

## A Domain Descriptions

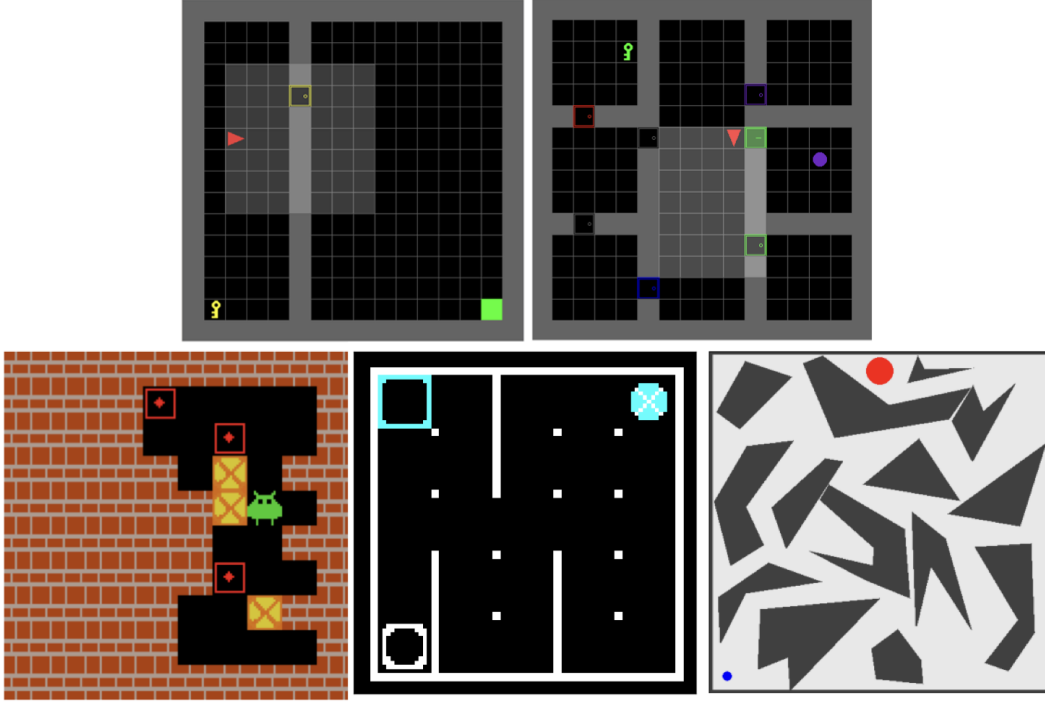


Figure 6: Domains used to test the IM-DSG agent. From top-left in left to right order: MINIGRID-DOORKEY, MINIGRID-KEYCORRIDOR, SOKOBAN, VISUALTAXI, and VISUALPINBALL.

1. **MINIGRID-DOORKEY**: To reach the goal, the agent must first pick up a key and then use that key to unlock a door. There is no intermediate reward for picking up the key, only a sparse terminating reward for reaching the goal. Each episode lasts a maximum of 200 steps.
2. **MINIGRID-KEYCORRIDOR**: This domain also has a key and a locked door, but additionally has other doors that can be open and closed. The goal is to pick up the purple ball that is placed in locked room (center-right room in Figure 6(c)). Each episode lasts a maximum of 1000 steps.
3. **SOKOBAN**: In this classic puzzle, the agent must place 3 boxes into their target locations (red border squares in Figure 6). The environment provides intermediate rewards for putting each box into its place and a terminating reward for correctly placing the final box. Each episode lasts a maximum of 200 steps.
4. **VISUALTAXI**: this is an image-based version of the classic Taxi problem (Dietterich, 2000). A passenger awaits in one of four depots and must be dropped off at a destination depot. Successful completion of the full task yields a sparse terminating reward of +1. Each episode lasts a maximum of 50 steps.
5. **VISUALPINBALL**: image-based version of the under-actuated Pinball domain (Konidaris & Barto, 2009; Bacon et al., 2017) in which the agent provides acceleration to the ball in one of 4 directions (or no-op), causing the velocity of the ball to change over time. We pick the hardest configuration from Konidaris & Barto (2009) during training, and sample goal locations at random during testing. Test-time goals are communicated to the agent as images of the pinball in the desired location. Each episode lasts a maximum of 1000 steps.

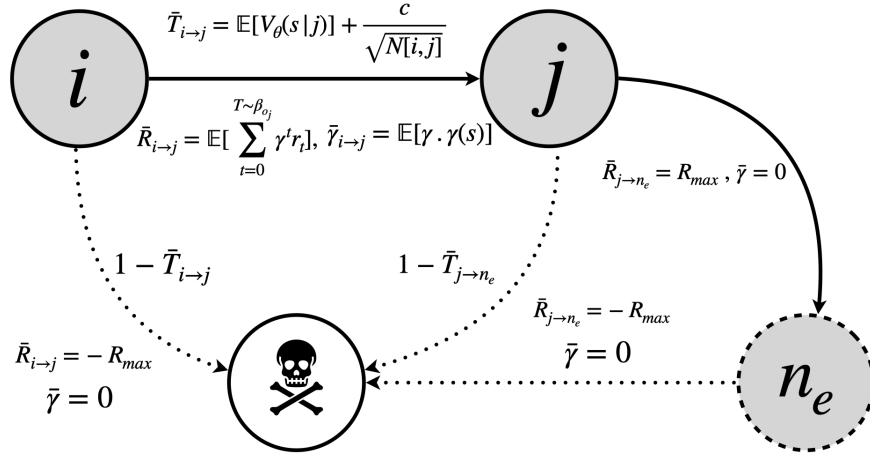


Figure 7: Summary illustration of a toy abstract MDP (AMDP) with 3 nodes.  $i$  and  $j$  are two ordinary nodes,  $n_e$  is the expansion node and the skull node shows a fictitious node that represents “falling off the graph”. The transition probabilities  $\bar{T}_{i \rightarrow j}$  are obtained using the UVFA  $V_\theta$  and edge-traversal counts  $N[i, j]$ ; the reward  $\bar{R}_{i \rightarrow j}$  and discount models  $\bar{\gamma}$  are obtained via monte carlo estimation from option executions.

## B VISUALPINBALL Experiment Details

Both the IM-DSG agent and the Hindsight Experience Replay (HER) agent get the same training budget of 50M frames. During evaluation, both algorithms are tested on the same set of 10k goals and are given a budget of 10M frames to achieve them. Each episode lasts for at most for 1k steps. Error bars are computed over 5 training runs of the respective agents.

To compute the coverage metric shown in Figure 5 (right), we first need to compute the maximum possible coverage in the maze. This is done by first discretizing the  $(x, y)$  plane into  $100 \times 100$  squares, and then subtracting the approximate area occupied by the obstacles in the maze. To compute the coverage achieved by the two agents, we consider what fraction of the free space is occupied by all the states visited by the two agents.

**Test-time modifications for VISUALPINBALL.** During training, IM-DSG picks the expansion node based on  $V_{\text{novelty}}$ . At test-time, this would not be a good strategy because there is no reason that  $V_{\text{novelty}}$  would guide the agent to the test-time goal. So instead, we pick the expansion node to be the one that is closest to the test-time goal  $g_{\text{test}}$ :

$$n_{\text{expansion}}^{\text{test}} = \underset{n}{\operatorname{argmax}} \mathbb{E}_{s \sim n} [V_{g_{\text{test}}}(s)] \quad (9)$$

Once the agent reaches  $n_{\text{expansion}}^{\text{test}}$ , it then executes its goal-conditioned policy  $\pi(s, g_{\text{test}})$  to reach the test-time goal  $g_{\text{test}}$ .

## C AMDP Construction

Figure 7 illustrates a 3 node toy AMDP:  $n_e$  is the expansion node,  $i$  and  $j$  are intermediate nodes and the skull represents the null node  $\varphi$ . Nodes of the graph are abstract states and outgoing edges from a node are available options. The abstract model, which is a combination of  $\bar{T}$ ,  $\bar{R}$ ,  $\bar{\gamma}$ , is estimated using the agent’s goal-conditioned value function  $V_g$  and Monte Carlo estimation respectively.

Hyperparameter	Value	Tuned
Learning rate	<b>0.0001</b> , 0.00001	Yes
Trace length	40	No
Sequence period	20	No
Batch size	32	No
Burn-in length	0	No
Max replay size	500000	No
Target update period	<b>600</b> , 1200	Yes
Discount factor	0.997	No
Number of actors	32	No

Table 1: R2D2 Hyperparameters.

Hyperparameter	Value	Tuned
$\tau$ from Eq 6	0.7, <b>0.8</b> , 0.9	Yes
$c$ from Eq 5	0.1, <b>0.2</b>	Yes
$k$ in Eq 7	<b>1</b> , 2	Yes
$\tau_{\text{template}}$ in Eq 8	0.5	No

Table 2: IM-DSG Specific Hyperparameters.

## D UVFA Training

We represent the agent’s goal-conditioned value function using the following neural network architecture: a convolutional neural network (CNN; Goodfellow et al., 2016) torso encodes the image representing the current state  $s$  into  $\Phi_1(s)$ ; another CNN encodes the goal  $g$  into  $\Phi_2(g)$ , where  $g$  is sampled from an option’s effect set.  $\Phi_1(s)$  is input into an LSTM (Goodfellow et al., 2016); the output of the LSTM,  $y_t$  is concatenated with  $\Phi_2(g)$  and then passed through a multi-layered perceptron (MLP), which outputs the Q-value for each action in the MDP’s action space. The goal encoder is identical to the state encoder, whose exact architecture, along with that of the LSTM and MLP, are taken without modification from Kapturowski et al. (2019). We follow Bagaria & Schaul (2023) and use novelty to pick which 5 achieved goals in a trajectory should be replayed in hindsight.

## E Hyperparameters

Table 1 lists the hyperparameters for R2D2, Table 2 lists those hyperparameter settings that are specific to IM-DSG, and finally, Table 3 lists those IM-DSG hyperparameters that were environment specific. For non-IM-DSG hyperparameters, we first tune them based on the performance of the CFN agent on the MINIGRID-KEYCORRIDOR problem. Then, the same CFN and R2D2 hyperparameters are used in the IM-DSG algorithm, and other domains, without modification.

The descendant threshold for VISUALTAXI is higher than 0 because the passenger’s destination is part of the state description. The threshold filters the descendant set  $\mathcal{D}(s_t)$  to only include nodes that match the current passenger destination, ensuring expansion candidates are contextually relevant to the current task.

Hyperparameter	DoorKey	KeyCorridor	VisualTaxi	VisualPinball	Sokoban
Option Horizon $H$	200	400	50	200	50
Descendant threshold	0	0	0.3	0	0

Table 3: IM-DSG Hyperparameter settings for different domains