

# Emergent Active Perception and Dexterity of Simulated Humanoids from Visual Reinforcement Learning

Zhengyi Luo<sup>1,2 \*</sup> Chen Tessler<sup>1,2 \*</sup> Toru Lin<sup>1,3</sup> Ye Yuan<sup>1</sup> Tairan He<sup>1,2</sup> Wenli Xiao<sup>1,2</sup>

Yunrong Guo<sup>1</sup> Gal Chechik<sup>1</sup> Kris Kitani<sup>2</sup> Linxi Fan<sup>1 †</sup> Yuke Zhu<sup>1 †</sup>

<sup>1</sup>Nvidia; <sup>2</sup>Carnegie Mellon University; <sup>3</sup>University of California, Berkeley

<https://zhengyiluo.github.io/PDC>



Figure 1. Perceptive Dexterous Control (PDC) enables a humanoid equipped with egocentric vision to search for, reach, grasp, and manipulate objects in cluttered kitchen scenes. We use visual perception as the sole interface for indicating which hand to use, which object to grasp, where to move, and which drawer to open.

## Abstract

*Human behavior is fundamentally shaped by visual perception — our ability to interact with the world depends on actively gathering relevant information and adapting our movements accordingly. Behaviors like searching for objects, reaching, and hand-eye coordination naturally emerge from the structure of our sensory system. Inspired by these principles, we introduce Perceptive Dexterous Control (PDC), a framework for vision-driven dexterous whole-body control with simulated humanoids. PDC operates solely on egocentric vision for task specification, enabling object search, target placement, and skill selection through visual cues, without relying on privileged state information (e.g., 3D object positions and geometries). This perception-as-interface paradigm enables learning a single policy to perform multiple household tasks, including reaching, grasping, placing, and articulated object manipulation. We also show that training from scratch with reinforcement learning can produce emergent behaviors such as active search. These results demonstrate how vision-driven control and complex tasks induce human-like behaviors and can serve as the key ingredients in closing the perception-action loop for animation, robotics, and embodied AI.*

## 1. Introduction

Human interaction with the world is fundamentally governed by active perception — a continuous process where vision, touch, and proprioception dynamically guide our movements and interactions [43, 44]. In this work, we study how a simulated humanoid, equipped only with egocentric perception, can perform tasks such as object transportation and articulated object manipulation in diverse household scenes. The combination of high-dimensional visual input and high-degree-of-freedom humanoid control makes this problem computationally demanding. As a result, humanoid controllers in animation and robotics often rely on privileged state information (e.g., precise 3D object pose and shape), sidestepping the challenges of processing noisy visual input. This raises a key question: How can we effectively close the egocentric perception-action loop to enable humanoids to interact with their environment using only their egocentric sensory capabilities?

Perception-driven humanoid control presents unique challenges because it involves two computationally intensive problems: dexterous whole-body control and visual perception in natural environments. While locomotion demands precise coordination of balance, limb movements, and dexterity, incorporating vision introduces additional complexities of partial observability and

\*Equal contribution, † GEAR Team Leads

the need for active information gathering. Due to these challenges, prior work [4, 30, 64] has relied mainly on privileged object-state information to achieve humanoid-object interaction, bypassing the difficulties of noisy sensory input and gaze control. While using privileged information provides a compact, pre-processed representation of the environment that includes objects outside the field of view, it fundamentally limits the emergence of human-like behaviors such as visual search since the humanoid has complete knowledge of object locations at all times. In contrast, learning from visual observations eliminates this limitation but introduces several key challenges: the policy must learn to handle partial observability and tackle the increased input dimensionality (such as  $128 \times 128 \times 3$  RGB images) — which is orders of magnitude larger than state representations. These challenges exacerbate the significant sample efficiency issues in reinforcement learning (RL) for whole-body humanoid control.

Another challenge in developing vision-driven humanoids is designing an input representation that facilitates learning multiple tasks simultaneously. Consider a kitchen cleaning scenario: the humanoid must identify specific objects in cluttered environments, understand task commands, and precisely place items in designated locations. Each of these steps requires clear task instructions. While natural language offers flexibility as an interface, it often lacks the precision required for exact object selection and placement in physical space. Previous approaches, such as Catch & Carry [38], have combined vision-based policies with phase variables, but this design limits scalability since each new task requires retraining the policy to handle new task variables. The ideal input representation should be task-agnostic, allowing the policy to learn in diverse manipulation scenarios without task-specific modifications.

To address these challenges, we introduce Perceptive Dexterous Control (PDC), a framework for learning human-like whole-body behaviors of simulated humanoids through perception-driven control. PDC controls humanoids to act solely from visual input (RGB, RGB-D, or Stereo) and proprioception. Rather than using state variables (*e.g.*, task phases, object information) to encode tasks, PDC specifies tasks directly through perception. Similarly to how augmented reality enhances human vision via informative markers and visual cues, we enhance the robot’s vision using object masks and 3D markers to select objects and specify target locations. This vision-centric approach supports continuously adapting the agent’s skills to new tasks and complex scenes. To tackle the challenging whole-body dexterous control task, we leverage control priors learned from large-scale motion capture data. Combined with household settings and dexterous hand tasks, these priors enable RL policies to develop emergent human-like behaviors — including active search and whole-body coordination.

To summarize, our contributions are as follows: (1) We demonstrate the feasibility of vision-driven, whole-body dexterous humanoid control in naturalistic household environments, a novel and challenging task that involves both perception and complex motor control; (2) We introduce a vision-driven task specification paradigm that uses visual cues for object selection, target placement, and skill specification, eliminating the need for predefined state variables; (3) We show that human-like behaviors, including active search and whole-body coordination, emerge from our vision-driven approach. Extensive evaluation shows how different visual modalities (RGB, RGBD, Stereo) and reward designs impact behaviors and task performance (*e.g.*, stereo outperforms RGB by 9% in success rate).

## 2. Related Work

**Visual Dexterous Manipulation.** Learning visual dexterous policies for simulated [63, 72] and real-world [5–7, 24–26, 29, 50–52, 56, 57, 68] dexterous hands has gained increasing attention due to its relevance to real-world tasks such as object picking and manipulation. These approaches can be roughly divided into RL [26, 29, 57, 72] and behavior cloning [7, 25, 52] methods. PDC falls under RL, where the dexterous hands’ behavior is learned through trial and error. Among RL methods, a popular paradigm is to first learn a state-space policy using privileged object information from simulation and then distill it into vision-based policies [29, 57, 63, 72]. However, to successfully distill, these methods assume a fixed camera pointing at a stationary table and do not involve active perception. For mobile and dexterous humanoids that loco-manipulate, where to look and how to pick up objects remain open questions.

**Whole Body Loco-Manipulation.** Controlling simulated [3, 4, 14, 30, 38, 62, 64, 67, 70, 71] and real humanoids [16, 28] to loco-manipulate requires precise coordination of the lower and upper body. Some prior work adopts a reference motion imitation approach [16, 28, 64, 70], where the humanoid learns from reference human-object interactions. Others follow a generative approach [3, 14, 30, 62, 67, 71], where the humanoid is provided high-level objectives (such as moving boxes [62], grasping objects [30], *etc.*) to complete but not a reference motion to follow. Closely related to ours, Omnigrasp [30] studies a state-space policy to grasp diverse objects using either hand (predominantly the right hand). HumanVLA [71] learns a language-conditioned visual policy for room arrangement via distillation. Compared to PDC, HumanVLA does not use dexterous hands and always assumes the object of interest and target location are within view of the humanoid during initialization. Similarly to our setting of egocentric vision, Catch & Carry [38] learns a vision-conditioned policy for carrying boxes and catching balls and uses phase variables and privileged infor-



Figure 2. **Kitchens:** Our agent is trained in parallel on a large set of (randomly) procedurally generated kitchens. Each generated kitchen is structurally and visually different. Objects are spawned in random locations on the counter, and the agent starts in a random position and orientation within the scene. This diversity encourages the agent to learn general behaviors, such as search, and robust interaction.

mation (object position for carrying). Compared to Catch & Carry, we involve dexterous hands, learn multiple tasks, and do not use privileged information.

**Perception-as-Interface.** The interface theorem of perception [17] argues that perception functions as an interface for guiding useful actions instead of reconstructing the 3D world. Recently, many [18, 41, 69] have explored using visual signals as indicators for policies to imbue [41] common sense or guide actions [69]. In games and augmented reality (AR), 3D markers and visual signals are also constantly used to indicate actions or give instructions. Building upon this idea, we leverage the redundant nature of images and use visual cues to replace state input for our RL policy.

**Hierarchical Reinforcement Learning.** The options framework, also known as skills [58], provides an abstraction layer for RL agents, enabling the separation between high-level planning and low-level execution [13, 29, 37, 46, 57, 61]. Reusable latent models have recently emerged as a promising hierarchical approach for humanoid control. First, a low-level latent-conditioned controller (LLC) is trained to generate a diverse repertoire of motions [32, 48, 60, 66]. Then, a high-level controller (HLC) is trained to utilize the LLC by predicting latents, effectively reusing the learned skills. This hierarchical structure has been used in complex scenarios involving humanoid-object interactions [4, 30], where the separation between high-level task planning and low-level motion execution can significantly reduce the learning complexity.

**Lifelong Learning.** Interactive agents face the challenge of unpredictable operational environments. The framework of lifelong learning addresses this by continuously acquiring and transferring knowledge across multiple tasks sequentially over the agent’s lifetime [55]. In reinforcement learning, lifelong learning approaches have demonstrated superior performance in complex tasks through knowledge reuse and retention [1, 36, 42, 61]. Our work encodes all tasks via visual cues, enabling the policy to learn new tasks through fine-tuning without architectural modifications. We show that this capability is crucial and allows the PDC to master simple skills before adapting to diverse, complex scenes.

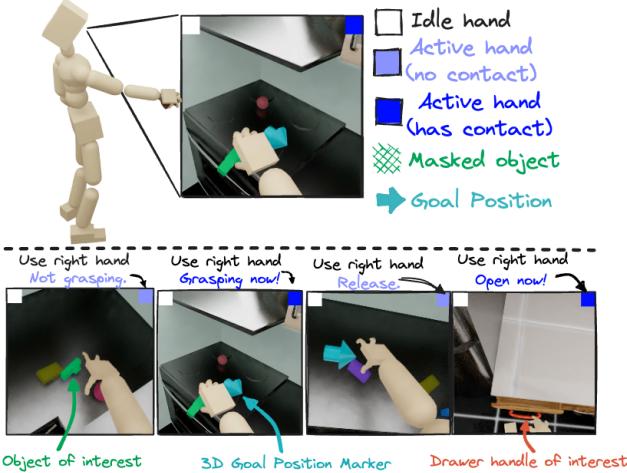
### 3. Method

To tackle the problem of visual dexterous humanoid control, we train a policy to output joint actuation based on current visual and proprioceptive observations. In Section 3.1, we define the task settings of our visual dexterous control problem. In Section 3.2, we describe our perception-as-interface design to enable vision-only task learning. Finally, in Section 3.3, we detail how we learn our control policy.

#### 3.1. Task Definitions

We study two scenarios for visual dexterous whole-body control: (1) a tabletop setting where the humanoid manipulates objects on an isolated table floating in midair, and (2) a naturalistic kitchen environment with diverse objects, randomized placements, and articulated cabinets, as visualized in Fig. 2. The tabletop scenario, inspired by Omnigrasp [30], is devoid of complex visual backgrounds, providing a controlled training environment for visual-dexterous grasp learning. Here, the humanoid is initialized facing the table with one object on the table. The policy is trained to approach the table, pick up an object, move it to a designated 3D location, and place it down. The kitchen scene presents a significantly more challenging setting, featuring a photorealistic background, distractor objects, and articulated drawers. In this environment, we consider both object transport and drawer manipulation. In the object transport task, the object position is unknown, and the humanoid must first locate the target object, pick it up, transport it to the specified 3D location, and release it. In the articulated cabinets task, the agent is tasked with opening and closing specified cabinets on command.

**Scene Generation.** To create diverse environments, we leverage SceneSynthesizer [12] to generate six distinct kitchen types (galley, flat, L-shaped, with island, U-shaped, and peninsula). For each type, we produce 80 randomized configurations with variations in both structural elements (*e.g.*, cabinet quantity and placement) and surface textures. We pre-compute 20 valid humanoid spawning positions and 64 object placement locations within each generated scene, ensuring all elements are positioned without geometric intersections or scene penetrations. We create 512 environments for training and 64 for testing.



**Figure 3. Perception-as-Interface:** PDC instructs the policy through visual signals. We overlay the object of interest using a green mask, use a 3D blue marker to indicate target location, and use colored 2D squares (top corners) to inform the agent which hand to use and when to grasp and release.

### 3.2. The Visual Perception Interface

To tackle the task of object search, grasp, goal reaching, release, and drawer opening, prior art resorts to phase variables [38] or modular state machines [19] to decide which part of the task the policy should execute on. While adding phase variables, object information, and task instructions such as 3D goal coordinates is helpful for state-space policy training, it limits the possibility of learning new tasks. Any change in the task requires re-training the policy to accommodate new state variables. We propose to use perception as the sole interface for task specification. Leveraging vision as a unified interface can support agents that can be adapted (through fine-tuning) to new tasks without any architectural changes. In an extreme example, visual signals could even be floating text for precise task instructions. Our approach embraces this principle, using visual indicators to specify tasks, as illustrated in Fig. 3.

**Object Selection.** The kitchen scenes contain multiple objects, making object specification a significant challenge, particularly when precise location information is unavailable. For instance, selecting a particular apple from a cluster becomes ambiguous without explicit spatial coordinates. To address this challenge, we leverage 2D semantic segmentation [21] as a visual selection mechanism. Our approach applies alpha blending using the segmentation mask to highlight the target object with a distinctive bright green overlay directly **in the image space**. This visual differentiation enables the agent to identify the correct object and transform the object selection problem into a visually guided task. To differentiate between object grasping and drawer opening, we paint drawer handles bright red.

**Object Reaching Goals.** Once the object is picked up, the next step is specifying its destination. While 3D target locations can specify goals, such input will become redundant for tasks like object search or cabinet opening and lacks adaptability to new scenarios. To overcome these limitations, we instead *render* a 3D arrow directly in the agent’s visual field to indicate the desired destination. This interface resembles information markers commonly used in 3D games or AR applications and provides an intuitive and visually grounded objective for manipulation tasks.

**Handedness, Pickup Time, and Release Time.** We train our agent for bimanual control, enabling it to use the left hand, right hand, or both hands on command. Additionally, the humanoid must be able to pick up and place objects as instructed. To achieve this, we color the visual field’s upper right and left corners to indicate which hand should grasp the object. A small colored block indicates hand usage: white signals the hand should remain idle, purple designates a hand that should be used for contact but should not currently be in contact, and blue signifies active contact now (grasping). The purple signal guides the humanoid in determining which hand to use to reach out to the object.

We believe that perception-as-interface [17] offers near-infinite extensibility for vision-based agents, with an extreme case being using floating text to convey instructions.

### 3.3. Learning Perceptive Dexterous Control

We formulate our task using the general framework of goal-conditioned RL. The learning task is formulated as a Partially Observed Markov Decision Process [2, POMDP] defined by the tuple  $\mathcal{M} = \langle \mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$  of states, observations, actions, transition dynamics, reward function, and discount factor. The simulation determines the state  $s_t \in \mathcal{S}$  and transition dynamics  $\mathcal{T}$ , and a policy computes the action  $a_t \in \mathcal{A}$  based on the partial observation  $o_t \in \mathcal{O}$ . The observation  $o_t \in \mathcal{O}$  is a partial representation of the full simulation state  $s_t$ , combined with the task instructions. We train a control policy  $\pi_{\text{PDC}}(a_t | o_t)$  to compute joint actuation  $a_t$  from observations. The reward function  $r_t = \mathcal{R}(s_t, a_t, s_{t+1})$  is defined using the full simulation state  $s_t$  to guide learning. The policy optimizes the expected discounted reward  $\mathbb{E} \left[ \sum_{t=1}^T \gamma^{t-1} r_t \right]$  and is trained using proximal policy optimization (PPO) [54].

**Observation.** The policy observes the proprioception (the observation of oneself) and camera image  $o_t \triangleq (o_t^p, I_t)$ . The proprioception  $o_t^p \triangleq (\theta_t, p_t, v_t, \omega_t, c_t)$  consists of the 3D joint rotation  $\theta_t \in \mathbb{R}^{J \times 3}$ , position  $p_t \in \mathbb{R}^{J \times 3}$ , linear velocity  $v_t \in \mathbb{R}^{J \times 3}$ , angular velocity  $\omega_t \in \mathbb{R}^{J \times 3}$ , and the contact forces on each hand  $c_t \in \mathbb{R}^{J_H \times 3}$ . Due to computational challenges posed by larger images, prior work focused on low-resolution inputs (*e.g.*,  $40 \times 30 \times 3$  [62]); our experiments demonstrate the perfor-

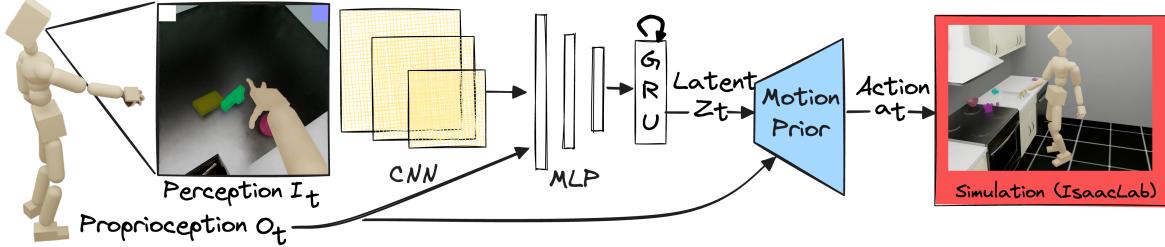


Figure 4. We use perception and proprioception as input to the network, processed by a simple CNN-GRU-MLP architecture.

mance benefits of higher-resolution visual inputs. We evaluate RGB ( $128 \times 128 \times 3$ ), RGBD ( $100 \times 100 \times 4$ ), and stereo ( $2 \times 80 \times 80 \times 3$ ), with the latter two using slightly smaller dimensions due to GPU memory constraints.

**Reward.** To train PDC, we provide it with dense rewards that help and guide its behavior, as RL agents struggle when trained with sparse objectives (e.g.,  $r_t = \mathbf{1}_{\text{object on marker}}$ ). For the grasping task, we designed two rewards: a reward aimed to measure grasp and object pose, and another (optional) to guide the gaze. The grasp reward is:

$$r_t^{\text{PDC}} = \begin{cases} r_t^{\text{approach}} + r_t^{\text{lookat}}, & \|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2 > 0.2 \text{ and } t < \lambda_{\text{start}} \\ r_t^{\text{pre-grasp}} + r_t^{\text{lookat}}, & \|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2 \leq 0.2 \text{ and } t < \lambda_{\text{start}} \\ r_t^{\text{obj}} + r_t^{\text{lookat}}, & \lambda_{\text{start}} \leq t < \lambda_{\text{end}} \\ (1 - \mathbf{1}_{\text{has-contact}}), & t \geq \lambda_{\text{end}} \end{cases} \quad (1)$$

This reward is split into 4 sequential segments: a time scheduler (provided by the user) determines whether the agent should be in the approach, grasp, or release phase. The pre-grasp phase is broken into 2 segments, reach and pre-grasp. The pre-grasp [8, 30] is provided when the hand is  $< 0.2m$  from the target object. This encourages the hand to match a pre-computed pre-grasp  $\hat{\mathbf{q}}^{\text{pre-grasp}} \triangleq (\hat{\mathbf{p}}^{\text{pre-grasp}}, \hat{\theta}^{\text{pre-grasp}})$ , a single hand pose consisting of the hand translation  $\hat{\mathbf{p}}^{\text{pre-grasp}}$  and rotation  $\hat{\theta}^{\text{pre-grasp}}$ . If the object is successfully grasped, we switch to the object’s 3D location reward  $r_t^{\text{obj}}$  to encourage transporting the object to specific locations. After the grasping should end ( $t \geq \lambda_{\text{end}}$ ), we encourage the agent to release the object:  $\mathbf{1}_{\text{has-contact}}$  determines if any hand has contact with the object. The optional look-at reward

$$r_t^{\text{lookat}} = \begin{cases} r_t^{\text{lookat-object}}, & \text{and } t < \lambda_{\text{start}} \\ r_t^{\text{lookat-marker}}, & \text{and } t \geq \lambda_{\text{start}} \end{cases}, \quad (2)$$

is also time-conditioned. First, the humanoid is rewarded for looking at the object. After the contact start time, the humanoid is encouraged to look at the 3D target marker. The look-at reward is computed as  $r_t^{\text{lookat}} = 1 - \sqrt{1 - (\mathbf{v}_{\text{gaze}} \cdot \mathbf{v}_{\text{eye-target}})}$ , where  $\mathbf{v}_{\text{gaze}}$  is the gaze vector and  $\mathbf{v}_{\text{eye-target}}$  is the eye to target (object or marker) vector. While we encourage the policy to always look in the object’s direction, search (looking left and right) emerges when the object is not always initialized in the view. To train the drawer open policy, we use the same pregrasp and

approach reward, and change the object reward position  $r_t^{\text{obj}}$  to the drawer opening reward  $r_t^{\text{drawer}}$ . For details about each reward term, please refer to Appendix C.3.

**Early Termination.** Equally important to reward design, early termination [47] also plays an important role in shaping the agent’s behavior as it provides a strong negative reward signal to the agent. We early terminate the training episode in the following scenarios: (1) the agent is not in contact with the object by  $\lambda_{\text{start}}$  grasping time, (2) the agent is still in contact of the object after contact end time  $\lambda_{\text{end}}$ , and (3) if the target object is more than 0.25m away from the marker for more than 2 seconds.

**Policy Architecture.** We use a simple CNN-GRU-MLP architecture for the policy, illustrated in Fig. 4. The CNN processes raw image inputs, extracting high-level spatial features. The GRU provides recurrent memory, enabling the agent to handle temporal dependencies of this partially observable task. Due to the large throughput of data when training with an image-conditioned RL policy, we use a simple three-layer CNN architecture that has been used in sim-to-real robotics settings [27]. For RGBD, the depth channel is treated as an additional image channel. For stereo, we use a Siamese network [22] and process the two images with the same CNN and concatenate the features. We also experiment with frozen pretrained visual encoders like ResNet [15] and ViT [10] that are pretrained on classification tasks, though they perform poorly in our setting.

**Humanoid Motion Representation.** Due to the difficulty of controlling a high-dimensional humanoid with dexterous hands, many prior works resort to reference motion imitation [64, 70] or overfitting to a single type of action [3]. We do not use any reference motion and utilize the recently proposed PULSE-X [30, 32] as the reusable, general-purpose, low-level controller. PULSE-X is a conditional VAE (cVAE) that is trained to reproduce motions from the AMASS MoCap dataset [34]. At a high level, PULSE-X learns a proprioception-conditioned latent space that can be decoded into human-like actions  $\mathcal{D}_{\text{PULSE-X}}(\mathbf{a}_t | \mathbf{z}_t^{\text{PDC}}, \mathbf{o}_t^{\text{P}})$ . Using PULSE-X, we define the action space of our policy with respect to the prior  $\mathbf{a}_t = \mathcal{D}_{\text{PULSE-X}}(\pi_{\text{PDC}}(\mathbf{z}_t^{\text{PDC}} | \mathbf{o}_t^{\text{P}}, \mathbf{I}_t), \mathbf{o}_t^{\text{P}})$  and train it using hierarchical RL. For more information about PULSE-X,

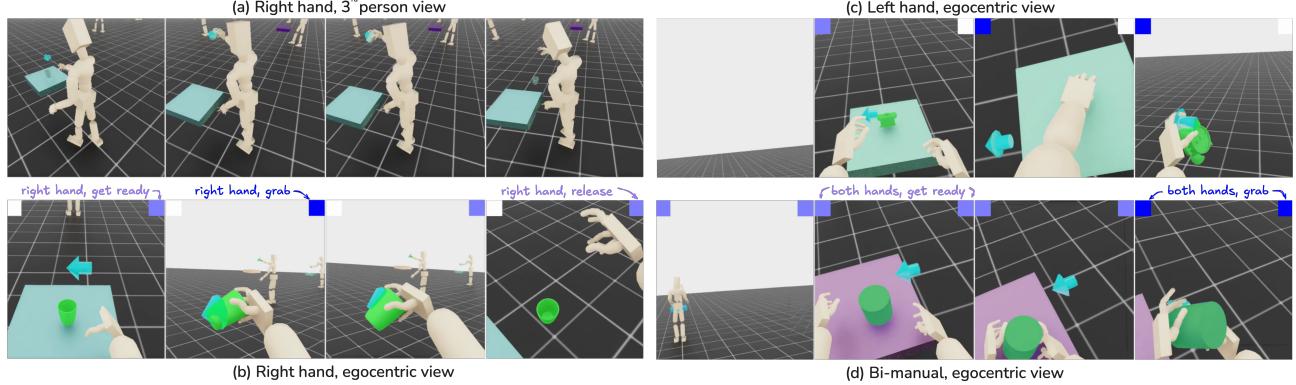


Figure 5. **Tabletop:** PDC is instructed directly from visual signals. A visual (top left and/or right) indicates in purple whether the corresponding hand should be prepared for contact. Changing to dark-blue indicates that contact should be made. Instructing the agent to use both hands enables it to transport larger objects. Changing the indicator back to purple instructs the agent to release the object.

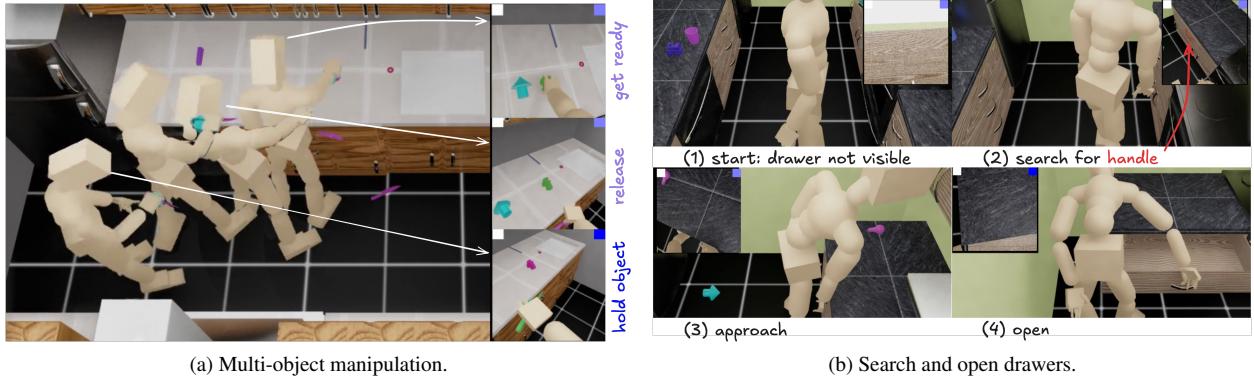


Figure 6. **Kitchen task:** Here, the agent must master multiple skills. First, the agent must search and find the objective. A target object is marked in green, whereas a target drawer handle in red. Once found, the agent must approach the target before it can interact with it.

please refer to Appendix. C.2.

**Tabletop training:** the humanoid is initialized at a random location facing (but not looking at) the object. The target goal position is sampled randomly above the tabletop and changes periodically. **Kitchen training:** the humanoid is initialized randomly in an empty space facing a random direction. Target object positions are sampled randomly on the support surface and changed periodically (see samples in [supplement site](#)). **Optimization:** Unlike imitation-based methods [64, 65, 70], we do not use any reference human motions during training. We follow the standard on-policy RL [54, PPO] training procedure, interleaving sample collection and policy updates. To successfully solve the kitchen scene, we warm-start the policy using the trained table-top policy. *This is enabled by our perception-as-interface paradigm, as we can reuse the grasping capability from the tabletop agent.* For the kitchen scene, half of the environments learn the grasping task while the other half opens drawers.

## 4. Experiments

**Baselines.** As few prior attempts at visual dexterous whole-body control exist, we compare against a modified version of a SOTA state-space policy, Omnigrasp [30]. The state-space policy always observes the object and target location and serves as the oracle policy. We extend the original Omnigrasp policy to support hand specification and placing down the object. The state-space policy is trained using the exact same reward as the visual controllers, including the look-at reward. Please refer to Appendix. C.1 for more information on this baseline.

**Implementation Details.** Simulation is conducted in NVIDIA IsaacLab [39] and rendered using tiled rendering. Due to the significantly increased data throughput for training visual RL, we simulate 192 environments per-GPU across 8 Nvidia L40 GPUs in parallel. Simulation is conducted at 120Hz while the control policy runs at 30Hz. The tabletop policy training takes around 5 days to converge. Fine-tuning a trained tabletop policy for the kitchen scene requires an additional 5 days. This results in approximately  $5 \times 10^9$  samples, roughly 4 years’ worth of experience.

Method	GRAB-Train (25 seen objects)								GRAB-Test (5 unseen objects)								
	Vision	Succ $\uparrow$	$E_{\text{pos}} \downarrow$	Succ <sub>right</sub> $\uparrow$	$E_{\text{pos}} \downarrow$	Succ <sub>left</sub> $\uparrow$	$E_{\text{pos}} \downarrow$	Succ <sub>bi</sub> $\uparrow$	$E_{\text{pos}} \downarrow$	Succ $\uparrow$	$E_{\text{pos}} \downarrow$	Succ <sub>right</sub> $\uparrow$	$E_{\text{pos}} \downarrow$	Succ <sub>left</sub> $\uparrow$	$E_{\text{pos}} \downarrow$	Succ <sub>bi</sub> $\uparrow$	$E_{\text{pos}} \downarrow$
Omnigrasp	$\times$	99.1%	8.9	99.1%	8.9	98.7%	13.0	99.7%	12.5	70.6%	11.2	70.4%	11.2	-	-	100.0%	18.7
PDC-RGB	$\checkmark$	87.5%	74.4	88.7%	72.6	83.0%	77.4	78.7%	98.3	90.1%	100.5	<b>90.0%</b>	101.0	-	-	<b>100.0%</b>	39.1
PDC-RGBD	$\checkmark$	86.9%	61.5	88.1%	60.7	80.1%	<b>68.6</b>	83.9%	<b>59.5</b>	85.2%	74.6	85.1%	74.8	-	-	<b>100.0%</b>	41.8
PDC-stereo	$\checkmark$	<b>96.4%</b>	<b>51.9</b>	<b>96.9%</b>	<b>47.7</b>	<b>92.9%</b>	79.2	<b>87.7%</b>	57.5	<b>91.8%</b>	<b>61.0</b>	85.8%	<b>61.1</b>	-	-	<b>100.0%</b>	<b>41.6</b>

Table 1. Quantitative results on visual object grasps and target goal reaching on the GRAB dataset. The training set contains 82.4% right hand, 12.1% left hand, and 5.4% both hands. The test set contains 99% right hand grasp and 1% both hands.

Method	GRAB-Train, Scene-Train			GRAB-Test, Scene-Train			GRAB-Test, Scene-Test			Drawer, Scene-Train		Drawer, Scene-Test				
	Search	Pick up	Goal Reaching	Search	Pick up	Goal Reaching	Search	Pick up	Goal Reaching	Search	Drawer Open	Search	Drawer Open			
PDC-Stereo	98.1%	85.1%	65.5%	162.2	98.9%	79.1%	52.8%	147.6	95.7%	80.2	53.6%	154.4	98.8%	63.7%	98.0%	64.0%

Table 2. Quantitative results on visual object grasps, target goal reaching, and drawer opening in the kitchen scene. We test on different combination of train/test scenes and train/test objects.

Our simulated humanoid follows the kinematic structure of SMPL-X [45] using the mean body shape [30, 31, 33]. This humanoid has  $J = 52$  joints, of which 51 are actuated. Between these joints, 21 joints belong to the body, and the remaining  $J_H = 30$  joints are for two hands. All joints are 3 degrees-of-freedom (DoF) spherical joints actuated with PD controllers, resulting in  $a_t \in \mathbb{R}^{51 \times 3}$ . To mimic human perception, we attach camera(s) to the head of the humanoid, roughly at the position of the eyes. For stereo, we place the cameras 6 cm apart, similar to human eye placements.

**Object Setups.** We use a subset of the GRAB [59] dataset (25 out of 45 objects) that contains household objects for training and keep the testing set (5 objects) for both the kitchen and tabletop setting. We obtain the pre-grasp for the reward and hand specification from the MoCap recordings provided by the GRAB dataset: *e.g.* if the pre-grasp uses both hands, the episode will use the corresponding hands. We also provide qualitative results on larger and more diverse objects on the OMOMO [23] and Oakink [73] datasets (see [supplement site](#)).

## 4.1. Results

As motion is best seen in videos, we strongly recommend that readers view the [qualitative videos](#).

**Metrics.** In the tabletop scenario, the agent is evaluated on its ability to pick up an object, track a marker located 30cm above the table, and then release the object. An episode is considered successful if the humanoid picks up the object using the correct hand, reaches the target location ( $<25\text{cm}$ ), and then places the object back down. The sequence of actions is time-conditioned and specified through visual cues. The policy has 2 seconds to pick up the object and 2 seconds to reach the target. We report the distance of the object from the visual marker  $E_{\text{pos}}$  (mm), after the object is picked up. We also provide a breakdown of the success rate based on which hand to use — right Succ<sub>right</sub>, left Succ<sub>left</sub>, and both hands Succ<sub>bi</sub>. The kitchen scene presents an additional challenge. The policy must search for the object, reach and pick

it up, and then follow the marker. Here, the marker moves across the counter (instead of straight lifting). We report the search performance Succ<sub>search</sub>, measuring whether the object comes into view; the grasp success Succ<sub>grasp</sub>, measuring whether the correct object is grasped; and trajectory following success Succ<sub>traj</sub>, which shows whether the object follows the trajectory ( $<25\text{cm}$ ) and is successfully released at the end. Finally, we measure the policy’s ability to open drawers Succ<sub>drawer</sub>. For the kitchen setup, we test in both training scenes and *unseen scenes*.

**Tabletop: Object Lifting.** We report the results in Table 1 and Fig. 5. The oracle state-space policy reaches a high success rate for the training set, yet its success rate drops on the test set. On the other hand, the visual policy reaches a similar success rate across both training and testing. This result shows that while the ground-truth object shape information used by the state-space policy (extracted using BPS [49] using the canonical pose) can help achieve a high success rate during training, vision provides better generalization capabilities. Since most of the reference pre-grasps in the data use the right hand, PDC performs better at using the right hand than using the left hand and both hands. In terms of vision format, stereo outperforms both RGBD and RGB, with RGBD coming in second. Surprisingly, stereo outperforms RGBD across all metrics, suggesting that stereo depth estimation emerges from grasping and reaching target goals.

**Kitchen Scenes: Search, Grasp, Move, and Drawers.** For the kitchen scene, as shown in Table 2 and Fig. 6, our policy achieves a high search and grasping success rate, even when the object is deep into the counter and the humanoid is required to lean on the counter to reach the object. The goal reaching success rate is lower compared to the tabletop setting, partially due to the harder goal positions in the kitchen setting (random positions above the counter). PDC is robust across unseen objects and kitchen scenes without large performance degradation on unseen objects and scenes. Finally, the drawer-opening task achieves a high success rate in finding and opening the drawer.

GRAB-Test (5 unseen objects))												
idx	Feature Extractor	Resolution	Input Modality	Lookat Reward	Distillation	PULSE-X	Succ <sub>grasp</sub> ↑	E <sub>pos</sub> ↓	Succ <sub>right</sub> ↑	E <sub>pos</sub> ↓	Succ <sub>obj</sub> ↑	E <sub>pos</sub> ↓
1	ViT	128	RGB	✓	✗	✓	61.4%	142.9	61.4%	142.8	60.0%	161.3
2	ResNet	128	RGB	✓	✗	✓	68.5%	194.7	68.7%	194.0	40.0	375.7
3	CNN	128	RGB	✓	✓	✓	68.2%	<b>16.2</b>	68.6	<b>16.3</b>	10.0%	79.0
4	CNN	128	RGB	✓	✓	✗	71.4%	161.1	74.1	41.6	10.0%	69.8
5	CNN	32	RGB	✓	✗	✓	80.7%	116.4	80.6%	115.7	<b>100.0%</b>	189.9
6	CNN	64	RGB	✓	✗	✓	80.2%	100.1	80.1%	100.2	90.0%	82.0
7	CNN	128	RGB	✗	✗	✓	89.6%	85.6	89.5%	85.9	100.0%	<b>48.0</b>
8	CNN	128	RGB	✓	✗	✓	<b>90.1%</b>	100.5	<b>90.0%</b>	101.0	<b>100.0%</b>	<b>39.1</b>

Table 3. Ablation on various strategies of training PDC.

## 5. Ablation and Analysis

**Pretrained Vision Encoders.** In Table 3 Row 1 (R1), R2, and R7, we study using visual encoders frozen pretrained using ImageNet [19]. The results show that although the policy learns to pick up using these pretrained features, it does not reach comparable success rates to learning from scratch. We hypothesize that the features trained from ImageNet are insufficient to close the perception-action loop, and training from scratch or fine-tuning the visual feature extractor is needed to learn better visual features.

**Distillation vs From Scratch.** Comparing R3 and R8, we can see that distillation achieves a much lower success rate compared to training from scratch. Distillation does reach a better position error, indicating that predicting the visual marker position in 3D can be a good auxiliary loss.

**Humanoid Motion Prior .** R4 vs R8 shows that the humanoid motion prior (PULSE-X) can lead to better overall performance. Upon visual inspection, the behavior learned using PULSE-X is also more human-like.

**Resolution.** Comparing R5, R6, and R8: the policy success rate increases as the resolution of the images increases. This shows that higher resolution does have benefits in terms of grasp success and goal reaching. Notice that the stereo-model using two 80x80 resolution cameras achieves the highest success rate within our computational budget.

**Look-at Reward.** R7 vs R8 shows that without the look-at reward, the policy can still learn the table-top policy and achieves comparable results. However, the look-at reward helps shape the search behavior for the kitchen scene.

**Analysis: Emergent Search Behavior.** For the kitchen scene, our policy learns interesting emergent behavior due to the setup of the task. As can be seen from the [supplement videos](#), the policy learns to look left & right and scans the room (sometimes turns 360) while searching for the object. Also, we observe that the humanoid learns to scan the counter to find the object, a behavior that emerges from our object spawn locations. Also, the marker is not guaranteed to be in view when the object has been grasped. We observe that the humanoid learns to search left & right for the marker. Such behavior emerges from the reward to look at the object and the complex kitchen setting and is not

observed in the simpler tabletop setting.

**Analysis: Object Selection.** Our masking-as-object-selection design enables PDC to grasp objects of interest, even when objects of the same shape and geometry are in the view. In Table 2, results are reported when 6 objects are together in the scene. These results show high search and pickup rates, showing that PDC can find the right object.

**Analysis: Multi-Task Learning.** Our kitchen policy is warm-started from our trained tabletop policy. While the trained-from-scratch policy does learn to search (achieving a 64.3% success rate in search) for the object, it does not succeed in picking up the objects. Our perception-as-interface enables continuously adapting the policy to learn in different scenarios (table top → kitchen) and different tasks (object transportation and drawer opening).

## 6. Discussion

**Failure Cases and Limitations.** The vision-centric design in PDC leads to emergent search behaviors and success rates that outperform the state-based benchmark. However, the kitchen scene provides some challenging scenarios. For example, as the agent reaches towards objects located near the wall (on the far end of the counter), low-hanging cabinets and the steam-collector may interfere with the agent’s vision. The lack of visual clarity may lead to failing to grasp the object. In addition, if the grasp fails, our agent does not attempt to re-grasp. Instead, it shifts its focus to tracking the marker, despite not holding the object. Our policy also tends to shake its head from time to time and could benefit from image-stabilization techniques. Future work may overcome these limitations by providing additional and looser rewards. Such as enabling re-grasping, or using more advanced vision analysis to ensure the object is in view instead of naive angle comparison.

**Conclusions.** We propose PDC, a framework for learning multiple vision-guided tasks using a single policy via perception-as-interface and visual reinforcement learning. We observe that training in complex and diverse scenes leads to the emergence of behaviors such as search. As one of the first to tackle visual dexterous humanoid control, we believe our method, task setting, and perception-as-interface paradigm open many doors for future research.

## References

- [1] David Abel, Yuu Jinnai, Sophie Yue Guo, George Konidaris, and Michael Littman. Policy and value transfer in lifelong reinforcement learning. In *International conference on machine learning*, pages 20–29. PMLR, 2018. 3
- [2] Karl J Astrom. Optimal control of markov decision processes with incomplete state estimation. *J. Math. Anal. Applic.*, 10: 174–205, 1965. 4
- [3] Jinseok Bae, Jungdam Won, Donggeun Lim, Cheol-Hui Min, and Young Min Kim. Pmp: Learning to physically interact with environments using part-wise motion priors. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–10, 2023. 2, 5
- [4] Jona Braun, Sammy Christen, Muhammed Kocabas, Emre Aksan, and Otmar Hilliges. Physically plausible full-body hand-object interaction synthesis. *International Conference on 3D Vision (3DV)*, 2024. 2, 3, 14
- [5] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023. 2
- [6] Zerui Chen, Shizhe Chen, Etienne Arlaud, Ivan Laptev, and Cordelia Schmid. Vividex: Learning vision-based dexterous manipulation from human videos. *arXiv preprint arXiv:2404.15709*, 2024.
- [7] Xuxin Cheng, Jialong Li, Shiqi Yang, Ge Yang, and Xiaolong Wang. Open-television: Teleoperation with immersive active visual feedback. *arXiv preprint arXiv:2407.01512*, 2024. 2
- [8] Sudeep Dasari, Abhinav Gupta, and Vikash Kumar. Learning dexterous manipulation from exemplar object trajectories and pre-grasps. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3889–3896. IEEE, 2023. 5
- [9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 8
- [10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5
- [11] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *arXiv preprint arXiv:1702.03118*, 2017. 14
- [12] Clemens Eppner, Adithyavairavan Murali, Caelan Garrett, Rowland O’Flaherty, Tucker Hermans, Wei Yang, and Dieter Fox. scene\\_synthesizer: A python library for procedural scene generation in robot manipulation. *Journal of Open Source Software*, 2024. 3
- [13] Tuomas Haarnoja, Kristian Hartikainen, Pieter Abbeel, and Sergey Levine. Latent space policies for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pages 1851–1860. PMLR, 2018. 3
- [14] Tuomas Haarnoja, Ben Moran, Guy Lever, Sandy H Huang, Dhruva Tirumala, Markus Wulfmeier, Jan Humpelik, Saran Tunyasuvunakool, Noah Y Siegel, Roland Hafner, Michael Bloesch, Kristian Hartikainen, Arunkumar Byravan, Leonard Hasenclever, Yuval Tassa, Fereshteh Sadeghi, Nathan Batchelor, Federico Casarini, Stefano Saliceti, Charles Game, Neil Sreendra, Kushal Patel, Marlon Gwira, Andrea Huber, Nicole Hurley, Francesco Nori, Raia Hadsell, and Nicolas Heess. Learning agile soccer skills for a bipedal robot with deep reinforcement learning. *arXiv preprint arXiv:2304.13653*, 2023. 2
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 5
- [16] Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. In *arXiv*, 2024. 2
- [17] Donald D Hoffman, Manish Singh, and Chetan Prakash. The interface theory of perception. *Psychonomic bulletin & review*, 22:1480–1506, 2015. 3, 4
- [18] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024. 3
- [19] Yifeng Jiang, Michelle Guo, Jiangshan Li, Ioannis Exarchos, Jiajun Wu, and C Karen Liu. Dash: Modularized human manipulation simulation with vision and language for embodied ai. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–12, 2021. 4
- [20] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, pages 1–14, 2014. 14
- [21] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023. 4
- [22] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, pages 1–30. Lille, 2015. 5
- [23] Jiaman Li, Jiajun Wu, and C Karen Liu. Object motion guided human motion synthesis. *ACM Transactions on Graphics (TOG)*, 42(6):1–11, 2023. 7
- [24] Toru Lin, Zhao-Heng Yin, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Twisting lids off with two hands. *arXiv:2403.02338*, 2024. 2
- [25] Toru Lin, Yu Zhang, Qiyang Li, Haozhi Qi, Brent Yi, Sergey Levine, and Jitendra Malik. Learning visuotactile skills with two multifingered hands. *arXiv preprint arXiv:2404.16823*, 2024. 2
- [26] Toru Lin, Kartik Sachdev, Linxi Fan, Jitendra Malik, and Yuke Zhu. Sim-to-real reinforcement learning for vision-based dexterous manipulation on humanoids. *arXiv preprint arXiv:2502.20396*, 2025. 2

- [27] Toru Lin, Kartik Sachdev, Linxi Fan, Jitendra Malik, and Yuke Zhu. Sim-to-real reinforcement learning for vision-based dexterous manipulation on humanoids, 2025. 5
- [28] Chenhao Lu, Xuxin Cheng, Jialong Li, Shiqi Yang, Mazeyu Ji, Chengjing Yuan, Ge Yang, Sha Yi, and Xiaolong Wang. Mobile-television: Predictive motion priors for humanoid whole-body control. *arXiv preprint arXiv:2412.07773*, 2024. 2
- [29] Tyler Ga Wei Lum, Martin Matak, Viktor Makoviychuk, Ankur Handa, Arthur Allshire, Tucker Hermans, Nathan D Ratliff, and Karl Van Wyk. Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics. *arXiv preprint arXiv:2407.02274*, 2024. 2, 3
- [30] Zhengyi Luo, Jinkun Cao, Sammy Christen, Alexander Winkler, Kris M Kitani, and Weipeng Xu. Omnidgrasp: Grasping diverse objects with simulated humanoids. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2, 3, 5, 6, 7, 12, 13, 14
- [31] Zhengyi Luo, Ryo Hachiuma, Ye Yuan, and Kris Kitani. Dynamics-regulated kinematic policy for egocentric pose estimation. *NeurIPS*, 34:25019–25032, 2021. 7
- [32] Zhengyi Luo, Jinkun Cao, Josh Merel, Alexander Winkler, Jing Huang, Kris Kitani, and Weipeng Xu. Universal humanoid motion representations for physics-based control. *arXiv preprint arXiv:2310.04582*, 2023. 3, 5
- [33] Zhengyi Luo, Jinkun Cao, Alexander W. Winkler, Kris Kitani, and Weipeng Xu. Perpetual humanoid control for real-time simulated avatars. In *International Conference on Computer Vision (ICCV)*, 2023. 7, 13, 14
- [34] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-Octob: 5441–5450, 2019. 5
- [35] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 13
- [36] Yuan Meng, Zhenshan Bing, Xiangtong Yao, Kejia Chen, Kai Huang, Yang Gao, Fuchun Sun, and Alois Knoll. Preserving and combining knowledge in robotic lifelong reinforcement learning. *Nature Machine Intelligence*, pages 1–14, 2025. 3
- [37] Josh Merel, Arun Ahuja, Vu Pham, Saran Tunyasuvunakool, Siqi Liu, Dhruva Tirumala, Nicolas Heess, and Greg Wayne. Hierarchical visuomotor control of humanoids. *arXiv preprint arXiv:1811.09656*, 2018. 3
- [38] Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch and carry: Reusable neural controllers for vision-guided whole-body tasks. *ACM Trans. Graph.*, 39, 2020. 2, 4
- [39] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich,
- [40] Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6): 3740–3747, 2023. 6, 13
- [41] Gyeongsik Moon, Shunsuke Saito, Weipeng Xu, Rohan Joshi, Julia Buffalini, Harley Bellan, Nicholas Rosen, Jesse Richardson, Mize Mallorie, Philippe Bree, Tomas Simon, Bo Peng, Shubham Garg, Kevyn McPhail, and Takaaki Shiratori. A dataset of relighted 3D interacting hands. In *NeurIPS Track on Datasets and Benchmarks*, 2023. 13
- [42] Soroush Nasiriany, Fei Xia, Wenhai Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024. 3
- [43] Saptarshi Nath, Christos Peridis, Eseoghene Ben-Iwhiwhu, Xinran Liu, Shirin Dora, Cong Liu, Soheil Kolouri, and Andrea Soltoggio. Sharing lifelong reinforcement learning knowledge via modulating masks. In *Conference on Lifelong Learning Agents*, pages 936–960. PMLR, 2023. 3
- [44] Alva Noë. *Action in perception*. MIT press, 2004. 1
- [45] J Kevin O’regan and Alva Noë. A sensorimotor account of vision and visual consciousness. *Behavioral and brain sciences*, 24(5):939–973, 2001. 1
- [46] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A A Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:10967–10977, 2019. 7
- [47] Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel Van De Panne. Deeploco: dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Trans. Graph.*, 36:1–13, 2017. 3
- [48] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018. 5
- [49] Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *arXiv preprint arXiv:2205.01906*, 2022. 3
- [50] Sergey Prokudin, Christoph Lassner, and Javier Romero. Efficient learning on point clouds with basis point sets. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4332–4341, 2019. 7, 13
- [51] Haozhi Qi, Brent Yi, Sudharshan Suresh, Mike Lambeta, Yi Ma, Roberto Calandra, and Jitendra Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023. 2
- [52] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022.
- [53] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter

- Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *arXiv preprint arXiv:2307.04577*, 2023. 2
- [53] Stephane Ross, Geoffrey J Gordon, and J Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *arXiv preprint arXiv:1011.0686*, 2010. 13
- [54] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. 4, 6
- [55] Daniel L Silver, Qiang Yang, and Lianghao Li. Lifelong machine learning systems: Beyond learning algorithms. In *AAAI Spring Symposium: Lifelong Machine Learning*, 2013. 3
- [56] Himanshu Gaurav Singh, Antonio Loquercio, Carmelo Sferrara, Jane Wu, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Hand-object interaction pretraining from videos. *arXiv preprint arXiv:2409.08273*, 2024. 2
- [57] Ritvik Singh, Arthur Allshire, Ankur Handa, Nathan Ratliff, and Karl Van Wyk. Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands. *arXiv preprint arXiv:2412.01791*, 2024. 2, 3
- [58] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999. 3
- [59] Omid Taheri, Nima Ghorbani, Michael J Black, and Dimitrios Tzionas. Grab: A dataset of whole-body human grasping of objects. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 581–600. Springer, 2020. 7, 13
- [60] Chen Tessler, Israel Yoni Kasten, Israel Yunrong Guo, and Canada Nvidia. Calm: Conditional adversarial latent models for directable virtual characters. 3
- [61] Chen Tessler, Shahar Givony, Tom Zahavy, Daniel Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in minecraft. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. 3
- [62] Dhruva Tirumala, Markus Wulfmeier, Ben Moran, Sandy Huang, Jan Humplik, Guy Lever, Tuomas Haarnoja, Leonard Hasenclever, Arunkumar Byravan, Nathan Batchelor, et al. Learning robot soccer from egocentric vision with deep reinforcement learning. *arXiv preprint arXiv:2405.02425*, 2024. 2, 4
- [63] Weikang Wan, Haoran Geng, Yun Liu, Zikang Shan, Yaodong Yang, Li Yi, and He Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3891–3902, 2023. 2
- [64] Yinhuai Wang, Jing Lin, Ailing Zeng, Zhengyi Luo, Jian Zhang, and Lei Zhang. Physhoi: Physics-based imitation of dynamic human-object interaction. *arXiv preprint arXiv:2312.04393*, 2023. 2, 5, 6
- [65] Jungdam Won, Deepak Gopinath, and Jessica Hodgins. Control strategies for physically simulated characters performing two-player competitive sports. *ACM Trans. Graph.*, 40:1–11, 2021. 6
- [66] Jungdam Won, Deepak Gopinath, and Jessica Hodgins. Physics-based character controllers using conditional vaes. *ACM Trans. Graph.*, 41:1–12, 2022. 3, 14
- [67] Zhaoming Xie, Jonathan Tseng, Sebastian Starke, Michiel van de Panne, and C Karen Liu. Hierarchical planning and control for box loco-manipulation. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 6(3):1–18, 2023. 2
- [68] Kelvin Xu, Zheyuan Hu, Ria Doshi, Aaron Rovinsky, Vikash Kumar, Abhishek Gupta, and Sergey Levine. Dexterous manipulation from images: Autonomous real-world rl via sub-step guidance. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5938–5945. IEEE, 2023. 2
- [69] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024. 3
- [70] Sirui Xu, Hung Yu Ling, Yu-Xiong Wang, and Liang-Yan Gui. Intermimic: Towards universal whole-body control for physics-based human-object interactions, 2025. 2, 5, 6
- [71] Xinyu Xu, Yizheng Zhang, Yong-Lu Li, Lei Han, and Cewu Lu. Humanvla: Towards vision-language directed object rearrangement by physical humanoid. *Advances in Neural Information Processing Systems*, 37:18633–18659, 2025. 2
- [72] Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4737–4746, 2023. 2
- [73] Lixin Yang, Kailin Li, Xinyu Zhan, Fei Wu, Anran Xu, Liu Liu, and Cewu Lu. OakInk: A large-scale knowledge repository for understanding hand-object interaction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 7

# Appendix

<b>A Introduction</b>	<b>12</b>
<b>B Supplementary Website</b>	<b>12</b>
B.1. Teaser . . . . .	12
B.2. Tabletop . . . . .	12
B.3. Kitchens . . . . .	12
B.4. Failure cases . . . . .	12
<b>C Implementation Details</b>	<b>12</b>
C.1. State-space Policy . . . . .	12
C.2. Humanoid Motion Prior . . . . .	13
C.3. Details about PDC . . . . .	14
C.4. Details about Dataset . . . . .	14
<b>D Supplementary Results</b>	<b>15</b>
D.1. Per-object Success Rate . . . . .	15
D.2 Additional Qualitative Results . . . . .	15

## Appendices

### A. Introduction

In this document, we include additional details about PDC that are omitted from the main paper due to space limits. In Appendix. B, we provide descriptions of the results shown in our [supplement site](#). In Appendix. C, we include implementation details about the state-space oracle policy (Appendix. C.1) and humanoid motion prior (Appendix. C.2). In Appendix. D, we include additional results like per-object success rate breakdown. The code and models will be open-sourced.

### B. Supplementary Website

As the resulting behaviors are hard to convey through text and images, we also provide [supplement videos](#).

#### B.1. Teaser

The teaser video on the top illustrates the diverse environments in which PDC is trained. Across 6 different types of kitchens, spanning 80 random configurations and textures each. This diversity enables our agent to learn general behaviors, resulting in emergent search and dexterity.

#### B.2. Tabletop

The tabletop videos showcase the 3 control schemes. We provide a 3rd person video in addition to the agent’s own visual (rendered at a higher quality for ease of viewing). The visual markers in the top corners of the agent’s view indicate what each hand should do — stay idle, be ready to engage, engage, and release. We show examples of right hand, left hand, and also both-hand manipulation.

This is not limited to seen objects. Our PDC agent generalizes to new and unseen objects, directly from vision.

### B.3. Kitchens

#### B.3.1. Object transport

In the kitchen demonstrations, we first showcase the task of object transport. Here, the agent needs to identify the target object. There are 5 distractors (objects that are not the target). The target object is marked using a green mask overlay.

Once the object is picked up, the agent needs to find the target marker and bring the object towards it.

Unlike the tabletop which mainly focuses on lifting the object, here the agent needs to transport the object to a new location in the kitchen. Every 300 frames (10 seconds) we instruct it to release the object and randomly sample a new target object.

These scenes were not observed during training, showcasing the abilities of our agent to generalize. It exhibits traits such as search, where it scans the scene for the object. Moreover, an interesting emergent property is that it scans the counter-tops, as it has learned that is where kitchen objects are located (in our tasks).

#### B.3.2. Articulated drawers

Following the object transport, we showcase the ability of the agent to open drawers. These are highlighted in the agents point of view using a red overlay. The agent learns to insert its fingers through the handle in order to obtain a better grip. It then leans back with its body in order to pull the drawer open.

### B.4. Failure cases

Finally we showcase some common failure cases. For example, although our interface enables selecting which hand to utilize, it requires a system to determine the hand-to-use apriori. When an object can not be picked up with the selected hand, the agent will fail. In these videos we show what happens when a single hand is selected to pick up a large object. The same objects are successfully picked up in the tabletop examples above, using both hands, yet fails when attempted with a single hand.

## C. Implementation Details

### C.1. State-space Policy

We learn a state-space policy similar to Omniprism [30] but with the modifications that instead of using desired object trajectory, we only provide a single target position. Also, Omniprism does not have the ability to specify which hand to use, nor can it put the object down on command. To enable these capabilities, we add additional phase variables to inform the policy of the current command. Specifically,

GRAB-Train (25 Seen Objects)										
Object Success Rate	bowl	hammer	flashlight	mouse	duck	wineglass	scissors	airplane	stapler	torusmedium
	100%	98.5%	95.8%	94.6%	90.4%	96.7%	89.1%	87.8%	95.5%	100%
Object Success Rate	banana	cylindersmall	waterbottle	watch	stamp	alarmclock	headphones	phone	cylindermedium	flute
	100%	100%	100%	99.0%	100%	95.6%	97.1%	93.2%	98.3%	95.8%
Object Success Rate	cup	fryingpan	lightbulb	toothbrush	knife					
	92.6%	98.1%	100%	98.0%	93.0%					
GRAB-Test (5 Unseen Objects)										
Object Success Rate	apple	binoculars	camera	mug	toothpaste					
	97.4%	65.5%	99.3%	93.3%	95.7%					

Table 4. Per-object success breakdown for the PDC-stereo policy in the tabletop setting.

GRAB-Train (25 Seen Objects)										
Object Success Rate	bowl	hammer	flashlight	mouse	duck	wineglass	scissors	airplane	stapler	torusmedium
	50.0%	62.5%	72.0%	74.5%	87.0%	77.0%	65.0%	60.5%	70.5%	80.5%
Object Success Rate	banana	cylindersmall	waterbottle	watch	stamp	alarmclock	headphones	phone	cylindermedium	flute
	76.0%	56.0%	60.5%	69.5%	63.5%	58.5%	64.0%	66.5%	80.5%	74.5%
Object Success Rate	cup	fryingpan	lightbulb	toothbrush	knife					
	66.0%	43.5%	77.5%	42.0%	39.5%					
GRAB-Test (5 Unseen Objects)										
Object Success Rate	apple	binoculars	camera	mug	toothpaste					
	63.0%	31.0%	62.0%	53.0%	59.0%					

Table 5. Per-object success breakdown for the PDC-stereo policy in the kitchen setting.

Table 6. Imitation result on AMASS (14889 sequences).

the observation input for our omnigrasp policy is

$$\mathbf{o}_t^{\text{Omnigrasp}} \triangleq (\mathbf{o}_t^p, \mathbf{p}_t^{\text{obj}}, \boldsymbol{\theta}_t^{\text{obj}}, \boldsymbol{\sigma}^{\text{obj}}, \hat{\mathbf{p}}_{t+1}^{\text{obj}}, \mathbf{h}_t),$$

where  $\mathbf{p}_t^{\text{obj}} \in \mathcal{R}^3$  object translation,  $\boldsymbol{\theta}_t^{\text{obj}} \in \mathcal{R}^6$  object rotation, and object shape latent code  $\boldsymbol{\sigma}^{\text{obj}} \in \mathcal{R}^{512}$  are privileged information provided by simulation. We compute the shape latent code  $\boldsymbol{\sigma}^{\text{obj}}$  using BPS [49] with 512 randomly sampled points. The target object position  $\hat{\mathbf{p}}_{t+1}^{\text{obj}} \in \mathcal{R}^3$  specifies the 3D position of where the object’s centroid should be, and hand information  $\mathbf{h}_t \in \mathcal{R}^{512}$  provides information about which hand to use and when to grasp and put down.  $\mathbf{h}_t \triangleq (\mathbf{h}_t^{\text{left}}, \mathbf{h}_t^{\text{right}}, \mathbf{h}_t^{\text{time}})$  contains the indicator variables to indicate whether to use the left  $\mathbf{h}_t^{\text{left}} \in \mathcal{R}^1$  or the right  $\mathbf{h}_t^{\text{right}} \in \mathcal{R}^1$  hand. The time variable  $\mathbf{h}_t^{\text{time}} \in \mathcal{R}^{10}$  encodes whether the policy should grasp or release the object within the next second, sampled at 0.1s intervals.

We train the state-space policy using the same training procedure as our vision-guided policy and use the same architecture networks (removing the visual encoder). The humanoid motion prior is also the same as the vision-conditioned policy. The state-space policy is trained using a similar number of samples as the visual policy via multi-GPU RL training.

## C.2. Humanoid Motion Prior

We use a similar training procedure to obtain our humanoid motion representation (PULSE-X) as Omnigrasp [30].

AMASS-Train					
Method	Succ $\uparrow$	$E_{\text{g-mpipe}} \downarrow$	$E_{\text{mpipe}} \downarrow$	$E_{\text{acc}} \downarrow$	$E_{\text{vel}} \downarrow$
PHC-X – IsaacGym	99.9 %	29.4	31.0	4.1	5.1
PULSE-X – IsaacGym	99.5 %	42.9	46.4	4.6	6.7
PHC-X – IsaacLab	99.9 %	25.4	26.7	4.8	5.2
PULSE-X – IsaacLab	99.6 %	29.1	30.1	4.2	5.1

First, we train a motion imitator, PHC-X [33], on the cleaned AMASS dataset augmented with hand motion. Then, we distill the PHC-X policy into the PULSE-X CVAE using DAgger [53].

**Motion Imitator, PHC-X.** The task of motion imitation involves training a control policy to follow per-frame reference motion given input. For the motion imitation task, the input is defined as  $\mathbf{o}_t^{\text{imitation}} \triangleq (\mathbf{o}_t^p, \mathbf{o}_t^{\text{g-mimic}})$ , consisting of the proprioception  $\mathbf{o}_t^p$  and imitation goal  $\mathbf{o}_t^{\text{g-mimic}}$  (one frame difference between current pose and reference motion. We follow PHC’s [33] state, reward, and policy design to train one policy to imitate all motion sequences from the AMASS dataset. To increase the dexterity of the policy, we follow Omnigrasp [30] and augment the AMASS dataset’s whole body motion with randomly selected hand motion from the Interhand [40] and GRAB [59] dataset. To transition from IsaacGym [35] to IsaacSim (the underlying simulation engine for IsaacLab [39]), we implement the PHC-X policy in IsaacLab and train it for 5 days across using 8 GPUs. We

Method	# Envs	Learning Rate	$\sigma$	$\gamma$	$\epsilon$	# of samples
PHC-X	$3072 \times 8$	$2 \times 10^{-5}$	0.05	0.99	0.2	$\sim 10^{10}$
PULSE-X	$3072 \times 8$	$5 \times 10^{-4}$	—	—	—	$\sim 10^9$
Omnigrasp	$2048 \times 8$	$2 \times 10^{-5}$	0.36	0.99	0.2	$\sim 10^9$
PDC	$192 \times 8$	$2 \times 10^{-5}$	0.36	0.99	0.2	$\sim 10^9$

Table 7. Hyperparameters for PDC, PHC-X, and PULSE-X.  $\sigma$ : fixed variance for policy.  $\gamma$ : discount factor.  $\epsilon$ : clip range for PPO.

report success rate, mean per joint position error  $E_{\text{g-mpjpe}}$  (mm), local joint position error  $E_{\text{mpjpe}}$  (mm), acceleration error  $E_{\text{acc}}$  (mm/frame<sup>2</sup>), and velocity error  $E_{\text{vel}}$  (mm/frame) to evaluate the performance of our policies. All metrics are completed between the simulated humanoid motion and the reference motion. From Table 6, we can see that our IsaacLab implementation achieves comparable motion imitation results to those in IsaacGym.

**Humanoid Motion Prior, PULSE-X.** PHC-X learns the motor skills required to perform most of the actions in the AMASS dataset, and then PULSE-X learns a CVAE-like motion latent space. Specifically, PULSE learns the encoder  $\mathcal{E}_{\text{PULSE-X}}$ , decoder  $\mathcal{D}_{\text{PULSE-X}}$ , and prior  $\mathcal{P}_{\text{PULSE-X}}$  to compress motor skills into a latent representation. The encoder  $\mathcal{E}_{\text{PULSE-X}}(\mathbf{z}_t | \mathbf{o}_t^{\text{p}}, \mathbf{o}_t^{\text{g-mimic}}) \sim \mathcal{N}(\mathbf{z}_t | \boldsymbol{\mu}_t^{\text{e}}, \boldsymbol{\sigma}_t^{\text{e}})$  computes the latent code distribution based on current input states. The decoder  $\mathcal{D}_{\text{PULSE-X}}(\mathbf{a}_t | \mathbf{o}_t^{\text{p}}, \mathbf{z}_t)$  produces action (joint actuation) based on the latent code  $\mathbf{z}_t$ . The prior  $\mathcal{P}_{\text{PULSE-X}}(\mathbf{z}_t | \mathbf{o}_t^{\text{p}}) \sim \mathcal{N}(\mathbf{z}_t | \boldsymbol{\mu}_t^{\text{p}}, \boldsymbol{\sigma}_t^{\text{p}})$  defines a Gaussian distribution based on proprioception and replaces the unit Gaussian distribution used in VAEs [20]. The prior increases the expressiveness of the latent space and guides downstream task learning by forming a residual action space.

After the encoder, decoder, and prior is trained, similar to downstream task policies in PULSE, we form the action space of  $\pi_{\text{PDC}}$  as the residual action with respect to prior’s mean  $\boldsymbol{\mu}_t^{\text{p}}$  and compute the PD target  $\mathbf{a}_t$ :

$$\mathbf{a}_t = \mathcal{D}_{\text{PULSE-X}}(\pi_{\text{PDC}}(\mathbf{z}_t^{\text{PDC}} | \mathbf{o}_t^{\text{p}}, \mathbf{o}_t^{\text{g}}) + \boldsymbol{\mu}_t^{\text{p}}, \mathbf{o}_t^{\text{p}}), \quad (3)$$

where  $\boldsymbol{\mu}_t^{\text{p}}$  is computed by the prior  $\mathcal{P}_{\text{PULSE-X}}(\mathbf{z}_t | \mathbf{o}_t^{\text{p}})$ . The policy  $\pi_{\text{PDC}}$  computes  $\mathbf{z}_t^{\text{PDC}} \in \mathcal{R}^{48}$  instead of the target  $\mathbf{a}_t \in \mathcal{R}^{51 \times 3}$  directly, and leverages the latent motion representation of PULSE-X to produce human-like motion. Table 6 shows that our implementation achieves a high success rate on the AMASS dataset.

### C.3. Details about PDC

**Hyperparameters.** Hyperparameters for training PHC-X, PULSE-X, and PDC can be found in Appendix C.3. We do not change the hyperparameters significantly between training the visual policy and the state-space policy, demonstrating the robustness of PPO across different tasks.

**Rewards.** For our reward,

$$r_t^{\text{PDC}} = \begin{cases} r_t^{\text{approach}} + r_t^{\text{lookat}}, & \|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2 > 0.2 \text{ and } t < \lambda_{\text{start}} \\ r_t^{\text{pre-grasp}} + r_t^{\text{lookat}}, & \|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2 \leq 0.2 \text{ and } t < \lambda_{\text{start}} \\ r_t^{\text{obj}} + r_t^{\text{lookat}}, & \lambda_{\text{start}} \leq t < \lambda_{\text{end}} \\ (1 - \mathbf{1}_{\text{has-contact}}), & t \geq \lambda_{\text{end}} \end{cases} \quad (4)$$

$\lambda_{\text{start}}$  indicates the frame in which grasping should occur, and  $\lambda_{\text{end}}$  is when the frame should end and the object released.  $\mathbf{p}_t^{\text{H}}$  indicates the hands’ position. When the object is far away from the hands ( $\|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2 > 0.2$ ) and before grasping should start, the approach reward  $r_t^{\text{approach}}$  is similar to a point-goal [33, 66] reward  $r_t^{\text{approach}} = \|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2 - \|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_{t-1}^{\text{hand}}\|_2$ , where the policy is encouraged to get close to the pre-grasp. After the hands are close enough ( $\leq 0.2\text{m}$ ), we use a more precise hand imitation reward:  $r_t^{\text{pre-grasp}} = w_{\text{hp}} e^{-100\|\hat{\mathbf{p}}^{\text{pre-grasp}} - \mathbf{p}_t^{\text{H}}\|_2} \times \mathbf{1}\{\|\hat{\mathbf{p}}^{\text{pre-grasp}} - \hat{\mathbf{p}}_t^{\text{obj}}\|_2 \leq 0.2\} + w_{\text{hr}} e^{-100\|\hat{\theta}^{\text{pre-grasp}} - \theta_t^{\text{hand}}\|_2}$ , to encourage the hands to be close to pre-grasps. After the grasp start time, we switch to the object 3D location reward  $r_t^{\text{obj}} = e^{-5\|\mathbf{p}_t^{\text{target}} - \mathbf{p}_t^{\text{obj}}\|_2} \times \mathbf{1}_{\text{correct-hand}}$  to encourage the object being moved to specific locations.  $\mathbf{1}_{\text{correct-hand}}$  is the indicator random variable to determine if the correct hand has contact with the object. After the grasping should end ( $t \geq \lambda_{\text{end}}$ ), we encourage the agent not to be in contact with the object:  $\mathbf{1}_{\text{has-contact}}$  determines if any hand has contact with the object. The drawer open reward  $r_t^{\text{drawer}}$  is defined as how many degrees (angles defined by IsaacLab) the drawer is opened, clipped to 0 and 1.

**Network Architecture .** For our networks, we use a two-layer CNN with each with 32 channels. Our CNN produces a latent feature space of  $\mathcal{R}^{128}$ . We use 6-layer MLPs of units (2048, 2048, 1024, 1024, 512, 512) with silu activation [11]. Our GRU has one layer and 128 hidden units.

### C.4. Details about Dataset

We use the same train/test sequence split from Omnidgrasp [30] and Braun *et al.* [4]. For training, we use a subset of the objects and pick the ones that are more common in the households. Specifically, we use the following 25 categories:

torusmedium	flashlight	bowl
lightbulb	alarmclock	hammer
scissors	cylindermedium	stapler
phone	duck	airplane
knife	cup	wineglass
fryingpan	cylindersmall	waterbottle
banana	mouse	flute
headphones	stamp	toothbrush
watch		

Out of the 1016 training sequences from GRAB, 571 is picked. For testing, we use the same 140 sequences and

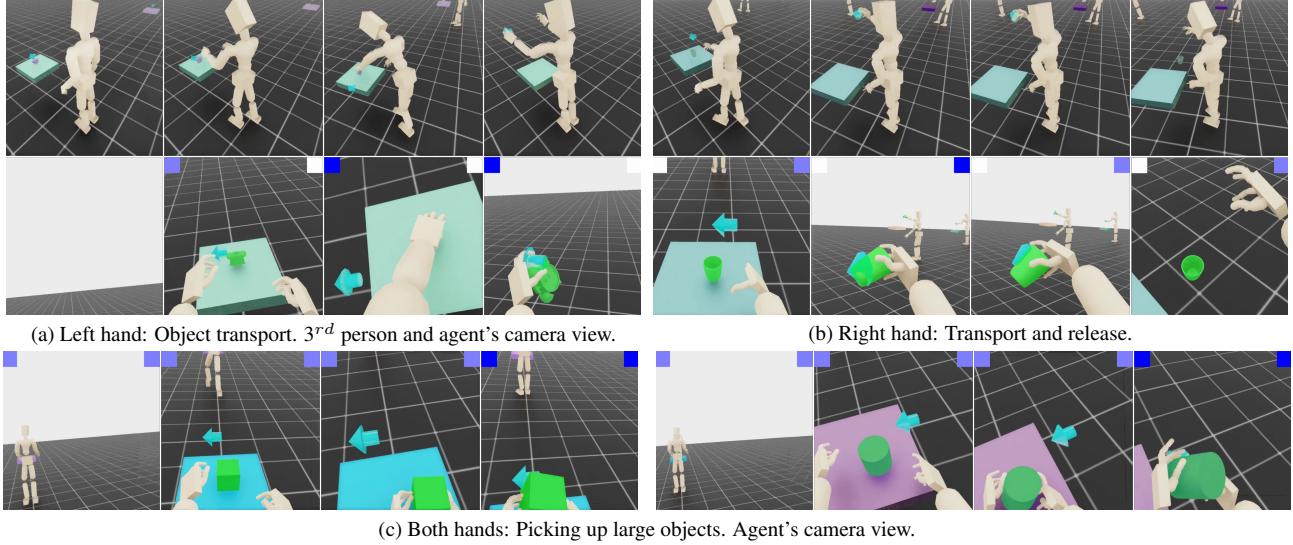


Figure 7. **Tabletop:** PDC is instructed directly from visual signals. A visual (top left and/or right) indicates in purple whether the corresponding hand should be prepared for contact. Changing to dark-blue indicates contact should be made. Instructing the agent to use both hands enables it to transport larger objects. Changing the indicator back to purple instructs the agent to release the object.

object types are apple, binoculars, camera, mug, toothpaste.

## D. Supplementary Results

### D.1. Per-object Success Rate

In Table 4 and Table 5, we report the per-object success rate for our stereo policy in the tabletop and kitchen scenes on the grab dataset. From the result, we can see that objects such as a bowl and a water bottle are easy to grasp, while objects with irregular shapes, like airplanes, are harder. Small and thin objects like a knife are also hard to pick up and have one of the lowest success rates across the kitchen and tabletop scene. Test objects such as binoculars are harder since they are bigger and difficult to grasp with one hand. The kitchen grasping paints a similar picture. No object has a zero success rate.

### D.2. Additional Qualitative Results

In Fig. 7 and Fig. 8 we present additional qualitative examples of the resulting controllers. They showcase the ability of the agent to handle objects in diverse scenes, using on-screen indicators as instructions. By training directly from vision within complex and diverse tasks, we observe search and dexterity emerges.

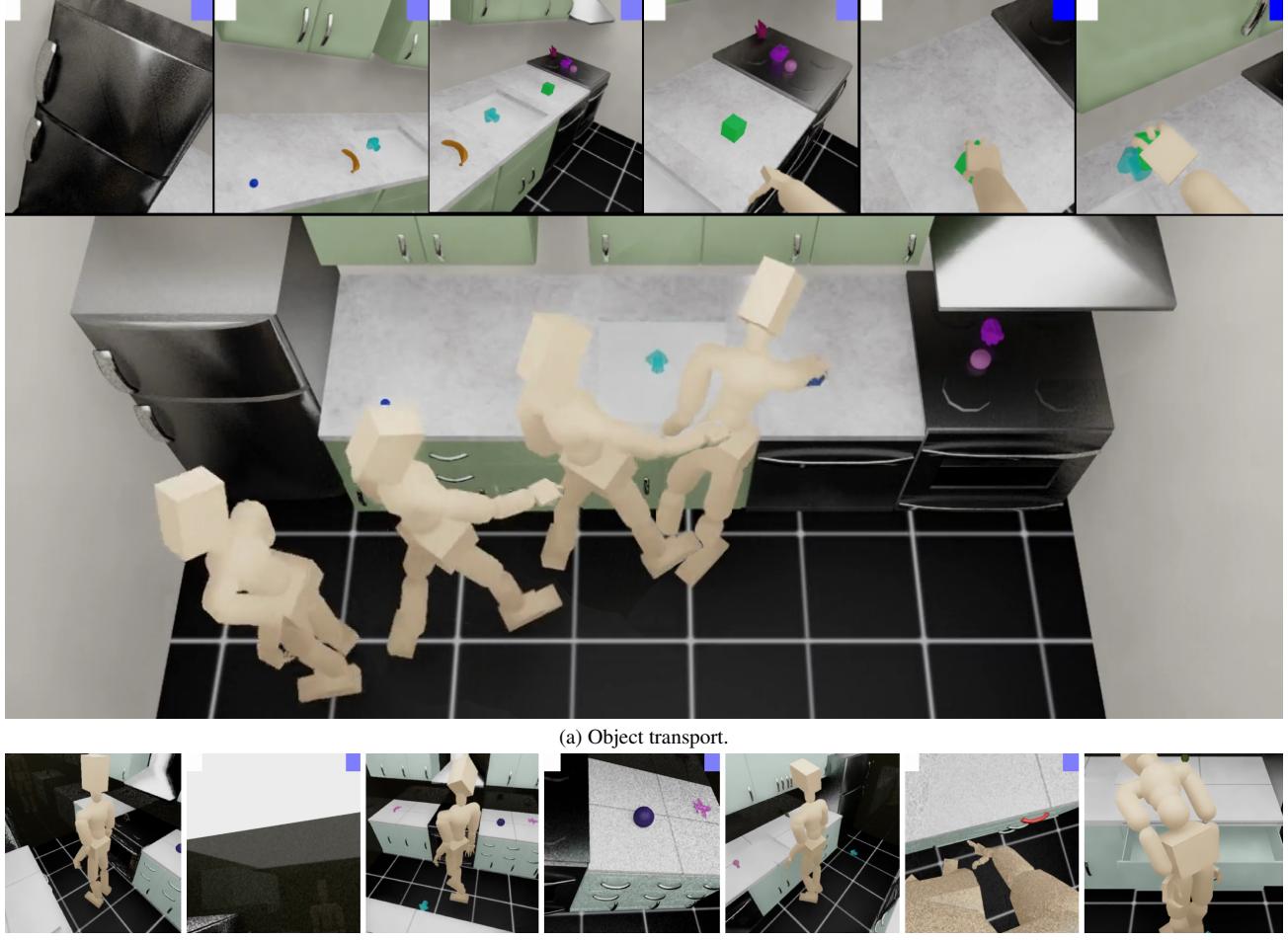


Figure 8. **Emergent search in the kitchen task:** By training the agent in diverse and complex scenes, it learns generalizable behaviors and avoids overfitting. We observe that behaviors such as searching emerge. When the object is not in view, the agent scans the counter top and top drawers in order to learn of its objective and execute on it.