# Supplementary Material for 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

We provide additional details and analyses of the proposed method in this supplementary material. Section A provides detailed algorithm and theoretical analyses for group trajectory comparison. Section B provides implementation details and experiment settings. Section C provides evaluation results for RoboTHOR with detailed analyses on both ProcTHOR and RoboTHOR experiments. Section D provides implementation details and detailed results for the ablation study.

## A. Group Trajectory Comparison

### A.1. Algorithm

Algorithm 1 summarizes the reward weight prediction process from preference feedback on group trajectory comparisons. Suppose we are analyzing human preferences between two groups of weights in a simplex $\Delta_K$. The groups are defined as $G_1 = \{\mathbf{w} \mid \mathbf{a}^\intercal\mathbf{w} > b+1\} \subseteq \Delta_K$ and $G_2 = \{\mathbf{w} \mid \mathbf{a}^\intercal\mathbf{w} < b\} \subseteq \Delta_K$, where $\mathbf{a} \in \mathbb{R}^K$ and $b \in \mathbb{R}$. Suppose we collect two groups of trajectories corresponding to $G_1$ and $G_2$ as $\mathcal{T}_i = \{\{\tau_{i,j}\}_{j=1}^M \mid \tau_{i,j} \sim \pi(\cdot|\mathbf{w}) \text{ s.t. } \mathbf{w} \sim \text{Unif}(G_i)\}$ for $\forall i \in \{1,2\}$ where $M = |\mathcal{T}_1| = |\mathcal{T}_2|$. We define group preference that $G_1$ is preferred to $G_2$ if $|\{\tau_{1,j} \succ \tau_{2,j}\}| > \alpha M$, where $M$ is the size of $\mathcal{T}_1$ and $\mathcal{T}_2$, $\tau_{i,j} \in \mathcal{T}_i$ is a trajectory generated with a reward weight vector from $G_i$ in the $j^{th}$ episode, and $\alpha \geq 1/2$ is a threshold that determines the group preference. As described in Algorithm 1, we eliminate the volume corresponding to $G_2$ if the human prefers $G_1$ over $G_2$. Conversely, if $G_2$ is preferred over $G_1$, the volume in $G_1$ is removed. This process is repeated for multiple iterations and we perform constrained optimization to find the reward weight vector from the left weight space that maximizes the likelihood of group preference.

### A.2. Theoretical Analyses

For theoretical analysis, we assume the well-constructed set of trajectory pairs $\{\tau_{1,i}, \tau_{2,i}\}_{i=1}^M$ with a specific reward vector so that $\tau_{1,i}$ can be regarded as a trajectory sampled from $\mathbf{w}$ in $G_1$, and $\tau_{2,i}$ can be regarded as a trajectory sampled from $\mathbf{w}$ in $G_2$. This is not exactly the same as the sampling we did in Algorithm 1, where we uniformly sample $\mathbf{w}$ from each group $G_i$. The justification for this assumption will be thoroughly detailed in this section.

**Construction of trajectory pair set.** We will *construct and present pairs of trajectories*, denoted as $\{(\tau_{1,i}, \tau_{2,i})\}_{i=1}^M$, where $M$ represents the number of pairs in the group comparisons. For each trajectory $\tau_{j,i}$, it is associated with a reward, denoted as $r_{j,i,k}$. Here, $k \in [K]$ specifies the $k$th objective, $j = 1, 2$ indicates the first or second component of each pair, and $i \in [M]$ corresponds to the $i$th pair in the group comparison data. Then, if a human prefer $\tau_{i_p,i}$ to $\tau_{3-i_p,i}$ for every $i \in [M]$, (i.e., $i_p$ is the index of preferred group for $i$th pair, which is 1 or 2),

$$P(\{\tau_{i_p,i} \succ \tau_{3-i_p,i}\}_{i=1}^M) = \prod_{i=1}^M \frac{\exp(\mathbf{w}^{*\intercal}\mathbf{r}(\tau_{i_p,i}))}{\exp(\mathbf{w}^{*\intercal}\mathbf{r}(\tau_{i_p,i})) + \exp(\mathbf{w}^{*\intercal}\mathbf{r}(\tau_{3-i_p,i}))} \quad (1)$$

holds due to the Bradley-Terry Model. Since we can construct $\tau_i$ (as we have a lot of trajectories), we that $r_{2,i,k} - r_{1,i,k} = c_i a_k - c_i \frac{b_1+b_2}{2}$ for all $i \in [M], k \in [K]$. Without loss of generality, we set $b_1 - b_2 = 0.5$. We also set $c_i$ as a constant $1.4/(b_1 - b_2)$. Then, Equation (1) can be written as

$$P(\{\tau_{i_p,i} \succ \tau_{3-i_p,i}\}_{i=1}^M) = \prod_{i=1}^M \frac{1}{1 + \exp((-1)^{\mathbf{1}(i_p=1)} c_i(\mathbf{w}^{*\intercal}\mathbf{a} - \frac{b_1+b_2}{2})}. \quad (2)$$

Here, Equation (2) holds due to dividing Equation (1)'s denominator and numerator by $\exp(\mathbf{w}^{*\intercal}\mathbf{r}(\tau_{i_p,i}))$.

**The justification for the assumption.** A careful construction of $r_{1,i}$ and $r_{2,i}$ indicates that if $w^\star \in G_1$, i.e. $w^\star a > b_1$, $P(\tau_{1,i} \succ \tau_{2,i}) = \frac{1}{1+\exp(-c_i(\mathbf{w}^{*\intercal}\mathbf{a} - \frac{b_1+b_2}{2})} >$

**Algorithm 1:** Group Trajectory Comparison with Probabilistic Rejective Sampling

```
Require: # Queries: $N$, Rejection Probability: $1-\delta$;
# Objectives: $K$, # Trajectories per Group: $M$;
Initialize: Preference Buffer $\mathcal{D} \leftarrow \emptyset$, Constraint Buffer $\mathcal{C} \leftarrow \emptyset$
for $n = 1$ to $N$ do
    // Sample two groups of weights $G_1^n, G_2^n$ from the set of all possible weights $\Delta_K$
    // Generate $M$ trajectories for each group using the weights in the group
    // Calculate sub-rewards for $2 \cdot M$ trajectories across $K$ objectives
    $\mathbf{R}_i^n \leftarrow [\mathbf{r}(\tau_1^i),...,\mathbf{r}(\tau_M^i)]$ s.t. $\tau_j^i \sim \pi(\cdot|\mathbf{w}), \mathbf{w} \in G_i^n$;
    // Ask the user to give a preference label for these two groups
    $y_n \leftarrow$ preference label for pair $(G_1^n, G_2^n)$
    // Determine an inequality constraint based on the user's preference
    Choose $\mathbf{a}_n \in \mathbb{R}^K, \mathbf{c}_n \in \mathbb{R}^K, b_n \in \mathbb{R}$ s.t. $\mathbf{R}_2^n - \mathbf{R}_1^n = \mathbf{a}_n{\mathbf{c}_n}^\intercal - (b_n + 0.5)\mathbf{1}{\mathbf{c}_n}^\intercal$
    if User prefers the first group then
        Ensure the probability of this preference being correct is above the threshold $\delta$
    else
        Ensure the probability of the opposite preference being correct is above the threshold $\delta$
    end
    // Update Preference Buffer with the user's choice
    $\mathcal{D} \leftarrow \mathcal{D} \cup ((\sigma_n, \sigma_{n+1}), y_n)$
    // Update Constraint Buffer with the new constraint
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\mathbf{a}_n^\intercal \mathbf{w} < b_2\}$
end
Constrained Optimization:
// Maximize the expected likelihood of the user's preferences being correct, considering all constraints
$\max_{\mathbf{w} \in \Delta_K} \mathbb{E}[\log P(y_n | G_1^n, G_2^n; \mathbf{w})]$ s.t. $\mathbf{w} \in \cap_{n=1}^N C_i$
```

$1/2$, which says that the possibility of preferring $\tau_{1,i}$ to $\tau_{2,i}$ is larger than preferring $\tau_{2,i}$ to $\tau_{1,i}$. In the same way, if $w^\star \in G_2$, then they will probably more prefer $\tau_{2,i}$ to $\tau_{1,i}$. Therefore, we can regard preferring $\tau_{j,i}$ to $\tau_{3-j,i}$ as preferring $j$ to $3-j$. In this line, we assume that group $g$ will be said as preferred if $|\{\tau_{g,i} \succ \tau_{3-g,i}\}_{i=1}^M| \geq \alpha M$ where $\alpha > 1/2$.

**Probabilistic Guarantee.** Theorem 1 shows that with a sufficiently large group size, group comparison can achieve a probabilistic guarantee of predicting group preference with an error less than $\delta$.

***Theorem* 1.** *If* $M \geq \frac{-\log\delta - 1/2}{\alpha(1+0.25c)-1-(1-\alpha)\log(1-\alpha)}$ *holds,* $P(G_2 \succ G_1 \mid \mathbf{w}^* \in G_1) \leq \delta$ *with probability at least* $1-\delta$.

*Proof.* We will use the trajectory pairs defined in the aforementioned section. For each comparison between groups, we can decide the values of $a$ and $b_1, b_2$, thereby shrinking the domain of possible $\mathbf{w}$ values in $\Delta_K$ using a specific affine hyperplane. Iteratively employing these group comparisons will provide a high confidence level in determining the real value of $\mathbf{w}^*$.

Assume $\mathbf{w}^\star \in G_1$, but the case that human will select $G_2$ is

$$
\begin{aligned}
P(G_2 \succ G_1 \mid w^\star \in G_1) &= P_{\mathbf{w}^*}(|\{\tau_{2,i} \succ \tau_{1,i}\}_{i=1}^M| \geq \alpha M) \\
&= \sum_{k \geq \alpha M} \binom{M}{k} \left(\frac{1}{1+\exp(c_i({\mathbf{w}^*}^\intercal \mathbf{a} - \frac{b_1+b_2}{2})}\right)^k \\
&\quad \left(\frac{1}{1+\exp(-c_i({\mathbf{w}^*}^\intercal \mathbf{a} - \frac{b_1+b_2}{2})}\right)^{M-k} \\
&\leq \sum_{k \geq \alpha M} \binom{M}{k} \left(\frac{1}{1+\exp(c_i\frac{b_1-b_2}{2})}\right)^{\alpha M} \\
&\quad \left(\frac{1}{1+\exp(c_i\frac{b_2-b_1}{2})}\right)^{M-\alpha M} && (3) \\
&\leq \frac{M^{M-\alpha M+1}}{e\sqrt{M-\alpha M}\left(\frac{M-\alpha M}{e}\right)^{M-\alpha M}} \left(\frac{1}{1+\exp(c\frac{b_1-b_2}{2})}\right)^{\alpha M} \\
&\quad \left(\frac{1}{1+\exp(c\frac{b_2-b_1}{2})}\right)^{M-\alpha M} && (4) \\
&\leq \exp\left(M - \alpha M - \frac{b_1-b_2}{2}c\alpha M + \frac{1}{2}\right)/(1-\alpha)^{M-\alpha M} \\
&\leq \delta
\end{aligned}
$$

if $M \geq \frac{-\log\delta - 1/2}{\alpha\left(1+\frac{b_1-b_2}{2}c\right)-1-(1-\alpha)\log(1-\alpha)}$ holds, so we can get exponentially small error rate with respect to $M$. Here, Equation (3) used that $b_1 - b_2 > 0$, Equation (4) used $\binom{M}{k} \leq \binom{M}{\alpha M}$ and using Stirling formula of $\binom{M}{\alpha M}$. $\square$

Since $c_i > \frac{2-\log(2)}{b_1-b_2}$ and $\frac{1+(1-\alpha)\log(1-\alpha)}{\alpha} < 2 - \log(2)$ for any $\alpha > \frac{1}{2}$, $\alpha\left(1 + \frac{b_1-b_2}{2}c\right) - 1 - (1-\alpha)\log(1-\alpha) > 0$. Suppose the threshold of group preference $\alpha = 2/3$ and we want an accuracy of $\delta = 0.05$. Then, $P(G_2 \succ G_1 \mid \mathbf{w}^* \in G_1) \leq 0.05$ if $M > 4.996$. This theoretical analysis implies that we can efficiently remove a volume of the weight space by rejecting highly unprobable weights. We can do the same analysis for $P(G_1 \succ G_2 \mid w^* \in G_2)$. Therefore, even if we have a small $M$, we can efficiently make the error smaller, which means that group comparison accurately removes the volume.

## B. Experiment Setup

### B.1. Implementation Details

The agent observes an RGB image observation with a field of view of $63.453^\circ$ at each timestep. We define the time efficiency reward as a negative constant value at each timestep, fixed as $-0.01$ for all experiments. The path efficiency reward is defined as $\max(d_{t-1}, d_t)$, where $d_t$ is the distance from the agent's current location to the target location and $d_{t-1}$ is the distance from the agent to the target location at the previous timestep. House exploration reward considers the navigation history, by encouraging the agent to visit new areas with its corresponding reward as a constant $r_{house\_explore}$ if the agent is visiting its current location for the first time and the target object is not observed yet. If the target object is observed or the agent revisits a location, the house exploration reward is $0$. $r_{house\_explore}$ is $0.1$ for all experiments. The object exploration reward is calculated by counting the number of observed objects, where the reward is defined as $r_{object\_found} * n_{new\_visible}/n_{total}$, where $r_{object\_found}$ is a constant $4.0$, $n_{new\_visible}$ is the number of objects newly observed at the current timestep, and $n_{total}$ is the number of all objects in the environment. Safety reward is defined as $-r_{safety} \cdot n_{unreachable}$ if the target object is not visible in current observation and $0$ otherwise, where $r_{safety}$ is $0.005$. The number of unreachable locations near the agent $n_{unreachable}$ is higher than a threshold $n_{safety\_threshold}$. We draw a square grid of size $13 \times 13$ with the agent's current location in the center of the grid and a distance $0.25m$ unit grid size. Contrary to the sub-rewards for preference objectives that can be scaled with reward weights, we use a default success reward for ObjectNav to encourage the agent to find the target object. The default success reward is set to $10.0$ when the agent identifies the target object for the first time and $0$ otherwise. During training, we add the default success reward to the weighted sum of sub-rewards for preference objectives.

### B.2. List of Scenarios

- **Urgent:** The user is getting late to an important meeting and needs to quickly find an object in the house.
- **Energy Conservation:** The user wants to check an appliance in the house while the user is away, but the robot that has a limited battery life.
- **New Home:** The user just moved in and wants to find which furniture or object is located while inspecting the layout of the house as a video.
- **Post-Rearrangement:** After rearranging the house, the user does not remember where certain objects were placed. The user wants to find a specific object, while also inspecting other areas to confirm the new arrangement.
- **Quiet Operation:** At midnight, the user wants to find an object in the house without disturbing a sleeping child with any loud noise.

### B.3. Language Instruction Templates.

The input template for the language model is as follows:

```
{Task Description}
Objectives are {Definitions}.
Given an instruction, I want to know the weights
over the {K} objectives {Names of Objectives}.
The weights should be spiked, meaning that the
weight of the most important objective should be
much higher than the weight of the least important
objective.

Here are some examples.
1. Instruction: I want the agent to explore  ...
   Rationale: The instruction describes that ...
   Answer: [0.05,0.1,0.65,0.1,0.1]
...

Instruction: {Instruction}
Rationale:
Answer:
```

For ObjectNav, we provide the following template:

```
"""
In the object-goal navigation task in ProcTHOR,
an agent is placed within a simulated environment
containing various rooms and objects. The
agent's main goal is to find a specific object
in this environment. To assist the agent in its
navigation, it can be given different objectives
that determine how it behaves during its search.

Objectives are:
1. Time Efficiency: Aim to find the target object
using as few steps as possible.
2. Path Efficiency: Approach the goal using the
most direct route. Consider if you're taking the
shortest possible path.
3. House Exploration: Strive to explore the house
thoroughly. This involves checking many different
areas/rooms until you locate the target object.
4. Safety: Navigate while avoiding obstacles and
areas where you could get trapped or stuck.
```

```
5. Object Exploration: While finding the target
object, try to inspect as many objects as you
encounter.

Given a scenario, I want to know the weights
over the five objectives (Time Efficiency, Path
Efficiency, House Exploration, Safety, Object
Exploration).
The weights should be spiked, meaning that the
weight of the most important objective should
be much higher than the weight of the least
important objective.
The answer should be a list of five float
numbers, summed to 1.

Here are some examples.
1. Scenario: My kid is asleep. Navigate to an
apple in the kitchen without making any noise."
   Rationale: Based on the scenario, the agent
should prioritize safety the most, assigning 0.6.
Other objectives are not mentioned, assigning
(1-0.6)/4=0.1 for each objective.
   Answer: [0.1,0.1,0.1,0.6,0.1]
2. Scenario: I am in hurry. I want to find an
object before I am late for work.
    Rationale: Based on the scenario, time
efficiency is the most important, assigning 0.6.
Other objectives are not mentioned, assigning
(1-0.6)/4=0.1 for each objective.
    Answer: [0.6,0.1,0.1,0.1,0.1]
3. Scenario: I want to find a missing object in
my house. I looked into every room briefly but I
couldn't find it.
    Rationale: Based on the scenario, the agent
should explore the house thoroughly, assigning
0.4 for both house exploration and object
exploration. Other objectives are not mentioned,
assigning (1-0.4-0.4)/3=0.067 for each objective.
    Answer: [0.067,0.067,0.4,0.067,0.4]
4. Scenario: I bought an expensive furniture in
my house. I want to find an object, but I don't
want to damage the furniture.
    Rationale: Based on the scenario, the agent
should prioritize safety the most, assigning 0.6.
Other objectives are not mentioned, assigning
(1-0.6)/4=0.1 for each objective.
    Answer: [0.1,0.1,0.1,0.6,0.1]
5. Scenario: I'm recording a video in the living
room. While I'm working on this, I want the agent
to find an object for me. I don't want the agent
to move around too much since it might be too
noisy and appear a lot in the video.
    Rationale: Based on the scenario, the
agent should prioritize path efficiency and
time efficiency, assigning 0.4 for each.
Other objectives are not mentioned, assigning
(1-0.4-0.4)/3=0.067 for each objective.
    Answer: [0.4,0.4,0.067,0.067,0.067]
6. Scenario: I will have a home party this week,
but can't find where I put the vase to put on the
table. I want to find it surely by today. I have
enough time, so I just want the robot to find it.
    Rationale: Based on the scenario, the agent
should prioritize house exploration the most,
assigning 0.6. Object exploration is also
important, assigning 0.3. Other objectives are
not mentioned, assigning (1-0.6-0.3)/3=0.033 for
each objective.
    Answer: [0.033,0.033,0.6,0.033,0.3]

Scenario: {}
Rationale:
Answer:
"""
```

The test instructions are as follows:

```
instructions = [
"I'm getting late to an important
meeting. I need to quickly find an
object in the house.",
"I want to check an appliance in the
house while I'm away, but I forgot to
charge the robot last night. It seems
that the robot that has a limited
battery life, so I don't want the
robot to waste time while looking into
unnecessary regions.",
"I just moved in and want to find which
furniture or object is located while
inspecting the layout of the house as a
video.",
"After rearranging the house, I can't
remember where certain objects were
placed. I want to find a specific
object, while also inspecting other
areas to confirm the new arrangement.",
"My house has lots of valuable and
fragile artifacts. I want to find a
special-edition item among those.",
"It's in the midnight, and I want to
find an object in the house without
making any loud noise. I don't want to
disturb my child who is sleeping.",
]
```

For FleeNav, we use the following template:

```
"""
In the flee navigation task in ProcTHOR, an agent
is placed within a simulated environment with
the aim to move as far away as possible from its
starting position. The task tests the agent's
ability to maximize the distance from its initial
location while considering various objectives
that determine its behavior.

Objectives are:
1. Time Efficiency: Aim to find the target object
using as few steps as possible.
2. House Exploration: Strive to explore the house
thoroughly. This involves checking many different
areas/rooms until you find the farthest point
from the agent's initial location.
3. Safety: Navigate while avoiding obstacles and
areas where you could get trapped or stuck.
```

| **Method** | **Multi-Objective** | | **Prioritized Objective** | **Success** | **SPL** | **Distance to Goal** | **Episode Length** | **Sub Rewards ↑** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | ↑ | ↑ | ↓ | ↓ | Time Efficiency | Path Efficiency | House Exploration | Object Exploration | Safety |
| Promptable Behaviors (Ours) | Single-Policy | a | - | 0.470 | 0.190 | 1.632 | 182.540 | -1.825 | 1.179 | 3.370 | 2.095 | -3.162 |
| | | b | Time Efficiency | 0.410 | 0.185 | 1.736 | 156.470 | **-1.565** | 0.959 | 3.112 | 2.066 | -1.500 |
| | | c | Path Efficiency | 0.420 | 0.164 | 1.656 | 199.330 | -1.993 | **1.279** | 3.452 | 2.131 | -1.488 |
| | | d | House Exploration | **0.480** | 0.184 | 1.558 | 183.340 | -1.833 | 1.193 | **3.681** | 2.125 | -1.885 |
| | | e | Object Exploration | **0.480** | 0.207 | 1.643 | 182.070 | -1.821 | 1.246 | 3.534 | **2.137** | -1.573 |
| | | f | Safety | 0.450 | 0.175 | 1.629 | 156.770 | -1.568 | 1.247 | 3.122 | 2.098 | **-1.485** |

Table 1. **Performance in RoboTHOR ObjectNav.** We evaluate each method in the validation set with five different configurations of objective prioritization: uniform reward weights among all objectives and prioritizing a single objective 10 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 across all methods. Colored cells indicate the highest values in each sub-reward column.

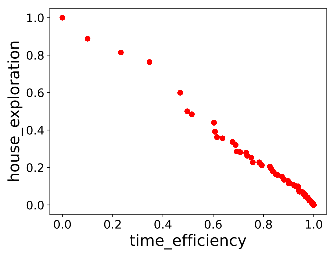
| **Method** | **Multi-Objective** | | **Prioritized Objective** | **Success** | **PLOPL** | **Distance to Furthest** | **Episode Length** | **Sub Rewards ↑** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | ↑ | ↑ | ↓ | ↓ | Time Efficiency | House Exploration | Safety |
| Promptable Behaviors (Ours) | Single-Policy | a | - | 0.876 | 0.950 | 3.360 | 57.560 | 0.784 | 0.629 | 0.688 |
| | | b | Time Efficiency | 0.890 | 0.946 | 3.501 | 56.030 | **0.920** | 0.053 | 0.367 |
| | | c | House Exploration | **0.902** | 0.954 | 3.017 | 59.320 | 0.627 | **0.907** | 0.932 |
| | | d | Safety | 0.854 | 0.946 | 3.630 | 63.890 | 0.220 | 0.730 | **0.934** |

Table 2. **Performance in RoboTHOR FleeNav.** We evaluate each method in the validation set with four different configurations of objective prioritization: uniform reward weight across all objectives and prioritizing a single objective 10 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 across all methods. Colored cells indicate the highest values in each sub-reward column.

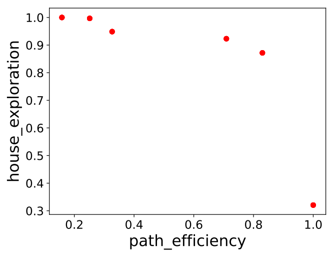
```
Given an instruction, I want to know the weights
over the four objectives (Time Efficiency, House
Exploration, Safety).
The weights should be spiked, meaning that the
weight of the most important objective should
be much higher than the weight of the least
important objective.

Here are some examples.
1. Instruction: Prioritize getting as far from
your starting point as possible, regardless of
the number of steps.
   Rationale: The instruction describes that time
efficiency is the least important, assigning 0.2.
House exploration and safety are not mentioned
but should be important than time efficiency, so
they are assigned 0.4 each.
   Answer: [0.2,0.4,0.4]
2. Instruction: Explore the environment
thoroughly while avoiding colliding to walls
and obstacles.
   Rationale: The instruction describes that
house exploration is the most important,
assigning 0.5. Safety is the second priority,
assigning 0.4. Time efficiency is not mentioned,
so it is assigned 0.1.
   Answer: [0.1,0.5,0.4]
3. Instruction: Your main goal is to explore
while distancing from the start.
   Rationale: The instruction describes that
house exploration is the most important,
assigning 0.6. Safety and time efficiency are
not mentioned, so they are assigned 0.2 each.
   Answer: [0.2,0.6,0.2]
4. Instruction: Safety is key. Move away, but
avoid any and all obstacles.
   Rationale: The instruction describes that
safety is the most important, assigning 0.6.
House exploration and time efficiency are not
mentioned, so they are assigned 0.2 each.
   Answer: [0.2,0.2,0.6]
5. Instruction: Avoid taking too many steps.
   Rationale: The instruction describes that time
efficiency is the most important, assigning 0.6.
House exploration and safety are not mentioned,
so they are assigned 0.2 each.
   Answer: [0.6,0.2,0.2]
6. Instruction: Prioritize safety first, then
exploration.
   Rationale: The instruction describes that
safety is the most important, assigning 0.6.
House exploration is the second important,
assigning 0.4. Time efficiency is not mentioned,
so it is assigned 0.0.
   Answer: [0.0,0.4,0.6]

Instruction: {}
Rationale:
Answer:
"""
```

## B.4. Human Evaluation Details

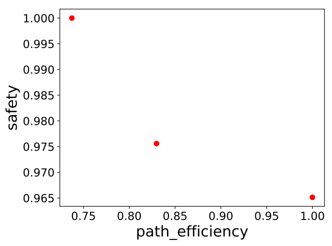
For human evaluation, we asked five participants to evaluate trajectories generated by the agent policy, conditioned on the predicted reward weights. Specifically, in each comparison, given a language instruction that described the user's situation, the participant had to choose the more preferred trajectory from a pair of trajectories. Each participant observed 50 pairs of trajectories for each weight prediction method in Table 4 in the main paper. For statistical testing, we performed a paired t-test, and the average p-value was calculated to be 0.021.



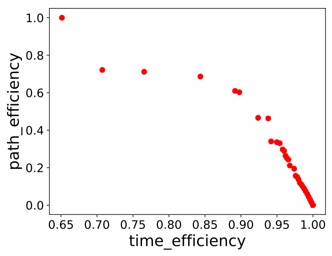
(a) Time Efficiency vs. House Exploration



(b) Path Efficiency vs. House Exploration



(c) Path Efficiency vs. Safety



(d) Time Efficiency vs. Path Efficiency

Figure 1. **Pareto front plots in ProcTHOR ObjectNav.** We plot the Pareto front on four different combinations of objective pairs in ProcTHOR ObjectNav. Data samples are generated with five different reward configurations, each prioritizing a single objective 4 times more than other objectives. Each data point corresponds to a sample with rewards on two objectives. Rewards are normalized for each objective.

## C. Additional Results and Analyses

In this section, we provide experiment results for RoboTHOR ObjectNav and FleeNav. Additionally, we visualize trajectories for numerous episodes and analyze those qualitatively. Lastly, detailed analyses for reward weight prediction experiments are provided.

### C.1. RoboTHOR Experiment Results

**RoboTHOR ObjectNav.** Results in Table 1 show that the proposed *Promptable Behaviors* outputs different agent behaviors and performances based on the prioritization of objectives. Regarding the general performance, prioritizing house exploration or object exploration shows the highest success rate of 48.0%. When time efficiency is prioritized (row b in Table 1), the agent achieves % higher time efficiency reward compared to the case when no objective is prioritized (row a in Table 1). Similarly, prioritizing a single objective improves the corresponding sub-reward.

**RoboTHOR FleeNav.** Results in Table 2 also show that our method can effectively prompt agent behaviors by adjusting the reward weights across multiple objectives. Among all reward weight configurations, prioritizing house exploration shows the highest success rate. When safety is prioritized, the agent shows a higher safety reward compared to all other reward weight configurations.

### C.2. Pareto Front Analysis

**Efficiency and house exploration are conflicting objectives.** Pareto front plots in Figure 1 (a) and (b), and Figure 2 show that time efficiency and path efficiency are conflicting objectives. Comparing Figure 1 (a) and (b), path efficiency could be interpreted as more conflictive against house exploration since the area below the Pareto front in (b) is larger than the area in (a). Interestingly, Figure 1 (d) illustrates a nearly convex Pareto front when time efficiency and path efficiency are compared. This implies that although time efficiency and path efficiency both aim *efficiency* on the agent's behavior, a notable difference exists between these two objectives. Comparing safety with path efficiency in Figure 1 (c), the Pareto front is illustrated as a concave curve. Comparing the curve with the Pareto front in Figure 1 (b), path efficiency is more conflictive with house exploration than safety.

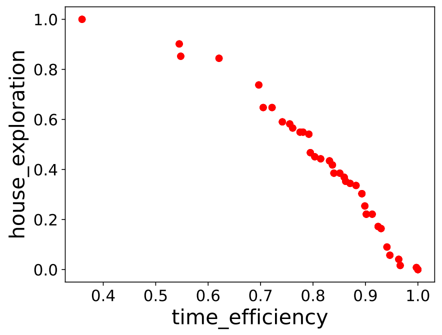
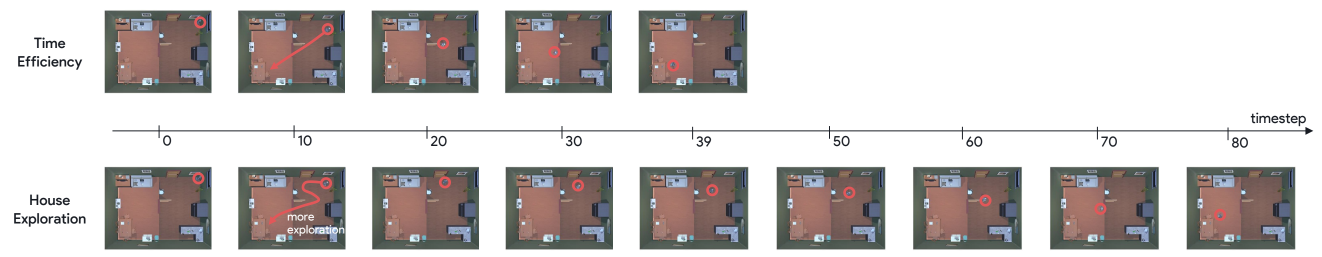


Figure 2. **Pareto front comparing time efficiency and house exploration in ProcTHOR FleeNav.** We plot the Pareto front in ProcTHOR FleeNav comparing time efficiency and house exploration. Data samples are generated with three different reward configurations, each prioritizing a single objective 3 times more than other objectives. Each axis denotes the reward for the corresponding objective and the rewards are normalized following Table 1.

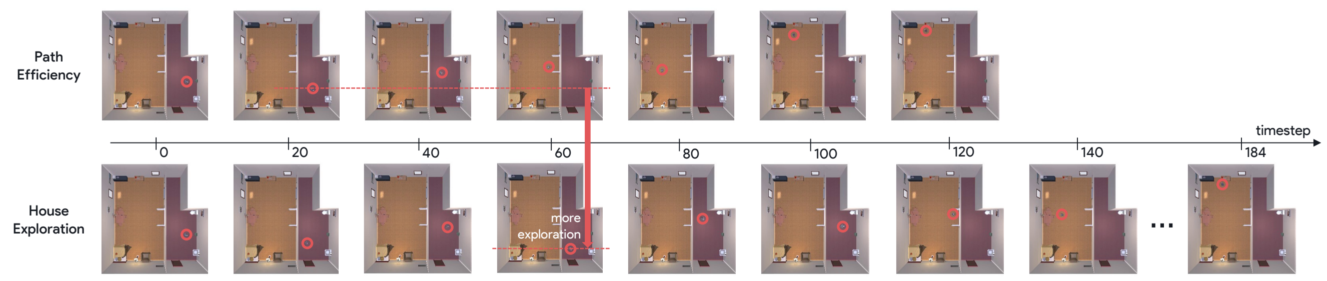
### C.3. Trajectory Visualizations

We visualize trajectories to compare how agent behaviors are different when different objectives are prioritized.

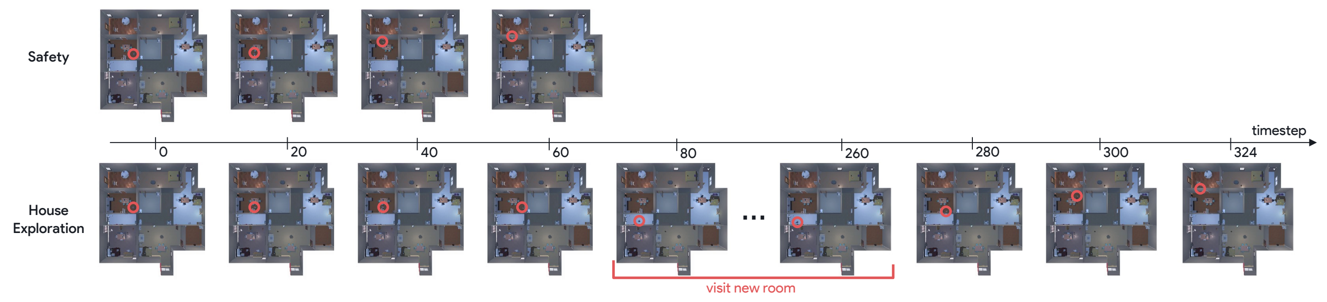
**Time efficiency saves time.** Figure 3 (a), Figure 4, Figure 6 (a), and Figure 5 shows that within the same episode, the agent finds the target object faster when time efficiency is prioritized than other objectives. In Figure 3 (a), the agent that prioritizes time efficiency directly gets out of a room and finds the target object earlier than the case when house exploration is prioritized. Even comparing time efficiency against path efficiency in Figure 4, prioritizing time efficiency encourages the agent to find the target object earlier.



(a) Time Efficiency vs. House Exploration



(b) Path Efficiency vs. House Exploration



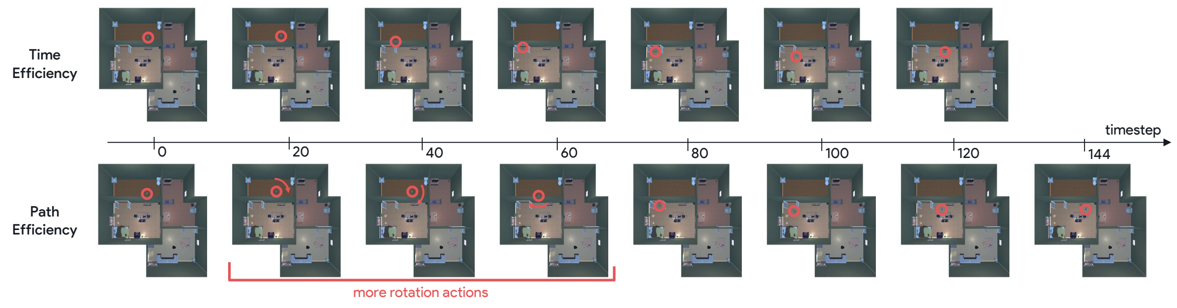
(c) Safety vs. House Exploration

Figure 3. **Comparing House Exploration with other objectives.** We compare agent trajectories when house exploration is prioritized with when one of three objectives (time efficiency, path efficiency, and safety) is prioritized.

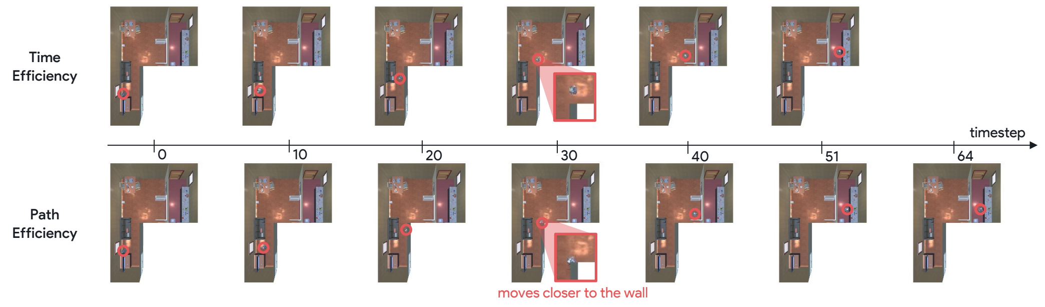
Notably, we found that time efficiency and path efficiency are correlated but different in some cases. As shown in Figure 4 (a), prioritizing path efficiency often causes more rotation actions than prioritizing time efficiency. This could be because changing the agent's orientation can help the agent find shorter paths. Figure 4 describes another difference: prioritizing path efficiency encourages the agent to move closer to a corner of the wall. This could be due to the nature of calculating the geodesic shortest path since the shortest path is usually a series of straight-line segments that connect corner places in the environment. In Figure 5, prioritizing time efficiency results in a shorter path than prioritizing object exploration. These results imply that the agent learns to finish the episode faster when time efficiency is prioritized.

**House exploration enables the agent to explore the house thoroughly.** Figure 3 visualizes how agent trajectories are changed when house exploration is prioritized. In Figure 3 (a), compared to the trajectory that prioritizes time efficiency, the agent prioritizing house exploration explores the room more before getting out of the room. Similarly, Figure 3 (b) illustrates an episode where the agent explores more when house exploration is prioritized than when path efficiency is prioritized. In Figure 3 (c), the agent visited more rooms when house exploration is prioritized, compared to the case when safety is prioritized. These qualitative results demonstrate the effectiveness of *Promptable Behaviors* that prioritizing house exploration encourages the agent to explore the house thoroughly.

**Object exploration encourages the agent to inspect more objects.** Figure 5 visualizes the RGB observations with objects detected in each trajectory. The figure shows that prioritizing object exploration makes the agent inspect objects more in detail, often encouraging getting close to the observed objects. For instance, the agent observed a bed, a chair, and a cabinet closer than the trajectory that prioritizes time efficiency. Also, the agent changed the orientation of



(a) House 18 Episode - Find AlarmClock



(b) House 82 Episode - Find HousePlant

Figure 4. **Time Efficiency vs. Path Efficiency.** We compare agent trajectories when time efficiency is prioritized and when path efficiency is prioritized in the same episode. Both episodes in (a) and (b) show that prioritizing time efficiency encourages the agent to end the episode faster than prioritizing path efficiency. Prioritizing path efficiency showed interesting behaviors, such as performing more rotation actions in (a) and moving closer to the wall in (b), compared to prioritizing time efficiency.

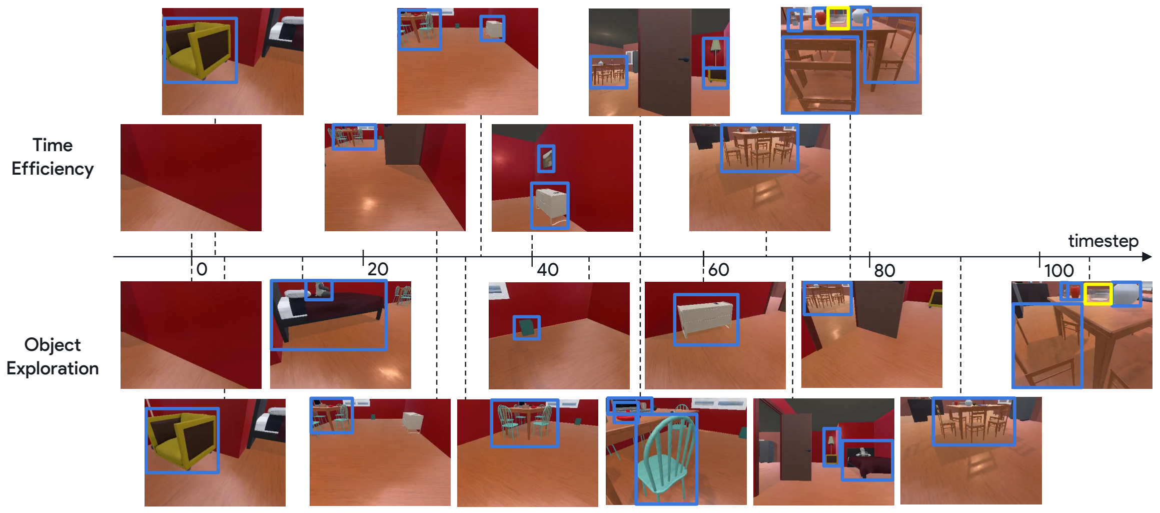
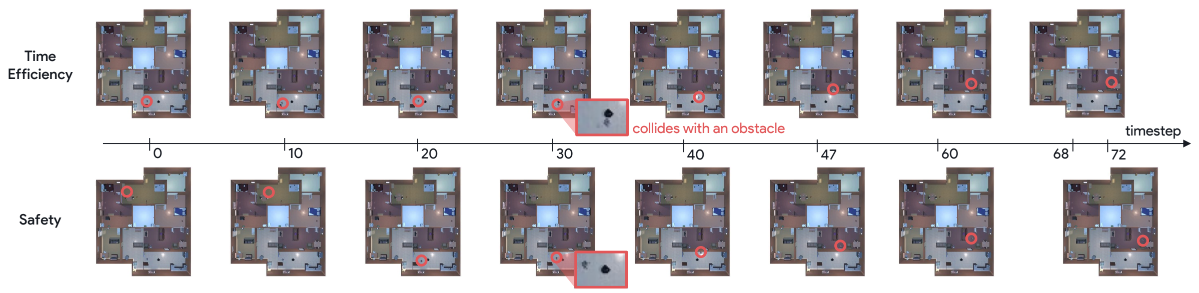
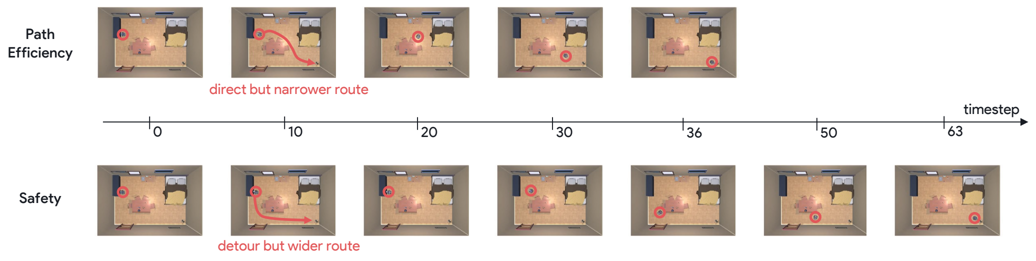


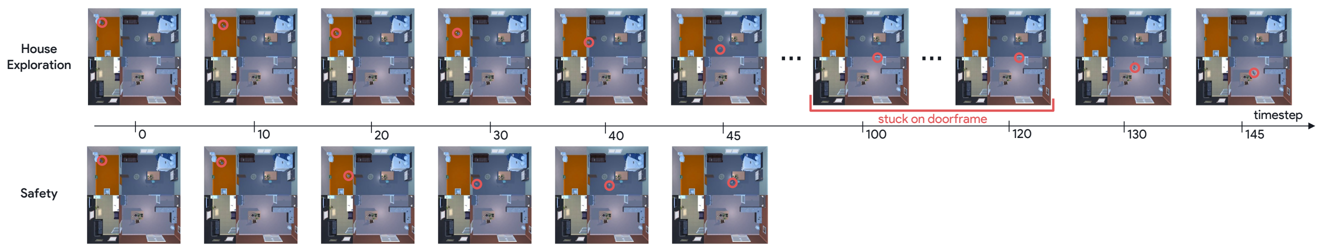
Figure 5. **Time Efficiency vs Object Exploration.** We compare agent trajectories when time efficiency is prioritized with when object exploration is prioritized. Blue boxes denote objects detected in each image and yellow box describes the bounding box of the target object.



(a) Time Efficiency vs. Safety



(b) Path Efficiency vs. Safety



(c) House Exploration vs. Safety

Figure 6. **Comparing Safety with other objectives.** We compare agent trajectories when safety is prioritized with when one of three objectives (time efficiency, path efficiency, and house exploration) is prioritized. (a) The agent collides with an obstacle when time efficiency is prioritized the most. When safety is prioritized, the agent does not collide with any object and smoothly finds the target object. (b) When path efficiency is prioritized, the agent follows a direct but narrower route compared to the case when safety is prioritized. (c) In this episode, multiple objects corresponding to the target object category exist in the house. The agent gets stuck on a doorframe when house exploration is prioritized, while the agent smoothly passes the door when safety is prioritized.

the view by performing `LookUp` and `LookDown` actions before leaving the room when object exploration is prioritized. In contrast, the agent prioritizing time efficiency did not perform any `LookUp` or `LookDown` actions and got out of the room in an earlier timestep.

**Safety avoids colliding with objects, doors, and walls.** Figure 6 describes agent trajectories that prioritize safety or one of three objectives (time efficiency, path efficiency, and house exploration). In Figure 6 (a), prioritizing time efficiency resulted in a collision between the agent and an obstacle in the middle of a room, while the safety-prioritized agent did not collide with any objects. In Figure 6 (b), the agent prioritizing safety took a detour but a wider route to move towards the target object, while the agent prioritizing path efficiency followed a direct and narrower route. In Figure 6, the agent prioritizing house exploration got stuck on a doorframe for 20 timesteps, while the safety-prioritized agent did not get stuck in any places. These results imply that the agent learns to avoid colliding with objects and places that could potentially make the agent get stuck.

**Our agent covers various combinations of preference objectives.** Although we primarily evaluate the policy based on cases where a single objective is prioritized, our policy can handle any combination of objectives in the reward weight space. Since we randomly sample the reward weights during training, the policy has the capability to cover various combinations of objectives, corresponding to diverse preferences of multiple users. In contrast, Prioritized EmbCLIP learns a set of policies focusing on one objective per policy and therefore can only generate a single agent behavior at a time. Our method is more flexible: rather than choosing just one objective, weighting across possible objectives enables the agent to cover a broader range of human preferences over agent behaviors.

## C.4. Reward Weight Prediction

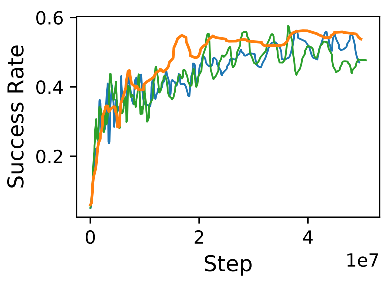
Table 3 shows the full results of reward weight prediction experiments. Group trajectory comparison when $M = 5$ aligns with the probabilistic guarantee of group preference with an error less than $\delta = 0.01$, resulting in 86.2% prediction accuracy with only 10 group comparisons. The results show that group trajectory comparison with $M = 5$ and 10 feedback shows a similar performance of pairwise trajectory comparison with 500 feedback. This implies that group trajectory comparison effectively reduces human effort on providing preference feedback. Comparing ChatGPT [1] and Llama-2-70B [2], Llama-2-70B model generated more peaked reward weights compared to ChatGPT, while the prediction accuracy was higher using ChatGPT.

## D. Ablation Study

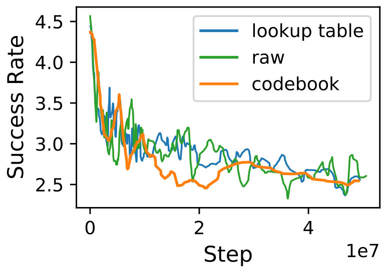
**Is Codebook Effective in MORL?** We examine the effectiveness of the codebook module for encoding reward weight vector in MORL. The training curves, shown in Figure 7, illustrate that using codebook results in a more sta-

| **Weight Prediction Methods** | | | **Acc.** ↑ | **GGI** ↑ |
|---|---|---|---|---|
| Input | Model | $N$ | | |
| Human Demonstrations | - | 1 | 0.707 | 0.347 |
| Preference Feedback | Pairwise Comparison (M=1) | 10 | 0.369 | 0.800 |
| | | 20 | 0.356 | 0.800 |
| | | 50 | 0.358 | 0.800 |
| | | 100 | 0.505 | 0.800 |
| | | 200 | 0.587 | 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 |
| | Llama-2-70B | 1 | 0.314 | 0.608 |
| | w/ ICL | 1 | 0.360 | 0.675 |
| | w/ CoT | 1 | 0.313 | 0.582 |
| | w/ ICL + CoT | 1 | 0.287 | 0.568 |

Table 3. **Comparison of Three Weight Prediction Methods.** We predict the optimal reward weight vector from the collected human demonstrations, preference feedback on trajectory comparisons, and language instructions. For trajectory comparison methods, the total number of observed trajectories is $2NM$. mention tha tthis is objectnav



(a) Success Rate



(b) Distance to Target

Figure 7. **Training Curves with Different Weight Embedding Methods in ProcTHOR ObjectNav.** We compare three weight embedding methods: raw, lookup table, and codebook.

| Method | Input Type | Input Range | Success ↑ | SPL ↑ |
|---|---|---|---|---|
| Raw | Conti. | [0.0, 1.0] | 0.393 ± 0.064 | 0.259 ± 0.020 |
| Lookup Table | Discrete | [0, 10] | 0.409 ± 0.007 | 0.284 ± 0.003 |
| Codebook (Ours) | Conti. | [0.0, 1.0] | 0.396 ± 0.027 | 0.298 ± 0.022 |

Table 4. **Comparison of Reward Weight Encoding Methods.** We evaluate each method in the validation set with 11 different configurations of objective prioritization: uniform reward weights among all objectives, prioritizing a single objective 4 times as much as other objectives, and prioritizing a single objective 10 times as much as other objectives.

ble performance compared to using a lookup table extended from [3] and using raw reward weight vectors without encoding. After training with each reward weight encoding method for $50M$ steps, we evaluate the agent with 11 different reward configurations, as described in Table 4. The proposed encoding method using codebook improved the average success rate by $0.76\%$ and the standard deviation of success rate by $57.8\%$ compared to using raw reward weight vectors without encoding. This indicates that the training process is stabilized through the codebook module. While the lookup table method did exhibit a higher average success rate and SPL than using codebook, we address that using integer weights to represent reward configurations has a critical limitation: ambiguity in the reward weight prediction phase. For instance, different integer weight vectors like $[4, 1, 1, 1, 1]$ and $[8, 2, 2, 2, 2]$ may represent identical preferences, leading to multiple potential solutions and complicating the inference of optimal reward weights from human preferences. In contrast, our method represents human preferences using continuous real-values weight vectors, such as $[0.5, 0.125, 0.125, 0.125, 0.125]$.

## References

[1] OpenAI. ChatGPT. `https://openai.com/blog/chatgpt`, 2022. 10

[2] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 10

[3] 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. 11