2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

COMPASS: Cross-embOdiment Mobility Policy via ResiduAl RL and Skill Synthesis

Wei Liu[1], Huihua Zhao[1], Chenran Li[1], Yuchen Deng[1], Joydeep Biswas[1,2], Soha Pouya[1], Yan Chang[1]

***Abstract*— As robots are increasingly deployed in diverse application domains, enabling robust mobility across different embodiments has become a critical challenge. Classical mobility stacks, though effective on specific platforms, require extensive per-robot tuning and do not scale easily to new embodiments. Learning-based approaches, such as imitation learning (IL), offer alternatives, but face significant limitations on the need for high-quality demonstrations for each embodiment.**

To address these challenges, we introduce COMPASS, a unified framework that enables scalable cross-embodiment mobility using expert demonstrations from only a single embodiment. We first pre-train a mobility policy on a single robot using IL, combining a world model with a policy model. We then apply residual reinforcement learning (RL) to efficiently adapt this policy to diverse embodiments through corrective refinements. Finally, we distill specialist policies into a single generalist policy conditioned on an embodiment embedding vector. This design significantly reduces the burden of collecting data while enabling robust generalization across a wide range of robot designs. Our experiments demonstrate that COMPASS scales effectively across diverse robot platforms while maintaining adaptability to various environment configurations, achieving a generalist policy with a success rate approximately 5X higher than the pre-trained IL policy, and further demonstrates zero-shot sim-to-real transfer.

Project page: `https://nvlabs.github.io/COMPASS`

I. Introduction

While robotics has made significant strides in both industry and daily life, achieving robust mobility across diverse robot embodiments remains a fundamental open challenge. Morphological differences in hardware, kinematics and sensing configurations [1]–[3] create large variations in dynamics and perception, complicating efforts to build a universal and adaptable mobility policy for the real world.

Classical mobility stacks [4], [5] excel in specific robots, especially wheeled platforms, but often require extensive manual retuning or redevelopment when ported to new embodiments with distinct sensor suites and physical constraints. This reliance on per-robot engineering has motivated interest in end-to-end learning approaches, particularly imitation learning (IL) [6], [7], which aim to scale mobility policies across multiple robots by leveraging data-driven training rather than manual redesign.

However, scaling IL to diverse embodiments introduces a major bottleneck: each new robot typically requires its own set of high-quality demonstrations. As morphological and sensor differences increase, so does the need for additional embodiment-specific data. For complex modalities such as humanoids, collecting demonstrations can be prohibitively expensive or impractical. In parallel, IL can also suffer from the distribution shift [8], leading to failures when the policy encounters out-of-distribution states during deployment. While advances in machine learning architectures [9]–[11] and data augmentation techniques [12] help mitigate the distribution shift within a given embodiment, extending these methods to accommodate greater morphological diversity greatly amplifies data requirements and training complexity, further complicating the scalability of pure IL approaches.

Nevertheless, we observe that many key aspects of mobility are inherently shared across robot embodiments, such as environmental understanding, obstacle avoidance, and goal reaching. This observation motivates our central insight: it should be possible to build a cross-embodiment mobility policy by leveraging expert demonstrations from only a single embodiment.

Based on this principle, we propose COMPASS, a unified three-stage workflow that combines imitation learning, residual reinforcement learning, and policy distillation, as shown in Figure 1. We first pre-train a mobility policy on a single embodiment using IL, leveraging expert demonstrations to train a shared representation of mobility priors over environmental states and robot actions. Following the X-Mobility approach [6], this pre-trained model is decomposed into two components: (i) a world model that encodes environmental dynamics, and (ii) a policy that learns to produce actions. To efficiently adapt this policy to various embodiments, we integrate residual reinforcement learning (RL) [13]–[15] into our workflow. Rather than training policies from scratch, residual RL enables each embodiment-specific specialist to refine the pre-trained policy through small corrective adjustments, improving closed-loop performance with fewer environment interactions. Finally, we collect data from these specialists and apply policy distillation to merge their expertise into a single cross-embodiment model, conditioned on an embodiment embedding vector. This workflow amortizes the demonstration effort, enabling robust generalization across diverse platforms while requiring expert demonstrations from only a single embodiment. Experiments show that COMPASS achieves a 5X increase in task success rate and a 3X improvement in travel efficiency compared to the IL-only policy, demonstrating both its effectiveness and scalability.

To the best of our knowledge, this three-stage framework is the first end-to-end pipeline for cross-embodiment mobility that systematically addresses morphological diversity, domain shift, and data efficiency. Our key contributions are as follows:

- We propose a unified three-stage workflow that com-

[1]Wei Liu, Huihua Zhao, Chenran Li, Yuchen Deng, Joydeep Biswas, Soha Pouya, and Yan Chang are with Nvidia, Santa Clara, CA, USA ({liuw, huihuaz, chenranl, yuchendeng, jbiswas, spouya, yachang}@nvidia.com).

[2]Joydeep Biswas is also with The University of Texas at Austin, Austin, TX, USA. (joydeepb@cs.utexas.edu)

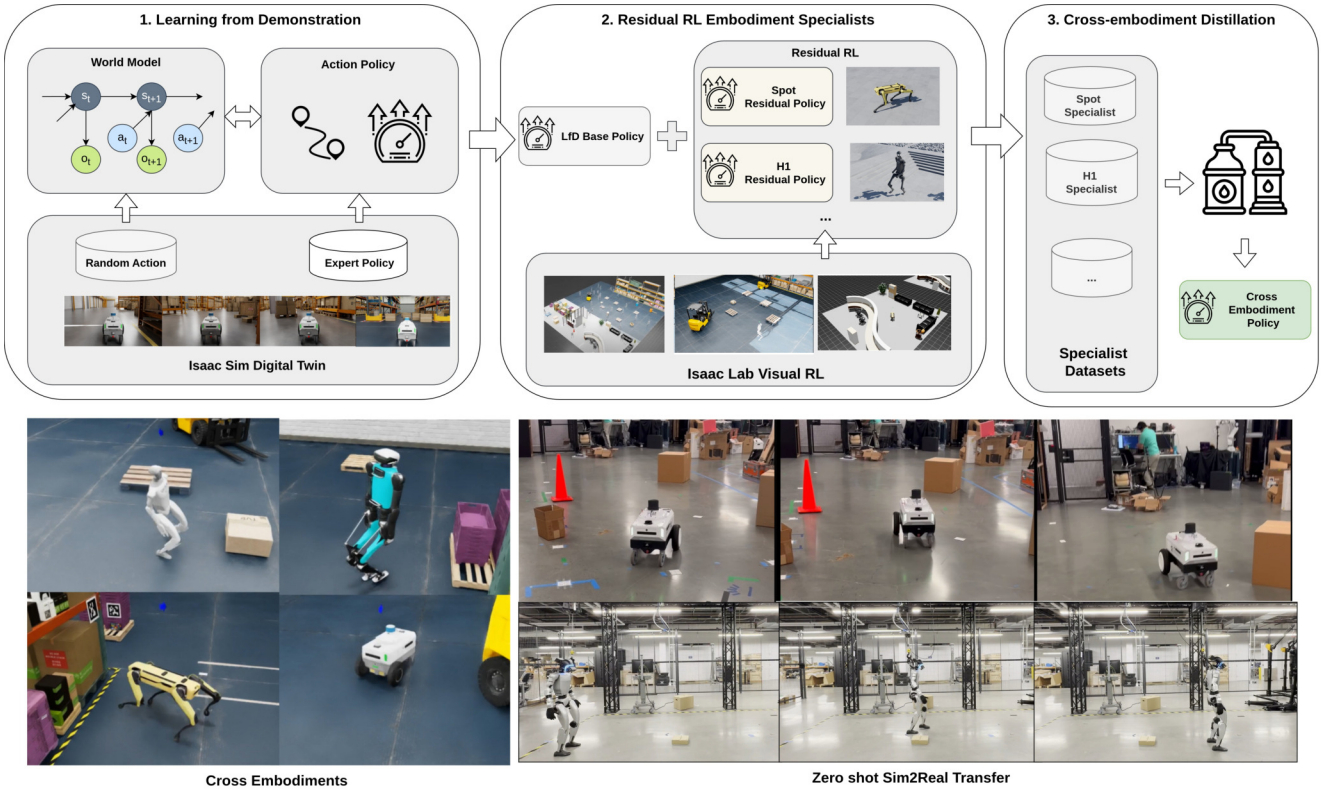

Fig. 1: High-level overview of the COMPASS workflow: (1) Imitation learning produces a base policy and world model using readily available teacher policies on a mobile robot. (2) Residual RL fine-tunes the base policy for multiple embodiments, optimizing for physical constraints and sensor modalities. (3) Policy distillation consolidates these embodiment-specialist policies into one robust cross-embodiment policy. The learned policy can transfer across heterogeneous embodiments and achieve zero-shot sim-to-real transfer in cluttered environments.

bines imitation learning, residual reinforcement learning, and policy distillation to achieve scalable, data-efficient cross-embodiment mobility from demonstrations collected on a single embodiment.
- We develop a visual RL approach that leverages a world model to efficiently refine policies across embodiments by incorporating temporal and spatial structure.
- We demonstrate extensive benchmarks across diverse robot platforms, showing that the proposed approach achieves robust generalization while preserving the adaptability needed to succeed in varied environments and zero-shot sim-to-real transfer.

II. Method

We present a three-stage workflow aimed at building robust cross-embodiment mobility policies (see Fig. 1). First, we train a base policy via IL, which captures general mobility priors from teacher demonstrations on mobile robots. Next, we refine this base policy into embodiment-specific specialists via residual RL. Finally, policy distillation combines these specialists into a single model suitable for multi-platform deployment.

A. Problem Statement

We focus on the task of point-to-point mobility across different robotic embodiments, each characterized by unique kinematics and dynamics. At time step t, let the robot observe a state

$$\mathbf{x}_t = \big(\mathbf{I}_t, \mathbf{v}_t, \mathbf{g}_t, \mathbf{e}\big),$$

where $\mathbf{I}_t$ is current camera input (RGB images), $\mathbf{v}_t$ is the measured velocity, $\mathbf{g}_t$ provides route or goal-related information (e.g., goal position in robot frame), and $\mathbf{e}$ is an embodiment embedding that specifies the robot's morphology. Although $\mathbf{e}$ remains constant for a single robot during an episode, it varies across different embodiments.

We aim to learn a policy π_θ that maps $\mathbf{x}_t$ to a velocity command $\mathbf{a}_t = (v_t, \omega_t)$, which is then consumed by a low-level controller for joint-level actuation. The environment's transition dynamics $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{a}_t)$ depend on both the robot's embodiment and external factors in the scene. We define a reward function $R(\cdot)$ that encourages efficient, collision-free progress to the goal. The objective is to maximize the expected discounted return

$$J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R(\mathbf{x}_t, \mathbf{a}_t)\right].$$

The challenge is to design a single policy that leverages the embodiment embedding $\mathbf{e}$, allowing shared knowledge yet accommodating distinct morphological constraints.

B. Step 1: Imitation Learning for Mobility Priors

The initial step uses IL to acquire a generic mobility baseline. We rely on readily available teacher policies—often

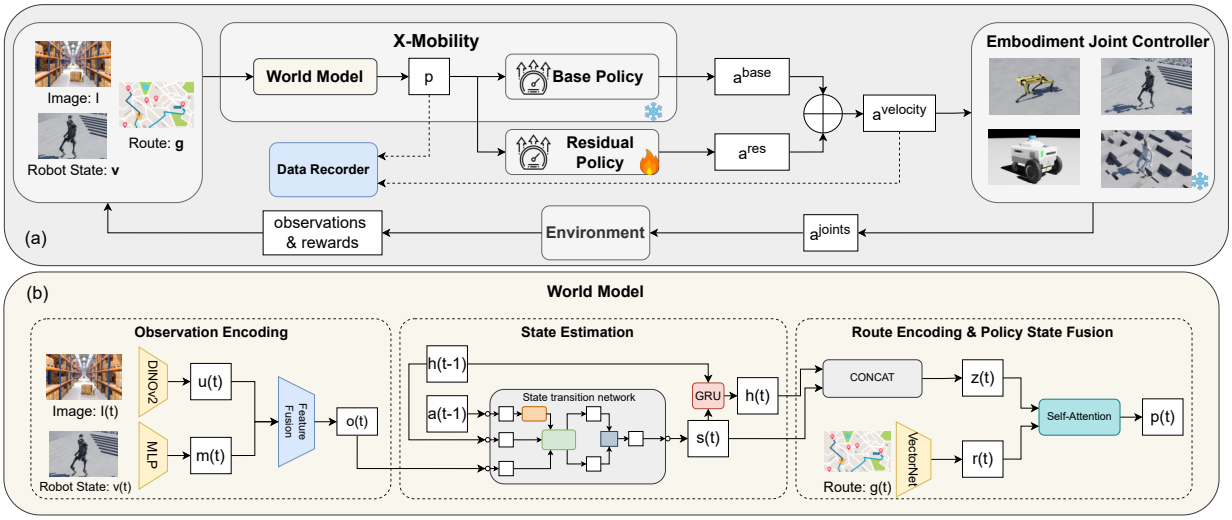

Fig. 2: Residual RL architecture: (a) residual RL loop and (b) world model architecture. The world model processes the same inputs as the IL approach to produce the policy state, while the imitation-learned base policy generates a base action. The residual policy refines this action with a correction term, producing the final velocity command for embodiment-specific joint controllers. With the joint actions, the robot interacts with the environment and receives new observations and rewards. The data recorder records the pairs of policy state and action for policy distillation.

classical mobility stacks—for standard mobile robots, which typically provide reliable demonstrations.

1) Latent State Modeling: We introduce a latent state $\mathbf{s}_t$ to capture environment dynamics. Let $\mathbf{o}_t = (\mathbf{I}_t, \mathbf{v}_t)$ denote raw observations, including RGB images and robot velocities. Our goal is to learn a world model that predicts transitions in this latent space:

$$\mathbf{s}_t = f_\phi(\mathbf{s}_{t-1}, \mathbf{a}_{t-1}), \quad \widehat{\mathbf{o}}_t = g_\psi(\mathbf{s}_t),$$

where f_ϕ updates the latent state based on the previous action, and g_ψ attempts to reconstruct or predict $\mathbf{o}_t$. Training minimizes reconstruction and predictive losses over a dataset of expert demonstrations

$$\mathcal{D} = \left\{(\mathbf{o}_0, \mathbf{a}_0^\star, \ldots, \mathbf{o}_T, \mathbf{a}_T^\star)\right\},$$

where $\mathbf{a}_t^\star$ are the expert (teacher) actions.

2) Policy Learning in Latent Space: Having learned a latent transition, we then train a policy π_θ^{IL} that takes policy state $\mathbf{p}_t$, which is fused from the latent state $\mathbf{s}_t$ and route embedding $\mathbf{r}_t$, to predict $\mathbf{a}_t$:

$$\begin{aligned}\mathbf{r}_t &= f_\theta(\mathbf{g}_t),\\ \mathbf{p}_t &= \Phi(\mathbf{s}_t, \mathbf{r}_t),\\ \mathbf{a}_t &= \pi_\theta^{\text{IL}}(\mathbf{p}_t),\end{aligned}$$

where f_θ denotes route encoding and Φ is a learnable feature fusion block.

We minimize the action discrepancy between the policy's outputs and the teacher's actions:

$$\min_\theta \sum_t \ell\left(\pi_\theta^{\text{IL}}(\mathbf{p}_t), \mathbf{a}_t^\star\right),$$

where $\ell(\cdot)$ could be a simple regression loss. This yields an IL-based mobility priors. The world model helps the policy generalize to out-of-distribution states by predicting future observations and latent transitions, thereby providing a robust encoded representation for decision-making.

3) X-Mobility: We leverage the X-Mobility [6] as our base policy, which integrates an autoregressive world model (see Fig. 2) with a velocity-prediction policy model. The learned latent state $\mathbf{s}_t$ encapsulates environment dynamics and constraints, while the policy head combines this state with route information to generate velocity commands. X-Mobility's strong generalization performance indicates that its learned representation readily adapts to different embodiments.

C. Step 2: Residual RL to Fine-Tune Specialists

Having trained a promising general mobility policy through IL, we refine it for embodiment-specific needs via residual RL. This stage addresses robot-specific kinematics, sensor configurations, and other constraints that the base IL policy may not fully capture.

1) Residual Policy Setup: Let $\mathbf{a}_t^{\text{base}} = \pi_\theta^{\text{base}}(\mathbf{p}_t)$ be the action from the IL baseline. We introduce a residual policy π_θ^{res} that takes residual policy state $\hat{\mathbf{p}}_t$ and outputs $\mathbf{a}_t^{\text{res}}$. The final action is

$$\mathbf{a}_t = \pi_\theta^{\text{base}}(\mathbf{p}_t) + \pi_\theta^{\text{res}}(\hat{\mathbf{p}}_t).$$

The role of π_ϕ^{res} is to adapt the base policy to nuances of a specific embodiment's characteristics.

2) Residual Policy State: To enable effective residual policy learning, we introduce a residual policy state $\hat{\mathbf{p}}_t = \mathcal{G}(\mathbf{p}_t)$, which augments the base policy state $\mathbf{p}_t$ with additional task-relevant information. This design allows the residual policy to complement the base policy by incorporating inputs that were not considered during the base policy training. For example, if the base policy omits goal heading information, we can inject its embedding into the residual policy state, allowing the final composite policy to account for the desired heading and achieve more stable goal-reaching behavior. This modular formulation offers flexibility to extend the capabilities of the base policy without retraining it from scratch.

3) Residual Policy Network Architecture: The residual policy network reuses the same policy model as in the base policy. We initialize it by copying the weights of the base action policy and introducing a linear projection layer to align the dimensionality between the base policy state and the residual policy state. The final output layer is reinitialized to exclusively learn the residual correction. This initialization strategy stabilizes training and encourages the residual component to specifically address embodiment-specific discrepancies. The critic network is implemented as a standard multi-layer perceptron (MLP), taking the same residual policy state as input.

4) Reward Design: We define a reward function R to promote safe and efficient mobility, consisting of the following components:

- Progress: Positive reward proportional to the reduction in distance to the goal.
- Collision avoidance: Penalties for collisions or fall downs.
- Goal completion: Large positive reward on reaching the destination with zero velocities.

We adopt this simple formulation to facilitate training, while acknowledging that more sophisticated reward shaping could potentially yield better performance.

5) Training Loop: We employ a PPO-based RL optimizer [16] for the residual policy π_ϕ^{res}. As illustrated in Fig. 2, each training iteration proceeds as follows:

1) The agent receives the current state $\mathbf{x}_t$, processes it through the world model to form the policy state $\mathbf{p}_t$, then generates the base action $\mathbf{a}_t^{\text{base}}$ from the IL policy and the residual action $\mathbf{a}_t^{\text{res}}$ from the residual network.
2) The combined action $\mathbf{a}_t$ is executed in the simulation environment via an embodiment-specific joint controller.
3) The agent observes the next state $\mathbf{x}_{t+1}$ and the reward R associated with the transition.
4) The residual policy π_ϕ^{res} is updated via gradient-based methods, while the IL policy π_θ^{base} remains frozen.

The environment resets if the robot reaches the destination, collides with an obstacle, or times out. Upon receiving the reset signal, history states within the world model are also cleared.

By building on a robust pre-trained base policy, the residual RL framework mitigates the sparse sampling challenge. This enables faster convergence to high-performance policies for each embodiment.

D. Step 3: Policy Distillation to Combine Specialists

After separately training residual RL specialists for each robot embodiment, we consolidate them into a single multi-embodiment policy. This "distilled" policy captures the collective knowledge of all specialist policies while using an embodiment embedding to generalize across different robot platforms.

1) Data Collection from Specialists: After the residual RL training, we record each specialist's input and output distributions, including:

- Policy state from the world model.
- One-hot embodiment identifier $\mathbf{e}$.

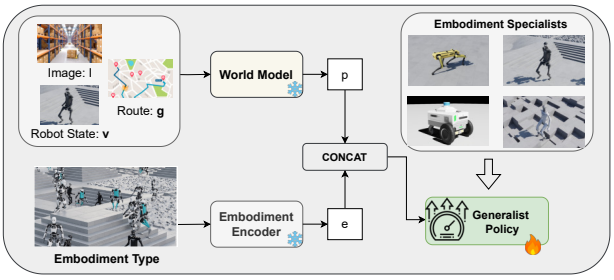

Fig. 3: Policy distillation aggregates multiple expert specialists (one per embodiment). The final multi-embodiment policy uses a one-hot or learned embedding to condition decisions on robot morphology.

- Mean and variance of the Gaussian action distribution used in PPO.

This logged dataset forms the basis for distillation.

2) Distillation Method: Let $\pi_\phi^{(i)}$ denote the specialist policy for the i-th embodiment. Each specialist produces a normal distribution $\mathcal{N}(\boldsymbol{\mu}^{(i)}(\mathbf{p}), \sigma^2)$ over actions. We define a distilled policy π_θ^{dist} that outputs $\boldsymbol{\mu}_\theta(\mathbf{p}, \mathbf{e})$ given $\mathbf{p}$ and an embodiment embedding $\mathbf{e}$. To match the specialists' distributions, we minimize the KL divergence:

$$\min_\theta \sum_{i \in \mathcal{E}} \sum_{d^i \in \mathcal{D}_i} \text{KL}\Big(\mathcal{N}(\boldsymbol{\mu}^{(i)}(\mathbf{p}), \sigma^2) \,\|\, \mathcal{N}(\boldsymbol{\mu}_\theta(\mathbf{p}, \mathbf{e}^{(i)}), \sigma_\theta^2)\Big),$$

where $d^i = (\mathbf{p}, \boldsymbol{\mu}^{(i)}, \sigma^2)$ and $\mathcal{D}^{(i)}$ is the dataset of recorded state-action distributions from the i-th specialist. $\mathbf{e}^{(i)}$ is the corresponding embodiment embedding.

3) Embodiment Embedding: A key component of policy distillation is the embodiment embedding $\mathbf{e}$, which captures the morphological and dynamical characteristics of each embodiment. In the simplest version, we use a one-hot encoding vector of length N, where N denotes the number of robot embodiments. Each position in this vector corresponds to a specific robot. This straightforward approach is efficient when N is small and the robots differ substantially. We anticipate that a learnable embedding could better generalize to new, unseen embodiments by interpolating within the embedding space, and we leave such zero-shot generalization for future work.

4) Distilled Policy Network Architecture: The distilled policy retains the same latent processing pipeline but additionally conditions on the embodiment embedding before generating the final action distribution (see Fig. 3). The network consists of an MLP for mean prediction and a global variance parameter, resulting in a single policy that achieves near-expert performance across all considered robot types.

Consequently, our three-step framework—imitation learning, residual RL, and policy distillation—bridges the gap between generic mobility knowledge and highly specialized embodiment constraints, yielding a unified cross-embodiment mobility policy.

III. Experimental Setup

A. Training

1) IL Base Policy: For the initial IL stage, we employ a X-Mobility checkpoint pre-trained on Carter dataset. We freeze this checkpoint, which then serves as the base network for subsequent RL refinements.

Fig. 4: Residual RL training environment. Multiple instances are tiled and run in parallel to accelerate data collection and model updates.

2) Residual RL: We utilize Nvidia Isaac Lab [17] to train our policies in a parallelized visual RL environment, enabling efficient data collection and rapid training updates.

To avoid overfitting and preserve the base policy's ability to generalize across environments, we construct a diverse set of training scenarios (Fig. 4) that accommodate four distinct robot embodiments: Nova Carter (wheeled), Unitree H1 (humanoid), Unitree G1 (humanoid), and Spot Mini (quadruped). For the humanoid and quadruped robots, we employ RL-based locomotion policies, trained within Isaac Lab, to map velocity commands to joint-level controls. Due to limited wheel-physics support in Isaac Lab, Nova Carter instead uses a custom controller that directly adjusts the robot's root state based on velocity commands.

Each embodiment is trained in a unified environment that randomly initializes the agent's pose and the goal location, where the goal distance is uniformly sampled between 2m and 5m from the robot's starting position. The straight line between the robot and the goal serves as a simplified route, providing short-horizon guidance within the camera's field of view. Each episode spans up to 256 timesteps and resets if the agent collides, reaches its goal, or exceeds the maximum episode length. We train each embodiment specialist for 1,000 episodes with 64 environments in parallel using 2 Nvidia L40 GPUs, except for Carter, which is trained for only 300 episodes to mitigate overfitting. This reduced training schedule for Carter is necessary because X-Mobility is already trained on Carter dataset, making it prone to overfitting if extensively fine-tuned.

3) Policy Distillation: To distill the learned specialists into a single unified policy, we record 320 trajectories per embodiment using the same environment as residual RL training. Each trajectory spans 128 steps, yielding approximately 40k frames per embodiment. We then perform policy distillation training on 4 Nvidia H100 GPUs by aligning each specialist's output distribution.

B. Benchmark

1) Metrics: We evaluate both efficiency and safety using two key metrics:

- Success Rate (SR): Fraction of trials that reach the goal region without collision or timeout.
- Weighted Travel Time (WTT): Total travel time to reach the goal, conditioned on success, divided by SR.

2) Scenarios: We evaluate our algorithm in four environments of increasing difficulty as in Fig. 5. In the simplest environment, only sparse low lying obstacles are present, while in the complex one, a warehouse with multiple racks requires long-horizon mobility under minimal route guidance. To further assess the policy's robustness and generalization, we also vary textures and lighting conditions across these environments.

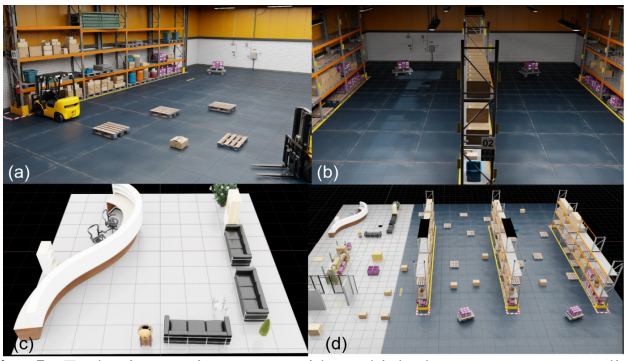

Fig. 5: Evaluation environments with multiple layouts to assess policy generalization: (a) warehouse with single rack ($24m \times 38m$), (b) warehouse with multi racks ($24m \times 38m$), (c) office ($10m \times 10m$), and (d) combined scenes with multi racks ($30m \times 38m$).

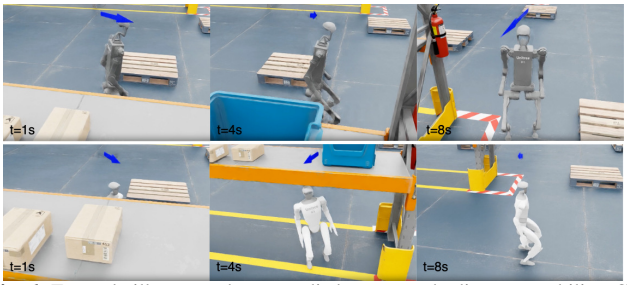

Fig. 6: Example illustrates the generalist's cross-embodiment capability: G1 shortcuts beneath the shelf, while H1 must detour due to insufficient height clearance.

For each environment, we conduct 640 trials with randomly sampled initial and goal poses. Each trial is terminated after 25.6s of execution. We then collect performance metrics for each embodiment and environment type, allowing direct comparison of mobility performance across different settings.

IV. Results

This section addresses four primary questions:

- Does residual RL effectively yield high-performance embodiment specialists from the IL baseline?
- Can policy distillation consolidate these specialists into a single robust generalist without a considerable performance drop?
- How does our approach compare to alternative strategies, such as training RL from scratch?
- Can COMPASS policy trained in simulation be applied to real robot platforms?

A. Does residual RL produce high-performance embodiment specialists?

To investigate whether residual RL can improve performance relative to the IL baseline (X-Mobility), we compared the specialist policies for each embodiment against X-Mobility, and the results are summarized in Table I. In all evaluations, the specialist policies significantly outperform X-Mobility across both success rate (SR) and mobility efficiency (WTT). The SR is improved with a factor ranging

TABLE I: Benchmark Results

Embodiment	Model	Warehouse Single Rack		Warehouse Multi Rack		Office		Combined	
		SR(%)	WTT(s)	SR(%)	WTT(s)	SR(%)	WTT(s)	SR (%)	WTT(s)
Carter	X-Mobility	42.3	8039	37.9	9351	50.6	5508	50.3	5319
	Carter Specialist	**91.5**	2282	91.8	**2283**	72.0	**2340**	**85.4**	**2442**
	Generalist	90.6	**2241**	**91.9**	2361	**73.4**	2504	85.3	2527
H1	X-Mobility	17.5	12588	9.2	12234	25.6	10791	22.0	11519
	H1 Specialist	**94.5**	**4123**	87.9	**4283**	**66.7**	3300	82.8	**3698**
	Generalist	93.4	4129	**88.9**	4319	66.2	**3260**	**84.4**	3802
Spot	X-Mobility	5.7	13320	4.5	12007	6.1	7740	11.7	11426
	Spot Specialist	84.5	**3183**	**93.5**	**3204**	**76.2**	3587	77.1	3490
	Generalist	**84.7**	3207	93.2	3255	74.8	**3512**	**77.9**	**3485**
G1	X-Mobility	3.6	13913	1.8	13781	2.8	13276	10.0	11865
	G1 Specialist	95.6	**4667**	93.7	**5254**	**77.0**	**3960**	90.0	**3968**
	Generalist	**95.7**	4717	**94.5**	5360	76.7	4031	**90.6**	3991

from 5x to 40x, and the WTT is improved 3x on average. This performance boost suggests that residual RL effectively leverages environment semantics and embodiment-specific physics. In particular, by exploring each robot's physical limits during training, the specialist can produce commands that maximize efficiency while maintaining balance and stability.

A notable observation is that while X-Mobility performs best on the Carter robot, its zero-shot performance on other embodiments degrade significantly. This aligns with expectations and underscores the need for embodiment-specific policy adaptations to ensure robustness. Compared to the original X-Mobility paper, we observe a performance drop for Carter in warehouse environments within Isaac Lab, primarily due to simplified routes and variations in image rendering quality. This motivates future improvements, such as incorporating a hierarchical mobility stack for more sophisticated routing and exposing the model to diverse rendering conditions during latent state training.

B. Can policy distillation produce a robust generalist?

We next evaluate whether policy distillation can merge the specialists into a single generalist policy without sacrificing overall performance. As shown in Table I, the generalist achieves performance comparable to, and sometimes exceeding, that of the specialist policies. This is especially evident on the G1 robot, likely due to increased diversity in training scenarios and embodiments during distillation. These results confirm that our distillation approach is effective and that building a robust generalist from multiple specialists is both feasible and efficient.

As a case study of cross-embodiment capabilities, we designed a long-horizon mobility scenario in a multi-rack warehouse. As illustrated in Fig. 6, G1 can lean slightly to the right and pass under a shelf with just enough head clearance, whereas H1 lacks sufficient clearance and must take a detour. This example highlights how the generalist effectively accounts for each embodiment's distinct morphological constraints.

C. How does COMPASS compare to training from scratch?

We also explored whether we could train a policy from scratch using the same RL setup with the latent state

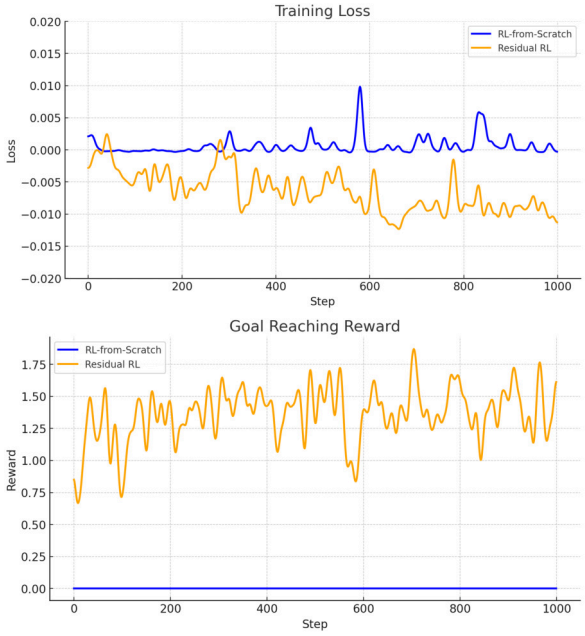

Fig. 7: RL from scratch versus Residual RL.

but without leveraging the IL base actions. We observed that this RL-from-scratch approach struggled to converge even after 1,000 episodes as in Fig. 7, indicating significantly lower training efficiency. In contrast, our residual RL method exhibited notably faster convergence, underscoring how leveraging an IL baseline resolves the sparsity issues that commonly plague RL in mobility tasks.

D. Can COMPASS Policy trained in simulation be applied to real robot platforms?

To evaluate the Sim2Real transferability, we deployed it on two real-world robotic platforms powered by the Nvidia Jetson Orin: Carter and Unitree G1 (Fig. 1). Leveraging the world model latent state representation, along with Isaac Lab's high-quality visual rendering and accurate physics simulation, both robots achieved robust zero-shot transfer, successfully navigating cluttered environments without any additional fine-tuning. Across 20 trials with randomly placed obstacles, the policy attained a success rate of approximately 80%, consistently avoiding collisions and reaching the des-

ignated targets. We anticipate that RL fine-tuning focused on semantic understanding of the environment, particularly when combined with a Real2Sim pipeline [18], can further narrow the Sim2Real gap.

Moreover, onboard inference with TensorRT requires only $\sim$30 ms per step, demonstrating that the policy is computationally efficient and well-suited for real-time deployment.

V. Ablation Studies

A. Residual RL

In this section, we present ablation studies evaluating the various design decisions in our residual RL approach. We exclude the Carter robot from these experiments as its cross-embodiment effect is less pronounced.

1) Curriculum on Goal Distance: In our RL training setup, we currently do not use a curriculum. To investigate whether such a strategy could improve training efficiency, we introduced a curriculum based on goal distance. Rather than enforcing a minimum goal distance from the outset, we gradually increased the probability of applying this constraint as training progressed to enable smoother convergence.

However, the results in Table II indicate that the curriculum-based approach did not outperform the baseline without a curriculum. A likely explanation is that the IL base policy already provides a strong initialization, and reducing exposure to challenging scenarios with the curriculum may have limited the policy's capacity to explore and succeed on more difficult tasks.

2) Critic State: In our default setup, the critic network uses the same policy state as the actor, under the assumption that the policy state already encapsulates sufficient information to learn accurate value functions. While this approach often works well, it depends on implicit information, which may make learning the critic network more challenging. To investigate potential improvements, we explored different configurations for the critic's input. In one variant, we provided a depth image explicitly to the value function by employing a ResNet18 for feature extraction, then concatenating the resulting features with velocity and route embeddings to form the critic state. As shown in Table II, this strategy enhanced performance in office scenarios—where tall obstacles exist—but inadvertently reduced performance in warehouse environments presented with more low-lying obstacles. These findings suggest that the additional signal from the depth image can boost performance under certain conditions, indicating a promising direction for further exploration.

3) Training Environment: In another ablation study, we evaluate the impact of the training environment. Our default setup uses a combined environment consisting of warehouse, office, and lab scenarios. To assess the effect of environmental diversity, we also conducted a separate training run using only the warehouse environment with H1 robot. The results, shown in Table II, indicate that restricting training to warehouse scenarios slightly enhances performance in warehouse scenes but reduces performance in the office and combined settings. These findings underscore the value of diverse training environments for improving generalization. Notably, even though the office setting was excluded from this specialized training, the model still performs reasonably well when tested there, demonstrating a degree of environment generalizability.

B. Policy Distillation

In our final ablation study, we want to examine the policy distillation process. The default setup trains the distilled policy to minimize KL divergence on an unfiltered dataset that may include failure cases. Our first goal is to assess whether capturing the full policy distribution is more beneficial than merely imitating the mean. To this end, we trained another model using only an MSE loss on the mean predictions. As shown in Table III, this approach led to slightly inferior performance, particularly on the H1 robot, suggesting that KL divergence—which preserves variance—provides more flexibility for managing training noise and architectural differences.

We also investigated the role of dataset quality by constructing a new dataset from the same scenarios but excluding all failure cases, hypothesizing that this might yield better overall performance. Contrary to our expectations, the results in Table III improved only marginally. We suspect that removing failure cases also eliminates corner cases that could otherwise offer valuable learning opportunities for the generalist policy.

VI. Future Work

Our findings suggest several key takeaways and directions for future work:

- Embodiment Encoding Strategy: We find that one-hot encoding is sufficient when only a few distinct platforms are considered. For a larger or more continuous spectrum of embodiments, a learned embedding might offer richer generalization [19].
- Distillation Trade-offs: Although policy distillation successfully merges expertise from specialized policies, it can also lead to an averaging effect across embodiments. This phenomenon contributes to embodiment-dependent performance variations.
- Transferability vs. Specialization: While residual RL can adapt rapidly to new embodiments, using the same set of hyperparameters can lead to performance divergence, as demonstrated in our studies. Consequently, specialized designs and tuned hyperparameters are often required to achieve optimal results.
- Hierarchical Mobility Stack: We observe a decreased success rate in environments such as offices and multi-rack warehouses, primarily for tasks requiring long-horizon route planning. This underscores the importance of a hierarchical mobility stack with a graph-based route planner to handle more complex mobility scenarios.
- VLA Training: The RL specialist policy can serve as a valuable resource for generating large-scale datasets to fine-tune foundational VLAs for object-oriented navigation, a task that typically requires massive amounts of training data.

References

[1] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine, "Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation," *arXiv preprint arXiv:2408.11812*, 2024.

TABLE II: Ablation Studies on Residual RL.

Embodiment	Model	Warehouse Single Rack		Warehouse Multi Rack		Office		Combined	
		SR(%)	WTT(s)	SR(%)	WTT(s)	SR(%)	WTT(s)	SR (%)	WTT(s)
H1	**Default Setting**	94.5	4123	87.9	4283	66.7	3300	**82.8**	3698
	w/ Curriculum	96.1	4045	88.7	4155	66.5	3148	83.9	3933
	Depth Critic	90.9	4245	87	4254	72.3	3530	85.4	3764
	Warehouse Only	94.6	3237	88.7	3249	64.0	2850	81.0	3152
Spot	**Default Setting**	84.5	3183	93.5	3204	76.2	3587	77.1	3490
	w/ Curriculum	81.5	3079	86.2	3128	66.1	3112	71.5	3238
	Depth Critic	83.4	3090	92.1	3125	77.9	3645	77.6	3226
G1	**Default Setting**	95.6	4667	**93.7**	5254	**77.0**	3960	**90.0**	3968
	w/ Curriculum	96.4	4376	86.5	4992	69.6	3478	89.0	4018
	Depth Critic	95.7	4600	87.9	5153	78.4	4091	91.4	4018

Green/Red: Better/Worse than default setting

TABLE III: Ablation study on policy distillation

Embodiment	Koss / Dataset	Warehouse Single Rack		Warehouse Multi Rack		Office		Combined	
		SR(%)	WTT(s)	SR(%)	WTT(s)	SR(%)	WTT(s)	SR (%)	WTT(s)
Carter	KL / All	90.6	2241	91.9	2361	73.4	2504	85.3	2527
	MSE / All	89.8	2201	92.1	2331	73.7	2585	85.1	2452
	KL / Good	93.1	2246	92.0	2388	72.3	2427	83.1	2744
H1	KL / All	93.4	4129	88.9	4319	66.2	3260	84.4	3802
	MSE / All	93.3	4082	87.6	4260	66.5	3251	82.1	3717
	KL / Good	93.6	4043	88.4	4218	66.1	3288	88.4	4218
Spot	KL / All	84.7	3207	93.2	3255	74.8	3512	77.9	3485
	MSE / All	84.3	3180	93.6	3218	75.0	3512	79.2	3516
	KL / Good	84.6	3234	94.3	3200	75.1	3462	78.4	3410
G1	KL / All	95.7	4717	94.5	5360	76.7	4031	90.6	3991
	MSE / All	96.8	4596	94.3	5336	77.6	4030	89.8	3934
	KL / Good	95.1	4622	93.4	5180	75.4	3884	90.0	3946

Green/Red: Better/Worse than baseline

[2] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song, "XSkill: Cross embodiment skill discovery," in *7th Annual Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=8L6pHd9aS6w

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

[4] S. Macenski, F. Martín, R. White, and J. Ginés Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. [Online]. Available: https://github.com/ros-planning/navigation2

[5] W. Liu, Z. Weng, Z. Chong, X. Shen, S. Pendleton, B. Qin, G. M. J. Fu, and M. H. Ang, "Autonomous vehicle planning system design under perception limitation in pedestrian environment," in *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. IEEE, 2015, pp. 159–166.

[6] W. Liu, H. Zhao, C. Li, J. Biswas, B. Okal, P. Goyal, Y. Chang, and S. Pouya, "X-mobility: End-to-end generalizable navigation via world modeling," *arXiv preprint arXiv:2410.17491*, 2024.

[7] Z. Xu, X. Xiao, G. Warnell, A. Nair, and P. Stone, "Machine learning methods for local motion planning: A study of end-to-end vs. parameter learning," in *2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2021, pp. 217–222.

[8] J. Chang, M. Uehara, D. Sreenivas, R. Kidambi, and W. Sun, "Mitigating covariate shift in imitation learning via offline data with partial coverage," *Advances in Neural Information Processing Systems*, vol. 34, pp. 965–979, 2021.

[9] X. Wang, Z. Zhu, G. Huang, X. Chen, J. Zhu, and J. Lu, "Drivedreamer: Towards real-world-driven world models for autonomous driving," *arXiv preprint arXiv:2309.09777*, 2023.

[10] A. Popov, A. Degirmenci, D. Wehr, S. Hegde, R. Oldja, A. Kamenev, B. Douillard, D. Nistér, U. Muller, R. Bhargava *et al.*, "Mitigating covariate shift in imitation learning for autonomous vehicles using latent space generative world models," *arXiv preprint arXiv:2409.16663*, 2024.

[11] T. Feng, W. Wang, and Y. Yang, "A survey of world models for autonomous driving," *arXiv preprint arXiv:2501.11260*, 2025.

[12] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[13] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6023–6029.

[14] L. Ankile, A. Simeonov, I. Shenfeld, M. Torne, and P. Agrawal, "From imitation to refinement–residual rl for precise visual assembly," *arXiv preprint arXiv:2407.16677*, 2024.

[15] C. Li, C. Tang, H. Nishimura, J. Mercat, M. Tomizuka, and W. Zhan, "Residual q-learning: Offline and online policy customization without value," *Advances in Neural Information Processing Systems*, vol. 36, pp. 61 857–61 869, 2023.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[17] M. Mittal, P. Roth, J. Tigue, A. Richard *et al.*, "Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning," 2025. [Online]. Available: https://arxiv.org/abs/2511.04831

[18] S. Zhu, L. Mou, D. Li, B. Ye, R. Huang, and H. Zhao, "Vr-robo: A real-to-sim-to-real framework for visual robot navigation and locomotion," *arXiv preprint arXiv:2502.01536*, 2025.

[19] J. Johannemann, V. Hadad, S. Athey, and S. Wager, "Sufficient representations for categorical variables," 2021. [Online]. Available: https://arxiv.org/abs/1908.09874