ATK: Automatic Task-driven Keypoint Selection for Robust Policy Learning

Yunchu Zhang Shubham Mittal Zhengyu Zhang Liyiming Ke
Siddhartha Srinivasa Abhishek Gupta

Paul G. Allen School of Computer Science and Engineering
University of Washington

Abstract: Visuomotor policies often suffer from perceptual challenges, where visual differences between training and evaluation environments degrade policy performance. Policies relying on state estimations, like 6D pose, require task-specific tracking and are difficult to scale, while raw sensor-based policies may lack robustness to small visual disturbances. In this work, we leverage 2D keypoints — spatially consistent features in the image frame — as a flexible state representation for robust policy learning and apply it to both sim-to-real transfer and real-world imitation learning. However, the choice of which keypoints to use can vary across objects and tasks. We propose a novel method, ATK, to automatically select keypoints in a task-driven manner so that the chosen keypoints are predictive of optimal behavior for the given task. Our proposal optimizes for a minimal set of keypoints that focus on task-relevant parts while preserving policy performance and robustness. We distill expert data (either from an expert policy in simulation or a human expert) into a policy that operates on RGB images while tracking the selected keypoints. By leveraging pre-trained visual modules, our system effectively encodes states and transfers policies to the real-world evaluation scenario despite wide scene variations and perceptual challenges such as transparent objects, fine-grained tasks, and deformable objects manipulation. We validate ATK on various robotic tasks, demonstrating that these minimal keypoint representations significantly improve robustness to visual disturbances and environmental variations. See all experiments and more details on our website.

1 Introduction

Though powerful in principle, visuomotor policy learning in practice often requires a significant number of samples to learn robust, generalizable policies [1, 2, 3, 4, 5]. To make this paradigm more practical, many methods use *pretrained visual representations* [6, 7, 8]; these representations, often obtained through self-supervised learning objectives (such as reconstruction [9], future prediction [10] or contrastive learning [6, 11, 12]), improve sample efficiency and robustness in many domains. However, despite this pretraining, the resulting policies can remain *brittle*, i.e., responsive to distractors, object changes, and lighting changes, making them difficult to broadly deploy [13, 14]. This problem raises the question that motivates our research: *How can we design general-purpose yet tailorable representations of visual input that make policies robust to environmental variations but still transferable across scenarios?*

In this work, we propose the use of *keypoints* —a set of 2D pixel points in RGB images that can be tracked over time—as general-purpose visual state representations for robotic manipulation policies. Extensively used in computer vision [15, 16], keypoints track specific semantically meaningful points on an image and have demonstrated robustness even in the presence of occlusion, lighting changes, and scale variations. Unlike pose-based methods, they do not rely on rigid structures, making them more suitable for tracking articulated and deformable objects. Additionally, keypoints naturally

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

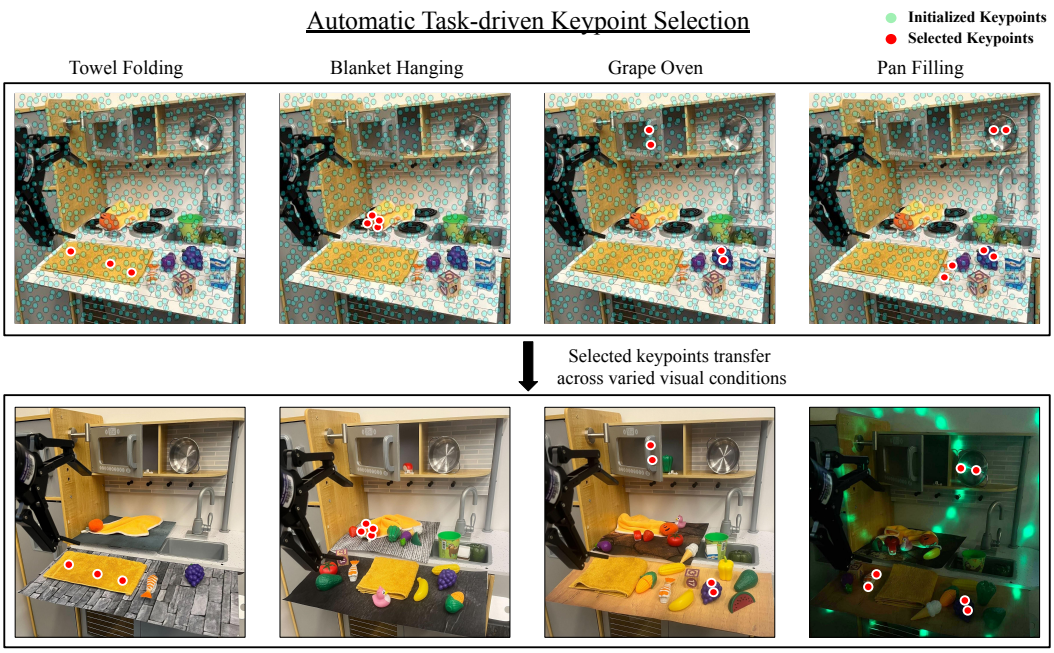


Figure 1: **(Top:)Automatically selected keypoint representations for different tasks in the same scene.** Scene representations vary depending on the desired functionality. **(Bottom:)** Robust generalization of policies learned keypoint representations to various positions, backgrounds, distractors, and lighting changes.

generalize across extreme object appearances, such as transparent, reflective, or fine-grained. Recent advances in keypoint tracking, driven by models trained on large-scale web data [15, 17], reveal that keypoint tracking is surprisingly robust across diverse visual domains. These advantages support our contention that, when chosen appropriately, keypoints are promising candidates for powerful, robust visual representations.

Having established the value of keypoints, we next ask: *What is the minimal set of task-relevant keypoints that can serve as an effective state representation for decision making?* Simply using all keypoints in a scene leads to inefficient redundancy, increases computational burden, and complicates tracking due to occlusion and point interference. Random sampling of points or selecting too few points risks overlooking critical task-relevant information, making the optimal policy unrealizable in the representation class. Crucially, the ideal set of keypoints varies from task to task, as shown in Fig 1. Each task requires focusing on different parts of the scene, indicating that *the minimal set of keypoints must be inherently task driven*.

Our key insight, then, is that task objectives should inform both the selection of compact keypoint representations and the training of an optimal policy. We infer that a suitable task-driven representation is the *minimal* set of keypoints that can sufficiently predict the optimal policy. Intuitively, this suggests that points that are not predictive of optimal actions can be dropped since they are not necessary for decision making. However, such reasoning creates a chicken-and-egg problem: a robust state representation is needed to learn the optimal policy, yet optimal policies are necessary to identify the appropriate state representation. Fortunately, when synthesizing policies via common data-driven learning methods, e.g., imitation learning [18] or student-teacher distillation of policies for sim-to-real transfer [19, 20], optimal actions are available to inform the choice of minimal, task-relevant keypoints. We propose a distillation-based algorithm, ATK, that uses a masking architecture to jointly select a minimal set of task-relevant keypoints and train a keypoint conditional policy via supervised learning. This minimal, task-specific keypoint representation retains the necessary task-

relevant information, making the resulting policies naturally robust to environmental variations and transferable across scenarios with considerable visual differences, e.g., sim-to-real transfer.

In sum, this work contributes:

- A methodology for jointly **selecting a minimal set of task-relevant keypoints** and learning a policy conditioned on these keypoints.
- An empirical validation of this approach across a variety of real-world robot manipulation tasks in the **sim-to-real** setting, demonstrating robust sim-to-real transfer in settings with considerable visual variety.
- A demonstration of the efficacy of the proposed methodology in the **imitation learning** setting, with the resulting policies showing strong visual generalization while retaining high-precision dexterity.

2 Problem Formulation

We study decision making in finite-horizon Markov Decision Processes (MDPs) defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, \rho_0, \mathcal{R}, \gamma)$, where $\mathcal{S}$ represents the Lagrangian state space (the compact, physical state of the system), $\mathcal{O}$ is the observation space, $\mathcal{A}$ is the action space, $\mathcal{P}(s'|s, a)$ defines the transition dynamics, ρ_0 is the initial state distribution, $\mathcal{R}$ is the reward function, and γ is the discount factor. In simulation, agents have access to the Lagrangian state $\mathcal{S}$, which provides a compact, complete description of the environment (e.g., object positions, velocities, etc.). In the real world, agents can access only sensor observations $\mathcal{O}$ (e.g., RGB images). Although the real world might be partially observable, we assume that the current observation $o \in \mathcal{O}$ is sufficient to make optimal decisions. The observation o is produced by an invertible emission function f, such that $o = f(s)$. Our goal is to derive a visuomotor policy π_θ that is near optimal in the real world when acting on observations o_t.

Simulation to Reality Transfer. We aim to derive policies that operate from perceptual inputs for transfer from simulation to reality. Though we have "privileged" information s_t in simulation that can enable rapid learning of (near) expert policies $\pi^*(\cdot|s_t)$ via standard decision-making algorithms (including imitation learning, reinforcement learning, trajectory optimization, or motion planning), transferring these *perception-based* policies $\pi_\theta(\cdot|o_t)$ from simulation to the real world operation is difficult. A key challenge is the perceptual gap between simulation and the real world. This can be formalized using two MDPs: $\mathcal{M}_{\text{sim}} = (\mathcal{O}_{\text{sim}}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \rho_0, \mathcal{R}, \gamma)$ for simulation, and $\mathcal{M}_{\text{real}} = (\mathcal{O}_{\text{real}}, \mathcal{A}, \mathcal{P}, \rho_0, \mathcal{R}, \gamma)$ for the real world. The same underlying state s goes through different emission functions and leads to different observations (such as RGB images) in simulation, $o_{\text{sim}} = f_{\text{sim}}(s)$, vs the real world, $o_{\text{real}} = f_{\text{real}}(s)$. The challenge in transferring end-to-end visuomotor policies $\pi^*(a_t|o_t)$ from simulation to the real world lies in the mismatch between $\mathcal{O}_{\text{sim}}$ and $\mathcal{O}_{\text{real}}$.

Imitation Learning. The imitation learning setting is provided with an offline dataset of expert behavior data in lieu of a simulator. The dataset $\mathcal{D} = \{(o_i, a_i)\}_{i=1}^M$ is drawn from an expert – π^*. Imitation learning settings both train and test in the real world from raw camera observations. However, for imitation learning to be visually "robust," we consider training on a setting with one emission function, $f_{\text{train}}(s)$, and evaluating with another emission function, $f_{\text{test}}(s)$. We can formulate this via a simple dual MDP formulations: $\mathcal{M}_{\text{train}} = (\mathcal{O}_{\text{train}}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \rho_0, \mathcal{R}, \gamma)$ for training and $\mathcal{M}_{\text{test}} = (\mathcal{O}_{\text{real}}, \mathcal{A}, \mathcal{P}, \rho_0, \mathcal{R}, \gamma)$ for evaluation. The same underlying state s goes through different emission functions and leads to different observations at training, $o_{\text{train}} = f_{\text{train}}(s)$, and testing, $o_{\text{test}} = f_{\text{test}}(s)$. For instance, this approach could involve training for robotic manipulation on a clear tabletop and then testing with changes in background, lighting or distractor objects. The challenge lies in the mismatch between $\mathcal{O}_{\text{train}}$ and $\mathcal{O}_{\text{test}}$.

To address both challenges, we select a state representation that retains invariance between simulation and real-world $g_{\text{sim}}(o_{\text{sim}}) = g_{\text{real}}(o_{\text{real}})$, or train and test observations $g_{\text{train}}(o_{\text{train}}) = g_{\text{test}}(o_{\text{test}})$. Though many such choices are feasible, this work focuses on keypoint-based representations.

3 Task-Driven Automatic Keypoint Selection for Robust Policy Learning

We aim to provide an input representation that can enable policy generalization and robustness for the transfer settings mentioned in Section 2. To this end, we propose the use of **2D keypoints** as the perceptual representation for sim-to-real transfer (Sec. 3.1). The crux of our proposal lies in transferring only task-relevant parts of the observation by automatically *selecting* a set of task-relevant keypoints. We propose an algorithm that integrates keypoint selection with policy training using a distillation process that relies on expert data to propose keypoints and obtain corresponding keypoint-based policies (Sec. 3.2). Finally, we explain how to deploy these chosen keypoints and their trained policies in real-world evaluations (Sec. 3.3).

3.1 Keypoints as Policy Representations

Keypoints, a widely used representation in computer vision, are salient locations in an image that are useful for identifying and describing objects or features. Formally, a keypoint is defined as a specific position $k_i^t = (x_i^t, y_i^t)$ in the 2D image plane at time t. A set of N keypoints, $\{k_i^t\}_{i=1}^N = (x_1^t, y_1^t, ..., x_N^t, y_N^t)$, provides a compact scene representation. The number and selection of keypoints can be dynamically adjusted based on task complexity and requirements. However, what makes keypoints impactful for robust policy learning is the proliferation of robust tracking algorithms [16, 15], trained on web-scale data, that maintain dense correspondences across frames despite visual scene-level and instance-level variations. We bring this robustness to bear on policy learning. To track keypoints over time, we initialize keypoints $\{k_i^t\}_{i=1}^N$ at $t = 0$ and then use tracking methods [21, 16, 15] to maintain robust semantic correspondences of these points across time steps. We formalize tracking as a correspondence function $h_{\mathcal{C}}$ that updates keypoint locations at each time step while providing correspondence measurement scores.

Given a set of initial keypoint positions, $\{k_i^t\}_{i=1}^N = (x_1^t, y_1^t, ..., x_N^t, y_N^t)$, a particular set of their positions at time t is updated as: $\{k_i^t\}_{i=1}^N = h_{\mathcal{C}}(\{k_i^{t-1}\}_{i=1}^N, I_t)$, where I_t is the current image. The observation at time t, o_t, is an (ordered) set of N keypoints $\{k_i^t\}_{i=1}^N$ that is updated iteratively through the correspondence function $h_{\mathcal{C}}$. The keypoint locations $k_i^t = (x_i^t, y_i^t)$ correspond to the *current* planar positions of points that semantically correspond to the initially chosen points $\{k_i^0\}_{i=1}^N$.

The core challenges in leveraging keypoints as a policy representation are the selection of initial keypoints $\{k_i^0\}_{i=1}^N$, the tracking of them through time, and the transferring of resulting keypoint-based policies across widely varying deployment scenarios. The choice of keypoints is inherently task-specific since different tasks require focusing on distinct elements in the scene. For example, in the kitchen scene shown in Fig 1, the keypoints on the blanket are crucial for the blanket-hanging task, whereas the pan-placement and grasping tasks require keypoints on both the pan and other objects.

3.2 Automatic Task-Driven Keypoint Selection: Training

Our work selects a *minimal* set of task-relevant keypoints as a representation to enable robust policy transfer. Formally, we aim to identify a minimal set of N task-relevant keypoints, $\{k_i\}_{i=1}^N$, that enable training a near-optimal policy while being easily tracked with $h_{\mathcal{C}}$. Our keypoint selection is based on two criteria: (1) **realizability of the optimal policy**, i.e., the selected keypoints must capture all necessary information to learn a near optimal policy for the task, and (2) **trackability**, i.e., the chosen keypoints must be reliably and consistently trackable using an available correspondence function $h_{\mathcal{C}}$. Realizability and trackability are naturally interconnected concepts; representations that are not trackable are not consistent, making it impossible to realize an optimal policy. *Given the full set of N candidate keypoints $\{k_i^t\}_{i=1}^N$, how do we select a minimal set of K keypoints, with $K \ll N$, to satisfy realizability and trackability?*

Our insight is that the key workhorse in both imitation learning and sim-to-real distillation—*supervised action prediction*—can directly inform both keypoint selection and subsequent policy learning. Given a dataset of expert observation-action tuples $\mathcal{D} = \{(o_i, a_i)\}_{i=1}^M$, we typically learn a visuomotor policy via supervised learning: $\theta^* = \arg\min_\theta \frac{1}{M} \sum_{i=1}^M \left\| \pi_\theta(o_i) - a_i \right\|^2$. This can be

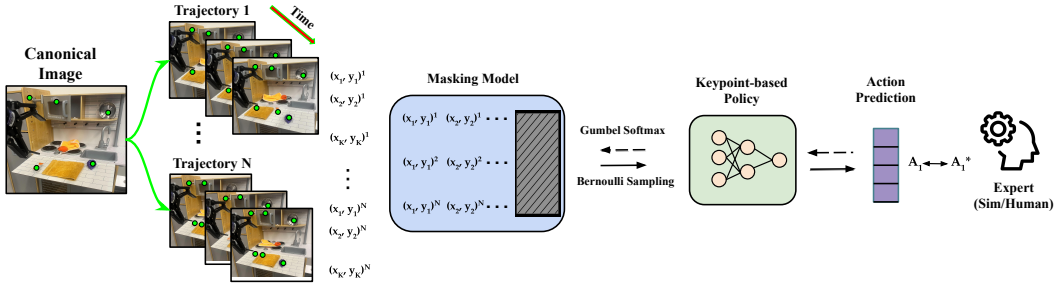


Figure 2: **ATK automatically selects minimal yet necessary information for task execution** by distilling expert data (either from an expert policy in simulation or a human expert) into a policy that operates on a selective subset of keypoints and optimizing the selection mask. Once the keypoints are identified, they are transferred from the training set to the real-world evaluation scenario. Finally, the keypoint-based policy is transferred to the evaluation scenario, taking as input RGB images while tracking the transferred keypoints.

easily generalized to more expressive maximum likelihood objectives [5, 1]. Supervised learning objectives underpin both imitation learning and sim-to-real distillation. For imitation learning, the expert dataset comes from a human expert; in sim-to-real distillation, it comes from a privileged expert in simulation. In this work, we propose a *masking-based keypoint selection mechanism* that uses gradients from supervised learning to select a minimal set of task-relevant keypoints.

To ensure that each keypoint remains semantically and spatially consistent across trajectories, we first identify a single image frame that captures the complete task context for solving the task, denoted as the *canonical template* I_{can}. We then randomly sample C candidate keypoints $\{k_j^0\}_{j=1}^C$ on the canonical image I_{can}. Given an expert dataset $\mathcal{D} = \left\{(o_i, a_i)\right\}_{i=1}^M$, we use the correspondence function $h_{\mathcal{C}}$ to align and propagate these candidates across all trajectories and timesteps, producing a keypoint-annotated dataset $\mathcal{D}_k = \left\{(o_{i,t},\, K_{i,t},\, a_i)\right\}_{i=1,t=1}^{M,T_i}$, where $K_{i,t}$ is the set of *all* tracked keypoint positions for observation $o_{i,t}$. From $\mathcal{D}_k$, we jointly learn a sparse masking model $\mathbb{M}_\phi$ and a downstream keypoint-based policy π_θ^k. As shown in Fig. 2, the M candidate keypoints are fed into $\mathbb{M}_\phi$, which outputs M independent Bernoulli probabilities, with each representing the likelihood of retaining the corresponding keypoint from the input candidate set. Sampling a binary mask $m \in \{0,1\}^M$ then zeroes out the unselected keypoints, yielding a reduced set $\tilde{K}$. Finally, $\tilde{K}$ is passed to π_θ^k, which produces the action distribution, as is standard in imitation learning. This unified architecture first applies pointwise masking to select the minimal keypoint subset and then predicts actions from those selected points.

To select the *minimal* set of K keypoints, we enforce a sparse information bottleneck on the mask:

$$\min -\mathbb{E}_{(k,a)\sim\mathcal{D}} \log \pi_\theta^k\left(a_t^* \,\middle|\, \mathbb{M}_\phi\big(\{k_i^t\}_{i=1}^N\big)\right) + \alpha\|\mathbb{M}_\phi(\{k_i^t\}_{i=1}^N)\|_1.$$

Intuitively, this training procedure filters out points that are (1) irrelevant to predicting optimal actions and (2) challenging to track using $h_{\mathcal{C}}$ since their representations over time $(\{k_i^t\}_{i=1}^N)$ are unreliable markers of optimal actions. Since the mask sampling is discrete and non-differentiable, we employ the Gumbel-softmax relaxation [22] to enable gradient-based optimization.

3.3 Inference with Task-Driven Keypoints

The preceding procedure learns a masking model $\mathbb{M}_\phi$ and a policy $\pi_\theta^k\left(a_t^* \,\middle|\, \mathbb{M}_\phi\big(\{k_i^t\}_{i=1}^N\big)\right)$. How can we use them for robust policy inference at test time? Whether it be a transfer from simulation to reality or from one imitation learning scenario at training time to another at test time, the inference procedure remains the same. For inference, it is important to transfer the minimal set of keypoints selected at training time to a variety of visually diverse test time scenarios. Since tracking of keypoints is performed by robust web-scale visual trackers [16, 15], once the *initial*

selected set of keypoints is identified in the real world at test time, subsequent tracking is not affected by the visual gap. This lets us focus solely on transferring the *initial* set of keypoints. Assuming the correspondence function $h_{\mathcal{C}}$ provides confidence scores for each candidate pair, at test time we select the initial-image points with the highest scores relative to the training canonical keypoints $\{k_i\}_{i=1}^{N}$, ensuring accurate matches. These transferred keypoints are then tracked via $h_{\mathcal{C}}$ and used directly as input to the deployed policy π_θ without requiring additional masking at test time. Though many implementations of $h_{\mathcal{C}}$, we adopt the method using diffusion-feature [17, 23] for initial set transfer from canonical image due to its strong visual robustness and reliable matching, and employ Cotracker [16] for subsequent tracking. We provide the pseudocode for applying ATK on a new task in Algorithm 1 and recommended hyperparameter selection in Table 7.

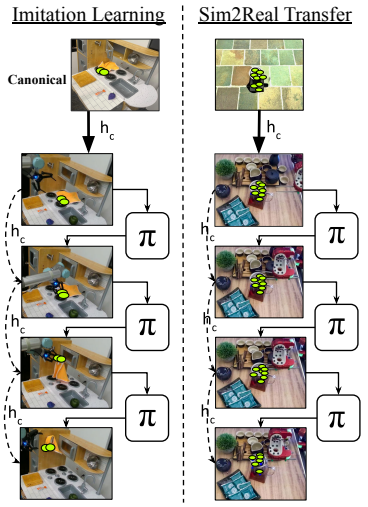


Figure 3: **Inference loop with keypoint-based representations.**

4 Experiment Evaluation and Results

Our experiments aim to answer the following questions: **(1) Sim-to-real transfer:** How well do the keypoints and policies learned in simulation *transfer* to the real world? **(2) Imitation robustness:** How well does the proposed keypoint selection and policy learning method work in the imitation learning setting? **(3) Interpretability and task-relevant features:** Are the chosen keypoints interpretable and relevant to different task objectives in multi-functional environments?

4.1 Experiment Setup

4.1.1 Simulation-to-Reality Transfer

Tasks and Challenges. We consider *three fine manipulation tasks for quantitative analysis*, shown in Fig 4. (1) The *sushi pick-and-place task* requires grasping a piece of sushi in a cluttered environment with distracting objects. (2) The *glasspot tip lifting task* requires precise grasping and lifting of the tip of a glass pot. This task is particularly challenging due to the pot's reflective surface and the small size of the tip. (3) The *clock manipulation task* contains two distinct subtasks: turning the button at the top of the clock or turning the clock hand on its surface, requiring task-specific representations for manipulation of articulated objects. Each task involves the challenges of tracking difficulty, precision of manipulation, persistence of task-specific focus, and management of variations in scene configuration.

4.1.2 Robust Imitation Learning

Tasks and Challenges. We consider *four manipulation tasks for quantitative analysis* in a multifunctional kitchen environment as shown in Fig 4. (1) The *grape-oven* task requires grasping a small grape toy and placing it in the microwave. (2) The *blanket hanging* task requires picking up and hanging a deformable object on a hook. (3) The *towel folding* task involves manipulating precise deformable objects without many distinct markers. (4) The *pan filling* task involves transporting and placing a pan on the burner and then placing two different objects, sushi and grapes, into this pan. Notably, each task occurs in the same environment, making it natural to have a focused task-specific representation that is specified to each problem.

4.1.3 Baselines and Evaluation

Baselines. For both sim-to-real and imitation learning experiments, we compare our approach to two groups of three baselines. *(1) Input modality:* Policies trained with different input types: **RGB images**, **Depth images**, and **Point clouds**. We use pre-trained visual encoders [8] for RGB and Depth

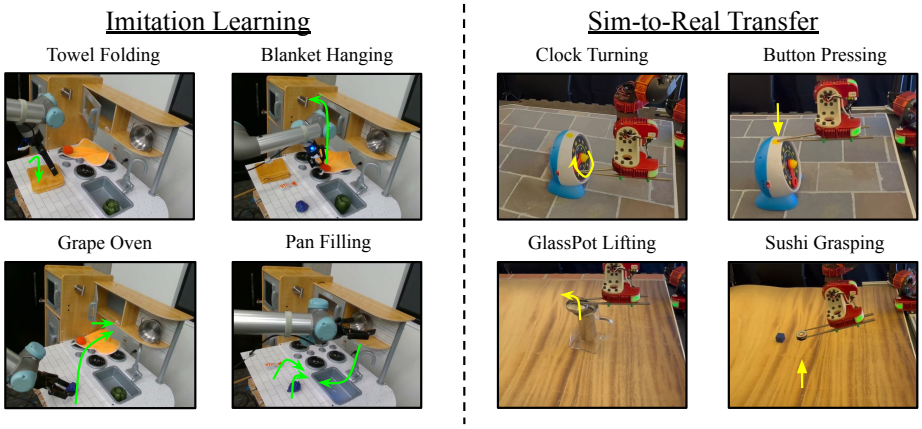


Figure 4: **Overview of evaluation tasks.** *Left:* Tasks used in an imitation learning setting. *Right:* Tasks used to assess sim-to-real transfer.

baselines and train pointcloud baselines end-to-end. For detailed training details and architecture, see Appendix D. *(2) Keypoint selection methods:* We consider three more baselines: **FullSet** uses all sampled keypoints across the image plane; **Random Select** randomly selects the same number of keypoints as our method; and **GPTSelect** uses GPT-4 to select the same number of keypoints based on the image and task.

Evaluation. For each setting, we evaluate each agent on 20 trajectories in the real world, all with varying initial configurations. We illustrate a small range of the randomizations in Fig 1 and a complete range of randomizations during evaluations in the Appendix. To assess robustness and generalization, we introduce disturbances: **RP** (random object poses), **RB** (background texture shuffling), **RO** (random distractor objects), and **Light** (altered lighting). For further details on these components, see Appendix.D.

4.2 Sim-to-Real Transfer of Keypoint-based Policies

ATK **transfers from sim-to-real demonstrating visual robustness.** As shown in Fig 5, keypoint-based policies maintain high success rates in the real world compared to alternative modalities, showcasing strong resilience against randomized object poses or background variations. We provide aggregate performance across different distractors, providing a detailed per-disturbance breakdown in the Appendix. E. Although extreme distractions, such as flashing light or occlusions, can disrupt tracking and decrease performance, ATKconsistently outperforms RGB, depth, and point-cloud based policies at transfer. The gap is worth noting in tasks involving transparent objects (e.g., glass) and fine-grained manipulation (e.g., clock tasks).

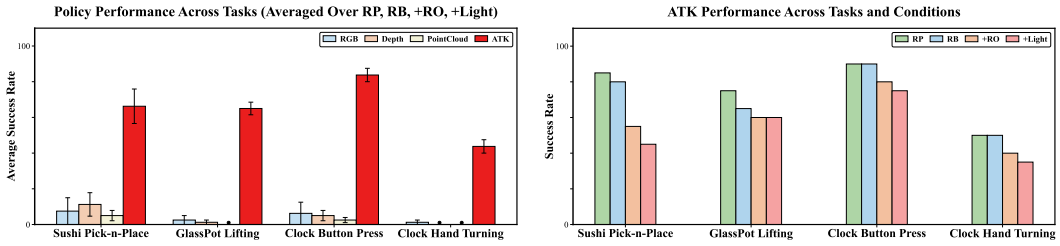


Figure 5: **Sim-to-real policy success rates in real world.** *Left:* Aggregated results across real-world evaluation conditions—random pose, background variation, distractor objects, and lighting changes—show that ATK outperforms other methods using different input modalities. *Right:* ATK demonstrates strong robustness under positional variation and various visual perturbations.

4.3 Robust Imitation Learning with Keypoint-based Policies

Policies learned via imitation learning with ATK are resilient to visual disturbances. We show that policies learned atop the ATKselected representations can perform tasks given significant visual variations. We introduce variations in object positions, backgrounds, distractor objects, and lighting conditions during evaluation. Despite not being trained in these conditions, the learned policies show significantly better transfer performance than other representations (RGB/depth/pointclouds). Moreover, we find that the particular choice of keypoints is crucial for robust transfer performance: FullSet has redundant keypoints that vary significantly, while RandomSelect and GPTSelect often miss important portions of the scene.

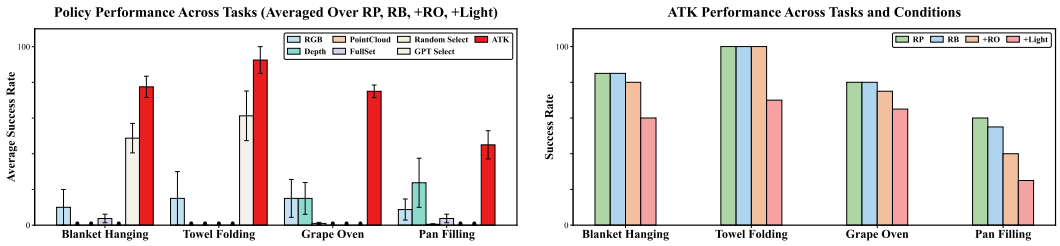


Figure 6: **Imitation policy success rates.** *Left:* Aggregated results across diverse evaluation conditions show that ATK outperforms other methods based on different input modalities and selection strategies. *Right:* ATK demonstrates strong robustness under positional variation and various visual perturbations.

4.4 Qualitative Visualization of Keypoints Learned by ATK

ATK chooses interpretable and task-relevant keypoints. In Fig 7, we show the keypoints ATK selects, focusing on task-relevant parts. Corresponding visualizations for imitation experiments are done in Fig 1.The chosen keypoints correspond to semantically meaningful elements of the scene. In multifunctional cases (e.g., kitchen, clock), they are specialized to the utility. We see that chosen keypoints transfer accurately from simulation to the real world and from training to testing. The selected keypoints are resilient to visual variations in the scene, including distractors, lighting changes, and background changes.

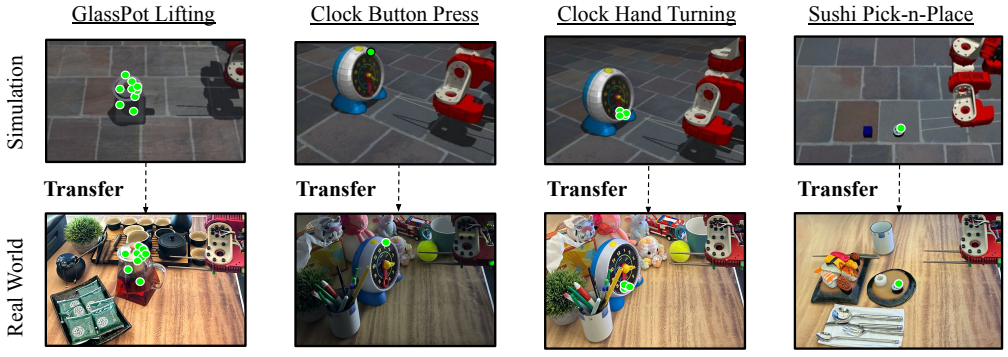


Figure 7: **Qualitative visualization of task-relevant keypoint selection and transfer.** Keypoints selected in simulation transfer to real-world scenes across various object positions, backgrounds, distractors, and lighting.

5 Related Work

Visual Representations. Prior research has explored various visual representation learning approaches for robotics [24, 25, 8, 26, 27] that use both self-supervised and supervised objectives [27, 24]. These representations often rely on large-scale pretraining or auxiliary information-theoretic objectives [25, 26]. Although such representations accelerate policy learning, they often entangle task-irrelevant features and become brittle under distribution shift. In contrast, this work focuses on exploiting privileged expert demonstrations to derive task-driven visual representations suitable for sim-to-real transfer and robust imitation learning.

Sim-to-real Transfer. Bridging the perceptual gap between simulation and the real world remains a significant challenge due to discrepancies in the observation space. Though simulations have become more photorealistic [28], the direct transfer of policies across domains continues to suffer from performance degradation. Prior work has proposed various methods to mitigate this gap, including domain randomization [29], latent representation learning [8, 30], unsupervised image translation [31, 32], depth-based policy [33, 34] and explicit pose estimation [35]. Though promising, these methods still face challenges in handling complex, precise tasks, or they rely on task-specific scaffolding (to estimate the pose of a certain object) or restrictive assumptions (e.g., the availability of accurate depth sensors). In this work, we focus on task-driven objectives for visual representations that are intrinsically robust to sim-to-real perturbations.

Keypoints as Representations for Learning-based Control. Keypoints have been utilized as robust state representations for robotic manipulation in several prior works [36, 37, 38, 39]. They have been applied in areas including deformable object manipulation [40, 41], few-shot imitation learning [37], model-based reinforcement learning [39], and learning from videos [36, 42]. However, these approaches often rely on heuristic or manual keypoint selection [43, 44]. Inspired by [45], our work differs by introducing a task-driven method for automatic keypoint selection. The procedure produces flexible but expressive representations that generalize across rigid, deformable, and transparent objects, and—crucially—are robust to challenging visual disturbances.

6 Limitations and Conclusion

We present ATK, a system for automatically selecting task-relevant keypoints, learning keypoint-based policies from these representations, and successfully transferring them to the real-world evaluation scenario. Though promising, the system faces challenges in tracking and optimization. The use of 2D keypoints makes the policy sensitive to camera perspective changes, and off-the-shelf tracking modules may lack robustness for robotic applications with uncommon visual data. Additionally, the method is sensitive to hyperparameters due to the non-smooth nature of the optimization problem, making tuning difficult. Currently, the method trains different models for each individual task. A promising future direction is to incorporate a task-conditional module that can isolate task-specific keypoints and support a unified multi-task selector for diverse manipulation skills. Nevertheless, our work demonstrates the robustness of keypoint-based policies and provides an effective approach for automatic keypoint selection. Developing more automated and robust techniques to address these challenges would further extend its applicability.

Acknowledgments

This work was (partially) funded by grants from the National Science Foundation NRI (#2132848), DARPA RACER (#HR0011-21-C-0171), the Office of Naval Research (#N00014-24-S-B001 and #2022-016-01 UW), and funding from the Toyota Research Institute through the University Research Program 3.0. We gratefully acknowledge gifts from Amazon, Collaborative Robotics, Cruise, and others. We would also like to thank Emma Romig, Grant Shogren for their help in setting up the hardware environments for this effort. We thank members of the Washington Embodied Intelligence and Robotics Development Lab and the Personal Robotics Lab for their thoughtful comments and feedback on versions of this draft.

References

[1] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control. *CoRR*, abs/2410.24164, 2024. doi:10.48550/ARXIV.2410.24164. URL https://doi.org/10.48550/arXiv.2410.24164.

[2] M. J. Kim, K. 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.

[3] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17:39:1–39:40, 2016. URL https://jmlr.org/papers/v17/15-522.html.

[4] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *Int. J. Robotics Res.*, 37 (4-5):421–436, 2018. doi:10.1177/0278364917710318. URL https://doi.org/10.1177/0278364917710318.

[5] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, editors, *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023. doi:10.15607/RSS.2023.XIX.026. URL https://doi.org/10.15607/RSS.2023.XIX.026.

[6] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P. Huang, S. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=a68SUt6zFt.

[7] S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. Gupta. The unsurprising effectiveness of pre-trained vision models for control. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 17359–17371. PMLR, 2022. URL https://proceedings.mlr.press/v162/parisi22a.html.

[8] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3M: A universal visual representation for robot manipulation. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Conference on Robot Learning, CoRL 2022, 14-18 December 2022, Auckland, New Zealand*, volume 205 of *Proceedings of Machine Learning Research*, pages 892–909. PMLR, 2022. URL https://proceedings.mlr.press/v205/nair23a.html.

[9] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 15979–15988. IEEE, 2022. doi:10.1109/CVPR52688.2022.01553. URL https://doi.org/10.1109/CVPR52688.2022.01553.

[10] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. URL http://arxiv.org/abs/1807.03748.

[11] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 2020. URL http://proceedings.mlr.press/v119/chen20j.html.

[12] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. Á. Pires, Z. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent - A new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/f3ada80d5c4ee70142b17b8192b2958e-Abstract.html.

[13] A. Majumdar, M. Sharma, D. Kalashnikov, S. Singh, P. Sermanet, and V. Sindhwani. Predictive red teaming: Breaking policies without breaking robots. *CoRR*, abs/2502.06575, 2025. doi:10.48550/ARXIV.2502.06575. URL https://doi.org/10.48550/arXiv.2502.06575.

[14] A. Hancock, A. Z. Ren, and A. Majumdar. Run-time observation interventions make vision-language-action models more visually robust. *CoRR*, abs/2410.01971, 2024. doi:10.48550/ARXIV.2410.01971. URL https://doi.org/10.48550/arXiv.2410.01971.

[15] C. Doersch, Y. Yang, M. Vecerík, D. Gokay, A. Gupta, Y. Aytar, J. Carreira, and A. Zisserman. TAPIR: tracking any point with per-frame initialization and temporal refinement. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 10027–10038. IEEE, 2023. doi:10.1109/ICCV51070.2023.00923. URL https://doi.org/10.1109/ICCV51070.2023.00923.

[16] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker: It is better to track together. *CoRR*, abs/2307.07635, 2023. doi:10.48550/ARXIV.2307.07635. URL https://doi.org/10.48550/arXiv.2307.07635.

[17] L. Tang, M. Jia, Q. Wang, C. P. Phoo, and B. Hariharan. Emergent correspondence from image diffusion. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/0503f5dce343a1d06d16ba103dd52db1-Abstract-Conference.html.

[18] M. Zare, P. M. Kebria, A. Khosravi, and S. Nahavandi. A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, 2024.

[19] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: rapid motor adaptation for legged robots. In D. A. Shell, M. Toussaint, and M. A. Hsieh, editors, *Robotics: Science and Systems XVII, Virtual Event, July 12-16, 2021*, 2021. doi:10.15607/RSS.2021.XVII.011. URL https://doi.org/10.15607/RSS.2021.XVII.011.

[20] T. Chen, J. Xu, and P. Agrawal. A system for general in-hand object re-orientation. In A. Faust, D. Hsu, and G. Neumann, editors, *Conference on Robot Learning, 8-11 November 2021, London, UK*, volume 164 of *Proceedings of Machine Learning Research*, pages 297–307. PMLR, 2021. URL https://proceedings.mlr.press/v164/chen22a.html.

[21] J. Yang, M. Gao, Z. Li, S. Gao, F. Wang, and F. Zheng. Track anything: Segment anything meets videos. *CoRR*, abs/2304.11968, 2023. doi:10.48550/ARXIV.2304.11968. URL https://doi.org/10.48550/arXiv.2304.11968.

[22] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

[23] Y. Ju, K. Hu, G. Zhang, G. Zhang, M. Jiang, and H. Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. *arXiv preprint arXiv:2401.07487*, 2024.

[24] M. Laskin, A. Srinivas, and P. Abbeel. CURL: contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5639–5650. PMLR, 2020. URL http://proceedings.mlr.press/v119/laskin20a.html.

[25] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. VIP: towards universal visual reward and representation via value-implicit pre-training. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=YJ7o2wetJ2.

[26] A. Zhang, R. T. McAllister, R. Calandra, Y. Gal, and S. Levine. Learning invariant representations for reinforcement learning without reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=-2FCwDKRREu.

[27] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1134–1141. IEEE, 2018. doi:10.1109/ICRA.2018.8462891. URL https://doi.org/10.1109/ICRA.2018.8462891.

[28] N. Morrical, J. Tremblay, Y. Lin, S. Tyree, S. Birchfield, V. Pascucci, and I. Wald. Nvisii: A scriptable tool for photorealistic image generation, 2021.

[29] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, pages 23–30. IEEE, 2017. doi:10.1109/IROS.2017.8202133. URL https://doi.org/10.1109/IROS.2017.8202133.

[30] A. Yu, A. Foote, R. Mooney, and R. Martín-Martín. Natural language can help bridge the sim2real gap. *CoRR*, abs/2405.10020, 2024. doi:10.48550/ARXIV.2405.10020. URL https://doi.org/10.48550/arXiv.2405.10020.

[31] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 12627–12637. Computer Vision Foundation / IEEE, 2019. doi:10.1109/CVPR.2019.01291. URL http://openaccess.thecvf.com/content_CVPR_2019/html/James_Sim-To-Real_via_Sim-To-Sim_Data-Efficient_Robotic_Grasping_via_Randomized-To-Canonical_Adaptation_Networks_CVPR_2019_paper.html.

[32] D. Ho, K. Rao, Z. Xu, E. Jang, M. Khansari, and Y. Bai. Retinagan: An object-aware approach to sim-to-real transfer. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*, pages 10920–10926. IEEE, 2021. doi:10.1109/ICRA48506.2021.9561157. URL https://doi.org/10.1109/ICRA48506.2021.9561157.

[33] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *CoRR*, abs/2403.03949, 2024. doi:10.48550/ARXIV.2403.03949. URL https://doi.org/10.48550/arXiv.2403.03949.

[34] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023. doi:10.1126/scirobotics.adc9244. URL https://www.science.org/doi/abs/10.1126/scirobotics.adc9244.

[35] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. V. Wyk, A. Zhurkevich, B. Sundaralingam, and Y. S. Narang. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*, pages 5977–5984. IEEE, 2023. doi:10.1109/ICRA48891.2023.10160216. URL https://doi.org/10.1109/ICRA48891.2023.10160216.

[36] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. *CoRR*, abs/2401.00025, 2024. doi:10.48550/ARXIV.2401.00025. URL https://doi.org/10.48550/arXiv.2401.00025.

[37] M. Vecerík, C. Doersch, Y. Yang, T. Davchev, Y. Aytar, G. Zhou, R. Hadsell, L. Agapito, and J. Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. In *IEEE International Conference on Robotics and Automation, ICRA 2024, Yokohama, Japan, May 13-17, 2024*, pages 5397–5403. IEEE, 2024. doi:10.1109/ICRA57147.2024.10611409. URL https://doi.org/10.1109/ICRA57147.2024.10611409.

[38] L. Manuelli, W. Gao, P. R. Florence, and R. Tedrake. KPAM: keypoint affordances for category-level robotic manipulation. In T. Asfour, E. Yoshida, J. Park, H. Christensen, and O. Khatib, editors, *Robotics Research - The 19th International Symposium ISRR 2019, Hanoi, Vietnam, October 6-10, 2019*, volume 20 of *Springer Proceedings in Advanced Robotics*, pages 132–157. Springer, 2019. doi:10.1007/978-3-030-95459-8_9. URL https://doi.org/10.1007/978-3-030-95459-8_9.

[39] L. Manuelli, Y. Li, P. R. Florence, and R. Tedrake. Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning. In J. Kober, F. Ramos, and C. J. Tomlin, editors, *4th Conference on Robot Learning, CoRL 2020, 16-18 November 2020, Virtual Event / Cambridge, MA, USA*, volume 155 of *Proceedings of Machine Learning Research*, pages 693–710. PMLR, 2020. URL https://proceedings.mlr.press/v155/manuelli21a.html.

[40] X. Ma, D. Hsu, and W. S. Lee. Learning latent graph dynamics for visual manipulation of deformable objects. In *2022 International Conference on Robotics and Automation, ICRA 2022, Philadelphia, PA, USA, May 23-27, 2022*, pages 8266–8273. IEEE, 2022. doi:10.1109/ICRA46639.2022.9811597. URL https://doi.org/10.1109/ICRA46639.2022.9811597.

[41] P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, J. Ichnowski, E. R. Novoseller, M. Hwang, M. Laskey, J. Gonzalez, and K. Goldberg. Untangling dense non-planar knots by learning manipulation features and recovery policies. In D. A. Shell, M. Toussaint, and M. A. Hsieh, editors, *Robotics: Science and Systems XVII, Virtual Event, July 12-16, 2021*, 2021. doi:10.15607/RSS.2021.XVII.013. URL https://doi.org/10.15607/RSS.2021.XVII.013.

[42] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation, 2024.

[43] F. Liu, K. Fang, P. Abbeel, and S. Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[44] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning, 2023.

[45] M. Sharma and O. Kroemer. Generalizing object-centric task-axes controllers using keypoints. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7548–7554. IEEE, 2021.

[46] C. E. Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[47] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[48] J. Lu, Z. Liang, T. Xie, F. Ritcher, S. Lin, S. Liu, and M. C. Yip. Ctrnet-x: Camera-to-robot pose estimation in real-world conditions using a single camera. *arXiv preprint arXiv:2409.10441*, 2024.

[49] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi:10.1109/IROS.2012.6386109.

[50] Y. Zhang, L. Ke, A. Deshpande, A. Gupta, and S. Srinivasa. Cherry-picking with reinforcement learning: Robust dynamic grasping in unstable conditions. *arXiv preprint arXiv:2303.05508*, 2023.

[51] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation, 2022. URL https://arxiv.org/abs/2203.12601.

[52] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations, 2024. URL https://arxiv.org/abs/2403.03954.

[53] M. Vecerik, C. Doersch, Y. Yang, T. Davchev, Y. Aytar, G. Zhou, R. Hadsell, L. Agapito, and J. Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5397–5403, 2024. doi:10.1109/ICRA57147.2024.10611409.

A ATK implementation details

A.1 Mask network

To ensure the selection of the minimal set of K keypoints, we enforce a sparsity penalty on the mask network. We denote the masking distribution over binary masks $m \in \{0,1\}^N$ for candidate keypoints K as $\mathbb{M}_\phi(m \mid K)$, and the action distribution conditioned on the masked keypoints as $\pi_\theta(a \mid \mathbb{M}_\phi)$. Because sampling $m \sim \mathbb{M}_\phi(m \mid K)$ is discrete and non-differentiable, we employ the Gumbel–softmax relaxation [22] to enable gradient-based optimization. Concretely, a N-dimensional learnable parameters are used as logits for N independent Bernoulli probabilities, with each representing the likelihood of retaining the corresponding keypoint from the input candidate set. We then add Gumbel noise to each logit, divide by a temperature τ, and apply softmax to form a continuous "soft" mask y_M^{Soft}. During the forward pass, we take $\arg\max_i \ y_i^{\text{Soft}}$, to get a hard one-hot selection, but in the backward pass we use the smooth "soft" mask for gradient calculation. The policy π_θ can be instantiated by any policy class model—e.g., a multi-layer perceptron, a Gaussian policy [46] or a score-based diffusion model [47]. The final objective jointly optimizes the policy $\pi_\theta(a \mid \mathbb{M}_\phi)$ and the masking distribution $\mathbb{M}_\phi(m \mid K)$, yielding a minimal, task-relevant keypoint representation.

A.2 Viewpoint robustness

"Viewpoint robustness" is another critical metric for gauging how well a policy holds up when the camera's perspective shifts. It measures the policy's performance under changed camera viewpoints. In our method, we assume access to both the original and shifted camera intrinsic and extrinsic matrices—a practical assumption given modern computer vision advances(e.g., extrinsic estimation via CtRNet [48]). In our experiments, we utilized a fixed ArUco marker's coordinate to get the cameras' extrinsic. We use $h_{\mathcal{C}}$ to find correspondence 2D keypoints in the new camera view and project them back into the original camera frame where the policy was trained. This reprojection compensates for viewpoint shifts, allowing the policy to operate as if the view had not changed. We show that in our video, after changing the camera to three different angles like Figure 8, the reprojected keypoints still enable the policy to succeed.

Figure 8: Different camera viewpoint

A.3 Category-Level Generalization

To evaluate whether this method can generalize to different objects within the same category, we tested it on two tasks as shown in Figure 9. We assume that simulated objects may differ in color, size, or shape from their real-world counterparts; as long as they share the same category, a web-scale pretrained correspondence model can still match them reliably across various visual domain shifts. To measure the robustness of the correspondence matching algorithm, we consider two metrics: (1) Confidence Score, the mean cosine similarity of matched feature vectors, and (2) Mean Distance Error, the average pixel offset between manual annotations and the correspondences predicted by h_C. As shown in the Table 1 , the correspondence matching method achieves a high confidence score (0.76–0.82) and a low distance error (6 pixels, < 5% of the object size), demonstrating accurate sim-to-real keypoint transfer.

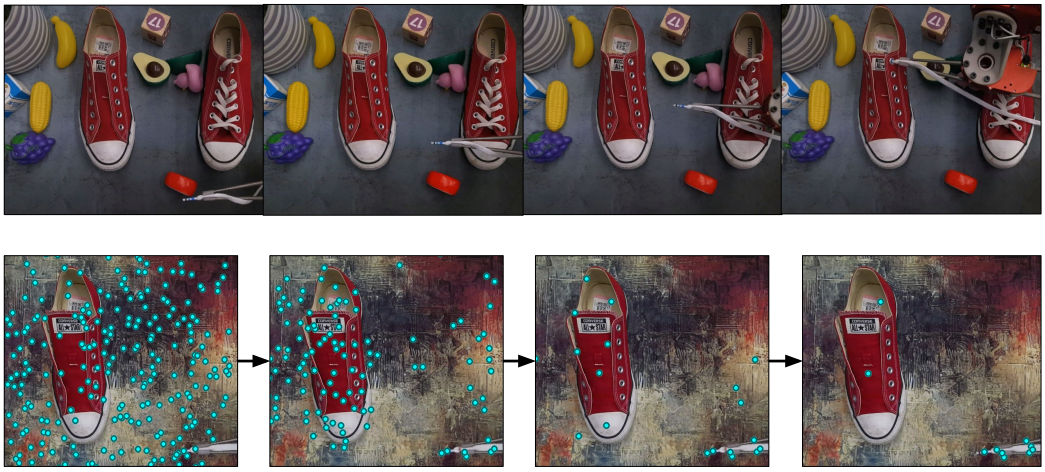


Figure 10: **The shoe lacing task.** The first row shows a successful rollout of the policy, performing shoelace insertion with varying backgrounds and distractors. The second row illustrates the keypoint distillation process.

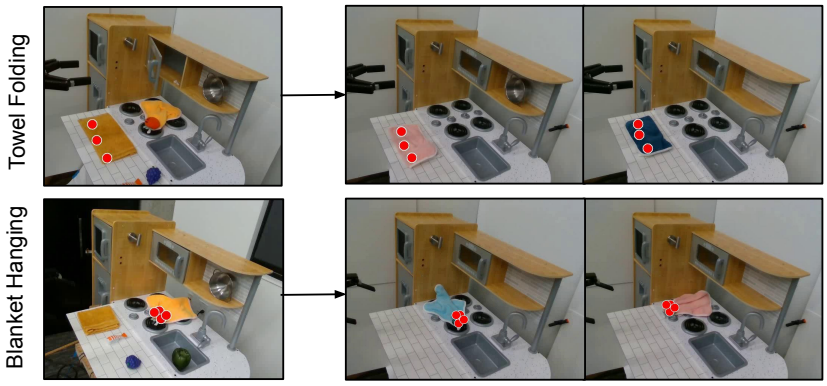


Figure 9: Category Level Generalization

Method	Sushi	Glass	Clock Button	Clock Turning
Confidence Score	0.78	0.76	0.80	0.82
Mean Distance Error (pixels)	3.24	5.21	2.79	6.75

Table 1: Correspondence Matching Performance Metrics

A.4 High-precision task

We also evaluate ATK on a **high-precision manipulation task, i.e., shoe lacing**. For this fine-grained insertion task, the robot needs to insert a shoelace into a shoe eyelet. The task requires high precision since the shoe eyelet has a diameter of 5 mm and the lace's radius is approximately 3.2 mm, leaving only a 1.7 mm tolerance—significantly tighter than the tolerances typically encountered in standard picking or grasping tasks. We tested the robustness of our method under challenging conditions, including varying background textures, random distractors, and changes in lighting. Despite these perturbations, our approach demonstrates high performance.

B Tasks evaluation procedure

B.1 Evaluation metrics

Random pose (RP): As shown in Figure 11, we show the distribution of the object's pose during task evaluation.

Algorithm 1 ATK: Automatic Task-Driven Keypoint Selection for Robust Policy Learning

Input: Expert dataset $\mathcal{D} = \{(o_i, a_i)\}_{i=1}^{M}$, correspondence function $h_{\mathcal{C}}$
Output: Mask model $\mathbb{M}_\phi$, keypoint policy π_θ

// Initialize keypoints on canonical frame
Choose canonical image I_{can} capturing full task context
Sample C candidate keypoints $K_{\text{can}} = \{k_j\}_{j=1}^{C}$ on I_{can}
// Propagate & label entire dataset
for $i \leftarrow 0$ **to** $M - 1$ **do** ▷ for all trajectories
  for $t \leftarrow 0$ **to** $T_i - 1$ **do**
    if $t = 0$ **then**
      $K_{i,0} \leftarrow h_{\mathcal{C}}\big(K_{\text{can}},\, o_{i,0},\, I_{\text{can}}\big)$ ▷ initial alignment to canonical frame
    else
      $K_{i,t} \leftarrow h_{\mathcal{C}}\big(K_{i,t-1},\, o_{i,t}\big)$ ▷ track from previous frame
    end if
  end for
end for
Construct $\mathcal{D}_k = \{(o_{i,t}, K_{i,t}, a_i)\}$
//Training
while not converged **do**
  Sample mini-batch $\mathcal{B} \subset \mathcal{D}_k$
  Draw mask $m \sim \mathbb{M}_\phi(K)$ ▷ Gumbel–Softmax
  $\widetilde{K} \leftarrow m \odot K$ ▷ retain selected points
  $\mathcal{L} \leftarrow -\log \pi_\theta(a \mid \widetilde{K}) + \alpha \|m\|_1$
  Update ϕ, θ with Adam to minimise $\mathcal{L}$
end while
return $(\mathbb{M}_\phi, \pi_\theta)$

function EVALUATION
  // Transfer keypoints to test scene
  $K_0 \leftarrow h_{\mathcal{C}}\big(K_{\text{can}},\, I_{\text{can}},\, O_{init}\big)$ ▷ highest-score matches
  Track $K_t = h_{\mathcal{C}}(K_{t-1}, o_t)$ for $t \geq 1$
  return $\pi_\theta(K_t)$ at each step
end function

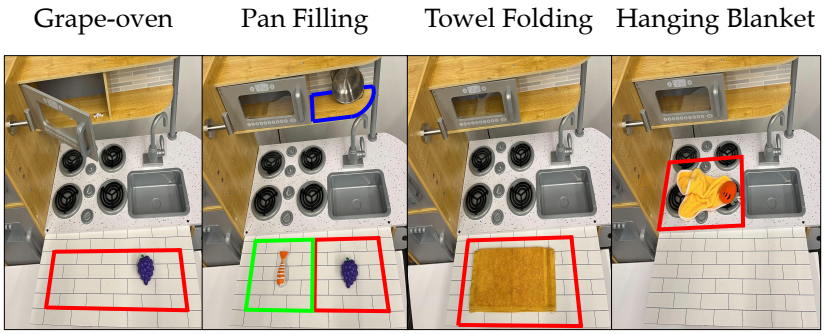


Figure 11: Highlighted regions for different objects during evaluation.

Random background (RB): As shown in Figure 12, we show different backgrounds used for evaluation in each task

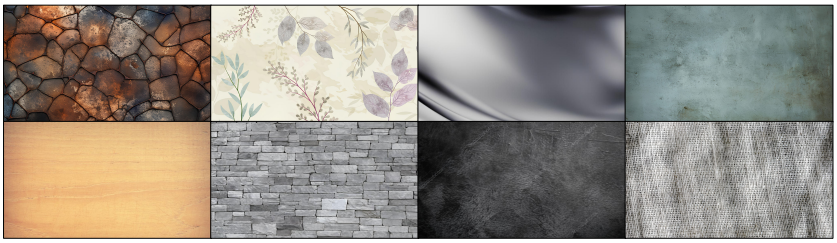


Figure 12: Different backgrounds used for evaluation

Random distractor objects (RO): As shown in Figure 13, we show distractor objects used for evaluation in each task.

Figure 13: Distractor Objects used in evaluation

Light: As shown in Figure 14, we show the different colored lights used for evaluation in each task

Figure 14: Different Light Conditions used for evaluation

C Infrastructural setup

Sim-to-Real Setup: We create MuJoCo [49] simulation environment using an iPhone app, Scaniverse, to scan and import the meshes of real-world objects and add joints for articulated objects. We conduct real-world transfer experiments using a 6-DOF Hebi robot arm equipped with chopsticks, following [50]. For RGB and depth streaming, we employ Azure Kinect RGB-D cameras.

Imitation Setup: Imitation learning is purely tested in the real world. We test these methods on the UR5e robot equipped with a Robotiq 2F-145 gripper, running a joint PD controller. This robot is tasked with manipulating various objects in a miniature kitchen. As previously mentioned, for RGB and depth streaming, we employ Azure Kinect RGB-D cameras. We used 80, 50, 50, 30 number of demonstrations to train policies for pan filling, grape-oven, towel folding, hanging blanket respectively.

D Baseline implementation details

ResNet18 Encoder: We use a modified ResNet18 architecture as the encoder for depth inputs. The depth data is first resized to 224×224 and duplicated across three channels to match the expected input format of the ResNet18 backbone. Then, we take the ResNet's 512-dimensional feature embedding as a feature map and concatenate it with a 7-dimensional robot state (six joint values and a binary gripper state). The resulting vector is then passed to the diffusion policy [47] for action prediction.

R3M Encoder: To handle RGB inputs, we first resize the RGB images from 640×480 resolution to 224×224. Then, we use R3M[51] to extract a 512-dimensional feature embedding given image inputs and concatenate this embedding with the 7-dimensional joint and gripper state vector to form the input to the diffusion policy [47].

DP3 Encoder: For encoding 3D point cloud data, we follow the encoder from DP3[52], a CNN-based architecture that utilizes layer normalization. We first project the depth map into 3D and do a spatial crop to get dense point clouds. We then uniformly down-sample 4096 points as the input for the 3D encoder.

The output 3D feature embedding with size 1024 is concatenated with the same 7-dimensional robot state vector (six joint values and a binary gripper flag) before being used as input to the diffusion policy [47].

Baseline Performance Notes: All baselines (RGB, depth, point-cloud) first used Appendix E.1 encoders with an MLP policy, but performance was weak, so we switched to the official diffusion-policy. With the same 50–60 demos as ATK, each baseline achieved >0% success on at least one task. However, visual domain shifts and sensor noise especially for depth and pointcloud caused failures, whereas ATK remained robust.

Hyperparameter Selection: We provide recommended ranges for the core hyperparameters used across our experiments in Table 7. The default learning rate of 1×10^{-5} is effective across most tasks. Increasing the batch size often improves performance by providing richer contextual information, which is especially beneficial for stable keypoint masking. The sparsity weight λ controls the keypoint numbers; for tasks requiring high precision or fine-grained manipulation, a lower sparsity weight is recommended to preserve more expressive and informative keypoints.

E Detailed evaluation results

E.1 Sim-to-real performance

	Sushi				Glass			
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light
RGB	**0.453±0.262**	0.076±0.041	**0.027±0.020**	**0.010±0.014**	**0.253±0.154**	0.109±0.098	**0.020±0.021**	0.000±0.000
Depth	0.255±0.199	0.255±0.199	0.020±0.021	**0.010±0.014**	0.110±0.001	**0.110±0.001**	0.000±0.000	0.000±0.000
Pointcloud	0.277±0.088	**0.277±0.088**	0.020±0.021	0.000±0.000	0.033±0.047	0.033±0.047	0.000±0.000	0.000±0.000
ATK	**0.893±0.073**	**0.893±0.073**	**0.893±0.073**	**0.893±0.073**	**0.933±0.034**	**0.933±0.034**	**0.933±0.034**	**0.933±0.034**
	Clock button				Clock turning			
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light
RGB	**0.456±0.293**	0.046±0.017	**0.013±0.019**	0.000±0.000	**0.367±0.205**	0.093±0.020	**0.013±0.012**	0.000±0.000
Depth	0.290±0.150	**0.290±0.150**	0.000±0.000	0.000±0.000	0.256±0.264	**0.256±0.264**	0.000±0.000	**0.020±0.021**
Pointcloud	0.107±0.056	0.107±0.056	0.010±0.014	0.000±0.000	0.077±0.056	0.077±0.056	0.010±0.014	0.010±0.014
ATK	**0.970±0.024**	**0.970±0.024**	**0.970±0.024**	**0.970±0.024**	**0.903±0.028**	**0.903±0.028**	**0.903±0.028**	**0.903±0.028**

Table 2: **Simulator** Policy Success Rates using *different input modalities* over 3 random seeds. Keypoint-based policies are easier to distill in simulator than other baselines with alternative sensor modalities.

	Sushi Pick-n-Place				GlassPot Lift				Clock Button Press				Clock Hand Turning				Total
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	
RGB	**0.30**	0.00	0.00	0.00	**0.10**	0.00	0.00	0.00	**0.25**	0.00	0.00	0.00	**0.05**	0.00	0.00	0.00	0.04
Depth	0.25	**0.20**	0.00	0.00	0.05	0.00	0.00	0.00	0.10	**0.10**	0.00	0.00	0.00	0.00	0.00	0.00	0.04
Pointcloud	0.10	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.02
ATK	**0.85**	**0.80**	**0.55**	**0.45**	**0.75**	**0.65**	**0.60**	**0.60**	**0.90**	**0.90**	**0.80**	**0.75**	**0.50**	**0.50**	**0.40**	**0.35**	**0.64**

Table 3: **Real-world** Policy Success Rates. Varying conditions including RP (random pose), RB (background), RO (distractor object), Light. ATK consistently outperforms baseline methods using alternative modalities in sim-to-real transfer.

	Sushi				Glass			
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light
FullSet	0.122±0.057	0.053±0.036	0.010±0.008	0.013±0.012	**0.311±0.150**	**0.069±0.056**	0.013±0.012	**0.013±0.012**
RandomSelect	**0.337±0.315**	**0.246±0.360**	**0.233±0.370**	**0.226±0.375**	0.120±0.082	0.031±0.044	**0.116±0.151**	0.006±0.009
GPTSelect	0.032±0.009	0.020±0.008	0.013±0.005	0.006± 0.004	0.133±0.188	0.020±0.028	0.010±0.014	0.010±0.014
ATK	**0.893±0.073**	**0.893±0.073**	**0.893±0.073**	**0.893±0.073**	**0.933±0.034**	**0.933±0.034**	**0.933±0.034**	**0.933±0.034**
	Clock button				Clock turning			
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light
FullSet	0.474±0.317	0.126±0.090	0.026±0.030	0.020±0.016	0.253±0.183	0.083±0.880	0.010±0.014	0.010±0.014
RandomSelect	0.107±0.030	0.080±0.045	0.036±0.032	0.026±0.020	**0.253±0.166**	0.076±0.088	0.000±0.000	0.000±0.000
GPTSelect	**0.913±0.041**	**0.913±0.041**	**0.913±0.041**	**0.913±0.041**	0.065±0.053	**0.146±0.179**	**0.077±0.088**	**0.020±0.028**
ATK	**0.970±0.024**	**0.970±0.024**	**0.970±0.024**	**0.970±0.024**	**0.903±0.028**	**0.903±0.028**	**0.903±0.028**	**0.903±0.028**

Table 4: **Simulator** Policy Success rate using **different keypoint selection methods** over 3 random seeds. ATK consistently outperforms alternative keypoint selection methods using random sampling or ChatGPT selection.

E.2 Imitation learning performance

	Towel Hanging				Towel Folding				Grape Oven				Pan Filling			
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light
RGB	**0.40**	0.00	0.00	0.00	**0.60**	0.00	0.00	0.00	**0.45**	0.15	0.00	0.00	**0.25**	0.10	0.00	0.00
Depth	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.35	**0.25**	0.00	0.00	0.50	0.45	0.00	0.00
Pointcloud	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.00	0.00	0.01	0.01	0.00	0.00
ATK	**0.85**	**0.85**	**0.80**	**0.60**	**1.00**	**1.00**	**1.00**	**0.70**	**0.80**	**0.80**	**0.75**	**0.65**	**0.60**	**0.55**	**0.40**	**0.25**

Table 5: **Real World Imitation Learning** Success Rates. Varying conditions including RP (random pose), RB (background), RO (distractor object), Light. ATK consistently outperforms baseline methods using alternative modalities.

	Towel Hanging				Towel Folding				Grape Oven				Pan Filling			
	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light	RP	RB	+RO	+ Light
FullSet	0.10	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.05	0.00	0.00
RandomSelect	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GPTSelect	0.60	0.60	0.50	0.25	1.00	0.60	0.50	0.35	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ATK	**0.85**	**0.85**	**0.80**	**0.60**	**1.00**	**1.00**	**1.00**	**0.70**	**0.80**	**0.80**	**0.75**	**0.65**	**0.60**	**0.55**	**0.40**	**0.25**

Table 6: **Real World Imitation Learning** Success rate using **different keypoint selection methods**. ATK consistently outperforms alternative keypoint selection methods using random sampling or ChatGPT selection.

Table 7: Core hyperparameters and their recommended ranges.

Parameter	**Default**	**Typical Range**
Learning rate	1×10^{-5}	1×10^{-5}–1×10^{-4}
Batch size	4096	2048–4096
Sparsity Weight λ	0.002	0.001–0.005

E.3 GPT selection details

Prompt for keypoint selection using GPT for Grape-oven task

Consider the image:

Task Description:
You are training a robotic arm policy to perform a task that involves:

1. Identifying and picking up a grape object in an image.
2. Placing the object inside an open gated oven.
3. Closing the oven door.

Keypoint Requirements:
The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 4 critical keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image

GPT Output

Here are the 2D pixel coordinates for the selected keypoints:

On the Grape Object

- Left side of the object: **(210, 375)**
- Right/top side of the object: **(225, 368)**

On the Oven (Above the Stove)

- Inside the oven tray (for placing the object): **(305, 190)**
- Oven handle (for closing the door): **(270, 160)**

Prompt for keypoint selection using GPT for Pan-filling task

Consider the image:

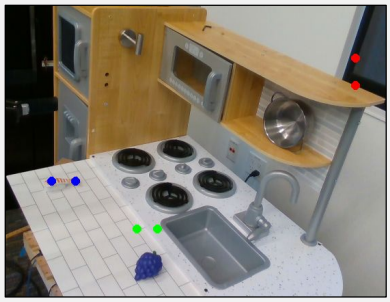
Task Description:
You are training a robotic arm policy to perform a task that involves:

1. Identifying and picking up a pan placed on a shelf above the sink in the above image.
2. Placing the pan on the right bottom gas burner.
3. Picking up sushi, placing it inside the pan, and finally picking up a grape and placing it inside the pan.

Keypoint Requirements:
The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 6 critical 2D keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above-given image

GPT Output



Here are the 2D pixel coordinates for the selected keypoints: Sushi Keypoints:

- sushi_1: (105, 230)
- sushi_2: (125, 240)

Grape Keypoints:

- grape_1: (215, 350)
- grape_2: (225, 370)

Pan Keypoints:

- pan_1: (495, 160)
- pan_2: (525, 180)

Prompt for keypoint selection using GPT for Towel Folding task

Consider the image:

Task Description: You are training a robotic arm policy to perform a task that involves identifying and folding a towel lying flat on the kitchen counter:
Keypoint Requirements:
The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 6 critical 2D keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image

GPT Output

Here are the 2D pixel coordinates for the selected keypoints:

- Left Edge: (85, 365)
- Right Edge: (225, 365)
- Top Edge: (225, 320)

Prompt for keypoint selection using GPT for Hanging Blanket task

Consider the image:

Task Description: You are training a robotic arm policy to perform a task that involves identifying and picking up a blanket lying flat on the gas stove and hang it on a hook attached on a side wall:
Keypoint Requirements:
The robot's control policy uses relevant visual keypoints as input to solve the above task. Your objective is to:

- Identify and output 4 critical 2D keypoints in the provided image that will help the policy learn this task.
- Draw your output keypoints on the above given image

GPT Output

Here are the 2D pixel coordinates for the selected keypoints:

- Grasp Ball (Pickup): (330, 300)
- Towel Top Edge: (280, 280)
- Towel Center: (300, 320)
- Towel Side Edge: (260, 340)

E.4 Additional KeyPoint Selection Baselines

To make fairer comparison, we provide GPT with the same set of candidate keypoints as used for the initial set of keypoints on the canonical image from ATK. As illustrated in the figure 15 , the GPT-Select approach occasionally suffers from spatial mismatches (incorrectly locating the relevant point in the image) or language mismatches (incorrectly interpreting the textual prompt), which leads to erroneous keypoint selections. We also evaluated Robotap[53] keypoint's selector on our tasks. As

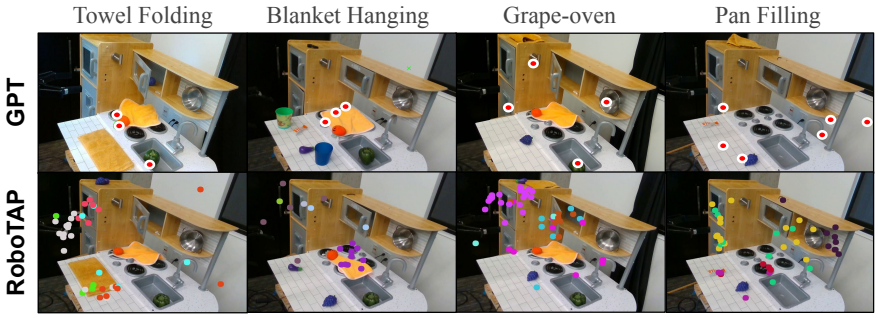


Figure 15: First row: GPT results when prompted with the initial set of candidate keypoints. Second row: RoboTAP selection results. Different colors denote distinct clusters of keypoints remaining after selection.

shown in the above figure, its design for ego-centric images prevents it from filtering out irrelevant points effectively in our setup.