DYNAMICAL SYSTEM EMBEDDING FOR EFFICIENT INTRINSICALLY MOTIVATED ARTIFICIAL AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Mutual Information between agent Actions and environment States (MIAS) quantifies the influence of agent on its environment. Recently, it was found that intrinsic motivation in artificial agents emerges from the maximization of MIAS. For example, empowerment is an information-theoretic approach to intrinsic motivation, which has been shown to solve a broad range of standard RL benchmark problems. The estimation of empowerment for arbitrary dynamics is a challenging problem because it relies on the estimation of MIAS. Existing approaches rely on sampling, which have formal limitations, requiring exponentially many samples. In this work, we develop a novel approach for the estimation of empowerment in unknown arbitrary dynamics from visual stimulus only, without sampling for the estimation of MIAS. The core idea is to represent the relation between action sequences and future states by a stochastic dynamical system in latent space, which admits an efficient estimation of MIAS by the "Water-Filling" algorithm from information theory. We construct this embedding with deep neural networks trained on a novel objective function and demonstrate our approach by numerical simulations of non-linear continuous-time dynamical systems. We show that the designed embedding preserves information-theoretic properties of the original dynamics, and enables us to solve the standard AI benchmark problems.

1 INTRODUCTION

Deep Reinforcement Learning (DRL) provides a solid framework for calculating an optimal policy given a reward function, which is provided to the agent by an external expert with specific domain knowledge. This dependency on the expert domain knowledge may restrict the applicability of DRL techniques because, in many domains, it is hard to define an ultimate reward function.

On the other hand, intrinsically-motivated artificial agents do not require external domain knowledge but rather get a reward from interacting with the environment. This motivates the study of intrinsically-motivated AI in general, and to develop efficient intrinsically-motivated methods in particular as an alternative and/or complementary approach to the standard reinforcement learning setting.

In recent works, different intrinsically-motivated approaches were introduced (Klyubin et al., 2005; Klyubin et al., 2005; Wissner-Gross & Freer, 2013; Salge et al., 2013b; Pathak et al., 2017). Among others, the empowerment method reviewed in Salge et al. (2014) is an information-theoretic quantity that proposes to use the diversity of future states distinguishably-achievable by agent as an intrinsic reward. Previously, empowerment has proved to solve a broad range of the standard AI benchmarks (Salge et al., 2014; 2013b; Tiomkin et al., 2017; Klyubin et al., 2005). Despite the empirical success, the computational burden of the estimation of empowerment is significant, because it involves the estimation of mutual information between high-dimensional random variables, which is known to be a hard problem (McAllester & Statos, 2018). The latter is usually done by extensive sampling methods. The computation burden is high even for known dynamics and fully observable state space. This significantly limits the applicability of empowerment to real-life scenarios, ubiquitous in biology and engineering, where an agent observes the environment through sensors (e.g. vision) and does not know an exact dynamical model of the environment.

In this work, we present a novel approach for efficient estimation of empowerment from visual observations (images) in an unknown dynamical environment. The new approach enables us to avoid computationally expensive sampling in high dimensional state/action spaces, which is required for the estimation of MIAS.

The ability to efficiently estimate empowerment from images opens new directions in the study of intrinsic motivation in both biological and engineering systems, where the interaction between the agent and the environment can be represented by a perception-action cycle, schematically represented in Figure 1.

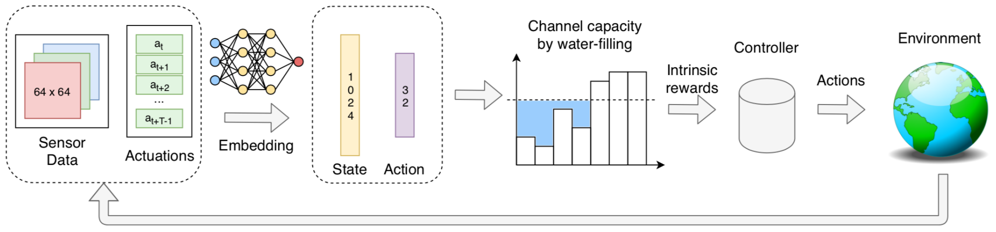


Figure 1: **I**ntrinsically **M**otivated **P**erception and **A**ction **C**ycle. Left-to-right: (a) the observations with the corresponding action sequences are embedded to the latent space by DNNs, where the final states are constrained to be linear in the action sequences (non-linear in the initial states). (b) Gaussian channel capacity between the embedded actions and the embedded final states is computed efficiently by the 'Water-filling' algorithm. (c) an intrinsic reward is derived, triggering the agent to derive a corresponding action, which is injected to the environment. The observations from the environment are perceived by the agent's sensors, closing the loop.

This figure provides a broader perspective of our approach, elucidating our contribution to intrinsically motivated perception action cycle, **IMPAC**, comprising of (a) sensors for information collection, (b) embedding to the latent space, (c) efficient estimation of intrinsic rewards by maximization of MIAS in the latent space, (d) deriving the control signal from intrinsic reward, and (e) applying the control signal to environment by actuators.

While our proposed scheme is broadly applicable, we demonstrate the new scheme in two different domains: intrinsically motivated stabilization of non-linear dynamical systems and intrinsically-safe deep reinforcement learning. Our approach for the estimation of MIAS in the embedded space provides a qualitatively similar MIAS landscape compared to the corresponding one in the original space. We study in details the embedded MIAS of inverted pendulum, which is an important non-linear dynamics and a prototype for upright stabilization in various biological and engineering systems.

The paper is organized as follows: in Section 2 we compare our work with previous relevant works. Especially, table 1 in provides a thorough comparison between existing methods for the estimation of empowerment with regard to essential criteria, and differentiates our work from the existing state of the art. In Section 3 we provide the necessary formal background to present our approach. In Section 4 we present the method, with the implementation described in Section 5. In Section 6 we demonstrate the proposed method by numerical simulations, and compare to the analytical solution derived in the previous work (Salge et al., 2013a). Finally, in Section 7 we conclude, and propose promising continuations of this work.

2 Related works

In this section, we compare our approach to the existing approaches for the estimation of empowerment. The criteria for comparison are: (a) knowledge of the dynamical model, (b) the ability to estimate MIAS from high dimensional visual input, (e.g. images/video) and (c) the need for sampling for the estimation of MIAS. The first criterion is important because the dynamics model is often inaccessible in real-life environments. The second criterion is important in that the interaction (perception-action cycle) between the agent and environment involves the use of sensors (e.g., cameras). The third criterion is important because sampling of high-dimensional data is hard and it is desired to have an alternative approach for the estimation of MIAS. The hardness of the estimation of mutual estimation is a theoretical one, as it requires an exponential number of samples to estimate mutual information (McAllester & Statos, 2018). One of the contributions of this work is an efficient representation scheme, achieved by deep neural networks, which does not require sampling for the estimation of mutual information. This is the first work that makes it possible to estimate empowerment from images with unknown dynamics and without sampling.

Table 1: Comparison of methods for the estimation of empowerment.

Method	Unknown dynamics	Visual input	Sampling
Salge et al. (2013a)	no	no	no
Mohamed & Rezende (2015)	yes	yes	yes
Gregor et al. (2016)	yes	yes	yes
Karl et al. (2017)	yes	no	yes
Tiomkin et al. (2017)	no	no	no
our work	yes	yes	no

As shown at Table 1, the proposed method has the least assumptions, (unknown dynamics, visual input, no sampling for the estimation of MIAS) compared to existing methods, which makes it applicable to a broader class of problems. Also, this approach eliminates the need for externally provided rewards, (which is usually hand-shaped and fine-tuned), in certain classical RL problems.

3 Preliminaries

3.1 Markov Decision Process

A Markov decision process (MDP) is a discrete-time control model with parameters S: the state space, A: the action space, $p(s' \mid s, a)$: the transition probability model, $r(s, a) \in \mathbb{R}$: the reward function, p_0: the initial state distribution, H: the horizon in number of steps and γ: the reward discount factor.

In reinforcement learning, we try to find policy π to maximize the expected sum of returns along a trajectory, $\max_\theta \mathbb{E}\big[\sum_{t=0}^{H-1} r(s_t, a_t)|\pi_\theta\big]$.

3.2 Empowerment

In this work, we use the established definition of empowerment as a function of state (Klyubin et al., 2005). The empowerment at state s_t is the channel capacity between the action sequence $a_t^T \doteq (a_t, a_{t+1}, \dots, a_{T-1})$ and the final state s_T.

$$Emp(s_t) = \max_{\omega(a_t^T \mid s_t)} \mathbf{I}(s_T, a_t^T \mid s_t),$$

where $\mathbf{I}(s_T, a_t^T \mid s_t)$ is the corresponding mutual information functional, and $\omega(\cdot \mid \cdot)$ is a probability distribution of action trajectories conditioned on s_t.

In order to estimate empowerment one needs to estimate information capacity over the probability distributions of action sequences, which is a hard optimization problem.

4 Proposed Approach

The key idea of our approach is to represent the relation between action sequences and corresponding final states by a linear function in the latent space and to construct a Gaussian linear channel in this space. Importantly, even though the relation between action sequences and future states is linear, the dependency of future states on initial states is non-linear. This representation of a non-linear dynamical system enables us to compute empowerment by an efficient “water-filling” algorithm in the latent space, as explained in Section 4.4.

4.1 Interaction Model

We assume that an agent can control a dynamical system, $f : \mathbb{S} \times \mathbb{A} \mapsto \mathbb{S}$, by its actions:

$$s_{t+1} = f(s_t, a_t, \eta_t), \tag{1}$$

where $\mathbb{S}$ is the state space, $\mathbb{A}$ is the action space, s_t is the state at time t, a_t is the action at time t, and η_t is assume to be Gaussian process noise. The agent does not know the dynamical model, f, but rather it observes the current state, s_t, through its visual sensors, providing it with an image ν_t. Then, it derives and applies an action, a_t, and the system moves to the next state, s_{t+1}. The agent has access to the previous state and action trajectories, $\{\tau_i\}_{i=1}^N$, $\tau_i = \{\nu_t^i, a_t^i\}_{t=1}^T$, where N and T are the number of trajectories and the length of each trajectory, respectively.

4.2 Embedded Spaces

From the collected data, $\{\tau_i\}_{i=1}^N$, we embed tuples in the form of (ν_i, a_i^{i+k-1}, ν_{i+k}) where $a_i^{i+k-1} = (a_i, a_{i+1}, \cdots, a_{i+k-1})$. Observations and actions are embedded by deep neural networks to the latent spaces $\mathbb{Z} \in \mathbb{R}^{d_z}$ and $\mathbb{B} \in \mathbb{R}^{d_b}$, respectively. These two spaces are related by a linear equation:

$$z_{t+k} = A(z_t) b_t^{t+k-1}, \tag{2}$$

where z_t is the embedded representation of the image ν_t. $A(z_t)$ is the state dependent matrix that relates the embedded action sequence, b_t^{t+k-1} to the future embedded state, z_{t+k}. This representation was suggested previously by Salge et al. (2013a) in the original state space with known dynamics. The existence of such representation in the embedded space, and a practical method for its calculation from images is one of the contributions of our work. The dimensions of A are $d_z \times d_b$, where d_z is the dimension of the embedded state, and d_b is the dimension of the embedded action sequence, respectively. The architecture for the embedding and the objective function is provided in Section 5

4.3 Information Channel in Embedded Space

To compute mutual information between embedded action sequences, b_t^{t+k-1} and embedded states, z_{t+k}, we inject Gaussian noise, $\eta \sim \mathcal{N}(\mathbf{0}_{d_z \times d_z}, \mathbf{I}_{d_z \times d_z})$, in the embedded space. Finally, we construct the Gaussian linear channel in embedded spaces:

$$z_{t+k} = A(z_t) b_t^{t+k-1} + \eta \tag{3}$$

This formulation of dynamical control systems in the latent space admits an efficient estimation of MIAS as explained in the next section.

4.4 Channel capacity in Latent Space

To compute the information capacity of the channel given in Eq. 4.3, we apply the "Water-Filling" algorithm (Cover & Thomas, 2012), which is a computationally efficient method for the estimation of capacity, C^*, of a linear Gaussian channel given by Eq. 4.3:

$$C^*(z_t) = \max_{p_i} \frac{1}{2} \sum_{i=1}^{n} \log(1 + \sigma_i(z_t) p_i), \tag{4}$$

under the constraint on the total power, $\sum_{i=1}^n p_i = P$, where $\{\sigma_i(z_t)\}_{i=1}^n$ are the singular values of the channel matrix, $A(z_t)$, and P is the total power in the latent space.

5 Architecture

The most crucial component of the IMPAC scheme is the embedding from sensor inputs (in our case, images) and action sequences, into latent state and action tensors while remaining consistent with the real environment dynamics. In this section, we organize our discussion around how a single sample: $\{S_t, a_t, \cdots, a_{t+T-1}, S_{t+T}\}$ fits in the training of our deep neural networks, as illustrated in Figure 2. We will explain each individual components of the block diagram and wrap up by discussing the whole picture.

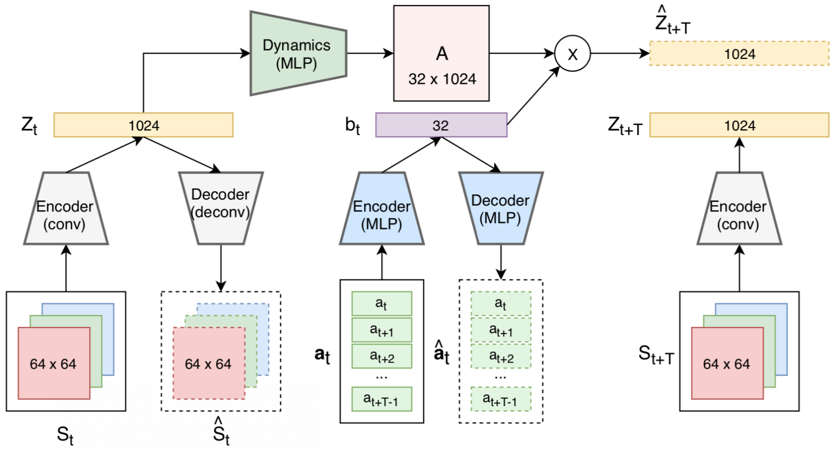


Figure 2: An illustration of data flow around a single training sample.

5.1 Embedding of image observations

The left side and right side of the block diagram in Figure 2 embeds raw observations (images).

In this work, we take images scaled down to 64×64 as observations. The image resolution is chosen as a balance of clarity and performance: visual features of the environments are retained while training batches can be fit on a single GPU. Since one single RGB image is not enough to capture some particular information in the environment (e.g. velocity), we concatenate images from a few consecutive states to form one observation when needed. When encoding multi-channel images into meaningful latent vectors, we use an auto-encoder and decoder pair parameterized by convolutional neural networks. Given a training sample $\{S_t, a_t, \cdots, a_{t+T-1}, S_{t+T}\}$, both S_t and S_{t+T} are encoded into their respective latent vectors Z_t and Z_{t+T} using a same CNN encoder. When we embed images into latent space, we want state information to be retained. Thus, the latent tensor Z_t is decoded back using a deconvolutional network for reconstruction of image $\hat{S}_t$. The objective is to make S_t and $\hat{S}_t$ stay close to each other. The encoder and decoder network details are presented in Appendix B.

5.2 Embedding of action sequences

The mid-lower section of Figure 2 embeds action sequences.

Standard MDPs (Section 3.1) models environment dynamics in the granularity of single steps, which results in an exponentially large action space over long horizons. In this work, we aim to compute information-theoretic quantities across multiple time-steps while avoiding exponential running time. We consider a k-step dynamic model directly by embedding action sequence $\{a_t, a_{t+1} \cdots a_{t+k-1}\}$ into a single action b_t in the latent space (1 step in the latent space corresponds to k steps in the original space). Actions inside the sequence are concatenated into a vector and passed into an auto-encoder to get b_t. Again, to ensure that necessary information is retained in the latent action tensor, we decode b_t for reconstruction of $\{a_t, a_{t+1} \cdots a_{t+k-1}\}$. In our implementation, the encoder and decoder are both fully connected MLPs. The detailed information about each layer can be found in Appendix B.

5.3 Dynamical model in latent space

The top part of Figure 2 fits a dynamics model linear in action (non-linear in state).

As discussed in our IMPAC scheme, the water-filling algorithm computes channel capacities efficiently. But as a requirement, we need to model our dynamics such that given the current state Z_t, the next state Z_{t+T} is an affine transformation of action b_t. Mathematically, if latent states

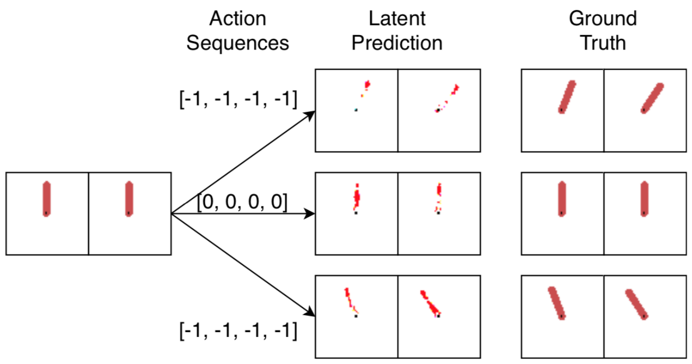


Figure 3: Reconstruction through the latent dynamics in comparison with the ground truth. Starting from the same upright position, a different action sequence is taken in each row. In all 3 cases, reconstruction matches the actual observation after the actions are taken.

Z_t, Z_{t+T} have dimension d_z and latent action b_t have dimension d_b, we want to find a mapping $A : \mathbb{R}^{d_z} \rightarrow \mathbb{R}^{d_z \times d_b}$ such that $Z_{t+T} \approx A(Z_t) \times b$.

The mapping A from Z_t to an $d_z \times b_z$ matrix is parameterized by a fully connected MLP with $d_z \times d_b$ output neurons. The final layer is then reshaped into a matrix. (See appendix B)

5.4 The entire scheme

In the previous 3 sub-sections, we introduce 3 objectives: 1) Find an embedding of observations that retains their information. 2) Find an embedding of action sequences that retain their information. 3) Find a dynamics model linear in the latent action. When these three objectives are trained jointly, the neural networks learn towards a latent space that extract information from the original space, with a constrained dynamics model that is consistent with the actual environment. In practice, regularization on the latent tensors is enforced at training time to encourage a better structured latent space. The details about our choice of regularization are discussed in appendix A.

6 Experiments

6.1 Inverted Pendulum

Inverted pendulum is a prototypical system for upright bi-pedal walking, and it often serves to check new AI algorithms. Moreover, inverted pendulum is a baseline for the estimation of empowerment in dynamical systems because its landscape is well-studied in numerous previous studies (Salge et al., 2013b; 2014; 2013a; Karl et al., 2017; Mohamed & Rezende, 2015). So, we chose inverted pendulum for testing our new method, and comparing it to the previous methods.

6.1.1 Reconstruction through latent dynamics

Prior to applying the proposed model, given by Eq. 4.3, to the estimation of MIAS in the latent space, we need to verify that our latent representation correctly encodes the dynamics of our system. One way to do so is by comparing reconstruction of latent prediction with ground truth observation. In the case of pendulum environment, two consecutive images are used to capture the state information (angle and angular velocity). When we supply the initial two images for time t, along with choices of different action sequences of length k, we expect the reconstructions to match the corresponding observations at time $t + k$. As shown in Figure 3 the model is capable of reconstructing the future images from the current images and action sequences.

6.1.2 Empowerment landscape

In this section, we represent the empowerment landscape computed from images by our new approach, and the optimal trajectory for swinging up the pendulum from the bottom position to the upright position. The optimal trajectory is computed by the standard PPO algorithm with an intrinsic reward given by the values of empowerment (Schulman et al., 2017b). Figure 4 represent the empowerment landscape estimated from images:

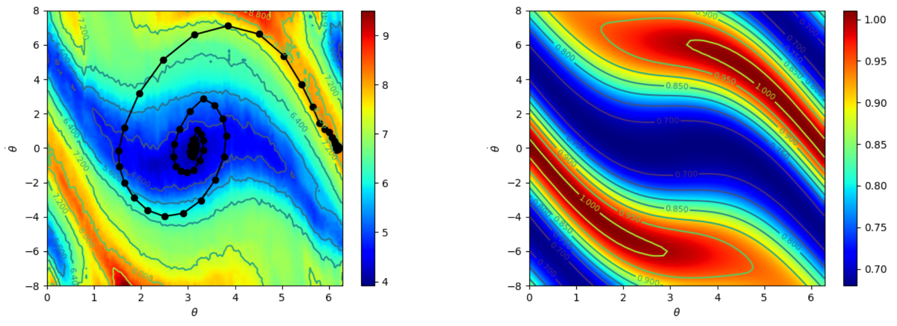


Figure 4: **Left:** Empowerment landscape and the optimal trajectory. **Right:** Analytical solution from Salge et al. (2013a). The x-axis is θ (angle) and the y-axis is $\dot{\theta}$ (angular velocity). The angle is measured from the upright position, ($\theta = 0\,\mathrm{rad}$ corresponds to the upright position). The black curve is the optimal trajectory for swinging up the pendulum, which starts from the bottom with $\theta = \pi\,\mathrm{rad}$ and $\dot{\theta} = 0\,\mathrm{rad}$.

As shown in Figure 4, at the beginning the pendulum swings with a small amplitude (the dense black dots around $\theta = \pi\,\mathrm{rad}$, $\dot{\theta} = 0\,\mathrm{rad\,s^{-1}}$), then, once accumulated enough energy, it starts traversing the state space, and arrives at the top position, where it stabilises, (dense black points around $\theta = 0\,\mathrm{rad}$, $\dot{\theta} = 0\,\mathrm{rad\,s^{-1}}$). Strikingly, the empowerment landscape computed by the proposed method from images coincides with the empowerment landscape computed by an analytical solution (cf. Figure 3 at Salge et al. (2013a)).

6.2 Safety of RL agent

We hypothesize that a state is intrinsically safe for an agent when the agent has a high diversity of future states, achievable by its actions. In this context, the higher its empowerment value, the safer the agent is. In this experiment, we test how empowerment augmented reward function will affect the agent's preference between a shorter but more dangerous path and a longer but safer one.

Environment: a double tunnel environment implemented with to the OpenAI Gym API (Brockman et al., 2016). Agent (blue) is modeled as a ball of radius 1 inside a 20×20 box. The box is separated by a thick wall (gray) into top and bottom section. Two tunnels connect the top and bottom of the box. The tunnel in middle is narrower but closer to the goal compared to the one on the right.

Control: In each time step, the agent can move at most 0.5 unit length in each of x and y direction. If an action takes the agent into the walls, it will be shifted back out to the nearest surface.

Reset criterion: each episode has a maximum length of 200 steps. The environment resets when time runs out or when the agent reaches the goal.

Since the tunnel in the middle is narrower, the control of the agent is damped in 2 ways:

1. In a particular time step, it's more likely for the agent to bump into the walls. When this happens, the agent is unable to proceed as far as desired.
2. The tunnel is 10 units in length. When the agent is around the middle, it will still be inside the tunnel in the 5-step horizon. Thus, the agent has fewer possible future states.

We trained the agent with PPO algorithm (Schulman et al., 2017a) from OpenAI baselines (Dhariwal et al., 2017) using an empowerment augmented reward function. After a parameter search, we used discount factor $\gamma = 0.95$ over a total of 10^6 steps. The reward function that we choose is:

$$R(s, a) = \mathbf{1}_{goal} + \beta \times Emp(s)$$

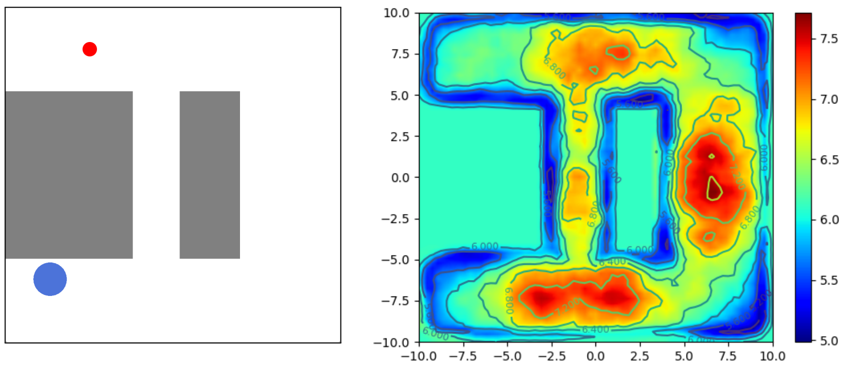
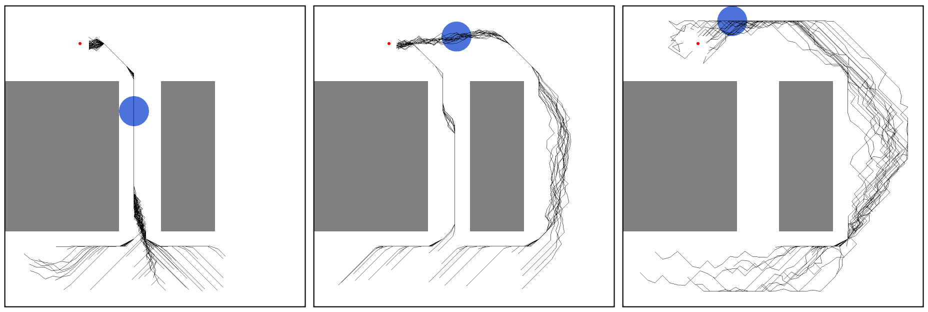


Figure 5: **Left sub-figure**: Double tunnel environment. The goal is marked in red. The agent in blue. **Right sub-figure**: Empowerment landscape for the tunnel environment. The values of empowerment reduce at the corner and inside the tunnel where the control of the agent is less effective compared to more open locations.

where β balances the relative weight between the goal conditioned reward and the intrinsic safety reward. With high β, we expect the agent to learn a more conservative policy.



(a) $\beta = 0$ (b) $\beta = 1/1600$ (c) $\beta = 1/800$

Figure 6: Trajectories of trained policy.

The trajectory plots shown in Figure 6 are generated using the final trained policy of three different β values. For clarity, only trajectories starting from the bottom are shown in the plot. Just as expected, we observe that as β increase, the agent develops a stronger preference over a safer route, sacrificing hitting time.

7 Discussion

Intrinsically motivated artificial agents do not rely on external reward and thus, do not need domain knowledge for solving a broad class of AI benchmark problem such as stabilization, tracking, etc. In this work, we introduce a new method for efficient estimation of a certain type of information-theoretic intrinsic motivation, known as empowerment. Our method estimates of mutual information between action trajectories and final states under the minimal assumptions: estimation from images with unknown dynamics. The solution is based on a particular embedding of states and action sequences, which is constrained to be linear in embedded action sequences, but non-linear in embedded states.

With this work, we also introduced the IMPAC framework, which allows for a systematic study of intrinsically motivated agents in broader scenarios, including real-world physical environments like robotics. Such real-world application of our presented approach might provide an easier and more efficient way to incorporate intrinsic motivation compared to existing methods.

REFERENCES

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. Openai baselines. `https://github.com/openai/baselines`, 2017.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Maximilian Karl, Maximilian Soelch, Philip Becker-Ehmck, Djalel Benbouzid, Patrick van der Smagt, and Justin Bayer. Unsupervised real-time control through variational empowerment, 2017.

A. S. Klyubin, D. Polani, and C. L. Nehaniv. Empowerment: a universal agent-centric measure of control. In *2005 IEEE Congress on Evolutionary Computation*, volume 1, pp. 128–135 Vol.1, Sep. 2005. doi: 10.1109/CEC.2005.1554676.

Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. All else being equal be empowered. In *European Conference on Artificial Life*, pp. 744–753. Springer, 2005.

David McAllester and Karl Statos. Formal limitations on the measurement of mutual information. *arXiv preprint arXiv:1811.04251*, 2018.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pp. 2125–2133, 2015.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 488–489, 2017.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Approximation of empowerment in the continuous domain. *Advances in Complex Systems*, 16(02n03):1250079, 2013a.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment and state-dependent noise-an intrinsic motivation for avoiding unpredictable agents. In *Artificial Life Conference Proceedings 13*, pp. 118–125. MIT Press, 2013b.

Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pp. 67–114. Springer, 2014.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017a.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017b.

Stas Tiomkin, Daniel Polani, and Naftali Tishby. Control capacity of partially observable dynamic systems in continuous time. *arXiv preprint arXiv:1701.04984*, 2017.

Alexander D Wissner-Gross and Cameron E Freer. Causal entropic forces. *Physical review letters*, 110(16):168702, 2013.

A Details of training algorithm

1. Collect trajectories using random policy and random reset.
2. Separate out tuples $(S_t, a_t, a_{t+1}, \cdots, a_{t+T-1}, S_{t+T})$ from trajectories. For simplicity of notation, let a_t^T represent the concatenated action sequences $a_t, a_{t+1}, \cdots, a_{t+T-1}$
3. Use randomly initialized encoder networks to map the observations and action sequence into latent space. This gives us tuples (Z_t, b_t^T, Z_{t+T}).
4. Use randomly initialized MLP network to get the corresponding transformation matrices $A(Z_t)$.
5. Calculate the predicted next state in latent space $\hat{Z}_{t+T} = A(Z_t)b_t^T$
6. Use randomly initialize decoder networks to reconstruct images and action sequences from latent vectors.
   $\tilde{S}_t = Dec(Z_t)$
   $\tilde{a}_t^T = Dec(b_t^T)$
   $\tilde{S}_{t+T} = Dec(\hat{Z}_{t+T})$ Note that $\tilde{S}_{t+T}$ is reconstructed from latent prediction.
7. Calculate the following loss terms:
   (a) Observation reconstruction error: $L_{obs} = ||S_t - \tilde{S}_t||_2^2$
   (b) Action sequence reconstruction error: $L_{action} = ||a_t^T - \tilde{a}_t^T||_2^2$
   (c) Prediction error in latent space: $L_{latent} = ||Z_{t+T} - \hat{Z}_{t+T}||_2^2$
   (d) Prediction error in original space: $L_{org} = ||S_{t+T} - \tilde{S}_{t+T}||_2^2$
8. In additional to the loss terms, we add regularization terms to prevent latent vectors from shrinking. This help us get consistent and comparable empowerment values across different trials and even different environments.
   (a) Regularization of latent state: $Reg_z = |1 - \frac{|Z_t|_2^2}{d_z}|$
   (b) Regularization of latent action: $Reg_b = |1 - \frac{|b_t^T|_2^2}{d_b}|$
9. Finally, we use batched gradient descent on the overall loss function to train all the neural networks in the same loop.
   $$L = \alpha_{obs}L_{obs} + \alpha_{action}L_{action} + \alpha_{latent}L_{latent} + \alpha_{org}L_{org} + \alpha_{reg}(Reg_z + Reg_b)$$
   For both of our experiments, we chose
   $$\alpha_{obs} = \alpha_{org} = 100$$
   $$\alpha_{action} = 10$$
   $$\alpha_{latent} = \alpha_{reg} = 1$$
   and were able to produce desired empowerment plots.

B Details on neural network layers

B.1 Convolutional net for image encoding

(h1) 2D convolution: 4 filters, stride 2, 32 channels, ReLU activation
(h2) 2D convolution: 4 filters, stride 2, 64 channels, ReLU activation
(h3) 2D convolution: 4 filters, stride 2, 128 channel, ReLU activations
(h4) 2D convolution: 4 filters, stride 2, 256 channel, ReLU activations
(out) Flatten each sample to a 1D tensor of length 1024

B.2 Deconvolutional net for image reconstruction

(h1) Fully connected layer with 1024 neurons, ReLU activation
(h1') Reshape to 1×1 images with 1024 channels
(h2) 2D conv-transpose: 5 filters, stride 2, 128 channels, ReLU activation
(h3) 2D convolution: 5 filters, stride 2, 64 channels, ReLU activation
(h4) 2D convolution: 6 filters, stride 2, 32 channel, ReLU activations
(out) 2D convolution: 6 filters, stride 2, C channel

B.3 MLP FOR ACTION SEQUENCE ENCODING

(h1) Fully connected layer with 512 neurons, ReLU activation
(h2) Fully connected layer with 512 neurons, ReLU activation
(h3) Fully connected layer with 512 neurons, ReLU activation
(out) Fully conected layer with 32 output neurons

B.4 MLP FOR ACTION SEQUENCE RECONSTRUCTION

(h1) Fully connected layer with 512 neurons, ReLU activation
(h2) Fully connected layer with 512 neurons, ReLU activation
(out) Fully conected layer with $T \times d_a$ neurons, tanh activation then scaled to action space

B.5 MLP FOR TRANSITION MATRIX A

(h1) Fully connected layer with 1024 neurons, ReLU activation
(h2) Fully connected layer with 4096 neurons, ReLU activation
(h2) Fully connected layer with 8192 neurons, ReLU activation
(out) Fully conected layer with $d_z \times d_b$ (1024×32) neurons