Promptable Behaviors: Personalizing Multi-Objective Rewards from Human Preferences

Minyoung Hwang[1], **Luca Weihs**[1], **Chanwoo Park**[2], **Kimin Lee**[3],
Aniruddha Kembhavi[1], **Kiana Ehsani**[1]
[1]PRIOR @ Allen Institute for AI, [2]Massachusetts Institute of Technology,
[3]Korea Advanced Institute of Science and Technology

Abstract

Customizing robotic behaviors to be aligned with diverse human preferences is an underexplored challenge in the field of embodied AI. In this paper, we present Promptable Behaviors, a novel framework that facilitates efficient personalization of robotic agents to diverse human preferences in complex environments. We use multi-objective reinforcement learning to train a single policy adaptable to a broad spectrum of preferences. We introduce three distinct methods to infer human preferences by leveraging different types of interactions: (1) human demonstrations, (2) preference feedback on trajectory comparisons, and (3) language instructions. We evaluate the proposed method in personalized object-goal navigation and flee navigation tasks in ProcTHOR [18] and RoboTHOR [17], demonstrating the ability to prompt agent behaviors to satisfy human preferences in various scenarios.
Project page: https://promptable-behaviors.github.io

1. Introduction

Imagine a robot navigating in a house at midnight, asked to find an object without disturbing a child who just fell asleep. The robot is required to explore the house thoroughly in order to find the target object, but not collide with any objects to avoid making unnecessary noise. In contrast to this *Quiet Operation* scenario, in the *Urgent* scenario, a user is in a hurry and expects a robot to find the target object quickly rather than avoiding collisions. These contrasting scenarios highlight the need for customizing robot policies to adapt to diverse and specific human preferences.

Although learning-based approaches [57, 68] have significantly advanced the capability of robots to solve numerous tasks successfully, using these methods to customize robots for diverse human preferences remains a challenge [28]. Common practices in embodied AI [17, 18] use reinforcement learning with a reward function designed for specific agent behaviors. However, hand-crafting a reward function by human experts is time-consuming and difficult for agents with complex dynamics and large state and action spaces. To simplify the reward design process for non-expert users, recent methods [27, 30, 41] intuitively acquire reward models from human feedback. Yet, they have shortcomings in dealing with diverse preferences, since the agent has to be re-trained for each unique human preference.

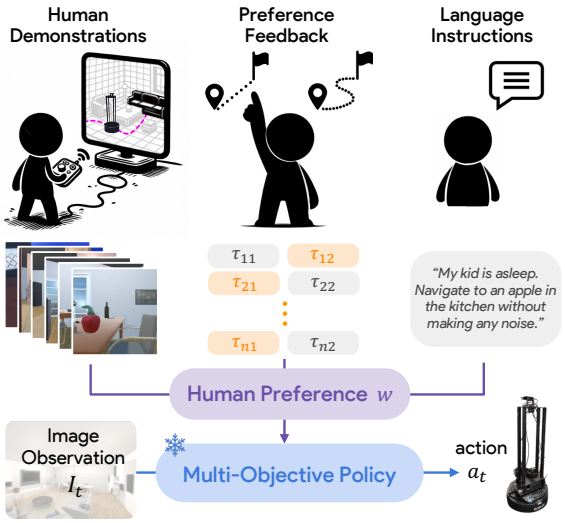

Figure 1. **Overview.** *Promptable Behaviors* captures human preferences across multiple objectives. We first train a multi-objective policy conditioned on the reward weight vector. After training, we freeze the policy and humans can provide their preferences with a wide range of options: (1) human demonstrations, (2) preference feedback on trajectory comparisons, and (3) language instructions.

We propose *Promptable Behaviors*, a novel personalization framework that deals with diverse human preferences without re-training the agent. The key idea of our method is to use *multi-objective reinforcement learning* (MORL) as the backbone of personalizing a reward model. We take a modular approach: (1) training a policy conditioned on a reward weight vector across multiple objectives and (2) inferring the reward weight vector that aligns with the user's preference. Using MORL, agent behaviors become promptable through adjustments in the reward weight vector during inference, without any policy fine-tuning. This significantly simplifies customizing robot behaviors to inferring a low-dimensional reward weight vector.

We provide a variety of options for users to provide their preferences to the agent. Specifically, we introduce three distinct methods of reward weight prediction, see Figure 1, leveraging different types of interaction: (1) human demonstrations, (2) preference feedback on trajectory comparisons, and (3) language instructions. Given human demonstrations in simulated environments, the agent can extract preferences from user-specific behaviors. Preferences could also be inferred from binary feedback on trajectory comparisons, enabling users to evaluate and contrast different agent behaviors rather than providing direct demonstrations. Finally, we utilize large-language models (LLMs) to translate language instructions into reward weight vectors. Using LLMs, even indirect or implicit instructions can be effectively interpreted based on extensive world knowledge.

We demonstrate *Promptable Behaviors* in two personalized navigation tasks, object-goal navigation and flee navigation, in two environments, ProcTHOR [18] and RoboTHOR [17]. Experimental results show that the agent behavior can be effectively prompted in both tasks. While the three reward weight prediction methods have their own advantages, preference feedback on trajectory comparisons shows the highest performance. In particular, our human evaluations demonstrate the effectiveness of our method.

In summary, our main contributions include:

- A novel framework for personalized learning that enables robots to align with diverse human preferences in complex embodied AI tasks without any policy fine-tuning.
- Three methods for inferring human preferences using human demonstrations, preference feedback on trajectory comparisons, and language instructions, each offering unique advantages.
- Demonstrations in two long-horizon personalized navigation tasks shows the effectiveness of our approach in prompting agent behaviors to satisfy human preferences.

2. Related Work

2.1. Multi-Objective Reinforcement Learning

Existing MORL algorithms are categorized into two main types [24]: single-policy and multi-policy. *Multi-policy* methods [6, 13, 38, 40, 51, 52, 60, 69, 70] train multiple policies, each corresponding with a single combination of objectives. However, in complex environments, training separate policies for each objective combination can be inefficient and resource-intensive. On the other hand, *single-policy* methods [44, 46, 53, 56, 61] transform the multi-objective problem into a single-objective problem through reward scalarization. [56] present multi-objective forms of existing RL algorithms (e.g., PPO [55] and A2C [39]) and focuses on learning a single policy conditioned on the combination of multiple objectives. While previous work show success in simple environments [5], tasks and objectives are often unrealistic. Recent work [14, 19] apply MORL on collision-aware navigation tasks but use multi-policy methods, while ours uses single-policy MORL. We demonstrate single-policy MORL in complex, realistic robotic tasks, utilizing high-dimensional observations. Ask4Help [58] trains a policy conditioned on user preference, where the policy determines when to request expert intervention. However, Ask4Help only considers user preference in a single dimension and discrete weight space, while ours deal with at least three objectives and continuous weight space.

2.2. Learning from Demonstrations

Given expert demonstrations, imitation learning (IL) and inverse-reinforcement learning (IRL) are the two prominent methods that guide agents to perform tasks by replicating and understanding observed behaviors, respectively. IL [29, 48, 49, 71] typically requires a large amount of high-quality expert data [49]. IRL aims to understand the underlying reward functions motivating expert behaviors rather than just copying the observed behaviors [1, 7, 22], but also requires a sufficient amount of demonstrations and the learned reward can be overfit to the collected data. Compared to IL and IRL, our method requires significantly fewer demonstrations to make the agent behavior satisfy the user's preference because the agent can efficiently generalize to diverse preferences without any policy fine-tuning.

2.3. Learning from Human Feedback

Learning from human feedback such as ratings, rankings, or expert interventions has been studied in numerous prior works [2–4, 15, 16, 21, 23, 50, 59, 65, 66]. Extending [15] which uses preference feedback on pairwise trajectory comparisons, recent developments [8, 9, 25–27, 30, 35–37, 41, 45, 54] have enhanced sample and feedback efficiency. While these methods have shown promise in natural language processing [43] and simplified settings in robotics with low-dimensional state and action spaces, their scalability to more complex, long-horizon, robotic tasks remains underexplored. We show that training a multi-objective policy with MORL and subsequently optimizing the reward weight vector using human feedback greatly enhances *the efficiency of handling diverse preferences*, especially in complex and long-horizon tasks. Our framework opens a new paradigm of using *user-friendly and intuitive human feedback* to predict the reward weights that satisfy users instead of asking users to explicitly choose the weights. Each of our weight prediction methods shows an advantage compared to traditional learning approaches that require intensive expert-level supervision or labeling efforts.

3. Method

We propose a novel framework, *Promptable Behaviors*, for personalized robot learning. *Promptable Behaviors* is an

adaptable policy that can update its behavior to various objectives and user preferences. Our method is divided into two primary components: (1) training a promptable multi-objective policy, and (2) capturing the agent's desired behavior through interactions. The overview of the proposed method is illustrated in Figure 1. Our multi-objective policy is adapted to individual users by adjusting the reward weights without any policy fine-tuning. For instance, suppose we wish to find reward weight vectors that align with the *Quiet Operation* and *Urgent* scenarios introduced in Section 1. For the *Quiet Operation* scenario, it is desirable to give high weight to safety like $[0, 0.3, 0.7]$ where the dimensions correspond to time efficiency, house exploration, and safety, respectively. On the other hand, reward weights such as $[1, 0, 0]$ would be aligned with the *Urgent* scenario.

3.1. Problem Formulation

We solve two navigation tasks using a robotic agent: object-goal navigation and flee navigation. We follow the task definition of object-goal navigation in previous work [17, 18], where the agent has to find an object of a given object category and execute an explicit `Done` action when the object is within 1m and visible in the agent's camera. The agent is allowed a maximum time horizon of $T = 500$. Flee navigation requires the agent to maximize its distance from the initial location. This task is useful when the robot has to autonomously relocate to a distant location in the house through spatial reasoning. In both tasks, the agent uses an RGB image observation o_t and outputs an action a_t at time t. The action is chosen from `[MoveAhead, RotateRight, RotateLeft, Done, LookUp, LookDown]`. The agent state s_t at time t is set as $o_{1:t}$. Most of the previous work aim to improve general performance such as success rate and success weighted by path length (SPL) [17, 18, 32]. In this paper, we open a new paradigm to consider the agent's behavior beyond rapid task completion. Such agent behavior can be measured and categorized as the sub-rewards on K objectives, where each objective reflects a fundamental aspect of navigation. Detailed definitions of the objectives and the evaluation metrics in each task are provided in Section 4.1.

3.2. Promptable Multi-Objective Policy Training

Our approach aims to develop a promptable and efficient framework for embodied AI tasks. Contrary to traditional RL methods that require a significant amount of time and resources to optimize for a single combination of different objectives, our method focuses on training a policy that can handle any linear combination of different objectives at test time. This reduces the dependency of the agent's behavior on the reward design choice of the practitioner who trains the policy. A naïve approach is to apply multi-policy MORL, train multiple policies with different reward configurations, and choose the most appropriate policy output for inference. This divides the multi-objective problem into a series of single-objective problems and ensures the agent optimizes the policies on each combination. However, this training process is inefficient and cannot cover the entire set of combinations because the size of the weight space increases exponentially with the number of objectives.

Inspired by Ask4Help [58] that trains a policy conditioned on the reward configuration, we condition our agent's policy on randomly sampled reward configuration during training and allow the agent to adapt to the user at inference time without additional training. While Ask4Help focuses on changing the reward configuration in a single dimension, we implement single-policy MORL [56] to handle multiple objectives. Figure 2 illustrates the network architecture of the policy in our method. We train a single policy with a scalarized reward function $r^{\mathbf{w}} = \mathbf{w}^\intercal \mathbf{r}$, which combines multiple objectives with a reward weight vector $\mathbf{w}$ randomly sampled from a K-dim simplex $\Delta_K = \{\mathbf{w} \in \mathbb{R}_+^K | \, ||\mathbf{w}||_1 = 1\}$. While most RL frameworks in embodied AI have a pre-defined and fixed $\mathbf{w}$, our policy is conditioned on the combination itself and explores various combinations during training. This makes the trained policy adaptable to various human preferences through the adjustment of the reward weight vector without any additional policy training.

Visual Encoder Using CLIP. Recent work [32] has shown the strength of visual backbones of CLIP [47] in embodied AI tasks. As described in Figure 2, we use a pre-trained CLIP ResNet-50 to encode $3 \times 224 \times 224$ RGB image into a $2048 \times 7 \times 7$ tensor. Since the pretrained model has shown its effectiveness in various visual navigation tasks, we freeze the weights of the encoder while training the policy. The CLIP embedding is merged with a 32-dim goal embedding resulting in a shape of $64 \times 7 \times 7$. The concatenated tensor is passed into a CNN and is flattened into a 1568-dim feature.

Reward Weight Encoder. Since the goal of the proposed method is to handle various combinations of objectives with a single policy, we randomly sample reward weights during training. At each episode, a reward weight vector $\mathbf{w}$ is uniformly sampled from a K-dim simplex Δ_K and the agent calculates the scalarized rewards based on $\mathbf{w}$ throughout the whole episode. Bringing insight from the recent success of using codebook as an effective representation module [20], we use a feed-forward neural network (FFNN) to expand the dimension of $\mathbf{w}$ to 30 and then pass it through a codebook with 30 learnable $K \cdot 12$-dim latent codes. This makes the policy handle unseen reward combinations using the learned codes. We also compare this method with an encoding approach extended from [58], where an integer weight vector is encoded using a lookup table.

Navigation Policy. We implement a multi-objective version of DD-PPO [62, 64] to maximize the expected reward.

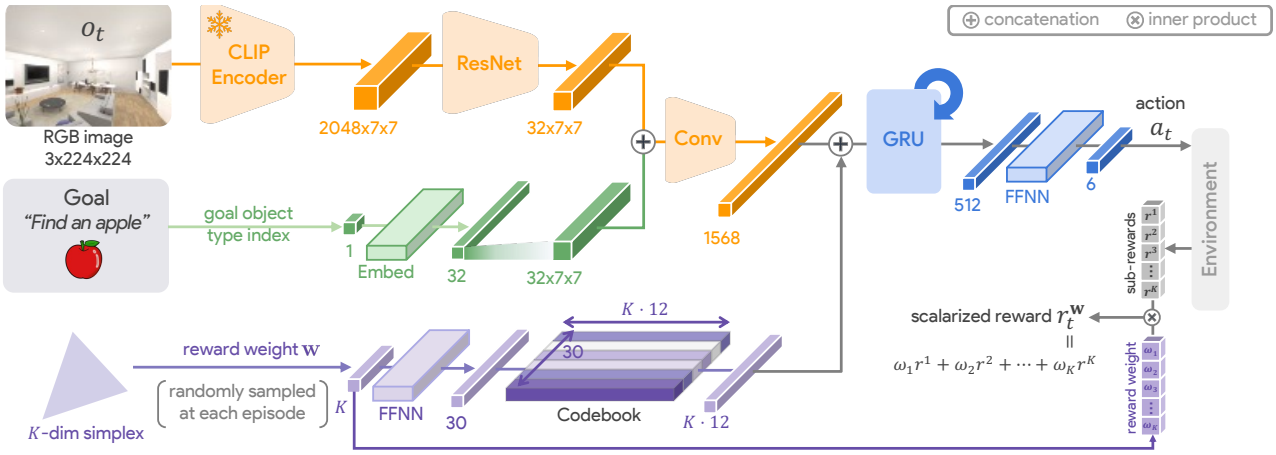

Figure 2. **Network Architecture.** The figure illustrates a single-policy MORL architecture for object-goal navigation. A CLIP encoder alongside a ResNet processes the visual observation. This image embedding, concatenated with the goal embedding, is then fed into a convolutional layer and concatenated with the reward weight vector encoded via codebook, forming the input for the recurrent policy. The policy is trained to be multi-objective, modulated by the reward weight vector $\mathbf{w}$. During training the policy, the agent receives the reward as the weighted sum of sub-rewards from K objectives, determined by the reward weight vector $\mathbf{w}$.

A fundamental difference compared to the traditional DD-PPO is that the policy π is conditioned on the reward weight vector $\mathbf{w}$, which is randomly sampled at each episode. The agent calculates the rewards for multiple objectives at each timestep, and updates the policy based on the scalarized rewards. The RL loss tries to maximize the expected return averaged among different $\mathbf{w}$ and episodes.

3.3. Reward Weight Prediction via Interaction

Effectively aligning agent behavior with human preferences is a key challenge in our work. The proposed framework focuses on predicting the optimal reward weight vector representing human preferences, based on different forms of interactions. As illustrated in Figure 1, we explore three distinct interactions: (1) human demonstrations, (2) preference feedback on trajectory comparisons, and (3) language instructions. Each method offers a unique perspective and mechanism for capturing human preferences, thereby handling a diverse range of scenarios and user interactions. Following the general context in MORL [24], we assume that human preferences remain constant over time and each human preference is captured through a linear combination of multiple objectives in the environment. Under these assumptions, a human user's true preference is represented as a reward weight vector $\mathbf{w} \in \Delta_K$.

3.3.1 Human Demonstrations

Getting human demonstrations is a direct and intuitive way for humans to express their preferences. Given a demonstration $\tau_h = (s_1, a_1, ..., s_{T_h}, a_{T_h})$ $(T_h \leq T)$ from a human user, we can infer the user's inherent values and priorities. We identify the reward weight vector that most accurately reflects these preferences by maximizing the expected log-likelihood between the demonstrated action and the action distribution from the policy π conditioned on the reward weight $\mathbf{w}$. The weight prediction loss is defined as follows:

$$\mathcal{L}_{demo}(\mathbf{w}; \tau_h) = -\sum_{t=1}^{T_h} \log \pi(a_t \mid s_t ; \mathbf{w}). \quad (1)$$

For the optimization process, we use multiple initialization weights, including a uniform vector $[1/K, ..., 1/K]$. We then apply gradient descent from each of the initialization weights until the loss converges. By averaging the results from diverse initial conditions, we enhance the robustness and reliability of our weight prediction. Also, our method is faster than traditional IRL approaches since we only optimize K parameters, with K being at most 5 in our setting.

3.3.2 Preference Feedback on Trajectory Comparisons

We further develop weight prediction methods that take human feedback, in the form of comparisons between agent trajectories, as input. We also propose a novel trajectory comparison method called group trajectory comparison, which is more feedback-efficient than conventional pairwise comparison [15].

Pairwise Trajectory Comparison. In pairwise trajectory comparison, the human user is asked to select a trajectory that better aligns with their preference from a pair of trajectories. Given N trajectory pairs, the user will provide N binary preference labels. We denote the preference data as a set of trajectory pairs and preference labels $\mathcal{S} = \{(\tau_{i1}, \tau_{i2}, y_i) | 1 \leq i \leq N\}$, where τ_{i1} and τ_{i2} are the two trajectories that the user observes at the i^{th} trajectory comparison, $y_i = 1$ indicates τ_{i1} is preferred to τ_{i2}, and $y_i = 0$ indicates otherwise. Notably, we do not consider the ties and provide the human user the ability to skip indistinguishable queries. We use a common assumption in

preference-based learning [67] that given a reward weight vector $\mathbf{w}$ that reflects human preference, the user chooses trajectory τ_1 to be preferred over τ_2 with a preference probability based on the Bradley-Terry model [10] as follows:

$$P(\tau_1 \succ \tau_2; \mathbf{w}) = \frac{\exp(\mathbf{w}^\intercal \mathbf{r}(\tau_1))}{\exp(\mathbf{w}^\intercal \mathbf{r}(\tau_1)) + \exp(\mathbf{w}^\intercal \mathbf{r}(\tau_2))},$$

where $\tau_1 \succ \tau_2$ denotes τ_1 is preferred to τ_2 and $\mathbf{r}(\tau) = \sum_{t=0}^{T} \mathbf{r}(s_t, a_t)$. The model handles the inherent stochasticity and inconsistency in human preferences, rather than assuming human preferences are deterministic and perfectly rational. We solve an optimization problem that maximizes the expected log-likelihood of preferences in pairwise trajectory comparisons as follows:

$$\max_{\mathbf{w} \in \Delta_K} \mathop{\mathbb{E}}_{(\tau_1, \tau_2, y) \in \mathcal{S}} \log \left(y P(\tau_1 \succ \tau_2; \mathbf{w}) + (1-y) P(\tau_1 \prec \tau_2; \mathbf{w}) \right).$$

Group Trajectory Comparison. We also introduce a novel trajectory comparison method called group trajectory comparison, where the human user observes groups of trajectories instead of individual pairs. For instance, consider comparing a group emphasizing safety against another set, prioritizing path efficiency. The difference between the two groups becomes clearer, making the process of giving preference labels more straightforward for users. At each iteration, we sample two groups of M trajectories, each trajectory group generated with the same reward weights, while different groups have different reward weights. We ask the user to compare the groups of trajectories and provide a preference label. Each group comparison yields an inequality constraint, filtering out a volume of the reward weight space less likely to match the user's preferences. We repeat this process for N iterations with different groups of trajectories and different reward weights. By adding N constraints and performing constrained optimization, we effectively narrow down the search area for the most probable reward weight vector. In Section 4.3, we show that group comparison is 17.8% more effective than conventional pairwise trajectory comparison in human evaluation. It is also less ambiguous for the human user to compare groups that have distinct differences. Detailed theoretical analyses are included in the supplementary material.

3.3.3 Language Instructions

We leverage the power of LLMs to interpret language instructions and quantify human preferences as numerical reward weights. LLMs can adapt to the nuances of user instructions, and the models can translate natural language instructions into reward weights. We ask ChatGPT [42] to output the optimal reward weight vector from a language instruction of the user given the task description and definitions of the objectives. We use in-context learning (ICL) [11] by providing six examples of instruction and answer pairs, collected from six human experts. We also apply chain-of-thought (CoT) reasoning [33] to handle the complex, multi-step process of deciding reward weights on multiple, often conflicting, objectives. This method is highly beneficial in scenarios requiring rapid adaptation to user preferences without domain knowledge because LLMs can infer the importance to place on each objective based on the context in the instruction utilizing its world knowledge.

4. Experiments

We evaluate our method on personalized object-goal navigation (ObjectNav) and flee navigation (FleeNav) in ProcTHOR [18] and RoboTHOR [17], environments in the AI2-THOR [34] simulator. For ObjectNav, there are 16 and 12 target object categories in ProcTHOR and RoboTHOR, respectively. The policy is evaluated across various scenarios to ensure that it aligns with human preferences and achieves satisfactory performance in both tasks. We show that the proposed method effectively prompts agent behaviors by adjusting the reward weight vector and infers reward weights from human preferences using three distinct reward weight prediction methods.

4.1. Experiment Settings

Training details. We train our models using the Allen-Act [62] reinforcement learning framework. In ProcTHOR ObjectNav, we train our policy for $130M$ steps over $10k$ houses and validate with 100 episodes in 67 unseen houses. In ProcTHOR FleeNav, we train for $50M$ steps over $10k$ houses and validate in 100 episodes in 71 unseen houses. In RoboTHOR ObjectNav and FleeNav, we train for $100M$ steps in 60 houses and validate with 100 episodes in 15 unseen houses. All methods are trained with 8 NVIDIA RTX A6000 GPUs and 80 samplers. We use Adam optimizer with a learning rate of 0.0003 for all training.

Objectives. In personalized object-goal navigation, we define five objectives: time efficiency, path efficiency, house exploration, object exploration, and safety. *Time efficiency* is designed to encourage the agent to complete the episode quickly, while *path efficiency* aims to find the target object via the shortest path. House exploration and object exploration encourage the agent to explore more at the expense of efficiency. *House exploration* is designed to favor covering a larger area, while *object exploration* aims to observe more objects within the agent's camera. *Safety* encourages the agent to avoid colliding with obstacles and visiting areas where you could get trapped or stuck. We provide the exact equations for calculating sub-rewards for each objective in the supplementary material.

In personalized flee navigation, there are three objectives: time efficiency, house exploration, and safety. We follow the definitions of the objectives in object-goal navigation. We do not consider preferences over success, since

achieving the goal is a default expectation in both tasks.

Baselines. We use EmbCLIP [32] as the single-objective RL (SORL) baseline. Also, we implement a multi-policy MORL baseline, prioritized EmbCLIP, that trains K policies separately, where each policy is trained with a fixed and spiked reward weight vector prioritizing one specific objective ν times more than the other objectives. We set ν as 4 and 3 for ObjectNav and FleeNav, respectively.

Evaluation Metrics. In ObjectNav, we evaluate the general performance of the agent using success rate, success weighted by path length (SPL), distance to goal, and episode length. An episode is recorded as a success if the agent executes a `Done` action within $1m$ of the target object and the object is visible in the agent's last observation. Success rate is measured as the number of succeeded episodes divided by the total number of episodes, $|E|$. SPL is calculated as $\frac{1}{|E|}\sum_{i=1}^{|E|} S_i \frac{\ell_i^{min}}{max(\ell_i^{min},\ell_i)}$, where S_i, ℓ_i, and ℓ_i^{min} denote the binary success value, path length, and the shortest path length at the i^{th} episode. In FleeNav, we evaluate the agent using success rate, path length weighted by path length (PLOPL), distance to the farthest point, and episode length. Success at each episode is determined as ℓ/ℓ^{max}, where ℓ and ℓ^{max} denote the Euclidean distance from the initial point to the last point and the distance from the initial point to its farthest point, respectively. PLOPL is measured as the path length divided by the maximum path length at each episode. SPL and PLOPL consider the length of the trajectory, not only the initial and last positions of the agent.

To evaluate three reward weight prediction methods, we collect demonstrations and feedback from real human users for five different scenarios, each prioritizing one or two objectives in object-goal navigation. We perform human evaluations by calculating the win rate [31], showing a pair of trajectories to the user and asking which trajectory is more preferred in each objective. The win rate is calculated as $\frac{1}{K(K-1)}\sum_{i=1}^{K}\sum_{j\neq i} H(\zeta_i,\tau_i,\tau_j)$, where τ_i is generated for the i^{th} scenario ζ_i and $H(\zeta_i,\tau_i,\tau_j)$ is 1 if $\tau_i \succ \tau_j$ in ζ_i. For instance, suppose we have generated τ_1 and τ_2 for the *Quiet Operation* and *Urgent* scenarios in Section 1, respectively. Presenting the *Quiet Operation* scenario with the two trajectories to the user, we get a score of 1 when the user prefers τ_1 in the given situation.

Furthermore, we measure the weight prediction performance as the cosine similarity between the estimated reward weight and the reward weight determined by human experts. We also measure the Generalized Gini Index (GGI) [12, 63] to statistically measure the disparity across multiple objectives in the predicted weights. A higher GGI indicates the weight vector is concentrated or peaked towards a few specific objectives. More scenarios and experiment details are provided in the supplementary material.

In our experiments, prioritizing different objectives demonstrates diverse human preferences in everyday scenarios. By evaluating how the agent's behavior changes with varied objective prioritization, we gain insights into the adaptability of *Promptable Behaviors* to diverse preferences. We first show that the multi-objective policy outputs different agent behaviors based on objective prioritization. Then, we demonstrate how the agent can induce its behavior to satisfy user preference, given human demonstrations, preference feedback, and language instructions.

4.2. Promptable Behaviors in Embodied AI

The first question we would like to answer is "How well does *Promptable Behaviors* adapt its policy to reflect the changes in the input reward weights?". The goal is to show that our method can adapt its behavior to the reward weights during inference time more effectively than the baseline. To test this hypothesis, we first measure our model's behavior given a reward weight vector prioritized on a single objective. For instance, the policy should achieve higher exploration when the exploration reward's weight is peaked than when no objective is prioritized or when the reward weight is peaked for safety. Thus, we evaluate *Promptable Behaviors* and Prioritized EmbCLIP across $K+1$ reward weights, including one uniform weight and K peaked weights. For peaked reward weight vectors, we set the weight for the prioritized objective ν times greater than the weights for other objectives. Among all methods, our approach most effectively reflects the prioritization of objectives in agent behaviors. Results in Table 1 and Table 2 show that the proposed method outperforms the baseline in ProcTHOR ObjectNav and FleeNav while showing efficiency in training, requiring much less computational resources compared to Prioritized EmbCLIP. Note that the performance of EmbCLIP does not perfectly match the performance reported in [20] since we re-implement the method with our network architecture and evaluate it in a smaller validation set. Results in the RoboTHOR environment and detailed analyses of all experiments are provided in the supplementary material.

Our method achieves high success rates while efficiently optimizing the agent behavior for each objective. In Table 1, Prioritized EmbCLIP and *Promptable Behaviors* show different performance based on the prioritization of objectives. When prioritizing house exploration, both MORL methods achieve higher house exploration reward than the SORL baseline, EmbCLIP. When house exploration is prioritized, the proposed method shows the highest success rate (row j in Table 1), 11.3% higher than EmbCLIP, while Prioritized EmbCLIP shows the lowest success rate (row d in Table 1) among all methods and reward configurations. Additionally, our method achieves the highest SPL and the path efficiency reward when path efficiency is prioritized (row i in Table 1), outperforming EmbCLIP by 19.3% and 56.1%, respectively. This implies that the

Method	Multi-Objective		Prioritized Objective	Success ↑	SPL ↑	Distance to Goal ↓	Episode Length ↓	Sub Rewards ↑				
								Time Efficiency	Path Efficiency	House Exploration	Object Exploration	Safety
EmbCLIP [32]	✗	a	-	0.611	0.455	1.677	105.389	0.767	0.581	0.703	0.731	0.556
Prioritized EmbCLIP	Multi-Policy	b	Time Efficiency	0.560	0.445	2.803	52.060	**0.926**	0.317	0.136	0.247	0.746
		c	Path Efficiency	0.611	0.449	2.038	106.444	0.764	0.515	0.590	0.731	0.693
		d	House Exploration	0.200	0.113	3.921	350.960	0.033	**0.677**	**2.868**	0.161	0.012
		e	Object Exploration	0.611	0.513	2.439	138.389	0.668	0.414	0.703	**0.731**	0.556
		f	Safety	0.480	0.391	3.237	56.620	0.912	0.016	0.130	0.004	**0.834**
Promptable Behaviors (Ours)	Single-Policy	g	-	0.600	0.496	2.526	86.070	0.824	0.589	0.336	0.412	0.770
		h	Time Efficiency	0.560	0.492	2.675	51.760	**0.927**	0.375	0.078	0.301	0.772
		i	Path Efficiency	0.650	**0.543**	2.213	115.350	0.737	**0.907**	0.451	0.674	0.665
		j	House Exploration	**0.680**	0.506	2.253	159.440	0.605	0.902	**0.995**	0.705	0.563
		k	Object Exploration	0.650	0.525	2.198	94.890	0.798	0.829	0.358	**0.725**	0.754
		l	Safety	0.500	0.446	2.875	51.890	0.927	0.211	0.083	0.096	**0.829**

Table 1. **Performance in ProcTHOR ObjectNav.** We evaluate each method in the validation set with six different configurations of objective prioritization: uniform reward weight across all objectives and prioritizing a single objective 4 times as much as other objectives. Sub-rewards for each objective are accumulated during each episode, averaged across episodes, and then normalized using the mean and variance calculated from values in rows g-l. Colored cells indicate the highest values in each sub-reward column.

(a) Prioritizing Safety in ObjectNav

(b) Trajectory Visualizations with Different Prioritizations on Objectives

Figure 3. **Prompting Agent Behaviors by Adjusting Reward Weights.** (a) As we prioritize safety more, the average safety reward increases while the average reward of a conflicting objective, house exploration, decreases. We normalize the rewards for each objective using the mean and variance calculated across all weights. (b) In each figure, agent trajectory is visualized when an objective is prioritized 10 times as much as other objectives. The agent's final location is illustrated as a star.

proposed method effectively maintains general performance while satisfying the underlying preferences in various prioritizations. In contrast, Prioritized EmbCLIP fails to improve path efficiency reward when the corresponding objective is prioritized. This could be due to the design choice of ν, which determines the sensitivity of the prioritized objective. Trying various ν might improve the alignment, but it is challenging to train the policy multiple times with different ν. Selecting a proper ν is much easier in *Promptable Behaviors* because our policy has already observed random reward weight vectors during training. For FleeNav, Table 2 shows that both MORL methods successfully prompt agent behaviors through reward weight adjustments. Our method achieves success rates higher than 0.7 in all cases, while Prioritized EmbCLIP shows a success rate of 0.691 when time efficiency is prioritized (row a in Table 2).

To check how conflicting objectives affect each other, we assess an experiment to evaluate the trained policy by adjusting the weight of the most prioritized objective from 0.2 to 0.9. Figure 3 (a) shows an example when we observe trade-offs between two conflicting objectives in ObjectNav: safety and house exploration. As we increase the weight for safety, the safety reward increases while the reward for its conflicting objective, house exploration, decreases. Figure 3 (b) visualizes five trajectories that prioritize four different objectives in the same episode. The difference between trajectories implies that prioritizing time efficiency or path efficiency encourages the agent to move through a shorter path while prioritizing house exploration or object exploration encourages the agent to explore the house more thoroughly. The safety reward column in Table 1 shows that the agent receives a higher safety reward when safety is prioritized, which means that the agent learns to avoid visiting narrow places or moving closely to near objects and walls.

4.3. Reward Weight Prediction

In the previous section, we have shown that our policy can effectively adjust its behavior to reflect the reward weights during inference. In this section, we compare three reward weight prediction methods and show the results of *Promptable Behaviors* for the full pipeline. As mentioned in Section 3.3, the users have three distinct options to describe their preferences to the agent: (1) demonstrating a trajectory, (2) labeling their preferences on trajectory comparisons, and (3) providing language instructions. Table 3 shows the quantitative performance of the three weight prediction methods, each with its own advantage.

Weight Prediction Performance. Weight optimization from human demonstrations shows 70.7% cosine similarity between the predicted weights and the weights designed by human experts only using a single human demonstration. Preference feedback on group trajectory comparisons shows the highest prediction performance, 93.5%, when each group contains two trajectories. Utilizing ChatGPT

Method	Multi-Objective		Prioritized Objective	Success ↑	PLOPL ↑	Distance to Furthest ↓	Episode Length ↓	Sub Rewards ↑		
								Time Efficiency	House Exploration	Safety
Prioritized EmbCLIP	Multi-Policy	a	Time Efficiency	0.691	0.810	7.360	57.090	**0.875**	0.420	0.138
		b	House Exploration	**0.759**	0.872	6.704	58.330	0.839	**0.835**	0.215
		c	Safety	0.723	0.856	7.391	57.640	0.859	0.676	**0.487**
Promptable Behaviors (Ours)	Single-Policy	d	-	0.700	0.805	7.013	69.020	0.531	0.365	0.522
		e	Time Efficiency	0.728	0.832	6.592	66.490	**0.604**	0.434	0.563
		f	House Exploration	0.737	0.861	6.317	71.500	0.460	**0.813**	0.089
		g	Safety	0.711	0.814	6.735	67.830	0.566	0.227	**0.776**

Table 2. **Performance in ProcTHOR FleeNav.** We evaluate each method in the validation set with five different configurations of objective prioritization: uniform reward weight across all objectives and prioritizing a single objective 3 times as much as other objectives. The displayed sub-reward values are normalized for each objective following Table 1.

Weight Prediction Methods			Sim ↑	GGI
Input	Model	N		
Human Demonstrations	-	1	0.707	0.347
Preference Feedback	Pairwise Comparison (M=1)	20	0.356	0.800
		50	0.358	0.800
		500	0.897	0.800
	Group Comparison (M=2)	5	0.689	0.626
		10	0.793	0.618
		25	**0.935**	0.657
	Group Comparison (M=5)	2	0.722	0.634
		4	0.682	0.762
		10	0.862	0.641
Language Instructions	ChatGPT	1	0.530	0.388
	w/ ICL	1	0.529	0.379
	w/ CoT	1	0.614	0.391
	w/ ICL + CoT	1	0.482	0.347

Table 3. **Comparison of Three Weight Prediction Methods in ProcTHOR ObjectNav.** We predict the optimal reward weights from human demonstrations, preference feedback on trajectory comparisons, and language instructions. We measure the cosine similarity (Sim) between the predicted weights and the weights designed by human experts. We also calculate generalized gini index (GGI) which measures the peakedness of the predicted weights.

with four different settings based on the use of ICL and CoT, using ChatGPT with CoT resulted in the best performance.

Peakedness of Weights. In Table 3, preference feedback on pairwise comparison shows the most peaked predicted weights while using human demonstrations outputs the least peaked weights. This could be due to the ambiguity lying in human demonstrations, where multiple similar reward weight vectors can produce the same trajectory. Although there appears to be no direct correlation between peakedness and weight prediction performance, this analysis provides valuable insights into the distinct characteristics of the different weight prediction methods.

Human Evaluation. We also perform human evaluations by asking five participants to compare trajectories generated with the predicted reward weights for different scenarios. Results in Table 4 show that group trajectory comparison, especially with two trajectories per group, achieves the highest win rate, significantly outperforming other methods by up to 17.8%. This high win rate indicates that the generated trajectories closely align with the intended scenarios. Among the weight prediction methods using preference feedback, group comparison with a group size of two requires only half the binary feedback compared to pairwise comparison, yet it improves the win rate significantly. More efficiently, group comparison with five trajectories per group needs just 10 user feedback while still managing a 6.5% higher win rate than pairwise comparison. Interestingly, using language instructions to infer reward weights shows the second-best performance among all methods in Table 4, demonstrating the potential of LLMs in understanding and translating complex human preferences into reward weights using world knowledge. Details for the evaluation and statistical test are in the supplementary.

Weight Prediction Methods			Win Rate ↑
Input	Model	N	
Human Demo.	-	1	0.556
Preference Feedback	Pairwise Comparison (M=1)	50	0.552
	Group Comparison (M=2)	25	**0.650**
	Group Comparison (M=5)	10	0.588
Language Instruction	ChatGPT w/ CoT	1	0.600

Table 4. **Human Evaluation on Scenario-Trajectory Matching.** Participants evaluate trajectories generated with the trained policy and the reward weights predicted for five scenarios in ObjectNav.

Ablation Study. Ablation studies on codebook and group trajectory comparison are provided in the supplementary.

5. Conclusion

This paper proposes *Promptable Behaviors*, a novel framework that advances the personalization of robotic behaviors in complex environments, efficiently adapting to diverse human preferences with minimal user interaction. By leveraging MORL and three weight prediction methods, we have demonstrated the ability to prompt agent behaviors through reward weight adjustments in object-goal and flee navigation. For future work, we will demonstrate our method in various tasks such as manipulation. Additionally, since we assume static and linear preferences, we will extend our approach to consider dynamic and non-linear preferences.

References

[1] Stephen Adams, Tyler Cody, and Peter A Beling. A survey of inverse reinforcement learning. *Artificial Intelligence Review*, 55(6):4307–4346, 2022. 2

[2] Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I 11*, pages 12–27. Springer, 2011. 2

[3] Riad Akrour, Marc Schoenauer, and Michèle Sebag. April: Active preference learning-based reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23*, pages 116–131. Springer, 2012.

[4] Riad Akrour, Marc Schoenauer, Michèle Sebag, and Jean-Christophe Souplet. Programming by feedback. In *International Conference on Machine Learning*, number 32, pages 1503–1511. JMLR. org, 2014. 2

[5] Lucas N. Alegre, Florian Felten, El-Ghazali Talbi, Grégoire Danoy, Ann Nowé, Ana L. C. Bazzan, and Bruno C. da Silva. MO-Gym: A library of multi-objective reinforcement learning environments. In *Proceedings of the 34th Benelux Conference on Artificial Intelligence BNAIC/Benelearn 2022*, 2022. 2

[6] Lucas N Alegre, Ana LC Bazzan, Diederik M Roijers, Ann Nowé, and Bruno C da Silva. Sample-efficient multi-objective learning via generalized policy improvement prioritization. *arXiv preprint arXiv:2301.07784*, 2023. 2

[7] Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021. 2

[8] Erdem Biyik and Dorsa Sadigh. Batch active preference-based learning of reward functions. In *Conference on robot learning (CoRL)*. PMLR, 2018. 2

[9] Erdem Bıyık, Daniel A Lazar, Dorsa Sadigh, and Ramtin Pedarsani. The green choice: Learning and influencing human decisions on shared roads. In *2019 IEEE 58th conference on decision and control (CDC)*. IEEE, 2019. 2

[10] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. 5

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 5

[12] Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, and Shie Mannor. Multi-objective bandits: Optimizing the generalized gini index. In *International Conference on Machine Learning*, pages 625–634. PMLR, 2017. 6

[13] Xin-Qiang Cai, Pushi Zhang, Li Zhao, Jiang Bian, Masashi Sugiyama, and Ashley Juan Llorens. Distributional pareto-optimal multi-objective reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 2

[14] Guangran Cheng, Yuanda Wang, Lu Dong, Wenzhe Cai, and Changyin Sun. Multi-objective deep reinforcement learning for crowd-aware robot navigation with dynamic human preference. *Neural Computing and Applications*, pages 1–19, 2023. 2

[15] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems (NeurIPS)*, 2017. 2, 4

[16] Christian Daniel, Oliver Kroemer, Malte Viering, Jan Metz, and Jan Peters. Active reward learning with a novel acquisition function. *Autonomous Robots*, 39:389–405, 2015. 2

[17] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020. 1, 2, 3, 5

[18] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35, 2022. 1, 2, 3, 5

[19] Niranjan Deshpande, Dominique Vaufreydaz, and Anne Spalanzani. Navigation in urban environments amongst pedestrians using multi-objective deep reinforcement learning. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 923–928. IEEE, 2021. 2

[20] Ainaz Eftekhar, Kuo-Hao Zeng, Jiafei Duan, Ali Farhadi, Ani Kembhavi, and Ranjay Krishna. Selective visual representations improve convergence and generalization for embodied ai. *arXiv preprint arXiv:2311.04193*, 2023. 3, 6

[21] Layla El Asri, Bilal Piot, Matthieu Geist, Romain Laroche, and Olivier Pietquin. Score-based inverse reinforcement learning. 2016. 2

[22] Qiang Fang, Wenzhuo Zhang, and Xitong Wang. Visual navigation using inverse reinforcement learning and an extreme learning machine. *Electronics*, 10(16):1997, 2021. 2

[23] Johannes Fürnkranz, Eyke Hüllermeier, Weiwei Cheng, and Sang-Hyeun Park. Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine learning*, 89:123–156, 2012. 2

[24] Conor F Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M Zintgraf, Richard Dazeley, Fredrik Heintz, et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):26, 2022. 2, 4

[25] Joey Hejna and Dorsa Sadigh. Inverse preference learning: Preference-based rl without a reward function. *arXiv preprint arXiv:2305.15363*, 2023. 2

[26] Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W Bradley Knox, and Dorsa Sadigh. Contrastive prefence learning: Learning from human feedback without rl. *arXiv preprint arXiv:2310.13639*, 2023.

[27] Donald Joseph Hejna III and Dorsa Sadigh. Few-shot preference learning for human-in-the-loop rl. In *Conference on Robot Learning*, pages 2014–2025. PMLR, 2023. 1, 2

[28] Mehdi Hellou, Norina Gasteiger, Jong Yoon Lim, Minsu Jang, and Ho Seok Ahn. Personalization and localization in human-robot interaction: A review of technical methods. *Robotics*, 10(4):120, 2021. 1

[29] Minyoung Hwang, Jaeyeon Jeong, Minsoo Kim, Yoonseon Oh, and Songhwai Oh. Meta-explore: Exploratory hierarchical vision-and-language navigation using scene object spectrum grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[30] Minyoung Hwang, Gunmin Lee, Hogun Kee, Chan Woo Kim, Kyungjae Lee, and Songhwai Oh. Sequential preference ranking for efficient reinforcement learning from human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 1, 2

[31] Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023. 6

[32] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: Clip embeddings for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14829–14838, 2022. 3, 6, 7

[33] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022. 5

[34] 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. 5

[35] Kimin Lee, Laura Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 2

[36] Xinran Liang, Katherine Shu, Kimin Lee, and Pieter Abbeel. Reward uncertainty for exploration in preference-based reinforcement learning. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[37] Runze Liu, Fengshuo Bai, Yali Du, and Yaodong Yang. Meta-reward-net: Implicitly differentiable reward learning for preference-based reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[38] Haoye Lu, Daniel Herman, and Yaoliang Yu. Multi-objective reinforcement learning: Convexity, stationarity and pareto optimality. In *The Eleventh International Conference on Learning Representations*, 2022. 2

[39] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016. 2

[40] Hossam Mossalam, Yannis M Assael, Diederik M Roijers, and Shimon Whiteson. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707*, 2016. 2

[41] Vivek Myers, Erdem Biyik, Nima Anari, and Dorsa Sadigh. Learning multimodal rewards from rankings. In *Conference on Robot Learning*, pages 342–352. PMLR, 2022. 1, 2

[42] OpenAI. ChatGPT. https://openai.com/blog/chatgpt, 2022. 5

[43] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022. 2

[44] Anqi Pan, Wenjun Xu, Lei Wang, and Hongliang Ren. Additional planning with multiple objectives for reinforcement learning. *Knowledge-Based Systems*, 193:105392, 2020. 2

[45] Jongjin Park, Younggyo Seo, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. Surf: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 2

[46] Markus Peschl, Arkady Zgonnikov, Frans A Oliehoek, and Luciano C Siebert. Moral: Aligning ai with human norms through multi-objective reinforced active learning. *arXiv preprint arXiv:2201.00012*, 2021. 2

[47] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 3

[48] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5173–5183, 2022. 2

[49] Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. *arXiv preprint arXiv:2301.07302*, 2023. 2

[50] Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. *arXiv preprint arXiv:2307.01928*, 2023. 2

[51] Mathieu Reymond, Eugenio Bargiacchi, and Ann Nowé. Pareto conditioned networks. *arXiv preprint arXiv:2204.05036*, 2022. 2

[52] Diederik M Roijers. Multi-objective decision-theoretic planning. *AI Matters*, 2(4):11–12, 2016. 2

[53] Diederik M Roijers, Denis Steckelmacher, and Ann Nowé. Multi-objective reinforcement learning for the expected utility of the return. In *Proceedings of the Adaptive and Learning Agents workshop at FAIM*, 2018. 2

[54] Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. In *Proceedings of Robotics: Science and Systems (RSS)*, 2017. 2

[55] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[56] Umer Siddique, Paul Weng, and Matthieu Zimmer. Learning fair policies in multi-objective (deep) reinforcement learning with average and discounted rewards. In *International Conference on Machine Learning*, pages 8905–8915. PMLR, 2020. 2, 3

[57] Bharat Singh, Rajesh Kumar, and Vinay Pratap Singh. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*, pages 1–46, 2022. 1

[58] Kunal Pratap Singh, Luca Weihs, Alvaro Herrasti, Jonghyun Choi, Aniruddha Kembhavi, and Roozbeh Mottaghi. Ask4help: Learning to leverage an expert for embodied tasks. *Advances in Neural Information Processing Systems*, 35:16221–16232, 2022. 2, 3

[59] Hiroaki Sugiyama, Toyomi Meguro, and Yasuhiro Minami. Preference-learning based inverse reinforcement learning for dialog control. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012. 2

[60] Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *The Journal of Machine Learning Research*, 15(1): 3483–3512, 2014. 2

[61] Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. Scalarized multi-objective reinforcement learning: Novel design techniques. In *2013 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*, pages 191–199. IEEE, 2013. 2

[62] Luca Weihs, Jordi Salvador, Klemen Kotar, Unnat Jain, Kuo-Hao Zeng, Roozbeh Mottaghi, and Aniruddha Kembhavi. Allenact: A framework for embodied ai research. *arXiv preprint arXiv:2008.12760*, 2020. 3, 5

[63] John A Weymark. Generalized gini inequality indices. *Mathematical Social Sciences*, 1(4):409–430, 1981. 6

[64] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *arXiv preprint arXiv:1911.00357*, 2019. 3

[65] Aaron Wilson, Alan Fern, and Prasad Tadepalli. A bayesian approach for policy learning from trajectory preference queries. *Advances in neural information processing systems*, 25, 2012. 2

[66] Christian Wirth, Johannes Fürnkranz, and Gerhard Neumann. Model-free preference-based reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016. 2

[67] Christian Wirth, Riad Akrour, Gerhard Neumann, Johannes Fürnkranz, et al. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017. 5

[68] Xuesu Xiao, Bo Liu, Garrett Warnell, and Peter Stone. Motion planning and control for mobile robot navigation using machine learning: a survey. *Autonomous Robots*, 46(5):569–597, 2022. 1

[69] Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. Prediction-guided multiobjective reinforcement learning for continuous robot control. In *International conference on machine learning*, pages 10607–10616. PMLR, 2020. 2

[70] Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. *Advances in neural information processing systems*, 32, 2019. 2

[71] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699, 2021. 2