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

Yell At Your Robot 🗣️
Improving On-the-Fly from Language Corrections

Lucy Xiaoyang Shi[1] Zheyuan Hu[2] Tony Z. Zhao[1] Archit Sharma[1]
Karl Pertsch[1,2] Jianlan Luo[2] Sergey Levine[2] Chelsea Finn[1]
[1]Stanford University [2]UC Berkeley

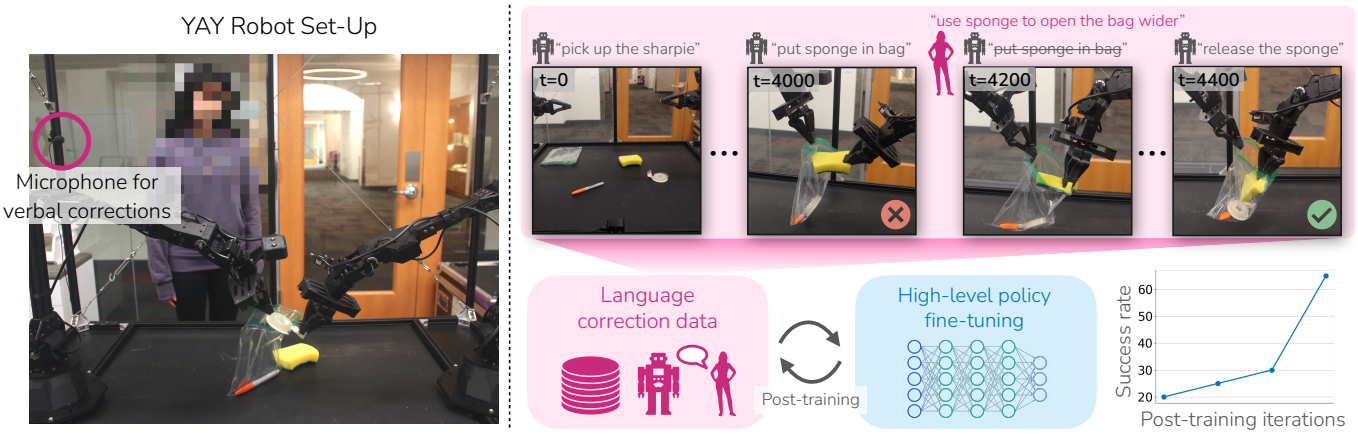


Fig. 1: Our approach enables robots to leverage verbal corrections to improve performance on complex long-horizon tasks like packing a ziploc bag and preparing trail-mix. It can incorporate verbal corrections in real-time (top) and for continuous improvement (bottom).

***Abstract*—Hierarchical policies that combine language and low-level control have been shown to perform impressively long-horizon robotic tasks, by leveraging either zero-shot high-level planners like pretrained language and vision-language models (LLMs/VLMs) or models trained on annotated robotic demonstrations. However, for complex and dexterous skills, attaining high success rates on long-horizon tasks still represents a major challenge – the longer the task is, the more likely it is that some stage will fail. In principle, a robust high-level controller can compensate for low-level failures by dynamically deploying corrections and adjustments, but training such high-level controllers in a way that is aware of the physical capabilities of the low-level skills requires costly demonstrations of entire multi-stage tasks. Can humans help the robot to continuously improve its long-horizon task performance through intuitive and natural feedback? In this paper, we make the following observation: high-level policies that index into sufficiently rich and expressive low-level language-conditioned skills can be readily supervised with human feedback in the form of language corrections. We show that even fine-grained corrections, such as small movements ("move a bit to the left"), can be effectively incorporated into high level policies, and that such corrections can be readily obtained from humans observing the robot and making occasional suggestions. This framework enables robots not only to rapidly adapt to real-time language feedback, but also incorporate this feedback into an iterative training scheme that improves the high-level policy's ability to correct errors in both low-level execution and high-level decision-making purely from verbal feedback. Our evaluation on real hardware shows that this leads to significant performance improvement in long-horizon, dexterous manipulation tasks without the need for any additional teleoperation. Videos and code are available at https://yay-robot.github.io/.**

I. Introduction

Complex robotic tasks may require sequencing multiple individual primitives. For example, packing multiple items into a bag, as shown in Figure 1, requires grasping each object in turn, maneuvering it near the bag opening, and inserting it. An appealing framework for addressing such multi-stage tasks is via *hierarchical abstraction*, where a high-level policy commands specific behaviors that are then performed by the low-level policy [36, 68, 69, 18]. One intuitive method for parameterizing such policies is via language, with a high-level policy selecting among possible language instructions at each stage [8, 1]. Unfortunately, as the number of stages in a task increases, there are more points of failure – if every stage needs to succeed, the overall probability of failure goes up exponentially. However, a robust high-level policy can compensate for low-level failures, deploying corrections and adjustments as needed. Therefore, successful completion of such multi-stage tasks depends critically on the ability to train such high-level policies in a way that is robust, aware of the limitations of the low-level primitives, and well adapted to the dynamics of the problem.

Unfortunately, this is difficult to do in a scalable way. Complete end-to-end on-robot demonstrations of long-horizon tasks provide "gold standard" supervision, because they allow the high level to be fully aware of the intricacies of low-level control, but they are costly and time consuming to collect at scale because the tasks are so long. Knowledge transfer

from other sources, such as large language models (e.g., as in works that use LLMs or VLMs for planning [1, 24, 23]) provides an appealing alternative, but this knowledge is not grounded in robotic behaviors, leaving the high-level policy relatively brittle because it does not know which skills are more or less effective *for this particular robot and in this particular situation*. Other modes of indirect supervision, such as language-only supervision [17] or human videos [44, 5, 58] similarly provide indirect supervision. Therefore, the challenge of training robust high-level policies can be seen as the challenge of obtaining scalable and high-quality training data for the high level.

What if instead of requiring extensive demonstration supervision for such high-level policies, we could instead train them with natural feedback from humans in the form of language corrections? Such feedback is natural for humans to provide, and might even be gathered naturally in the course of the robot's day-to-day work, as it attempts the desired tasks and receives feedback from human users. Our key insight in this work is that this sort of language feedback can be readily incorporated into hierarchical policies when the high-level policy outputs language commands, and in fact this procedure can be seen as a high-level analogue of the widely used DAgger [52, 28] algorithm – a technique where the robot iteratively incorporates feedback from a human supervisor, typically in the form of low-level motor actions – but over the action space of a high-level instruction policy, i.e., over language instructions. By incorporating language *corrections*, the high-level policy is equipped to correct mistakes made by both the low-level policy and the high-level policy. This facilitates robust recovery from failures, which is critical for the success in long-horizon tasks.

Based on this insight, we propose Yell At Your Robot (YAY Robot), a system for improving robot post-training through natural language feedback. Unlike previous efforts that focus on post-hoc action relabeling or one-time correction [35, 12], YAY Robot aims for a more organic integration of language into policy improvement through: (a) on-the-fly adaptation to language feedback, and (b) continuous improvement from user interaction.

First, we enable robots to adapt their behaviors to diverse, often contextual language commands in real time. We use an end-to-end Language-Conditioned Behavior Cloning (LCBC) policy to learn a variety of skills specified through language within a single neural network, facilitating versatile low-level behaviors in response to diverse language inputs. As a result, users can interact with our system using free-form language, ranging from low-level commands like "move the sponge a little lower" or "turn your right gripper towards me" to high-level preferences such as "I want more M&M's in my trail mix bag".

Moreover, robots should improve from user feedback over time, avoiding perpetual corrections and learning to handle complex, long-horizon tasks more effectively without frequent interventions. Hence, we learn a high-level policy to mimic human instruction patterns and assimilate human feedback. As illustrated in Figure 1, our high-level policy generates language instructions for the low-level LCBC policy autonomously. When users choose to intervene, their commands temporarily override the high-level policy and directly feed into the low-level policy, allowing for immediate adaptation from the robot. After user interactions, the high-level policy is finetuned on human language interventions to enhance its ability to predict better instructions and corrections in the future.

The primary contribution of our work is YAY Robot, a learning-based system for long-horizon tasks that allows robots to both (a) incorporate language corrections on-the-fly and (b) continuously improve from this feedback. In our experiments, we consider three bi-manual multi-stage manipulation tasks: packing three items into a ziploc bag, making a bag of trail mix by scooping ingredients, and cleaning gummies off a plate. By leveraging human corrections on-the-fly, our approach leads to real-time improvement from 15% to 50%. Moreover, incorporating these corrections into high-level policy training increases performance from 15% to 45%. Crucially, YAY Robot allows users to guide robots naturally and intuitively, representing a step towards enabling end-users to directly teach robots through natural language in everyday scenarios.

II. Related Work

The idea of learning a single policy to complete several tasks has a rich history in robotics [13, 29, 14, 48, 62, 4, 42]. As the number of behaviors increases, natural language provides a simple and interpretable abstraction to index these behaviors, and thus, several robotic systems map natural language instructions to actions [40, 30, 43, 45, 59, 37, 38, 56]. This idea has been explored in a rich body of work spanning semantic parsing for robotics [31, 63, 66] and task and motion planning (TAMP) [25, 20, 15]. More recently, the advent of large robotic interaction datasets has enabled learning language-conditioned multi-task policies [26, 8, 9] that can generalize to novel scenes, objects and even natural language instructions.

Representing the low-level behaviors using natural language has several desirable properties for high-level decision making. First, natural language offers a compact and compositional representation, that can allow combinatorial generalization [57, 3, 27]. This makes it a compelling abstraction to tackle long-horizon tasks [27, 54, 19, 2], especially with the increasing use of large language models for high-level planning [1, 23, 32, 24, 65]. Second, language provides a natural interface for humans to guide, correct, and improve a robot by simply *talking* to it. Human-in-the-loop imitation learning techniques often require people to intervene and correct robot behaviors using teleoperation or kinesthetic teaching [52, 28, 41, 22, 21, 34]. Natural language is more accessible in contrast, and recent works have shown that it can enable humans to interact and guide robots in real-time [39], provide on-the fly corrections for both high-level plans [7, 55] or low-level behaviors [10, 12] in a shared autonomy setup, modify the code [33], or even update low-level behaviors post-hoc by converting language corrections into new target actions [35]. Our work combines the benefits of both these properties; YAY Robot learns a

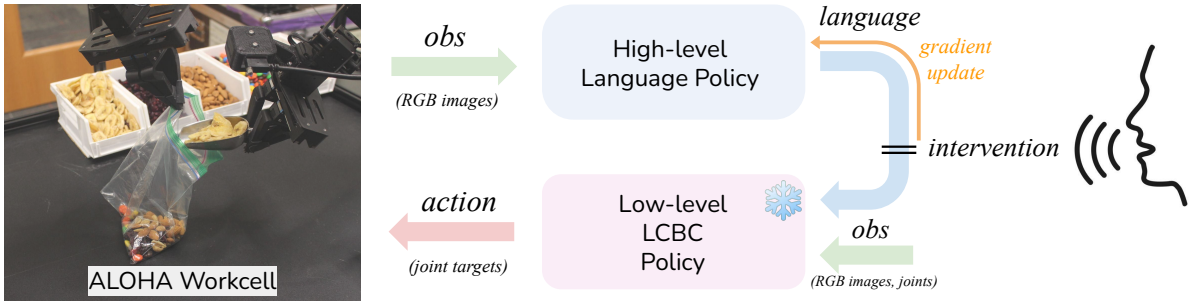


Fig. 3: **Overview of YAY Robot.** We operate in a hierarchical setup where a high-level policy generates language instructions for a low-level policy that executes the corresponding skills. During deployment, humans can intervene through corrective language commands, temporarily overriding the high-level policy and directly influencing the low-level policy for on-the-fly adaptation. These interventions are then used to finetune the high-level policy, improving its future performance.

language-conditioned low-level policy and trains a high-level policy to output natural language instructions to combine low-level skills for long-horizon tasks. Further, YAY Robot can fine-tune the high-level policy after deployment with verbal corrections, leading to consistent improvements in performance on hard long-horizon tasks. In contrast to OLAF [35], YAY Robot can modify robot behaviors on-the-fly from human language interventions and can learn end-to-end from raw pixel observations without any explicit state estimation. And in contrast to other works [7, 10, 12, 67] that are designed for shared autonomy, and thus require humans to provide corrections perpetually, YAY Robot is designed to operate autonomously while still being able to improve from verbal corrections when provided. One concurrent work, RT-H [6], shares a similar idea of learning from language corrections. However, RT-H's correction is limited to a fixed set of spatial movements, whereas ours allows flexible, diverse user input such as "use the sponge to open the bag wider" to tackle long-horizon, bi-manual dexterous manipulation tasks.

III. Yell At Your Robot

In this section, we describe each component of our system that enables on-the-fly and continual improvement from language corrections. We will first describe the overall problem set-up and define notation before detailing the low-level and high-level policy components. Last, but most importantly, we will describe how we integrate verbal corrections into the framework.

A. Preliminaries

We formulate our robot manipulation task as a Markov Decision Process (MDP), denoted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, p_0)$, where $\mathcal{S}$ represents the state space, $\mathcal{A}$ the action space, $\mathcal{P}$ the transition probabilities, and p_0 the initial state distribution. The robot does not have access to the true states $s \in \mathcal{S}$; instead, it receives observations $\mathcal{O}$ that are partially observed.

For training the robot's high-level policy π_{H} and low-level policy π_{L}, we utilize a base dataset $\mathcal{D}$, composed of sequences (o_t, a_t, l_t), where o_t represents the observation, including RGB images and robotic proprioceptive information, a_t is the robot's action, and l_t is a language instruction given at time t. This dataset is assumed to cover a broad range of visuomotor skills necessary for completing tasks, alongside various mistakes and recovery behaviors.

After the initial training phase, we finetune the robot's high-level policy to align it with human verbal feedback. This is done using a correction dataset $\mathcal{D}_{\text{corr}}$ consisting of online user interaction data, (o_t, l^{user}) – data that is naturally produced in the course of humans interacting with robots. Unlike the base dataset, $\mathcal{D}_{\text{corr}}$ omits a_t because we keep the low-level policy frozen during this post-training phase and only update the parameters of the high-level policy.

B. Low-Level Language-Conditioned Behavior Cloning

The low-level policy enables the robot to interpret and execute a wide array of skills articulated through natural language commands. Learning an expanded set of skills that are not just the minimal set for completing the task provides more flexibility, as it allows the high-level policy to have the latitude to orchestrate skills that correct for previous mistakes, and the low-level policy to accommodate these corrections and adjustments on-the-fly. Motivated by the need for such flexibility, the low-level policy is implemented as a deep neural network trained end-to-end on datasets encompassing diverse visuomotor skills, ranging from task-centric instructions, such as "pour into the bag", to task-agnostic corrections like "move the left arm towards me".

The low-level policy, $\pi_{\text{L}}(a_t|o_t, l_t)$, maps the current observation o_t and language instruction l_t to action a_t. The policy is trained using the standard Behavior Cloning (BC) objective:

$$\min_{\pi_{\text{L}}} \mathbb{E}_{(o_t, l_t, a_t)\sim\mathcal{D}} \left[\mathcal{L}_{\text{BC}}(\pi_{\text{L}}(a_t|o_t, l_t), a_t)\right] \tag{1}$$

where $\mathcal{L}_{\text{BC}}$ is a loss function (e.g., ℓ_1 or ℓ_2) comparing the predicted continuous action to the ground truth action.

C. High-Level Policy for Autonomous Instruction Generation

A hierarchical setup could allow the robot to reuse primitive skills. Therefore, we learn a high-level policy to generate language instructions that guide the low-level policy. This

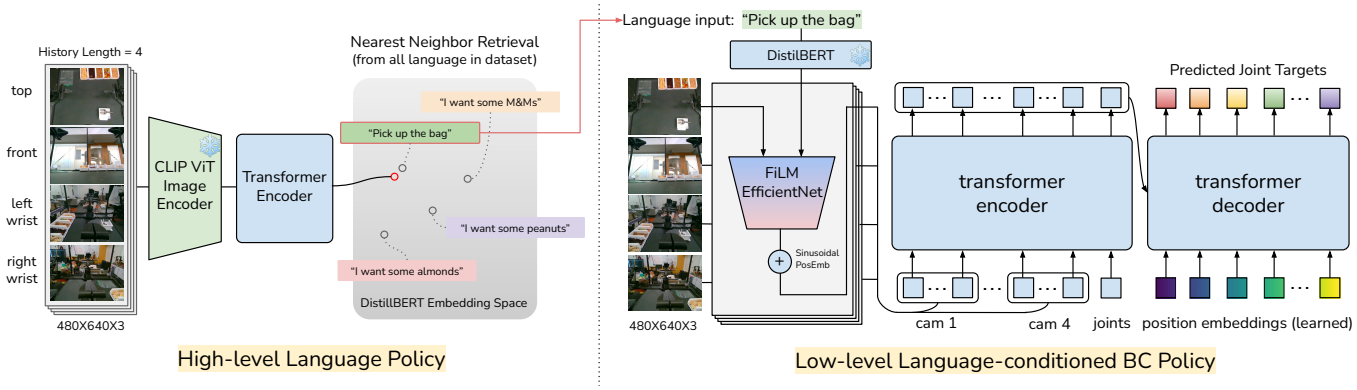


Fig. 4: **Policy Architecture.** Our system processes RGB images and the robot's current joint positions as inputs, outputting target joint positions for motor actions. The high-level policy uses a Vision Transformer to encode visual inputs and predicts language embeddings. The low-level policy uses ACT, a Transformer-based model to generate precise motor actions for the robot, guided by language instructions. This architecture enables the robot to interpret commands like "Pick up the bag" and translate them into targeted joint movements.

policy is also trained using behavior cloning to predict language instructions conditioned on the current observation. Similar to a vision-language model (VLM), it processes image observations and outputs language embeddings. Since the same observation can lead to different commands (e.g., "tilt the scoop down" and "go a bit higher" are both reasonable when the robot is close to food containers), we contextualize the instructions by conditioning on a brief history of observations.

In our approach, the high-level policy π_{H} outputs a language embedding for a given temporal context of observations, $(o_{t-k:t})$. To train this policy, we first compute the cosine similarity between the predicted language embedding and each language command in the training dataset. These similarity scores are transformed into logits:

$$\text{logits} = \text{cosine\_similarity}(\pi_{\mathrm{H}}(l_t|o_{t-k:t}), \text{all\_embeddings})$$

A softmax function with a learned temperature parameter, τ [47], is then applied to these logits to obtain a probability distribution over all possible language commands:

$$P(l_t|o_{t-k:t}) = \text{softmax}\left(\frac{\text{logits}}{\tau}\right)$$

This probability distribution is utilized in the cross-entropy loss function $\mathcal{L}_{\mathrm{CE}}$ to optimize π_{H}, minimizing the discrepancy between the predicted and true language commands:

$$\min_{\pi_{\mathrm{H}}} \mathbb{E}_{(o_{t-k:t},l_t)\sim\mathcal{D}} \left[\mathcal{L}_{\mathrm{CE}}(\pi_{\mathrm{H}}(l_t|o_{t-k:t}), l_t)\right] \tag{2}$$

During inference, the model identifies the most appropriate language command by selecting the nearest neighbor. This is determined by the highest cosine similarity score between the predicted language embedding and those in the dataset. This method ensures that the output is a valid and contextually relevant instruction based on the observed scenario.

D. Policy Integration and Adaptation to Human Feedback

YAY Robot integrates these policies to create a cohesive system capable of adapting to real-time language feedback. As shown in Figure 3, the high-level policy generates language instructions for the low-level policy, which executes the corresponding skills. During deployment, the human may intervene to correct erroneous behaviors or indicate preferences by providing a corrective language command. The user's verbal interventions temporarily override the high-level policy's output, directly influencing the low-level policy to enable on-the-fly adaptation:

$$\pi_{\text{deploy}}(a_t|o_{t-k:t}) = \begin{cases} \pi_{\mathrm{L}}(a_t|o_t, l^{\text{user}}) & \text{if intervention} \\ \pi_{\mathrm{L}}(a_t|o_t, l^{\mathrm{H}}) & \text{otherwise} \end{cases} \tag{3}$$

where l^{user} denotes the language command provided by the user, and $l^{\mathrm{H}} = \arg\max \pi_{\mathrm{H}}(\cdot \mid o_{t-k:t})$. This intervention is recorded as a new data point $(o_{t-k:t}, l^{\text{user}})$ in a correction dataset $\mathcal{D}_{\text{corr}}$, which is subsequently used to finetune π_{H}.

E. Continuous Improvement from Human Feedback

Building upon the real-time adaptation capability, YAY Robot aims to learn continuously to minimize the need for perpetual corrections and better align with user preferences over time. The continuous improvement of YAY Robot is driven by incorporating human language feedback into the high-level policy. This process is conceptually similar to performing Human-Gated DAgger (HG-DAgger [28]) on the high-level policy, whose actions are language commands. Crucially, unlike typical uses of DAgger and HG-DAgger, interventions are only provided in *natural language* rather than low-level robot actions. The low-level policy is kept frozen throughout this post-training process.

1) Finetuning High-Level Policy: We finetune the high-level policy on the correction dataset $\mathcal{D}_{\text{corr}}$ along with the base dataset $\mathcal{D}$. The combined dataset $\mathcal{D} \cup \mathcal{D}_{\text{corr}}$ enhances the policy's exposure to diverse scenarios that result from mistakes by either the low-level or high-level policy, both of

which necessitate corrections. This process aligns the policy's predictions with both the initial training instructions and the human corrective feedback. The optimization objective is the same as in Equation (2).

2) Iterative Improvement: After each iteration of user interaction and feedback collection in Section III-D, we fine-tune the policy to reflect the new data. This iterative process can be depicted as:

$$\pi_{\text{H}}^{(n+1)} = \text{Post-Training}(\pi_{\text{H}}^{(n)}, \mathcal{D} \cup \bigcup_{i=1}^{n} \mathcal{D}_{\text{corr}}^{(i)}) \tag{4}$$

Here, $\pi_{\text{H}}^{(n)}$ and $\pi_{\text{H}}^{(n+1)}$ denote the high-level policy before and after the n-th iteration of finetuning, respectively. $\mathcal{D}_{\text{corr}}^{(n)}$ is the dataset of corrective feedback obtained at the n-th iteration. This fine-tuning process ensures that YAY Robot progressively improves in handling complex tasks autonomously without frequent interventions.

IV. Practical Instantiation and Implementation

This section outlines our implementation, focusing on data collection, policy architecture, and the post-training procedure.

A. Pre-Training Data Collection and Processing

1) Language Annotation: We need language-annotated robotic data to pre-train the base policy. Traditional methods involve post-hoc language annotation, where operators watch robot videos and annotates each skill segment with start and end timestamps. However, this process is laborious, particularly for long-horizon tasks involving numerous skill segments.

To streamline this process, we adopt a more efficient data collection method: live narration. By placing a microphone near the robot, the operator can narrate the skill they are performing in real-time. They first verbalize the intended skill and then teleoperate the robot to perform it. The recorded audio is then transcribed into text using the Whisper [51] model and synchronized with the robot's trajectory. Our code, which automates this process, is open-sourced to facilitate research that involves language-annotated data collection.

2) Filtering Mistakes: Our data includes both successful executions and errors which lead to subsequent recoveries. If trained on all the data, a model might learn to intentionally make mistakes (Section V-E). An intuitive idea is to filter out the segments preceding corrections, since they are likely erroneous or suboptimal and should be excluded from training. However, identifying errors traditionally requires expensive post-hoc labeling. We simplify this by differentiating instructions from corrections.

To implement this distinction, we use foot pedals during data collection. When narrating a new skill (instruction), the operator steps on the instruction pedal. If a correction is necessary, they step on the correction pedal. This method allows for quick filtering of segments leading up to corrections, ensuring they are not used for training.

3) Collection of Low-Level Correction Skills: We assume the base dataset includes diverse, reusable correction skills. This way, we can efficiently collect verbal corrections during interaction and do not need to collect additional low-level demonstrations for post-training.

Which correction skills would be useful to collect for the base dataset? For dexterous, long-horizon tasks, operators naturally make mistakes, and their recovery behavior informs the collection of correction skills. Nonetheless, the situations where the operator makes mistakes and the robot makes mistakes could be different. Therefore, to ensure the collection of relevant and practical correction skills for robots, we assess policy performance to identify which skills to collect. We train robots using existing data, and during policy rollouts, whenever an operator observes, for example, "I wish I could just tell the robot to adjust its gripper slightly" when the robot behaves suboptimally, such corrections are narrated and included in future training data collection.

B. Low-Level Policy

The low-level policy is designed to perform precise and complex robotic actions based on both visual and language inputs. To achieve this, we use Action Chunking with Transformers (ACT) [70], a state-of-the-art imitation learning architecture that has demonstrated efficacy in handling robotic tasks requiring high precision and robustness [70].

ACT learns a generative model over action sequences ("action chunking"). We modify ACT such that it predicts action chunks based on the current observation and language instruction. The modified training objective for language-conditioned ACT is expressed as follows:

$$\min_{\pi_{\text{L}}} \mathbb{E}_{(o_t, l_t, a_t:a_{t+n})\sim\mathcal{D}} \left[\mathcal{L}_{\text{BC}}(\pi_{\text{L}}(a_t : a_{t+n}|o_t, l_t), a_t : a_{t+n})\right]$$

where $a_t : a_{t+n}$ represents a sequence of actions from time t to $t+n$. The $\mathcal{L}_{\text{BC}}$ is an ℓ_1 loss.

For visual processing, we use EfficientNet b3 [60] as the visual backbone to encode RGB images captured from each camera. In the original ACT model, image features are concatenated with the robot's proprioceptive state information as the input to the policy. To additionally condition on language, we modify ACT by incorporating FiLM [49] layers to fuse the visual and language inputs, similar to RT-1 [8]. We encode language instructions in the dataset using DistilBERT [53].

C. High-Level Policy

The high-level policy generates language commands autonomously. This policy is based on a visual backbone using a Vision Transformer (ViT) [16] initialized with pretrained CLIP [50] weights. These weights are kept frozen throughout training. The visual feature is then processed through additional Transformer [64] and MLP layers to produce a language encoding. For ground truth encoding, we utilize DistilBERT [53] to process language instructions in the dataset.

To account for temporal context, the model employs sinusoidal position embeddings. These are applied to a sequence of historical observations for each camera at every timestep. In

selecting history observations, in case of high-frequency control, the immediate previous image has high similarity to the current frame. Instead of the immediate previous images, we choose up to four images spaced one second apart, providing a broader temporal context.

During training, the output language embedding is compared to existing language commands in the dataset using cosine similarity. This comparison yields logits, which are then evaluated using cross-entropy loss with a learned temperature parameter [47]. To encourage the model to predict upcoming instructions rather than current ones, we offset the target prediction by a small margin to increase the likelihood of the model predicting the next instruction as it nears the end of the current skill segment. At inference time, the most probable language command is selected as the high-level policy output.

D. Post-Training

During post-training, the high-level policy is queried at fixed intervals, specifically every 4 seconds as the average skill length, to generate language instructions for the robot.

To facilitate real-time human feedback, a microphone is placed near the robot to capture verbal commands from users. If the user wishes to intervene, they can verbally instruct the robot to stop with a simple command such as "stop" (or a more polite alternative like "pardon"). Following this interruption, the user can provide verbal correction to guide the robot.

To continually improve the high-level policy, we record the verbal corrections provided by the user along with the corresponding observations. Additionally, considering the human reaction time, the system also saves data from 2 seconds prior to the intervention for more context. This data is then used to fine-tune the high-level policy, as detailed in Section III-E. The specifics of the data, including the base dataset and the post-training dataset, are provided in Appendix V-B.

V. Experiments

Since YAY Robot proposes a hierarchical system that interfaces the high-level and low-level policies through natural language, we want to answer the following questions: (1) Can human interventions through natural language change robot behavior meaningfully to improve task success on-the-fly? (2) Can these verbal corrections improve the autonomous performance of the robotic system on long-horizon tasks? (3) How does the hierarchical approach in YAY Robot compare with non-hierarchical imitation learning methods? Further, we evaluate the importance of learning and improving a high-level policy in YAY Robot, by ablating it with scripted policies and VLMs like GPT-4V.

A. Robot Setup and Tasks

We conduct experiments on a set of real-robot tasks using ALOHA, a low-cost, open-source bimanual hardware setup proposed in Zhao et al. [70]. The setup includes a 14-dimensional action space corresponding to bimanual target joint positions and an observation space comprising RGB images from four cameras. Two cameras are mounted on the wrists for detailed object views during fine manipulation, and the other two are positioned at the front and top for broader perspectives. The demonstration data includes these camera feeds and the robots' joint positions at 50Hz. Further hardware specifics can be found in prior works by ALOHA 2 Team [61] and Zhao et al. [70].

We select three long-horizon tasks emphasizing precision, coordination, and contact-rich manipulation, involving challenging manipulation of deformable and transparent objects (Figure 5). Each task is designed to reflect useful, everyday robotic applications:

Bag Packing: This task involves packing three randomly positioned objects – a sharpie, a tape holder, and a sponge – into a small ziploc bag. We selected these items specifically as they present distinct challenges: the sharpie and tape holder require high precision for successful grasping and insertion (e.g., the tape holder has a clearance of ˜2mm), while the sponge, due to its size and deformable nature, may require squeezing to fit into the bag. The robot must perform a series of skills: picking up each object, inserting it into a bag, and arranging items within the bag to avoid obstruction to subsequent objects. Additionally, this task demands correction skills such as directional adjustments (right, left, towards, away, higher, lower), gripper rotation (clockwise, counter-clockwise), shaking the bag, and using the object to widen the bag opening, among others.

Trail Mix Preparation: This task involves creating a bag of trail mix from various ingredients (almonds, peanuts, M&M's, cranberries, and banana chips). Essential skills include using tools like a metal scoop, as well as precise maneuvers to scoop specific ingredients, adjust the quantity as per user instruction, and pour them into a transparent ziploc bag. Additionally, it involves corrections like adjusting the scoop's tilt and orientation, manipulating the bag for easier filling, and avoiding contact with food container edges.

Plate Cleaning: This task emulates a daily task in the real world, the cleaning of a dirty plate, requiring both precise and contact-rich manipulation and coordination from both arms for effective cleaning. For hygiene and safety reasons, we choose gummy bear candies that can stick to the plate surface as the food debris to be scrubbed. This task consists of four sets of skills: picking up the plate, reaching and grasping the sponge, moving the plate to a bowl, and wiping the gummy bears off the plate using the sponge. This task exhibits two common modes of failure: imperfect reaching or grasping of the plate and the sponge, and insufficient coverage of the wiping motion. Hence, correction skills such as "pick up the sponge again", "wipe the right side", and "clean the bottom" are crucial to clean the plate thoroughly.

B. Data

1) Base Dataset: For each of the three tasks, we collect a base dataset consisting of human teleoperated trajectories, annotated by language. In Table I, we detail the number of trajectories and language commands collected for each task during this stage of data collection.

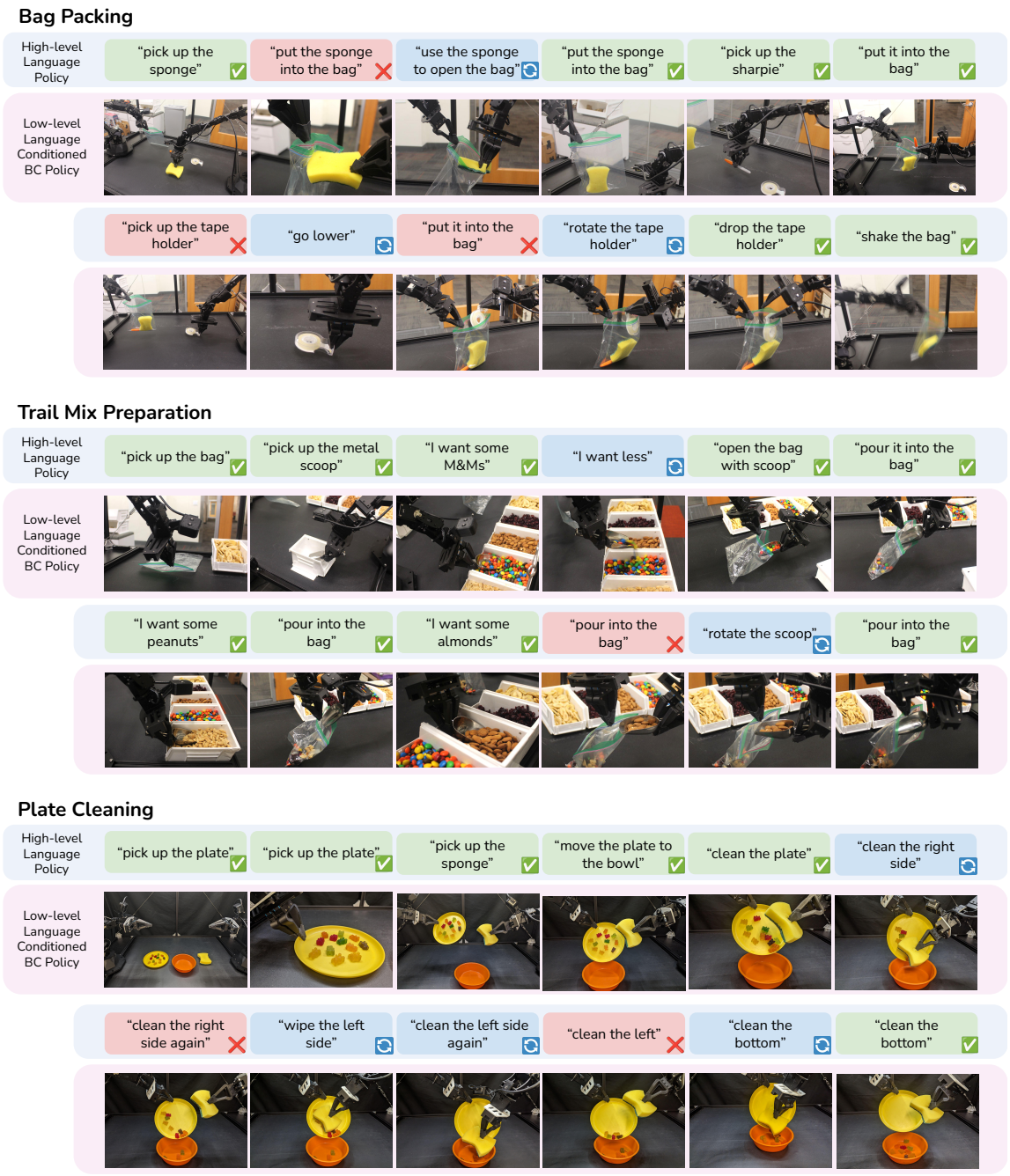


Fig. 5: **Real-World Task Rollouts with Language Corrections.** For each of the 3 long-horizon bimanual manipulation tasks, we illustrate the sub-tasks, common failure modes, examples of verbal corrections, and robot's corrective behaviors.

2) Post-Training Dataset: For high-level policy fine-tuning, we collect language-only human intervention data using a microphone. In Table II, we describe the aggregated dataset gathered after 2-3 iterations of post-training. The post-training dataset has significantly fewer skill segments compared to the base dataset. Additionally, we might have collected more post-training data than needed: we naturally collected lots of verbal interaction data when people come to the lab to play with the robot and when we continuously evaluate the policy over months of development.

3) Data Visualization: We visualize the language commands in the bag packing task by making a word cloud out of the most frequent 200 commands in the datasets, as shown in Figure 11. The most frequently-appeared skills are task-oriented. Each correction skill has a lower frequency, but the correction skills are more diverse (e.g. "wiggle", "rotate the gripper clockwise", "shake the bag a little").

C. Methods and Evaluation Protocol

In our study, we first train the low-level and high-level policies (**Base Policy**). Next, we consider YAY Robot in the

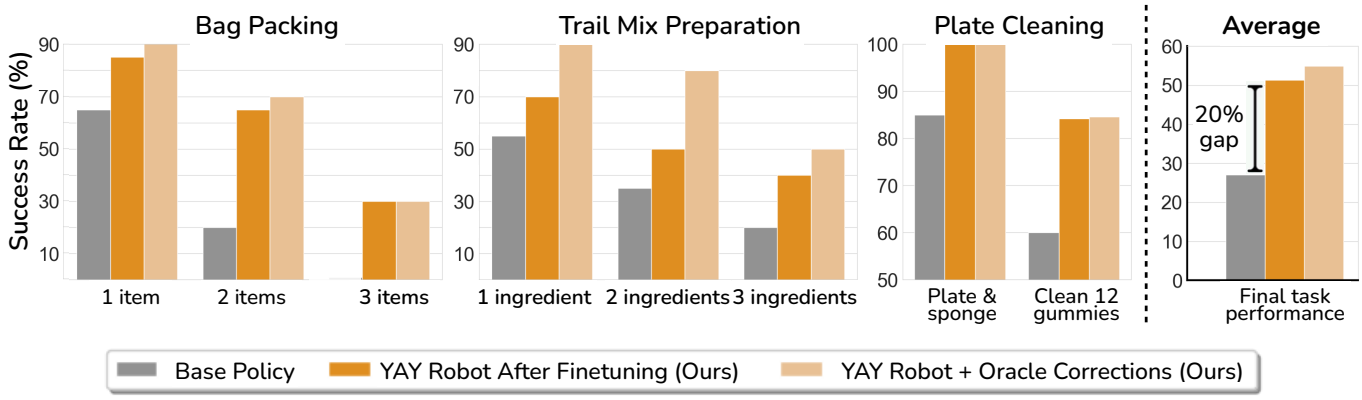


Fig. 6: **Quantitative Evaluations.** Our system demonstrates a 20% improvement in success rates over the base policy across three long-horizon bi-manual manipulation tasks, attributed to language corrections. These language corrections not only improve task success in real-time, but also substantially enhance the autonomous policy's performance at each stage of the tasks through fine-tuning.

	Bag Backing	Trail Mix	Plate Cleaning
Episodes	1170	317	265
Episode Length (timesteps)	2000 - 5000	3000	1200
Skill Segments	41517	7008	3236
Unique Commands	1054	104	33
Commands Appeared $\geq$ 3 Times	862	47	25

TABLE I: **Base Dataset.** Summary of data specs across different tasks. "Unique Commands" denote the number of unique language strings in the dataset.

	Bag Backing	Trail Mix	Plate Cleaning
Iterations	3	3	2
Skill Segments	2028	292	348
Ratio to the Base Dataset	0.048	0.042	0.108
Unique Commands	120	36	27
Commands Appeared $\geq$ 3 Times	92	17	18

TABLE II: **Post-Training Dataset.** Summary of data specs in the post-training dataset for each task. The post-training dataset is significantly smaller than the base dataset – the number of skill segments is 4%-11% of those in the base dataset.

interactive setting, where the high-level policy controls the low-level policy through natural language instructions but a human can override the high-level command and issue a natural language instruction (**YAY Robot + Oracle Corrections (Ours)**). This method allows us to evaluate the ability of the system to modulate its behavior on-the-fly and whether such interventions can improve the task performance, and we expect the task performance in this setting to be an upper bound on the performance of the best high-level policy. The final step for YAY Robot finetunes the high-level policy with data from human interventions collected in the shared autonomy setup, according to Section III. We evaluate the autonomous performance of YAY Robot after fine-tuning (**YAY Robot after Finetuning (Ours)**), with no additional human interventions. Notably, for all evaluation tasks, operators without a background in computer science or robotics research collect the teleoperation data and provide verbal feedback to YAY Robot, which also helps us assess the viability for real-world use in everyday settings by non-expert users.

To compare the performance of hierarchical policies with flat policies for long-horizon tasks, we benchmark our method against Action Chunking with Transformers (ACT [70]; **Flat-BC**), a state-of-the-art imitation learning method. We train ACT on the same training data as all the baselines above.

We conduct 20 trials for each evaluation. Given the long-horizon nature of the tasks, we also measure the sub-task success rate to provide more granular insights. We detail the success criteria for each task in Appendix VI-A1. Except for **YAY Robot + Oracle Corrections**, which allows *verbal* interventions from humans but no physical interventions, all evaluations are fully autonomous for both low-level and high-level policies.

D. Key Results

Language corrections enhance task success on the fly. As shown in Figure 6, our experiments show significant improvements in success rates at each stage when incorporating real-time language corrections: 25%-50% in Bag Packing, 30%-45% in Trail Mix Preparation, and 15%-25% in Plate Cleaning. Despite the high-level policy being trained on the entire dataset, including correction skills, we observe that it predominantly predicts task-specific instructions and rarely issues corrections prior to fine-tuning, as shown in our supplementary video.

We also demonstrate in our supplementary video how simple human language feedback helps address common failures – for instance, in the Bag Packing task, when the robot struggles with precise grasping, a simple human direction like "move to the left" enables the robot to adjust its position for a successful grasp. Furthermore, language feedback is particularly useful in out-of-distribution (OOD) scenarios, such as when the sharpie slides beneath the transparent bag during insertion or when objects entangle with the bag opening due to its deformability. In scenarios involving irreversible actions, such as in Trail Mix Preparation, humans can provide useful preemptive interventions, such as letting the robot adjust the angle of the

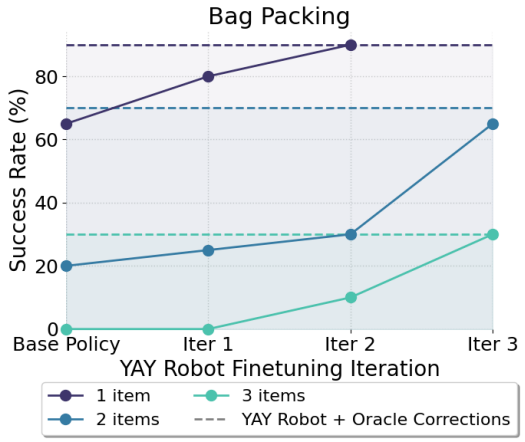


Fig. 7: **Iterative Improvement.** YAY Robot's success rates for packing different numbers of items show significant improvement with each iteration of user verbal feedback collection and fine-tuning, approaching the oracle's performance (dashed lines) at each stage of the task.

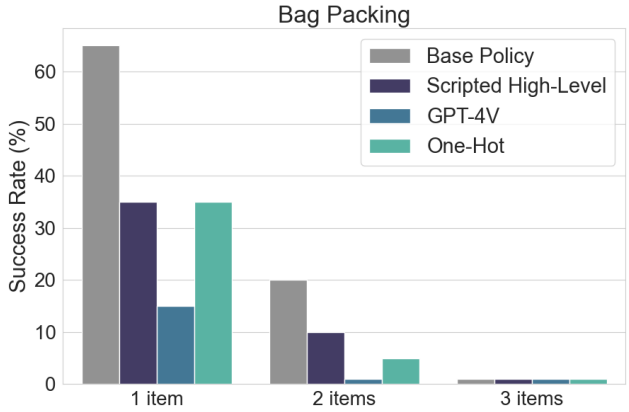


Fig. 8: **Ablation on High-Level Policy.** Our results show that 1) replacing a learned high-level policy with a scripted one leads to performance decrease, 2) off-the-shelf VLM performs poorly on complex long-horizon task, and 3) replacing language with one-hot encodings hurts model performance.

end effector or the position of the bag to prevent pouring nuts on the table. We save such interventions in the dataset to fine-tune the high-level policy, so that the robot can self-correct in subsequent interactions.

Finetuning on language corrections improves autonomous policy performance. Compared to the base policy, finetuning with our approach improves success rates at each stage by 20%-45% in Bag Packing, 15%-20% in Trail Mix Preparation, and 15%-25% in Plate Cleaning. After fine-tuning, the high-level policy starts to autonomously generate corrections, a behavior that evolves through iterative post-training (Figure 7). For example, in the Bag Packing task, the high-level policy begins to instruct the robot to "open the gripper wider" before reattempting to grasp, "rotate the right gripper clockwise" for successful insertion, or to "wiggle" when an object becomes entangled with the bag. In the Plate Cleaning task, removing all food debris covering the surface

Task	Substage	Base Policy	Flat BC
Bag Packing	1 item	**65**	25
	2 items	**20**	0
	3 items	0	0
Trail Mix Preparation	1 ingredient	55	**60**
	2 ingredients	**35**	20
	3 ingredients	**20**	5
Plate Cleaning	Prepare plate & sponge	**85**	80
	Clean 12 gummies	**60**	55

TABLE III: **Comparison to Flat Policy.** Overall, our hierarchical approach achieves higher success rates (%) than the non-hierarchical imitation learning method on long-horizon tasks.

requires multiple wipes covering different regions of the plate. Language corrections like "clean the" + "right", "left", "top", and "bottom" allow the robot to focus on specific areas of the plate and thus maximizing the cleaning results. We present qualitative and quantitative analysis of the policy behavior prior to and following fine-tuning in Figure 9 and Figure 10.

The hierarchical policy outperforms the flat policy on long-horizon tasks. As shown in Table III, success rates of hierarchical policies are generally higher than Flat-BC. The difference between hierarchical and flat policies is more pronounced in later stages of each task, indicating the hierarchical policy's better handling of compounding errors.

E. Ablation Studies

Next, we explore the necessity of a learned high-level policy. We compare our approach with both a predefined sequence of language instructions (**Scripted High-Level**) and a state-of-the-art vision-language model, GPT-4V (OpenAI et al. [46]; **GPT-4V**), as high-level policies. The GPT-4V policy is informed about the environment, task specifics, camera setups, and useful language instructions and corrections relevant to the task through a carefully crafted prompt (Appendix VI-A3). To examine the role of language conditioning on performance, we substitute language embeddings with one-hot skill encodings (**One-Hot**; Appendix VI-A2). Additionally, we investigate the trade-off between data quality and quantity by comparing ACT trained on filtered data that has removed suboptimal behaviors against training with the complete dataset (**All-Data**).

Scripted High-Level Policy: We replace our learned high-level policy with a predefined sequence of language instructions. As shown in Figure 8, we observe a marked decrease in task performance, with the policy performing worse than the Base Policy by up to 30%. Despite providing the most optimal skill sequence observed in policy rollouts, the scripted policy's inability to react to mistakes and adapt to unforeseen scenarios during deployment leads to failures. This finding underscores the necessity of a high-level policy that can dynamically address failures, especially given the imperfections of the low-level policy in handling all possible deployment scenarios.

GPT4-V as a High-Level Policy: We also explore using GPT4-V as a high-level policy. While it generates plausibly sounding reasoning steps, it frequently errs in understanding spatial relationships and the state of object manipulation, such

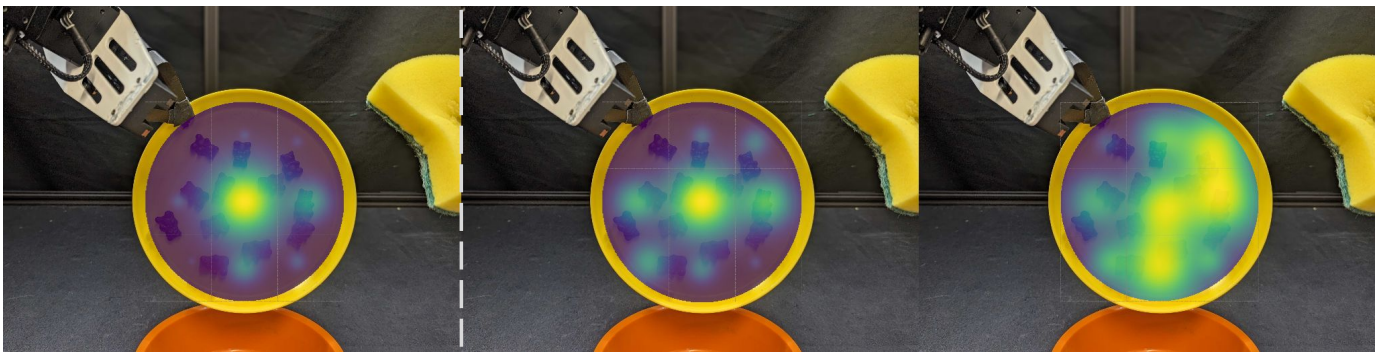


Fig. 9: **Policy Proficiency Increase through Fine-Tuning.** Through heatmaps, we visualize the cleaning efficacy across the plate surface, where brighter areas denote higher frequencies of effective wiping. YAY Robot demonstrates wider cleaning coverage after fine-tuning the high-level policy with human verbal feedback.

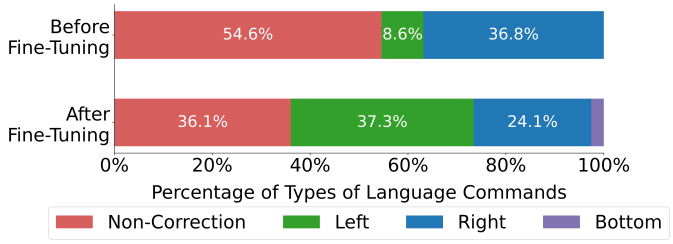


Fig. 10: **Non-Correction vs. Correction Skill Ratio.** Our analysis illustrates a shift from non-correction to corrective cleaning commands – cleaning left, right, and bottom – following policy fine-tuning, which leads to a notable enhancement in cleaning coverage (Figure 9), and thus more effective cleaning.

Task	Substage	Filtered Data	All Data
Bag Packing	1 item	**25**	10
	2 items	0	**5**
	3 items	0	0
Trail Mix Preparation	1 ingredient	**60**	55
	2 ingredients	**20**	**20**
	3 ingredients	**5**	0

TABLE IV: **Ablation on Data.** The policy's performance deteriorates when trained on larger datasets of mixed quality compared to smaller datasets of higher quality.

as instructing the robot to "lower the gripper" when it has already touched the table, or to "release the sponge" when doing so would result in the sponge falling outside, as most of it is still outside the bag. These errors persist even when GPT4-V is provided with visual inputs from most convenient camera angles from our phones. This suggests that VLMs, without training on interaction data, are not sufficiently reliable for being high-level policies in robotic tasks. Notably, our approach is complementary to pretrained VLMs. It would be interesting to explore our approach of high-level policy fine-tuning with a pretrained VLM, which we leave to future work.

One-Hot vs. Language Embedding: Replacing language embeddings with one-hot encodings in our model leads to inferior performance, occasionally resulting in unreasonable outputs. Given the large and diverse set of instructions in our dataset (e.g., 1,200 unique strings for the Bag task), we hypothesize language is crucial in leveraging the semantic similarities among instructions for model learning.

Data Quality vs. Quantity: Finally, we assess the importance of data quality in training. As shown in Table IV, while training with the entire dataset (All Data) offers more data points, we observe that the training loss is less stable or slightly higher, especially on more challenging tasks. Besides, the robot occasionally exhibits suboptimal behavior even when the observation appears to be in-distribution. This highlights the need for high-quality data or careful data filtering in training effective robotic policies.

VI. Conclusion and Limitations

We presented a framework for enabling robots to respond to and improve using verbal corrections. On three long-horizon bi-manual manipulation tasks, our system reaches a 20% higher success rate than the base policy, where the only additional supervision comes from verbal corrections. Despite promising results, our approach has a number of important limitations. To successfully handle language corrections both on-the-fly and in continuous improvement, our system critically relies on a performant low-level policy that can successfully react to many distinct language commands. Indeed, the performance of our approach with on-the-fly language corrections, which is essentially a near-optimal high-level policy, is far from perfect and not far above the performance using our fine-tuned high-level policy. This suggests that further performance improvements must come from improving the performance and flexibility of the low-level policy. Large-scale language-conditioned imitation learning approaches [26, 8], including methods that leverage vision-language models [9], have shown promise in both improving performance and expanding the vocabulary of language-conditioned policies. Beyond verbal corrections, it may also be beneficial for people to communicate in other ways, e.g. by pointing [11] or making other gestures. Our system is not well-equipped to handle such non-verbal communication. Looking forward, we hope that future research can further enable robots to improve with natural forms of human supervision, ultimately towards empowering anyone to help teach robots.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, K. Gopalakrishnan, Karol Hausman, Alexander Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, A. Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, N. Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, S. Levine, Yao Lu, Linda Luu, Carolina Parada, P. Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, D. Reyes, P. Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, F. Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. Do as i can, not as i say: Grounding language in robotic affordances. *Conference On Robot Learning*, 2022.

[2] Arun Ahuja, Kavya Kopparapu, Rob Fergus, and Ishita Dasgupta. Hierarchical reinforcement learning with natural language subgoals. *arXiv preprint arXiv:2309.11564*, 2023.

[3] Jacob Andreas, Dan Klein, and Sergey Levine. Learning with latent language. *arXiv preprint arXiv:1711.00482*, 2017.

[4] Himani Arora, Rajath Kumar, Jason Krone, and Chong Li. Multi-task learning for continuous control. *arXiv preprint arXiv:1802.01034*, 2018.

[5] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. *Computer Vision And Pattern Recognition*, 2023. doi: 10.1109/CVPR52729.2023.01324.

[6] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv: 2403.01823*, 2024.

[7] Alexander Broad, Jacob Arkin, Nathan Ratliff, Thomas Howard, and Brenna Argall. Real-time natural language corrections for assistive robotic manipulators. *The International Journal of Robotics Research*, 36(5-7):684–698, 2017.

[8] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, K. Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, A. Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, S. Levine, Yao Lu, U. Malla, D. Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, M. Ryoo, Grecia Salazar, P. Sanketi, Kevin Sayed, Jaspiar Singh, S. Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Q. Vuong, F. Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. *Robotics: Science And Systems*, 2022. doi: 10.48550/arXiv.2212.06817.

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[10] Arthur Bucker, Luis Figueredo, Sami Haddadin, Ashish Kapoor, Shuang Ma, Sai Vemprala, and Rogerio Bonatti. Latte: Language trajectory transformer. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7287–7294. IEEE, 2023.

[11] Stefan Constantin, Fevziye Irem Eyiokur, Dogucan Yaman, Leonard Bärmann, and Alex Waibel. Multimodal error correction with natural language and pointing gestures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1976–1986, October 2023.

[12] Yuchen Cui, Siddharth Karamcheti, Raj Palleti, Nidhya Shivakumar, Percy Liang, and Dorsa Sadigh. "no, to the right" - online language corrections for robotic manipulation via shared autonomy. *Ieee/acm International Conference On Human-robot Interaction*, 2023. doi: 10.1145/3568162.3578623.

[13] Bruno Da Silva, George Konidaris, and Andrew Barto. Learning parameterized skills. *arXiv preprint arXiv:1206.6398*, 2012.

[14] Marc Peter Deisenroth, Peter Englert, Jan Peters, and Dieter Fox. Multi-task policy search for robotics. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 3876–3881. IEEE, 2014.

[15] Yan Ding, Xiaohan Zhang, Chris Paxton, and Shiqi Zhang. Task and motion planning with large language models for object rearrangement. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2086–2092, 2023. doi: 10.1109/IROS55552.2023.10342169.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, S. Gelly, Jakob Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference On Learning Representations*, 2020.

[17] Danny Driess, F. Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Q. Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, P. Sermanet, Daniel Duckworth, S. Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Peter R. Florence. Palm-e: An embodied multimodal language model. *International Conference On Machine Learning*, 2023. doi: 10.48550/arXiv.2303.03378.

[18] Anqing Duan, R. Camoriano, Diego Ferigo, Yanlong Huang, Daniele Calandriello, L. Rosasco, and D. Pucci. Learning to sequence multiple tasks with competing

constraints. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2672–2678. IEEE, 2019.

[19] Divyansh Garg, Skanda Vaidyanath, Kuno Kim, Jiaming Song, and Stefano Ermon. Lisa: Learning interpretable skill abstractions from language. *Advances in Neural Information Processing Systems*, 35:21711–21724, 2022.

[20] Caelan Reed Garrett, Rohan Chitnis, Rachel M. Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annu. Rev. Control. Robotics Auton. Syst.*, 4:265–293, 2021. doi: 10.1146/ANNUREV-CONTROL-091420-084139.

[21] Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S Brown, and Ken Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. *arXiv preprint arXiv:2109.08273*, 2021.

[22] Ryan Hoque, Ashwin Balakrishna, Carl Putterman, Michael Luo, Daniel S Brown, Daniel Seita, Brijen Thananjeyan, Ellen Novoseller, and Ken Goldberg. Lazydagger: Reducing context switching in interactive imitation learning. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pages 502–509. IEEE, 2021.

[23] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 2022.

[24] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[25] Y.K. Hwang, P.C. Chen, and P.A. Watterberg. Interactive task planning through natural language. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 24–29 vol.1, 1996. doi: 10.1109/ROBOT.1996.503568.

[26] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[27] Yiding Jiang, Shixiang Shane Gu, Kevin P Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[28] Michael Kelly, Chelsea Sidrane, K. Driggs-Campbell, and Mykel J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. *Ieee International Conference On Robotics And Automation*, 2018. doi: 10.1109/ICRA.2019.8793698.

[29] Jens Kober, Andreas Wilhelm, Erhan Oztop, and Jan Peters. Reinforcement learning to adjust parametrized motor primitives to new situations. *Autonomous Robots*, 33:361–379, 2012.

[30] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Toward understanding natural language directions. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 259–266. IEEE, 2010.

[31] Thomas Kollar, Stefanie Tellex, Deb K. Roy, and Nicholas Roy. Grounding verbs of motion in natural language commands to robots. In *International Symposium on Experimental Robotics*, 2010.

[32] Jacky Liang, Wenlong Huang, F. Xia, Peng Xu, Karol Hausman, Brian Ichter, Peter R. Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. *Ieee International Conference On Robotics And Automation*, 2022. doi: 10.1109/ICRA48891.2023.10160591.

[33] Jacky Liang, Fei Xia, Wenhao Yu, Andy Zeng, Montserrat Gonzalez Arenas, Maria Attarian, Maria Bauza, Matthew Bennice, Alex Bewley, Adil Dostmohamed, Chuyuan Kelly Fu, Nimrod Gileadi, Marissa Giustina, Keerthana Gopalakrishnan, Leonard Hasenclever, Jan Humplik, Jasmine Hsu, Nikhil Joshi, Ben Jyenis, Chase Kew, Sean Kirmani, Tsang-Wei Edward Lee, Kuang-Huei Lee, Assaf Hurwitz Michaely, Joss Moore, Ken Oslund, Dushyant Rao, Allen Ren, Baruch Tabanpour, Quan Vuong, Ayzaan Wahid, Ted Xiao, Ying Xu, Vincent Zhuang, Peng Xu, Erik Frey, Ken Caluwaerts, Tingnan Zhang, Brian Ichter, Jonathan Tompson, Leila Takayama, Vincent Vanhoucke, Izhak Shafran, Maja Mataric, Dorsa Sadigh, Nicolas Heess, Kanishka Rao, Nik Stewart, Jie Tan, and Carolina Parada. Learning to learn faster from human feedback with language model predictive control. 2024.

[34] Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. *arXiv preprint arXiv:2211.08416*, 2022.

[35] Huihan Liu, Alice Chen, Yuke Zhu, Adith Swaminathan, Andrey Kolobov, and Ching-An Cheng. Interactive robot learning from verbal correction. *arXiv preprint arXiv:2310.17555*, 2023.

[36] Jianlan Luo, Charles Xu, Xinyang Geng, Gilbert Feng, Kuan Fang, L. Tan, S. Schaal, and S. Levine. Multi-stage cable routing through hierarchical imitation learning. *Ieee Transactions On Robotics*, 2023. doi: 10.48550/arXiv.2307.08927.

[37] Corey Lynch and Pierre Sermanet. Grounding language in play. *arXiv preprint arXiv:2005.07648*, 3, 2020.

[38] Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

[39] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

[40] Matt MacMahon, Brian Stankiewicz, and Benjamin

Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. *Def*, 2(6):4, 2006.

[41] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

[42] Jiayuan Mao, J. B. Tenenbaum, Tom'as Lozano-P'erez, and L. Kaelbling. Learning reusable manipulation strategies. *Conference On Robot Learning*, 2023. doi: 10.48550/arXiv.2311.03293.

[43] Hongyuan Mei, Mohit Bansal, and Matthew Walter. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[44] Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Structured world models from human videos. *Robotics: Science And Systems*, 2023. doi: 10.15607/RSS.2023.XIX.012.

[45] Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, 35(1-3):281–300, 2016.

[46] OpenAI, Josh Achiam, Steven Adler, et al. Gpt-4 technical report. *arXiv preprint arXiv: 2303.08774*, 2023.

[47] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in neural information processing systems*, 31, 2018.

[48] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015.

[49] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. *Aaai Conference On Artificial Intelligence*, 2017. doi: 10.1609/aaai.v32i1.11671.

[50] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. *International Conference On Machine Learning*, 2021.

[51] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision. *International Conference On Machine Learning*, 2022. doi: 10.48550/arXiv.2212.04356.

[52] Stéphane Ross, Geoffrey J. Gordon, and J. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *International Conference On Artificial Intelligence And Statistics*, 2010.

[53] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *NEURIPS*, 2019.

[54] Pratyusha Sharma, Antonio Torralba, and Jacob Andreas. Skill induction and planning with latent language. *arXiv preprint arXiv:2110.01517*, 2021.

[55] Pratyusha Sharma, Balakumar Sundaralingam, Valts Blukis, Chris Paxton, Tucker Hermans, Antonio Torralba, Jacob Andreas, and Dieter Fox. Correcting robot plans with natural language feedback. *arXiv preprint arXiv:2204.05186*, 2022.

[56] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[57] Tianmin Shu, Caiming Xiong, and Richard Socher. Hierarchical and interpretable skill acquisition in multi-task reinforcement learning. *arXiv preprint arXiv:1712.07294*, 2017.

[58] Aravind Sivakumar, Kenneth Shaw, and Deepak Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. *Robotics: Science And Systems*, 2022. doi: 10.15607/rss.2022.xviii.023.

[59] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

[60] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *International Conference On Machine Learning*, 2019.

[61] ALOHA 2 Team. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation, 2024. URL https://aloha-2.github.io/.

[62] Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[63] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew Walter, Ashis Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pages 1507–1514, 2011.

[64] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv: 1706.03762*, 2017.

[65] Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20, 2023.

[66] Nick Walker, Yu-Tang Peng, and Maya Cakmak. *Neural Semantic Parsing with Anonymization for Command Understanding in General-Purpose Service Robots*, pages 337–350. Springer International Publishing, 2019. doi: 10.1007/978-3-030-35699-6_26.

[67] Huaxiaoyue Wang, Kushal Kedia, Juntao Ren, Rahma

Abdullah, Atiksh Bhardwaj, Angela Chao, Kelly Y Chen, Nathaniel Chin, Prithwish Dan, Xinyi Fan, Gonzalo Gonzalez-Pumariega, Aditya Kompella, Maximus Adrian Pace, Yash Sharma, Xiangwan Sun, Neha Sunkara, and Sanjiban Choudhury. Mosaic: A modular system for assistive and interactive cooking. *arXiv preprint arXiv: 2402.18796*, 2024.

[68] Fan Xie, A. Chowdhury, M. Clara De Paolis Kaluza, Linfeng Zhao, Lawson L. S. Wong, and Rose Yu. Deep imitation learning for bimanual robotic manipulation. *Neural Information Processing Systems*, 2020.

[69] Jianjun Yu, Jie Jia, Guoyu Zuo, Xuchen Li, Yilin Cao, and Zihao Zhang. Multi-step tasks learning based on bayesian segmentation and dynamic movement primitive. In *2021 IEEE 11th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 265–270. IEEE, 2021.

[70] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv: 2304.13705*, 2023.

APPENDIX

A. Experiment Details

1) Task Success Criterion: We evaluate each method for 20 trials for all three tasks and report their success rates. The success criteria for each task are defined as follows:

1) Bag Backing:
   - "1 item" is successful if the robot successfully picks up the bag, grasps one of the objects, and inserts it into the bag.
   - "2 items" is successful if the robot successfully picks up the second object and inserts it into the bag without making the first item fall out of the bag.
   - "3 items" is successful if the robot successfully picks up the third object, inserts it into the bag without making the first or the second item fall out of the bag, and releases the bag without making any object fall out of the bag.
2) Trail Mix Preparation
   - "1 ingredient" is successful if the robot successfully picks up the bag, brings the bag closer to the user, picks up the scoop, scoops one ingredient, and pours it into the bag.
   - "2 ingredients" is successful if the robot successfully moves the gripper outside the bag, scoops the second ingredient, and pours it into the bag. If there exists any spill, the spill should be less than 1/4 of the total amount within the scoop.
   - "3 ingredients" is successful if the robot successfully moves the gripper outside the bag, scoops the third ingredient, pours it into the bag, and releases the bag.
3) Plate Cleaning:
   - Preparing the plate is successful if the robot grasps the plate and lifts up the plate mid-air.
   - Preparing the sponge is successful if the robot grasps the sponge and lifts up the sponge mid-air.
   - Preparing for cleaning is successful if the robot holds and moves the plate to the bowl, and moves the sponge near the plate.
   - Cleaning the plate success rate is measured by how many gummies are cleaned out of a total of 12 gummies on the plate.

2) One-Hot: We index all the 1054 unique language commands in the bag packing dataset. The model predicts logits for the indices (i.e., the output of the high-level policy is now a 1054-dimensional vector instead of language embeddings). We then train the model with cross-entropy loss. At the inference time, we select the index that has the highest probability (argmax).

3) GPT-4V: To test the capability of pre-trained VLMs in reasoning about complex, long-horizon tasks, we prompt GPT-4V to provide instruction for the language-conditioned low-level policy. We use the following prompt in Figure 12 to

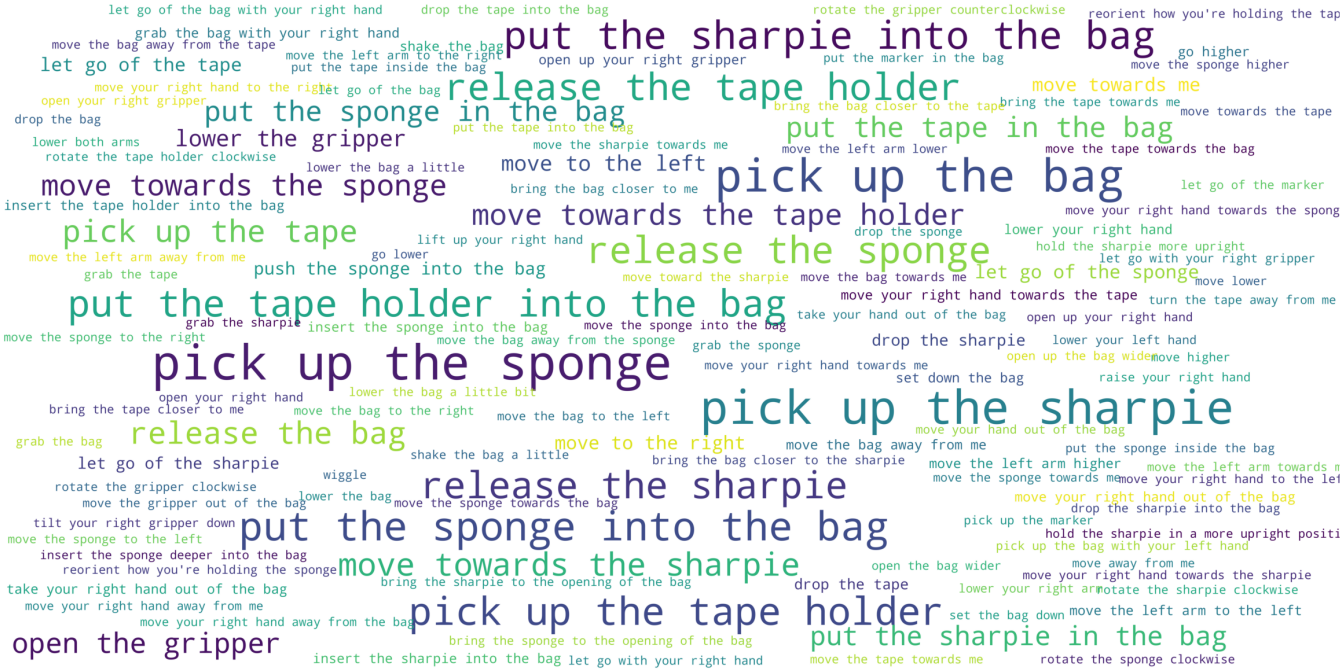


Fig. 11: Word cloud of the most frequent 200 commands in bag packing datasets.

inform GPT-4V about the task, camera set-up, and the most useful instructions and corrections for the bagging task.

GPT-4V is able to make reasonable predictions at the beginning, as shown in Figure 13. However, usually after 2-3 reasonable predictions, its spatial reasoning becomes incorrect. For example, in Figure 14, even though the sponge is still on the table and far from the robot gripper, the model reasons that "The sponge has been picked up by the robot" and asks the robot to "put the sponge in the bag". To check whether the suboptimal behavior is caused by the confusion over multiple camera observations, we tried to feed GPT-4V with only the third-person view from the front camera and adjust the prompt accordingly. It turns out that the model makes similar mistakes. As shown in Figure 15, the model similarly predicts "put the sponge into the bag" when the robot has not picked up the sponge yet. When we remind it of this fact through text, GPT-4V changes its prediction to "lower the gripper", while the gripper is already touching the table and cannot get lower. Additionally, we tried to make spatial reasoning easier by providing image observations from the most convenient camera angles (taken by phone). However, the model likewise makes mistakes in understanding spatial relationships, such as taking an object that is still outside the bag to be inside the bag (Figure 16). Given that changing camera angles did not help, we used the same multi-camera set-up as our learned high-level policy for evaluation.

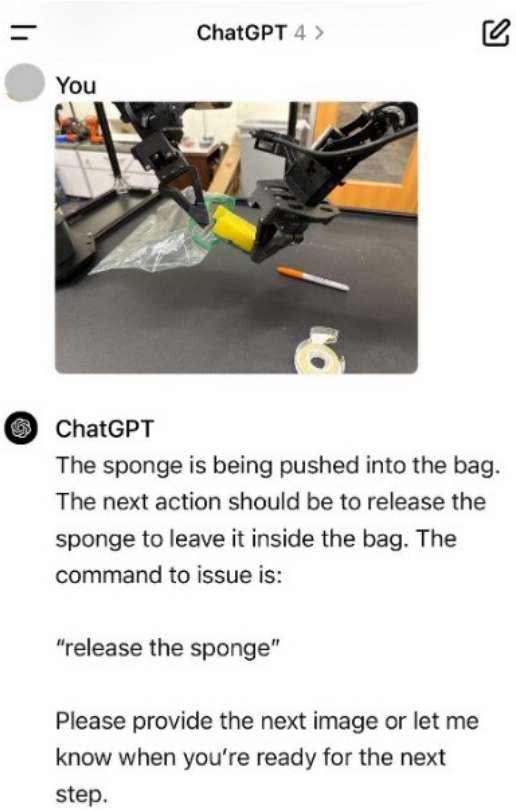


Fig. 16: GPT-4V is limited by the lack of knowledge regarding specific robotic capabilities.

B. Lessons

Below, we summarize our additional learnings from this project for future reference.

Technical Findings:

- Multimodal Integration: Our initial attempts to integrate language using cross-attention mechanisms were unsuccessful, which was unexpected. We found that Feature-wise Linear Modulation (FiLM) was more effective for language conditioning for ACT.
- Camera-Specific Backbones: Implementing a single visual backbone to process inputs from multiple cameras did not perform well on complex tasks. Separate backbones for each camera significantly improved task success rates, from 60% to 90% in a pick-and-place debugging task.
- Model Fine-tuning: Continuous training alongside the pretraining dataset is essential to prevent catastrophic forgetting.
- History Conditioning: Conditioning the high-level policy on a longer, subsampled observation history helps to reduce ambiguity and task switching, which are prevalent when only immediate observations are used.
- Instruction Dependency: We observed that conditioning on instruction history can lead to the model repetitively copying the last command as its prediction, especially in scenarios where the instruction frequency is high.

Data Collection and Augmentation:

- Iterative Data Collection: When uncertain of what data to collect next, it's beneficial to gather preliminary data, use it to train a model, and then identify failure modes to guide further data collection.
- Handling Distribution Shift: Techniques such as random cropping and brightness/contrast augmentation are effective for adapting to changes in camera positions and lighting conditions at the test time.
- Efficient Label Collection: For language labels, recording audio during data collection and subsequently transcribing it proved to be efficient for language label collection.

Meta Lessons:

- Data Inspection: Hours spent understanding the data is helpful for policy debugging and improvement. By identifying issues such as camera angle adjustments, shifts in data distribution at test time, motion smoothness, and overall data quality, this examination informs subsequent data collection.
- Simplified Datasets: Collecting small, curated robotic datasets proved invaluable for debugging and rapid iteration. These datasets allow for faster cycles of hypothesis testing and modification than larger datasets, which are heterogeneous and slow to iterate with.

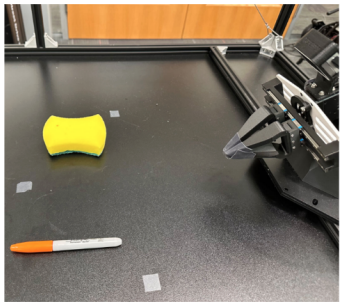


Fig. 17: An example of the test setup for image encoders.

C. Language and Visual Encoders

Language Embeddings: We conducted experiments to assess the policies trained with different language embeddings, specifically comparing DistilBERT and CLIP. The results demonstrated comparable performance across these embeddings as well as similar training and validation losses. Moreover, both setups exhibited robust generalizations in language understanding; for instance, when the robot was instructed with a variant term "grasp X" instead of "pick up X" (where X represents an object), it successfully identified and executed the correct action despite the term "grasp" not being present in this training dataset.

Image Encoders: We also evaluated the performance of our approach using different image encoders: EfficientNet b3, EfficientNet b0, and ResNet18. These encoders were tested in tasks involving the grasping of objects positioned randomly within a 20cm x 20cm area with 50 demonstrations (Figure 17). Each encoder was assessed in 10 evaluations, focusing on their ability to correctly identify and interact with objects based on given prompts:

- The robots achieved 100% accuracy in reaching for the correct object.
- Specific strategies such as rotating the gripper for sponges and adjusting the height for picking up thinner objects like sharpies were successfully executed.
- The pick-up success rates were as follows: EfficientNet b3 achieved 90%, while both EfficientNet b0 and ResNet18 achieved 70%.

These findings indicate that the choice of visual backbone impacts performance. Future work could benefit from a broader exploration of visual backbones to enhance the robustness and efficacy of robotic policies in real-world scenarios.

You're an AI assistant guiding a bimanual robot in bagging groceries into a ziploc bag over 6000 timesteps (2 minutes). Every 200 timesteps (4 seconds), you can issue one instruction from a provided list. You'll receive images at these intervals from four cameras: top, front, left wrist, and right wrist (left to right). You will select your instruction from the following list:

1. "pick up the sponge"
2. "pick up the sharpie"
3. "pick up the bag"
4. "put the sponge into the bag"
5. "put the sharpie into the bag"
6. "pick up the tape holder"
7. "put the tape holder into the bag"
8. "release the sharpie"
9. "release the tape holder"
10. "release the sponge"
11. "move towards the sharpie"
12. "release the bag"
13. "move towards the tape holder"
14. "move towards the sponge"
15. "pick up the tape"
16. "put the tape in the bag"
17. "put the sponge in the bag"
18. "open the gripper"
19. "put the sharpie in the bag"
20. "lower the gripper"
21. "let go of the sponge"
22. "let go of the tape"
23. "push the sponge into the bag"
24. "let go of the sharpie"
25. "drop the sharpie"
26. "drop the tape"
27. "move to the left"
28. "open up your right gripper"
29. "grab the bag with your right hand"
30. "insert the sponge into the bag"
31. "insert the sharpie into the bag"
32. "move towards me"
33. "insert the tape holder into the bag"
34. "lower your right hand"
35. "set down the bag"
36. "take your right hand out of the bag"
37. "move the left arm higher"
38. "move your right hand out of the bag"
39. "move your right hand towards the tape"
40. "shake the bag"
41. "move the left arm to the left"
42. "drop the sponge"
43. "rotate the tape holder clockwise"
44. "wiggle"
45. "move lower"
46. "move your right hand towards the sponge"
47. "move the bag away from me"
48. "move your right hand away from the bag"
49. "move to the right"
50. "rotate the sponge clockwise"

Fig. 12: Prompt to GPT-4V for the multi-camera setup. It includes the task information, camera setup, and 50 most useful instructions for the task.

You

t=0

ChatGPT

Based on the images provided, it seems we are at the starting point of the task with all items—sponge, sharpie, and tape holder—placed on the table and the robot's grippers open and ready for action.

The first instruction to issue to the robot is to pick up the bag, which will be necessary to place all the other items into. So, the instruction is:

1. "pick up the bag"

Please provide the next set of images once 200 timesteps have passed, or advise if you would like to continue with a different action.

Fig. 13: GPT-4V makes a reasonable choice at the beginning of the task.

You

t=600

ChatGPT

The sponge has been picked up by the robot. The next step is to place the sponge into the bag.

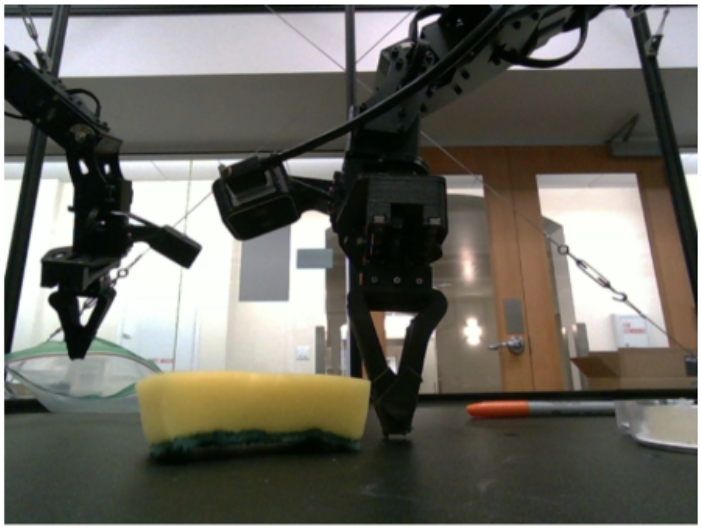
The instruction is:

1. "put the sponge in the bag"

Please send the next set of images after 200 timesteps, or let me know if you'd like a different action to be taken.

Fig. 14: As the robot proceeds, the spatial reasoning of the VLM becomes incorrect.

You



t=600

ChatGPT

The sponge is now within the robot's grasp. The next instruction is:

"put the sponge into the bag"

Proceed with this action and show me the next image when complete.

You

Note that the robot has not picked up the sponge yet

ChatGPT

Thank you for the clarification. In that case, the correct instruction should be:

"lower the gripper"

Please execute this action to pick up the sponge and provide the next image when ready.

Fig. 15: In the single-camera setup, the model still makes incorrect reasoning about the spatial relationship among the sponge, gripper, and table.