Vision-based Manipulation from Single Human Video with Open-World Object Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract: We present an object-centric approach to empower robots to learn vision-based manipulation skills from human videos. We investigate the problem of imitating robot manipulation from a single human video in the *open-world* setting, where a robot must learn to manipulate novel objects from one video demonstration. We introduce ORION, an algorithm that tackles the problem by extracting an object-centric manipulation plan from a single RGB-D video and deriving a policy that conditions on the extracted plan. ORION enables the robot to learn from videos captured by daily mobile devices such as an iPad and generalize the policies to deployment environments with varying visual backgrounds, camera angles, spatial layouts, and novel object instances. We systematically evaluate ORION on both short-horizon and long-horizon tasks, demonstrating the efficacy of ORION in learning from a single human video in the open world.

Keywords: Robot Manipulation, Imitation From Human Videos

1 Introduction

A critical step toward building robot autonomy is developing sensorimotor skills for perceiving and interacting with unstructured environments. Conventional methods for acquiring skills necessitate manual engineering and/or costly data collection [1–5]. A promising alternative is teaching robots through human videos of manipulation behaviors situated in everyday scenarios. These methods have great potential to tap into the readily available source of Internet videos that encompass a wide distribution of human activities, paving the ground for scaling up skill learning.

Prior work on learning from human videos has focused on pre-training representations and value functions [6–10]. However, they do not explicitly capture object states and their interactions in 3D space where robot motions are defined. Consequently, they require separate teleoperation data for each set of objects in each location and even for each possible change in visual background, e.g., the scene background or lighting conditions [11]. In contrast, our goal is for a robot to imitate a task robustly in the “open world”, i.e., under varying visual and spatial conditions from a single human video, without prior knowledge of the object models or the behaviors shown. Since we consider actionless videos that are equivalent to state-only demonstrations in the problem of “Imitation from Observation”[12], we refer to our problem setting as *open-world imitation from observation*.

Developing a method in this setting is only possible due to the recent advances in vision foundation models [13, 14]. These models, pre-trained on Internet-scale visual data, excel at understanding open-vocabulary visual concepts and enable robots to recognize and localize objects in videos without known object categories or access to physical states. This work marks the first step toward achieving our vision of open-world imitation from observation, where a robot imitates how to interact with objects given *a single video* while deployed in environments with different visual backgrounds and unseen spatial configurations. In this work, we consider using RGB-D video demonstrations where a person manipulates a small set of task-relevant objects with their single *hand*,

Submitted to the 8th Conference on Robot Learning (CoRL 2024). Do not distribute.

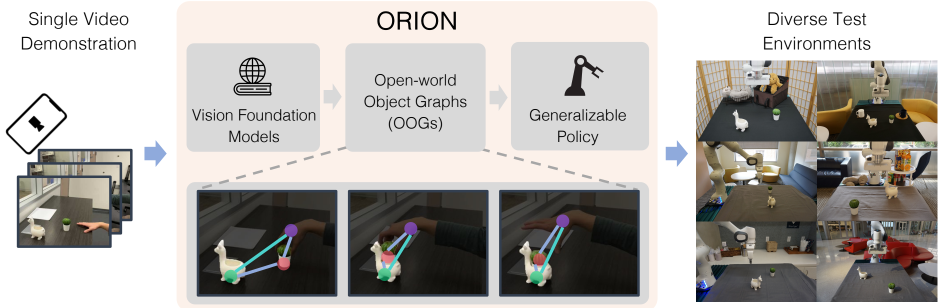

Figure 1: **Overview.** ORION tackles the problem of imitating manipulation from single human video demonstrations. ORION first extracts a sequence of Open-World Object Graphs (OOGs), where each OOG models a keyframe state with task-relevant objects and hand information. Then ORION leverages the OOG sequence to construct a manipulation policy that generalizes across varied initial conditions, specifically in four aspects: visual background, camera shifts, spatial layouts, and novel instances from the same object categories.

recorded with a stationary camera. These videos are actionless or state-only, as they do not come with any ground-truth action labels for the robot.

We introduce our method ORION, short for **O**pen-wo**R**ld video **I**mitati**ON**. Figure 1 visualizes a high-level overview of ORION. The core innovation lies in creating an object-centric spatiotemporal abstraction that effectively bridges the observational gap between human demonstration and robot execution. The design of ORION stems from our insight that manipulation tasks center around object interaction, and task completion depends on whether specific intermediate states, so-called *subgoals*, are reached. To capture the object-centric information in the video, we design a graph-based, object-centric representation, called Open-world Object Graphs (OOGs), to model the states of task-relevant objects and their relationships. An OOG has a two-level hierarchy. The high level consists of the object nodes and a hand node, where object nodes identify and localize the relevant objects by leveraging outputs from vision foundation models, while the hand node encodes the interaction information between the hand and objects, such as where to grasp. The low level consists of point nodes, which correspond to object keypoints, and the node features detail the motions of object keypoints in the 3D space.

ORION extracts a manipulation plan from the video as a sequence of OOGs and uses the plan to construct a generalizable policy. In experiments, ORION constructs a policy robust to conditions vastly different from the one in the video. Using only an iPhone or an iPad to record a human performing tasks in everyday environments (e.g., an office or a kitchen), ORION policies are deployed in workspaces with drastically different visual backgrounds, camera angles, and spatial arrangements, and even generalize to manipulating unseen object instances of the same categories.

In summary, our contribution is three-fold: 1) We pose the problem of learning vision-based robot manipulation from a single human video in the open-world setting; 2) We introduce Open-world Object Graphs (OOGs), a graph-based, object-centric representation for modeling the states and relations of task-relevant objects; and 3) We present ORION, an algorithm that uses a single video to construct a manipulation policy, which generalizes to conditions that differ in four key ways: visual backgrounds, camera perspectives, spatial configurations, and new object instances.

2 Problem Formulation

In this paper, we consider a vision-based, tabletop manipulation task, formulated as a finite-horizon Markov Decision Process (MDP) described by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, H, R, \mu \rangle$, where $\mathcal{S}$ is the state space of raw sensory data including RGB-D images and robot proprioception, $\mathcal{A}$ is the action space of low-level robot commands, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the transition dynamics, H is the maximal task horizon, R is the sparse reward function, and μ is the initial state distributions of a task. In this work, we consider the case where task reward functions are defined based on the contact relations between a small set of *task-relevant objects*. For example, a mug is placed on top of a coaster, or a spoon is put inside a bowl. A reward function returns 1 if all object relations of a task are satisfied and 0 otherwise. The primary objective of solving a manipulation task is to find a visuomotor policy π that maximizes the expected task success rate from a wide range of initial configurations, characterized by μ, where the states vary across the following four dimensions: 1) changing visual backgrounds, 2) different camera angles, 3) different object instances from the same categories, and 4) varied spatial layouts of the task-relevant objects.

We assume a robot does not have direct access to the ground-truth task reward or the physical states of task-relevant objects. We consider a setting where a single *actionless* video [15, 16] V is provided as a *state-only* demonstration. We assume V to be a video stream of a person manipulating the task-relevant objects with their single *hand*, captured as a sequence of RGB-D images using a stationary camera. V is an arbitrarily long video that involves a manipulation sequence where the contact relations among task-relevant objects and the hand change (e.g., an object is grasped or an object is placed on top of another). The assumption about V refers to tasks that involve diverse manipulation behaviors such as pick-and-place, assembly, object insertion. To avoid the inherent ambiguities of videos due to the distraction of irrelevant objects or ambiguities of what a user wants to specify (whether the color of a task-relevant mug matters to the task or not), each V is accompanied by a complete list of English descriptions of the task-relevant objects with their complete feature descriptions such as their color that a user wants, uniquely defining the object instances in V. Such a list is represented as a comma-separated list; an example is "[‘`small red block`’, ‘`boat body`’]" for the task shown in Figure 2. In this scenario, however, the robot is not pre-programmed to have access to ground-truth categories and locations of the task-relevant objects in V. We refer to this challenging setting as "open-world" [17], as the robot must imitate from V while not pre-programmed or trained to interact with the objects in V. To allow a robot to operate in this "open-world" setting, we assume access to common sense knowledge through large models pre-trained on internet-scale data, i.e., foundation models. For evaluation, we adopt the following procedure. Given a single video V that accomplishes a task instance drawn from μ, the performance of an approach is quantified by the average rewards received when evaluating new task instances drawn from the same μ.

3 Method

We introduce ORION (**O**pen-wo**R**ld video **I**mitati**ON**), an algorithm that allows a robot to mimic how to perform a manipulation task given a single human video, V. To effectively construct a policy π from V, ORION employs a learning objective based on an object-centric prior. The goal is to create a policy π that directs the robot to move objects along 3D trajectories that mimic the directional and curvature patterns observed in V, relative to the objects' initial and final positions. This objective is based on the observation that objects are likely to achieve target configurations by moving along trajectories similar to those in V. Key to ORION is generating a manipulation plan from V, which serves as the spatiotemporal abstraction of the video that guides the robot to perform a task. A plan is a sequence of object-centric keyframes that each specifies an initial or a subgoal state captured in V. We first introduce our formulation of the object-centric representation of a state, Open-world Object Graph (OOG), used in ORION, and then describe the algorithm that constructs a robot policy given a human video.

3.1 Open-world Object Graph

At the core of our approach is a graph-based, object-centric representation, Open-world Object Graphs (OOGs). OOGs use open-world vision models that model the visual scenes with task-relevant objects and the hand such that they naturally exclude the distracting factors in visual data and localize the task-relevant objects regardless of their spatial locations (see Section 3.2).

We denote an OOG as $\mathcal{G}$. At the high level, each object node corresponds to a task-relevant object from the result of open-world vision models. Every object node comes with node features, consisting of colored 3D point clouds derived from RGB-D observations. This node feature indicates both what

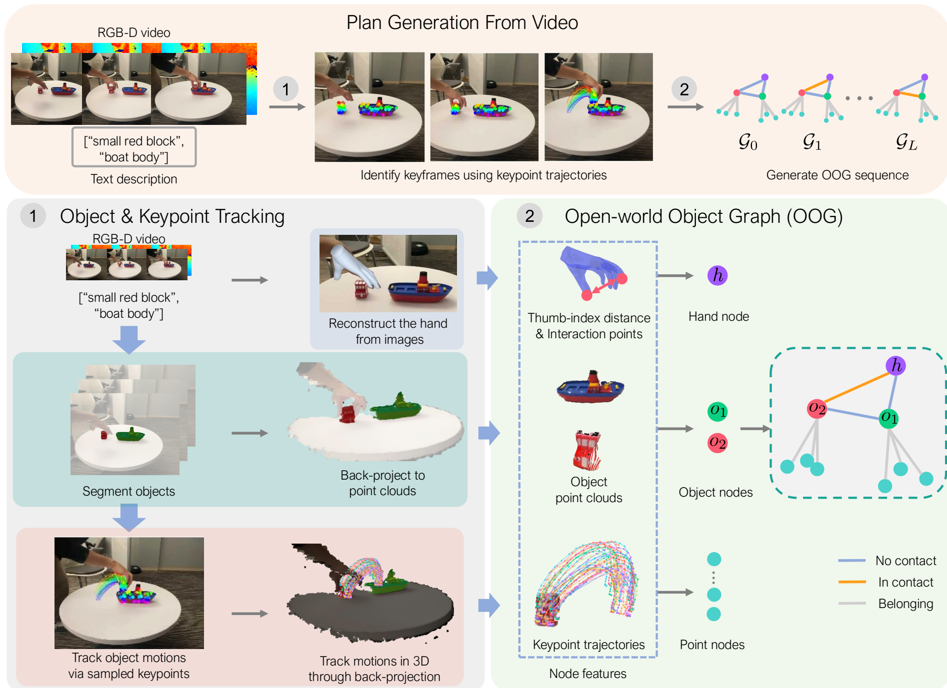

Figure 2: [Figure updated] **Overview of plan generation in ORION.** ORION generates a manipulation plan from a given video V in order for subsequent policies to synthesize actions. ORION first tracks the objects and keypoints across the video frames. Then keyframes are identified based on the velocity statistics of the keypoint trajectories. Then ORION generates an Open-world Object Graph (OOG) for every keyframe, resulting in a sequence of OOGs that serves as the spatiotemporal abstraction of the video. The figure is viewed best in color.

and where objects are and also represents their geometry information. Additionally, to inform the robot where to interact with objects (e.g. where to grasp), we introduce the specialized "hand node", which stores the interaction cues such as contact points and the grip status (open or closed) that can be directly mapped to the robot end-effector during execution. At the low level, each point node corresponds to a keypoint that belongs to a task-relevant object. Every point node comes with the feature, namely the 3D motion trajectories. The feature explicitly models how an object should be moved during a manipulation task. In the rest of the paper, by motion features of a point node in G_l, we mean 3D trajectory between keyframe l and $l+1$.

In an OOG, all the object nodes and the hand node are fully connected, reflecting real-world spatial relationships. Each edge is augmented with a binary attribute that indicates if two objects or objects and the hand are in contact. This attribute allows our algorithm to check the set of satisfied contact relations, retrieving the matched OOG from the generated plan (see Section 3.2). The low-level point nodes are connected to their respective object node, indicating a belonging relationship. We denote node entities from human videos with a superscript V, and denote the ones from the robot rollout with a superscript Ro. Table 1 in the appendix also summarizes the variables needed in an OOG.

3.2 Manipulation Plan Generation From V

We describe the first part of ORION (see Figure 2), which automatically annotates the video and generates a manipulation plan from V. Here, a manipulation plan is a spatiotemporal abstraction of V that centers around the object states and their motions over time. Our core insight is that a task can be cost-effectively modeled with object locations at some keyframe states where the set of satisfied contact relations are changed, and abstract the rest of the states into 3D motions of objects. Concretely, a plan is represented as a sequence of OOGs, $\{\mathcal{G}_l\}_{l=0}^{L}$ which corresponds to $L+1$ keyframes in V, with $\mathcal{G}_0$ representing the initial state.

Tracking task-relevant objects. ORION first localizes task-relevant objects in the video V. Given V and the list of object descriptions mentioned in Section 2, ORION uses an open-world vision model, Grounded-SAM [18], to annotate video frames with segmentation masks of the task-relevant objects. In practice, due to the demanding computation of using open-world vision models, we reduce the computation by exploiting object permanence to track the objects. Specifically, ORION annotates the first video frame with Grounded-SAM, and then propagates the segmentation to the rest of the video using a Video Object Segmentation Model, Cutie [19].

Discovering keyframes. After annotating the locations of task-relevant objects, we track their motions across the video to discover the keyframes based on the velocity statistics of object motions. This design is based on the observation that changes in object contact relations due to manipulation are often accompanied by sudden changes in object motions (e.g., transitioning from free space motion to grasping an object). However, keeping full track of object point motion using techniques like optical flow estimation requires heavy computation and the tracking quality is susceptible to noisy observations, largely due to occlusions during manipulation. We use a Track-Any-Point (TAP) model, namely CoTracker [20], to track a subset of points in a long-term video with explicit occlusion modeling, which has been successfully applied to track object motions in robot manipulation [21, 22]. Specifically, we first sample keypoints within the object segmentation of the first frame and track the trajectories across the video. The changes in velocity statistics are straightforward to detect based on the TAP trajectories, where we discover the keyframes using a standard unsupervised changepoint detection algorithm [23], a common technique that has been used in robotics applications [24, 25].

Generating OOGs from V. Once ORION discovers the keyframes, it generates an OOG at each keyframe to model the state of task-relevant objects and the human hand in V. The creation of OOG nodes reuses the results from the annotation process: for object nodes, the point clouds for node features are back-projected from the object segmentation using depth data; for the point nodes, each node corresponds to the sampled keypoints, and their motion features, 3D trajectories, are back-projected from the TAP trajectories using depth. Additionally, hand information is required to specify the interaction points with task-relevant objects and the grip status to be mapped to the robot gripper. We use a hand-reconstruction model, HaMeR [26], which gives a reconstructed hand mesh that pinpoints the hand locations at each keyframe. The distances between the fingertips of the mesh help determine the grip status, i.e., whether it is open or closed.

With all the node information, ORION establishes the edge connections between nodes in OOGs, representing contact relations. Since all object and hand locations are computed in the camera frame while the camera extrinsic of V is unknown, there is ambiguity when deciding the spatial relations between objects. We exploit the assumption of tabletop manipulation, where a table is always present with its normal direction aligned with the z-axis of the world coordinate system. So ORION estimates the transformation matrix of the table plane and transforms all the point cloud features in OOGs to align with the xy plane of the world coordinate (Full details appear in Appendix C.2). Then, the contact relations in each state can be determined based on the spatial relations and the computed distances between point clouds. The relations allow ORION to match the test-time observations with a keyframe state from the plan and subsequently decide which object to manipulate (see Section 3.3). In the end, ORION generates a complete OOG for each discovered keyframe.

3.3 Robot Policy To Synthesize Actions

Given a manipulation plan, ORION derives a manipulation policy that synthesizes actions based on the aforementioned objective to achieve object motion similarities (detailed in Figure 3). The action synthesis comprises three major steps: identify a keyframe from the plan that matches the current observation, predict object motions, and use the predictions to optimize the robot actions. These three steps are repeated until a task is completed or fails, detailed in Appendix E. The resulting ORION policy is robust to visual variations due to the use of open-world vision models. It also generalizes to different spatial locations due to our choice of representing object locations in object-centric frames and the optimization process that is not constrained to specific positions.

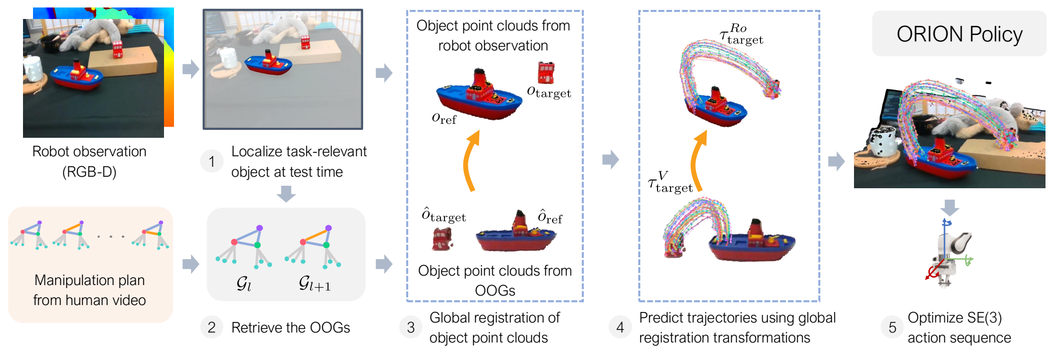

Figure 3: **Overview of the ORION Policy**. (1) ORION first localizes task-relevant objects at test time. (2) Next, ORION retrieves the matched OOGs from the generated manipulation plan. (3) ORION obtains the point clouds of the target object from the observation and the OOGs, namely o_{target} and $\hat{o}_{\text{target}}$, and those of the reference object, o_{ref} and $\hat{o}_{\text{ref}}$. Global registration is then performed to compute two transformations, one from $\hat{o}_{\text{target}}$ to o_{target}, and the other from $\hat{o}_{\text{ref}}$ to o_{ref}. (4) ORION then uses the computed transformations to warp τ^{V}_{target}, keypoint trajectories of the target object from the OOGs, into the workspace (details in the main text). The trajectory warping results in a predicted trajectory $\tau^{Ro}_{\text{target}}$. (5) ORION then uses $\tau^{Ro}_{\text{target}}$ to optimize the SE(3) action sequence of the robot end effector, which is subsequently used to command the robot.

Retrieving OOGs from the plan. ORION identifies the keyframe and retrieves OOGs to help decide what next actions to take. At test-time, ORION localizes objects in the new observations and estimates contact relations using the same vision pipeline as described in Section 3.2. Then ORION retrieves the OOG that has the same set of relations as the current state, allowing us to identify a pair $(\mathcal{G}_l, \mathcal{G}_{l+1})$, where $\mathcal{G}_l$ is the retrieved graph and $\mathcal{G}_{l+1}$ the graph of the next keyframe. This pair of graphs provides sufficient information to decide which object to manipulate next, termed the *target object*, and we denote its point cloud at keyframe l as $\hat{o}_{\text{target}}$, and its keypoint trajectories as τ^{V}_{target}. A target object is the one in motion due to manipulation between two keyframes, and it is determined by computing the average velocity per-object using motion features in G_l. At the same time, another object, called the *reference object*, serves as a spatial reference for the target object's movement when contact state relations change from $\mathcal{G}_l$ to $\mathcal{G}_{l+1}$. We use the point cloud of the reference object at *next keyframe* $l+1$, as object interactions might cause state changes of the reference object, and the information from the next keyframe gives us an accurate prediction of the trajectories. Once the target and reference objects are determined, we localize the corresponding objects in the new observations and denote their point clouds as o_{target} and o_{ref}, respectively.

Predicting object motions. Given the target and reference objects from keyframes l, and $l+1$, we predict the motion of the target object in the current state by warping the keypoint trajectories estimated from V. To warp the trajectories, we first identify the initial and goal locations of keypoints in the new configuration by leveraging information given by the OOG pair. We use global registration of point clouds [27] to align $\hat{o}_{\text{target}}$ with o_{target} and $\hat{o}_{\text{ref}}$ with o_{ref}, giving us two transformations to compute the new starting and goal positions of target object keypoints conditioned on where the reference object is. Then we normalize τ^{V}_{target} with its starting and goal locations, obtaining $\hat{\tau}_{\text{target}}$. $\hat{\tau}_{\text{target}}$ only contains the directional and curvature patterns that are independent of the absolute location of the initial and the goal keypoints. Then we scale it back to the workspace coordinate frame using the new starting and goal locations, resulting in new keypoint trajectories of the target object $\tau^{Ro}_{\text{target}}$.

Optimizing robot actions. Once we obtain $\tau^{Ro}_{\text{target}}$, we optimize for a sequence of SE(3) transformations that guide the robot end-effector to move. The SE(3) transformations are optimized to align the keypoint locations from previous frames to the next frames along the predicted trajectories:

$$\min_{T_0, T_1, \ldots, T_{t_{l+1}-t_l}} \sum_{i=0}^{t_{l+1}-t_l} \left(\tau^{Ro}_{\text{target}}(i+1) - T_i \tau^{Ro}_{\text{target}}(i)\right) \tag{1}$$

where $\tau^{Ro}_{\text{target}}(i)$ $(0 \leq i \leq t_{l+1} - t_l)$ represents the keypoint locations at timestep i along the trajectory. This optimization process naturally allows generalizations over spatial variations, as the action sequence always conditions on a new location instead of overfitting to fixed locations. To further specify where the gripper should interact with the object and whether it should be open or closed,

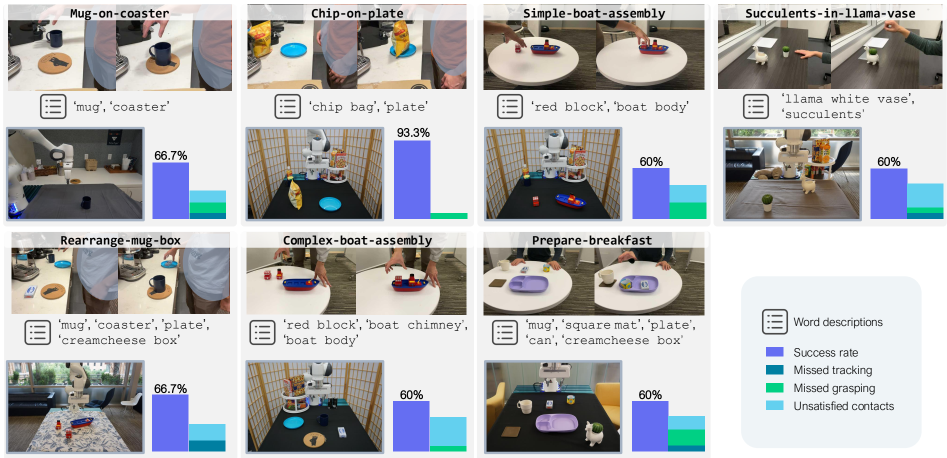

Figure 4: [Figure updated] This figure includes the following: task names, the initial and final frames of human videos, the list of word descriptions provided along with videos, snapshots of robot evaluation, and overall policy evaluation over all seven tasks, including the success rates and the quantification of failed trials, separated by failure mode. "Missed tracking" is the perception failure due to the vision foundation models, specifically the case of test-time object localization using Grounded-SAM.

we augment the resulting SE(3) sequence with the interaction information stored in the hand node h. To determine the initial pose of the end-effector in the sequence, ORION maps the two contact points using the computed transformation between $\hat{o}_{\text{target}}$ and o_{target}. The mapped points correspond to the two finger tips of the robot gripper, and the robot's gripper pose is determined by solving a simple inverse kinematics problem using the robot URDF file. We implement a combination of inverse kinematics (IK) and joint impedance control to achieve precise and compliant execution.

4 Experiments

In this section, we report on experiments to answer the following questions regarding the effectiveness of ORION and the important design choices. 1) Is ORION effective at constructing manipulation policies given a single human video in the open-world setting? 2) To what extent does the object-centric abstraction improve the policy performance? 3) How critical is it to model the object motions with keypoints and the TAP formulation? 4) How consistent is the performance of ORION's policy given videos taken in different conditions? 5) How effectively does ORION scale to long-horizon manipulation tasks?

4.1 Experiment Setup

Task descriptions. We design seven tasks to evaluate ORION poliies: 1) `Mug-on-coaster`: placing a mug on the coaster; 2) `Simple-boat-assembly`: putting a small red block on a toy boat; 3) `Chips-on-plate`: placing a bag of chips on the plate; 4) `Succulents-in-llama-vase`: inserting succulents into the llama vase; 5) `Rearrange-mug-box`: placing a mug on a coaster and placing a cream cheese box on a plate consecutively; 6) `Complex-boat-assembly`: placing both a small red block and a chimney-like part on top of a boat. 7) `Prepare-breakfast`: placing a mug on a coaster and putting a food box and can on the plate. The first four are "short-horizon" tasks that only require one contact relation between two objects, and the last three are "long-horizon" tasks that require more than one contact relation. Detailed success conditions of all tasks are described in Appendix E. Details about video recording, robot setup and evaluation can be found in Appendix B.

Baselines. To understand the model capacity and validate our design choices, we compare ORION with baselines. Since no prior work exists that matches the exact setting of our approach, we adopt the most important components from prior works and treat them as baselines to our model. Specifically, we implement the following two baselines: 1) HAND-MOTION-IMITATION [9, 28] is a baseline

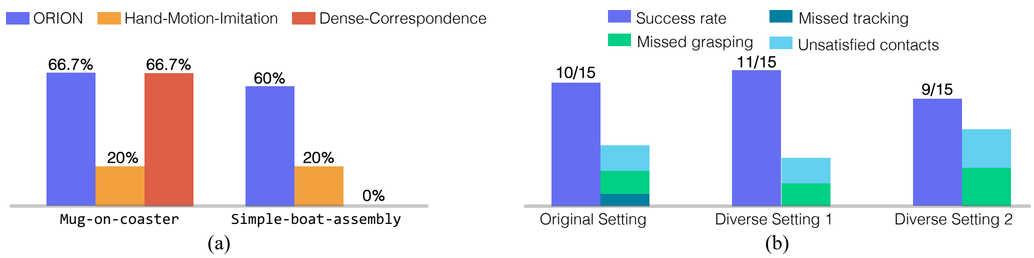

Figure 5: (a) Comparison experiments between ORION and the two baselines, namely HAND-MOTION-IMITATION and DENSE-CORRESPONDENCE. (b) Ablation study on using different videos of the task `Mug-on-coaster`. We show the number of successful trials out of 15 total trials on the bar plots for each setting. Figure 6 in Appendix F visualizes the different settings in this experiment.

that predicts robot actions by learning from the hand trajectories. The rest of the parts remain the same as ORION. We use this baseline to show whether it is critical to compute actions centering around objects. 2) DENSE-CORRESPONDENCE [15, 29] is a baseline that replace the TAP model in ORION with a dense correspondence model, optical flows. This baseline is used to evaluate whether our choice of TAP model is a better design. For this ablative study, we conduct experiments on `Mug-on-coaster` and `Simple-boat-assembly` to validate our model design, covering the distribution of common daily objects and assembly manipulation that requires precise control.

4.2 Experimental Results

Our evaluations are presented in Figures 4 and 5. We answer question (1) by showing the successful deployment of the ORION policies, while no other methods are designed to be able to operate in our setting. Furthermore, ORION yields an average of 66.7% success rates, which validates our model design in imitating from a single human video in the open-world setting.

We then answer question (2), showing the comparison results in Figure 5(a) against the baseline, HAND-MOTION-IMITATION, which yields low success rates in both tasks. Concretely, HAND-MOTION-IMITATION typically succeeds in trials where the initial spatial layouts are similar to the one in V. Its major failure mode is not being able to reach the target object configuration, e.g., misplacing the mug on the table while not achieving contact with the coaster. These results imply that learning from human hand motion from V results in poor generalization abilities of policies, supporting the design choice of ORION which focuses on the object-centric information.

We further answer question (3) by comparing the performance between ORION and the optical flow baseline, DENSE-CORRESPONDENCE. The baseline performs drastically worse on `Simple-boat-assembly` than on `Mug-on-coaster`. Our further investigation shows that the baseline discovers keyframes in the middle of smooth transitions as opposed to changes in object contact relations, resulting in a manipulation plan that computes completely wrong actions.

To answer question (4), we conduct controlled experiments using the task `Mug-on-coaster`. We record two additional videos of the task in very different visual conditions and spatial layouts (see pictures in Appendix F) and construct a policy from each video. Then, we compare the two policies against the original one using the same set of evaluation conditions. The result in Figure 5(b) shows that there is no statistically significant difference in the performance, demonstrating that ORION is robust to videos taken under different visual conditions. Finally, we show that ORION is effective in scaling to long-horizon tasks. This conclusion is supported by the performance among the pairs of `Mug-on-coaster` versus `Rearrange-mug-box`, and `Simple-boat-assembly` versus `Complex-boat-assembly`. Both the short-horizon tasks are subgoals of their long-horizon counterparts, yet we do not see any performance drop between the two. Such result shows that ORION excels at scaling to long-horizon tasks without a significant drop in performance.

References

[1] M. Dalal, D. Pathak, and R. R. Salakhutdinov. Accelerating robotic reinforcement learning via parameterized action primitives. *Advances in Neural Information Processing Systems*, 34: 21847–21859, 2021.

[2] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *arXiv preprint arXiv:2003.06085*, 2020.

[3] S. Nasiriany, H. Liu, and Y. Zhu. Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7477–7484. IEEE, 2022.

[4] R. Zhang, S. Lee, M. Hwang, A. Hiranaka, C. Wang, W. Ai, J. J. R. Tan, S. Gupta, Y. Hao, G. Levine, et al. Noir: Neural signal operated intelligent robots for everyday activities. *arXiv preprint arXiv:2311.01454*, 2023.

[5] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. *arXiv preprint arXiv:2210.11339*, 2022.

[6] A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021.

[7] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[8] 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. *arXiv preprint arXiv:2210.00030*, 2022.

[9] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[10] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg. Learning by watching: Physical imitation of manipulation skills from human videos. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7827–7834. IEEE, 2021.

[11] Y. Zhu, Z. Jiang, P. Stone, and Y. Zhu. Learning generalizable manipulation policies with object-centric 3d representations. In *7th Annual Conference on Robot Learning*, 2023.

[12] F. Torabi. *Imitation Learning from Observation*. PhD thesis, University of Texas at Austin, 2021. PhD Thesis.

[13] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

[14] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[15] P.-C. Ko, J. Mao, Y. Du, S.-H. Sun, and J. B. Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023.

[16] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

[17] K. Joseph, S. Khan, F. S. Khan, and V. N. Balasubramanian. Towards open world object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5830–5840, 2021.

[18] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[19] H. K. Cheng, S. W. Oh, B. Price, J.-Y. Lee, and A. Schwing. Putting the object back into video object segmentation. *arXiv preprint arXiv:2310.12982*, 2023.

[20] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023.

[21] 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. *arXiv preprint arXiv:2308.15975*, 2023.

[22] B. Wen, W. Lian, K. Bekris, and S. Schaal. You only demonstrate once: Category-level manipulation from single visual demonstration. *arXiv preprint arXiv:2201.12716*, 2022.

[23] R. Killick, P. Fearnhead, and I. A. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598, 2012.

[24] C. Wen, J. Lin, J. Qian, Y. Gao, and D. Jayaraman. Keyframe-focused visual imitation learning. *arXiv preprint arXiv:2106.06452*, 2021.

[25] S. Niekum, S. Osentoski, C. G. Atkeson, and A. G. Barto. Online bayesian changepoint detection for articulated motion models. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 1468–1475. IEEE, 2015.

[26] G. Pavlakos, D. Shan, I. Radosavovic, A. Kanazawa, D. Fouhey, and J. Malik. Reconstructing hands in 3d with transformers. *arXiv preprint arXiv:2312.05251*, 2023.

[27] S. Choi, Q.-Y. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5556–5565, 2015.

[28] H. Bharadhwaj, A. Gupta, S. Tulsiani, and V. Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.

[29] N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada. Ditto: Demonstration imitation by trajectory transformation. *arXiv preprint arXiv:2403.15203*, 2024.

[30] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[31] S. Kumar, J. Zamora, N. Hansen, R. Jangir, and X. Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, pages 55–66. PMLR, 2023.

[32] Y. Liu, A. Gupta, P. Abbeel, and S. Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1118–1125. IEEE, 2018.

[33] P. Sharma, D. Pathak, and A. Gupta. Third-person visual imitation learning via decoupled hierarchical controller. *Advances in Neural Information Processing Systems*, 32, 2019.

[34] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos. *arXiv preprint arXiv:1912.04443*, 2019.

[35] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song. Xskill: Cross embodiment skill discovery. In *Conference on Robot Learning*, pages 3536–3555. PMLR, 2023.

[36] R. Lee, J. Abou-Chakra, F. Zhang, and P. Corke. Learning fabric manipulation in the real world with human videos. *arXiv preprint arXiv:2211.02832*, 2022.

[37] K. Shaw, S. Bahl, A. Sivakumar, A. Kannan, and D. Pathak. Learning dexterity from human hand motion in internet videos. *The International Journal of Robotics Research*, 43(4):513–532, 2024.

[38] A. Bahety, P. Mandikal, B. Abbatematteo, and R. Martín-Martín. Screwmimic: Bimanual imitation from human videos with screw space projection. *arXiv preprint arXiv:2405.03666*, 2024.

[39] H. Bharadhwaj, A. Gupta, V. Kumar, and S. Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. *arXiv preprint arXiv:2312.00775*, 2023.

[40] B. S. Pavse, F. Torabi, J. Hanna, G. Warnell, and P. Stone. Ridm: Reinforced inverse dynamics modeling for learning from a single observed demonstration. *IEEE Robotics and Automation Letters*, 5(4):6262–6269, 2020.

[41] H. Karnan, F. Torabi, G. Warnell, and P. Stone. Adversarial imitation learning from video using a state observer. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2452–2458. IEEE, 2022.

[42] F. Torabi, G. Warnell, and P. Stone. Imitation learning from video by leveraging proprioception. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3585–3591, 2019.

[43] F. Torabi, G. Warnell, and P. Stone. Generative adversarial imitation from observation. In *Imitation, Intent, and Interaction (I3) Workshop at ICML 2019*, June 2019.

[44] F. Torabi, G. Warnell, and P. Stone. Behavioral cloning from observation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4950–4957, 2018.

[45] Y. Duan, M. Andrychowicz, B. Stadie, O. Jonathan Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. *Advances in neural information processing systems*, 30, 2017.

[46] N. Di Palo and E. Johns. Learning multi-stage tasks with one demonstration via self-replay. In *Conference on Robot Learning*, pages 1180–1189. PMLR, 2022.

[47] S. Haldar, V. Mathur, D. Yarats, and L. Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, pages 32–43. PMLR, 2023.

[48] S. Haldar, J. Pari, A. Rai, and L. Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023.

[49] E. Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 4613–4619. IEEE, 2021.

[50] E. Valassakis, G. Papagiannis, N. Di Palo, and E. Johns. Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8614–8621. IEEE, 2022.

[51] A. Jonnavittula, S. Parekh, and D. P. Losey. View: Visual imitation learning with waypoints. *arXiv preprint arXiv:2404.17906*, 2024.

[52] N. Di Palo and E. Johns. Dinobot: Robot manipulation via retrieval and alignment with vision foundation models. *arXiv preprint arXiv:2402.13181*, 2024.

[53] D. Guo. Learning multi-step manipulation tasks from a single human demonstration. *arXiv preprint arXiv:2312.15346*, 2023.

[54] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv preprint arXiv:1809.10790*, 2018.

[55] S. Tyree, J. Tremblay, T. To, J. Cheng, T. Mosier, J. Smith, and S. Birchfield. 6-dof pose estimation of household objects for robotic manipulation: An accessible dataset and benchmark. *arXiv preprint arXiv:2203.05701*, 2022.

[56] T. Migimatsu and J. Bohg. Object-centric task and motion planning in dynamic environments. *IEEE Robotics and Automation Letters*, 5(2):844–851, 2020.

[57] D. Wang, C. Devin, Q.-Z. Cai, F. Yu, and T. Darrell. Deep object-centric policies for autonomous driving. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8853–8859. IEEE, 2019.

[58] C. Devin, P. Abbeel, T. Darrell, and S. Levine. Deep object-centric representations for generalizable robot learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7111–7118. IEEE, 2018.

[59] J. Shi, J. Qian, Y. J. Ma, and D. Jayaraman. Plug-and-play object-centric representations from "what" and "where" foundation models.

[60] N. Di Palo and E. Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024.

[61] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, B. Zitkovich, F. Xia, C. Finn, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[62] K. Mo, P. Guerrero, L. Yi, H. Su, P. Wonka, N. Mitra, and L. J. Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575*, 2019.

[63] Y. Huang, A. Conkey, and T. Hermans. Planning for multi-object manipulation with graph neural network relational classifiers. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1822–1829. IEEE, 2023.

[64] A. H. Qureshi, A. Mousavian, C. Paxton, M. C. Yip, and D. Fox. Nerp: Neural rearrangement planning for unknown objects. *arXiv preprint arXiv:2106.01352*, 2021.

[65] Y. Zhu, J. Tremblay, S. Birchfield, and Y. Zhu. Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6541–6548. IEEE, 2021.

[66] M. Sieb, Z. Xian, A. Huang, O. Kroemer, and K. Fragkiadaki. Graph-structured visual imitation. In *Conference on Robot Learning*, pages 979–989. PMLR, 2020.

[67] C. Truong, L. Oudre, and N. Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

[68] Q.-Y. Zhou, J. Park, and V. Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018.

[69] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009.

[70] Z. Teed and J. Deng. Tangent space backpropagation for 3d transformation groups. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10338–10347, 2021.

A Related Work

Learning Manipulation From Human Videos. Human videos offer a rich repertoire of object interaction behaviors, making them an invaluable data source for manipulation. A large body of work has explored how to leverage human video data for learning robot manipulation [9, 10, 30–35], either through pre-training a single latent representation [7, 9, 35], learning representations of perception or action priors [36, 37], learning an implicit reward function [6, 8], learning 6D representations of actions [38], or learning generative models that in-paint human morphologies [15, 28, 30, 39]. However, they either require additional robot data from the target tasks or paired data between humans and robots. Our approach takes a novel direction by tackling how a robot can imitate or learn from a single human video only: the robot does not rely on pre-existing data, models, or ground-truth annotations *in scenes* where video recording and robot evaluation take place. We refer to such a setting as *open-world imitation from observation*, where the robot is not programmed or trained to interact with the objects in the video *a priori* and the video data does not come with any robot actions. Our setting is closely related to the problem of "Imitation Learning from Observation" [12], where state-only demonstrations are used to construct policies for physical interaction. However, this line of prior work assumes simulators of demonstrated tasks exist and physical states of the agents or objects are known [40–44]. In contrast, our setting does not assume the digital replica of real-world tasks, and all the object information is only perceived through RGB-D videos.

Learning Manipulation From a Single Demonstration. Studies have delved into learning manipulation policies from one demonstration. A notable one is one-shot imitation learning within meta-learning framework proposed by Duan et al. [45]. While prior works on one-shot imitation learning have shown a robot performing new tasks from one demonstration, they require extensive in-domain data and a well-curated set of meta-training tasks beforehand, leading to significant data collection costs and restricted policy generalization at test time due to the tailored nature of the training.

An alternative approach involves using a single demonstration for initial guidance, refining the policy through real-world self-play [22, 46–51]. However, this approach mainly applies to reset-free tasks and struggles with scaling to multi-stage tasks where resetting to the task initial conditions does not come free. Recently, foundation models are used to enable learning manipulation from a single demonstration, but existing works require ground-truth access to the robot action through kinesthetic teaching [52].

Our work aligns with these studies in using a single demonstration for learning manipulation, but stands out by not needing prior data or self-play. Recent or concurrent works have also explored using a single video demonstration only [29, 53], but they either assume known object instances or lack in formulating systematic generalization in an open-world setting described in Section 2. With just one single human video, our method constructs a policy that successfully completes the task, while adapting to a wide range of visual and spatial differences from the task instance of video demonstration.

Object-Centric Representation for Learning Robot Manipulation. The concept of object-centric representation has long been recognized for its potential to enhance robotic perception and manipulation by focusing on the objects within a scene. Prior works have shown effectiveness of such representation in downstream manipulation tasks by factorizing visual scenes into disentangled object concepts [54–58], but these works are typically confined to known object categories or instances. Recent developments in foundation models allow robots to access the open-world object concepts through pre-trained vision models [13, 14], enabling a wide range of abilities such as imitation of long-horizon tabletop manipulation [5, 59], in-context learning of tabletop manipulation [60], or mobile manipulation in the wild [61]. Building upon these advances, our work focuses on leveraging open-world, object-centric concepts in imitating manipulation behaviors from actionless human videos. We propose a graph-based representation called Open-world Object Graph (OOG), which allows a robot to imitate from a human video by leveraging the object-centric concepts. This proposed representation shares a similar vein with prior works that factorize scene or task-relevant visual concepts into scene graphs [31, 62–66]. However, our representation is tailored to integrate open-world object concepts and enable generalization across different embodiments, specifically a human and a robot.

B Additional Details of Experimental Setup

Experimental setup. We design experiments to fully test the efficacy of our method by providing the robot with videos captured in everyday scenarios, which naturally encompass visual backgrounds and camera setups that are different from the one for the robot. Specifically, we record an RGB-D video of a person performing each of the seven tasks in everyday scenarios, such as an office or a kitchen. We use an iPad for recording, which comes with a TrueDepth Camera, and we fix it on a camera stand. The videos can be found in the supplementary materials. During test time, the robot receives visual data through a single RGB-D camera, Intel Realsense435, and performs manipulation in its workstation to evaluate policies. We use the 7DoF Franka Emika Panda robot for all the experiments.

Evaluation protocol. As we describe in the experimental setup, the videos naturally include various visual backgrounds and camera perspectives that are significantly different from the robot workspace. Therefore, we only intentionally vary two dimensions before evaluating each trial of robot execution, namely the spatial layouts and the new object instances. Furthermore, the new object generalizations are included in the tasks `Mug-on-coaster` and `Chips-on-plate` as mugs and chip bags have many similar instances. As for the other three tasks, there are no novel objects involved, but we extensively vary the spatial layouts of task-relevant objects for evaluation. The policy performance of a task is the averaged success rates over 15 real-world trials. Aside from the success rates, we also group the failed executions into three types: *Missed tracking* of objects due to failure of the vision models, *Missed grasping* of objects during execution, and *Unsatisifed contacts* where the target object configurations are not achieved for reasons other than the previous two failure types.

C Additional Technical Details

C.1 Data Structure of an OOG.

For easy reproducibility of the proposed method, we present a table that explains the data structure of an OOG.

Node/Edge	Type	Attributes
$\mathcal{G}.vo_i$	Object Node	3D point cloud of an object.
$\mathcal{G}.vh$	Hand Node	Hand mesh and locations of the thumb and index finger.
$\mathcal{G}.vp_{ij}$	Point Node	A trajectory of a TAP keypoint between two keyframes, recorded in xyz positions.
$\mathcal{G}.eo_{ik}$	Object-Object Edge	A binary value of contact or not.
$\mathcal{G}.eh_i$	Object-Hand Edge	A binary value of contact or not.
$\mathcal{G}.ep_{ij}$	Object-Point Edge	The presence of an edge represents the belonging relation, and no specific feature is attached.

Table 1: Data Structure of an OOG. For a given OOG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, it has $\mathcal{V} = \{\mathcal{G}.vo_i\} \cup \{\mathcal{G}.vh\} \cup \{\mathcal{G}.vp_{ij}\}$, and $\mathcal{E} = \{\mathcal{G}.eo_{ik}\} \cup \{\mathcal{G}.eh_i\} \cup \{\mathcal{G}.ep_{ij}\}$.

C.2 Implementation Details

Changepoint detections. We use changepoint detection to identify changes in velocity statistics of TAP keypoints. Specifically, we use a kernel-based changepoint detection method and choose radial basis function [23]. The implementation of this function is directly based on an existing library Ruptures [67].

Plane estimation. In Section 3.2, we mentioned using the prior knowledge of tabletop manipulation scenarios and transforming the point clouds by estimating the table plane. Here, we explain how the plane estimation is computed. Concretely, we rely on the plane estimation function from Open3D [68], which gives an equation in the form of $ax + by + cz = d$. From this estimated plane equation, we can infer a normal vector of the estimated table plane, (a, b, c), in the camera coordinate frame. Then, we align this plane with xy plane in the world coordinate frame, where we compute a transformation matrix that displaces the normal vector (a, b, c) to the normalized vector $(0, 0, 1)$ along the z-axis of the world coordinate frame. This transformation matrix is used to transform point clouds in every frame so that the plane of the table always aligns with the xy plane of the world coordinate.

Object localization at test time. When we localize objects at test time, there could be some false positive segmentation of distracting objects. Such vision failures will prevent the robot policy from successfully executing actions. To exclude such false positive object segmentaiton, we use Segmentation Correspondence Model (SCM) from GROOT [11], where SCM filters out the false positive segmentation of the objects by computing the affinity scores between masks using DINOv2 features.

Global registration. In this paper, we use global registration to compute the transformation between observed object point clouds from videos and those from rollout settings. We implement this part using a RANSAC-based registration function from Open3D [68]. Specifically, given two object point clouds, we first compute their features using Fast-Point Feature Histograms (FPFH) [69], and then perform a global RANSAC registration on the FPFH features of the point clouds [27].

Implementation of SE(3) optimization. We parameterize each homogeneous matrix T_i into a translation variable and a rotation variable and randomly initialize each variable using the normal distribution. We choose quaternions as the representation for rotation variables, and we normalize the randomly initialized vectors for rotation so that they remain unit quaternions. With such parameterization, we optimize the SE(3) end-effector trajectories $T_0, T_1, \ldots, T_{t_{l+1}-t_l}$ over the Objective (1). However, jointly optimizing both translation and rotation from scratch typically results in trivial solutions, where the rotation variables do not change much from the initialization due to the vanishing gradients. To avoid trivial solutions, we implement a two-stage process. In the first stage, we only optimize the rotation variables with 200 gradient steps. Then, the optimization proceeds to the second stage, where we optimize both the rotation and translation variables for another 200 gradient steps. In this case, we prevent the optimization process from getting stuck in trivial solutions for rotation variables. We implement the optimization process using Lietorch [70].

D System Setup

Details of camera observations. As mentioned in Section 4, we use an iPad with a TrueDepth camera for collecting human video demonstrations. We use an iOS app, Record3D, that allows us to access the depth images from the TrueDepth camera. We record RGB and depth image frames in sizes 1920×1080 and 640×480, respectively. To align the RGB images with the depth data, we resize the RGB frames to the size 640×480. The app also automatically records the camera intrinsics of the iPhone camera so that the back-projection of point clouds is made possible.

To stream images at test time, we use an Intel Realsense D435i. In our robot experiments, we use RGB and depth images in the size 640×480 or 1280×720 in varied scenarios, all covered in our evaluations. Evaluating on different image sizes showcases that our method is not tailored to specific camera configurations, supporting the wide applicability of constructed policy.

Implementation of real robot control. In our evaluation, we reset the robot to a default joint position before object interaction every time. Then we use a reaching primitive for the robot to reach the interaction points. Resetting to the default joint position enables an unoccluded observation of task-relevant objects at the start of each decision-making step. Note that the execution of object interaction does not necessarily require resetting. To command the robot to interact with objects, we convert the optimized SE(3) action sequence to a sequence of joint configurations using inverse kinematics and control the robot using joint impedance control. We use the implementation of Deoxys [5] for the joint impedance controller that operates at 500 Hz. To avoid abrupt motion and make sure the actions are smooth, we further interpolate the joint sequence from the result of inverse kinematics. Specifically, we choose the interpolation so that the maximal displacement for each joint does not exceed 0.5 radian between two adjacent waypoints.

E Success conditions of tasks

We describe the success conditions for each of the tasks in detail:

- `Mug-on-coaster`: A mug is placed upright on the coaster.
- `Simple-boat-assembly`: A red block is placed in the slot closest to the back of the boat. The block needs to be upright in the slot.
- `Chips-on-plate`: A bag of chips is placed on the plate, and the bag does not touch the table.
- `Succulents-in-llama-vase`: A pot of succulents is inserted into a white vase in the shape of a llama.
- `Rearrange-mug-box`: The mug is placed upright on the coaster, and the cream cheese box is placed on the plate.
- `Complex-boat-assembly`: The chimney-like part is placed in the slot closest to the front of the boat. The red block is placed in the slot closest to the back of the boat. Both blocks need to be upright in the slots.
- `Prepare-breakfast`: The mug is placed on top of a coaster, the cream cheese box is placed in the large area of the plate, and the food can is placed on the small area as shown in the video demonstration.

In practice, we record the success and failure of a rollout as follows: If the program in ORION policy returns true when matching the observed state with the final OOG from a plan, we mark a trial as success as long as we observe that the object state indeed satisfies the success condition of a task as described above. Otherwise, if the robot generates dangerous actions (bumping into the table) or does not achieve the desired subgoal after executing the computed trajectory, we consider the rollout as a failure and we manually record the failure.

F Additional Details on Experiments

Diverse video recordings used in the ablation study. Figure 6 shows the three videos taken in very different scenarios: kitchen, office, and outdoor. The video taken in kitchen scenario is used in the major quantitative evaluation, termed “Original setting”. The other two settings are termed “Diverse setting 1” and “Diverse setting 2.” We conduct an ablation study where we compare policies imitated from these three videos, which inherently involve varied visual scenes, camera perspectives. The result of the ablation study is shown in Figure 5.

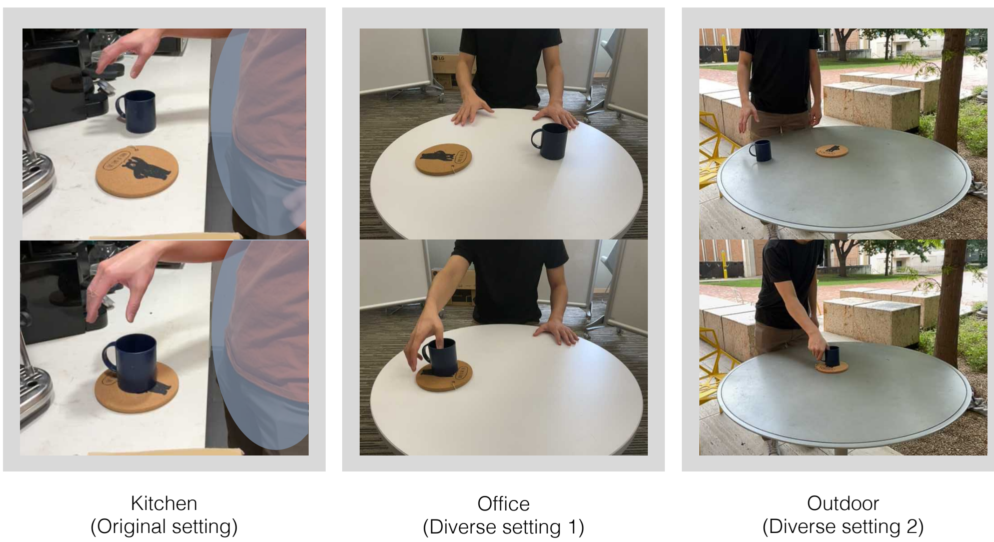

Figure 6: This figure visualizes the initial and final frames of the three videos of the same task `Mug-on-coaster`.