PnP-Nav: Plug-and-Play Policies for Generalizable Visual Navigation Across Robots

Anonymous Author(s)
Affiliation
Address
email

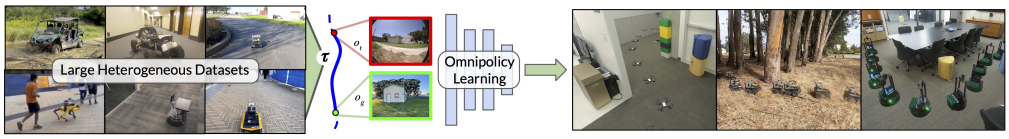


Figure 1: Plug-and-Play navigation policies to control any robot. By training on diverse, heterogeneous datasets, a single "omnipolicy" can control a variety of robots in challenging environments, including *new* robots, without any robot-specific data collection.

Abstract

Learning provides a powerful tool for vision-based navigation, but the capabilities of learning-based policies are constrained by limited training data. If we could combine data from all available sources, including multiple kinds of robots, we could train more powerful navigation models. In this paper, we study how goal-conditioned policies for vision-based navigation can be trained on data obtained from many distinct but structurally similar robots, and enable broad generalization across environments and embodiments. We analyze the necessary design decisions for effective data sharing across different robots, including the use of temporal context and standardized action spaces, and demonstrate that an *omnipolicy* trained from heterogeneous datasets outperforms policies trained on any single dataset. We curate 60 hours of navigation trajectories from 6 distinct robots, and deploy the trained omnipolicy on a range of new robots, including an underactuated quadrotor. We also find that training on diverse, multi-robot datasets leads to robustness against degradation in sensing and actuation. Using a pre-trained *base* navigational omnipolicy with broad generalization capabilities can bootstrap navigation applications on novel robots going forward, and we hope that PnP represents a step in that direction. For more information on the datasets, implementation, and videos, please check out our project page.

1 Introduction

Machine learning methods have enabled broad generalization with real-world applicability in natural language processing [1], visual perception [2–4], and other domains [5, 6] by leveraging Internet-scale data. Such generalization typically requires learning general patterns from diverse datasets, which are usually collected once and then reused for various purposes. Such large-scale models also support the ability to be adapted for new tasks by reusing the representations learned from broader, larger, and more general datasets, for example by or zero-shot transfer [7–9], or fine-tuning

Submitted to Workshops at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022), New Orleans, USA.

on target-domain data. Although this paradigm has been very successful, it is difficult to apply in robotics due to the sheer diversity of environments and platforms across researchers. Control policies learned end-to-end usually require separate data collection for each robotic platform, leading to "fragmentation" in progress — where every researcher works with their own robot-specific dataset and policies — making it infeasible to accumulate large enough datasets. Can we overcome this challenge by training policies on more general and reusable cross-robot datasets?

We study this question in the context of visual navigation, where heterogeneity between robots might include different camera hardware, viewpoints, dynamics, and more broadly, embodiments, but where the over-arching navigation objective looks similar irrespective of these differences. A wheeled robot, quadruped, or a drone all have the same abstract objectives: to explore the environment, plan a path to the goal, and avoid collisions. Leveraging this shared abstraction across robots and training a navigational *omnipolicy* from large-scale data could enable broad generalization to novel environments, unseen sensor parameters (e.g., camera intrinsics and extrinsics), and new robot configurations.

In this paper, we propose to take a step towards this kind of data sharing by training an embodiment-agnostic *omnipolicy* from an aggregated multi-robot navigation dataset. The primary contribution of our work is a framework for training a shared omnipolicy from multi-robot datasets, with empirical evidence that such an omnipolicy can effectively learn from heterogeneous datasets and generalize to novel robot platforms. To facilitate this, we aggregate a large heterogeneous dataset of navigation trajectories collected across 6 robots, spanning 60 hours of interactions in challenging indoor and outdoor environments. We train a shared navigation policy on this dataset and deploy it on 4 distinct robot platforms, including 2 new robots. We show that *a single* learned omnipolicy can be used across multiple robots to perform goal-reaching in challenging indoor and outdoor environments, and outperforms policies trained with any single dataset. We also report robustness to degradation in parameters like camera intrinsics, tire damage, and other gradual changes that the robot may experience over its lifetime.

We hope that this represents a step towards both general-purpose multi-robot datasets, and *visual navigational models* that can be deployed on a wide range of structurally similar robotic platforms — similar to how practitioners currently use pre-trained models in vision and language, such models could constitute pre-trained backbones for visual navigation.

2 Related Work

Learning from large, diverse robotic datasets has been studied for various robotic applications where data sharing across *similar* robots helps scale learning to challenging environments [10–12]. However, for applications such as ground or aerial navigation, with different sensors and robot dynamics, current approaches tend to rely on learning from small datasets which are only representative of a single robotic platform. Our paper proposes learning navigation behavior from heterogeneous robot datasets, collected across multiple embodiments.

Our work is closely related to transfer learning, where the objective is to train policies that transfer across domains, such as across dynamics [13–15], environments [16], morphologies [17–19], viewpoints [20], and embodiments [21]. Our focus is not on designing specific domain adaptation algorithms or hand-engineered augmentations [21] for transfer, but rather studying how direct generalization of simple, high-capacity models trained on real-world data can provide a path to broadly applicable navigational policies. Towards this, our work is also closely related to DroNet [22], which imitates expert on-road driving data to control a quadrotor. We take this paradigm one step further, showing that we can train *goal-conditioned* policies on data from *multiple* robots and control new ones, including a quadrotor.

Prior work has also explored learning of visual representations or end-to-end policies from passive data, such as YouTube videos, which can be scaled up massively without real-world data collection [23–25]. We explore a complementary direction, studying how readily available on-robot data

	Dataset	Platform	Speed	Amt.	Environment
1	GoStanford [26]	TurtleBot2	0.5m/s	14h	office
2	RECON [32]	Jackal	1m/s	25h	off-road
3	CoryHall [35]	RC Car	1.2m/s	2h	hallways
4	Berkeley [33]	Jackal	2m/s	4h	suburban
5	SCAND-S [36]	Spot	1.5m/s	8h	sidewalks
6	SCAND-J [36]	Jackal	2m/s	1h	sidewalks
7	Seattle [37]	Warthog	5m/s	1h	off-road
8	TartanDrive [38]	ATV	10m/s	5h	off-road
	PnP (Ours)			60h	

Table 1: The Plug-and-Play dataset contains 60 hours of navigation data in diverse environments across 6 different robots.

(also passive) can lead to generalizable policies. This is particularly relevant for navigation, where data is plentiful, and trajectories from multiple robots can directly train a policy, as opposed to two-stage methods that use Internet data for representation learning followed by in-domain adaptation.

Following a large body of research in visual navigation [26–30], we use a combination of topological graphs for high-level planning and image-goal policies for low-level control, which gives us an efficient way to scale reactive policies for long-range navigation [29, 31]. Prior work has also extended this framework for complex tasks beyond goal-reaching, such as exploration [28, 32, 33], instruction following [8], and reinforcement learning [34]. We show that that our omnipolicy can be coupled with such topological graphs to scale image-goal navigation to new robots.

3 The Plug-and-Play Dataset

Our aim is to train a general *visual navigation model* that can learn broadly applicable navigational affordances across a variety of distinct robotic systems. To facilitate such large-scale policy learning, we aggregated a heterogeneous dataset of navigation trajectories sourced from 8 datasets collected on robotic platforms with varying dynamics, sensors, and behaviors. The datasets contain a variety of challenging indoor and off-road environments (Table 1 and Fig. 1). We intend to publicly release this dataset on the project page.

The PnP dataset contains over 60 hours of real-world navigation trajectories: a combination of tele-operated and autonomous navigation behaviors collected across 6 distinct robotic platforms, including 4 commercially available platforms (TurtleBot, Clearpath Jackal, Warthog and Spot) and 2 custom platforms (Yamaha Viking ATV, RC Car). The trajectories contain widely varying robot dynamics and top speeds ranging between 0.2 and 10m/s, operating in a diverse set of environments (e.g., office buildings, hallways, suburban, off-road trails, university campus etc.).

To train navigation policies that can operate solely from egocentric visual observations, the dataset contains forward-facing RGB images paired with the robot's commanded actions and local odometry measurements. Each robot has different camera parameters, necessitating any successful policy to generalize across variations in camera pose and intrinsic parameters, though all platforms use the same type of sensor (monocular RGB camera). It is straightforward to further expand PnP by adding other datasets of relevant navigation behaviors [39, 40], or mix-and-match subsets of the dataset based on the desired application,

4 Training Multi-Robot Policies

To study a common navigation task across robots and environments, we consider the problem of image-goal navigation [41], where a robot is tasked with navigating to a goal location G specified as an image observation o_G taken at G. Unlike PointGoal [42], GPS navigation, or semantic objectives [43], image-goal navigation is a general framework that does not rely on ground truth localization or semantic labels, and allows us to formulate a very general navigation task that can be trained with any visual navigation dataset. Our goal is to train a goal-reaching policy $\pi(o_t, o_G)$ that

can navigate solely from egocentric visual observations. To provide a general task representation for this policy, we condition it on the desired goal o_G and integrate it into a navigational system based on topological graphs [26–29].

Such systems have shown great navigation results in a variety of indoor and outdoor environments — what would it take to train such a policy *across* robots, with varying controllers, dynamics and sensor placements? We highlight two key ingredients in training multi-robot policies: (i) carefully choosing the right action representation that facilitates transfer across robots, and (ii) conditioning the policies on a "summary" vector that allows it to deduce the properties of the robot it is controlling, so different robots can exhibit different, valid capabilities. Although we found the particular design decisions described in this section to be important for good performance, as we discuss in our experiments (Sec. 5.3), we emphasize that the primary contribution of our work is *not* a novel learning algorithm, but an empirical demonstration that policies learned from heterogeneous datasets can generalize broadly to new environments and new robots.

4.1 A Shared Abstraction Across Robots

While the general task of navigation from egocentric images is common across robots, the specific inputs (camera observations) and outputs (actions, dynamics) can vary substantially: a TurtleBot is differential-drive, expects low-level velocity commands, and has a top speed of 0.5m/s, whereas an ATV uses Ackermann steering, expects throttle and steering commands, and drives up to 20× faster. Learning a common control policy that operates directly on these raw, unstructured outputs can be challenging due to these inconsistencies and high-variance outputs (e.g., speed $\in [0.2, 10]$m/s). This is further exacerbated when generalizing to new robots, where the policy might need to "guess" how fast it should move.

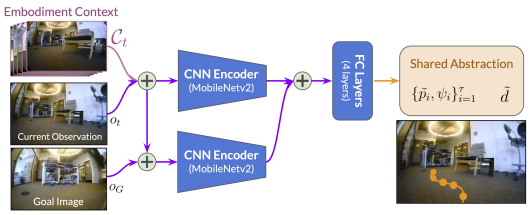


Figure 2: Omnipolicy architecture. We modify a typical goal-conditioned architecture (purple) by conditioning it on additional context from the target robot (pink) and making predictions in a shared, normalized action space (yellow).

To this end, we propose using a shared abstraction to allow the goal-reaching policies to operate in a *transformed* action space that is consistent across robots, making the data points look "similar" and easier to learn common patterns from. In our experiments, we found this to be important to be able to learn from multiple datasets (see Sec. 5.3.1 for analysis). We use a combination of *relative* waypoints $p(x, y)$ and yaw change ψ as a mid-level action space. Labels for these actions can be obtained by using local odometry, which are easily available across datasets. Additionally, the policy also predicts the temporal distance to the goal d, as a measure of traversability, which is used by the navigation system to estimate the connectivity of the topological graph.

While this gives a shared action space across robots, we found that the varying dynamics (e.g., different top speeds) across robots can make it challenging for learning algorithms to learn a joint policy. To alleviate this, we propose using a *normalized* action space $\{\tilde{p}(x, y), \psi\}$, where $\tilde{p} := \frac{1}{\alpha}p$ is scaled by a robot-specific factor α corresponding to the top speed of the robot. The temporal distance $\tilde{d}$ is also estimated in this normalized scale. Given this abstract action space, a robot-specific controller can be used to (i) *unnormalize* the waypoints, and (ii) *track* them (e.g., PID, MPPI) to extract low-level commands (e.g., velocities or motor commands).

4.2 Embodiment Context

When deployed on an arbitrary robot, the policy must infer the capabilities of that particular robot. For instance, a TurtleBot can spin in-place but not go over bumps on the road, whereas an RC Car can easily traverse small bumps but has a limited turning radius. A simple way to provide such awareness to the policy is to condition it on hand-designed parameters that provide a concise "summary" of

capabilities, such as its size, turning radius etc. Defining these parameters by hand presents a barrier to fast and easy deployment of the policy to new robots, and requires human intuition to identify and define a relevant set of parameters. Instead, we propose a simple and automatic approach: rather than manually defining parameters that fully identify the robot, we use a sequence of consecutive past observations from the robot's viewpoint to infer a learned *embodiment context* $\mathcal{C}_t$, and condition the learned policy on this context in addition to the observations. This context contains information about the robot's configuration and dynamics, which can be used to condition the behavior of the policy.

While this context may not contain *all* information to fully identify the robot, we hypothesize that it is sufficient to effectively control the robot. Our experiments show that the embodiment context allows the same policy to be deployed on novel robot configurations without designing any hand-engineered robot representation. We empirically evaluate different ways of providing context in Sec. 5.3.2 and find that the most effective representation is achieved by using a temporally consistent context $\mathcal{C}_t$ that conditions the policy on k consecutive past observations $\{o_{(t-k):(t-1)}\}$.

Figure 3: Depoying Plug-and-Play policies. We evaluate on 4 different robots in challenging indoor and outdoor environments.

4.3 Implementation Details

A combination of conditioning the policies on embodiment context and transforming the action space can allow a simple goal-reaching policy to be trained from heterogeneous datasets. It is important to note that the proposed modifications are orthogonal to the choice of downstream policy architecture and learning algorithm, and we could use different encoders or train with reinforcement learning.

Architecture: We use a goal-conditioned policy architecture that takes as input the current observation o_t and goal observation o_G, and predicts *normalized* waypoints and distances. Additionally, we condition on temporal context $\mathcal{C}_t$, which is constructed by stacking the past $k = 5$ consecutive observations. Visual inputs to the network are provided as 160×120 RGB images for all observations. Following prior work [32, 44], we train context-conditioned representations by using separate MobileNetv2 encoders for (i) the current observation $\{o_t, \mathcal{C}_t\}$, and (ii) conditional goal observation, as shown in Fig. 2. The two embeddings are concatenated and passed through three fully-connected layers to two prediction heads: normalized temporal distance $\tilde{d}_t$ and a sequence of $\tau = 5$ normalized future waypoints $\{\tilde{p}_i, \psi_i\}_{i=1}^{\tau}$.

Training: Following the procedure of Shah et. al. [27], we use a combination of image-goal pairs sampled from the same trajectory in the dataset as "positives", and "negatives" sampled from *different* trajectories, to obtain training data pairs. The distance head is trained on both positives and negatives, whereas the action head is only trained on positives. We train the two heads jointly with supervised learning using an ℓ_2 regression loss. We use multi-GPU training with batch sizes between 400–1200 and perform gradient updates using the Adam optimizer [45] with a learning rate of 5×10^{-4}.

Deployment: We combine this goal-reaching policy with a topological map $\mathcal{M}$, where nodes are represented by the robot's observations (augmented with the embodiment context), and edges are computed using the temporal distance estimates d from the trained policy, following the setup of ViNG [27]. At every time step, the robot associates its current and goal observations in $\mathcal{M}$, i.e., finds the node with smallest temporal distance to it, and computes the optimal sequence of subgoals $\{s_i\}$ using Dijkstra's algorithm. The policy π is queried with the current observation $\{o_t, \mathcal{C}_t\}$ and

immediate subgoal s_1 to obtain a sequence of waypoints $\{\tilde{p}_i, \psi_i\}_{i=1}^{\tau}$, which are tracked by a robot-specific low-level controller.

5 Deploying the Omnipolicy Across Robots

We deploy our learned omnipolicy in a variety of challenging indoor and outdoor environments on four different robot platforms. We designed out experiments to answer the following questions:

- **Q1.** Can multi-robot training enable generalization to *novel* robots and environments?
- **Q2.** Do policies trained with PnP outperform policies trained solely on single-domain data?
- **Q3.** How important are the design choices made in Sec. 4 for attaining good performance with the omnipolicy?
- **Q4.** Are multi-dataset policies more robust to degradation than single-domain policies?

5.1 Meet the Robots

We deploy the omnipolicy on four distinct robotic platforms, including a quadrotor and two other *novel* robots with no corresponding training data, as shown in Fig 3.

Vizbot: A custom-built robot platform inspired by the design of Niwa et. al. [46], based on a Roomba. It is equipped with an off-the-shelf PCB-mounted fisheye camera. *There is no data from a Vizbot, or any other Roomba-like robot, in PnP.*

DJI Tello: A commercially available quadrotor equipped with a forward-facing camera. *There is no training data from any quadrotor in PnP.* We restrict the drone to a horizontal plane 1m off the ground, to mimic ground navigation.

Clearpath Jackal UGV: A commercially available off-road platform equipped with an off-the-shelf PCB-mounted fisheye camera. *This system resembles the data collection platform used for the RECON, Berkeley, and SCAND-J datasets*, but has a different camera and mounting height.

LoCoBot: A popular open-source platform based on a Kobuki, equipped with an off-the-shelf PCB-mounted fisheye camera. *There is no data from a LoCoBot in PnP*, although GS was collected on a similar TurtleBot2, albeit with a different spherical camera at a lower height.

5.2 Zero-Shot Deployment

Towards answering **Q1**, we deploy the *same* trained omnipolicy on four distinct robotic platforms *without* any fine-tuning per robot. Fig. 3 and Table 2 summarize our evaluation in a variety of indoor and outdoor environments on 4 different robots, all using the same omnipolicy. Most notably, the omnipolicy can control a Tello, despite never having seen any trajectories from aerial robots in PnP. A policy trained on PnP consistently outperforms single robot policies across all tested robots, performing up to 5x better in some cases. We also observe generalization to massively out-of-distribution (OOD) settings, like a LoCoBot navigating outdoors on a sidewalk, or a Jackal navigating inside an office building, which were not present in the training data. This suggests that training on heterogeneous datasets can enable generalization to novel environment-robot pairs, as well as entirely new robots.

Dataset(s)	LoCoBot	Tello	Vizbot	Jackal
GS	0.26	0.21	0.51	0.31
RECON	0.62	0.79	0.26	0.68
PnP (Ours)	**0.96**	**0.99**	**0.93**	**0.94**

Table 2: Summary of navigation across robots. A single policy trained on PnP-Mid outperforms the best single-robot policy for *each* robot used in our experiments, mean success rate reported.

To better understand how data sharing benefits performance (**Q2**), we quantitatively evaluate the navigation performance of policies trained with heterogeneous datasets in an assortment of 20 indoor and outdoor environments on the Jackal and LoCoBot platforms (Tables 5, 3). To project the

performance trends with varying amounts of data, we train policies from increasingly diverse subsets of PnP— "Small", "Mid", and "Large", corresponding to data from the first 2, 4, and 6 datasets listed in Table 1. We quantify performance in terms of success rates, measured as the mean progress made towards the goal. For videos of our experiments and more information on the testing environments, please check out the supplementary video and project page.

Deploying on a LoCoBot, which is an *unseen robot* with no corresponding data present in the dataset, we find that policies trained on a single dataset (e.g., GoStanford (GS) [26] or CoryHall [35]) fail to generalize to a new embodiment with different sensors. Fine-tuning visual representations trained for task-agnostic datasets like ImageNet, which is a popular strategy for pre-training in many vision-based applications [47, 48], improves a bit but still struggles in a majority of the environments. However, policies trained by sharing task-relevant datasets across robots significantly outperform these single-domain policies, as shown in Table 5. We also observe that adding more and diverse datasets (PnP-Large) contributes towards improvements in performance, despite the additional data coming from seemingly unrelated tasks (e.g., off-road driving). Fig. 4 shows an example office environment where increasing the diversity of training data improves performance.

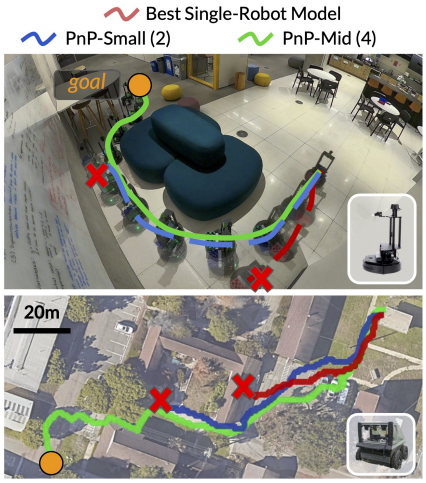


Figure 4: Qualitative comparison. Policies trained with increasingly diverse data lead to better generalization to a LoCoBot (top) and Jackal (bottom). Both robots were controlled by the *same* policy.

We observe similar trends on a Jackal, which is deployed on a variety of *previously unseen* outdoor and indoor environments (Table 3). Unsurprisingly, a single-domain policy trained on off-road RECON data [32] performs well for many outdoor environments, but struggles with navigating indoors, which is OOD for the RECON dataset. Similarly, a GS policy struggles in outdoor environments but succeeds in some easy indoor environments. Omnipolicies trained on subsets of PnP are able to generalize better to a variety of indoor and "Hard" outdoor environments, which can be over 100m long, significantly outperforming the single-domain policies (Fig. 4).

Dataset(s)	#	Indoor		Outdoor
		Easy	Moderate	
CoryHall	1	0.22	0.13	0.29
GS	1	0.25	0.16	0.44
–"– +ImageNet	1	0.35	0.35	0.57
PnP-Small	2	0.82	0.59	**1.0**
PnP-Mid	4	**1.0**	**0.97**	0.83
PnP-Large	6	**1.0**	**1.0**	0.83

Figure 5: Navigation success rates on a LoCoBot. Policies trained with PnP (green) result in increasingly capable navigation, in both indoor and outdoor enviroments, on an *unseen* robot.

5.3 A Systematic Analysis of the Design Space

Towards answering **Q3**, we perform a systematic analysis of the design choices presented in Sec. 4. We evaluate each design choice on a LoCoBot, which is an *unseen robot* with no corresponding training data, in indoor environments with varying levels of complexity, where "Easy" environments have wide passages and smooth turns, "Moderate" environments have tight passages or sharp turns, and "Hard" environments are larger (up to 50m) with a combination of tight passages and multiple turns.

5.3.1 Shared Action Space

Action Space	Easy	Moderate
Velocities	0.73	0.54
Waypoints	0.42	0.26
Norm. Waypt.	**1.0**	**0.95**

Architecture	Easy	Moderate
Stacked [28]	0.52	0.72
Siamese [26, 27, 49]	0.73	0.26
Conditioned [32, 44]	**1.0**	**0.95**

Context	Easy	Moderate	Hard
None	**1.0**	0.79	0.36
Static	**1.0**	0.86	0.5
Temporal	**1.0**	**0.92**	**0.7**

Table 4: A systematic analysis of the design choices in Sec. 5.3 reveals that choosing the right action representation (left), goal-conditioned architecture (center), and conditioning on embodiment context (right) are really important to facilitate multi-robot learning.

We compare the three action spaces discussed in Sec. 4.1 by training three different policies on PnP-Mid and evaluating them in 10 environments (Table 4). While using velocities as an action space works well for most easy environments, often outperforming the policy using waypoints, both these policies struggle in environments requiring dynamic maneuvers like sharp turns. A policy based on normalized waypoints, on the other hand, significantly outperforms the others, including in the challenging environments. This suggests that normalizing the action space indeed allows the policies to learn more effectively and generalize to new robots.

Dataset(s)	#	Outdoor		Indoor
		Easy	Hard	
GS	1	0.25	0.05	0.40
RECON	1	0.67	0.48	0.36
–"– +ImageNet	1	0.72	0.52	0.31
PnP-Small	2	0.75	0.52	0.42
PnP-Mid	4	**1.0**	**1.0**	**0.82**
PnP-Large	6	**1.0**	**1.0**	**0.88**

Table 3: Navigation success rates on a Jackal. By leveraging heterogeneous datasets, omnipolicies trained on PnP (green) can drive a Jackal *better* than a policy trained on a Jackal-specific dataset (RECON), also generalizing to novel indoor environments.

5.3.2 Embodiment Context

We consider two ways to represent the embodiment context: (i) temporally consistent context containing k consecutive past observations $\{o_{(t-k):(t-1)}\}$, and (ii) static context, containing a *fixed* set of k past observations from the robot in the target environment. Comparing these choices in environments of varying complexities (Table 4), we find that adding either form of context significantly boosts the navigation performance in the harder environments, which require the robot to navigate tight passages with multiple obstacles and sharp turns. This suggests that the context helps the polices generalize better due to the additional information about the embodiment (e.g., viewpoint, speed etc.). Between the two, we found the temporal variant superior, suggesting that the temporal information (e.g., speed, turning radius etc.) is important to enable this generalization. In our main experiments discussed in Sec. 5.2 and Fig. 3, we use a temporally consistent context with $k = 5$.

5.3.3 Policy Architecture

We also compared different policy architectures to encode the goal information: (i) single-encoder stacking, where the observation and goal images are stacked along the channel dimension [28], (ii) a Siamese architecture, where the images are processed with independent encoders and the resulting embeddings are combined [26, 27, 49], and (iii) the conditional architecture in Fig. 2, with an additional pathway from the observation to the policy outputs [32, 44]. We found that the choice of architecture significantly affects the navigation performance, with the conditional model being the most performant. We hypothesize that this is due to the additional pathway that allows the learned embeddings to be conditioned on the current observations, leading to more generalizable representations, as studied in prior work [32].

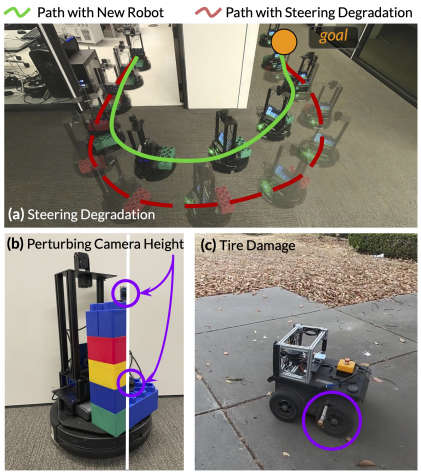
5.4 Robustness to Degradation

A key strength of training on heterogeneous datasets is that learning across varied parameters encourages the policy to learn shared affordances across robots, thus being robust to small variation in robot parameters, such as sensor placement and mechanical properties. We show that the learned omnipolicy can indeed offer such robustness by testing it under some example degradation scenarios shown in Fig. 6.

When testing the trained policy with a steering degradation (Fig. 6a), where the robot's maximum angular velocity is clipped, we find that the omnipolicy can compensate for the degradation by taking a longer, smoother path towards the goal without any localization failures. We also tested the omnipolicy while perturbing the position of the camera and physically affecting the dynamics by damaging the robot *during navigation*, and find that it can successfully reach the goals despite the degradation (Fig. 6d). Please see the supplemental video for these experiments.

6 Discussion

In this paper, we demonstrated that a simple goal-conditioned navigation policy trained from navigation datasets collected by multiple distinct robots, ranging from RC cars to large ATVs, can control *new* ground robots in challenging environments. The design of our learning framework is simple, and largely follows prior work: the novel observation is that a set of relatively simple decisions, such as including a temporal context and standardizing the action space, is sufficient to enable broad generalization from heterogeneous data. Empirically, we show that our approach can enable real-world navigation for a range of robots, including some not seen in training, and even an underactuated quadrotor.



(d)	Steering	Viewpoint	Physical
Single	0.30	0.17	0.81
PnP (Ours)	**0.89**	**0.81**	**1.0**

Figure 6: Policies trained with PnP are more robust to degradation in parameters such as (a) actuation, (b) perturbed sensor viewpoint, and (c) physical damage, than single-domain policies (d).

Our specific instantiation of this principle does have some limitations. Most prominently, our system does not explicitly account for differences in capabilities: we assume all robots are ground robots (though we study generalization to a quadrotor) with a forward-facing RGB camera. Handling diverse sensing, actuation (beyond variability in speed and steering), and traversability, would be an exciting direction for future work. Secondly, our dataset could be much larger: while we observe exciting generalization from 60 hours of data, a much larger and broader dataset could enable even better generalization in the future.

The promise of such *omnipolicies* trained on diverse data is that they might provide a pre-trained *base* model for a variety of downstream navigation applications. In the same way that computer vision researchers and practitioners typically start off by downloading a pre-trained backbone to use for their task, we hope that future navigation projects might use a pre-trained navigational omnipolicy that generalizes broadly enough to offer a "universal" starting point.

References

[1] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018. 1

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1

[3] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[4] K. Grauman *et al.*, "Ego4d: Around the world in 3,000 hours of egocentric video," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[5] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," 2022. 1

[6] M. Chen *et al.*, "Evaluating large language models trained on code," *arXiv*, 2021. 1

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *International Conference on Machine Learning (ICML)*, 2020. 1

[8] D. Shah, B. Osiński, B. Ichter, and S. Levine, "Robotic navigation with large pre-trained models of language, vision, and action," in *Conference on Robot Learning (CoRL*, 2022. 3

[9] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Finetuned language models are zero-shot learners," in *International Conference on Learning Representations (ICLR)*, 2022. 1

[10] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, "Learning modular neural network policies for multi-task and multi-robot transfer," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2169–2176. 2

[11] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," in *Conference on Robot Learning (CoRL)*, 2020.

[12] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[13] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," in *Robotics: Science and Systems*, 2017. 2

[14] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[15] G. Feng, H. Zhang, Z. Li, X. B. Peng, B. Basireddy, L. Yue, Z. Song, L. Yang, Y. Liu, K. Sreenath, and S. Levine, "GenLoco: Generalized Locomotion Controllers for Quadrupedal Robots," in *Conference on Robot Learning (CoRL)*, 2022. 2

[16] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," in *Robotics: Science and Systems*, 2021. 2

[17] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2017. 2

[18] W. Huang, I. Mordatch, and D. Pathak, "One policy to control them all: Shared modular policies for agent-agnostic control," in *International Conference on Machine Learning (ICML)*, 2020.

[19] K. Kang, G. Kahn, and S. Levine, "Hierarchically integrated models: Learning to navigate from heterogeneous robots," in *Conference on Robot Learning (CoRL)*, 2021. 2

[20] F. Sadeghi, A. Toshev, E. Jang, and S. Levine, "Sim2real viewpoint invariant visual servoing by recurrent control," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[21] N. Hirose, D. Shah, A. Sridhar, and S. Levine. ExAug: Robot-Conditioned Navigation Policies via Geometric Experience Augmentation. [Online]. Available: http://sites.google.com/view/exaug/ 2

[22] A. Loquercio, A. I. Maqueda, C. R. del Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, 2018. 2

[23] M. Chang, A. Gupta, and S. Gupta, "Semantic visual navigation by watching youtube videos," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[24] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," in *Conference on Robot Learning (CoRL)*, 2022.

[25] M. Hahn, D. S. Chaplot, S. Tulsiani, M. Mukadam, J. M. Rehg, and A. Gupta, "No RL, no simulation: Learning to navigate without navigating," in *Advances in Neural Information Processing Systems (NeurIPS*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021. 2

[26] N. Hirose, F. Xia, R. Martín-Martín, A. Sadeghian, and S. Savarese, "Deep visual MPC-policy learning for navigation," *IEEE Robotics and Automation Letters*, 2019. 3, 4, 7, 8

[27] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, "ViNG: Learning Open-World Navigation with Visual Goals," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021. 5, 8

[28] D. Singh Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3, 8

[29] X. Meng, N. Ratliff, Y. Xiang, and D. Fox, "Scaling Local Control to Large-Scale Topological Navigation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 3, 4

[30] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-Parametric Topological Memory for Navigation," in *International Conference on Learning Representations*, 2018. 3

[31] B. Eysenbach, R. R. Salakhutdinov, and S. Levine, "Search on the Replay Buffer: Bridging Planning and RL," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[32] D. Shah, B. Eysenbach, N. Rhinehart, and S. Levine, "Rapid exploration for open-world navigation with latent goal models," in *Conference on Robot Learning (CoRL)*, 2021. 3, 5, 7, 8

[33] D. Shah and S. Levine, "ViKiNG: Vision-Based Kilometer-Scale Navigation with Geographic Hints," in *Robotics: Science and Systems (RSS)*, 2022. 3

[34] D. Shah, A. Bhorkar, H. Leen, I. Kostrikov, N. Rhinehart, and S. Levine, "Offline reinforcement learning for customizable visual navigation," in *Conference on Robot Learning (CoRL)*, 2022. 3

[35] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-Supervised Deep RL with Generalized Computation Graphs for Robot Navigation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018. 3, 7

[36] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone, "Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation," *IEEE Robotics and Automation Letters*, 2022. 3

[37] A. Shaban, X. Meng, J. Lee, B. Boots, and D. Fox, "Semantic terrain classification for off-road autonomous driving," in *Conference on Robot Learning (CoRL)*, 2021. 3

[38] S. Triest, M. Sivaprakasam, S. J. Wang, W. Wang, A. M. Johnson, and S. Scherer, "Tartandrive: A large-scale dataset for learning off-road dynamics models," 2022. 3

[39] J. Biswas and M. M. Veloso, "Localization and navigation of the cobots over long-term deployments," *The International Journal of Robotics Research*, 2013. 3

[40] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, 2016. 3

[41] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 3

[42] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir, "On Evaluation of Embodied Navigation Agents," 2018. 3

[43] S. Wani, S. Patel, U. Jain, A. Chang, and M. Savva, "Multion: Benchmarking semantic map memory using multi-object navigation," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[44] A. Nair, S. Bahl, A. Khazatsky, V. Pong, G. Berseth, and S. Levine, "Contextual imagined goals for self-supervised robotic learning," in *Conference on Robot Learning (CoRL)*, 2019. 5, 8

[45] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations (ICLR)*, 2015. 5

[46] T. Niwa, S. Taguchi, and N. Hirose, "Spatio-temporal graph localization networks for image-based navigation," 2022. 6

[47] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 2014. 7

[48] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 7

[49] K. Yadav, R. Ramrakhya, A. Majumdar, V.-P. Berges, S. Kuhar, D. Batra, A. Baevski, and O. Maksymets, "Offline visual representation learning for embodied navigation," 2022. 8