$\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, Ury Zhilinsky
Physical Intelligence
research@physicalintelligence.company

Abstract: In order for robots to be useful, they must perform practically relevant tasks in the real world, outside of the lab. While vision-language-action (VLA) models have demonstrated impressive results for end-to-end robot control, it remains an open question how far such models can generalize in the wild. We describe $\pi_{0.5}$, a new model based on π_0 that uses co-training on heterogeneous tasks to enable broad generalization. $\pi_{0.5}$ uses data from multiple robots, high-level semantic prediction, web data, and other sources to enable broadly generalizable real-world robotic manipulation. Our system uses a combination of co-training and hybrid multi-modal examples that combine image observations, language commands, object detections, semantic subtask prediction, and low-level actions. Our experiments show that this kind of knowledge transfer is essential for effective generalization, and we demonstrate for the first time that an end-to-end learning-enabled robotic system can perform long-horizon and dexterous manipulation skills, such as cleaning a kitchen or bedroom, in entirely new homes.

Website: www.pi.website/blog/pi05

Keywords: VLA, Generalization, Mobile Manipulator

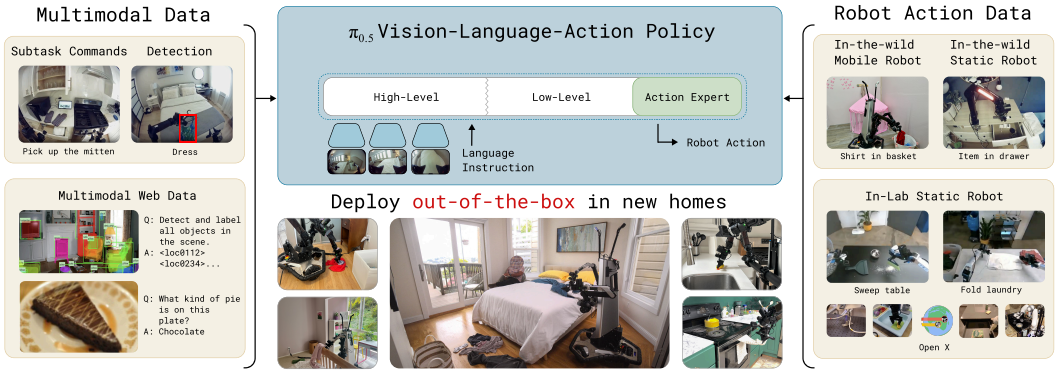


Figure 1: The $\pi_{0.5}$ model transfers knowledge from a heterogeneous range of data sources, including other robots, high-level subtask prediction, verbal instructions, and data from the web, to enable broad generalization across environments and objects. $\pi_{0.5}$ can control a mobile manipulator to clean kitchens and bedrooms in new homes that were not present in the training data, performing complex behaviors with durations of 10+ minutes.

1 Introduction

Open-world generalization represents one of the biggest open problems in physical intelligence, and scalable learning systems offer a path to enable such generalization, as they have in domains ranging from natural language processing [1, 2, 3, 4] to computer vision [5, 6, 7, 8]. However, the diversity of situations that a robot might encounter in the real world requires more than just scale: we need to

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

Figure 2: $\pi_{0.5}$ **cleaning a new kitchen.** The robot is tasked with cleaning a kitchen in a home that was not in the training data. Given general tasks (close the cabinets, put the items in the drawer, wipe the spill, and put the dishes in the sink), the model predicts subtasks (e.g., pick up the plate) and emits low-level actions.

design training recipes that can provide the breadth of knowledge that will allow robots to generalize at many levels of abstraction, from physical behaviors (e.g., how to pick up a new plate) to scene semantics (e.g., which object on the counter is most likely to be a drying rack). How can we structure a training recipe for a robotic learning system that can enable this kind of flexible generalization?

A person can draw on a lifetime of experience to synthesize appropriate solutions to each of these challenges. Analogously, generalizable robotic learning systems must be able to transfer experience and knowledge from a variety of information sources, from firsthand experience on relevant tasks to other robots and even other data types, such as verbal instructions, perceptual tasks based on web data, or prediction of high-level semantic commands. The heterogeneity of these different sources of data present a major obstacle, but recent advances in vision-language-action (VLA) models provide us with a toolkit that can make this possible: by casting different modalities into the same sequence modeling framework, VLAs can be adapted to train on robot data, language data, computer vision tasks, and combinations of the above.

We leverage this observation to design a co-training framework for VLAs that can utilize heterogeneous and diverse knowledge sources to enable broad generalization, creating the $\pi_{0.5}$ model ("pi oh five"), which can control mobile manipulators to perform a variety of household tasks even in homes that were never seen during training. $\pi_{0.5}$ draws on experience from many sources: in addition to a medium-sized dataset collected directly with mobile manipulators in a variety of real homes (about 400 hours), $\pi_{0.5}$ uses data from other non-mobile robots, data of related tasks collected under laboratory conditions, training examples that require predicting "high-level" semantic tasks based on robot observation, verbal language instructions provided to the robot by human supervisors, and a variety of multi-modal examples created from web data (see Figure 1). The overwhelming majority of training examples provided to $\pi_{0.5}$ (97.6% during the first training phase) do not come from mobile manipulators performing household tasks. $\pi_{0.5}$ is able to control mobile manipulators in entirely new homes not seen during training, perform intricate tasks such as making beds, and perform complex multi-stage tasks, like putting all the dishes into a sink.

Our central contribution is a system for training a highly generalizable VLA, $\pi_{0.5}$, together with a proof of concept that generalization can emerge from this model when it is trained on appropriately diverse data. $\pi_{0.5}$ follows a simple hierarchical architecture: we first pre-train the model on the heterogeneous mixture of training tasks, and then fine-tune it specifically for mobile manipulation with both low-level action examples and high-level "semantic" actions, which correspond to predicting subtask labels such as "pick up the cutting board" or "rearrange the pillow." At runtime, during each step of inference, the model first predicts the semantic subtask, inferring the behavior that is appropriate to perform next based on the task structure and the semantics of the scene, and then predicts the robot action chunk based on this subtask. This provides the model with an ability to reason through long horizon behaviors and leverage different sources of knowledge at both the low and high level. We provide a detailed empirical evaluation of both $\pi_{0.5}$'s generalization capabilities and the relevance of different co-training ingredients. To our knowledge, our work is the first to demonstrate an end-to-end learning-enabled robotic system that can perform long-horizon and dexterous manipulation skills, such as cleaning a kitchen or bedroom, in entirely new homes. Our experiments and comparisons further show that this is enabled by transferring knowledge from other robots, high-level semantic prediction, verbal language instruction from human supervisors, web data, and other sources.

2 Related Work

Generalist robot manipulation policies. Recent works have demonstrated that broadening the training data distribution for robot manipulation policies allows the resulting policies to not only solve a wider range of tasks out of the box, but also improves their ability to generalize [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 12, 19, 20]. Vision-language-action models (VLAs) [21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35] offer an appealing solution to continue scaling to *generalist* policies: by fine-tuning pre-trained vision-language models for robot control, VLAs can leverage the semantic knowledge acquired from web-scale pretraining and bring it to bear on the robotics problem. When combined with highly expressive action decoding mechanisms like flow matching [24], diffusion [27, 33, 36], or advanced action tokenization schemes [31], VLAs can perform a wide range of complex manipulation tasks. While some studies suggest that simple skills like picking up objects generalize simply by collecting robot data in a broader set of environments [16, 31, 37, 38, 39], it is challenging to apply the same approach to more complex, long-horizon tasks like cleaning up a kitchen, where covering all plausible scenarios with brute-force scaling of robot data collection is infeasible. In our experiments, we present an architecture and training procedure, $\pi_{0.5}$, that can generalize to entirely new scenes by leveraging not only direct experience on the target mobile manipulator platform, but also information from other data sources.

Non-robot data co-training. A number of prior works have sought to use diverse *non-robot* data to improve the generalization of robot policies through initializing vision encoders from computer vision datasets [40, 41, 42, 43], leveraging off-the-shelf task planners [44, 45, 46, 47], or training VLA policies initialized from a pre-trained vision-language model [21, 22, 23]. Prior works have demonstrated that co-training VLAs with data mixtures used for VLM training [21, 22, 48] can improve generalization, e.g., when interacting with new objects or unseen scene backgrounds. In this work, we go beyond VLM data co-training and design a system for co-training VLAs with a broader set of robotics-relevant supervision sources, including data from other robots, high-level semantic subtask predictions, and verbal language instructions. While multitask training and co-training are not new ideas, we show that the specific combination of data sources in our system enables mobile robots to perform complex and long-horizon behaviors in entirely new environments.

Robot reasoning and planning with language. A number of prior works have shown that augmenting end-to-end policies with high-level reasoning can improve performance [49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 32, 67], particularly when high-level subtask inference can benefit from large pre-trained LLMs and VLMs. Our method also uses a two-stage inference procedure, where we first infer a high-level semantic subtask (e.g., "pick up the plate"), and then predict the action based on this subtask. Many prior methods have employed two separate models for this purpose, with a VLM predicting semantic steps and a separate low-level policy executing those steps [49, 53, 61, 62, 65, 66, 68]. Our method uses the same model for both high-level and low-level inference, in a recipe that more closely resembles chain-of-thought [69] or test-time compute [70] methods, though unlike embodied chain-of-thought methods [58, 71, 72], the high-level inference process still runs at a lower frequency than low-level action inference.

Robotic learning systems with open-world generalization. While most robotic learning systems are evaluated in environments that closely match the training data, a number of prior works have explored broader open-world generalization. When the robot's tasks are restricted to a more narrow set of basic primitives, such as picking up objects, methods that allow for task-specific assumptions (e.g., grasp prediction, or incorporating model-based planning and control) have been shown to generalize broadly, even to entirely new homes [73, 74, 75, 76, 77]. However, such methods do not readily generalize to the full range of possible tasks that a generalist robot might need to perform. More recently, large-scale datasets collected across many domains [12, 13, 16, 37, 39, 78] have been shown to enable generalization of simple but end-to-end learned tasks to new environments [16, 31, 38, 39, 79, 80, 81, 82]. However, the tasks in these demonstrations are still relatively simple, typically less than a minute in length and often with relatively low success rates. We show that $\pi_{0.5}$ can perform long, multi-stage tasks, such as putting all of the dishes in the sink or picking all of the clothing off the floor of a new bedroom, while generalizing to entirely new homes.

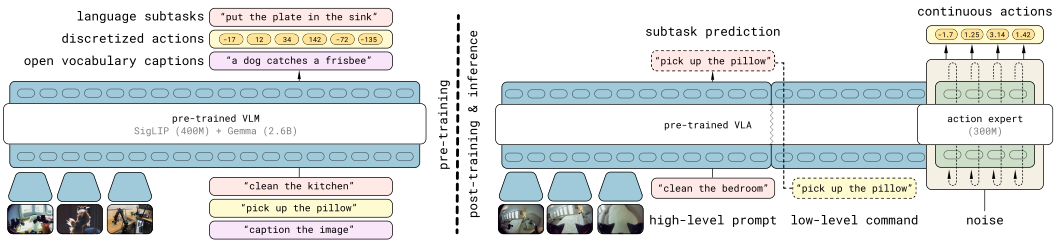


Figure 3: **Model overview.** $\pi_{0.5}$ is trained in two stages. First, a pre-training stage combines all of the different data sources to produce an initial VLA with discrete tokens. This stage uses data from diverse robotic platforms, high-level semantic action prediction, and data from the web. Robotic data uses the FAST action tokenizer to represent actions as discrete tokens [31]. Second, a post-training stage specializes the model for low-level and high-level inferences for mobile manipulation, leveraging the most task-relevant data, including verbal instructions from human supervisors. This stage uses flow matching to represent the action distribution, enabling efficient real-time inference and the ability to represent fine-grained continuous action sequences. At inference time, the model first infers a high-level subtask, and then predicts the actions based on this subtask.

3 The $\pi_{0.5}$ Model and Training Recipe

We provide an overview of the $\pi_{0.5}$ model and training recipe in Figure 3 and additional preliminaries in Section A. The model weights are initialized from a standard VLM trained on web data, and training then proceeds in two stages: a pre-training stage intended to adapt the model to diverse robotic tasks, and a post-training stage intended to specialize it to mobile manipulation and equip it with the mechanisms for efficient test-time inference. During pre-training, all tasks, including tasks with robot actions, are represented with discrete tokens, which leads to scalable, and efficient training [31]. During post-training, we adapt the model to also have an action expert, as with π_0, in order to both represent actions with finer granularity and enable more compute-efficient inference for real-time control. At inference-time, the model first produces a high-level subtask for the robot to perform and then, conditioned on this subtask, predicts the low-level actions via the action expert.

3.1 The $\pi_{0.5}$ architecture

The $\pi_{0.5}$ architecture can flexibly represent both action chunk distributions and tokenized text outputs, with the latter used both for co-training tasks (e.g., question-answering) and for outputting high-level subtask predictions during hierarchical inference. The distribution captured by the model can be written as $\pi_\theta(\mathbf{a}_{t:t+H}, \hat{\ell} | \mathbf{o}_t, \ell)$, where consists of the images from all of the cameras and the robot's configuration (joint angles, gripper pose, torso lift pose, and base velocity), ℓ is the overall task prompt (e.g., "put away the dishes"), $\hat{\ell}$ represents the model's textual output, which could be either a predicted high-level subtask (e.g., "pick up the plate") or the answer to a vision-language prompt in web data, and $\mathbf{a}_{t:t+H}$ is a predicted action chunk. We decompose the distribution as

$$\pi_\theta(\mathbf{a}_{t:t+H}, \hat{\ell} | \mathbf{o}_t, \ell) = \pi_\theta(\mathbf{a}_{t:t+H} | \mathbf{o}_t, \hat{\ell}) \pi_\theta(\hat{\ell} | \mathbf{o}_t, \ell),$$

where the action distribution does not depend on ℓ, only on $\hat{\ell}$. Thus, high-level inference captures $\pi_\theta(\hat{\ell} | \mathbf{o}_t, \ell)$, and low-level inference captures $\pi_\theta(\mathbf{a}_{t:t+H} | \mathbf{o}_t, \hat{\ell})$. The model corresponds to a transformer that takes in N multimodal input tokens $x_{1:N}$ (we use the term token loosely here, referring to both discretized and continuous inputs) and produces a sequence of multimodal outputs $y_{1:N}$, which we can write as $y_{1:N} = f\big(x_{1:N}, A(x_{1:N}), \rho(x_{1:N})\big)$. Each x_i can be a text token ($x_i^w \in \mathbb{N}$), an image patch ($x_i^I \in \mathbb{R}^{p \times p \times 3}$), or an intermediate denoising value of a robot action in flow matching ($x_i^a \in \mathbb{R}^d$). The observations $\mathbf{o}_t$ and ℓ form the prefix part of $x_{1:N}$. Depending on the token type, as indicated by $\rho(x_i)$, each token can be processed not only by a different encoder, but also by different expert weights within the transformer. For example, image patches are fed through a vision encoder, and text tokens are embedded with an embedding matrix. Following π_0 [24], we linearly project action tokens x_i^a into the transformer embedding space and use separate expert weights in the transformer to process the action tokens. The attention matrix $A(x_{1:N}) \in [0, 1]^{N \times N}$ indicates if a token can attend to another token. As we want our model to output both text (to answer questions about the scene or to output next tasks to accomplish) and actions (to act in the world), the output

of f is split into text token logits and action output tokens, respectively $\left(y^{\ell}_{1:M}, y^{a}_{1:H}\right)$. The first M correspond to text token logits that can be used to sample $\hat{\ell}$ and the later H tokens are produced by a separate action expert, as in π_0, and projected via a linear mapping to continuous outputs used to obtain $\mathbf{a}_{t:t+H}$ (see next section). Note that $M + H \leq N$, i.e., not all outputs are associated with a loss. More details about the architecture are in Appendix B.1.

3.2 Combining discrete & continuous action representations

Similarly to π_0, we use flow-matching [83] to predict continuous actions in the final model. Given $\mathbf{a}^{\tau,\omega}_{t:t+H} = \tau \mathbf{a}_{t:t+H} + (1-\tau)\omega$, $\omega \sim \mathcal{N}(0, \mathbf{I})$, where $\tau \in [0,1]$ is the flow matching time index, the model is trained to predict the flow vector field $\omega - \mathbf{a}_t$. However, as shown in [31], VLA training can be much faster when actions are represented by discrete tokens. Unfortunately, such discrete representations are less well-suited for real-time inference, because they require expensive autoregressive decoding for inference [31]. Therefore, an ideal model design would train on discretized actions but still allow for use of flow matching to produce continuous actions at inference time.

Our model is therefore trained to predict actions *both* through autoregressive sampling of tokens (using the FAST tokenizer) and iterative integration of the flow field, combining the best of both worlds. We use the attention matrix to ensure that the different action representations do not attend to each other. Our model is optimized to minimize the combined loss

$$\mathbb{E}_{\mathcal{D},\tau,\omega}\Big[H\big(x_{1:M}, f^{\ell}_{\theta}(\mathbf{o}_t, \ell)\big) + \alpha \left\|\omega - \mathbf{a}_{t:t+H} - f^{a}_{\theta}(\mathbf{a}^{\tau,\omega}_{t:t+H}, \mathbf{o}_t, \ell)\right\|^2\Big], \tag{1}$$

where $H(x_{1:M}, y^{\ell}_{1:M})$ is the cross entropy loss between the text tokens and predicted logits (including the FAST encoded action tokens), $y^{a}_{1:H} = f^{a}_{\theta}(\mathbf{a}^{\tau,\omega}_{t:t+H}, \mathbf{o}_t, \ell)$ is the output from the (smaller) action expert, and $\alpha \in \mathbb{R}$ is a trade-off parameter. This scheme enables us to first pre-train our model as a standard VLM transformer model by mapping actions to text tokens ($\alpha = 0$), and then add additional action expert weights predicting continuous action tokens in a non-autoregressive fashion for fast inference in a post-training stage. At inference time we then use autoregressive decoding for text tokens $\hat{\ell}$ followed by 10 denoising steps, conditioned on text tokens, to produce actions $\mathbf{a}_{t:t+H}$.

3.3 Training

Our training proceeds in two stages, with additional detail in Appendix Section B and Figure 12.

Pre-training. $\pi_{0.5}$ is trained as a standard auto-regressive transformer, performing next-token prediction of text, object locations, and FAST encoded action tokens. This stage includes data from Mobile Manipulators **(MM)** performing household tasks in about 100 different home environments, Multi-Environment **(ME)** non-mobile robots performing similar tasks in home environments, Cross-Embodiment **(CE)** data collected in a laboratory setting for a wide range of tasks (e.g., bussing a table, folding shirts), High-Level **(HL)** bounding box and subtask prediction data breaking down high-level task commands such as "clean the bedroom" into shorter subtasks like "pick up pillow", and finally **(WD)** multi-modal Web Data (including image captioning, VQA, and object location tasks). For all action data, we train the model to predict target joint and end-effector poses. We set the dimensionality of the action $\mathbf{a}$ to a fixed number to accommodate the largest action space among all the datasets. For robots with lowntirelyr-dimensional configuration and action spaces, we zero-pad the action vectors.

Post-training. After pre-training the model with discrete tokens for 280k gradient steps, we perform a second stage. The purpose of this stage is to both specialize the model to our use-case (mobile manipulation in homes), and to add an action expert that can produce continuous action chunks via flow matching. This stage jointly trains with next-token prediction, to preserve text prediction capabilities, and flow matching for the action expert (which is initialized with random weights at the beginning of post-training). We optimize the objective in Equation (1), with $\alpha = 10.0$ for 80k additional steps. The post-training action dataset consists of the MM and ME robot data, filtered down to successful episodes that are below a fixed length threshold, web data (WD), and the

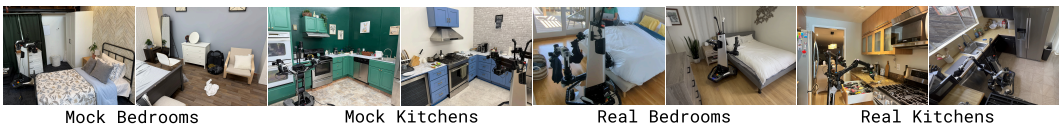


Figure 4: **Evaluation environments.** We evaluate $\pi_{0.5}$ in entirely new kitchens and bedrooms that were not seen during training, with novel objects, backgrounds, and layouts. We use a set of mock rooms for controlled, reproducible quantitative comparisons (left) and real homes for a realistic final evaluation (right).

multi-environment slice of HL data. Additionally, to improve the model's ability to predict appropriate high-level subtasks, we collect *verbal instruction* demonstrations (VI), which are constructed by expert users providing "language demonstrations," selecting appropriate sub-task commands to command the robot to perform mobile manipulation tasks step by step. These examples are collected by "teleoperating" the robot in real time with language to perform tasks with the learned low level policy, providing demonstrations of good high-level subtask outputs for a trained policy.

4 Experimental Evaluation

The $\pi_{0.5}$ model is designed to generalize broadly to new environments. While it is common to evaluate VLAs in environments that match the training data, we conduct all of our experiments in novel environments that were not seen in training. For quantitative comparisons, we use a set of mock home environments to provide a controlled and reproducible setup, while the most realistic final evaluation is conducted in three real homes that were not part of the training set (see Figure 4). The two mobile manipulators used in our experiments are illustrated in Figure 13 (see Section B.4 for more details). Our experiments explore $\pi_{0.5}$'s generalization abilities (4.1), environment scaling (4.2), low-level data (4.3) and model (4.4) ablations, and high-level ablations (D.1).

4.1 Can $\pi_{0.5}$ generalize to real homes?

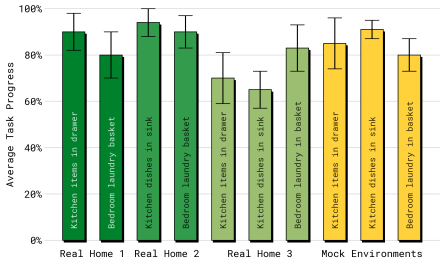
We evaluated $\pi_{0.5}$ in three real homes that were not present in the training set, using both types of robots. In each of the homes, the robots were instructed to perform bedroom and kitchen cleaning tasks. The evaluation rubrics for each task are provided in Appendix C and roughly correspond to the percentage of steps in each task that were completed successfully (e.g., placing half the dishes in the sink corresponds to around 50%). The results in Figure 5 show that $\pi_{0.5}$ was able to consistently succeed on a variety of tasks in each home (we note that, additionally, the model is capable of performing many more tasks than used in our quantitative evaluation). Many of the tasks involve multiple stages (e.g., moving multiple objects) lasting about 2 to 5 minutes. For these trials, the model is provided with a simple high-level command (e.g., "place the dishes in the sink"), and the high-level inference process autonomously determines appropriate steps (e.g., "pick up the cup"). This level of in-the-wild generalization goes significantly beyond the results demonstrated with prior vision-language-action models, both in terms of the degree of novelty that the model must handle, and the task duration and complexity.

4.2 How does generalization scale with the number of scenes?

We aim to measure how generalization scales with the number of environments. Since applying the entire pre-training and post-training recipe to each of these datasets is prohibitively compute-intensive, for these experiments we pre-train on the mixture of robot action prediction data *without* mobile manipulation data, and then compare models post-trained on datasets that comprise mobile manipulation data from varying numbers of environments (3, 12, 22, 53, 82, and 104). While the datasets split by location in principle differ in size, in practice the number of training steps (40k) is chosen such that each model sees the same number of unique data samples, which allows us to control for dataset size when varying the number of locations used within a post-training experiment.

Each model is evaluated in the mock environments shown in Figure 4, which are not seen in training. We conduct two types of evaluations: (1) overall performance on multi-stage tasks using the

(a) **Example rollouts.** We visualize an exemplary $\pi_{0.5}$ episode for one task from each home. Top to bottom: putting items in a drawer in Home 1, followed by putting dishes in the sink in Home 2, and putting clothes in the laundry basket in Home 3. The human instruction for each is given on the left, and the high-level subtask prediction from $\pi_{0.5}$ is shown beneath each frame in blue.



(b) **Quantitative evaluation.** We show the task progress per task and environment averaged over 10 trials. We find that $\pi_{0.5}$'s performance in the mock evaluation setups is representative of its performance in real homes.

Figure 5: **Evaluation in real homes.** We evaluated $\pi_{0.5}$ in three kitchens and three bedrooms in real homes that were not seen during training. We evaluate the tasks 'items in drawer', 'laundry basket', and 'dishes in sink,' and find $\pi_{0.5}$ to be successful at these tasks in these completely new, real homes.

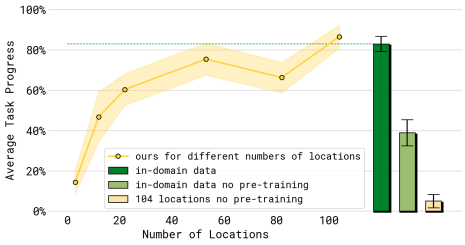


Figure 6: **Evaluating performance with different numbers of locations.** Performance over the four test tasks — "dishes in sink", "items in drawer", "laundry basket", "make bed" — improves with more training environments. The green bar shows a baseline model with the test homes in the training set. Compared to this model, our best model achieves similar performance, without any data from the test homes.

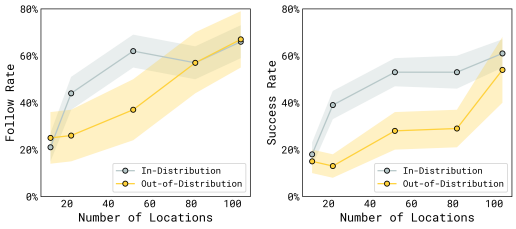


Figure 7: **Evaluating language following with different numbers of training locations.** We evaluate language following rate and success rate for picking up user-indicated items and placing them into drawers or sinks, averaged over seen object categories ("in-distribution") or unseen categories ("out-of-distribution"). Performance increases steadily as we increase the number of training locations.

standard rubric in Appendix C and (2) a more fine-grained evaluation of each model's ability to follow language instructions and interact with novel objects, where the robot must pick up specific in- and out-of-distribution objects from a kitchen counter based on language commands. Details of this experiment are discussed in Appendix D.

The results of the first experiment are shown in Figure 6. The average performance among the tasks generally improves with more training locations and achieves parity with a control (shown in green) trained directly on data from the test home. This suggests that our co-training recipe effectively enables broad generalization. To confirm that this generalization performance requires our full co-training recipe, we additionally include two baselines that *do not* use any of the other co-training tasks in the pre-training phase, but instead train directly on either data from the test environment (light green) or mobile manipulation data from the 104 training locations (light yellow). The performance for both those baselines is significantly worse.

The results of the second experiment are shown in Figure 7. For seen and unseen objects, we report the language following rate, which measures how often the robot selects the object indicated in the language command, and success rate, which measures how often the robot successfully places that object in the correct location. As the number of locations in the training data increases, both language following performance and success rate improve. As expected, the performance on in-distribution objects improves more quickly than that of out-of-distribution objects.

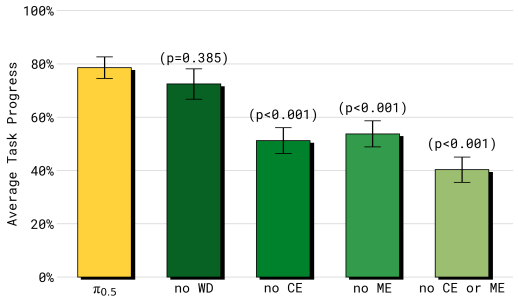


Figure 8: **Training recipe ablations.** We ablate parts of the training mixture on four test tasks (10 trials per task). Including cross-embodiment data, both in diverse environments (ME) and for diverse tasks in lab settings (CE) is important for good performance. Web data (WD) does not make a significant difference, but we will see in Figures 9, 16 that it impacts object generalization and high-level performance.

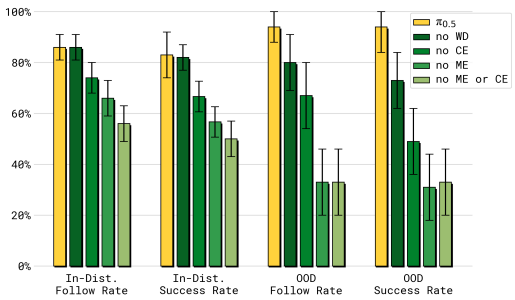


Figure 9: **Training recipe ablations.** We evaluate language following with in-distribution (ID) and out-of-distribution (OOD) objects. Including web data (WD) is important for OOD performance, while cross-embodiment (CE) and diverse environment (ME) data have a large impact on both ID and OOD performance. We note the tasks are different for ID and OOD and thus only comparable within categories.

4.3 How important is each part of our co-training recipe?

We compare our full $\pi_{0.5}$ model to other training mixtures. The task performance results on home tasks are shown in Figure 8 (detailed breakdown in Appendix E) and language following in Figure 9. For both experiments we see in the results that excluding *either* of the two cross-embodiment data sources significantly degrades performance, indicating that $\pi_{0.5}$ benefits considerably from cross-embodiment transfer, from both other environments (ME) and other tasks (CE). Excluding both sources harms performance even more. Interestingly, the difference in performance with the **no WD** ablation is not statistically significant for the performance experiment, though we see an impact in OOD language following and for high-level subtask inference (Sec. D.1).

4.4 How does $\pi_{0.5}$ compare to other VLAs?

We compare $\pi_{0.5}$ to π_0 as well as an improved version of π_0 denoted as π_0-FAST+Flow, which is trained via the joint flow matching and FAST action prediction formulation from Equation (1), but on action data only, without the HL or WD datasets. These models provide a strong point of comparison, since π_0 has been demonstrated to perform strongly on complex, dexterous tasks, and the enhancement in π_0-FAST+Flow brings it as close to $\pi_{0.5}$ as possible. All models receive the same robot action data set and are trained for a comparable number of steps. Results in Figure 10 show that $\pi_{0.5}$ significantly outperforms π_0, π_0 trained for longer to 300k steps, and our enhanced version.

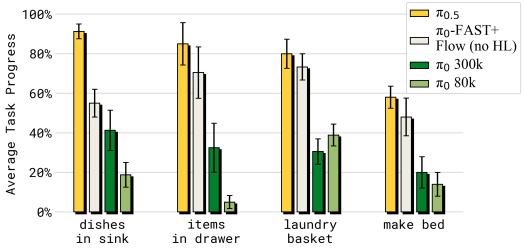


Figure 10: **Comparing $\pi_{0.5}$ with other models.** Our full model significantly outperforms both π_0 and π_0-FAST+Flow in the mock home test environments.

5 Conclusions

We described $\pi_{0.5}$, a co-trained model that builds on the π_0 VLA to integrate a variety of data sources and enable generalization to new environments. The $\pi_{0.5}$ VLA can control mobile manipulators to perform tasks in homes that were never seen in the training data, cleaning kitchens and bedrooms, making beds, hanging towels, and performing other multi-stage and dexterous behaviors. $\pi_{0.5}$ is trained on about 400 hours of mobile manipulation data, but includes a much larger amount of data from other robots, including non-mobile manipulators in diverse environments and data collected under laboratory conditions. It is also co-trained jointly with data from the web, as well as high-level prediction data for outputting language commands based on robot observations. The generalization capabilities of $\pi_{0.5}$ demonstrate that this co-training recipe facilitates effective transfer.

6 Limitations and Future Work

$\pi_{0.5}$ is not without its limitations. While our VLA exhibits broad generalization, it still makes mistakes. Some environments present persistent challenges (e.g., unfamiliar handles on drawers, or cabinets that are physically hard for the robot to open), some behaviors present challenges with partial observability (e.g., the robot arm occluding a spill that should be wiped), and in some cases the high-level sub-task inference is easily distracted (e.g., closing and opening a drawer multiple times while putting away items). Addressing these challenges with better co-training, transfer, and larger datasets is a promising direction for future work. Other future work directions could address the technical constraints of our method. While $\pi_{0.5}$ can perform a variety of behaviors to clean up kitchens and bedrooms, it processes relatively simple prompts. The complexity of the prompts that the model can accommodate is determined by the training data, and more complex preferences and instructions could be incorporated by producing more intricate and diverse annotations, either with human labelers or synthetically. The model also uses a relatively modest context, and incorporating richer context and memory could make the model significantly more capable in settings with more partial observability, such as tasks that require navigating between different rooms or remembering where objects are stored. More broadly, $\pi_{0.5}$ explores a particular combination of heterogeneous data sources, but the specific sources of data can be explored even more broadly. For instance, the ability of our system to learn from verbal instructions provides a powerful new supervision modality, and future work could explore this and other ways that people can provide robots with additional contextual knowledge. We hope that our work will serve as a foundation for a new generation of VLAs that exhibit broad generalization to diverse real-world environments.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.

[4] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[7] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15979–15988, 2022.

[8] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. arXiv preprint arXiv:2304.02643, 2023.

[9] 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. *CoRL*, 2019.

[10] 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. *arXiv preprint arXiv:2109.13396*, 2021.

[11] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. BridgeData v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[12] Open X-Embodiment Collaboration, A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, A. Raffin, A. Wahid, B. Burgess-Limerick, B. Kim, B. Schölkopf, B. Ichter, C. Lu, C. Xu, C. Finn, C. Xu, C. Chi, C. Huang, C. Chan, C. Pan, C. Fu, C. Devin, D. Driess, D. Pathak, D. Shah, D. Büchler, D. Kalashnikov, D. Sadigh, E. Johns, F. Ceola, F. Xia, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Schiavi, H. Su, H.-S. Fang, H. Shi, H. B. Amor, H. I. Christensen, H. Furuta, H. Walke, H. Fang, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Kim, J. Schneider, J. Hsu, J. Bohg, J. Bingham, J. Wu, J. Wu, J. Luo, J. Gu, J. Tan, J. Oh, J. Malik, J. Tompson, J. Yang, J. J. Lim, J. Silvério, J. Han, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Zhang, K. Majd, K. Rana, K. Srinivasan, L. Y. Chen, L. Pinto, L. Tan, L. Ott, L. Lee, M. Tomizuka, M. Du, M. Ahn, M. Zhang, M. Ding, M. K. Srirama, M. Sharma, M. J. Kim, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, P. R. Sanketi, P. Wohlhart, P. Xu, P. Sermanet, P. Sundaresan, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Martín-Martín, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Moore, S. Bahl, S. Dass, S. Song, S. Xu, S. Haldar, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Dasari, S. Belkhale, T. Osa, T. Harada, T. Matsushima, T. Xiao, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, V. Jain, V. Vanhoucke, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Wang, X. Zhu, X. Li, Y. Lu, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Xu, Y. Wang, Y. Bisk, Y. Cho, Y. Lee, Y. Cui, Y. hua Wu, Y. Tang, Y. Zhu, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Xu, and Z. J. Cui. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023.

[13] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, D. A. Herrera, M. Heo, K. Hsu, J. Hu, D. Jackson, C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, 2024.

[14] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4788–4795. IEEE, 2024.

[15] 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. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 653–660. IEEE, 2024.

[16] N. M. M. Shafiullah, A. Rai, H. Etukuru, Y. Liu, I. Misra, S. Chintala, and L. Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023.

[17] AgiBot-World-Contributors, Q. Bu, J. Cai, L. Chen, X. Cui, Y. Ding, S. Feng, S. Gao, X. He, X. Hu, X. Huang, S. Jiang, Y. Jiang, C. Jing, H. Li, J. Li, C. Liu, Y. Liu, Y. Lu, J. Luo, P. Luo, Y. Mu, Y. Niu, Y. Pan, J. Pang, Y. Qiao, G. Ren, C. Ruan, J. Shan, Y. Shen, C. Shi, M. Shi, M. Shi, C. Sima, J. Song, H. Wang, W. Wang, D. Wei, C. Xie, G. Xu, J. Yan, C. Yang, L. Yang, S. Yang, M. Yao, J. Zeng, C. Zhang, Q. Zhang, B. Zhao, C. Zhao, J. Zhao, and J. Zhu. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

[18] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

[19] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[20] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *Conference on Robot Learning*, 2024.

[21] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[22] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

[23] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[24] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[25] J. Wen, Y. Zhu, J. Li, M. Zhu, K. Wu, Z. Xu, N. Liu, R. Cheng, C. Shen, Y. Peng, F. Feng, and J. Tang. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2409.12514*, 2024.

[26] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan. 3d-vla: 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

[27] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

[28] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[29] S. Belkhale and D. Sadigh. Minivla: A better vla with a smaller footprint, 2024. URL https://github.com/Stanford-ILIAD/openvla-mini.

[30] A. Szot, B. Mazoure, O. Attia, A. Timofeev, H. Agrawal, D. Hjelm, Z. Gan, Z. Kira, and A. Toshev. From multimodal llms to generalist embodied agents: Methods and lessons. *arXiv preprint arXiv:2412.08442*, 2024.

[31] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. FAST: Efficient action tokenization for vision-language-action models. *Robotics: Science and Systems*, 2025.

[32] G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[33] J. Wen, Y. Zhu, J. Li, Z. Tang, C. Shen, and F. Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.

[34] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[35] H. Huang, F. Liu, L. Fu, T. Wu, M. Mukadam, J. Malik, K. Goldberg, and P. Abbeel. Otter: A vision-language-action model with text-aware visual feature extraction. *arXiv preprint arXiv:2503.03734*, 2025.

[36] J. Liu, H. Chen, P. An, Z. Liu, R. Zhang, C. Gu, X. Li, Z. Guo, S. Chen, M. Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[37] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[38] H. Etukuru, N. Naka, Z. Hu, S. Lee, J. Mehu, A. Edsinger, C. Paxton, S. Chintala, L. Pinto, and N. M. M. Shafiullah. Robot utility models: General policies for zero-shot deployment in new environments. *arXiv preprint arXiv:2409.05865*, 2024.

[39] F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao. Data scaling laws in imitation learning for robotic manipulation. *arXiv preprint arXiv:2410.18647*, 2024.

[40] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[41] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. In *CoRL*, 2022.

[42] A. Majumdar, K. Yadav, S. Arnaud, J. Ma, C. Chen, S. Silwal, A. Jain, V.-P. Berges, T. Wu, J. Vakil, et al. Where are we in the search for an artificial visual cortex for embodied intelligence? *Advances in Neural Information Processing Systems*, 36:655–677, 2023.

[43] S. Dasari, M. K. Srirama, U. Jain, and A. Gupta. An unbiased look at datasets for visuo-motor pre-training. In *Conference on Robot Learning*, pages 1183–1198. PMLR, 2023.

[44] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR, 2022.

[45] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[46] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

[47] S. Wang, M. Han, Z. Jiao, Z. Zhang, Y. N. Wu, S.-C. Zhu, and H. Liu. Llmˆ 3: Large language model-based task and motion planning with motion failure reasoning. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12086–12092. IEEE, 2024.

[48] J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang, et al. Magma: A foundation model for multimodal ai agents. *arXiv preprint arXiv:2502.13130*, 2025.

[49] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.

[50] Y. Hu, F. Lin, T. Zhang, L. Yi, and Y. Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. *arXiv preprint arXiv:2311.17842*, 2023.

[51] B. Li, P. Wu, P. Abbeel, and J. Malik. Interactive task planning with language models, 2023.

[52] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, B. Zitkovich, F. Xia, C. Finn, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[53] L. X. Shi, Z. Hu, T. Z. Zhao, A. Sharma, K. Pertsch, J. Luo, S. Levine, and C. Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024.

[54] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh. Rt-h: Action hierarchies using language, 2024. URL https://arxiv.org/abs/2403.01823.

[55] Y. Dai, J. Lee, N. Fazeli, and J. Chai. Racer: Rich language-guided failure recovery policies for imitation learning. *International Conference on Robotics and Automation (ICRA)*, 2025.

[56] H. Chen, Y. Yao, R. Liu, C. Liu, and J. Ichnowski. Automating robot failure recovery using vision-language models with optimized prompts. *arXiv preprint arXiv:2409.03966*, 2024.

[57] P. Liu, Y. Orru, J. Vakil, C. Paxton, N. M. M. Shafiullah, and L. Pinto. Ok-robot: What really matters in integrating open-knowledge models for robotics. *arXiv preprint arXiv:2401.12202*, 2024.

[58] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic control via embodied chain-of-thought reasoning. In *Conference on Robot Learning*, 2024.

[59] F. Liu, K. Fang, P. Abbeel, and S. Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[60] S. Nasiriany, F. Xia, W. Yu, T. Xiao, J. Liang, I. Dasgupta, A. Xie, D. Driess, A. Wahid, Z. Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.

[61] A.-C. Cheng, Y. Ji, Z. Yang, Z. Gongye, X. Zou, J. Kautz, E. Bıyık, H. Yin, S. Liu, and X. Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024.

[62] R. Shah, A. Yu, Y. Zhu, Y. Zhu, and R. Martín-Martín. Bumble: Unifying reasoning and acting with vision-language models for building-wide mobile manipulation. *arXiv preprint arXiv:2410.06237*, 2024.

[63] P. Zhi, Z. Zhang, Y. Zhao, M. Han, Z. Zhang, Z. Li, Z. Jiao, B. Jia, and S. Huang. Closed-loop open-vocabulary mobile manipulation with gpt-4v. *arXiv preprint arXiv:2404.10220*, 2024.

[64] D. Qiu, W. Ma, Z. Pan, H. Xiong, and J. Liang. Open-vocabulary mobile manipulation in unseen dynamic environments with 3d semantic maps. *arXiv preprint arXiv:2406.18115*, 2024.

[65] L. X. Shi, B. Ichter, M. Equi, L. Ke, K. Pertsch, Q. Vuong, J. Tanner, A. Walling, H. Wang, N. Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

[66] Y. Li, Y. Deng, J. Zhang, J. Jang, M. Memmel, R. Yu, C. R. Garrett, F. Ramos, D. Fox, A. Li, et al. Hamster: Hierarchical action models for open-world robot manipulation. *arXiv preprint arXiv:2502.05485*, 2025.

[67] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *Computer Vision and Pattern Recognition (CVPR)*, 2025.

[68] J. Duan, W. Yuan, W. Pumacay, Y. R. Wang, K. Ehsani, D. Fox, and R. Krishna. Manipulate-anything: Automating real-world robots using vision-language models. *arXiv preprint arXiv:2406.18915*, 2024.

[69] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[70] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[71] X. Li, C. Mata, J. Park, K. Kahatapitiya, Y. S. Jang, J. Shang, K. Ranasinghe, R. Burgert, M. Cai, Y. J. Lee, et al. Llara: Supercharging robot learning data for vision-language policy. *arXiv preprint arXiv:2406.20095*, 2024.

[72] D. Niu, Y. Sharma, G. Biamby, J. Quenum, Y. Bai, B. Shi, T. Darrell, and R. Herzig. Llarva: Vision-action instruction tuning enhances robot learning. *arXiv preprint arXiv:2406.11815*, 2024.

[73] J. L. Jones. Robots at the tipping point: the road to irobot roomba. *IEEE Robotics & Automation Magazine*, 13(1):76–78, 2006.

[74] Dempsey. Reviews-consumer technology. the teardown-amazon astro consumer robot. *Engineering & Technology*, 18(2):70–71, 2023.

[75] H. Nguyen and C. C. Kemp. Autonomously learning to visually detect where manipulation will succeed. *Autonomous Robots*, 36:137–152, 2014.

[76] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[77] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 39(5):3929–3945, 2023.

[78] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023.

[79] A. Gupta, A. Murali, D. P. Gandhi, and L. Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. *Advances in neural information processing systems*, 31, 2018.

[80] T. Gervet, S. Chintala, D. Batra, J. Malik, and D. S. Chaplot. Navigating to objects in the real world. *Science Robotics*, 8(79):eadf6991, 2023.

[81] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine. ViNT: A foundation model for visual navigation. In *7th Annual Conference on Robot Learning*, 2023. URL https://arxiv.org/abs/2306.14846.

[82] K. Ehsani, T. Gupta, R. Hendrix, J. Salvador, L. Weihs, K.-H. Zeng, K. P. Singh, Y. Kim, W. Han, A. Herrasti, et al. Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. *arXiv preprint arXiv:2312.02976*, 2023.

[83] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[84] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[85] Q. Liu. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022.

[86] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

[87] O.-E. Collaboration, A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv:2310.08864*, 1(2), 2023.

[88] Q. Yu, Q. Sun, X. Zhang, Y. Cui, F. Zhang, Y. Cao, X. Wang, and J. Liu. Capsfusion: Rethinking image-text data at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14022–14032, 2024.

[89] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[90] P. Tong, E. Brown, P. Wu, S. Woo, A. J. V. IYER, S. C. Akula, S. Yang, J. Yang, M. Middepogu, Z. Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.

[91] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models. *arXiv preprint arXiv:2409.17146*, 2024.

[92] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

A Preliminaries

Vision-language-action models (VLAs) are typically trained via imitation learning on diverse robot demonstration datasets $\mathcal{D}$, by maximizing the log-likelihood of an action $\mathbf{a}_t$ (or, more generally, an action *chunk* $\mathbf{a}_{t:t+H}$) given an observation $\mathbf{o}_t$ and a natural language task instruction ℓ: $\max_\theta \mathbb{E}_{(\mathbf{a}_{t:t+H}, \mathbf{o}_t, \ell) \sim \mathcal{D}} \log \left(\pi_\theta(\mathbf{a}_{t:t+H} | \mathbf{o}_t, \ell) \right)$. The observation typically contains one or more images $\mathbf{I}_t^1, ..., \mathbf{I}_t^n$ and proprioceptive state $\mathbf{q}_t$, which captures the position of the robot's joints. VLA architectures follow the design of modern language and vision-language models, with modality-specific tokenizers that map inputs and outputs to discrete ("hard") or continuous ("soft") token representations, and a large, auto-regressive transformer backbone that is trained to map from input to output tokens. The weights of these models are initialized from pre-trained vision-language models. By encoding policy inputs and outputs into tokenized representations, the imitation learning problem described above can be cast as a simple next-token-prediction problem over a sequence of observation, instruction and action tokens, and we can leverage the scalable tools of modern machine learning to optimize it. In practice, the choice of tokenizers for image and text inputs follows those of modern vision-language models. For actions, prior work has developed effective, compression-based tokenization approaches [31], which we use in this work during pretraining. A number of recent VLA models have also proposed to represent the action distribution via diffusion [27, 33, 36] or flow matching [24], providing a more expressive representation over continuous-valued action chunks. During the post-training phase of our model, we will build on the design of the π_0 model [24], which represents the action distribution via flow matching. In this design, the tokens corresponding to actions receive the partially denoised actions from the previous step of flow matching as input, and output the flow matching vector field. These tokens also use a different set of model weights, which we refer to as an "action expert," analogously to a mixture of experts architecture. This action expert can specialize to flow matching-based action generation, and can be significantly smaller than the rest of the LLM backbone.

B Supplemental for Section 3: $\pi_{0.5}$ Model and Training Recipe

B.1 Model technical details

The $\pi_{0.5}$ model builds upon π_0 and adopts the PaliGemma VLM [84] as the backbone for visual-language understanding as well as an "action expert" for fast action generation. The VLM backbone takes in a sequence of images $[\mathbf{I}_t^1, \ldots, \mathbf{I}_t^n]$ and a language prompt ℓ as in π_0, but also the robot's proprioceptive state q_t in tokenized form and tokenized actions [31], which will be auto-regressively predicted. The action expert is a smaller transformer that takes in a sequence of noisy action tokens $\mathbf{a}_{t:t+H}^{\tau,\omega}$ for an action horizon of 50, i.e. $H = 49$, and is trained with the flow matching objective. The noisy action chunk (with action dimension d) is first projected to the transformer embedding dimension using a single linear layer. Unlike π_0 that fuses the flow-matching timestep τ with the noisy action before being fed into the transformer, $\pi_{0.5}$ uses a separate MLP for projecting τ only and then applies adaptive RMSNorm to inject the timestep information to each layer of the action expert. The timestep MLP takes in the form of $\text{swish}(W_2 \cdot \text{swish}(W_1 \cdot \phi(\tau)))$, where $\phi : \mathbb{R} \to \mathbb{R}^w$ is a sinusoidal positional encoding function [1] and $W_1, W_2 \in \mathbb{R}^{w \times w}$. The action expert outputs action tokens $y_{1:H}^a$, which are then decoded into the target vector field using a final linear projection.

The dimensions of the two transformers are the same as π_0: {*width*=2048, *depth*=18, *mlp_dim*=16,384, *num_heads*=18, *num_kv_heads*=1, *head_dim*=256} for the 2B VLM initialized

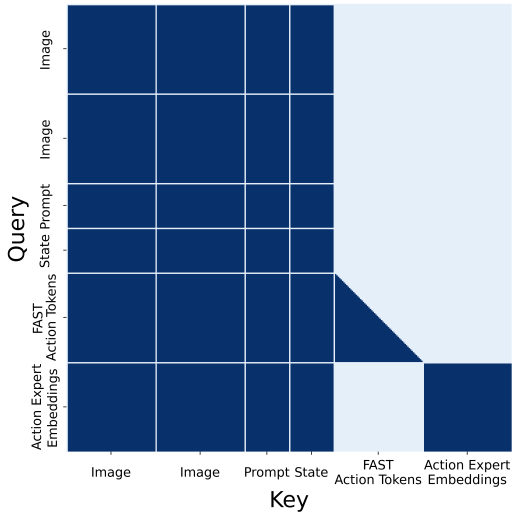


Figure 11: Example of the $\pi_{0.5}$ attention masking pattern.

from PaliGemma weights, and the same except for $\{$*width*=1024, *mlp_dim*=4096$\}$ for the action expert with 300M parameters.

The robot proprioceptive state is discretized and input to the model as text tokens. Embeddings from the VLM and action expert interact only through self-attention. A full prefix mask is used on images, prompt tokens, and proprioceptive state; FAST action tokens attend to this prefix and auto-regressively on previous action tokens. Embeddings from the action expert embeddings attend to the prefix and to one another, but do not attend to FAST action tokens to avoid information leakage between the two representations of actions. In effect, information flows unidirectionally from the VLM to the action expert; no VLM embedding attends to the action expert. An example of the attention mask at each layer is visualized in Figure 11.

We follow π_0 for sampling the flow-matching timestep τ. In summary we deviate from standard uniform sampling $\tau \sim \mathcal{U}(0,1)$ [83, 85] or methods emphasizing midrange timesteps [86], and instead use a time-step sampling distribution that emphasizes low time-steps [24], given by $p(\tau) = \text{Beta}(\frac{s-\tau}{s}; \alpha = 1.5, \beta = 1)$. Timesteps above the threshold s are excluded from sampling, as they are not needed if the integration step δ satisfies $\delta > 1 - s$. We use $s = 0.999$ in our experiments, which accommodates up to 1,000 integration steps ($\delta > 0.001$).

We apply image augmentation (random crop, resizing, rotation, and color jittering) to all input images using the following hyper-parameters and in this order

```
transforms = [
augmax.RandomCrop(int(width * 0.95), int(height * 0.95)),
    augmax.Resize(width, height),
    augmax.Rotate((-5, 5)),
    augmax.ColorJitter(brightness=0.3, contrast=0.4, saturation=0.5),
]
```

B.2 Pre-training

In the first training stage, $\pi_{0.5}$ is trained with a broad range of robot and non-robot data, which we summarize below and illustrate in Figure 12. It is trained as a standard auto-regressive transformer, performing next-token prediction of text, object locations, and FAST encoded action tokens.

Diverse *M*obile *M*anipulator data (MM). We use about 400 hours of data of mobile manipulators performing household tasks in about 100 different home environments, some of which are shown in Figure 5, using the robots in Section B.4. This slice of the training set is the most directly relevant

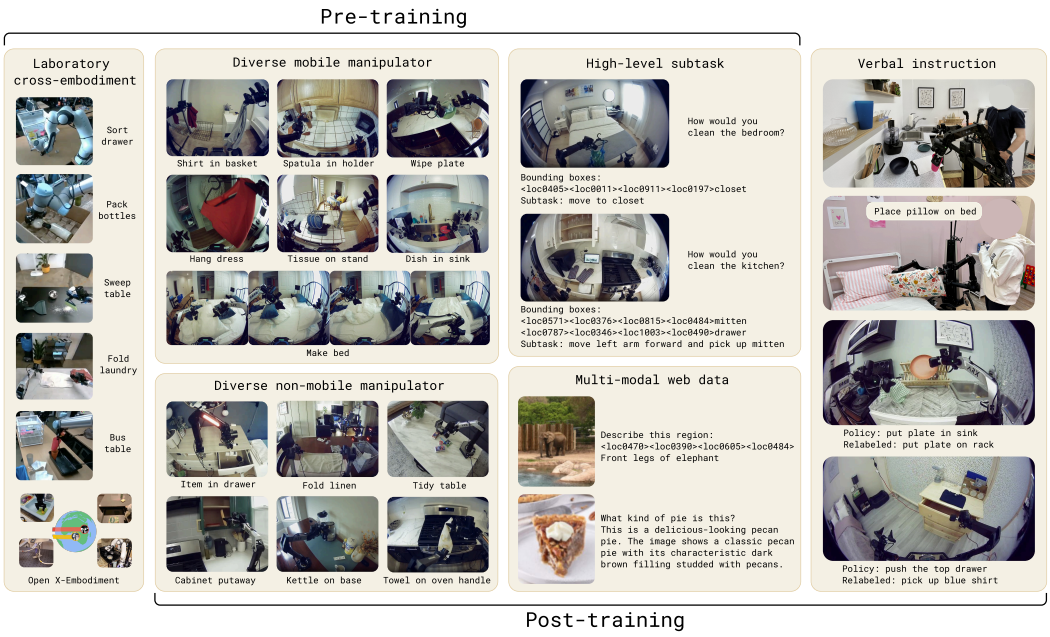


Figure 12: **Examples from pre-training and post-training tasks.** $\pi_{0.5}$ is pre-trained on data from mobile manipulators (MM), non-mobile robots in diverse environments (ME), and cross-embodiment data collected under laboratory conditions (CE), as well as high-level subtask prediction (HL), and multi-modal web data (WD). In a post-training phase, we additionally use verbal instructions (VI), and omit the laboratory cross-embodiment data (CE) to focus the model on mobile manipulation and diverse environments. The figure displays an exemplary subset of the tasks in each category.

to our evaluation tasks, which consist of similar cleaning and tidying tasks in new, unseen, home environments.

Diverse *M*ulti-*E*nvironment non-mobile robot data (ME). We also collected non-mobile robot data, either with a single arm or two arms, in a variety of home environments. These arms were fixed to surfaces or mounting platforms, and because they are significantly lighter and easier to transport, we were able to gather a more diverse dataset in a wider range of homes with them. However, this ME data comes from a different embodiment than the mobile robots.

***C*ross-*E*mbodiment laboratory data (CE).** We collected data for a wide range of tasks (e.g., bussing a table, folding shirts) in the laboratory, with simpler tabletop environments and a variety of robot types. Some of these tasks are highly relevant to our evaluation (e.g., putting dishes in a bin), while others are not (e.g., grinding coffee beans). This data includes single-arm and dual-arm manipulators, and both static and mobile bases. We also include the open-source OXE dataset [87]. This dataset is an extended version of the dataset used by π_0[24].

***H*igh-*L*evel subtask prediction (HL).** Breaking down high-level task commands such as "clean the bedroom" into shorter subtasks like "adjust the blanket" and "pick up pillow", similar to chain-of-thought prompting for language models, can help a trained policy reason about the current scene and better determine the next action. For robot data in MM, ME, and CE where the task involves multiple subtasks, we manually annotate all data with semantic descriptions of the subtasks and train $\pi_{0.5}$ to jointly predict the subtask labels (as text) as well as the actions (conditioned on the subtask label) based on the current observation and high-level command. This naturally leads to a model that can act both as a high-level policy (outputting subtasks) and low-level policy that executes actions for these subtasks. We also label relevant bounding boxes shown in the current observation and train $\pi_{0.5}$ to predict them before predicting the subtask.

Multi-modal *W*eb *D*ata (WD). Finally we include a diverse set of web data involving image captioning (CapsFusion [88], COCO [89]), question answering (Cambrian-7M [90], PixMo [91], VQAv2 [92]), and object localization in pre-training. For object localization, we further extend the

standard datasets with additional web data of indoor scenes and household objects with bounding box annotations.

For all action data, we train the model to predict target joint and end-effector poses. To differentiate the two, we add '<control_mode> joint/end effector <control_mode>' to the text prompt. All action data is normalized to $[-1, 1]$ using the 1% and 99% quantile of each action dimension of the individual dataset. We set the dimensionality of the action $\mathbf{a}$ to a fixed number to accommodate the largest action space among all the datasets. For robots with lower-dimensional configuration and action spaces, we zero-pad the action vectors.

The pre-trained model was trained for 280k gradient steps with a batch size of 2048.

B.3 Post-training

After pre-training the model with discrete tokens for 280k gradient steps, we perform a second stage of training that we refer to as post-training. The purpose of this stage is to both specialize the model to our use-case (mobile manipulation in homes), and to add an action expert that can produce continuous action chunks via flow matching. This stage jointly trains with next-token prediction, to preserve text prediction capabilities, and flow matching for the action expert (which is initialized with random weights at the beginning of post-training). We optimize the objective in Equation (1), with $\alpha = 10.0$ for 80k additional steps at a batch size of 384. The post-training action dataset consists of the MM and ME robot data, filtered down to successful episodes that are below a fixed length threshold (set as the 90th-percentile of task length). We include web data (WD) to preserve the model's semantic and visual capabilities, and the slice of HL data corresponding to the multi-environment datasets. Additionally, to improve the model's ability to predict appropriate high-level subtasks, we collect *verbal instruction* demonstrations (VI), which are constructed by expert users providing "language demonstrations," selecting appropriate sub-task commands to command the robot to perform mobile manipulation tasks step by step. These examples are collected by "tele-operating" the robot in real time with language to perform tasks with the learned low level policy, essentially providing demonstrations of good high-level subtask outputs for a trained policy.

B.4 Robot system details

The robot systems used in our mobile manipulation experiments are illustrated in Figure 13. We conducted all of our experiments using those two types of mobile manipulators. Both platforms are equipped with two 6 DoF arms with parallel jaw grippers and wrist-mounted monocular RGB cameras, a wheeled holonomic base, and a torso lift mechanism. The state and action spaces for the base correspond to linear (2D) and angular (1D) velocity, and the torso lift mechanism is either 1D (up/down) or 2D (up/down and forward/backward). In addition to the two wrist cameras, the robots have a forward and backward facing camera mounted between the arms. We use all four cameras for high-level inference, and the wrist and forward cameras for the low-level inference process. The total dimensionality of the state and action spaces is 18 or 19, depending on the platform.

The control system is very simple: the $\pi_{0.5}$ model directly commands target poses for the arms, gripper, and torso lift, and the target base velocities at 50 Hz (with action chunking). These targets are tracked with simple PD controllers, without any additional trajectory planning or collision detection. All manipulation and navigation control is fully end-to-end.

C Task evaluation rubric

For a quantitative evaluation of our method we performed rigorous evaluation of a subset of four tasks that are included in the training dataset (but evaluated in entirely new scenes and configurations). Among these are two kitchen cleanup tasks and two bedroom cleanup tasks. Each task is evaluated with a consistent set of items for each of the policies within a comparison (but items varied between locations) in three different homes and three different mock kitchens and mock bedrooms respectively (a total of 12 different locations). For each evaluation and each policy, unless

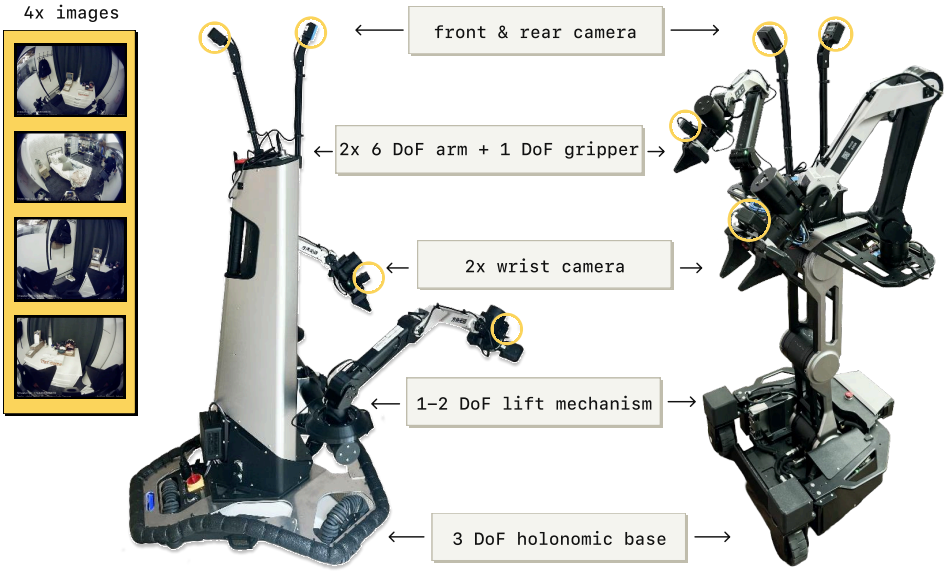


Figure 13: **Robot system overview.** We use two mobile manipulator platforms – each has four cameras (forward, backward, and both wrists), two 6 DoF arms with parallel jaw grippers, a mobile base, and a torso lift mechanism. The $\pi_{0.5}$ model controls the joints and grippers of each arm, base velocity, and the lift position, resulting in 18-19 DoF state and action spaces.

otherwise stated, we perform 10 evaluations per task; note that each of these evaluation episodes can span multiple minutes and they are thus time intensive. We present results as percent of total points achieved in each evaluation rubric (as outlined below) and present either per task metrics or metrics averaged across all tasks in four different locations, that are consistent for all policies in a comparison, leading to a total of 40 evaluations per policy for our standard evaluations. Evaluations were carried out by interleaving execution of policies to control for environmental changes. Some evaluations include cancelled episodes due to robot failures, time limitations or other causes, which are removed. In all cases we control the sample size to be close and report statistical significance according to a two-sided t-test assuming variable number of trials within the plots. The language following evaluations follow a different protocol as described in the main text.

The evaluation metrics for the kitchen cleanup tasks, which include placing dishes into a sink and storing items in a drawer, are detailed below.

- *Dishes in Sink:* The task begins with 4 dishes (e.g., plates, bowls, cutting boards, utensils) placed near a sink. The robot's goal is to place all of them in the sink.

  +1 For each item picked up.

  +1 For each item placed in the sink.

  Maximum score: 8 points.

- *Items in Drawer:* The task begins with an item on a countertop. The robot must place the item into a drawer beneath the counter.

  +1 Picking up the object.

  +1 Opening the drawer.

  +1 Putting the object into the drawer.

  +1 Closing the drawer (if the object is inside).

  Maximum score: 4 points.

Next, we outline the evaluation metrics for the bedroom cleanup tasks: putting laundry away and making a bed.

- *Laundry in Basket:* The task begins with an article of clothing lying on the ground. The robot's goal is to pick up the laundry and place it in the laundry basket.

  +1 Navigating to and picking up the clothing.

  +1 Placing the clothing into or on the laundry basket.

  +1 Clothing is fully inside the basket.

  Maximum score: 3 points.

- *Make the Bed:* The bed starts unmade. The robot must tidy the blanket and place two pillows at the head of the bed.

  +1 Straightening the blanket so it covers the sheets.

  +1 Placing one pillow at the head of the bed.

  +1 Placing the second pillow at the head of the bed.

  +1 Blanket is straightened very neatly.

  +1 Both pillows are placed very neatly.

  Maximum score: 5 points.

D Language following experiment setup

The language following experiments use two unseen kitchen scenes to test how well the model follows more specific user commands, such as *"put the scissors in the drawer"* or *"put the cutting board into the sink"*. Each trial requires the robot to interpret the instruction, identify the correct object amidst distractors, and perform the task. We evaluate on two scenarios:

1. **Items in the drawer:** common kitchen items (tongs, wooden serving spoon, can opener, scissors, and small yellow mustard).
2. **Items in the sink:** common dining items (cup, bowl, plate, plastic spoon, and cutting board).

In each trial, the robot is presented with five objects and is instructed to move one of them. To discourage shortcut behaviors, the target object is placed further away than the distractors, such that a policy that is unable to interpret the command should achieve only $\sim$20% language following accuracy. We report two metrics, averaged over both scenarios: **language following rate**, which measures whether the correct object was selected, and **task success rate**, which evaluates whether the object was successfully placed in the specified location. We further investigate how the number of distinct training environments influences the model's ability to generalize to previously unseen objects. We design a similar **Items in the drawer** task with novel household items (a funnel, a pill bottle, a grill lighter, a lighter, and a pair of safety goggles). None of these object categories were present in the training set, ensuring that this task tests the robot's performance on out-of-distribution (OOD) objects. We show the example initial scene of each task in Figure 14.

Along with data ablation experiments in Figure 9 and location scaling experiments in Figure 7, Figure 15 presents language following results across model classes. We find that $\pi_{0.5}$ follows language at a slightly higher rate than π_0-FAST+Flow, even though the high-level inference mode is not used for this evaluation, indicating that training with high-level data improves both overall performance and language following. We also find that $\pi_{0.5}$ significantly outperforms π_0, indicating the importance of discrete token training on language following abilities.

D.1 How important is high-level inference?

We evaluate the importance of high-level inference, and compare the performance of several alternative high-level inference methods. The high-level inference mechanism in $\pi_{0.5}$ takes in a high-level command (e.g., "clean the bedroom") and outputs the subtask to complete (e.g., "pick up pillow"),

(a) In-distribution objects, items in drawer (b) In-distribution objects, dishes in sink (c) OOD objects, items in drawer

Figure 14: Example initial states of different language following experiments.

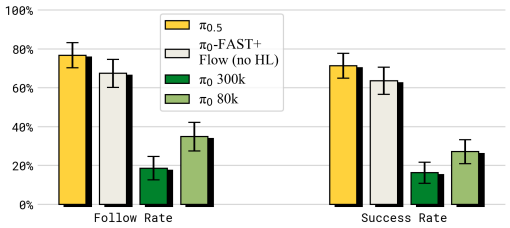


Figure 15: **Comparing $\pi_{0.5}$ with other models on language following.** We evaluate language following capabilities of $\pi_{0.5}$, π_0, and π_0-FAST+Flow, finding $\pi_{0.5}$ outperforms each, and π_0 by a wide margin.

which is then used as context for inferring the lower-level actions, analogously to chain of thought inference [69]. While $\pi_{0.5}$ uses a unified architecture where the *same* model performs both high-level and low-level inference, we can also construct baseline methods that either forego the high-level inference process and feed the task prompt directly into the low-level system, as is common in standard VLA models [22, 24], or use another model for high-level inference to ablate the importance of different dataset components in terms of their impact on the high-level policy. We consider the following methods and ablations, all of which use the full $\pi_{0.5}$ low-level inference process with different high-level policies:

1. $\pi_{0.5}$ model for high-level and low-level inference.
2. **no WD**: an ablation of $\pi_{0.5}$ that excludes web data.
3. **no VI**: an ablation of $\pi_{0.5}$ that excludes the verbal instruction (VI) data.
4. **implicit HL**: no high-level inference at runtime but includes high-level data in training, which may teach the model about subtasks implicitly.
5. **no HL**: no high-level inference, and no high-level data in training at all.
6. **GPT-4**: use GPT-4 as the high-level policy, evaluating the importance of training the high-level policy on robot data. To align the model with our domain, we prompt GPT-4 with a description of the task and a list of the most used labels to choose from.
7. **human HL**: use an expert hu bound on performance.

The results of these experiments are shown in Figure 16. The full $\pi_{0.5}$ model performs the best, and outperforms even the **human HL** "oracle" baseline. Perhaps surprisingly, the second best model is the **implicit HL** ablation, which does *not* perform any high-level inference, but includes the full data mixture, i.e. also subtask prediction, in training. This strongly suggests the importance of the co-training recipe used by our model: while there is a benefit to explicitly infer high-level subtasks, a significant portion of that benefit is already obtained simply by including subtask *prediction* data in the training mixture. The **no HL** ablation, excluding HL task even in training, performs significantly worse. The results also

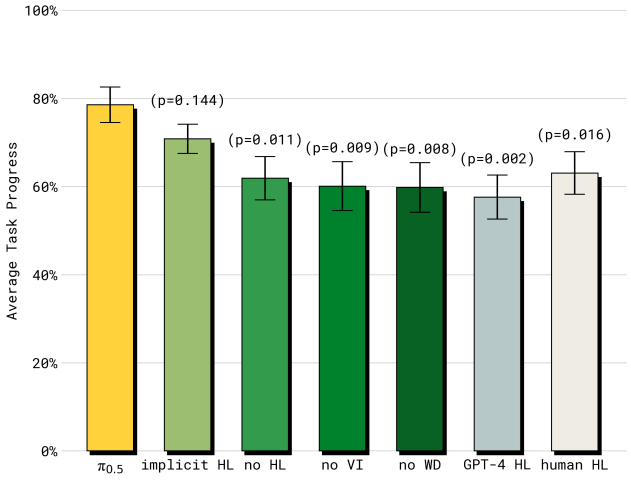


Figure 16: **Evaluation of the high-level inference process.** While the full $\pi_{0.5}$ model with high-level and low-level inference attains the best results, using only low-level inference ("implicit HL") with the full $\pi_{0.5}$ model also benefits from the inclusion of high-level subtask examples in training. In contrast, excluding verbal instructions (no VI) or web data (no WD) leads to a significant degradation in performance, and zero-shot prompting a large API-based model (GPT-4) performs worse.

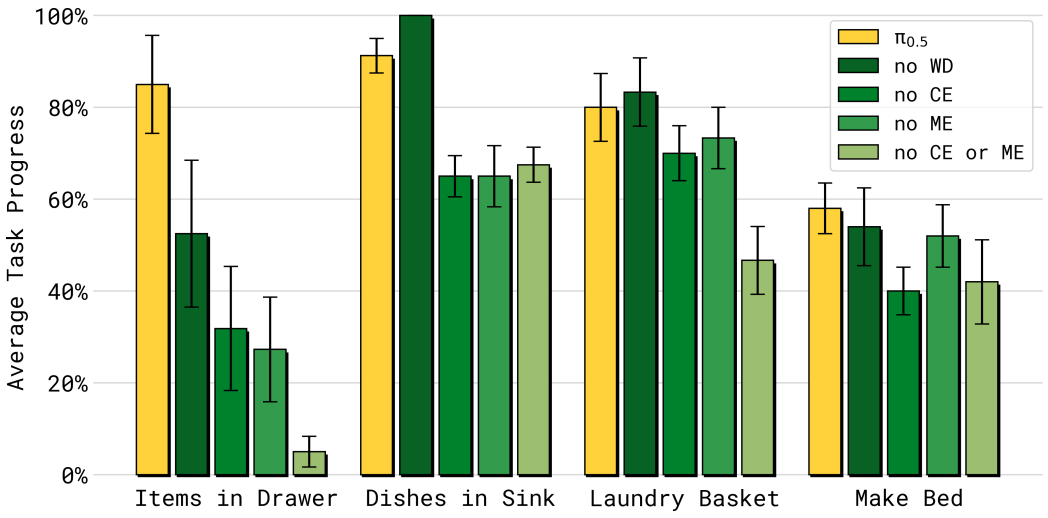


Figure 17: **Per-task performance breakdown for training recipe ablations.** We evaluate each training mixture variant on four representative household tasks: Items in Drawer, Dishes in Sink, Laundry Basket, and Make Bed. Removing cross-embodiment data (ME or CE) leads to significant degradation in specific tasks, particularly Items in Drawer and Dishes in Sink. Web data (WD) shows greater effect on the task (Items in Drawer) where the broad knowledge of the scene is desired.

show that the relatively small verbal instruction dataset, which only constitutes about 11% of the high-level mobile manipulation examples, is critical to strong performance as the **no VI** ablation is significantly weaker. The **no WD** ablation is also significantly worse, indicating that much of the benefit of web data (perhaps unsurprisingly) lies in improving the high-level policy. Finally, the zero-shot **GPT-4** ablation attains the worst performance, indicating the importance of adapting VLMs with robot data. We provide a detailed breakdown of performance on each task in Appendix E, Figure 18.

E Per-task performance breakdown

Co-training recipe ablations To better understand the influence of different training data sources on specific task categories, we provide a per-task performance breakdown (Figure 17). Here we consider four representative household tasks: *Items in Drawer*, *Dishes in Sink*, *Laundry Basket*, and *Make Bed*. In summary, the results indicate that cross-embodiment transfer and diverse data co-training are critical for generalization across a range of tasks, with varying degrees of reliance depending on task requirements.

For *Items in Drawer*, performance drops substantially when cross-embodiment data (ME or CE) or web data (WD) is removed, with the largest degradation observed when all are excluded. This task requires recognizing and understanding a very broad class of common objects, and such knowledge may be learned from diverse data sources. In contrast, *Dishes in Sink* remains relatively robust to the removal of web data (WD) but degrades when cross-embodiment data (ME or CE) is excluded, anchoring the intuition that this task primarily requires general manipulation strategies learned from robotic data. *Laundry Basket* and *Make Bed* also exhibit performance degradation when cross-embodiment data is removed, but are generally less sensitive to other changes in the data mixture.

High-level model analysis For a more granular view of how different high-level inference methods affect specific task categories, we again provide a per-task breakdown (Figure 18). We evaluate the full $\pi_{0.5}$ model and all high-level inference baselines across four representative tasks: *Items in Drawer*, *Dishes in Sink*, *Laundry Basket*, and *Make Bed*. The results show that explicit high-level inference improves performance across tasks, with the full $\pi_{0.5}$ model achieving the best overall results.

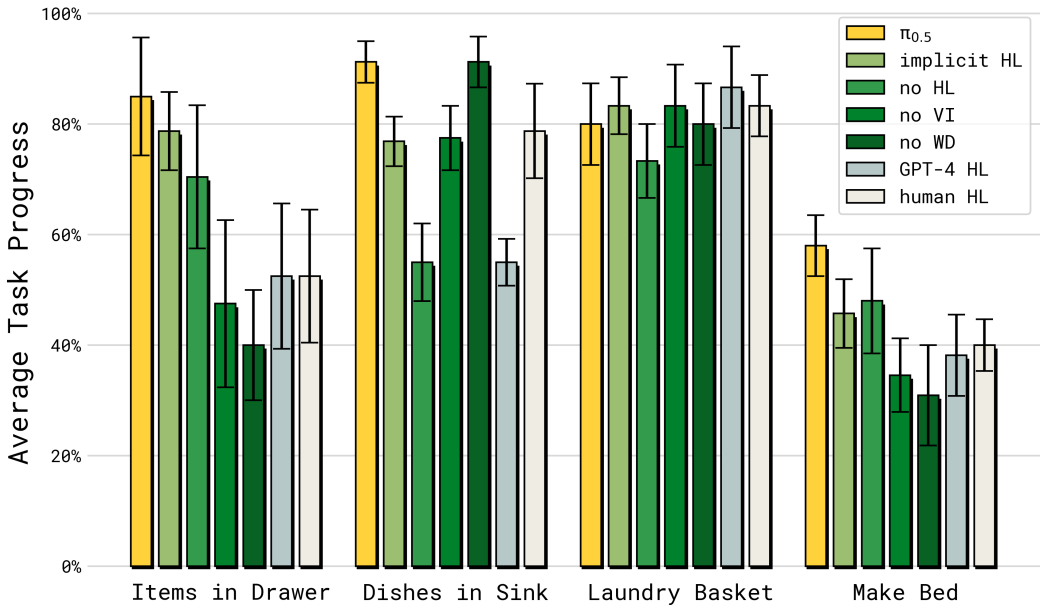


Figure 18: **Per-task performance breakdown for high-level inference methods.** We evaluate the full $\pi_{0.5}$ model and various high-level inference baselines across four representative household tasks.

For *Items in Drawer* and *Dishes in Sink*, high-level inference is critical: performance drops substantially with the *no HL* variant, indicating the importance of structured subtask prediction and long-horizon planning. In these two tasks, the $\pi_{0.5}$ model also outperforms *GPT-4 HL*, showing the benefit of in-domain fine-tuning and demonstrating that the high-level model learns strategies that help the low-level policy succeed. In *Items in Drawer*, performance also declines sharply when web data is removed — this echos the result from the co-training recipe ablation and highlights the importance of semantic knowledge for generalizing to less seen objects. For *Laundry Basket* and *Dishes in Sink*, the model is less sensitive to the choice of the high-level policy. These tasks are either relatively shorter in horizon or require less detailed semantic reasoning.