

A APPENDIX

Problem Definition The demonstrations are captured as egocentric videos with a non-static camera perspective that naturally follows the user’s hand during manipulation. The task we define is a one-shot learning task, as our setting provides only a single real demonstration per test task, and the experimental task is not identical to the pretraining data. Since the dynamic viewpoint may not consistently capture both the hand and the object in every frame, our framework models the hand’s motion trajectory as the primary signal, using object observations as reference cues only when they are visible.

Simulation Data To equip our policy with basic manipulation primitives, we pre-train it on simulated pick-and-place trajectories derived from common daily activities. These include reaching, grasping, and placing a variety of simple objects (e.g., controllers, boxes) in MuJoCo, without any particular selection to closely match the testing tasks. The simulated trajectories provide diverse motion examples that can be reused across tasks, effectively bootstrapping the policy prior to fine-tuning on the single available real-world demonstration.

Object and Morphology Comparison For each task we first segment the “interactive region” in the wrist-camera RGB-D stream, for example, the box and its contents in the “open box” task, by generating a binary mask of pixels corresponding to the object(s) of interest. Segmentation results assist the model to focus on the interactive object in the RGB image and the depth map. We then extract two parallel feature streams: (1) RGB branch: a standard 3-channel ResNet backbone (2) Depth branch: a single-channel ResNet backbone. Their outputs are concatenated and passed through a lightweight fusion layer that learns per-modality weights. The resulting fused feature vector encodes the object’s morphology, which we use to compare against each policy’s reference morphology during ensemble-weight computation. We estimate the hand–object distance directly in the camera coordinate frame by projecting both the segmented hand and object point clouds from the same RGB-D image using the camera’s intrinsic parameters. Because the hand and object are observed under identical lighting and viewpoint, this relative distance remains consistent and reliable. We extract “key frames” from the egocentric video only when the entire hand is fully visible, and we segment both the hand and object using the Segment Anything Model (SAM) to ensure precise mask generation before computing their spatial relationship.

Ensemble Weight At every coarse-level decision step, we update the ensemble weights by measuring the similarity between the current object morphology and each policy’s stored morphology prototype. The policy’s stored morphology prototype is collected from the one-shot egocentric video demonstration. 1. Observation: The wrist camera provides an RGB image and a depth map. 2. Feature extraction and fusion: As described above, these inputs are processed to yield a morphology embedding f . 3. Similarity: We compute the cosine similarity between the policy’s stored object feature f_p and the observed object feature f_o , obtained after projection from the raw input. 4. Normalization: We apply a softmax over these cosine scores to produce positive, sum-to-one weights $\{w_i\}$. These weights are recomputed at each coarse-level step based on the latest observation, allowing the ensemble to dynamically adapt to changes in object appearance and pose.

RL Details The fine-level control will apply after determine the ensemble weights. The frequency ratio between coarse-level control (Ensemble Action Prediction) and fine-level control (Reinforcement Action Refinement) is 1:10. We define the reward r_t as: $r_t = -\alpha ||p_t - p_{\text{target}}||_2 + k \mathbf{1}(\Delta\theta_t \leq \theta_{\text{max}}) + \beta \mathbf{1}(d_m \leq d_c \wedge \Delta\theta_t \leq \theta_{\text{max}}) - \delta \mathbf{1}(\Delta\theta_t > \theta_{\text{max}})$, where the function represents the distance reward and the action space reward for fine-level control. The coefficient α, k, β, δ can adjust the reward. The robot gets a higher reward when end-effector p_t is close to the target position p_{target} . The action space reward is based on the direction of the next position and the moving distance d_m of each step. The moving distance threshold of each step is d_c , and the direction angle threshold is θ_{max} . The robot is encouraged to move forward in the action space aligned with the coarse-level direction and receives a smaller penalty when the end-effector is closer to the target.

Baseline Comparison The pull-drawer task in our setup closely mirrors the drawer-opening behavior used in BiDP, whereas the other two tasks involve actions that differ substantially from those in BiDP. To ensure a fair comparison, all baseline methods were trained exclusively on the demonstrations recorded for this study.