Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

Pushing the Limits of Cross-Embodiment Learning for Manipulation and Navigation

Jonathan Yang*, Catherine Glossop†, Arjun Bhorkar†, Dhruv Shah†, Quan Vuong‡,
Chelsea Finn*, Dorsa Sadigh*, Sergey Levine†
* Stanford University
† University of California, Berkeley
‡ Google Deepmind

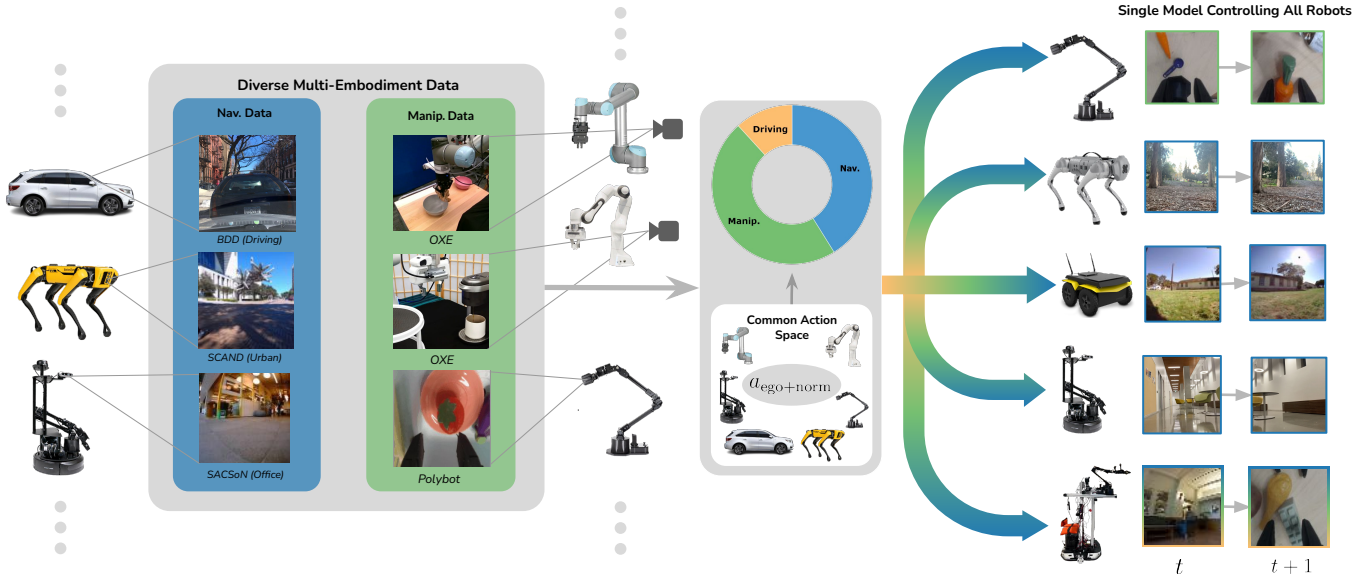

Fig. 1: Heterogeneous cross-embodiment learning. We test the limits of cross-embodiment learning by training a *single goal-conditioned policy* across 18 manipulation, navigation, and driving datasets. Our policy can control a variety of manipulators, wheeled, and legged robots, as well as novel embodiments such as drones and mobile manipulators, in challenging real-world environments.

***Abstract*—Recent years in robotics and imitation learning have shown remarkable progress in training large-scale foundation models by leveraging data across a multitude of embodiments. The success of such policies might lead us to wonder: just how diverse can the robots in the training set be while still facilitating positive transfer? In this work, we study this question in the context of heterogeneous embodiments, examining how even seemingly very different domains, such as robotic navigation and manipulation, can provide benefits when included in the training data for the same model. We train a single goal-conditioned policy that is capable of controlling robotic arms, quadcopters, quadrupeds, and mobile bases. We then investigate the extent to which transfer can occur across navigation and manipulation on these embodiments by framing them as a single goal-reaching task. We find that co-training with navigation data can enhance robustness and performance in goal-conditioned manipulation with a wrist-mounted camera. We then deploy our policy trained only from navigation-only and static manipulation-only data on a mobile manipulator, showing that it can control a novel embodiment in a zero-shot manner. These results provide evidence that large-scale robotic policies can benefit from data collected across various embodiments. Further information and robot videos can be found on our project website.**[1]

[1]https://extreme-cross-embodiment.github.io

I. Introduction

The advent of large-scale foundation models in machine learning has enabled harnessing diverse datasets to enhance sample efficiency, improve generalization, and facilitate transfer to novel domains [1]. Recent years in robotics have seen an acceleration in the collection and consolidation of large-scale datasets in the hopes of obtaining similar benefits. These datasets have contained demonstrations spanning many scenes, observations, viewpoints, tasks, and embodiments in a wide range of robotics domains such as manipulation [2–4], navigation [5, 6], autonomous driving [7], and others [8]. However, these prior works typically restrict their investigations to sets of similar embodiments – e.g., arms with parallel jaw grippers. In contrast, the most successful large-scale foundation models are typically trained on highly heterogeneous data, such as large text corpora mined from the web. This raises the question: what degree of embodiment diversity can we include when training broadly capable "generalist" robot policies?

We study this problem in the context of heterogeneous embodiments, aiming to understand whether large-scale policies can benefit from data across navigation and manipulation.

Enabling this transfer of knowledge can eliminate the need to recollect datasets containing information present in one domain but not the other. For example, navigation data can help manipulators understand spatial relationships between different poses. Similarly, manipulation data can help navigators with object-centric reasoning. This is particularly crucial for mobile manipulators, which need both the mobile base and the robotic arm to approach certain objects.

Why might we hope for positive transfer across navigation and manipulation? While these domains seemingly differ significantly in terms of hardware, observations, and action representations, they contain many similar sensorimotor principles. For example, both domains require the learned robot policy to have an understanding of collisions and geometry. Both domains also require the agent to perform some form of visual servoing. In manipulation, the robot analyzes its observation to determine the position of its end-effector with respect to a target object, and then moves towards it. Similarly, in visual navigation, the robot examines the spatial relationship between its current location and goal, as inferred from image observations, and determines how to move toward the goal. If the manipulation task uses a wrist-mounted camera and the navigation task uses a forward-facing camera, both embodiments have the same equivariance between pose changes and camera observations – i.e., moving "left" with respect to the image will transform the observations in egocentric manipulation and navigation in a similar manner.

In this paper, we empirically investigate the benefits of including navigation data for robotic manipulation, and vice versa. We present, to our knowledge, the first results demonstrating a large-scale policy trained jointly on navigation and manipulation data from many different robots, showing that such a policy can control robotic arms, drones, quadrupeds, mobile bases, and mobile manipulators. We then demonstrate that a co-trained policy can achieve a 20% higher success rate over a manipulation-only policy. Interestingly, the same co-trained policy achieves a $5-7\%$ improvement over a navigation-only policy on 4 different robots. This suggests that robotic agents can benefit from data collected across significantly different embodiments. We then characterize which datasets are most useful for manipulation and demonstrate that navigation data helps the policy learn embeddings that are more informative of distance to the goal in novel manipulation environments. We finally show that our policy can generalize to two new robots: a mobile manipulator and a quadrotor, without any data specific to these embodiments. While the particular training methodology and model architecture are based on prior techniques, the empirical findings are a novel contribution of our work, demonstrating for the first time that navigation data can provide quantifiable benefits for robotic manipulation in the cross-embodied policy learning setting.

II. Related Work

Traditionally, robotic learning has involved training policies using datasets specifically gathered for each robot and its designated task. However, the substantial cost of data collection and the ensuing lack of diversity in these datasets have resulted in policies with notably limited generalization ability. To alleviate this issue, several prior works have investigated cross-embodiment transfer at a small scale to enable the reuse of robotic datasets. In this section, we will first describe the body of work focused on cross-embodiment transfer at a small scale. We will then discuss several efforts in collecting large robotics datasets and training policies in these settings.

Cross-embodiment transfer. Prior works on cross-embodiment transfer have typically focused on transferring to novel robot parametrizations in simulation [9–11], novel morphologies in the real world [12–14], and via sim-to-real transfer [15–18]. By conditioning policies on embodiment [9, 10, 13, 19–24], unifying action abstractions [5, 25–31], and using domain adaptation [14, 18, 32–38], these works showed that robotic policies can generalize to new tasks and environments on different embodiments. Due to the large diversity of robot hardware, enabling the capability to learn from one robot hardware to control another is crucial to leveraging the robot data that is currently available. Our work specifically focuses on the problem of transferring knowledge between real-world robotic policies across a heterogeneous set of embodiments, or embodiments spanning navigation and manipulation with large amounts of variety in hardware.

Large-scale robotic datasets and policies. While these smaller-scale projects have demonstrated great success in facilitating multi-robot transfer, it has become clear that training policies that apply to a large variety of embodiments and domains would require learning from large and diverse datasets. To address this issue, researchers have created real-world robotic datasets for manipulation [2, 3, 39–44], navigation [5, 45, 46], and autonomous driving [47–52]. The increased availability of open-source robotic datasets has enabled training large-scale robotic foundation models that leverage data across many embodiments [4, 5, 53–56]. These foundation models include object tracking models [57–59], representation learning models [60, 61], and predictive world models [2, 62–64].

A number of robotic foundation models referred to in recent work as Generalist Robot Policies (GRPs) have been trained on large, diverse datasets to directly predict low-level robot actions from image observations [4, 5, 31, 55, 65, 66]. These foundation models have typically been co-trained [4] or pre-trained [53, 55] with data across multiple embodiments. GRPs have demonstrated the ability to fit a broad range of embodiments and significantly enhance their performance in new tasks and environments [4, 55]. While these models have previously been trained on datasets exclusively consisting of manipulation, navigation, or driving data, we propose to train a single model that can benefit from robotic data across these domains. We then investigate the extent to which this generalization can occur across these significantly different embodiments.

III. Preliminaries

We will study heterogeneous cross-embodiment robotic learning in the context of goal-conditioned imitation learning and training policies to reach visually indicated goals from data. Let us define datasets as $D_e := \{\tau_1, \tau_2, \ldots, \tau_k\}$ consisting of k demonstrations for embodiment e. Each trajectory $\tau \in D_{e_m}$ consists of a sequence of observations (images) and actions. That is, $\tau := \{o_0, a_0, o_1, a_1, \ldots\}$. The objective of goal-conditioned imitation learning is to train a policy $\pi(a|o, o_g)$ to output actions that control a particular embodiment given the current and goal observations.

Goal-conditioned manipulation. In goal-conditioned manipulation, the policy must learn to output a sequence of actions that are converted to joint velocities and given to a lower-level controller. Manipulation datasets D_{e_m} (e_m referring to manipulation for the embodiment) typically consist of teleoperated demonstrations from a remote controller, VR headset, or a haptic device. These different modalities can lead to demonstrations that contain many different choices for actions such as absolute and delta Cartesian control, absolute and delta joint control, or operational space control. Even with a similar controller, differences in coordinate frame and gains can cause discrepancies in action interpretation between robots.

Visual navigation. The objective of visual robotic navigation is to direct a robotic agent to move to a goal $g \in G$ while avoiding obstacles. The robot is not given any ground-truth localization information, GPS readings, or semantic maps requiring it to output a series of waypoints or velocities given only its observation history and goal image. In addition, the agent predicts a distance function $d(\cdot|o_{t-k:t}, o_g)$ to determine the distance between its current observation and its goal. During evaluation time, the robot is given a topological map $\mathcal{M}$, which is a sequence of image subgoals. The agent must first determine a feasible subgoal $o_g \in \mathcal{M}$ and then determine how to move to this subgoal using a local policy. The subgoal is determined by querying the distance function on all of the goal images in the topological map, then determining the closest image to the robot.

IV. Heterogeneous Cross-Embodiment Learning

In this work, we study cross-embodiment robotic learning with embodiments that include navigation platforms and robotic arms. We refer to this as heterogeneous cross-embodiment, to distinguish it from earlier works that studied cross-embodiment with data from similar robots and near-identical action spaces [2, 12]. Given manipulation datasets $D_{e_{m,\cdot}}$ and navigation datasets $D_{e_{n,\cdot}}$, we would like to learn a *single* policy $D_{e_{m,1}} \cup D_{e_{m,2}} \cup \ldots \cup D_{e_{n,1}} \cup D_{e_{n,2}}$ that can control robots in both domains. To solve this problem, we train a goal-conditioned policy $\pi(a|o, o_g)$ that outputs k actions into the future given a context of c observations. While we could simply train a single policy across all of the navigation and manipulation datasets to output action labels that match each specific dataset (using padding or sequence models in case dimensionalities do not match), we propose a unified action and observation representation that is specifically designed for our cross-embodied training setting in the following sections.

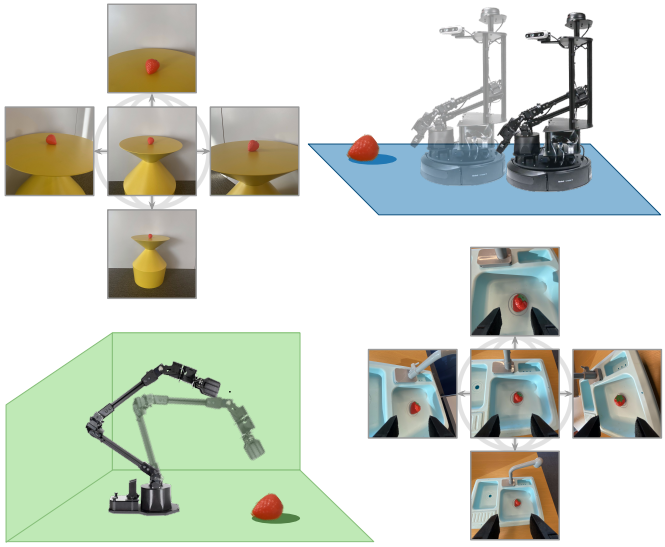

Fig. 2: Unifying Manipulation and Navigation. Despite having fundamentally different objectives, similar actions lead to similar transformations in the egocentric observations for both manipulators and navigators. We hypothesize that this equivariance can assist training of shared egocentric control policies across both morphologies.

A. Manipulation and Navigation as Unified Goal-Reaching

Consider a manipulator reaching for an object from egocentric observations and a navigator trying to reach a waypoint with an onboard front-facing camera. While these tasks span significantly different embodiments and have different action representations, their objectives are similar: to predict a sequence of actions that moves them from the current state to the desired goal. However, their similarities do not end there. Let us define a shared action space coordinate system where $+x$ denotes moving into the image, $+y$ denotes moving left with respect to the image, and $+z$ denotes moving up with respect to the image. Given a certain action and assuming that the robots can move in all directions from any state, the manner in which the observations for a robotic arm transform in response to actions is equivariant to that of a navigation platform. That is, moving "left" will change the manipulator's observation in the same manner as the navigator's. Figure 2 depicts this structure, showing that similar actions will lead to similar homographies of a desired object.

Based on this observation, we unify navigation and manipulation into a single task. Consider a trajectory $\tau \in D_{e,z}$. For two observations $o_i, o_j, \in \tau$ that are temporally close together, define the action a^* as the difference in the poses of the cameras that generated these observations. Note that a^* is *agnostic to embodiment*, meaning that optimizing an action prediction loss $\mathcal{L}(f(o_i, o_j), a^*)$, where $f(o_i, o_j)$ tries to predict a^* given its current and goal observation, will not result in fitting a different target regardless of whether o_i and o_j come

from a manipulation dataset $D_{e_{m,\cdot}}$ or a navigation dataset $D_{e_{n,\cdot}}$. Given that i and j are close enough, we would expect the dataset's action at o_i, or a_i to be similar in direction to a^*. This is because the robot's motion is continuous, and therefore, locally travels in a straight line. Under these assumptions, training our policy to predict action a_i would allow us to learn from $D_{e_{m,1}} \cup D_{e_{m,2}} \cup \ldots \cup D_{e_{n,1}} \cup D_{e_{n,2}}$ with a single, well-defined objective.

Of course, in practice, the limitations of physical robotic systems make this equivalence imperfect. First, the dataset action label might not correspond to a straight-line path from the current observation o_i to the goal when the goal is further away – a manipulator might need to grasp an object, and a ground robot might need to drive around an obstacle. Secondly, the assumption that any robot can maneuver in any direction from any state is simply untrue. Manipulators are constrained by their joint limits and degrees of freedom. Ground robots cannot move "upwards" against gravity. In addition, unifying each robot's actions such that moving in a certain direction will correspond to the same change in its egocentric camera parameters is infeasible. The camera parameters of many robotic datasets are not available. Finally, the action magnitudes of different embodiments such as cars and drones may operate on a different scale. However, we still expect that the *local equivariance* provided by this representation should make cross-embodiment training significantly easier.

B. Aligning the Action Coordinate Frames

To approximate the ideal action space, where the action corresponds to a change in the camera frame, to the greatest extent possible, we learn a *normalized, embodiment-specific* direction. This direction outlines a broad delta-position vector that each robot must move towards, yet allows variations in the scale and strategies that different robots employ to move there. Firstly, for each training dataset D_e, we normalize the distribution of actions to lie between -1 and 1. This allows the policy to handle different action magnitudes across datasets, which otherwise would cause instability in the action loss. Then, we align the action coordinates across datasets such that each of the action dimensions corresponds to similar transformations in the robots' observations. Ideally, this would be done by first computing the delta Cartesian actions for each robot, then applying a rigid transformation to each action a that maps it to the coordinate frame defined by the camera's extrinsic parameters. However, since most previous large-scale manipulation datasets do not contain this information, in practice, we swap each dataset's action dimensions such that they point in the same direction. Further details are described in the following section.

C. Datasets and Postprocessing

We train our policy on a mixture of our own small manipulation dataset (see Section V-C), 9 datasets from OXE [4], a large-scale dataset for wheeled robot navigation, and a large-scale dataset for autonomous driving. The manipulation datasets from OXE include Bridge [43], Fractal [65],

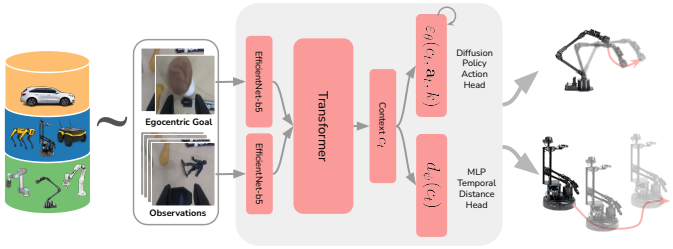

Fig. 3: Policy Architecture. We use separate observation and goal convolutional encoders to tokenize visual observations, which are passed through a Transformer block. The resulting features are used to predict the temporal distance to the goal d_ψ and future actions $\mathbf{a}_t$, using a conditional diffusion process.

Taco Play [67], Jaco Play [68], Roboturk [69], NYU Door Opening [70], Viola [71], Berkeley Autolab UR5 [72], and Toto [73]. The navigation datasets from GNM [5] include GO Stanford [45], SCAND-S/J [46], RECON [74], Cory Hall [75], Seattle [76], and TartanDrive [77]. Additionally, we also train on SACSoN [78] and Berkeley Deep Drive [51]. Each of these datasets is converted to the RLDS format [79]. We upweight the frequency at which navigation data appears such that 50% of the entire data mixture is navigation data and 50% is manipulation. This ensures that the policy fits evenly to the two domains, which is important not to degrade the performance of one domain in favor of another. For the manipulation share of data mixture, we weight the datasets by a similar split to prior work [55]. For navigation, we weight the datasets by their relative number of trajectories. For further information, we refer the reader to Section IX-A.

We post-process the manipulation datasets by aligning the coordinate frames of the actions as described in the previous section. Note that OXE does not contain consistent action coordinate systems for each robot. After converting each of the datasets to delta Cartesian actions, the dimension 0 can correspond to the robot moving left, right, or forward depending on their control scheme. Therefore, we alter the datasets inside our mixture by correcting each of the dimensions of the action coordinates frames such that each dimension of the action corresponds to the same general direction of the end effector. This is done by manually sampling (observation, action, next observation) pairs for each dataset and observing the change in robot pose with respect to the actions. For coordinate systems that don't align with the coordinate system of our manipulation dataset, we swap the dimensions and signs of the actions to be more consistent. For manipulation, we use a 7-dimensional action space with zero-indexed dimensions $0-2$ as delta Cartesian actions, $3-5$ as delta rotations, and 6 as gripper open/close.

Each of the navigation datasets contains a sequence of egocentric images and states containing a position $p = (x, y)$ as well as a yaw ϕ. For each action, we subtract the current state from the next 5 future states, then transform these differences with a rotation matrix defined by the yaw in order to get egocentric waypoints. These actions are transformed into

the manipulation coordinate frame as described in the previous section. Since the egocentric camera is pointed downwards and towards the gripper for our manipulation tasks, we map the "forward" $+y$ axis in our navigation datasets to the $-z$ downwards direction in our manipulation datasets. We also map the "left" $+x$ direction in our navigation dataset to the "left" $+y$ direction in our manipulation datasets. Therefore, we translate navigation action $a \in D_{e_n,t}$ to $(0, a[1], -a[0], 0, 0, 0, 0)$.

D. Policy Architecture

To fit a single policy to datasets across a wide variety of embodiments, the neural architecture must be both scalable and expressive. We design our model to scale up a simple transformer backbone. At a high level, we want our model to process its observations using some encoder, feed its embeddings into a transformer, and then output both an action and the distance to its goal. We use combined insights from previous large-scale robot models in manipulation [65] and navigation [5] to motivate our design decisions. Firstly, our choice of observation encoders is EfficientNet ConvNets [80], which have been used successfully in robot learning for both navigation and manipulation. For our action output head, we chose to use a diffusion policy [81] to account for noise in human demonstrations as well as different strategies that may exist to reach the goal state. In addition, we both incorporate history and predict future actions, parameterizing our policy as $\pi(a_{t:t+k-1}|o_{t-c+1:t}, o_g)$. This decision stems from prior work indicating that policies trained with past states and future actions exhibit significant improvements in their ability to fit to teleoperated demonstration data [44, 82].

Our heterogeneous cross-embodiment model consists of five different components: two observation encoders, a transformer, a diffusion policy action head [81], and an MLP distance prediction head for navigation with topological graphs. Similar to the scheme proposed by NoMaD [83], we tokenize the observation history $o_{t-k:t}$ into $k+1$ embeddings of size 2048 with an EfficientNet-b5 encoder. Each token corresponds to a frame in the observation history. We then concatenate the current observation o and goal observation o_g in a channel-wise manner and compute an additional embedding with a separate EfficientNet-b5 encoder. The resulting $k+2$ embeddings are then fed to the transformer and averaged to obtain a context $c_t = f_\theta(o_{t-k:t}, o_g)$. Further architecture details are provided in Table VI.

The context is then fed into an action prediction head and a distance prediction head. The action prediction is reshaped into a tensor of size $(b, n, 7)$, while the distance prediction is reshaped into a tensor of size $(b, 1)$, where b is the batch size and n is the number of actions we predict into the future. Note that this distance prediction is used by navigation policies to localize the robot with respect to a topological map, but not used by manipulation policies. Regardless, distance to the goal is a well-defined task, even in manipulation.

We train our policy with diffusion denoising loss (refer to Appendix IX-B for more details)

$$\mathcal{L}_{\text{diffusion}}(\theta, \psi) = ||\epsilon_k - \epsilon_\phi(f_\theta(o_{t-k:t}, o_g), a_t^0 + \epsilon_k, k)||_2^2,$$

and a distance prediction loss

$$\mathcal{L}_{\text{distance}}(\theta, \psi) = ||d_\psi(f_\theta(o_{t-k:t}, o_g)) - d_t||_2^2.$$

Our overall objective is the weighted combination of these two losses:

$$\mathcal{L}(\theta, \phi, \psi) = \mathcal{L}_{\text{diffusion}}(\theta, \psi) + \lambda \mathcal{L}_{\text{distance}}(\theta, \psi).$$

In practice, we find that $\lambda = 0.001$ is a reasonable value, which ensures that the distance head is trained but doesn't interfere with the action loss. $f_\theta(o_{t-k:t}, o_g)$ denotes the observation encoder and transformer, ϵ_ϕ denotes the noise prediction head, and d_ψ denotes the noise prediction head. The noise prediction network tries to predict the noise at iteration k, or ϵ_k from the noisy action $a_t^0 + \epsilon_k$, where a_t^0 denotes a flattened action from the dataset. In addition, d_t denotes the distance in timesteps from the current observation and goal observation. The goal image is sampled uniformly at random 20 to 40 timesteps into the future from the current observation. This firstly provides local goals such that the direction between the current observation and the goal and be ascertained. Secondly, this allows our method to scale to datasets within OXE that contain long sequences of observations.

Since the majority of OXE consists of third-person observations without wrist-image counterparts, when sampling new batches, we select between these two viewpoints uniformly at random if both exist or use the available one. Our experiments show that in certain domains, co-training with 3rd-person images can greatly increase the success rate of the policy. For these results, we refer the reader to Appendix IX-F. As more large-scale wrist-camera datasets are released in the future, we believe that this gap will close.

V. Evaluation

Our goal is to evaluate the performance of heterogeneous cross-embodiment policies in solving real-world manipulation and navigation tasks on a variety of embodiments. In addition, we aim to investigate the possibility of knowledge transfer across these embodiments. To this end, we seek to answer the following questions:

1) Can a single goal-conditioned policy successfully control widely varying embodiments for both navigation and manipulation?
2) Can co-training with navigation data provide generalization benefits to manipulation policies?
3) How does navigation data help manipulators generalize?
4) What kind of navigation data enables better transfer to manipulation tasks?
5) Can co-training with manipulation data provide generalization benefits to navigation policies?
6) Can heterogeneous cross-embodiment policies generalize zero-shot to new embodiments?

A. Evaluation Embodiments

In order to demonstrate the ability of our policy to fit to a wide range of embodiments, we evaluate on five low-cost,

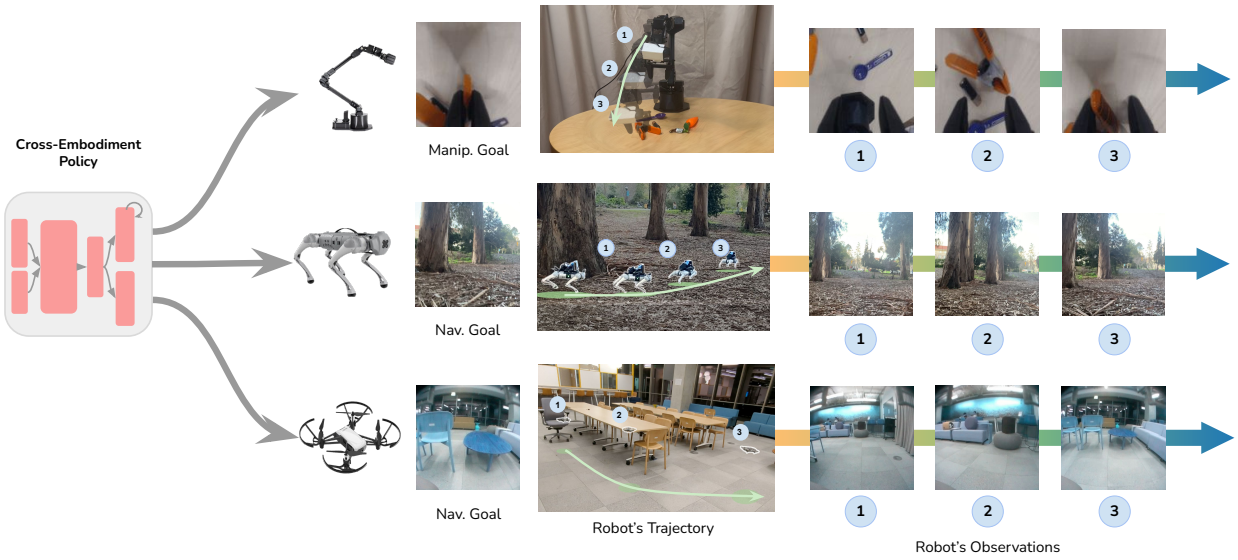

Fig. 4: Qualitative examples of the *same policy checkpoint* deployed on a tabletop manipulator solving the "Cluttered Grasp" task (top), a quadruped navigating to a goal in a forest (middle), and a drone navigating a cluttered office environment (bottom).

open-source robot manipulators and mobile robots, including a mobile manipulator (see Figure 1):

- **WidowX250S:** A 6-DoF robotic arm with a parallel jaw gripper, with a camera on the wrist.
- **LoCoBot:** An indoor mobile robot with a forward-facing camera.
- **Clearpath Jackal:** A fast mobile robot with a forward-facing camera capable of moving indoors and outdoors.
- **DJI Tello:** A quadrotor with a forward-facing camera.
- **Unitree Go1:** A quadruped with a forward-facing camera.
- **Mobile ALOHA:** A bimanual mobile manipulation platform with a mobile base and two ViperX arms [84]. Each arm has a camera on the wrist, and the base has a forward-facing camera.

B. Manipulation and Navigation Tasks

We aim to test whether navigation data can provide generalization benefits to manipulators. We evaluate our method on 5 tasks outlined below. These tasks were constructed to require information from the goal image for the policy to obtain high success.

1) **Two-object Reaching.** A simple environment with two different objects to the left and to the right of the manipulator. Given the goal image, the manipulator needs to move to the correct object, similar to the setup in navigation.
2) **Cluttered Grasp.** A grasping task where the robot has to pick the correct object out of 5 different objects seen in its training data. The positions and interactions between the objects are randomized and may not be seen in the training dataset.
3) **Novel Cluttered Grasp.** A cluttered grasping task with 5 held-out objects. The manipulator needs to pick the correct object as specified by the goal image.
4) **Toy Kitchen.** A more semantically meaningful environment where the robot needs to pick up a strawberry and eggplant from the sink. This exact environment is not explicitly seen in the training data, although similar toy kitchen setups exist in the BRIDGE dataset.
5) **Shelf Manipulation.** A task where the WidowX250S has to pick out the right object from a shelf compartment. This location of the shelf is randomized. This task evaluates the ability of the policy to generalize to variations in distance to an axis perpendicular to the egocentric camera while avoiding colliding with the shelf.

To evaluate our navigation policies, we chose two novel locations that were not seen in any of the training datasets.

1) **Office Hallway:** A navigation task in a hallway with clutter. The robot must navigate around two corners without colliding with obstacles and stop at the last goal location.
2) **Office Kitchen:** A navigation task in a more open kitchen environment. As with the previous task, the robot must navigate to the final goal image location without colliding with obstacles.

Before rolling out our policy for manipulation tasks, we collect a goal image for each task by rearranging the environment and teleoperating the manipulator to the desired state. For navigation, we create a topological map $\mathcal{M}$ by recording the robot's observations with a frequency of 4 Hz while moving the robot base throughout the environment. We ensure that this map has sufficient coverage of the locations which the robot may traverse during evaluation time.

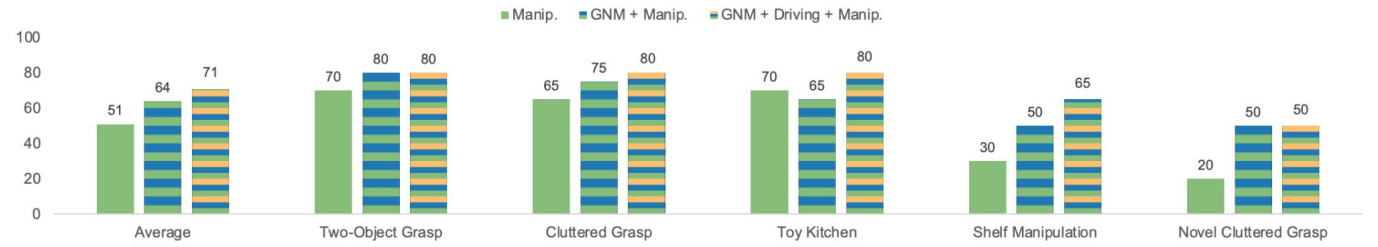

Fig. 5: Does navigation help manipulation? By aligning action coordinate frames, training on navigation and driving datasets results in a 20% improvement across five challenging tabletop manipulation tasks (success % on y-axis).

C. Experiment Setup

To investigate the impact of certain navigation datasets on manipulation and vice versa, we ablate including the various datasets. For clarity, we label the GNM dataset as **GNM**, the BDD100k dataset as **Driving** and the combination of OXE and our own tabletop manipulation dataset as **Manip**. Although we can obtain successful results on our navigation robots by solely using prior open-source datasets, collecting our own dataset specific to our embodiments and control schemes was necessary for manipulation. While navigation policies generalize in zero-shot to new embodiments, current manipulation policies require in-distribution data for the target embodiment and control scheme in order to identify the correct action space and understand important visual features of the robot [4]. Therefore, to ensure a nonzero success rate for our method on our embodiments, we collect our own manipulation data. This dataset contains expert demonstrations on the WidowX250s and ViperX300s collected via VR teleoperation. The WidowX250s dataset consists of manipulation with a set of 5 training objects. For each in-distribution object, we collect 50 grasping demonstrations while randomizing the locations of the other objects. The ViperX300s dataset consists of 50 pick and place trajectories each for a set of 3 training objects. This entire dataset spans 400 trajectories collected over the course of 8 hours.

VI. Analysis

A. Can a single goal-conditioned policy successfully control widely varying embodiments for both navigation and manipulation?

We report evaluation results for our method for manipulation in Figure 5 and for navigation results in Figure 6. Our method obtains an average of 71% success rate over 5 different manipulation tasks, and an average of 80% success rate on 2 navigation tasks each on 4 different embodiments. This demonstrates our policy's ability to fit to both manipulation and navigation datasets. In addition, our policy can identify and output an action for the appropriate embodiment given its current and goal observations. For example, when given observations from a mobile manipulator, the policy outputs a waypoint in the second and third dimensions along with values that are near 0 in the other dimensions. This is consistent with the scheme we used to align manipulation and navigation actions.

B. Can co-training with navigation data provide generalization benefits to manipulation policies?

Figure 5 shows the success rate of various dataset mixtures on manipulation tasks. Training our policy on a manipulation and navigation data split had a 20% greater success rate over 5 tasks compared to training only on manipulation data. The largest gap in performance between the joint navigation-manipulation policies and the manipulation-only policies was in the Novel Cluttered Grasp and Shelf Manipulation scenarios. These scenarios involve spatial reasoning in novel environments (e.g., in Shelf Manipulation, the policy must learn which actions don't collide with the shelf), requiring the policy to understand the location of its current state with respect to the goal.

For the Cluttered Grasp tasks, the gap in performance between the joint navigation-manipulation policy is larger in the out-of-distribution variant than the in-distribution variant. A plausible explanation could be that the navigation data regularizes the policy's intermediary representations to capture relative spatial information between current and goal images. In Shelf Manipulation, the robot needs to grasp an object located on a shelf with randomized positions. This requires the robot to avoid colliding with the shelf as well as gauge its distance to the object, which is fundamentally similar to the collision avoidance task in ground navigation. Gauging object distance is analogous to testing the robustness to a change in table height in tabletop manipulation, which previous works have identified as a common distribution shift artifact leading to failure [85, 86].

C. Can co-training with manipulation data provide generalization benefits to navigation policies?

Figure 6 reports our navigation results. Each dataset mixture was evaluated on four different robots across two indoor domains, then averaged to get a success rate. Three of these robots–the LoCoBot, Jackal, and Unitree Go1–were present in the training dataset, while the DJI Tello is a novel embodiment. Due to a difference in the camera lens used by the DJI tello, we noticed that the performance of the drone degraded significantly in environments without clear markers of corners. Therefore, while the LoCoBot, Clearpath Jackal, and Unitree Go1 were evaluated in everyday environments, the DJI Tello had to be evaluated in scenes with bright objects to indicate corners at which to turn.

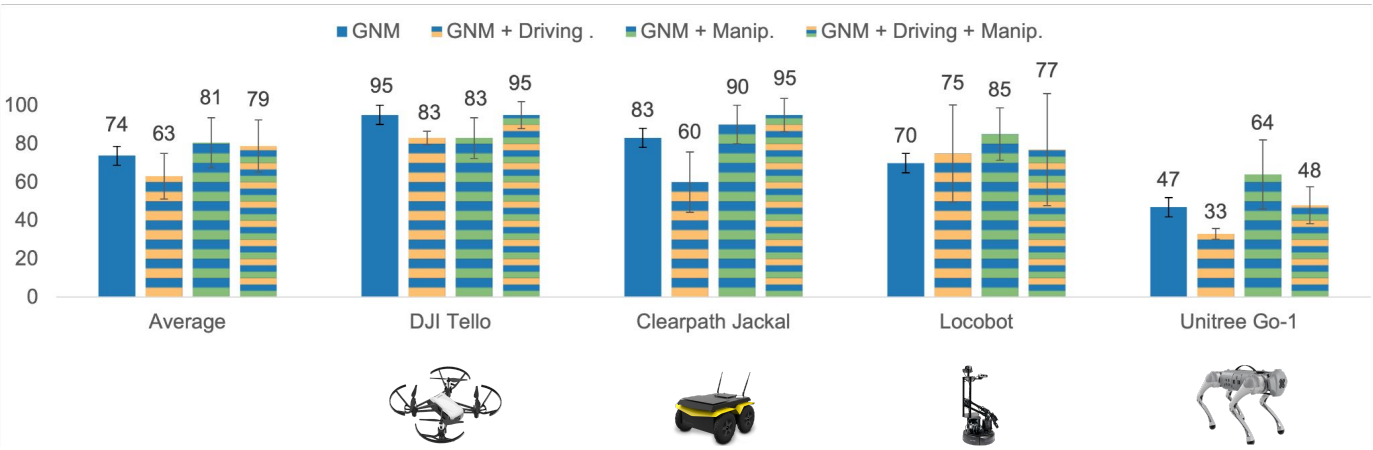

Fig. 6: Does manipulation help navigation? Across three different robots in challenging indoor and outdoor environments, adding manipulation datasets leads to $5-7\%$ improvement in navigation performance (success % on y-axis).

On the Jackal, LoCoBot, and Unitree Go1, we observed greater success rates for policies that were co-trained with navigation and manipulation data, with *GNM + Driving + Manip* being 12%, 7% and 1% better than *GNM-only* respectively. With *GNM + Manip*, the success rate is 7%, 15%, and 17% respectively. On the DJI Tello, the *GNM + Driving + Manip* performs similarly to the *GNM-only* policy, each with 95% success rate. Averaged over the embodiments, the policy trained with manipulation data had $5-7\%$ higher success than the navigation-only policy. While we qualitatively observed that these policies had better estimates for the closest node and had less collision with the environment, we acknowledge that the $5-7\%$ difference is not particularly large and can potentially be explained due to the variance between evaluation runs. We believe that this gap will widen more in-the-wild manipulation data or mobile manipulation data in the future.

D. How does navigation data help manipulators generalize?

Datasets	Cluttered	Novel Cluttered	Shelf
GNM + BDD + Manip	0.749	0.740	0.644
GNM + Manip	0.743	0.735	0.648
Manip-only	0.733	0.703	0.614

TABLE I: Embedding Analysis. Transformer features for policies co-trained with all datasets have a stronger linear correlation (R^2 coeff.) with the temporal distance than manip.-only policies.

To investigate our hypothesis that navigation data can help a manipulator understand its position with respect to its goal, we collected a small dataset of trajectories for each policy and computed our policy's embedding before the action and distance heads. Then, we ran canonical correlation analysis between these features and the temporal distance between the current state and the goal and recorded the resulting coefficients of determination (R^2) in Table I. Our results show a positive correlation between ratio of the coefficient of determination between data splits and the ratio of the success rates on manipulation tasks. In particular, the ratio of the performance of the full manipulation-navigation policy and the manipulation-only policy are 1.230, 2.75, and 1.14 on the Cluttered Grasp, Novel Cluttered Grasp, and Shelf Manipulation tasks. The ratio of the R^2 coefficients between these policies are 1.011, 1.061, and 1.049 respectively. The observation that higher correlation values are indicative of better performance supports our hypothesis that goal-conditioned policies co-trained with navigation data better understand their relationship with a goal image.

To further examine whether information from the goal image is essential to transferring navigation data to manipulation, we ran an ablation of our method without goal-conditioning. Table II shows the results of these policies in the novel cluttered grasp task. Note that while goal-conditioned experiments record the proportion of trials in which the robot grasped the *correct* object, the unconditioned experiments record the proportion of trials in which the robot grasped *any* object. The gap in performance between adding *GNM + Manip* and *Manip-only* in the goal-conditioned and the unconditioned policies are 35% and 5% respectively. Operating under the assumption that the diffusion policy is powerful enough to model the different possible tasks from the current observation without conditioning on the goal image, we can conclude that information transfer between manipulation and navigation policies is negligible without goal conditioning.

E. What kind of navigation data enables better transfer to manipulation tasks?

We ablate which datasets inside of GNM [5] we co-train with to investigate which types of navigation environments are more conducive to transfer to manipulation. We provide further information about the location and length of each dataset in Appendix IX-D. Figure 7 indicates that trail-like outdoor datasets such as Tartan and Seattle do not provide positive transfer to manipulation scenarios. Meanwhile co-training with indoor manipulation environments such SACSoN and GO Stanford lead to significant improvements in manipulation performance. We hypothesize that this difference is due to

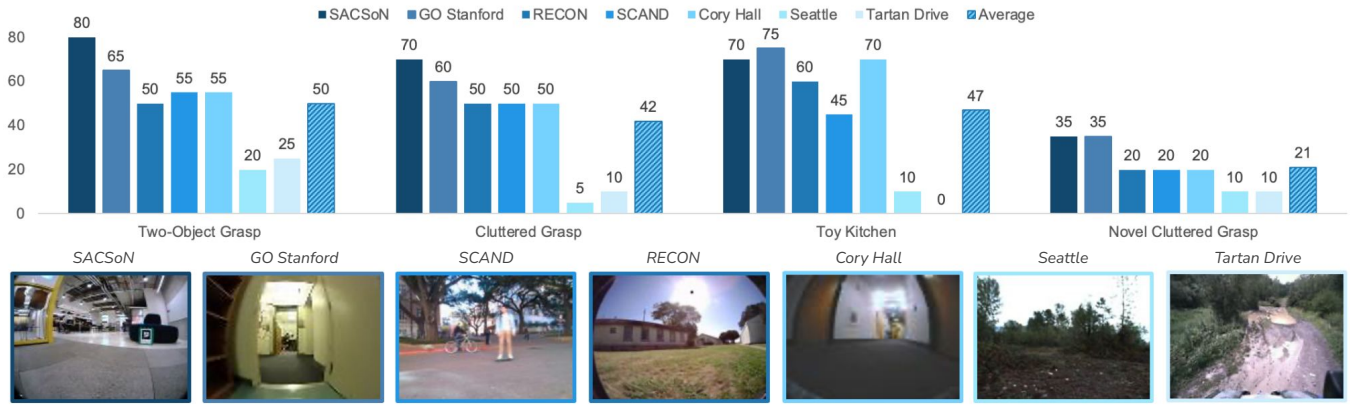

Fig. 7: What type of navigation data helps positive transfer? Manipulation policies co-trained with indoor and outdoor navigation data on sidewalks perform better than policies co-trained on forest trails and off-road environments.

the presence of sharper angles and objects with well-defined boundaries in indoor navigation data.

GNM + Manip GC	Manip-only GC	GNM + Manip UC	Manip-only UC
55%	20%	45%	40%

TABLE II: Is goal-conditioning important for transfer? There is a 30% higher gap in performance between goal-conditioned (GC) co-trained policies and manipulation-only policies compared to unconditioned (UC).

F. Can heterogeneous cross-embodiment policies generalize zero-shot to new embodiments?

By training our policies with both manipulation and navigation data, heterogeneous cross-embodiment policies can allow robots that require both manipulation and navigation to leverage preexisting domain-specific large-scale datasets. To test the limit of this generalization capability, we evaluate our policy on the Mobile Aloha platform [84]. While the robot is capable of bimanual mobile manipulation, we simplify the platform by only using the right manipulator. To obtain actions for both navigation and manipulation, we run our policy twice: one from an egocentric camera mounted to the robot arm, and one from the navigation camera. We threshold the magnitude of the policy's action prediction to determine when to run the manipulation policy. Namely, if the policy is close to the final goal image and the actions are small, we allow the manipulation policy to control the robot arm.

We evaluate our policy on the *Egg Nav/Pick/Place* task, where the robot has to approach a table, pick up an egg, and place it onto a plate (see Fig. 8). Despite the fact that neither the table nor the egg was seen in the training data of the policy, the robot achieves a 50% success rate, demonstrating the method's effectiveness in controlling a new embodiment. Qualitatively, the robot succeeds when the robot manipulator is located in a good position with respect to the object. However, small changes in the mobile base can elicit large changes in position of the robot arm with respect to the scene, and the robot can fail to move its base such that its arm into a favorable position before executing the task.

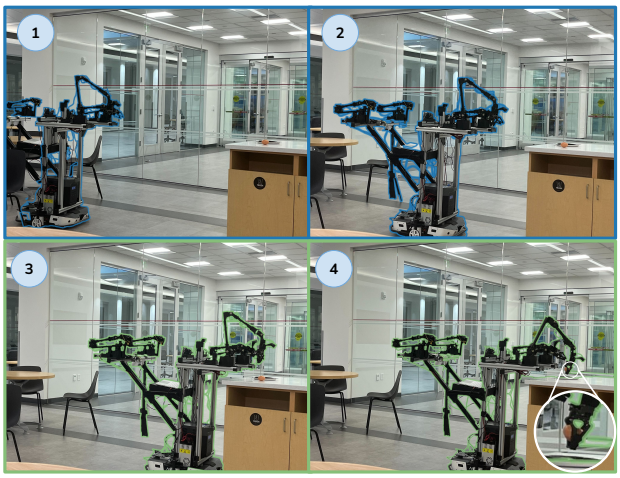

Fig. 8: Our cross-embodiment policy trained on manipulation and navigation data zero-shot generalizes to a mobile manipulator, succeeding in the "Egg Nav/Pick/Place" task.

Datasets	Egg Nav/Pick/Place
GNM + Driving + Manip	50%

TABLE III: Zero-shot Embodiment Generalization Experiments. Our policy demonstrates the ability to transfer to new embodiments that were not seen in the training data.

VII. Conclusion

In this paper, we analyze the effect of dataset diversity in cross-embodiment learning by blurring the boundary between navigation and manipulation. We hypothesize that projecting the different robotic tasks into a unified goal-reaching framework can lead to improved transfer of learned behaviors across embodiments, as well as to novel embodiments. We train the first *heterogeneous* cross-embodiment policy capable of controlling a variety of diverse robots — robotic arms, wheeled and legged mobile platforms, drones, and mobile manipulators

— in diverse real-world environments. We conduct over 1000 experiments to empirically characterize the effects of dataset size and variability, model size, and architecture choices. Our experiments reveal that policies co-trained with all manipulation and mobile data demonstrate an average of 20% improvement over 5 different manipulation tasks than training with manipulation data alone, and a $5-7\%$ improvement over 4 different navigation platforms. We believe such results are not merely quirks of the chosen datasets, but signs that there exists valuable information to transfer across seemingly different robot embodiments. Generalist manipulation agents would benefit from the large perceptual diversity and rich spatial relationships captured by navigation datasets, and generalist navigation agents would benefit from more the rich object-centric interactions present in manipulation datasets.

Our methodology does have a number of limitations. First, although we show that our policies can control both manipulators and mobile robots, our framework does not support systems that require controlling varying degrees of freedom. For example, although we show control of a quadrupedal robot, we are still controlling it at the level of overall heading rather than at the individual joints. Some robots, such as multi-fingered hands, do not readily support such abstraction, and extending our framework to handle varying numbers of degrees of freedom is an exciting direction for future. Second, all of our results focus on goal-conditioned policies that are tasked with a goal image. This modality is pragmatic and easy to evaluate, but not necessarily the most convenient for human users. Other task modalities, such as language, might be more useful in practice, and extending our framework to support this would also be valuable. These future improvements would make cross-embodiment training even more useful, and we hope that our work represents a step toward achieving greater synergy between robotic embodiments in the future, towards the goal of a true "robot foundation model" that can leverage data from all robots and control any robot out of the box.

ACKNOWLEDGMENTS

This research was supported by ONR grants N00014-22-1-2621 and N00014-22-1-2293, ARL DCIST CRA W911NF-17-2-0181, and NSF IIS-2150826, Ford, and Volkswagen. The authors would like to thank Pete Florence, Laura Smith, and Colin Li, for their helpful feedback on the paper. In addition, the authors would like to thank the IRIS, ILIAD, and RAIL labs for the numerous discussions about training generalist robotic policies and cross-embodiment learning.

VIII. CONTRIBUTIONS

Jonathan Yang: Led model development and tuning, wrote dataloader/data processing, wrote manipulation controllers, ran manipulation evaluations, ran experiments on the mobile manipulator.

Catherine Glossop: Tuned cross-embodiment models, implemented and trained ablations models, evaluated runs for the locobot, created plots/videos for the paper.

Arjun Bhorkar: Worked on data processing and model training for navigation, created navigation pipelines and ran evaluations for the DJI Tello, Clearpath Jackal and Unitree Go1.

Dhruv Shah: Provided guidance for the project and technical report, helped resolve issues with navigation.

Quan Vuong: Provided guidance for the project, helped with utilizing and dataloading from RT-X.

Chelsea Finn, Dorsa Sadigh, Sergey Levine: Provided guidance for the project and the technical report.

REFERENCES

[1] R. Bommasani *et al.*, "On the opportunities and risks of foundation models," 2022.

[2] 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 *Annual Conference on Robot Learning (CoRL)*, 2019.

[3] H.-S. Fang, H. Fang, Z. Tang, J. Liu, C. Wang, J. Wang, H. Zhu, and C. Lu, "Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot," 2023.

[4] Open X-Embodiment Collaboration *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," *arXiv preprint arXiv:2310.08864*, 2023.

[5] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine, "GNM: A General Navigation Model to Drive Any Robot," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[6] Y.-H. H. Tsai, V. Dhar, J. Li, B. Zhang, and J. Zhang, "Multimodal large language model for visual navigation," 2023.

[7] T.-H. Wang, A. Maalouf, W. Xiao, Y. Ban, A. Amini, G. Rosman, S. Karaman, and D. Rus, "Drive anywhere: Generalizable end-to-end autonomous driving with multi-modal foundation models," *arXiv preprint arXiv:2310.17642*, 2023.

[8] J. Lin, A. Zeng, S. Lu, Y. Cai, R. Zhang, H. Wang, and L. Zhang, "Motion-x: A large-scale 3d expressive whole-body human motion dataset," *Advances in Neural Information Processing Systems*, 2023.

[9] 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 (RSS)*, 2017.

[10] T. Chen, A. Murali, and A. Gupta, "Hardware conditioned policies for multi-robot transfer learning," 2019.

[11] H. You, T. Yang, Y. Zheng, J. Hao, and E. Taylor, Matthew, "Cross-domain adaptive transfer reinforcement learning based on state-action correspondence," in *Uncertainty in Artificial Intelligence*, 2022.

[12] E. S. Hu, K. Huang, O. Rybkin, and D. Jayaraman, "Know thyself: Transferable visual control policies through robot-awareness," in *International Conference on Learning Representations (ICLR)*, 2022.

[13] G. Salhotra, xI Chun Arthur Liu, and G. Sukhatme, "Bridging action space mismatch in learning from demonstrations," *arXiv preprint arXiv:2304.03833*, 2023.

[14] J. H. Yang, D. Sadigh, and C. Finn, "Polybot: Training one policy across robots while embracing variability," in *Annual Conference on Robot Learning (CoRL)*, 2023.

[15] P. F. Christiano, Z. Shah, I. Mordatch, J. Schneider, T. Blackwell, J. Tobin, P. Abbeel, and W. Zaremba, "Transfer from simulation to real world through learning deep inverse dynamics model," *ArXiv preprint arXiv:1610.03518*, 2016.

[16] F. Sadeghi, A. Toshev, E. Jang, and S. Levine, "Sim2real view invariant visual servoing by recurrent control," in *International Conference on Robotics and Automation (ICRA)*, 2017.

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

[18] Q. Zhang, T. Xiao, A. A. Efros, L. Pinto, and X. Wang, "Learning cross-domain correspondence for control with dynamics cycle-consistency," in *International Conference on Learning Representations (ICLR)*, 2021.

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

[20] T. Wang, R. Liao, and S. F. Jimmy Ba, "Nervenet: Learning structured policy with graph neural networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[21] 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.

[22] A. Ghadirzadeh, X. Chen, P. Poklukar, C. Finn, M. Björkman, and D. Kragic, "Bayesian meta-learning for few-shot policy adaptation across robotic platforms," in *International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[23] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "Exaug: Robot-conditioned navigation policies via geometric experience augmentation," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[24] M. Attarian, M. A. Asif, J. Liu, R. Hari, A. Garg, I. Gilitschenski, and J. Tompson, "Geometry matching for multi-embodiment grasping," 2023.

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

[26] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks," 2019.

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

[28] L. Shao, F. Ferreira, M. Jorda, V. Nambiar, J. Luo, E. Solowjow, J. A. Ojea, O. Khatib, and J. Bohg, "Unigrasp: Learning a unified model to grasp with multi-fingered robotic hands," *IEEE Robotics and Automation Letters*, 2020.

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

[30] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak, "Affordances from human videos as a versatile representation for robotics," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 01–13.

[31] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "ViNT: A foundation model for visual navigation," in *Annual Conference on Robot Learning (CoRL)*, 2023.

[32] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," 2016.

[33] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," 2017.

[34] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, "Learning invariant feature spaces to transfer skills with reinforcement learning," 2017.

[35] K. Fang, Y. Bai, S. Hinterstoisser, S. Savarese, and M. Kalakrishnan, "Multi-task domain adaptation for deep learning of instance grasping from simulation," in *International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3516–3523.

[36] N. H. Kim, Z. Xie, and M. van de Panne, "Learning to correspond dynamical systems," 2020.

[37] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *International Conference on Computer Vision (ICCV)*, 2017, pp. 2242–2251.

[38] H. You, T. Yang, Y. Zheng, J. Hao, and E. Taylor, Matthew, "Cross-domain adaptive transfer reinforcement learning based on state-action correspondence," in *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, ser. Proceedings of Machine Learning Research, J. Cussens and K. Zhang, Eds., vol. 180. PMLR, 01–05 Aug 2022, pp. 2299–2309.

[39] P. Sharma, L. Mohan, L. Pinto, and A. Gupta, "Multiple interactions made easy (mime): Large scale demonstrations data for imitation," 2018.

[40] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," in *Annual Conference on Robot Learning (CoRL)*, 2018.

[41] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," 2020.

[42] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Conference on Robot Learning*, 2021.

[43] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," in *Robotics: Science and Systems*, 2022.

[44] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," 2023.

[45] N. Hirose, F. Xia, R. Martín-Martín, A. Sadeghian, and S. Savarese, "Deep visual mpc-policy learning for navigation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3184–3191, 2019.

[46] H. Karnan *et al.*, "Socially CompliAnt Navigation Dataset (SCAND): A Large-Scale Dataset Of Demonstrations For Social Navigation," *IEEE Robotics and Automation Letters*, 2022.

[47] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3213–3223.

[48] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361.

[49] J. Geyer *et al.*, "A2d2: Audi autonomous driving dataset," 2020.

[50] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The apolloscape open dataset for autonomous driving and its application," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[51] 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," 2020.

[52] S. Ettinger *et al.*, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *International Conference on Computer Vision (ICCV)*, 2021.

[53] S. Reed *et al.*, "A generalist agent," 2022.

[54] K. Bousmalis *et al.*, "Robocat: A self-improving foundation agent for robotic manipulation," *ArXiv*, 2023.

[55] Octo Model Team *et al.*, "Octo: An open-source generalist robot policy," https://octo-models.github.io, 2023.

[56] A. Hu, L. Russell, H. Yeo, Z. Murez, G. Fedoseev, A. Kendall, J. Shotton, and G. Corrado, "Gaia-1: A generative world model for autonomous driving," 2023.

[57] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner, "Zero-shot category-level object pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[58] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Imitation learning for vision-based manipulation with object proposal priors," 2023.

[59] Y. Zhu, Z. Jiang, P. Stone, and Y. Zhu, "Learning generalizable manipulation policies with object-centric 3d representations," in *7th Annual Conference on Robot Learning*, 2023.

[60] R. Bonatti, S. Vemprala, S. Ma, F. Frujeri, S. Chen, and A. Kapoor, "Pact: Perception-action causal transformer for autoregressive robotics pre-training," 2022.

[61] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang, "Language-driven representation learning for robotics," in *Robotics: Science and Systems (RSS)*, 2023.

[62] Y. Du, M. Yang, B. Dai, H. Dai, O. Nachum, J. B. Tenenbaum, D. Schuurmans, and P. Abbeel, "Learning universal policies via text-guided video generation," *arXiv e-prints*, pp. arXiv–2302, 2023.

[63] Z. Xian, T. Gervet, Z. Xu, Y.-L. Qiao, T.-H. Wang, and Y. Wang, "Towards generalist robots: A promising paradigm via generative simulation," 2023.

[64] M. Yang, Y. Du, K. Ghasemipour, J. Tompson, L. Kaelbling, D. Schuurmans, and P. Abbeel, "Learning interactive real-world simulators," 2024.

[65] A. Brohan *et al.*, "Rt-1: Robotics transformer for real-world control at scale," 2023.

[66] ——, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," 2023.

[67] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard, "Latent plans for task agnostic offline reinforcement learning," in *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[68] S. Dass, J. Yapeter, J. Zhang, J. Zhang, K. Pertsch, S. Nikolaidis, and J. J. Lim, "Clvr jaco play dataset," 2023. [Online]. Available: https://github.com/clvrai/clvr_jaco_play_dataset

[69] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1048–1055.

[70] J. Pari, M. Shafiullah, S. Arunachalam, and L. Pinto, "Visual imitation through nearest neighbors (vinn) implementation," 2021.

[71] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Imitation learning for vision-based manipulation with object proposal priors," *6th Annual Conference on Robot Learning (CoRL)*, 2022.

[72] L. Y. Chen, S. Adebola, and K. Goldberg, "Berkeley UR5 demonstration dataset," https://sites.google.com/view/berkeley-ur5/home.

[73] G. Zhou, V. Dean, M. K. Srirama, A. Rajeswaran, J. Pari, K. Hatch, A. Jain, T. Yu, P. Abbeel, L. Pinto, C. Finn, and A. Gupta, "Train offline, test online: A real robot learning benchmark," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[74] D. Shah, B. Eysenbach, N. Rhinehart, and S. Levine, "Rapid exploration for open-world navigation with latent goal models," in *Annual Conference on Robot Learning (CoRL)*, 2022.
[75] G. Kahn, A. Villaflor, B. Ding, P. Abbeel, and S. Levine, "Self-Supervised Deep RL with Generalized Computation Graphs for Robot Navigation," in *International Conference on Robotics and Automation (ICRA)*, 2018.
[76] 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)*, 2022.
[77] S. Triest *et al.*, "TartanDrive: A Large-Scale Dataset for Learning Off-Road Dynamics Models," in *International Conference on Robotics and Automation (ICRA)*, 2022.
[78] N. Hirose, D. Shah, A. Sridhar, and S. Levine, "Sacson: Scalable autonomous control for social navigation," *IEEE Robotics and Automation Letters*, 2024.
[79] S. Ramos, S. Girgin, L. Hussenot, D. Vincent, H. Yakubovich, D. Toyama, A. Gergely, P. Stanczyk, R. Marinier, J. Harmsen, O. Pietquin, and N. Momchev, "Rlds: an ecosystem to generate, share and use datasets in reinforcement learning," 2021.
[80] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 6105–6114.
[81] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.
[82] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *arXiv preprint arXiv:2108.03298*, 2021.
[83] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "Nomad: Goal masked diffusion policies for navigation and exploration," 2023.
[84] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," 2024.
[85] K. Hsu, M. J. Kim, R. Rafailov, J. Wu, and C. Finn, "Vision-based manipulators need to also see from their hands," 2022.
[86] A. Xie, L. Lee, T. Xiao, and C. Finn, "Decomposing the generalization gap in imitation learning for visual robotic manipulation," 2023.
[87] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," 2022.

IX. Appendix

A. Data Postprocessing

1) Dataset Mixture: The following table shows the data mixture we used to train our policy:

Datasets	Split
Ours	1.3%
Bridge [43]	5.3%
Fractal [65]	1.3%
Taco Play [67]	2.6%
Jaco Play [68]	5.3%
Roboturk [40]	2.6%
NYU Door Opening [70]	1.3%
Viola [71]	13.3%
Berkeley Autolab UR5 [72]	5.3%
Toto [73]	5.3%
GNM [5]	46.3%
BDD100k [51]	10%

TABLE IV: Data Splits

Ours denotes a manipulation dataset we collected via teleoperation containing 300 WidowX and ViperX trajectories. We weight the datasets such that navigation accounts for roughly half of the data split. This ensures that the policy would be able to fit our control scheme and egocentric viewpoint. Since BDD100k is comparatively larger than the rest of the datasets, we subsample this dataset to make data loading more efficient.

2) Data Loading: To load our datasets in an efficient manner, we use a tensorflow dataloader with the RLDS format [79]. Although all of RT-X is already in this format, we write conversion scripts for the navigation datasets as well. We filter each of the datasets for the relevant observations and actions before loading them. Since RLDS shards entire trajectories together, this is significantly more efficient than filtering the datasets after loading. To determine the correct coordinate frame alignment, we visualize trajectories from each dataset and analyze how transformations in observations correspond to changes in each dimension of the actions. Then, after loading each dataset, we map each action to its dataset-specific coordinate frame transformation.

B. Diffusion Policy Head Details

In order to model multimodal action distributions that may occur in manipulation and navigation datasets, we parameterize our action head with a diffusion policy [81]. To obtain an action, the model first samples a noisy action a_t^K of size (b, n, 7) from a Gaussian distribution $\mathcal{N}(0, \sigma^2 I)$. b denotes the batch size, n denotes the number of actions to predict into the future, and 7 denotes the action dim. Then, the model iteratively denoises the action using the following update rule:

$$a_t^{k-1} = \alpha(a_t^k - \gamma\epsilon_\theta(c_t, a_t^k, k) + \mathcal{N}(0, \sigma^2 I))$$

The noise prediction network is parameterized by a U-Net similar to the one described in Janner et al. [87]. The U-Net consists of k downsampling blocks followed by k upsampling blocks with residual connections. The noisy action is first transposed into size (b, 7, n). Each layer of the U-Net first processes the context via an MLP and the action via a temporal convolution. Then, after adding these embeddings together, it applied a group norm layer and Mish nonlinearity to the resulting vector. This vector is then passed through another temporal convolution layer, group norm layer, and Mish nonlinearity. Finally, the action is transposed back into its original shape of (b, n, 7). The channel sizes of each downsampling block can be found in Table VI.

C. GNM Navigation Data

The following table describes the datasets inside GNM that we used to train our policy, as well as their environment and split inside the dataset mixture. Further information can be found in the GNM paper [5]. Each of these datasets its weighted proportionally to its size.

Datasets	Environment	Split
SACSoN [78]	Indoors	7%
GO Stanford [78]	Indoors	9.8%
SCAND [46]	Outdoors, sidewalk	6.3%
RECON [74]	Outdoors, off-road	17.6%
Cory Hall [75]	Indoors	1.4%
Seattle [76]	Outdoors, trail	0.7%
Tartan Drive [77]	Outdoors, trail	3.5%

TABLE V: GNM Splits

D. Navigation Details

In this section, we provide further details on how we evaluate our policy on a mobile robot. Firstly, we move the robot around the environment and record a video at a $4hz$ frequency. The video is then converted into a topological map of goals. This map contains images that the navigation policy may potentially reach when navigating in the environment. During evaluation time, the agent first estimates the temporal distance of all the images in the topological map with respect to its current location. The robot then chooses a goal that is close to its current state, but also progresses towards its final destination. This is done by finding the image that minimizes the temporal distance prediction, and then choosing a goal image that is closer to its destination, but not too far. Finally, the robot uses the output of the policy as a waypoint. The waypoint is converted into linear and angular velocities, and then passed as commands to the robot's servos.

E. Hyperparameters

We train our policy with the following hyperparameters on a single 48G gpu for around 48 gpu-hours.

F. Egocentric Cotraining Ablation

We compare the performance of our method with and without co-training with third person viewpoints. Our results show a drop in performance when training with only wrist observations. This gap is more pronounced in the Toy Kitchen and Cluttered Grasp Novel settings. We hypothesize that this is due to the RT-X dataset mixture not containing many observations from wrist-mounted cameras. For example, less than 10% of the BRIDGE dataset [43] contains wrist

Fig. 9: Navigation and Manipulation Tasks. We evaluate our policy's manipulation performance on the two-object grasp, cluttered grasp, toy kitchen, cluttered grasp novel, and shelf manipulation tasks (from left to right). To evaluate our policy's navigation performance, we run our policies on the LoCoBot, Unitree Go-1 and ClearPath Jackal (from left to right).

Hyperparameter	Value
Batch Size	256
Learning Rate	1e-4
Learning Rate Scheduler	Cosine
Observation Encoder	EfficientNet-b5
Transformer Attention Heads	8
Transformer Attention Layers	8
Transformer Hidden Dim	2048
Diffusion Iterations	10
Diffusion UNet Dims	128, 256, 512, 1024

TABLE VI: Hyperparameters

camera observations. As a result, while the performance gap for environments that were seen in the datasets with wrist camera observations is small, the gap for novel environments is greater. We believe that this gap will reduce in the future as more large-scale datasets with egocentric camera are released.

Datasets	Cluttered Grasp	Cluttered Grasp Novel	Toy Kitchen
Wrist + Third	80%	55%	80%
Wrist-only	75%	40%	25%

TABLE VII: Egocentric Manipulation with Third-person Co-training. Policies co-trained with third-person observations have a 42% higher success rate on average than wrist-only policies.

G. Evaluation Tasks

Figure 9 depicts the navigation and manipulation tasks used for evaluation. For manipulation, we evaluate our policies on the two-object grasp, cluttered grasp, toy kitchen, cluttered grasp novel, and shelf manipulation tasks (from left to right). For each environment, we roll out the policy 20 times to account for higher variances in success when interacting with multiple objects. For navigation, we evaluate our policies on the LoCoBot, Unitree Go-1 and ClearPath Jackal (from left to right). For each embodiment, we average the success rate over 2 different environments: the office lobby and the kitchen.

H. Additional Discussion

1) Scaling Experiments: Tables XII and XIII show results for models with different numbers of parameters on the manipulation and navigation evaluation tasks. Interestingly, we see a clear positive correlation between model capacity and performance. The 27M param model achieves an average of 33% success on manipulation and 44% on navigation. Meanwhile, the $186M$ param model achieves averages of 68% and 55% success on manipulation and navigation respectively. We hypothesize that the visual diversity across manipulation and navigation contributes to this trend. We record the hyperparameters for the policy architectures we used sorted by parameter count in Table IX. Encoder represents the size of the EfficientNet image encoder.

Datasets	Two-Object Grasp	Cluttered Grasp	Novel Cluttered Grasp	Toy Kitchen	Shelf Manipulation
Ours + OXE (0%)	30%	30%	10%	0%	25%
Ours + OXE (33%)	50%	45%	15%	40%	30%
Ours + OXE (66%)	65%	65%	20%	55%	30%
Ours + OXE (100%)	70%	65%	20%	70%	30%
Ours + OXE (0%) + GNM	40%	45%	25%	0%	55%
Ours + OXE (33%) + GNM	65%	50%	35%	50%	60%
Ours + OXE (66%) + GNM	75%	70%	45%	70%	65%
Ours + OXE (100%) + GNM	80%	80%	50%	80%	65%

TABLE VIII: Impact of Manipulation Dataset Size vs Navigation Data In some environments such as Toy Kitchen, more OXE data helps more than navigation data. In OOD environments that requires spacial knowledge of the goal such as shelf manipulation, navigation data leads to more performance increases.

Datasets	U-Net dims	Layers	Encoder	Embedding Size
27M	128, 256, 512	4	b0	1024
56M	128, 256, 512, 1024	8,8,8	b0	1024
66M	128, 256, 512, 1024	8,8,8	b0	2048
112M	128, 256, 512, 1024	8,8,8	b0	2048
176M	128, 256, 512, 1024	8,8,8	b5	1024
186M	128, 256, 512, 1024	8,8,8	b5	2048

TABLE IX: Architectures by Parameter Count. We record the policies architectures we used for our parameter scaling experiments.

2) Discretization Experiments: Table XIV records the results of training with discretization on various manipulation tasks. Similar to the scheme proposed in RT-1 [65], we discretize each action dimension uniformly into 256 bins. For the model capacities we trained on, we find that our model performs poorly with this discretization scheme, with only the 180M parameter model having any success. Navigation results for discretization are not included in this table because they are flat and cause numerous collisions. We believe that a large model capacity is essential for models with this discretization scheme to perform reasonably. While we were unable to further scale our discretized models due to computation constraints, we believe that performance will increase if the number of parameters are appropriately scaled.

3) Impact of Manipulation Dataset Size vs Navigation Data: Table VIII records results for a OXE dataset size vs navigation data ablation. We first train policies on a mixture of our own dataset, $n\%$ of OXE where n varies from $0\%, 33\%, 66\%,$ and 100%. Then, we train policies on our own dataset, $n\%$ of OXE, and navigation data. Interestingly, in certain domains, using more manipulation data from OXE leads to greater performance increases than using navigation data. For instance, in the toy kitchen scenario, OXE data is essential for achieving meaningful performance because the policy lacks training exposure to kitchen setups beyond those included in OXE.

In addition, the results indicate that adding navigation data is beneficial to performance on certain tasks compared to OXE data even if the budget for navigation data is significantly lower than that of OXE data. Compared to the policy trained on the OURS + OXE (33%) datasets, OURS + OXE (0%) + GNM has better performance on the *Novel Cluttered Grasp* and *Shelf Manipulation* tasks(15%, 30% versus 25%, 55% respectively). Note that (33% of OXE contains on the order of one million trajectories, while GNM contains around 50 thousand trajectories. We conjecture that this improvement can be attributed to the fact that navigation data provides extra information that might not be present in OXE alone. For instance, navigation data includes details about spatial relationships between the current and goal images, particularly useful in tasks where the end-effector isn't directly aimed at a table or countertop—a scenario not extensively covered in OXE.

Datasets	Two-Object Grasp	Cluttered Grasp	Novel Cluttered Grasp	Toy Kitchen	Shelf Manipulation
GNM + M3ED + Driving + Manip	85%	80%	55%	80%	60%
GNM + Driving + Manip	80%	80%	50%	80%	65%
GNM + M3ED + Manip	80%	70%	55%	80%	50%
M3ED + Driving + Manip	65%	55%	25%	70%	35%
GNM + Manip	80%	75%	50%	65%	50%
Manip-only	70%	65%	20%	70%	30%

TABLE X: Manipulation Evaluations. By aligning action coordinate frames, training on navigation and driving datasets results in a 19% improvement across five challenging tabletop manipulation tasks.

	Locobot			DJI Tello*			Unitree Go-1		Clearpath Jackal	
Datasets	**OB**	**K**	**OL**	**CR**	**K**	**OL**	**K**	**OL**	**K**	**OL**
GNM-only	77 ± 40	70 ± 26	70 ± 0	95 ± 7	80 ± 0	65 ± 7	53 ± 4	40 ± 0	88 ± 18	78 ± 4
GNM + Driving	60 ± 36	63 ± 25	87 ± 3	83 ± 4	50 ± 28	38 ± 11	35 ± 0	30 ± 0	60 ± 28	60 ± 14
GNM + Manip	97 ± 6	87 ± 23	83 ± 6	83 ± 11	95 ± 8	80 ± 28	50 ± 14	77 ± 4	100 ± 0	80 ± 0
GNM + Driving + Manip	77 ± 40	63 ± 35	90 ± 0	95 ± 7	40 ± 0	10 ± 14	$50n \pm 14$	45 ± 7	100 ± 0	90 ± 14

TABLE XI: Navigation Evaluations. Across three different robots in challenging indoor and outdoor environments, adding manipulation datasets leads to $5-7\%$ improvement in navigation performance. The values in the table are the percent success or the completed portion of a given trajectory before failure. The abbreviations of the environments are Office Lobby (OL), Office Bay (OB), Kitchen (K), and Conference Room (CR). *Note that we only use the Conference room environment for the results of the drone evaluations due to variance across environments.

Datasets	Two-Object Grasp	Cluttered Grasp	Cluttered Grasp OOD	Toy Kitchen	Cluttered Grasp
27M Params	50%	45%	20%	40%	10%
56M Params	55%	45%	15%	40%	20%
66M Params	75%	55%	20%	70%	45%
112M Params	85%	60%	45%	70%	40%
186M Params	80%	80%	50%	80%	50%

TABLE XII: Manipulation Scaling Policies with 186M parameters have a 15% higher success rate on average than policies with 27M parameters.

	Locobot		DJI Tello	Unitree Go-1		Jackal	
Num. Parameters	**K**	**OL**	**CR**	**K**	**OL**	**K**	**OL**
27M Params	95 ± 5	68 ± 34	80 ± 0	20 ± 14	30 ± 0	43 ± 3	63 ± 4
56M Params	53 ± 12	77 ± 15	68 ± 11	20 ± 14	25 ± 7	50 ± 0	65 ± 0
66M Params	50 ± 0	87 ± 6	85 ± 7	40 ± 0	30 ± 0	73 ± 11	53 ± 18
112M Params	85 ± 22	90 ± 0	65 ± 7	73 ± 39	58 ± 32	73 ± 11	85 ± 21
176M Params	73 ± 23	83 ± 12	90 ± 14	53 ± 10	40 ± 0	75 ± 7	75 ± 7
186M Params	63 ± 35	90 ± 0	95 ± 7	50 ± 14	45 ± 7	100 ± 0	90 ± 14

TABLE XIII: Navigation Scaling. Policies with 186M parameters have a 20% higher success rate on average than policies with 27M parameters on manipulation tasks. The values in the table are the percent success or the completed portion of a given trajectory before failure. The abbreviations of the environments are Office Lobby (OL), Kitchen (K), and Conference Room (CR).

Datasets	Two-Object Grasp	Cluttered Grasp	Cluttered Grasp OOD
40M	0%	0%	0%
100M	0%	0%	0%
180M	10%	5%	5%

TABLE XIV: Discretization Experiments. Policies trained with a discretization head and cross-entropy loss under 100M parameters fail to have nonzero success.

Datasets	Two-Object Grasp	Cluttered Grasp	Novel Cluttered Grasp	Toy Kitchen
Polybot + RT-X + Sacson	80%	70%	35%	70%
Polybot + RT-X + Go Stanford	65%	6%	35%	75%
Polybot + RT-X + Tartan	25%	10%	10%	0%
Polybot + RT-X + Recon	50%	50%	20%	60%
Polybot + RT-X + Cory Hall	55%	5%	20%	70%
Polybot + RT-X + Seattle	20%	5%	10%	10%
Polybot + RT-X + Scand	55%	50%	20%	45%

TABLE XV: GNM Ablations. Manipulation policies co-trained with indoor and outdoor navigation data on sidewalks perform better than policies co-trained on outdoor navigation data in off-road or trail-like environments.