

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

Physical Intelligence

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

<https://pi.website/blog/pi05>

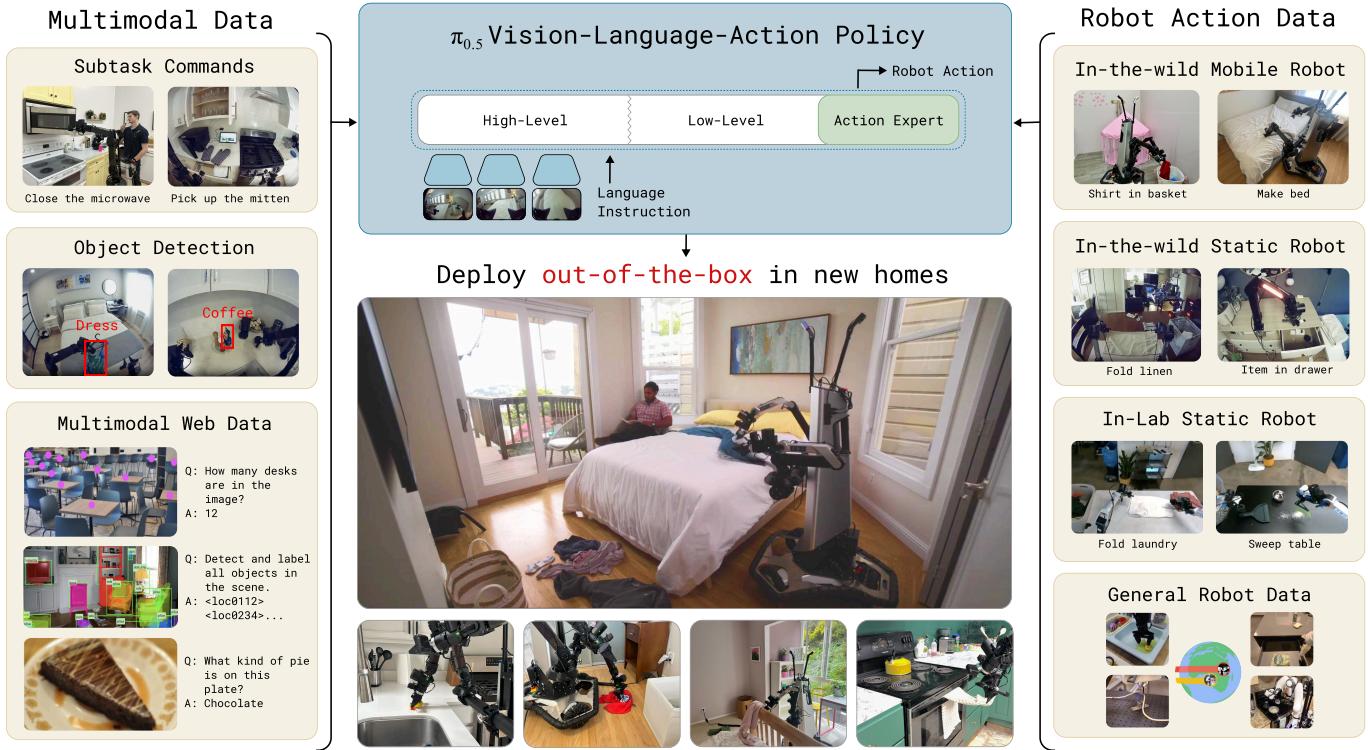


Fig. 1: The $\pi_{0.5}$ model transfers knowledge from a heterogeneous range of data sources, including other robots, high-level subtask prediction, verbal instructions, and data from the web, in order to enable broad generalization across environments and objects. $\pi_{0.5}$ can control a mobile manipulator to clean kitchens and bedrooms in new homes that were not present in the training data, performing complex multi-stage behaviors with durations of 10 to 15 minutes.

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.

I. INTRODUCTION

Stuff your eyes with wonder... See the world. It's more fantastic than any dream made or paid for in factories.

Ray Bradbury, *Fahrenheit 451*

Open-world generalization represents one of the biggest open problems in physical intelligence: embodied systems such as robotic arms, humanoids, and autonomous vehicles only truly become useful when they can leave the lab and handle the diverse situations and unexpected events that occur in the real world. Learning-based systems offer a path to en-



Fig. 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. The model is given general tasks (close the cabinets, put the items in the drawer, wipe the spill, and put the dishes in the sink), which it performs by both predicting subtasks to accomplish (e.g., pick up the plate) and emitting low-level actions.

abling broad generalization, particularly with recent advances that have enabled scalable learning systems in domains ranging from natural language processing [79, 21, 10, 78] to computer vision [34, 66, 35, 43]. However, the diversity of situations that a robot might encounter in the real world requires more than just scale: we need to design training recipes that can provide the breadth of knowledge that will allow robots to generalize at many levels of abstraction. For example, if a mobile robot is asked to clean up a kitchen that it has never seen before, some behaviors generalize readily if they are well represented in the data with a sufficient range of scenes and objects (e.g., picking up a knife or plate), others might require adapting or modifying existing skills to use them in a new way or in a new sequence, and yet others might require understanding the semantics of the scene based on prior knowledge (e.g., which drawer to open, or 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. Not all of this experience is firsthand, and not all of it comes from rote practice – for example, we might use facts that we were told by others or read in a book, together with bits of insight from other tasks we have performed in different contexts, combined with direct experience in the target domain. Analogously, we might hypothesize that generalizable robotic learning systems must be able to transfer experience and knowledge from a variety of information sources. Some of these sources are firsthand experience with direct relevance to the task at hand, some require transfer from other robot embodiments, environments, or domains, and some represent entirely different 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 fortunately 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.

In this paper, we leverage this observation to design a co-training framework for VLAs that can utilize heterogeneous and diverse knowledge sources to enable broad generalization.

Building on the π_0 VLA, we propose to include a range of different data sources to create 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, such as image captioning, question answering, and object localization (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, but from these other sources, such as other robots or data from the web. Nonetheless, $\pi_{0.5}$ is able to control mobile manipulators in entirely new homes not seen during training, perform intricate tasks such as hanging up towels or making beds, and can carry out long-horizon manipulation skills 10 to 15 minutes in length, cleaning an entire kitchen or bedroom based on only a high-level prompt.

The design of $\pi_{0.5}$ 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 low-level robot action chunk based on this subtask. This simple architecture provides both the ability to reason about long-horizon multi-stage tasks and the ability to leverage different sources of knowledge for the two levels: the low-level action inference procedure readily benefits from action data collected by other robots, including simpler static robots in other environments, while the high-level inference procedure benefits from semantic examples from the web, high-level annotation prediction, and even verbal commands that can be provided to the robot by human “supervisors” that walk the

robot through complex tasks step by step, instructing it (much like how they might instruct a person) on the appropriate subtasks to perform to complete a complex task such as cleaning a room. We illustrate this design in Figure 1.

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. 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.

II. RELATED WORK

Generalist robot manipulation policies. Recent works have demonstrated that broadening the training data distribution for robot manipulation policies from narrow, single-task datasets to diverse datasets that span many scenes and tasks [17, 25, 80, 63, 41, 6, 30, 67, 1] allows the resulting policies to not only solve a wider range of tasks out of the box, but also improves their ability to generalize to *new* scenes and tasks [9, 63, 62, 22]. Training such *generalist* policies requires new modeling approaches that can handle the scale and diversity of datasets that often span hundreds of different tasks and scenes. Vision-language-action models (VLAs) [23, 92, 42, 8, 83, 90, 55, 45, 3, 75, 64, 76, 84, 7, 37] offer an appealing solution: 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 [8], diffusion [55, 84, 52], or advanced action tokenization schemes [64], VLAs can perform a wide range of complex manipulation tasks in the real world. However, despite impressive language following abilities, VLAs are still typically evaluated in environments that closely match their training data. While some studies suggest that simple skills like picking up objects or opening drawers can be made to generalize simply by collecting robot data in a broader set of environments [14, 67, 28, 49, 64], it is challenging to apply the same approach to more complex, long-horizon tasks like cleaning up a kitchen, where achieving broad coverage of plausible scenarios via brute-force scaling of robot data collection is infeasible. In our experiments, we evaluate $\pi_{0.5}$ in entirely new scenes, such as new kitchens and bedrooms that were not seen in training, showing that our VLA can generalize to entirely new scenes by leveraging not only direct first-hand experience on the target mobile manipulator platform, but also information from other data sources. These

sources include data from other (non-mobile) robots, high-level semantic subtask prediction, and data from the web.

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. Prior methods have explored initializing vision encoders from computer vision datasets [85, 58, 57, 18], or leveraging off-the-shelf task planners [38, 48, 73, 81]. VLA policies are typically initialized from a pre-trained vision-language model, which has been exposed to large amounts of internet vision and language data [23, 92, 42]. Notably, the VLA architecture is flexible and allows to map between input and output sequences of multi-modal vision, language, and action tokens. As such, VLAs broaden the design space of possible transfer approaches beyond simple weight initialization, by supporting the *co-training* of a single, unified architecture on not just robot action imitation data, but any dataset that interleaves one or multiple of the aforementioned modalities. Prior works have demonstrated that co-training VLAs with data mixtures used for VLM training [23, 92, 86] can improve their generalization ability, 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. We believe that this level of generalization, particularly when accounting for the complexity of the tasks, goes significantly beyond the results demonstrated in prior works.

Robot reasoning and planning with language. A number of prior works have shown that augmenting end-to-end policies with high-level reasoning can significantly improve performance for long-horizon tasks [2, 36, 44, 74, 71, 4, 16, 11, 53, 88, 51, 59, 13, 70, 91, 65, 72, 47, 76, 89], 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 [2, 71, 13, 24, 70, 72, 47]. Our method uses the same exact model for both high-level and low-level inference, in a recipe that more closely resembles chain-of-thought [82] or test-time compute [39] methods, though unlike embodied chain-of-thought methods [88, 46, 61], 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

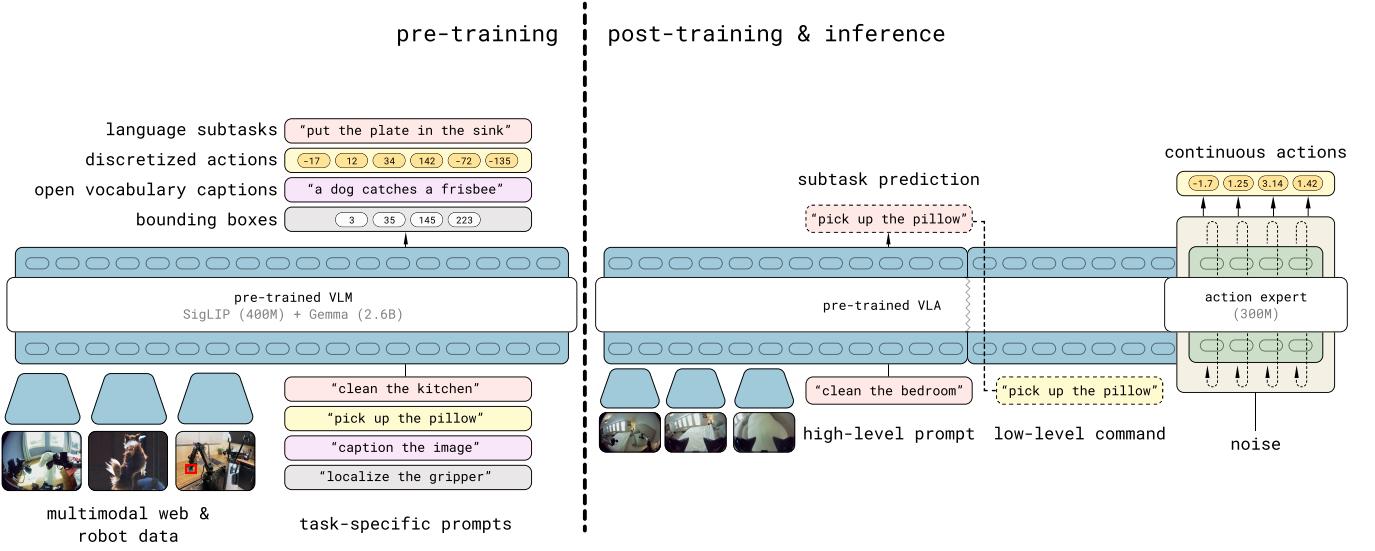


Fig. 3: **Model overview.** $\pi_{0.5}$ is trained in two stages. First, a pre-training stage combines all of the different data sources to produce an initial VLA with discrete tokens. This stage uses data from diverse robotic platforms, high-level semantic action prediction, and data from the web. Robotic data uses the FAST action tokenizer to represent actions as discrete tokens [64]. 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.

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 [40, 20, 60, 56, 29]. 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 [41, 68, 63, 67, 14, 49] have been shown to enable generalization of simple but end-to-end learned tasks to new environments [33, 31, 67, 69, 26, 49, 28, 64]. 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.

III. 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 : $\max_{\theta} \mathbb{E}_{(\mathbf{a}_{t:t+H}, \mathbf{o}_t, \ell) \sim \mathcal{D}} \log (\pi_{\theta}(\mathbf{a}_{t:t+H} | \mathbf{o}_t, \ell))$. The observation typically contains one or more images $\mathbf{I}_t^1, \dots, \mathbf{I}_t^n$ 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, autoregressive 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 [64], which we use in this work during pretraining. A number of recent VLA models have also proposed to represent the action distribution via diffusion [55, 84, 52] or flow matching [8], 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 [8], 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.

IV. THE $\pi_{0.5}$ MODEL AND TRAINING RECIPE

We provide an overview of the $\pi_{0.5}$ model and training recipe in Figure 3. The model weights are initialized from a standard VLM trained on data from the web, 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 simple, scalable, and efficient training [64]. 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. We describe the model architecture below, followed by a description of each of the phases and their corresponding training tasks.

A. 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 $\mathbf{o}_t = [\mathbf{I}_t^1, \dots, \mathbf{I}_t^n, \mathbf{q}_t]$ 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 (tokenized) 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 $\pi_\theta(\mathbf{a}_{t:t+H}|\mathbf{o}_t, \hat{\ell})$, with both distributions represented by the same model.

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 $y_{1:N} = f(x_{1:N}, A(x_{1:N}), \rho(x_{1:N}))$. 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 [8], 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. Compared to standard causal attention

in LLMs, image patch, textual prompt, and continuous action tokens use bidirectional attention.

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 ($y_{1:M}^\ell, y_{1:H}^a$). 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. The robot proprioceptive state is discretized and input to the model as text tokens. More details about the architecture are in Appendix E.

B. Combining discrete & continuous action representations

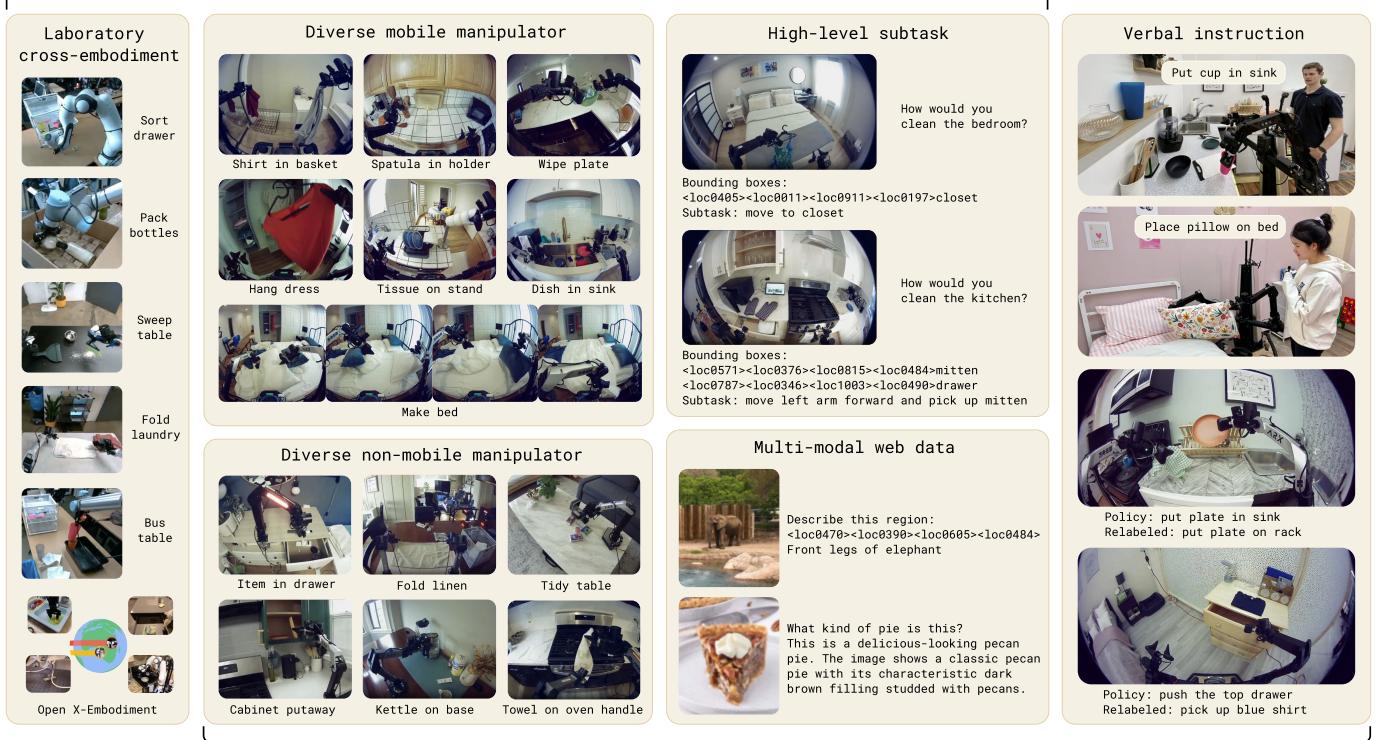
Similarly to π_0 , we use flow-matching [50] to predict continuous actions in the final model. Given $\mathbf{a}_{t:t+H}^{\tau, \omega} = \tau \mathbf{a}_{t:t+H} + (1 - \tau) \omega$, $\omega \sim \mathcal{N}(0, \mathbf{I})$, where $\tau \in [0, 1]$ is the flow matching time index, the model is trained to predict the flow vector field $\omega - \mathbf{a}_t$. However, as shown in [64], VLA training can be much faster when actions are represented by discrete tokens, particularly when using a tokenization scheme that is efficient for compressing the action chunks (e.g., FAST). Unfortunately, such discrete representations are less well-suited for real-time inference, because they require expensive autoregressive decoding for inference [64]. 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

$$\begin{aligned} \mathbb{E}_{\mathcal{D}, \tau, \omega} \left[H(x_{1:M}, f_\theta^\ell(\mathbf{o}_t, \ell)) \right. \\ \left. + \alpha \| \omega - \mathbf{a}_{t:t+H} - f_\theta^a(\mathbf{a}_{t:t+H}^{\tau, \omega}, \mathbf{o}_t, \ell) \|^2 \right], \quad (1) \end{aligned}$$

where $H(x_{1:M}, y_{1:M}^\ell)$ is the cross entropy loss between the text tokens and predicted logits (including the FAST encoded action tokens), $y_{1:H}^a = f_\theta^a(\mathbf{a}_{t:t+H}^{\tau, \omega}, \mathbf{o}_t, \ell)$ 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. We find that following this procedure, which is further explained below, leads to stable pre-training and excellent language following abilities of the VLA model. At inference time we then use standard autoregressive decoding for text tokens $\hat{\ell}$ followed by 10 denoising steps, conditioned on text tokens, to produce actions $\mathbf{a}_{t:t+H}$.

Pre-training



Post-training

Fig. 4: **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.

C. 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 4. It is trained as a standard auto-regressive transformer, performing next-token prediction of text, object locations, and FAST encoded action tokens.

Diverse Mobile Manipulator 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 7, using the robots in Section IV-E. This slice of the training set is the most directly relevant to our evaluation tasks, which consist of similar cleaning and tidying tasks in new, unseen, home environments.

Diverse Multi-Environment 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.

Cross-Embodiment 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 [15]. This dataset is an extended version of the dataset used by π_0 [8].

High-Level 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 Web Data (WD). Finally we include a diverse set of web data involving image captioning (CapsFusion [87], COCO [12]), question answering (Cambrian-7M [77], PixMo [19], VQAv2 [32]), 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 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.

D. 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. 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. 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 “teleoperating” 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.

E. Robot system details

The robot systems used in our mobile manipulation experiments are illustrated in Figure 5. We conducted all of our experiments using 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.

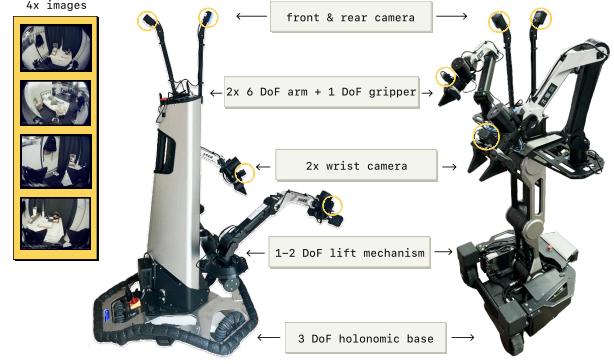


Fig. 5: **Robot system overview.** We use two mobile manipulator platforms – each has four cameras (forward, backward, and both wrists), two 6 DoF arms with parallel jaw grippers, a mobile base, and a torso lift mechanism. The $\pi_{0.5}$ model controls the joints and grippers of each arm, base velocity, and the lift position, resulting in 18-19 DoF state and action spaces.

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.

V. 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 6). Our experiments focus on the following questions:

- 1) Can $\pi_{0.5}$ effectively generalize to complex multi-stage tasks in entirely new homes?
- 2) How does the generalization of $\pi_{0.5}$ scale with the number of distinct environments in the training data?
- 3) How do the individual co-training ingredients in the $\pi_{0.5}$ training mixture contribute to its final performance?
- 4) How does $\pi_{0.5}$ compare to the π_0 VLA?
- 5) How important is the high-level inference component of $\pi_{0.5}$, and how does it compare to flat, low-level inference as well as oracle high-level baselines?

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

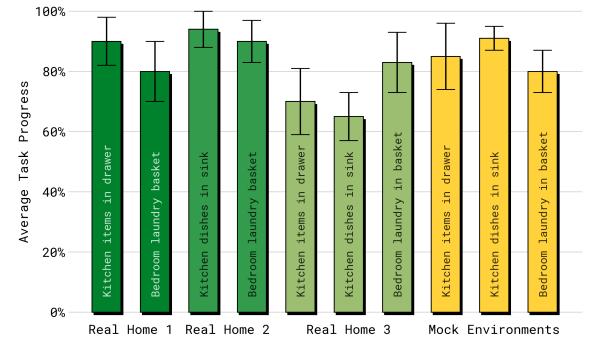
To answer Question (1), we evaluated $\pi_{0.5}$ in three real homes that were not present in the training set, using both types of robots. In each of the homes, the robots were instructed to perform a bedroom and kitchen cleaning task. The evaluation rubrics for each task are provided in Appendix B 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 7



Fig. 6: **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).



(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.

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.

B. How does generalization scale with the number of scenes?

In the next set of experiments, we aim to measure how generalization scales with the number of environments seen in the training data. We vary the number of environments in the mobile manipulation data and measure its impact on generalization by training with data from 3, 12, 22, 53, 82, and 104 locations. 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. 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 6, which are not seen in training. We conduct two types of evaluations. First, to evaluate overall performance on multi-stage tasks, we use the standard rubric in Appendix B and the mock test homes to evaluate each model's end-to-end performance on putting dishes in the sink, packing items into a drawer, putting away laundry, and making a bed. Second, we conduct 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 objects from a kitchen counter based on language commands. These experiments use both in-distribution objects from similar categories as

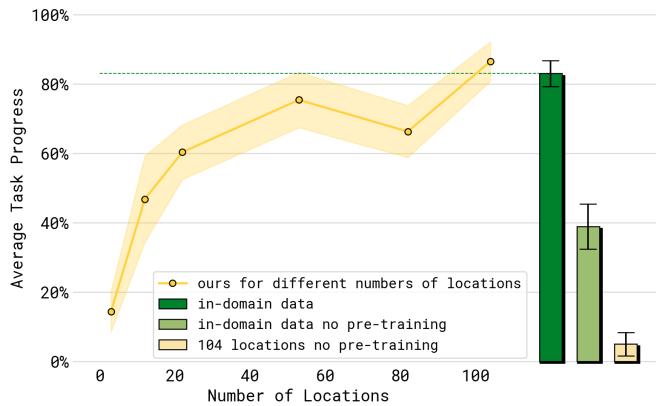


Fig. 8: Evaluating performance with different numbers of locations. Performance over the four test tasks — “dishes in sink”, “items in drawer”, “laundry basket”, “make bed” — improves with more training environments. The dashed green line and green bar show a baseline model that includes the test homes in the training set. Compared to this model, our best model achieves similar performance, despite not seeing any data from the test homes.

those in the training data (but new instances), as well as out-of-distribution objects from unseen categories. The latter necessitates broad semantic generalization.

The results of the first experiment are shown in Figure 8. The average performance among the tasks generally improves with more training locations. To quantify how much the final model (with 104 locations) bridges the generalization gap, we include a control (shown in green) that is trained directly on data from the test homes. This control attains similar performance as the final 104-location model, suggesting that our co-training recipe effectively enables broad generalization, reaching similar performance to a model trained on the test environment. 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 — this indicates that the other data sources leveraged by our full training recipe are essential for good generalization, even when the policy has seen robot data from test homes. When not using data from test homes, pre-training with our recipe is especially important, as can be seen by the large gap between the green bars and light yellow bar in Figure 8.

The results of the second experiment (language following) are shown in Figure 9. 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 (either inside the drawer or inside the sink, depending on the test scenario). We separately measure performance on object categories seen in training (but new object instances) and unseen (“out-of-distribution”) object categories. Details of this experiment are shown and discussed in Appendix C. Figure 9 shows that, as the number

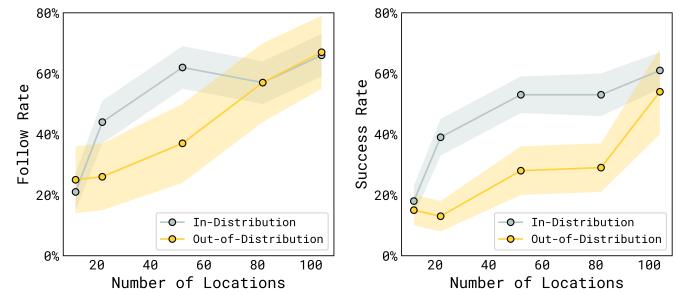


Fig. 9: Evaluating language following with different numbers of training locations. We evaluate language following rate and success rate for picking up user-indicated items and placing them into drawers or sinks, averaged over seen object categories (“in-distribution”) or unseen categories (“out-of-distribution”). Performance increases steadily as we increase the number of training locations.

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. As each new environment introduces new household items, the model becomes generally more robust and starts to generalize to task categories that were not present in the training data.

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

To study Question (3), we compare our full $\pi_{0.5}$ model to other training mixtures to study the importance of each mixture component, again using end-to-end task performance in the mock homes and the language following evaluation described in Section V-B. As a reminder, our full recipe uses data from mobile manipulators in many environments (MM), static manipulators in many environments (ME), and diverse cross-embodiment data collected in laboratory settings (CE). It also includes high-level data where the prediction corresponds to a high-level language command (HL), and web data corresponding to captioning, VQA, and object localization tasks (WD). Post-training also uses verbal instruction data (VI), which we analyze in Section V-E. In these experiments, we ablate different parts of the mixture:

- 1) **no WD**: this ablation excludes web data.
- 2) **no ME**: this ablation excludes multi-environment non-mobile data.
- 3) **no CE**: this ablation excludes the laboratory cross-embodiment data.
- 4) **no ME or CE**: this ablation excludes both data sources from other robots, such that the model is trained on only data from the target mobile manipulator platform as well as web data.

The results on the full mock home tasks are shown in Figure 10 (detailed breakdown of performance on each task in Appendix D). First, we see in the results that excluding *either* of the two cross-embodiment data sources (ME and CE) 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

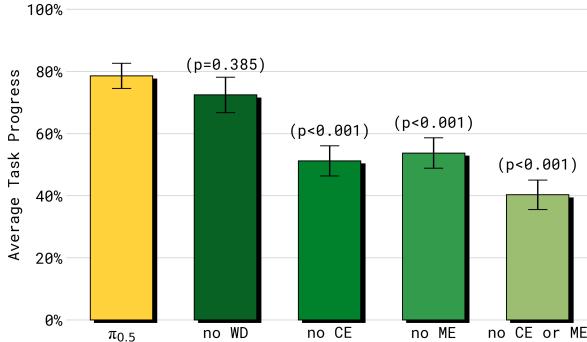


Fig. 10: **Training recipe ablations, mock homes.** We evaluate variants of our model that exclude different parts of the training mixture on all four test tasks (10 trials per policy and task). Including cross-embodiment data, both in diverse environments (ME) and for diverse tasks in laboratory settings (CE) is important for good performance, with large degradation when either or both of these data sources are removed. Web data (WD) does not make a significant difference in these experiments, but we will see in Figures 11 and 13 that it impacts object generalization and high-level performance.

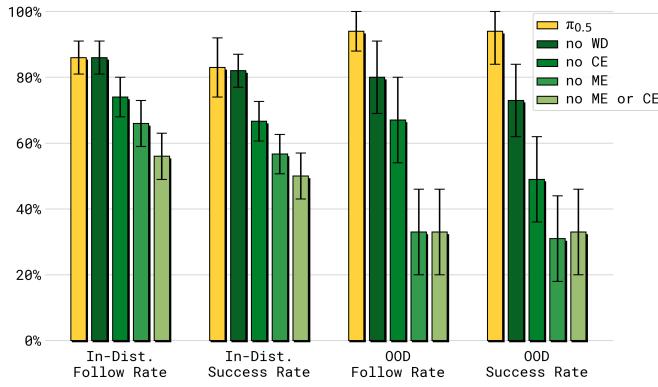


Fig. 11: **Training recipe ablations, language following.** Evaluating language following with in-distribution and out-of-distribution objects after training on different numbers of locations. Including web data (WD) is important for out-of-distribution (OOD) performance in particular. Cross-embodiment (CE) and diverse environment (ME) data both have a large impact on in-distribution and out-of-distribution performance.

difference in performance with the **no WD** ablation is not statistically significant in this experiment, though we show later that web data has a large impact on language following (below) and high-level subtask inference (Section V-E).

The results of the language following experiment, shown in Figure 11, show a similar trend as Figure 10 — excluding ME or/and CE data leads to a significant degradation in performance. What differs now is that removing web data (**no WD**) causes significantly worse performance on out-of-distribution (OOD) objects — we conjecture that training with web data, which contains very broad knowledge of physical objects, allows the model to understand and follow language commands involving *unseen* object categories.

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

We compare $\pi_{0.5}$ to the original π_0 VLA as well as an improved version of π_0 which we denote as π_0 -FAST+Flow. This version is trained via the joint diffusion and FAST action prediction formulation from Equation (1), but on action data



Fig. 12: **Comparing $\pi_{0.5}$ with other models.** Our full model significantly outperforms both π_0 and π_0 -FAST+ in the mock home test environments.

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 and dexterous mobile manipulation tasks, and the enhancement in π_0 -FAST+Flow brings it as close to $\pi_{0.5}$ as possible. $\pi_{0.5}$ builds on these models with a combination of co-training tasks. For a fair comparison, all models receive the same cross-embodiment robot training set and are trained for a comparable number of steps. The differences then are: (1) $\pi_{0.5}$ additionally uses HL and WD data; (2) $\pi_{0.5}$ uses a hybrid training procedure, with discrete tokenized training in the pre-training phase, and training with a flow matching action expert *only* in the post-training phase, while π_0 always uses the action expert. π_0 -FAST+Flow follows the hybrid training recipe but is trained only with data containing robot actions and thus cannot perform high-level inference. The results in Figure 12 show that $\pi_{0.5}$ significantly outperforms both π_0 and our enhanced version. This result holds even when we allow for longer training up to 300k training steps of π_0 , confirming that as in Pertsch et al. [64] training with FAST tokens is more effective in terms of compute than pure diffusion based training.

E. How important is high-level inference?

Finally, 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”), which is then used as context for inferring the lower-level actions, analogously to chain of thought inference [82]. 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 [92, 8], 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.

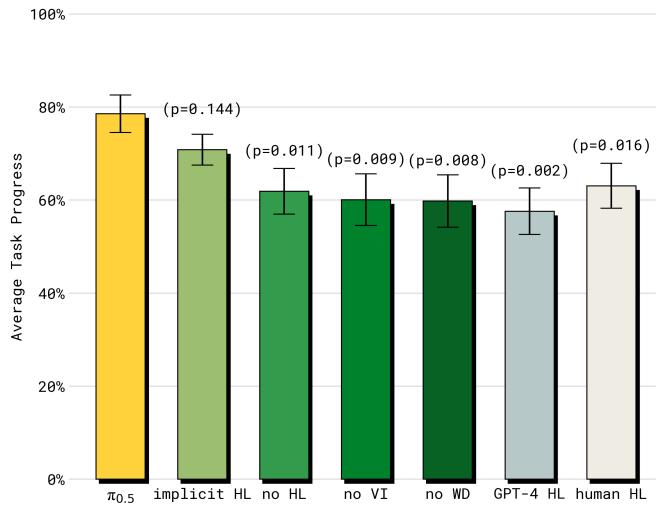


Fig. 13: **Evaluation of the high-level inference process.** While the full $\pi_{0.5}$ model with high-level and low-level inference attains the best results, using only low-level inference (“implicit HL”) with the full $\pi_{0.5}$ model also benefits from the inclusion of high-level subtask examples 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.

- 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 human as an “oracle” high-level policy, to provide an upper bound on performance.

The results of these experiments are shown in Figure 13. 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 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 D, Figure 17.

VI. DISCUSSION AND FUTURE WORK

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, enabling highly generalizable control of a mobile manipulator with only a medium-sized mobile manipulation dataset.

$\pi_{0.5}$ 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.

ACKNOWLEDGEMENTS

We thank our robot operators for data collection, evaluations, logistics, and video recording. See Appendix A for a full contributions statement.

REFERENCES

- [1] AgiBot-World-Contributors, Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, Shu Jiang, Yuxin Jiang, Cheng Jing, Hongyang Li, Jialu Li, Chiming Liu, Yi Liu, Yuxiang Lu, Jianlan Luo, Ping Luo, Yao Mu, Yuehan Niu, Yixuan Pan, Jiangmiao Pang, Yu Qiao, Guanghui Ren, Cheng Ruan, Jiaqi Shan, Yongjian Shen, Chengshi Shi, Mingkang Shi, Modi Shi, Chonghao Sima, Jianheng Song, Huijie Wang, Wenhao Wang, Dafeng Wei, Chengan Xie, Guo Xu, Junchi Yan, Cunbiao Yang, Lei Yang, Shukai Yang, Maoqing Yao, Jia Zeng, Chi Zhang, Qinglin Zhang, Bin Zhao, Chengyue Zhao, Jiaqi Zhao, and Jianchao Zhu. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.
- [2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.
- [3] Suneel Belkhale and Dorsa Sadigh. Minivla: A better vla with a smaller footprint, 2024. URL <https://github.com/Stanford-ILIA/opencvla-mini>.
- [4] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debiddatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language, 2024. URL <https://arxiv.org/abs/2403.01823>.
- [5] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.
- [6] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash 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.
- [7] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.
- [8] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.
- [10] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.
- [11] Hongyi Chen, Yunchao Yao, Ruixuan Liu, Changliu Liu, and Jeffrey Ichnowski. Automating robot failure recovery using vision-language models with optimized prompts. *arXiv preprint arXiv:2409.03966*, 2024.
- [12] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.
- [13] An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Bıyık, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv*

- preprint arXiv:2412.04453*, 2024.
- [14] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.
- [15] OX-Embodiment 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.
- [16] Yinpei Dai, Jayjun Lee, Nima Fazeli, and Joyce Chai. Racer: Rich language-guided failure recovery policies for imitation learning. *International Conference on Robotics and Automation (ICRA)*, 2025.
- [17] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *CoRL*, 2019.
- [18] Sudeep Dasari, Mohan Kumar Srirama, Unnat Jain, and Abhinav Gupta. An unbiased look at datasets for visuomotor pre-training. In *Conference on Robot Learning*, pages 1183–1198. PMLR, 2023.
- [19] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art multimodal models. *arXiv preprint arXiv:2409.17146*, 2024.
- [20] Dempsey. Reviews-consumer technology. the teardown-amazon astro consumer robot. *Engineering & Technology*, 18(2):70–71, 2023.
- [21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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.
- [22] Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *Conference on Robot Learning*, 2024.
- [23] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palme: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.
- [24] Jiafei Duan, Wentao Yuan, Wilbert Pumacay, Yi Ru Wang, Kiana Ehsani, Dieter Fox, and Ranjay Krishna. Manipulate-anything: Automating real-world robots using vision-language models. *arXiv preprint arXiv:2406.18915*, 2024.
- [25] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.
- [26] Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Luca Weihs, Kuo-Hao Zeng, Kunal Pratap Singh, Yejin Kim, Winson Han, Alvaro Herrasti, et al. Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. *arXiv preprint arXiv:2312.02976*, 2023.
- [27] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.
- [28] Haritheja Etukuru, Norihito Naka, Zijin Hu, Seungjae Lee, Julian Mehu, Aaron Edsinger, Chris Paxton, Soumith Chintala, Lerrel Pinto, and Nur Muhammad Mahi Shafullah. Robot utility models: General policies for zero-shot deployment in new environments. *arXiv preprint arXiv:2409.05865*, 2024.
- [29] Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhui Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 39(5):3929–3945, 2023.
- [30] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu 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.
- [31] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *Science Robotics*, 8(79):eadf6991, 2023.
- [32] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi 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.
- [33] Abhinav Gupta, Adithyavairavan Murali, Dhiraj Prakashchand Gandhi, and Lerrel Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. *Advances in neural information processing systems*, 31, 2018.
- [34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [35] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15979–15988, 2022.
- [36] Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-

- 4v in robotic vision-language planning. *arXiv preprint arXiv:2311.17842*, 2023.
- [37] Huang Huang, Fangchen Liu, Letian Fu, Tingfan Wu, Mustafa Mukadam, Jitendra Malik, Ken Goldberg, and Pieter Abbeel. Otter: A vision-language-action model with text-aware visual feature extraction. *arXiv preprint arXiv:2503.03734*, 2025.
- [38] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR, 2022.
- [39] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.
- [40] Joseph L Jones. Robots at the tipping point: the road to irobot roomba. *IEEE Robotics & Automation Magazine*, 13(1):76–78, 2006.
- [41] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, 2024.
- [42] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [43] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.
- [44] Boyi Li, Philipp Wu, Pieter Abbeel, and Jitendra Malik. Interactive task planning with language models, 2023.
- [45] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.
- [46] Xiang Li, Cristina Mata, Jongwoo Park, Kumara Khatapitiya, Yoo Sung Jang, Jinghuan Shang, Kanchana Ranasinghe, Ryan Burgert, Mu Cai, Yong Jae Lee, et al. Llara: Supercharging robot learning data for vision-language policy. *arXiv preprint arXiv:2406.20095*, 2024.
- [47] Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, et al. Hamster: Hierarchical action models for open-world robot manipulation. *arXiv preprint arXiv:2502.05485*, 2025.
- [48] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy 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.
- [49] Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. *arXiv preprint arXiv:2410.18647*, 2024.
- [50] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.
- [51] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey 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.
- [52] Jiaming Liu, Hao Chen, Pengju An, Zhuoyang Liu, Renrui Zhang, Chenyang Gu, Xiaoqi Li, Ziyu Guo, Sixiang Chen, Mengzhen Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.
- [53] Peiqi Liu, Yaswanth Orru, Jay Vakil, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Ok-robot: What really matters in integrating open-knowledge models for robotics. *arXiv preprint arXiv:2401.12202*, 2024.
- [54] Qiang Liu. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022.
- [55] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun

- Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.
- [56] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken 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.
- [57] Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Tingfan Wu, Jay 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.
- [58] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *CORL*, 2022.
- [59] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.
- [60] Hai Nguyen and Charles C Kemp. Autonomously learning to visually detect where manipulation will succeed. *Autonomous Robots*, 36:137–152, 2014.
- [61] Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. Llarva: Vision-action instruction tuning enhances robot learning. *arXiv preprint arXiv:2406.11815*, 2024.
- [62] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.
- [63] Open X-Embodiment Collaboration, Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, Antonin Raffin, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Brian Ichter, Cewu Lu, Charles Xu, Chelsea Finn, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Chuer Pan, Chuyuan Fu, Coline Devin, Danny Driess, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Federico Ceola, Fei Xia, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Giulio Schiavi, Hao Su, Hao-Shu Fang, Haochen Shi, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homer Walke, Hongjie Fang, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jaehyun Kim, Jan Schneider, Jasmine Hsu, Jeannette Bohg, Jeffrey Bingham, Jiajun Wu, Jialin Wu, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jitendra Malik, Jonathan Tompson, Jonathan Yang, Joseph J. Lim, João Silvério, Junhyek Han, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Zhang, Keyvan Majd, Krishan Rana, Krishnan Srinivasan, Lawrence Yunliang Chen, Lerrel Pinto, Liam Tan, Lionel Ott, Lisa Lee, Masayoshi Tomizuka, Maximilian Du, Michael Ahn, Mingtong Zhang, Mingyu Ding, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Norman Di Palo, Nur Muhammad Mahi Shafullah, Oier Mees, Oliver Kroemer, Pannag R Sanketi, Paul Wohlhart, Peng Xu, Pierre Sermanet, Priya Sundaresan, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Sherry Moore, Shikhar Bahl, Shivin Dass, Shuran Song, Sichun Xu, Siddhant Haldar, Simeon Adebola, Simon Guiist, Soroush Nasiriany, Stefan Schaaf, Stefan Welker, Stephen Tian, Sudeep Dasari, Suneel Belkhale, Takayuki Osa, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Vidhi Jain, Vincent Vanhoucke, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiaolong Wang, Xinghao Zhu, Xuanlin Li, Yao Lu, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yueh hua Wu, Yujin Tang, Yuke Zhu, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zhuo Xu, and Zichen Jeff Cui. Open X-Embodiment: Robotic learning datasets and RT-X models. <https://arxiv.org/abs/2310.08864>, 2023.
- [64] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. *Robotics: Science and Systems*, 2025.
- [65] Dicong Qiu, Wenzong Ma, Zhenfu Pan, Hui Xiong, and Junwei Liang. Open-vocabulary mobile manipulation in unseen dynamic environments with 3d semantic maps. *arXiv preprint arXiv:2406.18115*, 2024.
- [66] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [67] Nur Muhammad Mahi Shafullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023.
- [68] Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023.

- [69] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A foundation model for visual navigation. In *7th Annual Conference on Robot Learning*, 2023. URL <https://arxiv.org/abs/2306.14846>.
- [70] Rutav Shah, Albert Yu, Yifeng Zhu, Yuke Zhu, and Roberto Martín-Martín. Bumble: Unifying reasoning and acting with vision-language models for building-wide mobile manipulation. *arXiv preprint arXiv:2410.06237*, 2024.
- [71] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024.
- [72] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolò Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.
- [73] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh 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.
- [74] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Brianna Zitkovich, Fei Xia, Chelsea Finn, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.
- [75] Andrew Szot, Bogdan Mazoure, Omar Attia, Aleksei Timofeev, Harsh Agrawal, Devon Hjelm, Zhe Gan, Zsolt Kira, and Alexander Toshev. From multimodal llms to generalist embodied agents: Methods and lessons. *arXiv preprint arXiv:2412.08442*, 2024.
- [76] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.
- [77] Peter Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Adithya Jairam Vedagiri IYER, Sai Charitha Akula, Shusheng Yang, Jihan Yang, Manoj Middepogu, Ziteng Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024.
- [78] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [79] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- [80] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. BridgeData v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.
- [81] Shu Wang, Muzhi Han, Ziyuan Jiao, Zeyu Zhang, Ying Nian Wu, Song-Chun Zhu, and Hangxin Liu. Llm³: 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.
- [82] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [83] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, Yixin Peng, Feifei Feng, and Jian Tang. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2409.12514*, 2024.
- [84] Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.
- [85] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.
- [86] Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, et al. Magma: A foundation model for multimodal ai agents. *arXiv preprint arXiv:2502.13130*, 2025.
- [87] Qiying Yu, Quan Sun, Xiaosong Zhang, Yufeng Cui, Fan Zhang, Yue Cao, Xinlong Wang, and Jingjing 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.
- [88] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *Conference on Robot Learning*, 2024.
- [89] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *Computer Vision and Pattern Recognition (CVPR)*, 2025.
- [90] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: 3d vision-language-action generative world model.

arXiv preprint arXiv:2403.09631, 2024.

- [91] Peiyuan Zhi, Zhiyuan Zhang, Yu Zhao, Muzhi Han, Zeyu Zhang, Zhitian Li, Ziyuan Jiao, Baoxiong Jia, and Siyuan Huang. Closed-loop open-vocabulary mobile manipulation with gpt-4v. *arXiv preprint arXiv:2404.10220*, 2024.
- [92] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan 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.

APPENDIX

A. Contributions

Data collection and operations. Noah Brown, Michael Equi, Chelsea Finn, Lachy Groom, Suraj Nair, Lucy Xiaoyang Shi, Anna Walling.

Annotation and supplemental data. Danny Driess, Chelsea Finn, Niccolo Fusai, Lachy Groom, Brian Ichter, Karl Pertsch, Allen Z. Ren, Laura Smith, Kyle Stachowicz, Quan Vuong, Anna Walling, Lili Yu.

Policy training and research. Kevin Black, Danny Driess, Michael Equi, Chelsea Finn, Niccolo Fusai, Dibya Ghosh, Brian Ichter, Liyiming Ke, Sergey Levine, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Quan Vuong, Homer Walke, Lili Yu.

Policy infrastructure. Kevin Black, Karan Dhabalia, Danny Driess, Manuel Y. Galliker, Dibya Ghosh, Adrian Li-Bell, Quan Vuong, Haohuan Wang, Ury Zhilinsky.

Robot hardware. Noah Brown, Adnan Esmail, Tim Jones, Devin LeBlanc, Mohith Mothukuri.

Robot infrastructure. James Darpinian, Adnan Esmail, Manuel Y. Galliker, Karol Hausman, Szymon Jakubczak, James Tanner.

Writing and illustration. Kevin Black, Danny Driess, Chelsea Finn, Karol Hausman, Brian Ichter, Sergey Levine, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Jost Tobias Springenberg.

B. 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 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.

C. 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 objects. We show the example initial scene of each task in Figure 14.

Along with data ablation experiments in Figure 11 and location scaling experiments in Figure 9, 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, and a much higher rate than π_0 , indicating the importance of discrete token training on language following abilities.



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

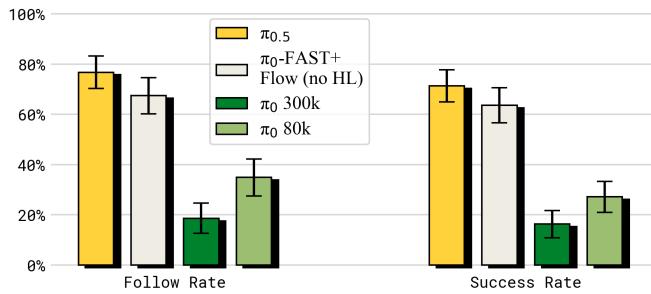


Fig. 15: Comparing $\pi_{0.5}$ with other models on language following. We evaluate language following capabilities of $\pi_{0.5}$, π_0 , and π_0 -FAST+Flow, finding $\pi_{0.5}$ outperforms each, and π_0 by a wide margin.

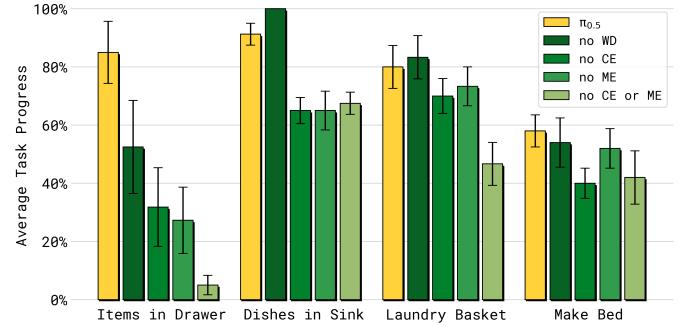


Fig. 16: Per-task performance breakdown for training recipe ablations. We evaluate each training mixture variant on four representative household tasks: *Items in Drawer*, *Dishes in Sink*, *Laundry Basket*, and *Make Bed*. Removing cross-embodiment data (ME or CE) leads to significant degradation in 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.

D. Per-task performance breakdown

a) *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 16). 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.

b) *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 17). 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.

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

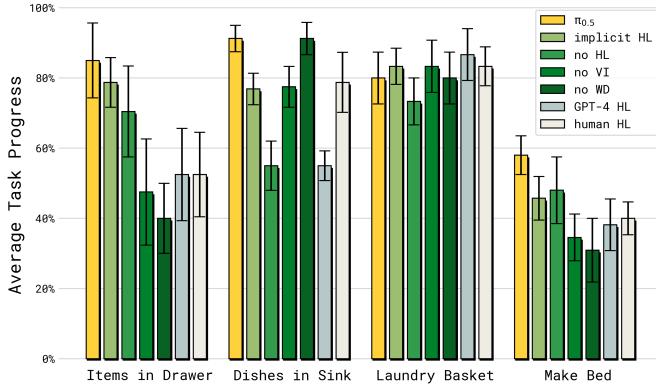


Fig. 17: Per-task performance breakdown for high-level inference methods. We evaluate the full $\pi_{0.5}$ model and various high-level inference baselines across four representative household tasks.

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.

E. Model technical details

The $\pi_{0.5}$ model builds upon π_0 and adopts the PaliGemma VLM [5] 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 $[I_t^1, \dots, I_t^n]$ and a language prompt ℓ as in π_0 , but also the robot’s proprioceptive state q_t in tokenized form and tokenized actions [64], which will be auto-regressively predicted. The action expert is a smaller transformer that takes in a sequence of noisy action tokens $a_{t:t+H}^{\tau, \omega}$ 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 $\phi : \mathbb{R} \rightarrow \mathbb{R}^w$ is a sinusoidal positional encoding function [79] 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 from PaliGemma weights, and the same except for $\{width=1024, mlp_dim=4096\}$ for the action expert with 300M₆ parameters.

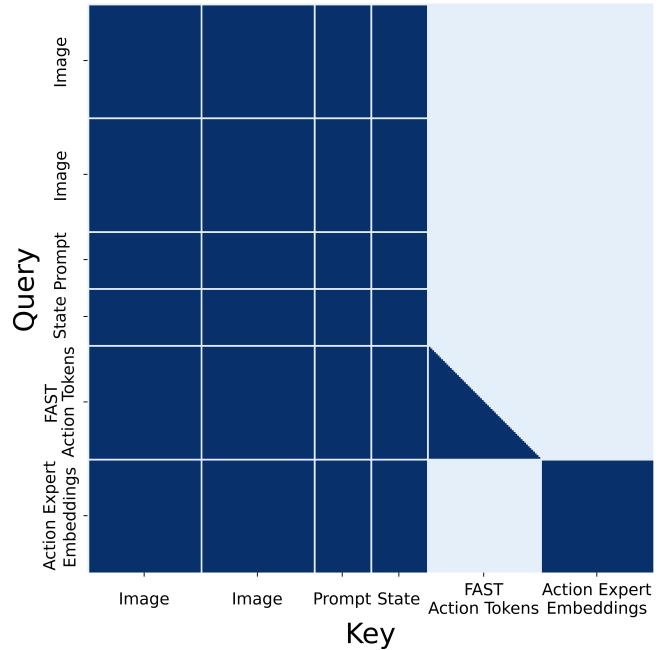


Fig. 18: Example of the $\pi_{0.5}$ attention masking pattern.

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 18.

We follow π_0 for sampling the flow-matching timestep τ . In summary we deviate from standard uniform sampling $\tau \sim \mathcal{U}(0, 1)$ [50, 54] or methods emphasizing midrange timesteps [27], and instead use a time-step sampling distribution that emphasizes low time-steps [8], 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),
]
```