Mini-BEHAVIOR: A Procedurally Generated Benchmark for Long-horizon Decision-Making in Embodied AI

Emily Jin[1*] **Jiaheng Hu**[2*] **Zhuoyi Huang** [1]
Ruohan Zhang[1] **Jiajun Wu**[1] **Li Fei-Fei**[1] **Roberto Martín-Martín**[2]
[1]Stanford University [2]University of Texas at Austin
{emilyjin,zhuoyih,zharu}@stanford.edu
{jhu, robertomm}@cs.utexas.edu
{jiajunwu, feifeili}@cs.stanford.edu

Abstract

We present Mini-BEHAVIOR, a novel benchmark for embodied AI that challenges agents to use reasoning and decision-making skills to solve complex activities that resemble everyday human challenges. The Mini-BEHAVIOR environment is a fast, realistic Gridworld environment that offers the benefits of rapid prototyping and ease of use while preserving a symbolic level of physical realism and complexity found in complex embodied AI benchmarks. We introduce key features such as procedural generation, to enable the creation of countless task variations and support open-ended learning. Mini-BEHAVIOR provides implementations of various household tasks from the original BEHAVIOR benchmark, along with starter code for data collection and reinforcement learning agent training. In essence, Mini-BEHAVIOR offers a fast, open-ended benchmark for evaluating decision-making and planning solutions in embodied AI. It serves as a user-friendly entry point for research and facilitates the evaluation and development of solutions, simplifying their assessment and development while advancing the field of embodied AI. Code is publicly available at `https://github.com/StanfordVL/mini_behavior`.

1 Introduction

Embodied AI focuses on developing agents that can interact with the environment and perform complex tasks. As in other AI fields, progress in embodied AI has largely been driven and supported by a rich set of benchmarks that enable researchers to focus on common challenges and measure progress in equal and fair conditions [2, 6, 13, 16, 10, 20]. The impressive progress in the field has led to increasingly richer and more complex benchmarks, pushing the boundaries of AI agents to perform navigation [22, 21], stationary manipulation [14, 12], and their combination in mobile manipulation [7, 9]. However, these more complex benchmarks are costly to run and train agents for, leading to a high entry point and slow development.

A complex benchmark with those characteristics is BEHAVIOR, *a benchmark for everyday household activities in virtual, interactive, and ecological environments*. BEHAVIOR (both the 100 and 1K variants) seeks to drive the development of embodied agents by challenging them to perform everyday household tasks that are diverse, realistic, and long horizon [11, 17]. With the support of advanced physics simulators [10] and several interactive simulated scenes, BEHAVIOR defines multiple and diverse household tasks, ranging from `installing a printer` to `cleaning up the kitchen` to `preparing a salad`. These tasks are long-horizon and heterogeneous: they require thousands

[*]Equal Contribution

2nd Workshop on Agent Learning in Open-Endedness (ALOE) at NeurIPS 2023.

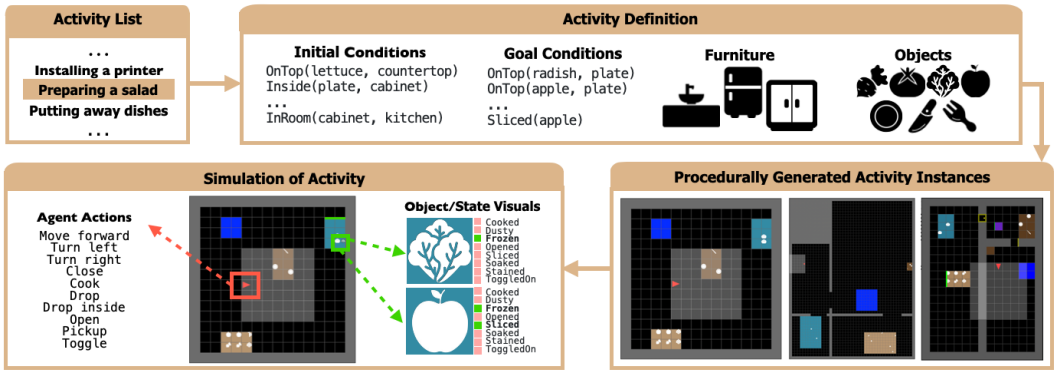


Figure 1: From Activity Definition to Simulated Activity Instance in Mini-BEHAVIOR: Given a long-horizon household activity definition in first-order logic, together with a dataset of simple object and furniture models, our Mini-BEHAVIOR benchmark procedurally generates unlimited activity instances in a fast, easy-to-use 3D Gridworld environment.

of decision-making steps to be completed and for the agent to master a variety of skills, from pick and place, to open/close, to switch on/off. A significant challenge in BEHAVIOR arises from the need to generalize to all possible variations of houses the AI agent may encounter, which requires open-ended training in different scenes.

To facilitate the process of creating solutions for embodied AI, researchers and AI practitioners developed simplified versions of these complex benchmarks [1, 18, 3]. These environments are simple, lightweight, and fast to run, supporting primitive decision-making tasks. One of the most used ones is MiniGrid [4], with simplified versions of navigation. In MiniGrid, dynamics, actions, and states are discretized, simplifying simulation and decision-making. Due to its simplicity, it is highly reconfigurable and easy to extend, becoming one of the most used tools by embodied AI practitioners to create, test, and develop algorithmic ideas. However, MiniGrid and the other reduced AI benchmarks fail to provide a simpler developing version of complex benchmarks for long-horizon, heterogeneous AI tasks such as BEHAVIOR, combining navigation and manipulation, with richer semantics, involving complex task planning, and with practical applications such as household activities.

We present a new simplified benchmark, **Mini-BEHAVIOR**, that provides the benefits of simple benchmarks for fast prototyping and training, while keeping the most relevant problem structure and the characteristics from the task-level decision-making challenge in BEHAVIOR: the long-horizon, heterogenousness of the activities, and the multi-object, multi-state, high variability of household environments. Mini-BEHAVIOR re-defines BEHAVIOR tasks in a novel 3D Gridworld environment with support for procedural generation for potentially infinite variants of each task.

We extend over previous gridworld environments by including semantically classified objects – `printer`, `plate`, `apple`,...– with multiple possible states –`frozen`, `onTop`, `closed`,...– in 2D gridworlds with rooms and furniture where we added a third dimension to enable a simplified vertical axis –`plates` can be `onTop` a `table`. We also extended the agents' capabilities to be able to change those objects' states and observe them. We provide implementation for multiple household activities from the original BEHAVIOR, and starter code to collect data and train reinforcement learning agents, while keeping the simplicity and speed of the MiniGrid benchmark.

In summary, the Mini-BEHAVIOR benchmark contains multiple long-horizon heterogenous tasks, is simple and fast, and open-ended through procedural generation. We hope that with Mini-BEHAVIOR, we provide the community with:

1. a simple tool to standardize the evaluation of task-level decision-making and planning solutions, going beyond pure symbolic domains into problems with the constraints of embodied AI;

2. an open-ended generative system that can create potentially infinite variations of each task;

3. an easy starting point and fast prototyping platform for embodied AI solutions, keeping enough similarities to its more realistic, fully simulated version that we hope it will enable the easy transfer of some of the solutions;

2 Related Work

Datasets and benchmarks have long played an important role in driving progress in AI. In embodied AI, several benchmarks have been developed to calibrate progress in the field, including Rearrangement, TDW Transport Challenge, VirtualHome, ALFRED, Interactive Gibson Benchmark, MetaWorld, and BEHAVIOR [2, 6, 13, 16, 10, 20]. Of these, BEHAVIOR is the most comprehensive benchmark, with realistic, diverse, and complex tasks. BEHAVIOR tasks include household activities that are representative of the kinds of activities that humans face in their everyday lives. BEHAVIOR activities are also diverse in the sense that they involve a variety of scenes, objects, and actions, and they are complex since they require both low-level manipulation and high-level planning skills.

One key feature of BEHAVIOR and other embodied AI benchmarks is that they use complex simulator environments to physically represent the real world. While this physical realism makes these benchmarks highly valuable to the robot learning community, they can incur huge computational costs and drastically reduce the prototyping speed, making them unsuitable for rapid experimentation and development.

To address the need for more accessible benchmarking tools, researchers have created simplified versions of these benchmarks in 2D-Gridworld environments to benchmark decision-making algorithms. Environments such as Griddly, MazeBase, Overcooked, and MiniGrid were developed to benchmark RL agents for 2D-games and goal-oriented tasks [1, 18, 3, 4]. Thus far, MiniGrid is the best for supporting various goal-oriented tasks, and is known to be fast and easily customizable.

Even so, MiniGrid is only able to support a small range of goal-oriented tasks and lacks the complexity to support realistic activities such as the ones in BEHAVIOR. It provides few actions for interacting with the environment, limited objects, and physical grid constraints. Each cell is only allowed to have one object, and each object takes up exactly one cell. As a result, MiniGrid is 2D, and there are no notions of size, height, and vertical or internal placement. For example, both an apple and a bed have the same effective size and height, and it is not possible to place a pan on top of a counter or an apple inside the refrigerator.

By comparison, Mini-BEHAVIOR extends MiniGrid's capabilities by adding procedural generation capabilities, as well as support for symbolic states, more objects and primitive actions, a third vertical dimension, and multi-cell objects. In the meantime, Mini-BEHAVIOR is built on top of the MiniGrid library and inherits aspects such as its speed and simplicity [4]

3 Mini-BEHAVIOR Benchmark Features

With Mini-BEHAVIOR, we hope to bridge the gap between the complexities of benchmarks like BEHAVIOR and the simplicity of MiniGrid. The Mini-BEHAVIOR benchmark includes 20 household tasks of varying difficulty. The tasks are instantiated in a simulated household environment with support for 96 objects, 23 states, and 15 actions. The three key features of the benchmark are:

1. **diverse and complex tasks:** a standardized set of 20 tasks for reproducible research that require reasoning and high-level planning skills;

2. **fast simulation environment:** a simple, lightweight, fast, and easy-to-use Gridworld environment, suitable for rapid prototyping of decision-making algorithms;

3. **procedural generation:** ability to generate unlimited activity instances with physically realistic object placement to support open-ended learning

We proceed by discussing the three key features in greater detail.

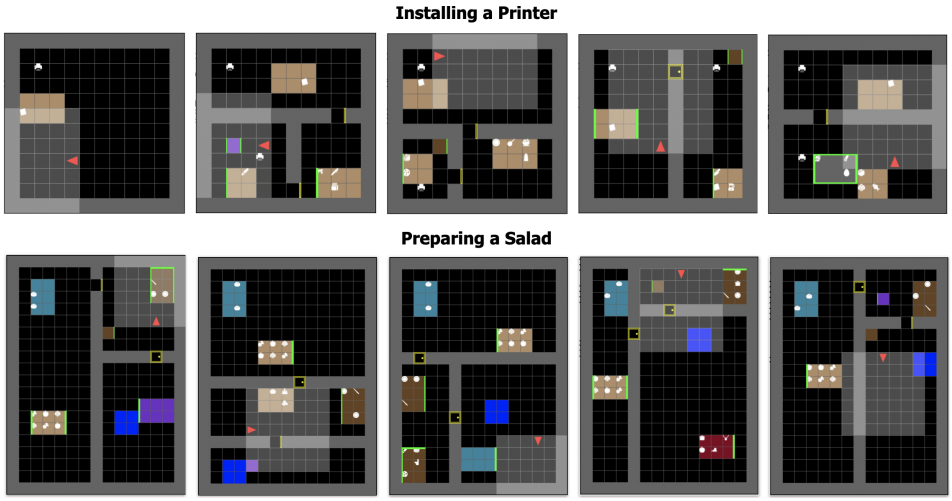


Figure 2: Examples of Procedural Generated Activity Instances: A few procedurally generated grid environments from procedural environment generation for two activities: `Installing a printer` and `Preparing a salad`.

3.1 Diverse and Complex Tasks: For Reasoning and High-Level Planning

In order to ensure that embodied AI agents learn the decision-making skills needed to complete real-world activities, Mini-BEHAVIOR tasks must reflect the wide range of difficulties and skills encountered in the real world.

Mini-BEHAVIOR provides a standardized set of 20 tasks chosen from BEHAVIOR-100 tasks. The chosen tasks represent a realistic distribution of diverse and complex household activities, including cleaning, organizing, and cooking tasks (see appendix A for a full list of activities and descriptions). The tasks are diverse in the sense that the activities involve a variety of household objects and require different states changes, such as moving objects, cleaning objects, and opening furniture. They are complex – not only because they are long-horizon, but also since they require advancing reasoning skills for high-level planning.

Of all of the activities, the most simple one is `installing a printer`, in which the scene is initialized with a printer and table on the floor. To complete this, the agent must move the printer onto the table and toggle it on. This is simple since it involves interacting with only one object and relatively few number of high-level steps.

On the other hand, `washing pots and pans` is a hard activity that requires advanced high-level reasoning skills. The objective is to clean the stained kitchen appliances (teapot, kettle, pans) on the kitchen countertop and place them in the cabinets. To successfully accomplish this activity, the agent must execute a series of steps: navigate to find the relevant objects, pick up an unsoaked scrub brush, turn on the sink, soak the scrub brush in sink water, pick up the soap, clean each appliance using the soaked scrub brush and soap, and finally, place all of the items in the cabinet. This complex task requires the agent to interact with over 10 different objects and perform hundreds of individual actions in the correct sequence. This requires high-level planning and makes it difficult for the agent to complete.

By providing a standardized set of diverse and complex tasks with ranging difficulty, we hope that Mini-BEHAVIOR will facilitate more reasoning and high-level planning capabilities for embodied AI agents.

3.2 Fast Simulation Environment: For Ease of Use and Rapid Prototyping

Mini-BEHAVIOR contains a novel Gridworld environment with the capabilities to support diverse and complex tasks, yet is also designed to be lightweight and have minimal dependencies. The Mini-BEHAVIOR environment is built on top of MiniGrid, and it inherits its design philosophy for easiness of understanding and customization.

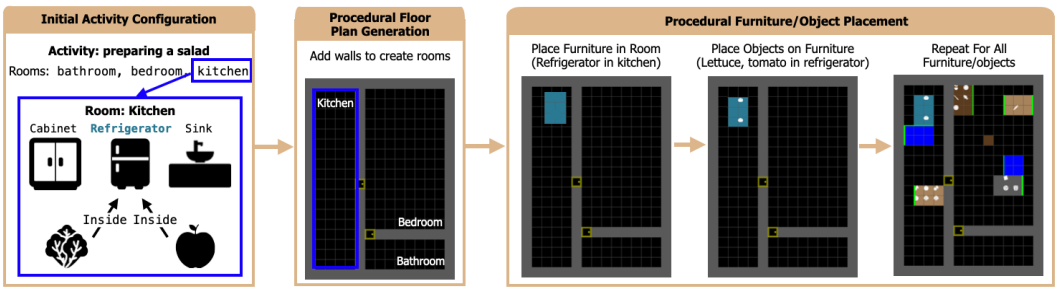


Figure 3: Procedural Generation Flow Chart: We allow procedural environment generation by simply taking in an initial activity configuration, with specifications for the grid environment, rooms, furnitures and objects. First, we generate a grid with a specified width and height, and then we procedurally generate a floor plan by adding walls and doors to create rooms. Then, we procedurally place the furnitures and objects: we add furniture to the specified room in the initial configuration, and then we place objects on the corresponding furniture with the correct symbolic states.

It also has a comparably high simulation speed that makes it suitable for fast prototyping. On a regular laptop computer with Intel i7 eight-core processor at 1.90GHz, Mini-BEHAVIOR (without parallelization) can run at around 600 FPS.

3.3 Procedural Generation: For Open-Ended Learning

In the real-world, agents constantly encounter previously unseen environments, and they must be able to navigate and accomplish tasks under these novel settings. Mini-BEHAVIOR aims to support such open-ended learning by using procedural generation to provide an unlimited number of diverse activity settings. Inspired by ProcTHOR [5], our procedural generation is achieved through procedural floor plan generation and procedural furniture/object placement, as shown in Fig. 3.

Procedural Floor Plan Generation. The floor plan of the Gridworld household environment is determined by the layout and arrangement of rooms, walls, and doors. Mini-BEHAVIOR allows procedural floor plan generation by first generating the surrounding walls of the grid, randomly generating a wall with a door to split the grid into two connected rooms, choosing a room, and then repeating the wall generation process to further split the grid until a desired number of rooms is reached.

Procedural Furniture/Object Placement. The number and types of furniture and objects in the environment are predefined for the provided activities in their configuration files. Alternatively, they can be randomly sampled, where the number of furniture in each room will fall in the range $[1, \max(2, \frac{room_width \times room_height}{12}))$ and the number of objects on each furniture will fall in the range $[1, furniture_width \times furniture_height)$. Given the number and types of furniture in each room defined, the pieces of furniture are randomly placed in the rooms one by one without space overlap. Similarly, with the number and types of objects on each piece of furniture defined, the objects are randomly placed on the corresponding furniture without space overlap. We also introduce a reachability check to make sure the floorplan and placements are valid by checking if all empty tiles are all connected.

This results in a procedurally generated activity instance that aids in open-ended learning, with examples shown in Fig. 2.

4 Mini-BEHAVIOR Environment Design

The Mini-BEHAVIOR environment is designed to balance simplicity and speed while accommodating the complexity required to effectively support a wide range of long-horizon tasks. In addition to incorporating procedural generation, we extend beyond MiniGrid's capabilities to encompass symbolic states, an additional vertical dimension, multi-cell objects, and an abundance of actions, and objects. These enhancements introduce a level of realism essential for tackling complex tasks. Moreover, we incorporate various visual and control modes to enable different learning approaches – such as visual agents and learning from human demonstrations.

4.1 Motivation of Environment Features: Activity Definitions and Symbolic States

To motivate the needs of our environment, it is important to understand how activities are defined and the role of symbolic states. Following BEHAVIOR, we define Mini-BEHAVIOR activities using the Behavior Domain Definition Language (BDDL), which is a predicate-based logic language that allows for infinite activity instances and flexible goal states.

In BDDL, an activity definition consists of a set of objects, initial conditions, and goal conditions. Given an activity, Mini-BEHAVIOR procedurally generates environment instances with objects satisfying the initial conditions, and the agent completes the task by navigating and interacting with the environment until it fulfills all of the goal conditions.

For example, the simplest activity of `Installing a Printer` is defined by the initial conditions `OnTop(printer, floor)`, `not ToggledOn(printer)`, `InRoom(table, office)`, and the task is considered complete when `OnTop(printer, table)` and `ToggledOn(printer)`.

At any point in time, we use these symbolic states to describe the scene based on the physical objects' states (i.e. `ToggledOn`), their relationships to each other (i.e. `OnTop`), and their relationships to the agent (i.e. `InHandOf`). They are essential to our activity definitions and scene representations, so we design our environment to support these states both symbolically and visually (see appendix B for a list of all symbolic states and descriptions).

We introduce three novel features to support symbolic states realistically – a third environment dimension, multi-cell objects, and more comprehensive primitive actions.

4.2 A Novel Gridworld Environment

In order to realistically simulate complex tasks and the physical relationships between objects, Mini-BEHAVIOR is a Gridworld environment that is designed to support vertical relationships between objects and notions of height. Each Mini-BEHAVIOR environment is a 3D world with x-y-z axes and $n \times m \times 3$ cells. The z-axis represents a vertical dimension (top, middle, bottom) and introduces a notion of height and vertical placement to the environment. This is necessary to support more binary symbolic states and relations (i.e. on top, under, inside). Inspired by MiniGrid, each cell can contain at most one object. The limitation of 3 z-axis coordinates is still not entirely reflective of the real world, but choose this limitation to balance simplicity and realism in our environment. It is enough to support all of the BEHAVIOR tasks and, we believe, most real-world scenarios as well.

4.3 Multi-Cell Objects

We also introduce multi-cell furniture objects to reflect notions of size, which is important for physically representing the real world. Mini-BEHAVIOR supports two classes of objects: an object class and a furniture class. To reflect the real world, *objects* are small objects that we often move or pick up in our everyday lives, such as a book or apple. On the other hand, *furniture* are larger and stationary, and they include common household furniture such as a table, bed, and stove. In Mini-BEHAVIOR, the two distinctions between *objects* and *furniture* are that 1) *objects* are $1 \times 1 \times 1$ in size while *furniture* can span multiple cells, and 2) *objects* are movable while *furniture* are not.

4.4 Agent

Action Space. By default, Mini-BEHAVIOR agents use an action space that consists of 15 primitive actions – 3 for navigation (forward, turn left, turn right) and 12 primitive actions for interacting with objects (close, cook, drop (3D), drop-in, open, pickup (3D), slice, toggle). This results in an 15-dimensional discrete action space. Under this setting, the same action space is shared across different tasks, allowing the study of transfer learning, multi-task learning, and meta-learning.

A drawback of the primitive action space is that the agent can only carry one object at a time to prevent action ambiguity, which may cause inefficiency. Therefore, Mini-BEHAVIOR supports an additional type of action space - the Cartesian action space - which allows for the simultaneous carrying of multiple objects. The Cartesian action space consists of the Cartesian product between the primitive actions and available objects for a given task, with invalid actions (e.g. "slice a refrigerator") removed. Such an action space makes carrying and dropping multiple objects possible, and is often better suited for studying how to solve a single task. However, the Cartesian action space for different

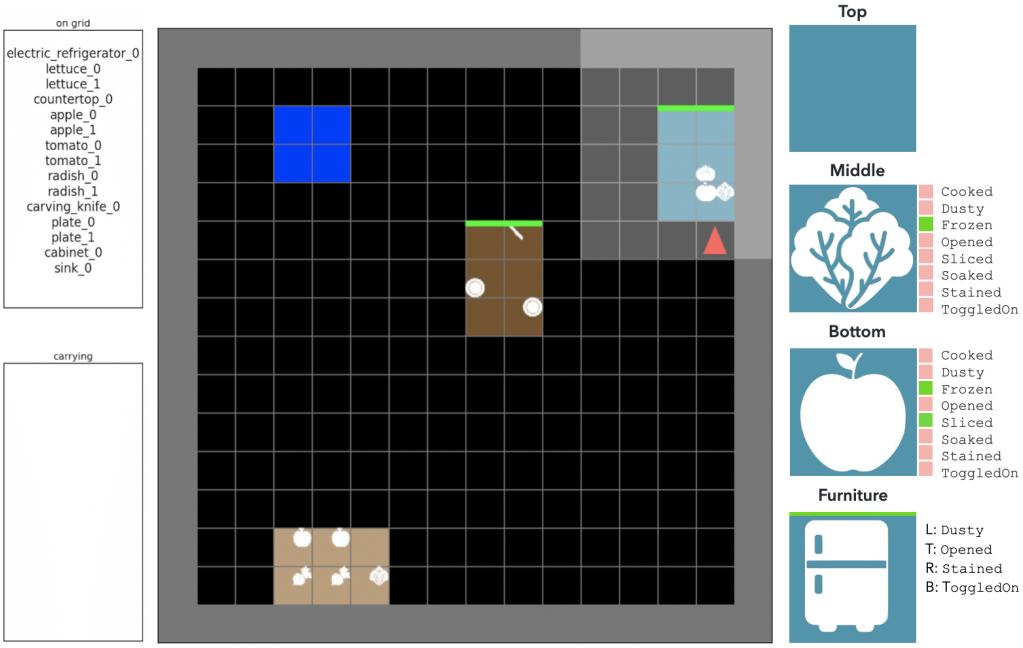


Figure 4: Visualization of the Mini-BEHAVIOR Environment: showing all objects as icons and all furniture as a colored background. All furniture states are shown in the grid with a green edge, while object states are only visualized for the cell directly in front of the agent by a red or green box.

tasks will be different, based on the available objects. For a task where each object i has n_i possible actions, the action dimension of the Cartesian action space is equal to $4 + \sum_i n_i$. This ranges from 5 (for the simplest task of installing a printer) to 54 (for preparing a salad).

Observation Space. Mini-BEHAVIOR supports both fully observable and partially observable environments. In the partially observable case, the observation space of the agent is a $7 \times 7 \times 31$ ego-centric grid; when fully observable, the observation space is a $n \times n \times 31$ global grid, where n is the size of the map. Each 31-dimension pixel encodes the states of the furniture and up to three objects that occupy a given cell on the map.

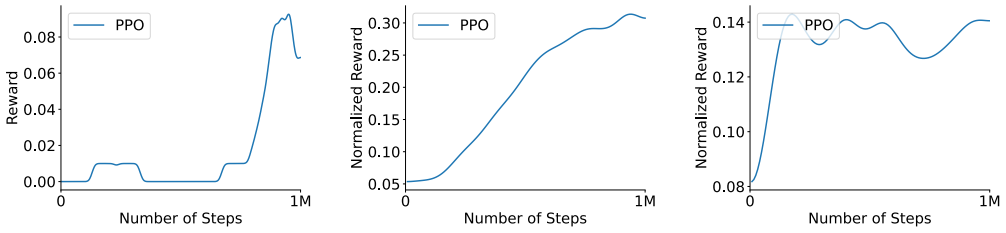
Reward. By default, Mini-BEHAVIOR uses sparse rewards for all tasks, where the agent receives +1 for successfully completing the task, and 0 otherwise. We provide APIs such that the user can easily implement dense rewards for each environment, and toggle it on / off during environment initialization, exemplified in the following two tasks: putting away dishes after cleaning, and washing pots and pans.

4.5 Additional Features to Support Research Needs

Mini-BEHAVIOR is easy to use and customizable to fit the needs of researchers, and we also provide different visualization modes and agent control modes to support a variety of learning methods such as from visual input and human demonstrations.

Customizability. By default, the layout of the environment is a 20×20 grid with walls along the outside. However, the layout is easily customizable. There are parameters which can be used to set the size and create grids of rooms, and we also provide some default layouts that correspond to scene layouts from iGibson, with examples shown in the Appendix. In addition, it is easy to create new tasks, objects, and actions using the classes and modules that we provide.

Simulation Visualization. To support visual agents and manual control of the environment, we provide various ways to visualize the environment as a grid rendering of $n \times m$ cells. The agent is always represented by a red triangle, and walls are represented by gray cells. Objects are represented by a corresponding object icon that is easy to recognize, and states are represented by a colored square. To visualize a 3D environment in a 2D format, we provide different views of the environment



(a) Printer (Sparse) (b) Dishes (Dense) (c) PotWash (Dense)

Figure 5: Preliminary results on the Mini-BEHAVIOR benchmark. We show results of vanilla PPO across three tasks, ranging from easy to hard. While simple and lightweight, Mini-BEHAVIOR is very challenging for cutting-edge decision-making algorithms, especially when a dense reward signal is not available.

to provide a full view of the 3D environment. For example, the default view provides a birds-eye view of all objects in the environment (Fig 4), and a single-dimension view visualizes all objects in a single dimension and their unary states. We provide further discussion in the Appendix.

Control Modes We support two modes: agent control and manual control. In the agent control mode, the agent is controlled by an algorithm, and this is useful for training different RL and planning agents. In manual control, a user can manually navigate and interact with the environment using keyboard controls. This is useful for creating human demonstrations.

5 Example Benchmark Results

Mini-BEHAVIOR provides APIs both for training an agent with reinforcement learning and for collecting demonstration data that can be used for imitation learning. In this section, we provide preliminary results of training agents using proximal policy optimization (PPO) [15] on 3 different tasks – `installing a printer`, `putting away dishes after cleaning`, and `washing pots and pans`. Based on the optimal number of steps to complete these tasks, we classify the difficulty of these tasks as easy, medium, and hard, respectively. For all of these tasks, we use a grid size of 10×10, with a time limit of 1000 steps per episode and a total of 1e6 training steps.

We first examine these tasks under the sparse reward setting, where the agent will only receive a reward of +1 if it successfully completes the task. We found that, under the sparse reward setting, within 1M steps, `installing a printer` (the simplest out of the three) is the only task where the agent can achieve occasional success. We show the reward curve for `installing a printer` in Fig. 5a, and omit the curves for the other two tasks due to them being zero throughout the training.

To gain further insights into the difficulty of the two harder tasks: `putting away dishes after cleaning` and `washing pots and pans`, we show additional results by hand-crafting a dense reward function for each of them. With dense reward, the agent receives a +1 reward every time it makes progress towards achieving the final goal (e.g. for `putting away dishes`, the agent receives a reward for opening the cabinet). We show the reward curves for the dense reward setting in Fig. 5b and Fig. 5c. Notice that we normalize the reward in the plots such that a reward of 1 would correspond to successfully completing the task. Even with the dense reward, it is still difficult for vanilla PPO to learn a good policy for the medium and hard tasks, where the medium-level task agent has made a moderate amount of progress while the hard-level task agent is stuck at a relatively early stage.

These results together show that Mini-BEHAVIOR, while lightweight in terms of dependencies and computation, still provides a considerable amount of challenges necessary to evaluate state-of-the-art algorithms in decision-making and embodied AI.

6 Conclusion

In this paper, we introduce Mini-BEHAVIOR, a novel 3D gridworld simulation environment and benchmark of tasks to advance and support the development of high-level decision-making algorithms

for embodied AI. Mini-BH is built on top of MiniGrid to be simple, lightweight, and fast, and it includes additional features to support simulating a set of 20 long-horizon, human-centered tasks for benchmarking. Mini-BH has already been used in other research projects [19, 8], and we hope that the embodied AI community as a whole will find value in using Mini-BH to prototype and benchmark decision-making algorithms that can ultimately be used to solve tasks in a realistic, physical simulation like those in BEHAVIOR.

References

[1] Chris Bamford, Shengyi Huang, and Simon M. Lucas. Griddly: A platform for AI research in games. *CoRR*, abs/2011.06363, 2020. URL `https://arxiv.org/abs/2011.06363`.

[2] Dhruv Batra, Angel X. Chang, Sonia Chernova, Andrew J. Davison, Jia Deng, Vladlen Koltun, Sergey Levine, Jitendra Malik, Igor Mordatch, Roozbeh Mottaghi, Manolis Savva, and Hao Su. Rearrangement: A challenge for embodied AI. *CoRR*, abs/2011.01975, 2020. URL `https://arxiv.org/abs/2011.01975`.

[3] Micah Carroll, Rohin Shah, Mark K. Ho, Thomas L. Griffiths, Sanjit A. Seshia, Pieter Abbeel, and Anca D. Dragan. On the utility of learning about humans for human-ai coordination. *CoRR*, abs/1910.05789, 2019. URL `http://arxiv.org/abs/1910.05789`.

[4] Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831, 2023.

[5] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation, 2022.

[6] Chuang Gan, Siyuan Zhou, Jeremy Schwartz, Seth Alter, Abhishek Bhandwaldar, Dan Gutfreund, Daniel L. K. Yamins, James J. DiCarlo, Josh H. McDermott, Antonio Torralba, and Joshua B. Tenenbaum. The threedworld transport challenge: A visually guided task-and-motion planning benchmark for physically realistic embodied AI. *CoRR*, abs/2103.14025, 2021. URL `https://arxiv.org/abs/2103.14025`.

[7] Jiaheng Hu, Peter Stone, and Roberto Martín-Martín. Causal policy gradient for whole-body mobile manipulation. In *arXiv preprint arXiv:2305.04866*, 2023.

[8] Andrey Kurenkov, Michael Lingelbach, Tanmay Agarwal, Emily Jin, Chengshu Li, Ruohan Zhang, Li Fei-Fei, Jiajun Wu, Silvio Savarese, and Roberto Martın-Martın. Modeling dynamic environments with scene graph memory. In *International Conference on Machine Learning*, pages 17976–17993. PMLR, 2023.

[9] Chengshu Li, Fei Xia, Roberto Martín-Martín, and Silvio Savarese. Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators. In *CoRL*, pages 603–616. PMLR, 2020.

[10] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey Kurenkov, Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 455–465. PMLR, 08–11 Nov 2022. URL `https://proceedings.mlr.press/v164/li22b.html`.

[11] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, Mona Anvari, Minjune Hwang, Manasi Sharma, Arman Aydin, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Silvio Savarese, Hyowon Gweon, Karen Liu, Jiajun Wu, and Li Fei-Fei. BEHAVIOR-1k: A benchmark for embodied AI with 1,000 everyday activities and realistic simulation. In *6th*

Annual Conference on Robot Learning, 2022. URL https://openreview.net/forum?id=_8DoIe8G3t.

[12] Jan Matas, Stephen James, and Andrew J Davison. Sim-to-real reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 734–743. PMLR, 2018.

[13] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. *CoRR*, abs/1806.07011, 2018. URL http://arxiv.org/abs/1806.07011.

[14] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

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

[16] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. *CoRR*, abs/1912.01734, 2019. URL http://arxiv.org/abs/1912.01734.

[17] Sanjana Srivastava, Chengshu Li, Michael Lingelbach, Roberto Martín-Martín, Fei Xia, Kent Vainio, Zheng Lian, Cem Gokmen, Shyamal Buch, C. Karen Liu, Silvio Savarese, Hyowon Gweon, Jiajun Wu, and Li Fei-Fei. BEHAVIOR: benchmark for everyday household activities in virtual, interactive, and ecological environments. *CoRR*, abs/2108.03332, 2021. URL https://arxiv.org/abs/2108.03332.

[18] Sainbayar Sukhbaatar, Arthur Szlam, Gabriel Synnaeve, Soumith Chintala, and Rob Fergus. Mazebase: A sandbox for learning from games. *CoRR*, abs/1511.07401, 2015. URL http://arxiv.org/abs/1511.07401.

[19] Zizhao Wang, Jiaheng Hu, Peter Stone, and Roberto Martin-Martin. Elden: Exploration via local dependencies. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[20] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *CoRR*, abs/1910.10897, 2019. URL http://arxiv.org/abs/1910.10897.

[21] Kai Zhu and Tao Zhang. Deep reinforcement learning based mobile robot navigation: A review. *Tsinghua Science and Technology*, 26(5):674–691, 2021.

[22] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

A Task Descriptions

Here, we describe the 20 tasks in Mini-BEHAVIOR.

1. **Boxing books up for storage:** There are books and a box on the floor, and the agent must put all of the books in the box.
2. **Cleaning a car:** There is initially a dusty car and unsoaked rag, and the agent must use the rag, soap, and sink to clean the car. After, the agent must place the rag and soap to a bucket.
3. **Cleaning shoes:** There are 2 stained shoes and 2 dusty shoes in the bedroom, and the agent must clean them using a towel and sink in the bathroom.
4. **Cleaning up the kitchen only:** There is a messy kitchen with a dusty floor, dusty pan, dusty cabinet, stained plate, and stained blender. The agent must find the broom, rag, and sink and use them to clean all of the dusty and stained objects.
5. **Collect misplaced items:** There are items such as a gym shoe, necklace, notebook, and sock, misplaced around the living room and dining room. The agent must navigate these rooms, find the items, and place them on the table.
6. **Installing a printer:** There is a printer on the floor, and the agent must place it on the table and toggle it on.
7. **Laying wood floors:** There is are pieces of plywood, a saw, and a hammer on the floor. The agent must lay all of the plywood in the kitchen next to each other.
8. **Making tea:** There is a kitchen with a teapot, teabag, lemon, and knife hidden in various places like the cabinet and refrigerator. The agent must find these items and make tea by slicing a lemon, placing the teapot on the stove, placing the teabag inside of the teapot, and turning on the stove.
9. **Moving boxes to storage:** There are two cartons on the floor of the living room, and the agent must move these to the storage room and stack them on top of one another.
10. **Opening packages:** There are two unopened packages on the floor, and the agent must open both of them.
11. **Organizing file cabinet:** There are various office objects (markers, folders, and documents) that are scattered on different furniture (chair, table, cabinet) in a private office. The agent must find the objects and place the markers on the table and documents and folders in the cabinet.
12. **Preparing salad:** There are different salad ingredients (lettuces, apples, tomatoes, radishes), a plate, and a carving knife in the kitchen, and they are initially on the countertop, in the cabinet, or in the refrigerator. The agent must find these items, use the carving knife to slice the apples and tomatoes, and then place all of the salad ingredients on top of the plate.
13. **Putting away dishes after cleaning:** There are plates on the kitchen countertop, and the agent must place all of them in the kitchen cabinet.
14. **Setting up candles:** There are candles in a box, and the agent must take them out and place them on the table.
15. **Sorting books:** There are books and hardbacks on the table, and the agent must place these on top of each other on the shelf.
16. **Storing food:** There are boxes of oatmeal, chips, bottles of olive oil, and jars of sugar on the countertop. The agent must place all of these food items in the cabinet.
17. **Thawing frozen food:** There are various frozen foods (fish, dates, and olives) in the refrigerator. The agent must take all of them out of the refrigerator and place them in the sink.
18. **Throwing away leftovers:** There are hamburgers on plates, which are on countertops in the kitchen. The agent must throw all of the hamburgers into the trash can.
19. **Washing pots and pans**: There are stained kitchen appliances (teapot, kettle, pans) on the kitchen countertop. The agent must clean the appliances using the scrub brush, soap, and sink and then place them in the cabinet.
20. **Watering houseplants:** There are potted plants, and the agent must water them using the sink in the bathroom.

B Symbolic States

Mini-BEHAVIOR supports 3 types of symbolic states: agent-related, absolute, and relative object states:

- **absolute object state**: a state that depends on a single object (i.e. `Cooked` and `ToggledOn`)
- **agent-related state**: a state that represent relationships between the agent and object (i.e. `InHand` and `InSameRoom`)
- **relative object state**: a state based on the physical relationships between two objects (i.e. `NextTo` and `OnTop`)

This table lists the name, type, and description for all supported symbolic states:

Symbolic State	Type	Description
InFOV	agent	object is in the cell in front of the agent
InHand	agent	object is in the hand of the agent
InReach	agent	object is either in front or in hand of agent
InSameRoom	agent	object is in the same room as the agent
Cooked	absolute	the object is cooked (only food objects)
Dusty	absolute	the object is dusty
Frozen	absolute	the object is frozen (only food objects)
Opened	absolute	the object is open (refregirator, box, etc)
Sliced	absolute	the object is sliced (only food objects)
Soaked	absolute	the object is soaked
Stained	absolute	the object is stained
ToggledOn	absolute	the object is on
OnFloor	absolute	object is on the floor
AtSameLocation	relative	object1 and object2 are in the same cell
Inside	relative	object1 is inside object2
NextTo	relative	object1 and object2 are in adjacent cells
OnTop	relative	object1 is one dimension above object2
Under	relative	object1 is one dimension below object2

C Room Size Customization

Mini-BEHAVIOR allows for customization of environment layout size. In Fig. 6, we show examples of rooms with customized layout, including one which corresponds to the scene layouts from iGibson.

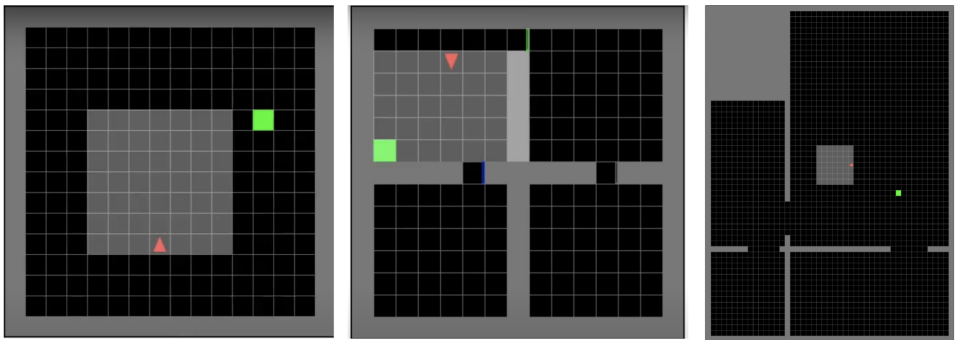


Figure 6: Examples of Mini-BEHAVIOR environment layouts: one 20×20 room (left), four 6×6 rooms (middle), iGibson `Rs_int` scene layout (right)

D Visualizations

D.1 State Visualization

Only unary states are visualized, which means that the possible states are cooked, dusty, frozen, opened, sliced, soaked, stained, and toggledon for *objects* (8 total) and dusty, opened, stained, toggledon for *furniture* (4 total).

To visualize the *object* states, there are 8 squares to the right edge of the object cell to represent the 8 possible states. The square is green if the state is True and red otherwise.

For *furniture* states, note that all furniture is constrained to be rectangular, so we use the 4 edges to visualize the states (left: dusty, top: opened, right: stained, bottom: toggledon). In particular, a green edge indicates that a state is True.

D.2 Scene Visualization

We provide three ways to view the grid, each with its own characteristics and trade-offs.

Default View: By default, the scene visualization is a birds-eye-view of the environment that shows all of the objects in all dimensions (Fig. 4). *Objects* are visualized as a simple yet easily recognizable icon, and are placed in different areas of the cell depending on the vertical dimension that it is in. *Furniture* is rendered as a colored background. In this view, *object* states are not shown, but *furniture* states are. Still, it is possible to view the states of all objects in the cell directly in front of the agent as this includes all of the objects that the agent can interact with at that point. The user is able to open a 'Closeup' of the cell on the right of the grid, which displays 4 squares for the three dimensions and furniture in the cell. For each square, the icon and states are rendered.

Furniture Only: The dimension view (Fig 7, right) is used to visualize a single vertical dimension (top, middle, or bottom), and it shows all objects, furniture, and states in the specified dimension. Similar to the default view, objects are rendered as icons while furniture is a colored background. The key difference is that the object icon takes up the full cell, and the object states are visualized in addition to the furniture states.

Single Dimension: In the furniture-only view (Fig 7, left), the *furniture* icons are displayed, and no *objects* or object states are shown.

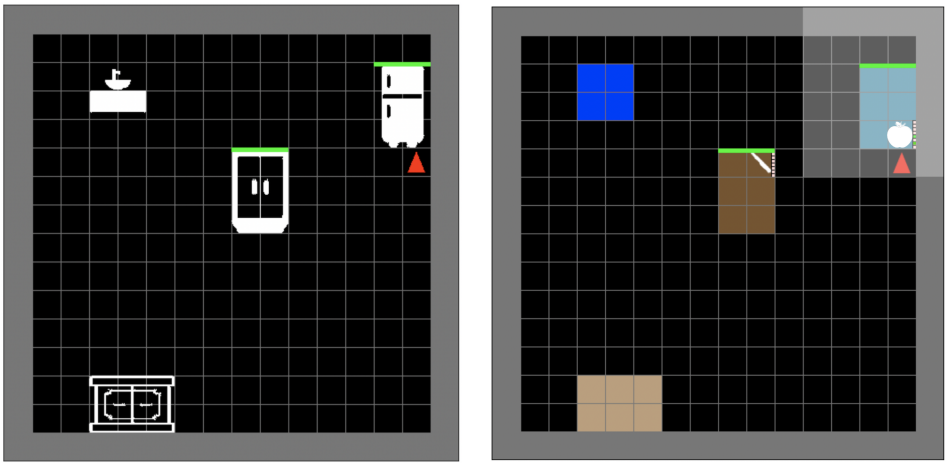


Figure 7: Other views of the grid – (left) Furniture only view only shows furniture and furniture state. (right) Single dimension view of the bottom dimension, which shows all furniture and objects in the bottom dimension with their states. Comparing with the default view (Figure 4), we see that the 'apple' from the closeup corresponds to the cell in front of the agent in the single dimension view