Episodic Memory for Subjective-Timescale Models

Alexey Zakharov [1 2] **Matthew Crosby** [3 1] **Zafeirios Fountas** [4 2]

Abstract

Planning in complex environments requires reasoning over multi-step timescales. However, in model-based learning, an agent's model is more commonly defined over transitions between consecutive states. This leads to plans using intermediate states that are either unnecessary, or worse, introduce cumulative prediction errors. Inspired by the recent works on human time perception, we devise a novel approach for learning a transition dynamics model based on the sequences of episodic memories that define an agent's subjective timescale – over which it learns world dynamics and over which future planning is performed. We analyse the emergent benefits of the subjective-timescale model (STM) by incorporating it into two disparate model-based methods – Dreamer and deep active inference. Using 3D visual foraging tasks, we demonstrate that STM can systematically vary the temporal extent of its predictions and is more likely to predict future salient events (such as new objects coming into view). In comparison to the agents trained using objective timescales, STM agents also collect more rewards due to their ability to perform flexible planning and a more pronounced exploratory behaviour.

1. Introduction

An agent endowed with a model of its environment has the ability to predict the consequences of its actions and perform planning into the future. Models allow agents to simulate the possible action-conditioned futures from their current state, even if the state was never visited during learning. As a result, *model-based* approaches can provide agents with better generalization abilities across both states and tasks in an environment, compared to their *model-free* counterparts

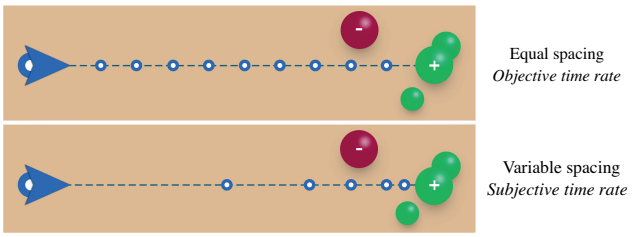


Figure 1. In standard approaches an agent's model inherits environment's object time rate, resulting in equally-spaced predictions and likely including redundant intermediate states (top). In our proposed STM model, the temporal extent of predictions varies automatically depending on the context, resulting in more informative planning (bottom).

(Racanière et al., 2017; Mishra et al., 2017).

The most popular framework for developing agents with internal models is model-based reinforcement learning (RL). Model-based RL has seen great progress in recent years, with a number of proposed architectures attempting to improve both the quality and the usage of these models (Racanière et al., 2017; Kansky et al., 2017; Hamrick, 2019; Kaiser et al., 2020). Nevertheless, learning internal models affords a number of unresolved problems. The central one of them is model-bias, in which the imperfections of the learned model result in unwanted over-optimism and sequential error accumulation for long-term predictions (Deisenroth & Rasmussen, 2011). Long-term predictions are additionally computationally expensive in environments with slow temporal dynamics, given that all intermediary states must be predicted. Moreover, slow world dynamics[1] can inhibit the learning of dependencies between temporally-distant events, which can be crucial for environments with sparse rewards. Finally, the temporal extent of future predictions is limited to the *objective* timescale of the environment over which the transition dynamics has been learned. This leaves little room for flexible and context-dependent planning over varying timescales which is characteristic to animals and humans (Clayton et al., 2003; Buhusi & Meck, 2005; Cheke & Clayton, 2011).

The final issue exemplifies the disadvantage of the classical view on internal models, in which they are expected

[1]Imperial College London, UK [2]Emotech Labs, UK [3]Leverhulme Center for the Future of Intelligence, UK [4]WCHN, University College London, UK. Correspondence to: Alexey Zakharov <az519@ic.ac.uk>.

Proceedings of the Unsupervised Reinforcement Learning Workshop at the 38th International Conference on Machine Learning, PMLR 139, 2021.

[1]Worlds with small change in state given an action

to capture the ground-truth transition dynamics of the environment. Furthermore, in more complex environments with first-person observations, this perspective does not take into account the apparent subjectivity of first-person experiences. In particular, the agent's learned representations of the environment's transition dynamics implicitly include information about *time*. Little work has been done to address the concept of time perception in model-based agents. Empirical evidence from the studies of human and animal cognition has led some researchers to posit that intelligent biological organisms do not perceive time precisely and do not possess an explicit clock mechanism responsible for keeping track of it (Hills, 2003; Roseboom et al., 2019; Sherman et al., 2020). For instance, humans tend to perceive time slower in environments rich in perceptual content (e.g. busy city), and faster in environments with little perceptual change (e.g. empty field). The mechanisms of subjective time perception still remain unknown; however, recent computational models based on episodic memory were able to closely model the deviations of human time perception from veridical perception (Fountas et al., 2020b).

Inspired by this account, in this work we propose *subjective-timescale model* (STM), an alternative approach to learning a transition dynamics model, by replacing the objective timescale with a subjective one. The latter represents the timescale by which an agent perceives events in an environment, predicts future states, and which is defined by the sequences of episodic memories. These memories are accumulated on the basis of saliency (i.e. how poorly an event was predicted by the agent's transition model), which attempts to mimic the way humans perceive time and results in an agent with an ability to plan over varying timescales and construct novel future scenarios.

We employ two disparate agents to analyse the emergent characteristics of STM and demonstrate its applicability and usefulness to a variety of model-based agents. We particularly emphasise the ease with which STM can be plugged into an existing model-based algorithm to significantly improve its performance. For this work, we use Dreamer by Hafner et al. (2020) and deep active inference by Fountas et al. (2020a). While these agents involve considerably different training and planning procedures, their underlying nature implies the central role of a generative model, including the learned transition dynamics model. Thus, as will be seen, both of the agents benefit from the incorporation of STM.

Using these two agents and a combination of visually-complex first-person foraging tasks, we demonstrate that the resulting characteristics of STM allow the agent to automatically perform both short- and long-term planning using the same computational resources and without any explicit mechanism for adjusting the temporal extent of its predictions. Furthermore, for long-term predictions STM systematically performs temporal jumps (skipping intermediary steps), thus providing more informative future predictions and reducing the detrimental effects of one-step prediction error accumulation. Additionally, being trained on salient events, STM much more frequently imagines futures that contain epistemically-surprising events, which incentivises exploratory behaviour. In both cases, the STM version outperforms the objective-timescale models, achieving higher rewards with significantly fewer penalties.

2. Related Work

Model-based RL. Internal models are extensively studied in the field of model-based RL. Using linear models to explicitly model transition dynamics has achieved impressive results in robotics (Bagnell & Schneider, 2001; Abbeel et al., 2006; Levine & Abbeel, 2014a;b; Watter et al., 2015; Levine et al., 2016; Kumar et al., 2016). In general, however, their application is limited to low-dimensional domains and relatively simple environment dynamics. Similarly, Gaussian Processes (GPs) have been used (Ko et al., 2007; Deisenroth & Rasmussen, 2011). Their probabilistic nature allows for state uncertainty estimation, which can be incorporated in the planning module to make more cautious predictions; however, GPs struggle to scale to high-dimensional data. An alternative and recently more prevalent method for parametrising transition models is to use neural networks. These are particularly attractive due to their recent proven success in a variety of domains, including deep model-free RL (Silver et al., 2017), ability to deal with high-dimensional data, and existence of methods for uncertainty quantification (Blundell et al., 2015; Gal & Ghahramani, 2016). Different deep learning architectures have been utilised including fully-connected neural networks (Nagabandi et al., 2018; Feinberg et al., 2018; Kurutach et al., 2018) and autoregressive models (Racanière et al., 2017; Ha & Schmidhuber, 2018; Ke et al., 2019), showing promising results in environments with relatively high-dimensional state spaces. In particular, autoregressive architectures, such as Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997), are capable of modelling non-Markovian environments and of learning temporal dependencies. Nevertheless, LSTMs are still limited in their ability to learn relations between temporally-distant events, which is exacerbated in environments where little change occurs given an action.

Uncertainty quantification using ensemble methods (Kalweit & Boedecker, 2017; Buckman et al., 2018; Clavera et al., 2020) or Bayesian neural networks (McAllister & Rasmussen, 2016; Depeweg et al., 2017) have been proposed to tackle model bias and sequential error accumulation. Other works have focused on techniques to create more accurate

long-term predictions. Mishra et al. (2017) used a segment-based approach to predict entire trajectories at once in an attempt to avoid one-step prediction error accumulation. A work by Ke et al. (2019) used an autoregressive model and introduced a regularising auxiliary cost with respect to the encodings of future observations, thus forcing the latent states to carry useful information for long-horizon predictions. In contrast, the work presented in this paper re-focuses the objective from attempting to create better parametrisation techniques or mitigating methods to simply transforming the timescale over which the dynamics of an environment is learned. As will be seen, our approach can lead to more informative and efficient long-term predictions without compromising agent's ability to plan over short time-horizons.

Time and Models. Relatively little work has been done in the field to study the role and practical use of time perception for artificial agents, despite its apparent importance for intelligent behaviour (Deverett et al., 2019). For instance, Braylan et al. (2015) showed that the degree of frame-skipping can have a significant impact on an agent's performance in Atari 2600 games. Nevertheless, several works have proposed methods for learning models with particular temporal characteristics, such as time-agnostic models (Jayaraman et al., 2019) or 'jumpy' models with fixed (Gregor & Besse, 2019; Buesing et al., 2018) or adaptive frame intervals (Goyal et al., 2019; Neitz et al., 2018; Pertsch et al., 2020). In line with some of these works, our approach is designed to perform adaptive frame skips, but unlike all of them it is characterised by the simplicity of use and incorporation into existing model-based methods. Furthermore, our method aims to mimic human time perception, being inspired by the latest works from computational neuroscience.

Active Inference. Until now, most of the work on active inference has been done in low-dimensional and discrete state spaces (Friston et al., 2015; 2017a;b;c). Recently, however, there has been a rising interest in scaling active inference and applying it to environments with continuous and/or large state spaces (Ueltzhöffer, 2018; Tschantz et al., 2019; Çatal et al., 2019; Millidge, 2019; Fountas et al., 2020a). Although these works used deep learning techniques, their generative models have so far been designed to be Markovian and trained over the objective timescale of the environment.

Episodic Memory. In neuroscience, episodic memory is used to describe autobiographical memories that link a collection of first-person sensory experiences at a specific time and place (Tulving, 1972). Past studies in the field suggest that episodic memory plays an important role in human learning (Mahr & Csibra, 2017), and may capture a wide range of potential functional purposes, such as construction of novel future scenarios (Schacter et al., 2007; Hassabis et al., 2007; Schacter et al., 2012), mental time-travel (Michaelian, 2016) or assisting in the formation of new semantic memories (Greenberg & Verfaellie, 2010). A recent computational model of episodic memory (Fountas et al., 2020b) also relates it to the human ability to estimate time durations.

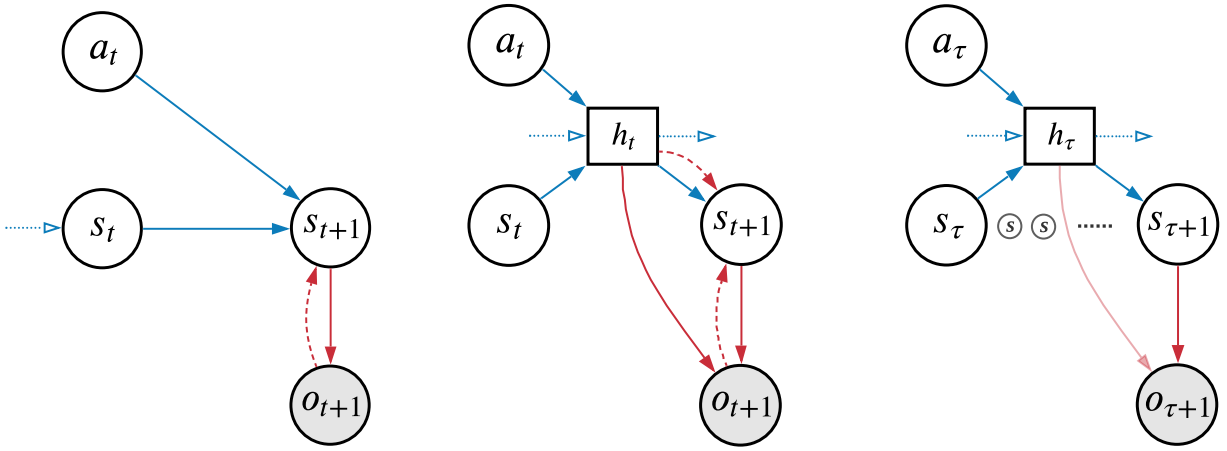
The application of episodic memory in reinforcement learning has been somewhat limited. Some works have employed simple forms of memory to improve the performance of a deep model-free RL agent via experience replay (Mnih et al., 2015; Schaul et al., 2016; Espeholt et al., 2018). However, these methods do not incorporate information about associative or temporal dependencies between the memories (Hansen et al., 2018). Read-write memory banks have also been implemented alongside gradient-based systems (memory-augmented neural networks) for assisting in learning and prediction (Graves et al., 2014; 2016; Oh et al., 2016; Jung et al., 2018; Hung et al., 2019). Further, episodic memory has been used for non-parametric Q-function approximation (Hansen et al., 2018; Blundell et al., 2016; Pritzel et al., 2017; Zhu et al., 2020). It has also been proposed to be used directly for control as a faster and more efficient alternative to model-based and model-free approaches in RL, such as instance-based control (Lengyel & Dayan, 2007; Gershman & Daw, 2017; Botvinick et al., 2019) and one-shot learning (Kaiser et al., 2017). In contrast, our paper considers a novel way of using episodic memories – in defining the agent's subjective timescale of the environment and training a transition dynamics model over the sequences of these memories.

3. Baseline Agents

To test the versatility of STM, we integrate it into two types of model-based agents that represent fundamentally different approaches to building model-based agents. We consider a partially-observable Markov decision process (POMDP) with discrete time t. At every time step, agents receive observation $o_t \in \mathcal{O}$ upon taking a discrete action $a_{t-1} \in \mathcal{A}$. Latent states inferred by an agent are denoted by $s_t \in \mathcal{S}$. For Dreamer, an agent also observes a scalar reward $r_t \in \mathcal{R}$.

3.1. Dreamer

Dreamer (Hafner et al., 2020) is a well-established model-based RL agent that uses a recurrent generative model (Hafner et al., 2019) and latent imagination to learn a reward-maximising policy. Specifically, it consists of: a representation model $q(s_t|s_{t-1}, a_{t-1}, o_t; \theta)$, a transition model $p(s_t|s_{t-1}, a_{t-1}; \theta)$, a reward model $q(r_t|s_t; \theta)$, an action model $q(a_t|s_t; \phi)$, and a value model $v(s_t; \psi)$ (Fig. 2b), where $\{\theta, \phi, \psi\}$ are parameters of the models. The dynam-



(a) Deep active inference (b) Dreamer (c) Subjective generative model

Figure 2. Graphical models. Circles and rectangles denote stochastic and deterministic variables, respectively. Full lines represent generative models, while dotted lines inference models. We also use blue to denote the transition dynamics model, and red for perception models. Figure (a) shows the Markovian deep active inference agent. Figure (b) shows the graphical model of the Dreamer agent, including the additional recurrent state h. Figure (c) shows the subjective-timescale generative model that performs adaptive skips and that operates over the subjective time variable τ. Light red line indicates the dependency that is only used in the Dreamer agent.

ics model trained from environment observations is used to perform latent roll-outs up to horizon H while predicting rewards $\hat{r}_t$ and values $v(s_t)$, and then updating the action and value models using the value estimates $V_\lambda(s_t)$, such that the action model maximises expected rewards, while the value model regresses the value of a state:

$$\max_{\phi} \mathbb{E}_{p_\theta, q_\phi} \Big(\sum_{t=k}^{k+H} V_\lambda(s_t) \Big), \tag{1}$$

$$\max_{\psi} \mathbb{E}_{p_\theta, q_\phi} \Big(\frac{1}{2} \sum_{t=k}^{k+H} ||v(s_t) - V_\lambda(s_t)||^2 \Big). \tag{2}$$

In turn, value estimates V_λ are calculated via a combination of predicted rewards and values (see Appendix A.2). Dreamer performs competitively on a wide range of traditional RL tasks.

3.2. Deep Active Inference

The second baseline is a deep active inference (DAI) agent with Monte-Carlo tree search (Fountas et al., 2020a). Its selection is motivated by an observation-based objective and a purely model-based planning procedure, which allow for a more detailed analysis of STM's properties. The generative model of this agent is defined as $p(o_{1:t}, s_{1:t}, a_{1:t}; \theta)$, where θ are parameters of the model. Specifically, it includes two factors: a transition model $p(s_t|s_{t-1}, a_{t-1}; \theta)$ and a latent state decoder $p(o_t|s_t; \theta)$ parametrised by feed-forward neural networks (Fig. 2a). The agent also possesses two inference models, which are trained using amortized inference: a habitual network $q(a_t; \phi_a)$ and observation encoder $q(s_t; \phi_s)$ parametrised by ϕ_a and ϕ_s, respectively. Further, we use π to denote a sequence of actions up to some time horizon H. Training is performed using the variational free energy at an arbitrary time-step t:

$$F_t = -\mathbb{E}_{q(s_t)} \left[\log p(o_t|s_t; \theta) \right] \tag{3a}$$

$$+ D_{KL} \left[q(s_t; \phi_s) \| p(s_t|s_{t-1}, a_{t-1}; \theta) \right] \tag{3b}$$

$$+ \mathbb{E}_{q(s_t)} \left[D_{KL} \left[q(a_t; \phi_a) \| p(a_t) \right] \right], \tag{3c}$$

where $p(a) = \sum_{\pi: a_1 = a} p(\pi)$ is the summed probability of all policies π beginning with action a. The expected free energy (EFE) of the generative model up to some time horizon H is defined as:

$$G(\pi) = \sum_{k=t}^{H} \mathbb{E}_{\tilde{q}} \left[\log q(s_k, \theta|\pi) - \log p(o_k, s_k, \theta|\pi) \right], \tag{4a}$$

where $\tilde{q} = q(o_k, s_k, \theta|\pi)$ and $p(o_k, s_k, \theta|\pi) = p(o_k|\pi) q(s_k|o_k, \pi) p(\theta|s_k, o_k, \pi)$. Action selection is aided with Monte Carlo tree search (MCTS), ensuring a more efficient trajectory search. Specifically, MCTS generates a weighted tree that is used to sample policies from the current timestep, where the weights refer to the agent's estimation of the EFE given a state-action pair, $\tilde{G}(s, a)$. The nodes of the tree are predicted via the transition model, $p(s_t|s_{t-1}, a_{t-1}; \theta)$. At the end of the search, MCTS is used to construct the action prior, $p(a_t) = N(a_t, s_t) / \sum_j N(a_{j,t}, s_t)$, where $N(a_t, s_t)$ is the number of times action a_t has been taken from state s_t. More details

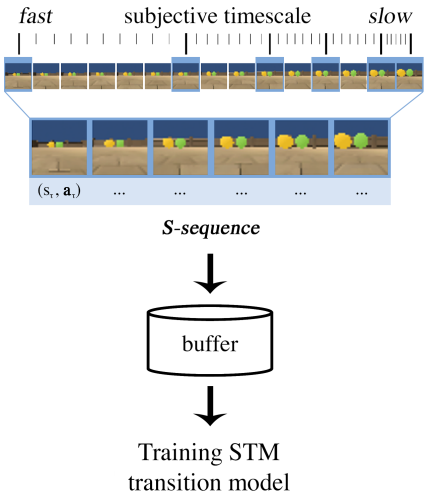


Figure 3. STM pipeline. As the agent moves through the environment, states s that exceeded a pre-defined threshold are recorded along with all successive actions $\boldsymbol{a}$ in an S-sequence. S-sequences are saved in a buffer at the end of each episode and are later sampled for training a subjective-timescale transition model.

on active inference and MCTS planning procedure can be found in Appendix A and C.2.

4. Subjective-Timescale Model

We introduce subjective-timescale model (STM) – a transition dynamics model trained over sequences of episodic memories that define a more useful timescale. As such, the system consists of a memory accumulation module to selectively record salient events and an autoregressive transition model (Figure 3).

We define a *ground-truth sequence* as a sequence of *all* states experienced in an environment during a single episode denoted with t, $S_g = \{s_0, ..., s_t, s_{t+1}, ..., s_T\}$, and an *S-sequence* (subjective sequence) as a sequence of states selectively picked by our system, and over which the new transition model would be learned, $S_e = \{s_0, ..., s_\tau, s_{\tau+1}, ..., s_\mathrm{T}\}$, denoted with τ. Each unit in an S-sequence is called *an episodic memory* and consists of a set of sufficient statistics, $s = \{\boldsymbol{\mu}_s, \boldsymbol{\sigma}_s\}$, where $\boldsymbol{\mu}_s$ and $\boldsymbol{\sigma}_s$ are mean and variance vectors of a Gaussian-distributed state s, respectively. Additionally, each episodic memory contains a reference to its preceding (parent) episodic memory and all actions until the next one. We call the process of recording S-sequences *memory accumulation*.

4.1. Memory Accumulation

Previous work on time perception and episodic memory (Fountas et al., 2020b) employed *saliency* of an event, or the generative model's prediction error, as the memory formation criterion. Selection of this criterion is informed by the experimental evidence from neuroscience on episodic memory (Greve et al., 2017; Jang et al., 2018; Rouhani et al., 2018) and recent time perception models (Roseboom et al., 2019). Inspired by this account, our memory accumulation system uses the prediction error (or the free energy) of the objective-timescale *transition model* as a measure of event saliency, and forms memories when a pre-defined threshold is exceeded.

Accumulation To train STM, an agent moves in the environment under a pre-trained generative model. During this process, each transition is evaluated based on the objective transition model prediction error (or free energy), which represents the degree of surprise experienced by the transition model upon taking an action. If the value exceeds a pre-defined threshold, ϵ, a memory is formed and placed into an S-sequence. At the end of each episode, the recorded S-sequence is saved for later use. Over the course of training, we increase the threshold using $\epsilon = \epsilon_{\text{max}}(1 - \exp(-\lambda \cdot i))$, where ϵ_{max} is the maximum allowable threshold, λ is the half-life of the exponential decay term and i is training iteration.

Recorded memories We can categorise the transitions that cause higher values of transition model prediction errors into two main groups: epistemic surprise and model-imperfection surprise. The former refers to transitions that the model could not have predicted accurately due to the lack of information about the current state of the environment (e.g. objects coming into view). The latter refers to the main bulk of these high prediction-error transitions and stems from the inherent imperfections of the learned dynamics. Specifically, less frequently-occurring observations with richer combinatorial structure would systematically result in higher compounded transition model errors, given that these would be characteristic of more complex scenes. Empirical evidence of these trends can be found in Appendix D. As will become apparent, the presence of these two categories in the recorded S-sequences results in the model's ability to vary its prediction timescale based on the perceptual context and systematically imagine future salient events. Two examples of S-sequences are shown in Figure 4, where colours are used to indicate parts of the sequences with different accumulation rates.

A transition dynamics model is necessarily trained with respect to actions that an agent took to reach subsequent states. However, STM records memories over an arbitrary number of steps, thus leaving action sequences of variable length. For the purposes of this paper, we implement a simple heuristic to summarise agent's trajectories, which is enough to provide our system with the necessary information to learn action-conditioned predictions. We do it by

estimating the angle between the agent's initial position and its final position at the time-step of the subsequent memory (full details can be found in Appendix B.3).

4.2. Transition Dynamics Model

As mentioned, S-sequences are characterised by the presence of epistemically-surprising and salient events squeezed together in the recorded episodes. As a result, training on these sequences is more conducive for learning *temporal dependencies* between important states. For this reason, we employ a recurrent transition dynamics model (LSTM (Hochreiter & Schmidhuber, 1997) for active inference and GRU (Cho et al., 2014) for Dreamer), which utilises internal memory states to store information about preceding inputs. As such, hidden states h_τ at subjective time τ are calculated via a deterministic mapping, $h_\tau = f_{\theta_h}(s_\tau, a_\tau, h_{\tau-1})$, where s_τ and a_τ are the latent state and action, respectively, and f_{θ_h} is a deterministic function of the used recurrent unit with parameters θ_h. The hidden state h_τ is then mapped to a latent state $s_{\tau+1}$ at the next subjective time $\tau + 1$. Importantly, the parameters of STM can be trained normally according to the procedure of the model-based agent or in isolation via loss function defined as,

$$\mathcal{L}_{STM} = \frac{1}{\mathrm{T}} \sum_{\tau}^{\mathrm{T}} D_{KL}\Big[q(s_{\tau+1}) \| p(s_{\tau+1}|h_\tau)\Big]. \quad (5)$$

The STM generative model is shown in Figure 2c and the procedure is outlined in Algorithm 1. Architectural and training details of the model can be found in Appendix B.

Algorithm 1 Training with STM

Initialise :
Pre-trained objective-timescale generative model, $p(o, s, a; \theta)$
Memory threshold, ϵ
Episodic buffer, $\mathcal{D} \leftarrow \{\}$
for $episode = 1$ ***to*** ∞ **do**
  Initialise S-sequence with the first observation and state, $\mathcal{S} \leftarrow \{s_1, o_1\}$
  for $t = 1$ ***to*** *max length* **do**
    `// Interact and collect memories`
    Sample an action and get a new observation
    Compute surprise of the objective-timescale transition model, L_{obj}
    if $L_{obj} \geq \epsilon$ **then**
      Add memory: $\mathcal{S} \leftarrow \mathcal{S} \cup \{a_t, s_{t+1}, o_{t+1}\}$
    else
      Add action: $\mathcal{S} \leftarrow \mathcal{S} \cup \{a_t\}$
    `// Training`
    Sample a batch of S-sequences: $\mathcal{B}_i \sim \mathcal{D}$
    Train the models: $\mathbb{E}_{\mathcal{B}_i}[\mathcal{L}]$

5. Experiments

To analyse the properties and benefits of STM, we use a combination of visually-complex 3D foraging tasks from DeepMind Lab (DMLab) (Beattie et al., 2016) and Animal-AI (AAI) (Crosby et al., 2020; Crosby, 2020). DMLab and AAI afford suitable testing conditions given their large size, slow objective temporal dynamics, and low density of rewards.

Foraging with sparse rewards Within DMLab and AAI, we use sparsely populated arenas with positive[2] and negative rewards. The chosen map configuration is motivated by the fact that a successful agent would be forced to perform both short- and long-distance planning, as well as a more extensive exploration of the environment.

Foraging in a maze Further, we test the Dreamer agent within a more complex maze configuration in DMLab. These maps include additional levels of complexity such as epistemic distractions (wall imagery) and more difficult terrain with obstructions. More pronounced exploratory navigation and reward signals are crucial in this setting.

Training Across all tasks, we use the environment's intrinsic frame rate (i.e. no action repetitions) and an episode length of 1000 steps. Observations are of size $32 \times 32 \times 3$ and $64 \times 64 \times 3$ for AAI and DMLab, respectively. We train the objective- and subjective-timescale agents using the same hyperparameters on a single NVIDIA Quadro RTX 6000 GPU.

5.1. Reward-based Results

Enhanced reward seeking Upon training completion, we compare the objective- and subjective-timescale agents performance in randomly-generated environments and validate a statistically significant improvement in solving the tasks by agents with STM. Table 1 summarises reward-based results. This empirical evidence indicates that training and planning over the subjective timescale can be considerably advantageous to model-based agents.

More cautious behaviour The ability of STM to perform both short- and long-term planning allows the agent to be more cautious in selecting actions that may lead to negative rewards in the future. Table 1 shows a large drop in the collected negative rewards when the agents employ STM.

5.2. Varying Prediction Timescale

Much like human perception of time changes depending on the perceptual content of the surroundings, our agent

[2] In AAI, we use one green (terminal) and one yellow rewards. In DMLab, there is only one type of positive reward.

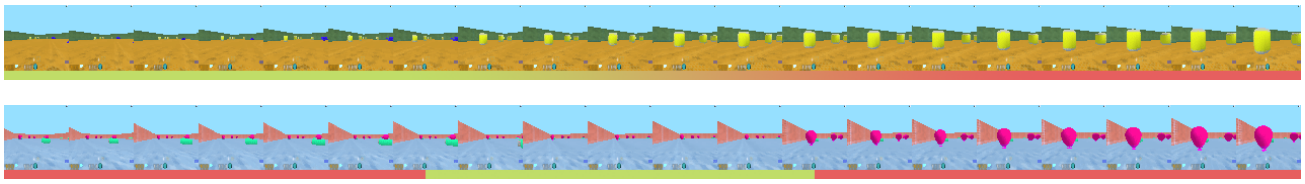


Figure 4. Examples of collected S-sequences. S-sequences tend to perform temporal jumps in visually simple settings, thus defining a more useful timescale. We use red for slower and green for faster parts of the sequences. As evident, STM can produce large skips of uninteresting areas of the arena, while slowing down in more important settings.

Task	Agent	Rewards	p-value	Penalties
AAI Sparse Rewards	DAI	1.06 ± 4.12	< 0.001	0.22 ± 0.024
	DAI + STM	2.70 ± 3.53		0.12 ± 0.002
DMLab Sparse Rewards	Dreamer	1.05 ± 0.89	< 0.001	0.24 ± 0.019
	Dreamer + STM	1.26 ± 0.46		0.01 ± 0.003
DMLab Maze Rewards	Dreamer	2.81 ± 4.20	0.043	n/a
	Dreamer + STM	3.46 ± 5.83		n/a

Table 1. Agent performances (per episode)

varies the prediction timescale depending on the context it finds itself in. Specifically, in a static environment the complexity of any given observation is primarily driven by its configurable parts (e.g. objects, which may appear in different sizes, colours, locations, etc.). As a result, our agent tends to predict farther into the future in the absence of any nearby objects, and slows its timescale, predicting at finer temporal rate, when the objects are close.

The EFE planning objective of the active inference agent allows us to visualise this property of STM by means of an energy map. Figure 6a shows the agent's values of EFE at different locations in the environment calculated using the active inference agent with and without STM using identical MCTS parameters. Specifically, the map contains a single green sphere placed in the middle, and the agent's orientation is always towards the top of the image. Regions with lower values of EFE (more blue) indicate the agent's ability to 'see' the sphere in its imagination roll-outs during planning, since its appearance decreases the value of the observational prior component. The comparison of these regions produced by the two models indicate two important things. First, the significantly larger blue region of DAI with STM (right) points to the model's ability to predict farther into the future, when placed at a distance from the object of interest, in contrast to the agent with the objective-timescale transition model. Second, STM's blue region also extends to the regions very close to the sphere, suggesting that short-term planning is not compromised.

Performing temporal jumps and skipping unnecessary intermediary steps affords greater computational efficiency and reduces the detrimental effects of sequential error accumulation, as can be seen in Figure 5. Similarly, while STM is able to predict far ahead, its inherent flexibility to automatically predict over varying timescales does not com-

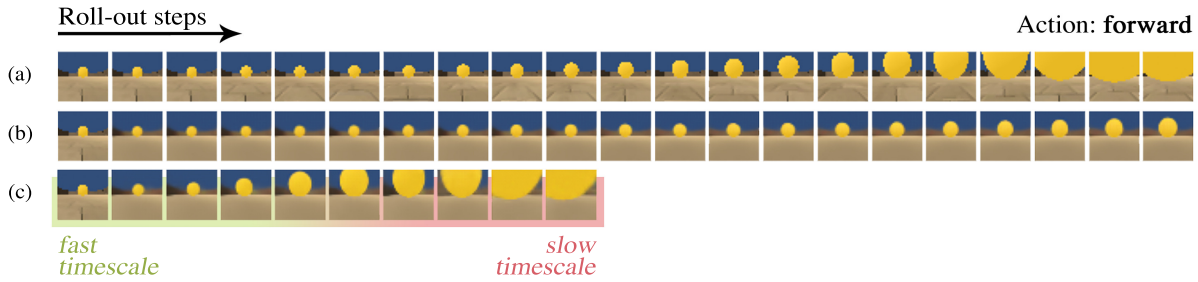
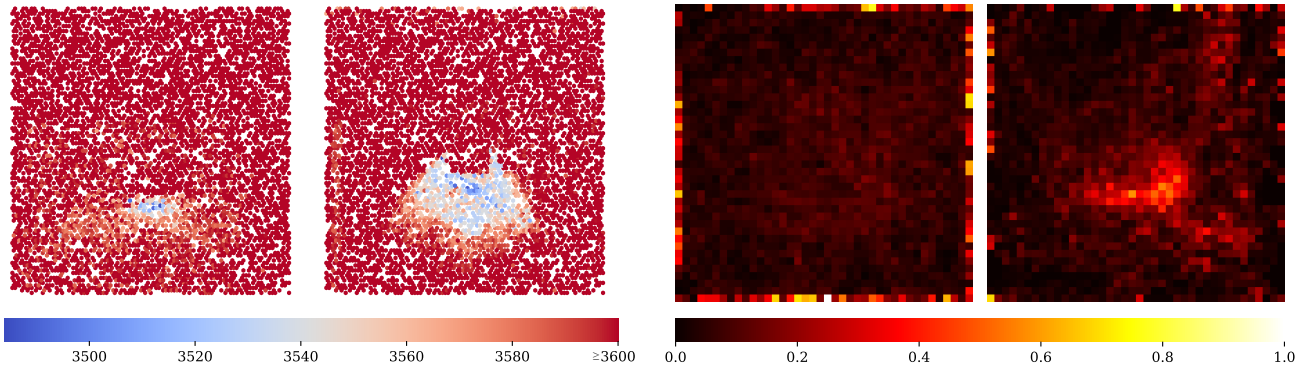


Figure 5. STM can vary prediction timescale. (a) Ground-truth observations shown for reference. (b) DAI without STM suffers from slow-timescale predictions and error accumulation, resulting in poorly informative roll-outs. (c) DAI with STM predicts a rapid approach to the object of interest, later becoming more fine-grained.



(a) Energy map with a central sphere

(b) Visited locations in a map with border attractors

Figure 6. (a) Visualisation of EFE values calculated by DAI without (left) and with STM (right) at different locations of the arena containing a green sphere in the centre. Larger low-value region of STM points to the ability to perform temporally-flexible planning (b) Normalised heatmap of visited locations (in an empty map). STM agent (right) explores the environment much more extensively than the baseline (left) and is less prone to spend time at suboptimal border locations.

promise the agent's performance when the states of interest are close. Thus, a separate mechanism for adjusting how far into the future an agent should plan is not necessary and is implicitly handled by our model. Finally, STM allows the agent to make more *informed* decisions in an environment, as it tends to populate the roll-outs with salient observations of the short- and long-term futures depending on the context. As such, STM effectively re-focuses the central purpose of a transition model from most accurately modeling the ground-truth dynamics of an environment to predicting states more informative with respect to the affordances of its surroundings.

5.3. Imagining Surprising Events

In the absence of interesting surroundings, it is useful for an agent to imagine the kinds of observations it *could* see in the future. We show that since S-sequences frequently include epistemically-surprising memories, STM favours prediction of salient events. For instance, in the context of the selected environments, these events mostly constitute objects coming into agent's view upon turning. To investigate this, we compared 1000 roll-outs using DAI with and without STM (example shown in Fig.7), counting objects that the models imagine starting from an initial observation of an empty arena. For roll-outs conditioned on turning right, we detect 1274 objects in roll-outs made with STM compared to 97 without STM; similarly, for turning left, 941 with STM compared to 0 without STM.

The ability of the STM to imagine novel and salient future events encourages exploratory behaviour that is distinct from the agents' information seeking motivations. To test this, we create an empty AAI arena with light brown walls and set a pre-trained active inference agent's observational prior to a green colour. In this scenario, brown walls act as suboptimal attractors that reduce agent's EFE, meaning that the agent is likely to stick around them unless it can imagine more optimal states elsewhere. Figure 6b shows a striking difference in the heat map of visited locations between DAI without and with STM (after 50k steps). As can be seen, the STM agent explores the environment even in the presence of the ubiquitous suboptimal border states (acting as rewarding distractions), indicating stronger exploratory motives.

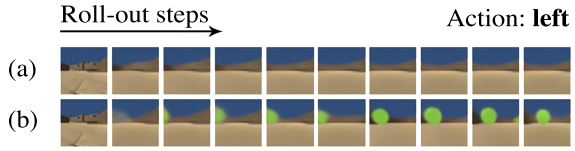


Figure 7. STM can imagine surprising events. (a) Prediction of DAI agent without STM. (b) Prediction of DAI with STM. Despite the fact that appearance of objects is a rare event, STM frequently predicts them in the roll-outs.

6. Conclusion

We proposed STM, a novel approach to learning a transition dynamics model with the use of sequences of episodic memories, which define an agent's more useful, subjective timescale. As shown, the emergent properties of STM can be useful to different model-based agents that rely on the transition dynamics model in either training or inference. We demonstrated its effectiveness within two types of agents and by using several complex 3D foraging tasks within DMLab and AAI. Nevertheless, a limitation of the current version of STM is its action summarisation heuristic. We pose that integrating general techniques for summarising sequences of actions will allow for more advanced planning and application to environments with more demanding control requirements.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Abbeel, P., Quigley, M., and Ng, A. Y. Using inaccurate models in reinforcement learning. In *ICML '06*, 2006.

Bagnell, J. A. and Schneider, J. G. Autonomous helicopter control using reinforcement learning policy search methods. *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, 2:1615–1620 vol.2, 2001.

Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., Schrittwieser, J., Anderson, K., York, S., Cant, M., Cain, A., Bolton, A., Gaffney, S., King, H., Hassabis, D., Legg, S., and Petersen, S. Deepmind lab. gpl-2.0 license. *ArXiv*, abs/1612.03801, 2016. URL https://github.com/deepmind/lab.

Beyret, B., Hernández-Orallo, J., Cheke, L. G., Halina, M., Shanahan, M., and Crosby, M. The Animal-AI Environment: Training and testing animal-like artificial cognition. apache-2.0 license. *ArXiv*, abs/1909.07483, 2019. URL https://github.com/beyretb/AnimalAI-Olympics.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *ArXiv*, abs/1505.05424, 2015.

Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J. Z., Rae, J. W., Wierstra, D., and Hassabis, D. Model-free episodic control. *ArXiv*, abs/1606.04460, 2016.

Botvinick, M. M., Ritter, S., Wang, J. X., Kurth-Nelson, Z., and Hassabis, D. Reinforcement learning, fast and slow. *Trends in cognitive sciences*, 23 5:408–422, 2019.

Braylan, A., Hollenbeck, M., Meyerson, E., and Miikkulainen, R. Frame skip is a powerful parameter for learning to play atari. In *AAAI Workshop: Learning for General Competency in Video Games*, 2015.

Buckman, J., Hafner, D., Tucker, G., Brevdo, E., and Lee, H. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *NeurIPS*, 2018.

Buesing, L., Weber, T., Racanière, S., Eslami, S., Rezende, D. J., Reichert, D. P., Viola, F., Besse, F., Gregor, K., Hassabis, D., and Wierstra, D. Learning and querying fast generative models for reinforcement learning. *ArXiv*, abs/1802.03006, 2018.

Buhusi, C. V. and Meck, W. H. What makes us tick? functional and neural mechanisms of interval timing. *Nature Reviews Neuroscience*, 6:755–765, 2005.

Cheke, L. and Clayton, N. Eurasian jays (garrulus glandarius) overcome their current desires to anticipate two distinct future needs and plan for them appropriately. *Biology Letters*, 8:171 – 175, 2011.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://www.aclweb.org/anthology/D14-1179.

Clavera, I., Fu, Y., and Abbeel, P. Model-augmented actor-critic: Backpropagating through paths. *ArXiv*, abs/2005.08068, 2020.

Clayton, N. S., Bussey, T. J., and Dickinson, A. Can animals recall the past and plan for the future? *Nature Reviews Neuroscience*, 4:685–691, 2003.

Crosby, M. Building thinking machines by solving animal cognition tasks. *Minds and Machines*, 2020.

Crosby, M., Beyret, B., Shanahan, M., Hernández-Orallo, J., Cheke, L., and Halina, M. The Animal-AI testbed and competition. apache-2.0 license. volume 123 of *Proceedings of Machine Learning Research*, pp. 164–176. PMLR, 2020. URL https://github.com/beyretb/AnimalAI-Olympics.

Deisenroth, M. P. and Rasmussen, C. E. Pilco: A model-based and data-efficient approach to policy search. In *ICML*, 2011.

Depeweg, S., Hernández-Lobato, J. M., Doshi-Velez, F., and Udluft, S. Learning and policy search in stochastic dynamical systems with bayesian neural networks. *ArXiv*, abs/1605.07127, 2017.

Deverett, B., Faulkner, R., Fortunato, M., Wayne, G., and Leibo, J. Z. Interval timing in deep reinforcement learning agents. *ArXiv*, abs/1905.13469, 2019.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *ArXiv*, abs/1802.01561, 2018.

Feinberg, V., Wan, A., Stoica, I., Jordan, M. I., Gonzalez, J. E., and Levine, S. Model-based value expansion for efficient model-free reinforcement learning. 2018.

Fountas, Z., Sajid, N., Mediano, P. A., and Friston, K. Deep active inference agents using monte-carlo methods. In *NeurIPS*, 2020a.

Fountas, Z., Sylaidi, A., Nikiforou, K., Seth, A. K., Shanahan, M., and Roseboom, W. A predictive processing model of episodic memory and time perception. *bioRxiv*, 2020b.

Friston, K. J. A free energy principle for a particular physics. *arXiv: Neurons and Cognition*, 2019.

Friston, K. J., Rigoli, F., Ognibene, D., Mathys, C., FitzGerald, T. H. B., and Pezzulo, G. Active inference and epistemic value. *Cognitive Neuroscience*, 6:187 – 214, 2015.

Friston, K. J., FitzGerald, T. H. B., Rigoli, F., Schwartenbeck, P., O'Doherty, J., and Pezzulo, G. Active inference and learning. *Neuroscience and Biobehavioral Reviews*, 68:862 – 879, 2016.

Friston, K. J., FitzGerald, T. H. B., Rigoli, F., Schwartenbeck, P., and Pezzulo, G. Active inference: A process theory. *Neural Computation*, 29:1–49, 2017a.

Friston, K. J., Lin, M., Frith, C. D., Pezzulo, G., Hobson, J. A., and Ondobaka, S. Active inference, curiosity and insight. *Neural Computation*, 29:2633–2683, 2017b.

Friston, K. J., Rosch, R. E., Parr, T., Price, C. J., and Bowman, H. Deep temporal models and active inference. *Neuroscience and Biobehavioral Reviews*, 90:486 – 501, 2017c.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Gershman, S. J. and Daw, N. D. Reinforcement learning and episodic memory in humans and animals: An integrative framework. *Annual Review of Psychology*, 68:101–128, 2017.

Goyal, A., Islam, R., Strouse, D., Ahmed, Z., Botvinick, M., Larochelle, H., Levine, S., and Bengio, Y. Infobot: Transfer and exploration via the information bottleneck. *ArXiv*, abs/1901.10902, 2019.

Graves, A., Wayne, G., and Danihelka, I. Neural turing machines. *ArXiv*, abs/1410.5401, 2014.

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S. G., Grefenstette, E., et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476, 2016.

Greenberg, D. L. and Verfaellie, M. Interdependence of episodic and semantic memory: evidence from neuropsychology. *Journal of the International Neuropsychological Society : JINS*, 16 5:748–53, 2010.

Gregor, K. and Besse, F. Temporal difference variational auto-encoder. *ArXiv*, abs/1806.03107, 2019.

Greve, A., Cooper, E., Kaula, A., Anderson, M. C., and Henson, R. N. A. Does prediction error drive one-shot declarative learning? *Journal of Memory and Language*, 94:149 – 165, 2017.

Ha, D. R. and Schmidhuber, J. Recurrent world models facilitate policy evolution. In *NeurIPS*, 2018.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/hafner19a.html.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

Hamrick, J. B. Analogues of mental simulation and imagination in deep learning. *Current Opinion in Behavioral Sciences*, 29:8–16, 2019.

Hansen, S., Sprechmann, P., Pritzel, A., Barreto, A., and Blundell, C. Fast deep reinforcement learning using online adjustments from the past. In *NeurIPS*, 2018.

Hassabis, D., Kumaran, D., Vann, S. D., and Maguire, E. A. Patients with hippocampal amnesia cannot imagine new experiences. *Proceedings of the National Academy of Sciences*, 104:1726 – 1731, 2007.

Hills, T. T. Towards a unified theory of animal event timing. 2003.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

Hung, C.-C., Lillicrap, T., Abramson, J., Wu, Y., Mirza, M., Carnevale, F., Ahuja, A., and Wayne, G. Optimizing agent behavior over long time scales by transporting value. *Nature Communications*, 10, 2019.

Jang, A. I., Nassar, M. R., Dillon, D. G., and Frank, M. J. Positive reward prediction errors strengthen incidental memory encoding. *bioRxiv*, 2018.

Jayaraman, D., Ebert, F., Efros, A. A., and Levine, S. Time-agnostic prediction: Predicting predictable video frames. *ArXiv*, abs/1808.07784, 2019.

Jung, H., Han, M., Kang, M., and Hwang, S. J. Learning what to remember: Long-term episodic memory networks for learning from streaming data. *ArXiv*, abs/1812.04227, 2018.

Kaiser, L., Nachum, O., Roy, A., and Bengio, S. Learning to remember rare events. *ArXiv*, abs/1703.03129, 2017.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Sepassi, R., Tucker, G., and Michalewski, H. Model-based reinforcement learning for atari. *ArXiv*, abs/1903.00374, 2020.

Kalweit, G. and Boedecker, J. Uncertainty-driven imagination for continuous deep reinforcement learning. In *CoRL*, 2017.

Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, S., and George, D. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. *ArXiv*, abs/1706.04317, 2017.

Ke, N. R., Singh, A., Touati, A., Goyal, A., Bengio, Y., Parikh, D., and Batra, D. Learning dynamics model in reinforcement learning by incorporating the long term future. *ArXiv*, abs/1903.01599, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Ko, J., Klein, D. J., Fox, D., and Hähnel, D. Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pp. 742–747, 2007.

Kumar, V., Todorov, E., and Levine, S. Optimal control with learned local models: Application to dexterous manipulation. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 378–383, 2016.

Kurutach, T., Clavera, I., Duan, Y., Tamar, A., and Abbeel, P. Model-ensemble trust-region policy optimization. *ArXiv*, abs/1802.10592, 2018.

Lengyel, M. and Dayan, P. Hippocampal contributions to control: The third way. In *NIPS*, 2007.

Levine, S. and Abbeel, P. Learning neural network policies with guided policy search under unknown dynamics. In *NIPS*, 2014a.

Levine, S. and Abbeel, P. Learning neural network policies with guided policy search under unknown dynamics. In *NIPS*, 2014b.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17:39:1–39:40, 2016.

Mahr, J. B. and Csibra, G. Why do we remember? the communicative function of episodic memory. *The Behavioral and brain sciences*, pp. 1–93, 2017.

McAllister, R. and Rasmussen, C. E. Improving pilco with bayesian neural network dynamics models. 2016.

Michaelian, K. Mental time travel: Episodic memory and our knowledge of the personal past. 2016.

Millidge, B. Deep active inference as variational policy gradients. *ArXiv*, abs/1907.03876, 2019.

Mishra, N., Abbeel, P., and Mordatch, I. Prediction and control with temporal segment models. *ArXiv*, abs/1703.04070, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Nagabandi, A., Kahn, G., Fearing, R. S., and Levine, S. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7559–7566, 2018.

Neitz, A., Parascandolo, G., Bauer, S., and Schölkopf, B. Adaptive skip intervals: Temporal abstraction for recurrent dynamical models. In *NeurIPS*, 2018.

Oh, J., Chockalingam, V., Singh, S. P., and Lee, H. Control of memory, active perception, and action in minecraft. *ArXiv*, abs/1605.09128, 2016.

Pertsch, K., Rybkin, O., Yang, J., Zhou, S., Derpanis, K., Daniilidis, K., Lim, J. J., and Jaegle, A. Keyframing the future: Keyframe discovery for visual prediction and planning. In *L4DC*, 2020.

Pritzel, A., Uria, B., Srinivasan, S., Badia, A. P., Vinyals, O., Hassabis, D., Wierstra, D., and Blundell, C. Neural episodic control. *ArXiv*, abs/1703.01988, 2017.

Racanière, S., Weber, T., Reichert, D. P., Buesing, L., Guez, A., Rezende, D. J., et al. Imagination-augmented agents for deep reinforcement learning. In *NIPS*, 2017.

Roseboom, W., Fountas, Z., Nikiforou, K., Bhowmik, D., Shanahan, M., and Seth, A. Activity in perceptual classification networks as a basis for human subjective time perception. *Nature Communications*, 10, 2019.

Rouhani, N., Norman, K. A., and Niv, Y. Dissociable effects of surprising rewards on learning and memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 44:1430–1443, 2018.

Sajid, N., Ball, P. J., and Friston, K. J. Active inference: demystified and compared. *arXiv: Artificial Intelligence*, 2019.

Schacter, D., Addis, D., Hassabis, D., Martín, V. C., Spreng, R. N., and Szpunar, K. The future of memory: Remembering, imagining, and the brain. *Neuron*, 76:677–694, 2012.

Schacter, D. L., Addis, D. R., and Buckner, R. L. Remembering the past to imagine the future: the prospective brain. *Nature Reviews Neuroscience*, 8:657–661, 2007.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *CoRR*, abs/1511.05952, 2016.

Sherman, M. T., Fountas, Z., Seth, A. K., and Roseboom, W. Accumulation of salient perceptual events predicts subjective time. *bioRxiv*, 2020.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., et al. Mastering the game of go without human knowledge. *Nature*, 550:354–359, 2017.

Tschantz, A., Baltieri, M., Seth, A. K., and Buckley, C. L. Scaling active inference. *ArXiv*, abs/1911.10601, 2019.

Tulving, E. Episodic and semantic memory. 1972.

Ueltzhöffer, K. Deep active inference. *Biological Cybernetics*, 112:547–573, 2018.

Watter, M., Springenberg, J. T., Boedecker, J., and Riedmiller, M. A. Embed to control: A locally linear latent dynamics model for control from raw images. In *NIPS*, 2015.

Zhu, G., Lin, Z., Yang, G., and Zhang, C. Episodic reinforcement learning with associative memory. In *ICLR*, 2020.

Çatal, O., Nauta, J., Verbelen, T., Simoens, P., and Dhoedt, B. Bayesian policy selection using active inference. *ArXiv*, abs/1904.08149, 2019.

A. Preliminaries

A.1. Active Inference

Active inference is a corollary of the free-energy principle applied to action (Friston et al., 2016; Friston, 2019; Sajid et al., 2019). In this framework, an agent embedded in an environment aims to do two things: (i) minimise surprisal from the observations of the environment under the agent's internal model of this environment, and (ii) perform actions so as to minimise the expected surprisal in the future. More formally, an agent is equipped with a generative model $p(o_t, s_t; \theta)$, where o_t is the agent's observation at time t, s_t is the hidden state of the environment, and θ denotes the parameters of the generative model. The agent's surprise at time t is defined as the negative log-likelihood, $-\log p(o_t; \theta)$.

We can upper-bound this intractable expression using variational inference by introducing an approximate posterior distribution, $q(s_t)$, over s_t, such that:

$$-\log p(o_t;\theta) \leq \mathbb{E}_{q(s_t)}\left[\log q(s_t) - \log p(o_t, s_t;\theta)\right] = \mathcal{F}, \tag{6}$$

where $\mathcal{F}$ is the *variational free energy*. The minimisation of this quantity realises objective (i) and is performed by optimising the parameters of the generative model, θ. It is also equivalent to the maximisation of model evidence, which intuitively implies that the agent aims to perfect its generative model at explaining the sensory observations from the environment. To realise objective (ii), the agent must select actions that lead to the lowest *expected* surprise in the future, which can be calculated using the expected free energy (EFE), G:

$$G(\pi,k) = \mathbb{E}_{p(o_k|s_k)}\Big[\underbrace{\mathbb{E}_{q(s_k|\pi)}\left[\log q(s_k|\pi) - \log p(o_k, s_k|\pi)\right]}_{\text{variational free energy, } \mathcal{F}}\Big], \tag{7}$$

where $k > t$ and $\pi = \{a_t, a_{t+1}, ..., a_{k-1}\}$ is a sequence of actions (policy) between the present time t and the future time k. The free-energy minimising system must, therefore, imagine the future observations given a policy and calculate the expected free energy conditioned on taking this policy. Then, actions that led to lower values of the EFE are chosen with higher probability, as opposed to actions that led to higher values of EFE, such that:

$$p(\pi) = \sigma\left(-\gamma G(\pi)\right), \tag{8}$$

where $G(\pi) = \sum_{k>t} G(\pi, k)$, γ is the temperature parameter, $\sigma(\cdot)$ denotes a softmax function, and t is the present timestep.

A.2. Dreamer

As explained in the main body of the paper, Dreamer operates by learning a policy function $q(a_t|s_t;\phi)$ using latent roll-outs. While the generative model (p_θ) is trained to perform accurate reconstructions of the future, the action (q_ϕ) and value (v_ψ) models are trained to maximise rewards (Eq. 1) and predict state values (Eq. 2).

As per the original paper by Hafner et al. (2020), we outline how the value estimate V_λ is calculated. Specifically, given an imagined trajectory of length H, the value estimate is defined such that:

$$V_N^k(s_j) = \mathbb{E}_{p_\theta, q_\phi}\Big[\sum_{n=j}^{h-1} \gamma^{n-j} r_n + \gamma^{h-j} v_\psi(s_h)\Big], \quad \text{where } h = \min(j+k, t+H), \tag{9}$$

$$V_\lambda(s_j) = \mathbb{E}_{p_\theta, q_\phi}\Big[(1-\lambda)\sum_{n=1}^{H-1} \lambda^{n-1} V_N^n(s_j) + \lambda^{H-1} V_N^H(s_j)\Big], \tag{10}$$

where j denotes the timestep from which the roll-out is performed.

B. Architectural Details and Training

B.1. Deep active inference (DAI)

Baseline system In order to ensure the strongest performance of the baseline agent, we have made slight adjustments to the architecture and training procedure of the original DAI agent by Fountas et al. (2020a), which we report next. In DAI, each component of the generative and inference models is parametrised by feed-forward neural networks (including fully-connected, convolutional and transpose-convolutional layers), whose architectural details can be found in Table 2. The latent bottleneck of the autoencoder, s, was of size 10. Similarly, we restricted the action space to 3 – forward, left, right. The networks were trained using separate Adam optimisers (Kingma & Ba, 2015) for stability reasons. The habitual and transition networks are trained with a learning rate of 0.0001; the autoencoder's optimiser had a learning rate of 0.001. The batch size was set to 50 and the model was trained for 1×10^6 iterations under a green observational prior. All of the networks were implemented using Tensorflow v2.2 (Abadi et al., 2015). Tests were performed in Animal-AI v2.0.1 (Beyret et al., 2019).

For the objective-timescale version of DAI, we also employ a *top-down* attention mechanism, as outlined in Fountas et al. (2020a). Introducing variable ω that modulates uncertainty about hidden states is aimed at promoting latent state disentanglement and more efficient learning. Specifically, the latent state distribution is defined as a Gaussian

such that $s \sim \mathcal{N}(s; \boldsymbol{\mu}_s, \boldsymbol{\sigma}_s/\omega)$, where $\boldsymbol{\mu}_s$ and $\boldsymbol{\sigma}_s$ are the mean and the diagonal covariance, and ω is a decreasing logistic function over the divergence $D_{KL}\left[q(a;\phi_a)\|p(a)\right]$. During training, the hyperparameters of the top-down attention mechanism were: $a = 2$, $b = 0.5$, $c = 0.1$, and $d = 5$, chosen to match those in Fountas et al. (2020a).

Each network was trained with its corresponding loss function, which are the constituent parts of the total variational free energy. In particular, the autoencoder was trained using Eqs. 11a and 11b, transition model using Eq. 11b, and habitual network using Eq. 11c:

$$F_t = -\mathbb{E}_{q(s_t)}\left[\log p(o_t|s_t;\theta_o)\right] \tag{11a}$$
$$+ D_{KL}\left[q(s_t;\phi_s)\|p(s_t|s_{t-1},a_{t-1};\theta_s)\right] \tag{11b}$$
$$+ \mathbb{E}_{q(s_t)}\left[D_{KL}\left[q(a_t;\phi_a)\|p(a_t)\right]\right]. \tag{11c}$$

Furthermore, following the training procedure from Fountas et al. (2020a), we stabilise the convergence of the autoencoder by modifying the loss function to:

$$\begin{aligned}\mathcal{L}_{\text{autoencoder}} = &-\mathbb{E}_{q(s_t)}\left[\log p(o_t|s_t;\theta_o)\right] \\ &+ \gamma D_{\text{KL}}\left[q(s_t;\phi_s)||p(s_t|s_{t-1},a_{t-1};\theta_s)\right] \\ &+ (1-\gamma) D_{\text{KL}}\left[q(s_t;\phi_s)||\mathcal{N}(0,I)\right],\end{aligned} \tag{12}$$

where γ is a hyperparameter that gradually increases from 0 to 0.8 during training.

As part of the system's training procedure, we used a replay buffer with random sampling to mitigate the detrimental effects of on-line learning. Full training procedure is shown in Algorithm 2.

Adding STM The STM module replaces the Markovian transition model of the DAI agent. We train the subjective transition model using batch size 15 and a learning rate of 0.001. Each batch consisted of zero-padded S-sequences with length 50. We use a Masking layer to ignore zero-padded parts of the sequences in the computational graph. The training was stopped at approximately 800×10^3 training iterations. The memory formation threshold, ϵ, was set to increase via $\epsilon = \epsilon_{\text{max}}(1 - \exp(-\lambda \cdot i))$, where threshold ceiling is $\epsilon_{\text{max}} = 5$, i is training iteration, $\lambda = \ln 2/t_{1/2}$ with half-life $t_{1/2} = 1 \times 10^5$ training iterations.

B.2. Dreamer

Baseline system For the Dreamer agent, we follow the training details outlined in Hafner et al. (2020), in order to ensure the strongest performance of the objective-timescale baseline. The only minor adjustment included setting action repetition to 1, thus forcing the agent to operate over the objective timescale of the environment. The baseline Dreamer was trained for 1.8M steps in DMLab Sparse Rewards and 2.8M steps in DMLab Maze Rewards.

Algorithm 2 Objective-timescale DAI

Initialise :
  Empty replay buffer, $\mathcal{D} \leftarrow \{\}$
  Random generative and inference model parameters, $\{\theta, \phi\}$
for $episode = 1$ ***to*** ∞ **do**
  Reset and randomise the environment
  for $t = 1$ ***to*** 250 **do**
    Retrieve observation: $\tilde{o}_t$
    Sample and take an action $\tilde{a}_t \sim p(a_t)$ using the planner
    Retrieve next observation: $\tilde{o}_{t+1}$
    Add to replay buffer, $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\tilde{o}_t, \tilde{a}_t, \tilde{o}_{t+1})\}$
    if $\mathcal{D}$ ***sufficient size*** **then**
      Sample a batch of transitions: $\mathcal{B}_i \sim \mathcal{D}$
      For every tuple in $\mathcal{B}_i$, encode the preceding and the subsequent observations, $(\tilde{o}_{t_1}, \tilde{o}_{t_2}) \rightarrow (q(s_{t_1};\phi_s), q(s_{t_2};\phi_s))$
      Compute $q(a_{t_1}|s_{t_1};\phi_a)$ via sampled $\tilde{s}_{t_1} \sim q(s_{t_1};\phi_s)$
      Calculate $p(a_{t_1})$ using the planner
      Train habitual network ▷ Eq. 11c
      Compute ω_{t_2} with the use of Eq. 11c
      Compute next state sufficient statistics $(\boldsymbol{\mu}_{t_2}, \boldsymbol{\sigma}_{t_2})$ using $p(s_{t_2}|s_{t_1}, a_{t_1};\theta_s)$
      Train transition model ▷ Eq. 11b
      Train variational autoencoder ▷ Eqs. 11a, 11b

Adding STM Because the Dreamer already involved a recurrent dynamics model, no changes to the architecture of the Dreamer+STM agent needed to be done. As such, the training procedure with all concomitant hyperparameters were kept the same. The threshold was set to a constant value, $\epsilon = 7$ for Sparse Rewards and $\epsilon = 8$ for Maze Rewards. Dreamer+STM was trained for 1.8M steps in DMLab Sparse Rewards and 3.5M steps in DMLab Maze Rewards.

B.3. Action Heuristic

To train the STM transition model in the AAI and DMLab environments, we implement a simple heuristic that is used to summarise a sequence of actions taken by the agent from one memory to reach the next one. A sequence of actions, $A = \{a_{\tau_1}, a_{\tau_1+1}, ...a_{\tau_1+(N-1)}\}$, takes the agent from a recorded memory s_{τ_1} to memory s_{τ_2}, where the time between these states $\tau_2 - \tau_1 = N$, and $a \in \{a_{\text{forward}}, a_{\text{right}}, a_{\text{left}}\}$. We employ polar coordinates relative to the agent's initial position in Cartesian coordinates at time τ_1 and perform iterative updates of its position after every action until the time-step of the next episodic memory,

Table 2. DAI Architecture

Task	Layer	Size	Activation
Habitual	Linear	128	ReLU
	Linear	128	ReLU
	Linear	128	ReLU
	Linear	128	ReLU
	Linear	3	-
Transition	Linear	512	ReLU
	Linear	512	ReLU
	Linear	512	ReLU
	Linear	512	ReLU
	Linear	20	-
Encoder	Conv2D	32 filters, kernel (3, 3), stride (2, 2)	ReLU
	Conv2D	64 filters, kernel (3, 3), stride (2, 2)	ReLU
	Conv2D	128 filters, kernel (3, 3), stride (2, 2)	ReLU
	Conv2D	256 filters, kernel (3, 3), stride (2, 2)	ReLU
	Linear	256	ReLU
	Linear	256	ReLU
	Linear	256	ReLU
	Linear	20	-
Decoder	Linear	256	ReLU
	Linear	256	ReLU
	Linear	256	ReLU
	Linear	$16^2 * 64$	ReLU
	Conv2DTranspose	256 filters, kernel (3, 3), stride (1, 1)	ReLU
	Conv2DTranspose	128 filters, kernel (3, 3), stride (2, 2)	ReLU
	Conv2DTranspose	64 filters, kernel (3, 3), stride (1, 1)	ReLU
	Conv2DTranspose	3 filters, kernel (3, 3), stride (1, 1)	Sigmoid
STM Transition	Linear	256	ReLU
	LSTM	512	-
	LSTM	512	-
	LSTM	512	-
	LSTM	512	-
	Linear	20	-

τ_2, is reached. Given the agent's orientation in the environment, θ, the next position of the agent given a forward action is calculated using,

$$p_{t+1} = p_t + \begin{bmatrix} \sin\theta \\ \cos\theta \end{bmatrix}, \text{ where } p_t = \begin{bmatrix} 0 \\ 0 \end{bmatrix}_{t=\tau_1} \tag{13}$$

Finally, we retrieve angle ϕ, which describes the direction in which the agent has travelled with respect to its initial position and orientation. This angle is used to decide on the action that summarises the trajectory (see Figure 8) using

$$a = \begin{cases} a_{\text{forward}} & |\phi| \leq 22.5^\circ \\ a_{\text{right}} & 22.5^\circ < \phi < 180^\circ \\ a_{\text{left}} & -22.5^\circ > \phi \geq -180^\circ. \end{cases}$$

Although this heuristic provided satisfactory results, trajectory encoding is one of the most limiting parts of the STM. Nevertheless, we believe such simplistic encoding was sufficient to demonstrate the primary characteristics of the STM – relating to time and prediction saliency. As such, we strongly encourage further research into more sophisticated and generalisable techniques of summarising action sequences.

C. Testing

C.1. Animal-AI Environment

We expect some readers to be unfamiliar with the Animal-AI environment (Beyret et al., 2019), and thus provide a few snapshots from the environment in Figure 9.

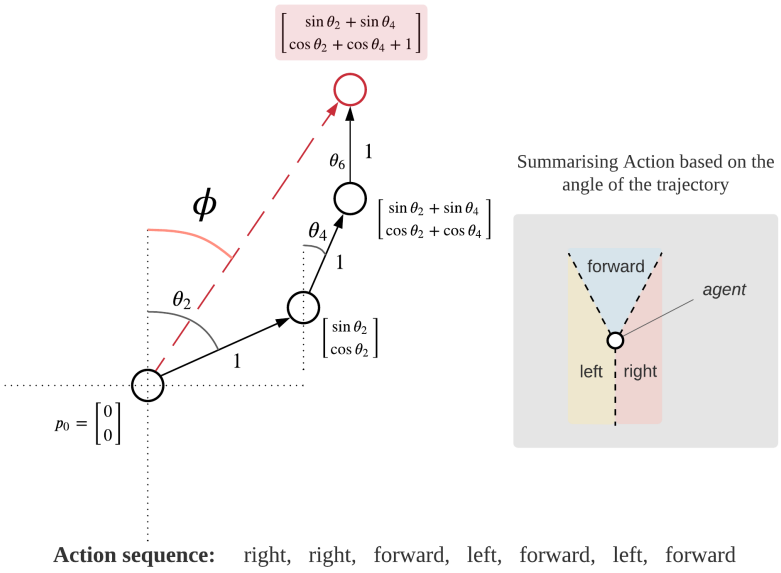


Figure 8. Action encoding for STM.

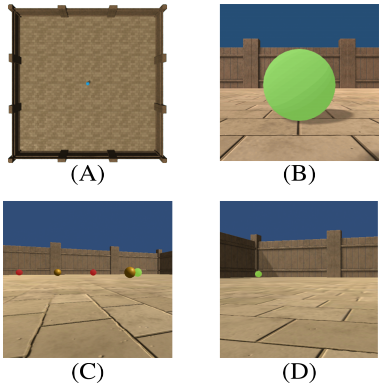


Figure 9. (A) Top view of an empty arena with an agent in the center. (B) Close-up of a green sphere. (C) Green, yellow, and red spheres. (D) View of a green sphere in the corner from the center of the arena.

C.2. Monte Carlo Tree Search of DAI

Following Fountas et al. (2020a), we define the MCTS upper confidence bound as,

$$U(s_t, a_t) = \tilde{G}(s_t, a_t) \; + c_{explore} \cdot q(a_t|s_t;\phi_a) \cdot \frac{1}{N(a_t, s_t)+1}, \tag{14a}$$

where $\tilde{G}(s_t, a_t)$ is agent's estimation of the expected free energy given action a_t from state s_t, $N(a_t, s_t)$ is the number of times a_t was taken from s_t during tree search, and $c_{explore}$ is a tree exploration hyperparameter.

For testing, the MCTS parameters of the baseline and STM were kept the same to ensure a fair comparison, with the exception of the exploration hyperparameter, $c_{explore}$, which was optimised separately for both to yield the best performance of the agents: $c_{explore} = 1$ for STM and $c_{explore} = 0$ for the baseline. Otherwise, both agents used 15 MCTS simulation loops with depth 3.

D. Empirical Trends

As discussed in Section 4.1, prediction error (or free energy) of the transition model was used as the criterion for memory formation. Although the idea is largely inspired by the neuroscience literature on episodic memory and human time perception, it is useful to examine the practical implications of this criterion in the context of model-based agents.

In particular, we consider the necessity for *context-aware* adaptation of the temporal extent of future predictions during planning. We pose that using shorter extents in perceptually-complex scenes is conducive to the agent's ability to navigate difficult or important parts of the environment. In contrast, longer extents should be used in scenes with little perceptual complexity, as the agent plans to get to the objects of its interest. This is further justified by the STM agent's significantly improved performance at avoiding negative rewards.

The chosen criterion allows the STM transition model to learn just that – due to the fact that the frequency of memory formation increases in more complex or rare scenes, as the prediction error gets larger. Empirically, this is demonstrated in Figure 10, in which observations from a replay

buffer were sorted based on the DAI agent's free energy of the transition model. It can be seen that observations with higher free energy contain more objects and, generally, constitute more perceptually-complex scenes.

At the same time, higher values of transition model free energy can be driven by epistemically-surprising events (e.g. sphere entering the agent's view upon turning). Indeed, these types of observations can also be observed in the higher levels of Figure 10. As demonstrated in the paper, such observations appearing in the recorded S-sequences drive the STM to producing more salient predictions of the future, which in turn encourages exploratory behaviour.

E. Additional Results

Collected S-sequences Figure 11 shows additional examples of collected S-sequences in DMLab. In particular, we draw reader's attention to how the model is able to record memories at different temporal rates, in which different segments are approximately denoted with different colours. In visually simple settings, the model tends to skip more frames than in more complex scenes, thus resulting in the STM's ability to vary the extent of its predictions. To better understand STM memory collection, access the following anonymised link, which contains a number of videos of episodes as perceived by the objective- and subjective-timescale agents. https://www.notion.so/Dreamer-STM-Episodes-47692c7680b541d69e13983d04256aec.

STM roll-outs Figure 13 shows random STM roll-outs generated by the system. These diverse roll-outs demonstrate that STM is able to: i) make correct *action-conditioned* predictions, ii) *speed up* its prediction timescale when objects are far away, iii) *slow down* the prediction timescale when objects are nearby. Additionally, for initial observations without any objects in the frame, the agent is capable of imagining their presence in the future.

Exploration heatmaps The provided exploration heatmaps in the main body of the paper were concerned with showcasing the STM's ability to exert more pronounced exploratory behaviour in the presence of rewarding distractions. Here, we additionally demonstrate the exploratory behaviour of Dreamer with and without STM in an empty DMLab environment (in the absence of any rewards) – see Figure 14. Specifically, Figure 14a shows the exploration heatmap comparison, while Figure 14b shows 20 random samples of the different paths that the agents took in an empty arena. We emphasise the considerable differences in the agents' behaviours – STM agent employs a consistent strategy with extensive and persistent exploration of the environment, while the objective-timescale agent resorts to more chaotic behaviour.

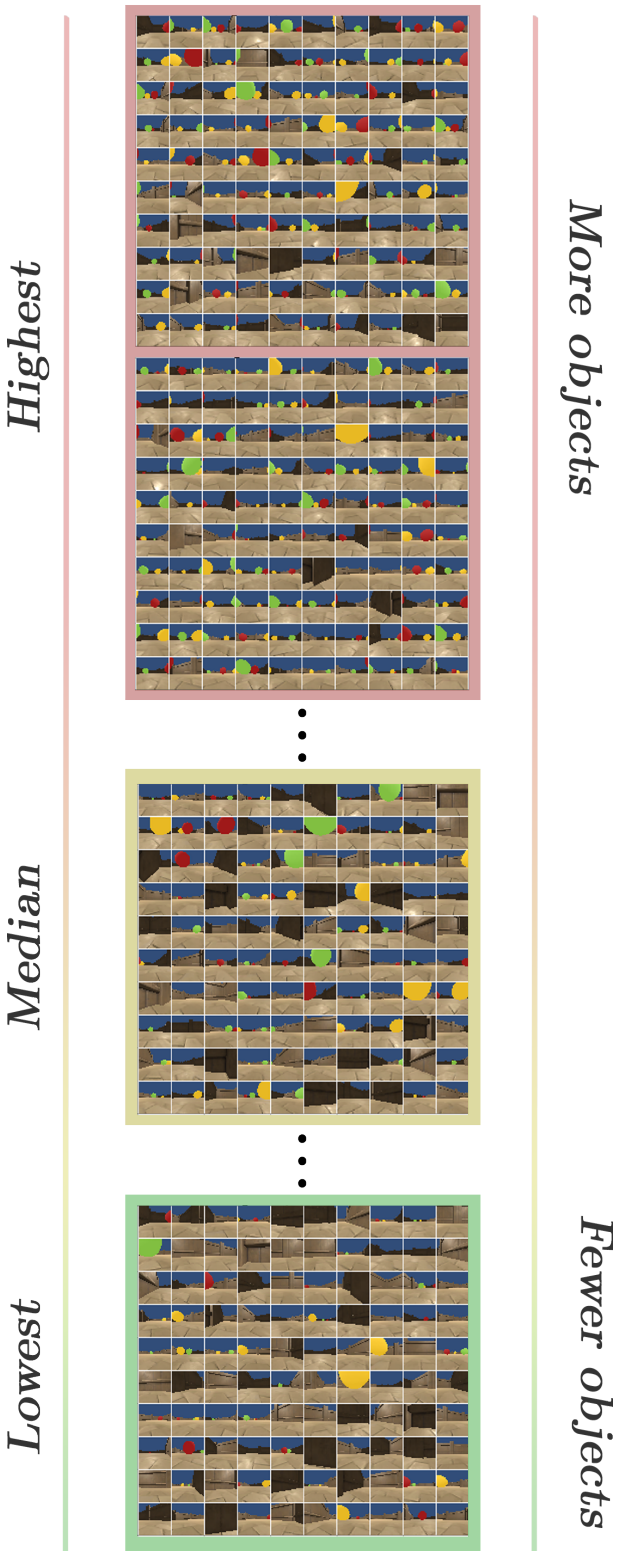


Figure 10. Observations sorted by their corresponding recorded value of the objective-timescale transition model free energy (using DAI agent) in the descending order. States with higher values on average contain more objects, constituting more complex settings.

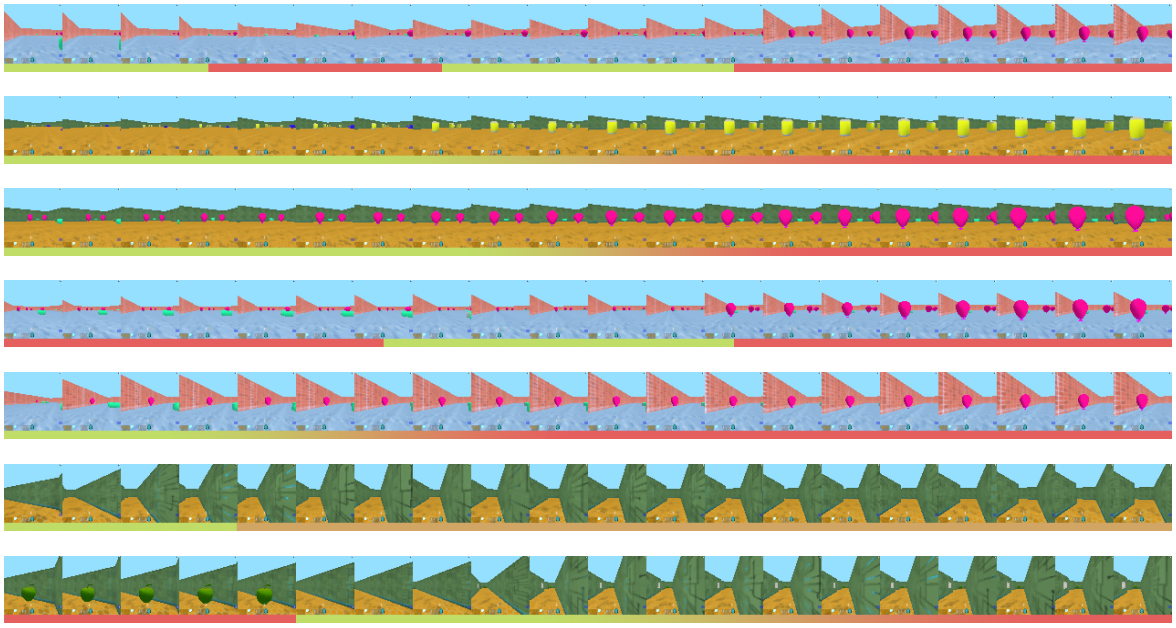


Figure 11. Examples of collected S-sequences in DMLab. Red is used for slower and green for faster parts of the sequences. For better dynamic visualisations of the collected S-sequences, you can access https://www.notion.so/Dreamer-STM-Episodes-47692c7680b541d69e13983d04256aec.

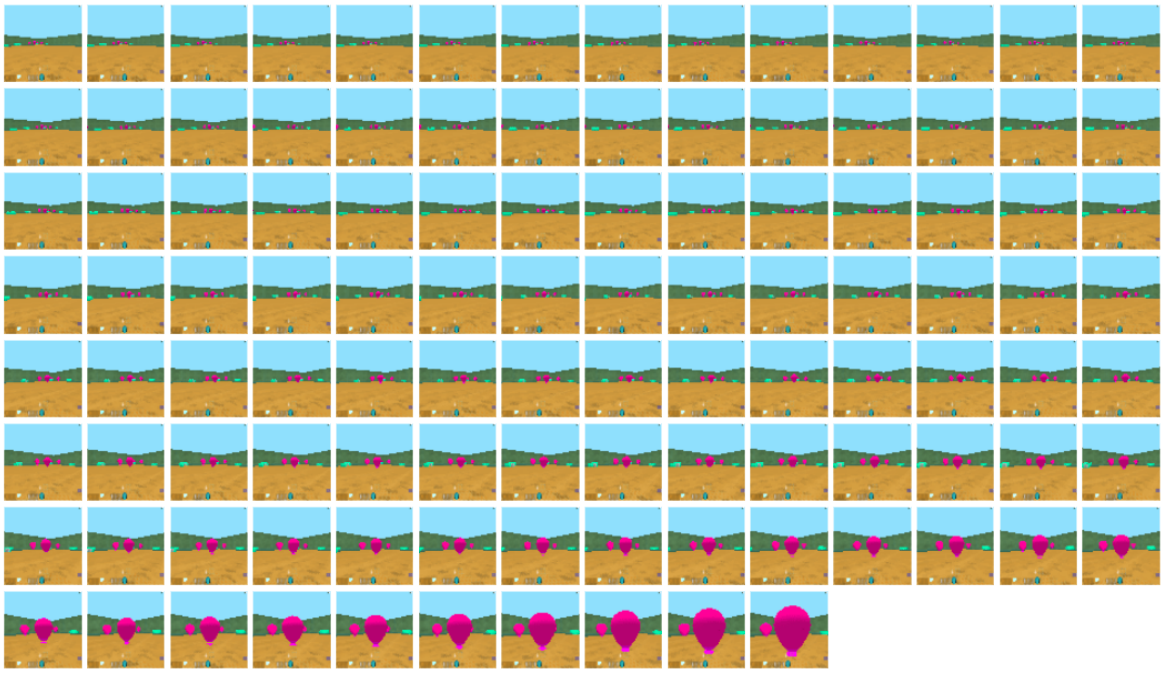


Figure 12. Single example of a recorded sequence using the objective timescale – shown for reference. In contrast to the presented S-sequences, an approach of similar distance takes significantly higher number of episode frames.

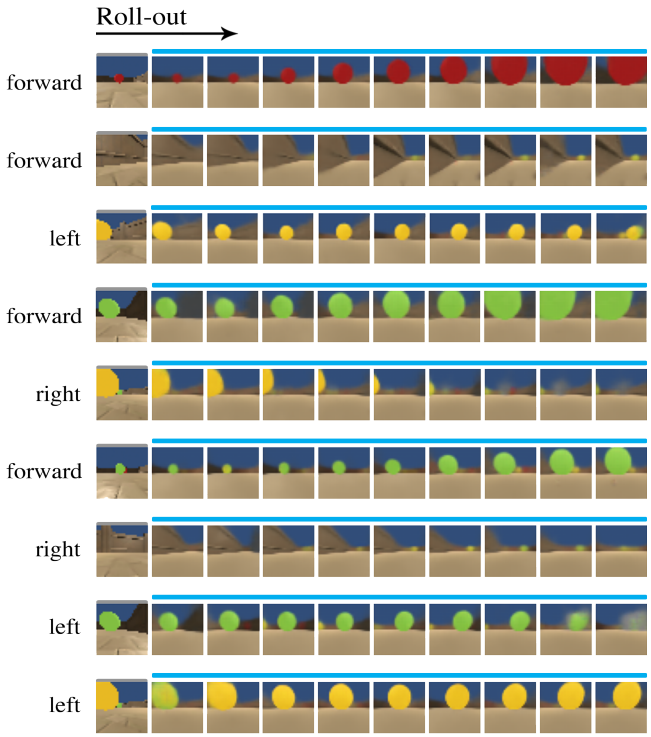
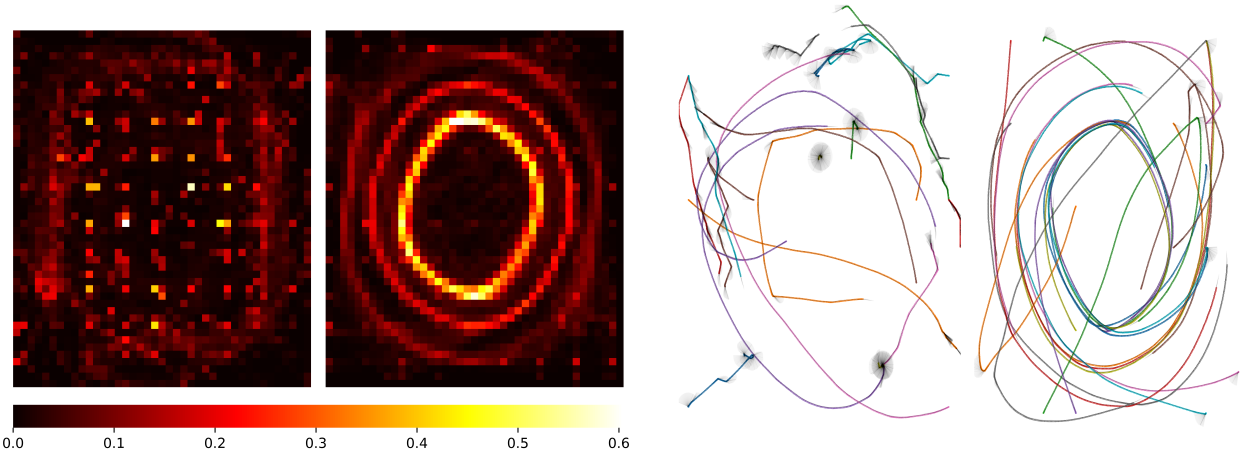


Figure 13. Representative roll-outs generated with the DAI-STM agent which are meant to showcase several important characteristics. Firstly, the agent can successfully model the action-conditioned dynamics of the environment, justifying the appropriateness of the used action heuristic. Secondly, in line with the analysis from the main body of the paper, STM is able to vary the extent of its predictions depending on the perceptual context. For instance, the top roll-out illustrates a rapid approach to a red sphere, while slowing down close to it. This is akin to some of the presented S-sequences in Figure 11, which illustrate a very similar behaviour. Thirdly, in roll-outs with uninformative initial states (such as the ones with the agent facing a wall), STM agent tends to imagine observing an object in the future and with generally increased speed of future predictions.



(a) Exploration heatmap in an empty arena

(b) Samples of Dreamer's exploration paths

Figure 14. (a) Normalised heatmap of visited locations (in an empty map). Dreamer+STM agent (right) exhibits a significantly different behaviour to the objective-timescale Dreamer (left), employing an active and consistent exploratory strategy. (b) Samples of the paths taken by both agents. Dreamer+STM agent (right) employs a clear exploratory strategy, characterised by perpetual loops in an empty environment. On the other hand, objective-timescale Dreamer (left) seems to produce inconsistent and less favourable strategy, in which the agent is more prone to travel in straight lines and not explore beyond its current location.