The One RING: a Robotic Indoor Navigation Generalist

Ainaz Eftekhar [1,2] **Luca Weihs** [2] **Rose Hendrix**[2]
Ege Caglar[1] **Jordi Salvador** [2] **Alvaro Herrasti** [2] **Winson Han**[2]
Eli VanderBil[2] **Aniruddha Kembhavi** [1,2] **Ali Farhadi** [1,2] **Ranjay Krishna**[1,2]
Kiana Ehsani[*2] **Kuo-Hao Zeng**[*2]
[1]University of Washington
[2]Allen Institute for AI

one-ring-policy.allen.ai

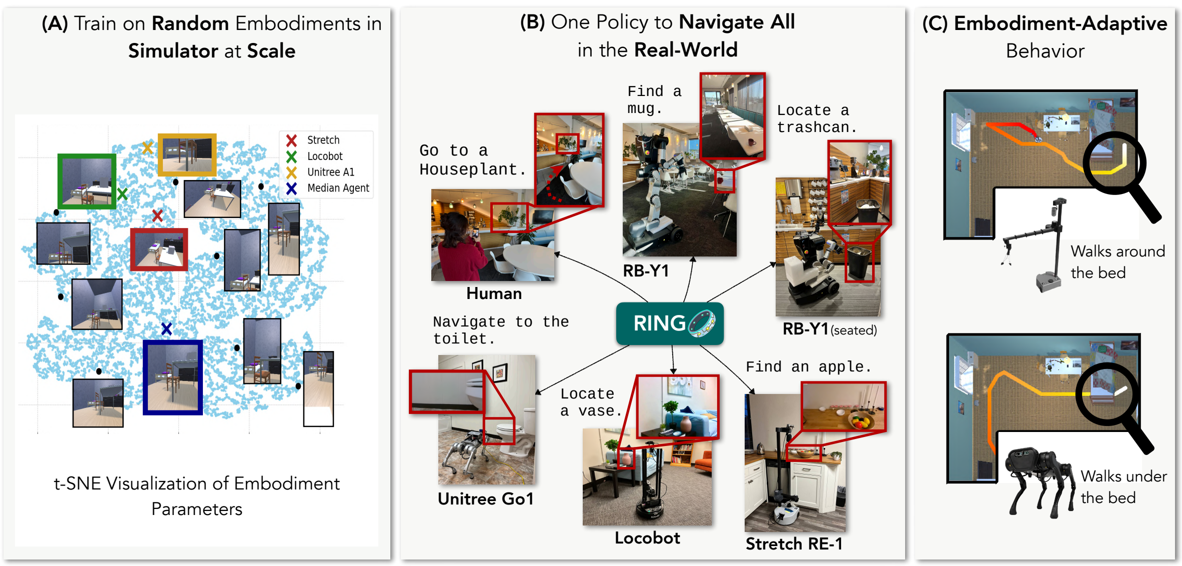

Figure 1. We show that training on *one million* randomly generated embodiments in simulation (varying camera configurations, body size, and rotation pivot point) results in RING, a generalist navigation policy that works across various robot embodiments in the real world. (A) A t-SNE visualization of embodiment parameters for 30k random agents and three real robots (*we do not train on any real robot embodiment parameters*). Egocentric views from the first camera are shown for 10 sample agents. (B) RING transfers zero-shot to a wide range of embodiments in the real-world including Stretch RE-1, LoCoBot, Unitree Go1, RB-Y1 wheeled humanoid. (C) The policy displays embodiment-adaptive behavior, adjusting its navigation strategy based on its embodiment.

Abstract

Modern robots vary significantly in shape, size, and sensor configurations used to perceive and interact with their environments. However, most navigation policies are embodiment-specific—a policy trained on one robot typically fails to generalize to another, even with minor changes in body size or camera viewpoint. As custom hardware becomes increasingly common, there is a growing need for a single policy that generalizes across embodiments, eliminating the need to (re-)train for each specific robot. In this paper, we introduce RING *(**R**obotic **I**ndoor **N**avigation **G**eneralist), an embodiment-agnostic policy that turns any mobile robot into an effective indoor semantic navigator. Trained entirely in simulation,* RING *leverages large-scale randomization over robot embodiments to enable robust generalization to many real-world platforms. To support this, we augment the AI2-THOR simulator to instantiate robots with controllable configurations, varying in body size, rotation pivot point, and camera parameters. On the visual object-goal navigation task,* RING *achieves strong cross-embodiment (XE) generalization—72.1% average success rate across* 5 *simulated embodiments (a 16.7% absolute improvement on the* CHORES-$\mathbb{S}$ *benchmark) and 78.9% across* 4 *real-world platforms, includ-*

ing Stretch RE-1, LoCoBot, and Unitree Go1—matching or even surpassing embodiment-specific policies. We further deploy RING *on the RB-Y1 wheeled humanoid in a real-world kitchen environment, showcasing its out-of-the-box potential for mobile manipulation platforms.*

1. Introduction

Robot embodiments are diverse and are constantly evolving to better suit new environments and tasks. This range in body configurations—differences in size, shape, wheeled or legged locomotion, and sensor configurations—not only shapes how robots perceive the world but also how they act in it. A robot with a wide field of view (FoV) or multiple cameras can scan its surroundings quickly, while one with a narrower view might need to more actively explore a room. A small robot can squeeze through tight spaces, a low-profile one can duck under furniture, and a larger robot may need to follow more conservative routes. The influence of embodiment on behavior means a policy trained on one design, or even several, often does not perform well out of domain.

There has been progress towards scalable cross-embodiment training [13, 34, 46, 48, 58]. While these methods demonstrate some transfer to unseen embodiments, they still suffer from performance degradation with relatively small changes in embodiment (e.g., camera pose modification on the same robot) [38, 53]. Potentially, this is due to these methods relying on the small amount of real-world data available in public datasets-around 20 embodiments in total [34]. Similarly, general-purpose navigation policies [43–45] are trained on datasets with relatively few embodiments (e.g., 8 robots in [44]), limiting their generalization. A more comprehensive solution is needed—one that can robustly handle the full spectrum of possible embodiments without retraining or additional adaptation.

We introduce RING, a **R**obotic **I**ndoor **N**avigation **G**eneralist. RING is trained *exclusively in simulation*, without any use of real-world robot embodiments. In other words, all robot platforms we evaluate on (i.e., Stretch RE-1, LoCoBot, Unitree's Go1, RB-Y1) are *unseen* by RING during training. We leverage simulation to randomly sample **1 Million** agent body configurations, varying the robot's camera parameters, collider sizes, and center of rotation. Concretely, each embodiment consists of a collider box of varying dimensions and cameras with randomized parameters, placed randomly within the collider box. Fig.1-A presents a t-SNE[47] visualization of body parameters for 30k such agents. Our approach builds on the success of prior works that achieve strong real-world performance through large-scale simulation-only training [17, 24, 65]. Simulation enables training across a vast distribution of environments (150k ProcTHOR houses [12]) and objects (40k+ 3D objects in Objaverse [11]) in the AI2-THOR simulator. Extensive domain randomization on visual observations and the use of pre-trained visual encoders then allows simulation-trained policies to bridge the sim-to-real gap. We follow the training procedure in FLaRe [24], first training our policy on expert trajectories collected from 1M randomized embodiments and subsequently fine-tuning it with on-policy reinforcement learning (RL) in the simulator.

Our results demonstrate generalization to *truly unseen embodiments*. RING transfers to diverse real-world embodiments without any adaptation, despite being trained entirely in simulation without access to the real robot configurations. We evaluate in a zero-shot setting across Stretch RE-1, LoCoBot, Unitree's Go1, RB-Y1 wheeled humanoid, and even "Navigation Assistants," where a human user captures egocentric observations on their phone and prompts RING to predict actions. RING achieves 72.1% average success rate in simulation (16.7% absolute improvement on CHORES-$\mathbb{S}$ benchmark) and 78.9% on real robot platforms—matching or even surpassing embodiment-specific policies. RING can be further adapted to an *embodiment-specialized* policy with even better performance (up to 10% absolute improvement) with minimal finetuning. RING is easy to install, and is ready for use by the community. We will release our pretrained models, code, and data.

2. RING

With the growing diversity of robots used in research labs and real-world applications, there remains a need for a policy that can operate a wide range of embodiments and transfer, in a zero- or few-shot manner, to unseen robots. We introduce RING, a generalist policy for indoor visual navigation that *learns from a broad spectrum of embodiments, trained exclusively in simulation, without any direct use of actual robot embodiments*. We show that training on an extensive range of ~1M random embodiments results in a robust navigation policy, enabling zero-shot transfer to unseen real-world robots. To train RING, we define the space of random embodiments (Sec. 2.1), enable generation of expert trajectories for random embodiments in simulation (see Appendix 7), and use state-of-the-art architecture designs (Appendix 9) to train with a combination of IL and RL methods (Sec. 2.2).

2.1. Embodiment randomization at scale

Domain randomization [8] is a class of methods in which policies are trained across a wide range of simulated *environmental* parameters; the aim is to enable robustness to unseen environments. Our approach is complementary yet orthogonal; we apply embodiment randomization to train policies on a diverse set of *robot body* parameters, enabling robust deployment to unseen real-world robots.

Parameters	Training Range
Collider Size $(\alpha_x, \alpha_y, \alpha_z)$	$[0.2, 0.5]$, $[0.3, 1.5]$, $[0.2, 0.5]$
Rotation Center (o_x, o_y, o_z)	$[-\alpha_x/2, \alpha_x/2]$, $[-\alpha_y/2, \alpha_y/2]$, $[-\alpha_z/2, \alpha_z/2]$
Vertical FoV (cam1, cam2)	$[40, 100]$, $[40, 100]$
Horizontal FoV (cam1, cam2)	$[40, 120]$, $[40, 120]$
Camera Pitch (cam1)	$[-20, 40]$
Camera Pitch (cam2)	$[-20, 60]$
Camera Yaw (cam1, cam2)	always 0, $[0, 360]$
Camera Position (x) (cam1, cam2)	$[-\alpha_x/2, \alpha_x/2]$, $[-\alpha_x/2, \alpha_x/2]$
Camera Position (y) (cam1, cam2)	$[0.3, \alpha_y]$, $[0.3, \alpha_y]$
Camera Position (z) (cam1, cam2)	$[-\alpha_z/2, \alpha_z/2]$, $[-\alpha_z/2, \alpha_z/2]$
RGB dimensions (H, W)	$[112, 448]$, $[112, 448]$

Table 1. **Random Embodiment Parameters.** We generate 1M different embodiments sampled from the ranges above.

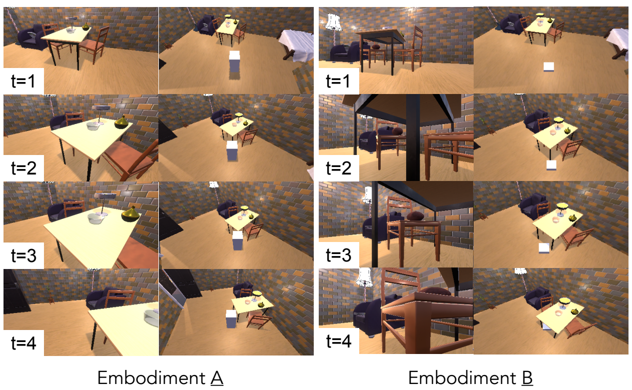

Figure 2. **Different embodiments exhibit different behaviors.** Embodiment A (shown on the left) has a bigger body size compared to Embodiment B (shown on the right). As a result, B can go under the table to get to the chair but A collides with the table and has to go around.

We model the body of the agent as an invisible collider box in the AI2-THOR [29] simulator. Each agent can have 1 or 2 RGB cameras placed at a random pose within the collider box. Parameters corresponding to both the body and the cameras are sampled randomly from the ranges specified in Tab. 1. We also modify the process of generating expert trajectories to account for the diversity of embodiments, for details see Appendix 7. Below, we detail the parameters varied in our embodiment randomization: **Collider size** $(\alpha_x, \alpha_y, \alpha_z)$. The agent's body is modeled as a collider box. We use three scale factors $(\alpha_x, \alpha_y, \alpha_z)$ to scale the box along x, y, z axis. **Rotation center** (o_x, o_y, o_z). These coordinates define the agent's pivot point. While this center is typically near (0,0), it can vary across different robots. We sample o_x from the range $[-\frac{\alpha_x}{3}, \frac{\alpha_x}{3}]$ and o_y from the range $[-\frac{\alpha_y}{3}, \frac{\alpha_y}{3}]$, with the sampling ranges determined by the collider size. **Camera parameters**. Each agent is equipped with two RGB cameras placed within the collider box. We randomize several camera parameters, including position, rotation, FoV, and aspect ratio. While the first camera always faces forward, the second camera can rotate up to 360° in z-axis, enabling it to face forward, to the sides, or backward.

2.2. Training paradigm

We adopt the training recipe of pretraining our policy on expert trajectories collected from randomized embodiments (Sec. 2.1), followed by finetuning with on-policy RL using the randomized embodiments in the AI2-THOR simulator [29].

Large-scale imitation learning with random embodiments: We train our policy using expert trajectories collected from 1M randomized embodiments across 50k procedurally generated PROCTHOR houses [12], containing approximately 40k annotated 3D objects [18]. At each time step, the linear actor-critic head in the Causal Transformer Decoder predicts action logits, and a cross-entropy loss is computed between the predicted logits π^t and the expert action. We use a batch size of 240 trajectories, each with a temporal context window of 100 steps. Training is conducted on 8× H100 GPUs (80 GB each) using the AdamW optimizer with a learning rate of $2 \cdot 10^{-4}$ for 80k iterations.

Large-scale RL finetuning with random embodiments: Following the training recipe in FLaRe [24], we perform large-scale RL fine-tuning using AllenAct [50] on randomized embodiments in simulation. This fine-tuning phase is critical for enabling the policy to learn through trial and error how to navigate diverse embodiments. We use DD-PPO with 64 parallel environments and 128 rollout steps across 4 machines (each with 8× H100 GPUs), training for 40M steps using the AdamW optimizer with a learning rate of $2{\cdot}10^{-5}$. As in FLaRe [24], we disable the entropy term in the PPO loss to prevent catastrophic forgetting. For fair comparison, we adopt the reward function from [12]: $r_t = \max(0, min\Delta_{0:t-1} - \Delta_t) + s_t - \rho$, where $\min \Delta_{0:t-1}$ is the minimum L2 distance between the agent and the target object up to time $t-1$, Δ_t is the current L2 distance, s_t is a success reward, and $\rho = 0.01$ is a step penalty encouraging task efficiency. The agent must explicitly issue `Done` to receive the success reward ($s_t = 10$); otherwise, $s_t = 0$.

3. Experiments

Our experiments show that RING operates effectively across a wide range of embodiments, including *Stretch RE-1*, *LoCoBot*, *Unitree Go1*, and *RB-Y1*, despite being trained exclusively in simulation **without** any direct exposure to real robot embodiments. Our key results are: 1) RING generalizes **zero-shot** to 5 *truly* unseen embodiments, despite never being trained on them, and achieves state-of-the-art performance across multiple benchmarks (Sec. 3.1). 2) Our policy, trained solely in simulation on randomized embodiments, transfers directly to the **real-world**, on 5 real robots and as navigation assistants (Sec. 3.2). 3) RING can be easily adapted to **embodiment-specialized** policies with minimal finetuning. It achieves better performance on each specific robot (Sec. 3.3). 4) RING shows **embodiment-**

Model	Loss	Train Embodiment	Benchmark Embodiment					Average
			Stretch	**Stretch** (Nav Cam)	**Stretch** (Factory Config)	**LoCoBot**	**Unitree A1**	
SPOC [17]	IL only	Stretch	57.0 (38.1)*	37.9 (19.0)	33.0 (19.3)	16.2 (5.4)	2.1 (1.6)	29.2 (16.7)
SPOC-2.3M	IL only	Stretch	60.0 (30.3)*	37.5 (17.9)	46.0 (19.4)	24.0 (7.9)	10.0 (5.2)	35.5 (16.1)
POLIFORMER [65]	RL only	Stretch	81.0 (58.1)*	65.0 (35.5)	47.5 (25.6)	27.5 (14.8)	42.6 (25.1)	52.7 (31.8)
		LoCoBot	56.0 (32.9)	56.5 (34.7)	52.0 (27.7)	61.5 (44.7)*	50.5 (34.2)	55.4 (34.9)
		Unitree A1	40.0 (25.2)	39.0 (22.5)	35.5 (20.9)	30.0 (17.4)	55.3 (48.2)*	40.0 (26.8)
FLARE [24]	IL + RL	Stretch	82.0 (63.5)*	55.5 (37.9)	38.0 (19.6)	21.5 (10.9)	27.0 (15.1)	44.8 (29.4)
RING-ZERO-SHOT	IL + RL	RING-Random	**76.0 (55.9)**	**74.0 (52.5)**	**72.0 (52.7)**	**66.5 (45.3)**	**72.0 (58.6)**	**72.1 (53.0)**

Table 2. **Zero-shot Results.** RING shows zero-shot generalization to four unseen embodiments. Unless otherwise specified, "Stretch" refers to the two-camera variant of the RE-1 platform, used in[17]. All prior methods fail to generalize effectively to embodiments beyond those seen during training. Gray* numbers indicate evaluation on the training embodiment; all others reflect zero-shot performance on unseen embodiments.

adaptive behavior, adjusting its strategies based on the agent's body (Sec. 3.4). 5) We perform a **collision analysis** showing that RING remains as safe—*and in some cases even safer*—than embodiment-specific policies(Sec. 10).

3.1. RING generalizes zero-shot to unseen embodiments

We perform zero-shot evaluations of all policies on four robot embodiments: Stretch RE-1 (with 1 or 2 cameras), LoCoBot, and Unitree A1 in simulation.

Baselines. We select prior works from both imitation learning (IL) and reinforcement learning (RL) for comparison. Each baseline is trained on a specific embodiment and evaluated in a zero-shot setting across four different embodiments. SPOC [16] is a supervised IL baseline trained on shortest-path expert trajectories in AI2-THOR. PoliFormer [65] is a state-of-the-art transformer-based policy for object goal navigation, trained from scratch using RL. FLaRe [24] combines IL and RL for efficient policy fine-tuning. All baselines use similar architectures and comparable data, except for SPOC (reason to include SPOC 2.3M). Specifically, SPOC [17] (SPOC-2.3M) is trained with IL on Stretch RE-1 using 100k (2.3M) expert trajectories; Poliformer [65] is trained from scratch on each embodiment individually over 300M RL steps (more than other baselines in terms of training frames); and FLaRe [24] finetunes SPOC on Stretch RE-1 with an additional 20M RL steps.

Experimental details. RING is first trained with IL on 1M expert trajectories collected from randomized embodiments in simulation, followed by finetuning with RL for an additional 40M steps on the randomized embodiments. *Note that all four target embodiments were unseen during training*. We evaluate on the navigation benchmark in CHORES-$\mathbb{S}$ [17], a simulation benchmark for household robot with 200 tasks across 200 scenes. For Unitree A1, we create a new, similar benchmark with 200 tasks adjusted for the robot's lower height to ensure that all targets are feasible.

Results. Tab. 2 presents the zero-shot evaluation of all policies across four embodiments. We compare *Success Rate* and *Success Weighted by Episode Length (SEL [15])*, a metric measuring efficiency. The results indicate that all single-embodiment baselines struggle to generalize effectively to new embodiments, with performance declining as embodiment differences increase. In contrast, RING exhibits strong generalization across all embodiments, despite not being trained on any of them, achieving an average absolute improvement of 16.7% in Success Rate. In some cases, it outperforms the baseline trained on the target embodiment: PoliFormer trained on LoCoBot *(*$61.5 \rightarrow 68.5$*)* and Unitree A1 ($55.3 \rightarrow 72.0$). These 2 more challenging embodiments (lower FoV, low camera placement) make RL from scratch less effective. RING benefits from more efficient learning by training across random embodiments at scale with more diverse navigation behaviors.

3.2. RING transfers to real-world embodiments despite being purely trained in simulation

Robot evaluation. We zero-shot evaluate our policy on 4 unseen robots in a multi-room real-world apartment (Fig.5) without any real-world-specific finetuning (Tab. 3). We use the same evaluation set of 15 tasks for LoCoBot[10, 12, 65] (3 start poses × 5 targets) and 18 tasks for Stretch RE-1 [17, 24, 65] (3 poses × 6 goals). For Unitree Go1, we create a new set with 3 start poses and 4 objects (*toilet, sofa, TV, trashcan*) placed to match its lower viewpoint. RING matches or outperforms specialized policies, likely due to cross-embodiment (XE) training enabling robust sim-to-real transfer in the presence of real-world noise. We also deploy RING on the RB-Y1 wheeled humanoid in an unstructured kitchen, where it successfully navigates to targets (*trashcan, apple, houseplant, mug*) at two different heights (standing/seated), using an iPhone 16 Pro camera mounted on the robot to stream visual observations (Fig. 1-B). (*More detail in Appendix 6.1*)

3.3. RING can efficiently adapt to an embodiment-specialized policy with minimal finetuning

Although RING generalizes zero-shot across diverse embodiments, some scenarios benefit from embodiment-specialized policies for optimal performance. Here, we

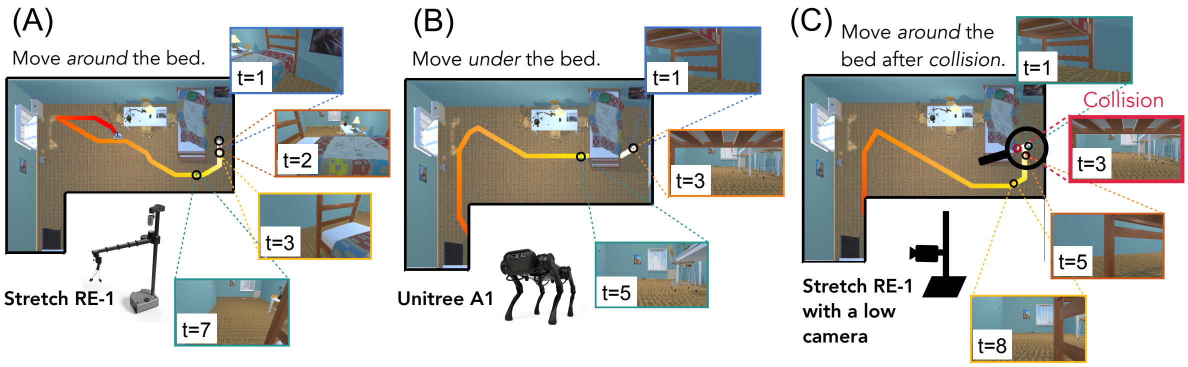

Figure 3. RING has **embodiment-adaptive** behavior, adjusting its navigation strategy based on the embodiment. The quadruped robot (B), due to its lower height, walks under the bed, while the taller Stretch-RE1 robot (A) navigates around it. In (C), an agent with the same height as Stretch-RE1 but a lower camera position initially attempts to move under the bed, assuming a shorter height. After colliding, it adapts its strategy and navigates around the bed, similar to Stretch-RE1.

Model	Train Embodiment	Eval Embodiment			
		Stretch	**Stretch (FC)**	**LoCoBot**	**Unitree Go1**
ProcTHOR [12]	LoCoBot	-	-	26.7	-
Phone2Proc [10]	LoCoBot	-	-	66.7	-
SPOC [16]	Stretch	50.0	-	-	-
POLIFORMER [65]	Stretch	83.3	33.3	-	-
	LoCoBot	-	-	80.0	-
	Unitree Go1	-	-	-	41.7
FLARE [24]	Stretch	94.4	-	-	-
RING-ZERO-SHOT	RING-Random	83.3	**72.2**	**80.0**	**80.0**

Table 3. **Real-world Results**. RING transfers zero-shot to the real-world without any finetuning. Gray numbers are evaluated on same embodiment as their training. RING achieves 78.9% success rate on average across 4 real-world robots.

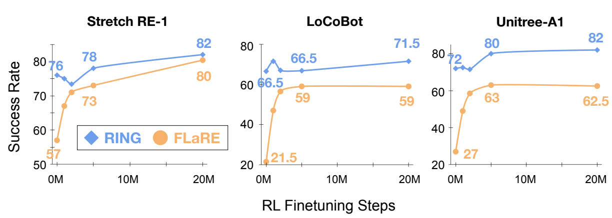

Figure 4. **Embodiment-Specialized Adaptation.** RING, pretrained on randomized embodiments, adapts efficiently to individual robots with minimal fine-tuning.

show that RING can be easily adapted to a *robot-specialized policy* through minimal fine-tuning. **Baselines.** We compare with FLaRe [24], which demonstrates effective adaptation to new tasks and embodiments. It is pretrained on Stretch RE-1 and finetuned on each of the three test embodiments using up to 20M RL steps. **Implementation.** We finetune RING, pretrained on randomized embodiments, on each robot for up to 20M RL steps, using the same hyperparameters as FLaRe for fair comparison. Following FLaRe, we repurpose `RotateBase(±6°)` to `TiltCamera(±30°)` for LoCoBot, enabling camera control not available during zero-shot evaluation. **Results.** As shown in Fig. 4, RING adapts efficiently, achieving superior performance with minimal fine-tuning. For LoCoBot and Unitree-A1, FLaRe underperforms compared to Stretch RE-1, suggesting that pretraining on a single embodiment limits generalizability. This underscores the value of policies like RING that can adapt quickly and consistently to new embodiments.

3.4. RING changes its behavior across different embodiments

Ideally, an optimal policy would modify its behavior depending on the embodiment. For instance, a thinner robot can navigate through narrow hallways or under furniture, and a wider agent may need to take more conservative paths. Our qualitative results show that RING exhibits embodiment-adaptive behavior. In Fig. 3-A,B, both Stretch RE-1 and Unitree A1 begin behind a bed. The low-profile quadruped moves directly under it, while Stretch RE-1 navigates around—demonstrating that RING *implicitly* infers embodiment characteristics from visual input, without access to privileged body information. Visual input reveals cues like camera specs and, in some cases, the agent's height. However, vision alone can be ambiguous, prompting the agent to rely on physical interactions—such as collisions—to refine its understanding. (Collision feedback may come from actual impacts or from sensors that anticipate collisions before they occur.) In Fig. 3-C, an agent with a low-mounted camera but tall body misjudges its own height, initially attempts to go under the bed, collides, and then reroutes like Stretch RE-1. This behavior is not present in the expert data but emerges during reinforcement learning through training across diverse embodiments.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *CoRL*, 2022. 11

[2] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. ObjectNav revisited: On evaluation of embodied agents navigating to objects. *CoRR*, abs/2006.13171, 2020. 10, 12

[3] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *ECCV*, 2024. 10

[4] Nico Bohlinger, Grzegorz Czechmanowski, Maciej Piotr Krupka, Piotr Kicki, Krzysztof Walas, Jan Peters, and Davide Tateo. One policy to run them all: Towards an end-to-end learning approach to multi-embodiment locomotion. In *CoRL*, 2024. 10

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael S. Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong T. Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: vision-language-action models transfer web knowledge to robotic control. *CoRR*, abs/2307.15818, 2023. 10

[6] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. *ICLR*, 2020. 10

[7] Lawrence Yunliang Chen, Chenfeng Xu, Karthik Dharmarajan, Zubair Irshad, Richard Cheng, Kurt Keutzer, Masayoshi Tomizuka, Quan Vuong, and Ken Goldberg. Rovi-aug: Robot and viewpoint augmentation for cross-embodiment robot learning. In *CoRL*, 2024. 10

[8] Xiaoyu Chen, Jiachen Hu, Chi Jin, Lihong Li, and Liwei Wang. Understanding Domain Randomization for Sim-to-real Transfer. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. 2

[9] Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Mart'in-Mart'in, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding,

Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. In *ICRA*, 2024. 10

[10] Matt Deitke, Rose Hendrix, Luca Weihs, Ali Farhadi, Kiana Ehsani, and Aniruddha Kembhavi. Phone2Proc: Bringing robust robots into our chaotic world, 2022. 4, 5, 14

[11] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A Universe of Annotated 3D Objects. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13142–13153, 2022. 2, 10

[12] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ProcTHOR: Large-scale embodied AI using procedural generation. In *NeurIPS*, 2022. 2, 3, 4, 5

[13] Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. *arXiv preprint arXiv:2408.11812*, 2024. 2, 10

[14] Jiafei Duan, Wentao Yuan, Wilbert Pumacay, Yi Ru Wang, Kiana Ehsani, Dieter Fox, and Ranjay Krishna. Manipulate-anything: Automating real-world robots using vision-language models. *arXiv preprint arXiv:2406.18915*, 2024. 10

[15] Ainaz Eftekhar, Kuo-Hao Zeng, Jiafei Duan, Ali Farhadi, Ani Kembhavi, and Ranjay Krishna. Selective visual representations improve convergence and generalization for embodied ai. In *ICLR*, 2023. 4, 10

[16] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, et al. Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. In *CVPR*, 2024. 4, 5, 14

[17] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, et al. Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. In *CVPR*, pages 16238–16250, 2024. 2, 4, 10, 11, 12, 13, 14, 16

[18] Allen Institute for AI. ObjaTHOR: Python package for importing and loading external assets into ai2thor. `https://github.com/allenai/objathor`, 2024. 3

[19] Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. In *CoRL*, 2024. 10

[20] Aric A. Hagberg, Daniel A. Schult, Pieter Swart, and JM Hagberg. Exploring Network Structure, Dynamics, and Function using NetworkX. *Proceedings of the Python in Science Conference*, 2008. 12

[21] Sudarshan Harithas and Srinath Sridhar. Motionglot: A multi-embodied motion generation model. *arXiv preprint arXiv:2410.16623*, 2024. 10

[22] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybern.*, 4:100–107, 1968. 12

[23] Joey Hejna, Chethan Bhateja, Yichen Jian, Karl Pertsch, and Dorsa Sadigh. Re-mix: Optimizing data mixtures for large scale imitation learning. In *CoRL*, 2024. 10

[24] Jiaheng Hu, Rose Hendrix, Ali Farhadi, Aniruddha Kembhavi, Roberto Martin-Martin, Peter Stone, Kuo-Hao Zeng, and Kiana Ehsan. Flare: Achieving masterful and adaptive robot policies with large-scale reinforcement learning fine-tuning. *arXiv preprint arXiv:2409.16578*, 2024. 2, 3, 4, 5, 10, 11, 12, 14, 16

[25] Kushal Kedia, Prithwish Dan, and Sanjiban Choudhury. One-shot imitation under mismatched execution. *arXiv preprint arXiv:2409.06615*, 2024. 10

[26] Mukul Khanna, Ram Ramrakhya, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goat-bench: A benchmark for multi-modal lifelong navigation. In *CVPR*, 2024. 10

[27] 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. In *arXiv preprint arXiv:2403.12945*, 2024. 10

[28] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. In *CoRL*, 2024. 10

[29] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Kumar Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *ArXiv*, abs/1712.05474, 2017. 3

[30] Antonio Loquercio, Ana I Maqueda, Carlos R Del-Blanco, and Davide Scaramuzza. Dronet: Learning to fly by driving. In *RA-L*, 2018. 10

[31] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. ZSON: zero-shot object-goal navigation using multimodal goal embeddings. In *NeurIPS*, 2022. 10

[32] Soroush Nasiriany, Fei Xia, Wenhao 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. *ICML*, 2024. 11

[33] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey

Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *RSS*, 2024. 10

[34] Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 2

[35] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision, 2023. 10

[36] Austin Patel and Shuran Song. GET-Zero: Graph Embodiment Transformer for Zero-shot Embodiment Generalization. *CoRR*, abs/2407.15002, 2024. 10

[37] Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, Vladimir Vondrus, Théophile Gervet, Vincent-Pierre Berges, John M. Turner, Oleksandr Maksymets, Zsolt Kira, Mrinal Kalakrishnan, Jitendra Malik, Devendra Singh Chaplot, Unnat Jain, Dhruv Batra, Akshara Rai, and Roozbeh Mottaghi. Habitat 3.0: A Co-Habitat for Humans, Avatars and Robots. *CoRR*, abs/2310.13724, 2023. 10

[38] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. *arXiv preprint arXiv:2402.08191*, 2024. 2

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 16

[40] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Real-world humanoid locomotion with reinforcement learning. *Science Robotics*, 2024. 10

[41] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5163–5173. IEEE, 2022. 10

[42] Milad Shafiee, Guillaume Bellegarda, and Auke Ijspeert. Manyquadrupeds: Learning a single locomotion policy for diverse quadruped robots. In *ICRA*, 2024. 10

[43] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023. 2, 10

[44] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023. 2, 10

[45] Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *ICRA*, 2023. 2, 10

[46] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. 2

[47] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9 (11), 2008. 2

[48] Lirui Wang, Xinlei Chen, Jialiang Zhao, and Kaiming He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. In *NeurIPS*, 2024. 2, 10

[49] Saim Wani, Shivansh Patel, Unnat Jain, Angel X. Chang, and Manolis Savva. MultiON: Benchmarking Semantic Map Memory using Multi-Object Navigation. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 10

[50] Luca Weihs, Jordi Salvador, Klemen Kotar, Unnat Jain, Kuo-Hao Zeng, Roozbeh Mottaghi, and Aniruddha Kembhavi. AllenAct: A framework for embodied ai research. *arXiv preprint arXiv:2008.12760*, 2020. 3

[51] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 10

[52] Wenli Xiao, Haoru Xue, Tony Tao, Dvij Kalaria, John M Dolan, and Guanya Shi. Anycar to anywhere: Learning universal dynamics model for agile and adaptive mobility. *arXiv preprint arXiv:2409.15783*, 2024. 10

[53] Annie Xie, Lisa Lee, Ted Xiao, and Chelsea Finn. Decomposing the generalization gap in imitation learning for visual robotic manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3153–3160. IEEE, 2024. 2

[54] Mengda Xu, Zhenjia Xu, Cheng Chi, Manuela Veloso, and Shuran Song. Xskill: Cross embodiment skill discovery. In *CoRL*, 2023. 10

[55] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. In *CoRL*, 2024. 10

[56] Zhuo Xu, Hao-Tien Lewis Chiang, Zipeng Fu, Mithun George Jacob, Tingnan Zhang, Tsang-Wei Edward Lee, Wenhao Yu, Connor Schenck, David Rendleman, Dhruv Shah, et al. Mobility vla: Multimodal instruction navigation with long-context vlms and topological graphs. In *CoRL*. 11

[57] Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Sym-*

posium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation', pages 146–151. IEEE, 1997. 10

[58] Jonathan Yang, Catherine Glossop, Arjun Bhorkar, Dhruv Shah, Quan Vuong, Chelsea Finn, Dorsa Sadigh, and Sergey Levine. Pushing the limits of cross-embodiment learning for manipulation and navigation. In *arXiv preprint arXiv:2402.19432*, 2024. 2, 10

[59] Joel Ye, Dhruv Batra, Abhishek Das, and Erik Wijmans. Auxiliary tasks and exploration enable objectnav. *CoRR*, abs/2104.04112, 2021. 10

[60] Chengyang Ying, Hao Zhongkai, Xinning Zhou, Xuezhou Xu, Hang Su, Xingxing Zhang, and Jun Zhu. Peac: Unsupervised pre-training for cross-embodiment reinforcement learning. *Advances in Neural Information Processing Systems*, 37:54632–54669, 2024. 10

[61] Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon Ha. Hm3d-ovon: A dataset and benchmark for open-vocabulary object goal navigation. *IROS*, 2024. 10

[62] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024. 11

[63] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *CoRL*, 2022. 10

[64] Kuo-Hao Zeng, Luca Weihs, Ali Farhadi, and Roozbeh Mottaghi. Pushing it out of the way: Interactive visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 10

[65] Kuo-Hao Zeng, Zichen Zhang, Kiana Ehsani, Rose Hendrix, Jordi Salvador, Alvaro Herrasti, Ross Girshick, Aniruddha Kembhavi, and Luca Weihs. Poliformer: Scaling on-policy rl with transformers results in masterful navigators. In *CoRL*, 2024. 2, 4, 5, 10, 14, 15, 16

[66] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. *ICCV*, abs/2303.15343, 2023. 10

Appendices for *The One RING: a Robotic Indoor Navigation Generalist*

The following items are provided in the Appendix:

- *Problem formulation (App. 4)*
- *Related work (App. 5)*
- *Real-world robot platforms and human evaluation details (App. 6)*
- *Data collection using expert planners in simulation for randomized embodiments (App. 7)*
- *Full experimental setup (App. 8)*
- *Model architecture details (App. 9)*
- *Collision analysis (App. 10)*
- *Impact of using a more powerful visual encoder (App. 11)*
- *Visualization of the randomized training embodiments and their 5 nearest neighbors to each real robot (Stretch-RE1, LoCoBot, Unitree A1) (App. 12)*
- *Out-of-distribution generalization across embodiment parameters (App. 13)*
- *Limitations (App. 14)*

On our website (one-ring-policy.allen.ai), we have

- *Real-world qualitative videos of evaluating* RING *zero-shot on five different robot platforms, including Stretch RE-1 with our camera setup, Stretch RE-1 with factory camera configuration, LoCoBot, Unitree GO1, and RB-Y1 wheeled humanoid,*
- *Qualitative videos for human evaluation, using* RING *as navigation assistant,*
- *Videos showing our dataset of trajectories collected from random embodiments in simulation.*

4. Problem formulation

We define the space of possible embodiments as E, where each embodiment $e \in E$ is characterized by a configuration vector $\mathbf{c}_e$, including parameters such as camera settings, agent collider size, and center of rotation. Each task can be modeled as a Partially Observable Markov Decision Process (POMDP), denoted as $(S, A, E, O_e, T_e, R, L, P(s_0), \gamma)$, where S and A are the state and action spaces. The observation space O_e varies across embodiments due to differences in camera parameters. The observation at time t for embodiment e, $o_t^e = O_e(s_t, \mathbf{c}_e)$, is a function of both the state s_t and embodiment parameters $\mathbf{c}_e$. Given an action a_t, the next state follows the transition dynamics $s_{t+1} \sim T_e(s_{t+1}|s_t, a_t, \mathbf{c}_e)$, which depends on the embodiment (due to variations in collider size and rotation center). Fig. 2 shows trajectories from two different embodiments starting at the same location and following the same sequence of actions. They have distinct visual observations and follow different transition dynamics—one agent moves under the table, while the other collides with it.

Except where otherwise specified, we assume that all embodiments share the same discrete action space {`MoveBase(±20cm)`, `RotateBase(±6°, ±30°)`, `Done`}. These actions are executed using robot-specific low-level controllers during deployment. This simple and platform-agnostic action space enables effective cross-embodiment (XE) transfer, as demonstrated across both holonomic and differential-drive robots in our experiments. Investigating more expressive XE action spaces beyond this sufficient version is left for future work.

5. Related work

Cross-embodiment. Cross-embodiment training has received substantial attention from the research community. Arguably the most representative of a large body of recent work [3, 7, 13, 14, 19, 21, 23, 25, 28, 30, 33, 48, 54, 55, 58, 63], Open-X-Embodiment (OXE) [9] is the fruit of a large collaboration to cover many robotic tasks, with special emphasis in manipulation. Its usage in RT-X results in a notable performance gain in emergent skill evaluations in comparison to RT-2 [5]. Despite the 1.5 million trajectories across 22 embodiments present in their dataset, the enormous cost of data collection in the real world makes further scaling challenging. CrossFormer [13] trains a transformer-based policy on 900k trajectories spanning 30 robots, drawing from OXE, navigation data from GNM [43], manipulation data from DROID [27], and additional sources. However, the limited diversity of embodiments and focus on low-level control highlight the need for denser embodiment coverage. GET-zero [36], focused on dexterous manipulation, incorporates embodiment structure via a connectivity graph to guide attention. In contrast, we generate an arbitrarily large set of randomized embodiments during training, allowing our policy to generalize zero-shot to novel embodiments without requiring access to their structure.

Foundational navigation policies Following the success in recent developments for point-goal navigation [51], locomotion [4, 40, 42, 60], agile control [52], exploration [6, 57, 59], and social navigation [37], comparable results in more nuanced tasks like semantic or object-goal navigation (ObjectNav) [2, 15, 26, 31, 41, 49, 61, 64] remain elusive due to a lack of efficient exploration and semantic understanding capabilities. Recently, with powerful pretrained vision models [35, 66] and large-scale procedurally generated virtual environments [11], notable progress in end-to-end ObjectNav policy learning *for specific embodiments* has been achieved by means of imitation learning (IL) from shortest-path trajectories [17], RL [65], or combinations thereof [24]. In image-goal navigation, NoMaD [45], which extends ViNT [44], uses a diffusion policy to control a single embodiment. With the same goal in mind, GNM [43] trains navigation policies across 6 embodiments using IL. In contrast, our policy benefits from RL fine-tuning, improving robustness to compounding errors. Leveraging large-scale training with randomized embodi-

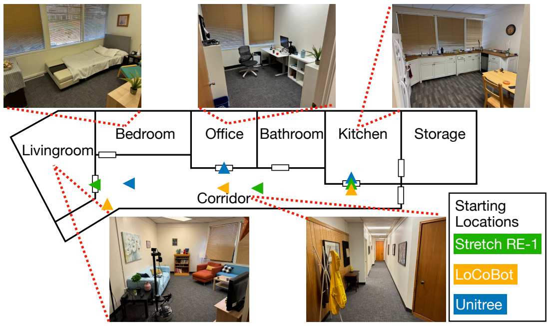

Figure 5. **Real Evaluation Environment**. Our real-world evaluations are performed in a multi-room apartment with a long corridor, shown here with the three starting locations for three different robots' evaluations.

ments in simulation, RING learns a single policy that generalizes to *any* embodiment, including *truly* unseen robot platforms in the real world. Unlike NoMaD, ViNT, GNM, and Mobility VLA [56], which rely on topological map or graph reconstruction for high-level planning, our approach is fully end-to-end and can explore dynamic novel scenes without requiring an explicit map. While prior work [1, 32, 62] has explored embodiment-agnostic policies using LLMs or VLMs, these methods are limited to short-horizon navigation and single-step prediction. In contrast, RING incorporates temporal context via a transformer decoder.

6. Real Robot Platforms and Human Evaluation Setup.

6.1. Stretch RE-1, LoCoBot, Unitree GO1, RB-Y1

We use Stretch RE-1, LoCoBot, Unitree GO1, and RB-Y1 wheeled humanoid as our robot platforms for real-world evaluations, shown in Fig. 7. For Stretch RE-1, we evaluate two different sensor configurations: the factory configuration and the configuration suggested by SPOC [17]. For RB-Y1, we include both standing and seated configurations and use an iPhone 16 Pro camera mounted on the robot to stream visual observations. The main differences among these platforms are summarized in Tab. 4. For robot movements, we either implement a Kalman filter or wrap around provided robot APIs to realize low-level controllers for a discrete action space {`MoveBase(±20cm)`, `RotateBase(±6°, ±30°)`, `Done`} across all platforms. It is important to note that during the training stage, we do not use any embodiment configurations from these robots to generate imitation learning data or to initialize RL finetuning embodiments.

6.2. Human Evaluations

To further show RING's generalization to unseen embodiments, we evaluate our policy as a navigation assistant with

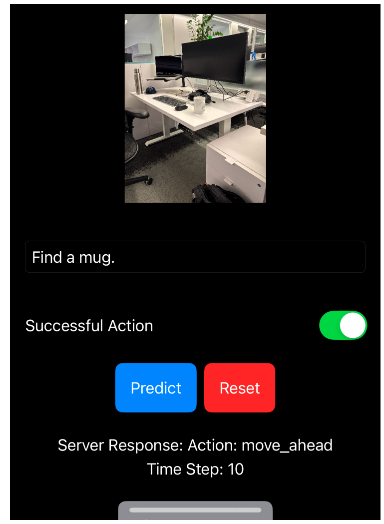

Figure 6. **iOS app for human evaluation**. We developed a simple iOS app that enables human participants to text goal, capture an image using iPhone's back camera, send both to a remote server, and receive the predicted action from our RING policy.

humans as novel embodiments. Five participants navigated a real-world kitchen by following policy outputs on their phones. Each had unique characteristics (e.g., step size, height, rotation, camera posture) and was tasked with reaching three objects (`Mug`, `Apple`, `Houseplant`), yielding 15 trajectories. We compare RING to FLaRe [24], trained only on Stretch RE-1. As shown in Tab.6, RING consistently outperforms FLaRe across objects and users. Fig.9 shows two qualitative examples. **Human participants**. We asked five human participants to use RING as a navigation assistant and evaluated its performance across a diverse range of human embodiments. These embodiment variations arose from differences in camera-holding posture, participant height, step size, and rotation angle. Tab. 5 summarizes these variations across participants. As a result, each participant contributed a distinct set of evaluation embodiments and sensor configurations.

Human Evaluation Details. We developed a simple iOS app (Fig. 6) that allows human participants to input a text prompt (e.g., "Find a mug"), capture an image using the iPhone's back camera, and send both the prompt and image to a remote server. The server processes this input using our RING policy, predicts action probabilities, samples an action, and returns it to the app for display. The action space available in the app mirrors that of our real-world robot.

Participants follow the suggested action at their own pace and chosen rotation degree, as specified in Tab. 5. After each step, they tap the `Predict` button to repeat the process: capturing a new image and sending it, along with the original prompt, to the server. This continues until either

	Stretch RE-1	Stretch RE-1 (Factory)	LoCoBot	Unitree GO1	RB-Y1 (standing)	RB-Y1 (seated)
Body dimension (cm)	33×34×141	33×34×141	35×35×89	64.5×28×40	60×69×140	60×69×92
Camera model	2× D455	D435	D435	D435	iPhone 16 Pro Camera	iPhone 16 Pro Camera
Camera vertical FoV	59°	69°	42°	42°	73°	73°
Camera horizontal FoV	90°	42°	68°	68°	53°	53°
Camera height (cm)	140	130	87	28	140	92
Camera pitch	27°	30°	0°	0°	0°	0°

Table 4. **Details about evaluation robot platforms.** Our four robot platforms have varying dimensions and camera configurations, resulting in diverse evaluation embodiments.

	H1	H2	H3	H4	H5
Height	6′3″	5′10″	5′5″	6′1″	5′11″
Step size	0.25m	0.35m	0.4m	0.3m	0.3m
Rotation Degrees	30°	45°	45°	35°	30°

Table 5. **Details about human evaluators.** Our five human participants have varying heights, step size, and rotation degrees, resulting in different evaluation embodiments.

Model	Train Embodiment	Object	Human Participants					Average
			H1	**H2**	**H3**	**H4**	**H5**	
FLaRE [24]	Stretch RE-1	🍎	✓	✗	✗	✗	✗	40.0%
		🌱	✗	✓	✓	✓	✓	
		☕	✗	✗	✗	✗	✓	
RING-ZERO-SHOT	RING-Random	🍎	✓	✗	✗	✗	✓	**73.3%**
		🌱	✓	✓	✓	✓	✓	
		☕	✗	✓	✓	✓	✓	

Table 6. **Human Evaluation.** Five individuals navigate to 3 different objects (🍎 `Apple`, 🌱 `Houseplant`, ☕ `Mug`) following the policy's output actions on their phones in a kitchen area (example trajectories in Fig. 9). RING-ZERO-SHOT shows much better generalization to human embodiment than the FLaRE baseline trained on Stretch RE-1.

the `Done` action is returned or 100 steps have been executed. An episode is considered successful if the target object is visible in the final image, within 1 meter, when RING issues the `Done` action.

Fig. 9 shows the layout of the evaluation scene and two sample trajectories, including two locations for a `Mug`, three for a `Houseplant`, and one for an `Apple`. Participants always begin in the bottom-left corner of the scene (results shown in Tab. 6).

7. Data Generation with Expert Planners

Expert planners introduced by [17] are not efficient and robust for random embodiments. As a result, we made major improvements to the planners to allow for better trajectories.

The major factor in this improvement is to consider *safety* of the policy (defined as the avoidance of approaching any obstacles along the way.) We use A* [20, 22] to generate safe navigation trajectories for training as follows: **1)** Extract reachable locations in a scene on a finely spaced grid, ensuring that the agent's collider does not intersect with any object's collider. Thus, different embodiments yield different reachable locations according to their collider. **2)** Compute a clipped Euclidean distance to the nearest obstacle. Then, for each location, set the cost of visiting it as the inverse of the third power of the distance. **3)** Construct a grid-like graph where each reachable location is a node connected to its immediate neighbors. For each connection, assign a cost equal to the maximum cost of visiting either of the two connected nodes. **4)** Extract a minimum-cost path connecting the reachable positions in the graph nearest to the source and to the target via A*. **5)** Extract waypoints by skipping over points in the A* path as long as skipping them doesn't increase the total path cost from the latest waypoint. **6)** The expert linearly interpolates between waypoints up to the precision reachable by the action space to generate each trajectory.

8. Additional Benchmark/Experiment Details

Action Space. Following on prior work with AI2-THOR, we discretize the action space for all agents in our training: {`MoveAhead`, `MoveBack`, `RotateRight`, `RotateLeft`, `RotateRightSmall`, `RotateLeftSmall`, `Done`}. Here, `MoveAhead` advances the robot by 0.2 meters, `MoveBack` moves the robot backward by 0.2 meters, `RotateRight`/`RotateRightSmall` rotates it clockwise by 30° / 6° around the yaw axis, and `RotateLeft`/`RotateLeftSmall` rotates it counterclockwise by 30° / 6° around the yaw axis, and `Done` indicates the agent has located the target, ending the episode. We evaluate RING zero-shot on all robots (Stretch-RE, LoCoBot, Unitree Go1) with the same action space using their low-level controllers. When finetuning for embodiment-specialized policies, we finetune for a slightly different action space for LoCoBot: {`MoveAhead`, `MoveBack`, `RotateRight`, `RotateLeft`, `LookUp`, `LookDown`, `Done`}. `LookUp` tilts the camera up by 30° around the roll axis and `LookDown` tilts the camera down by 30° around the roll axis. All baselines are trained and evaluated with the same action space for fair comparison.

Success Criteria. We follow the definition of Object Goal Navigation from [2], where an agent must explore its environment to locate and navigate to a specified object within a maximum of n steps. To indicate it has found the target, the agent must execute the `Done` action. Success is determined

Figure 7. **Robot platforms**. We use 3 different platforms, including Stretch RE-1, LoCoBot, and Unitree GO1 for our real-world evaluations.

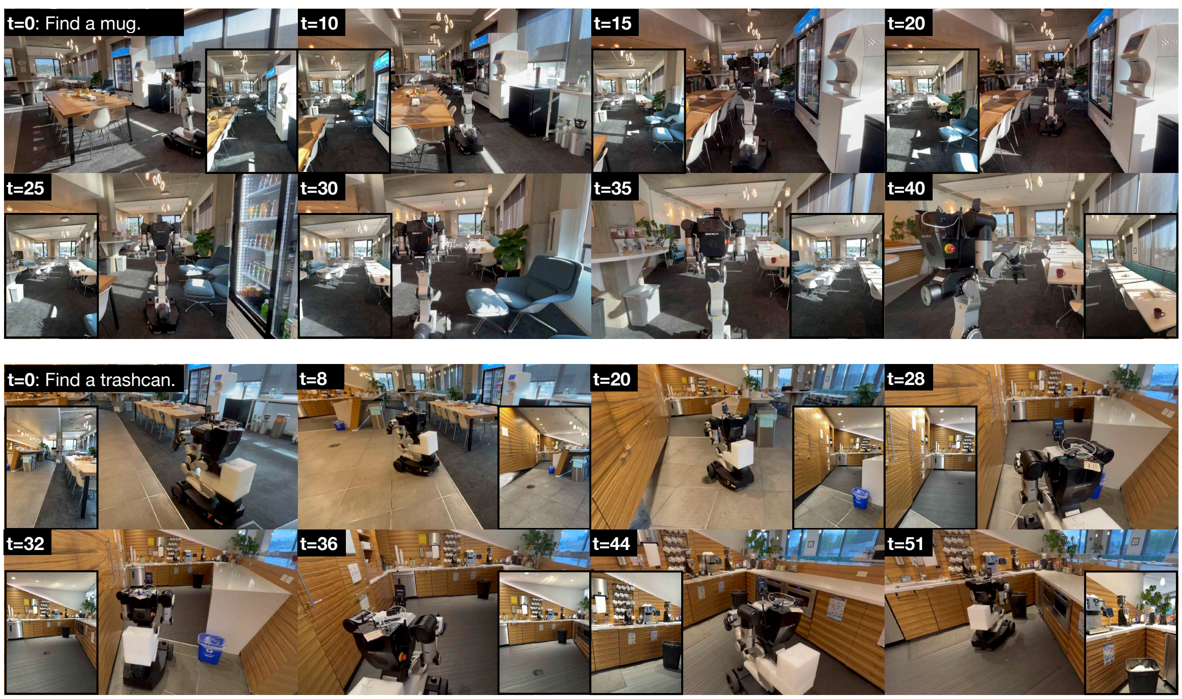

Figure 8. **RB-Y1 Trajectories.** We deploy the policy on the wheeled humanoid in an unstructured kitchen area (layout shown in Fig. 9) to navigate to different objects. We include both seated and standing configurations and use an iPhone 16 Pro camera to stream the visual observations.

by the environment based on whether the agent is within a distance d of the target and if the target is visible in its view. If the agent exceeds n steps without executing the `Done` action, the episode is considered a failure. For simulation benchmarks, we follow CHORES-S [17] with $n = 600$ and $d = 2$. For real-world evaluations, we use $n = 300$ and $d = 1$.

Success weighted by collision (SC). Collision is one of the main challenges for a unified policy operating across diverse embodiments in visual navigation tasks. Previous works measure the collision rate ($\frac{\#collisions}{\#steps}$) to understand how often a policy collides with objects in a scene. However, this does not reflect the effectiveness of the policy at the task level. For example, in a successful episode, a single collision and multiple collisions should have different impacts on the performance measurement. As a results, inspired from Success Weighted by Episode Length (SEL), we propose Success Weighted by Collision (SC),

$$SC = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{1}{1+c_i}, \tag{1}$$

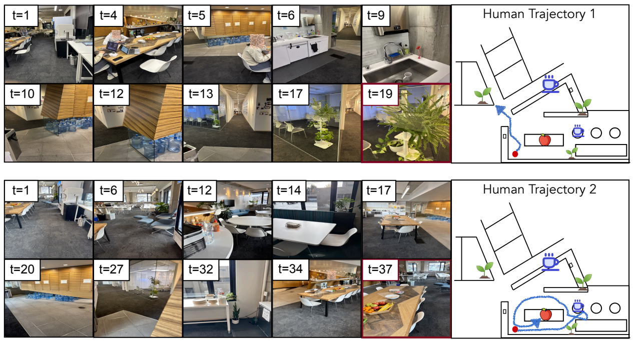

Figure 9. **Human Trajectories**. Two sample trajectories from two individuals navigating to a houseplant and an apple using RING.

where S_i is a binary indicator of success for episode i, c_i is the number of collisions in episode i, and N is the number of evaluation episodes. In this metric, the policy is penalized most heavily for a single collision, with the penalization decreasing for each additional collision, as the penalty diminishes inversely with the number of collisions. Intuitively, > 0 collisions are much worse than 0, as a real robot may suffer damage from one bad collision, but the difference between 10 and 11 collisions is a more marginal difference.

Hyparameters. We list the hyperparameters used in training and the architecture in Table 7.

Imitation Learning	
Batch Size	224
Context Length	100
Learning Rate	0.0002
RL Finetuning	
Total Rollouts	64
Learning Rate	0.0002
Mini Batch per Update	1
Update Repeats	4
Max Gradient Norm	0.5
Discount Value Factor γ	0.99
GAE λ	0.95
PPO Surrogate Objective Clipping	0.1
Value Loss Weight	0.5
Entropy Loss Weight	0.0
Steps for PPO Update	128
Model Architecture	
Transformer State Encoder Layers	3
Transformer State Encoder Hidden Dims	512
Transformer State Encoder Heads	8
Causal Transformer Deocder Layers	3
Causal Transformer Deocder Hidden Dims	512
Causal Transformer Deocder Heads	8

Table 7. **Hyperparameters for training and model architecture.**

8.1. Real-World Benchmarks

We evaluate 3 of the robots (Stretch-RE1, LoCoBot, Unitree Go1) in a multi-room apartment shown in Fig. 5. Based on the embodiment, the benchmark has different starting locations and objects. Among our target object categories, `Apple` can be found in the Living room and Kitchen, `Bed` can only be found in the Bedroom, `Sofa` and `Television` can only be found in the Living room, `Vase` can be found in the Livingroom, Corridor, Office, and Kitchen, `Chair` can be found in the Office and Kitchen, `HousePlant` can be found in the Living room, Office, and Kitchen.

We also deploy the policy on the RB-Y1 wheeled humanoid in an unstructured kitchen environment—similar to the human evaluation setting.

- **LoCoBot**: Following Phone2Proc [10], use the same five target object categories, including `Apple`, `Bed`, `Sofa`, `Television`, and `Vase`, and the three starting poses shown in 5.
- **Stretch RE-1**: We follow SPOC [16] to use the same six target object categories, including `Apple`, `Bed`, `Chair`, `HousePlant`, `Sofa`, and `Vase`, and the three starting poses, shown in Fig. 5. We consider 2 different camera configurations for Stretch: 1) off-the-shelf camera equipped on the Stretch RE-1 (D435 with a vertical field of view of 69° and resolution of 720×1280), 2) following [17], we use 2 Intel RealSense 455 fixed cameras, with a vertical field of view of 59° and resolution of 1280×720. The cameras are mounted facing forward but pointing downward, with the horizon at an angle of 27°.
- **Unitree Go1**: We create a new evaluation set for Unitree Go1 with 3 starting poses (Fig. 5) and 4 objects (`toilet, sofa, TV, trashcan`) positioned to accommodate the robot's lower height, ensuring that the objects can be visible from its lower viewpoint.
- **RB-Y1**: The wheeled humanoid navigates to various target objects in a real kitchen area, including a `mug`, `apple`, `houseplant`, and `trashcan` (2 example trajectories shown in Fig. 8).

9. Model Architecture Details

We will now detail RING's architecture (see Fig. 10), which is inspired by previous works POLIFORMER [65] and FLARE [24].

Visual encoder. We use the Vision Transformer from the pretrained SIGLIP-VIT-B/16 as our visual encoder. Since the RGB images vary in dimensions across different embodiments, we include an additional preprocessing step before feeding them into the encoder. Specifically, we pad each RGB image to a square and then resize it to 256×256. In addition, we mask the image from the

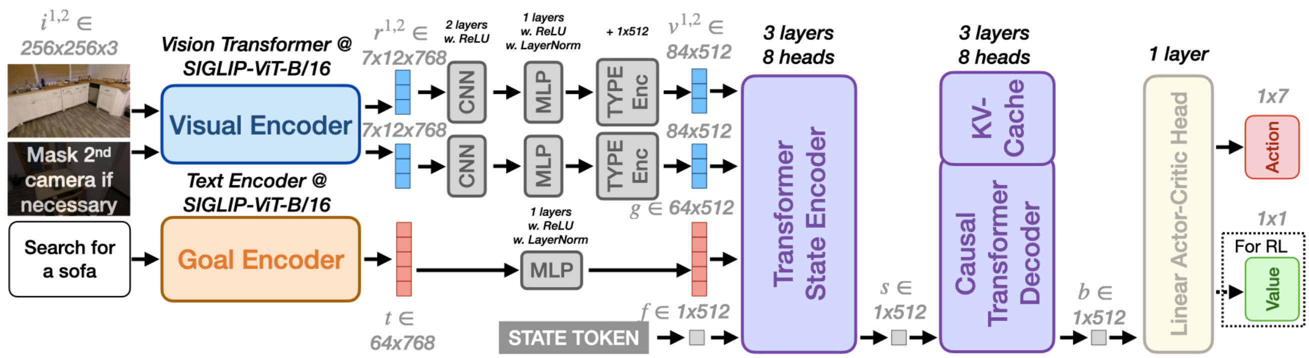

Figure 10. **RING Architecture.** Black text denotes module hyperparameters; gray text indicates hidden feature vectors. RING takes as input visual observations and a language instruction, and outputs an action. During RL finetuning, it also predicts a value estimate. For embodiments with only one camera (e.g., LoCoBot and Unitree), we mask the second camera image with zeros. We use the Vision Transformer and Text Encoder from SIGLIP-ViT-B/16 for visual and goal encoding, respectively. Their outputs are projected to v and g with dimension d. A Transformer State Encoder integrates v, g, and a `state` token embedding f into a state feature s. A Causal Transformer Decoder uses s and past experiences from a KV-Cache to produce a belief state b, which is passed to a Linear Actor-Critic Head to predict actions and (during finetuning) value estimates.

2^{nd} camera with zeros for the embodiments with only one camera. The visual backbone takes the RGB observation $i \in \mathbb{R}^{256\times256\times3}$ as input and produces a patch-wise representation $r \in \mathbb{R}^{\frac{256}{16}\times\frac{256}{16}\times h}$, where $h = 768$ is the hidden dimension of the visual representation. We reshape the visual representation into a $\ell \times h$ matrix, $\ell = 256\cdot256/16\cdot16$, and project the representation to produce $v \in \mathbb{R}^{\ell\times d}$, where $d = 512$ is the input dimension to the transformer state encoder. Note that since we have two RGB images from two cameras, we produce two visual representations $v^{1,2}$ at the end of this module. The vision encoder remains frozen through training.

Goal encoder. We follow the Text Transformer from the pretrained SIGLIP-ViT-B/16 to encode the given natural language instruction into goal embedding $t \in \mathbb{R}^{64\times h}$, where $h = 768$ is the hidden dimension and this Text Transformer returns 64 tokens after padding. Before passing the goal embedding to the transformer state encoder, we always project the embedding to the desired dimension $d = 512$, resulting in $g \in \mathbb{R}^{64\times512}$.

Transformer State Encoder. This module summarizes the state at each timestep as a vector $s \in \mathbb{R}^d$. The input to this encoder includes two visual representations $v^{1,2}$, the goal feature g, and an embedding f of a `STATE` token. These features are concatenated and fed to the non-causal transformer encoder. The output corresponding to the `STATE` token is the state feature vector $s \in \mathbb{R}^d$ which summarizes the state at each timestep. This feature vector is a goal-conditioned visual state representation.

Causal transformer decoder. We use a causal transformer decoder to perform explicit memory modeling over time. This can enable both long-horizon (e.g., exhaustive exploration with backtracking) and short-horizon (e.g., navigating around an object) planning. Concretely, the causal transformer decoder constructs its state belief b^t using the sequence of state features $\mathbf{s} = \{s^j|_{j=0}^{j=t}\}$ within the same trajectories. To avoid recomputing the attention on the previous state features, we follow PoliFormer [65] to use KV-Cache to store the past **K**ey and **V**alue into two cache matrices in each attention layer. Therefore, we only perform feedforward computation for the most current state feature s^t.

Linear actor-critic head. With the latest state belief b^t, we simply use a linear actor-critic head to project it to predict action logits over the action space. For RL-finetuning, the linear actor-critic head also predicts a value estimate about the current state.

		Nearest Neighbors				
	Stretch RE-1	N1	N2	N3	N4	N5
Camera Position (x) (meters	0	-0.06	0.11	0	-0.08	0.03
Camera Position (y) (meters	1.44	1.13	0.67	0.24	0.72	0.32
Camera Position (z) (meters	0.07	0.03	0.06	0.07	0.07	-0.03
Camera Pitch (degrees)	27	29	33	34	32	33
Camera Yaw (degrees)	0	0	0	0	0	0
Vertical FoV (degrees)	59	57	56	54	59	54
RGB Resolution (H)	224	224	224	224	224	224
RGB Resolution (Y)	396	394	394	396	396	398
Rotation Center (x) (meters)	0	0	0.09	-0.17	0	0.02
Rotation Center (z) (meters)	0.11	0.02	0.02	-0.08	0.04	-0.12
Collider Size (x) (meters)	0.34	0.23	0.28	0.49	0.33	0.24
Collider Size (y) (meters)	1.41	1.41	0.9	0.84	1.23	0.43
Collider Size (z) (meters)	0.33	0.27	0.41	0.29	0.44	0.38
distance	-	0.38	0.7	0.79	0.8	0.92

Table 8. **Five Nearest Neighbor Embodiments for Stretch RE-1 in Training Data.**

10. Collision analysis

RING is trained with randomized body dimensions and is not explicitly provided with real embodiment informa-

	Locobot	Nearest Neighbors N1	N2	N3	N4	N5
Camera Position (x) (meters)	0	-0.09	0.12	0.03	-0.06	-0.1
Camera Position (y) (meters)	0.87	1.01	0.81	0.39	0.85	0.42
Camera Position (z) (meters)	0	-0.1	-0.05	-0.1	-0.02	0.09
Camera Pitch (degrees)	0	0	0	-1	0	1
Camera Yaw (degrees)	0	0	0	0	0	0
Vertical FoV (degrees)	42	45	44	42	45	45
RGB Resolution (H)	224	224	224	224	224	224
RGB Resolution (Y)	396	396	394	394	392	392
Rotation Center (x) (meters)	0	0.04	0.1	0.1	-0.02	-0.15
Rotation Center (z) (meters)	0	-0.13	0	0.13	0.02	-0.12
Collider Size (x) (meters)	0.35	0.27	0.36	0.37	0.27	0.42
Collider Size (y) (meters)	0.89	1.28	1.23	0.86	1.46	0.59
Collider Size (z) (meters)scale z	0.4	0.43	0.23	0.36	0.36	0.45
distance	-	0.18	0.22	0.34	0.41	0.42

Table 9. **Five Nearest Neighbor Embodiments for LoCoBot in Training Data.**

	Unitree A1	Nearest Neighbors N1	N2	N3	N4	N5
Camera Position (x) (meters)	0.01	0.08	0.03	-0.01	-0.04	0.1
Camera Position (y) (meters)	0.3	0.56	0.37	0.85	0.55	0.82
Camera Position (z) (meters)	0.27	-0.11	0.06	0	0.12	0.02
Camera Pitch (degrees)	0	-3	-2	-4	-5	-5
Camera Yaw (degrees)	0	0	0	0	0	0
Vertical FoV (degrees)	42	49	49	51	50	51
RGB Resolution (H)	270	224	224	224	224	224
RGB Resolution (Y)	480	448	446	448	446	446
Rotation Center (x) (meters)	0	-0.07	0.05	-0.07	-0.09	-0.14
Rotation Center (z) (meters)	0.04	-0.02	0	-0.12	0.12	0.11
Collider Size (x) (meters)	0.3	0.46	0.27	0.35	0.27	0.49
Collider Size (y) (meters)	0.34	1.24	0.45	1.47	0.67	1.39
Collider Size (z) (meters)	0.64	0.34	0.37	0.36	0.33	0.39
distance	-	0.76	0.78	1.04	1.1	1.12

Table 10. **Five Nearest Neighbor Embodiments for Unitree A1 in Training Data.**

Model	Train Embodiment	Safe Episode ↑ Stretch	LoCoBot	Unitree A1
POLIFORMER [65]	Stretch	45.0	-	-
	LoCoBot	-	42.0	-
	Unitree A1	-	-	**49.25**
FLARE [24]	Stretch	62.0	-	-
RING-ZERO-SHOT	RING-Random	**67.0**	**64.5**	48.5

Table 11. RING has more **Safe Episodes** (with no collisions) compared to embodiment-specific baselines.

tion. Regardless, the learned policy remains as safe—*and in some cases even safer*—than embodiment-specific policies (Tab. 11). Collision-avoidance behavior can be further improved by incorporating a small collision penalty into the RL reward (Tab. 12).

RING learns to take conservative paths without explicit knowledge of its collider size. Expert trajectories are generated using A*, finding minimum-cost paths where the cost is defined as the inverse of the Euclidean distance to the nearest obstacles. During reinforcement learning, collisions slow down the agent's progress, leading to lower rewards through a step penalty. Since collisions cause no meaningful state changes and only waste time, the policy learns to avoid them to complete tasks more efficiently. We evaluate RING against embodiment-specific baselines using the metric *Safe Episodes*—the percentage of episodes completed without any collisions. As shown in Tab. 11, RING achieves a higher percentage of Safe Episodes compared to the baselines. By training across a large number of embodiments and without access to exact body size information, RING learns to take more conservative actions through reinforcement learning, promoting safer navigation.

Model	Collision Penalty	Metrics Success ↑	SEL ↑	SC ↑	CR ↓	Safe Episode ↑
RING	✗	**67.62**	56.24	42.53	7.77	46.90
	✓	66.33	**56.87**	**49.05**	**4.03**	**60.57**

Table 12. **Collision Penalty.** Adding a small collision penalty (0.1) to the reward function results in 50% less collision, forcing the policy to take more conservative paths.

Include collision penalty to take safer routes. Adding a small collision penalty of 0.1 to the reward function can further reduce collision rate (*CR*) by 50%. The resulting policy is more conservative, regardless of embodiment size. To quantify these results, we created a custom benchmark similar to CHORES-$\mathbb{S}$ [17], consisting of $2{,}000$ random embodiments across $2{,}000$ scenes. We evaluate 2 different versions of our policy on this benchmark, comparing metrics such as *Success Rate*, *Success Weighted by Collision (SC)*, *Collision Rate (CR)*, and *Safe Episode*. As shown in Tab. 12, adding the collision penalty reduces the collision rate (CR) ($7.77\% \rightarrow 4.03\%$) as well as increases the percentage of trajectories without collisions ($46.90\% \rightarrow 60.57\%$).

11. A More Powerful Pretrained Visual Encoder.

The default vision encoder used in our policies is the pretrained SIGLIP-VIT-B/16. In this section, we examine the impact of using a more powerful visual encoder on RING's performance. We train RING-LARGE using OpenAI's VIT-L/14 336PX CLIP model [39]. Table 13 compares the results, showing that a stronger visual encoder significantly improves zero-shot performance across all four embodiments (approximately 9% improvement on average). A larger visual encoder is particularly beneficial in our policy, as the visual observations are highly varied due to randomized camera parameters. To ensure fair comparison with the baselines and because VIT-L/14 is more computationally demanding, we chose to use the VIT-B/16 encoder for our main experiments. We will release the training code for the community for those interested in training with the larger visual encoder.

12. Nearest Neighbor Embodiments to Real Robots in our Training Data

Fig. 12 presents a t-SNE visualization of the embodiment parameters $\mathbf{c}_e \in \mathbb{R}^{19}$ for 50k samples from the random embodiments in our training set (examples showin in Fig. 11). We also show the corresponding parameters for Stretch, Lo-

Figure 11. **Random embodiments in the AI2-THOR simulator**. Right column shows the egocentric view from the main camera and the left column shows a third-person view of the agent –white boxes indicate the robot colliders for visualization purposes only.

Model	Visual Encoder	Benchmark Embodiment			
		Stretch	**Stretch** (Nav)	**LoCoBot**	**Unitree A1**
RING	SIGLIP-VIT-B/16	76.0	74.0	66.5	72.0
RING-LARGE	VIT-L/14 336PX CLIP	**83.8**	**77.7**	**75.3**	**79.9**

Table 13. **A Stronger Visual Encoder**. Using a more powerful vision encoder significantly improves the zero-shot performance across all embodiments.

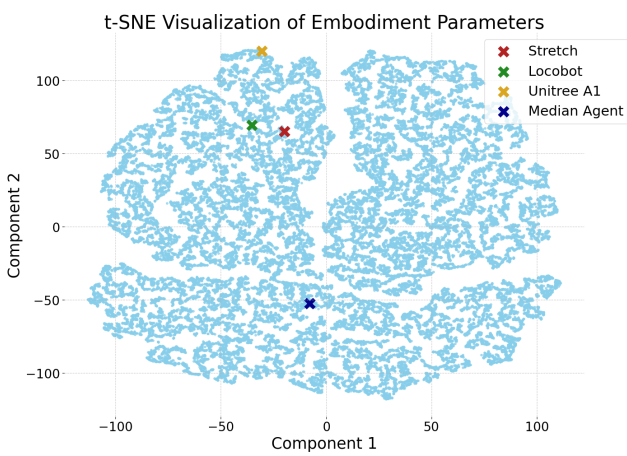

Figure 12. t-SNE visualization of the embodiment parameters $\mathbf{c}_e \in \mathbb{R}^{19}$ for 50k random agents. The three specific robots are also shown for visualization (they are not included in our training set).

CoBot, and Unitree A1 for visualization purposes. Our random embodiments range widely over the space of possible embodiments, with many closely approximating each of the three real robots. Tables 8, 9, and 10 list the five nearest neighbors to each robot in the compressed t-SNE space and their corresponding embodiment parameters. Although the nearest neighbors do not exactly match each robot's embodiment, they are sufficiently similar across different parameters. This extensive coverage of the embodiment space and proximity to real-world embodiments ensure consistent

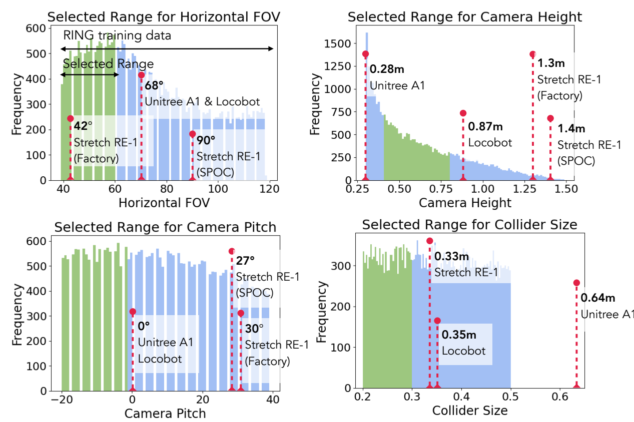

Figure 13. **Selected training ranges for the four embodiment parameters (camera height, camera FOV, camera pitch, and collider size)**. The green regions represent the narrower ranges used during training, excluding the values corresponding to the real robots. The results of policies trained on each selected range are presented in Table 14.

zero-shot generalization to all three robots.

13. Generalization to out-of-distribution embodiment parameters

The random embodiments in our training set span a wide range of possible configurations, with many closely approximating each of the three real robots. Although the training data covers the full range of each embodiment parameter individually, the specific combination of parameters corresponding to each real robot is not explicitly included. This is demonstrated by the nearest-neighbor embodiments shown in Appendix 12. In this section, we examine the extent to which the policy generalizes to out-of-distribution values of individual embodiment parameters.

We focus on four specific parameters: *camera height, camera field of view (FOV), camera pitch*, and *collider size*. For each parameter, we define a narrower range that ex-

Embodiment Parameter	Training Range	Success Rate↑ / Collision Rate ↓				
		Stretch	**Stretch** (Factory Config)	**LoCoBot**	**Unitree A1**	**Average**
Camera Height	[0.4, 0.8]	55.3 / **9.3**	51.0 / 9.8	51.5 / 9.8	59.3 / 9.2	54.3 / 9.5
Camera FoV	[40, 60]	54.0 / 14.3	51.6 / 15.5	53.4 / 12.8	**61.5** / 11.5	55.1 / 13.5
Camera Pitch	[-20, -2]	54.5 / 12.9	53.5 / 9.7	**56.5** / 11.2	59.8 / 12.7	56.1 / 11.6
Collider Size	[0.20, 0.32]	**60.5** / 18.0	53.5 / 21.4	55.0 / 14.9	54.0 / 18.6	55.7 / 18.2
No Filter	-	58.8 / 9.6	**60.0 / 9.5**	**56.5 / 7.9**	60.9 / **8.3**	**59.1 / 8.8**

Table 14. **Out-of-distribution (OOD) generalization for different embodiment parameters.** For each of the four embodiment parameters, we select 50k random embodiments from a narrow training range that excludes the parameter values of the real robots (as shown in Fig. 13). Zero-shot evaluations are performed on four real robots, each with parameter values outside the training distribution. Success and collision rates are reported for each robot and averaged across all robots.

Model	Ablations	Success ↑	SEL ↑	SC ↑	CR ↓	Safe Episode ↑
	Body Config					
RING	✗	67.62	56.24	42.53	**7.77**	**46.90**
RING-EMB-COND	✓	**69.44**	**57.42**	**44.69**	8.0	46.54

Table 15. **Conditioning RING on embodiment parameters**. We explicitly provide the embodiment parameters to the policy (RING-EMB-COND) and compare with RING without any information about the embodiment. Both policies are evaluated on a custom benchmark consisting of 2000 random embodiments in 2000 scenes.

cludes the values corresponding to the real robots. From the training data, we filter the random embodiments to select 50k samples within each of these specified ranges. For comparison, we also train a version of the policy using 50k unfiltered embodiments that span the full range of each parameter. The selected training ranges for each parameter are illustrated in Fig. 13.

We then perform zero-shot evaluations of the policies trained on each selected range using four robots whose parameters lie outside the training ranges. The success rate and collision rate are summarized in Table 14. The results indicate that policies trained on narrower ranges still generalize to out-of-distribution parameters, achieving only a slightly lower success rate. However, evaluation on unseen embodiment parameters leads to a significantly higher collision rate, particularly for the policy trained with a narrower range of collider sizes. This suggests that the agent may rely more on physical contact with the environment to infer its embodiment configurations. Comparing Table 14 and Table 2, the average success rate drops by 13%, emphasizing that the number of random embodiments used during training is crucial to develop an embodiment-agnostic policy capable of effectively handling a wide range of embodiments.

14. Limitations

Although RING has the advantage of being deployable on a wide range of embodiments without any privileged information about its current body, when available it may be beneficial to have a policy explicitly conditioned on the current embodiment specification. This might lead to improved performance and more desirable behaviors, such as increased efficiency and collision avoidance.

We train RING-EMB-COND by explicitly providing the embodiment information to the policy. The embodiment parameters are represented as a configuration vector $\mathbf{c}_e \in \mathbb{R}^{19}$, with each dimension corresponding to a specific embodiment parameter listed in Table 1. This information is passed as an additional token to the Transformer State Encoder. We use a simple MLP to project $\mathbf{c}_e$ to the desired feature dimension $e \in \mathbb{R}^{1\times512}$ before passing it to the encoder. Tab. 15 evaluates the 2 versions of the policy on our custom benchmark consisting of 2,000 random embodiments across 2,000 scenes, comparing metrics such as *Success Rate*, *Success Weighted by Collision (SC)*, *Collision Rate (CR)*, and *Safe Episode (percentage of episodes without any collisions)*.

The results do not show a clear benefit to conditioning the policy on embodiment information. This could be due to several reasons. It is possible that most relevant information about environment hazards and agent motion can be already inferred from visual observations. It is also possible that a significant fraction portion of collisions (both with an without embodiment specification provided) occur with objects that never enter the agent's visual field, in which case extra information about its own embodiment would not help. Alternatively, a more effective method for conditioning the policy on the parameters may exist. Future work should explore this with additional examination of agent-environment collision and designing improved policy architectures to better integrate embodiment parameters, ultimately training a more efficient and robust policy that explicitly incorporates embodiment information.