IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

ProVox: Personalization and Proactive Planning for Situated Human-Robot Collaboration

Jennifer Grannen[†], Siddharth Karamcheti[†], Blake Wulfe[‡], Dorsa Sadigh[†]

***Abstract*—Collaborative robots must quickly adapt to their partner's intent and preferences to proactively identify helpful actions. This is especially true in situated settings where human partners can continually teach robots new high-level behaviors, visual concepts, and physical skills (e.g., through demonstration), growing the robot's capabilities as the human-robot pair work together to accomplish diverse tasks. In this work, we argue that robots should be able to *infer their partner's goals* from early interactions and use this information to *proactively plan behaviors* ahead of explicit instructions from the user. Building from the strong commonsense priors and steerability of large language models, we introduce ProVox ("Proactive Voice"), a novel framework that enables robots to efficiently personalize and adapt to individual collaborators. We design a *meta-prompting protocol* that empowers users to communicate their distinct preferences, intent, and expected robot behaviors ahead of starting a physical interaction. ProVox then uses the personalized prompt to condition a *proactive language model task planner* that anticipates a user's intent from the current interaction context and robot capabilities to suggest helpful actions; in doing so, we alleviate user burden, minimizing the amount of time partners spend explicitly instructing and supervising the robot. We evaluate ProVox through user studies grounded in household manipulation tasks (e.g., assembling lunch bags) that measure the efficiency of the collaboration, as well as features such as perceived helpfulness, ease of use, and reliability. Our analysis suggests that both meta-prompting and proactivity are critical, resulting in 38.7% faster task completion times and 31.9% less user burden relative to non-active baselines.**[1]

***Index Terms*—Personalization, Proactive Planning, Situated Collaboration**

I. Introduction

Collaborative robots must be able to continually infer their partner's intent, adapting from this information to personalize and proactively suggest helpful actions. This is especially true in the context of *situated human-robot collaboration* [1–4], where robots and humans share the same physical space – a setting that spans increasingly important applications such as household robotics, elderly or assistive care, warehouse manufacturing, and robot-assisted surgery, amongst others [5, 6]. Across these applications, effective collaboration is challenging due to the sheer diversity of human partners, each with their own distinct goals and preferences.

Fig. 1. We present **ProVox** ("Proactive Voice"), a framework for personalization and proactive planning in the context of situated human-robot collaborations. In the first phase of a collaboration [**Top**], a human communicates their goals and distinct preferences, enabling the robot to *personalize*. Throughout the rest of the collaboration [**Bottom**], the robot continues to incorporate and anticipate their partner's intent to *proactively suggest helpful actions* (e.g. "Should I put the hand sanitizer in next?") ahead of explicit instruction, reducing the user's mental load while they assemble the sandwich.

Consider the example in Fig. 1 of a household robot working with a person to assemble multiple lunch bags on a busy morning. Different people express different constraints on the overarching task – for example, the person in Fig. 1 wants each bag to contain snacks, a sandwich, and a hand sanitizer because his kids have been sick, while in Fig. 2, another person needs the bag to contain Skittles, their favorite snack. In both cases, the human needs to perform the dexterous, fine-grained task of making the sandwich (i.e., grabbing two slices of bread, spreading the jelly and cream cheese, slicing off the crusts); however, without any other information, the division of work between the robot and human for the rest of the task is ambiguous. In such a scenario, a passive robot [7–9] might wait for explicit instructions, expecting the human to context-switch between making the sandwich and monitoring the robot's progress – for example, repeatedly instructing the robot to "put the Rice Krispies treat in the lunchbox" followed by similar instructions for the hand sanitizer and candy (and again for the next lunch bag) – a process that is as inefficient as it is frustrating. Instead, a more productive collaboration might

Manuscript received: December 12, 2024; Revised: March 31, 2025; Accepted: May 29, 2025.

This paper was recommended for publication by Editor Angelika Peer upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported by the Toyota Research Institute (TRI), the DARPA Friction for Accountability in Conversational Transactions (FACT) Program, the AFOSR Young Investigator Program, the Stanford Institute for Human-Centered AI (HAI), the Cooperative AI Foundation, the NSF (Awards #1941722, #2006388, #2125511), the Office of Naval Research (ONR Award #N000142112298), and DARPA (Grant #W911NF2210214).

The authors are with [†]Stanford University and [‡]Toyota Research Institute. jgrannen@stanford.edu, skaramcheti@cs.stanford.edu, blake.wulfe@tri.global, and dorsa@cs.stanford.edu.

[1]Videos and Supplementary Material: https://provox-2025.github.io

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

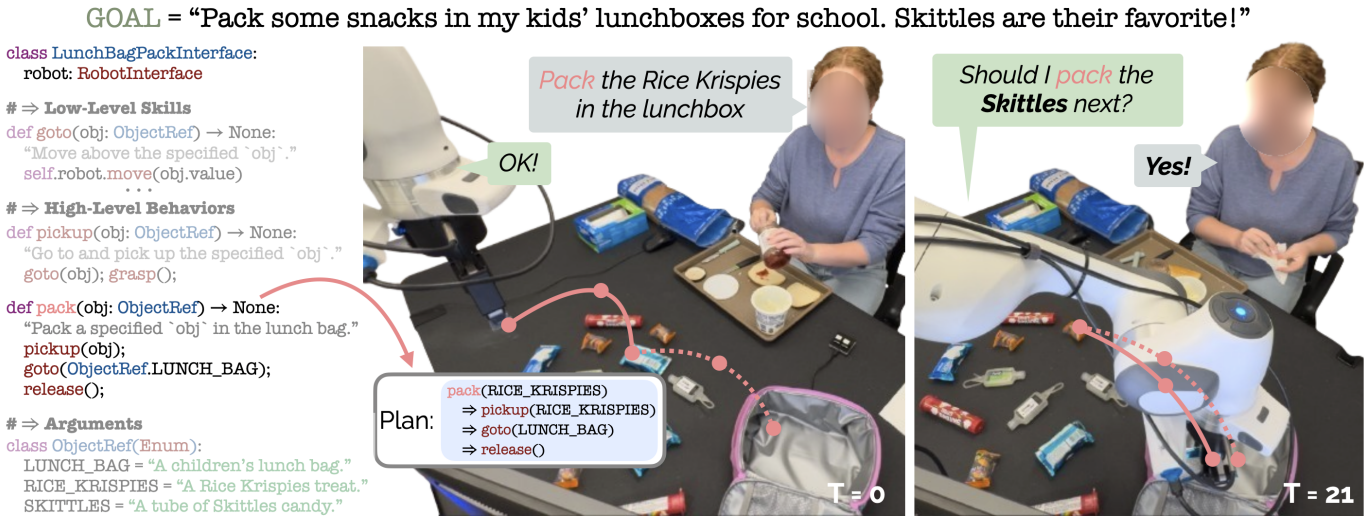

Fig. 2. **ProVox Motivating Example.** Existing frameworks for situated human-robot collaboration tend to assume static, hard-coded APIs to inform task planning (gray) that cannot be adapted to new individuals with distinct objectives and preferences. Instead, ProVox allows users to provide high-level goals [**Top**] and define task-relevant actions (e.g., pack on the [**Left**]). This enables *personalization* to a user's specific vocabulary and commands [**Middle**], and *proactive planning* [**Right**], where the robot suggests helpful behaviors to accomplish the goal.

start with a *handshake*: an explicit protocol where the human iterates with the robot to build up a shared understanding of their intent. Doing so allows the robot to *personalize* and adapt to each individual; now, as soon as the interaction starts, the robot might work from this shared understanding to *proactively suggest a helpful plan*. From Fig. 1, this might be as simple as extrapolating from the user's comment about his sick kids to suggest hand sanitizer as a possible addition to each lunch bag ("Should I put the hand sanitizer in next?").

In this work, we formalize this process by introducing **ProVox** ("Proactive Voice"), a new framework for developing personalizable and proactive collaborative robots that adapt online from language-based interactions with a partner. ProVox builds on top of prior work for situated collaboration [8, 9] that leverage the commonsense priors and steerability of large language models [LMs; 7, 10–12] for task planning. Our first contribution is a *meta-prompting protocol* that equips users with a natural interface for not only communicating their overall objectives to the robot, but also for specifying concrete examples to seed the robot with an understanding of the user's distinct vocabulary and preferences (Fig. 1; Top). We use the resulting prompt to condition a *proactive language model planner* that anticipates what the robot should do next from the interaction context, suggesting helpful plans that work to alleviate user burden and improve the efficiency of the collaboration (Fig. 1; Bottom).

We evaluate our technical contributions through two user studies. In our first study, we evaluate our meta-prompting protocol, performing a survey ($N = 26$) that demonstrates the flexibility and effectiveness of our proposed protocol relative to existing meta-prompting approaches. We find that our meta-prompting procedure universally improves our language model's ability to proactively suggest helpful plans while handling cross-user diversity relative to baselines. Finally, we perform a real-world within-subjects user study comparing ProVox to a state-of-the-art passive, user-agnostic system [9] grounded in the lunch bag packing scenario seen in Fig. 1 and Fig. 2. We find that ProVox enables 38.7% faster collaborative task performance, with participants strongly preferring our system due to its ease of use (+27.3%) and helpfulness (+18.4%), as well as their willingness to use it again (+26.5%).

II. Related Work

ProVox builds on prior work that propose new learning frameworks for situated human-robot collaboration, methods that leverage the commonsense reasoning and in-context learning ability of large language models for task planning, and general approaches for developing personalized and proactive robots in the context of human-robot interaction.

Learning Frameworks for Situated Collaboration. An expansive body of work frames human-robot collaboration as turn-taking, where a robot executes actions conditioned on a partner's prompt, spanning modalities such as natural language instructions [13, 14] or gestures [15, 16]. Realizing the lack of adaptivity in these approaches, subsequent work develop methods for adapting robot behavior online, from more dynamic inputs such as real-time language corrections [17, 18] or physical interventions [19]. More recently, work such as MOSAIC [8] and Vocal Sandbox [9] extend such methods to develop fully-fledged systems that connect diverse modalities for interaction and teaching to build collaborative robots that can sustain long-horizon interaction for predefined tasks (e.g., cooking a fixed recipe). We build ProVox on top of these works, specifically extending the Vocal Sandbox [9] framework to generalize to users with distinct goals and preferences while also enabling proactive planning (Fig. 2).

Language Models as Task Planners. Many approaches that map language to robot behavior do so by leveraging the commonsense priors and steerability afforded by large, pretrained language models (LMs). These approaches often use LMs for *task planning*, mapping complex user instructions to structured intermediate representations [7, 20] that are used to inform robot behavior. Many recent methods formalize task plans as executable programs [11, 21, 22], using LMs to generate sequences of function calls subject to a predefined API – for

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

example, a Python class defining primitive robot behaviors such as `grasp()` or `pickup(obj: ObjectRef)`.

While specifying task plans as executable programs offers benefits such as interpretability, runtime validation (i.e., always ensuring that generated task plans compile), and extensibility [23–25], code-based planners introduce new challenges when used in the context of human-robot collaboration. Specifically, a new partner trying to instruct such a robot needs a robust understanding of the robot's underlying API *as well as* a rudimentary understanding of what language instructions invoke different functionality. Prior work attempts to overcome these issues through solutions that either make exclusionary assumptions about the types of people who can use such systems [e.g., assuming enough code and LM literacy to understand what to say to induce a given behavior; 11], or otherwise place undue burden on the user to work out what they can or cannot say through trial-and-error, or through expensive "practice" sessions with the robot [8, 9]. This need to "naturalize" [26] a planning interface to a given person motivates the meta-prompting contributions in this work.

Personalization and Proactivity in HRI. The proactive planning component of ProVox is informed by a wealth of work that learns to infer a user's intent through interaction. For example, algorithms for preference-based learning [27–29] model the space of user intents by parameterizing reward functions based on a predefined set of features, resolving ambiguity by actively querying the user (e.g., asking them to rank or score different robot behaviors), often while minimizing some auxiliary cost (e.g., frequency of user queries, time to recover the ground-truth reward, etc.). Other work seeks to explicitly learn predictive models of user behavior [30–32] from a combination of offline and online interaction data; by predicting what the user might do next, robots can proactively choose their actions in a way that offers maximal assistance, while minimizing any associated cost. While we do not explicitly learn models of user behavior or reward in this work, our proactive planner tries to infer actions based on the full interaction history – a history that encompasses the full sequence of actions taken by both the human and the robot.

III. Problem Formulation

ProVox builds on Vocal Sandbox [9], a framework for situated collaboration that allows human partners to teach robots new behaviors and concepts *online*, during an interaction. We first formalize and highlight the key mechanisms of Vocal Sandbox, while the following section (§IV) introduces our novel contributions for personalization and proactive planning.

A. Vocal Sandbox – Preliminaries

Vocal Sandbox consists of a high-level language model (LM) planner and a family of low-level skills. The LM task planner maps spoken utterances from the human collaborator to code that subscribes to a predefined API; the Python-based API we use for our main user study is shown in Fig. 2 (Left). Each function in the API (e.g., `goto(obj: ObjectRef)`) corresponds to an individual skill that is parameterized by arguments as shown in the function signature. "Executing" a function means rolling out the corresponding skill with the provided arguments (e.g., `goto(LUNCH_BAG)`).

A critical affordance uniquely provided by Vocal Sandbox is the ability for human collaborators to use language to teach the robot new behaviors (i.e., functions) online, throughout the course of an interaction. Fig. 2 provides a concrete example in which the robot's base API (gray) consists of simple motion primitives such as `goto(obj: ObjectRef)`, and `pickup(obj: ObjectRef)`, as well as operations for opening/closing the gripper. While this base API covers the space of motions the robot can perform, it is not ergonomic or natural – a limitation demonstrated in Fig. 2 (Middle), when the user asks the robot to "pack the Rice Krispies in the lunchbox." To teach this new behavior, the user verbally *decomposes* their utterance into a sequence of functions that already exist in the API – in this case, via the program `pickup(RICE_KRISPIES); goto(LUNCH_BAG); release()`. Critically, after providing this decomposition, the Vocal Sandbox task planner uses the initial utterance and the corresponding program to *synthesize* a new function `pack(obj: ObjectRef)` – with the appropriate type signature and documentation – adding it to the API so that it can immediately be used for the remainder of the interaction.

In Grannen et al. [9], users can teach new behaviors during a collaboration. However, this procedure requires substantial user effort during an interaction – users must continually instruct the robot by continuously giving commands and decomposing unknown behaviors. Instead, ProVox proposes a meta-prompting protocol, where users provide context and preferences to naturalize the system prior to task execution. Then, a proactive LM planner consumes this context to suggest helpful actions during a collaboration.

B. Formalizing Task Planning and Teaching

Vocal Sandbox formalizes task planning as conditional language generation; here, we define notation for this problem setting. In §IV, we discuss ProVox also using this notation.

At a given interaction step t, the language model LM_θ attempts to generate a programmatic plan p_t conditioned on the user's natural language utterance u_t, the full interaction history $h_t = [(u_1, p_1), (u_2, p_2), \dots (u_{t-1}, p_{t-1})]$, the current API Λ_t, and a global collaboration goal prompt u_{fixed}. Critically, u_{fixed} is hand-designed by Grannen et al. [9] and ***held fixed for all users***, serving to outline the full scope of the collaboration. This includes the specific task the robot and user are to complete, examples of possible language instructions, and expectations of how the robot should behave. Establishing a meta-prompting protocol for personalizing the goal prompt u^k_{prompt} to a specific individual k with distinct preferences and goals is a key contribution of ProVox (§IV-A).

Teaching is formalized as a separate conditional generation task. Given an example $(\hat{u}_t, \hat{p}_t)$ consisting of a trigger utterance $\hat{u}_t$ (e.g., "Pack the Rice Krispies in the lunchbox" from §III-A), and the corresponding program decomposition $\hat{p}_t$ (e.g., `pickup(RICE_KRISPIES); goto(LUNCH_BAG); release()`), the goal is to *synthesize* a new function to be added to the robot's API Λ_{t+1}. As described in §III-A, the output of the synthesis step is a new function name (e.g.,

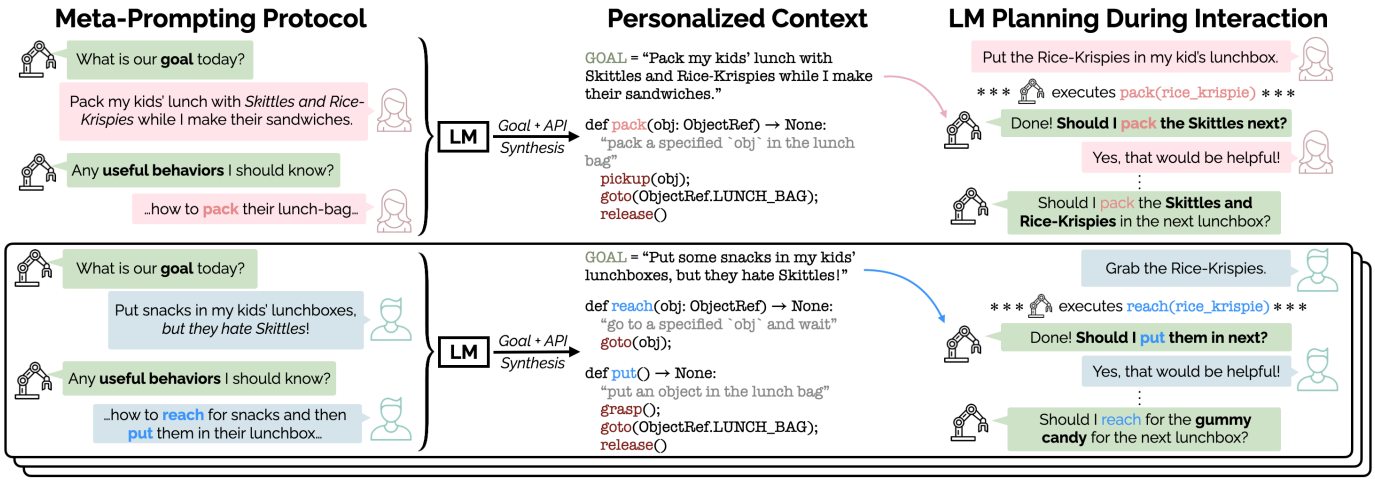

Fig. 3. **Meta-Prompting Protocol & Proactive Planning.** ProVox develops a novel meta-prompting protocol to collect two critical pieces of information from an individual: their specific goal, and an API of useful behaviors. Crucially, each user has a distinct set of preferences, yielding different goals and behaviors. The female user [**Top-Left**] wants her children's lunch to contain Skittles and Rice-Krispies, and teaches the robot to pack objects, with full confidence in its ability to identify, grasp, and move objects. In contrast, the male user [**Bottom-Left**] is more hesitant in trusting the robot; thus, he separates pick-and-place into two parts: a motion to `reach` for an object (moving above it without grasping), followed by a `put` behavior to complete the motion. The language model task planner then leverages this meta-prompted context to proactively suggest helpful, personalized behaviors over the course of the interaction [**Right**].

`pack`), type signature (e.g., `(obj: ObjectRef)` $\rightarrow$ `None`), informative documentation string (e.g., *"Pack a specified object in the lunch bag"*), and function body (e.g., `pickup(obj); goto(LUNCH_BAG); release()`). To generate the function type signature and body from program decomposition $\hat{p}_t$, Vocal Sandbox employs prior unification-based algorithms for program induction [26, 33], while the language model LM_θ generates both the function name and documentation from the interaction – see Grannen et al. [9] for further detail.

ProVox employs a similar teaching approach to Vocal Sandbox with one key difference: individual users can teach behaviors *before* a physical interaction begins. Teaching in advance of an interaction yields a more capable robot partner at task time, maximizing efficiency during costly robot interactions. We describe ProVox's novel contributions further in §IV.

IV. ProVox – Personalization & Proactivity

To enable personalization and proactive planning, ProVox introduces two novel contributions: 1) a *meta-prompting protocol* (§IV-A) that enables individual users to communicate their distinct objectives, preferences, and expectations to the robot, and 2) a *proactive language model planner* that suggests helpful actions ahead of explicit instructions (§IV-B).

A. Designing a Meta-Prompting Protocol

As discussed in §III-B, a key limitation of Vocal Sandbox and similar systems [8, 9, 21] is the reliance on a hand-designed global prompt u_{fixed} and base API Λ_{base} that outlines the full scope of a collaboration – a scope that encompasses the specific task to complete, examples of "valid" language instructions, and expected robot behaviors. Beyond the question of generalizing to different users with distinct goals and preferences, relying on this global prompt also *unfairly homogenizes the nature of a human-robot collaboration*; different users are expected to use the same vocabulary (e.g., behavior or object names), adopt the same roles, and use the same conventions prescribed by the system designers, which can be damaging in the presence of diverse users who want to assert different amounts of autonomy or trust. Fig. 3 shows an example of these disparities. Not only do the users have unique goals and preferences as to what should go in each lunch box, but they even define different behaviors for placing objects in each bag; rather than let the robot perform the entire pick-and-place motion continuously, the second user wants to verify how individual objects will be grasped prior to transferring them to the lunch box (Fig. 3; Bottom).

We address these questions of per-user personalization by defining a novel *meta-prompting protocol* that allows an individual k to naturally communicate their preferences and goals to arrive at a personalized prompt u^k_{prompt} and API Λ^k. We express this protocol via a graphical user interface, with an overview of the core features visualized in Fig. 3. [2] Intuitively, the interface gives users the ability to directly work with the language model task planner, iteratively building u^k_{prompt} and Λ^k through a "query-driven development" workflow [34, 35].

A user starts by describing their individual goal and preferences in natural language (e.g., "Pack my kids' lunch with Skittles and Rice-Krispies while I make their sandwiches" in the `GOAL` field of Fig. 3; Middle), which the LM planner interprets to set u^k_{prompt} for the personalized context. Individuals then *iteratively test and verify the outputs of the LM planner* (in isolation, prior to interacting with the physical robot) by providing test utterances such as "can you put the cereal bar in the bag?" or "how about packing the hand sanitizer?". Before teaching any new behaviors, the LM may output a valid plan given its reasoning and common sense abilities (i.e., `pickup(cereal), goto(lunchbox), release()`). However, in most instances, the LM fails to initially generate a satisfying plan. Users then have the option to explicitly teach new API functions themselves by specifying: (1) function name (e.g., "pack"), (2) expected behavior (e.g., "packing food for lunch"), and (3) the function body through a drop-down menu (e.g., `pickup(food), goto(lunchbox), release()`). Users

[2] We provide additional figures and code on our project page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

may choose to iterate on their taught behaviors, trying out what they have taught to see if the LM planner matches the expected behavior (e.g., "Put the cereal bar in my lunch." maps to `pack(cereal)`). Giving users explicit insight into the API in this manner is a key difference from the LM-driven function synthesis procedure in Grannen et al. [9]; we evaluate the benefits of doing this with user studies (§VI-A).

We provide users with the ability to view the history of taught actions and LM outputs, as well as freely edit and delete their goal text and taught functions. For simplicity, we do not track the entire interaction history $h_t = [(u_1, p_1), (u_2, p_2), \dots (u_{t-1}, p_{t-1})]$ when using the LM to generate plans. Note that teaching is not limited to definitions via this interface; if the user realizes they want to define new behaviors while physically interacting with the robot, they can do so via the teaching-via-synthesis procedure in §III-B.

We provide the context output (a goal u^k_{prompt} and an API Λ^k) to the LM planner, thereby enabling it to proactively suggest actions (see Fig. 3; Right). We emphasize the novelty of our meta-prompting protocol for collaborative task planning: where prior situated collaboration works lack personalization [8, 9], ProVox's meta-prompting protocol enables users to communicate their personal objectives and preferences *before an interaction begins*. We evaluate the impact of meta-prompting relative to global prompting via an isolated, component-wise user study (§VI-A), and the impact of meta-prompting in the context of a full system user study (§VI-B).

B. Proactive Planning

Another limitation of prior systems such as Vocal Sandbox is the lack of an ability to proactively suggest plans, preventing the robot from assuming more autonomy and initiative over the course of a collaboration. The robot executes the user's instruction and then waits idly for the next instruction (e.g. in Fig. 3, "Put the Rice-Krispies in my kids lunchbox." would invoke only `pickup(rice_krispies)`). This limitation again stems from the expensive upfront cost of having new users build a mental model and "naturalize" [26, 36] to a system reflecting the conventions and expectations of another person (i.e., the initial system designer). The second contribution of ProVox builds on top of the grounded understanding of an individual's goals, preferences, and desired behaviors obtained as a result of the meta-prompting procedure, yielding a *proactive language model planner* – a LM planner that pairs the commonsense abilities embedded in LMs with the information encoded in u^k_{prompt} and Λ^k to progressively suggest actions that help maximize the efficiency of the collaboration.

We implement proactivity as a straightforward extension of the base task planner described in §III-B. After each executed task plan p_t, we invoke the LM planner again by prompting it with the following trigger string: "Propose an action to perform next to perform [user-provided goal u^k_{prompt}]." For example, in Fig. 3 the robot proactively suggests helpful actions after completing each instruction – from finishing the current lunchbox ("Should I pack the Skittles next?") to proactively suggesting longer plans for packing a full lunchbox ("Should I pack the Skittles and the Rice Krispies in the next lunchbox?"). We do this continually at each interaction step, using the full interaction history (and potentially updated API) to shape the planner's suggested behaviors. For safety and transparency, we do not automatically execute proposed plans. Rather, the robot displays its intended plan on an interface and vocalizes it, then gates on user confirmation before execution. We note that if the proactive LM planner fails, users retain the ability to instruct the robot as in Grannen et al. [9].

The novelty of ProVox's proactive planner is that it actively suggests helpful actions that we show minimize robot downtime and streamline collaborations through a full system user study (§VI-B), evaluating ProVox against a non-active Vocal Sandbox baseline. We measure the efficiency of the collaboration as well as qualitative metrics such as ease of use, predictability, and perceived helpfulness.

V. Implementation & Reproducibility

In this work, we implement the ProVox framework mostly following the design decisions in Grannen et al. [9] with minor changes. First, due to OpenAI's deprecation warning of the GPT-3.5 Turbo suite of language models used in the original Vocal Sandbox work [9], we adopt the more recent GPT-4 Turbo [v04-09-2024; 10] as our base language model LM_θ. As our full system user study (§VI-B) focuses on a different application (lunch bag packing), we define a new base API to reflect the new objects and motion primitives; the full Python implementation can be found on our website. As in Grannen et al. [9] we format the API as an object-oriented Python API, formatted as a Markdown code block. We use the function-calling capabilities provided by OpenAI to constrain the language model to generate valid programs.

Robot Platform & Motion Primitives. We use a Franka Panda fixed-arm manipulator equipped with a Robotiq 2F-85 gripper following the hardware specification in Khazatsky et al. [37]. We also assume an extrinsics-calibrated ZED 2 stereo camera to obtain point cloud observations. As participants and the robot share a physical workspace, we run a compliant controller with a contact force threshold of 40 N, and torque threshold of 30 Nm. We implement two software-based kill switches (one controlled by the participant, the other by the study proctor), as well as two hardware emergency stops. We implement our `goto(obj)` and `pickup(obj)` motion primitives by identifying heuristic offsets relative to the 3D centroid of each object in the robot's coordinate frame. We obtain these centroid coordinates from off-the-shelf vision models following Grannen et al. [9], first obtaining a 2D segmentation mask via FastSAM [38], then retrieving a point cloud via backprojection through our calibrated camera.

VI. User Studies

We evaluate ProVox through two user studies. The first study (§VI-A; $N = 26$) serves to evaluate the meta-prompting protocol in isolation, evaluating the proposed procedure's ability to capture different user preferences and effectively inform proactive planning in a controlled experiment. We then perform a full-system study on a real-world robot platform (§VI-B; $N = 9$) that has participants evaluate ProVox against

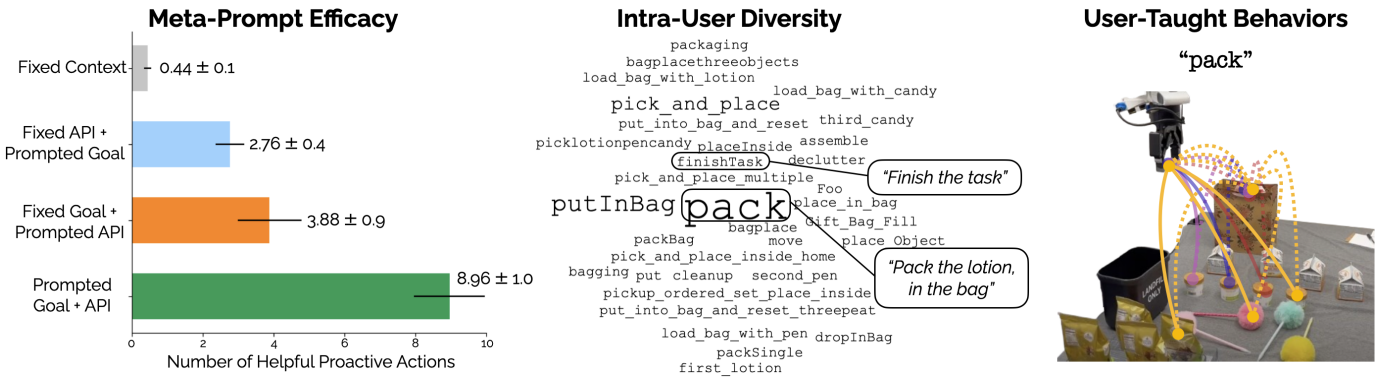

Fig. 4. **Meta-Prompting User Study Results.** We present results evaluating the efficacy of ProVox's meta-prompt protocol through a $N = 26$ user study, as well as visualize the diversity of user-defined behaviors. We report the mean and standard error values of the number of helpful, proactive actions suggested by the LM planner given full, partial, or no access to the meta-prompted context [**Left**]. We observe that both parts of the meta-prompted context (goal and API) are crucial for downstream proactive suggestion capabilities. We also highlight the intra-user diversity with a word cloud of 33 user-taught behavior names as well as two corresponding commands [**Middle**], with the majority of the behavior names being unique with the exception of `pack` (5X), `putInBag` (3X), and `pick_and_place` (2X). Interestingly, while five behavior instances are named pack, each corresponds to a unique program decompositions, visualized in different colors on the [**Right**], underscoring the need for personalization – even `pack` means different behaviors to different people!

a (non-proactive) Vocal Sandbox system for a collaborative lunch bag packing application. All studies were IRB approved.

A. *Meta-Prompting for Grocery Bagging*

In this study, participants evaluate our meta-prompting protocol by attempting to specify a goal and API to accomplish a given task (shown to participants as a video of a robot rolling out in a controlled environment). We aim to measure the diversity of goals and behaviors our procedure can cover, as well as quantitative metrics that speak to how well the resulting meta-prompted contexts inform proactive planning.

Participants and Procedure. We conducted this study with a population of $N = 26$ participants (within-subjects – 9 female, 17 male; ages between 21 and 69) with some amount of prior experience working with robots (an average self-reported experience score of $3.65/7$). After providing informed consent, participants were shown a 26 second video of a Franka Emika Panda robot bagging three items (a bottle of lotion, a pen, and candy) and tasked with reproducing the behavior in as few language instructions as possible. Participants were given access to a list of initial robot behaviors (e.g., `pickup`), a set of example "trigger" utterances (e.g., "pick up the ..."), and the corresponding robot demonstration videos. We asked each participant to engage with the meta-prompting protocol over the course of 20 minutes, working to produce a goal u^k_{prompt} and API Λ^k they felt best represented the initial video.

Independent Variables: Meta-Prompt Information. We evaluate the efficacy of our meta-prompting protocol by using each participant's resulting prompt (e.g., the context u^k_{prompt}, Λ^k) to condition the proactive language model planner; if the participant was able to properly communicate their goal and expected behaviors, the planner should be able to suggest a maximal sequence of behaviors that make progress towards replicating the actions taken in the provided robot video. We compare our proposed protocol (*Prompted Goal + API*) to three baselines that ablate specific components. *Fixed Goal + Prompted API* conditions the language model planner with a fixed, neutral goal u_{fixed} = "to help the user with a tabletop task," but using the participant's taught behaviors Λ^k. *Fixed API + Prompted Goal* does the opposite, using the participant's prompted goal u^k_{prompt}, but does not allow access to the taught behaviors, instead using the base API Λ_{base}. Finally, *Fixed Context* assumes a neutral goal and fixed API.

Dependent Measures. We measure the efficacy of each approach by generating the proactive task plan p_{pro} following §IV-B. We compare p_{pro} with the ground truth task plan p_{ref} depicted in the initial demonstration, reporting the count of overlapping function invocations (individual behaviors represented in the reference video) as *Meta-Prompt Efficacy*. Large overlaps indicate that the proactive planner and the corresponding meta-prompting method are effective in suggesting helpful actions that mimic the ground truth task plan p_{ref}.

Hypotheses. This intuition informs our first hypothesis (**H1**) that proactive planners with access to parts of the meta-prompted context (*Fixed API + Prompted Goal* and *Fixed Goal + Prompted API*) are more effective in suggesting helpful actions than a fixed context planner (*Fixed Context*). Furthermore, our second hypothesis (**H2**) affirms that the proactive planner with full access to the meta-prompted context (*Prompted Goal + API*) would be more effective in suggesting helpful actions than any other method.

Results. We report *Meta-Prompt Efficacy* for each of the methods in Fig. 4 (Left). Our graph clearly shows that the ProVox meta-prompting protocol (*Prompted Goal + API*) achieves the highest efficacy with an average of 8.96 ± 1.0 helpful actions proposed, supporting (**H2**). We also observe that the methods with partial access to the meta-prompted context, *Fixed API + Prompted Goal* and *Fixed Goal + Prompted API*, suggest an average of 3.88 ± 0.9 and 2.76 ± 0.4 helpful proactive actions respectively; this is higher than the average 0.44 ± 0.1 helpful actions suggested by the *Fixed Context* method, supporting (**H1**).

We run two-way repeated-measures ANOVA tests to assess the significance of these results. We find that having partial or full access to the meta-prompted context has a statistically significant effect ($p \leq 0.05$) on meta-prompt efficacy compared to using a fixed context. We find that using the full meta-prompted context has a significant effect ($p \leq 0.05$) on

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

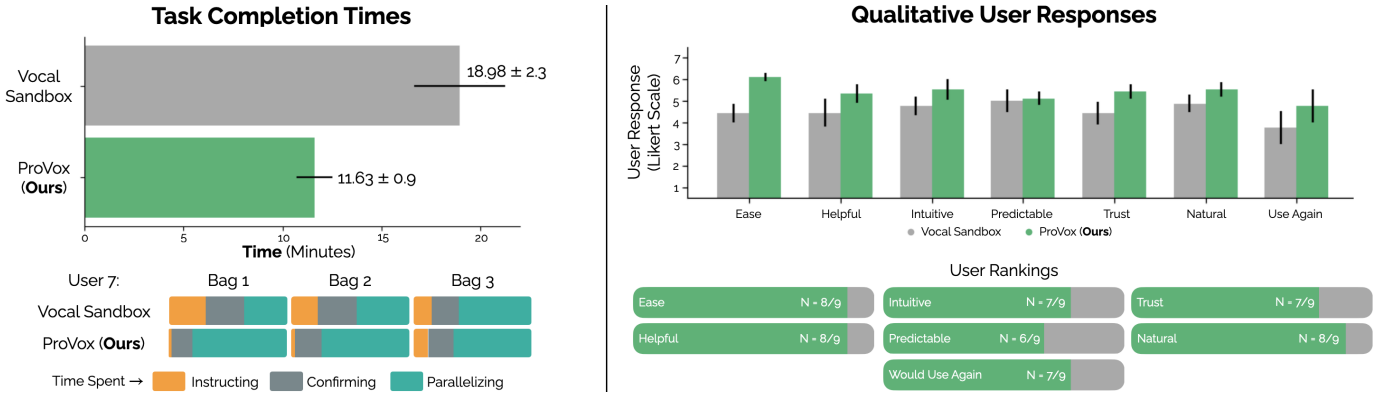

Fig. 5. **Full System User Study Results.** We present quantitative and qualitative results from our real-world user study of collaborative lunch bag packing. We report the average and standard error task completion times for users collaborating with a ProVox or a Vocal Sandbox system to pack three lunch bags [**Top, Right**]. We further visualize the breakdown of how a specific user spends their time throughout an interaction [**Bottom, Right**], highlighting the minimal instruction needed in the ProVox collaboration. On the [**Top, Right**], we report the average and standard error user responses on a 1–7 Likert scale across seven questions. We also ask users to explicitly rank which of the two methods they favor across these seven categories [**Bottom, Right**]. We observe in both the responses that users strongly favor ProVox over the baseline collaborator in terms of ease, helpfulness, trustworthiness, and willingness to use again.

meta-prompt efficacy compared to partial access.

Beyond efficacy, we visualize the diversity of taught behaviors across participants in Fig. 4, highlighting the need for per-user personalization. Across $N = 26$ participants, our user-specific meta-prompted contexts consist of 39 unique behaviors with 33 distinct names (Fig. 4; Middle). Even within taught behaviors of the same name, participants elect to teach vastly different program decompositions. For example, there are five instances of taught behaviors with the most common name, pack, however there are three unique program decompositions of these behaviors (visualized in Fig. 4; Right).

B. Full-System Evaluation: Lunch Bag Packing

In this full-system study, participants collaborate with a ProVox system to pack three lunch bags. Each bag must contain two snack items, a bottle of hand sanitizer, and one prepared sandwich. Participants may interact with the robot to pack items in each lunch bag while they focus on the dexterous task of making the sandwich (i.e., spreading the jam and cream cheese, cutting off the crusts, and placing it a Ziploc bag).

Participants and Procedure. We conducted the full-system study with a population of $N = 9$ participants (within-subjects – 4 female, 5 male; ages between 25 and 28), again with a some amount experience of working with robots (an average self-reported experience of $3/7$). Each participant performed the task with each of two methods (ProVox and Vocal Sandbox), with a random ordering across users. After providing informed consent, we provided each participant with a sheet describing the robot's base API (consisting of 5 initial behaviors) and teaching interfaces. We also offered the opportunity to try a practice task ("clear the table" – unrelated to the lunch bag packing task). To systematically evaluate different preferences, we assigned each participant 3 items to pack in the lunch bag (in addition to the prepared sandwich), from the set consisting of hand sanitizer, Rice Krispies treat, Skittles, and gummy candy. If applicable, participants were asked to engage with the meta-prompting protocol to collect their personalized context ahead of interacting with the robot.

Independent Variables: Personalization & Proactivity. We compare ProVox against an instantiation of a (non-proactive) Vocal Sandbox system [9]; running this head-to-head comparison allows us to evaluate our technical contributions while controlling for the remaining system capabilities.

Dependent Measures. We consider both objective and subjective metrics to evaluate our framework. For each method, we report the time needed to pack three lunch bags (*Task Completion Times*) as well as the number of user or robot-lead plan proposals throughout the task. After completing the task, participants complete a survey of 7-point Likert scale questions to assess the qualitative characteristics of the given method: *Easy to Use*, *Helpful*, *Intuitive*, *Predictable*, *Trust*, *Natural*, and *Willingness to Use Again*. After interacting with both methods, we additionally asked participants to explicitly rank each method subject to the same criteria.

Hypotheses. Our first hypothesis (**H1**) asserts that participants will complete the lunch bag packing task faster with the ProVox system compared to the Vocal Sandbox system; intuitively, a faster task completion time validates the overall utility of our proposed contributions. Furthermore, we affirm that (**H2**) the ProVox system reduces the burden on each user to provide explicit instructions by proposing helpful plans. Finally, we expect that (**H3**) participants qualitatively prefer ProVox to Vocal Sandbox across all criteria due to the utility provided by personalization and proactivity.

Results. We report quantitative and qualitative results in Fig. 5. In collaborating with the ProVox system, participants complete the task in an average of 11.63 ± 0.9 minutes, significantly faster than the 18.98 ± 2.3 minutes needed for the Vocal Sandbox system, supporting (**H1**). We note that these times only include the (situated) human-robot collaboration, and exclude the time spent in the meta-prompting interface; if we look at meta-prompting times alone, we see an average of 5.58 ± 0.9 minutes. Note that time spent engaging with the meta-prompting protocol is a fixed cost at the beginning of each interaction that does not scale with the horizon of the rest of the task. We additionally probe the proactivity of ProVox by computing the ratio of user-initiated behaviors to robot-initiated behaviors (Fig. 5; Left). Notably, participants accede to robot-proposed plans 31.9% of the time, strongly supporting (**H2**). In further visualizing the breakdown of how

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

participants spend time during the course of an interaction, we observe that users also spend 19.0% *less time* explicitly instructing ProVox relative to Vocal Sandbox.

Fig. 5 (Right) plots the responses to the subjective questions. Participants strongly prefer ProVox in terms of ease of use, helpfulness, intuitiveness, trustworthiness, naturalness and willingness to use again, supporting (**H3**). We also report user rankings for each of these criteria to explicit measure the trade-off between the two methods, where the trends are clearer.

We assess the significance of these results through one-way repeated-measures ANOVA tests. We find that collaborating with the ProVox system has a statistically significant effect on task completion times, as well as user ratings for ease and helpfulness. The remaining subjective results are not statistically significant (an artifact of the relative small participant pool). In general, participants see clear gains from the personalization and proactivity provided by ProVox, both in terms of collaboration efficiency and subjective perception.

VII. Discussion

We present ProVox, a novel framework for developing personalizable and proactive robots in the context of situated collaboration. ProVox proposes two novel technical contributions: (1) developing a meta-prompting protocol to personalize to an individual user's objectives and expected robot behaviors as well as (2) a proactive language model task planner that suggests helpful actions ahead of explicit user instruction. Through these contributions, ProVox demonstrates the ability to not only generalize to a broad population of participants with distinct preferences, but also improve the efficiency of a human-robot collaboration. In general, ProVox systems produce more easy to use (+27.3%), helpful (+18.4%) and trustworthy (+22.5%) robot collaborators, with 38.7% faster collaborative task performance and 31.9% less user burden compared to state-of-the-art baselines [9]. That said, ProVox is only a start, with multiple avenues for future work.

Towards Diverse Feedback Modalities. While ProVox provides a framework for personalizing a language-based goal and API to a user, it fails to consider other feedback modalities that implicitly encode preferences, such as kinesthetic demonstrations or gestures. In general, a key limitation of ProVox is its reliance on an *ungrounded* language model; this opens up an interesting avenue of future work for integrating vision-language models [39, 40] or even vision-language-action models [41] for human-robot collaboration.

References

[1] G. Hoffman and C. L. Breazeal, "Collaboration in human-robot teams," *AIAA 1st Intelligent Systems Technical Conf.*, 2004.

[2] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. O. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human–robot collaboration," *Autonomous Robots*, vol. 42, pp. 957 – 975, 2017.

[3] J. Brawer, O. Mangin, A. Roncone, S. Widder, and B. Scassellati, "Situated human–robot collaboration: predicting intent from grounded natural language," *Int. Conf. on Intelligent Robots and Systems*, 2018.

[4] L. Shaikewitz *et al.*, "In-mouth robotic bite transfer with visual and haptic sensing," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2023.

[5] J. Krüger, T. Lien, and A. Verl, "Cooperation of human and machines in assembly lines," *CIRP Annals*, vol. 58, no. 2, pp. 628–646, 2009.

[6] A.-N. Sharkawy, "A survey on applications of human-robot interaction," *Sensors & Transducers*, vol. 251, no. 4, pp. 19–27, 2021.

[7] M. Ahn *et al.*, "Do as i can and not as i say: Grounding language in robotic affordances," in *Conf. on Robot Learning*, 2022.

[8] H. Wang *et al.*, "Mosaic: A modular system for assistive and interactive cooking," in *Conf. on Robot Learning*, 2024.

[9] J. Grannen, S. Karamcheti, S. Mirchandani, P. Liang, and D. Sadigh, "Vocal Sandbox: Continual learning and adaptation for situated human-robot collaboration," in *Conf. on Robot Learning*, 2024.

[10] OpenAI, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[11] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2023.

[12] W. Huang *et al.*, "Inner monologue: Embodied reasoning through planning with language models," in *Conf. on Robot Learning*, 2022.

[13] S. Tellex *et al.*, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, 2020.

[14] A. Brohan *et al.*, "RT-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[15] C. Matuszek, L. Bo, L. Zettlemoyer, and D. Fox, "Learning from unscripted deictic gesture and language for human-robot interactions." in *Proceedings of the AAAI Conf. on Artificial Intelligence*, vol. 28, 2014.

[16] L.-H. Lin *et al.*, "Gesture-informed robot assistance via foundation models," in *Conf. on Robot Learning*, 2023.

[17] Y. Cui *et al.*, ""No, to the right"– online language corrections for robotic manipulation via shared autonomy," in *ACM/IEEE HRI*, 2023.

[18] L. X. Shi *et al.*, "Yell at your robot: Improving on-the-fly from language corrections," *ArXiv*, vol. abs/2403.12910, 2024.

[19] A. Bajcsy *et al.*, "Learning robot objectives from physical human interaction," in *Conf. on Robot Learning*, 2017.

[20] I. Singh *et al.*, "Progprompt: Generating situated robot task plans using large language models," *IEEE Int. Conf. Robotics and Automation*, 2022.

[21] P. Liu, Y. Orru, C. Paxton, N. M. M. Shafiullah, and L. Pinto, "Ok-robot: What really matters in integrating open-knowledge models for robotics," in *Proc. Robotics: Science and Systems*, 2024.

[22] J. Varley *et al.*, "Embodied ai with two arms: Zero-shot learning, safety and modularity," *arXiv preprint arXiv:2404.03570*, 2024.

[23] W. Huang *et al.*, "Grounded decoding: Guiding text generation with grounded models for robot control," in *Proc. Advances in Neural Information Processing Systems*, 2023.

[24] M. G. Arenas *et al.*, "How to prompt your robot: A promptbook for manipulation skills with code as policies," *IEEE Int. Conf. on Robotics and Automation*, pp. 4340–4348, 2024.

[25] OpenAI, "GPT-3.5 – Function calling and other updates," https://openai.com/index/function-calling-and-other-api-updates/, 2023.

[26] S. I. Wang *et al.*, "Naturalizing a programming language via interactive learning," in *Association for Computational Linguistics*, 2017.

[27] A. Jain *et al.*, "Learning preferences for manipulation tasks from online coactive feedback," *Int. Journal of Robotics Research*, 2015.

[28] E. Biyik and D. Sadigh, "Batch active preference-based learning of reward functions," in *Conf. on Robot Learning*, 2018.

[29] E. Biyik, A. Talati, and D. Sadigh, "Aprel: A library for active preference-based reward learning algorithms," *ACM/IEEE HRI*, 2021.

[30] S. Nikolaidis *et al.*, "Efficient model learning from joint-action demonstrations for human-robot collaborative tasks," *ACM/IEEE HRI*, 2014.

[31] D. Sadigh *et al.*, "Information gathering actions over human internal state," *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016.

[32] S. Javdani *et al.*, "Shared autonomy via hindsight optimization for teleoperation and teaming," *Int. Journal of Robotics Research*, 2018.

[33] S. Karamcheti, D. Sadigh, and P. Liang, "Learning adaptive language interfaces through decomposition," in *EMNLP Workshop for Interactive and Executable Semantic Parsing*, 2020.

[34] Beck, "Test driven development: By example," 2002. [Online]. Available: https://api.semanticscholar.org/CorpusID:262220275

[35] P. Liu *et al.*, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, pp. 1 – 35, 2021.

[36] S. Nikolaidis and J. Shah, "Human-robot teaming using shared mental models," *ACM/IEEE HRI*, 2012.

[37] A. Khazatsky *et al.*, "DROID: A large-scale in-the-wild robot manipulation dataset," in *Robotics: Science and Systems*, 2024.

[38] X. Zhao, W. Ding, Y. An, Y. Du, T. Yu, M. Li, M. Tang, and J. Wang, "Fast segment anything," *arXiv preprint arXiv:2306.12156*, 2023.

[39] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," *arXiv preprint arXiv:2310.03744*, 2023.

[40] OpenAI, "GPT-4v(ision) system card," 2023.

[41] M. J. Kim *et al.*, "Openvla: An open-source vision-language-action model," in *Conf. on Robot Learning*, 2024.