R-AIF: Solving Sparse-Reward Robotic Tasks from Pixels with Active Inference and World Models Implementation Details Documentation

Viet Dung Nguyen
Rochester Institute of Technology
vn1747@rit.edu

Zhizhuo Yang
Rochester Institute of Technology
zy8981@rit.edu

Christopher L. Buckley
University of Sussex
c.l.buckley@sussex.ac.uk

Alexander Ororbia
Rochester Institute of Technology
ago@cs.rit.edu

A Recurrent State Space Model and World Model

Temporal Information. In active inference, the hidden state inferred by the agent is often computed by a likelihood matrix $\mathbb{R}^{m \times n}$ where m is the number of possible state values and n is the number of possible observation values [6]. A single observation from the environment can then be directly mapped to a state using this scheme. Similarly, in the amortized inference context, the recognition density parameterized by an artificial neural network (ANN) is often used to estimate the posterior probability density over the hidden states of the environment [4]. However, in the POMDP setting, an observation from a single time step would not provide sufficient information about the state as is done in classic active inference literature with a likelihood matrix. For example, higher-order information, such as velocity and acceleration of particular variables, cannot be captured in one single image but instead must be inferred from a sequence of images (or manually integrated [22]). Therefore, it is crucial for every active inference model in POMDP environments to maintain temporal information or deterministic beliefs throughout time as additional information is input into the generative model framework.

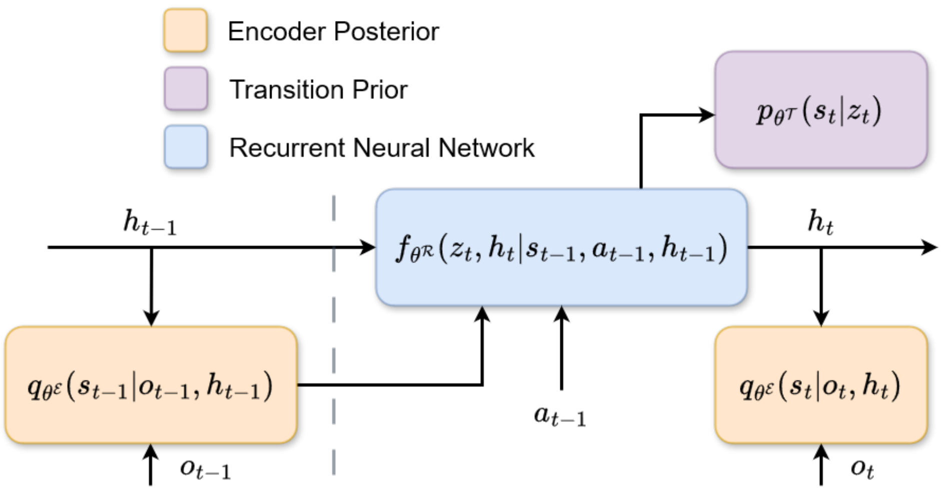
Since active inference posits that an agent finds a policy, i.e, a sequence of actions, based on the estimated future state distribution [8, 27, 7, 5, 17], one approach is to predict the next state s_{t+1} given the current state s_t and action a_t. When operating in POMDP environments, this methodology involves predicting the next partial observation o_{t+1} given the previous partial observation o_t and the action a_t. In order to integrate temporal information into this generative model, we can use the 'carried-over' recurrent state in some forms of RNN such as a gated recurrent unit [2] as was done in [12, 13, 14, 18]. Therefore, the prior $p(s)$ and posterior $q(s)$ distributions over states are able to encapsulate the temporal information h embodied in previous observations ; see Figure 1 for a visual representation of this process.

B Improving Numerical Stability

Minimizing the complexity (term) aids the agent in closing the gap between its prior and its approximate posterior whereas minimizing the accuracy (term) improves the model's future observation estimation. We utilize the world model which has a discretized state space [13] where each hidden state is represented by a vector of discrete distributions instead of a vector of Gaussian distribution parameters as is done in other deep active inference formulations. We also employ the KL balancing [13] and applying "symlog" function to inputs [14] for numerical stability:

$$\begin{aligned} \mathrm{D}_{\mathrm{KL}}\left[q_\theta(s_t|o_t) \parallel p_\theta(s_t|s_{t-1}, a_{t-1})\right] \leftarrow \eta_{\text{rep}} &\mathrm{D}_{\mathrm{KL}}\left[q_\theta(s_t|o_t) \parallel \mathrm{sg}(p_\theta(s_t|s_{t-1}, a_{t-1}))\right] \\ &+ \eta_{\text{dyn}} \mathrm{D}_{\mathrm{KL}}\left[\mathrm{sg}(q_\theta(s_t|o_t)) \parallel p_\theta(s_t|s_{t-1}, a_{t-1})\right] \end{aligned} \tag{1}$$

where sg is the stop gradient operation, η_{rep} and η_{dyn} are the coefficients for representation and dynamics KL losses, respectively. We also clip the KL term to a minimum value of free bits [12] and finally apply the $symlog$ function [14] on its inputs for numerical stability.



Encoder Posterior
Transition Prior
Recurrent Neural Network
$p_{\theta^{\mathcal{T}}}(s_t|z_t)$
h_{t-1}
$f_{\theta^{\mathcal{R}}}(z_t, h_t|s_{t-1}, a_{t-1}, h_{t-1})$
h_t
$q_{\theta^{\mathcal{E}}}(s_{t-1}|o_{t-1}, h_{t-1})$
a_{t-1}
$q_{\theta^{\mathcal{E}}}(s_t|o_t, h_t)$
o_{t-1}
o_t

Figure 1: **The temporal generative dynamics model.** A depiction of the generative model that R-AIF uses to make use of past information; this is equivalent to a RSSM operating in latent space.

C The prior preference self-revision mechanism

Algorithm 1 The Prior *Self-Revision* Mechanism (per Episode)

```
Initialize positive and negative preference rate arrays $m^+$, $m^-$.
$\{p_t\}_{i=1}^T \leftarrow \{p_t \odot k\}_{i=1}^T$                ▷ Ensure good expert actions
// Compute positive preference rate
$c \leftarrow \max(p_T, 0)$                          ▷ Initialize carry variable
for $t \in \{T..1\}$ do
    $c \leftarrow \max(c \odot \alpha, d_t)$               ▷ Discount positive rate
    $c \leftarrow c \odot (c \geq \epsilon)$               ▷ Quantize to 0 if smaller than $\epsilon$
    $m^+ \leftarrow \{c\} \cup m^+$                        ▷ Append to front
end for
$m^+ \leftarrow \{\text{clip}(p + m_t^+, 0, 1)\}_{t=1}^T$          ▷ Value expert states more
// Compute negative preference rate
$c \leftarrow -1/\beta$                              ▷ Initialize carry variable
for $t \in \{T..1\}$ do
    $c \leftarrow c \odot \beta$                           ▷ Decay backward from the episode end
    $m^- \leftarrow \{c\} \cup m^-$                        ▷ Append to front
end for
$m^- \leftarrow \{m_t^- \odot (1-k)\}_{t=1}^T$         ▷ No negative rate when successful
$m^- \leftarrow \{m_t^- \odot (1-p_t)\}_{t=1}^T$       ▷ No negative rate in expert steps
$\rho \leftarrow \{0\} \cup \{\text{clip}(m_t^- + m_t^+, 0, 1)\}_{t=2}^T$     ▷ First state is neutral
return $\rho$
```

Prepare episode data:
Successful at least once boolean k.
Successful step boolean d_t.
Step is taken by the expert boolean p_t.
Episode length T.

Hyperparameters:
Successful discount rate α.
Failure decay rate β.
Positive signal quantization threshold ϵ.

For each step in a trajectory e, we record the following statistics: **1)** a boolean showing whether the step was carried out by an expert p_t or not, **2)** a boolean representing whether the agent immediately achieved the goal at a step d_t, and **3)** a boolean indicating whether the agent succeeded in reaching its goal at least one time within the episode k. For each time step, we want to craft a decaying signal that emphasizes the degree of preference $\rho_t \in [-1, 1]$. In order to do so, we equip the agent with a *self-revision* mechanism which allows the agent to "look back" on how it performed and determine whether a certain state is preferred or not (see Algorithm 1). Note that we complete the for-loop for positive samples and set the preference at its highest value at the state and decay this backward whenever there is a successful state. In contrast, when the episode fails, the agent only needs to decay negatively backward from the end of the episode.

D Computing the Information Gain using a Network Ensemble

Taking actions that reduce the uncertainty of model parameters requires estimating the uncertainty in the first place. One can compute such a term from an explicit ANN ensemble or by using Monte Carlo dropout [10] to compute information gain [4]; however, as the world model grows in size/complexity, it becomes impractical to maintain a collection of multiple world models or to sample from a large state distribution. Therefore, we instead construct a separate ensemble of small multi-layer perceptrons (MLPs) [16, 11] to estimate the next state based on the current state and action – this is the "information gain network ensemble." Formally, we learn a collection of N transition prior models $\{p_{\omega_i}(s_t|s_{t-1}, a_{t-1})\}_{i=1}^{N}$. We next optimize the network ensemble by minimizing the negative log likelihood between the predicted state distribution (Gaussian) and the actual state produced by the world model. In essence, we engage in maximum likelihood estimation and update ensemble parameters by maximizing Gaussian likelihood for a predicted state Gaussian with $\mu_{\omega_i}, \sigma_{\omega_i} \in \mathbb{R}^D$ and an observed next state s_θ from the world model. This can be formally stated as:

$$\arg\min_{\omega_i} \mathcal{L}(\omega_i) = -\left(-\frac{D}{2}\ln(2\pi\sigma_{\omega_i}^2) - \sum_{j=1}^{D}\left(\frac{(s_{\theta,j} - \mu_{\omega_i})^2}{2\sigma_{\omega_i}^2}\right)\right). \quad (2)$$

When training the actor π_ψ and the value network f_χ, we may compute the information gain by computing the difference between the entropy of the mixture-average of the underlying Gaussian and the average of the entropy of all Gaussian outputs from the information gain network ensemble at each rolled-out step τ in imagination space. This is also the main reason why a collection of models are used instead of a single one as was done in [26, 29]. Given the Gaussian entropy formula $\text{ent}(\sigma) = \frac{\ln(2\pi\sigma^2)}{2} + \frac{1}{2}$, we formulate the information gain (or the uncertainty associated with model parameters) estimator in the following way:

$$\text{IG} = \text{ent}\left(\text{std}\left(\{s_i\}_{i=1}^{N}\right)\right) - \mathbb{E}_i\left[\text{ent}(\sigma_{\omega_i})\right], s_i \sim \mathcal{N}(\mu_{\omega_i}, \sigma_{\omega_i}). \quad (3)$$

E Using Percentage Exponential Moving Average (PEMA)

We introduce and set the coefficient of the generative world model entropy ζ and actor distribution entropy $\eta = 3 \times 10^{-4}$ and perform percentile exponential moving average normalization as in [14]; these coefficients depend on both the reward value [14] and model parameters, and therefore, given that they would be impractical to dynamically adjust, we choose to keep ζ and η fixed and small enough for numerical stability. As a result, normalizing the return to the range between 0 and 1 can align with ζ and η range [14]. We then divide the difference between the return and the computed value by the range S as in [14]

$$\text{PEMA}(\mathcal{G}_\tau, f_\chi(v_\tau|s_\tau)) = \big(\mathcal{G}_\tau - f_\chi(v_\tau|s_\tau)\big)/\max\big(1, S\big), \quad (4)$$

where S is computed using percentile (Per) exponential moving average (EMA) [14]:

$$S = \text{EMA}(\text{Per}(\mathcal{G}_\tau, 95) - \text{Per}(\mathcal{G}_\tau, 5), 99). \quad (5)$$

F R-AIF Agent Algorithm

See implementation details in Algorithm 2.

G Implementation of environments

Pixel-level Mountain Car. The mountain car environment [28] is a standard testing problem in reinforcement learning in which an agent (the car) has to drive uphill. A difficult aspect of this problem is that the gravitational force is greater than full force that can be exerted by the car such that it cannot go uphill by simply moving forward. The agent has to learn to build up the car's potential energy by driving to the opposite hill (behind it) in order to create enough acceleration to reach the goal state in front of it. The original mountain car problem was proposed as an MDP where the environment state included the exact position and velocity of the car at any step in time. In the POMDP extension of the task, agents are not permitted to use this state directly. Instead, an agent must use rendered pictures (height 64, width 64) as its observations (and must infer useful internal states that aid it in its completion of the taks). Critically, the reward signal provided by this task is very sparse, i.e., it is -1 for every step and 0 when the agent reaches the goal, and the action space is continuous. In the actual environment, we modify it slightly to provide a dark background and light objects, e.g., white car, to improve visualization for human experimenters.

Meta-World. In this environment, the agent (a controllable robotic arm) has to control the proprioceptive joints (represented as a vector of continuous actions) in a velocity-based control system [30]. Since the workspace is 3D, an observation from a single viewpoint might cause the agent to struggle when inferring environment hidden

Algorithm 2 R-AIF Agent Algorithm

```
Initialize $\theta, \phi, \psi, \chi, \{\omega_i\}_{i=1}^N$
Initialize $E$, $\mathcal{M}$, and $\mathcal{M}^+$
Collect a small number of successful trajectories for $\mathcal{M}^+$
while total environment steps < max environment total steps do
    for environment step $t$ in $1..T$ do
        $s_t \sim q_\theta(s_t|s_{t-1}, a_{t-1}, o_t)$
        $a_t \sim \pi_\psi(a_t|s_t)$
        if done then
            Compute $\rho$ from Algorithm 1
            $\mathcal{M} \leftarrow \mathcal{M} \cup \{(o_t, a_t, r_t, \rho_t)\}_{t=1}^T$
            if successful at least once step then
                $\mathcal{M}^+ \leftarrow \mathcal{M}^+ \cup \{(o_t, a_t, r_t, \rho_t)\}_{t=1}^T$
            end if
        end if
    end for
    for gradient step in $1..S$ do
        if $i \mod 2 = 0$ then
            $\{(o_t, a_t, r_t, \rho_t)\}_t^{t+L} \sim \mathcal{M}$          ▷ Draw normal buffer
        else
            $\{(o_t, a_t, r_t, \rho_t)\}_t^{t+L} \sim \mathcal{M}^+$         ▷ Draw positive buffer
        end if
        $\theta \leftarrow \theta - \xi\nabla_\theta\mathbb{E}\big[\mathcal{L}_t(\theta)\big]$
        $\phi \leftarrow \phi - \xi\nabla_\phi\mathbb{E}\big[\mathcal{L}_t(\phi)\big]$
        $\{\omega_i\}_{i=1}^N \leftarrow \{\omega_i - \xi\nabla_{\omega_i}\mathbb{E}\big[\mathcal{L}_t(\omega_i)\big]\}_{i=1}^N$
        Imagine trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^H$ from each $s_t$
        Imagine prior preference $\{\hat{s}_\tau\}_{\tau=t}^H$ from each $s_t$
        Predict $q_\theta(r_\tau|s_\tau), f_\chi(v_\tau|s_\tau)$
        Compute information gain $\text{IG}_\tau$
        Compute target advantage value $\mathcal{G}_\tau$
        $\chi \leftarrow \chi - \xi\nabla_\chi\mathbb{E}\big[\mathcal{L}_\tau(\chi)\big]$
        $\psi \leftarrow \psi + \xi\nabla_\psi\mathbb{E}\big[\mathcal{L}_\tau(\psi)\big]$
    end for
end while
```

Prepare models' parameters:
Generative world model θ.
CRSPP model ϕ.
Policy network ψ.
Value function χ.
Information gain ensemble $\{\omega_i\}_{i=1}^N$.

Prepare other components:
Environment E.
Memory buffer $\mathcal{M}$.
Positive memory buffer $\mathcal{M}^+$.

Hyperparameters:
Gradient update steps S.
Imagination horizon H.
Batch size B.
Sequence length L.
Learning rate ξ.
Ensemble size N.

states, e.g., the height of the goal can never be inferred from the top-down view. Therefore, we construct a raw pixel observation image using three different (camera) viewpoints: these include a top-down, an agent workspace, and a side camera view. Furthermore, we ensure that the reward space for the tasks in this environment are sparse, similar in form to the sparse signals produced by the mountain car environment with a reward of 0 provided when the agent achieves the control objective (it reaches a successful state) and -1 everywhere else. Note that, unlike the mountain car, the environment simulation continues even after the agent reaches the goal state.

robosuite. Similar to the Meta-World environment, robosuite [31] simulates a robotics environment where the agent controls different joints as continuous actions. Since the robot's workspace in robosuite is larger, we utilized four different camera viewpoints as streams of pixel observations for the agent instead of three as we did in Meta-World, namely: bird's-eye view (similar to the top-down view in Meta-World), agent view (the agent's perspective of the workspace), side view, and front view. Additionally, we modified the environment to have a sparse reward system, yielding a reward of 1 when the agent successfully achieves the task goal, and 0 otherwise. Finally, the environment continues even after the agent successfully reaches the goal state.

H Derivation of Expected Free Energy

According to [25, 4, 9], the expected free energy given the planned action distribution π can be formulated as:

$$G_\tau(\pi) = \mathbb{E}_{\tilde{Q}}[\ln Q(s_\tau, \theta|\pi) - \ln Q(o_\tau, s_\tau, \theta|\pi)] \tag{6}$$

with $\tilde{Q} = Q(o_\tau, s_\tau, \theta|\pi)$ the joint probability of future observation, state, and model parameter given planned action π. This expected free energy formula can be further decomposed into:

$$G(\pi,\tau) = \mathbb{E}_{\tilde{Q}}[\ln Q(\theta|\pi) - \ln Q(\theta|o_\tau, s_\tau, \pi)] + \mathbb{E}_{\tilde{Q}}[\ln Q(s_\tau|\theta,\pi) - \ln Q(s_\tau|o_\tau,\pi)] - \mathbb{E}_{\tilde{Q}}[\ln P(o_\tau)] \quad (7)$$

with the first term is the model parameter exploration, denoting the mutual information between the model parameter before and after making an observation and state. The second term is the mutual information of agent's hidden state before and after making a new observation. The third term is realizing preference term where the agent tries to compute the amount of information about the observation that matches its prior preference.

For ease of computation, we decompose each terms into a form that could be implemented practically by our active inference agent. Firstly, we can further decompose the parameter exploration term:

$$\begin{aligned}
\mathbb{E}_{\tilde{Q}}[\ln Q(\theta|\pi) - \ln Q(\theta|o_\tau, s_\tau, \pi)] &= \mathbb{E}_{\tilde{Q}}[\ln Q(\theta|\pi) + \ln Q(\pi|o_\tau, s_\tau) + \ln Q(o_\tau, s_\tau) - \ln Q(\theta, \pi, o_\tau, s_\tau)] \\
&= \mathbb{E}_{\tilde{Q}}[\ln Q(\pi|o_\tau, s_\tau) + \ln Q(o_\tau, s_\tau) - \ln Q(o_\tau, s_\tau|\theta,\pi) - \ln Q(\pi)] \\
&= \mathbb{E}_{\tilde{Q}}[\ln Q(\pi|o_\tau, s_\tau) - \ln Q(\pi)] + \mathbb{E}_{\tilde{Q}}[\ln Q(o_\tau, s_\tau) - \ln Q(o_\tau, s_\tau|\theta,\pi)] \\
&= \mathbb{E}_{\tilde{Q}}[\ln Q(\pi|o_\tau, s_\tau) - \ln Q(\pi)] \\
&\quad + \mathbb{E}_{\tilde{Q}}[\ln Q(o_\tau|s_\tau) - \ln Q(o_\tau|s_\tau,\theta,\pi)] \\
&\quad + \mathbb{E}_{\tilde{Q}}[\ln Q(s_\tau) - \ln Q(s_\tau|\theta,\pi)] \\
&= \mathbb{E}_{Q(\theta|o_\tau,s_\tau,\pi)Q(o_\tau,s_\tau|\pi)}[\ln Q(\pi|o_\tau,s_\tau) - \mathbb{E}_{Q(o_\tau,s_\tau)} \ln Q(\pi|o_\tau,s_\tau)] \\
&\quad + \mathbb{E}_{Q(o_\tau|s_\tau,\theta,\pi)Q(s_\tau|\theta,\pi)Q(\theta|\pi)}[\mathbb{E}_{Q(\theta,\pi)} \ln Q(o_\tau|s_\tau,\theta,\pi) - \ln Q(o_\tau|s_\tau,\theta,\pi)] \\
&\quad + \mathbb{E}_{Q(o_\tau|s_\tau,\theta,\pi)Q(s_\tau|\theta,\pi)Q(\theta|\pi)}[\mathbb{E}_{Q(\theta,\pi)} \ln Q(s_\tau|\theta,\pi) - \ln Q(s_\tau|\theta,\pi)] \\
&= \mathbb{E}_{Q(\theta|o_\tau,s_\tau,\pi)}\mathbf{H}[Q(\pi|o_\tau,s_\tau)] - \mathbf{H}[\mathbb{E}_{Q(o_\tau,s_\tau)}Q(\pi|o_\tau,s_\tau)] \\
&\quad + \mathbb{E}_{Q(s_\tau|\theta,\pi)}\mathbf{H}[\mathbb{E}_{Q(\theta,\pi)}Q(o_\tau|s_\tau,\theta,\pi)] - \mathbb{E}_{Q(s_\tau|\theta,\pi)Q(\theta|\pi)}\mathbf{H}[Q(o_\tau|s_\tau,\theta,\pi)] \\
&\quad + \mathbf{H}[\mathbb{E}_{Q(\theta,\pi)}Q(s_\tau|\theta,\pi)] - \mathbb{E}_{Q(\theta|\pi)}\mathbf{H}[Q(s_\tau|\theta,\pi)] \\
&\approx \mathbb{E}_{Q(\theta|o_\tau,s_\tau,\pi)}\mathbf{H}[Q(\pi|o_\tau,s_\tau)] - \mathbf{H}[\mathbb{E}_{Q(o_\tau,s_\tau)}Q(\pi|o_\tau,s_\tau)] \\
&\quad + \mathbf{H}[\mathbb{E}_{Q(\theta,\pi)}Q(s_\tau|\theta,\pi)] - \mathbb{E}_{Q(\theta|\pi)}\mathbf{H}[Q(s_\tau|\theta,\pi)]
\end{aligned} \quad (8)$$

where the term $\mathbb{E}_{Q(\theta|o_\tau,s_\tau,\pi)}\mathbf{H}[\ln Q(\pi|o_\tau,s_\tau)]$ is defined as entropy of the predicted action distribution for each estimated future state and observation, whereas the term $\mathbf{H}[\mathbb{E}_{Q(o_\tau,s_\tau)}Q(\pi|o_\tau,s_\tau)$ is defined as the entropy of the policy in the Gaussian mixture averaging all possible future observation and states. In our experiment, we minimize $-\mathbf{H}[\mathbb{E}_{Q(o_\tau,s_\tau)}Q(\pi|o_\tau,s_\tau)$ similar to the reinforcement learning literature where the agent get some intrinsic reward when the policy entropy increases. The term $\mathbb{E}_{Q(s_\tau|\theta,\pi)}\mathbf{H}[\mathbb{E}_{Q(\theta,\pi)}Q(o_\tau|s_\tau,\theta,\pi)] - \mathbb{E}_{Q(s_\tau|\theta,\pi)Q(\theta|\pi)}\mathbf{H}[Q(o_\tau|s_\tau,\theta,\pi)]$ is "the parameter exploration or active learning" term [25, 9] in predicting the future observation, and the term $\mathbf{H}[\mathbb{E}_{Q(\theta,\pi)}Q(s_\tau|\theta,\pi)] - \mathbb{E}_{Q(\theta|\pi)}\mathbf{H}[Q(s_\tau|\theta,\pi)]$ is the "parameter exploration or active learning" term in predicting future states. Note that, since we do not specifically compute/rollout future observations due to impracticality, the parameter exploration over future observation is omitted.

Secondly, we decompose the second term of the expected free energy function as followed:

$$\begin{aligned}
\mathbb{E}_{\tilde{Q}}[\ln Q(s_\tau|\theta,\pi) - \ln Q(s_\tau|o_\tau,\pi)] &= \mathbb{E}_{Q(o_\tau,s_\tau,\theta|\pi)}[\ln Q(s_\tau|\theta,\pi) - \ln Q(s_\tau|o_\tau,\pi)] \\
&= \mathbb{E}_{Q(o_\tau|s_\tau,\theta,\pi)Q(\theta|\pi)}\mathbf{H}[Q(s_\tau|\theta,\pi)] - \mathbb{E}_{Q(\theta|o_\tau,s_\tau,\pi)Q(o_\tau|\pi)}\mathbf{H}[Q(s_\tau|o_\tau,\pi)].
\end{aligned} \quad (9)$$

This term can be explained as the "hidden state exploration or active inference" [25, 9], which can be defined as the entropy of future prior state distribution rolled out from the world model minus the entropy of future state posterior estimated from future predicted observation. In our experiment, since we only roll-out from the state, and not observation, we can omit the term $-\mathbb{E}_{Q(\theta|o_\tau,s_\tau,\pi)Q(o_\tau|\pi)}\mathbf{H}[Q(s_\tau|o_\tau,\pi)]$. Additionally, instead of minimizing the entropy of future states $\mathbb{E}_{Q(\theta|\pi)}\mathbf{H}[Q(s_\tau|\theta,\pi)]$, we maximize it to balance the omitted term and to give the agent more incentive in exploring un-visited states which has higher state entropy. This is also similar to providing agent with more motivation [1, 21, 3] when there is a certain level of uncertainty estimated to be in future states. The "hidden state exploration" can then be approximately equal to $-\mathbb{E}_{Q(\theta|\pi)}\mathbf{H}[Q(s_\tau|\theta,\pi)]$.

Lastly, for the third term, we are maximizing the probability of future predicted observation which match our prior preference at the same time step. Since closer states s_τ eventually leads to closer observation o_τ, we can formulate this instrumental term as:

$$\ln P(o_\tau) \approx r_\tau + \text{ sim}(s_\tau, \hat{s}_\tau) \tag{10}$$

where r_τ is the predicted future reward observation, and $\text{sim}(s_\tau, s^*_\tau)$ is the similarity measure that we have defined in the manuscript.

Overall, we have the full form of the expected free energy formula given the policy π:

$$\begin{aligned} G_\tau(\pi) = & -\mathbf{H}[\mathbb{E}_{Q(o_\tau, s_\tau)} Q(\pi | o_\tau, s_\tau)] \\ & + \mathbf{H}[\mathbb{E}_{Q(\theta,\pi)} Q(s_\tau|\theta, \pi)] - \mathbb{E}_{Q(\theta|\pi)} \mathbf{H}[Q(s_\tau|\theta, \pi)] \\ & - \mathbb{E}_{Q(\theta|\pi)} \mathbf{H}[Q(s_\tau|\theta, \pi)] \\ & - r_\tau - \text{ sim}(s_\tau, \hat{s}_\tau). \end{aligned} \tag{11}$$

I Training R-AIF Agent

In contrast to on-policy learning algorithms [23, 24], we train our agent in an off-policy fashion (using memory buffers). Specifically, we utilize two replay buffers: a standard replay buffer $\mathcal{M}$ stores all agent's encountered transitions, and a positive replay buffer $\mathcal{M}^+$ only contains episodes with at least one step successfully-reached goal state. While training, we sample from these buffers equally as training with more successful samples is found to boost the convergence of CRSPP. We then simulate the agent in the environment and train the world model, CRSPP, information gain ensemble [29], policy network, and value function periodically (see Algorithm 2 for specific details).

J Discussion

Being able to estimate a particular goal or preferred state at any particular time step is very useful for cognitive control agent. As our results demonstrate, the agent can then learn to adapt to take actions that lead from a specific state to its estimated goal(s). In contrast to the approaches taken in the imitation learning and behavior cloning literature, which train the agent's policy based on a fixed collected expert dataset, in our R-AIF framework, the expert signal (the preferred observation) is estimated dynamically through an adaptive prior preference model, closing the domain gap between the actual trajectories and the collected training imitation (preferred) trajectories.

Note that, with respect to our CRSPP sub-module, our model utilizes contrastive objective to adapt its parameters (i.e., it solely learns how to estimate the world model's encoded states while pushing itself away from undesired world model states/trajectories) and thus does *not* require or learn any decoder. We only make use of the learned decoder in the generative world model to visualize the preferred observation from the estimated CRSPP's states. Experimentally, we remark that using an auxiliary decoder proved useful for clearly visualizing the preferred observation $\hat{o}_t$ given the produced preferred state $\hat{s}_t$ at each time step.

Theoretically, behavior cloning will be unable to achieve a great of success in multi-goal environments due to its reliance on a fixed, finite-size imitation sample pool. The expert trajectories in this fixed pool might further differ from the actual trajectories that the agent needs to take to solve the problem at hand, i.e., there is a distributional gap in observations, and therefore require different sets of actions to be taken. On the other hand, CRSPP learns to produce a dynamic goal state while jointly training the agent's core policy to reach task goal states. Therefore, as we empirically confirmed in simulation, R-AIF does not suffer from goal-mismatch problem in the training phase that other AIF schemes would.

Broader Impacts. In line with active inference process theory's focus on optimizing a policy that minimizes future expected free energy, R-AIF agents do so by taking actions that they are able to predict will lead to their preferences (in line AIF's instrumental signal), while also jointly taking actions that they are mostly sure about and working to reduce uncertainty by taking intelligent explorative actions of their niches (in line with AIF's epistemic signals). This is practical for different robotic tasks, particularly those with potential dangers in their operation/functioning (i.e., when human safety must be considered). For example, a self-driving car agent can take the actions that it is sure to be safe rather than focusing exploring wildly (as random exploration policies encourage, potentially causing traffic accidents. Furthermore, the R-AIF framework could prove useful to the imitation learning research community, as its ability to optimize a policy that achieves dynamic goals as produced from an adaptive prior preference model was found to be quite useful for the more complex POMDP tasks we sought to solve in this work.

This carries with it possible positive implications for practical application in downstream tasks such as autonomous driving [19] and complex robotic control and navigation [15, 20].

References

[1] BARTO, A., MIROLLI, M., AND BALDASSARRE, G. Novelty or surprise? *Frontiers in Psychology 4* (2013).

[2] CHUNG, J., ÇAGLAR GÜLÇEHRE, CHO, K., AND BENGIO, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv abs/1412.3555* (2014).

[3] DECI, E. L., AND RYAN, R. M. *Intrinsic Motivation and Self-Determination in Human Behavior*. Springer US, 1985.

[4] FOUNTAS, Z., SAJID, N., MEDIANO, P., AND FRISTON, K. Deep active inference agents using monte-carlo methods. *Advances in neural information processing systems 33* (2020), 11662–11675.

[5] FRISTON, K. Life as we know it. *Journal of the Royal Society, Interface / the Royal Society 10* (06 2013), 20130475.

[6] FRISTON, K., FITZGERALD, T., RIGOLI, F., SCHWARTENBECK, P., AND PEZZULO, G. Active inference: a process theory. *Neural computation 29*, 1 (2017), 1–49.

[7] FRISTON, K., RIGOLI, F., OGNIBENE, D., MATHYS, C., FITZGERALD, T., AND PEZZULO, G. Active inference and epistemic value. *Cognitive neuroscience* (02 2015).

[8] FRISTON, K. J. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience 11* (2010), 127–138.

[9] FRISTON, K. J., LIN, M., FRITH, C. D., PEZZULO, G., HOBSON, J. A., AND ONDOBAKA, S. Active Inference, Curiosity and Insight. *Neural Computation 29*, 10 (10 2017), 2633–2683.

[10] GAL, Y., AND GHAHRAMANI, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning* (New York, New York, USA, 20–22 Jun 2016), M. F. Balcan and K. Q. Weinberger, Eds., vol. 48 of *Proceedings of Machine Learning Research*, PMLR, pp. 1050–1059.

[11] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[12] HAFNER, D., LILLICRAP, T., BA, J., AND NOROUZI, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations* (2020).

[13] HAFNER, D., LILLICRAP, T. P., NOROUZI, M., AND BA, J. Mastering atari with discrete world models. In *International Conference on Learning Representations* (2021).

[14] HAFNER, D., PASUKONIS, J., BA, J., AND LILLICRAP, T. Mastering diverse domains through world models. *ArXiv abs/2301.04104* (2023).

[15] KRAYANI, A., ALAM, A. S., MARCENARO, L., NALLANATHAN, A., AND REGAZZONI, C. A novel resource allocation for anti-jamming in cognitive-uavs: An active inference approach. *IEEE Communications Letters 26*, 10 (2022), 2272–2276.

[16] LECUN, Y., BENGIO, Y., AND HINTON, G. Deep learning. *Nature 521* (05 2015), 436–44.

[17] MILLIDGE, B., TSCHANTZ, A., AND BUCKLEY, C. Whence the expected free energy? *Neural Computation 33* (01 2021), 1–36.

[18] NOEL, A. D., VAN HOOF, C., AND MILLIDGE, B. Online reinforcement learning with sparse rewards through an active inference capsule. *ArXiv abs/2106.02390* (2021).

[19] NOZARI, S., KRAYANI, A., MARIN-PLAZA, P., MARCENARO, L., GÓMEZ, D. M., AND REGAZZONI, C. Active inference integrated with imitation learning for autonomous driving. *IEEE Access 10* (2022), 49738–49756.

[20] OLIVER, G., LANILLOS, P., AND CHENG, G. An empirical study of active inference on a humanoid robot. *IEEE Transactions on Cognitive and Developmental Systems 14*, 2 (2022), 462–471.

[21] OUDEYER, P.-Y., AND KAPLAN, F. What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics 1* (2007).

[22] PARR, T., AND FRISTON, K. J. Generalised free energy and active inference. *Biological cybernetics 113*, 5 (2019), 495–513.

[23] SCHULMAN, J., LEVINE, S., ABBEEL, P., JORDAN, M., AND MORITZ, P. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning* (Lille, France, 07–09 Jul 2015), F. Bach and D. Blei, Eds., vol. 37 of *Proceedings of Machine Learning Research*, PMLR, pp. 1889–1897.

[24] SCHULMAN, J., WOLSKI, F., DHARIWAL, P., RADFORD, A., AND KLIMOV, O. Proximal policy optimization algorithms. *CoRR abs/1707.06347* (2017).

[25] SCHWARTENBECK, P., PASSECKER, J., HAUSER, T. U., FITZGERALD, T. H., KRONBICHLER, M., AND FRISTON, K. J. Computational mechanisms of curiosity and goal-directed exploration. *eLife 8* (may 2019), e41703.

[26] SHYAM, P., JAŚKOWSKI, W., AND GOMEZ, F. J. Model-based active exploration. In *International Conference on Machine Learning* (2018).

[27] SMITH, R., FRISTON, K. J., AND WHYTE, C. J. A step-by-step tutorial on active inference and its application to empirical data. *Journal of Mathematical Psychology 107* (2022), 102632.

[28] SUTTON, R. S., AND BARTO, A. G. *Reinforcement Learning: An Introduction*, second ed. The MIT Press, 2018.

[29] TSCHANTZ, A., MILLIDGE, B., SETH, A. K., AND BUCKLEY, C. L. Reinforcement learning through active inference. *CoRR abs/2002.12636* (2020).

[30] YU, T., QUILLEN, D., HE, Z., JULIAN, R., HAUSMAN, K., FINN, C., AND LEVINE, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)* (2019).

[31] ZHU, Y., WONG, J., MANDLEKAR, A., MARTÍN-MARTÍN, R., JOSHI, A., NASIRIANY, S., AND ZHU, Y. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293* (2020).