---

**Article:**
Chen, L., Russell, D., Qiu, Y. et al. (2026) Cross-Behavior Learning with Object Flow Prediction for Robotic Manipulation. IEEE Transactions on Robotics. ISSN: 1552-3098

https://doi.org/10.1109/tro.2026.3701576

---

# Cross-Behavior Learning with Object Flow Prediction for Robotic Manipulation

Longrui Chen, *Student Member*, David Russell, Yulei Qiu, Mehmet Dogar, *Member*

***Abstract*—Cross-embodiment learning enables robots to acquire manipulation skills by learning from demonstrations provided by different embodiments. However, most existing research on cross-embodiment learning focuses on transferring similar manipulation behaviors. For the same task, different embodiments may need different *behaviors*; e.g., it may be easier for a human to push an object to a goal position, while a robot may use a pick-and-place to perform the same task. Making use of such cross-embodiment *and* cross-behavior demonstrations becomes essential for large-scale imitation learning. In this work, we propose a novel framework for cross-behavior learning based on object flow. Object flow represents the task in a manner that is independent from the embodiment and the particular behavior used during the demonstration. By shifting the focus from the manipulator to objects, our framework enables learning from cross-behavior manipulation data rather than merely imitating the behavior of a specific embodiment. Our results on task representations show that, with only 20 robot demonstrations, integrating object flow prediction improves success rates by up to 91% in-domain and 87% out-of-domain. Adding 40 human demonstrations in addition to the 20 robot demonstrations further boosts out-of-domain performance by 143%.**

***Index Terms*—Learning from Demonstration, Task Planning, Manipulation Planning, Cross-Behavior Learning**

## I. Introduction

GENERALIZATION ability [1]–[4] is a fundamental challenge hindering the transition of robotic manipulation from laboratory settings to real-world applications, and even between different lab environments. Unlike humans, who develop object manipulation skills through innate training [5], [6], robots typically learn from scratch through a limited set of predefined tasks. The constrained environment and experience make it difficult to transfer robotic skills across different scenes [7]–[9] and objects [10], thereby restricting robots to operate effectively only within the domains they have been explicitly trained on.

To bridge the generalization gap, two primary approaches have emerged: learning from large-scale robotic datasets [11]–[13] and leveraging online human data [14]–[16]. While the success of scaling laws in large language models (LLMs) [17]–[19] raises hopes for similar advancements in robotics [20]–[24], fundamental differences exist. Unlike LLMs, which process sequences of words with a uniform probabilistic framework, robotic learning systems exhibit significant variations in observation and action spaces due to differences in system configurations, robot types, and end-effector designs.

The authors are with the Robot Manipulation Lab, School of Computer Science, University of Leeds,LS2 9JT Leeds, U.K (Corresponding e-mail: sclch@leeds.ac.uk)

M. Dogar was supported by the UK Engineering and Physical Sciences Research Council [EP/V052659/1]

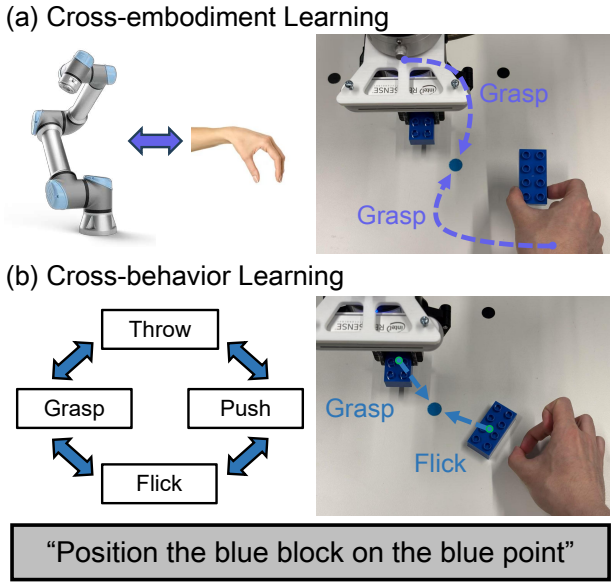


Fig. 1: (a) Cross-embodiment learning transfers skills across different embodiments with similar behaviors. (b) Cross-behavior learning can improve a policy even if the demonstrations use a different behavior to complete the same task; e.g., the human can use flicking behavior to move the block onto the blue point, while the learned robot policy may use a grasping behavior.

To expand the training dataset available for policy learning, cross-embodiment learning [25]–[28] extends learning across different robotic embodiments. Similar tasks or embodiments share underlying properties that can be exploited to extract high-level guidance for policy learning. Many studies attempt to pair manipulation trajectories across different embodiments to learn task representations [29], [30]. Temporal pairing generates similar trajectories, while spatial pairing links skills across tasks. However, pairing is effective only under high behavior similarity; otherwise, it requires expensive human annotation or additional model training to establish meaningful correspondences. To generalize further, latent space alignment [28], [31] constructs a shared latent space from one embodiment or task and reuses it across different domains. This method reduces the requirement for direct behavioral similarity, though adaptability remains proportional to behavior similarity.

When robotic manipulators vary [32] (e.g., transitioning from two-finger grippers to multi-finger dexterous hands) and manipulation behaviors shift [33] (e.g., from picking to flicking), manipulator-specific motion patterns become in-

creasingly diverse, while the task objective remains invariant (Fig. 1). This motivates task-centric representations that prioritize task completion over behavior imitation.

We formalize this perspective as *cross-behavior learning*, which aims to transfer task understanding across distinct manipulation behaviors arising from different embodiments or control strategies (e.g., a human hand throwing versus a robot gripper pushing), without requiring motion-level similarity.

To this end, we adopt an object-centric task representation based on object flow, which captures essential task information while discarding embodiment-specific motion and other task-irrelevant information, enabling efficient and generalizable policy learning across behaviors.

Our methodology consists of three stages: (i) Extracting object flow from cross-embodiment and cross-behavior demonstrations (Fig. 2). (ii) Building an object flow predictor to guide task execution. (iii) Training an action policy that utilizes the object flow to complete the task.

In the first stage, we collect a cross-embodiment and cross-behavior dataset. Objects' 2D shapes and positions are then extracted using an off-the-shelf tracking model, with the first frame of each demonstration manually labeled to ensure high-quality results. In the second stage, we train a transformer-based prediction model to infer the object flow in the environment. The predicted object flow serves as a structured guide for task execution, effectively breaking long-horizon tasks into shorter, more manageable steps. In the third stage, the action policy is designed to encode the object flow predicted in the previous stage, generating actions that manipulate objects in accordance with the predicted object flow sequence.

The contributions of this paper are as follows. We:

1) propose a novel pipeline for learning cross-behavior, as well as cross-embodiment manipulation from diverse embodiments. Our task-centric information extraction approach significantly enhances the utility of existing datasets, including action-free cross-embodiment and cross-behavior videos featuring various manipulators;
2) develop a robust and efficient object extraction pipeline, which generates reliable task-level representations from video data with minimal human effort;
3) demonstrate that incorporating object flow into guided policies improves policy performance by 38% on average. Moreover, cross-behavior learning enhances generalization to unseen scenarios with limited robot demonstrations;
4) investigate the role of object flow prediction horizon, human demonstration, and analyze the generalization capabilities of the proposed method.

## II. Related Work

### A. Cross-embodiment Learning

Learning from demonstrations across different embodiments has been widely studied in robotic manipulation. Most prior work on cross-embodiment imitation learning [34], [35] assumes strong behavioral similarity between the source and target domains, typically treating the human hand as a direct analogue of a robot gripper. Representative approaches align demonstrations via shared visual features [35], graph-based correspondence [36], [37], or explicitly designed embodiment-invariant representations [27]. While effective, these methods implicitly constrain the learning setting to behaviorally matched demonstrations, where the interaction patterns, contact modes, and temporal structure are largely preserved.

In practice, however, in human demonstrations and/or Internet-scale human videos, the same task can be achieved with substantial behavioral diversity, e.g., via pushing, flicking, throwing, or tool-assisted interactions that differ fundamentally from robotic manipulation primitives. Learning from such data without manual filtering or behavior alignment remains largely unexplored.

This gap motivates the cross-behavior learning setting considered in this work. Unlike prior cross-embodiment approaches that assume behavior-level correspondence, we explicitly address scenarios where the same task objective is achieved through heterogeneous behaviors across embodiments. Rather than translating human motions into robot actions, we discard manipulator-specific information altogether and focus on learning task-relevant object motions. This shift enables the use of diverse, weakly structured human demonstrations that are otherwise incompatible with standard cross-embodiment imitation learning.

### B. Task Representation for Generalizable Manipulation

The choice of task representation is critical for enabling generalization across tasks, embodiments, and behaviors. Pure behavior cloning [38] often suffers from long-horizon execution failures [39] and poor multi-task generalization [40], as it lacks explicit task structure and tends to entangle task objectives with embodiment-specific motion patterns. To address this limitation, prior work has introduced task-level guidance in the form of goals [41], language instructions [42], or intermediate abstractions such as skills and subgoals [43], [44]. These approaches highlight the importance of representations that explicitly encode task progress rather than raw actions.

Among existing representations, language-based task specifications [45] provide a compact and intuitive interface, but often function as static task identifiers when not grounded in execution dynamics [46]. Image- or mesh-based predictions [41], [47], [48] preserve rich spatial information but require high-dimensional representations, making them data-intensive and difficult to scale to multi-object settings. Furthermore, full image-based representations entangle task-relevant motion with background appearance and manipulator geometry.

To better capture task dynamics, recent work has explored flow-based representations derived from trajectory or object motion. Trajectory-guided representations, including RT-Trajectory [49] and HAMSTER [50], encode robot-centric trajectories or 2D sketches as intermediate supervision signals. Although successful for similar-embodiment transfer, these methods rely on robot data or explicit trajectory annotations, limiting their applicability to learning from heterogeneous human demonstrations with diverse interaction strategies.

General Flow [51] and Im2Flow2Act [52] represent tasks by predicting point-level object flow from 3D point clouds

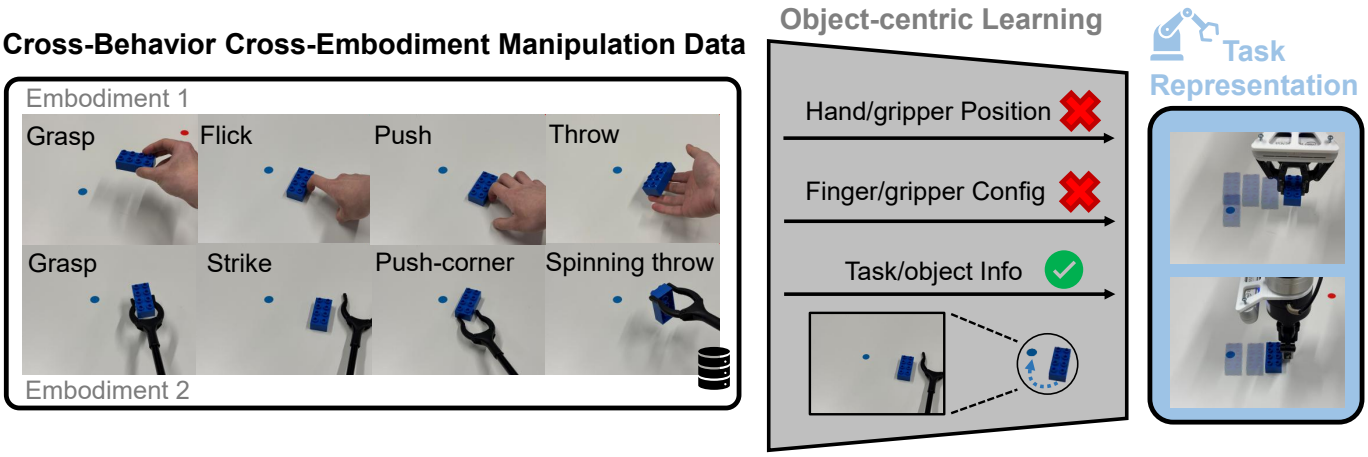


Fig. 2: An example of cross-behavior and cross-embodiment object-centric learning. **Left:** The figure shows how the same task (transporting a blue object to the blue sticker) can be completed using different manipulators/embodiments (top row showing a human hand, bottom row showing a black litter picker) and different manipulation behaviors (grasping, flicking, pushing, etc.). **Middle:** To transfer such cross-behavior and cross-embodiment manipulation data to the target robot domain, information of manipulator position and configuration are filtered out, and the task/object information is kept (object-centric learning). **Right:** The filtered task/object information works as the task representation and guides the robot to finish the task.

or 2D images, demonstrating improved transfer across embodiments. However, these methods typically rely on preselected object regions, explicit object grounding, or separate segmentation modules. In particular, Im2Flow2Act depends on external grounding and segmentation models at test time to generate bounding boxes and sample object points, making inference sensitive to perception errors under occlusion, clutter, or illumination changes. Moreover, extending point-based representations to multi-object manipulation requires additional mechanisms for point selection, association, and disambiguation, substantially increasing system complexity.

Other approaches, such as Track2Act [53] and ATM [54], leverage dense point tracking as task guidance for cross-embodiment imitation. While effective, point-level flow representations are inherently sensitive to tracking noise and point drift, especially when objects have many similar looking points (e.g., repeated textures or large uniform regions, which are extremely common on everyday objects), or occlusion.

In contrast to prior work, we introduce *segment-level object flow* as a task representation that captures task-relevant objects' motion while discarding manipulator-specific information. Compared to image-based representations, object flow significantly reduces irrelevant background information and removes appearance details that are hard to predict. Compared to point-level flow, segment-level flow avoids point sampling and enables tracking relevant objects altogether, provides richer information about object position and shape, and naturally extends to multi-object settings. Our object flow is predicted directly from RGB observations, using object masks only during offline dataset preparation; at training or inference time, no object localization or external perception modules are required.

### C. Object-centric Learning

Object-centric learning has proven to be effective in both task reasoning [55], [56] and modeling object dynamics [57], [58]. By incorporating object-relevant information, rather than relying solely on end-to-end training, this approach enhances the generalization capabilities of learned policies. However, most existing methods focus on a limited set of known object categories, often requiring detailed object textures or physical properties. In contrast, we propose a novel approach that leverages off-the-shelf object segmentation and tracking models to extract object information from the scene. Rather than relying on detailed object-specific attributes or explicitly estimating full physical dynamics, our method uses object flow as a lightweight representation that captures only the essential motion and spatial relations needed for task execution. This design reduces the required object information while still providing a scalable object-centric interface for policy learning.

## III. Problem Formulation

We define the control environment as a partially observable Markov decision process (POMDP), denoted by $M = \langle S, A, P, O \rangle$. Here, $S$ represents the state space, i.e., the information defining the current environment state, which includes observable and unobservable information. $A \subseteq \mathbb{R}^7$ defines the robot action space, which includes gripper width and relative 6D motion of the end-effector with respect to its current pose. $P(s_{t+1} \mid s_t, a_t)$ represents the state transition probability. $O$ is the observation space representing the information that can be observed. The observation $o \in O$ includes robot proprioception $x_t$, task instruction $l_t$ and RGB images $I_t$ from two cameras, one from the agent-view $I_t^a$ and one from the hand-view $I_t^h$. The robot's proprioception includes the gripper width and end-effector pose in the base frame. The orientations in observation and action are represented using

Euler angles. The initial state of the environment is represented as $s_0$, which follows the distribution $p(s_0)$.

The control problem is to model the action distribution based on the observation, giving the action policy $a_t = \pi(o_t)$. If the observation also includes past observations, and the action is an action chunk [59] predicting a sequence of actions, the action policy is formulated as $a_{t:t+h_a} = \pi(o_{t-h_o:t})$, where $h_a$ and $h_o$ are the action horizon and observation horizon respectively. During policy evaluation, we execute only the first $h_e$ steps of the predicted action chunk $a_{t:t+h_e}$. Building the action distribution based on past observations helps improve task understanding, and the action chunk smooths the actions learned from imperfect expert data.

### A. Cross-Embodiment Learning

Cross-embodiment learning aims to leverage behaviorally similar data collected from different embodiments to improve policy performance on target embodiments. It is widely used in scenarios where data collected from one manipulator (e.g., a human hand or a simulated robot) can benefit another manipulator with different kinematics or control constraints. The key idea is that, despite embodiment differences, demonstrations may share common behavioral structures—such as manipulator motion patterns, task goals, and temporal action dependencies—that can be transferred to improve the target policy.

Given a dataset $D_s$ with a source embodiment, and another dataset $D_g$ with a target embodiment, cross-embodiment learning exploits the shared behavioral patterns between the two to train a policy for the target embodiment. In order to make the source dataset $D_s$ useful for the target embodiment, often a conversion operation is performed, such as kinematic conversion and visual impainting between embodiments, generating an aligned dataset $D_s^g$. The learning objective is then to train a policy $\pi_g$ for the target embodiment by minimizing the loss over the *combined dataset* from both embodiments:

$$\arg\min_{\pi_g} \mathbb{E}_{(o,a)\sim D_s^g \cup D_g} \left[ \mathcal{L}\big(\pi_g(o_{t-h_o:t}), a_{t:t+h_a}\big) \right], \quad (1)$$

where $\mathcal{L}$ denotes the training loss function. This formulation enables the target policy to benefit from additional data provided by the source embodiment dataset, potentially improving generalization and reducing data collection on the target embodiment.

There are two major issues in current cross-embodiment learning approaches. First, when the source embodiment dataset lacks explicit action labels (which is often the case, for example with humans as the source embodiment), additional techniques are required to infer the corresponding actions. If the source embodiment is trackable (e.g., a human hand), motion tracking methods can be used to extract action labels [60]. Otherwise, one must synthesize a surrogate dataset by replacing the source embodiment with a simulated or proxy version of the target embodiment [61], [62] to enable imitation learning. Alternatively, behavior representation methods [34], [63] can be used to extract task-relevant features from the source embodiment, which can then be leveraged to guide the target policy.

Second, existing cross-embodiment learning methods typically assume that either the embodiments or the behaviors are closely aligned. This restricts the diversity of usable source data and limits the generalization capacity of the learned policies. Consider a source dataset $D_s = \{D_s^{grasp} \cup D_s^{flick} \cup D_s^{throw} \cup \dots\}$ consisting of *diverse behaviors* such as grasping, flicking, and throwing to perform the *same task* (Fig. 2-Left). If the target embodiment is a speed-limited robot, then throwing data (which demonstrates fast motion) would be useless. Similarly, flicking data may not be useful for a target robot that has a parallel two-finger gripper. In such cases, each behavioral subset must be either fine-tuned for the target embodiment or discarded entirely due to incompatibility.

### B. Cross-Behavior Learning

One way to bypass these two issues is to extract information about the *task*, instead of the embodiment or behavior. To enable effective cross-behavior policy learning across diverse embodiments, a key challenge lies in capturing task intent in a form that is transferable across embodiment-specific constraints, as shown in Fig. 2.

We propose an object-centric policy learning framework that leverages a shared representation $R$ to distill task intent from the source ($D_s$) or combined ($D_s \cup D_g$) dataset and guide the learning of the target policy. This approach avoids explicit dataset conversion and instead learns a high-level representation that encapsulates future object-centric changes, serving as a latent task descriptor across behaviors.

Specifically, instead of the typical cross-embodiment formulation (Eq. 1), we propose a target policy $\pi_g$ conditioned on both recent observations and a learned task representation:

$$\arg\min_{\pi_g} \mathbb{E}_{(o,a)\sim D_g} \left[ \mathcal{L}(\pi_g(o_{t-h_o:t}, R_t), a_{t:t+h_a}) \right], \quad (2)$$

where $R_t$ is the task representation at timestep $t$, which incorporates the object-centric task-related features, and can be learned from the source or combined datasets. $\mathcal{L}$ is the BC loss.

## IV. Method

In this work, as task representation $R_t$, we propose to use an *object flow*, $f_{t:t+h_f}$, where $h_f$ is the object flow horizon, and each $f_t$ represents objects' information at time $t$. This future-centric representation encodes task intent in a way that is invariant to specific embodiment dynamics, enabling policy $\pi_g$ to condition on a task-level goal signal rather than raw observations alone.

For objects' information $f_t$, we use 2D segmentations of all task-relevant objects (i.e., each $f_t$ is a binary mask), as segmentation provides a combination of position and shape information that are crucial in manipulation. The object flow $R_t$ constructed from these segmentations serves as a compact, embodiment-agnostic representation, focusing on how objects move and interact within the scene.

This object-centric representation does not exclude other types of task information. In practice, the policy $\pi_g$ (Eq. 2), in addition to $R_t$, also receives the raw multi-modal observations,

including the agent-view and hand-view RGB images, the end-effector pose, gripper state, and the task instruction. This design ensures that the policy can still access fine-grained interaction cues—such as contact geometry, object texture, and manipulator motion—while leveraging the object flow as a high-level prior to guide task reasoning. In this way, object flow complements rather than replaces interaction information, helping the policy disentangle manipulator-specific behaviors from general task dynamics and improving transferability across different embodiments.

During policy execution, the object flow is predicted from the agent-view images, $I^a$, in the raw observations using a high-level predictor $\pi_f$:

$$R_t = \hat{f}_{t:t+h_f} = \pi_f(I^a_{t-h_o:t}, l_t), \tag{3}$$

where $\hat{f}_{t:t+h_f}$ represents the predicted object flow. We construct the object flow using the agent-view observation only, since the task representation primarily depends on the global scene context provided by the agent-view camera rather than the local hand-view image [51], [52]. The agent-view captures holistic spatial information—including object layout, inter-object relations, and global motion cues—that are crucial. In contrast, the hand-view camera mainly reflects local ego-motion of the end-effector and often suffers from self-occlusion, which introduces unstable or noisy flow estimation. Therefore, the agent-view flow is adopted as a stable and task-informative representation for modeling object-centric motion.

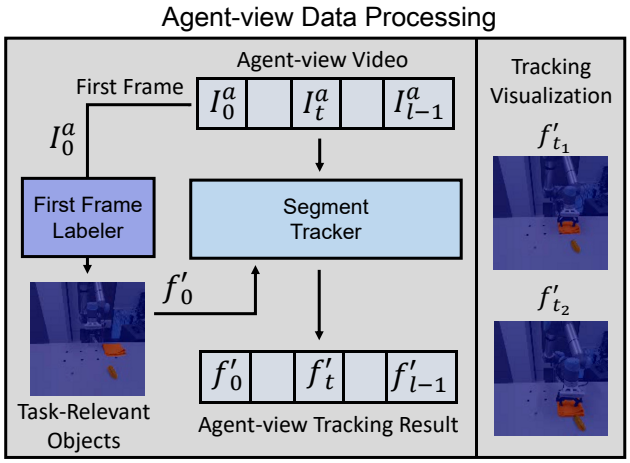


Fig. 3: Data process for agent-view images. The segment tracker takes in the agent-view video and the first frame mask, predicting the agent-view object flow.

Both the object flow predictor and the action policy are trained in a multi-task setting. As such, both models take a task instruction as input to distinguish between tasks. The task instruction $l_t$ is encoded via a pretrained BERT model [64].

Section IV-A details the extraction of object flow from the source (or combined) dataset using off-the-shelf models. Section IV-B then describes the training of the object flow predictor, $\pi_f$, based on the extracted object flows. Finally, Section IV-C presents the training procedure for the action policy $\pi_g$.

### A. Object Flow Extraction

To train $\pi_f$, we first generate an object flow dataset using the demonstrations in the source (or combined) dataset. This involves associating an object flow $f_{t:t+h_f}$ with each timestep of each demonstration in the source (or combined) dataset.

To extract such segmentations from the demonstration images, we initially experimented with SAM [65], Grounding DINO [66], and Grounded SAM [67] for object segmentation. However, these models struggled to segment objects accurately: the parameters of the models and text descriptions needed to be tuned carefully to match each object in each task. In some cases even fine-tuned parameters failed to detect target objects.

Instead, we formulate the problem as object-based segment tracking with object IDs, which provides more temporally robust segmentation. Rather than directly extracting unlabeled binary masks $f_t$, we first track labeled object segments. We denote the labeled mask as $f'_t$, which differs from $f_t$ only in that pixels belonging to the same object segment share a consistent object ID within each demonstration. The IDs are arbitrary and need not be consistent across demonstrations. Given a demonstration of length $L$, the tracking problem is defined as

$$f'_{0:L-1} = \mathcal{F}(I^a_{0:L-1}, f'_0), \tag{4}$$

where $I^a_{0:L-1}$ denotes the agent-view images and $f'_0$ is the seed segmentation mask with object IDs. The tracker $\mathcal{F}$ then propagates the labeled segments through the full demonstration.

The method requires a seed mask, which we obtain using a lightweight RITM-based labeling interface [68]. A human annotator marks task-relevant objects in the first frame with a few clicks, taking about one second per demonstration on average and producing high-quality masks for flow estimation. If the target object is not visible in the first frame, an intermediate visible frame is selected as the seed, and bidirectional propagation is used to obtain consistent masks over the full trajectory. We use the off-the-shelf tracker Cutie [69] as $\mathcal{F}$, as illustrated in Fig. 3.

Once we have $f'_{0:L-1}$ for all demonstrations, we remove the object IDs from the segments and assign a value of 1 instead, yielding the binary object flow $f_{0:L-1}$. Retaining distinct object IDs could resolve overlaps but would require a fixed object-ID list; thus, we discard them to keep the representation object-agnostic.

### B. Object Flow Prediction

We use a transformer model to predict the object flow from the observation, as shown in Fig. 4. The object flow prediction transformer $\pi_f$ is trained in a supervised manner:

$$\arg\min_{\pi_f} \mathcal{L}_{\text{flow}}(\pi_f(I^a_{t-h_o:t}, l_t), f_{t:t+h_f}) \tag{5}$$

over all timesteps $t$ for all demonstrations in the source embodiment dataset $D_s$ or the combined dataset $D_s \cup D_g$. Large-scale external video datasets can also be used to create a more general object flow predictor, even though we did not do that in this paper. Incorporating the target embodiment dataset

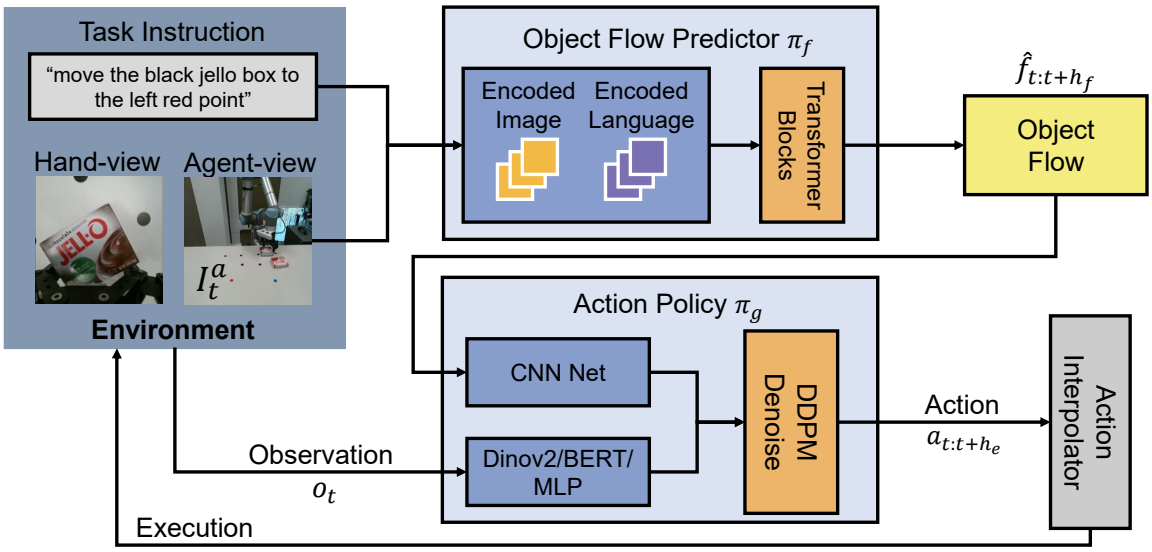


Fig. 4: An illustration of the system pipeline. The object flow predictor takes the task instruction and the agent-view image to predict the object flow in the next $h_f$ steps. The action policy takes all observations and the object flow from the predictor and outputs an action chunk. The action interpolator generates a tool center point (TCP) trajectory to the desired pose.

into training allows the predictor to fine-tune the object flow predictions for the specific target embodiment.

The loss function $\mathcal{L}_{\text{flow}}$ needs to be well designed, as we want the shape reconstruction and prediction to be sufficient to infer the objects' movement, and small objects are easy to be ignored. We use a combination of focal and dice loss as a basic loss function [70]. Focal loss addresses class imbalance by focusing on hard examples, while dice loss improves overlap quality in segmentation tasks:

$$\mathcal{L}_{\text{focal}}(p, y) = \frac{1}{4hw} \sum_{ij} \left[(p + y - (p \circ y))^2\right] \mathcal{L}_{\text{BCE}}(p, y), \quad (6)$$

$$\mathcal{L}_{\text{dice}}(p, y) = -\frac{2\sum_{ij}(p \circ y) + \epsilon}{\sum_{ij} p + \sum_{ij} y + \epsilon}, \quad (7)$$

where $p$ and $y$ are binary images of the same dimensions ($h$ and $w$), $(p \circ y)$ denotes element-wise (Hadamard) multiplication, $\sum_{ij}$ sums over all pixels, the square operation is applied element-wise, $\mathcal{L}_{\text{BCE}}$ is the binary cross entropy loss with logits, and $\epsilon$ is a small number (we use $10^{-6}$).

As we are predicting the object flow, objects that remain static for a long time can cause the network to overfit to a local optima that outputs a prediction which is the same as the current observation. To enforce motion prediction, we combine frame-by-frame loss with frame-combination loss:

$$\mathcal{L}_{\text{flow}} = \alpha \mathcal{L}_{\text{ff}} + \mathcal{L}_{\text{fc}}, \quad (8)$$

$$\mathcal{L}_{\text{ff}} = \sum_{k=t}^{t+h_f} [1 + \mathcal{L}_{\text{dice}}(\hat{f}_k, f_k) + \mathcal{L}_{\text{focal}}(\hat{f}_k, f_k)], \quad (9)$$

$$\mathcal{L}_{\text{fc}} = 1 + \mathcal{L}_{\text{dice}}(\hat{c}, c) + \mathcal{L}_{\text{focal}}(\hat{c}, c), \quad (10)$$

where $\alpha$ is a scaling factor (a value of 5 is used in this work), and $\hat{f}_k$ and $f_k$ denote the predicted and ground-truth object segmentations, respectively. $\hat{c}$ and $c$ denote the merged predictions and targets across $h_f$ frames, defined as:

$$\hat{c} = \bigvee_{k=t}^{t+h_f} \hat{f}_k, \quad c = \bigvee_{k=t}^{t+h_f} f_k, \quad (11)$$

where $\bigvee$ denotes elementwise logical OR.

The effect of each component is illustrated in Fig. 5. We plot the object flow above current agent-view image, with the left image being the ground truth merged $c$ (object flow extracted as in Section IV-A) and the right image being the merged prediction $\hat{c}$. The motion speed gap between different demonstrations is lessened using the frame-combination loss.

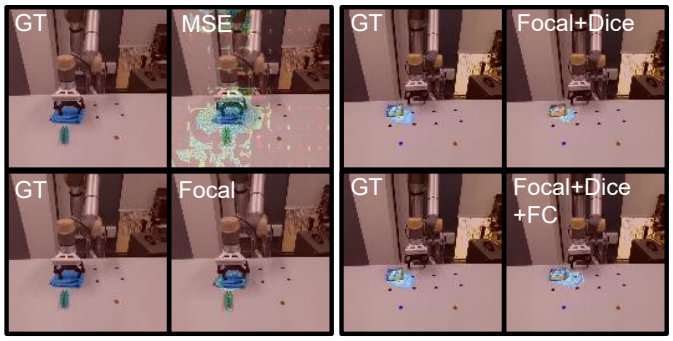


Fig. 5: Examples of the importance of different loss functions for object flow prediction. In each block, the left image is the ground truth (GT) and the right image shows the output of our object flow predictor. Each block shows a different combination of loss functions (MSE, Focal+Dice, Focal and Focal+Dice+FC). With MSE, the predictor fails to capture the object information. With only focal loss, the predictor could not capture the objects' shape well. With focal and dice, the flow has the trend of moving but on a small scale. With the frame-combination loss, the flow shows the future trajectory.

In practice, demonstrations collected from human teleoperation and robot control exhibit distinct execution speeds (e.g., the average duration of a human demonstration is 2.42 s, whereas that of a robot demonstration is 6.61 s). Under the frame-by-frame loss, each human demonstration advances faster through the sequence than its robot counterpart, resulting in temporal misalignment between the two. In contrast, the frame-combination loss jointly considers overlapping temporal windows, effectively increasing the correspondence between frames of different speeds and thereby reducing the inconsistency caused by temporal variations across demonstrations.

To improve generalization during training, we apply data augmentation techniques to the input images. Specifically, we randomize brightness, contrast, and saturation to simulate varying visual conditions, thereby enhancing the model's robustness to appearance changes. Additionally, we introduce random spatial translations to both the input images and corresponding object flow maps, which promotes invariance to object displacement and camera motion.

### C. Action Policy

The action policy $\pi_g$ is trained following Eq. 2, where $R_t$ denotes the object flow predicted by $\pi_f$. Formally, the action policy is defined as:

$$\pi_g(a_{t:t+h_a} \mid x_{t-h_o:t}, l_t, R_t, I^a_{t-h_o:t}, I^h_{t-h_o:t}) \tag{12}$$

where $a_{t:t+h_a}$ represents a sequence of delta end-effector poses and target gripper states over a prediction horizon $h_a$. The generated sequence thus corresponds to an action chunk, which are later interpolated into a smooth, continuous trajectory executable by the low-level controller.

**Network architecture** We employ a diffusion policy network [71] as the base policy, composed of a CNN-based denoiser and a DDPM scheduler. Unlike the original implementation, we adopt Dino-v2 [16] as the visual encoder. We use the smallest Dino-v2 variant and freeze its parameters during training for computational efficiency. Empirically, Dino-v2 yields superior performance compared to R3M—whether pretrained, trained from scratch, or frozen—suggesting stronger visual feature transferability.

The predicted object flow $R_t$ is processed through a CNN trained from scratch. The robot proprioceptive states are mapped to the latent space using a lightweight MLP.

**Training Objective** The policy $\pi_g$ is implemented as a conditional diffusion model, where the denoising network $\epsilon_\theta$ predicts the injected Gaussian noise, thereby implicitly defining the conditional action distribution. During training, the policy is optimized using the standard diffusion denoising objective:

$$\mathcal{L}_{\pi_g} = \mathbb{E}_{t,\epsilon}\left[\|\epsilon - \epsilon_\theta(a_{t:t+h_a} + \sigma_t \epsilon)\|_2^2\right], \tag{13}$$

where $\sigma_t$ is determined by the diffusion noise schedule. This objective trains the policy to predict the added noise, enabling iterative reverse diffusion to sample coherent action chunks conditioned on multimodal observations.

## V. Research Questions

This section formulates the research questions that guide our experimental evaluation. Our goal is to understand whether object flow can serve as a general, manipulator-agnostic task representation for learning and transferring manipulation skills across embodiments *and* behaviors.

Specifically, we aim to answer the following questions:

1) **Task representation effectiveness:** How does object flow compare with alternative task representations—such as image-based planning and point-based flow—in guiding manipulation policies?
2) **Cross-behavior generalization:** Can a single object-flow representation capture task semantics shared across distinct manipulation strategies, rather than overfitting to a specific behavior pattern?
3) **Cross-embodiment transfer:** Can object flow learned from human demonstrations be directly transferred to robot execution while preserving task-relevant information?
4) **Robustness and scalability:** Is object flow robust to visual distractions, distribution shifts, and clutter, and does it generalize to long-horizon, multi-object, or precision-critical manipulation tasks?

To address these questions systematically, we organize our experiments into three groups that correspond to the structure of the experimental section.

*Experiments comparing Task Representations:* In Section VI, we investigate the effectiveness of object flow as a task representation. Using five non-prehensile pushing tasks, we compare object flow against image-based planning and point-flow baselines under different dataset sizes and configurations (Section VI-C). We further analyze key factors that influence representation quality, including cross-behavior supervision, flow prediction accuracy, temporal horizon, and policy smoothness through controlled ablation studies (Section VI-D).

*Cross-Behavior Generalization:* In Section VII-A1, we explicitly evaluate whether object flow captures task-level structure beyond specific behaviors. We design demonstrations with aligned and distinct human behaviors and examine how these differences affect prediction and policy learning. We further study whether object flow learned solely from human demonstrations can be transferred to robots without additional task-specific adaptation (Section VII-A2).

*Generalization Tests:* Finally, in Section VII-B and Section VII-C, we assess the robustness and scalability of the learned object-flow representation. These tests include performance under visual distractions, deployment in cluttered scenes, and execution of complex tasks involving multiple objects, articulated mechanisms, and long-horizon planning. Together, these experiments evaluate whether object flow remains stable and interpretable when applied beyond the original training conditions.

## VI. Experiments Comparing Task Representations

This section focuses on comparing object flow with other task representations under identical training and evaluation

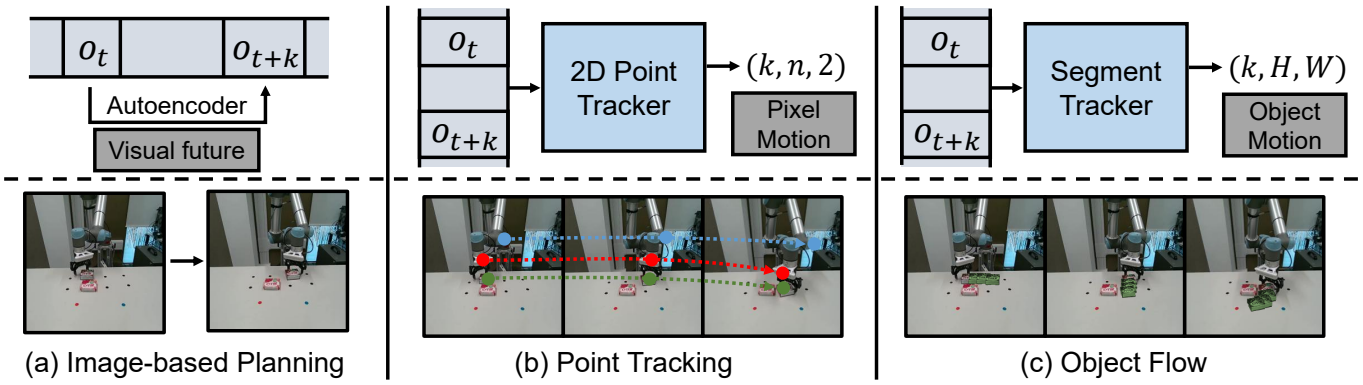


Fig. 6: Three representations of task flow. (a) Image-based: Predicts future visual observations in pixel space using an autoencoder. While expressive, this representation is high-dimensional, often containing noise, and is computationally expensive. (b) Point tracking: Models task flow by tracking the movement of 2D pixel points in an image over the horizon. (c) Object flow: Represents task flow as a sequence of object translations and shapes over the horizon.

protocols. Task descriptions are provided in Sec. VI-A and baselines are in Sec. VI-B. Results are presented in Sec. VI-C. Ablation studies and analysis are presented in Sec. VI-D.

### A. Tasks and Dataset

We designed five tasks to evaluate the effectiveness and transferability of the proposed pipeline facing different behaviors. Each task is defined by a natural task instruction, which is also used as input to the language-conditioned modules. The instructions are as follows:

1) "Push the black jello box to the left red point."
2) "Push the black jello box past the red jello box to the left red point."
3) "Push the black jello box to the right blue point."
4) "Push the black jello box past the red jello box to the right blue point."
5) "Wipe the green pea off the table with the yellow towel."

The objects used in the experiments are shown in Fig. 7. The task trajectory design for training and testing is shown in Fig. 8. Tasks 1 and 3 are easier to learn, but distinguishing between them is crucial, as different goals require the robot to approach the corresponding object corner and move in the correct direction. The training samples cover all 8 black points, and the goal is either the red or blue point.

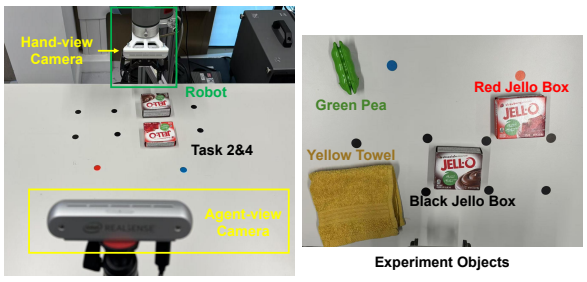


Fig. 7: The left image shows the general experiment setup. There are two cameras, an agent view camera statically observing the scene and a hand-view camera attached to the robot (UR5). The current layout of objects is an example starting position for tasks 2&4. The right image shows the different experimental objects used in our real-world experiments.

Tasks 2 and 4 are more challenging, as the policy must infer the relationship between the two jello boxes. In data collection, the red jello box (acting as an obstacle) is placed in the second row, while the black jello box starts in the first row. The obstacle's position is randomized, and it may either block the path or not. During testing, we consistently placed the red jello box in the path of the black jello box.

Task 5 differs from the first four in the action and language feature domain. During data collection for task 5, the yellow towel is randomly placed on the black points in the first row, while the pea is placed at either of the two goal points. In testing, both the towel and pea positions are randomized.

We collected a total of 200 human demonstrations ($D_s$) and 100 robot demonstrations ($D_g$) for each of the manipulation tasks. Each demonstration consists of time-synchronized multi-modal observations, including agent-view and hand-view RGB images at 30 Hz, and robot proprioceptive states.

Half of the human demonstrations are performed with a bare hand, while the other half use a black litter picker to emulate a robot gripper (Fig. 2). Human demonstrations are intentionally collected to provide approximately balanced coverage of diverse behaviors (including grasping, pushing, rotating, and throwing) so that the policy is exposed to a broad range of motion primitives. Robot demonstrations are collected via a 6-DoF SpaceMouse controller that teleoperates the robot in Cartesian space.

To reduce the effect of static frames at the beginning and end of demonstrations—caused by delays in starting or stopping teleoperation—we remove these segments. Pauses during execution are kept, as they reflect cautious human teleoperator behavior.

### B. Baselines and Implementations

The proposed framework comprises two design choices that are evaluated through comparative experiments. The first is the high-level task representation, $R_t$, instantiated in this

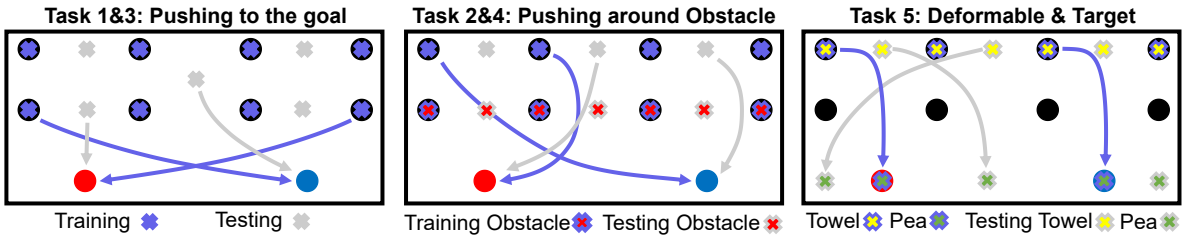


Fig. 8: The five tasks designed for the real-world experiments. Eight black circle stickers are placed on the table to mark the initial state, and one blue and one red circle mark the goal states. Tasks 1&3 involve pushing the black jello box to the goal, tasks 2&4 introduce an obstacle, and task 5 involves wiping the pea off the table with a towel.

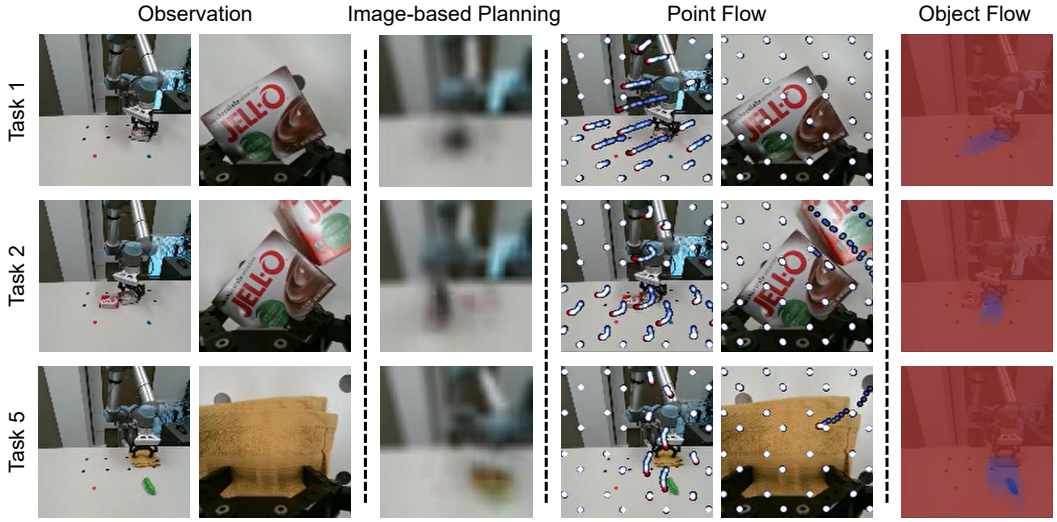


Fig. 9: Visualization of subgoal, point flow, and object flow predictions for tasks 1, 2, and 5. The leftmost images represent the observations, including both agent-view and hand-view perspectives. The image-based planning exhibits significant uncertainty around the robot and experimental regions. The point flow captures the robot's movement trajectory but misjudges the table and fails hand view. meaningful movements in the hand-view. The object flow focuses on experimental objects, providing a more accurate representation of future movements.

work as an object-flow representation. The second is the low-level action policy, $\pi_g$, which in our implementation adopts a modified diffusion-based policy network (see Sec. IV-C).

**High-level Task Representation** For the high-level task representation, $R_t$, we compare our proposed object flow to commonly used image-based planning and point flow methods (shown in Fig. 6 and 9), as well as pure behavior cloning, as baselines. These are:

- **BC**: Behavior cloning without task guidance ($R_t$), serving as a no-task-representation baseline.
- **IMAGE**: Image-based planning, which predicts future visual observations as task guidance. We adopt a transformer-based architecture for implementation.
- **ATM [54]**: One of our point-flow baselines, which predicts grid-based point flow to infer task-relevant motion features. ATM serves as a point-flow baseline representing structured visual motion reasoning.
- **Im2Flow2Act [52]**: An object-centric point-flow method, which constructs object-level point flow to transfer motion features across different embodiments.
- **OF**: Object flow (ours), which models the motion of task-relevant objects as a manipulator-agnostic task representation to guide the action policy.

We visualize the learned task representations for some tasks in Fig. 9 as comparison. All methods use an observation horizon of $h_o = 3$ and a prediction horizon of $h_f = 16$.

**Low-level Action Policy** For the low-level action policy, we refer to our modified diffusion policy as *DP*. DP was trained with an action horizon of $h_a = 16$. During execution, we executed $h_e = 4$ of these actions. The baseline we compare against is the open-source ATM's transformer-based action policy architecture [54].

The above high-level task representation methods and low-level action policy methods can be combined in different ways to give us different overall methods (Table I). For example, OF-DP employs object flow as the task representation generator and diffusion policy as the action policy. The high-level task representation can be trained on the same or a different dataset from that used for the action policy (Eq. 2). Therefore, when naming the different baselines we used, after each method's name, we also indicate the number of Robot and Human demonstrations used to train that method. For example, OF-100Robot-200Human-DP-100Robot indicates a method where the object flow predictor is trained with 100

TABLE I: Success rates for policies under different task representations.

| Task Representation | Action Policy | Evaluation setting | Task-1 | Task-2 | Task-3 | Task-4 | Task-5 | Total |
|---|---|---|---|---|---|---|---|---|
| **No Task Representation** | | | | | | | | |
| BC-0 | DP-20Robot | seen scenarios | 7/10 | 1/10 | 10/10 | 3/10 | 3/10 | 24/50 |
| | | unseen scenarios | 5/10 | 1/10 | 8/10 | 1/10 | 1/10 | 16/50 |
| BC-0 | DP-100Robot | seen scenarios | 8/10 | 6/10 | 7/10 | 7/10 | 8/10 | 36/50 |
| | | unseen scenarios | 6/10 | 5/10 | 7/10 | 4/10 | 2/10 | 24/50 |
| **Image Prediction as Task Representation** | | | | | | | | |
| IMAGE-100Robot-2views | DP-100Robot | seen scenarios | 10/10 | 5/10 | 9/10 | 4/10 | 10/10 | 38/50 |
| | | unseen scenarios | 6/10 | 4/10 | 6/10 | 2/10 | 3/10 | 21/50 |
| **Point Flow as Task Representation** | | | | | | | | |
| ATM-100Robot-2views | ATM-100Robot | seen scenarios | 1/10 | 1/10 | 1/10 | 1/10 | 1/10 | 5/50 |
| | | unseen scenarios | 0/10 | 0/10 | 0/10 | 0/10 | 0/10 | 0/50 |
| ATM-100Robot | DP-100Robot | seen scenarios | 10/10 | 8/10 | 10/10 | 9/10 | 9/10 | 46/50 |
| | | unseen scenarios | 6/10 | 5/10 | 10/10 | 5/10 | 1/10 | 27/50 |
| ATM-100Robot-2views | DP-100Robot | seen scenarios | 9/10 | 9/10 | 8/10 | 5/10 | 10/10 | 41/50 |
| | | unseen scenarios | 10/10 | 4/10 | 10/10 | 4/10 | 3/10 | 31/50 |
| ATM-100Robot-200Human | DP-100Robot | seen scenarios | 10/10 | 9/10 | 10/10 | 7/10 | 9/10 | 45/50 |
| | | unseen scenarios | 6/10 | 4/10 | 7/10 | 3/10 | 1/10 | 21/50 |
| Im2Flow2Act-100Robot-200Human | DP-100Robot | seen scenarios | 6/10 | 2/10 | 1/10 | 1/10 | 6/10 | 16/50 |
| | | unseen scenarios | 3/10 | 3/10 | 1/10 | 1/10 | 6/10 | 14/50 |
| **Object Flow as Task Representation** | | | | | | | | |
| OF-20Robot | DP-20Robot | seen scenarios | 10/10 | 10/10 | 10/10 | 7/10 | 9/10 | 46/50 |
| | | unseen scenarios | 7/10 | 8/10 | 6/10 | 6/10 | 3/10 | 30/50 |
| OF-100Robot | DP-100Robot | seen scenarios | 9/10 | 10/10 | 10/10 | 10/10 | 10/10 | 49/50 |
| | | unseen scenarios | 6/10 | 7/10 | 6/10 | 7/10 | 7/10 | 33/50 |
| OF-20Robot-40Human | DP-20Robot | seen scenarios | 10/10 | 9/10 | 10/10 | 9/10 | 10/10 | 48/50 |
| | | unseen scenarios | 9/10 | 9/10 | 8/10 | 7/10 | 6/10 | 39/50 |
| OF-100Robot-200Human | DP-100Robot | seen scenarios | 10/10 | 10/10 | 10/10 | 10/10 | 10/10 | **50/50** |
| | | unseen scenarios | 7/10 | 8/10 | 8/10 | 9/10 | 8/10 | **40/50** |

robot and 200 human demonstrations and the action policy is trained with 100 robot demonstrations. Since the hand-view image is not available in human demonstrations, incorporating human data for task-representation training sacrifices the task representation in the hand-view. When using only Robot data to train the task representations we could use both views, shown with a label of *2views* at the end of the method names.

### C. Results

As shown in Table I, we evaluated 12 different methods (where each method is a combination of a Task Representation and a low-level Action Policy). We evaluated each method over 100 rollouts, totaling to 1200 real robot evaluations. Out of the 100 for each method, 50 are tested from states included in the *seen scenarios*, and the remaining 50 are used to evaluate generalization in previously *unseen scenarios*. In each case, the 50 runs are divided over the five tasks (i.e., 10 runs per task).

With 20 robot demonstrations, BC-0-DP-20Robot succeeds on Tasks 1 and 3 in both in-distribution and out-of-distribution settings, but fails on Tasks 2, 4, and 5. Increasing the number of demonstrations to 100 improves performance on the more demanding Tasks 2, 4, and 5. Despite this improvement, BC-0-DP-100Robot still shows limited generalization to unseen configurations, particularly on Task 5.

The image-based planning baseline (IMAGE) achieves performance comparable to BC-0-DP-100Robot, which is trained with the same number of robot demonstrations. The predicted image-based subgoals exhibit high uncertainty, particularly in dynamic regions, limiting their effectiveness as guidance for the action policy.

The ATM model [54], trained with 100 robot demonstrations, performs poorly across both in-distribution and out-of-distribution scenarios. To verify the correctness of our implementation, we additionally evaluated ATM in simulation under the same task in the original work, obtaining results consistent with those reported.

Replacing the transformer-based policy in ATM with the proposed diffusion-based policy leads to consistent performance improvements. In particular, ATM-100Robot-2views-DP-100Robot outperforms BC-0-DP-100Robot, highlighting the benefit of incorporating task representation. Removing the hand-view input causes ATM-100Robot-DP-100Robot to overfit to in-distribution settings, reducing generalization. More-

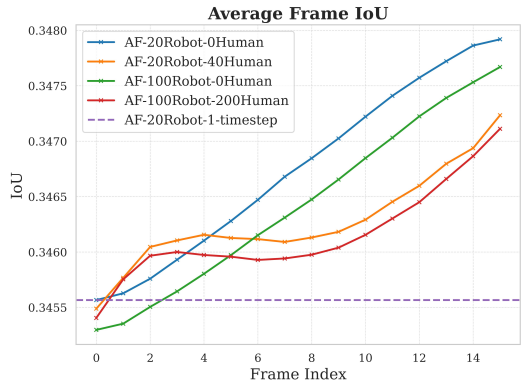


Fig. 10: IoU accuracy of object flow prediction under different settings.

over, incorporating human demonstrations with diverse manipulation strategies (ATM-100Robot-200Human-DP-100Robot) degrades out-of-distribution performance, indicating sensitivity to distribution mismatch.

Im2Flow2Act [52], which relies on Grounding-DINO [16] for object localization and point sampling, performs reasonably when the target object is large and visually distinct from the background (e.g., Task 5). However, the method is not end-to-end and is highly sensitive to the quality of external segmentation. In scenarios involving illumination changes, clutter, occlusion, or small objects (Tasks 1–4), segmentation failures frequently lead to inaccurate point-flow estimation and degraded policy performance, in some cases falling below behavior cloning. In addition, the method requires manual tuning of object-specific textual prompts and confidence thresholds, and is limited to single-object settings.

Integrating object flow prediction leads to substantial performance gains. With only 20 robot demonstrations, OF-20Robot-DP-20Robot outperforms BC-0-DP-100Robot by 91% in-domain and 87% out-of-domain, despite using one-fifth of the training data. Moreover, OF-20Robot-DP-20Robot achieves performance comparable to OF-100Robot-DP-100Robot, indicating that object flow significantly reduces the dependence on large-scale demonstrations.

Augmenting training with out-of-distribution human demonstrations further improves robustness. With 40 additional human demonstrations, OF-20Robot-40Human-DP-20Robot improves performance by 100% in-domain and 143% out-of-domain. When trained with 100 robot and 200 human demonstrations, the policy achieves perfect performance on training tasks and strong generalization to unseen configurations.

Overall, these results demonstrate that object flow substantially enhances policy robustness and data efficiency, enabling effective learning from limited and cross-behavior demonstrations without requiring strict temporal or spatial alignment.

TABLE II: Dice and focal loss of flow prediction under different settings.

| **Type** | $\mathcal{L}_{\text{ff}}$-Dice | $\mathcal{L}_{\text{ff}}$-Focal | $\mathcal{L}_{\text{fc}}$-Dice | $\mathcal{L}_{\text{fc}}$-Focal |
|---|---|---|---|---|
| OF-20Robot-0Human | 0.515 | 0.0446 | 0.528 | 0.0646 |
| OF-20Robot-40Human | 0.514 | 0.0447 | 0.531 | 0.1579 |
| OF-100Robot-0Human | 0.515 | 0.0446 | 0.527 | 0.0697 |
| OF-100Robot-200Human | 0.514 | 0.0447 | 0.531 | 0.1623 |
| OF-20Robot-1-timestep | 0.514 | 0.0447 | 0.516 | 0.0645 |
| OF-20Robot-4-timestep | 0.514 | 0.0447 | 0.517 | 0.0631 |
| OF-20Robot-8-timestep | 0.514 | 0.0446 | 0.520 | 0.0635 |

TABLE III: Action jerkiness of low-level policies.

| **Type** | **Action Jerkiness** ($J$) |
|---|---|
| Dataset | 0.2712 |
| OF-20Robot-DP-20Robot | 0.0715 |
| OF-20Robot-40Human-DP-20Robot | 0.0712 |
| OF-100Robot-DP-100Robot | 0.0736 |
| OF-100Robot-200Human-DP-100Robot | 0.0734 |

### *D. Ablation Studies and Analysis*

In this section, we provide further analysis and ablation studies of the proposed method.

*1) Object Flow Prediction Accuracy:* We evaluate object flow prediction using dice and focal losses across all object flow–based policies (Table II), with results reported for 100 robot demonstrations. The frame-wise Dice and focal losses remain consistent across different prediction settings, indicating that as few as 20 demonstrations suffice for learning object flow within the domain.

When human demonstrations are incorporated, the frame-by-combination focal loss increases, reflecting greater temporal variability in object motion. While such demonstrations improve downstream policy generalization, they introduce higher uncertainty in long-horizon flow prediction due to non-uniform motion patterns.

To further analyze this effect, we measure the average frame IoU for the first four object flow–based policies (Fig. 10). Incorporating human demonstrations improves IoU in early timesteps, but degrades accuracy at later stages. Across all policies, long-horizon predictions are more accurate than short-horizon ones, which may be influenced by mask quality from Cutie [69] and occlusion effects.

*2) Policy Action Jerkiness:* To examine whether incorporating human demonstrations affects action stability, we evaluate action jerkiness for object flow–guided policies across all five tasks, with averaged results reported in Table III.

Action jerkiness is defined as the average squared norm of the second-order finite difference of the action sequence:

$$J = \frac{1}{T-2} \sum_{t=2}^{T-1} \left\| a_{t+1} - 2a_t + a_{t-1} \right\|^2, \tag{14}$$

where $a_t$ denotes the executed action at timestep $t$ and $T$ is the rollout length. Higher values indicate less smooth action trajectories.

Across all settings, object flow–guided policies exhibit substantially lower jerkiness than the demonstration data. As the number of robot demonstrations increases, action jerkiness shows a mild upward trend, whereas augmenting training with human demonstrations slightly reduces jerkiness. These results indicate that incorporating human demonstrations does not

TABLE IV: Success rates of the ablation study on object flow prediction horizon, the base model is OF-20Robot-DP-20Robot.

| **Prediction Horizon** | **Evaluation setting** | Task-1 | Task-2 | Task-3 | Task-4 | Task-5 | Total |
|---|---|---|---|---|---|---|---|
| 1-timestep | seen scenarios | 8/10 | 5/10 | 10/10 | 4/10 | 4/10 | 31/50 |
| | unseen scenarios | 4/10 | 1/10 | 6/10 | 3/10 | 3/10 | 17/50 |
| 4-timestep | seen scenarios | 10/10 | 5/10 | 10/10 | 4/10 | 8/10 | 37/50 |
| | unseen scenarios | 5/10 | 3/10 | 5/10 | 5/10 | 4/10 | 22/50 |
| 8-timestep | seen scenarios | 10/10 | 8/10 | 9/10 | 6/10 | 8/10 | 41/50 |
| | unseen scenarios | 7/10 | 5/10 | 6/10 | 5/10 | 3/10 | 26/50 |
| 16-timestep | seen scenarios | 10/10 | 10/10 | 10/10 | 7/10 | 9/10 | **46/50** |
| | unseen scenarios | 7/10 | 8/10 | 6/10 | 6/10 | 3/10 | **30/50** |
| 32-timestep | seen scenarios | 9/10 | 6/10 | 9/10 | 6/10 | 10/10 | 40/50 |
| | unseen scenarios | 7/10 | 4/10 | 3/10 | 5/10 | 4/10 | 22/50 |

TABLE V: Success rates of policies that learns task representation from pure human demonstrations.

| **Task Representation** | **Action Policy** | **Evaluation setting** | Task-1 | Task-2 | Task-3 | Task-4 | Task-5 | Total |
|---|---|---|---|---|---|---|---|---|
| BC-0 | DP-20Robot | seen scenarios | 7/10 | 1/10 | 10/10 | 3/10 | 3/10 | 26/50 |
| | | unseen scenarios | 5/10 | 1/10 | 8/10 | 1/10 | 1/10 | 16/50 |
| OFH-40Human-Aligned | DP-20Robot | seen scenarios | 9/10 | 6/10 | 10/10 | 5/10 | 7/10 | **37/50** |
| | | unseen scenarios | 3/10 | 6/10 | 4/10 | 5/10 | 7/10 | **25/50** |
| OFH-40Human-Distinct-1 | DP-20Robot | seen scenarios | 9/10 | 6/10 | 10/10 | 3/10 | 3/10 | 31/50 |
| | | unseen scenarios | 3/10 | 6/10 | 6/10 | 2/10 | 3/10 | 20/50 |
| OFH-40Human-Distinct-2 | DP-20Robot | seen scenarios | 8/10 | 7/10 | 9/10 | 4/10 | 8/10 | 32/50 |
| | | unseen scenarios | 4/10 | 3/10 | 5/10 | 5/10 | 6/10 | 23/50 |

introduce instability and can contribute to smoother action trajectories. Notably, learned policies are consistently smoother than teleoperated robot demonstrations, suggesting that policy learning attenuates high-frequency control noise present in raw demonstrations.

*3) Flow Prediction Horizon:* We ablate the object flow prediction horizon using OF-20Robot-DP-20Robot without human demonstrations, varying the horizon over $\{1, 4, 8, 16\}$ timesteps. A one-step horizon provides detection-like object localization, whereas longer horizons provide multi-step task guidance. The results are summarized in Table IV.

Longer horizons consistently improve policy performance and execution stability in both in-distribution and out-of-distribution settings, indicating that multi-step object flow is more robust to observation noise and uncertainty. Increasing the horizon does not noticeably affect the inference time of either the object flow model or the action policy.

## VII. Generalization Tests

### A. Cross-Behavior Ablation Study

To further assess the cross-behavior learning capability of the proposed pipeline, we conduct ablation studies using human demonstrations collected under three distinct manipulation strategies: **Aligned**, **Distinct-1**, and **Distinct-2**. These strategies are designed to vary the degree of behavioral alignment between human demonstrations and robot execution, while preserving the underlying task intent.

- **Aligned:** Human demonstrations that closely mirror the robot's manipulation behavior, serving as behaviorally aligned references.
- **Distinct-1:** Human demonstrations that achieve the same task objective using an alternative but consistent strategy. Examples include intermittent 2D sliding of a jello box instead of continuous pushing, or wiping by grasping the top center of a towel to allow stable vertical contact and wider motion range.
- **Distinct-2:** Human demonstrations that employ a substantially different manipulation strategy, such as transporting objects via pick-and-place or wiping by grasping the edge or corner of a towel.

Each task in every group comprised 40 newly collected human demonstrations, distinct from previously mixed behaviors. The ablation studies were structured as follows:

1) **Learning from Distinct Behaviors:** Followed the general comparison outlined in Table I.
2) **Human to Robot Transfer:** The object flow predictor is trained with only the human data, without seeing robot motion and how robot executes the task beforehand.

*1) Learning from Distinct Behaviors:* We evaluate cross-behavior learning primarily on Task 5 under unseen configurations, as OF-20Robot-DP-20Robot already achieves near-saturated performance on Tasks 1–4 without human demonstrations (Table I). Results for Tasks 1–4 are therefore omitted for brevity.

In unseen scenarios of Task 5, OF-20Robot-DP-20Robot

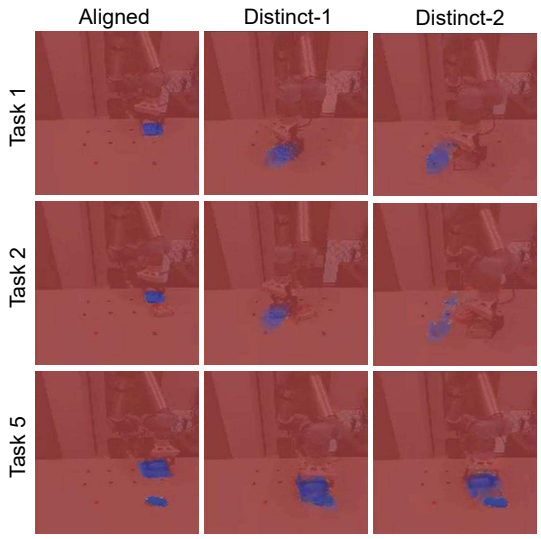


Fig. 11: Visualization of object flow from separate human demonstrated manipulation behaviors. The motion of objects in Aligned is slow as the robot; Distinct-1 shows longer object flow as motion is faster, while Distinct-2 presents 3D transferring and obstacle-crossing flow.

achieves 3/10 successes (Table I), indicating limited generalization when the target object is placed outside the training distribution. Incorporating human demonstrations substantially improves performance: Aligned and Distinct-1 strategies both achieve 7/10 successes, while Distinct-2 reaches 5/10. Despite the large behavioral differences across these strategies, all human-augmented settings yield clear gains over the robot-only baseline, demonstrating the ability of the proposed framework to learn from diverse manipulation behaviors.

*2) Human to Robot Transfer:* We evaluate human-to-robot transfer by training object flow predictors exclusively on human demonstrations, without any exposure to robot data (rows 2–4 in Table V). In this setting, the high-level predictors (*OFH* variants) encode task-level object motion without access to robot-specific motion or end-effector trajectories. This setup allows us to assess whether object flow can provide meaningful guidance for robot policy learning across embodiments. Representative object flow visualizations are shown in Fig. 11.

Across all tasks, human-trained *OFH* models consistently outperform the baseline without object flow (BC-0, row 1), demonstrating that the learned object flow captures transferable task structure. Performance varies across manipulation strategies: predictors trained on behaviorally aligned or moderately distinct human demonstrations yield more stable improvements, while predictors trained on highly distinct strategies exhibit greater sensitivity to task variations. For example, OFH-40Human-Distinct-1-DP-20Robot improves performance on Task 2 but degrades on the structurally similar Task 4, highlighting limitations of purely human-trained predictors under distribution shift.

When the object flow predictor is trained with both human and robot demonstrations (see Table. I), performance improves further, indicating better adaptation to robot-specific execution patterns. Importantly, even in this setting, human demonstrations continue to contribute positively, as object flow preserves task-level motion—such as object trajectories and goal states. Overall, these results show that object flow enables effective human-to-robot transfer, with robustness increasing as limited robot data is incorporated.

### *B. Performance under Distraction*

To evaluate the robustness of the learned policy under visual distractions, we conduct an additional test on **Task 1** and **Task 3** by introducing various irrelevant objects into the scene at test time. These distractors are placed such that they do not physically obstruct the intended manipulation trajectory, but increase visual clutter. As illustrated in Fig. 12, the distractors include objects of different shapes, colors, and sizes, as well as a visually similar variant of the target jello box.

This experiment is designed to assess whether the policy can (i) maintain task performance in the presence of unseen, irrelevant objects, and (ii) correctly identify and act on the original task-relevant object without being confused by visually similar distractors. For each task, we evaluate the policy on 20 randomized distraction configurations.

Across all trials, the policy maintains similar success rate (90%) as in the original (92.5%), distraction-free setting. Notably, even when the original jello box is placed at an out-of-distribution location and an additional, visually similar jello box is positioned at an in-distribution location, the policy consistently identifies the original target and successfully pushes it to the goal. This behavior demonstrates that the learned object-flow-based task representation is robust to visual distractions and spatial biases.

### *C. Policy on Diverse and Complex Tasks*

To further evaluate the generalization capability of the proposed approach, we design three additional manipulation tasks that go beyond the original benchmark settings. The tasks are shown in Fig. 13 These tasks collectively cover a wide range of real-world challenges, including high-precision manipulation

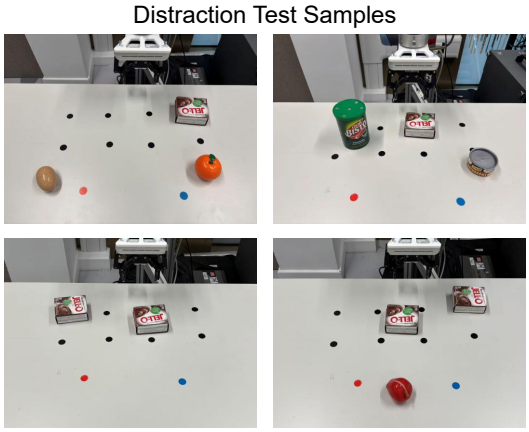


Fig. 12: Distraction test samples. Various irrelevant objects are introduced into the scene during testing, including a visually similar but dimensionally different jello box.

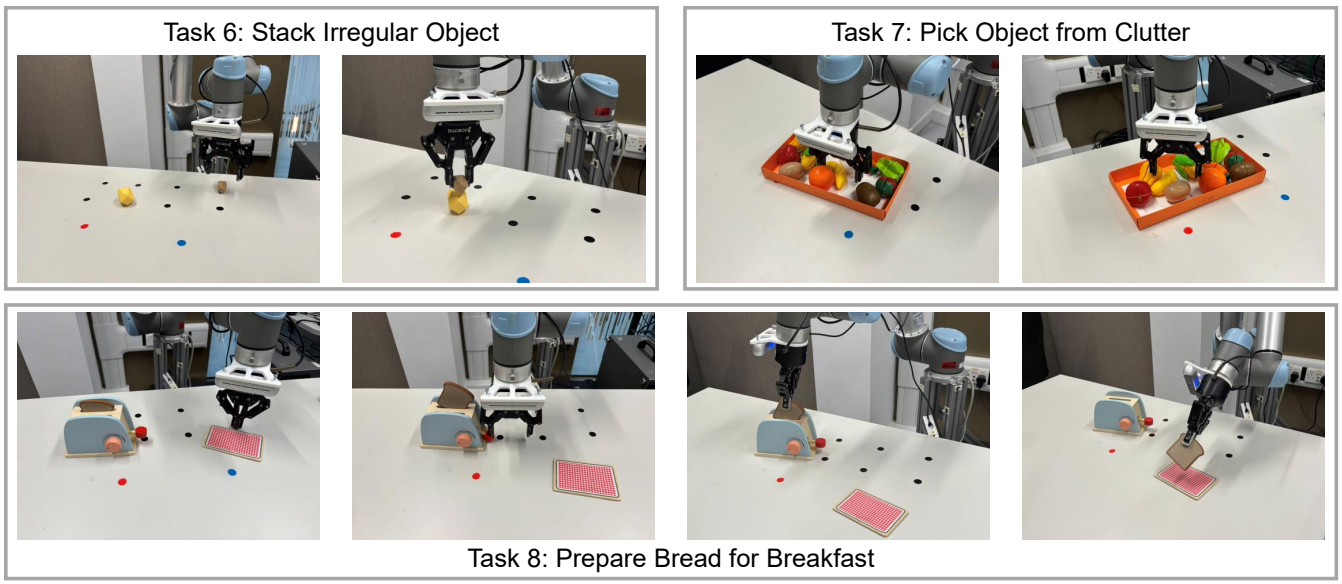


Fig. 13: Illustrations of task 6-8. **Task 6 (Stack Irregular Object):** The robot stacks an irregularly shaped object, requiring precise pose control and accurate contact handling. **Task 7 (Pick Object from Clutter):** The robot selects and picks a specific target object (banana) from a cluttered container, demonstrating robustness to occlusion and distractors. **Task 8 (Prepare Bread for Breakfast):** The robot pushes a plate to a target location, ejects toasted bread from a toaster, retrieves the bread, and places it onto the plate, illustrating multi-stage task execution and generalization to long-horizon object interactions.

TABLE VI: Success rates of task-stage execution under increasingly challenging generalization settings.

| **Policy** | **Evaluation setting** | **Task-6** | | **Task-7** | | **Task-8** | | | | **Total** |
|---|---|---|---|---|---|---|---|---|---|---|
| | | pick | stack | approach | pick | plate | button | take bread | place bread | all |
| BC-0 | seen scenarios | 2/10 | 1/10 | 0/10 | 0/10 | 2/10 | 1/10 | 0/10 | 0/10 | 6/80 |
| | unseen scenarios | 0/10 | 0/10 | 0/10 | 0/10 | 1/10 | 0/10 | 0/10 | 0/10 | 1/80 |
| ATM-100Robot | seen scenarios | 1/10 | 0/10 | 1/10 | 0/10 | 1/10 | 0/10 | 0/10 | 0/10 | 3/80 |
| | unseen scenarios | 0/10 | 0/10 | 0/10 | 0/10 | 0/10 | 0/10 | 0/10 | 0/10 | 0/80 |
| OF-100Robot | seen scenarios | 6/10 | 5/10 | 4/10 | 3/10 | 4/10 | 3/10 | 1/10 | 0/10 | 26/80 |
| | unseen scenarios | 1/10 | 0/10 | 3/10 | 2/10 | 2/10 | 1/10 | 0/10 | 0/10 | 9/80 |
| OF-100Robot-200Human | seen scenarios | 7/10 | 6/10 | 6/10 | 4/10 | 5/10 | 3/10 | 2/10 | 2/10 | **35/80** |
| | unseen scenarios | 6/10 | 4/10 | 5/10 | 3/10 | 4/10 | 5/10 | 1/10 | 1/10 | **29/80** |

(Task 6), cluttered scenes with distractors (Task 7), and long-horizon, multi-stage execution involving multiple objects and articulated mechanisms (Task 8). Compared to the original tasks, these settings require accurate contact handling, robust object selection, and temporally coordinated control across multiple subtasks.

In the stacking task (Task 6), training data are collected by placing the wooden blocks on predefined marked locations (black dots) on the table, referred to as *seen scenarios*. During evaluation, blocks are instead initialized at random continuous positions that do not coincide with these marked locations, forming the unseen setting. For the pick-from-clutter task (Task 7), the distinction between training and unseen configurations is defined by the position of the container. The container is placed at fixed locations during training and moved to novel locations during evaluation, while the positions of all objects inside the container are randomized in both training and testing. For the breakfast preparation task (Task 8), training and unseen settings are determined by the spatial arrangement of the plate and the bread machine. Their positions vary across training and evaluation.

For all policies, the low-level action policy is instantiated as the same diffusion policy and trained using 100 robot demonstrations. The key difference across methods lies in how the task representation is obtained and provided to the policy. Specifically, **BC-0** corresponds to behavior cloning without any task-level guidance, **ATM-100Robot** uses grid-based point-flow representations, and **OF** variants condition the policy on object-flow-based task representations. The OF-100Robot-200Human setting further incorporates human demonstrations to enrich the diversity of task-level motion patterns.

Quantitative results are reported in Table VI, where success is measured at each task stage as well as overall task completion. Across all three tasks, policies without explicit task representations (BC-0) or with point-flow guidance (ATM)

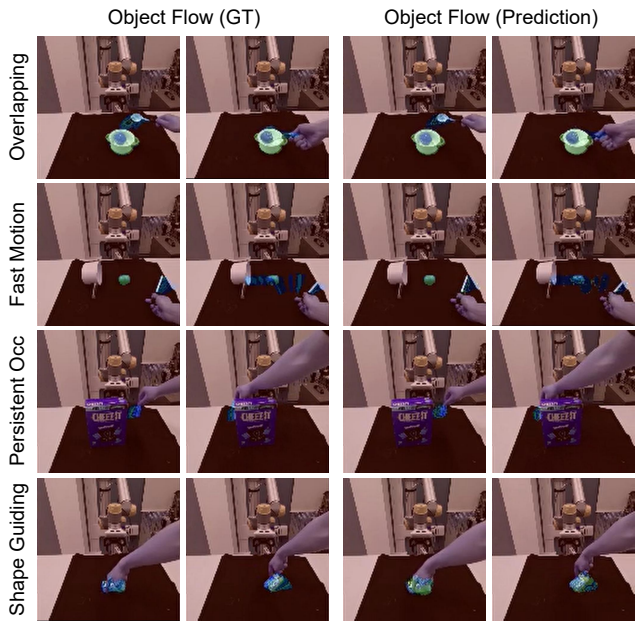


Fig. 14: Evaluation of object flow prediction in four challenging scenarios: overlapping objects, fast motion, persistent occlusion, and shape-guided manipulation under occlusion. Ground-truth object flow is obtained using Cutie [69]. Unified object masks introduce ambiguity under overlap, while fast motion causes temporal discontinuities. The model remains robust to occlusion and can recover object flow after occlusion-induced disturbances.

fail to reliably complete even the early stages, particularly under unseen scenarios. In contrast, object-flow-based policies achieve substantially higher success rates across all stages. Notably, incorporating human demonstrations further improves robustness, especially in unseen settings, leading to the highest overall success rate. These results indicate that learning a manipulator-agnostic object-flow task representation enables effective generalization to precision-critical, cluttered, and long-horizon manipulation tasks that were not encountered during training.

### D. Generalization of Object Flow

Beyond real-world policy execution, we evaluate object flow prediction under four challenging conditions: overlapping objects, fast motion, persistent occlusion, and shape-guided manipulation under occlusion, as shown in Fig. 14.

When multiple objects overlap, object flow predicted from a unified object mask cannot distinguish category-specific motion and tends to follow the dominant visible object region. This limitation suggests the need for incorporating object-category information in complex multi-object interactions.

Fast motion introduces temporal discontinuities in both ground-truth and predicted object flow. Although the prediction captures the coarse motion direction, high-speed displacement increases uncertainty, which may be mitigated by object velocity estimation or temporal interpolation.

Under persistent occlusion, the object remains trackable using Cutie [69], and the predicted flow recovers the post-occlusion object state. However, estimating flow during complete occlusion would require additional motion inference. In shape-guided manipulation, the prediction may temporarily drift to the human hand under heavy occlusion, but it refocuses on the object once visibility is restored, indicating robustness to transient occlusion-induced disturbances.

## VIII. Conclusion

We introduce *object flow* as a novel task representation to enable learning from multimodal cross-behavior manipulation data. Unlike conventional approaches that rely on detailed object and robot modeling, object flow encodes essential object motion information in a compact and structured manner, serving as a low-dimensional yet effective guidance for action policy learning. Our experiments demonstrate that learning object flow significantly outperforms state-of-the-art visual subgoal and point flow methods, leading to improved policy performance both within and beyond the training domain.

To further validate the effectiveness of our method, we conduct an in-depth analysis of object flow prediction accuracy and perform ablation studies on key components, including object flow length and cross-behavior learning. Our results indicate that predicting future object motion enables more robust action generation, thereby improving policy reliability. Moreover, the task representation extracted from pure human data could be seamlessly used for the robot policy, demonstrating its compatibility with diverse demonstration modalities. As the similarity between human and robot behaviors increases, the object flow becomes more informative, resulting in a more generalizable policy.

Finally, we discuss limitations and directions for future work. Our current approach treats all objects in a scene as belonging to a single category to promote broad generalization across tasks. However, this assumption can introduce ambiguity in scenarios with significant object overlap, thereby reducing the effectiveness of object flow estimation. In addition, object flow prediction remains challenging in high-speed human demonstrations, where capturing fine-grained motion states accurately is difficult. Despite these limitations, our results demonstrate that object flow remains effective even under severe occlusions, highlighting the robustness of the approach in complex manipulation scenarios.

## References

[1] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," in *Conference on Robot Learning*. PMLR, 2023, pp. 3766–3777.

[2] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," *arXiv preprint arXiv:2109.13396*, 2021.

[3] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar, "Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4788–4795.

[4] A. Xie, L. Lee, T. Xiao, and C. Finn, "Decomposing the generalization gap in imitation learning for visual robotic manipulation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 3153–3160.

[5] A. W. Needham and E. L. Nelson, "How babies use their hands to learn about objects: Exploration, reach-to-grasp, manipulation, and tool use." *WIREs: Cognitive Science*, vol. 14, no. 6, 2023.

[6] C. M. Heiman, W. G. Cole, D. K. Lee, and K. E. Adolph, "Object interaction and walking: Integration of old and new skills in infant development," *Infancy*, vol. 24, no. 4, pp. 547–569, 2019.

[7] J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, Y. Chen, W. Zeng, and P. S. Yu, "Generalizing to unseen domains: A survey on domain generalization," *IEEE transactions on knowledge and data engineering*, vol. 35, no. 8, pp. 8052–8072, 2022.

[8] F. Muratore, F. Ramos, G. Turk, W. Yu, M. Gienger, and J. Peters, "Robot learning from randomized simulations: A review," *Frontiers in Robotics and AI*, vol. 9, p. 799893, 2022.

[9] Y. Hu, Q. Xie, V. Jain, J. Francis, J. Patrikar, N. V. Keetha, S. Kim, Y. Xie, T. Zhang, S. Zhao *et al.*, "Toward general-purpose robots via foundation models: A survey and meta-analysis," *CoRR*, 2023.

[10] A. Gupta, A. Murali, D. P. Gandhi, and L. Pinto, "Robot learning in homes: Improving generalization and reducing dataset bias," *Advances in neural information processing systems*, vol. 31, 2018.

[11] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[12] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[13] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[14] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 892–909.

[15] J. Zeng, Q. Bu, B. Wang, W. Xia, L. Chen, H. Dong, H. Song, D. Wang, D. Hu, P. Luo *et al.*, "Learning manipulation by predicting interaction," *CoRR*, 2024.

[16] 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," *Transactions on Machine Learning Research Journal*, pp. 1–31, 2024.

[17] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[18] A. Aghajanyan, L. Yu, A. Conneau, W.-N. Hsu, K. Hambardzumyan, S. Zhang, S. Roller, N. Goyal, O. Levy, and L. Zettlemoyer, "Scaling laws for generative mixed-modal language models," in *International Conference on Machine Learning*. PMLR, 2023, pp. 265–279.

[19] Y. Ruan, C. J. Maddison, and T. Hashimoto, "Observational scaling laws and the predictability of language model performance," *arXiv preprint arXiv:2405.10938*, 2024.

[20] Z. Xue, S. Deng, Z. Chen, Y. Wang, Z. Yuan, and H. Xu, "Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning," *arXiv preprint arXiv:2502.16932*, 2025.

[21] F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao, "Data scaling laws in imitation learning for robotic manipulation," *arXiv preprint arXiv:2410.18647*, 2024.

[22] P. Sharma, L. Mohan, L. Pinto, and A. Gupta, "Multiple interactions made easy (mime): Large scale demonstrations data for imitation," in *Conference on robot learning*. PMLR, 2018, pp. 906–915.

[23] S. Haldar, J. Pari, A. Rai, and L. Pinto, "Teach a robot to fish: Versatile imitation from one minute of demonstrations," in *Robotics: Science and Systems*, 2023.

[24] J. Yang, Z. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg, "Equibot: Sim (3)-equivariant diffusion policy for generalizable and data efficient learning," in *8th Annual Conference on Robot Learning*.

[25] J. Yang, C. Glossop, A. Bhorkar, D. Shah, Q. Vuong, C. Finn, D. Sadigh, and S. Levine, "Pushing the limits of cross-embodiment learning for manipulation and navigation," *arXiv preprint arXiv:2402.19432*, 2024.

[26] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song, "Xskill: Cross embodiment skill discovery," in *Conference on robot learning*. PMLR, 2023, pp. 3536–3555.

[27] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi, "Xirl: Cross-embodiment inverse reinforcement learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 537–546.

[28] T. Wang, D. Bhatt, X. Wang, and N. Atanasov, "Cross-embodiment robot manipulation skill transfer using latent space alignment," *CoRR*, 2024.

[29] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," in *International Conference on Learning Representations*, 2017.

[30] P. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2019.

[31] P. Jian, E. Lee, Z. Bell, M. M. Zavlanos, and B. Chen, "Policy stitching: Learning transferable robot policies," in *Conference on Robot Learning*. PMLR, 2023, pp. 3789–3808.

[32] J. Shintake, V. Cacucciolo, D. Floreano, and H. Shea, "Soft robotic grippers," *Advanced materials*, vol. 30, no. 29, p. 1707035, 2018.

[33] I. M. Bullock and A. M. Dollar, "Classifying human manipulation behavior," in *2011 IEEE international conference on rehabilitation robotics*. IEEE, 2011, pp. 1–6.

[34] 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," in *Conference on Robot Learning*. PMLR, 2023, pp. 201–221.

[35] 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)*. IEEE, 2021, pp. 7827–7834.

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

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

[38] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *Conference on robot learning*. PMLR, 2022, pp. 158–168.

[39] D. J. Foster, A. Block, and D. Misra, "Is behavior cloning all you need? understanding horizon in imitation learning," *arXiv preprint arXiv:2407.15007*, 2024.

[40] Y. Ding, C. Florensa, P. Abbeel, and M. Phielipp, "Goal-conditioned imitation learning," *Advances in neural information processing systems*, vol. 32, 2019.

[41] 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.

[42] C. Lynch and P. Sermanet, "Language conditioned imitation learning over unstructured data," *arXiv preprint arXiv:2005.07648*, 2020.

[43] B. Brito, M. Everett, J. P. How, and J. Alonso-Mora, "Where to go next: Learning a subgoal recommendation policy for navigation in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4616–4623, 2021.

[44] Y. Lee, J. J. Lim, A. Anandkumar, and Y. Zhu, "Adversarial skill chaining for long-horizon robot manipulation via terminal state regularization," *arXiv preprint arXiv:2111.07999*, 2021.

[45] H. Zhou, X. Yao, Y. Meng, S. Sun, Z. Bing, K. Huang, and A. Knoll, "Language-conditioned learning for robotic manipulation: A survey," *arXiv preprint arXiv:2312.10807*, 2023.

[46] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, "Libero: Benchmarking knowledge transfer for lifelong robot learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 44 776–44 791, 2023.

[47] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, "Keto: Learning keypoint representations for tool manipulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7278–7285.

[48] Z. Huang, X. Lin, and D. Held, "Mesh-based dynamics with occlusion reasoning for cloth manipulation," *arXiv preprint arXiv:2206.02881*, 2022.

[49] J. Gu, S. Kirmani, P. Wohlhart, Y. Lu, M. G. Arenas, K. Rao, W. Yu, C. Fu, K. Gopalakrishnan, Z. Xu *et al.*, "Rt-trajectory: Robotic task generalization via hindsight trajectory sketches," *CoRR*, 2023.

[50] Y. Li, Y. Deng, J. Zhang, J. Jang, M. Memmel, R. Yu, C. R. Garrett, F. Ramos, D. Fox, A. Li *et al.*, "Hamster: Hierarchical action models for open-world robot manipulation," *arXiv preprint arXiv:2502.05485*, 2025.

[51] C. Yuan, C. Wen, T. Zhang, and Y. Gao, "General flow as foundation affordance for scalable robot learning," in *Conference on Robot Learning*. PMLR, 2025, pp. 1541–1566.

[52] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song, "Flow as the cross-domain manipulation interface," in *Conference on Robot Learning*. PMLR, 2025, pp. 2475–2499.

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

[54] 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.

[55] 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)*. IEEE, 2018, pp. 7111–7118.

[56] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2018, pp. 373–385.

[57] J. Wang, C. Hu, Y. Wang, and Y. Zhu, "Dynamics learning with object-centric interaction networks for robot manipulation," *IEEE Access*, vol. 9, pp. 68 277–68 288, 2021.

[58] Z. Weng, F. Paus, A. Varava, H. Yin, T. Asfour, and D. Kragic, "Graph-based task-specific prediction models for interactions between deformable and rigid objects," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5741–5748.

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

[60] H. Yuan, B. Zhou, Y. Fu, and Z. Lu, "Cross-embodiment dexterous grasping with reinforcement learning," 2024. [Online]. Available: https://arxiv.org/abs/2410.02479

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

[62] M. Lepert, J. Fang, and J. Bohg, "Phantom: Training robots without robots using only human videos," 2025. [Online]. Available: https://arxiv.org/abs/2503.00779

[63] P. Sharma, D. Pathak, and A. Gupta, "Third-person visual imitation learning via decoupled hierarchical controller," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 2597–2607.

[64] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.

[65] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," *arXiv:2304.02643*, 2023.

[66] 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.

[67] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang, "Grounded sam: Assembling open-world models for diverse visual tasks," 2024.

[68] K. Sofiiuk, I. A. Petrov, and A. Konushin, "Reviving iterative training with mask guidance for interactive segmentation," in *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2022, pp. 3141–3145.

[69] H. K. Cheng, S. W. Oh, B. Price, J.-Y. Lee, and A. Schwing, "Putting the object back into video object segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 3151–3161.

[70] S. A. Taghanaki, Y. Zheng, S. K. Zhou, B. Georgescu, P. Sharma, D. Xu, D. Comaniciu, and G. Hamarneh, "Combo loss: Handling input and output imbalance in multi-organ segmentation," *Computerized Medical Imaging and Graphics*, vol. 75, pp. 24–33, 2019.

[71] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Robotics: Science and Systems*, 2023.


**Longrui Chen** (Student Member, IEEE) received the B.Eng. degree in robotics from Beihang University, Beijing, China, and the M.Sc. degree from Imperial College London, London, U.K.

He is currently pursuing the Ph.D. degree with the University of Leeds, Leeds, U.K. His research interests include robot manipulation and imitation learning.

**David Russell** is a postdoctoral researcher specializing in robotic manipulation at the University of Leeds. He obtained both his Master's degree in Mechatronics and Robotics and his PhD in Robotic Manipulation from the same institution, where he developed a strong foundation in advanced robotic systems and trajectory optimization. His current research interests span manipulation in cluttered environments, robotic packing and integrated systems design, trajectory optimization, and model predictive control.

**Yulei Qiu** received the B.Eng. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2020, and the M.Sc. degree from Delft University of Technology, Delft, The Netherlands, in 2022.

He is currently pursuing the Ph.D. degree at the University of Leeds, Leeds, U.K. His research interests include robotic manipulation and imitation learning.

**Mehmet Dogar** received his PhD from the Robotics Institute at the Carnegie Mellon University in 2013. He is currently a Professor in Robotics and AI at the School of Computer Science, University of Leeds, UK. He was a Visiting Professor at ETH Zurich in 2023 and a post-doctoral researcher at CSAIL, MIT between 2013-2015. Prof. Dogar is an Associate Editor for IEEE Transactions on Robotics and also for the International Journal of Robotics Research. His research focuses on robotic object manipulation.