FAST EXPLORATION AND LEARNING OF LATENT GRAPHS WITH ALIASED OBSERVATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of quickly recovering the structure of a latent graph by navigating in it, when the agent can only perform stochastic actions and — crucially— different nodes may emit the same observation. This corresponds to learning the transition function of a partially observable Markov decision process (POMDP) in which observations are deterministic. This is highly relevant for partially observed reinforcement learning, where the agent needs to swiftly learn how to navigate new environments from sensory observations. The challenge involves solving two related problems: exploring the graph as fast as possible, and learning it from the obtained aliased observations, where the learning helps to explore faster. Our approach leverages a recently proposed model, the Clone Structured Cognitive Graph (CSCG), which can handle aliasing, and guide exploration. We provide empirical evidence that our model-based algorithm can recover graphs from a wide range of challenging topologies, and shows linear scaling with graph size even for severely aliased and loopy graph structures where model-free methods require an exponential number of steps.

1 INTRODUCTION

Graphs serve as a general representation for relational modeling (Battaglia et al., 2018). In this role, they provide a good substrate for Markov decision processes (MDPs) (Guestrin et al., 2011), with graph nodes corresponding to MDP states and graph edges representing the action-driven transitions between them. In addition to representing a generative model of the environment that is useful for downstream tasks such as planning (towards a variety of different goals), having such an explicit correspondence helps ground the model in the environment, and also adapt it to changes as the environment evolves.

The representational suitability of graphs extends from MDPs to the partially observable case (POMDPs) where distinct nodes in the graph may be perceptually aliased to the same observations. Perceptual aliasing is a key property of many interesting problems; examples abound in a variety of domains (Rosvall & Bergstrom, 2008; Xu et al., 2016; Lambiotte et al., 2019; George et al., 2021) from reinforcement learning (Whitehead & Ballard, 1991; Chrisman, 1992; Shani & Brafman, 2004) to simultaneous localization and mapping (Lajoie et al., 2019).

To build high-fidelity models (especially large graphs) with limited sample complexity, in settings where interactive access to the environment becomes a bottleneck, it is crucial that exploration be directed and efficient. Dechter et al. (2013) proposed an active exploration strategy: the Explore-Compress framework, where an agent alternates between building a model of the environment it has already explored, and using that to plan further exploration; we follow a similar approach.

Such an active exploration strategy requires a model-building approach powerful enough to resolve aliased nodes in a graph; while challenging, there exist methods (Xu et al., 2016; Lambiotte et al., 2019; George et al., 2021) which exploit the higher-order structure in the sequences to accomplish the same. In what follows, we choose to work with Clone-Structured Cognitive Graphs (CSCGs) introduced in George et al. (2021) because they allow for planning exploration through inference. However our focus in this work is on active exploration, and the proposed strategy might be combined with other model-building approaches. Previous work using the CSCG model was restricted to learning relatively small graphs (on the order of a hundred nodes or fewer), and used on a random

exploration of the environment. As we demonstrate in Section 4, random exploration scales poorly to larger and more intricately structured environments, and we provide a better exploration strategy.

In what follows, we elaborate on the problem setting and our proposal in the next section. In Section 3 we contextualize our approach against related work. Following that, we evaluate the performance of our proposal on a variety of challenging environments in Section 4, and conclude in Section 5 with some closing thoughts.

2 THE PROBLEM SETUP, AND OUR PROPOSAL

2.1 PROBLEM SETUP

Consider an environment $\mathcal{E}$ characterized by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \Omega, O)$ where $\mathcal{S}$ represents the finite set of states, $\mathcal{A}$ represents the finite set of possible actions at each state (which we will assume to be the same, without loss of generality), $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ represents the set of possible transitions, Ω represents the finite set of possible observations (also referred to as emissions), $O : \mathcal{S} \to \Omega$ represents the emission function mapping states to observations. We further assume O to be deterministic, such that each state produces a unique observation, and are interested in the "perceptually aliased" case where $O : \mathcal{S} \to \Omega$ might be many-to-one. The case of stochastic transition dynamics is described by a probability distribution $p_{\mathcal{T}}(s'|s, a)$ on the next state s', given action a from state s. The agent starts with observation x_1 (at its starting position), then on performing action a_i receives observation x_{i+1}, until it has performed the sequence of actions $a_1 \dots a_{N-1}$ (with discrete $a_i \in \mathcal{A}\}$, and received observations $x_1 \dots x_N$ (with discrete $x_i \in \Omega$). The goal is to recover the set of states $\mathcal{S}$ and the action-conditioned transitions $\mathcal{T}$ (which respectively form the nodes and edges[1] of the graph $\mathcal{G}$ representing $\mathcal{E}$), from such a sequence of actions and observations.

Note that much of the information characterizing this problem is similar to partially observable Markov decision processes (POMDPs) except it lacks any reward function; the goal here is to efficiently explore the environment and build a graph describing it.

2.2 MODEL LEARNING

We use CSCGs (George et al., 2021) to build models of perceptually aliased environments. A CSCG corresponds to a hidden Markov model (HMM) (Rabiner, 1989) with the emission matrix constrained such that each state corresponds to a unique observation, but each observation x may be emitted by several states $\mathcal{C}(x)$ – referred to as "clones" of x.

The conditional probability distribution for a sequence of observations $x_1, \dots, x_N$ given actions $a_1 \dots a_{N-1}$ is described by the following factorized formulation

$$P(x_1, \dots, x_N \mid a_1, \dots, a_{N-1}) = \sum_{\{z_n \in \mathcal{C}(x_n)\}_{n=1}^{N}} P(z_1) \prod_{n=1}^{N-1} P(z_{n+1}|z_n, a_n).$$

As an action-conditional HMM with a frozen emission matrix O, we can use the expectation-maximization (EM) algorithm to learn the action-conditioned transition matrix $\mathcal{T}$, where $\mathcal{T}_{ijk} = P(z_{n+1} = k|z_n = j, a_n = i)$. Refer to George et al. (2021) for the mathematical details. We assign a pseudocount to all possible transitions in the model, as smoothing to enable the model to generalize more robustly to transitions not seen in the training data. This is particularly important in cases where the agent has explored only a part of the environment, or when the dynamics are stochastic – requiring the model to generalize beyond the specific realizations of noisy transitions seen in the training data.

We restrict ourselves to the EM algorithm, since our focus here is primarily on model-driven exploration. However, the proposed exploration strategy is easily amenable to other learning algorithms for the same sequence model, and particularly suited to online variants such as online EM, or *Memorize-Generalize* introduced in Rikhye et al. (2019).

[1]Consistency demands that the probabilities associated with all the outgoing graph edges at a certain node, corresponding to a certain action, sum to one.

2.3 PLANNING AND EXPLORATION

The default (passive) approach to exploration involves the agent performing a random sequence of actions and building a model based on the received observations. We elaborate below our proposal for efficient (active) exploration (abbreviated as "eFeX") using the learned model.

In line with the compression phase in the Explore-Compress framework, we learn a model for the sequence of observations, as described in the previous section. Once EM has converged, we use the Viterbi algorithm (Viterbi, 1967; Forney et al., 2017) to get the maximum a posteriori assignment of latent states $s_1 \dots s_N$ corresponding to the aforementioned sequence of actions and observations.

The agent now needs to determine a state to explore next (which we term the "frontier state") and the best action to execute at that state (which we term the "frontier action"). In what follows, we explain how the agent plans for these.

The decoded sequence of states enables the agent to assign counts to action state pairs, constructing C_{ai}. We describe below how these counts may serve as a functional notion of exploration utility; the agent can actively seek to explore state-action pairs which have not previously been explored. A measure of utility for states reachable from the current state s_N is described by the number of unexplored actions, computed as follows

$$U_{\{i|i \text{ reachable from } N\}} = |\mathcal{A}| - \sum_{a|C_{ai}>0} 1.$$

The simplest measure of reachability for state s_i from s_N is the existence of a finite length path going from the latter to the former. It might often be advantageous (especially in stochastic settings) to choose not just the shortest length path, but the path with the highest likelihood of success (or the shortest path subject to a threshold probability p_{th} for successfully taking the agent from s_N to s_i).

With the above scheme describing the choice of path from s_N to s_i, let D_{Ni} parametrize the length of the path, defined by the number of actions to reach s_i (infinite if not reachable). Then, the most easily accessible state among those with optimal exploration utility would be given by

$$s_f = \underset{\{i|U_i=\max_i U_i\}}{\arg\min} D_{Ni}.$$

For this frontier state, the agent also chooses the least explored action as the frontier action

$$a_f = \arg\min_a C_{a\, s_f}.$$

In case of ties, the $\arg\min$ chooses one of the tied choices randomly. The agent then executes the planned path to the frontier node in a closed-loop manner, estimating its location on the graph after each step, and re-planning a path to the frontier. On reaching the frontier, the agent executes the frontier action, and then performs a random walk. Failing to reach the frontier in twice the number of steps as originally planned (due to a combination of stochasticity in the environment, and a partially learned model), the agent aborts and performs a round of random exploration. This whole process is iterated till the exploration budget is exhausted.

The overall algorithm, including the model learning and active exploration, is summarized in Algorithm 1. The main parameter choice in this algorithm corresponds to the length of (short) random walks L after which the agent will re-plan for a frontier. The optimal choice for this parameter will depend on the problem setting. The longer the agent walks before building models, the better its chance of disambiguating aliased nodes, traded off against shorter walks helping it target unexplored nodes & actions more often. Depending on the needs of the application, this parameter could also be tuned adaptively if necessary. In graphs where every node has a unique emission, even $L = 1$ would be sufficient.

In addition, the algorithm uses two ancilliary parameters: the probability threshold p_{th} for planning paths to the frontier, and the pseudocount for the transition matrix learning.

3 RELATED WORK

The question of exploratory coverage on graphs has a colorful history, going back to the "white screen problem", where Wilf (1989) observed that an animation coloring pixels on a screen based

Algorithm 1 Effective model-based exploration of aliased graphs (eFeX)

Initialize: s ▷ *Starting state*
Initialize: $D \leftarrow []$ ▷ *Experience buffer*
repeat
  for i in 1 . . . L **do** ▷ *Random Exploration*
    $a_i \sim \text{Categorical}(\mathcal{A})$
    $s \sim p_{\mathcal{T}}(s'|s, a_i)$ ▷ *Take one step*
    $x_i \leftarrow E(s)$ ▷ *Receive observation after action*
    $D_i \leftarrow [D_{i-1}, (x_i, a_i)]$ ▷ *Append to experience*
  Estimate $\mathcal{S}, \mathcal{T}$ from D by performing Expectation-Maximization
  Decode experience D into a sequence of latent model states $s_1 \dots s_\tau$
  Populate C_{as} with counts for (action, state) pairs from $a_1 \dots a_{\tau-1}$ and $s_1 \dots s_N$
  $a_1 \dots a_M \leftarrow$ Planned action sequence to reach frontier state, followed by frontier action
  repeat ▷ *Directed Exploration*
    $s \sim p_{\mathcal{T}}(s'|s, a_1)$ ▷ *Take one step*
    $x_i \leftarrow E(s)$ ▷ *Receive observation after action*
    $D_i \leftarrow [D_{i-1}, (x_i, a_i)]$ ▷ *Append to experience*
    Decode experience D into a sequence of latent model states $s_1 \dots s_N$
    $a_1 \dots a_M \leftarrow$ Planned action sequence to reach frontier state, followed by frontier action
  until Executed frontier action at frontier state, or up to $2M$ steps of directed exploration
until exploration budget exhausted

on the coverage of a random walk on the underlying grid tends to cover most of the screen area quite quickly, but takes a long time to cover the few uncolored pixels remaining at the end. While there is a rich and substantive literature studying the cover times of random walks on graphs, classical graph exploration approaches (Motwani & Raghavan, 1995) tend to typically assume distinctly identifiable nodes (i.e. full observability), or a mechanism for the agent to mark nodes as visited (thereby externalizing cognition). Greedy exploration and graph learning approaches which work in that setting (with depth-first search being a classic example) perform poorly in the aliased setting which requires subsequent (future) observations to disambiguate an agent's location in the graph structure at any given time.

Research on latent graph learning (see Dong et al. (2019) and references therein) tends to focus on discovering edges among given nodes, given a signal defined on each node; this is also the setting most obvious with graph neural networks. We are interested in a different setting where the signal is a sequence of observations corresponding to one node per exploration step, from which we seek to construct the graph.

While Dai et al. (2019) propose an interesting graph exploration approach, it avoids the challenge of actually learning the graph structure, instead being provided access to the *true* graph connectivity between visited locations.

As hinted by the connection with POMDPs, the setting we consider is quite relevant in reinforcement learning (RL). However, RL algorithms usually operate with a task-specific reward which the agent is learning to exploit, while also trying to explore actively (Thrun, 1992; Kearns & Singh, 2002), which leads to the exploration-exploitation trade-off (Sutton & Barto, 1998). One way to avoid this is to focus purely on exploration[2] driven by "intrinsic curiosity" (Schmidhuber, 1991; Oudeyer & Kaplan, 2007) or "novelty" (Lehman & Stanley, 2011). Our approach can be interpreted along these lines, since our algorithm proposes frontier states based on unexplored actions. As emphasized in Groth et al. (2021), even after thorough exploration, curiosity-driven RL approaches might still be plagued by the problem of forgetting previously learned regions of state space. Learning the graph, as we do, ensures that discovered states remain accessible as part of the model. Further, with the ability to efficiently plan a path to the frontier state by performing inference using the learned graph, we also avoid thorny issues of sparse rewards when the potential frontier states for exploration form a set of low measure.

[2]Refer Amin et al. (2021) for a review of exploration approaches in reinforcement learning, including particularly the pure exploration setting.

More specifically, our proposal is closely related to count-based exploration approaches in RL (Strehl & Littman, 2005; 2008; Kolter & Ng, 2009). Given our focus on learning graphs, our experiments are situated in the discrete setting. However, as demonstrated in Bellemare et al. (2016); Tang et al. (2017), count-based approaches generalize nicely to high-dimensional and/or continuous RL benchmarks.

Epistemic partially observable MDPs, introduced in Ghosh et al. (2021) provide an example of how perceptual aliasing might manifest itself in the RL setting. As they explain, instead of sticking to a fixed policy, an agent should ideally use any received test-time feedback to disambiguate between states that were a priori perceived as same, leveraging that to update its response strategy.

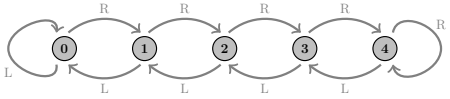
With their emphasis on ideas from classical planning (the notion of a frontier, and expanding a node with exploration, etc.), Ecoffet et al. (2021) propose the Go-Explore algorithm, which is close in spirit to our approach. While it is interesting to note that Go-Explore was able to perform much better than other RL algorithms, their setting has almost no perceptual aliasing – even in Atari games where the viewport only sees a part of the world, the actions that need to be taken need not vary drastically depending on the unobserved latent variables (in contrast with the aliased setting).

While Sharma et al. (2021) also consider the problem of efficiently mapping a novel environment, exploiting the structure of repeating map fragments which can be used to inductively compose a larger map. We focus on the more general problem of exploration without assuming that the same fragments recur, thereby being able to handle unstructured environments.

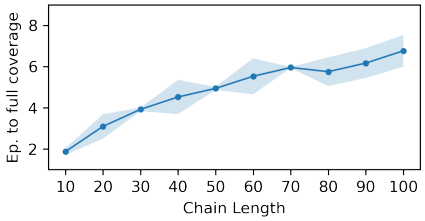
4 Experiments

4.1 Exploring "River Swim" inspired one-dimensional chain

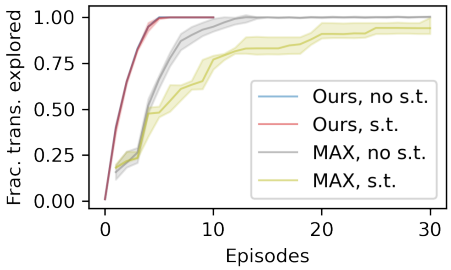
We consider the same one-dimensional chain environment studied in Shyam et al. (2019), and posed originally in Osband et al. (2016) (as a simplified version of the "River Swim" problem posed in Strehl & Littman (2008)). The environment consists of a chain of L uniquely-identifiable states $\{0 \dots L-1\}$. Each state has two actions – respectively linking to the state on either side – except for the two end states which each have one action ("self-loop") keeping the agent in the same state. The correspondence between the two outgoing links $\{\text{left, right}\}$ and the two local actions available to the agent are assigned randomly at each state. The exploration is broken into episodes of length $L+9$, at the end of each of which the agent position is reset to the second state from the left end (i.e. state 1). A variant of the problem incorporates a "stochastic trap" at state 0, such that both actions at state 0 have stochastic consequences, each transitioning to state 0 or 1 with 0.5 probability. Figure 1a illustrates the environment for the $L = 5$ case, without a stochastic trap.



(a) Chain env. with 5 states



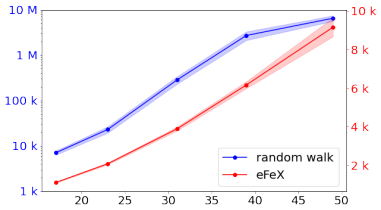
(b) Episodes to full coverage, with increasing chain length, for our approach



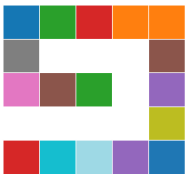
(c) Frac. transitions explored, for $L = 50$. Note that the red & blue curves overlap i.e. the stochastic trap barely affects the performance of our method.

Figure 1: 1D chain experiment (shaded bands span 25th to 75th percentile, estimated from five runs)

Proactive model-based exploration is crucial in this setting, as random exploration will explore states and transitions very slowly. Resets at the start of each episode make the setting more challenging,



(a) Scaling with env. size of steps to complete exploration. *Note the different scales on the respective Y-axes*



(b) Example graph with 17 states

(c) Recovered graph

Figure 2: Comparing the scaling behavior of random walks against our model-based exploration approach. Sub-figures (b) and (c) show an example aliased environment and its recovered graph structure.

as the agent has to use the learned model to make a beeline for the exploration frontier, to manage sufficient exploration in an episode before its position is reset again.

This problem therefore serves as a great test of our proposed active exploration approach. Specifically, in this example, we re-train the model after each step, and plan for the frontier. With the same setup, Shyam et al. (2019) demonstrated that their method MAX significantly outperformed several prior exploration proposals in RL, needing only eighteen episodes (including the three episodes of "warmup time") for exploring all the transitions in a 50-state chain. Figure 1c demonstrates that our approach performs exploration much faster than even MAX, rapidly exploring all the transitions in only five episodes. As shown in the same figure, our method can also handle the "stochastic trap" at the left-most node with hardly any degradation[3] in performance.

Figures 6a & 6b in the appendix show example traces for the two cases. Apart from getting stuck at the stochastic trap for a few extra steps (due to the impossibility of planning a way out when the transitions are completely stochastic), the algorithm is able to perform exploration just fine. In both cases, the algorithm clearly appears to be following what we (as humans) would expect to be the sensible exploration strategy; after the reset at the start of each episode, it aims straight for the exploration frontier and continues systematically performing local exploration around there, repeating this process until all transitions have been explored. As shown in Figure 1b, the exploration time for our method also scales much better with increasing chain length compared with MAX.

4.2 Scaling properties of model-based exploration

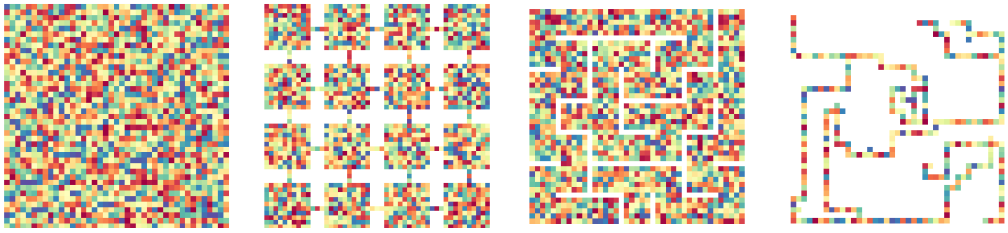
Here we study the scaling behavior of our model-based exploration algorithm contrasted with a model-free alternative in the random walk. We consider intricately structured maze-like graphs of various sizes, where a small fraction (10% in this case) of transitions where the agent hits the wall (and would have stayed in place) are modified into non-local jumps taking the agent back to its starting location. The loopy topology resulting from such long range transitions is a challenging setting reminiscent of one-way exits, or the video game setting where the agent encounters some harm and has to start over.

To explore this environment efficiently, it is crucial that an algorithm judiciously avoid the non-local jumps after they have been explored some minimal number of times (ideally once). Figure 2a shows the mean number of steps to complete full exploration of the environment, with the shaded bands representing one standard deviation for estimates of the mean. This experiment shows that the time taken by our algorithm to explore all transitions scales linearly with the number of states, whereas random walks take exponentially long in the same setting[4].

Figure 2c shows an example graph learned by our algorithm. We have drawn a graph with the discovered states (along with their respective emissions) registered to the locations they represent on a

[3] In fact, episodes prove to be too coarse a measure of sample complexity to distinguish performance degradation from the stochastic trap, for our approach. The retardation is usually only by a few steps, and the exploration curves for both cases therefore largely overlap.

[4] Note the logarithmic Y-axis for the random walk, compared with the linear Y-axis for eFeX.



(a) 2d grid (b) Inter-Connected Grid (c) Dense maze (d) Sparse maze

Figure 3: Exploration environments (3d room not visualized here). Colored locations are accessible, and white regions are inaccessible. The colors at each location represent the (categorical) observation the agent receives when there, including the pattern of aliasing.

map of the room (as seen in Figure 2b). We see clearly that the algorithm has managed to successfully discover the latent graph modeling the states and transitions in the environment – including the non-local jumps.

4.3 Exploring mazes and higher-dimensional rooms

Following the one-dimensional chain environment, we evaluate our proposal on more challenging higher-dimensional room and maze environments, visualized in Figure 3 as two dimensional images. The figure for each room shows the accessible (colored) and inaccessible (white) regions. The colors at each location indicate an example pattern of perceptual aliasing, representing the (categorical) observation received by the agent when at that location.

The transition matrix is determined from the layout, such that each cell is connected to its immediate neighbors on the lattice (six for the three-dimensional room, and four for the other rooms) on the respective lattices. On "hitting a wall" (encountering an inaccessible state), the agent stays at the same location. The actions corresponding to the respective transitions are shuffled at each site, preventing persistent walks unless the agent builds a model of the environment. The environment has stochastic walk dynamics (of a local form). To perform successful active exploration in this setting, the agent needs to be able to navigate to frontier nodes under stochastic walk dynamics – by calibrating its actual location based on (possibly aliased) observations, and then re-planning a path to the desired frontier node. For each room, we consider examples with three different levels of perceptual aliasing. Section A.1 of the Appendix elaborates on details about the room layouts, aliasing and stochasticity.

To study the performance of our proposed method, we use $L = 210$, $p_{\text{th}} = 0.8$ and a pseudocount of 2×10^{-4} for all the experiments, allowing us to compare plots across different amounts of aliasing, even though the optimal values of L might vary with environmental characteristics such as the amount of perceptual aliasing or stochasticity in actions.

To understand the efficiency of exploration, we measure the scaling of the fraction of explored environment transitions with the number of steps of the exploratory walk, graphed in Figure 4, with a 95% confidence interval on the estimate of the mean. To emphasize a different perspective on the same measure, Table 1 presents the mean number of steps needed to explore all the transitions in the environment (along with standard deviations of the underlying distributions). Note that the random walk is not just slower in exploration, but its time to completion tends to also have a very high variance.

We start with the relatively "open" environments corresponding respectively to three-dimensional and two-dimensional rooms. The rapidly increasing transition coverage (first two columns of Figure 4) for both methods indicate that at early times, random actions lead to exploring new transitions with a high probability. However, at late times, once most of the transitions in the room have been explored, the undirected exploration of random walks takes a long time to explore the straggling transitions, whereas our exploration method manages to use the learned model to explore leftover transitions more efficiently (even at challenging levels of aliasing). This is confirmed in the corresponding rows of Table 1, which measures steps to complete exploration.

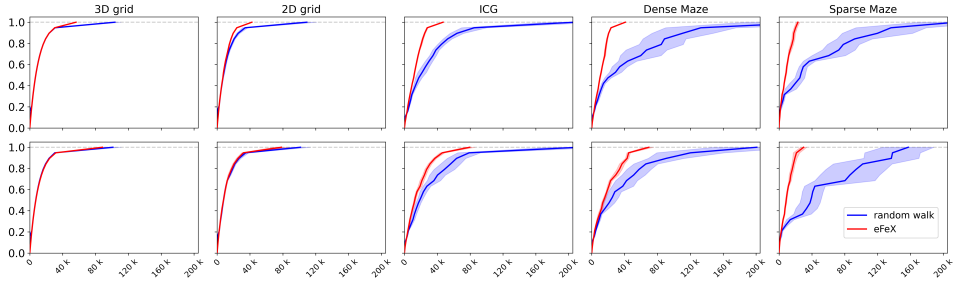


Figure 4: Fraction of transitions explored versus the number of exploration steps, for different amounts of aliasing: 100% (first row) & 30% (second row) unique emissions; eFeX in red, and random walk in blue. Bands indicate 95% confidence intervals on the estimated means. Curves stop when the respective method has explored all transitions in the environment.

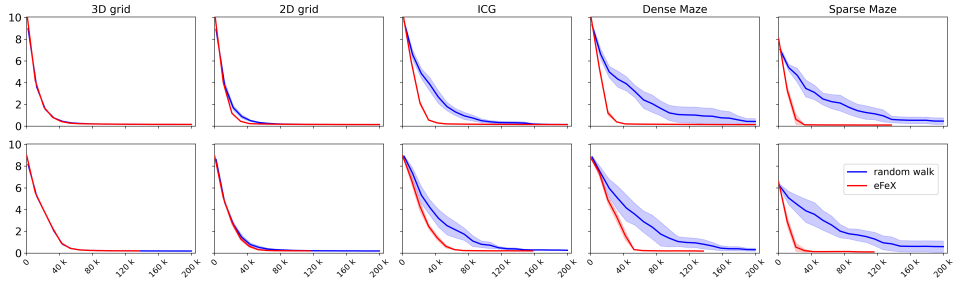


Figure 5: Negative log-likelihoods (bits per symbol) for learned models, evaluated on test sequences for different amounts of aliasing: 100% (first row) & 30% (second row) unique emissions; eFeX in red, and random walk in blue. Bands indicate 95% confidence intervals on the estimated means.

Next, we consider the "Inter-Connected Grid" (ICG), which is meant to evoke[5] a common pattern in many real-world problems where clusters of highly-connected states are then bridged by long narrow chains. Here we note that random walk tends to plateau coverage for substantial periods – getting stuck in the rooms and failing to move through the narrow corridors and explore new rooms. Our model-based active exploration approach manages to maintain the rapid exploration throughout, driving towards completion faster than random walks, again with much lower variance.

The latter two environments are mazes: the dense maze provides some room for undirected local exploration, but with a very constrained global structure and the sparse maze[6] demands that the agent perform long sequences of suitably correlated actions (necessitating accurate graph learning) to effective explore the whole environment. The environments therefore demand a highly deliberate exploration strategy, and provide a stringent test of active exploration capability. The plots (and tabular results) for these environments show that random exploration struggles miserably in these environments, while our active exploration approach is very effective in rapidly exploring all the transitions.

Since just exploring all transitions might not be sufficient to fully construct the graph (especially in the aliased setting), we need an appropriate measure to evaluate the quality of the learned graph. To this end, we evaluate the negative log-likelihood (in bits per symbol) of test sequences drawn from the ground truth graph, under a model (graph) learned from the exploratory walk. For an intuition for the meaning of this metric, it is worth noting that in the unaliased setting (with all unique states), validating the high likelihood of all one-step walks is akin to checking that all the edges in the graph have been learned correctly. This is insufficient in the aliased case, and longer walks are needed to

[5]They are also reminiscent of "lollipop graphs" which happen to be hard for random walks to cover efficiently (Chandra et al., 1989).

[6]It could be considered a far more challenging version of the one-dimensional chain, because, in addition to states being perceptually aliased, each state provides four actions, some of which (a priori unknown) move the agent forwards/backwards along the long chains, and others which keep the agent in place.

Room shape	Random Walk	eFeX	
	(indep. of aliasing)	u.f. 1.0 rooms	u.f. 0.3 rooms
3D grid	103800 ± 17330	**56476 ± 1214**	**88461 ± 6584**
2D grid	106200 ± 17816	**42358 ± 949**	**74909 ± 13269**
ICG	197266 ± 50705	**47023 ± 1256**	**79518 ± 10603**
Dense Maze	233800 ± 109030	**40879 ± 982**	**69621 ± 8862**
Sparse Maze	179925 ± 69494	**22966 ± 3515**	**29980 ± 6161**

Table 1: Number of steps taken by each method to explore all possible transitions, in each environment, with statistically significant winners bolded (comparing eFeX to random walk in each case). Note that these are variances in the underlying distribution, not variances in the estimate of the mean. The table indicates that our method performs substantially better, and has a much smaller variance in the number of steps taken for complete exploration.

disambiguate the higher-order context surrounding each node. A single long random walk carries the risk of over-emphasizing certain regions of the graph and under-emphasizing other regions, unless it mixes well. Therefore, a suitable scheme would be to consider the negative log-likelihood of long sequences drawn from the ground truth graph, averaged over many walks initialized randomly in different parts of the graph, which is the quantity we graph in Figure 5. Section A.2 of the Appendix elaborates on the precise setup of this computation.

The likelihood graphs in Figure 5 indicate that the algorithm is not just exploring all the transitions, but also learning the graph structure effectively

Taken together, these demonstrate that the proposed method is able to quickly learn good models of the underlying environments, and explore effectively using that. We also note that the variance in the number of exploration steps needed is highly controlled for our method, in sharp contrast with random walks, especially in intricate environments.

5 Conclusion

In this paper, we proposed an approach for efficiently learning large latent graphs with aliased observations. This setting is challenging because the graph states (in relation to the environment) are not known directly from observations, but must be discovered based on the long range context in observational sequences generated by exploration.

It is impractical to build models of large environments without efficient exploration, and building a model enables identifying already visited regions and efficiently directing exploration away from them. Using a strategy that thus interleaves model-building and exploration, our algorithm successfully learns graphs in a variety of topologies with perceptual aliasing and stochasticity, without making any assumptions about the geometry of the environment. We also demonstrated that random exploration can have a sample complexity that scales exponentially with the size of the environment, whereas our approach maintains a linear sample complexity in the same case.

We leveraged CSCG to learn a model that allows planning through inference, coupled with a count-based notion of exploration utility. The modular nature of our approach, however, lends itself nicely to several potentially interesting variants: reusing locally built models for further exploration, a non count-based notion of exploration utility, alternative approaches to planning that avoid re-planning each time from scratch, etc. We leave this exploration for future work.

References

Susan Amin, Maziar Gomrokchi, Harsh Satija, Herke van Hoof, and Doina Precup. A survey of exploration methods in reinforcement learning. *arXiv preprint arXiv:2109.00157*, 2021.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al.

Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

A K Chandra, P Raghavan, W L Ruzzo, and R Smolensky. The electrical resistance of a graph captures its commute and cover times. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, STOC '89, pp. 574–586, New York, NY, USA, February 1989. Association for Computing Machinery. ISBN 9780897913072. doi: 10.1145/73007.73062. URL https://doi.org/10.1145/73007.73062.

L Chrisman. Reinforcement learning with perceptual aliasing: The perceptual distinctions approach. *AAAI*, 1992. URL https://www.semanticscholar.org/paper/a1b055577a86141df13f13a3203c76a32bffdc3a.

Hanjun Dai, Yujia Li, Chenglong Wang, Rishabh Singh, Po-Sen Huang, and Pushmeet Kohli. Learning transferable graph exploration. *Advances in Neural Information Processing Systems*, 32, 2019.

Eyal Dechter, J Malmaud, Ryan P Adams, and J Tenenbaum. Bootstrap learning via modular concept discovery. *IJCAI: proceedings of the conference / sponsored by the International Joint Conferences on Artificial Intelligence*, 2013. ISSN 1045-0823. URL https://www.semanticscholar.org/paper/c51274cdd2a0305cab0bbaf0298cab63db1b9152.

Xiaowen Dong, Dorina Thanou, Michael Rabbat, and Pascal Frossard. Learning graphs from data: A signal representation perspective. *IEEE Signal Processing Magazine*, 36(3):44–63, 2019.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. First return, then explore. *Nature*, 590(7847):580–586, February 2021. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-020-03157-9. URL http://dx.doi.org/10.1038/s41586-020-03157-9.

Andrew Forney, Judea Pearl, and Elias Bareinboim. Counterfactual data-fusion for online reinforcement learners. In *International Conference on Machine Learning*, pp. 1156–1164. PMLR, 2017.

Dileep George, Rajeev V Rikhye, Nishad Gothoskar, J Swaroop Guntupalli, Antoine Dedieu, and Miguel Lázaro-Gredilla. Clone-structured graph representations enable flexible learning and vicarious evaluation of cognitive maps. *Nature communications*, 12(1):2392, April 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-22559-5. URL http://dx.doi.org/10.1038/s41467-021-22559-5.

Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P Adams, and Sergey Levine. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability. *Advances in Neural Information Processing Systems*, 34:25502–25515, 2021.

Oliver Groth, Markus Wulfmeier, Giulia Vezzani, Vibhavari Dasagi, Tim Hertweck, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Is curiosity all you need? on the utility of emergent behaviours from curious exploration. *arXiv preprint arXiv:2109.08603*, 2021.

Carlos Guestrin, Daphne Koller, Ronald Parr, and Shobha Venkataraman. Efficient solution algorithms for factored mdps. *CoRR*, abs/1106.1822, 2011.

Michael Kearns and Satinder Singh. Near-Optimal reinforcement learning in polynomial time. *Machine learning*, 49(2/3):209–232, 2002. ISSN 0885-6125, 1573-0565. doi: 10.1023/a:1017984413808. URL http://link.springer.com/10.1023/A:1017984413808.

J Zico Kolter and Andrew Y Ng. Near-Bayesian exploration in polynomial time. In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, New York, New York, USA, 2009. ACM Press. ISBN 9781605585161. doi: 10.1145/1553374.1553441. URL http://zicokolter.com/publications/kolter2009nearbayesian.pdf.

Pierre-Yves Lajoie, Siyi Hu, Giovanni Beltrame, and Luca Carlone. Modeling perceptual aliasing in SLAM via discrete–continuous graphical models. *IEEE robotics and automation letters*, 4(2): 1232–1239, April 2019. ISSN 2377-3766, 2377-3774. doi: 10.1109/lra.2019.2894852. URL https://ieeexplore.ieee.org/document/8624393/.

Renaud Lambiotte, Martin Rosvall, and Ingo Scholtes. From networks to optimal higher-order models of complex systems. *Nature physics*, 15(4):313–320, April 2019. ISSN 1745-2473. doi: 10.1038/s41567-019-0459-y. URL http://dx.doi.org/10.1038/s41567-019-0459-y.

Joel Lehman and Kenneth O Stanley. Abandoning objectives: evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, February 2011. ISSN 1063-6560, 1530-9304. doi: 10.1162/EVCO_a_00025. URL http://dx.doi.org/10.1162/EVCO_a_00025.

Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, August 1995. ISBN 9780521474658. doi: 10.1017/CBO9780511814075. URL https://play.google.com/store/books/details?id=QKVY4mDivBEC.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29, 2016.

Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, November 2007. ISSN 1662-5218. doi: 10.3389/neuro.12.006.2007. URL http://dx.doi.org/10.3389/neuro.12.006.2007.

L R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989. ISSN 1558-2256. doi: 10.1109/5.18626. URL http://dx.doi.org/10.1109/5.18626.

Rajeev V Rikhye, J Swaroop Guntupalli, Nishad Gothoskar, Miguel Lázaro-Gredilla, and Dileep George. Memorize-generalize: An online algorithm for learning higher-order sequential structure with cloned hidden markov models. *BioRxiv*, pp. 764456, 2019.

Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences of the United States of America*, 105(4):1118–1123, January 2008. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.0706851105. URL http://dx.doi.org/10.1073/pnas.0706851105.

J Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *From Animals to Animats*. The MIT Press, 1991. ISBN 9780262256674. doi: 10.7551/mitpress/3115.003.0030. URL https://direct.mit.edu/books/book/3865/chapter/162771/a-possibility-for-implementing-curiosity-and.

Shani and Brafman. Resolving perceptual aliasing in the presence of noisy sensors. *Advances in neural information processing systems*, 2004. ISSN 1049-5258. URL https://proceedings.neurips.cc/paper/2004/file/c315f0320b7cd4ec85756fac52d78076-Paper.pdf.

Sugandha Sharma, Aidan Curtis, Marta Kryven, Josh Tenenbaum, and Ila Fiete. Map induction: Compositional spatial submap learning for efficient exploration in novel environments. *arXiv preprint arXiv:2110.12301*, 2021.

Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International conference on machine learning*, pp. 5779–5788. PMLR, 2019.

Alexander L Strehl and Michael L Littman. A theoretical analysis of Model-Based interval estimation. In *Proceedings of the 22nd international conference on Machine learning - ICML '05*, New York, New York, USA, 2005. ACM Press. ISBN 9781595931801. doi: 10.1145/1102351.1102459. URL http://portal.acm.org/citation.cfm?doid=1102351.1102459.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, December 2008. ISSN 0022-0000. doi: 10.1016/j.jcss.2007.08.009. URL https://www.sciencedirect.com/science/article/pii/S0022000008000767.

Richard S Sutton and Andrew G Barto. *Reinforcement learning : an introduction*. MIT Press, Cambridge, Mass., 1998. ISBN 9780262193986. URL https://www.worldcat.org/title/37293240.

Tang, Houthooft, Foote, Stooke, Xi Chen, Duan, Schulman, DeTurck, and Abbeel. #exploration: A study of Count-Based exploration for deep reinforcement learning. *Advances in neural information processing systems*, 2017. ISSN 1049-5258. URL https://proceedings.neurips.cc/paper/2017/file/3a20f62a0af1aa152670bab3c602feed-Paper.pdf.

Sebastian Thrun. Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie Mellon University, Pittsburgh, PA, January 1992.

A Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on information theory / Professional Technical Group on Information Theory*, 13(2):260–269, April 1967. ISSN 0018-9448, 1557-9654. doi: 10.1109/TIT.1967.1054010. URL http://dx.doi.org/10.1109/TIT.1967.1054010.

Steven D Whitehead and Dana H Ballard. Learning to perceive and act by trial and error. *Machine learning*, 7(1):45–83, July 1991. ISSN 0885-6125, 1573-0565. doi: 10.1007/BF00058926. URL https://doi.org/10.1007/BF00058926.

Herbert S Wilf. The editor's corner: The white screen problem. *The American mathematical monthly: the official journal of the Mathematical Association of America*, 96(8):704–707, 1989. ISSN 0002-9890, 1930-0972. doi: 10.2307/2324718. URL http://www.jstor.org/stable/2324718.

Jian Xu, Thanuka L Wickramarathne, and Nitesh V Chawla. Representing higher-order dependencies in networks. *Science advances*, 2(5):e1600028, May 2016. ISSN 2375-2548. doi: 10.1126/sciadv.1600028. URL http://dx.doi.org/10.1126/sciadv.1600028.

A APPENDIX

A.1 ROOM DETAILS

The layouts for each of the room types are as follows:

- The 2d and 3d rooms are grids of respective dimensions, of size 41×41 and $12 \times 12 \times 12$ respectively.
- The ICG contains 16 room-like blocks of size 10×10 arranged in a square grid, connected by corridors of length 3.
- Dense mazes were generated by running depth-first search on a 2d room, and then blown up by a factor of five to make the corridors five times as thick as the walls, ultimately to a room size of 41×41.
- Sparse mazes were generated on a background of size 41×41 using the open source "lab-maze" library introduced in Beattie et al. (2016).

Each room has colors (categorical emissions) randomly assigned to each lattice position accessible to the agent. The provided numbers for aliasing ("unique fraction") correspond to the ratio of unique perceptions to the number of states in the room. For example, in a 10×10 2d room, a unique fraction of 0.8 implies 80 unique observations, which means 20 nodes would share emissions with a randomly sampled subset of 20 nodes (without repetition) from the 80 nodes with unique colors.

The dynamics are stochastic in a local manner, such that when the agent tries to take one step, the actual number of steps executed in the environment are sampled from a geometric distribution on $k \geq 1$ described by:

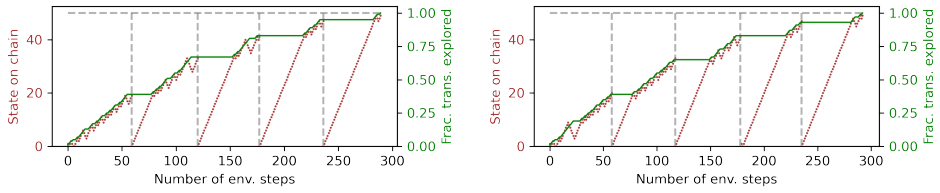
$$p_{\{k|k\geq 1, k\in\mathbb{N}\}} = p\,(1-p)^{k-1}$$

with a slippage rate $p_{\text{slip}} = 0.01$.

A.2 LIKELIHOODS OF LEARNED MODELS

To verify that the agent is not just exploring all possible transitions, but also effectively learning the appropriate models (transition graphs) for each room, we evaluate the likelihood of a bunch of test sequences (generated from the ground truth room graph) using the agent's model; see Figure 5. The computed likelihoods are averaged over random walk sequences (each of length 100) initialized in 1000 different locations in each room. Averaging over a large number of independent walks should ensure that these walks don't get stuck in some sub-region of the graph, and lengths of 100 on each one should ensure that we are evaluating that the agent also correctly learns the long-range context.

A.3 EXAMPLE TRACES FOR THE ONE-DIMENSIONAL CHAIN ENVIRONMENT



(a) Exploration trace for one-dimensional chain of length 50 -vs- number of steps. Actual states in brown, and fraction of transitions explored in green.

(b) Exploration trace for one-dimensional chain of length 50 (with stochastic trap) -vs- number of steps. Actual states in brown, and fraction of transitions explored in green.

Figure 6: Example traces for the one-dimensional chain experiment.