Smart Help: Strategic Opponent Modeling for Proactive and Adaptive Robot Assistance in Households

Zhihao Cao[1,2,*], Zidong Wang[1,2,*], Siwen Xie[3], Anji Liu[4], Lifeng Fan[1,✉]

[1] State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI)

[2] Department of Automation, Tsinghua University [3] Yuanpei College, Peking University

[4] Computer Science Department, University of California, Los Angeles

Abstract

Despite the significant demand for assistive technology among vulnerable groups (e.g., the elderly, children, and the disabled) in daily tasks, research into advanced AI-driven assistive solutions that genuinely accommodate their diverse needs remains sparse. Traditional human-machine interaction tasks often require machines to simply help without nuanced consideration of human abilities and feelings, such as their opportunity for practice and learning, sense of self-improvement, and self-esteem. Addressing this gap, we define a pivotal and novel challenge Smart Help, which aims to provide proactive yet adaptive support to human agents with diverse disabilities and dynamic goals in various tasks and environments. To establish this challenge, we leverage AI2-THOR [32] to build a new interactive 3D realistic household environment for the Smart Help task. We introduce an innovative opponent modeling module that provides a nuanced understanding of the main agent's capabilities and goals, in order to optimize the assisting agent's helping policy. Rigorous experiments validate the efficacy of our model components and show the superiority of our holistic approach against established baselines. Our findings illustrate the potential of AI-imbued assistive robots in improving the well-being of vulnerable groups.

1. Introduction

All of us may find ourselves within vulnerable demographics at some point. Throughout the human life span, we confront an array of challenges, whether originating from physical discomfort [12], emotional turmoil [53], or the inevitable march of aging [22], that can hinder our ability to perform even the simplest tasks that we once accomplished effortlessly (e.g., lifting a heavy object). This predicament often triggers a complex emotional response. The fear of being labeled as "special" or "disabled" can evoke feelings of diminished self-esteem, reduced self-efficacy and self-sufficiency, and a sense of personal boundary violation [8, 40]. However, previous research in human-robot interaction has primarily focused on pure cooperation [30, 38, 42, 43, 52, 57], and assistive technologies [4, 7] have been designed to simply take over everything for human users, often disregarding their emotional well-being. Considering the importance of sensitivity and consideration when offering help, we need new assistance technologies with an emphasis not only on the successful completion of the task but also on the recipient's emotional acceptance of the assistance [8, 40]. This introduces a new dimension to the concept of *learning to help* in AI, extending the boundaries beyond technical proficiency and into the realm of empathetic engagement.

To tackle this issue, we propose a novel and pivotal challenge *Smart Help*, aiming to provide both proactive and adaptive support to human agents with diverse disabilities and dynamic goals in different tasks and environments. Fig. 1 exemplifies the concept of the *Smart Help* strategy as applied to an assistive robot aiding a human in a household task. The *Basic Helper* fails to provide proactive assistance due to its inability to infer the user's goals accurately. The *Ordinary Helper* successfully infers the user's goals from observations and offers proactive help, but it lacks adaptability to different user needs. Its simple "take-over" helping strategy can cause a feeling of discomfort in the human user. In contrast, the *Smart Helper* not only infers both the user's goals distribution but also reasons about the user's capability of completing each goal independently. As a result, the *Smart Helper* can identify critical goals that hinder task completion and provide proactive and adaptive help that selectively addresses those bottlenecks. This approach allows the user to accomplish the task successfully and joyfully.

Leveraging AI2-THOR [32], a 3D interactive environ-

*Indicates equal contribution. Work was conducted during Zhihao Cao and Zidong Wang's internships at BIGAI. ✉ Corresponding author: Lifeng Fan (lifengfan@bigai.ai). Our environment, dataset, and codes are available at https://github.com/caozh20/SmartHelp.

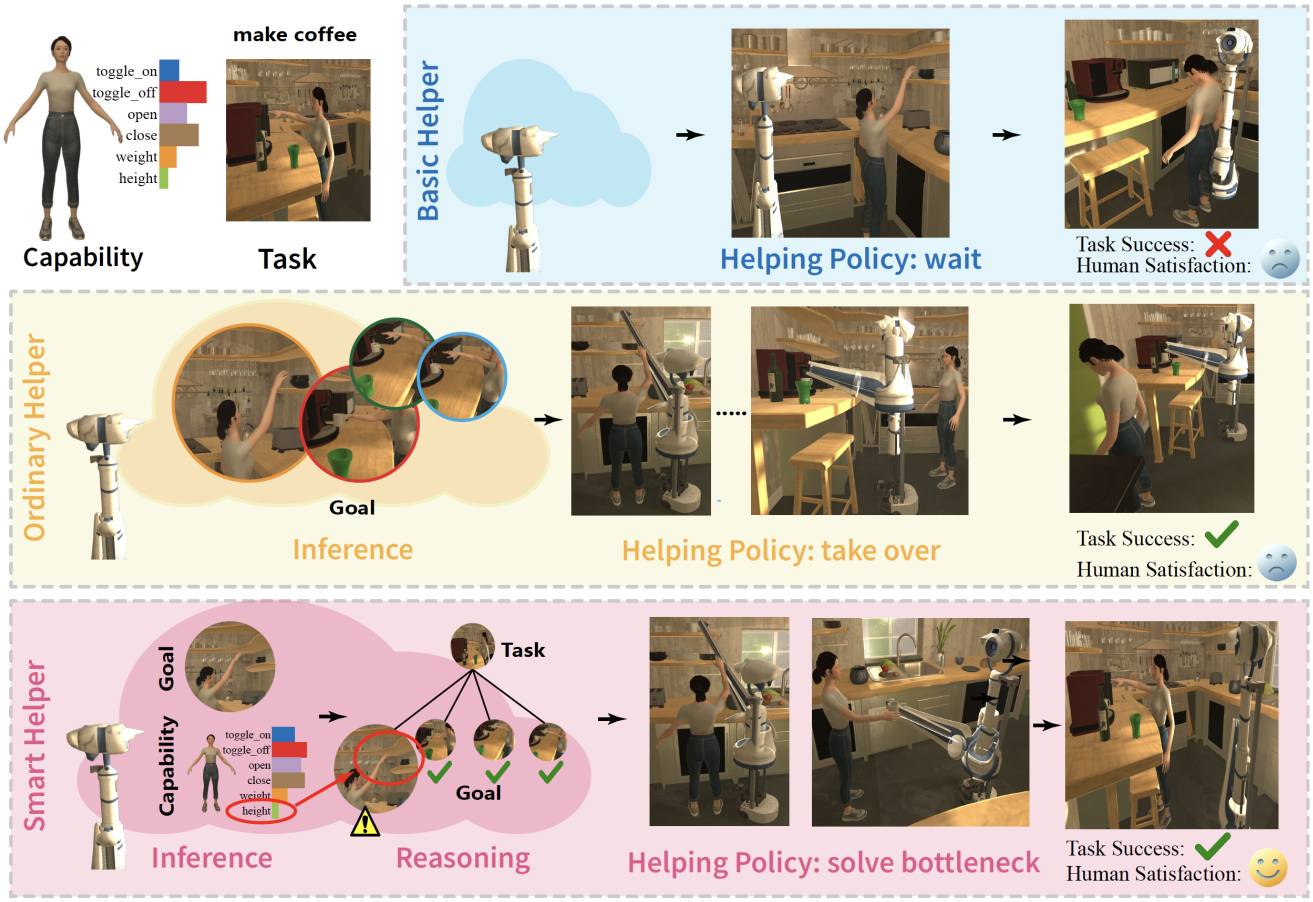

Figure 1. **An example of the *Smart Help* strategy for an assistive robot**. The top left shows the human user's capability distribution across several dimensions (e.g., *weight* for lifting heavy objects and *height* for grasping objects from high positions), as well as the current task (e.g., making coffee). The figure illustrates three types of helpers: (1) The ***Basic Helper*** merely observes the human user without inferring anything, and thus remains idle and waits until the human user fails the task, leading to frustration. (2) The ***Ordinary Helper*** infers the human user's goals through observations and always provides simple and direct assistance for each goal, taking over the entire task. While the task is successfully completed, the human user is left dissatisfied because the robot's helping strategy is overly intrusive, causing discomfort. (3) The ***Smart Helper*** infers both the human user's goals and her capability distribution. It reasons about whether the human user is capable of independently accomplishing each goal and thus identifies the bottleneck goal that hinders task completion. The proactive helper then adapts its strategy to assist solely with the bottleneck goal. As a result, the human user successfully completes the task with tailored support, feeling comfortable and satisfied.

ment engineered for realistic home simulations, we demonstrate a concrete realization of the proposed novel challenge *Smart Help*. We create a multi-agent interactive environment featuring a main agent with diverse capability distributions (representing the vulnerable group) across dimensions, such as toggling, opening, closing, weight, and height. Within this environment, the main agent faces challenges in achieving dynamic goals across various tasks, and an assistive robot is introduced to assist the main agent throughout their endeavors. Unlike previous assistive tasks that were highly specialized and constrained, such as the use of a robotic arm for the disabled [10, 17, 51], our work represents, to the best of our knowledge, the first construction of a 3D home environment designed to assist the vulnerable group with various general daily household tasks.

Furthermore, we build a new benchmark model for our proposed challenge, which consists of 1) an opponent modeling module that jointly optimizes goal inference and capability estimation, and 2) a helping policy module that reasons about the bottleneck, and learns an optimal *Smart Help* policy in an online interactive manner. To better evaluate performance in the *Smart Help* task, we also introduce six assessment metrics. Contrary to traditional evaluation metrics primarily focusing on cumulative reward or task completion, our metrics emphasize the helper's contribution to the task and the essentiality of assistance. Rigorous experiments validate the efficacy of our model components and show the superiority of our holistic approach against baselines. We believe our proposed task, environment, model, and benchmark will contribute to the development of next-generation advanced home assistive robot technologies.

Our **main contributions** can be summarized as follows:

- We propose a novel *Smart Help* challenge that aims at learning to provide both proactive and adaptive help to diverse human users (especially vulnerable groups) based on inferred goals and capabilities.
- To the best of our knowledge, we contribute the first 3D realistic home environment built upon AI2-THOR, that focuses on assisting the vulnerable group with daily household tasks in an online and interactive manner.
- We provide a benchmark model with a joint goal and capability inference, bottleneck reasoning, and helping policy improvement. Strict experiments and the proposed holistic evaluations validate the efficacy of our model.

2. Related Work

Assistive Robots. The domain of assistive robots encompasses a broad spectrum of research, focusing on enhancing the life quality for individuals with various needs and spanning diverse applications such as mobility aids [1, 20], companion robots [46], and robotic arms [21]. Feil-Seifer and Mataric [18] underscore that Socially Assistive Robotics entails robots assisting humans through effective interaction in tasks such as food delivery [49], healthcare [27], and other tasks necessitating social interaction. However, many of them only rely on simple rules or programs to achieve social interaction [27, 49]. The deployment of assistive robots has also raised ethical and safety considerations, particularly in terms of user autonomy, privacy, acceptance, trust, etc. [5, 6, 9, 18]. The objective of our work is to devise an effective algorithm for Socially Assistive Robotics to estimate people's goals and capabilities, thereby enabling comfortable and smart assistance.

Embodied Multi-agent Collaboration. A series of embodied collaboration tasks have been developed recently thanks to the development of Embodied AI simulators [13, 14, 32, 33, 41, 47, 48, 50]. However, these tasks always focus on some limited task completion, such as collaborative navigation [34, 56] and collaborative furniture rearrangement [28, 29]. Watch-and-Help [42] and NOPA [43] study the cooperative tasks between two agents with the same capabilities. Our work focuses on the collaboration between agents with different capabilities and study the strategic assistance for the vulnerable group on household tasks.

Opponent Modeling. Opponent modeling [16, 19, 24], a pivotal approach in multi-agent interaction tasks, leverages a variety of methodologies such as Inverse Reinforcement Learning [36], Bayesian methods [25, 26, 45, 52, 60], Deep Q-Networks (DQN) [23, 35], Variational Auto-Encoders [39], Markov Decision Process [36], etc. This technique is especially valuable in competitive settings like games and strategic decision-making, where understanding and anticipating an opponent's behavior or unseen traits (e.g., Theory of Mind modeling [44]) can significantly influence the outcome. It enables agents to predict future environmental state transitions and refine their strategies accordingly [58, 59]. Projects like Watch-and-Help [42] and NOPA [43] have explored estimating a main agent's goals to improve coordination. Yet, these initiatives often overlook the capabilities of the main agent, leading to a gap in developing effective support strategies for those in need. Our research addresses this gap by focusing on enhancing the acceptance of AI assistance among users, as emphasized in [9], highlighting the importance of aligning AI functionalities with user needs and preferences. Additionally, we propose to model the capability of the opponent and learn a smart, adaptive, and empathetic helping policy for vulnerable people in a challenging 3D household interactive environment with partial observation and high uncertainty.

3. The Smart Help Challenge

3.1. Problem Formulation

We model the interaction between the main agent and the helper with a multi-agent Partially Observable Markov Decision Process (POMDP[11]), which is formally defined as a tuple $G = <S, A, O, R, T, n, \gamma>$. S represents the state space, including physical states and mental states. A is the joint action space for n agents, whose local observations compose observation space O. $T(s'|s, a)$ denotes the transition probability. $R(s, a)$ denotes the shared reward function and $\gamma \in [0, 1)$ is the discount factor. Specifically, there are $n = 2$ agents in our *Smart Help* Challenge, i.e., a main agent and a helper agent. The objective is to train a helper agent to assist the main agent in achieving goals, considering the emotional state of the main agent simultaneously.

In previous work, the reward for the helper agent is:

$$R_h(s, a_h) = R^{g_h}(s, a_h) + \beta R^{g_m}(s, a_h), \tag{1}$$

where the $\beta \in (0, 1)$ is a factor controlling the level of *altruism* of the helper. $R^{g_h}(s, a_h)$ is the reward attributed to the successful completion of the helper agent's goal at state s through action a_h. $R^{g_m}(s, a_h)$ is the same for the main agent's goal achievement. This formulation assumes equal consideration for both agents when rewarding goal completion. However, when the helper fulfills a goal of the main agent, it triggers emotional responses in the main agent. Integrating this emotional component with Eq. (1), we have:

$$\begin{aligned} R_h(s, a_h) = & R^{g_h}(s, a_h) \\ & + \beta(R^{g_m}(s, a_h) + \lambda_e R^{e_m}(s, a_h)), \end{aligned} \tag{2}$$

where $R^{e_m}(s, a_h)$ means the reward correlated to the emotions of the main agent when the helper does the action a_h, and λ_e is a hyper-parameter that controls the helper's sensitivity to the emotional states of the main agent. In our *Smart Help* challenge, the helper agent's role is solely to assist the main agent in accomplishing his/her goals, without its own

goals. So we omit the term $R^{g_h}(s, a_h)$ and set $\beta = 1$, leading to the helper's reward as:

$$R_h(s, a_h) = R^{g_m}(s, a_h) + \lambda_e R^{e_m}(s, a_h). \quad (3)$$

Indeed, this is what sets *smart Help* apart from other assistance-related tasks. Through such design, we emphasize that AI agents must possess the capacity to discern not just how to aid humans, but also when their assistance is truly needed and will be valued.

3.2. Challenge Implementation

We implement a tangible version of *Smart Help* in the AI2THOR [32] simulator, which emphasizes adaptive helping policy for individuals with various disabilities, as shown in Fig. 1. Our new environment includes a main agent that simulates human behavior in various assistive household tasks. The main agent, assigned with a new task and a capability distribution at the beginning of each round, attempts to complete the task with an expert planning policy, which utilizes complete information to divide the task into distinct goals and plan intentional actions for the goals. We also implement a low-level planner that translates the intentional action to a sequence of executable actions in the simulator, so as to enable the main agent to reach the target state indicated by the input intentional action. The workflow of the low-level planner is a loop involving the following steps:

1. Navigate to the relevant object by following the shortest path determined by the object's position and the room layout; or be directly teleported to the target position.
2. Interact with the object in accordance with the goal.

However, actions taken by the main agent might fail due to certain disabilities or the physical constraints of the scene. Thus, the helper agent, as in Fig. 2, based on its symbolic observations of the world, should infer both the goal and capability of the main agent and provide proactive assistance with the bottleneck for the main agent. We will introduce the detailed environment settings in the following part.

Object State. We define object state of object i as $e_i = (type_i, pos_i, w_i, attr_i, pr_i)$, where:

- $type_i \in \mathbb{N}$ denotes the type of object i with an index;
- $pos_i \in \mathbb{R}^3$ records the position of object i's center;
- $w_i \in \mathbb{R}$ represents the weight of object i;
- $attr_i \in \{0, 1\}^{4+2}$ denotes the attributes of object i, including its 4 status (i.e., whether it is picked up, opened, cooked, and toggled on) and its visibility to the 2 agents;
- $pr_i \in \mathbb{R}$ signifies the type of the parent receptacle of the object i through an index; here, the parent receptacle means an object that serves as the container or the support for another subordinate object.

Action Space. We employ intentional actions to construct the action space, where intentional actions represent the target state that the agent wants to achieve. It is defined as $a_i = (predicate_i, e_i) \in A$; for instance, (PickUp, 3), where 3 is an object index, representing "Bread" in our environment. There are seven kinds of different intentional actions: (Wait), (PickUp, e_i), (Put, pr_i), (ToggleOn, e_i), (ToggleOff, e_i), (Open, e_i), and (Close, e_i). In total, there are $|A| = 373$ possible intentional actions.

Agent Capability. The variance of agent capability is manifested in the differences in the transition matrix, which means that different agents will produce different outcomes for the same action at the same state. For example, some agents may be able to pick an object up, while others cannot due to the its weight. Learning the matrix itself is impractical due to the vast size of the space. Instead, we select six parameters to represent the distinct characteristics of different matrices. Therefore, the capability of agent i is represented as $\eta_i = (\alpha_i, \beta_i, \gamma_i, \delta_i, \epsilon_i, \zeta_i)$ where:

- The maximum height an agent can reach $\alpha_i \in [0, 1]$ and the maximum weight an agent can lift $\beta_i \in [0, 1]$ determine whether an agent can execute the action *PickUp*;
- $\gamma_i, \delta_i, \epsilon_i, \zeta_i \in [0, 1]$ respectively represent the agent's ability to complete the actions *Open*, *Close*, *ToggleOn* and *ToggleOff*.

Agent State. We define the state of agent i as $h_i = (\eta_i, pos_i, rot_i, e_i, a_i, succ_i)$, wherein:

- $\eta_i \in C = [0, 1]^6$ denotes the capability of agent i,
- $pos_i, rot_i \in \mathbb{R}^3$ represent agent i's position and rotation,
- $e_i \in E$, $a_i \in A$, and $succ_i \in \{0, 1\}$ respectively represent the object held by agent i, the action undertaken by agent i in the last step, and whether the last action was a success or a failure.

World. The world is denoted as $W = (S, A, T)$, wherein:

- $S = \{e_i, h_j\}_{i=1,\dots,N_o, j=1,2}$ represents the world state, which encompasses the state of N_o objects and 2 agents;
- $A = \{a_i\}$ denotes the action space;
- $T = P(S'|S, A; \eta_i)$ refers to the transition matrix controlled by agent capability.

Task. Given our formulation of agent capabilities, we select three everyday tasks for practical implementations of our challenge: *Make Breakfast*, *Arrange Room*, and *Make Coffee*. These tasks respectively involves different capabilities: $(\alpha_i, \beta_i, \gamma_i, \delta_i, \epsilon_i, \zeta_i)$ for *Cook Potato*, $(\alpha_i, \beta_i, \gamma_i)$ for *Arrange Room*, and $(\alpha_i, \beta_i, \epsilon_i, \zeta_i)$ for *Make Coffee*.

Goal. Upon setting a task, the agent need to identify several key points (goals) that they must achieve. These goals are hidden from the helper, but affect the helper's reward for a particular action. Each goal is defined by the target state of the involved objects. The goal at time t is represented as:

$$g^t = (\text{target\_state}^t, e^t, pr^t) \quad (4)$$

Target state includes *Wait*, *Get*, *On*, *In*, *KeepOpen*, *KeepClose*, *KeepOn* and *KeepOff*, e.g., "(In, Cup, Cabinet)".

Observation Space. The proposed environment supports both symbolic and visual observations. This dual-mode observation system enables the helper to acquire and refine

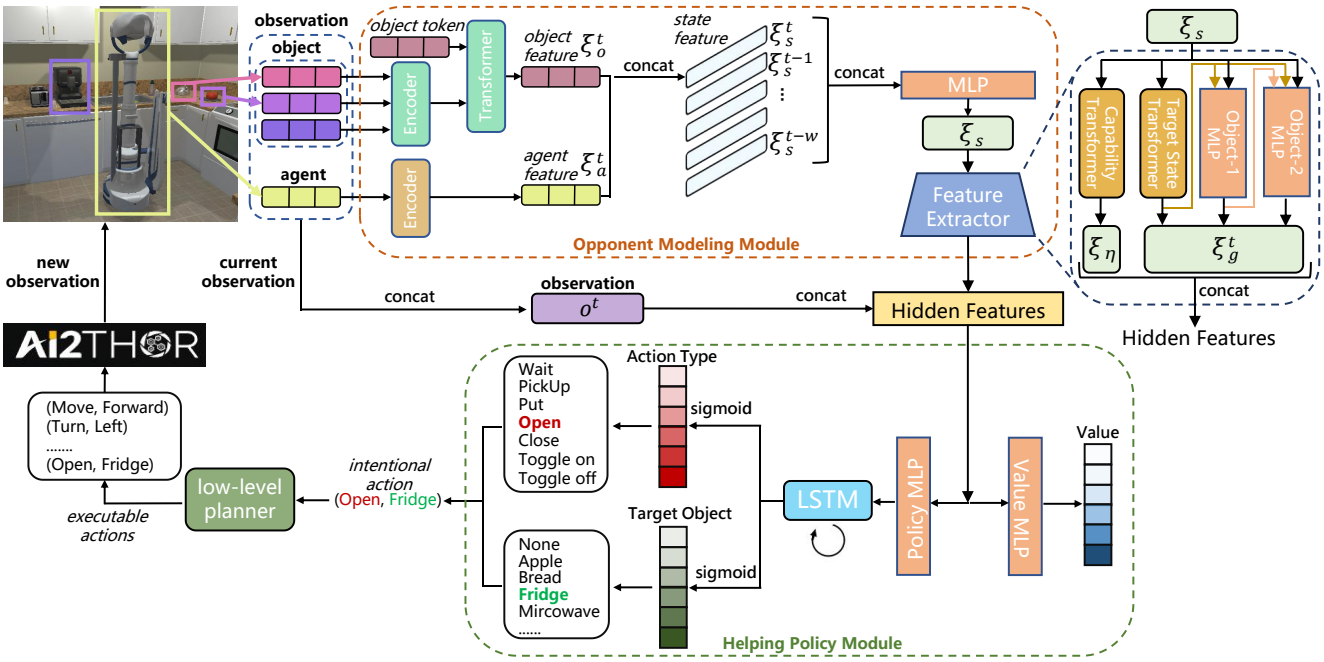

Figure 2. This figure depicts the architecture of our *smart help* model and its interaction mechanisms with the environment. The model is partitioned into two primary components: an opponent modeling module and a helping policy module. The opponent modeling module is designed to estimate the goal and capability of the main agent. It adopts a sliding window paradigm and utilizes several MLP layers to generate the state feature ξ_s. This feature is then processed by a feature extractor to derive the capability feature ξ_η and the goal feature ξ_g^t. The helping policy module is composed of two MLP layers to produce action decision and value estimation respectively. During interaction with the environment, the helping model outputs an intentional action, representing the target state it aims to reach. The environment employs a low-level planner to decompose the intentional action into a sequence of executable actions, which are then processed by the AI2THOR simulator, guiding the assistant toward the intended state. Finally, the simulator provides a new observation as feedback, triggering the next cycle of interaction.

helping behaviors in various contexts. Following previous work [42, 43], we use symbolic observations containing physical states of all perceivable objects and agents. Hidden states, like goals and capabilities, can not be observed.

4. Our Model

Our objective is to train a helper agent who can effectively adapt to main agents with varying capabilities and goals. There are two main challenges: 1) the challenge posed by the environment (e.g., partial observation and occlusion), and 2) the challenge posed by the task (i.e., how to estimate the current goal g^t and capabilities η of the main agent). Given the observation $o^{0:t}$, the action policy for the helper is formulated as:

$$P(a|o^{0:t}) = \sum_{g^t \in G, \eta \in C} P(a|g^t, \eta, o^{0:t}) P(g^t, \eta|o^{0:t}). \quad (5)$$

Therefore, as depicted in Fig. 2, our model consists of: 1) an opponent modeling module to estimate the current goal and capabilities of the main agent, i.e., $P(g^t, \eta|o^{0:t})$; and 2) a helping policy module to learn the distribution of the helper's action conditioned on the predicted goal and capabilities of the main agent, i.e., $P(a|g^t, \eta, o^{0:t})$.

The training of our model has two stages. In the first stage, we train an opponent modeling module using the collected main agent trajectories. This stage is crucial as it enables the helper agent to understand and anticipate the behavior of the main agent effectively. In the second stage, we utilize the pre-trained opponent modeling module to train the helping policy module, which handles dynamic and complex tasks by interacting in our environment.

4.1. The Opponent Modeling Module

To train this module, we collect simulated trajectories in our environment with the main agent following an expert policy and the helper moving randomly, where each frame contains the observations of the helper agent and the true labels of the main agent's goals and capabilities. Throughout data collection, we utilize a main agent with various tasks and capabilities and a helper executing random actions in the environment. See more details in the supplementary.

We use a sliding window trick when inferring goals and capabilities from the main agent's trajectory since goals are transient and typically persist for only a few steps. This

method enables examining a segment of the trajectory at any given time, and sliding the window as new actions are taken. It allows the helper agent to make more immediate and relevant inferences about the main agent's current goals and capabilities, enhancing its adaptability and responsiveness. Formally, we have:

$$P(\eta_i, g^t | o^{0:t}) \approx P(\eta_i, g^t | o^{t-w:t}), \tag{6}$$

where w is the window size.

We use an object encoder and an agent encoder to extract features from raw observations (details in supplementary). Similar to the "class token" in [2], we prepend a learnable embedding, which we call "object token" to the sequence of object feature. We follow the usage of "class token" [2], and only retain this object token to represent the object feature ξ_o^t after the transformer [54] layer (other tokens are removed). This object feature ξ_o^t, together with the agent feature ξ_a^t, constitute the state feature ξ_s^t. As in Eq. (6), we use a window size of w and the state features $\{\xi_s^{t-w}, \cdots, \xi_s^t\}$ are fed into an MLP layer to obtain the final state feature ξ_s. Furthermore, a feature extractor is employed to derive the capability feature ξ_η and the target state feature ξ_g^t of the main agent. A transformer layer is utilized for the capability feature. As for the goal feature ξ_g^t, as detailed in Eq. (4), one transformer layer is used for the target state feature, and two MLP layers are employed for the features of the related object and its target parent receptacle.

4.2. The Helping Policy Module

Our helping policy module has an actor-critic [55] architecture. At each time step t, given the current observation o^t, together with all the features ξ_η and ξ_g^t from the opponent modeling module, the helper agent needs to learn the helping policy $\pi_\theta(a|o^t, \eta, g^t)$. We employ an MLP layer for action selection (the "Policy MLP" in Fig. 2), coupled with an LSTM network for context memorization. Meanwhile, another MLP (the "Value MLP" in Fig. 2) is utilized for the value network to estimate the value of the current state.

5. Experimental Setup

5.1. Two-stage Learning

We apply a two-stage training technique to train our model. In the first stage, we use four auxiliary classifiers to respectively train the capability feature ξ_η, target state feature, object feature and parent receptacle feature (i.e., the goal feature ξ_g^t) in a supervised learning manner. These classifiers are lightweight and efficient, utilizing MLPs, and are trained using cross-entropy loss and the Adam optimizer, with a learning rate of 1×10^{-6} and a batch size of 32.

In the second stage, we remove these four classifiers, keep the pre-trained opponent modeling module frozen, directly use the learnt opponent features ξ_η and ξ_g^t, and focus on the training of helping policy module. We use 20 kitchen rooms for helping policy training and 10 for evaluation in AI2-THOR [32]. The RLlib [15] framework and PPO [31] algorithm are applied to train our helping policy module. Unless otherwise stated, we use the default settings of RLlib. During training, the scene is randomly initialized with a new room, a new task, and a new capability distribution of the main agent, at each episode.

We apply a progressive learning technique to train the help policy module. Firstly, we train the helper policy module for 840 epochs, with a constant learning rate of 5×10^{-5} without weight decay and a batch size of 128, using the reward defined in Eq. (3) with $\lambda_e = 0.0$. We find such "warm-up" quite important in helping the helper agent to familiarize itself with basic skills to complete goals and tasks. Secondly, we progressively train the model for 240 epochs with a learning rate of 5×10^{-7} and $\lambda_e = 1.0$ in Eq. (3) to enhance the smart help ability. In the evaluation phase, we test the helper in 10 rooms (never used in training) and with 14 different task-capability pairs, repeating three times with different random seeds. We exclude 7 task-capability pairs as they will enable the main agent to finish the task independently. See supplementary for more details.

5.2. Reward

The reward of the helper is influenced by the goal g_m and the capability η_m of the main agent. During training, the helper agent will get a reward of 20 after finishing a goal of the main agent, corresponding to the first term in Eq. (3). The second term $R^{e_m}(s, a_h)$ in Eq. (3) is influenced by the capability of the main agent. Specifically, if the helper finishes a goal that the main agent can handle independently, the helper will get a punishment: $R^{e_m}(s, a_h) = -30$. For every step, the helper agent will get a cost of -0.12 normally and a cost of -0.5 if the action is illegal (e.g., attempting to open something that can not be opened, such as an apple) in the environment. If the task is not finished when the scenario is ended, the helper will receive a punishment of -20.

5.3. Baselines

To make a fair comparison, the baselines are:

- **Random**. This helper randomly selects an action.
- **End2end-λ_e=0.0.** This model generates actions directly from the observation with MLPs, trained with $\lambda_e = 0.0$ in Eq. (3). We set learning rate to 5×10^{-5}, batch size to 128. The model is trained for 350 epochs before convergence.
- **End2end-λ_e=1.0.** Similar to *End2end*-λ_e=0.0, but with $\lambda_e = 1.0$ in Eq. (3).
- **MCTS.** This model uses a Monte-Carlo Tree Search (MCTS [3]) algorithm to search for an action with the highest value, given a predicted goal from the opponent modeling module. The value of each action is estimated by the status of goal completion after rollout, discounted

by the number of steps taken to achieve this result.

- **MCTS-heuristic.** This model combines the *MCTS* model with expert rules for the three selected tasks. Given a predicted goal from the opponent modeling module, it decides on which action to simulate at a probability $p = 0.5$ by chance and $p = 0.5$ by a rule-based selection of the next action to complete the goal.
- **MCTS$_{\text{TG}}$.** This is an implementation to reproduce the high-level planning policy of Watch-and-Help [42]. This model knows the true goal of the main agent.
- **MCTS$_{\text{RG}}$.** This model is almost the same as *MCTS$_{TG}$*, except that it follows a random goal.
- **LLM.** We use Large Language Model, i.e., *gpt-3.5-turbo-instruct* [37] as the helper. We compose a prompt based on the state observations from the AI2-THOR [32]. The prompt is fed into the LLM at every step. We extract the action decision generated from the LLM and evaluate its actual effectiveness in the AI2-THOR simulator. See supplementary for more details.

5.4. Ablation Study

To assess the contributions and efficacy of essential components in our method, we derive the following variants:

- **BaseModel.** This model is the aforementioned helping policy model with a pre-trained parameter-frozen opponent modeling module. As in Sec. 5.1, we set $\lambda_e = 0.0$ in Eq. (3), and train it for 840 epochs.
- **BaseModel-w/o. Capability.** Here we remove the capability embedding from the training of the helping policy.
- **BaseModel-λ_e=1.0.** This model is trained with $\lambda_e = 1.0$ in Eq. (3) for 840 epochs.
- **BaseModel-PL-λ_e=0.0.** Here, "PL" means the Progressive Learning technique (detailed in Sec. 5.1), but with $\lambda_e = 0.0$ in the second phase after the first warm-up phase. We simply continue to train the *BaseModel* with a smaller learning rate for 240 epochs (with no change in $\lambda_e = 0.0$).
- **BaseModel-PL-λ_e=1.0 (our full model).** We use the correct Progressive Learning technique (detailed in Sec. 5.1), and change to $\lambda_e = 1.0$ in the second phase after warming up the basic skills of the helper model.

5.5. Evaluation Metrics

To objectively evaluate an agent's performance, we utilize six distinct metrics, where *Helping Necessity (HN)* and *Helping Rate (HR)* are first proposed in our work to better assess the *Smart Help* policy. Let N denote the size of the test scenarios, and let each scenario i be characterized by an initial room state s_i^0 and a goal of transitioning the room to the target state s_i^*. Assuming that both the main agent and the helper require l_i steps (with a maximum of 30) to reach the final state s_i^l, we elaborate on these metrics as follows.

- **Success Rate (SR and GSR).** It includes *Task-conditioned Success Rate (SR)* and *Goal-conditioned Success Rate (GSR)*, respectively representing the completion degree of tasks and goals. The *SR* is defined as $SR = \frac{1}{N}\sum_{i=1}^{N} R_i$, where R_i is 1 if the task is finished. For task i, let $N_{g_i^m}$ denote the effective goals completed by the main agent, $N_{g_i^h}$ represent the effective goals completed by the helper, and $N_{g_i^a}$ be all the effective goals of the task. The *goal-conditioned success* for task i is: $GS_i = \frac{N_{g_i^m} + N_{g_i^h}}{N_{g_i^a}}$. Hence, the overall *GSR* is defined as $GSR = \frac{1}{N}\sum_{i=1}^{N} GS_i$. $R_i = 1$ if and only if $GS_i = 1$.
- **Helping Necessity (HN).** This metric reflects the necessity of the helper's involvement. Only when the capability required to finish the goal exceeds the main agent's capability, the helping is necessary. Let $N_{g^h_{\text{nec},i}}$ denote the necessary goals completed by the helper, the Helping Necessity is then calculated as $HN = \frac{1}{N}\sum_{i=1}^{N} \frac{N_{g^h_{\text{nec},i}}}{N_{g_i^h}} R_i$, where $R_i = 1$ if the task is completed, otherwise $R_i = 0$. This assesses the helper's ability to swiftly identify the main agent's capability and provide necessary assistance.
- **Helping Rate (HR).** *HR* represents the helper's initiative to help. $HR = \frac{N_{\text{help}}}{N_{\text{need\_help}}}$, reflecting the probability of helping when the main agent needs help.
- **Episode Length (EL).** We record the average *episode length* to reflect the efficiency of helping policy. It is computed as: $EL = \frac{1}{N}\sum_{i=1}^{N} l_i$.
- **Success-weighted by Path Length (SPL).** We also include *SPL* to have a comprehensive evaluation. It is computed as: $SPL = \frac{1}{N}\sum_{i=1}^{N} R_i \frac{d_i}{\max(d_i, l_i)}$, where $R_i \in \{0, 1\}$ denotes whether the task is successfully completed, d_i represents the minimum number of steps to finish the task i, and l_i is the actual steps.

Beyond these key metrics, we also incorporate the *average rewards* as a metric for evaluation. For the *Smart Help* challenge, we want to increase the *HN* and *HR*, while simultaneously descending the *EL*, as well as keeping the *SR*, *GSR*, and *SPL* as high as possible.

6. Results and Analyses

Comparison with baselines. As shown in Tab. 1, our full model exhibits superior performance compared with all other baseline models without ground truth knowledge of the main agent. The *Random* agent, selecting random actions, and the *MCTS$_{RG}$* model, following random goals, both have minimal ability to provide appropriate help. The superiority of our *BaseModel* compared to the *End2End* model underscores the value of opponent modeling in the context of assistance tasks. The *LLM* agent, although has the shortest *EL*, falls short on other metrics compared with our model. The *MCTS-heuristic* model, which incorporates rule-based heuristics, outperforms the pure *MCTS* model.

(a). BaseModel-PL-λ_e=0 (Normal Help) **(b). BaseModel-PL-λ_e=1 (Smart Help)**

Figure 3. Qualitative results of the learned *smart help* policy. In this example, the main agent moves directly to finish the goal "(In, Potato, Microwave)", but fails at the goal "(Open, Microwave)". (a) The *BaseModel-PL-*λ_e=0.0 model not only helps the main agent with the bottleneck "(Open, Microwave)", but also continues to help with another goal "(ToggleOff, Microwave)". (b) Our full model *BaseModel-PL-*λ_e=1.0 only offer necessary help with the bottleneck "(Open, Microwave)", and let the main agent to finish the rest goals independently, which helping policy is smarter since it considers the needs and emotional feelings of the main agent.

However, our model achieves higher scores in *SR*, *GSR*, *HN*, and *HR*, while maintaining similar values for *EL* and *SPL*, indicating the effectiveness of its assistive action planning. The $MCTS_{TG}$ model, knowing the true goal of the main agent, achieves the best performance among all the models, serving as an upper bound for the task. Notably, while it surpasses other models in most metrics, our model demonstrates competitive performance, particularly in HN. This suggests that our model achieves a balance by providing only necessary assistance and ensuring user comfort. The gap between our model and the upper bound shows potential for future research and further improvement.

Ablation analysis. Comparing the *BaseModel* with *BaseModel-*λ_e=1.0, we find that setting $\lambda_e = 0.0$ can augment the assisting actions of the helper. When compared with *BaseModel-w/o. Capability*, we find that the capability module contributes to the learning of the smart helping policy. Comparing the *BaseModel-PL-*λ_e=0.0 with *BaseModel-PL-*λ_e=1.0, we find that after the initial "warm-up" of the basic skills, setting $\lambda_e = 1.0$ in the second phase could greatly improve the *HN* while maintaining competitive performances across other metrics. Thus, as in Sec. 5.1, for our full model, in the *BaseModel* "warm-up" training phase, the model learns how to complete the goals and tasks, and the assistive actions are greatly encouraged; while in the second progressive learning phase, our full model focuses on improving the assistance strategy of the helper based on the main agent's needs and feelings.

Qualitative Results. Fig. 3 demonstrates two helping cases with a baseline model and our full model. The two models exhibit exploration behaviors in the scene and finally learn to solve the bottleneck problem of the main agent. In comparison, the *BaseModel-PL-*λ_e=0.0 only learns to help all the goals it successfully infers, while our full model *BaseModel-PL-*λ_e=1.0 learns to decide whether to help based on its inferred goals and capabilities, i.e., only to help with "(Open, Microwave)".

Method	SR(↑)	GSR(↑)	HN(↑)	HR(↑)	Reward(↑)	EL(↓)	SPL(↑)
Random	0.074	0.364	0.054	0.057	-31.339	28.567	0.054
End2end-λ_e=0.0	0.267	0.441	0.149	0.315	-23.218	24.343	0.212
End2end-λ_e=1.0	0.067	0.372	0.057	0.057	-26.183	28.726	0.049
LLM	0.345	0.506	0.382	0.381	/	**22.914**	0.210
MCTS	0.178	0.482	0.266	0.328	/	26.743	0.126
MCTS-heuristic	0.274	0.528	0.355	0.400	/	24.692	0.199
$MCTS_{TG}$	0.593	0.804	0.610	0.678	/	16.271	0.541
$MCTS_{RG}$	0.043	0.348	0.015	0.015	/	28.950	0.042
BaseModel	0.419	0.515	0.408	0.412	-15.179	24.945	0.176
BaseModel-w/o. Capability	0.374	0.504	0.375	0.379	-13.852	23.748	**0.235**
BaseModel-λ_e=1.0	0.181	0.466	0.193	0.200	-17.185	27.448	0.122
BaseModel-PL-λ_e=0.0	**0.488**	**0.556**	0.455	0.496	-10.959	24.383	0.198
BaseModel-PL-λ_e=1.0 (Ours)	0.483	0.548	**0.498**	**0.506**	**-10.625**	24.650	0.200

Table 1. The quantitative results of our experiments. The last ten environments in AI2-THOR are reserved for evaluation. The **best results** are highlighted in bold. Note that we include some oracle baselines and the upper bound performances are highlighted with underlines.

7. Conclusion

In this paper, we propose the novel challenge *Smart Help* and build an environment and model that fosters smarter and more harmonious interaction between humans and artificial agents. We hope our challenge, environment, dataset, model and benchmark results will serve as valuable resources to future studies of this important problem.

Acknowledgement This work is supported by the National Science and Technology Major Project (2022ZD0114900). We thank Zhenliang Zhang at BIGAI and Luyao Yuan at META for their helpful discussions.

References

[1] Peter Aigner and Brenan McCarragher. Shared control framework applied to a robotic aid for the blind. *IEEE Control Systems Magazine*, 19(2):40–46, 1999. 3

[2] Dosovitskiy Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, and Mostafa Dehghani et al. An image is worth 16x16 words: Transformers for image recognition at scale. 2020. 6

[3] Browne Cameron B., Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 2012. 6

[4] Roger Bemelmans, Gert Jan Gelderblom, Pieter Jonker, and Luc De Witte. Socially assistive robots in elderly care: a systematic review into effects and effectiveness. *Journal of the American Medical Directors Association*, 13(2):114–120, 2012. 1

[5] Elizabeth Broadbent, Rebecca Stafford, and Bruce MacDonald. Acceptance of healthcare robots for the older population: Review and future directions. *International journal of social robotics*, 1:319–330, 2009. 3

[6] Elizabeth Broadbent, I Han Kuo, Yong In Lee, Joel Rabindran, Ngaire Kerse, Rebecca Stafford, and Bruce A MacDonald. Attitudes and reactions to a healthcare robot. *Telemedicine and e-Health*, 16(5):608–613, 2010. 3

[7] Steven W Brose, Douglas J Weber, Ben A Salatin, Garret G Grindle, Hongwu Wang, Juan J Vazquez, and Rory A Cooper. The role of assistive robotics in the lives of persons with disability. *American Journal of Physical Medicine & Rehabilitation*, 89(6):509–521, 2010. 1

[8] Luca Buoncompagni, Barbara Bruno, Antonella Giuni, Fulvio Mastrogiovanni, and Renato Zaccaria. Towards a new paradigm for assistive technology at home: research challenges, design issues and performance assessment. *arXiv preprint arXiv:1710.10164*, 2017. 1

[9] Martina Čaić, Gaby Odekerken-Schröder, and Dominik Mahr. Service robots: value co-creation and co-destruction in elderly care networks. *Journal of Service Management*, 2018. 3

[10] Anthony Casey, Hannan Azhar, Marek Grzes, and Mohamed Sakel. Bci controlled robotic arm as assistance to the rehabilitation of neurologically disabled patients. *Disability and Rehabilitation: Assistive Technology*, 16(5):525–537, 2021. 2

[11] Anthony R Cassandra, Leslie Pack Kaelbling, and Michael L Littman. Acting optimally in partially observable stochastic domains. In *Aaai*, pages 1023–1028, 1994. 3

[12] Victor J Davila, Andrew J Meltzer, M Susan Hallbeck, William M Stone, and Samuel R Money. Physical discomfort, professional satisfaction, and burnout in vascular surgeons. *Journal of vascular surgery*, 70(3):913–920, 2019. 1

[13] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, and et al. Robothor: An open simulation-to-real embodied ai platform. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3164–3174, 2020. 3

[14] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4495–4504, 2021. 3

[15] Liang Eric, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph Gonzalez, Michael Jordan, and Ion Stoica. Rllib: Abstractions for distributed reinforcement learning. In *International conference on machine learning*, pages 3053–3062. PMLR, 2018. 6

[16] Richard Everett and Stephen Roberts. Learning against non-stationary agents with opponent modelling and deep reinforcement learning. In *2018 AAAI spring symposium series*, 2018. 3

[17] Cheikh Latyr Fall, Philippe Turgeon, Alexandre Campeau-Lecours, Véronique Maheu, Mounir Boukadoum, Sebastien Roy, Daniel Massicotte, Clément Gosselin, and Benoit Gosselin. Intuitive wireless control of a robotic arm for people living with an upper body disability. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 4399–4402. IEEE, 2015. 2

[18] David Feil-Seifer and Maja J Mataric. Defining socially assistive robotics. In *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pages 465–468. IEEE, 2005. 3

[19] Jakob N Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*, 2017. 3

[20] Jared Glover. A robotically-augmented walker for older adults. 2003. 3

[21] B Graf, A Hans, J Kubacki, and RD Schraft. Robotic home assistant care-o-bot ii. In *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society][Engineering in Medicine and Biology*, pages 2343–2344. IEEE, 2002. 3

[22] Denham Harman. Aging: overview. *Annals of the New York Academy of Sciences*, 928(1):1–21, 2001. 1

[23] He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, pages 1804–1813. PMLR, 2016. 3

[24] He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, pages 1804–1813. PMLR, 2016. 3

[25] Pablo Hernandez-Leal and Michael Kaisers. Learning against sequential opponents in repeated stochastic games. In *The 3rd Multi-disciplinary Conference on Reinforcement Learning and Decision Making, Ann Arbor*, page 16, 2017. 3

[26] Pablo Hernandez-Leal, Benjamin Rosman, Matthew E Taylor, L Enrique Sucar, and Enrique Munoz de Cote. A

bayesian approach for learning and tracking switching, non-stationary opponents. In *Proceedings of the 2016 international conference on autonomous agents & multiagent systems*, pages 1315–1316, 2016. 3

[27] Kaoru Inoue, Kazuyoshi Wada, and Yuko Ito. Effective application of paro: Seal type robots for disabled people in according to ideas of occupational therapists. In *Computers Helping People with Special Needs: 11th International Conference, ICCHP 2008, Linz, Austria, July 9-11, 2008. Proceedings 11*, pages 1321–1324. Springer, 2008. 3

[28] Unnat Jain, Luca Weihs, Eric Kolve, Mohammad Rastegari, Svetlana Lazebnik, Ali Farhadi, Alexander G Schwing, and Aniruddha Kembhavi. Two body problem: Collaborative visual task completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6689–6699, 2019. 3

[29] Unnat Jain, Luca Weihs, Eric Kolve, Ali Farhadi, Svetlana Lazebnik, Aniruddha Kembhavi, and Alexander Schwing. A cordial sync: Going beyond marginal policies for multi-agent embodied tasks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 471–490. Springer, 2020. 3

[30] Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *Advances in neural information processing systems*, 31, 2018. 1

[31] Schulman John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 6

[32] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 1, 3, 4, 6, 7

[33] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. *arXiv preprint arXiv:2108.03272*, 2021. 3

[34] Xinzhu Liu, Di Guo, Huaping Liu, and Fuchun Sun. Multi-agent embodied visual semantic navigation with scene prior knowledge. *IEEE Robotics and Automation Letters*, 7(2): 3154–3161, 2022. 3

[35] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. 3

[36] Eshed OhnBar, Kris Kitani, and Chieko Asakawa. Personalized dynamics models for adaptive assistive navigation systems. In *Conference on Robot Learning*, pages 16–39. PMLR, 2018. 3

[37] OpenAI. Gpt-3.5 turbo models. https://platform.openai.com/docs/models/gpt-3-5-turbo. 7

[38] Liviu Panait and Sean Luke. Cooperative multi-agent learning: The state of the art. *Autonomous agents and multi-agent systems*, 11:387–434, 2005. 1

[39] Georgios Papoudakis and Stefano V Albrecht. Variational autoencoders for opponent modeling in multi-agent systems. *arXiv preprint arXiv:2001.10829*, 2020. 3

[40] Monnie Parida and Manjira Sinha. Pandemic and disability: Challenges faced and role of technology. *Technology and Disability*, 33(4):245–252, 2021. 1

[41] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtua lhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502, 2018. 3

[42] Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Yuan-Hong Liao, Joshua B Tenenbaum, Sanja Fidler, and Antonio Torralba. Watch-and-help: A challenge for social perception and human-ai collaboration. *arXiv preprint arXiv:2010.09890*, 2020. 1, 3, 5, 7

[43] Xavier Puig, Tianmin Shu, Joshua B Tenenbaum, and Antonio Torralba. Nopa: Neurally-guided online probabilistic assistance for building socially intelligent home assistants. *arXiv preprint arXiv:2301.05223*, 2023. 1, 3, 5

[44] Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, SM Ali Eslami, and Matthew Botvinick. Machine theory of mind. In *International conference on machine learning*, pages 4218–4227. PMLR, 2018. 3

[45] Benjamin Rosman, Majd Hawasly, and Subramanian Ramamoorthy. Bayesian policy reuse. *Machine Learning*, 104: 99–127, 2016. 3

[46] Nicholas Roy, Gregory Baltus, Dieter Fox, Francine Gemperle, Jennifer Goetz, Tad Hirsch, Dimitris Margaritis, Michael Montemerlo, Joelle Pineau, Jamie Schulte, et al. Towards personal service robots for the elderly. In *Workshop on Interactive Robots and Entertainment (WIRE 2000)*, page 184. Citeseer, 2000. 3

[47] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, and Julian Straub et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019. 3

[48] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. igibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7520–7527. IEEE, 2021. 3

[49] Reid Simmons, Richard Goodwin, Karen Zita Haigh, Sven Koenig, and Joseph O'Sullivan. A layered architecture for office delivery robots. In *Proceedings of the first international conference on Autonomous agents*, pages 245–252, 1997. 3

[50] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, and Noah Maestre et al. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems*, pages 251–266, 2021. 3

[51] Hideyuki Uehara, Hiroki Higa, and Takashi Soken. A mobile robotic arm for people with severe disabilities. In *2010 3rd*

IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics, pages 126–129. IEEE, 2010. 2

[52] Tomer Ullman, Chris Baker, Owen Macindoe, Owain Evans, Noah Goodman, and Joshua Tenenbaum. Help or hinder: Bayesian models of social goal inference. *Advances in neural information processing systems*, 22, 2009. 1, 3

[53] Danielle D Van Jaarsveld, David D Walker, Simon Lloyd D Restubog, Daniel Skarlicki, Yueyang Chen, and Pascale H Frické. Unpacking the relationship between customer (in) justice and employee turnover outcomes: can fair supervisor treatment reduce employees' emotional turmoil? *Journal of Service Research*, 24(2):301–319, 2021. 1

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 6

[55] Mnih Volodymyr, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Silver, , and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *the 33rd International Conference on International Conference on Machine Learning*, pages 1928–1937. PMLR, 2016. 6

[56] Haiyang Wang, Wenguan Wang, Xizhou Zhu, Jifeng Dai, and Liwei Wang. Collaborative visual navigation. *arXiv preprint arXiv:2107.01151*, 2021. 3

[57] Ping Xuan, Victor Lesser, and Shlomo Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proceedings of the fifth international conference on Autonomous agents*, pages 616–623, 2001. 1

[58] Xiaopeng Yu, Jiechuan Jiang, Wanpeng Zhang, Haobin Jiang, and Zongqing Lu. Model-based opponent modeling. *Advances in Neural Information Processing Systems*, 35:28208–28221, 2022. 3

[59] Weinan Zhang, Xihuai Wang, Jian Shen, and Ming Zhou. Model-based multi-agent policy optimization with adaptive opponent-wise rollouts. *arXiv preprint arXiv:2105.03363*, 2021. 3

[60] Yan Zheng, Zhaopeng Meng, Jianye Hao, Zongzhang Zhang, Tianpei Yang, and Changjie Fan. A deep bayesian policy reuse approach against non-stationary agents. *Advances in neural information processing systems*, 31, 2018. 3