Ask4Help: Learning to Leverage an Expert for Embodied Tasks
– Supplementary Materials –

Kunal Pratap Singh *
PRIOR, Allen Institute for AI

Luca Weihs
PRIOR, Allen Institute for AI

Alvaro Herrasti
PRIOR, Allen Institute for AI

Jonghyun Choi
Yonsei University

Aniruddha Kembhavi
PRIOR, Allen Institute for AI

Roozbeh Mottaghi
PRIOR, Allen Institute for AI

Appendix

A Compute Resource Details

Our models are trained on an internal GPU cluster which has 8 NVIDIA TITAN V GPUs with 12 GB memory on each card. The server also has a 64 CPU core which allows us to run 60 processes in parallel, and hence speed up training. We achieve 600 frames-per-second (fps) for training the ASK4HELP policy on Object Navigation, and 150 fps for Room Rearrangement.

B Human Trial Setup

We modified AI2-THOR [2] to run custom actions that are intercepted and resolved in python before reaching unity. In unison, we modified AllenAct to allow standard input from the terminal and built a simple interface by customizing THOR's terminal interfacing tools, to display the agent's view of the scene and take user choices for actions to send them back to the model.

We conducted human trials internally with the members of our research organisation. We received verbal consent from each participant for the same. We provide a complete set of installation instructions to allow each participant to conduct a trial autonomously. Figure 1 shows the interface and instructions in the terminal provided to the user.

As seen in Figure 1, we present the user with a top-down view of the environment, and the egocentric view from the agent's camera. A red boundary is used to indicate to the user that the agent requires help. When the agent asks for help with target object is displayed on the egocentric window. The user can use W key to move forward, the arrow keys to rotate around and looking up and down. If the user believe they have found the object, they can press the E key to end the episode.

C CE-ϵ Result on Full Validation Set

We present the full validation set results of the CE-ϵ baseline in Table 1. These results reaffirm the findings presented in Section 4.3.3 of the main paper. Our ASK4HELP policy is able to achieve

*correspond to kunals@allenai.org for any queries/comments

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

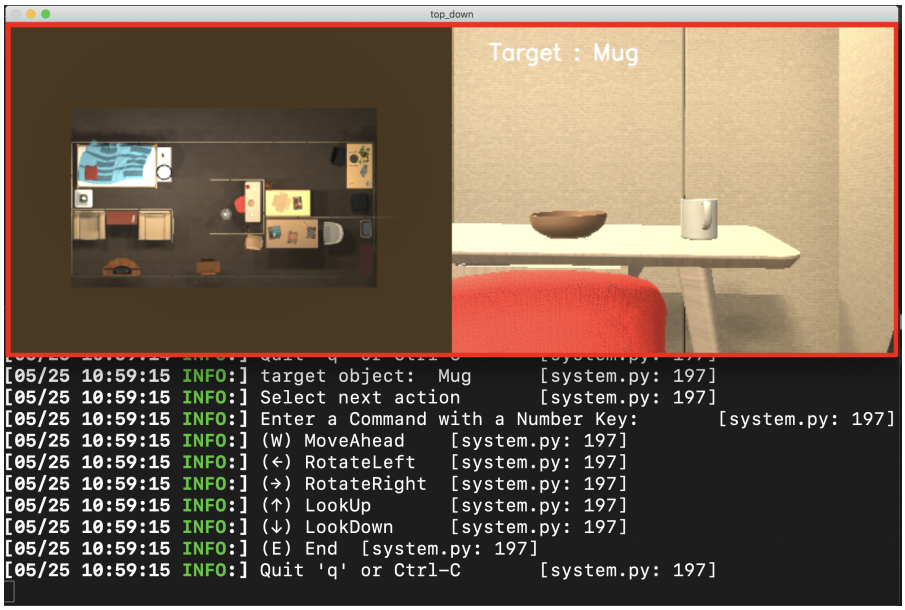

Figure 1: Human Trial Interface for ASK4HELP.

significantly performance performance improvements even with a noise corrupted version of the expert.

Expert	OP (%)	SR (%)	SPL (%)	EL
NavMesh [1]	11.18	86.22	32.81	169.5
CE-0.1	12.15	86.44	32.91	174.6
CE-0.2	12.3	84.39	32.48	171.8
CE-0.4	13.98	82.11	31.29	185.4
CE-0.8	20.89	67.1	27.1	231
None	0	52.4	24.7	168

Table 1: SR, SPL and EL represent the Success Rate, Success by path weighted length and episode length metrics respectively. Expert column indicates the oracle used. OP corresponds to the Oracle Proportion metric.

D Broader Societal Impact

We propose an ASK4HELP policy that can be plugged on Embodied AI models to improve their reliability and performance. This line of work can potentially boost the applicability of our current models at the expense of limited human intervention. This would allow us to deploy E-AI models in safety-critical situations like disaster relief, thereby reducing human risk and effort. Additionally, since we do not retrain the underlying E-AI model, we achieve higher performance with significantly less compute, thereby reducing the energy-footprint of these models.

Although, there are no direct negative impacts of our work, robotic agents can be used for malicious applications to harm life and property. Although, our work does not pose such a risk because we train and evaluate our agents in simulation, ethical deployment of these methods is something that needs to be carefully considered.

E Qualitative Results

We present qualitative results in the form of videos attached to the supplementary material. Subsequently, we describe various ASK4HELP behaviors in different episodes. The top left corner of each frame indicates the target object. The color of the progress bar indicates whether the agent or the expert governs control. If the progress bar is green, then the action at the current time step is taken by the agent policy, and if it is red, then the action was taken by the expert. All the episodes are from the unseen validation set.

In example 1, the agent is looking for a houseplant. As it can be seen in the video the underlying E-AI policy is having a hard time exploring the room. It keep going back and forth within the same region. The ASK4HELP policy allows the agent to attempt the task for some duration before requesting help from the expert. Once the expert takes over, it breaks the agent out of the loop, and brings it into the open hallway. Thereafter, the agent recognizes the houseplant and successfully ends the trajectory.

In example 2, the expert help allows the agent to bypass an immovable obstacle. The target object for this episode is a laptop. The agent policy does a good job in exploring the environment and finding the correct region where the laptop is located. However it fails to successfully accomplish the task, it does not know how to get close enough to its target by going around the chairs that are blocking its path. After letting the agent try the task by itself, the ASK4HELP policy requests expert help, and the expert takes the agent around the chair. Thereafter, the agent successfully recognizes the laptop and ends the trajectory.

In example 3, the target object Alarm Clock is on top of a cabinet. The agent goes around the room and explores various locations including around the cabinet. It fails to find the Alarm Clock, because it keeps facing straight and does not look up. The ASK4HELP policy fixes this by allowing the expert to correct the agent gaze, which helps it identify the Alarm Clock and succeed.

Example 4 represents a case where the agent policy fails to look for the object in all parts of the room. If we look at the top down view, it explores all regions except the bottom right where the target object (basketball in this case) ends up being. The ASK4HELP policy helps the agent by requesting the expert to take the agent to the correct region, after which the agent is able to find the basketball successfully.

In example 5, the agent is looking for a spray bottle. It explores both parts of the environment but fails to turn around in the left half (in the top down view), thereby missing the spray bottle. The ASK4HELP policy requests expert help, which turns the agent around at the appropriate moment, allowing the agent to see the spray bottle and navigate towards it.

Example 6 is also an example of incomplete exploration. The agent keep going around in one half of the environment, thereby failing to find the baseball bat. The expert help requested by the ASK4HELP policy takes it to the other half of the environment, thereby avoiding task failure with very limited assistance.

F Additional Baselines

We present a few additional baselines to highlight the efficacy of ASK4HELP.

• *Replacing* ASK4HELP *with a heuristic policy*: We implement a hard coded policy by running the pre-trained agent for M steps and then the expert for N steps, after M+N steps, the episode ends. For both M and N, we try the following values, [10,20,30,40]. We present the results in Table 2. To put the comparison into perspective, a policy trained with ASK4HELP uses an average of 17 expert steps and 157 agent steps, and achieves a success rate of 86.3.

G Clarification on Success Prediction Module's role

For some initial trials, we trained an offline model called the success prediction network that takes in the belief state at a particular time step and outputs the probability of task success at a given time step. We trained this network offline based on a pre-trained model's trajectories.

M ↓ \ N →	10	20	30	40
10	23.8	46.2	66.2	82.3
20	28.2	48.67	68.72	72.72
30	30.7	49.3	68.5	81.4
40	33.7	50.1	68.2	80.9

Table 2: Heuristic Policy results. M and N denote the number of agent and expert steps respectively.

For the task of Object Navigation, given its simple nature, the success prediction network is able to capture the characteristics of failures reasonably well. We find that by masking out either the belief input or the success prediction input (but not both), we achieve similar results to the ones presented in the main paper. Although, we would like to point out that the success prediction uses belief as an input. However, when we try masking out the belief input for Rearrangement, we find that the Ask4Help policy simply learns to request expert help 97% of the time, which is very undesirable from the user perspective. This is due to the complex nature of rearrangement which makes it much harder for success prediction to capture the modes of failure where the agent might require assistance.

We believe this indicates that having the agent belief as an input allows our method to be generally applicable to tasks of varying difficulty, especially for complex ones like rearrangement where purely relying on success prediction may not be enough.

References

[1] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. Unity: A general platform for intelligent agents. *arXiv*, 2018. 2

[2] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*, 2017. 1