A Environments

Our experiments utilize the MiniGrid [Chevalier-Boisvert et al., 2018] set of environments. These environments are partially observable, procedurally generated, and provide a sparse reward upon successful completion of the task.

The agent receives a 7×7 egocentric view of its surroundings, which is encoded in a $7 \times 7 \times 3$ observation to describe the contents of the visible grid cells. The agent cannot see behind walls or closed doors.

At every step, the agent has seven available actions: turn left, turn right, move forward, pick up, drop, toggle (used to open doors), and done.

A new configuration of the environment is generated for each episode. Episodes finish when the agent has interacted with the environment for the maximum number of steps permitted, or if the agent successfully completes the task, in which case the agent also receives a non-zero reward depending on the number of steps taken to complete the task.

MiniGrid-KeyCorridorS3R3-v0 The agent has to pick up a ball behind a locked door. To unlock the door, the agent must first find the key, which is also hidden.

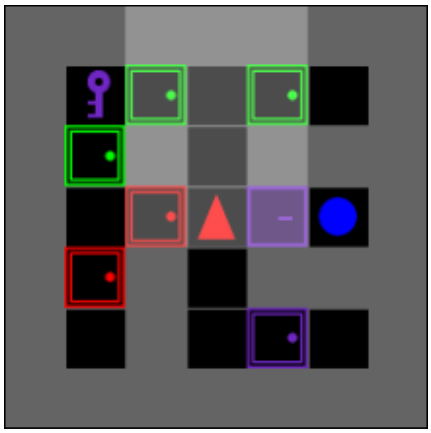

Figure 5: The KeyCorridor-S3R3 environment

MiniGrid-ObstructedMaze-2Dl-v0 The goal of the agent is to pick up an object behind one of the locked doors.

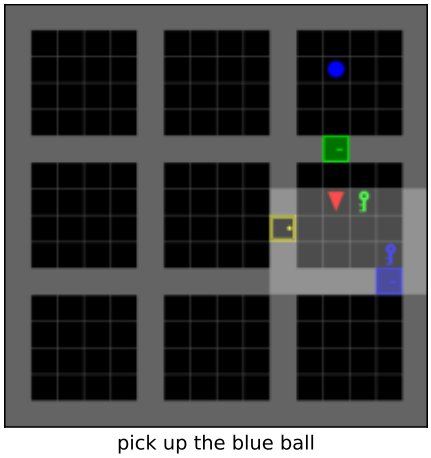

Figure 6: The ObstructedMaze-2Dl environment

MiniGrid-MultiRoom-N10-S4-v0 and MiniGrid-MultiRoom-N7-S8-v0 The goal of the agent is to reach the green goal state in the farthest room (see Figure 7). The notation N10-S4 signifies that the environment consists of 10 rooms with a maximum size of 4. These environments are not available in the default set of MiniGrid environments. We borrow the implementation of these environments from the previous work of Raileanu and Rocktäschel [2020].

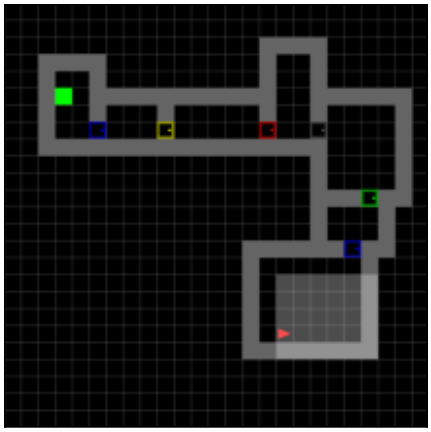

Figure 7: The MultiRoom-N7-S8 environment

B Implementation details

B.1 Base agent

We use the PPO implementation from the `rl-starter-files` [Willems, 2017] repository. We retain the same architecture for the actor and critic as provided in their implementation.

The actor and the critic share three convolution layers with 16, 32 and 64 output channels, with 2×2 kernels and a stride of 1. Each convolution layer is followed by a ReLU non-linearity. A 2×2 max-pooling is applied after the first convolution layer. The output from the convolution layers is flattened and provided to an LSTM with a hidden dimension of 64. Separate multi-layer perceptron (MLP) heads for the actor and critic operate on the output of the LSTM to output the action logits and the value. Each MLP has 64 hidden units with a hyperbolic tangent (tanh) activation function.

B.2 Pseudo-reward generator

The pseudo-reward generator is implemented as a convolutional neural network whose output is flattened to provide a vector of pseudo-rewards $z_{t+1} \in \mathbb{R}^d$. It has a similar architecture to the convolutional layers used in the base agent.

Concretely, this network consists of three convolution layers, with 16, 32, and d output channels respectively, with 2×2 kernels and a stride of 1. Here, d is the number of pseudo-rewards, which we set as $d = 128$ for our experiments. ReLU non-linearity is applied after the first two convolution layers. Additionally, a 2×2 max-pooling is applied after the first convolution layer.

B.3 Predictor network

In the case of RND, the predictor network has the same architecture as the pseudo-reward generator.

For RC-GVF, the predictor is recurrent. Specifically, it consists of three convolution layers with 16, 32 and 64 output channels, with 2×2 kernels and a stride of 1. Each convolution layer is followed by a ReLU non-linearity. A 2×2 max-pooling is applied after the first convolution layer. The output from the convolution layers is flattened and provided to a fully connected layer which provides a 32 dimensional input to an LSTM (with a hidden dimension of 32). A final linear layer maps the output of the LSTM to the pseudo-return predictions.

C Grid search and hyper-parameters

Table 1 provides the common hyperparameter settings used for the PPO base agent across all environments.

Table 1: Hyperparameters for PPO.

Hyperparameter	Value
Learning rate η	0.0002
Discount factor	0.99
PPO clip ϵ	0.2
Rollout length	128
Number of parallel environments	16
Number of epochs	4
Batch size	256
Generalized Advantage Estimation λ	0.95
Entropy coefficient	0
Value loss coefficient	0.5
Recurrence	4 (for truncated BPTT[])
Optimizer	Adam [Kingma and Ba, 2015]
Optimizer epsilon	$1e-08$

Utilizing an intrinsic reward based on RND/RC-GVF introduces additional hyperparameters. Two of them are the intrinsic reward coefficient β and the predictor's learning rate η_p. These parameters were chosen separately for each environment and approach. The values considered for the intrinsic coefficient were, $\beta \in \{0.0001, 0.0005, 0.001, 0.005, 0.1\}$. We considered two possible values for the learning rate of the predictor $\eta_p \in \{\eta, 2.5\eta\}$, where η is the learning rate of the PPO base agent. The definitive hyperparameters were selected based on a grid search over 3 seeds, and we report the final results of a further 10 runs for the chosen setting. We provide the resulting hyperparameters for each environment below

KeyCorridor-S3R3: RND ($\beta = 0.005$ and $\eta_p = 2.5\eta$) and RC-GVF ($\beta = 0.0005$ and $\eta_p = 2.5\eta$),
ObstructedMaze-2Dl: RND ($\beta = 0.001$ and $\eta_p = \eta$) and RC-GVF ($\beta = 0.0001$ and $\eta_p = \eta$),
MultiRoom-N10-S4: RND ($\beta = 0.005$ and $\eta_p = 2.5\eta$) and RC-GVF ($\beta = 0.0001$ and $\eta_p = 2.5\eta$),
MultiRoom-N7-S8: RND ($\beta = 0.0005$ and $\eta_p = 2.5\eta$) and RC-GVF ($\beta = 0.0001$ and $\eta_p = 2.5\eta$).

For all environments, we set the number of pseudo-rewards $d = 128$, the discount factor for the RC-GVF pseudo-returns $\gamma_z = 0.75$ and the associated λ-return parameter $\lambda_z = 0.95$.

In the analysis of different discount factors for RC-GVF (see Figure 3), we report the best results from a a grid search over the intrinsic reward coefficient $\beta \in \{0.0005, 0.001, 0.005\}$ for each discount factor $\gamma_z \in \{0, 0.007, 0.07, 0.25, 0.5, 0.7, 0.75\}$.

In the additional experiment with the entropy coefficient on the KeyCorridor-S3R3 environment (see Figure 4), we utilize an entropy coefficient of 0.0001 and use the same grid search as the main experiments to identify the best hyperparameters. For RND we select $\beta = 0.001$ and $\eta_p = 2.5\eta$, and for RC-GVF we select $\beta = 0.0001$ and $\eta_p = 2.5\eta$.