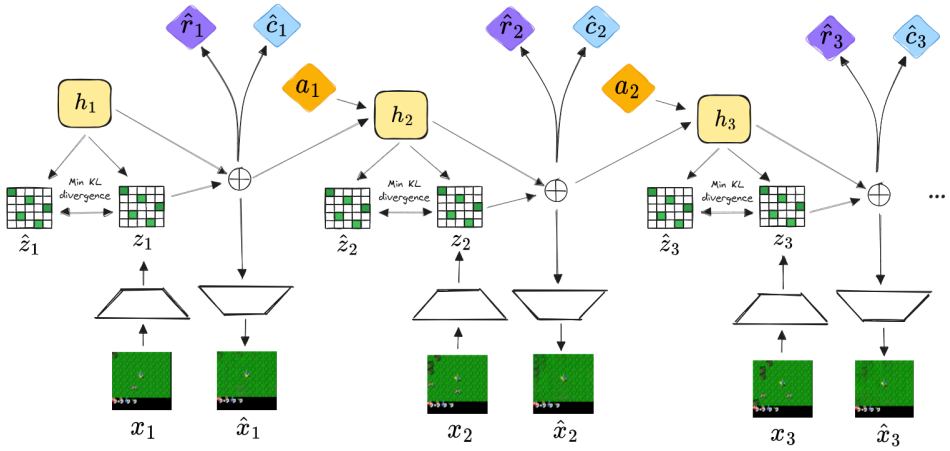


Figure 1: The World Model learning architecture of the Dreamer V3 algorithm. The learned latent-state representation consist of a deterministic component h, and a discrete stochastic component z. The representation is trained to be informative for reconstructing the observations $\hat{x}$, and predicting the environment's rewards and continuation $\hat{r}, \hat{c}$. The dynamics predictor $\hat{z}$ is trained to produce the stochastic component z *without* access to the environment observation(s), which is used downstream to enable planning in the world model.

A Technical Appendices and Supplementary Material

A.1 Broader Impacts

We hope that our work may increase the interpretability and trustworthiness of deep RL systems, such that a user can trust that what a Somniloquy-based agent says it will do is what it actually does. As with any deep learning based system, care needs to be paid as to the accuracy of the model (translation, in this case), and careful consideration made to any real world deployment.

A.2 Overview of Dreamer

As shown in Figure 1 and given formally below, Dreamer's world model consists of several components [1, p.648]:

$$
\begin{aligned}
\text{Sequence model: } & h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1}) \\
\text{Encoder: } & z_t \sim q_\phi(z_t \mid h_t, x_t) \\
\text{Dynamics predictor: } & \hat{z}_t \sim p_\phi(\hat{z}_t \mid h_t) \\
\text{Reward predictor: } & \hat{r}_t \sim p_\phi(\hat{r}_t \mid h_t, z_t) \\
\text{Continue predictor: } & \hat{c}_t \sim p_\phi(\hat{c}_t \mid h_t, z_t) \\
\text{Decoder: } & \hat{x}_t \sim p_\phi(\hat{x}_t \mid h_t, z_t)
\end{aligned}
$$

Where $\hat{r}_t$ denotes the model's predicted reward at timestep t, and $\hat{c}_t$ denotes the model's binary prediction as to whether the episode continues (i.e., if the current state is non-terminal) at timestep t.

Dreamer's model components are trained in a supervised manner, where batches of sequences of length T of observations, rewards, and continuation flags are sampled from a replay buffer. The world model loss consists of three components: a *prediction* loss, a *dynamics* loss, and a *representation* loss [1, p.648]:

$$
\begin{aligned}
\mathcal{L}_{pred} &\doteq -\ln p_\phi(x_t \mid z_t, h_t) - \ln p_\phi(r_t \mid z_t, h_t) - \ln p_\phi(c_t \mid z_t, h_t) \\
\mathcal{L}_{dyn} &\doteq \max(1, \mathrm{KL}[\, \mathrm{sg}(q_\phi(z_t \mid h_t, x_t)) \parallel p_\phi(z_t \mid h_t)\,]) \\
\mathcal{L}_{rep} &\doteq \max(1, \mathrm{KL}[\, q_\phi(z_t \mid h_t, x_t) \parallel \mathrm{sg}(p_\phi(z_t \mid h_t))\,])
\end{aligned}
$$

Where KL[‖] denotes the Kullback-Leibler divergence [2], and sg() denotes the 'stop gradient' operator, which prevents any model parameters wrapped by the operator from being updated via the gradients with respect to the current loss function.

The prediction loss is used to ensure that the learned latent representation is informative for reconstructing the observations, predicting rewards, and predicting terminal states. The dynamics loss trains the dynamics predictor to reconstruct the 'true' stochastic latent state, *without* access to the environment observation, whilst ensuring some differences to prevent a degenerate solution. The representation loss is used to make the 'true' dynamics easier to predict, whilst again enforcing some difference. These losses are summed together and averaged over the sampled sequence, with different weights β given to each of the three losses:

$$\mathcal{L}_{world} = \frac{1}{T}\sum_{t=1}^{T}[\beta_{pred}\mathcal{L}_{pred} + \beta_{dyn}\mathcal{L}_{dyn} + \beta_{rep}\mathcal{L}_{rep}]$$

The dynamics predictor is the component that enables planning from a given starting environment observation x_t. The observation is encoded and the initial latent state is computed using the learned world model, yielding (h_t, z_t). Then, an action is chosen from the agent's policy a_t. The action and latent state is then input to the sequence model to yield h_{t+1}, which is then input to the dynamics predictor to obtain $\hat{z}_{t+1}$, to form an estimate of the latent state at period $t+1$. The latent state estimate can then be used to query the policy for another action, and the process can repeat itself for an arbitrary number of steps, which results in a sequence of latent states and actions: a latent state plan.

The agent's policy is trained entirely within the world model, in an actor-critic based approach. Details are omitted for brevity, but to summarise, the actor is an MLP that learns to map from latent states to a distribution over actions, whilst the critic is an MLP that is trained to estimate the value of latent states.

A.3 Somniloquy Translation Evaluation

The evaluation algorithm for estimating the translation performance of Somniloquy in deterministic environments can be found below in Algorithm 1, and for stochastic environments in Algorithm 2.

Algorithm 1: Somniloquy Translation Evaluation for Deterministic Environments

parameters : number of evaluation episodes S, latent plan length T

translation_scores ← [ ];
for *episode in* S **do**
  starting_obs ← env.**reset()**;
  while *not terminal* **do**
    planned_actions, planned_latent_states, pred_terminal ← **plan_latent_sequence**(starting_obs, T);
    env_states, env_terminal ← **execute_plan**(env, planned_actions, starting_obs);
    translated_plan_description ← **latent_translator**(planned_latent_states);
    true_plan_description ← **narrator**(env_states);
    translation_scores.**append(score**(translated_plan_description,true_plan_description)**)**
    staring_obs ← true_obs$[-1]$;
    terminal ← pred_terminal or env_terminal;
  end
end
return *translation_scores.**mean()***

A.4 Environment Details

An overview and example observation for each environment used in our experiments can be found in Figure 2. As a reminder, to evaluate the translation performance of Somniliquy, we use Crafter [3] and AI2THOR [4] as our two deterministic environments, and create our own stochastic gridworld environment using the MiniGrid domain [5]. To evaluate the ability of Somniliquy to turn language

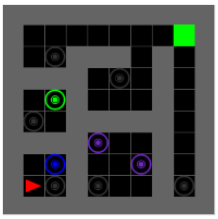
Algorithm 2: Somniloquy Translation Evaluation for Stochastic Environments

parameters : number of evaluation episodes S, latent plan length T, plan rollouts N

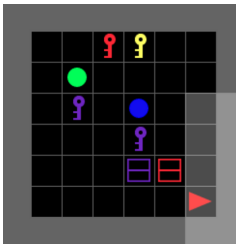
empirical_dynamics ← *None*;
true_dynamics ← env.**get_dynamics**();
for *episode in* S **do**
    starting_obs ← env.**reset**();
    while *not terminal* **do**
        sampled_translations ← [ ];
        for *plan_sample in* N **do**
            planned_actions, planned_latent_states ← **plan_latent_sequence**(starting_obs, T);
            translated_plan_description ← **latent_translator**(planned_latent_states);
            sampled_translations.**append**(translated_plan_description);
        end
        empirical_dynamics ← **update_estimate**(sampled_translations);
        terminal, true_obs ← **execute_policy**(env, agent, starting_obs, T);
        staring_obs ← true_obs$[-1]$;
    end
end
return ***score****(empirical_dynamics, true_dynamics)*

(a) An example observation from the Crafter environment. The observation shows the agent in the procedurally generated world, along with their status (health, hunger, thirst, and energy) and the items in their inventory. The agent is rewarded the first time it completes an achievement such as harvesting wood.

(b) An example observation from the AI2THOR environment. The observation shows a first person perspective from the robotic agent acting in a virtual kitchen. The agent is rewarded the first time it picks up each unique object in the scene, and the episode terminates after a fixed number of timesteps, or after every unique object has been picked up.



(c) An example observation from the Minigrid stochastic environment. The agent (red triangle) is tasked to navigate to the goal (green square) utilising a set of teleporters (one blue, one green, and two purple). Each teleporter randomly takes the agent to one of a sub-set of teleporter locations, denoted by the grey circles.



(d) An example observation from the BabyAI GoToLocal environment. The agent (red triangle) is tasked, in natural language, to go to one of the coloured objects present in the scene. To go to an object, the agent must both be in a cell adjacent to the object (non-diagonally), and be facing towards the object.

Figure 2: Environments used to evaluate Somniloquy, both for translation (a-c) and language to policy (d)

into a reward function, we use the BabyAI environment [6]. Details for each environment are as follows:

Crafter. We use an updated version of Crafter[1] that makes it compatible with the latest Gymnasium API. To make Crafter deterministic across a given seed, we use that given seed to fix the world generated across all training and evaluation episodes, since "all randomness in the environment is uniquely determined by an integer seed" [3, p.3]. The agent is given a reward of $+1$ the first time it unlocks an achievement (such as harvesting wood) in a given episode, and a small -0.1 and $+0.1$ reward for losing/regenerating health. The action space of Crafter consists of 17 discrete actions, and

[1] by Chris Taylor, available at `https://github.com/catid/crafter`

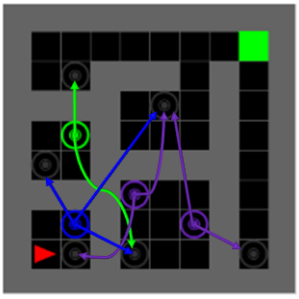


Figure 3: The stochastic minigrid environment with possible teleport locations for each teleporter are shown annotated with arrows.

each episode terminates when the agent's health reaches 0, or after a maximum number of steps is reached.

AI2THOR. We create our own task in the AI2THOR domain, using the `FloorPlan10` kitchen environment. The agent is given a reward of $+1$ the first time it picks up each unique, pickup-able, object in the scene, for a maximum reward of 33. We discretise the action space into 24 actions. For actions that require parameters (e.g., the `pickup` action requires the ID of the object to pickup), we automatically fill those parameters with the closest object that would satisfy them. If no such object exists (e.g., the agent chooses to take the `pickup` action, but there are no 'pickup-able' objects in range), the agent takes a `no-op` action. Each episode terminates once every unique item has been picked up at least once, or after a maximum number of steps is reached.

Stochastic MiniGrid. We introduce a teleporter object to the MiniGrid domain, and create our own stochastic environment in which an agent must navigate to the goal by using a set of teleporters. When an agent takes an action that would place it in the cell containing a teleporter, the agent is teleported to one of a set of teleporter destination cells, sampled according to a fixed probability distribution. The possible transitions for each teleporter object in our environment is shown in Figure 3, where the distribution is uniform for each teleporter. Consistent with other MiniGrid environments, the agent is given a reward of $1 - 0.9(\text{step\_count}/\text{max\_steps})$ if it reaches the goal, to encourage reaching the goal as quick as possible. The action space consists of 3 discrete actions for moving and turning the agent, and the episode terminates if the agent reaches the goal or after a maximum number of steps has been reached. We give the agent full RGB observability.

BabyAI GoToLocal. We use the BabyAI-GoToLocal-v0 configuration of the environment, with a fixed seed of 100 for environment generation across all our experiments, which results in a 6 by 6 grid of navigable cells. This version of the environment consists of two purple keys, a red key, a yellow key, a blue ball, a green ball, a purple box, and a red box. We restrict the agent to only using navigation actions, resulting in a discrete action space of three actions. In training, there are no terminal states, as we assume no extrinsic reward function. In evaluation, we write our own ground-truth reward function, and state termination function, for each of the four language goals. Namely, the agent is given a reward of 1.0 when it reaches the requested object (is adjacent and facing the object), which then terminates the environment in this state. We give the agent full RGB observability.

A.5 Narrator Details

We create a rule-based narrator for each environment that takes in a sequence of subsets of environment states, and outputs a natural language description regarding what happened to the agent and the world in the given sequence.

Crafter. The Crafter narrator requires the agent's inventory, status (hunger, thirst, and energy levels), achievement count (e.g., number of tables crafted), and local semantic grid that specifies the visible object types and entities in a 7x9 radius surrounding the agent. The narrator then generates text of the form: "I will see `<VISIBLE OBJECTS & ENTITIES>`, I will harvest `<INVENTORY CHANGES>`, I will craft `<INVENTORY CHANGES>`, I will fight/eat/place `<ACHIEVEMENTS>`, `<VITALS INFORMATION>`". If the agent dies or is hurt during the sequence, the `<VITALS INFORMATION>` will contain the name

of the monster that killed/attacked them. Including special tokens (beginning of sequence, end of sequence, etc.), there are a total of 97 unique tokens that can be output by the narrator.

AI2THOR. The AI2THOR narrator requires the agent's position (x, y, z) in the scene, along with all object interactions in the sequence. The object interactions contain information on all possible interaction types for each object, such as picking up, opening, closing, slicing, and turning on the object, etc. The narrator generates text of the form "I start near the <LANDMARK> and <RELATIVE MOVEMENT>. I will <OBJECT INTERACTION>". Where <RELATIVE MOVEMENT> is replaced with a string describing the temporal movement of the agent relative to landmarks in the scene, and <OBJECT INTERACTION> is replaced with the temporal order of interaction that the agent had with the object(s) in the sequence. Including special tokens, there are a total of 109 unique tokens that can be output by the narrator.

Stochastic MiniGrid. The Stochastic MiniGrid narrator requires the full semantic map of the grid world that contains the location, colour, and type, of entities in the grid. Each 'room' in the grid is given a unique name, and the narrator generates strings describing the agent's movement relative to the goal and room names, along with which teleporters were used and where they teleported the agent to, for example "I start in the blue teleporter room and I go through the blue teleporter to the purple teleporter room...". Including special tokens, there are a total of 36 unique tokens that can be output by the narrator.

BabyAI GoToLocal. The BabyAI narrator requires the full semantic map of the grid world that contains the location, colour, and type, of entities in the grid, plus the agent's directional vector. Given a sequence of such observations, the narrator generates strings describing what (if any) objects the agent went to (as defined by being adjacent, and facing, the object), such as "First I will go the the red key and then I will go to the green ball...". If the agent doesn't move, it outputs "I will not move", and if it doesn't go to any object it will output "I will move but not go to anything". Including special tokens, there are a total of 29 unique tokens that can be output by the narrator.

A.6 Model Details & Parameters

To learn the latent state representation, we utilise the open-source PyTorch implementation of the Dreamer V3 algorithm[2] (licensed under the MIT license), since we are more familiar with the PyTorch library. The PyTorch implementation has some differences over the official Dreamer V3 implementation, namely:

1. Use of ADAM optimizer instead of LaProp
2. Vanilla gradient clipping instead of Adaptive Gradient Clipping (AGC)
3. No replay loss in the Critic component
4. No GRU optimisation

These difference do not seem to have much (if any) of an impact on the performance of the algorithm, as can be seen from the results in the author's repository.

We use the 100M parameter version of Dreamer, as given in the Dreamer V3 paper [7, *Extended Data Table 4*]. Due to the non-optimisation of the GRU component, the PyTorch version gives around 200M parameters total when following the official 100M parameter specification. We provide the 100M Dreamer model size and parameter details, along with Somniloquy's encoder-decoder transformer parameters, in Table 1.

A.7 Model Hyperparameters

The world-model hyperparameters mostly follow that of Dreamer V3, as given in *Extended Data Table 5* in the Dreamer V3 paper [7]. Details of Somniloquy's hyperparameters can be found in Table 2, where differences with Dreamer's hyperparameters are highlighted in red.

[2]Created by Naoki Morihira, available at `https://github.com/NM512/dreamerv3-torch`

Table 1: Somniloquy Model Size

Name	Value
RSSM	
Deterministic latent units	6,144
Hidden size	768
Codes per stochastic latent	48
Number of stochastic latents	32
Base Encoder CNN channels	48
CNN Encoder kernel size	4
CNN Encoder stride	2
CNN Decoder	
Base CNN channels	48
Kernel size	4
Stride	2
Actor & Critic	
MLP units	768
MLP layers	3
Reward & Continuation Heads	
MLP units	768
MLP layers	1
Translation Component	
Latent state embedding dimension (d)	256
Token embedding dimension	256
Feed-forward units	256
Attention heads	2
Encoder attention layers	2
Decoder attention layers	2

A.8 Environment Specific Hyperparameters

Each environment has a specific set of hyperparameters that can be found in Table 3. *Max Steps* is the maximum number of environment steps allowed in an episode before it terminates; *Training Steps* are the total number of environment steps Somniloquy is trained on each each environment, which is approximate for AI2THOR and Crafter as we train for a total of 48-hours; *Eval Every* determines after how many environment steps the evaluation procedures are run; *Train Ratio* determines the ratio of model-update steps to environment update steps, the higher this is the higher model-update steps are relative to environment steps; *Vocab Size* is the number of unique possible words that the environment's narrator can output, plus special tokens; and *Translation Max Length* is the maximum number of tokens that can be generated by the decoder transformer for a single plan translation.

For the BabyAI environment, the parameters listed are for training the latent dynamics model. When evaluating the performance of the language-goal achieving policy, the maximum number of steps in the environment is equal to the planning horizon of the trained agent, which we found sufficient since all our trained agent's quickly learn to reach the goal in policy training once deployed in the real environment.

A.9 Translation Performance by Metric

In addition to the BLEU-4 [8] metric reported in the main text, we also evaluated the use of other metrics for computing the similarity between generated plan translations and the corresponding ground-truth description of the plan execution. Namely, we used the Word Error Rate (WER), Translation Edit Rate (TER) [9], Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [10],

Table 2: Somniloquy Hyperparameters. Differences with Dreamer are highlighted in red.

Name	Value
General	
Replay capacity	1×10^5
Batch size	16
Batch length	64
Activation	RMSNorm + SiLU
Learning rate	4×10^{-5}
Gradient clipping	1000
Optimiser	ADAM($\epsilon = 10^{-9}$)
World Model	
Reconstruction loss scale	1
Dynamics loss scale	1
Representation loss scale	0.1
Latent unimix	1%
Free nats	1
Actor & Critic	
Imagination horizon	15
Discount horizon	333
Return lambda	0.95
Critic EMA regulariser	1
Critic EMA decay	0.98
Actor entropy regulariser	3×10^{-4}
Actor unimix	1%
Actor RetNorm scale	$\text{Per}(R, 95) - \text{Per}(R, 5)$
Actor RetNorm decay	0.99
Translation Component	
Activation	ReLU
Dropout	0.1
Max latent sequence length	16
Token sampling method	Nucleus
Translation Loss Weighting	10.0

Table 3: Environment-specific hyperparameters

Env Name	Max Steps	Training Steps	Eval Every	Train Ratio	Vocab Size	Translation Max Length
AI2THOR	1K	∼400K	20K	64	109	200
Crafter	1K	∼230K	5K	512	97	150
MiniGrid	1K	100K	5K	512	36	200
BabyAI	1K	50K	5K	512	29	450

of which we present the ROUGE-L-F1 score, Metric for Evaluation of Translation with Explicit ORdering (METEOR) [11], and the CHaRacter n-gram F-score++ (chrf++) [12]. The results of these five metrics in the Crafter environment are given in Figure 4. As can be seen, there is very low variation across metrics, and all show the same as the BLEU-4 score results in the main text: in the Crafter environment, Somniloquy gives a consistently high translation performance throughout training, even whilst the agent's policy and the world model's latent representation is evolving.

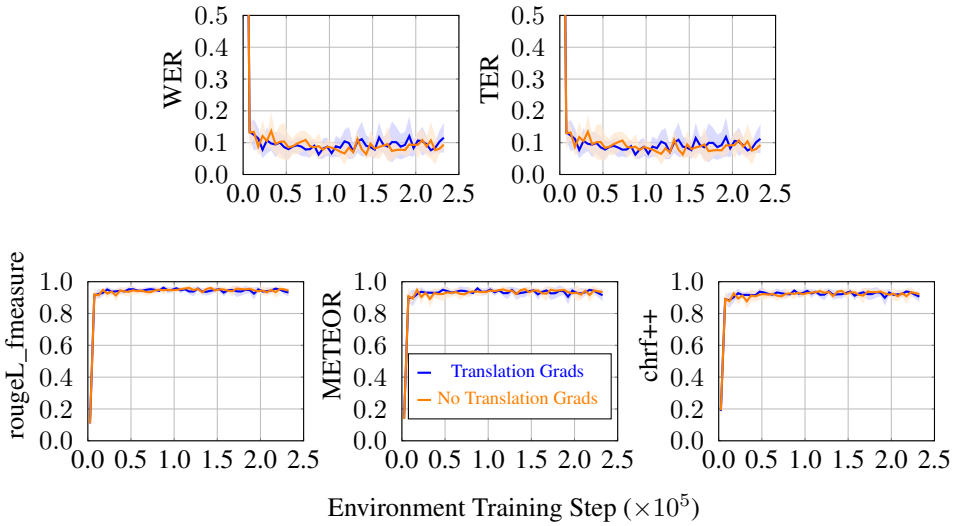


Figure 4: Crafter translation performance according to 5 different metrics. For WER and TER, the closest the score is to 0, the better the translation; for the rest, a value of 1 indicates a perfect translation. Each curve depicts the mean over 5 seeds, with 1 standard deviation shaded. All x-axes are the same and denote number of environment training steps.

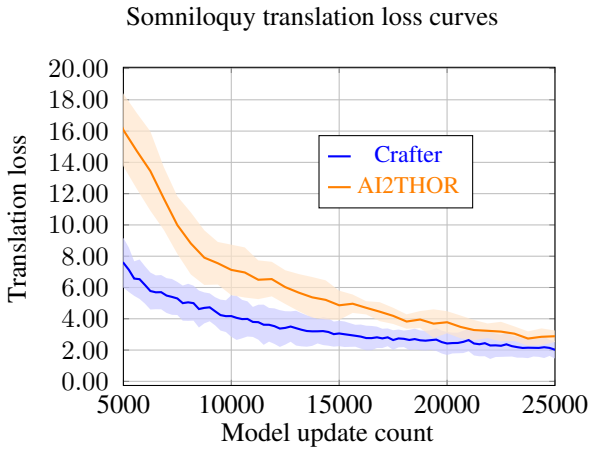


Figure 5: Somniloquy translation loss curves by number of model updates in the deterministic Crafter and AI2THOR environments. Curves depict the mean across 5 seeds, and shading shows 1 standard deviation.

A.10 Translation Loss Curves

As an estimate for the translation difficulty in the deterministic Crafter and AI2THOR environments, we present the translation loss curves during training below in Figure 5, for the default version of Somniloquy with translation loss gradients flowing into the latent representation. As the ratio of model update steps to environment training steps is different in each environment, we present the number of model update steps on the x-axis, as opposed to the number of environment steps, in order to align the translation loss across the two environments. Note that one model update is over a batch containing (batch_size $\times$ batch_length) transitions. As Figure 5 shows, Somniloquy requires a greater number of model update steps in the AI2THOR environment, relative to the Crafter environment, to achieve a small translation loss.

A.11 Example Latent Plan Translations

AI2THOR. We present the first 10 translations for an evaluation episode in Table 6, taken from the first evaluation episode of seed 100, after training for around 400K environment steps.

We present two detailed examples of latent plan translations, and the corresponding plan executions, in the AI2THOR environment, one correct and one incorrect. A correct plan translation can be found in Figure 6, along with some of the observations obtained from executing the plan. The agent correctly states where it will start in the scene, along with the set of objects that it will interact with. An incorrect plan translation can be found in Figure 7. In this case, the agent incorrectly states that it will first pickup an apple, when actually the first object it picks up is a pot. Interestingly, the decoded plan images also appear to show an apple, suggesting that the translation is a faithful representation of the latent states, but the world model itself is incorrect in predicting dynamics.

The correct example is taken from Somniloquy's training run with a seed of 100, from the first evaluation run after training for around 400K steps. The second example is taken from the same seed, and is the first evaluation run from earlier in training after training for around 200K steps.

Table 4: AI2THOR Example Translations. Taken from evaluation episode 1 of seed 100

Plan Timesteps	**Generated Translation**	**'True' Translation**	**BLEU Score**
0-16	"i will start near the stove and i wont move much first i will pickup a pan and then i will throw the pan and then i will pickup a dishsponge and then i will throw the dishsponge and then i will pickup a bowl and then i will throw the bowl and then i will pickup a potato"	"i will start near the stove and i wont move much first i will pickup a pan and then i will throw the pan and then i will pickup a dishsponge and then i will throw the dishsponge and then i will pickup a bowl and then i will throw the bowl and then i will pickup a potato"	1.0
16-32	"i will start near the stove and i wont move much first i will throw the potato and then i will pickup a plate and then i will throw the plate and then i will pickup a mug and then i will throw the mug and then i will pickup a soapbottle and then i will throw the soapbottle and then i will pickup a spoon and then i will throw the spoon and then i will pickup a fork and then i will throw the fork and then i will pickup a butterknife and then i will throw the butterknife"	"i will start near the stove and i wont move much first i will throw the potato and then i will pickup a plate and then i will throw the plate and then i will pickup a mug and then i will throw the mug and then i will pickup a soapbottle and then i will throw the soapbottle and then i will pickup a spoon and then i will throw the spoon and then i will pickup a fork and then i will throw the fork and then i will pickup a butterknife and then i will throw the butterknife"	1.0

Continued on next page

Plan Timesteps	Generated Translation	'True' Translation	BLEU Score
32-48	"i will start near the stove and i wont move much first i will throw the butterknife and then i will pickup a pot and then i will throw the pot and then i will pickup a cup and then i will throw the cup and then i will pickup a tomato and then i will throw the tomato and then i will pickup a peppershaker and then i will throw the peppershaker"	"i will start near the stove and i wont move much first i will throw the butterknife and then i will pickup a pot and then i will throw the pot and then i will pickup a cup and then i will throw the cup and then i will pickup a tomato and then i will throw the tomato and then i will pickup a peppershaker and then i will throw the peppershaker"	1.0
48-64	"i will start near the stove and i wont move much first i will throw the peppershaker and then i will pickup a spatula and then i will throw the spatula and then i will pickup a bread and then i will throw the bread and then i will pickup a apple and then i will throw the apple and then i will pickup a saltshaker"	"i will start near the stove and i wont move much first i will throw the peppershaker and then i will pickup a spatula and then i will throw the spatula and then i will pickup a bread and then i will throw the bread and then i will pickup a apple and then i will throw the apple and then i will pickup a saltshaker"	1.0
64-80	"i will start near the stove and i wont move much first i will throw the saltshaker and then i will pickup a papertowelroll and then i will throw the papertowelroll and then i will open the drawer and then i will pickup a knife and then i will throw the knife"	"i will start near the stove and i wont move much first i will throw the saltshaker and then i will pickup a papertowelroll and then i will throw the papertowelroll and then i will open the drawer and then i will pickup a knife and then i will throw the knife"	1.0
80-96	"i will start near the sink and i wont move much first i will pickup a creditcard and then i will throw the creditcard and then i will close the drawer"	"i will start near the sink and i wont move much first i will pickup a creditcard and then i will throw the creditcard and then i will close the drawer"	1.0
96-112	"i will start near the sink and then i will head towards the near window first i will pickup a statue and then i will throw the statue"	"i will start near the sink and i wont move much first i will pickup a statue and then i will throw the statue"	0.64
112-128	""i will start near the sink and then i will head towards the near window first i will pickup a statue and then i will throw the statue"	"i will start near the sink and then i will head towards the near window first i will pickup a statue and then i will throw the statue"	1.0

Continued on next page

Plan Timesteps	Generated Translation	'True' Translation	BLEU Score
128-144	"i will start near the stove and i wont move much i wont interact with any objects"	"i will start near the stove and i wont move much i wont interact with any objects"	1.0
144-160	"i will start near the stove and i wont move much i wont interact with any objects"	"i will start near the stove and i wont move much i wont interact with any objects"	1.0

***Plan Translation=Plan Execution Description**: "i will start near the stove and i wont move much first i will pickup a pan and then i will throw the pan and then i will pickup a dishsponge and then i will throw the dishsponge and then i will pickup a bowl and then i will throw the bowl and then i will pickup a potato"*

Figure 6: An example of a correct plan translation in the AI2THOR environment that exactly matches the description of the plan execution. Given the starting state (t=1), the agent uses the policy and world model to plan for 15 steps. The sequence of latent states is translated into the string shown in the green box. The agent then executes its plan in the environment, yielding the observations shown at the bottom of the image, which gives a description that exactly matches the verbalised plan.

Crafter. We present translations for a full evaluation episode in Table 6, taken from the first evaluation episode of seed 2000, after training for around 230K environment steps.

We present two detailed examples of latent plan translations, and the corresponding plan executions, in the Crafter environment, one correct and one incorrect. A correct plan translation can be found in Figure 8, along with some of the observations obtained from executing the plan. The agent correctly states all of the objects that it will see, states correctly that it will harvest wood and stone, and states correctly that it will drink some water. An incorrect plan translation can be found in Figure 9. In this case, the agent incorrectly states that it will get killed by a zombie. As with the incorrect example in the AI2THOR environment, this plan translation does seem somewhat consistent with what the actual latent plan may represent, since the world model does predict that the plan reaches a terminal state.

Both examples are taken from Somniloquy's training run with a seed of 2000, and from the final evaluation epoch after training for around 230K environment steps.

Plan Translation: *"i will start near the sink and i wont move much first i will pickup a apple and then i will throw the apple and then i will pickup a apple and then i will throw the apple and then i will pickup a bread and then i will throw the bread and then i will pickup a apple and then i will throw the apple"*

Plan Execution Description: *"i will start near the sink and i wont move much first i will throw the bread and then i will pickup a pot and then i will throw the pot and then i will pickup a peppershaker and then i will throw the peppershaker and then i will pickup a apple and then i will throw the apple"*

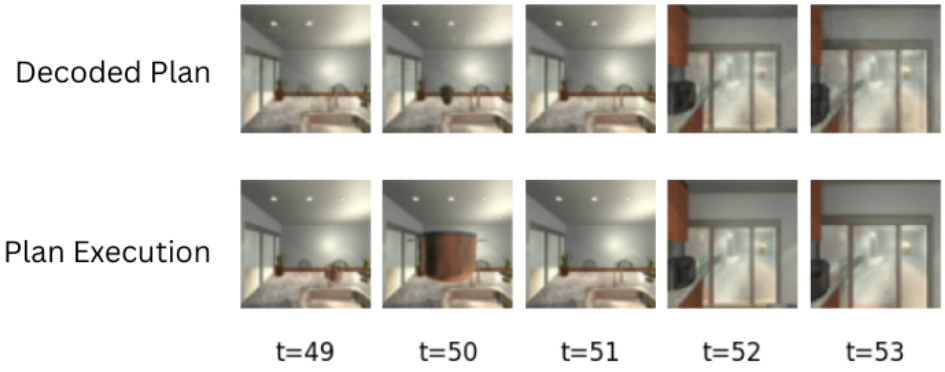


Figure 7: An example of a plan translation in the AI2THOR environment that does not match the description of the plan execution. Given the starting state (t=49), the agent uses the policy and world model to plan for 15 steps. The sequence of latent states is translated into the string shown in the red box. The agent then executes its plan in the environment, yielding the description shown in the green box. Observations from the plan execution can be seen in the bottom row of images, and decoded observations from the latent plan in the top row, showing that the agent incorrectly imagines that it will pick up an apple rather than a pot.

Table 5: Crafter Example Translations. Taken from evaluation episode 1 of seed 2000. Here, the world model incorrectly predicts that the episode terminates during the final plan.

Plan Timesteps	**Generated Translation**	**'True' Translation**	**BLEU Score**
0-16	"i will see cow grass sand tree and water i will harvest 1 sapling 1 wood i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	"i will see cow grass sand tree and water i will harvest 1 sapling 1 wood i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	1.0

Continued on next page

Plan Timesteps	Generated Translation	'True' Translation	BLEU Score
16-32	"ii will see cow grass plant sand table tree and water i will not harvest anything i will not craft anything i will eat 1 cows i will place 1 tables i will place 1 plants i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	"i will see cow grass plant sand table tree and water i will not harvest anything i will not craft anything i will eat 1 cows i will place 1 tables i will place 1 plants i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	1.0
32-48	"i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will drink 1 waters and refill some thirst i will not sleep and get more tired"	"i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will drink 1 waters and refill some thirst i will not sleep and get more tired"	1.0
48-64	"i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will sleep 1 times and refill some energy"	"i will see cow grass sand table tree and water i will harvest 1 wood i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will sleep 1 times and refill some energy"	0.91
64-80	"i will see grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will drink 2 waters and refill some thirst i will not sleep and get more tired"	"i will see coal cow grass sand stone table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will drink 1 waters and refill some thirst i will not sleep and get more tired"	0.82
80-96	"i will see coal cow grass sand stone table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will drink 2 waters and refill some thirst i will not sleep and get more tired"	"i will see coal cow grass sand stone table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	0.83

Continued on next page

Plan Timesteps	Generated Translation	'True' Translation	BLEU Score
96-112	"i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	"i will see coal cow grass sand stone table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	0.89
112-128	"i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	"i will see coal cow grass sand stone table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	0.89
128-144	"i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	"i will see cow grass sand table tree water and zombie i will harvest 1 wood i will not craft anything i will be attacked and hit by a zombie my health will regenerate 1 times i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	0.59
144-160	"i will see cow grass path sand water and zombie i will not harvest anything i will not craft anything i will be attacked and hit by a zombie i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	"i will see cow grass sand table water and zombie i will not harvest anything i will not craft anything i will be attacked and hit by a zombie i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"	0.94
160-169	"i will see cow grass sand skeleton table water and zombie i will not harvest anything i will not craft anything i will be attacked and hit by a zombie my hitpoints will reach zero and i will die"	"i will see cow grass sand table water and zombie i will not harvest anything i will not craft anything i will be attacked and hit by a zombie my hitpoints will reach zero and i will die"	0.93

Stochastic MiniGrid. We present one example of a latent plan translation, and the corresponding plan execution, in our stochastic MiniGrid environment. Due to the stochastic nature of the environment, the description of the plan execution is not expected to match the latent plan translation. As can be see in Figure 10, the latent plan translation, along with the decoded plan observations, has the

***Plan Translation=Plan Execution Description**: "i will see cow grass sand tree and water i will harvest 1 sapling 1 wood i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"*

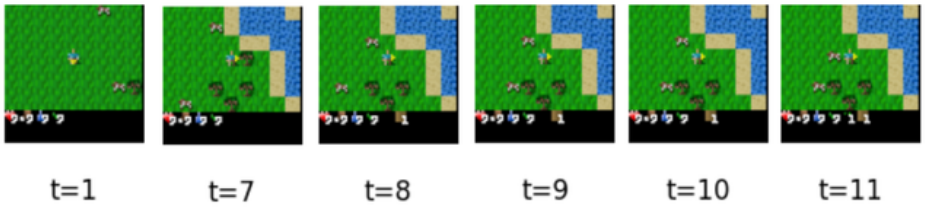


Figure 8: An example of a correct plan translation in the Crafter environment that exactly matches the description of the plan execution. Given the starting state (t=1), the agent uses the policy and world model to plan for 15 steps. The sequence of latent states is translated into the string shown in the green box. The agent then executes its plan in the environment, yielding the observations shown at the bottom of the image, which gives a description that exactly matches the verbalised plan.

***Plan Translation**: "i will see cow grass sand table tree and water i will not harvest anything i will not craft anything i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"*

***Plan Execution Description**: "i will see cow grass sand table tree water and zombie i will harvest 1 wood i will not craft anything i will be attacked and hit by a zombie my health will regenerate 1 times i will not eat anything and get more hungry i will not drink anything and get more thirsty i will not sleep and get more tired"*

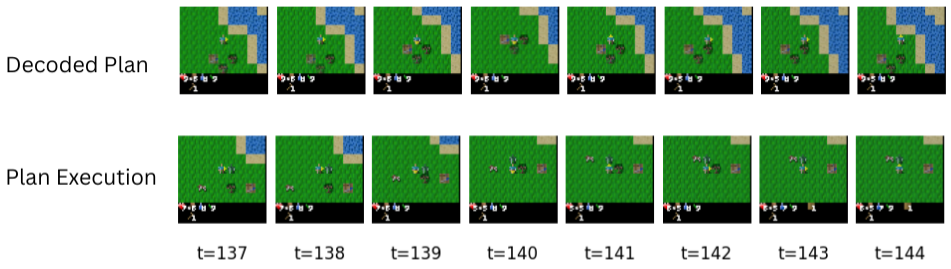


Figure 9: An example of a plan translation in the Crafter environment that does not match the description of the plan execution. Given the starting state (t=129), the agent uses the policy and world model to plan for 15 steps. The sequence of latent states is translated into the string shown in the red box. The agent then executes its plan in the environment, yielding the description shown in the green box. Observations from the plan execution can be seen in the bottom row of images, and decoded observations from the latent plan in the top row. In this case, the world model incorrectly predicts what the agent will observe, misses that the agent harvests wood, and fails to predict the agent encountering, and getting attacked by, a zombie.

agent going through the green teleporter and teleporting to the purple teleporter room. When the plan is executed, the stochastic nature of the environment means that the agent goes through the green teleporter and is teleported to the left goal corridor instead. Result are taken from the first evaluation episode of seed 100, after training for around 100K steps.

***Plan Translation**: "i start in the blue teleporter room and i go through the blue teleporter to the green teleporter room and then i go through the green teleporter to the purple teleporter room and then i go through the right purple teleporter to the middle goal corridor and then i will move towards the goal"*

***Plan Execution Description**: "i start in the blue teleporter room and i go through the blue teleporter to the green teleporter room and then i go through the green teleporter to the left goal corridor and then i will move towards the goal"*

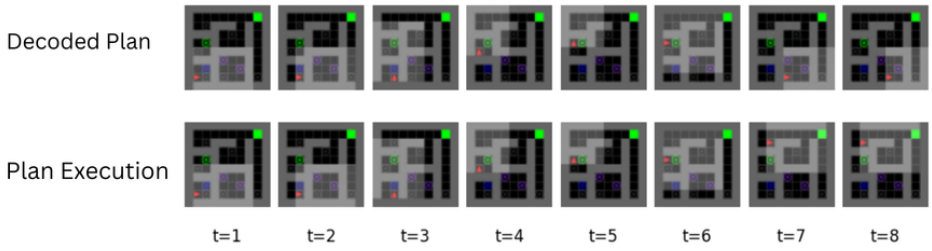


Figure 10: An example of a plan translation in the MiniGrid environment. Given the starting state (t=1), the agent uses the policy and world model to plan for 15 steps. The sequence of latent states is translated into the string shown in the red box. The agent then executes its plan in the environment, yielding the description shown in the green box. Observations from the plan execution can be seen in the bottom row of images, and decoded observations from the latent plan in the top row. Due to the stochasticity in the environment, the agent imagines that going through the green teleporter will take it to the purple teleporter room (at steps 6 and 7), but when the plan execution goes through the green teleporter, the agent is taken to the left goal corridor, causing the plan and plan execution to diverge.

References

[1] Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

[2] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[3] Danijar Hafner. Benchmarking the spectrum of agent capabilities. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=1W0z96MFEoH.

[4] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*, 2017.

[5] Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in Neural Information Processing Systems 36, New Orleans, LA, USA*, December 2023.

[6] Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. BabyAI: First steps towards grounded language learning with a human in the loop. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJeXCo0cYX.

[7] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, Apr 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-08744-2. URL https://doi.org/10.1038/s41586-025-08744-2.

[8] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[9] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, 2006.

[10] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[11] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[12] Maja Popović. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618, 2017.