GHOST: Hierarchical Sub-Goal Policies for Generalizing Robot Manipulation

Sriram Krishna[1], Ben Eisner[1], Haotian Zhan[1], Ying Yuan[1], Haoyu Zhen[2], Chuang Gan[2], Shubham Tulsiani[1], David Held[1]
[1]Robotics Institute, Carnegie Mellon University
[2]UMass Amherst
https://ghost-human-demo.github.io/

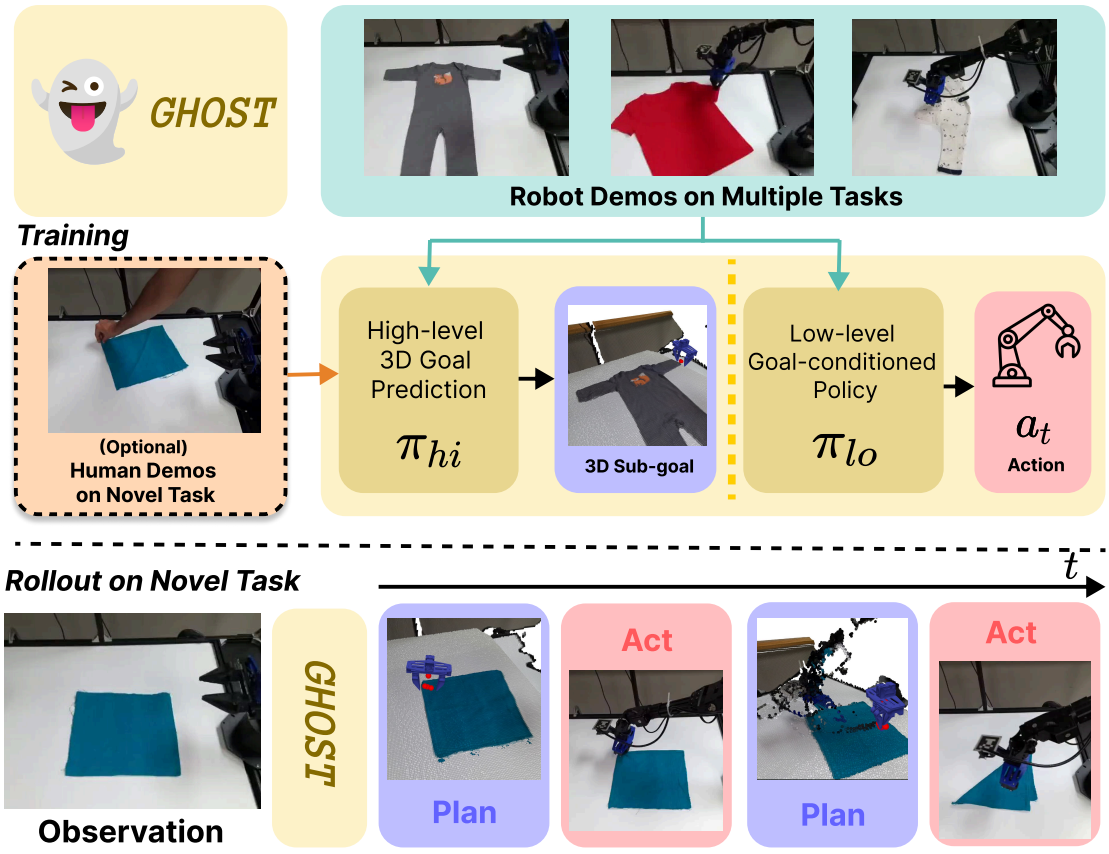

Fig. 1: GHOST learns skills from robot teleoperation data and optionally uses human demonstrations to generalize to novel tasks. We train a hierarchical policy that decouples embodiment-agnostic goal prediction (π_{hi}) from embodiment-specific action execution (π_{lo}). By training π_{hi} on both robot and human data and π_{lo} purely on robot data, GHOST transfers learned manipulation skills to out-of-distribution tasks with third person human video demonstrations.

***Abstract*—We present GHOST, a framework for learning visuomotor manipulation policies that *generalize* beyond the training distribution. GHOST factorizes control into (i) a high-level policy that predicts the next sub-goal as a *distribution* over 3D end-effector poses from multi-view RGB-D observations, and (ii) a low-level goal-conditioned controller that executes embodiment-specific actions. To condition image-based policies on 3D goals, we introduce a simple spatial interface that projects predicted goals into the image plane and represents them as *end-effector heatmaps*. Across a suite of manipulation tasks, this hierarchical factorization consistently improves performance and robustness compared to a flat Diffusion Policy.**

Further, we show that this hierarchical interface also makes it easy to incorporate human demonstrations without relying on (noisy) action retargeting. As sub-goals are largely embodiment-agnostic, we train the high-level policy on human video to specify how learned skills should be applied and composed, while keeping the low-level policy trained purely on robot data. This hierarchy enables adaptation to novel objects and task variations using a small number of human demonstrations.

I. Introduction

The dominant paradigm today for teaching robot policies to manipulate their environment is imitation learning (IL). Given a sufficiently large number of demonstration trajectories, imitation learning algorithms can learn precise, complex behaviors [4, 35]. However, to generalize to novel objects and

environments, they require massive amounts of data, typically collected through *teleoperation*, where a human operator controls a robot to complete a given task [2, 14, 26]. This recipe, while simple, is expensive and labor-intensive to scale.

In this work we focus on an alternate lever: *hierarchy*. Many manipulation tasks are naturally organized into phases - *e.g.,* a pick-and-place task may be decomposed into grasp, transport, and place phases. We argue that explicitly modeling this hierarchy improves robustness and data efficiency by separating *sub-goal selection* from *low-level execution*. Concretely, we define sub-goals as end-effector poses at phase boundaries, which provide a compact interface for composing skills and extending behavior over long horizons.

Based on this perspective, we propose a hierarchical policy that factorizes control into two modules. (i) A high-level policy π_{hi} predicts the next sub-goal from visual observations and a language instruction, and (ii) a low-level policy π_{lo} conditioned on the predicted goal to produce embodiment-specific actions. To connect these modules while retaining the benefits of image-based policies, we introduce a spatial goal representation that projects 3D sub-goals into the image plane and represents them as *end-effector heatmaps*, enabling effective goal-conditioning of image-based policies.

This hierarchy yields two practical benefits. First, even when trained only on robot demonstrations, our hierarchical factorization significantly improves success rates compared to a flat Diffusion Policy [4]. Second, our insight is that this sub-goal interface is largely embodiment-agnostic, making it a convenient point to incorporate additional supervision. As a secondary contribution, we show that we can use human videos to train the high-level policy for novel task variants, without requiring retargeted action labels, enabling out-of-distribution generalization.

Thus, we present **GHOST**: **G**eneralizing manipulation via **H**ierarchical end-effect**O**r **S**ub-goals for Skill **T**ransfer. GHOST is a hierarchical framework for training visuomotor policies that can generalize skills through human demonstrations. GHOST features: a) a 3D high-level goal prediction network that takes in RGB-D images to predict a distribution of sub-goal end-effector poses, and b) a low-level policy conditioned on goals represented as heatmap projections of the predicted end-effector sub-goals.

Our contributions are as follows:

- A hierarchical framework for training policies with 3D end-effector keypoint sub-goals and heatmap-based goal-conditioning that consistently improves in-distribution performance over flat policies.
- A demonstration that this factorization naturally enables cross-embodiment transfer from human video without action retargeting, by training π_{hi} on heterogeneous human and robot data while keeping π_{lo} grounded in robot demonstrations.

II. Related Work

A. Hierarchical Imitation Learning

Hierarchical Imitation Learning factorizes the policy into a high-level "planner" and a low-level "control" policy. Previous work has primarily focused on learning high- and low-level policies from robot data [17, 30, 16]. This structure is particularly attractive for manipulation, where tasks often decompose into phases and where long-horizon behaviors can be expressed as sequences of sub-goals. Recent work has shown that representing sub-goals directly in end-effector space can yield strong generalization for articulated manipulation [28]. In parallel, goal-conditioning has been explored through images and language, often by predicting intermediate targets and conditioning a controller on them [1, 29, 6]. Our work is most closely related to hierarchical approaches that use explicit spatial sub-goals. We differ in two ways: (i) we predict 3D end-effector sub-goals from multi-view RGB-D, and (ii) we introduce an *end-effector heatmap* interface that makes goal-conditioning compatible with image-based policies.

B. Learning from human demonstrations

Large-scale robot teleoperation datasets have enabled impressive generalization in visuomotor policies [14, 18, 3, 2]. However, collecting robot demonstrations remains expensive, motivating complementary sources of supervision such as human video [8, 9, 15]. One line of work attempts to directly learn robot actions from human demonstrations by estimating hand pose and retargeting to a robot end-effector [19, 21, 13, 22, 10, 15]. These approaches can suffer from noisy pose estimates and an embodiment mismatch. By restricting human supervision to embodiment-agnostic sub-goals, we avoid action retargeting and keep low-level control grounded in robot demonstrations. Another line of work learns higher-level representations or latent dynamics from video corpora and maps them to robot actions with additional robot data [31, 27]. Most similar to our work is MimicPlay [27], which learns a latent distribution over hand trajectories collected through play data. However, this approach relies on a video prompt of the task being provided at test-time, whereas we only require a language goal. In contrast, we treat human demonstrations as *optional* supervision for the high-level planner in a hierarchy.

III. Problem Statement and Assumptions

We consider manipulation tasks that can be naturally decomposed into a sequence of sub-goal states $\{g_i\}_{i\in[1\ldots M]}$, such that the task is completed when each sub-goal state is reached. We denote a *skill* $\pi(\cdot|l)$ as a reusable manipulation capability described by a language instruction l (*e.g.,* 'grasping', 'placing', 'folding') that can be instantiated across different objects and contexts [3]. We make the assumption that for every sub-goal transition $g_i \rightarrow g_{i+1}$ in a task, there exists a reusable language-conditioned skill $\pi(\cdot|l_{i+1})$ which can achieve the next desired sub-goal state (*e.g.,* a pick-and-place task would be decomposed into sub-goals {'pick', 'place'}, with corresponding reusable skills to accomplish each transition). In this work, we are concerned with how to both learn representations

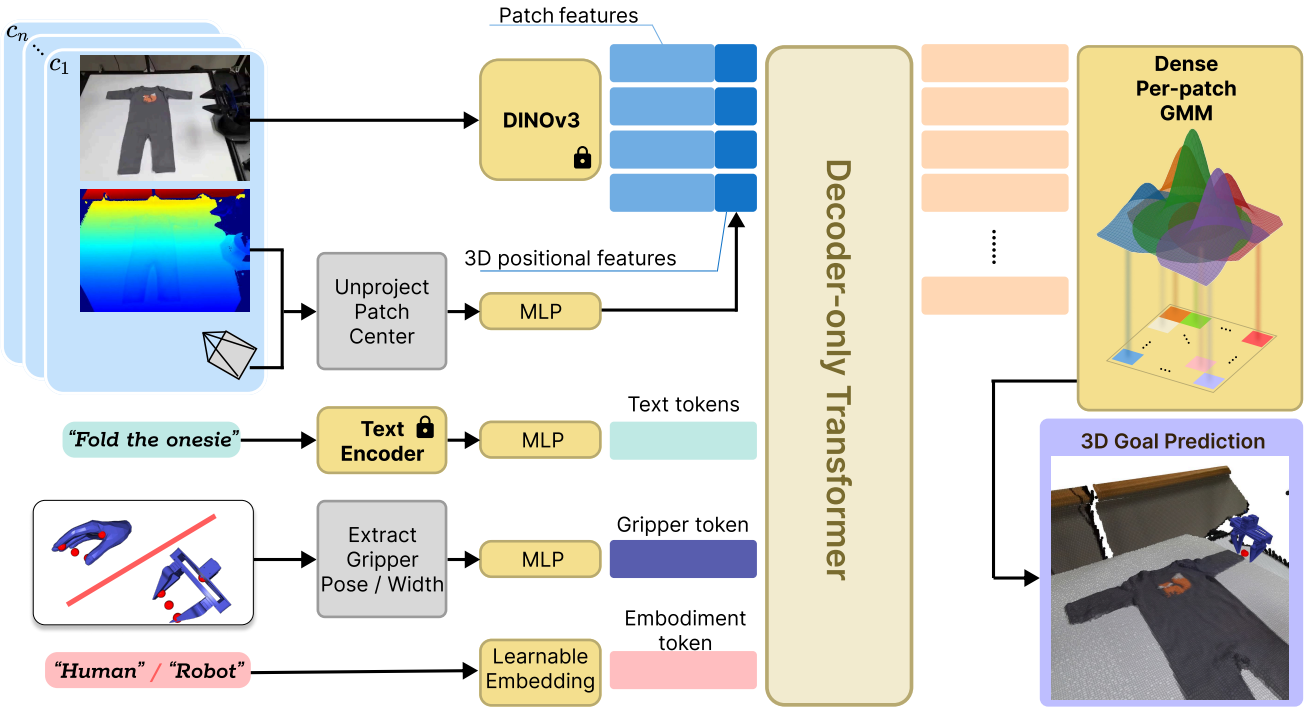

Fig. 2: **GHOST** High-level sub-goal prediction architecture: RGB-D observations from multiple cameras are processed with a DINOv3 encoder, with patch tokens augmented by 3D coordinates. Additional context (gripper state, language embedding, embodiment name) is encoded via separate MLPs. A decoder-only transformer processes all tokens, with each patch predicting the GMM parameters of a 3D sub-goal distribution over end-effector keypoints.

of these sub-goals g_i, and learn language-conditioned skills $\pi(\cdot|l_i)$ from demonstrations, such that agents can generalize to novel task variations.

Training data. We assume access to robot teleoperation demonstrations $\mathcal{D}_{robot} = \{(o_t, \text{aux}_t, a_t, e_t, g)\}$ - workspace cameras o_t, auxiliary robot observations aux_t (*e.g.,* wrist cameras, proprioception), actions a_t, end-effector poses e_t and a language goal g, where t denotes the timestep of the observation.

Optional human data. We optionally allow an additional dataset of human videos $\mathcal{D}_{human} = \{(o_t, \hat{e}_t, g)\}$, where $\hat{e}_t$ is an estimated hand pose from an off-the-shelf tracker [21, 32]. Human demonstrations do not provide action labels, so we use them only to train or adapt our high level policy π_{hi} on estimated end-effector poses $\hat{e}_t$, keeping the low-level policy π_{lo} trained purely on robot data.

Assumptions. We assume calibrated RGB-D workspace cameras for goal prediction, and that demonstrations contain sufficient visual cues to localize the hand near sub-goal boundaries. In addition, we assume that human demonstrations use similar grasp types as the robot gripper to minimize the embodiment gap. Full hardware configuration and per-task data collection details are provided in Appendix B and Appendix C respectively.

IV. GHOST

A. Overview

The GHOST framework learns a hierarchy over *sub-goal end-effector poses*. During training, sub-goal boundaries provide supervision for the high-level planner π_{hi}, while the full robot trajectories provide action supervision for the low-level controller π_{lo}. At test time, π_{hi} predicts the next end-effector sub-goal (as a distribution) from workspace observations and language, and π_{lo} executes goal-conditioned control toward that sub-goal. This separation isolates long-horizon reasoning in π_{hi} and maintains precise, embodiment-specific control in π_{lo}.

B. Data Collection and Preprocessing

Robot Data. We collect demonstrations through teleoperation across multiple task variants, each demonstrating the same skill (*e.g.,* pick-and-place). When possible, we extract sub-goals $s \in \mathcal{S}$ automatically by identifying the timesteps where the gripper state changes from open $\rightarrow$ close and vice versa.

Human Data. We optionally collect human demonstrations of the *same skill* in a novel setting, using either new objects or novel applications of the skill. We track 3D hand poses H_t using off-the-shelf hand pose estimators [21, 32]. We resolve the weak-perspective scale ambiguity of the detection by segmenting the hand using Grounded-SAM [23] and scaling the detection with the observed depth of the hand. We manually annotate sub-goals at the timesteps where a sub-goal

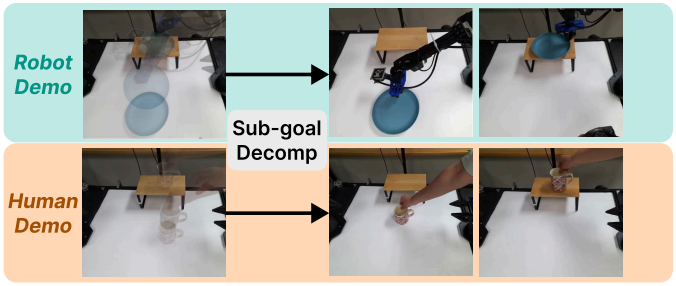

Fig. 3: Sub-goal decomposition for robot and human demonstrations. **Top**: Robot teleop demonstrations with sub-goals automatically extracted at gripper state transitions. **Bottom**: Human demonstrations with manually annotated sub-goals. Each sub-goal represents a semantically meaningful transition in the skill execution.

terminates. We treat sub-goal boundaries as supervision rather than discovering them; we discuss automated discovery as future work in Sec. VI. Figure 3 shows an example of our sub-goal decomposition for both robot and human demonstrations.

Representing the end-effector pose: Following prior work [28], we represent the end-effector pose as a sparse set of 3D points instead of a position and $SO(3)$ rotation. For robot data, we sample a point cloud P_R from the gripper mesh at each time-step and represent the end-effector pose e_t as a set of 4 3D points from P_R, located at the base of the gripper, the tips of the two gripper fingers and the grasping center. Similarly, for human data, we extract hand point clouds P_H from the MANO [24] mesh of the hand pose H_t and select 4 3D points - corresponding to points on the palm, tips of the thumb and index finger, and the grasping center. Figure 4 illustrates the keypoint locations for both gripper types. Thus, we extract a unified 3D end-effector representation for both human and robot data.

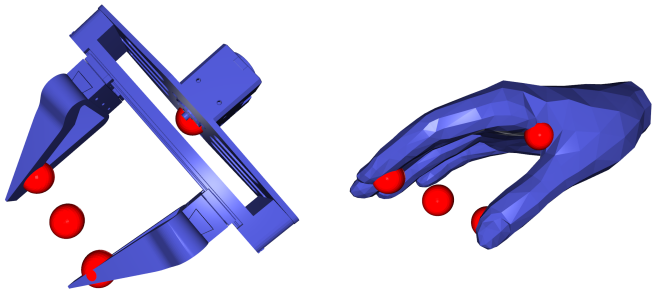

Fig. 4: End-effector representations for robot and human demonstrations. Instead of representing the gripper pose as a translation and SO(3) rotation, we represent the gripper pose as a set of 3D keypoints (red spheres): gripper base, fingertip locations, and grasping center. **Left**: Parallel-jaw gripper. **Right**: MANO hand.

C. Policy Learning

1) High-Level Goal Prediction Policy: The high-level policy π_{hi} predicts a distribution over the end-effector pose e_s at sub-goal timestep s. We parameterize π_{hi} as a decoder-only transformer [12] operating on RGB-D observations from C cameras. The architecture is shown in Figure 2. Each RGB-D image is processed independently with a frozen DINOv3 [25] encoder, producing $K \times C$ patch tokens (K per camera). Each patch token is augmented with the features of the 3D coordinate $[x, y, z]$ of the patch center, processed with an MLP. Additional tokens encode the current gripper pose and the task language embedding with Flan-T5 [5]. We also include a learnable token specifying whether the demonstration is from a human or robot.. All tokens are projected to the transformer's feature dimension through separate MLPs. We also add register tokens [7], as they have been shown to improve performance on dense prediction tasks without any change in training objective.

To handle the multimodality inherent in the demonstration data, we model the goal distributions as a dense per-patch Gaussian Mixture Model (GMM). Each patch token predicts: (1) a mixing weight w_i, and (2) four 3D residual vectors $\{\delta_{i,1}, \delta_{i,2}, \delta_{i,3}, \delta_{i,4}\}$ relative to the patch center p_i. These residual vectors are trained to predict the 3D points of the gripper. The global goal distribution is a mixture of $K \times C$ isotropic Gaussians (one per patch) with fixed variance σ[1]. We do not claim the Dense GMM as a novel contribution in this paper, though we believe that this paper is the first to apply the idea of a Dense GMM in this per-patch manner (as opposed to using a Dense GMM over point clouds and predicting a Gaussian per-point).

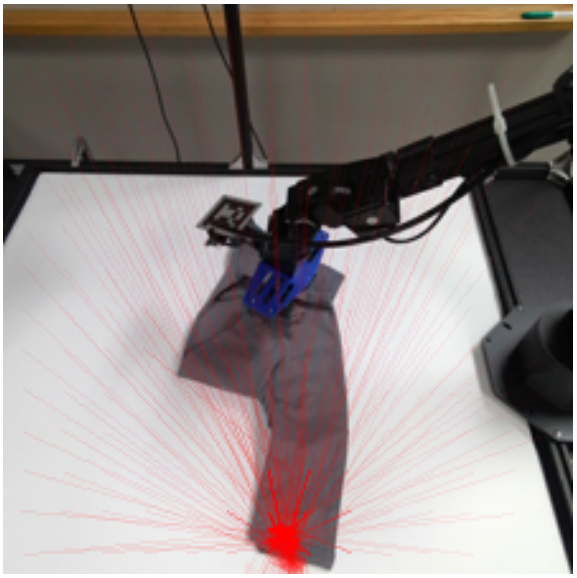

Fig. 5: High-level policy GMM predictions at inference. We visualize the mixture components of the GMM, with the opacity encoding mixing weight w_i and arrows showing projections of 3D residuals δ_i from patch centers p_i.

Training. We train π_{hi} by minimizing the negative log-

[1]In practice we sum NLL losses over a small set of fixed variances to capture sub-goal distributions at multiple scales; see Appendix A.1.

Task	Data Source	# Demos	Generalization Type
	Pick-and-Place		
`plate-on-table`	Robot	20	—
`plate-in-bin`	Robot	20	—
`mug-in-bin`	Robot	20	—
`mug-on-table`	Human	20	Object combination
	Cloth Folding		
`fold-onesie`	Robot	33	—
`fold-shirt`	Robot	50	—
`fold-onesie-ood`	Human	17	Object instance
`fold-towel`	Human	50	Object category + Skill composition
	Hammer Pin		
`hammer-pin`	Human+Robot	100	—

TABLE I: Overview of tasks. Blue: In-distribution (ID) tasks. Orange: Out-of-distribution (OOD) tasks requiring generalization.

likelihood on sub-goal transitions from a dataset of both human and robot data $\mathcal{D}' = \mathcal{D}_{robot} \cup \mathcal{D}_{human}$ that consists of observations o_t for each timestep t and end-effector poses e_s at the end of each sub-goal s:

$$\mathcal{L}_{hi} = \mathbb{E}_{(o_t, e_s)\sim\mathcal{D}'}\left[-\log\left(\sum_{i=1}^{K\times C} w_i \mathcal{N}(e_s; \mu_i, \sigma^2)\right)\right]$$

where μ_i is the predicted goal, given by the patch center + each of the residuals for patch i: $\mu_i = p_i + \delta_i$. Full architecture and training hyperparameters are listed in Appendix A.1.

Inference. During inference, we sample a patch i from the categorical distribution on mixing weights $\{w_i\}$. We then use the corresponding mean prediction $\mu_i = p_i + \delta_i$ to obtain the predicted end-effector pose. Figure 5 visualizes the components of the GMM, depicting the spatial distribution over the predicted residuals for a single keypoint in a single camera view.

2) Low-Level Goal Conditioned Policy: We instantiate π_{lo} as a Diffusion Policy [4], which generates action chunks $\{a_t, ..., a_{t+H}\}$ by denoising conditioned on observations.

Goal Conditioning. Each point in the 3D goal e_s is projected onto the image plane of each camera using the camera parameters, yielding sparse 2D coordinates $\mathbf{p}_i \in \mathbb{R}^2$ for $i \in \{1 \dots C\}$. These are converted to dense *end-effector heatmaps* H where each channel c encodes the pixel distance field from keypoint $\mathbf{p}_c$: $H_{xy}^{(c)} = \sqrt{\frac{\|\mathbf{x}_{xy} - \mathbf{p}_c\|_2}{d_{\max}}}$ where $\mathbf{x}_{xy}$ denotes the pixel coordinates, $d_{\max} = \sqrt{h^2 + w^2}$ normalizes by the image diagonal, and we use the square root for steeper gradients near the targets. We denote the resulting heatmap tensor as $\text{heatmap}(e_s) := H \in \mathbb{R}^{3\times h\times w}$, where $\mathbf{x}_{xy}$ denotes the pixel coordinates. We use heatmaps rather than single-pixel binary masks as goal representations, empirically we find them to yield better performance; we hypothesize this is due to the denser supervisory signal. For practical implementation reasons, we select three of the heatmap channels corresponding to non-collinear points in the goal[2], and pass these channels through standard 3-channel image encoder networks before incorporating into the policy network. This preserves spatial structure while densifying the sparse goal representation. This process is illustrated in Figure 7, and the architecture of the low-level goal conditioned policy is shown in Figure 6.

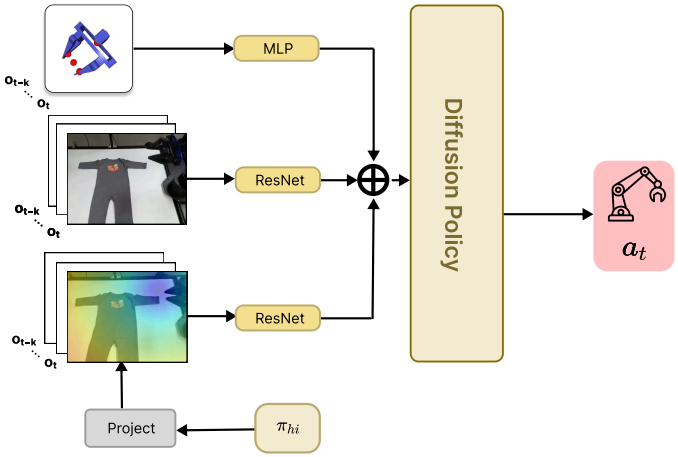

Fig. 6: **GHOST** Low-level goal-conditioned policy architecture. Images from each camera and the projected end-effector heatmap images are processed independently with ResNet [11] encoders and concatenated along with the proprioceptive input into a global conditioning vector for the Diffusion Policy.

Training. We use the standard diffusion loss, training on the robot-only data $\mathcal{D}_{robot}$:

$$\mathcal{L}_{lo} = \mathbb{E}_{\mathcal{D}_{robot},\epsilon}\left[\|\epsilon - \epsilon_\theta(a_t^{\text{noisy}}, o_t, \texttt{aux}_t, \text{heatmap}(e_s))\|^2\right]$$

where $\texttt{aux}_t$ are auxiliary robot observations (*e.g.,* wrist cameras, proprioception), a_t are low-level robot actions, a_t^{noisy} denotes the action corrupted according to the forward diffusion process and e_s is the gripper pose extracted from the robot

[2]We choose a point on the gripper wrist, as well as a point on each finger. Because of the rigid structure of the gripper, this minimal representation is sufficient to capture the state of the gripper, so discarding channels does not have a major effect on the policy.

demonstrations. We train π_{hi} and π_{lo} independently. Full hyperparameters for π_{lo} are listed in Appendix A.2.

Inference. At deployment, we execute π_{hi} and π_{lo} sequentially, where π_{hi} predicts a sub-goal conditioned on the current observation and π_{lo} predicts one action chunk conditioned on the heatmap representation of the sub-goal prediction.

Fig. 7: End-effector heatmap generation for goal conditioning. Predicted 3D keypoints are projected to 2D coordinates, and then converted to dense distance field heatmaps that encode spatial proximity to each keypoint.

V. Experiments and Results

We evaluate whether our hierarchical policy learning approach improves performance and enables skill generalization. Concretely, our experiments test two questions: **(Q1) In-distribution performance:** Do hierarchical policies improve in-distribution performance even without human data? **(Q2) Generalization:** Do human demonstrations enable transferring learned skills to novel object instances, categories, and contexts?

A. Tasks and Evaluation Protocol

Table I summarizes our evaluation tasks. We collect robot demonstrations on in-distribution (ID) tasks to train both π_{hi} and π_{lo}, and human demonstrations on out-of-distribution (OOD) variations to train only π_{hi}. For all experiments, we conduct $n = 30$ trials per method with randomized object configurations, reporting mean success rate $\pm$ half-width of the 95% percentile bootstrap CI (clipped to $[0, 100]$).

- **Pick-and-Place:** Grasp a specified object (plate, mug) and place on a target (table, bin). Robot demos include placing plates on a table/bin and placing a mug in the bin; human demos provide OOD (out of distribution) object-target combinations (see Table I). The OOD task `mug-on-table` evaluates whether models can demonstrate *compositional* generalization on a novel combination of objects. The task is considered a success if the object is placed stably on the target; objects placed successfully but knocked off during reset are given a score of 0.5.
- **Cloth Folding:** Long-horizon deformable manipulation requiring multiple sequential folds. Robot demos on folding onesies and shirts; human demos on novel instances (`fold-onesie-ood`) and applying the skill on entirely novel categories (`fold-towel`). k-step success indicates first k folds completed successfully.
- **Hammer Pin:** Pick up a hammer and strike a target pin. The blue, purple, red, and pink pins are initially raised. Robot demos include hammering the blue pin or the pink pin; human demos on novel target pin (purple pin). This task is considered a success if the target pin is fully driven in using the hammer.

B. Baselines

We evaluate GHOST against two baselines: (1) **Diffusion Policy (DP)** [4], a flat policy trained on robot data, and (2) **MimicPlay** [27], a hierarchical approach that learns a latent trajectory representation from human play data. For fair comparison, we train our high-level policy architecture with MimicPlay's representation and training objective, appending a CLS token to the input sequence as the latent plan with a separate MLP as the GMM decoder. This ensures that we measure the differences that stem from the goal representation rather than policy architecture. We also evaluate **GHOST (Robot Only)**, our method trained without human demonstrations, to isolate the benefit of hierarchy from the benefit of human data.

C. Implementation Details

We train a single multi-task policy with language conditioning for each method. Complete hyperparameters for both π_{hi} and π_{lo}, our hardware setup, and per-task data collection details are provided in Appendix A, Appendix B, and Appendix C respectively. For the Diffusion Policy baseline, we add language conditioning by appending the text embedding to the global conditioning vector. For hierarchical methods, language conditioning is only provided for the high-level policy. We train all DP variants for 300k steps with color jitter and random crop augmentations. The high-level policy is trained for 100 epochs with additional augmentations: adding noise to the gripper pose, blur, grayscale, and token dropout augmentations. At inference, we synchronously execute π_{hi} and π_{lo}, *i.e.,* the low-level predicts action chunks that fully execute before the next high-level inference step, enabling reactive replanning to changes in environment.

D. Results

Do hierarchical policies improve in-distribution performance even without human data? For `plate-on-table` (Table II) we see that nearly all methods saturate in performance, as the task is simple and sufficient training data is available. However, for a long-horizon complex task, we see a significant increase in the performance of "GHOST (Ours - Robot Only)" compared to "DP". As shown in Table III, on `fold-onesie` performance increases from 10% to 80% final success, showing large benefits from the hierarchical decomposition. Similarly, in `hammer-pin`, a task requiring precise grasping of the hammer tool and striking the correct pin, we see that performance significantly improves from DP (16.7%) to "GHOST (Ours - Robot Only)" (50%) (Table V). Appendix D further isolates the contribution of the DINOv3 backbone via an ablation with a tiny ViT trained from scratch; the hierarchy alone (with no pre-trained encoder) already substantially outperforms flat DP, while DINOv3 provides additional gains on long-horizon tasks.

Method	Success Rate (%) ↑	
	plate-on-table	mug-on-table
DP	80.0 ± 15.0	13.3 ± 9.2
MimicPlay	65.0 ± 15.0	28.3 ± 12.5
GHOST (Ours - Robot Only)	83.3 ± 10.8	55.0 ± 15.0
GHOST (Ours)	**98.3 ± 2.5**	**63.3 ± 13.3**

TABLE II: Pick-and-place success rates. Best: **shaded**, second-best: underline.

Task	Method	Success Rate (%) ↑				
		1-step	2-step	3-step	4-step	Final
fold-onesie	DP	90.0	76.7	53.3	40.0	10.0 ± 11.7
	MimicPlay	93.3	76.7	70.0	70.0	46.7 ± 16.7
	GHOST (Ours - Robot Only)	**100.0**	**100.0**	96.7	86.7	80.0 ± 15.0
	GHOST (Ours)	**100.0**	**100.0**	**100.0**	**90.0**	**83.3 ± 13.3**
fold-onesie-ood (Novel Object Instance)	DP	76.7	60.0	46.7	26.7	10.0 ± 10.0
	MimicPlay	63.3	36.7	30.0	20.0	0.0 ± 0.0
	GHOST (Ours - Robot Only)	**100.0**	**100.0**	73.3	60.0	43.3 ± 16.7
	GHOST (Ours)	**100.0**	**100.0**	**93.3**	**86.7**	**56.7 ± 16.7**

TABLE III: Onesie-folding success rates. k-step = first k folds completed. Final = all 5 folds completed. Best: **shaded**, second-best: underline.

Method	Success Rate (%) ↑
MimicPlay	16.7 ± 13.3
GHOST (Ours)	**36.7 ± 16.7**

TABLE IV: fold-towel (Novel Object Category + Skill Composition) success rates. Best: **shaded**.

Method	Success Rate (%) ↑
DP	16.7 ± 13.3
MimicPlay	33.3 ± 16.7
GHOST (Ours - Robot Only)	50.0 ± 16.7
GHOST (Ours)	**70.0 ± 16.7**

TABLE V: hammer-pin success rates. Best: **shaded**, second-best: underline.

Do human demonstrations enable transferring learned skills to novel object instances, categories, and contexts? As shown in Table II, Table III and Table IV human demonstrations unlock meaningful OOD transfer of learned skills to novel objects and novel skill compositions. GHOST achieves 63.3% success on mug-on-table, a task featuring a combination of objects unseen in robot demonstrations. On fold-onesie-ood, GHOST achieves 56.7% final success vs 43.3% for "GHOST (Ours - Robot Only)" and 0% for MimicPlay (Table III). Finally, on the hardest task of generalizing a policy to a novel object category and skill combination (fold-towel), GHOST achieves 36.7% success as compared to 16.7% with the MimicPlay baseline.

VI. Limitations and Future Work

Although GHOST shows significant gains in performance through a hierarchical framework, and promising results in generalizing learned skills across multiple axes, it is not without limitations. Like other methods that use hand pose estimators on human demonstrations [15, 22], our approach is limited by the quality of the estimated hand pose, and suffers in situations where hand pose estimation has errors. We extract sub-goals through heuristics (gripper open/close), which may not be optimal or even valid for tasks requiring continuous manipulation skills (pouring, stirring, or hammering), and through annotation on human demonstrations, which imposes an additional cost on the collection of demonstrations. Related work on automatic sub-goal discovery [20, 34] is an interesting direction for future research on mining sub-goals from collected human and robot demonstrations.

While GHOST generalizes to novel object instances and composes learned skills in novel sequences, we observe a steep drop in performance as the task horizon lengthens, which we attribute to the visual domain gap between human and robot observations in π_{hi}. To verify this, we conduct an oracle ablation (Appendix E): when π_{hi} is trained on a small set of robot demonstrations of fold-towel instead of human demonstrations—while π_{lo} remains identical and has never seen towel-folding data—final success increases from 40% to 90%. This indicates that π_{lo} generalizes zero-shot to novel object categories when given accurate sub-goals, and that the principal bottleneck for OOD generalization is the high-level visual domain gap rather than the low-level controller. Existing work on embodiment-invariant visual representations [27, 22] aims to close precisely this gap and represents an important direction for future work.

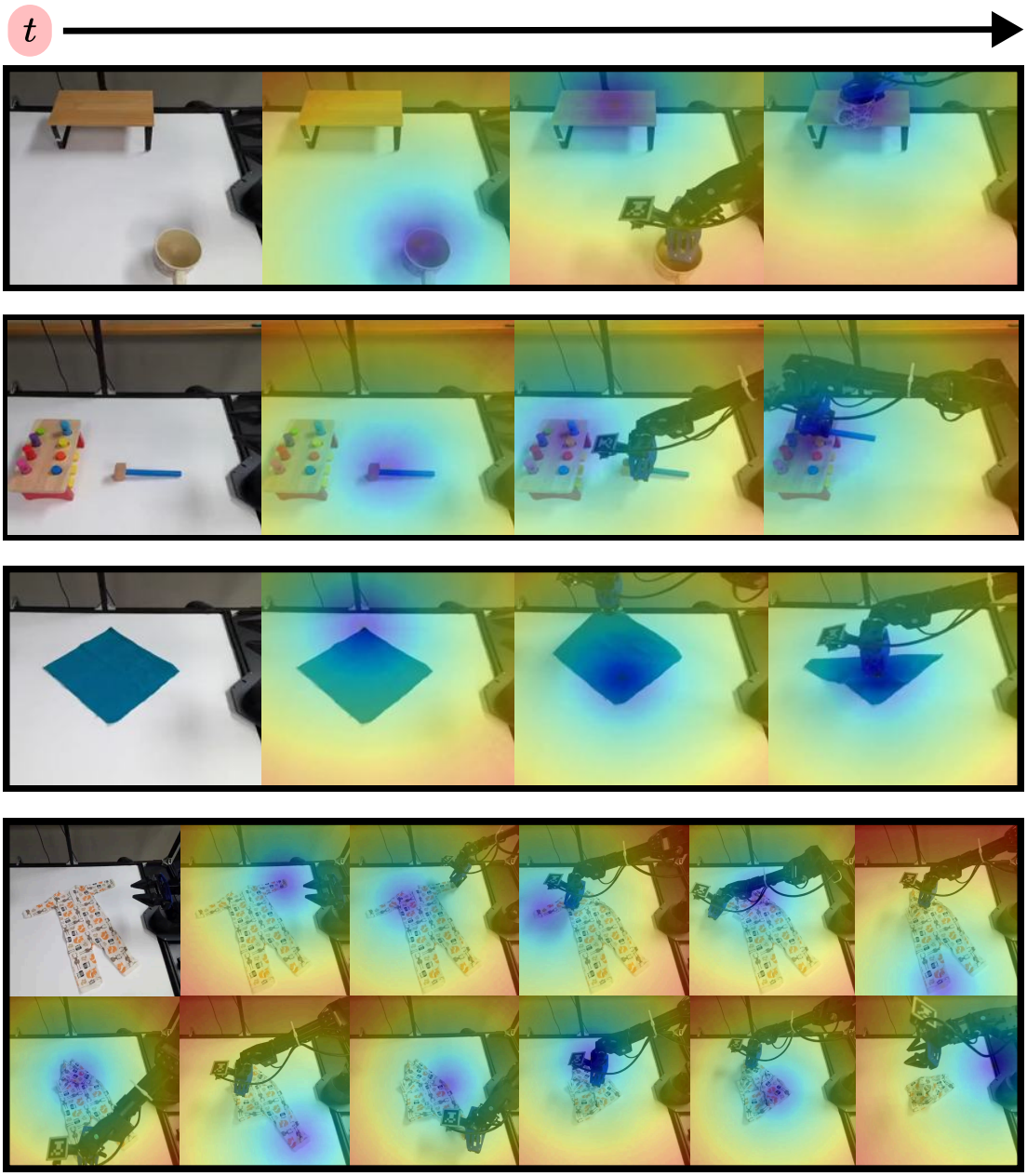

Fig. 8: We visualize qualitative results on various tasks with the **GHOST** framework. **Row 1**: `mug-on-table` (novel object combination). **Row 2**: `hammer-pin`. **Row 3**: `fold-towel` (novel category + skill composition). **Row 4**: `fold-onesie-ood` (novel instance). We visualize rollouts by overlaying a colormap on the projection of π_{hi}'s goal predictions at each timestep.

VII. Conclusion

We present GHOST, a hierarchical imitation learning framework that decouples embodiment-agnostic goal prediction from embodiment-specific action execution. This factorization enables two key benefits: improved in-distribution performance through explicit sub-goal modeling, and out-of-distribution generalization via human demonstrations.

We introduce a 3D goal prediction architecture that processes multi-view RGB-D observations to predict dense per-patch Gaussian mixture models over end-effector poses. We condition the low-level policy through 2D projections of 3D goals, represented as end-effector heatmaps. This representation bridges the gap between human and robot demonstrations without requiring explicit action retargeting. By training a high-level policy on heterogeneous human and robot data and a low-level policy purely on robot demonstrations, we generalize learned skills across novel object-context combinations and compositional skill generalization.

ACKNOWLEDGMENTS

We would like to thank Alexis Hao, Mino Nakura, Kallol Saha and Pratik Bhowal for helpful discussions and assistance with data collection. We thank the members of the R-PAD lab for their feedback. This material is based on work supported by the Toyota Research Institute, the National Science Foundation under NSF CAREER Grant No. IIS-2046491, and an unrestricted gift from Google.

REFERENCES

[1] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *CoRR*, 2024.

[2] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[3] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[4] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[5] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024. URL http://jmlr.org/papers/v25/23-0870.html.

[6] Jeremy A Collins, Loránd Cheng, Kunal Aneja, Albert Wilcox, Benjamin Joffe, and Animesh Garg. Amplify: Actionless motion priors for robot learning from videos. *arXiv preprint arXiv:2506.14198*, 2025.

[7] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers, 2024. URL https://arxiv.org/abs/2309.16588.

[8] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[9] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

[10] Siddhant Haldar and Lerrel Pinto. Point policy: Unifying observations and actions with key points for robot manipulation. *arXiv preprint arXiv:2502.20391*, 2025.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. Lvsm: A large view synthesis model with minimal 3d inductive bias. *arXiv preprint arXiv:2410.17242*, 2024.

[13] Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. Egomimic: Scaling imitation learning via egocentric video. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13226–13233. IEEE, 2025.

[14] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[15] Marion Lepert, Jiaying Fang, and Jeannette Bohg. Phantom: Training robots without robots using only human videos. *arXiv preprint arXiv:2503.00779*, 2025.

[16] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. Pmlr, 2020.

[17] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Silvio Savarese, and Li Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *arXiv preprint arXiv:2003.06085*, 2020.

[18] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya 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)*, pages 6892–6903. IEEE, 2024.

[19] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3d with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9826–9836, 2024.

[20] Karl Pertsch, Oleh Rybkin, Frederik Ebert, Shenghao Zhou, Dinesh Jayaraman, Chelsea Finn, and Sergey

Levine. Long-horizon visual planning with goal-conditioned hierarchical predictors. *Advances in Neural Information Processing Systems*, 33:17321–17333, 2020.

[21] Rolandos Alexandros Potamias, Jinglei Zhang, Jiankang Deng, and Stefanos Zafeiriou. Wilor: End-to-end 3d hand localization and reconstruction in-the-wild. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12242–12254, 2025.

[22] Juntao Ren, Priya Sundaresan, Dorsa Sadigh, Sanjiban Choudhury, and Jeannette Bohg. Motion tracks: A unified representation for human-robot transfer in few-shot imitation learning. *arXiv preprint arXiv:2501.06994*, 2025.

[23] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

[24] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), November 2017.

[25] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3, 2025. URL https://arxiv.org/abs/2508.10104.

[26] TRI LBM Team, Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, Naveen Kuppuswamy, Kuan-Hui Lee, Katherine Liu, Dale McConachie, Ian McMahon, Haruki Nishimura, Calder Phillips-Grafflin, Charles Richter, Paarth Shah, Krishnan Srinivasan, Blake Wulfe, Chen Xu, Mengchao Zhang, Alex Alspach, Maya Angeles, Kushal Arora, Vitor Campagnolo Guizilini, Alejandro Castro, Dian Chen, Ting-Sheng Chu, Sam Creasey, Sean Curtis, Richard Denitto, Emma Dixon, Eric Dusel, Matthew Ferreira, Aimee Goncalves, Grant Gould, Damrong Guoy, Swati Gupta, Xuchen Han, Kyle Hatch, Brendan Hathaway, Allison Henry, Hillel Hochsztein, Phoebe Horgan, Shun Iwase, Donovon Jackson, Siddharth Karamcheti, Sedrick Keh, Joseph Masterjohn, Jean Mercat, Patrick Miller, Paul Mitiguy, Tony Nguyen, Jeremy Nimmer, Yuki Noguchi, Reko Ong, Aykut Onol, Owen Pfannenstiehl, Richard Poyner, Leticia Priebe Mendes Rocha, Gordon Richardson, Christopher Rodriguez, Derick Seale, Michael Sherman, Mariah Smith-Jones, David Tago, Pavel Tokmakov, Matthew Tran, Basile Van Hoorick, Igor Vasiljevic, Sergey Zakharov, Mark Zolotas, Rares Ambrus, Kerri Fetzer-Borelli, Benjamin Burchfiel, Hadas Kress-Gazit, Siyuan Feng, Stacie Ford, and Russ Tedrake. A careful examination of large behavior models for multitask dexterous manipulation. 2025. URL https://arxiv.org/abs/2507.05331.

[27] Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Anima Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[28] Yufei Wang, Ziyu Wang, Mino Nakura, Pratik Bhowal, Chia-Liang Kuo, Yi-Ting Chen, Zackory Erickson, and David Held. Articubot: Learning universal articulated object manipulation policy via large scale simulation. *arXiv preprint arXiv:2503.03045*, 2025.

[29] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

[30] Danfei Xu, Suraj Nair, Yuke Zhu, Julian Gao, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Neural task programming: Learning to generalize across hierarchical tasks. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3795–3802. IEEE, 2018.

[31] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

[32] Yufei Ye, Yao Feng, Omid Taheri, Haiwen Feng, Shubham Tulsiani, and Michael J Black. Predicting 4d hand trajectory from monocular videos. *arXiv preprint arXiv:2501.08329*, 2025.

[33] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

[34] Zichen Zhang, Yunshuang Li, Osbert Bastani, Abhishek Gupta, Dinesh Jayaraman, Yecheng Jason Ma, and Luca Weihs. Universal visual decomposer: Long-horizon manipulation made easy. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6973–6980. IEEE, 2024.

[35] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[36] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5745–5753, 2019.

APPENDIX A
IMPLEMENTATION DETAILS

A. High-Level Policy

The high-level policy uses a decoder-only transformer operating on tokens extracted from a frozen DINOv3 [25] backbone. Each image is randomly cropped and resized, with additional color jitter, Gaussian blur, and grayscale augmentations applied stochastically. We also randomly drop a subset of DINOv3 image tokens as a regularization strategy. Further, we also apply uniform noise to the gripper token to be robust to noisy estimates.

After extracting patch tokens from DINOv3, we augment each token with a learned 3D positional embedding. Specifically, we unproject the center pixel of each patch of the depth image using camera parameters, and pass them through an MLP to produce the positional features.

The sub-goal distribution is parameterized as a dense GMM over all image patches. We train using multiple negative log-likelihood loss terms, with multiple fixed variances or using uniform mixing weights for the predicted distribution. Full hyperparameters are listed in Table VI.

B. Low-Level Policy

We adopt Diffusion Policy [4] (DP) as the low-level controller, using a ResNet-18 [11] visual backbone trained from scratch with color jitter and random crop augmentations. The goal-conditioned variant receives the end-effector heatmap as input and does not use language conditioning. The baseline without goal-conditioning (*i.e.,* DP) encodes language instructions with SigLIP [33] and concatenates the resulting features to form the global conditioning vector. We use receding horizon control as in the original Diffusion Policy implementation with the hyperparameters described in Table VII.

C. Hyperparameters

TABLE VI: High-level policy hyperparameters.

Hyperparameter	Value
Transformer	
Number of layers	4
3D Positional Features dim	128
Dropout	0.1
Number of register tokens	4
Dense Per-Patch GMM	
Fixed variances	`[0.01, 0.05, 0.1, 0.25, 0.5]`
Uniform weights coefficient	0.1
Data Augmentation	
Crop–resize probability	0.3
Crop size range	[192, 224]
Color jitter probability	0.3
Grayscale probability	0.1
Gaussian blur probability	0.1
Training	
Epochs	100
Learning Rate	1×10^{-4}
LR Scheduler	Cosine (100 warmup steps)
Optimizer	AdamW (β_1=0.95, β_2=0.999)
Batch Size	128

TABLE VII: Low-level policy hyperparameters.

Hyperparameter	Value
Observation / Action	
Observation horizon	2
Prediction horizon	16
Action horizon	8
Vision Backbone	
Architecture	ResNet-18
Crop size	700×700
Crop jitter	30
Spatial softmax keypoints	32
U-Net	
Down dimensions	`[512, 1024, 2048]`
Kernel size	5
Number of groups	8
Diffusion step embedding dim	128
Noise Scheduler	
Type	DDPM
Training timesteps	100
Beta schedule	`squaredcos_cap_v2`
Prediction type	ϵ
Clip sample range	$[-1.0, 1.0]$
Training	
Learning rate	1×10^{-4}
Optimizer	AdamW (β_1=0.95, β_2=0.999)
Weight decay	1×10^{-6}
LR scheduler	Cosine (500 warmup steps)
Batch Size	4

APPENDIX B
HARDWARE AND DATA COLLECTION SETUP

We make use of the ALOHA [35] robot platform, a bimanual robot consisting of WidowX arms for teleop (leader) and ViperX arms for manipulation (follower). However, we only make use of the right leader–follower pair; the left arm remains stationary throughout data collection and policy rollouts. All demonstrations are recorded at 30 Hz from two Azure Kinect workspace cameras at a resolution of 1280×720 and then downsampled to 15 Hz for policy learning. We also make use of an Intel Realsense wrist-mounted camera to augment the robot demonstrations. Rollouts are executed at 15 Hz. Actions are represented as 10-dimensional absolute end-effector poses (3 position, 6 for rotation in the 6D representation [36] and one for gripper width) for the single active arm.

We use standard ArUco-marker-based extrinsics calibration for the workspace cameras, and apply photometric and geometric augmentations during training of both π_{hi} and π_{lo}. We note that GHOST uses 3D information sparsely: π_{hi} unprojects only the center pixel of each DINOv3 patch, and π_{lo} conditions on heatmaps of a small set of projected keypoints rather than on dense point clouds. This sparse use of 3D reduces the surface area over which calibration or depth error can affect the policy compared to methods that consume dense point clouds or per-pixel 3D features.

APPENDIX C
TASK DESCRIPTIONS

A. Pick-and-Place

All pick-and-place tasks share a common workspace layout: the target object is placed at a randomized position on the table, while the receptacle (bin or toy table) is fixed at the center-back.

- **`mug-in-bin`**: 3 mugs of distinct color and shape. 20 robot demonstrations distributed across mugs.
- **`plate-on-table`**: 3 plates of distinct color. 20 robot demonstrations distributed across plates.
- **`plate-in-bin`**: Same 3 plates as above. 20 robot demonstrations distributed across plates.
- **`mug-on-table`**: Same 3 mugs as `mug-in-bin`. 20 human demonstrations distributed across mugs.

B. Cloth Folding

All cloth folding tasks place the garment at a random position and orientation on the table.

- **`fold-onesie`**: 2 onesies of different color. 33 robot demonstrations distributed evenly.
- **`fold-shirt`**: 2 shirts of distinct color. 50 robot demonstrations distributed evenly.
- **`fold-onesie-ood`**: A held-out onesie not seen during robot training. 17 human demonstrations.
- **`fold-towel`**: 3 towels of distinct color. 50 human demonstrations distributed evenly.

C. Hammer Pin

- **`hammer-pin`**: The hammer is placed at a random position; the pin board is placed near the table center. We collect robot demonstrations of striking the blue and pink pins, and human demonstrations of striking purple and blue+purple pins. We test with the blue pin for evaluation. 25 demonstrations for each of the 4 datasets.

APPENDIX D
ABLATION: VISUAL BACKBONE FOR HIGH-LEVEL POLICY

We ablate the choice of visual backbone by replacing the frozen DINOv3 encoder in π_{hi} with a tiny ViT (11M parameters, comparable to ResNet-18) trained from scratch. This isolates whether GHOST's gains stem from the pre-trained visual encoder or from the hierarchical factorization itself. We evaluate on **`fold-onesie`**, our most demanding in-distribution task.

	Success Rate (%) ↑				
Method	1-step	2-step	3-step	4-step	Final
DP	80	60	50	40	20
GHOST (ViT)	**100**	**100**	90	70	50
GHOST (DINOv3)	**100**	**100**	**90**	**90**	**100**

TABLE VIII: Ablation on the high-level visual backbone for **`fold-onesie`**. GHOST (ViT) uses a tiny ViT trained from scratch in place of the frozen DINOv3 encoder. Best: **shaded**, second-best: underline. $n = 10$ rollouts.

As shown in Table VIII, GHOST with a ViT trained from scratch (50% final success) still substantially outperforms the flat Diffusion Policy baseline (20%), confirming that the hierarchical sub-goal factorization is itself a significant contributor to performance. However, the frozen DINOv3 backbone provides a further large improvement (90%), indicating that a strong pre-trained visual encoder is important for accurate goal prediction, particularly over longer horizons where errors in π_{hi} compound. Thus, both the hierarchy and the visual backbone contribute meaningfully to GHOST's performance.

APPENDIX E
ABLATION: ORACLE HIGH-LEVEL POLICY

To disentangle the contribution of goal prediction quality (from human demonstrations) and goal-following ability, we collect an additional 25 robot teleoperation demonstrations of **`fold-towel`** and train the high-level policy on this data instead of human demonstrations. This gives us an "oracle" π_{hi} that operates without any embodiment gap, while π_{lo} remains identical to all other experiments—crucially, it has never seen **`fold-towel`** during training.

Method	Success Rate (%) ↑
MimicPlay	20
GHOST (human π_{hi})	40
GHOST (oracle π_{hi})	**90**

TABLE IX: Oracle high-level ablation on **`fold-towel`**. The oracle π_{hi} is trained on robot demonstrations; π_{lo} is unchanged and has never seen towel-folding data.

Replacing human demonstrations with robot data in π_{hi} increases success from 40% to 90%, while π_{lo} is held fixed. This reveals two things. First, the low-level policy can generalize zero-shot to a novel object category (towels) when given accurate sub-goals, confirming that the goal-conditioned factorization enables meaningful skill transfer. Second, the primary bottleneck for OOD generalization is the visual domain gap between human and robot observations in π_{hi}, rather than the low-level controller's ability to execute on unseen objects. Closing this embodiment gap in the high-level policy is an important direction for future work.

APPENDIX F
COMPUTATIONAL COST

We report wall-clock inference time for each method, averaged over 5 rollouts on a single NVIDIA RTX 4090 GPU. All methods predict an action chunk of length 16.

Method	**Inference time (ms / step)**
Diffusion Policy	60.1 ± 0.8
MimicPlay	127.8 ± 1.2
GHOST ($\pi_{hi} + \pi_{lo}$)	149.3 ± 2.1

TABLE X: Per-step inference time (mean $\pm$ std) on a single RTX 4090.