Supplementary Material for *Information-Theoretic World Model learning for Denoised Predictions*

1 Derivations

In this section, we derive equations from the Section "Denoised Predictive Imagination".

1.1 Derivation of Equation (7)

We aim to minimize the Mutual Information (MI) from the beginning to timestep t i.e. $\min I(z_{t^-}; z_t)$. To make our model action dependent, we introduce a conditional probability distribution $p(z_{t^-}, z_t|a_{t^-})$,

$$I(z_1; ...; z_t) = \mathbb{E}_{p(z_1,...,z_t)}\left[\log \frac{p(z_1, ..., z_t)}{\prod_{k=1}^{t} p(z_k)}\right], \tag{14}$$

$$= \mathbb{E}_{p(z_{1:t}, a_{1,t-1})}\left[\log \frac{p(z_{1:t})\, p(z_{1:t}|a_{1:t-1})}{p(z_{1:t}|a_{1:t-1}) \prod_{k=1}^{t} p(z_k)}\right], \tag{15}$$

$$= \mathbb{E}_{p(z_{1:t}, a_{1:t-1})}\left[\log \frac{p(z_{1:t}|a_{1:t-1})}{\prod_{k=1}^{t} p(z_k)}\right] - \mathbb{E}_{p(z_{1:t}, a_{1:t-1})}\left[\log \frac{p(z_{1:t}|a_{1:t-1})}{p(z_{1:t})}\right]. \tag{16}$$

The first term is similar to the variational upper bound introduced in Alemi et al. (2017). The second term is the KL-divergence between $p(z_{1:t}|a_{1:t-1})$ and $p(z_{1:t})$. Since the KL-divergence is always non-negative, the first term in the equation provides an upper bound on the MI objective we seek to optimize i.e.,

$$I(z_{1:t}) \leq \mathbb{E}_{p(z_{1:t}, a_{1:t-1})}\left[\log \frac{p(z_{1:t}|a_{1:t-1})}{\prod_{k=1}^{t} p(z_k)}\right]. \tag{17}$$

We can write the conditional distribution $p(z_{1:t}|a_{1:t-1})$ in its autoregressive form,

$$\begin{aligned} p(z_{1:t}|a_{1:t-1}) &= p(z_1, ..., z_t|a_1, ..., a_{t-1}), \\ &= p(z_t|z_{t-1}, a_{t-1}, ..., z_1, a_1)\, p(z_{t-1}, ..., z_1|a_{t-1}, ..., a_1), \\ &= p(z_t|z_{t-1}, a_{t-1}, ..., z_1, a_1)\, p(z_{t-1}|z_{t-2}, a_{t-2}, ..., z_1, a_1) \, ... \, p(z_1). \end{aligned} \tag{18}$$

To address past states and actions within the conditional distribution, we treat them as history. This history model is implemented through a Gated Recurrent Units (GRU, Cho et al. (2014)) that encapsulates these past variables into a single history variable, $h_t = \{z_{t-1}, a_{t-1}, ..., z_1, a_1\} = \{z_{t-1}, a_{t-1}, h_{t-1}\}$. Thus we can write our conditional probability in Equation (18) as,

$$p(z_{1:t}|a_{1:t-1}) = p(z_t|z_{t-1}, a_{t-1}, h_{t-1})\, p(z_{t-1}|z_{t-2}, a_{t-2}, h_{t-2}) \, ... \, p(z_1), \tag{19}$$

$$= p(z_1) \prod_{k=1}^{t-1} p(z_{k+1}|z_k, a_k, h_k). \tag{20}$$

We can substitute the conditional distribution from Equation (20) into the Upper bound in Equation (17),

$$I(z_{1:t}) \leq \mathbb{E}_{p(z_{1:t},a_{1:t-1})}\left[\log \frac{\cancel{p(z_1)}\prod_{k=1}^{t-1} p(z_{k+1}|z_k,a_k,h_k)}{\cancel{p(z_1)}\prod_{k=1}^{t-1} p(z_{k+1})}\right], \tag{21}$$

$$\leq \mathbb{E}_{p(z_{1:t},a_{1:t-1})}\left[\log \prod_{k=1}^{t-1} \frac{p(z_{k+1}|z_k,a_k,h_k)}{p(z_{k+1})}\right], \tag{22}$$

$$\leq \sum_{k=1}^{t-1} \mathbb{E}_{p(z_k,a_k)}\left[\log \frac{p(z_{k+1}|z_k,a_k,h_k)}{p(z_{k+1})}\right], \tag{23}$$

$$\leq \sum_{k=1}^{t-1} I\big(z_{k+1}; z_k, a_k, h_k\big). \tag{24}$$

1.2 Derivation of Equation (9)

We aim to minimize the Mutual Information (MI) between the latent variables z_t from the beginning to time step t and the observations o_t from the environment i.e. $\min I(z_{t^-,t}; o_{t^-,t})$, where $\cdot_{t^-,t}$ is the variable from timestep 1 to t,

$$I(z_1, ..., z_t; o_1, ..., o_t) = \mathbb{E}_{p(z_{1:t},o_{1:t})}\left[\log \frac{p(z_{1:t}|o_{1:t})}{p(z_{1:t})}\right]. \tag{25}$$

Here we introduce the conditional probability distribution $p(z_{t^-}, z_t|a_{t^-})$ with the aim of removing out the denominator and including actions into our model,

$$I(z_{1:t}; o_{1:t}) = \mathbb{E}_{p(z_{1:t},o_{1:t},a_{1:t-1})}\left[\log \frac{p(z_{1:t}|o_{1:t})\, p(z_{1:t}|a_{1:t-1})}{p(z_{1:t}|a_{1:t-1})\, p(z_{1:t})}\right], \tag{26}$$

$$= \mathbb{E}_{p(z_{1:t},o_{1:t})}\left[\log \frac{p(z_{1:t}|o_{1:t})}{p(z_{1:t}|a_{1:t-1})}\right] - \mathbb{E}_{p(z_{1:t},a_{1:t-1})}\left[\log \frac{p(z_{1:t})}{p(z_{1:t}|a_{1:t-1})}\right], \tag{27}$$

$$= \mathbb{E}_{p(z_{1:t},o_{1:t})}\left[\log \frac{p(z_{1:t}|o_{1:t})}{p(z_{1:t}|a_{1:t-1})}\right] - D_{KL}\big(p(z_{1:t})||p(z_{1:t}|a_{1:t-1})\big), \tag{28}$$

$$\leq \mathbb{E}_{p(z_{1:t},o_{1:t})}\left[\log \frac{p(z_{1:t}|o_{1:t})}{p(z_{1:t}|a_{1:t-1})}\right]. \tag{29}$$

The second term is the KL-divergence between $p(z_{1:t})$ and $p(z_{1:t}|a_{1:t-1})$, which is always non-negative, leading to Equation (29) being an upper bound on our MI objective. The encodings at every timesteps depends only on that observation's timestep i.e. $p(z_{1:t}|o_{1:t}) = \prod_{k=1}^{t} p(z_k|o_k)$. Autoregressing the denominator according to Equation (20), we get,

$$I(z_{1:t}; o_{1:t}) = \mathbb{E}_{p(z_{1:t},o_{1:t})}\left[\log \frac{p(z_1|o_1)\prod_{k=1}^{t-1} p(z_{k+1}|o_{k+1})}{p(z_1)\prod_{k=1}^{t-1} p(z_{k+1}|z_k,a_k,h_k)}\right], \tag{30}$$

$$= \sum_{k=1}^{t-1} \mathbb{E}_{p(z_k,o_k)}\left[\log \frac{p(z_{k+1}|o_{k+1})}{p(z_{k+1}|z_k,a_k,h_k)}\right] - D_{KL}\big(p(z_1)||p(z_1|o_1)\big). \tag{31}$$

In Equation (8), we approximate this with the transition function with variational function $q_\theta(z_{k+1}|\hat{z})$, where $\hat{z} = (z_k, a_k, h_k)$. The transition function is a neural network with parameters θ. This is the same transition function described in the Equation (9),

$$I(z_{1:t}; o_{1:t}) \leq \sum_{k=1}^{t-1} \mathbb{E}_{p(z_k,o_k)}\left[\log \frac{p(z_{k+1}|o_{k+1})}{q_\theta(z_{k+1}|\hat{z})}\right] - D_{KL}\big(p(z_{k+1}|\hat{z})||q_\theta(z_{k+1}|\hat{z})\big). \tag{32}$$

As KL-divergence is non-negative, this is the upper bound on our main objective,

$$I(z_{1:t}; o_{1:t}) \leq \sum_{k=1}^{t-1} \mathbb{E}_{p(z_k, o_k)} \left[\log \frac{p(z_{k+1}|o_{k+1})}{q_\theta(z_{k+1}|z_k, a_k, h_k)} \right]. \tag{33}$$

1.3 Derivation of Equation (10)

We aim to maximise the Mutual Information (MI) from the current timestep to the Horizon T i.e., $\max I(z_t; z_{t^+})$; where $t^+ = \{t+1, ..., T\}$,

$$I(z_t; ...; z_T) = \mathbb{E}_{p(z_{t:T})} \left[\log \frac{p(z_{t:T})}{\prod_{k=t}^{T} p(z_k)} \right]. \tag{34}$$

The numerator in Equation (34) can be factorised with chain rule,

$$\begin{aligned} p(z_t, ..., z_T) &= p(z_t|z_{t+1}, ..., z_T)\, p(z_{t+1}|z_{t+2}, ..., z_T) \,...\, p(z_T), && (35)\\ &= p(z_t|z_{t+1:T})\, p(z_{t+1}|z_{t+2:T}) \,...\, p(z_T), && (36)\\ &= p(z_T) \prod_{k=t}^{T-1} p(z_k|z_{k+1:T}). && (37) \end{aligned}$$

Integrating Equation (37) in Equation (34),

$$\begin{aligned} I(z_{t:T}) &= \mathbb{E}_{p(z_{t:T})} \left[\log \frac{\cancel{p(z_T)} \prod_{k=t}^{T-1} p(z_k|z_{k+1:T})}{\cancel{p(z_T)} \prod_{k=t}^{T-1} p(z_k)} \right], && (38)\\ &= \mathbb{E}_{p(z_{t:T})} \left[\log \prod_{k=t}^{T-1} \frac{p(z_k|z_{k+1:T})}{p(z_k)} \right]. && (39) \end{aligned}$$

Here we incorporate conditional probability $p(z_k|z_{k+1,a_k})$ to remove $p(z_k|z_{k+1:T})$ out of our equation.

$$\begin{aligned} I(z_{t:T}) &= \mathbb{E}_{p(z_{t:T}, a_{t:T})} \left[\log \prod_{k=t}^{T-1} \frac{p(z_k|z_{k+1:T})\, p(z_k|z_{k+1}, a_k)}{p(z_k|z_{k+1}, a_k)\, p(z_k)} \right], && (40)\\ &= \sum_{k=t}^{T-1} \mathbb{E}_{p(z_k, a_k)} \left[\log \frac{p(z_k|z_{k+1}, a_k)}{p(z_k)} \right] + \sum_{k=t}^{T-1} D_{KL}\big(p(z_k|z_{k+1:T})||p(z_k|z_{k+1}, a_k)\big), && (41)\\ &\geq \sum_{k=t}^{T-1} \mathbb{E}_{p(z_k, a_k)} \left[\log \frac{p(z_k|z_{k+1}, a_k)}{p(z_k)} \right], && (42)\\ &= \sum_{k=t}^{T-1} I\big(z_k; z_{k+1}, a_k\big). && (43) \end{aligned}$$

1.4 Derivation of Equation (12)

We aim to maximize the Mutual Information (MI) between the latent variables z_t and the observations o_t from current time step t to time-horizon T i.e. $\max I(z_{t:T}; o_{t:T})$

$$I(z_{t:T}; o_{t:T}) = \mathbb{E}_{p(z_{t:T}, o_{t:T})} \left[\log \frac{p(o_{t:T}|z_{t:T})}{p(o_{t:T})} \right], \tag{44}$$

$$= \mathbb{E}_{p(z_{t:T}, o_{t:T})} \left[\log \prod_{k=t}^{T} \frac{p(o_k|z_k)}{p(o_k)} \right]. \tag{45}$$

Introducing a tractable variational decoder with parameters ψ,

$$I(z_{t:T}; o_{t:T}) = \mathbb{E}_{p(z_{t:T}, o_{t:T})} \left[\log \prod_{k=t}^{T} \frac{p(o_k|z_k)\; r_\psi(o_k|z_k)}{r_\psi(o_k|z_k)\; p(o_k)} \right], \tag{46}$$

$$= \sum_{k=t}^{T} \mathbb{E}_{p(z_k, o_k)} \left[\log \frac{r_\psi(o_k|z_k)}{p(o_k)} \right] + \sum_{k=t}^{T} D_{KL}\big(p(o_k|z_k)||r_\psi(o_k|z_k)\big), \tag{47}$$

$$\geq \sum_{k=t}^{T} \mathbb{E}_{p(z_k, o_k)} \left[\log \frac{r_\psi(o_k|z_k)}{p(o_k)} \right], \tag{48}$$

$$= \sum_{k=t}^{T} \mathbb{E}_{p(z_k, o_k)} \Big[\log\; r_\psi(o_k|z_k) \Big] - \sum_{k=t}^{T} \mathbb{E}_{p(o_k)} \Big[\log\; p(o_k) \Big], \tag{49}$$

$$= \sum_{k=t}^{T} \mathbb{E}_{p(z_k, o_k)} \Big[\log\; r_\psi(o_k|z_k) \Big] + \sum_{k=t}^{T} H(o_k), \tag{50}$$

$$= \sum_{k=t}^{T} \mathbb{E}_{p(z_k, o_k)} \Big[\log\; r_\psi(o_k|z_k) \Big]. \tag{51}$$

The entropy term $H(o_k)$ is independent of the parameter ψ, and consequently, can be disregarded during optimization.

2 Extended Related Work

In this section, an extended related work discussion is provided.

2.1 Relation to Human Psychology

Predictive Information is maximized by the brain at a higher, more abstract level as a strategy to prevent sensory overload (Friston, 2005; Rao and Ballard, 1999). Imagine a scenario where you're driving a vehicle and nearing a bend in the road, beyond which visibility is limited. Based on the experience of having faced congested traffic thus far (for say), you may anticipate a similar traffic configuration beyond the bend. In these instances, you mentally simulate future possibilities based on the historical experience and using the current location as a reference point. Notably, during this mental forecast, you instinctively disregard exogenous noise like vehicle's number plate, cloud formations in the sky, or roadside billboards. This subconscious omission of inconsequential details significantly influences the agent's decision-making process (Nasr et al., 2008). While maintaining scholarly modesty, it's essential to clarify that our contribution in this paper does not constitute an ultimate solution to the challenges described. Instead, our work introduces alternative ideas, traversing similar territory and contributing fresh perspectives to the existing discourse.

2.2 Learning Representations and Reinforcement Learning

Recent methodologies (Chen et al., 2020; Henaff, 2020; Tian et al., 2020) have achieved notable success in learning representations from unlabeled data. Approaches like (Laskin et al., 2020; Oord et al., 2018; Shu et al., 2020; Ma et al., 2021) have effectively integrated these concepts into reinforcement learning (RL). Some RL strategies prioritize learning state representations that solely contain information beneficial for predicting future states (Oord et al., 2018; Ma et al., 2021; Hjelm et al., 2019). However, they do not strive to find representations that encapsulate task-relevant details, which are significant for decision-making. Diverging from previous studies, our strategy directly

quantifies and compresses the predictive data, thereby ensuring that the representation bypasses the incorporation of vast amounts of past information that holds no relevance for the future. Our idea and approach shares a conceptual similarity with PI-SAC (Lee et al., 2020), where Conditional Entropy Bottleneck (CEB) is utilised to find compact representations and data augmentation for accelerating sample efficiency. A recent study (Stone et al., 2021) shows that data augmentation aids in standard environments but falters when noisy distractors are introduced. Particularly, methods like Dynamic Bottleneck (DB, Bai et al. (2021)) and Sequential Information Bottleneck for Robust Exploration (SIBE, You et al. (2022)) aim at seeking compact representations under noisy conditions. However, they do not focus on achieving noiseless future predictions or treating temporal noise along representations.

3 Implementation Details

In this section further algorithmic implementation details are discused.

3.1 Algorithm

We jointly train DPI with Soft Actor-Critic by incorporating Equation (13) as an auxiliary objective. Soft Actor-Critic (SAC) (Haarnoja et al., 2018) is an off-policy actor-critic reinforcement learning algorithm designed to optimize stochastic policies. It incorporates maximum entropy framework, ensuring a stochastic policy that seeks to balance reward maximization with entropy maximization. SAC employs a value function and two Q-functions (or critics) to reduce value overestimation. We specifically utilise the same encoder architecture as in Yarats et al. (2021). It aims at learning the latent state representation and policy jointly.
The training algorithm for DPI with SAC is presented in Algorithm 1. E_{step} is the environment step function. φ and θ are the parameters of observation encoder and transition function respectively. They are jointly optimised. The parameters of the two Q-function and the policy π are denoted by $\{\phi_q^1, \phi_q^2\}$ and ϕ_a respectively. $\{\varphi_m, \hat{\phi}_q^1, \hat{\phi}_q^2\}$ are the parameters of the target encoder and target Q-functions respectively, which updated with an exponential moving average. α is the temperature parameter. $\lambda_Q, \lambda_\pi, \lambda_\alpha$ and λ_{DPI} are the learning rates for four different objective functions.

Algorithm 1 Training Algorithm for SAC with DPI

Require: $\mathrm{E}_{\text{step}}, \alpha, \varphi, \theta, \psi, \phi_a, \phi_q^1, \phi_q^2, \mathrm{L}$ ▷ Environment and initial parameters.
$D \leftarrow \emptyset$ ▷ Initialize replay buffer
for each initial collection step **do**
  $a_t \sim \pi_{\text{random}}(\cdot|o_t)$ ▷ Sample action from a random policy
  $o_{t+1}, r_{t+1} \sim \mathrm{E}_{\text{step}}(a_t)$ ▷ Apply action
  $D \leftarrow D \cup (o_{t+1}, a_t, r_{t+1})$ ▷ Append experience to replay buffer
end for
for every training step **do**
  $\{(o_t, a_t, r_t, o_{t+1})\}_{t=k}^{L+k} \sim D$ ▷ Sample minibatch of sample from buffer
  for $t = 1$ to L **do**
    $a_t \sim \pi_{\phi_a}(a_t|o_t)$ ▷ Sample action from the policy
    $o_{t+1}, r_{t+1} \sim \mathrm{E}_{\text{step}}(a_t)$
    $D \leftarrow D \cup (o_{t+1}, a_t, r_{t+1})$
    for each gradient step **do**
      $\{\phi_q^i, \varphi\} \leftarrow \{\phi_q^i, \varphi\} - \lambda_Q \nabla \mathcal{L}_Q(\phi_q^i, \varphi)$ for $i \in \{1, 2\}$ ▷ Update soft Q-functions
      $\phi_a \leftarrow \phi_a - \lambda_\pi \nabla \mathcal{L}_\pi(\phi_a)$ ▷ Update policy
      $\alpha \leftarrow \alpha - \lambda_\alpha \nabla \mathcal{L}_\alpha(\alpha)$ ▷ Adjust temperature
      $\{\varphi, \theta\} \leftarrow \{\varphi, \theta\} - \lambda_{DPI} \nabla \mathcal{L}_{DPI}(\varphi, \theta)$ ▷ Update encoder and transition model
      $\hat{\phi}_q^i \leftarrow \tau \phi_q^i + (1 - \tau)\hat{\phi}_q^i$ for $i \in \{1, 2\}$ ▷ Update target Q-function
      $\varphi_m \leftarrow \tau\varphi + (1 - \tau)\varphi_m$ ▷ Update target encoder
    end for
  end for
end for

3.2 Model Architecture Details

Our implementation of Soft Actor-Critic (Haarnoja et al., 2018) is implemented in PyTorch and is based on the implementation of SAC-AE (Yarats et al., 2021).

3.2.1 Critic and Actor Network

For our critic, we use double Q-learning, where each Q-function is a 3-layer MLP, using ReLU activations after every layer, except the final one. Similarly, the actor is structured as a 3-layer MLP with ReLUs, designed to produce the mean and covariance values of the diagonal Gaussian. The hidden dimensions are set to 50 for actor and critic.

3.2.2 Observation Encoder and Decoder Networks

Encoder. Our encoder architecture aligns with the design proposed by Yarats et al. (2021). The architecture starts with an initial convolutional layer featuring a 3 × 3 kernel and a stride of 2. Subsequent to this, there are three more convolutional layers, each characterized by a 3 × 3 kernel and a stride of 1, resulting in a total of four convolutional layers, which have RELU activations. The 50 dimensional output of the fully-connected layer is stabilized using layer normalization (Ba et al., 2016), then divided into mean and standard deviation. We add tanh non-linearity on the standard deviation, then perform reparameterization trick to produce encoder's representation from the given observation.

Decoder. Our decoder is structured with an initial fully connected linear layer, followed by three deconvolutional layers with a 3 × 3 kernel and with a stride of 1, and the last layer with the same kernel size and stride of 2.

3.2.3 Transition Network

Our transition model integrates representation z_t (from the encoder) and action a_t into a single encoding, denoted as za_t, of size 256 via a fully connected linear layer. This encoding is subsequently passed through three additional fully connected layers, each having the same size and all using the Exponential Linear Units (ELU) as the activation function. To incorporate temporal dependencies, the state-action encoding is merged with the past history variable h_{t-1} via a Gated Recurrent Unit (GRU) mechanism. On another hand, this state-action encoding is concatenated (z_t^{input}) and passed via a fully connected linear layer to generate the next representation mean $\mu_{z_{t+1}}$ and standard deviation $\sigma_{z_{t+1}}$. They are then reparameterised to produce the next representation z_{t+1}. The entire procedure is comprehensively detailed in Algorithm 2.

Algorithm 2 Transition Model Pseudo-code

Require: z_t, a_t, h_{t-1} ▷ Representation, Action and History
$za_t \leftarrow \text{cat}(z_t, a_t)$ ▷ Concatenate Representation and action
$za_t \leftarrow \text{ELU}(\text{fc}_1(za_t))$ ▷ Representation-action encoding
for $i = 2$ to 4 **do**
    $za_t \leftarrow \text{ELU}(fc_i(za_t))$
end for
$h_t \leftarrow \text{GRU}(za_t, h_{t-1})$ ▷ Current history variable for next representation
$z_t^{input} \leftarrow \text{cat}(za_t, h_{t-1})$ ▷ Input for encoding next representation
$\mu_{z_{t+1}} \leftarrow \text{ELU}(fc_\mu(z_t^{input}))$ ▷ Next representation mean
$\sigma_{z_{t+1}} \leftarrow \tanh(fc_\sigma(z_t^{input}))$ ▷ Next representation standard deviation
$z_{t+1} \leftarrow \mu_{z_{t+1}} + \epsilon \odot \exp(\sigma_{z_{t+1}})$ ▷ Reparameterization trick

3.3 Code details

Upon publication, all code will be made publicly available. Additionally, we intend to release the code for the benchmarked algorithms.

4 Hyperparameters

To ensure a fair comparison, we maintained the original hyperparameters for each method and used the code as provided by the authors. The only adjustment we made is in how background images are incorporated into the observation. The complete set of Hyperpameters essential to implement our approach are provided in the Table I.

Table I: Hyperparameter settings and descriptions for the SAC with DPI implementation

Parameter name	**Value**	**Description**
Replay buffer capacity	2.5×10^5	Maximum number of past experiences stored for off-policy learning.
Image size	$84 \times 84 \times 3$	RGB image of size 84×84.
Batch size	32	Number of experiences sampled from the replay buffer for each update.
Discount γ	0.99	Factor by which future rewards are discounted in the Q-function.
Optimizer	Adam	Optimization algorithm used for training; Parameters: $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon_{ADAM} = 10^{-7}$.
Critic learning rate	10^{-5}	Learning rate used to update the critic's parameters.
Critic target update frequency	2	Frequency of copying weights from the critic to the target critic.
Critic Q-function soft-update rate τ_Q	0.005	Rate of soft-updating the critic's Q-function.
Critic encoder soft-update rate τ_ϕ	0.005	Rate of soft-updating the critic's encoder.
Actor learning rate	10^{-5}	Learning rate used to update the actor's parameters.
Actor update frequency	2	Frequency of actor parameter updates.
Actor log stddev bounds	[-10, 2]	Bounds on the logarithm of the actor's policy standard deviation.
Encoder learning rate	10^{-5}	Learning rate used to update the encoder's parameters.
Decoder learning rate	10^{-5}	Learning rate used to update the decoder's parameters.
Temperature learning rate	10^{-4}	Learning rate for the temperature parameter in the SAC's objective.
Init temperature	0.1	Initial temperature parameter that scales the entropy term in SAC's objective.

4.1 Sequence Length

A crucial aspect in our method is selecting the length of the time sequence. Ideally, it could span from the trajectory's start to a certain time horizon in the future. In our method, we establish that each information term can be splitted in a Markovian fashion, due to the incorporation of the history variable. For our experiments, we've chosen a time sequence length of three timesteps.

4.2 Action Repeat

Following Dreamer (Hafner et al., 2020), we designate repeat action of 2 for each environment. We adopt the same settings for all our baselines.

4.3 Weighing Coefficients

We performed a grid search on the weighing coefficients from a range of 1 to 10^{-5}. We empirically found out that setting α_2 large makes the algorithm unstable, as the I_{CLUB} loss dominates other terms significantly. The best settings are shown in the Table II.

Table II: Environment and their Coefficients

Environment	Weighing Coefficients			
	α_1	α_2	β_1	β_2
Cheetah Run	10^{-1}	10^{-3}	10^{-2}	10^{-2}
Walker Walk	10^{-2}	10^{-4}	10^{-2}	1
Cartpole Swingup	10^{-2}	10^{-3}	10^{-1}	10^{-1}

The coefficients are as follows, α_1: Weighing coefficient for I_{LTC}, α_2: Weighing coefficient for I_{CLUB}, β_1: Weighing coefficient for I_{Rec} and β_2: Weighing coefficient for I_{Rec}.

5 Experiments and Analysis

5.1 Videos Configuration

In this study, we slightly modified the background from what has been traditionally done in previous research. These minor alterations significantly influenced the outcomes. Our experimental conditions closely resembles that of Temporal Predictive Coding (TPC, Nguyen et al. (2021)), but we find it crucial to articulate this explicitly here.

1. Contrary to the predominant use of grayscale images in benchmarking across numerous past studies, including Denoised MDPs (Wang et al., 2022), Task Informed Abstractions (TIA, Fu et al. (2021)), Deep Bismulation for Control (DBC, Zhang et al. (2021)),Dreamer (Hafner et al., 2020), with the notable exception of TPC (Nguyen et al. (2021)), our work deviates by employing RGB videos instead.

2. We eliminated the ground plane to fully expose the natural background in the observations.

3. In order to ensure generalizability, we leverage a large collection of videos, segregating them into distinct sets for training and testing. Specifically, we've independently sampled 100 videos each for both training and testing. These natural videos are incorporated from Kinetics 400 dataset (Kay et al., 2017) at random.

For transparent benchmarking and easy access, we will subsequently upload these videos to a cloud storage platform on publication.

5.2 Baseline Methods

DBC. We used the observation of size 84×84 and stacked 3 consecutive frames following the original work (Zhang et al., 2021). We used the same hyperparameters mentioned in its paper.

Others. Utilizing the Recurrent State-Space Model (RSSM) as their transition model (Hafner et al., 2019), these methods follow an identical training schedule. For all the methods, we use 64×64 images and use the same parameters described in their respective papers. In order to maintain homogeneity, we used the same number of actions for all the baselines. The author's open source-code are utilised for their implementation without any changes.

5.3 Results in Standard Settings

While our main focus isn't on noiseless environments, we evaluated our method against baseline approaches in such settings. We observed that Dreamer outperforms all the methods in most of the environment in these settings. As depicted in Figure 1, our method is competitive in most of the environments.

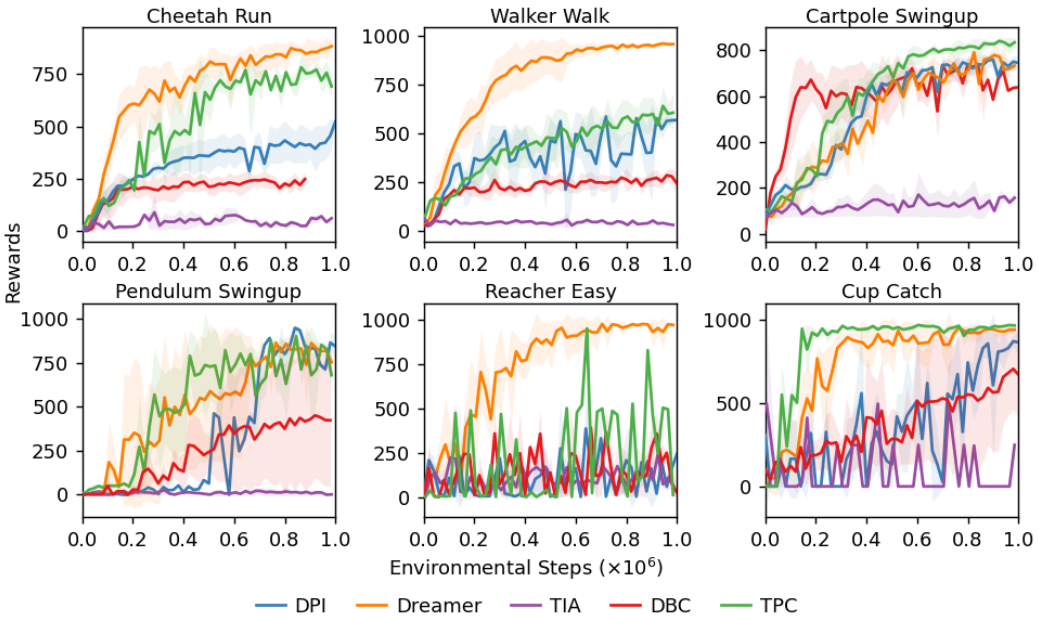

Figure 1: **Standard setting**. Performance comparison of our method (DPI) and baselines on six observation-based continuous control tasks from DMC Suite. Mean of 3 runs; shaded areas are 95% confidence intervals.

5.4 Results in Random Cartpole Settings

The presented results for the Cartpole swingup task in random background settings shows the performance of DPI in comparison with two relevant baselines: Dreamer-V2 (Hafner et al., 2021) and Self-Predictive Representations (SPR, Schwarzer et al. (2021)). As illustrated in Figure 2, it is evident that DPI outperforms the performance of both Dreamer-V2 and SPR in this setting.

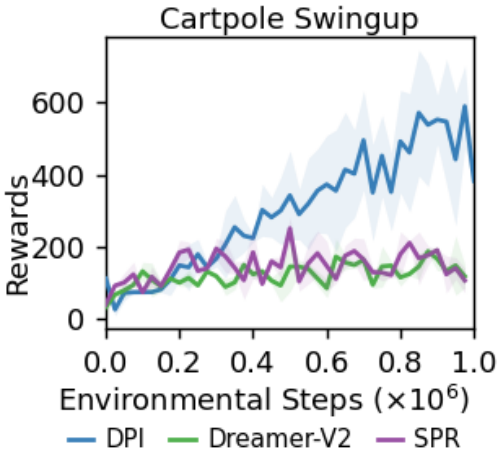

Figure 2: **Random setting**. Performance comparison of our method (DPI) and two relevant baselines on Cartpole swingup environment. Mean of 3 runs; shaded areas are 95% confidence intervals.

5.5 Computational Costs

All the experiments were done on a single GPU, that required atmost 8GB memory for all the tasks. We use multiple NVIDIA GPUs for training: 4070 (DBC and DPI), 4090 (DPI and Denoised MDPs), 3090 (TPC), P500 (TIA and Dreamer). Training time required for each run heavily depends on the CPU specification too. It also heavily relies on the batch size the algorithms are trained on. Single

seed of each method on average takes following time: DPI: $8 \sim 20$ hours, TIA: $15 \sim 24$ hours, Denoised MDPs: $5 \sim 8$ hours, TPC: $30 \sim 40$ hours, Dreamer: $15 \sim 24$ hours, DBC: $12 \sim 20$ hours.

6 Reconstructions

6.1 Reconstruction in the natural background setting

In our experiments, we explore the type of information encoded by different model encoders when trained in natural background settings. As depicted in Figure 3, while Dreamer (3rd row) attempts to encode both the agent and the background, DPI (2rd row) emphasizes on encoding the task-relevant agent, while the background is blurred. On the other hand, Denoised MDPs (Wang et al., 2022) also incorporate the background of other natural videos in the dataset, a consequence of overfitting on the training background noise, failing to generalise and separate the background from the agent.

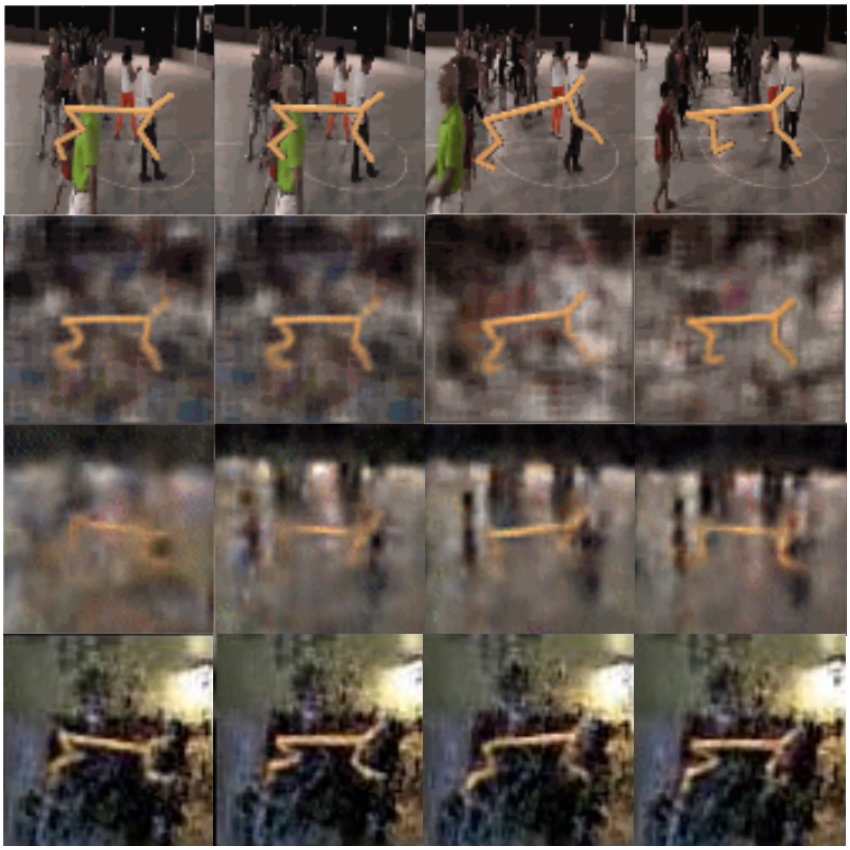

Figure 3: **Reconstruction**. Observation reconstruction of DPI versus Dreamer in the Natural background setting. First row: Ground Truth, Second row: DPI, Third row: Dreamer, Fourth Row: Denoised MDPs.

6.2 Reconstruction in blended backgrounds

We conduct experiments to investigate the challenges encountered in environments where the agent blends with their background due to similar colors. This phenomenon of color-based blending makes it difficult for the encoder to bifurcate between task-relevant features and background noise.

Figure 4: **Reconstruction in blended environments**. Observation reconstruction of DPI in the Natural background setting with similar color of agent and the background. First row: Ground Truth, Second row: DPI reconstruction

As illustrated in Figure 4, DPI prioritizes capturing task-relevant information and opts not to encode the background when it exhibits similar colors. In the reconstructions, the agent stands out distinctly, whereas the background appears blurred, underscoring DPI's focus on the agent over the surrounding noise.

6.3 Reconstruction of Cartpole swingup in random backgrounds

To investigate further into whether our method effectively emphasizes on relevant details, we carried out additional experiments on the Cartpole Swingup task. The findings from these experiments are shown in Figure 5.

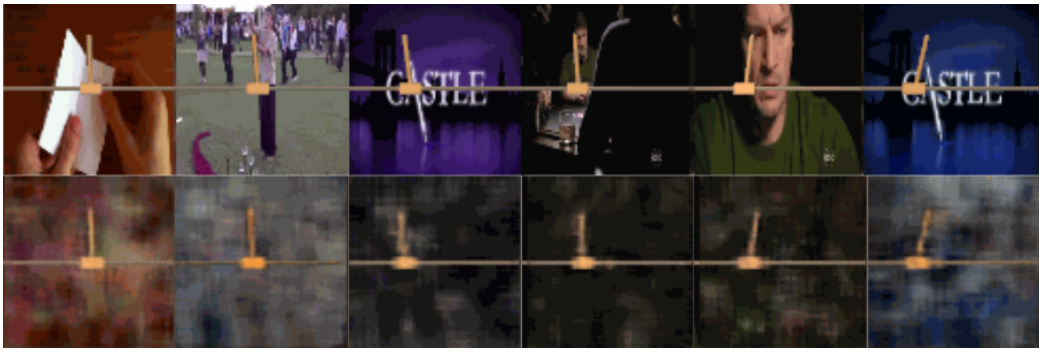

Figure 5: **Reconstruction in cartpole environment in random settings**. Observation reconstruction of DPI in the Cartpole environment in random background setting. First row: Ground Truth, Second row: DPI reconstruction

7 Ablation Analysis

In this section, we delve into an ablation study for the Cheetah Run environment, breaking down the components of the DPI model. Our experiment is conducted on various settings, each excluding distinct components in DPI (See Equation (13) for reference). Specifically, we consider:

A No latent consistency; removes I_{LTC} from $\mathcal{L}_{DPI}$ by setting $\alpha_1 = 0$.

B No upper bound minimization; removes I_{CLUB} from $\mathcal{L}_{DPI}$ by setting $\alpha_2 = 0$.

C No lower bound maximization; removes I_{NCE} from $\mathcal{L}_{DPI}$ by setting $\beta_2 = 0$.

D No reconstruction; removes I_{Rec} from $\mathcal{L}_{DPI}$ by setting $\beta_1 = 0$.

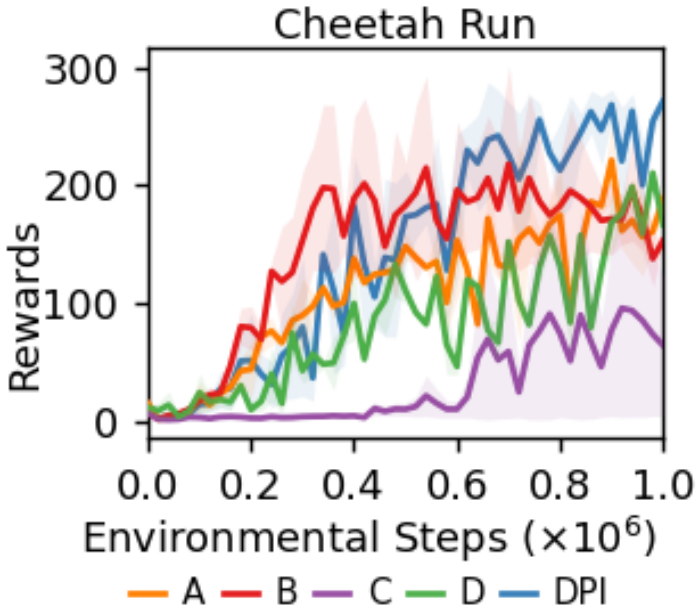

Figure 6: **Ablation Analysis**. Evaluating the impact of individual components removal on DPI's performance on Cheetah Run from DMC Suite. Mean of 3 runs; shaded areas are 95% confidence intervals.

The results of the experiments on Cheetah Run are illustrated in the Figure 6. Here we discuss the potential effects of these terms:

A No latent consistency settings eliminates the regularization of the latent representation from the transition from the observation encoder. This results into a drop in performance and noise addition from the past observations into the predicted observations (Figure 7, Third row).

B No upper bound minimization setting impacts the performance and stability in natural setting. This term is responsible for finding the current state representation from the past inputs. Exclusion of this term results in added noise in the current representations, potentially leading to higher variance and reduced performance. This can be seen in the Figure 7 (Fourth Row), where the learning algorithm is not able to accurately differentiate background video from the agent and as a result induces much more noise than in original DPI's reconstruction. The results are similar to A.

C No lower bound maximization; removes I_{NCE} from $\mathcal{L}_{DPI}$ by setting $\beta_2 = 0$. This term is responsible for predictive dynamics in the latent space. Based on our findings, omitting this term most profoundly diminishes the model's performance compared to the other components. A plausible explanation might be that this term prevents the representation from collapsing by incorporating the target encoder and updating it through a moving average. This is evident in the reconstructed image shown in Figure 7 (Fourth row), where all the observations converge to a singular representation, leading to similar outputs during reconstruction. It's worth mentioning that only the agent remains and the background is entirely eliminated in this scenario. This could be attributed to I_{CLUB} taking control and effectively filtering out all the noise.

D No reconstruction; removes I_{Rec} from $\mathcal{L}_{DPI}$ by setting $\beta_1 = 0$. As our approach is reconstruction based, not including reconstruction loss also has a profound impact on the learning.

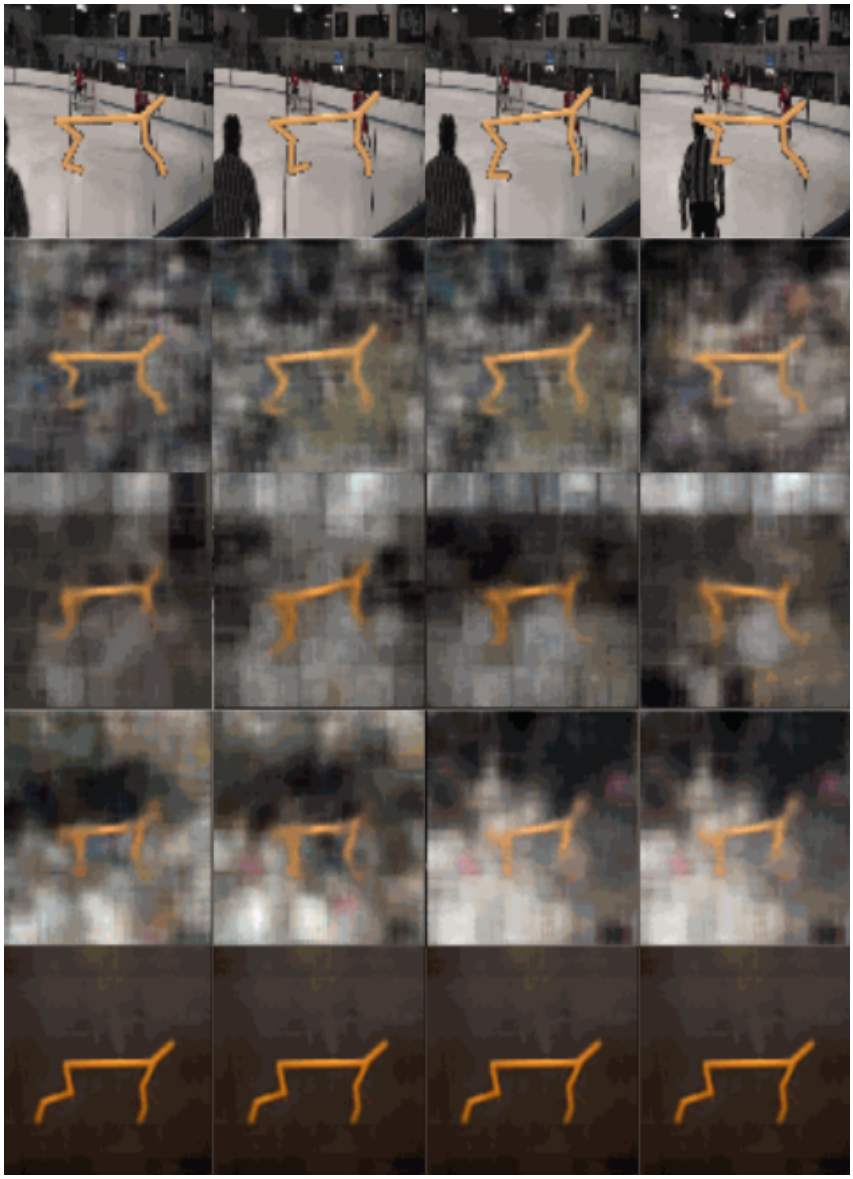

Figure 7: **Ablation Reconstruction**. Evaluating the impact of individual components removal on DPI's reconstruction on Cheetah Run from DMC Suite. First row: Ground Truth, Second row: DPI, Third row: A, Fourth row: B, Fifth row: C. We have not included D as it does not have the reconstruction.

References

Alemi, A. A.; Fischer, I.; Dillon, J. V.; Murphy, K. Deep Variational Information Bottleneck. International Conference on Learning Representations. 2017.

Cho, K.; van Merriënboer, B.; Bahdanau, D.; Bengio, Y. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation. Doha, Qatar, 2014; pp 103–111.

Friston, K. A theory of cortical responses. *Philosophical transactions of the Royal Society B: Biological sciences* **2005**, *360*, 815–836.

Rao, R. P.; Ballard, D. H. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience* **1999**, *2*, 79–87.

Nasr, S.; Moeeny, A.; Esteky, H. Neural correlate of filtering of irrelevant information from visual working memory. *PLoS One* **2008**, *3*, e3282.

Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. International Conference on Machine Learning. 2020; pp 1597–1607.

Henaff, O. Data-efficient image recognition with contrastive predictive coding. International Conference on Machine Learning. 2020; pp 4182–4192.

Tian, Y.; Krishnan, D.; Isola, P. Contrastive multiview coding. Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16. 2020; pp 776–794.

Laskin, M.; Srinivas, A.; Abbeel, P. Curl: Contrastive unsupervised representations for reinforcement learning. International Conference on Machine Learning. 2020; pp 5639–5650.

Oord, A. v. d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* **2018**,

Shu, R.; Nguyen, T.; Chow, Y.; Pham, T.; Than, K.; Ghavamzadeh, M.; Ermon, S.; Bui, H. Predictive coding for locally-linear control. International Conference on Machine Learning. 2020; pp 8862–8871.

Ma, X.; Chen, S.; Hsu, D.; Lee, W. S. Contrastive variational reinforcement learning for complex observations. Conference on Robot Learning. 2021; pp 959–972.

Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; Bengio, Y. Learning deep representations by mutual information estimation and maximization. International Conference on Learning Representations. 2019.

Lee, K.-H.; Fischer, I.; Liu, A.; Guo, Y.; Lee, H.; Canny, J.; Guadarrama, S. Predictive information accelerates learning in rl. *Advances in Neural Information Processing Systems (NeurIPS)* **2020**, *33*, 11890–11901.

Stone, A.; Ramirez, O.; Konolige, K.; Jonschkowski, R. The Distracting Control Suite–A Challenging Benchmark for Reinforcement Learning from Pixels. *arXiv preprint arXiv:2101.02722* **2021**,

Bai, C.; Wang, L.; Han, L.; Garg, A.; HAO, J.; Liu, P.; Wang, Z. Dynamic Bottleneck for Robust Self-Supervised Exploration. Advances in Neural Information Processing Systems (NeurIPS). 2021.

You, B.; Xie, J.; Chen, Y.; Peters, J.; Arenz, O. Self-supervised Sequential Information Bottleneck for Robust Exploration in Deep Reinforcement Learning. *arXiv preprint arXiv:2209.05333* **2022**,

Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; others Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905* **2018**,

Yarats, D.; Zhang, A.; Kostrikov, I.; Amos, B.; Pineau, J.; Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. Proceedings of the AAAI Conference on Artificial Intelligence. 2021; pp 10674–10681.

Ba, J. L.; Kiros, J. R.; Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450* **2016**,

Hafner, D.; Lillicrap, T.; Ba, J.; Norouzi, M. Dream to Control: Learning Behaviors by Latent Imagination. International Conference on Learning Representations. 2020.

Nguyen, T. D.; Shu, R.; Pham, T.; Bui, H.; Ermon, S. Temporal predictive coding for model-based planning in latent space. International Conference on Machine Learning. 2021; pp 8130–8139.

Wang, T.; Du, S. S.; Torralba, A.; Isola, P.; Zhang, A.; Tian, Y. Denoised MDPs: Learning World Models Better Than The World Itself. International Conference on Machine Learning. 2022.

Fu, X.; Yang, G.; Agrawal, P.; Jaakkola, T. Learning Task Informed Abstractions. Proceedings of the 38th International Conference on Machine Learning. 2021; pp 3480–3491.

Zhang, A.; McAllister, R. T.; Calandra, R.; Gal, Y.; Levine, S. Learning Invariant Representations for Reinforcement Learning without Reconstruction. International Conference on Learning Representations. 2021.

Kay, W.; Carreira, J.; Simonyan, K.; Zhang, B.; Hillier, C.; Vijayanarasimhan, S.; Viola, F.; Green, T.; Back, T.; Natsev, P.; others The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950* **2017**,

Hafner, D.; Lillicrap, T.; Fischer, I.; Villegas, R.; Ha, D.; Lee, H.; Davidson, J. Learning latent dynamics for planning from pixels. International Conference on Machine Learning. 2019; pp 2555–2565.

Hafner, D.; Lillicrap, T. P.; Norouzi, M.; Ba, J. Mastering Atari with Discrete World Models. International Conference on Learning Representations. 2021.

Schwarzer, M.; Anand, A.; Goel, R.; Hjelm, R. D.; Courville, A.; Bachman, P. Data-Efficient Reinforcement Learning with Self-Predictive Representations. International Conference on Learning Representations. 2021.