Efficient World Models with Context-Aware Tokenization

Vincent Micheli [* 1]   **Eloi Alonso** [* 1]   **François Fleuret** [1]

Abstract

Scaling up deep Reinforcement Learning (RL) methods presents a significant challenge. Following developments in generative modelling, model-based RL positions itself as a strong contender. Recent advances in sequence modelling have led to effective transformer-based world models, albeit at the price of heavy computations due to the long sequences of tokens required to accurately simulate environments. In this work, we propose Δ-IRIS, a new agent with a world model architecture composed of a discrete autoencoder that encodes stochastic deltas between time steps and an autoregressive transformer that predicts future deltas by summarizing the current state of the world with continuous tokens. In the Crafter benchmark, Δ-IRIS sets a new state of the art at multiple frame budgets, while being an order of magnitude faster to train than previous attention-based approaches. We release our code and models at `https://github.com/vmicheli/delta-iris`.

1. Introduction

Deep Reinforcement Learning (RL) methods have recently delivered impressive results (Ye et al., 2021; Hafner et al., 2023; Schwarzer et al., 2023) in traditional benchmarks (Bellemare et al., 2013; Tassa et al., 2018). In light of the evermore complex domains tackled by the latest generations of generative models (Rombach et al., 2022; Achiam et al., 2023), the prospect of training agents in more ambitious environments (Kanervisto et al., 2022) may hold significant appeal. However, that leap forward poses a serious challenge: deep RL architectures have been comparatively smaller and less sample-efficient than their (self-)supervised counterparts. In contrast, more intricate environments necessitate models with greater representational power and have higher data requirements.

Model-based RL (MBRL) (Sutton & Barto, 2018) is hypothesized to be the key for scaling up deep RL agents (LeCun, 2022). Indeed, world models (Ha & Schmidhuber, 2018) offer a diverse range of capabilities: lookahead search (Schrittwieser et al., 2020; Ye et al., 2021), learning in imagination (Sutton, 1991; Hafner et al., 2023), representation learning (Schwarzer et al., 2021; D'Oro et al., 2023), and uncertainty estimation (Pathak et al., 2017; Sekar et al., 2020). In essence, MBRL shifts the focus from the RL problem to a generative modelling problem, where the development of an accurate world model significantly simplifies policy training. In particular, policies learnt in the imagination of world models are freed from sample efficiency constraints, a common limitation of RL agents that is magnified in complex environments with slow rollouts.

Recently, the IRIS agent (Micheli et al., 2023) achieved strong results in the Atari 100k benchmark (Bellemare et al., 2013; Kaiser et al., 2020). IRIS introduced a world model composed of a discrete autoencoder and an autoregressive transformer, casting dynamics learning as a sequence modelling problem where the transformer composes over time a vocabulary of image tokens built by the autoencoder. This approach opened up avenues for future model-based methods to capitalize on advances in generative modelling (Villegas et al., 2022; Achiam et al., 2023), and has already been adopted beyond its original domain (comma.ai, 2023; Hu et al., 2023). However, in its current form, scaling IRIS to more complex environments is computationally prohibitive. Indeed, such an endeavor requires a large number of tokens to encode visually challenging frames. Besides, sophisticated dynamics may require to store numerous time steps in memory to reason about the past, ultimately making the imagination procedure excessively slow. Hence, under these constraints, maintaining a favorable imagined-to-collected data ratio is practically infeasible.

In the present work, we introduce Δ-IRIS, a new agent capable of scaling to visually complex environments with lengthier time horizons. Δ-IRIS encodes frames by attending to the ongoing trajectory of observations and actions, effectively describing stochastic deltas between time steps. This enriched conditioning scheme drastically reduces the number of tokens to encode frames, offloads the deterministic aspects of world modelling to the autoencoder, and lets the autoregressive transformer focus on stochastic dynamics.
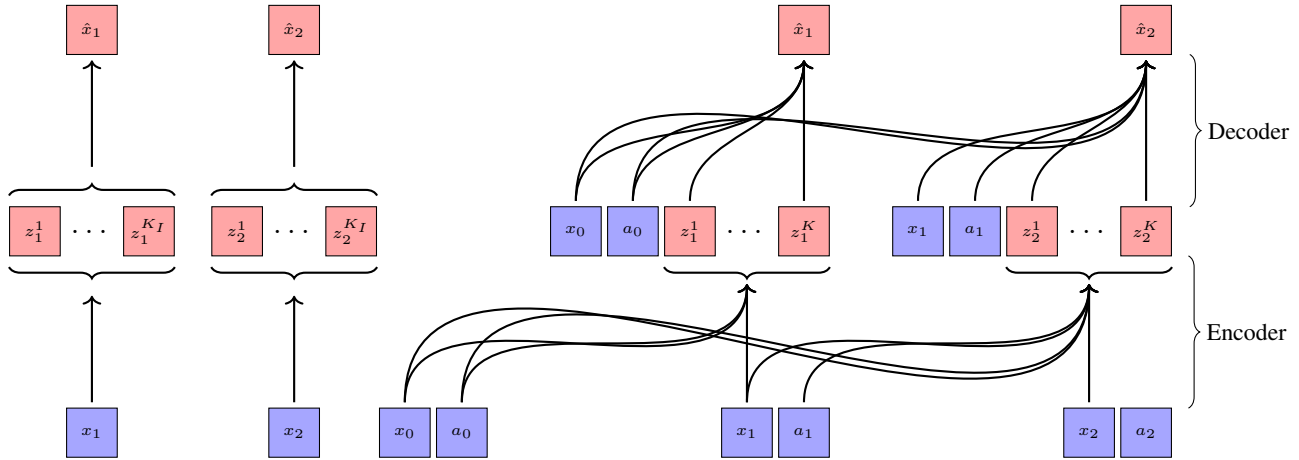
---
[*]Equal contribution [1]University of Geneva, Switzerland. Correspondence to: <first.last@unige.ch>.

Figure 1. Discrete autoencoder of IRIS (Micheli et al., 2023) (left) and Δ-IRIS (right). IRIS encodes and decodes frames independently, meaning that z_t has to carry all the information necessary to reconstruct x_t. On the other hand, Δ-IRIS' encoder and decoder are conditioned on past frames and actions, thus z_t only has to capture what has changed and that cannot be inferred from actions, i.e. the stochastic delta. This conditioning scheme enables us to drastically reduce the number of tokens required to encode a frame with minimal loss ($K \ll K_I$), which is critical to speed up the autoregressive transformer that predicts future tokens.

Nonetheless, substituting the sequence of absolute image tokens with a sequence of Δ-tokens makes the task of the autoregressive model more arduous. In order to predict the next transition, it may only reason over previous Δ-tokens, and thus faces the challenge of integrating over multiple time steps as a way to form a representation of the current state of the world. To resolve this issue, we modify the sequence of the autoregressive model by interleaving continuous I-tokens, that summarize successive world states with frame embeddings, and discrete Δ-tokens.

In the Crafter benchmark (Hafner, 2022), Δ-IRIS exhibits favorable scaling properties: the agent solves 17 out of 22 tasks after 10M frames of data collection, supersedes DreamerV3 (Hafner et al., 2023) at multiple frame budgets, and trains 10 times faster than IRIS. In addition, we include results in the sample-efficient setting with Atari games. Through experiments, we provide evidence that Δ-IRIS learns to disentangle the deterministic and stochastic aspects of world modelling. Moreover, we conduct ablations to validate the new conditioning schemes for the autoencoder and transformer models.

2. Method

We consider a Partially Observable Markov Decision Process (POMDP) (Sutton & Barto, 2018). The transition, reward, and episode termination dynamics are captured by the conditional distributions $p(x_{t+1} \mid x_{\leq t}, a_{\leq t})$ and $p(r_t, d_t \mid x_{\leq t}, a_{\leq t})$, where $x_t \in \mathcal{X} = \mathbb{R}^{3 \times h \times w}$ is an image observation, $a_t \in \mathcal{A} = \{1, \ldots, A\}$ a discrete action, $r_t \in \mathbb{R}$ a scalar reward, and $d_t \in \{0, 1\}$ indicates episode termination. The reinforcement learning objective is to find

a policy $p_\pi(a_t \mid x_{\leq t}, a_{<t})$ that maximizes the expected sum of rewards $\mathbb{E}_\pi[\sum_{t \geq 0} \gamma^t r_t]$, with discount factor $\gamma \in (0, 1)$.

Learning in imagination (Sutton, 1991; Sutton & Barto, 2018) consists of 3 stages that are repeated alternatively: experience collection, world model learning, and policy improvement. Strikingly, the agent learns behaviours purely within its world model, and real experience is only leveraged to learn the environment dynamics.

In the vein of IRIS (Micheli et al., 2023), our world model is composed of a discrete autoencoder (Van Den Oord et al., 2017) and an autoregressive transformer (Vaswani et al., 2017; Radford et al., 2019), albeit with new conditioning schemes and architectures. We first expose IRIS' world model in Section 2.1, then present Δ-IRIS' autoencoder and autoregressive model in Sections 2.2 and 2.3, respectively. Finally, we describe the policy improvement phase in Section 2.4. Appendix A gives a detailed breakdown of model architectures and hyperparameters.

2.1. Background: IRIS

High-dimensional images are converted into tokens with a discrete autoencoder (E_I, D_I) (Van Den Oord et al., 2017). The encoder $E_I : \mathbb{R}^{h \times w \times 3} \to \{1, \ldots, N_I\}^{K_I}$ maps an input image x_t into K_I tokens from a vocabulary of size N_I. The discretization is done by picking the index of the vector in the vocabulary embedding table that is closest to the encoder output $y_t \in \mathbb{R}^{K_I \times d}$. The K_I tokens are then decoded back into an image with $D_I : \{1, \ldots, N_I\}^{K_I} \to \mathbb{R}^{h \times w \times 3}$. This discrete autoencoder is trained with L_1 reconstruction, perceptual (Esser et al., 2021) and commitment losses (Van Den Oord et al., 2017) computed on collected frames.
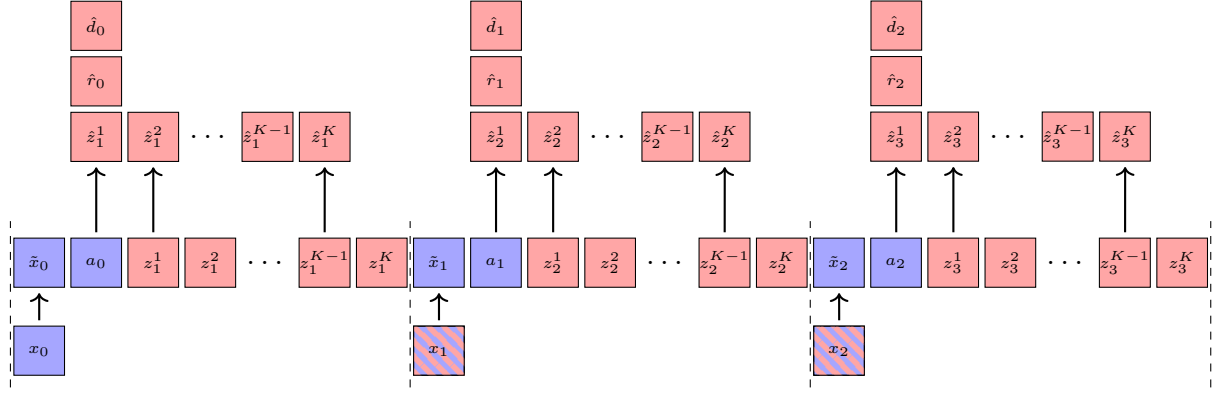
Figure 2. Unrolling dynamics over time. At each time step (separated by dashed lines), the GPT-like autoregressive transformer G predicts the Δ-tokens for the next frame, as well as the reward and a potential episode termination. Its input sequence consists of action tokens, Δ-tokens, and I-tokens, namely continuous image embeddings that alleviate the need to attend to past Δ-tokens for world modelling. More specifically, an initial frame x_0 is embedded into I-token $\tilde{x}_0$. From $\tilde{x}_0$ and a_0, G predicts the reward $\hat{r}_0$, episode termination $\hat{d}_0 \in \{0, 1\}$, and in an autoregressive manner $\hat{z}_1 = (\hat{z}_1^1, \ldots, \hat{z}_1^K)$, the Δ-tokens for the next frame. Note that, during the imagination procedure, the next frame (stripped box) is computed by the decoder D based on previous frames, actions, and the Δ-tokens generated by G, i.e. $x_1 = D(x_0, a_0, \hat{z}_1)$.

The transformer G_I models the environment dynamics by operating over an input sequence of image and action tokens $(z_0^1, \ldots, z_0^{K_I}, a_0, z_1^1, \ldots, z_1^{K_I}, a_1, \ldots, z_t^1, \ldots, z_t^{K_I}, a_t)$. Image and action tokens are embedded with learnt lookup tables. At each time step, G_I predicts the transition, reward, and termination distributions: $p_{G_I}(\hat{z}_{t+1}^k | z_{\leq t}, a_{\leq t})$ with $\hat{z}_{t+1}^k \sim p_{G_I}(\hat{z}_{t+1}^k | z_{\leq t}, a_{\leq t}, z_{t+1}^{<k})$, $p_{G_I}(\hat{r}_t | z_{\leq t}, a_{\leq t})$, and $p_{G_I}(\hat{d}_t | z_{\leq t}, a_{\leq t})$. The model is trained with a cross-entropy loss on segments sampled from past experience.

At a high level, the autoencoder builds a vocabulary of image tokens to encode each frame, and the transformer captures the environment dynamics by autoregressively composing the vocabulary over time. As a result, this world model is capable of attending to previous time steps to make its predictions, and models the joint law of future latent states.

2.2. Disentangling deterministic and stochastic dynamics

IRIS (Micheli et al., 2023) encodes frames independently, making no assumption about temporal redundancy within trajectories. One major drawback of this general formulation is that, in environments with visually challenging frames, a large number of tokens is required to encode frames losslessly. Consequently, computations with the dynamics model become increasingly prohibitive, as the attention mechanism scales quadratically with sequence length. Therefore, limiting computation under such a trade-off may result in degraded performance (Micheli et al. (2023) app. E)

One possible solution to achieve fast world modelling with minimal loss is to condition the autoencoder on previous frames and actions. Intuitively, encoding a frame given previous frames consists in describing what has changed, the delta, between successive time steps. In many environments, the delta between frames is often much simpler to describe than the frames themselves. As a matter of fact, when the transition function is deterministic, adding previous actions to the conditioning of the decoder results in a world model, without the need to encode any information between time steps. However, most environments of interest feature stochastic dynamics, and apart from aleatoric uncertainty, architectural limitations such as the agent's memory may induce additional epistemic uncertainty. Hence, the delta between two time steps usually consists of deterministic and stochastic components.

For instance, an agent moving from one square to another in a grid-like environment when pressing movement keys can be seen as a deterministic component of the transition. On the other hand, the sudden apparition of an enemy in a nearby square is a random event. Interestingly, only the stochastic features of a transition should be encoded, and the autoencoder could directly learn to model the deterministic dynamics, which do not require the expressivity and ability to handle multimodality of an autoregressive model. Therefore, when autoenconding frames by conditioning on previous frames and actions, a frame encoding may only consist of a handful of Δ-tokens, instead of a large number of image tokens describing frames independently.

Section 3.4 provides empirical evidence that Δ-IRIS' autoencoder learns to encode frames in such fashion, and Figure 1 illustrates the new conditioning scheme of the autoencoder.
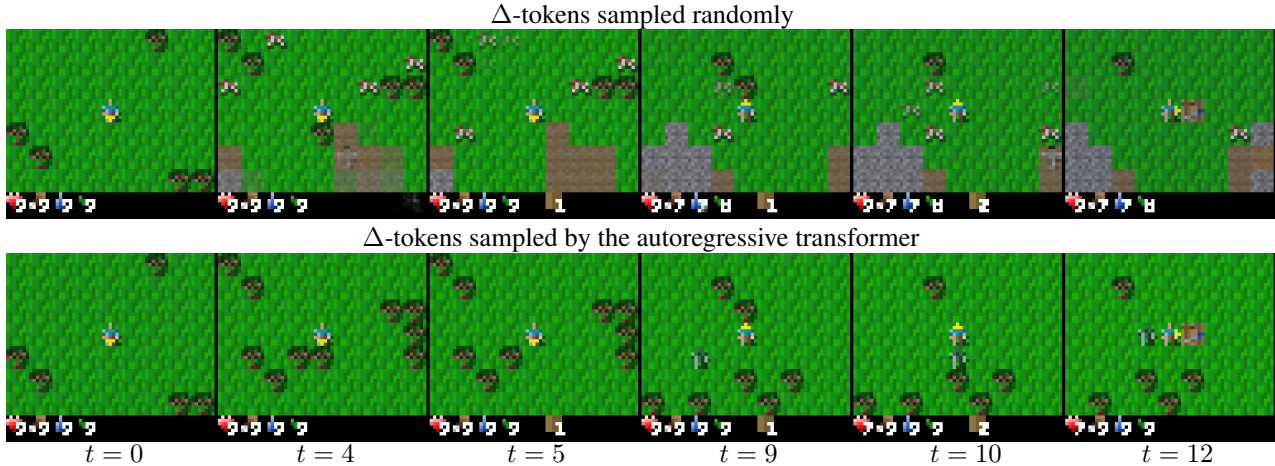
$t = 0 \qquad t = 4 \qquad t = 5 \qquad t = 9 \qquad t = 10 \qquad t = 12$

Figure 3. Evidence of dynamics disentanglement. Two trajectories are imagined with different ways of generating Δ-tokens. In the top trajectory, Δ-tokens are sampled randomly. In the bottom trajectory, the autoregressive transformer predicts future Δ-tokens. The same starting frame ($t = 0$) and sequence of actions are used. With random Δ-tokens, the deterministic aspects of the dynamics (layout, movement, items, crafting) are still properly modelled, but the stochastic dynamics (mobs, health indicators) become problematic. For instance, the agent successfully cuts down a tree between $t = 4$ and $t = 5$, and uses wood planks to build a crafting table between $t = 10$ and $t = 12$. We observe that these dynamics are modelled in the same way whether Δ-tokens are sampled randomly or not. However, in the top trajectory, large quantities of cows appear and disappear from the screen incoherently, whereas the bottom trajectory does not display such erratic patterns. This experiment shows that Δ-IRIS encodes stochastic deltas between time steps with Δ-tokens, and its decoder handles the deterministic aspects of world modelling. Appendix F contains additional examples.

More formally, for any set $\mathcal{Y}$, we denote $\mathcal{S}_n(\mathcal{Y}) = \bigcup_{i=1}^{n} \mathcal{Y}^i$ the set of tuples of elements from $\mathcal{Y}$ of maximum length n, and $\mathcal{S}(\mathcal{Y}) = \mathcal{S}_{\infty}(\mathcal{Y})$. Let $\mathcal{Z} = \{1, \ldots, N\}$ a vocabulary of discrete tokens. Given past images and actions $(x_0, a_0, \ldots, x_{t-1}, a_{t-1})$, the encoder $E : \mathcal{S}(\mathcal{X} \times \mathcal{A}) \times \mathcal{X} \to \mathcal{Z}^K$ converts an image x_t into $z_t = (z_t^1, \ldots, z_t^K)$, a sequence of K discrete Δ-tokens. The encoder is parameterized by a Convolutional Neural Network (CNN) (Le-Cun et al., 1989). Actions are embedded with a learnt lookup table and concatenated channel-wise with frames. We use vector quantization (Van Den Oord et al., 2017; Esser et al., 2021) with factorized and normalized codes (Yu et al., 2021) to discretize the encoder's continuous outputs. The CNN decoder $D : \mathcal{S}(\mathcal{X} \times \mathcal{A}) \times \mathcal{Z}^K \to \mathcal{X}$ reconstructs an image $\hat{x}_t$ from past frames, actions and Δ-tokens $(x_0, a_0, \ldots, x_{t-1}, a_{t-1}, z_t)$. Action and Δ-tokens are embedded with learnt lookup tables, and concatenated channel-wise with feature maps obtained by forwarding frames through an auxiliary CNN.

The discrete autoencoder is trained on previously collected trajectories with a weighted combination of L_1, L_2 and max-pixel (Anand et al., 2022) reconstruction losses, as well as a commitment loss (Van Den Oord et al., 2017). The codebook is updated with an exponential moving average (Razavi et al., 2019) and we use a straight-through estimator (Bengio et al., 2013) to enable backpropagation.

2.3. Modelling stochastic dynamics

While it should be possible to predict future Δ-tokens, given a starting image, past actions and Δ-tokens, we found this task much more difficult than simply predicting future image tokens, given past image tokens and actions, as in IRIS.

To better understand why this is the case, let us consider another example: in a grid environment, Δ-tokens may describe the unpredictable movement of an enemy, randomly jumping from one square to another at every time step. Based on the initial enemy location and after only a few time steps, it becomes increasingly difficult to predict if the enemy and the agent are located on the same square, which could trigger a battle and make the enemy disappear. Indeed, situating the two entities involves reasoning about the initial observation, and integrating over all of the previous action and Δ-tokens, which may have a complex dependence structure.

To address this problem, we alter the sequence of the dynamics model by interleaving continuous I-tokens, in reference to MPEG's I-frames (Richardson, 2004), and discrete Δ-tokens. I-tokens alleviate the need of integrating over past Δ-tokens to form a representation of the current state of the world, i.e. they deploy a "soft" Markov blanket for the prediction of the next Δ-tokens.

Table 1. Returns, number of parameters, and frames collected per second (FPS) for the methods considered. We compute FPS as the total number of environment frames collected divided by the training duration. Δ-IRIS outperforms DreamerV3 for larger frame budgets, and is 10x faster than IRIS (64 tokens).

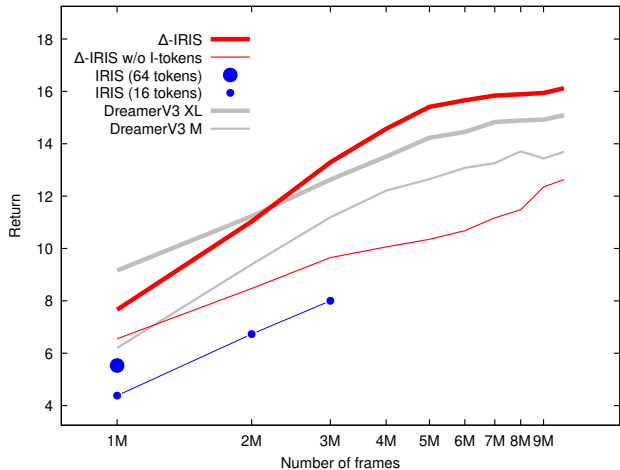| Method	Return @1M	Return @5M	Return @10M	#Parameters	FPS
Δ-IRIS	7.7 (0.5)	**15.4** (0.4)	**16.1** (0.1)	**25M**	20
DreamerV3 XL	**9.2** (0.3)	14.2 (0.2)	15.1 (0.3)	200M	**30**
IRIS (64 tokens)	5.5 (0.7)	-	-	48M	2
Δ-IRIS w/o I-tokens	6.6 (0.2)	10.4 (0.5)	12.6 (0.8)	24M	22
DreamerV3 M	6.2 (0.5)	12.6 (0.7)	13.7 (0.8)	37M	40
IRIS (16 tokens)	4.4 (0.1)	-	-	50M	6

Figure 4. Returns at multiple frame budgets in the Crafter benchmark. Δ-IRIS achieves higher returns than DreamerV3 beyond 3M frames, and surpasses IRIS for all frame budgets considered. Removing I-tokens from the input sequence of the autoregressive transformer significantly hurts performance.

We obtain I-tokens by forwarding frames through an auxiliary CNN at each time step. They are not produced by a discrete autoencoder. Since I-tokens are not predicted by the model but rather enrich its conditioning, there are no incentives to include a lossy discretization operator or to optimize a reconstruction loss. Instead, they are optimized end-to-end with the learning objectives of the dynamics model. With this improved conditioning, the dynamics model perceives the ongoing trajectory with a mixture of continuous and discrete representations, while making its predictions autoregressively in a discrete space.

Figure 2 displays the input sequence of the dynamics model and the quantities it predicts. Given a sequence of past I-tokens, action tokens, and Δ-tokens $(\tilde{x}_0, a_0, z_1^1, \ldots, z_1^K, \ldots, \tilde{x}_{t-1}, a_{t-1}, z_t^1, \ldots, z_t^k)$, the dynamics model G outputs a categorical distribution on $\mathcal{Z}$ for the next Δ-token $\hat{z}_t^{k+1} \sim p_G(\hat{z}_t^{k+1}|\tilde{x}_{<t}, z_{<t}, a_{<t}, z_t^{\leq k})$. It also predicts distributions for rewards $p_G(\hat{r}_t|\tilde{x}_{\leq t} z_{\leq t}, a_{\leq t})$ and episode terminations $p_G(\hat{d}_t|\tilde{x}_{\leq t}, z_{\leq t}, a_{\leq t})$.

G is parameterized by a stack of transformer encoder layers with causal self-attention (Vaswani et al., 2017; Radford et al., 2019). It is trained with a cross-entropy loss for transition and termination predictions, and we follow DreamerV3 (Hafner et al., 2023) in using discrete regression with two-hot targets and symlog scaling for reward prediction (Imani & White, 2018).

2.4. Policy improvement

During the policy improvement phase, the policy π learns in the imagination POMDP of its world model, composed of the autoencoder (E, D) and the dynamics model G.

At time step t, the policy observes a reconstructed image observation $\hat{x}_t$ and samples action $a_t \sim \pi(a_t|\hat{x}_{\leq t})$. The world model then predicts the reward $\hat{r}_t$, the episode end $\hat{d}_t$, and the next observation $\hat{x}_{t+1} = D(\hat{x}_{\leq t}, \hat{a}_{\leq t}, \hat{z}_{\leq t}, \hat{z}_{t+1})$, with $\hat{z}_{t+1} \sim p_G(\hat{z}_{t+1}|\hat{x}_{\leq t}, \hat{a}_{\leq t}, \hat{z}_{\leq t})$. The imagination procedure is initialized with a real observation x_0 sampled from past experience, and is rolled out for H steps. The procedure stops if an episode termination is predicted before reaching the imagination horizon.

For the sake of simplicity and to make comparisons easier, we employ the actor-critic training method used for IRIS (Micheli et al., 2023). A value baseline is trained to regress λ-returns (Sutton & Barto, 2018) with a mean squared error loss. The policy optimizes the REINFORCE with value baseline (Sutton & Barto, 2018) learning objective over imagined trajectories. Exploration is encouraged by adding an entropy maximization term to the policy's objective.

3. Experiments

In our experiments, we consider the Crafter benchmark (Hafner, 2022) to illustrate Δ-IRIS' ability to scale to a visually rich environment with large frame budgets. Besides, we also include Atari 100k games (Bellemare et al., 2013; Kaiser et al., 2020) in Appendix C to showcase the performance and speed of our agent in the sample-efficient setting.
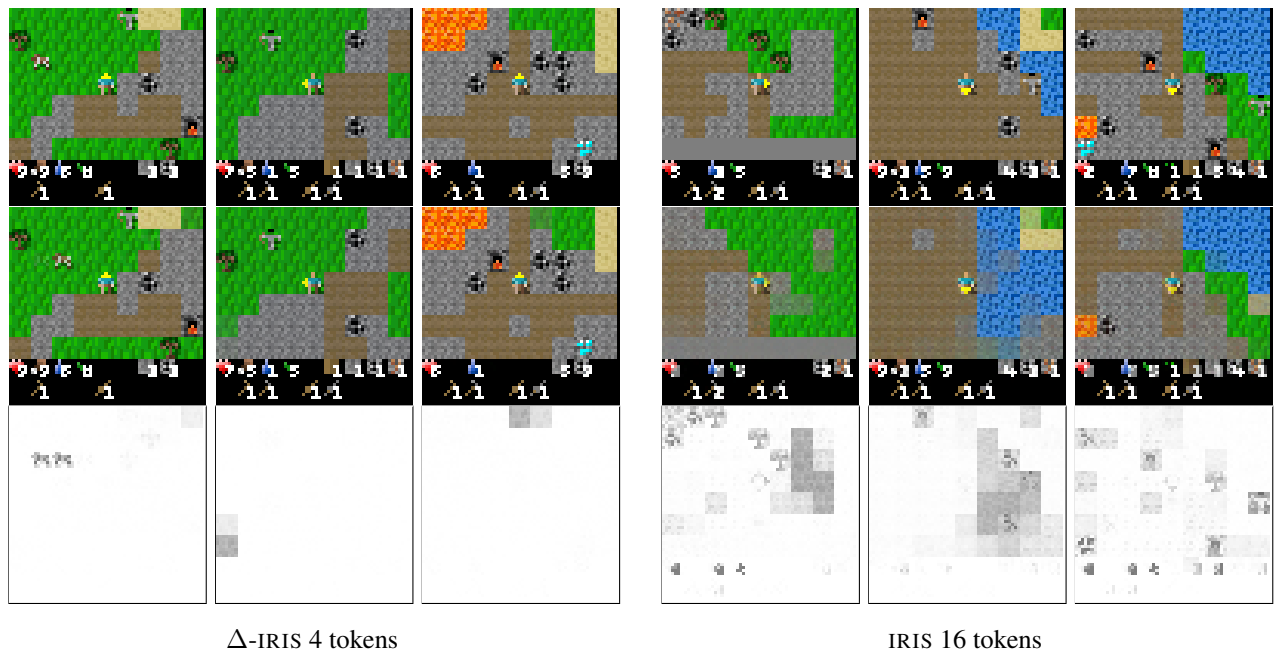
Δ-IRIS 4 tokens   IRIS 16 tokens

Figure 5. Bottom 1% test frames autoencoded by Δ-IRIS (4 tokens) and IRIS (Micheli et al., 2023) (16 tokens). Each token takes a value in $\{1, 2, \ldots, 1023, 1024\}$, i.e. Δ-IRIS encodes frames with $4 \times \log_2(1024) = 40$ bits while IRIS uses 160 bits. Original frames, reconstructions, and errors are respectively displayed in the top, middle, and bottom rows. Even in the worst instances, Δ-IRIS makes only minor errors, whereas IRIS fails to accurately reconstruct frames. These errors severely hamper the agent's performance, as it purely learns behaviours from frames generated by its autoencoder.

We introduce the Crafter benchmark and baselines in Section 3.1. Then, we present our results in Section 3.2. Finally, in Sections 3.3 and 3.4, we propose qualitative experiments to validate Δ-IRIS' world model architecture, and better our understanding of how the model represents information.

3.1. Benchmark and baselines

Crafter (Hafner, 2022) is a procedurally generated environment, inspired by the video game Minecraft, with visual inputs, a discrete action space and non-deterministic dynamics. By incorporating mechanics from survival games and a technology tree, this benchmark evaluates a broad range of agent capabilities such as generalization, exploration, and credit assignment. During each episode, the agent's goal is to solve as many tasks as possible, e.g. slaying mobs, crafting items, and managing health indicators.

Regarding baselines, we consider two model-based RL agents learning in imagination: IRIS (Micheli et al., 2023) and DreamerV3 (Hafner et al., 2023). We run several variants: IRIS (16 tokens), encoding frames with $K_I = 16$ tokens, IRIS (64 tokens), encoding frames with $K_I = 64$ tokens, and configurations of DreamerV3 of different sizes, namely DreamerV3 XL and DreamerV3 M. To demonstrate the importance of I-tokens, we also run Δ-IRIS without I-tokens in the sequence of the transformer, i.e. G only operates over the first frame as well as actions and Δ-tokens.

We keep a fixed imagined-to-collected data ratio of 64 to balance speed and performance. Our experiments run on a Nvidia A100 40GB GPU, with 5 seeds for all methods and ablations. We evaluate each run by computing the average return over 256 test episodes every 1M frames. Note that we stop the IRIS experiments before 10M frames because they are prohibitively slow.

3.2. Results

Table 1 exhibits key metrics and Figure 4 displays learning curves. After 10M frames of data collection, Δ-IRIS solves on average 17 out of 22 tasks, setting a new state of the art for the Crafter benchmark. Beyond the 3M frames mark, Δ-IRIS consistently achieves higher returns than DreamerV3, although DreamerV3 is better suited for the smallest frame budgets. A key difference between the two methods is that Δ-IRIS does not leverage the representations of its world model for policy learning, which may be especially useful in the scarce data regime. As our main objective is to develop world model architectures that scale to complex environments and larger frame budgets, we leave this exploration to future work. Δ-IRIS outperforms IRIS for all frame budgets considered, while training an order of magnitude faster. Finally, removing I-tokens from the sequence of the dynamics model drastically hurts performance.
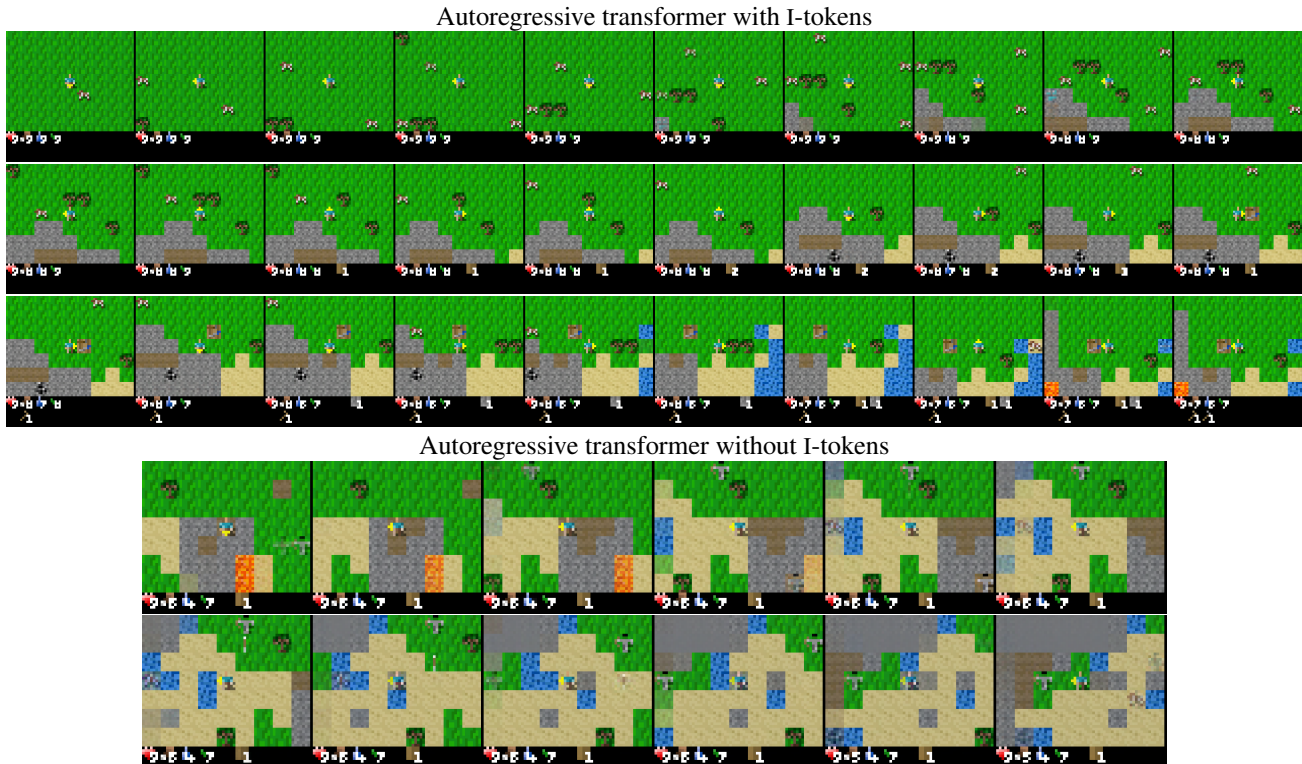
Autoregressive transformer with I-tokens

Autoregressive transformer without I-tokens

Figure 6. Trajectories imagined with (top) and without (bottom) I-tokens. In the top trajectory, we observe more than 30 seconds of gameplay generated by Δ-IRIS' world model. A wide variety of mechanics have been internalized: scrolling, chopping down trees, building a crafting table, mining iron, crafting pickaxes, etc. However, removing I-tokens from the sequence of the autoregressive transformer makes the task of predicting future Δ-tokens drastically harder as evidenced by the agent glitching through walls and water in the bottom trajectory. These mistakes ultimately hinder the policy improvement phase, since the agent will reinforce behaviours in a world that does not properly reflect its environment.

We believe that achieving higher returns at the 10M frames cap poses a hard exploration problem. Indeed, three of the missing four tasks require crafting new tools in the presence of a nearby crafting table and furnace. Discovering these tools with a naive exploration strategy is highly unlikely, and we have observed only a few occurrences of those events throughout training runs.

With too few training samples, the world model is unable to internalize these new mechanics and reflect them during the imagination procedure. We hypothesize that a biased data sampling procedure (Kauvar et al., 2023) could be the key to unlock the missing achievements.

3.3. World model analysis

In Section 3.2, we validated our design choices for Δ-IRIS with RL experiments. However, downstream RL performance is an imperfect proxy for the quality of a world model due to many possible confounding factors, e.g. the choice of the RL algorithm, entangled world model and policy architectures, or the continual learning loop. In this section, we directly focus on the abilities of the world model.

Figure 5 illustrates the bottom 1% autoencoded test frames with and without conditioning the autoencoder on the ongoing trajectory (i.e. reconstructions with Δ-IRIS vs IRIS). With as few as 4 tokens per frame, Δ-IRIS' autoencoder is able to encode frames with minimal loss. On the other hand, without access to previous frames and actions, and even with 16 tokens, IRIS' autoencoder produces poor reconstructions.

Figure 6 displays trajectories imagined to illustrate whether crucial mechanics have been internalized by the world model, when including I-tokens in the sequence of the autoregressive transformer or not. We observe that, with I-tokens, a multitude of game mechanics are well understood, but in the absence of I-tokens the world model is unable to simulate key concepts. Appendix B includes additional quantitative results.

3.4. Evidence of dynamics disentanglement

In Section 2.2, we argued that, by design, Δ-IRIS' encoder describes stochastic deltas between timesteps with Δ-tokens. In the present section, we propose to exhibit this phenomenon.

We pick a starting frame and a sequence of actions, and predict two different trajectories with the world model. In one case, we sample future Δ-tokens randomly. In the other case, Δ-tokens are produced by the autoregressive transformer. We consider a scenario where the agent collects wood then builds a crafting table in Figure 3. Appendix F displays two other scenarios where the agent explores its surroundings, and where it moves down then stands still.

We observe that, even when sampling Δ-tokens randomly, the deterministic aspects of the dynamics are properly modelled: grid layout, agent movement, wood level increasing, crafting table appearing, etc. On the other hand, stochastic dynamics become problematic: skeletons and cows appearing and disappearing, food and water indicators decreasing too early, unlikely quantities of enemies and objects, etc. These observations confirm that Δ-IRIS encodes stochastic deltas between time steps with Δ-tokens, and its decoder handles the deterministic aspects of world modelling.

4. Related Work

World Models and imagination

With Dyna, Sutton (1991) introduced the idea of learning behaviours in the imagination of a world model. Ha & Schmidhuber (2018) went beyond the tabular setting and proposed a new world model architecture, composed of a variational autoencoder (Kingma & Welling, 2013) and a recurrent network (Hochreiter & Schmidhuber, 1997; Gers et al., 2000), capable of simulating simple visual environments. Following this breakthrough, multiple generations of Dreamer agents (Hafner et al., 2020; 2021; 2023) were developed, with DreamerV2 being the first imagination-based agent to outperform humans in Atari games, and DreamerV3 being the first world model architecture applicable to a wide range of domains without any specific tuning. DreamerV2 learns in the imagination of a world model combining a convolutional autoencoder with a recurrent state-space model (RSSM) (Hafner et al., 2019). The key modifications that enabled DreamerV2 to improve over the original Dreamer agent were categorical latents and KL balancing between prior and posterior estimates. DreamerV3 builds upon DreamerV2 with more universal design choices such as symlog scaling of rewards and values, combining free bits (Kingma et al., 2016) with KL balancing, return scaling for static entropy regularization, and architectural novelties for model scaling. Variants of Dreamer such as TransDreamer (Chen et al., 2022) and STORM (Zhang et al., 2023) have also been explored, where transformers replace the recurrent network in the RSSM for dynamics prediction.

A potential limitation of RSSM-like architectures is that they do not model the joint distribution of future latent states, and instead predict product laws. One way to mitigate this discrepancy between the predicted distributions and the distributions of interest is to encourage factorized distributions (Hafner et al., 2023). On the other hand, autoregressive architectures (Micheli et al., 2023) do model the joint distribution and do not require to enforce independence, which may result in a more expressive model.

Trajectory and video autoencoders

The idea of encoding frames with respect to past frames predates modern deep learning, and is at the origin of efficient video compression algorithms, such as MPEG (Richardson, 2004). In recent years, multiple works have implemented variants of this approach. Ozair et al. (2021) propose an offline version of MuZero (Schrittwieser et al., 2020) equipped with an autoregressive transformer that performs search over trajectory-level discrete latent variables and actions. Phenaki (Villegas et al., 2022) is a text-to-video model composed of a spatio-temporal discrete autoencoder and a masked bidirectional transformer. TECO (Yan et al., 2022) is an action-conditional video prediction model composed of a discrete frame autoencoder conditioned on the previous frame, a temporal autoregressive transformer, and a spatial MaskGit (Chang et al., 2022). While these methods also encode frames by conditioning on past frames, their dynamics models purely operate over discrete tokens, and do not leverage continuous tokens to alleviate the need to integrate over multiple time steps in order to make the next prediction.

Hafner et al. (2019) acknowledge that modelling stochastic dynamics may be difficult, as it would involve remembering information from previous time steps. The authors propose to solve this problem by carrying a "deterministic" state over time via a recurrent network, at the core of their RSSM. We make a similar observation, and further show that this task is still difficult even when past information does not have to be carried by a recurrent state, as a transformer can attend to all previous Δ-tokens. Hence, it is not only a memory problem, but also a modelling one. Here, we address this issue in a manner that is compatible with autoregressive transformers, namely by injecting continuous I-tokens in the sequence of the dynamics model.

5. Conclusion

We introduced Δ-IRIS, a new model-based agent relying on an efficient world model architecture to simulate its environment and learn new behaviours. Δ-IRIS features a discrete autoencoder that encodes the stochastic aspects of world modelling with discrete Δ-tokens, and an autoregressive transformer leveraging continuous I-tokens to model stochastic dynamics.

Through experiments, we showed the ability of our agent to scale to the challenging Crafter benchmark, as well as its sample efficiency in Atari100k. Finally, we illustrated how its world model internalized environment dynamics, and conducted ablations to validate our proposed design choices.

In its current form, Δ-IRIS uses the same number of tokens to encode stochastic dynamics at each time step. However, the reality of most environments is such that periods of low uncertainty are quickly followed by moments of high randomness. Therefore, an improved version of the world model could possibly predict dynamically various numbers of tokens based on the current context. Besides, leveraging the internal representations of the world model could potentially result in a lightweight and more robust policy.

Impact Statement

The deployment of autonomous agents in real-world applications raises safety concerns. Agents learning new behaviours may harm individuals and damage property. With world models, we lower the amount of time spent interacting with the real world and thus mitigate risks. In this work, we propose a world model architecture that is amenable to scaling up to complex environments, where accurate simulations are even more critical given the usually higher stakes.

Acknowledgements

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34, 2021.

Anand, A., Walker, J. C., Li, Y., Vértes, E., Schrittwieser, J., Ozair, S., Weber, T., and Hamrick, J. B. Procedural generalization by planning with self-supervised world models. In *International Conference on Learning Representations*, 2022.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Chang, H., Zhang, H., Jiang, L., Liu, C., and Freeman, W. T. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

Chen, C., Wu, Y.-F., Yoon, J., and Ahn, S. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022.

comma.ai. commavq, 2023. URL https://github.com/commaai/commavq.

D'Oro, P., Schwarzer, M., Nikishin, E., Bacon, P.-L., Bellemare, M. G., and Courville, A. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *The Eleventh International Conference on Learning Representations*, 2023.

Esser, P., Rombach, R., and Ommer, B. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12873–12883, 2021.

Gers, F. A., Schmidhuber, J., and Cummins, F. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 2000.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Hafner, D. Benchmarking the spectrum of agent capabilities. In *International Conference on Learning Representations*, 2022.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Hafner, D., Lillicrap, T. P., Norouzi, M., and Ba, J. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104v1*, 2023.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Hu, A., Russell, L., Yeo, H., Murez, Z., Fedoseev, G., Kendall, A., Shotton, J., and Corrado, G. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Imani, E. and White, M. Improving regression performance with distributional losses. In *International conference on machine learning*, pp. 2157–2166. PMLR, 2018.

Kaiser, Ł., Babaeizadeh, M., Miłos, P., Osiński, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. Model based reinforcement learning for atari. In *International Conference on Learning Representations*, 2020.

Kanervisto, A., Milani, S., Ramanauskas, K., Topin, N., Lin, Z., Li, J., Shi, J., Ye, D., Fu, Q., Yang, W., Hong, W., Huang, Z., Chen, H., Zeng, G., Lin, Y., Micheli, V., Alonso, E., Fleuret, F., Nikulin, A., Belousov, Y., Svidchenko, O., and Shpilman, A. Minerl diamond 2021 competition: Overview, results, and lessons learned. In *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, Proceedings of Machine Learning Research, 2022.

Kapturowski, S., Ostrovski, G., Quan, J., Munos, R., and Dabney, W. Recurrent experience replay in distributed reinforcement learning. In *International conference on learning representations*, 2019.

Kauvar, I., Doyle, C., Zhou, L., and Haber, N. Curious replay for model-based adaptation. In *International Conference on Machine Learning*, 2023.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

LeCun, Y. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62, 2022.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Micheli, V., Alonso, E., and Fleuret, F. Transformers are sample-efficient world models. In *International Conference on Learning Representations*, 2023.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937. PMLR, 2016.

Ozair, S., Li, Y., Razavi, A., Antonoglou, I., Van Den Oord, A., and Vinyals, O. Vector quantized models for planning. In *International Conference on Machine Learning*, pp. 8302–8313. PMLR, 2021.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners, 2019.

Razavi, A., van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Richardson, I. E. *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia*. John Wiley & Sons, 2004.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T. P., and Silver, D. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., and Bachman, P. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021.

Schwarzer, M., Ceron, J. S. O., Courville, A., Bellemare, M. G., Agarwal, R., and Castro, P. S. Bigger, better, faster: Human-level atari with human-level efficiency. In *International Conference on Machine Learning*, 2023.

Sekar, R., Rybkin, O., Daniilidis, K., Abbeel, P., Hafner, D., and Pathak, D. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pp. 8583–8592. PMLR, 2020.

Sutton, R. S. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 1991.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Villegas, R., Babaeizadeh, M., Kindermans, P.-J., Moraldo, H., Zhang, H., Saffar, M. T., Castro, S., Kunze, J., and Erhan, D. Phenaki: Variable length video generation from open domain textual description. *arXiv preprint arXiv:2210.02399*, 2022.

Yan, W., Hafner, D., James, S., and Abbeel, P. Temporally consistent video transformer for long-term video prediction. *arXiv preprint arXiv:2210.02396*, 2022.

Ye, W., Liu, S., Kurutach, T., Abbeel, P., and Gao, Y. Mastering atari games with limited data. *Advances in neural information processing systems*, 34, 2021.

Yu, J., Li, X., Koh, J. Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., and Wu, Y. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

Zhang, W., Wang, G., Sun, J., Yuan, Y., and Huang, G. Storm: Efficient stochastic transformer based world models for reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

A. Architectures and hyperparameters

A.1. Discrete autoencoder

Table 2. Encoder / Decoder hyperparameters. We list the hyperparameters for the encoder, the same ones apply for the decoder.

Hyperparameter	Value
Frame dimensions (h, w)	64×64
Layers	5
Residual blocks per layer	2
Channels in convolutions per layer	$[64, 64, 128, 128, 256]$
Downsampling after layer n	$[1, 0, 1, 1, 0]$
Past actions embedding channels	4
Decoder past frames embedder arch.	Same as encoder
Decoder past frames embedder output feature map size	$8 \times 8 \times 8$
Conditioning time steps	1
L_1 loss weight	0.1
L_2 loss weight	1.0
Max-pixel loss weight	0.01
Commitment loss weight	0.02

Table 3. Embedding table and latent state hyperparameters.

Hyperparameter	Value
Vocabulary size (N)	1024
Tokens per frame (K)	4
Latent feature map size	$64 \times 8 \times 8$
Pre-discretization token size	$64 \times 4 \times 4$
Token embedding dimension	64
Codebook moving average coefficient	0.99

In early experiments, we used a transformer instead of a CNN for the architecture of the autoencoder. It had a much longer context size of twenty time steps. Although the transformer-based autoencoder performed better than its CNN counterpart on static datasets, we observed that the CNN would learn faster than the transformer in the continual learning setup. Besides, for the sake of simplicity, we decreased the initial conditioning of the CNN autoencoder from four time steps to one time step, as the slight increase in reconstruction losses did not significantly hinder agent performance. These observations are largely environment-dependent, thus the context size or the architecture of the autoencoder should most likely be adapted accordingly.

A.2. Autoregressive transformer

Table 4. Transformer hyperparameters.

Hyperparameter	Value
Timesteps	21
Embedding dimension	512
Layers	3
Attention heads	8
Weight decay	0.01
I-token frame embedder arch.	Same as encoder with halved channels per layer

A.3. Actor-Critic

We tie the weights of the actor and critic, except for the last layer. The actor-critic takes as input a frame, and forwards it through a convolutional neural network (LeCun et al., 1989) followed by an LSTM cell (Hochreiter & Schmidhuber, 1997; Gers et al., 2000; Mnih et al., 2016). For the CNN, we use the same architecture as the encoder, except that we halve the number of channels per layer. The dimension of the LSTM hidden state is 512.

Before starting the imagination procedure ($H = 15$) from a given frame, we burn-in (Kapturowski et al., 2019) the 5 previous frames to initialize the hidden state. The discount factor γ is 0.997, the parameter for λ-returns is set to 0.95, and the coefficient for the entropy maximization term is 0.001. Targets for value estimates are produced by a moving average of the critic network, with update parameter 0.995 (Mnih et al., 2015)

A.4. Training loop and shared hyperparameters

Table 5. Training loop and shared hyperparameters.

Hyperparameter	Value	Hyperparameter	Value
Epochs	1000	Autoencoder batch size	32
Environment steps first epoch	100000	Transformer batch size	32
Environment steps per epoch	10000	Actor-critic batch size	86
# Collection epochs	990	Learning rate	1e-4
Collection epsilon-greedy	0.01	Optimizer	Adam
Training steps per epoch	500	Max gradient norm	10.0

As mentioned in Section 2, the world model and policy are trained with temporal segments sampled from past experience. We use a count-based sampling procedure over the entire history of episodes, i.e. the likelihood that a given episode is chosen to produce the next sample is inversely proportional to the number of times it was previously used. We raise inverse counts to the power of 5 to further limit the bias towards older episodes.

B. Impact of design choices on key world modelling metrics

Metrics are computed on a held-out test set after training various world models on a dataset consisting of 10M frames collected by a Δ-IRIS agent throughout its training.

Table 6. Left: Impact of removing past frames and actions from the conditioning of the autoencoder (Δ-IRIS $\rightarrow$ IRIS). Right: Impact of removing I-tokens from the conditioning of the autoregressive transformer.

Method	L_2 loss
Δ-IRIS (4 tokens)	0.000185
IRIS (64 tokens)	0.001715
IRIS (16 tokens)	0.007496

Method	Next token loss (CE)	Reward loss (CE)
Δ-IRIS	1.57	0.108
Δ-IRIS w/o I-tokens	1.73	0.135

Table 7. Impact of discarding the auxiliary max-pixel loss.

Method	L_2 loss	Max-pixel loss
Δ-IRIS	0.000185	0.018
Δ-IRIS w/o max-pixel loss	0.000178	0.031

C. Atari 100k

The Atari 100k benchmark (Kaiser et al., 2020) features Atari games (Bellemare et al., 2013) with diverse mechanics. The specificity of this benchmark is the hard constraint on the number of interactions, namely one hundred thousand per environment. Compared to the standard Atari benchmark, this constraint results in a dramatic drop in real-time experience, from 900 hours to 2 hours.

Regarding baselines, we consider four model-based RL agents learning in imagination: SimPLe (Kaiser et al., 2020), DreamerV3 (Hafner et al., 2023), STORM (Zhang et al., 2023), and IRIS (Micheli et al., 2023). We note that the current best performing methods for Atari 100k resort to other approaches, such as lookahead search for EfficientZero (Ye et al., 2021), or self-supervised representation learning with periodic resets for BBF (Schwarzer et al., 2023).

The usual metric of interest is the HNS, the human-normalized score, based on the performance of human players with similar experience. A negative HNS indicates worse than random performance whereas an HNS above 1 signifies superhuman performance. We evaluate Δ-IRIS by computing an average over 100 episodes collected at the end of training for each game (5 seeds). For the baselines, we report the published results.

Table 8 displays returns across games and aggregate metrics (Agarwal et al., 2021). Δ-IRIS achieves higher aggregate metrics than IRIS, while training in 26 hours, a 5-fold speedup.

Table 8. Returns on the 26 games of Atari 100k after 2 hours of real-time experience, and human-normalized aggregate metrics.

Game	Random	Human	SimPLe	DreamerV3	STORM	IRIS	Δ-IRIS (ours)
Alien	228	7128	617	959	**984**	420	391
Amidar	6	1720	74	139	**205**	143	64
Assault	222	742	527	706	801	**1524**	1123
Asterix	210	8503	1128	932	1028	854	**2492**
BankHeist	14	753	34	649	641	53	**1148**
BattleZone	2360	37188	4031	12250	**13540**	13074	11825
Boxing	0	12	8	78	**80**	70	70
Breakout	2	31	16	31	16	84	**302**
ChopperCommand	811	7388	979	420	**1888**	1565	1183
CrazyClimber	10781	35829	62584	**97190**	66776	59324	57864
DemonAttack	152	1971	208	303	165	**2034**	533
Freeway	0	30	17	0	**34**	31	31
Frostbite	65	4335	237	909	**1316**	259	279
Gopher	258	2413	597	3730	**8240**	2236	6445
Hero	1027	30826	2657	**11161**	11044	7037	7049
Jamesbond	29	303	101	445	**509**	463	309
Kangaroo	52	3035	51	4098	**4208**	838	2269
Krull	1598	2666	2205	7782	**8413**	6616	5978
KungFuMaster	259	22736	14863	21420	**26182**	21760	21534
MsPacman	307	6952	1480	1327	**2674**	999	1067
Pong	-21	15	13	18	11	15	**20**
PrivateEye	25	69571	35	882	**7781**	100	103
Qbert	164	13455	1289	3405	**4523**	746	1444
RoadRunner	12	7845	5641	15565	**17564**	9615	10414
Seaquest	68	42055	683	618	525	661	**827**
UpNDown	533	11693	3350	**9234**	7985	3546	4072
#Superhuman	0	N/A	1	9	10	10	**11**
Mean	0.00	1.00	0.33	1.10	1.27	1.05	**1.39**
Interquartile Mean	0.00	1.00	0.13	0.50	0.64	0.50	**0.65**

D. Crafter scores and individual success rates

Table 9. Crafter scores, i.e. geometric mean of success rates.

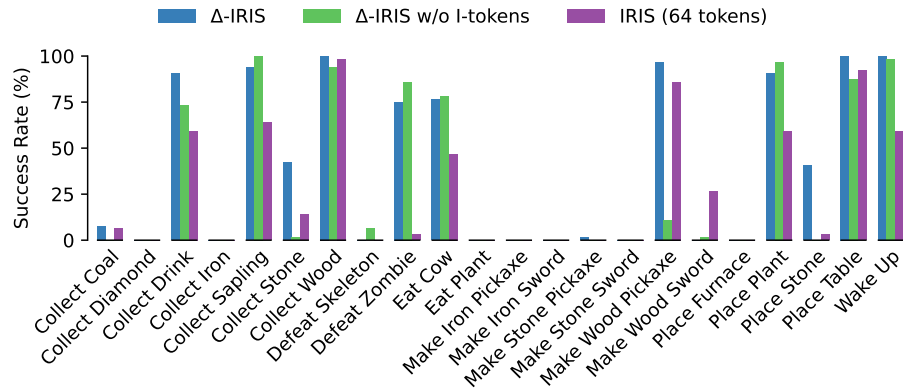| Method	Crafter score @1M	Crafter score @5M	Crafter score @10M
Δ-IRIS	9.30	39.67	42.47
Δ-IRIS w/o I-tokens	5.85	14.39	25.92
IRIS (64 tokens)	6.66	-	-

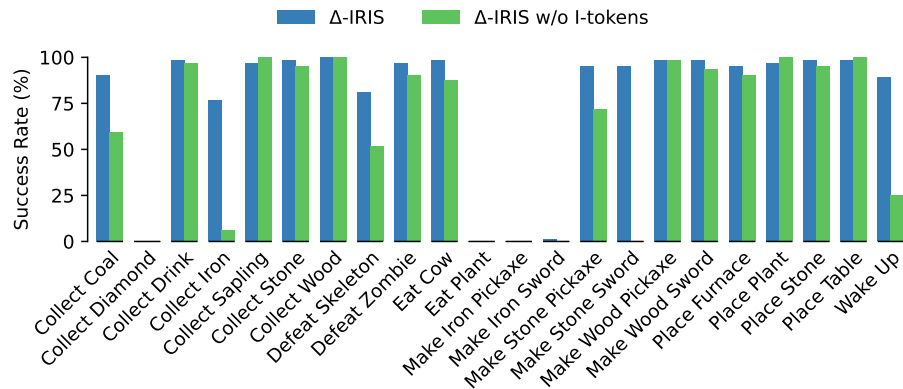*Figure 7.* Individual success rates after collecting 1M frames.

Figure 8. Individual success rates after collecting 10M frames.

E. Baselines

DreamerV3 results were obtained with commit 8fa35f8. We used the standard configuration for Crafter, and set the *run.train_ratio* variable controlling the imagined-to-collected data ratio to 64. Note that a new version of DreamerV3 was recently released in April 2024. This update includes additional and broadly applicable novelties for world model and policy learning.

IRIS results were obtained with commit ac6be40. For the training loop and shared hyperparameters, we picked the same values as in Table 5. We increased the dimension and attention heads of the transformer from 256 and 4 to 512 and 8, respectively. Finally, we used a replay buffer with a capacity of 1M frames.
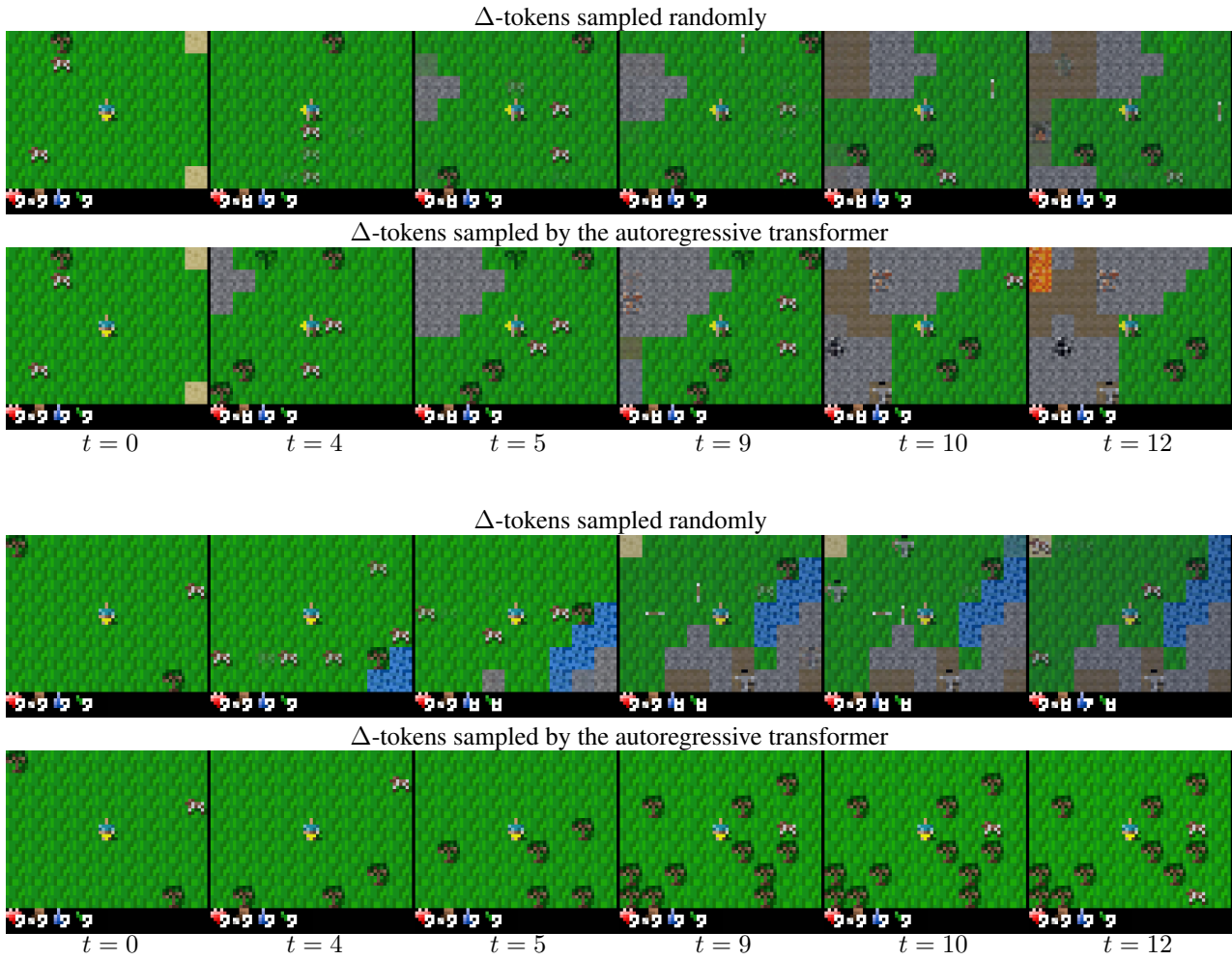
F. Evidence of dynamics disentanglement

Δ-tokens sampled randomly

Δ-tokens sampled by the autoregressive transformer

$t = 0$ $t = 4$ $t = 5$ $t = 9$ $t = 10$ $t = 12$

Δ-tokens sampled randomly

Δ-tokens sampled by the autoregressive transformer

$t = 0$ $t = 4$ $t = 5$ $t = 9$ $t = 10$ $t = 12$

Figure 9. Two additional examples of dynamics disentanglement, as discussed in Section 3.4 and Figure 3.