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

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

[†]Stanford University [‡]Toyota Research Institute

***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* [1–3], 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 [4–9]. Across these applications, effective collaboration is challenging due to the sheer diversity of human partners, each with their own distinct goals and preferences.

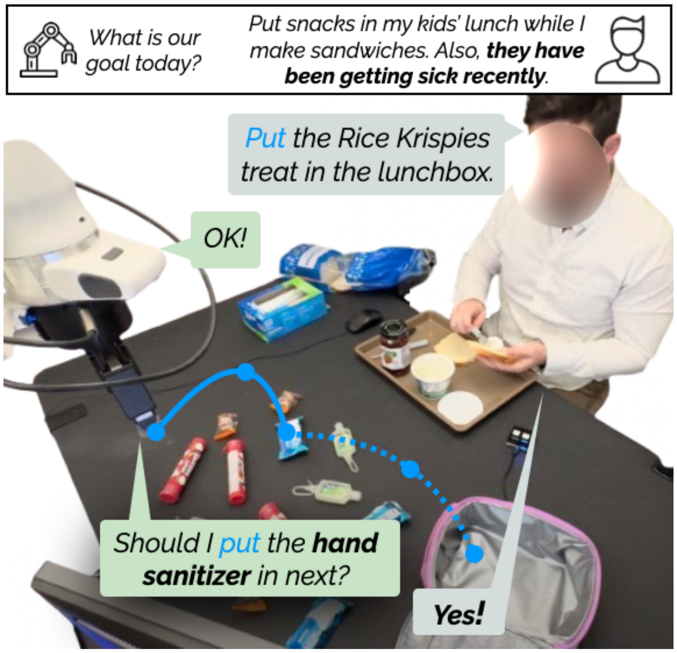
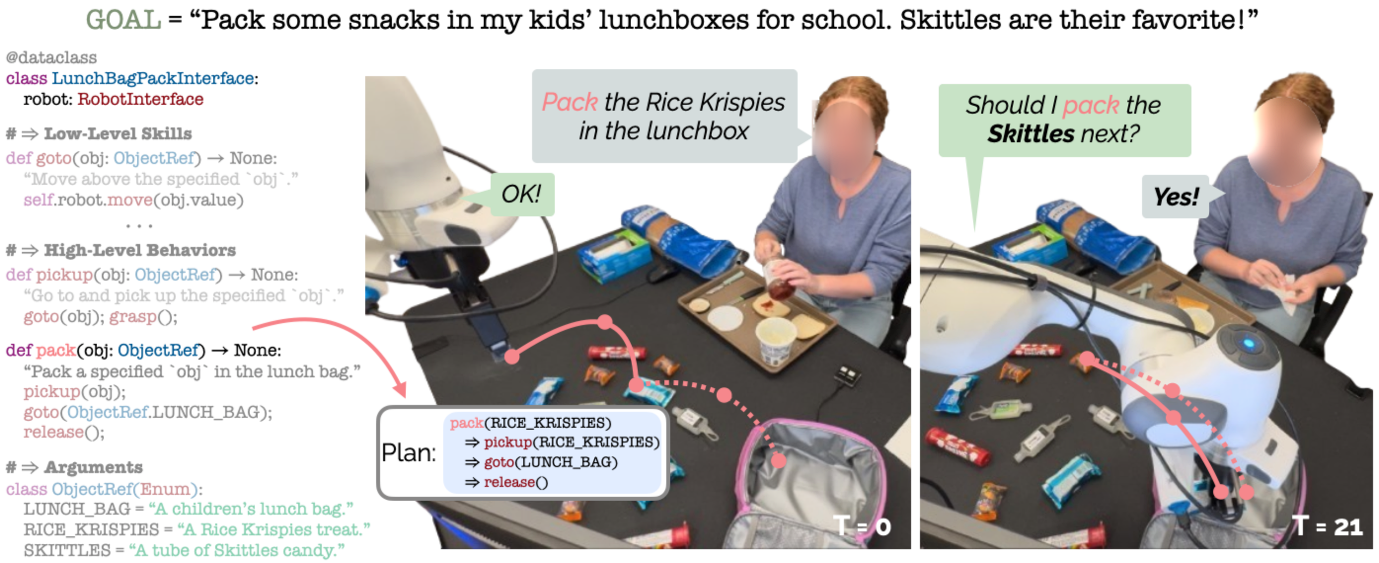

What is our goal today?
Put snacks in my kids' lunch while I make sandwiches. Also, they have been getting sick recently.
Put the Rice Krispies treat in the lunchbox.
OK!
Should I put the hand sanitizer in next?
Yes!

Fig. 1. We present **ProVox** ("Proactive Voice"), a framework for personalization and proactive planning in the context of a situated human-robot collaboration. In the first phase of a collaboration [**Top**], a human partner communicates their goals and distinct preferences, enabling the robot to *personalize* to an individual. 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 instructions, reducing the user's burden and mental load while they assembles 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 [10–12] 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

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



GOAL = “Pack some snacks in my kids’ lunchboxes for school. Skittles are their favorite!”

```
@dataclass
class LunchBagPackInterface:
   robot: RobotInterface

# ⇒ Low-Level Skills
def goto(obj: ObjectRef) → None:
   “Move above the specified `obj`.”
   self.robot.move(obj.value)
                  . . .
# ⇒ High-Level Behaviors
def pickup(obj: ObjectRef) → None:
   “Go to and pick up the specified `obj`.”
   goto(obj); grasp();

def pack(obj: ObjectRef) → None:
   “Pack a specified `obj` in the lunch bag.”
   pickup(obj);
   goto(ObjectRef.LUNCH_BAG);
   release();

# ⇒ Arguments
class ObjectRef(Enum):
   LUNCH_BAG = “A children’s lunch bag.”
   RICE_KRISPIES = “A Rice Krispies treat.”
   SKITTLES = “A tube of Skittles candy.”
```

```
Plan: pack(RICE_KRISPIES)
        ⇒ pickup(RICE_KRISPIES)
        ⇒ goto(LUNCH_BAG)
        ⇒ release()
```

Pack the Rice Krispies in the lunchbox
OK!
T = 0
Should I pack the Skittles next?
Yes!
T = 21

Fig. 2. **ProVox Motivating Example.** Existing frameworks for situated human-robot collaboration tend to assume static, hard-coded APIs to inform task planning (gray) – APIs 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 – for example pack on the [**Left**]. This enables *personalization* (e.g., to a user’s specific vocabulary and commands) [**Middle**], and *proactive planning* [**Right**], where the robot suggests helpful behaviors to accomplish the goal.

it is frustrating. Instead, a more productive collaboration might 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 [11, 12] that leverage the commonsense priors and steerability of large language models [LMs; 10, 13–15] 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 that compares ProVox to a state-of-the-art passive, user-agnostic system [12] grounded in the lunch bag packing scenario depicted 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%), helpfulness (+18.4%), and 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 [16–19], gestures [20–22], visually-grounded sketches of behaviors [23, 24], or other actions they take [25], amongst others. 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 [26–29], physical interventions [30–32], or other custom interfaces for shared control [33–35]. More recently, work such as InnerMonologue [15], MOSAIC [11], and Vocal Sandbox [12] extend such methods to develop fully-fledged systems that connect diverse modalities for interaction and teaching such as spoken dialogue, physical demonstrations, and visual keypoints to build collaborative robots that can sustain long-horizon interaction for predefined tasks (e.g., cooking a meal subject to a fixed recipe). We build ProVox on top of these works, specifically extending the Vocal Sandbox [12] framework with the ability to generalize to users with distinct goals and preferences while also enabling proactive planning (Fig. 2). We provide further detail about how we build on Vocal Sandbox in this work in §III.

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) such as GPT-4 and PaLM [13, 36, 37]. These approaches often use LMs for *task planning*, mapping complex user instructions to structured intermediate representations [38–40] that are used to inform robot behavior. While early work choose intermediate representations such as simpler language subtasks [15, 41] or parameterized reward functions [42, 43], many recent methods formalize task plans as executable programs [14, 44–47], using LMs to generate sequences of function calls subject to a predefined API – for 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 [48–50], 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; 14], 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 [11, 12]. Indeed, this need to "naturalize" [51] a planning interface to a given person is the motivation for the meta-prompting contributions we make in this work. We provide more details as to how we construct such an interface in §IV.

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 [52–54] 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 [34, 55, 56] 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.

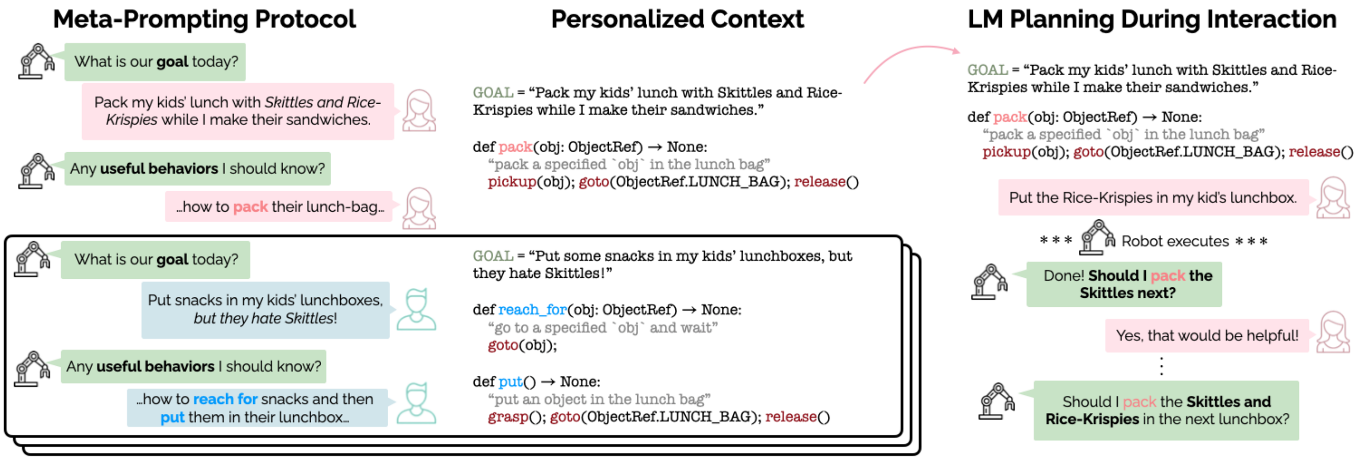
III. Problem Formulation

ProVox builds on Vocal Sandbox [12], a framework for situated collaboration that allows human partners to teach robots new behaviors and concepts *online*, during the course of an interaction. In this section, we formalize and highlight the key mechanisms of Vocal Sandbox, while the following section (§IV) introduces our novel contributions for enabling 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. As described in §II, 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 the same arguments in the function signature; "executing" a function simply means rolling out the corresponding skill with the provided arguments (e.g., `goto(LUNCH_BAG)`). While the formalism in Grannen et al. [12] permits arbitrary argument types and skill implementations, this work assumes a tabletop manipulation setting, where arguments refer to a single object in the scene (e.g., `LUNCH_BAG` or `GUMMY_CANDY`) and skills are engineered motion primitives; we provide more detail about how we implement these primitives in §V.

A critical affordance uniquely provided by Vocal Sandbox is the ability for human collaborators to use language and other feedback modalities such as gestures or physical demonstrations to teach the robot new behaviors (i.e., functions) and concepts (i.e., arguments) 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 robot's 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 *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 brand new function `pack(obj: ObjectRef)` – with the appropriate



What is our **goal** today?

Pack my kids' lunch with *Skittles and Rice-Krispies* while I make their sandwiches.

Any **useful behaviors** I should know?

..how to **pack** their lunch-bag..

```
GOAL = "Pack my kids' lunch with Skittles and Rice-
Krispies while I make their sandwiches."

def pack(obj: ObjectRef) → None:
  "pack a specified `obj` in the lunch bag"
  pickup(obj); goto(ObjectRef.LUNCH_BAG); release()
```

What is our **goal** today?

Put snacks in my kids' lunchboxes, *but they hate Skittles!*

Any **useful behaviors** I should know?

..how to **reach for** snacks and then **put** them in their lunchbox..

```
GOAL = "Put some snacks in my kids' lunchboxes, but
they hate Skittles!"

def reach_for(obj: ObjectRef) → None:
  "go to a specified `obj` and wait"
  goto(obj);

def put() → None:
  "put an object in the lunch bag"
  grasp(); goto(ObjectRef.LUNCH_BAG); release()
```

```
GOAL = "Pack my kids' lunch with Skittles and Rice-
Krispies while I make their sandwiches."

def pack(obj: ObjectRef) → None:
  "pack a specified `obj` in the lunch bag"
  pickup(obj); goto(ObjectRef.LUNCH_BAG); release()
```

Put the Rice-Krispies in my kid's lunchbox.

* * * Robot executes * * *

Done! **Should I pack the Skittles next?**

Yes, that would be helpful!

⋮

Should I pack the **Skittles and Rice-Krispies** in the next lunchbox?

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, as well as an API of useful behaviors. Crucially, each user has a distinct set of preferences, yielding different goals and behaviors. For example, the female user [**Top-Left**] wants to her children's lunch to contain Skittles, Rice-Krispies, and hand sanitizer, and teaches the robot to pack objects, with full confidence in its ability to identify, grasp, and move objects to the lunch bag. In contrast, the male user [**Bottom-Left**] is more hesitant in trusting the robot; as a result, 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 behaviors over the course of the interaction [**Right**].

type signature and documentation – adding it to the API so that it can immediately be used for the remainder of the interaction.

While Grannen et al. [12] introduce teaching-via-synthesis for the express goal of allowing users to define progressively more complex behaviors, we note that this procedure is much more general in scope. By providing alternative utterances for the same function or program (e.g., "add the Rice Krispies treat to the bag" instead of "pack"), or new referring expressions that map to the same argument (e.g., "cereal bar" instead of "Rice Krispies treat"), we unlock the ability to alias and reindex existing behaviors and concepts – a feature that we use to develop the personalization abilities of ProVox in §IV.

B. Formalizing Task Planning and Teaching

Vocal Sandbox formalizes task planning as conditional language generation. 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 $h_t = [(u_1, p_1), (u_2, p_2), \ldots (u_{t-1}, p_{t-1})]$, the current API Λ_t, and a global collaboration goal prompt u_{fixed}. Critically, u_{fixed} is hand-designed by Grannen et al. [12] 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. We note that 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 Fig. 2), 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}. Concretely, the output of the synthesis step is a new function name (e.g., `pack`), type signature (e.g., `(obj: ObjectRef)` → `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 "lifted" function body and type signature from program decomposition $\hat{p}_t$, Vocal Sandbox employs prior unification-based algorithms for program induction [51, 57, 58], while the language model LM_θ generates both the function name and documentation from the interaction. Note that beyond defining new functions, a near-identical procedure can be used to teach new arguments (i.e., novel objects in the scene) – see Grannen et al. [12] and the associated language modeling code on our project page for further detail.

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 uses the resulting prompt to suggest 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 [11, 12, 44] 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 that 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 bag, 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 bag (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 [60–62]. A user starts by describing their individual goal and preferences in natural language (i.e., the `GOAL` field in Fig. 3; Middle), setting u^k_{prompt} to the resulting string. 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?"). If the LM fails to generate a satisfying plan, users have the option to explicitly teach new API functions themselves, fully specifying the function name, expected behavior (documentation), and function body through a drop-down menu. Note that giving users explicit insight into the API in this manner is a key difference from the LM-driven function synthesis procedure used by Grannen et al. [12]; we evaluate the benefits of doing this through user studies in §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 $h_t = [(u_1, p_1), (u_2, p_2), \ldots (u_{t-1}, p_{t-1})]$ when using the LM to generate plans, instead conditioning on the empty history $h_t = []$. 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 defined in §III-B.

Given a meta-prompted context (a goal u^k_{prompt} and an API Λ^k), we seed the LM planner with the information to be used downstream for proactive suggestion synthesis, as shown in Fig. 3 (Right). We evaluate the impact of meta-prompting relative to global prompting and other ablations via an isolated, component-wise user study in §VI-A, and the impact of meta-prompting in the context of a full system user study in §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. This is again a limitation stemming from the expensive upfront cost of having new users build a mental model and "naturalize" [51, 63] 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 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 planner again by prompting it with the following trigger string: "Propose an action to perform next to perform [user-provided goal u^k_{prompt}]." We do this continually at each interaction step, using the full interaction history (and potentially updated API) to better shape the planner's suggested behaviors. For safety and transparency, we do not automatically execute the proposed plans, but rather gate on user confirmation. We evaluate the impact of proactivity through a full system user study in §VI-B, evaluating ProVox against a non-active Vocal Sandbox baseline, measuring the efficiency of the collaboration as well as qualitative metrics such as the ease of use, predictability, and perceived helpfulness of each system.

V. Implementation & Reproducibility

In this work, we implement the ProVox framework mostly following the design decisions in Grannen et al. [12] 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 [64], we adopt the more recent GPT-4 Turbo [v04-09-2024; 65] as our base language model LM_θ; we provide a cursory analysis of qualitative differences between the two generations of language models in §VII. 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 project page. As in Grannen et al. [12] we format the API as an object-oriented API implemented in Python, formatted as a Markdown code block. To constrain the language model to generate valid programs, we use the function-calling capabilities provided by the OpenAI text completion endpoint [50].

Robot Platform & Motion Primitives. For our robot platform, we use a Franka Emika Panda fixed-arm manipulator equipped with a Robotiq 2F-85 gripper following the hardware specification in Khazatsky et al. [66]. We also assume an

[2] The graphical interface was implemented using Gradio [59]; we provide figures and code on our project page: https://provox-hri-2025.github.io.

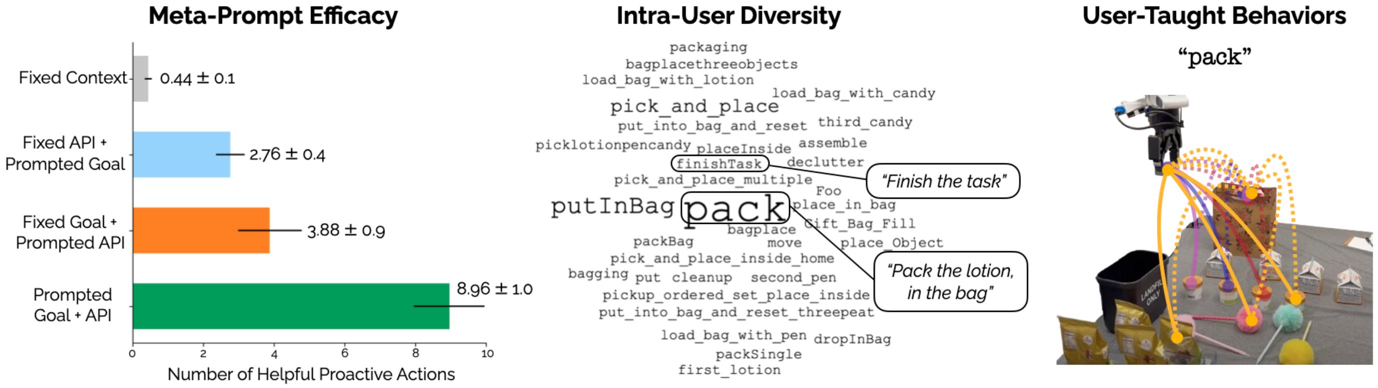


Fig. 4. **Meta-Prompting User Study Results.** We present results evaluating the efficacy of ProVox's meta-prompt protocol through a $N = 26$ user study, as well as visualize the diversity of user-defined behaviors. We report the mean and standard error values of the number of helpful, proactive actions 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!

extrinsics-calibrated ZED 2 stereo camera to obtain point cloud observations for guiding manipulation. As participants share the same physical workspace as the robot, 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, and one controlled 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 by employing off-the-shelf vision models following Grannen et al. [12], first obtaining a 2D segmentation mask via FastSAM [67], 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 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 $u_{\text{fixed}} =$ "to help the user with a tabletop task," but using the 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 $p_{\text{proactive}}$ following §IV-B. We compare $p_{\text{proactive}}$ to the ground truth task plan $p_{\text{reference}}$ depicted in the initial robot demonstration, reporting the count of overlapping function invocations (individual be-

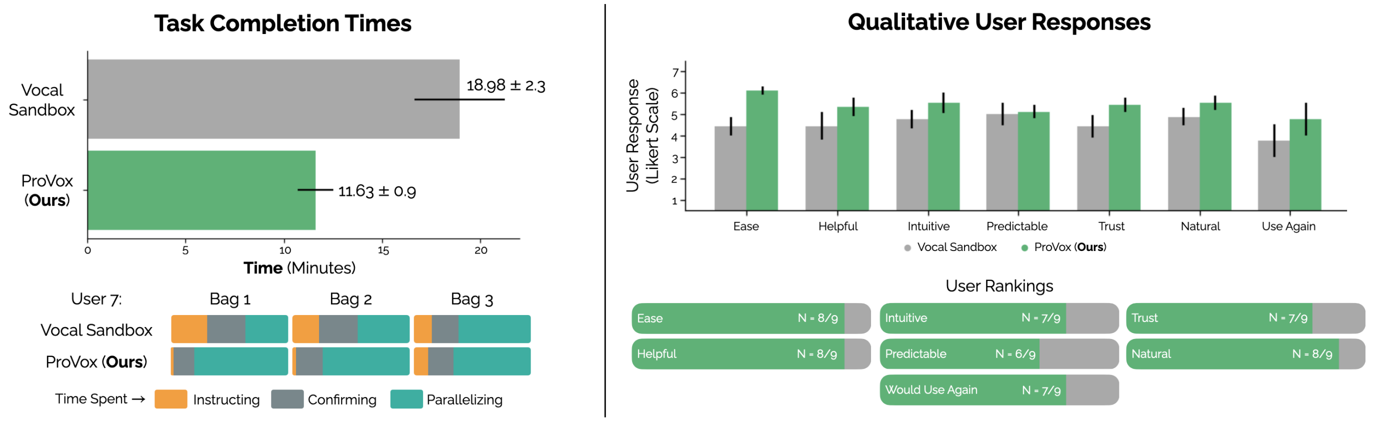


Fig. 5. **Full System User Study Results.** We present quantitative and qualitative results from our real-world user study of collaborative lunch bag packing. We report the average and standard error task completion times for users collaborating with a ProVox or a Vocal Sandbox robot system to pack three lunch bags [**Top, Right**]. We observe that the ProVox collaboration achieves faster task completion, and 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 about their interaction. We additionally ask users to explicitly rank which of the two methods they favor across these seven categories [**Bottom, Right**]. We observe for both the user responses that users strongly favor ProVox over the baseline collaborator in terms of ease, helpfulness, trustworthiness, and willingness to use again.

haviors represented in the reference video) as *Meta-Prompt Efficacy*. Intuitively, large overlaps indicate that the proactive planner – and by extension, the corresponding meta-prompting method – is able to perfectly suggest helpful actions.

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

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. We constrain the study such that 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 preparing 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 [12]; 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 our 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 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 [12]. That said, ProVox is only a start, with multiple avenues for future work.

Towards Better Language Model Task Planners? One of the peculiarities that surfaced in early pilot studies was an inconsistency in performance between different language models – specifically between the (now deprecated) GPT-3.5 Turbo [64] and the more recent GPT-4 Turbo [v04-09-2024; 65]. Specifically, even though GPT-4 Turbo is a strictly better model in terms of traditional language and coding benchmarks such as MMLU and HumanEval [68, 69], we found it disproportionally *worse* than GPT-3.5 at our specific teaching-via-synthesis subroutine. For example, we found GPT-4 Turbo failed to properly lift simple programs to modular abstractions (e.g., synthesizing a function `put_in`), instead synthesizing overly specific functions that are either irrelevant or just completely incorrect (e.g., `put_candy_in_candy`). In general, an open area of research is evaluating the stability of different LLMs across generations for applications beyond language – especially as we work towards integrating other foundation models into HRI systems.

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 real-time gestures. In general, a key limitation of ProVox is its reliance on an *ungrounded* language model that cannot perceive the physical world; this opens up an interesting avenue of future work for integrating vision-language models [70, 71] or even vision-language-action models [72, 73] for human-robot collaboration.

References

[1] Guy Hoffman and Cynthia Lynn Breazeal. Collaboration in human-robot teams. *AIAA 1st Intelligent Systems Technical Conference*, 2004. URL https://api.semanticscholar.org/CorpusID:1114471.

[2] Arash Ajoudani, Andrea Maria Zanchettin, Serena Ivaldi, Alin O. Albu-Schäffer, Kazuhiro Kosuge, and Oussama Khatib. Progress and prospects of the human–robot collaboration. *Autonomous Robots*, 42:957 – 975, 2017. URL https://api.semanticscholar.org/CorpusID:21722736.

[3] Jake Brawer, Olivier Mangin, Alessandro Roncone, Sarah Widder, and Brian Scassellati. Situated human–robot collaboration: predicting intent from grounded natural language. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 827–833, 2018. URL https://api.semanticscholar.org/CorpusID:51998195.

[4] C.A. Stanger, C. Anglin, W.S. Harwin, and D.P. Romilly. Devices for assisting manipulation: a summary of user task priorities. *IEEE Transactions on Rehabilitation Engineering*, 2(4):256–265, 1994. doi: 10.1109/86.340872.

[5] J. Krüger, T.K. Lien, and A. Verl. Cooperation of human and machines in assembly lines. *CIRP Annals*, 58(2): 628–646, 2009. ISSN 0007-8506. doi: https://doi.org/10.1016/j.cirp.2009.09.009.

[6] Abdel-Nasser Sharkawy. A survey on applications of human-robot interaction. *Sensors & Transducers*, 251 (4):19–27, 2021.

[7] Y. Rivero-Moreno, S. Echevarria, C. Vidal-Valderrama, L. Pianetti, J. Cordova-Guilarte, J. Navarro-Gonzalez, J. Acevedo-Rodríguez, G. Dorado-Avila, L. Osorio-Romero, C. Chavez-Campos, and K. Acero-Alvarracín. Robotic surgery: A comprehensive review of the literature and current trends. *Cureus*, 15(7)(e42370), 2023. doi: https://doi.org/10.7759/cureus.42370.

[8] Lorenzo Shaikewitz, Yilin Wu, Suneel Belkhale, Jennifer Grannen, Priya Sundaresan, and Dorsa Sadigh. In-mouth robotic bite transfer with visual and haptic sensing. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2023.

[9] Mariagrazia Costanzo, Rossana Smeriglio, and Santo Di Nuovo. New technologies and assistive robotics for elderly: A review on psychological variables. *Archives of Gerontology and Geriatrics Plus*, 1(4):100056, 2024. ISSN 2950-3078. doi: https://doi.org/10.1016/j.aggp.2024.100056.

[10] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *Conf. on Robot Learning (CoRL)*, 2022.

[11] Huaxiaoyue Wang, K. 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.

[12] Jennifer Grannen, Siddharth Karamcheti, Suvir Mirchandani, Percy Liang, and Dorsa Sadigh. Vocal Sandbox: Continual learning and adaptation for situated human-robot collaboration. In *Proceedings of the 8th Conference on Robot Learning (CoRL), November 2024*, 2024.

[13] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Benjamin Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simòn Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Raphael Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Lukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Ryan Kiros, Matthew Knight, Daniel Kokotajlo, Lukasz Kondraciuk, Andrew Kondrich, Aris Konstantini-

dis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Adeola Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel P. Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mèly, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Ouyang Long, Cullen O'Keefe, Jakub W. Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alexandre Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Pondè de Oliveira Pinto, Michael Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario D. Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin D. Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas A. Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Ceròn Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[14] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2023.

[15] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *Conf. on Robot Learning (CoRL)*, 2022.

[16] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2011.

[17] Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning. In *International Conference on Machine Learning (ICML)*, pages 1671–1678, 2012.

[18] Stefanie Tellex, Nakul Gopalan, Hadas Kress-Gazit, and Cynthia Matuszek. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1):25–55, 2020.

[19] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Krzysztof Choromanski, Tianli Ding, Danny Driess, Chelsea Finn, Peter R. Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil J. Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Sergey Levine, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Pierre Sermanet, Jaspiar Singh, Anika Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Ho Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Ted Xiao, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[20] Cynthia Matuszek, Liefeng Bo, Luke Zettlemoyer, and Dieter Fox. Learning from unscripted deictic gesture and language for human-robot interactions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

[21] David Whitney, Miles Eldon, John Oberlin, and Stefanie Tellex. Interpreting multimodal referring expressions in real time. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3331–3338, 2016. doi: 10.1109/ICRA.2016.7487507.

[22] Li-Heng Lin, Yuchen Cui, Yilun Hao, Fei Xia, and Dorsa Sadigh. Gesture-informed robot assistance via foundation models. In *Conference on Robot Learning*, 2023. URL https://api.semanticscholar.org/CorpusID:261557194.

[23] Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montse Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Ho Vuong, and Ted Xiao. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *ArXiv*, abs/2311.01977, 2023. URL https:

//api.semanticscholar.org/CorpusID:265018996.

[24] Priya Sundaresan, Quan Ho Vuong, Jiayuan Gu, Peng Xu, Ted Xiao, Sean Kirmani, Tianhe Yu, Michael Stark, Ajinkya Jain, Karol Hausman, Dorsa Sadigh, Jeannette Bohg, and Stefan Schaal. Rt-sketch: Goal-conditioned imitation learning from hand-drawn sketches. *ArXiv*, abs/2403.02709, 2024. URL https://api.semanticscholar.org/CorpusID:266369965.

[25] Dorsa Sadigh, Shankar Sastry, Sanjit A. Seshia, and Anca D. Dragan. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and Systems*, 2016. URL https://api.semanticscholar.org/CorpusID:7087988.

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

[27] 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. 2022.

[28] 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. In *ACM/IEEE International Conference on Human Robot Interaction (HRI)*, 2023.

[29] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections. *ArXiv*, abs/2403.12910, 2024. URL https://api.semanticscholar.org/CorpusID:268532135.

[30] Andrea Bajcsy, Dylan P. Losey, M. O'Malley, and A. Dragan. Learning robot objectives from physical human interaction. In *Conference on Robot Learning (CoRL)*, 2017.

[31] Dylan P Losey, Craig G McDonald, Edoardo Battaglia, and Marcia K O'Malley. A review of intent detection, arbitration, and communication aspects of shared control for physical human-robot interaction. *Applied Mechanics Reviews*, 70, 2018.

[32] Mengxi Li, Alper Canberk, Dylan P. Losey, and Dorsa Sadigh. Learning human objectives from sequences of physical corrections. In *International Conference on Robotics and Automation (ICRA)*, pages 2877–2883, 2021.

[33] Brenna D Argall. Autonomy in rehabilitation robotics: an intersection. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:441–463, 2018.

[34] Shervin Javdani, Henny Admoni, Stefania Pellegrinelli, Siddhartha S Srinivasa, and J Andrew Bagnell. Shared autonomy via hindsight optimization for teleoperation and teaming. *International Journal of Robotics Research (IJRR)*, 37:717–742, 2018.

[35] Dylan P. Losey, Hong Jun Jeon, Mengxi Li, Krishna Parasuram Srinivasan, Ajay Mandlekar, Animesh Garg, Jeannette Bohg, and Dorsa Sadigh. Learning latent actions to control assistive robots. *Autonomous Robots (AURO)*, pages 1–33, 2021.

[36] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, A. Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, B. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, M. Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, S. Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, D. Luan, Hyeontaek Lim, Barret Zoph, A. Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, T. S. Pillai, Marie Pellat, Aitor Lewkowycz, E. Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, K. Meier-Hellstern, D. Eck, J. Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling language modeling with pathways. *arXiv*, 2022.

[37] Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantòn Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundations and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[38] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Jayant Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego M Reyes, Pierre

Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[39] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530, 2022. URL https://api.semanticscholar.org/CorpusID:252519594.

[40] Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2motion: from natural language instructions to feasible plans. *Autonomous Robots*, Nov 2023. ISSN 1573-7527. doi: 10.1007/s10514-023-10131-7. URL https://doi.org/10.1007/s10514-023-10131-7.

[41] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Anand Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Ho Vuong, F. Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for real-world control at scale. In *Robotics: Science and Systems (RSS)*, 2023.

[42] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Manfred Otto Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and F. Xia. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

[43] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *ArXiv*, abs/2310.12931, 2023. URL https://api.semanticscholar.org/CorpusID:264306288.

[44] Peiqi Liu, Yaswanth Orru, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Ok-robot: What really matters in integrating open-knowledge models for robotics. In *Proc. Robotics: Science and Systems (RSS)*, 2024.

[45] Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas A. Funkhouser. Tidybot: Personalized robot assistance with large language models. In *International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[46] 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. In *Conference on Robot Learning (CoRL)*, 2023.

[47] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. In *Proc. Robotics: Science and Systems (RSS)*, 2024.

[48] Wenlong Huang, Fei Xia, Dhruv Shah, Danny Driess, Andy Zeng, Yao Lu, Pete Florence, Igor Mordatch, Sergey Levine, Karol Hausman, and Brian Ichter. Grounded decoding: Guiding text generation with grounded models for robot control. In *Proc. Advances in Neural Information Processing Systems*, 2023.

[49] Montse Gonzalez Arenas, Ted Xiao, Sumeet Singh, Vidhi Jain, Allen Ren, Quan Vuong, Jake Varley, Alex Herzog, Isabel Leal, Sean Kirmani, Mario Prats, Dorsa Sadigh, Vikas Sindhwani, Kanishka Rao, Jacky Liang, and Andy Zeng. How to prompt your robot: A promptbook for manipulation skills with code as policies. *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4340–4348, 2024. URL https://api.semanticscholar.org/CorpusID:271800417.

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

[51] Sida I. Wang, Sam Ginn, Percy Liang, and Christopher D. Manning. Naturalizing a programming language via interactive learning. In *Association for Computational Linguistics (ACL)*, 2017.

[52] Ashesh Jain, Shikhar Sharma, Thorsten Joachims, and Ashutosh Saxena. Learning preferences for manipulation tasks from online coactive feedback. *The International Journal of Robotics Research*, 34:1296 – 1313, 2015. URL https://api.semanticscholar.org/CorpusID:10851113.

[53] Erdem Biyik and Dorsa Sadigh. Batch active preference-based learning of reward functions. In *Conference on Robot Learning (CoRL)*, 2018.

[54] Erdem Biyik, Aditi Talati, and Dorsa Sadigh. Aprel: A library for active preference-based reward learning algorithms. *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 613–617, 2021. URL https://api.semanticscholar.org/CorpusID:237091752.

[55] Stefanos Nikolaidis, Ramya Ramakrishnan, Keren Gu, and Julie A. Shah. Efficient model learning from joint-action demonstrations for human-robot collaborative tasks. *2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 189–196, 2014. URL https://api.semanticscholar.org/CorpusID:9031520.

[56] Dorsa Sadigh, S. Shankar Sastry, Sanjit A. Seshia, and Anca D. Dragan. Information gathering actions over human internal state. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 66–73, 2016. URL https://api.semanticscholar.org/CorpusID:14170743.

[57] Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics (TACL)*, 1:49–62, 2013.

[58] Siddharth Karamcheti, Dorsa Sadigh, and Percy Liang. Learning adaptive language interfaces through decomposition. In *EMNLP Workshop for Interactive and Executable Semantic Parsing (IntEx-SemPar)*, 2020.

[59] Abubakar Abid, Ali Abdalla, Ali Abid, Dawood Khan, Abdulrahman Alfozan, and James Zou. Gradio: Hassle-free sharing and testing of ml models in the wild. *arXiv preprint arXiv:1906.02569*, 2019.

[60] Beck. Test driven development: By example. 2002. URL https://api.semanticscholar.org/CorpusID:262220275.

[61] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55:1 – 35, 2021. URL https://api.semanticscholar.org/CorpusID:236493269.

[62] O. Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. Dspy: Compiling declarative language model calls into self-improving pipelines. *ArXiv*, abs/2310.03714, 2023. URL https://api.semanticscholar.org/CorpusID:263671701.

[63] Stefanos Nikolaidis and Julie Shah. Human-robot teaming using shared mental models. *ACM/IEEE HRI*, 2012.

[64] OpenAI. Introducing ChatGPT and Whisper APIs. https://openai.com/index/introducing-chatgpt-and-whisper-apis/, 2022.

[65] OpenAI. GPT-4 Turbo. https://openai.com/index/new-models-and-developer-products-announced-at-devday/, 2023.

[66] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, P Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Ye Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sung Yul Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean-Pierre Mercat, Abdul Rehman, Pannag R. Sanketi, Archit Sharma, C. Blake Simpson, Quang Ho Vuong, Homer Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Da Ling Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan P. Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosa Maria Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J. Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems (RSS)*, 2024.

[67] Xu Zhao, Wenchao Ding, Yongqi An, Yinglong Du, Tao Yu, Min Li, Ming Tang, and Jinqiao Wang. Fast segment anything, 2023.

[68] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

[69] Anthony Chen, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Evaluating question answering evaluation. In *Workshop on Machine Reading for Question Answering (MRQA)*, 2019.

[70] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023.

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

[72] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alex Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*, 2023.

[73] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and

Chelsea Finn. Openvla: An open-source vision-language-action model. *ArXiv*, abs/2406.09246, 2024. URL https://api.semanticscholar.org/CorpusID:270440391.