

SKIL: Semantic Keypoint Imitation Learning for Generalizable Data-efficient Manipulation

Shengjie Wang^{1,2,3} Jiacheng You^{1,2,3} Yihang Hu¹ Jiongye Li⁴ Yang Gao^{1,2,3,†}
¹IIS, Tsinghua University ²Shanghai AI Laboratory
³Shanghai Qi Zhi Institute ⁴Department of Automation, Tsinghua University

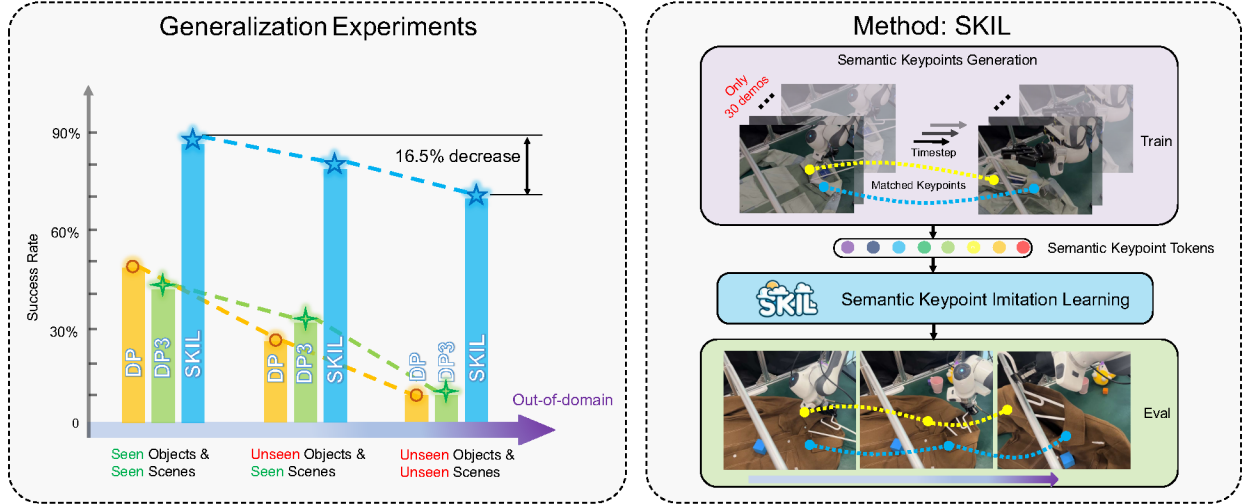


Fig. 1: Given limited robot demonstrations, our method, **Semantic Keypoint Imitation Learning (SKIL)**, can achieve superior performance for generalizable and long-horizon manipulation tasks, such as hanging a cloth on a rack. When encountering unseen objects and scenes, SKIL outperforms baselines by a large margin in four short-horizon tasks.

Abstract—Real-world tasks such as garment manipulation and table rearrangement demand robots to perform generalizable, highly precise, and long-horizon actions. Although imitation learning has proven to be an effective approach for teaching robots new skills, large amounts of expert demonstration data are still indispensable for these complex tasks, resulting in high sample complexity and costly data collection. To address this, we propose **Semantic Keypoint Imitation Learning (SKIL)**, a framework which automatically obtains semantic keypoints with help of vision foundation models, and forms the descriptor of semantic keypoints that enables efficient imitation learning of complex robotic tasks with significantly lower sample complexity. In real world experiments, SKIL doubles the performance of baseline methods in tasks such as picking a cup or mouse, while demonstrating exceptional robustness to variations in objects, environmental changes, and distractors. For long-horizon tasks like hanging a towel on a rack where previous methods fail completely, SKIL achieves a mean success rate of 70% with as few as 30 demonstrations. Furthermore, SKIL naturally supports cross-embodiment learning due to its semantic keypoints abstraction, our experiments demonstrate that even human videos bring considerable improvement to the learning performance. All these results demonstrate the great success of SKIL in achieving data-efficient generalizable robotic learning. Visualizations and code are available at: <https://skil-robotics.github.io/SKIL-robotics/>.

I. INTRODUCTION

End-to-end policy learning has gained significant attention in training robotic systems [27, 4, 57]. Imitation learning, in

particular, has been instrumental in enhancing the efficiency of end-to-end training by enabling robots to learn directly from expert demonstrations via supervised learning [35, 37, 62]. While current methods have shown success in various robotic manipulation scenarios, many real-world tasks, such as garment manipulation and table rearrangement, require policies that are generalizable and capable of highly precise or long-horizon actions [15, 63, 14].

We take the household task of hanging clothes as an illustrative example. Hanging clothes on a rack involves multiple stages, such as selecting a hanger, precisely inserting the garment onto the hanger, and subsequently placing it on the rack. This task also requires generalization across different types of clothing and varying positions. Consequently, the task inherently demands a large number of demonstrations, resulting in high sample complexity. However, data collection for such a complex task is both time-consuming and costly. A recent work [63] has demonstrated similar skills through nearly 10 thousand robot demonstrations. To address this challenge, prior approaches have focused on advancing data collection methods [63, 18, 6, 16, 3, 29] and developing advanced representation techniques [33, 23, 60, 47, 30, 3]. For instance, dataset-based methods emphasize the importance of constructing diverse datasets [63, 16, 29]. A recent study

finds that collecting data in a wide range of environments, each with unique manipulation objects and accompanying demonstrations, significantly enhances generalization capabilities [29]. However, even with these strategies, achieving zero-shot generalization to novel objects and environments may require a large amount of demonstrations for a single task [29]. In parallel, research into novel representations—such as pre-trained vision models [33, 7, 25, 3], 3D visual representations [17, 58, 47], and semantic-geometric features [23, 60, 48]—has aimed to overcome these limitations. These advanced representations improve sample efficiency, particularly by exhibiting spatial generalization [58, 47]. Despite this progress, these methods still tend to overfit to the seen objects and scenes during training, struggling to handle unseen objects and environments. Given these challenges, a critical question arises: *How can we reduce sample complexity to enable robots to learn data-efficient and generalizable manipulation tasks?*

In this paper, we propose Semantic Keypoints Imitation Learning (SKIL), an imitation learning framework that leverages a vision foundation model to identify semantic keypoints as observations. This sparse representation reduces the problem’s dimensionality, thereby achieving a lower sample complexity. By matching consistent keypoints between training and testing objects, SKIL utilizes their associated features and spatial information as conditional inputs to a diffusion-based action head, which outputs the robot’s actions. Additionally, SKIL inherently facilitates cross-embodiment learning through its abstraction of semantic keypoints. Our key contributions are summarized as follows:

- 1) We propose the Semantic Keypoint Imitation Learning (SKIL) framework, which automatically obtains the semantic keypoints through a vision foundation model, and forms the descriptor of semantic keypoints for downstream policy learning.
- 2) SKIL offers a series of advantages. First, the sparsity of semantic keypoint representations enables data-efficient learning. Second, the proposed descriptor of semantic keypoints enhances the policy’s robustness. Third, such semantic representations enables effective learning from cross-embodiment human and robot videos.
- 3) SKIL shows a remarkable improvement over previous methods in 6 real-world tasks, by achieving a success rate of 72.8% during testing, offering a 146% increase compared to baselines. SKIL can perform long-horizon tasks such as *hanging a towel or cloth on a rack*, with as few as 30 demonstrations, where previous methods fail completely.

II. RELATED WORK

A. Imitation Learning

Imitation learning from expert demonstrations has always been an effective approach for teaching robots skills [35, 37, 62, 14], in which behavior cloning (BC) serves as a most basic and straight-forward algorithm by directly taking expert actions as supervision labels [44, 11]. Considering the

challenges of obtaining accurate states while implementing in real-world environments, the most intuitive yet simple idea, end-to-end mapping from images to actions, has become one of the most popular choices of researchers in recent years [51, 62, 5]. For example, the ACT algorithm employs a transformer architecture to produce action tokens from encoded image tokens, and achieves accurate closed-loop control [62, 63, 15]. Diffusion Policy, on the other hand, leverages the diffusion process to model a conditional action distribution, therefore achieving multimodal behavior learning ability and stabler training [5, 6, 18].

Given the advantages of Diffusion Policy [5], recent research has focused on improving its representation capabilities. Some recent methods explore how to fuse information from 3D visual scenes, language instructions, and proprioception [17, 41, 60, 61, 23]. However, these approaches typically predict keyframes rather than continuous actions (e.g., Peract [41], Act3D [17], 3D Diffusor actor [23]), which makes them less effective at completing complex tasks. Other methods such as DP3 [58], RISE [47] and EquiBot [54] utilize 3D perception as observations and output the sequence of continuous actions. However, as demonstrated in our experiments, these methods severely lacks real-world generalization abilities with limited demonstrations. Furthermore, GenDP [48] computes dense semantic fields via cosine similarity with 2D reference features, and achieves category-level manipulation by taking semantic fields as input. However, semantic fields contain too much redundancy information, which harms the learning efficiency. In contrast, our method leverages semantic keypoints to construct a sparse representation, which, when conditioned on the policy, reduces the need of amount of demonstrations.

B. Keypoint-based Imitation Learning

Extracting point motions from visual images serves as a general feature representation method. Due to its inherent sparsity, this approach has proven to be data-efficient in robotic manipulation [50, 9, 39]. Early works typically required supervised training on large datasets from simulators or real world, to learn motion of points on related objects [59, 9, 45]. Recent approaches, such as ATM [49], Track2Act [2], GeneralFlow [56] and Im2Flow2Act [52], utilize an off-the-shelf tracker (e.g., Cotracker [22]) to observe the motion of points. These models support human-to-robot transfer by leveraging these point motion trajectories during policy training. However, despite these advances, observing accurate point motions remains challenging, as it struggles to generalize to unseen objects.

Keypoint representation dramatically reduces the dimensionality of the state, thereby achieving high efficiency in robot navigation and manipulation [32, 13, 12, 7]. Learning-based methods for keypoint extraction require large datasets and self-supervised training to generalize across object categories [12, 31, 53]. Recent advances in vision models, such as DINOv2 [34] and DiFT [43], allow the use of pre-trained models to extract semantic correspondence. DINOBot [7] and Robo-ABC [21] can retrieve visually similar objects from human demonstrations and align the robot’s end-effector with

new objects. However, this approach lacks feedback loops, limiting its use to offline planning.

Some recent works showed success in learning short-horizon tasks rapidly. ReKep [19] utilizes DINOv2 [34] for getting keypoint proposals and GPT-4 [20] for building relational constraints of keypoints, and then applies an optimization solver to generate robot trajectories. KALM [10] leverages Segment Anything (SAM) [26] and large language models (LLMs) to automatically generate task-relevant, consistent keypoints across instances. KAT [8] employs in-context learning (ICL) with LLMs, requiring only 5–10 demonstrations to teach the robot new skills, and their subsequent work, Instant Policy [46], extends ICL to a graph generation problem. However, these methods struggle to achieve precise or long-horizon motion planning due to the inaccuracy and latency of LLMs. In contrast, our method utilizes semantic keypoints as observations and applies a diffusion action head for real-time imitation learning with continuous actions. A very recent work [28] also uses semantic keypoints as observations of an imitation learning algorithm. However, they rely on off-the-shelf tracking models [22] to derive keypoints' positions, which restricts their application to long-horizon tasks. In contrast, our method SKIL enables learning long-horizon tasks such as hanging a towel or cloth on a hanger, with only 30 demonstrations.

III. METHOD

Our proposed method, SKIL, comprises two primary modules, which is shown in Figure 3. Based on RGBD input frames, the *Semantic Keypoints Description Module* first obtains the semantic keypoints and computes the descriptor of keypoints (Section III-B). The *Policy Module* then uses a transformer encoder to fuse the information of keypoints descriptor, and finally applies a diffusion action head to output robot actions (Section III-C). We also introduce an extra cross-embodiment learning version of SKIL in Section III-D.

A. Key Insight

Previous perception modules often tend to overfit specific training objects and scenes, struggling to handle objects with varying colors, textures, and geometries. However, practical manipulation tasks rely little on these detailed properties. For instance, when picking up a cup, a smart agent should focus mainly on the position of the handle instead of its color or shape. Similarly, when folding clothes, the positions of the collar and sleeves matter the most. In this context, sparse semantic keypoints, such as the handle of a cup or the collar and sleeves of a shirt, serve as the most critical task-relevant information. These keypoints remain highly consistent across different objects or scenes, enabling them to address the overfitting challenge. Furthermore, this simplified formulation can significantly reduce the need of extensive demonstrations, reaching a much higher sample efficiency.

Recently, vision foundation models have showed remarkable success across various downstream tasks, particularly excelling in semantic correspondence detection [34, 26, 43, 38]. This

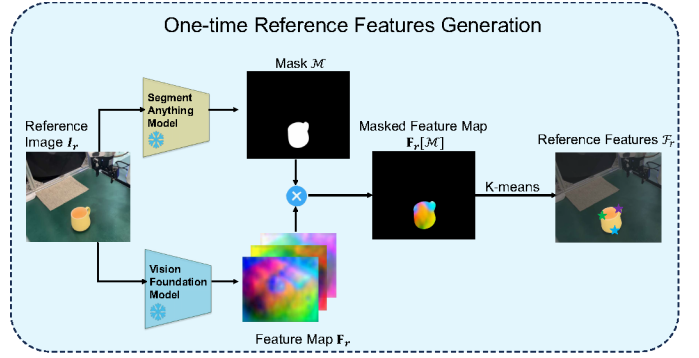


Fig. 2: Process of generating reference features, given a single reference image of the specific task: (1) Apply SAM [26] and Vision Foundation Model to obtain the mask $\mathcal{M}$ and the feature map $\mathbf{F}_r$ individually; (2) Cluster the masked features $\mathbf{F}_r[\mathcal{M}]$ to obtain the reference features $\mathcal{F}_r$ using K-means.

success motivates us to leverage vision foundation models to extract keypoints with semantic correspondence, as introduced later in the following sections.

B. Semantic Keypoints Description Module

In this module, we first obtain the reference features of the task, with the help of a vision foundation model. Based on the reference features, we build the cosine-similarity map of the current frame. Finally, we calculate the descriptor of semantic keypoints from the cosine-similarity map and the original depth image. The process can be seen in Figure 3.

One-time Reference Features Generation. For each task, we only require one single image of the task scene to automatically detect the reference keypoints and features, which are then used throughout the entire training and evaluation process. We illustrate this one-time reference features generation process in Figure 2. Given an RGB reference image $\mathcal{I}_r \in \mathbb{R}^{H \times W \times 3}$, we first extract patch-wise features using a vision foundation model (e.g., DiFT [43] and RADIO [38]) and apply bilinear interpolation to upsample the features to the original image size, $\mathbf{F}_r \in \mathbb{R}^{H \times W \times D}$. Meanwhile, we use Segment Anything Model (SAM) [26] to generate a mask $\mathcal{M}$ of all relevant objects. We then combine these two results to get the masked feature map $\mathbf{F}_r[\mathcal{M}]$, which contains $|\mathcal{M}|$ non-zero feature vectors of dimension D . Finally, we apply K-means to cluster these feature vectors into N clusters, with center pixel positions $\{(h_r^i, w_r^i) \in \mathcal{M} \mid 0 < i \leq N\}$. These cluster centers forms the reference keypoints, and their corresponding features form the set of reference features, which is $\mathcal{F}_r = \{\mathcal{F}_r^i = \mathbf{F}_r[h_r^i, w_r^i] \in \mathbb{R}^D \mid 0 < i \leq N\}$.

Note that N is a manually set hyperparameter, and K-means could be replaced by other keypoint proposal strategies. See Section V-C for more detailed discussions.

Cosine-similarity Map Generation. As shown in Figure 3, during the training and inference phases, the input image $\mathcal{I}_t \in \mathbb{R}^{H \times W \times 3}$ at current timestep is processed by the same vision foundation model to obtain the feature map at the original image size, $\mathbf{F}_t \in \mathbb{R}^{H \times W \times D}$. We compute the cosine-similarity

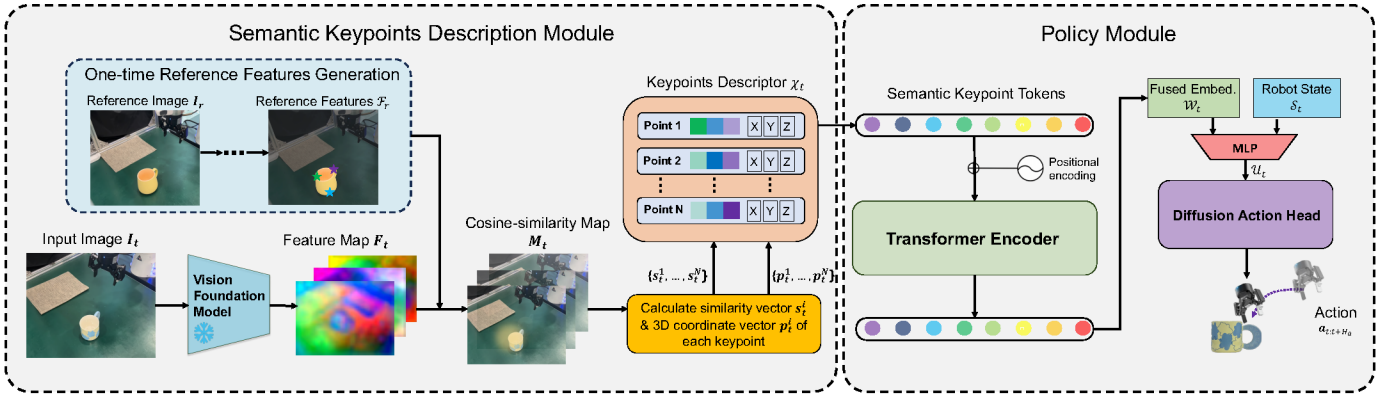


Fig. 3: Overview of our framework SKIL, including *Semantic Keypoints Description Module* and *Policy Module*. The first module computes descriptors for the semantic keypoints. Then, we apply a transformer encoder to obtain the fused embedding of the keypoints. Conditioned on the fused embedding and robot state, a diffusion action head outputs the final action sequence.

map between $\mathbf{F}_t$ and the reference features $\mathcal{F}_r$,

$$\mathbf{M}_t = \text{cosine_sim}(\mathbf{F}_t, \mathcal{F}_r), \quad (1)$$

where $\mathbf{M}_t \in \mathbb{R}^{H \times W \times N}$, whose i -th channel (denoted as $\mathbf{M}_t^i$ later) among the N channels represents the cosine-similarity map between the current frame $\mathcal{I}_t$ and the i -th reference feature.

Keypoints Descriptor Calculation. According to the similarity map $\mathbf{M}_t$, we can obtain the pixel coordinate (h_t^i, w_t^i) of each matched semantic keypoint, denoted as follows:

$$(h_t^i, w_t^i) = \arg \max_{(h, w)} (\mathbf{M}_t^i[h, w]), \quad 0 < i \leq N. \quad (2)$$

The pixel coordinate of each keypoint can serve as the intermediate representation in some flow-based policies, such as ATM [49] and Track2Act [2]. However, this representation is lacking for semantic and spatial description of keypoints, harming the downstream policy learning.

Therefore, we compute a descriptor for each matched keypoint, consisting of a similarity vector and a 3D coordinate vector. The similarity vector represents the cosine-similarities between the matched keypoint and all reference keypoints. The vector can identify the matched keypoint by its maximum value, and the magnitude of this value represents the confidence of this matching. Since the similarity map $\mathbf{M}_t$ stores the cosine-similarities between all pixels of the input image and reference keypoints, the similarity vector can be defined as $\mathbf{s}_t^i \in \mathbb{R}^N$:

$$\mathbf{s}_t^i = \mathbf{M}_t[h_t^i, w_t^i, :] , \quad 0 < i \leq N. \quad (3)$$

Based on the pointcloud derived from the depth image, we obtain the 3D coordinate vector of each matched keypoint, defined as $\mathbf{p}_t^i \in \mathbb{R}^3$. Overall, the descriptor of each matched keypoint can be denoted by

$$\chi_t^i = [\mathbf{s}_t^i, \mathbf{p}_t^i] , \quad 0 < i \leq N, \quad (4)$$

which is later fed into the next *Policy Module*.

C. Policy Module

Transformer Encoder. We first tokenize each descriptor χ_t^i into tokens of each keypoint. Specifically, each descriptor is first embedded into a d -dimensional latent space with positional encoding. As shown in Figure 3, a transformer encoder processes all tokens and we compute the mean of all output tokens to obtain the fused embedding of keypoints $\mathcal{W}_t$. We define this whole process as

$$\mathcal{W}_t = \text{Encoder}(\chi_t^1, \chi_t^2, \dots, \chi_t^N) \quad (5)$$

where t denotes the timestep, N denotes the number of keypoints. Note that we choose mean of tokens [36] instead of a [CLS] token, for its slightly better performance in our experiments.

Diffusion Action Head. Based on the aforementioned encoder, we obtain the fused embedding $\mathcal{W}_t$ of the keypoints. We concatenate $\mathcal{W}_t$ with the robot state $\mathcal{S}_t$ (including joint positions, end-effector position and orientation, gripper state, etc) and use a multi-layer perceptron (MLP) to fuse them into a compact representation

$$\mathcal{U}_t = \text{MLP}(\mathcal{S}_t, \mathcal{W}_t), \quad (6)$$

as shown in Figure 3.

Conditioned on the compact representation $\mathcal{U}_t$, a diffusion action head outputs the robot action. Following Diffusion Policy (DP) [5], we use a CNN-based U-Net as the noise prediction network. Detailed formulations are provided in Appendix H. To improve temporal consistency, we predict an action chunk in a single step, $\mathbf{a}_{t:t+H_a} := (\mathbf{a}_t, \dots, \mathbf{a}_{t+H_a-1})$, where H_a denotes the chunk size. For real-time inference, we utilize DDIM [42], a diffusion model sampling accelerator, to reduce the number of diffusion denoising steps.

Action Ensemble. Existing vision foundation models occasionally produce mismatching of keypoints, which causes motion jitter. To address this issue, we employ an ensemble approach for action planning. Specifically, during training, we randomly dropout 20% of the semantic keypoint tokens of each frame, before sending them to the transformer encoder.

During testing, we repeat the action inferring process (with this random dropout) 20 times, and get the median of all output actions to be the finally executed action. (Note that this repetition can be done parallelly across the batch dimension, so that introduces almost no extra latency.) This ensemble strategy ensures smoother and more reliable action execution.

D. Cross-embodiment Learning

In this section we define an extra cross-embodiment learning version of SKIL. Our motivation is that semantic keypoints abstraction avoids incorporating embodiment information, therefore enables the use of diverse data source (including human videos). Inspired by ATM [49], a cross-embodiment learning framework, we view the trajectory prediction of keypoints as an intermediate task. The predicted trajectories serve as effective guidance for learning policies. We name this cross-embodiment version **SKIL-H**, which involves 2 modules:

- 1) *Trajectory Prediction Module*:
 - predicts future keypoint positions from pure video data,
 - trained with both robot and human demonstrations;
- 2) *Trajectory-to-Action Module*:
 - maps the predicted trajectories into robot actions,
 - trained with only robot demonstrations.

As illustrated in Figure 4, at timestep t , the *Trajectory Prediction Module* of SKIL-H takes the fused embedding $\mathcal{W}_t$ (produced by original SKIL) as input, and predicts the future keypoint trajectories as

$$\hat{\tau}_{t:t+H_p} = \{\hat{\mathbf{p}}_q^i \mid t < q \leq t + H_p, 0 < i \leq N\}, \quad (7)$$

in which $\hat{\mathbf{p}}_q^i$ denotes the predicted 3D position of i -th matched keypoint at future timestep q , N is the number of predicted keypoints and H_p is the prediction horizon. We employ a diffusion model to build the *Trajectory Prediction Module*. The training labels of the model are obtained with the help of an off-the-shelf tracking model (e.g., CoTracker [22]). Specifically, we obtain the 2D flow of the matched keypoints from videos using the tracking model and project them back to 3D real trajectories $\tau_{t:t+H_p}$, as the training labels.

The next *Trajectory-to-Action Module* of SKIL-H takes the predicted trajectories $\hat{\tau}_{t:t+H_p}$ and the robot state $\mathcal{S}_t$ as input, and process them with a transformer encoder followed by a diffusion action head to output the final robot action $\mathbf{a}_{t:t+H_a}$. This module functions similarly as the origin *Policy Module* of SKIL (See section III-C), but with different input format and encoder architecture. All other settings including the training loss remain the same.

IV. EXPERIMENT SETUP

We introduce the experiment setup in this section, including the task definitions, data collection & evaluation settings, and baselines to be compared with.

A. Task Definitions

We use a Franka robot arm equipped with a Robotiq gripper to perform six real-world tasks, including the first four short-horizon tasks and the last two long-horizon ones. A brief

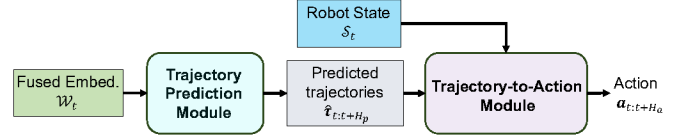


Fig. 4: Architecture of SKIL-H, comprising the *Trajectory Prediction Module* and the *Trajectory-to-Action Module*. The first module predicts the trajectories $\hat{\tau}_{t:t+H_p}$ of matched keypoints based on the fused embedding $\mathcal{W}_t$. The second module takes the predicted trajectories $\hat{\tau}_{t:t+H_p}$ and the robot state $\mathcal{S}_t$ as inputs, and outputs the robot action sequence $\mathbf{a}_{t:t+H_a}$.

overview of these tasks is listed below: (Visualizations are provided in Figure 5.)

- 1) **Pick Mouse**: The gripper grasps a mouse from the workspace and places it on the mouse mat.
- 2) **Grasp Handle of Cup**: The gripper grasps the cup's handle and places the cup on the right side of the table.
- 3) **Grasp Wall of Cup**: Instead of the handle, the gripper grasps the wall of the cup and places it on the right side of the table.
- 4) **Fold Towel**: The gripper grasps the left corner of the towel and lifts it toward the right corner.
- 5) **Hang Towel**: This multi-step task involves grasping a hanger from the table, placing it near the towel, pinching the towel's top edge to fold it through the hanger, and hanging the hanger on a rack. Visualization is provided in Figure 17 in Appendix F5.
- 6) **Hang Cloth**: This task involves grasping a hanger from the table, precisely inserting it into the cloth collar, rotating the hanger, and hanging the cloth on the rack. Visualization is provided in Figure 18 in Appendix F5.

The object poses and the joint positions of the Franka arm are randomly initialized throughout data collection and evaluation. See more details in Appendix B2.

Besides, we also conduct experiments on several simulation tasks. We select ten tasks from the MetaWorld [55] and DexArt [1] benchmarks. More details about the simulation tasks can be found in Appendix A.

B. Data Collection & Evaluation

Real-world expert demonstrations are collected through human teleoperation, following the collection process of Droid dataset [24]. The hardware setup is described in Appendix B1, where a Franka arm with a Robotiq gripper is teleoperated using a Meta Quest controller [24]. We collect 20 demonstrations for short-horizon tasks and 30 demonstrations for long-horizon tasks respectively. For all six tasks, we use 2 objects for training data collection, and we use 10 objects for the first four short-horizon tasks and 3~5 objects for the last two long-horizon ones during evaluation.

The action space contains the end-effector pose and gripper state, while observations include RGB images and corresponding depth images captured by a fixed third-view Zed2 camera, as shown in Figure 11 in Appendix B1.

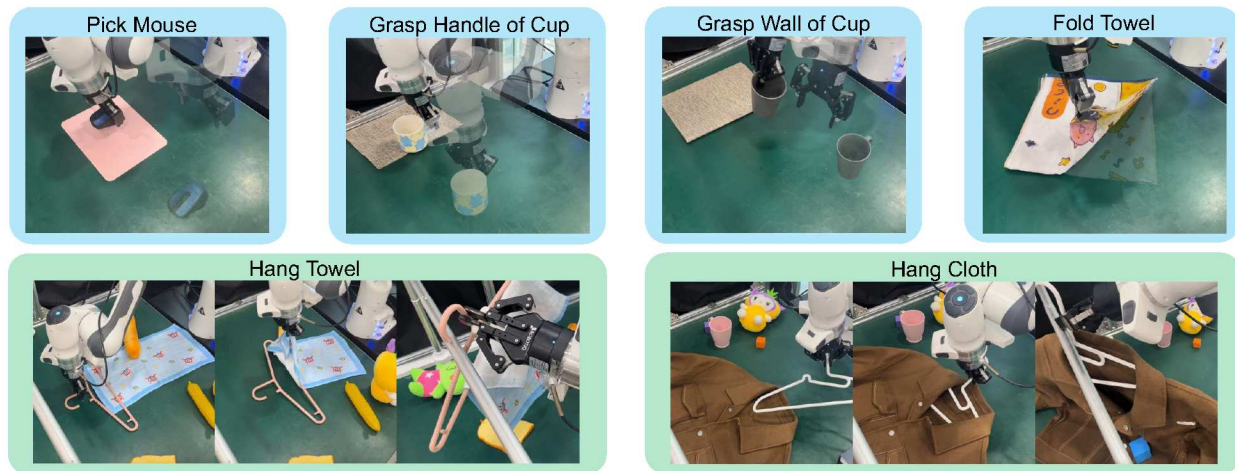


Fig. 5: Overview of our 6 real-world tasks, including the top 4 short-horizon ones and the bottom 2 long horizon ones.

TABLE I: Realworld results, measured by evaluation phase success rates on the unseen testing objects. SKIL outperforms baseline methods by a large margin on either training or testing objects. (All objects shown in Figures 21 and 22) Testing objects are unseen but belong to the same categories. For each object, we conduct five trials with random initialization. (The baselines with point clouds input perform poorly on *Grasp Wall of Cup*. We discuss a possible reason for this in Appendix F2.)

Method/Task	<i>Pick Mouse</i>		<i>Grasp Handle of Cup</i>		<i>Grasp Wall of Cup</i>		<i>Fold Towel</i>		<i>Hang Towel</i>		<i>Hang Cloth</i>		Average	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
DP	40%	4%	30%	36%	70%	36%	60%	50%	0%	0%	20%	12%	36.7%	23.0%
DP3	20%	18%	50%	32%	30%	18%	60%	58%	0%	0%	30%	24%	31.6%	25.0%
RISE	10%	14%	40%	34%	10%	12%	50%	32%	0%	0%	20%	16%	21.7%	18.0%
GenDP-S	40%	32%	50%	32%	40%	30%	60%	58%	0%	0%	30%	28%	36.7%	30.0%
SKIL (Ours)	90%	72%	90%	94%	90%	80%	90%	72%	70%	72%	60%	47%	81.7%	72.8%

As for the simpler simulation tasks, we collect 10 expert demonstrations for the chosen MetaWorld [55] and DexArt [1] tasks. More detailed settings can be found in Appendix A.

For all real-world and simulation tasks, we measure the performance of a specific method by its average success rate (on unseen testing objects for most tasks) in the evaluation phase.

C. Baselines

We compare SKIL with state-of-the-art imitation learning algorithms. **Diffusion Policy (DP)** [5] models the action distribution using a diffusion model and leverages RGB observations as conditions in the diffusion model. **DP3** [58] utilizes a similar diffusion architecture to Diffusion Policy and introduces a compact 3D representation instead of 2D images by employing an efficient MLP encoder. **RISE** [47] uses 3D point clouds to predict robot actions by first processing the data with a shallow 3D encoder and then mapping it to actions using a transformer. **GenDP-S**: GenDP [48] generates 3D descriptor fields from multi-view RGBD data, computes semantic fields via cosine similarity with 2D reference features, and uses PointNet++ and a diffusion model to predict robot actions. Note that we name our implementation GenDP-S because we build 3D descriptor fields from single view. More implementation details of SKIL and all these baselines

can be found in Appendix I and D respectively.

V. RESULTS & ANALYSIS

In this section, we present SKIL’s performance along with its comparison result with baseline methods, from which we can prove the strong generalization ability and the excelling data efficiency of SKIL. We also demonstrate the performance of SKIL-H, showing its cross-embodiment learning ability. Finally, we present ablation studies to assess our choices on each of SKIL’s components.

A. Performance & Comparison

Table I presents the main results on real-world tasks. SKIL significantly outperforms several strong baselines across all tasks, by achieving a mean success rate of 72.8% under unseen objects, comparing to the highest success rate of 30% achieved by baselines. Figure 5 presents snapshots of the real-world experiments. SKIL also reaches the best performance across the baselines on simulation tasks. Detailed results and analysis of simulation results are provided in Appendix E.

For an intuitive glance of SKIL’s keypoint-based representation, Figure 6 illustrates the moving trajectories of the matched semantic keypoints on several tasks. By comparing with the keypoints in the reference images, we observe that most of the matched semantic keypoints in the input images maintain

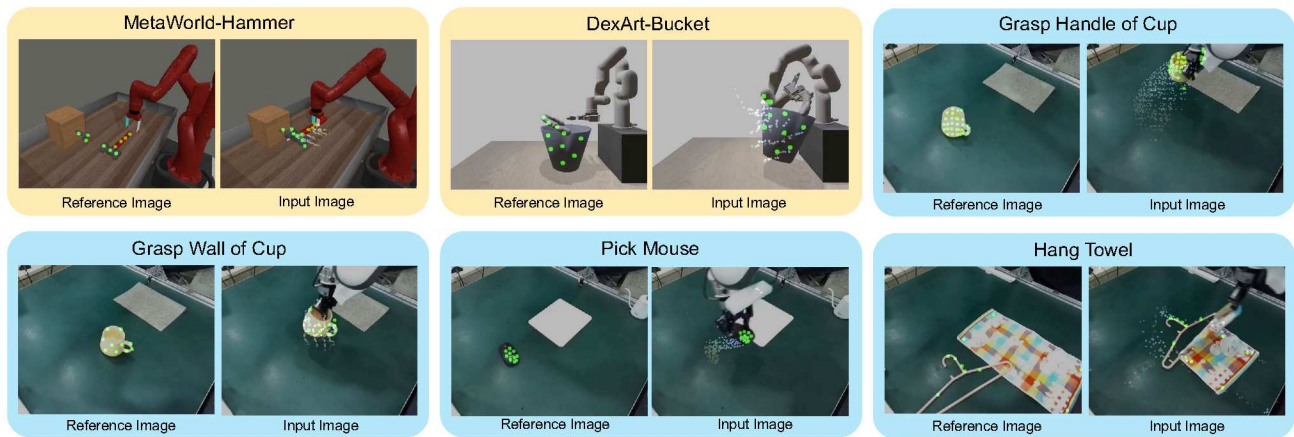


Fig. 6: Movement of semantic keypoints in SKIL. Green points represent the keypoints in the current frame, and the white trajectories show their movements in previous timesteps. Most keypoints maintain temporal consistency. Although some keypoints are occluded, the mismatched points remain close to the relevant object.



Fig. 7: Visualization of three types of environment changes. *Situation 1* adds distractors to the workspace, *Situation 2* changes the background, and *Situation 3* (the most difficult one) incorporates these two changes.

high temporal consistency. Even when some keypoints are occluded, the mismatched points remain close to the correct position. Additional visualizations of keypoint trajectories on different objects can be found in Appendix F1.

In the following, we analyze the generalization ability and data efficiency of SKIL with specific examples.

1) **Generalization:** We categorize generalization into three dimensions: *Spatial generalization*, *Object generalization* and *Environment generalization*.

a) **Spatial generalization:** We can see that the baselines achieve inferior performance on the *Pick Mouse* task. These methods often fail to grasp the mouse that is near the corner of the workspace. In contrast, SKIL is able to handle most of the workspace, as illustrated in Figure 14 in Appendix F3. Meanwhile, SKIL can pick up the towel’s corner more precisely than the baselines, as shown in Figure 15 in Appendix F3. The improvement is primarily due to the semantic keypoints located on relevant objects, which helps the policy better understand the pose of the objects.

b) **Object generalization:** The results in Table I demonstrate that SKIL maintains remarkable performance even on unseen objects. In contrast, DP, DP3, and RISE perform poorly on unseen objects. GenDP-S performs slightly better than these three, by utilizing semantic fields to capture critical task-relevant information. However, SKIL uses semantic keypoint

TABLE II: Average success rates of SKIL and baselines under the original *Situation 0* and all three situations with environmental changes. SKIL outperforms the baselines by a large margin in all situations.

Method	<i>Situation 0</i>	<i>Situation 1</i>	<i>Situation 2</i>	<i>Situation 3</i>
DP	4/10	4/10	0/10	0/10
DP3	5/10	1/10	4/10	0/10
RISE	4/10	3/10	2/10	1/10
GenDP-S	5/10	3/10	5/10	2/10
SKIL (Ours)	9/10	9/10	8/10	8/10

abstraction to obtain more concise and accurate representation. We take the *Grasp Handle of Cup* task as an example. As shown in Figure 13 in Appendix F1, different cups exhibit varying appearances (shape, color, etc.), but share common structures (including the cup body and handle), which matter the most for this manipulation task. In practice, the semantic keypoint descriptors of SKIL effectively capture the information of these structures, while disregarding redundant details related to appearance, thus provides excellent generalization across objects.

c) **Environment generalization:** We evaluate SKIL and other baselines on three new situations with environmental changes, including adding distractors (*Situation 1*), background color shifting (*Situation 2*) and their combination (*Situation 3*), as illustrated in Figure 7. For simplicity, we denote the original environment without any changes as *Situation 0*.

We report the comparison results of the task *Grasp Handle of Cup* in Table II, with the results of other tasks available in Appendix F4. As shown, SKIL maintains consistently high performance across all situations. In contrast, baselines experience a substantial drop in performance especially in the most difficult *Situation 3*. Specifically, DP with image input suffers a severe performance drop when the background changes. DP3 is more resilient to background changes because it disregards color channels, but it performs poorly with addi-

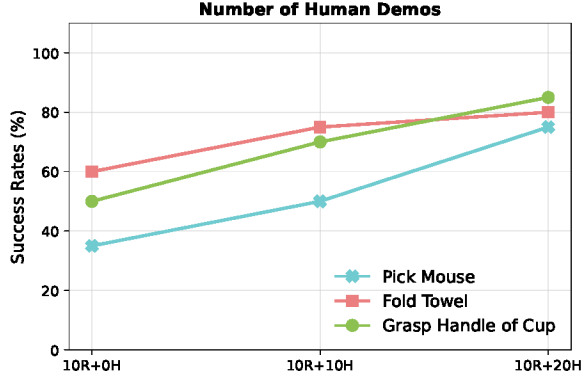


Fig. 8: The performance comparison between different numbers of human videos without any action labels. “10R+20H” represents 10 robot demonstrations and 20 human demonstrations.

tional distractors. GenDP performs better than other baselines but still suffers severe failures in *Situation 3*. SKIL’s semantic keypoints representation is least affected by environmental interference among these baselines, thus exhibiting superior generalization ability. For a more detailed visualization, please refer to the supplementary video.

2) **Data Efficiency:** Due to compounding errors, large amounts of data are indispensable for traditional imitation learning methods to get high performance, especially on long-horizon tasks. We consider the two long-horizon tasks, *Hang Towel* and *Hang Cloth*. These two tasks show different types of difficulty, one involves multiple pick-place actions, and the other requires precisely following a long spatial trajectory.

Despite these challenges, SKIL reaches high success rates with only 30 demonstrations, outperforming all baselines by a large margin. Particularly, SKIL achieves a success rate of 72% on *Hang Towel*, while all baselines fail completely. A prominent phenomenon is that they occasionally skip stages in the hanging process. Appendix F5 provides a detailed view of the task and illustrates the typical failure modes of the baselines.

Furthermore, we present the performance of SKIL and baselines with different numbers of demonstrations in Table III. It can be seen that with the increase in demonstration amounts, SKIL’s success rate grows much faster than the baselines. Specifically, the performance of SKIL with 10 demonstrations exceeds that of all baselines using 20 demonstrations on all tasks, showing SKIL’s excellent data efficiency.

B. Cross-embodiment Performance

By introducing a keypoint prediction model (Section III-D), SKIL-H enhances policy learning using extra human videos without action labels. We test SKIL-H on three tasks: *Pick Mouse*, *Grasp Handle of Cup*, and *Fold Towel*, with 10 robot demonstrations and 0~20 human demonstrations. Figure 19 in Appendix F6 provides snapshots of human demonstration videos on these tasks. Quantitative results of final performance

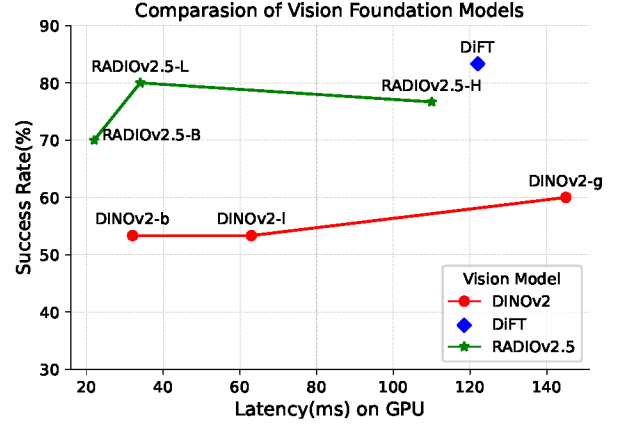


Fig. 9: A comparison of success rates and inference latency for different vision foundation models (DINOv2, DiFT, and RADIO v2.5) on an NVIDIA A10 GPU. The results highlight the trade-off between computational overhead (latency) and performance (success rate) across varying model scales.

are shown in Figure 8. We can see that success rates increase significantly with the growth of human demo amounts. Particularly on the relatively hardest task *Pick Mouse* among the three, 20 human demos lead to a dramatic 40% increase in success rate, comparing to the policy trained solely on 10 robot demos. Besides, we also observe that with more human videos, SKIL-H produces smoother action sequences during evaluation. All these results show that the *Trajectory Prediction Module* of SKIL-H do benefit from human videos, and further confirm the successful cross-embodiment semantic abstraction achieved by SKIL’s keypoint description process.

C. Ablations

Since the core contribution of SKIL lies in the design of a novel representation for semantic keypoints, we conduct ablation studies to evaluate the impact of our design choices in selecting keypoints. Specifically, we investigate the impact of different vision foundation models, keypoint numbers, and keypoint proposal strategies on three tasks: *Pick Mouse*, *Grasp Handle of Cup*, and *Fold Towel*.

Ablation on Vision Foundation Models. In SKIL we use DiFT [43] with Stable Diffusion 2.1 model, to extract features for later keypoint-related calculation, as described in Section III-B. We also tried 2 other recent models DINOv2 [34] and RADIO [38], which are good at object detection and segmentation. It can be seen from Figure 9 that DINOv2 performs far behind the others, regardless of the size of backbone used. We observe that the keypoints obtained by DINOv2 suffer from severe mismatches, especially when objects are partially occluded, as illustrated in Figure 20 in Appendix G1. On the other hand, the performance of RADIO models with ViT-L and ViT-H architectures are only slightly behind DiFT but with lower latencies, thus offering new choices for users to trade off performance and latency when implementing SKIL in specific scenes. Note that SKIL itself does not rely on

TABLE III: Realworld results with different numbers of demonstrations. Here we use the two seen objects in the training phase to test the success rate, and conduct five trials for each object with random initialization in each task.

Method/Task Number of Demos	<i>Pick Mouse</i>		<i>Grasp Handle of Cup</i>		<i>Fold Towel</i>		<i>Hang Towel</i>		<i>Hang Cloth</i>	
	10	20	10	20	10	20	20	30	20	30
DP	20%	40%	10%	30%	40%	60%	0%	0%	0%	20%
DP3	10%	20%	20%	50%	50%	60%	0%	0%	0%	30%
RISE	0%	10%	20%	40%	30%	50%	0%	0%	0%	20%
GenDP-S	20%	40%	30%	50%	40%	60%	0%	0%	0%	30%
SKIL (Ours)	50%	90%	70%	90%	60%	90%	40%	70%	40%	60%

TABLE IV: Average success rates with different keypoint numbers (N).

Num. keypoints (N)	<i>Pick Mouse</i>	<i>Fold Towel</i>	<i>Grasp Handle of Cup</i>
10	7/10	7/10	7/10
20	8/10	8/10	8/10
30	8/10	7/10	9/10

TABLE V: Average success rates with different keypoint proposal strategies. “Random” means selecting keypoints randomly inside the object mask, and “Manual” means manually selecting keypoints based on human knowledge. We choose to use K-means for SKIL.

Method	<i>Pick Mouse</i>	<i>Fold Towel</i>	<i>Grasp Handle of Cup</i>
Manual	7/10	7/10	9/10
Random	7/10	6/10	9/10
K-means(Ours)	8/10	7/10	9/10

any specific vision foundation model, so we believe that it could continue to benefit from any latest model to get even higher performance in the future. More details can be found in Appendix G1.

Ablation on Keypoint Numbers. We investigate the impact of the number of keypoints (N) set in the *Semantic Keypoints Description Module* (Section III-B). Experiment results are shown in Table IV, which illustrate that SKIL achieves similar performance with 10-30 keypoints. That means the performance is insensitive to the change of keypoint numbers in this range. Actually, higher N leads to higher dimensionality of keypoint descriptors, which means more information encoded with higher computing cost. Future users of SKIL may easily find appropriate values of N with few trials to reach enough performance with the least computing cost on new tasks.

Ablation on Keypoint Proposal Strategies. Keypoint proposal aims to identify the reference keypoints on the target objects from the reference image. We choose to use K-means for SKIL as illustrated in Section III-B. Here, we compare its effectiveness with 2 other strategies (selecting keypoints manually or randomly on objects in the reference image). As shown in Table V, these 2 strategies achieve very similar performance, with the random strategy slightly inferior only on the *Fold Towel* task. We speculate that the lack of keypoints on edge of the towel hinders accurate grasping of the edge. Our choice K-means performs best among these strategies on all tasks, with its strong clustering ability of object features

and independence of human inductive bias.

VI. LIMITATIONS

Although SKIL has demonstrated extraordinary performance in these manipulation tasks, its capability is strictly upper-bounded by the capability of its upstream vision foundation model. As an example, we have tried but struggled to complete a *Bulb Assembly* task with SKIL, because the precision of keypoints extracted by the current model (DiFT) could not reach the high requirement of such task. Another limitation is that current SKIL is unable to complete tasks that require detailed perception of environments, as it only extracts keypoints from the relevant objects. For instance, it might violate safety constraints on tasks with multiple obstacles. Future work may extend the capability of SKIL by developing an efficient keypoint-based environment representation.

VII. CONCLUSIONS

High sample complexity remains a significant barrier to advancing imitation learning for generalizable and long-horizon tasks. To address this challenge, we develop the Semantic Keypoints Imitation Learning (SKIL) algorithm. Leveraging a vision foundation model, SKIL obtains the semantic keypoints as sparse observations, significantly reducing the dimensionality of the problem, and the proposed descriptors of semantic keypoints substantially improve the policy’s generalization ability. Furthermore, the semantic keypoint abstraction of SKIL naturally supports cross-embodiment learning. Experiments demonstrate that SKIL achieves excellent data efficiency and strong generalization ability. We believe that our work can pave the way for the development of general-purpose robots capable of solving complicated open-world problems.

VIII. ACKNOWLEDGEMENT

This work is supported by the National Key R&D Program of China (2022ZD0161700), National Natural Science Foundation of China (62176135), Shanghai Qi Zhi Institute Innovation Program SQZ202306 and the Tsinghua University Dushi Program.

REFERENCES

- [1] Chen Bao, Helin Xu, Yuzhe Qin, and Xiaolong Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21190–21200, 2023.

- [2] H Bharadhwaj, R Mottaghi, A Gupta, and S Tulsiani. Track2act: Predicting point 250 tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv preprint* 251. *arXiv preprint arXiv:2405.01527*, 252, 2024.
- [3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [4] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023.
- [5] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.
- [6] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.
- [7] Norman Di Palo and Edward Johns. Dinobot: Robot manipulation via retrieval and alignment with vision foundation models. *arXiv preprint arXiv:2402.13181*, 2024.
- [8] Norman Di Palo and Edward Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024.
- [9] Ben Eisner, Harry Zhang, and David Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. *arXiv preprint arXiv:2205.04382*, 2022.
- [10] Xiaolin Fang, Bo-Ruei Huang, Jiayuan Mao, Jasmine Shone, Joshua B Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Keypoint abstraction using large models for object-relative imitation learning. *arXiv preprint arXiv:2410.23254*, 2024.
- [11] Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pages 158–168. PMLR, 2022.
- [12] Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5(2):492–499, 2019.
- [13] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 15–22. IEEE, 2014.
- [14] Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetstein, and Chelsea Finn. Humanplus: Humanoid shadowing and imitation from humans. *arXiv preprint arXiv:2406.10454*, 2024.
- [15] Zipeng Fu, Tony Z Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.
- [16] Jensen Gao, Annie Xie, Ted Xiao, Chelsea Finn, and Dorsa Sadigh. Efficient data collection for robotic manipulation via compositional generalization. *arXiv preprint arXiv:2403.05110*, 2024.
- [17] Theophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *7th Annual Conference on Robot Learning*, 2023.
- [18] Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. *arXiv preprint arXiv:2407.10353*, 2024.
- [19] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.
- [20] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- [21] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrun Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2025.
- [22] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *European Conference on Computer Vision*, pages 18–35. Springer, 2025.
- [23] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.
- [24] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.
- [25] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [26] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.
- [27] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter

- Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.
- [28] Mara Levy, Siddhant Haldar, Lerrel Pinto, and Abhinav Shirivastava. P3-po: Prescriptive point priors for visuo-spatial generalization of robot policies. *arXiv preprint arXiv:2412.06784*, 2024.
- [29] Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. *arXiv preprint arXiv:2410.18647*, 2024.
- [30] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.
- [31] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. kpm: Keypoint affordances for category-level robotic manipulation. In *The International Symposium of Robotics Research*, pages 132–157. Springer, 2019.
- [32] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.
- [33] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.
- [34] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- [35] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.
- [36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [37] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. State-only imitation learning for dexterous manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7865–7871. IEEE, 2021.
- [38] Mike Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Am-radio: Agglomerative vision foundation model reduce all domains into one. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12490–12500, 2024.
- [39] Daniel Seita, Yufei Wang, Sarthak J Shetty, Edward Yao Li, Zackory Erickson, and David Held. Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds. In *Conference on Robot Learning*, pages 1038–1049. PMLR, 2023.
- [40] Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Conference on Robot Learning*, pages 1332–1344. PMLR, 2023.
- [41] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.
- [42] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.
- [43] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. *Advances in Neural Information Processing Systems*, 36:1363–1389, 2023.
- [44] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.
- [45] Mel Vecerik, Carl Doersch, Yi Yang, Todor Davchev, Yusuf Aytar, Guangyao Zhou, Raia Hadsell, Lourdes Agapito, and Jon Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5397–5403. IEEE, 2024.
- [46] Vitalis Vosylius and Edward Johns. Instant policy: In-context imitation learning via graph diffusion. *arXiv preprint arXiv:2411.12633*, 2024.
- [47] Chenxi Wang, Hongjie Fang, Hao-Shu Fang, and Cewu Lu. Rise: 3d perception makes real-world robot imitation simple and effective. *arXiv preprint arXiv:2404.12281*, 2024.
- [48] Yixuan Wang, Guang Yin, Binghao Huang, Tarik Kestemur, Jiuguang Wang, and Yunzhu Li. Gendp: 3d semantic fields for category-level generalizable diffusion policy. In *8th Annual Conference on Robot Learning*, volume 2, 2024.
- [49] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.
- [50] Thomas Weng, Sujay Man Bajracharya, Yufei Wang, Khush Agrawal, and David Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In *Conference on Robot Learning*, pages 192–202. PMLR, 2022.
- [51] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2174–2182, 2017.
- [52] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024.

- [53] Zhengrong Xue, Zhecheng Yuan, Jiashun Wang, Xueqian Wang, Yang Gao, and Huazhe Xu. Useek: Unsupervised se (3)-equivariant 3d keypoints for generalizable manipulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1715–1722. IEEE, 2023.
- [54] Jingyun Yang, Zi-ang Cao, Congyue Deng, Rika Antonova, Shuran Song, and Jeannette Bohg. Equibot: Sim (3)-equivariant diffusion policy for generalizable and data efficient learning. *arXiv preprint arXiv:2407.01479*, 2024.
- [55] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.
- [56] Chengbo Yuan, Chuan Wen, Tong Zhang, and Yang Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024.
- [57] Maryam Zare, Parham M Kebria, Abbas Khosravi, and Saeid Nahavandi. A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, 2024.
- [58] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy. *arXiv preprint arXiv:2403.03954*, 2024.
- [59] Harry Zhang, Ben Eisner, and David Held. Flowbot++: Learning generalized articulated objects manipulation via articulation projection. *arXiv preprint arXiv:2306.12893*, 2023.
- [60] Tong Zhang, Yingdong Hu, Hanchen Cui, Hang Zhao, and Yang Gao. A universal semantic-geometric representation for robotic manipulation. *arXiv preprint arXiv:2306.10474*, 2023.
- [61] Tong Zhang, Yingdong Hu, Jiacheng You, and Yang Gao. Leveraging locality to boost sample efficiency in robotic manipulation. *arXiv preprint arXiv:2406.10615*, 2024.
- [62] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.
- [63] Tony Z Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Kamyar Ghasemipour, Chelsea Finn, and Ayzaan Wahid. Aloha unleashed: A simple recipe for robot dexterity. *arXiv preprint arXiv:2410.13126*, 2024.

A. Simulation Environment

Benchmarks. Metaworld consists of 50 distinct robotic manipulation tasks using a sawyer robot. The four-dimensional action space includes the relative changes in the end-effector position and gripper state, while observations consist of an RGB image and the corresponding depth image. Relative changes in the end-effector position range from -1 to 1 , and the gripper state ranges from 0 to 1 .

The DexArt benchmark consists of 4 dexterous manipulation tasks. The action space is 22-dimensional because DexArt employs a 16-DoF Allegro hand and a 6-DoF Xarm. Each dimension of the action space represents the relative change in joint position, ranging from -1 to 1 . Besides, the observations are the same as those in MetaWorld. Notably, DexArt uses different objects during the training and evaluation phases.

Tasks. For the simulation experiments, we select 6 tasks from the MetaWorld [55] and all 4 tasks in DexArt [1], as shown in Figure 10. The tasks in MetaWorld are categorized into different difficulty levels based on the criteria in [40]. We chose three easy-level tasks and three medium-level tasks.

Training Details. We collect expert demonstrations using scripted policies in MetaWorld, and reinforcement learning (RL) agents in DexArt. For each task, we collect 10 demonstrations and train the policies using 3 random seeds. During training, we evaluate the policies every 100 epochs over 10 episodes and report the average of the highest 3 success rates. The final performance is reported as the mean and standard deviation of the success rates across the 3 seeds.

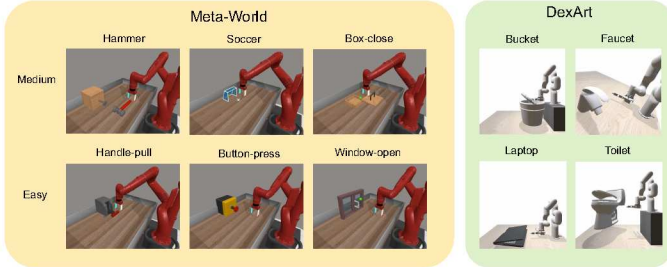


Fig. 10: Specific simulation tasks used in the MetaWorld and DexArt benchmarks.

B. Real-world Environment Setup

1) *Hardware:* Our real-world setup is shown in Figure 11. We primarily follow the hardware configuration of the Droid dataset [24]. As depicted in Figure 11, a wrist camera is mounted on the Franka arm similar to the Droid setup; however, we do not use the wrist camera throughout our experiments.

2) *Random Initialization:* The workspaces for all real-world tasks are shown in Figure 12. We consider the random positions and orientations of objects in each task. Note that the blue square represents the working space of the hanger hook in the two long-horizon tasks, *Hang Towel* and *Hang Cloth*.

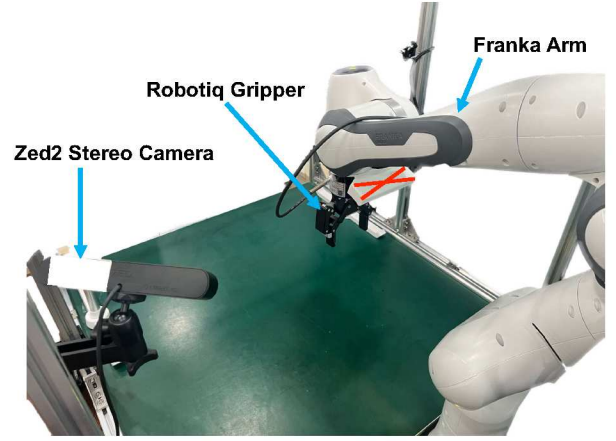


Fig. 11: Real-world hardware setup. We use a Franka arm equipped with a Robotiq gripper. A fixed Zed2 stereo camera is employed to capture visual and depth observations. Note that the wrist camera is not used in our experiments.

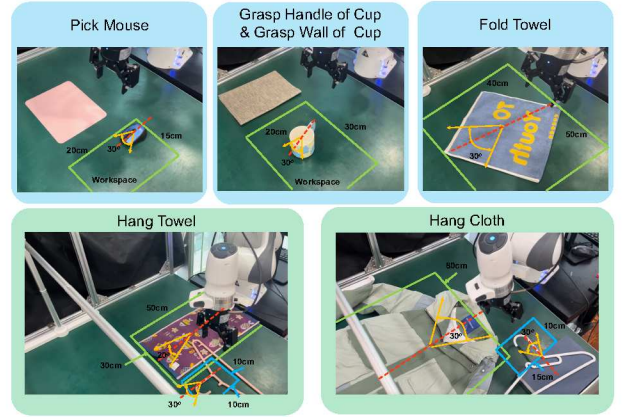


Fig. 12: Workspace of the six real-world tasks, in which we account for random positions and orientations of objects. The green and blue squares indicate the workspace, and the orange fan-shaped area represents the range of random orientations.

Throughout data collection and evaluation, we put each object at a random position inside the workspace, with a random orientation in a certain range.

C. Visualization of Real-world Task Objects

In Figures 21 and 22, we show the training and testing objects for all the real-world tasks *Pick Mouse*, *Grasp Handle of Cup*, *Grasp Wall of Cup*, *Fold Towel*, *Hang Towel*, and *Hang Cloth*, respectively.

D. Details of Baselines

We summarize the baselines as follows:

- 1) Diffusion Policy (DP) [5]: This method models the action distribution using a diffusion model, leveraging RGB observations as conditions within the diffusion process to generate robot actions.
- 2) DP3 [58]: DP3 adopts a similar diffusion architecture to DP but introduces a compact 3D representation instead

of using 2D images, leveraging an efficient MLP encoder for 3D data processing.

- 3) RISE [47]: RISE uses 3D point clouds to predict robot actions by processing the point cloud data with a shallow 3D encoder, which is then mapped to actions using a transformer-based network.
- 4) GenDP-S [48]: GenDP generates 3D descriptor fields from multi-view RGBD data and computes semantic fields by measuring the cosine similarity between these descriptors and 2D reference features. These semantic fields are then combined with the point cloud data and passed through PointNet++ and a diffusion policy to predict robot actions. Note that since we construct 3D descriptor fields using only a single camera, we refer to our implementation as GenDP-S.

Since all baselines use a diffusion-based action head, we maintain the same diffusion parameters as in our method, SKIL. For other parameters, we adhere to the settings outlined in the respective papers.

E. Simulation Experiment Results

Results of SKIL and baselines on simulation tasks are shown in Table VII and VI. We see that SKIL outperforms all baselines on all simulation tasks, but with smaller leading gaps than on real-world tasks. We list here the possible reasons we believe:

- Simulators always provide perfect observations, which lead to easier state abstraction and thus reduces the advantage of SKIL in this aspect.
- Most MetaWorld tasks are much simpler than real-world tasks, so that even the baselines could get high performance.
- The action dimension of DexArt tasks (22-Dof) are much higher than that of other tasks, so the much larger action space limits the performance of both SKIL and the baselines on these tasks..

TABLE VI: DexArt Results. We present the mean and standard deviation (std) of the success rates for each task.

Method/Task	<i>Bucket</i>	<i>Toilet</i>	<i>Faucet</i>	<i>Laptop</i>
DP	54.4(15.0)	36.7(12.5)	25.5(6.8)	28.9(1.6)
DP3	28.9(5.7)	45.5 (5.6)	14.4(3.1)	38.9(6.9)
RISE	52.2(12.2)	34.4(10.3)	11.1(1.6)	27.8(12.8)
GenDP-S	28.9(1.6)	32.2(4.2)	24.4(6.3)	43.3(2.7)
SKIL(Ours)	61.6 (1.6)	45.5 (5.6)	27.7 (4.1)	44.4 (4.2)

F. Real-world Experiment Results

1) *Visualization of Semantic Keypoints*: We visualize the semantic keypoints on different objects, as shown in Figure 13. The results demonstrate that temporal consistency is preserved across objects within the same category.

2) *Analysis of Grasp Wall of Cup*: Table I shows the baselines with point clouds input (e.g. DP3, RISE and GenDP-S) perform poorly on *Grasp Wall of Cup*. In contrast, DP achieves a success rate of 70% on the training objects. We

observe that the task requires grasping the inner wall of cup on the side away from the camera, but DP3, RISE and GenDP-S always perform grasping at a small distance from the cup. We suspect that the Zed2 stereo camera produces low-quality point clouds around the inner wall of the cups, due to the pure white color of that part. Thus, these methods with point clouds input observe the inaccurate positions of the cup wall. However, SKIL fuses the information of all keypoints using a transformer encoder, so that the perception errors on these small fraction of keypoints can be eliminated by their neighbours to some extent.

3) *Spatial Generalization of SKIL*: We find that our method can effectively handle objects located at the edges of the workspace, as shown in Figure 14. In contrast, baselines often fail to handle objects positioned at the corners of the workspace, especially when dealing with unseen objects. As shown in Figure 15, the SKIL method can precisely grasp the left corner of the towel, as demonstrated during the training. In contrast, while baselines can fold the towel, their policies often grasp the wrong region, missing the left corner.

4) *Environment generalization of SKIL*: Other than *Grasp Handle of Cup*, we also conduct experiments on environment generalization for *Pick Mouse* and *Fold Towel*. As shown in Table VIII and IX, all algorithms exhibit similar performance across these tasks. Therefore, we omit further analysis for *Pick Mouse* and *Fold Towel*. For *Hang Towel*, a long-horizon task, while all baselines fail to complete the task, SKIL achieves high performance under all situations.

5) *Long-horizon manipulation of SKIL*: Figure 17 and 18 illustrate the whole process of *Hang Towel* and *Hang Cloth* on a rack. Notice that baselines perform poorly on *Hang Towel*, and we show the classical failure mode of baselines in Figure 16. For example, the gripper cannot grasp the handle hook of hanger after folding the towel. This is because the position of the handle hook is disturbed behind folding the towel, as shown in Figure 17 (d, e, f). Due to limited demonstrations, baselines are unable to generalize to different positions of the handle. Besides, baselines occasionally skip necessary actions like folding the towel because of perception errors. Sometimes, wrong actions also occur. For example, the robot may put the towel directly onto the rack without the hanger. To conclude these, most failures result from high compounding errors in long-horizon tasks.

6) *Visualization of human videos*: We visualize the human videos in *Pick Mouse*, *Grasp the Handle of Cup*, and *Fold Towel*, as shown in Figure 19.

G. Ablation Study

1) *Ablation of Vision Foundation Models*: We present the results of different foundation models in three tasks, *Pick Mouse*, *Fold Towel*, and *Grasp Handle of Cup*. Table XI reports the success rate of the three tasks. The lowercase letters b, l, and g represent base, large, and giant, respectively, indicating different sizes of the ViT architecture. Similarly, the uppercase letters B, L, and H stand for Base, Large, and Huge, also denoting varying scales of the ViT architecture. Additionally,

TABLE VII: Metaworld Results. We present the mean and standard deviation (std) of the success rates for each task.

Method/Task	<i>Hammer</i>	<i>Handle-pull</i>	<i>Soccer</i>	<i>Box-close</i>	<i>Button-press</i>	<i>Window-open</i>
DP	54.3(6.1)	18.9(4.2)	21.1(4.2)	53.3(2.7)	100 (0)	85.5(3.2)
DP3	71.1(4.1)	62.2(3.2)	5.3(3.3)	70.3(2.3)	100 (0)	100 (0)
RISE	36.7(12.5)	17.8(3.1)	11.1(1.6)	43.3(9.8)	100 (0)	58.9(6.3)
GenDP-S	57.8(5.7)	31.1(6.9)	7.7(1.6)	51.1(3.1)	72.2(5.7)	46.7(5.4)
SKIL(Ours)	100 (0)	75.6 (4.2)	24.4 (3.1)	71.1 (8.7)	100 (0)	100 (0)



Fig. 13: Movement of semantic keypoints on different objects in SKIL. Green points represent the current keypoints, and the white flows show their previous trajectories. Temporal consistency is maintained across objects within the same category.

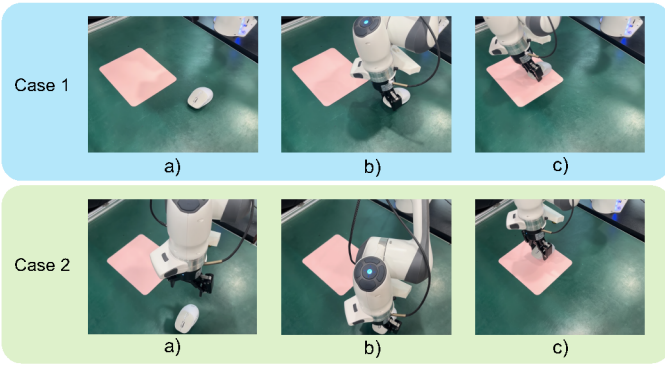


Fig. 14: SKIL successfully handles objects positioned at the edge of the workspace. We demonstrate its ability to grasp a mouse located in the corner of the workspace.

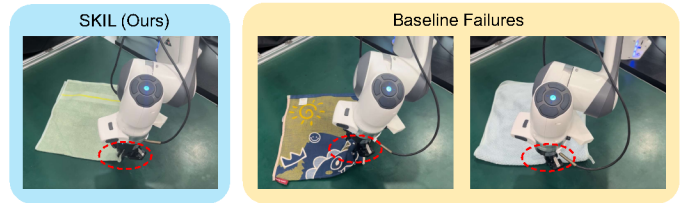


Fig. 15: SKIL accurately grasps the corner of the towel, whereas baselines struggle with precise grasping.

DiFT [43] utilizes Stable Diffusion 2.1 as its vision foundation model.

Figure 20 depicts the movement of semantic keypoints using different vision foundation models. We can see that the semantic keypoints obtained by DiFT [43] and RADIO [38] remain

relatively accurate during the evaluation, while the keypoints from DINOv2 [34] become detached from the relevant object's surface when meeting occlusions of keypoints.

2) *More Ablation Experiments: SKIL w/ tracking model.* We used a method similar to P3-PO [28], which utilizes CoTracker [22] to obtain keypoint positions. CoTracker has an offline mode that uses all frames and an online mode that uses only recent ones. In real-world settings, we use the online mode for fast inference. However, we found that its tracking performance deteriorates over time with an unknown video length, leading to a significant performance drop in long-

Baseline Failure Modes of *Hang Towel*



Fig. 16: Classical failure modes of baselines in *Hang Towel*, including grasping failure, skipping the folding of towel, and wrong action (directly putting the towel onto the rack without the hanger).

TABLE VIII: Average success rates of SKIL and baselines on the *Pick Mouse* task under three types of environmental changes. (*Situation 0* means without any environmental changes.)

Method	<i>Situation 0</i>	<i>Situation 1</i>	<i>Situation 2</i>	<i>Situation 3</i>
DP	2/10	2/10	0/10	0/10
DP3	3/10	1/10	3/10	0/10
RISE	1/10	0/10	1/10	0/10
GenDP-S	4/10	3/10	0/10	0/10
SKIL (Ours)	7/10	7/10	6/10	5/10

horizon tasks, as shown in Table XII. **SKIL w/o ensemble.** Table XII shows a nearly 27% performance drop across three tasks. The action ensemble strategy reduces motion jitter caused by mismatched keypoints. **SKIL w/o similarity vector.** The results can be found in Table XII. For *Grasp Handle of Cup*, it performs worse than SKIL. This is because the similarity vector contains the similarity between the current keypoints and the reference keypoints on the handle. As a result, the policy can distinguish keypoints on the handle from other keypoints, enabling a functional grasp.

SKIL and baselines with 20 ~ 100 demos. See Table XIII. As the amount of demo increases, SKIL achieves better performance by learning smoother actions with less motion jitter, for both in-domain and out-of-domain settings. Other baselines also benefit from more demos but struggle to match SKIL’s performance, especially on the out-of-domain setting.

H. Details of Diffusion Action Head

We provide a detailed formulation of the diffusion model as follows. Note that we refer to some formulations in [30].

First, the denoising process is represented as:

$$\mathbf{a}_t^{k-1} = \frac{\sqrt{\bar{\beta}^{k-1}}\gamma^k}{1 - \bar{\beta}^k}\mathbf{a}_t^0 + \frac{\sqrt{\beta^k}(1 - \bar{\beta}^{k-1})}{1 - \bar{\beta}^k}\mathbf{a}_t^k + \tau^k\mathbf{v}$$

Here, the parameters $\{\beta^k\}_{k=1}^K$ and $\{\tau^k\}_{k=1}^K$ are scalar coefficients from a predefined noise schedule. The terms are defined as $\gamma^k := 1 - \beta^k$ and $\bar{\beta}^{k-1} := \prod_{i=1}^{k-1} \beta^i$. Additionally, $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ when $k > 1$; otherwise, $\bar{\beta}^{k-1} = 1$ and $\mathbf{v} = \mathbf{0}$.

The network is trained by minimizing the mean-squared

TABLE IX: Average success rates of SKIL and baselines on the *Fold Towel* task under three types of environment changes. (*Situation 0* means without any environmental changes.)

Method	<i>Situation 0</i>	<i>Situation 1</i>	<i>Situation 2</i>	<i>Situation 3</i>
DP	6/10	5/10	5/10	3/10
DP3	6/10	6/10	5/10	2/10
RISE	5/10	4/10	2/10	4/10
GenDP-S	6/10	5/10	4/10	4/10
SKIL (Ours)	8/10	7/10	7/10	8/10

TABLE X: Average success rates of SKIL on the *Hang Towel* task under three types of environment changes. (*Situation 0* means without any environmental changes.)

Method	<i>Situation 0</i>	<i>Situation 1</i>	<i>Situation 2</i>	<i>Situation 3</i>
Baselines	0/10	0/10	0/10	0/10
SKIL (Ours)	7/10	7/10	7/10	6/10

error (MSE) between the predicted and true actions:

$$\mathcal{L}(\phi) := \text{MSE} \left(\mathbf{a}_t, D_{\theta} \left(\mathbf{o}_t, \sqrt{\bar{\beta}^k}\mathbf{a}_t + \sqrt{1 - \bar{\beta}^k}\epsilon, k \right) \right)$$

where $k \sim \text{Uniform}(\{1, \dots, K\})$, $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $(\mathbf{o}_t, \mathbf{a}_t)$ is sampled from the training dataset. For simplicity, noisy action inputs are denoted as $\tilde{\mathbf{a}}_t := \sqrt{\bar{\beta}^k}\mathbf{a}_t + \sqrt{1 - \bar{\beta}^k}\epsilon$, where the index k is omitted for clarity.

I. Hyperparameters

We report the main hyperparameters for SKIL and SKIL-H in Table XV and Table XVI, respectively. They share the same hyperparameters of the transformer encoders, as listed in Table XIV. Following DP3 [58], we set the action prediction and execution horizon to be $H = 4$ and $N_{act} = 2$ in MetaWorld [55] and DexArt [1]. We omit the diffusion model parameters in Table XVI since the same parameters are used for both SKIL and SKIL-H.

All models are trained on 2 NVIDIA 3090 GPUs, and the checkpoint with the lowest validation loss is saved as the final model for real-world performance evaluation. In simulation, we compute the average of the top three success rates every 100 training epochs.

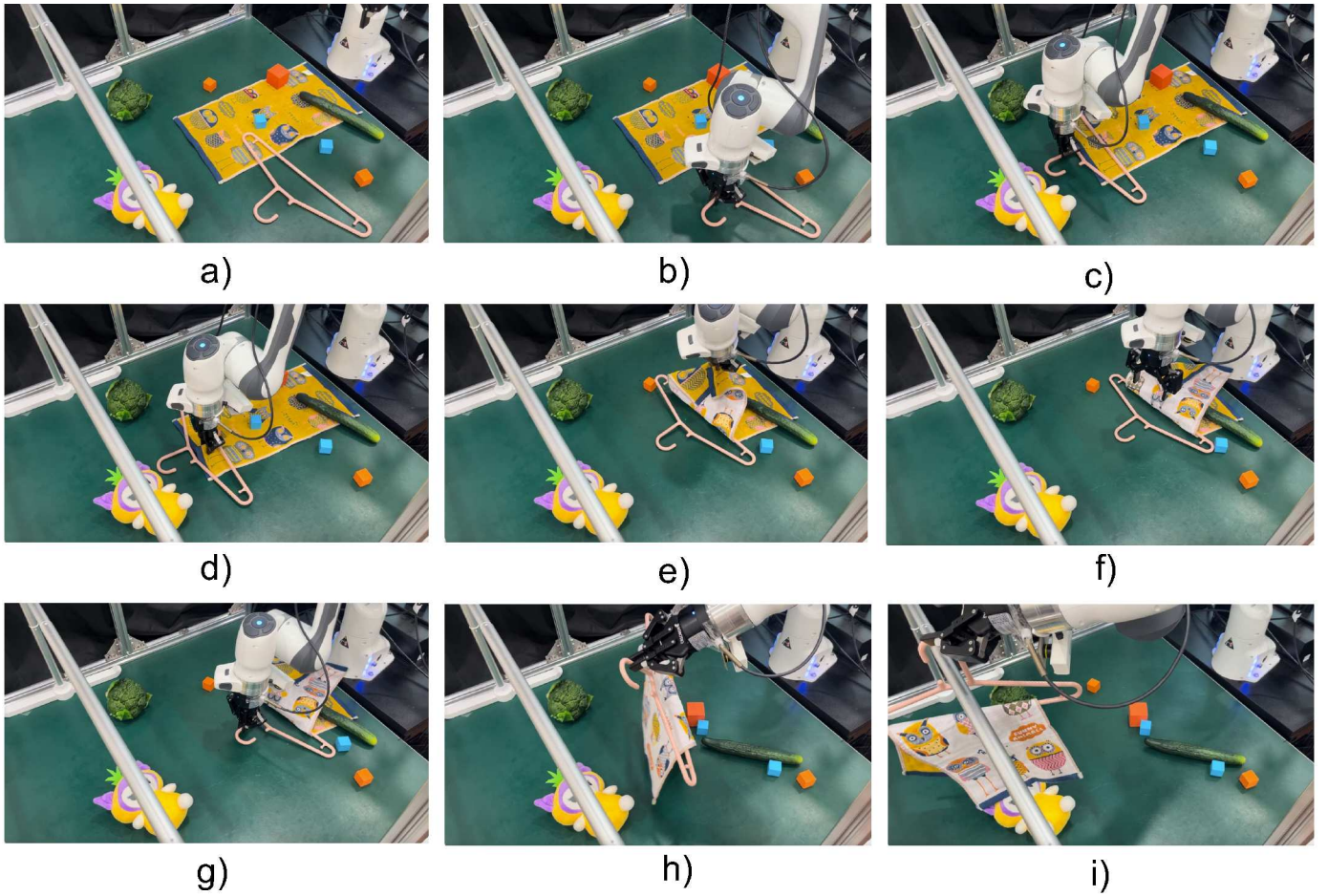


Fig. 17: Videosnaps of *Hang Towel* using SKIL. We can see that this task includes multiple stages: grasping the handle of hanger (a-b), placing the hanger on the towel’s edge (b-c), grasping and folding the towel (d-f), grasping the handle hook (f-g), and finally hanging it on the rack (h-i).

TABLE XI: Average success rates of various vision foundation models.

Method	<i>Pick Mouse</i>	<i>Fold Towel</i>	<i>Grasp Handle of Cup</i>
DINOv2-b	4/10	5/10	7/10
DINOv2-l	4/10	6/10	6/10
DINOv2-g	5/10	7/10	6/10
RADIOv2.5-B	6/10	6/10	9/10
RADIOv2.5-L	7/10	8/10	9/10
RADIOv2.5-H	7/10	7/10	9/10
DiFT	8/10	7/10	10/10

TABLE XII: Ablation results in the real-world. We conduct five trials with random initialization for each training object.

Method	<i>Pick Mouse</i>	<i>Grasp Handle of Cup</i>	<i>Hang Towel</i>
SKIL w/o ensemble	5/10	7/10	4/10
SKIL w/o sim. vector	7/10	6/10	5/10
SKIL w/ track model	8/10	7/10	0/10
SKIL (Ours)	8/10	9/10	7/10

TABLE XIII: Realworld results with different numbers of demonstrations. For each object, we conduct five trials with random initialization.

Method/Task Number of Demos	<i>Pick Mouse</i>			<i>Grasp Handle of Cup</i>		
	20	50	100	20	50	100
In Domain (2 training objects)						
DP	2/10	5/10	6/10	3/10	7/10	9/10
DP3	2/10	4/10	5/10	5/10	9/10	9/10
SKIL (Ours)	7/10	8/10	8/10	9/10	10/10	10/10
Out of Domain (3 testing objects)						
DP	2/15	3/15	3/15	5/15	7/15	8/15
DP3	3/15	6/15	8/15	7/15	10/15	10/15
SKIL (Ours)	9/15	10/15	11/15	12/15	13/15	13/15

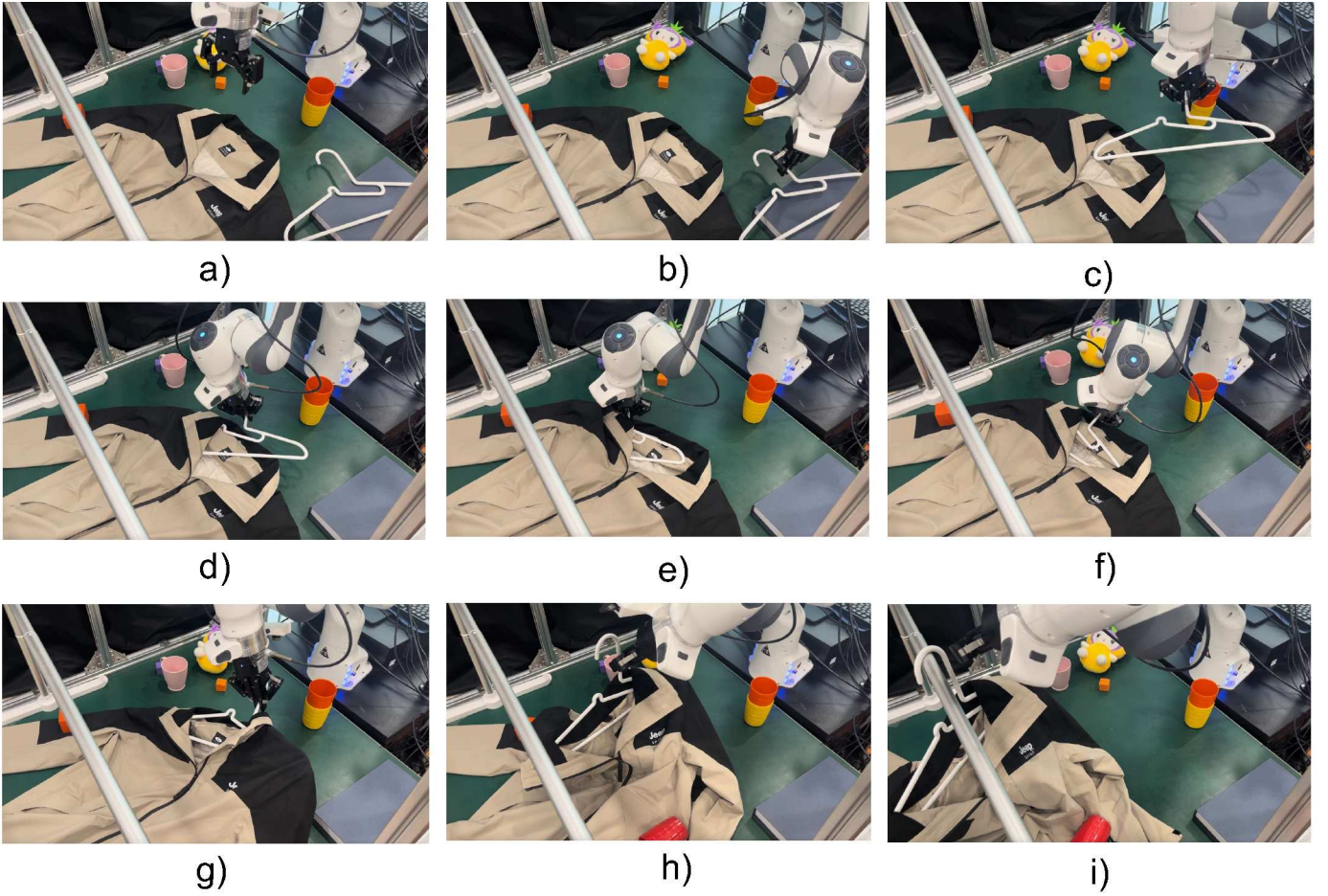


Fig. 18: Videosnaps of *Hang Cloth* using SKIL. Notice that the cloth and scene are unseen for the policy. The detailed stages include grasping the handle of hanger (a-b), inserting the hanger into the cloth (c-f), and hanging it onto the rack (g-i).

TABLE XIV: Main Parameters of the Transformer Encoder.

Parameter	Transformer Encoder
Number of layers	1
Hidden size	128
Number of attention heads	8
Feed-forward size	512
Dropout rate	0.1
Activation function	ReLU
Attention type	Self-Attention
Layer normalization	Post-LN
Embedding size	128
Positional encoding	Sinusoidal

TABLE XV: Hyperparameters of SKIL.

Hyperparameters	SKIL
Epoch	1000
Batch size	256
Optimizer	AdamW
Learning rate	1e-4
Weight decay	1e-6
Lr scheduler	Cosine
Diffusion Head in Policy Module	
Noise scheduler	DDIM
Denoising steps	100(train); 10(test)
Prediction horizon	16 (4 in simulation)
Observation horizon	4 (2 in simulation)
Action horizon	8 (2 in simulation)

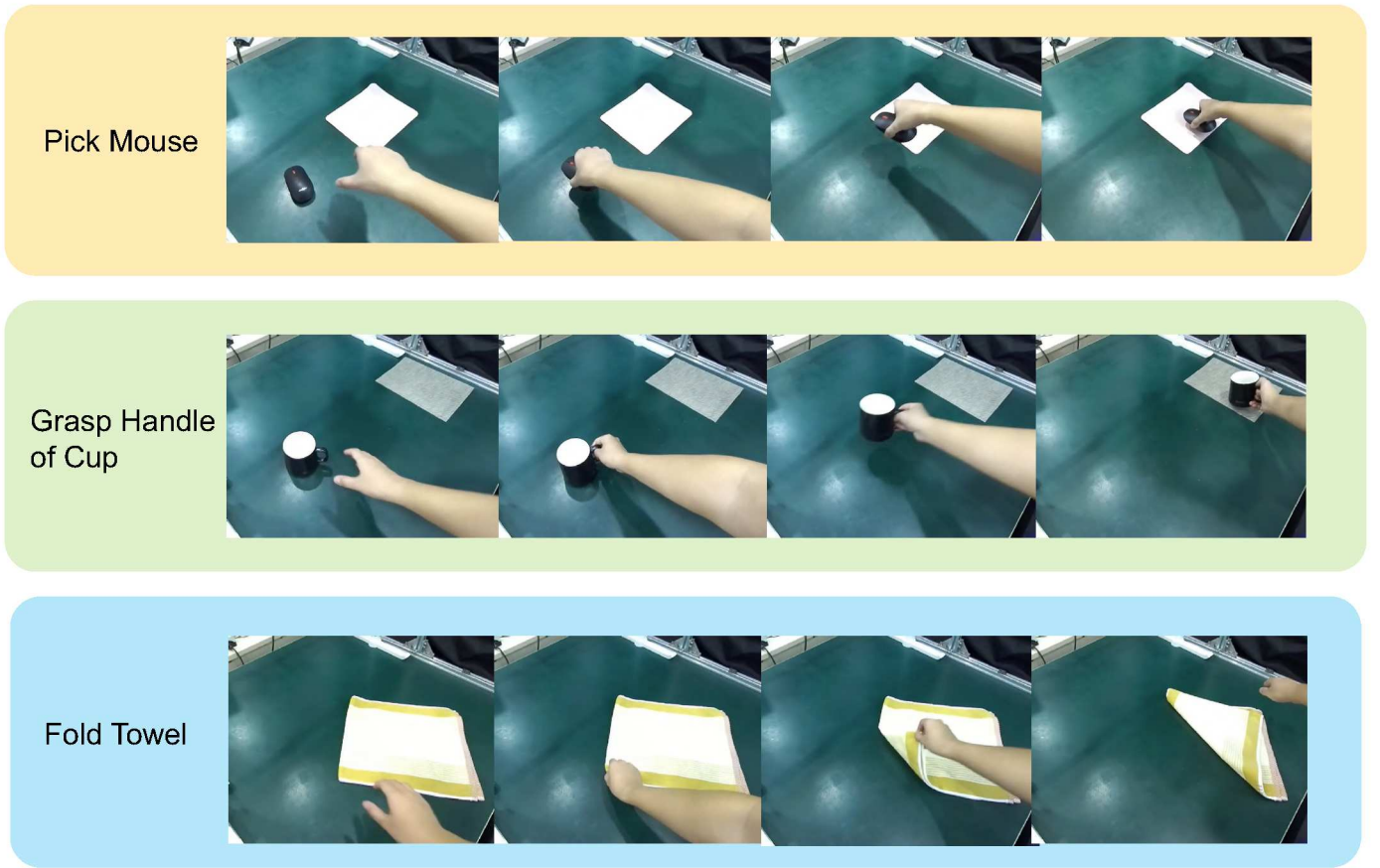


Fig. 19: Videosnaps of human demos collected in *Pick Mouse*, *Grasp Handle of Cup*, and *Fold Towel*.

TABLE XVI: Hyperparameters of SKIL-H.

Hyperparameters	SKIL-H
Epoch	1000
Batch size	256
Optimizer	AdamW
Learning rate	1e-4
Weight decay	1e-6
Lr scheduler	Cosine
Diffusion Head in Trajectory Prediction Module	
Noise scheduler	DDIM
Denoising steps	100(train); 10(test)
Prediction horizon	8
Observation horizon	2
Action horizon	4

DiFT



RADIO
v2.5-H



DINOv2-g



Fig. 20: Visualization of keypoints' movement using three vision foundation models on *Grasp Handle of Cup*.



Fig. 21: Visualization of training and testing objects. We use 2 objects for training and 10 objects for testing on short-horizon tasks, including *Pick Mouse*, *Grasp Handle of Cup*, *Grasp Wall of Cup*, and *Fold Towel*.



Fig. 22: Visualization of training and testing objects. We use 2 objects for training and 3 or 5 objects for testing on the long-horizon tasks, including *Hang Towel* and *Hang Cloth*.