Exploration-Driven Generative Interactive Environments

Supplementary Material

Contents

A. Experimental Protocol

A.1. Training Protocol of GenieRedux and GenieRedux-G.

The architecture and training parameters of the Tokenizer and the Dynamics module of GenieRedux-G are shown respectively in the Tab. 1 and Tab. 2. GenieRedux shares those choices, with the addition of LAM defined as in Tab. 3. For the purpose of the case study, we use 7 latent actions. Training parameters can be seen in Tab. 4.

We train the Tokenizer on 8 A100 GPUs for 72k iterations, with batch size 112 and patch size 4, on a dataset of all 483 environments (50 sessions per environment obtained with a random agent).

Component	Parameter	Value
Encoder	num_blocks	8
	d_model	512
	num_heads	8
Decoder	num_block	8
	d_model	512
	num_heads	8
Codebook	num_codes	1024
	latent_dim	32

Table 1. Tokenizer hyperparameters

Component	Parameter	Value
Architecture	num_blocks	12
	d_model	512
	num_heads	8
Sampling	temperature	1.0
	maskgit_steps	25

Table 2. Dynamics hyperparameters

Component	Parameter	Value
Encoder	num_blocks	8
	d_model	512
	num_heads	8
Decoder	num_blocks	8
	d_model	512
	num_heads	8
Codebook	num_codes	7
	latent_dim	32

Table 3. LAM hyperparameters

Parameter	Value
max_lr	1×10^{-4}
min_lr	5×10^{-5}
β_1	0.9
β_2	0.99
weight_decay	1×10^{-4}
linear_warmup_start_factor	0.5
warmup_steps	5000

Table 4. Optimizer Hyperparameters

Our Dynamics module is trained with sequences of 16 frames, processed by the pretrained tokenizer. Dynamics module is trained with batch size 80 on 8 A100 GPUs for 185k iterations on `Platformers-200` (GenieRedux-G-200), and fine-tuned for 80k iterations on

`Platformers-50` (GenieRedux-G-50), batch size 160.

For an agent (random or AutoExplore Agent), we obtain a dataset of 10k sequences of length 800. We fine-tune GenieRedux-G-50 on a set for 10k iterations to obtain GenieRedux-G-50-ft, with batch size 160.

We always use the Adam optimizer with a linear warm-up and cosine annealing strategy.

We note that GenieRedux has $\sim$ 350M total parameters, broken down as follows: Tokenizer (100M), LAM (170M), and Dynamics (80M). Meanwhile, GenieRedux-G has $\sim$180M total parameters: Tokenizer (100M) and Dynamics (80M).

A.2. Testing Protocol of GenieRedux and GenieRedux-G.

For our test set, we train Agent57 per environment, using the available environment reward. In order to have many diverse episodes in our datasets and all the actions to be represented, we mix, using an ϵ-greedy approach, the agent's actions with random actions. We collect 1000 episodes as the test set (with episode length 700) and evaluate on sequences of size 12 with step size 20 in two settings. While our model can handle a single frame as input, for a fair evaluation, we choose to provide two, as a single frame does not provide motion information and there are multiple valid solutions (see Sect. F.7). We provide two frames and predict the next 10, given all actions. In the usual case, we perform MaskGIT inference with 25 iterations for all 10 images at once. We obtain much fewer artifacts and higher level of control if we adapt an autoregressive approach - iteratively generating 2 frames at a time given all previous tokens in the sequence, each with 25 iterations. However, as this is computationally heavy, we provide autoregressive results for our best models only in our evaluations.

A.3. Training Protocol of AutoExplore Agent

For each of the environments, we train for 300 epochs with the following schedule for each epoch: 1. Run the current agent for 200 steps in 8 running environments in parallel to collect data in the replay buffer. Actions are sampled with temperature 1.0, with an epsilon-greedy algorithm with ϵ starting from 0.1 and linearly decaying to 0.01 over the course of 150 epochs. 2. Train the agent for 200 steps, sampling from the buffer, with batch size 128. Actor-critic loss is used with entropy regularization over the actions to prevent greedy behavior. In the end, we choose the agent with the highest evaluation return throughout training.

B. Super-resolution Network

We upscale the outputs of GenieRedux-G from 64x64 to 256x256 by a U-Net based super-resolution network, with MSE loss for both training and evaluation. The training data consists of 256x256 images that we captured from the original environments. Three configurations were tested: (1) a small U-Net with feature channel dimensions [64,128,256,512] and approximately 31 million parameters, trained on 16,000 images (12,000 for training and 4,000 for testing), achieving a test loss of 0.0081; (2) the same small U-Net trained on a larger dataset of 50,000 images (45,000 for training and 5,000 for testing), achieving a test loss of 0.0047; and (3) a larger U-Net with feature channel dimensions [128,256,512,1024] and 124 million parameters, trained on the same 50,000-image dataset, achieving a test loss of 0.0029. All models were trained with a batch size of 128, a learning rate of 0.0001, and a step-based scheduler (step_size=25, gamma=0.5) for 300 epochs, using Adam optimizer.

C. Multi-Environment Models Additional Experiments

C.1. Qualitative Results of GenieRedux-G-50

In Fig. 1 we show examples of 10-frame predictions from GenieRedux-G-50. They are sampled from the test set and the actions that resulted in the ground truth sequence were given to the model to produce the predictions. As seen, the model was able to produce outcomes from the actions that are close to the ground truth. In Fig. 2 is shown the developments of 3 actions over time for GenieRedux-G-50, showing a smooth trajectory and the action being executed.

In our experiments, we test the ability of our models to simulate multiple environments in virtual environments already observed by the models. For new unseen environments, our models show limited generalization abilities, characterized by pausing motion and visual artifacts. We believe that generalizability can be improved by training our tokenizer on a larger video dataset, however, with care taken to preserve the learned background token strategy learned, as it brings important properties to our exploration reward and the Dynamics module.

C.2. Autoregressive Evaluation

For one of the environments - `SuperMarioBros`, we provide comparison of all our models using autoregressive evaluation. This evaluation is more computationally heavy, so we originally compare them with a single-pass evaluation, and evaluate autoregressively only the fine-tuned models on both strategies - with a random agent and AutoExplore Agent. As for small characters and uniform backgrounds, single-pass evaluation appears to produce close results between all models in terms of controllability, we choose to perform full autoregressive evaluation on `SuperMarioBros` (where these conditions are present) to show the benefit of our approach. Results are shown in Tab. 5. The newly autoregressively evaluated models are GenieRedux-G-50 and GenieRedux. It can be concluded

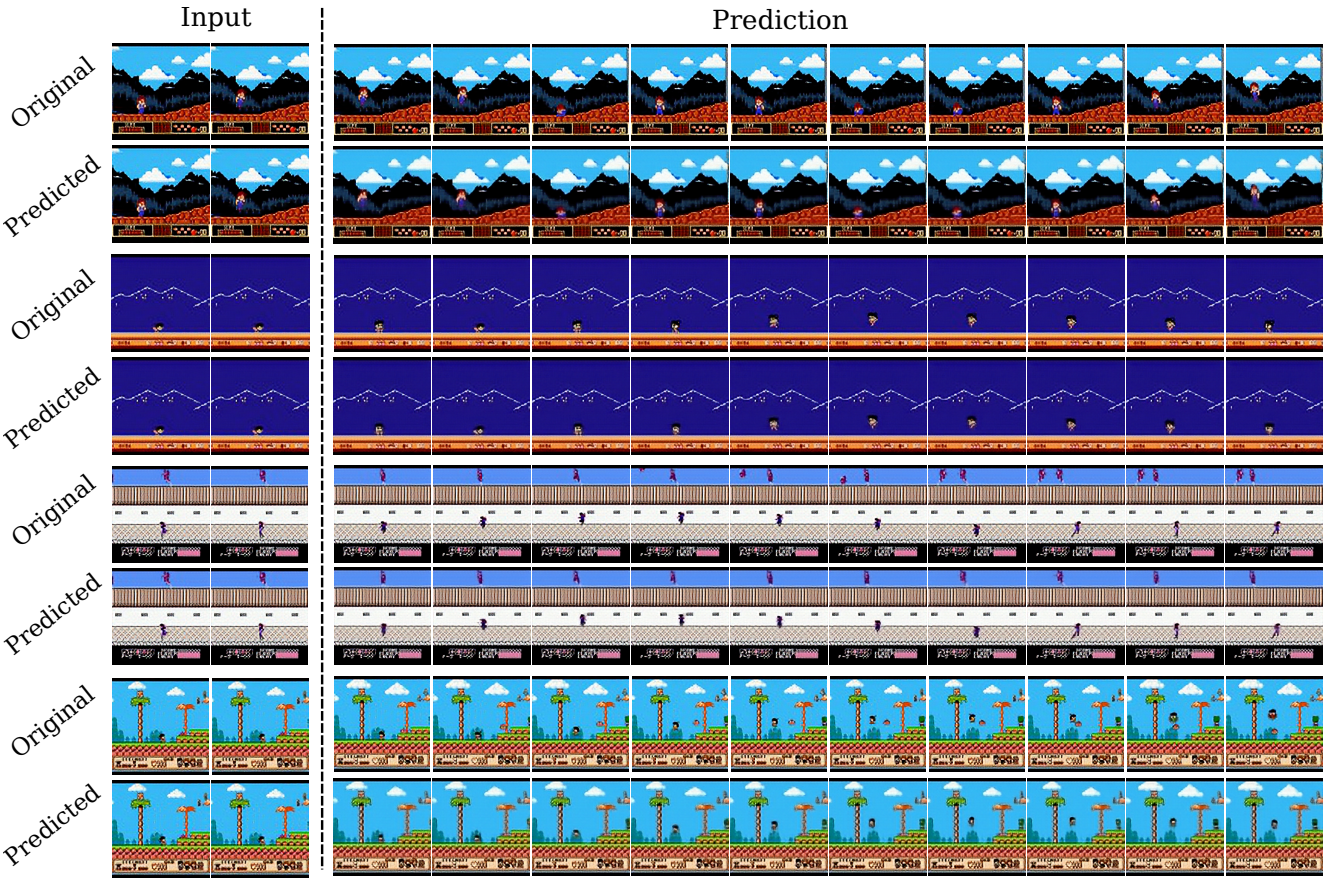


Figure 1. **Qualitative Results of GenieRedux-G-50.** Given 2 frames, we show the predictions of 10 future frames. Ground truth is shown for comparison.

Table 5. **`SuperMarioBros` Autoregressive Quantitative Evaluation.**

Environment	Strategy	Model	FID↓	PSNR↑	SSIM↑	ΔPSNR↑
Super Mario Bros.	Random Autoregressive	GenieRedux-G-50	30.48	34.59	0.94	0.55
		GenieRedux-G-50-ft	30.84	34.85	0.95	0.57
	Exploration Autoregressive	Tokenizer-ft	8.08	42.00	0.99	-
		GenieRedux-G	9.46	34.38	0.95	0.07
		GenieRedux-G-50-ft	**9.33**	**37.77**	**0.97**	**0.76**

Table 6. **Quantitative results of GenieRedux-G-50, fine-tuned on AutoExplore Agent's data of all three environments together.** Results show that our method can be used to improve the multi-environment performance of the model. GenieRedux-G-50 is our pretrained world model on 50 environments. GenieRedux-G-50-ft are fine-tuned models using data from a random agent or AutoExplore (Exploration). GenieRedux-G denotes a non-fine-tuned model, trained with the exploration data.

Environment	Strategy	Model	FID↓	PSNR↑	SSIM↑	ΔPSNR↑
Combined Environments	Random	GenieRedux-G-50	43.57	27.55	0.82	0.65
		GenieRedux-G-50-ft	43.98	27.74	0.82	0.78
	Exploration	Tokenizer-ft	14.02	37.98	0.98	-
		GenieRedux-G	14.88	28.91	0.88	0.25
		GenieRedux-G-50-ft	**14.61**	**31.29**	**0.91**	**1.09**
	Random Autoregressive	GenieRedux-G-50-ft	43.69	28.19	0.83	0.79
	Exploration Autoregressive	GenieRedux-G-50-ft	**14.49**	**33.14**	**0.93**	**1.46**

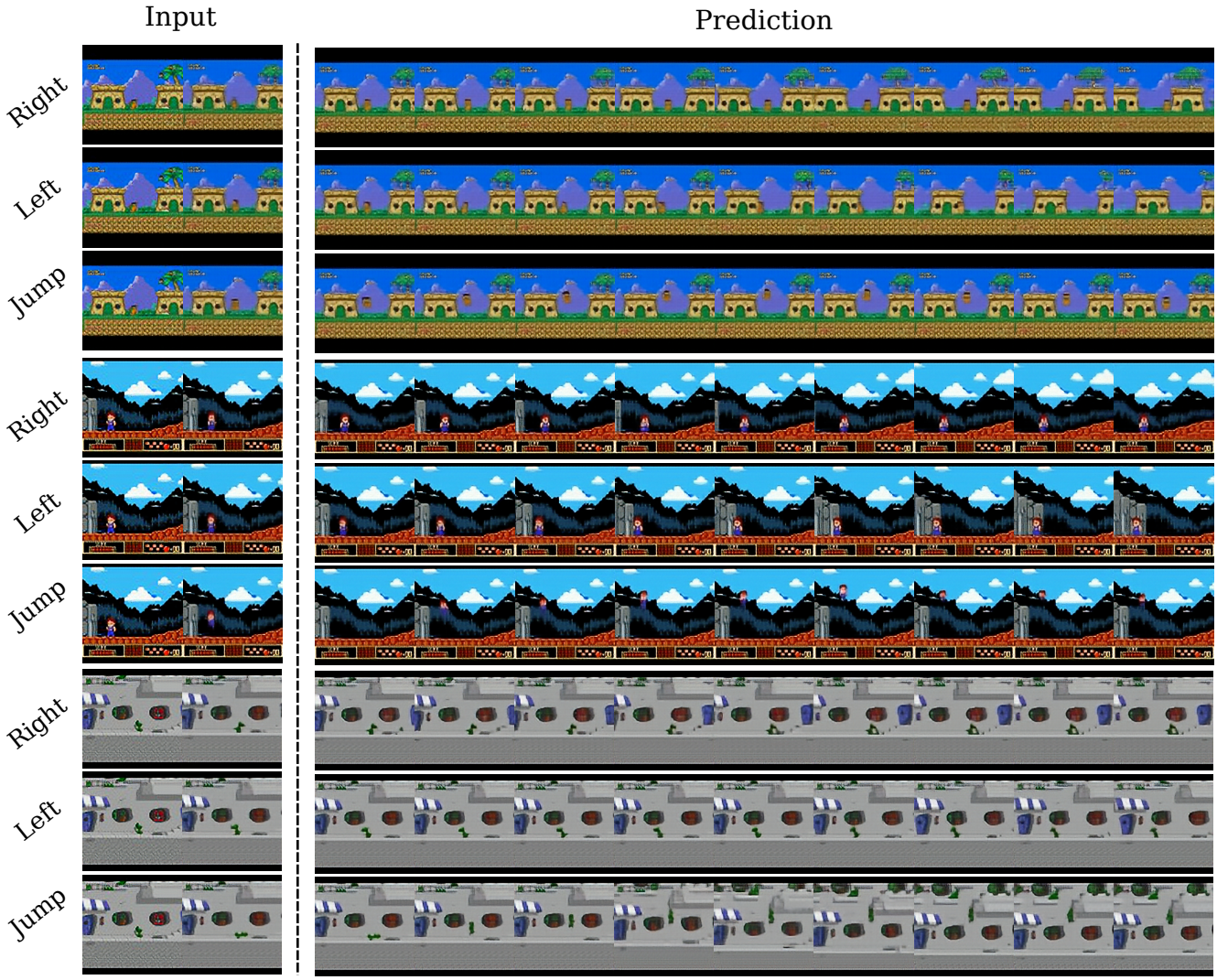


Figure 2. **Controllability of GenieRedux-G-50.** We show a detailed per frame prediction for right, left and jump actions, showing the development of the actions.

that, with our exploration approach, we obtain significantly better results in terms of visual fidelity and controllability.

C.3. Multi-Environment Fine-tuning

In this experiment, we take the diverse datasets, collected from the three environments we have studied - `AdventureIslandII`, `SuperMarioBros` and `Smurfs`, and fine-tune GenieRedux-G-50 on all of them together. In this way, we evaluate the effect of our method on multi-environment training. Results are shown in Tab. 6. Using AutoExplore Agent's data, the model has improved its visual fidelity and controllability across the test set, containing all three environments (equal number of samples each). This shows that our method is applicable for improving multi-environment training as well.

C.4. Qualitative Evaluation of Fine-tuned Models

In Fig. 3 are shown examples per environment of predictions from a model, fine-tuned on a random agent versus a model fine-tuned on AutoExplore Agent's data. The model, trained on AutoExplore Agent's data, exhibits much higher visual quality and less artifacts. We also note that the tokenizer plays a role in improving visual quality. After exploration, the tokenizer is able to fit better to new visuals of the environment, which reduces visual artifacts.

In Fig. 4 we show AutoExplore Agent's data helping to achieve better controllability compared to the random agent. As typical for controllability evaluation, we give a single frame as input. We observe that in cases where the motion is ambiguous (e.g. where a character might be going up or down), fine-tuning with exploration data leads to more con-

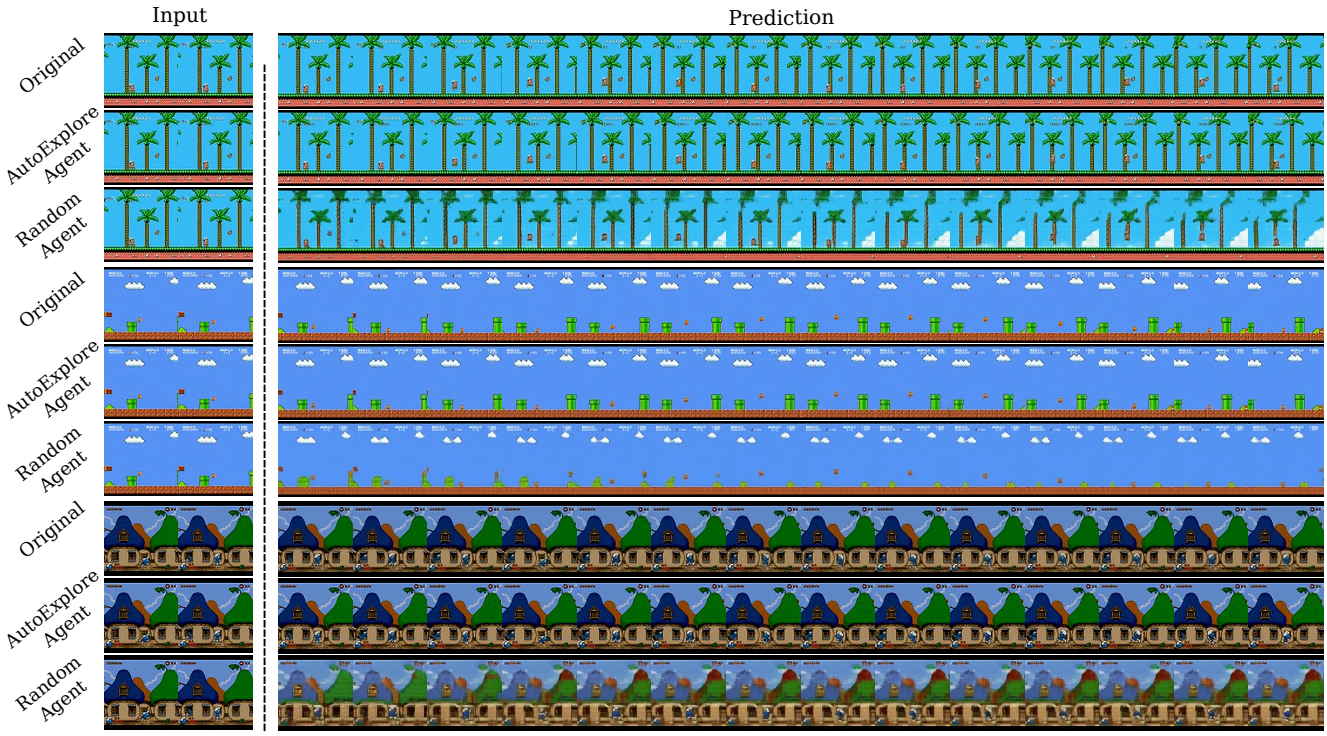


Figure 3. **Qualitative Results of GenieRedux-G-50-ft for random and AutoExplore agents.** They are compared to the ground truth (Original). It can be seen that AutoExplore Agent's data produces a model with significantly higher visual quality and less artifacts.

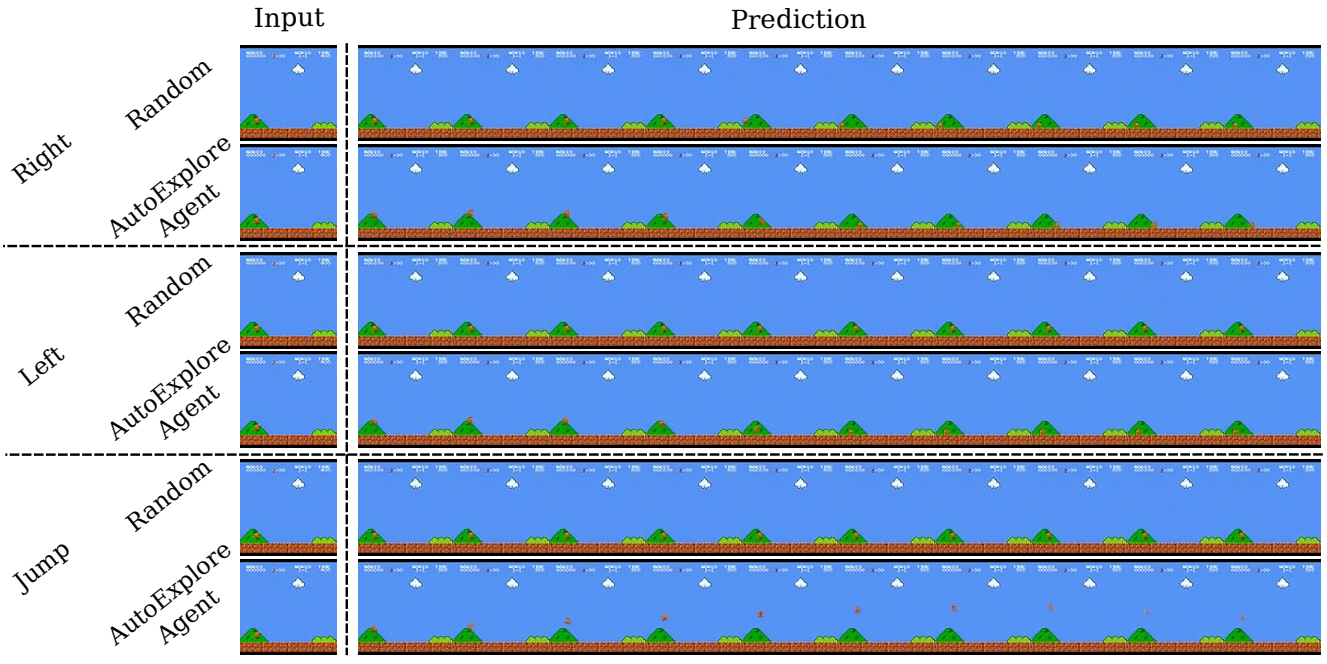


Figure 4. **Controllability of GenieRedux-G-50-ft for random and AutoExplore agents on `SuperMarioBros`.** In ambiguous cases like this where the agent can be going up or down, exploration data shows to improve performance.

fidence and hence realistic sequence generation. In contrast, models trained on random data cannot resolve the situation and copy the frame multiple times.

D. AutoExplore Agent Behavior Study

The agent was trained with the five actions that the world model was trained with. While initially in training the agent

Figure 5. **AutoExplore Agent Behavior.** We show the behavior of our AutoExplore Agent on the three environments studied. It can be seen that the agent learned to progress by moving right, jumping over obstacles and dealing with enemies.

learns simpler strategies like jumping, eventually it achieves better returns by learning to move forwards in an environment (and reveal new scenes). To progress even further, the agent learns to overcome obstacles and enemies. In Fig. 5, we show the behavior of the agent on the three environments used after training. The agent is observed to move forward in the environment, to overcome enemies, jump over obstacles. Interestingly, the strategy in Smurfs was to sometimes wait to be attacked by an enemy, which caused the player to disappear and the camera to move before spawning. This seems to cause an increase in world model uncertainty in that environment. In other cases (flying enemies), the agent tries to avoid. In Smurfs, there is an action of entering a door. We observe that sometimes the agent enters a door that causes the character to reappear from a different side on the screen.

E. User Studies

E.1. General Quality User Study Details

We provide extra details about our user study to evaluate the models fine-tuned on data from a random agent and from AutoExplore Agent. In Fig. 6 we show the interface for a single sample given to the user. A clip is shown with two parts that the user should compare and rate. The order of the samples is random. The instructions given to the users at the start of the study are provided below.

Thank you for participating in our study! You will watch a total of 120 video samples. Each sample consists of two clips:
***Top** clip: Reference*
***Bottom** clip: Comparison clip*
Please compare the two clips in each sample and rate how closely they match in terms of visual quality and content.
Use the scale provided:
***1** : The two clips completely differ in terms of visual quality and/or content*
***5** : The two clips closely match in terms of visual quality and content*

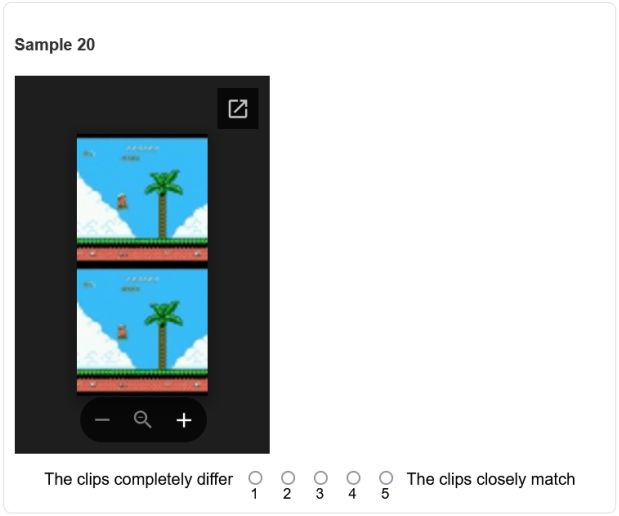


Figure 6. **General User Study Sample.**

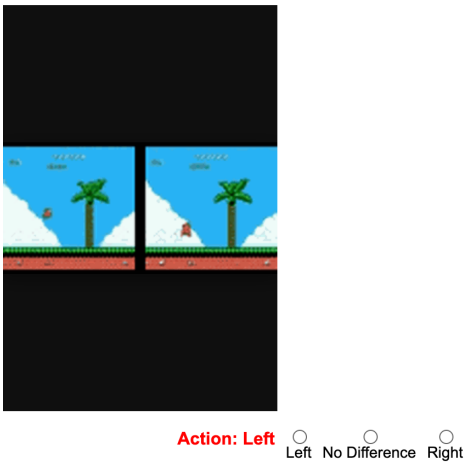


Figure 7. **Action Quality User Study Sample.**

Submit your rating for each sample through this form. Your feedback is important and greatly appreciated!

E.2. Action Quality User Study Details

We conduct a second user study to specifically evaluate the gains in action quality of our model fine-tuned on data from the AutoExplore agent over the baseline. Observing that our model is particularly beneficial in scenarios with ambiguous initial frames, we use this user study to test this. We use single initial frames with the agent in mid-jump. Participants interact with an interface shown in Fig.7. The user is shown pairs of synchronized videos, generated by the baseline and the exploration model (left/right position is randomized). Both videos depicted the same action — *left*, *jump*, or *right*—which was explicitly labeled in *bold red* below them. Participants were instructed to assess the qual-

Table 7. **Visual Fidelity** of TA models.

Model	Basic Test Set		
	FID↓	PSNR↑	SSIM↑
Tokenizer-TA	12.10	39.53	0.97
LAM-TA	47.73	28.24	0.85
GenieRedux-TA	13.26	25.47	0.82
GenieRedux-G-TA	**13.01**	**32.09**	**0.94**

ity of the action performed, disregarding any differences related purely to visual quality, and select one of the following options:

- ***Left:*** *The left clip depicts the action more accurately.*
- ***No Difference:*** *Both clips depict the action equally well.*
- ***Right:*** *The right clip depicts the action more accurately.*

F. GenieRedux Evaluation on CoinRun Case Study

In this section, we qualitatively evaluate our Genie implementation - GenieRedux and its variant GenieRedux-G on the CoinRun case study. We also quantitatively and qualitatively study the effect of using data from a trained agent in the Coinrun environment (GenieRedux and GenieRedux-G). We study the behavior and limitations of the model and compare our implementation with a concurrent one.

F.1. CoinRun Case Study Details

We train the Tokenizer and the Dynamics module on CoinRun environment datasets, one obtained from a random agent, and one obtained from a trained agent using environment reward.

For training the agent for exploration, we enable velocity maps on CoinRun. These maps also need to be enabled for the agent during data collection. When evaluating models trained on different datasets (random agent vs. trained agent), to be fair, we exclude the velocity map regions by setting their pixels to black on all sets.

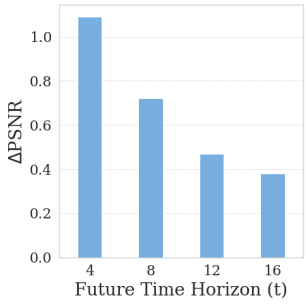
Throughout the training, we use a batch size of 84 and a patch size of 4 for all components. We use the Adam Optimizer with a linear warm-up and cosine annealing strategy.

We refer to the test set obtained from a random agent as Basic Test Set and to the one obtained from a trained agent as Diverse Test Set.

F.2. Comparison of GenieRedux with Jafar

We compare with Jafar Willi et al. [1] - a concurrent with our implementation of Genie (in JAX). We obtain and train their model as instructed. We train GenieRedux with Jafar's model parameters and like them separate LAM from Dynamics in training. The latter significantly worsened

Figure 8. **GenieRedux-G-TA Controllability Across Horizons.**



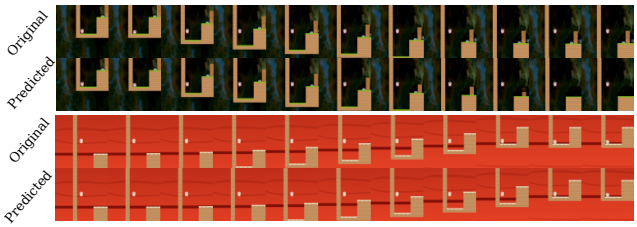


Figure 9. **GenieRedux Quantitative Evaluation.** We present a few sequences from the test set with predictions from GenieRedux. On the example at the top we show a successful jump action. On the example at the bottom we show a successful motion progression.

GenieRedux's action representation. Despite that, GenieRedux shows significantly better visual fidelity metrics, achieving 17.91 PSNR (46.12 FID), compared to Jafar's 12.66 PSNR (154.12 FID). GenieRedux does not exhibit Jafar's artifacts or the reported problematic "hole digging" behavior (more in App. F.6).

F.3. Prediction Horizon Evaluations

We evaluate the controllability of our best model (at 50k iterations) over varying prediction horizons in Fig. 8. As expected, predictions become more challenging further into the future. The first prediction is also difficult due to insufficient motion information - we obtain 0.4 Δ_tPSNR for $t = 1$. To address this issue, we provide the model with 4 frames and actions (predicting 10), and observe an improvement of our best model (GenieRedux-G-TA) from 34.79 PSNR (12.75 FID) in our results in the main paper to **38.31** PSNR (12.29 FID) on Diverse Test Set.

F.4. GenieRedux-G Qualitative Evaluation

In Fig. 9 we show quantitative results demonstrating that GenieRedux-G can perform motion progression and action execution.

F.5. GenieRedux-TA Qualitative Evaluation

In Fig. 10 we demonstrate that GenieRedux-TA is able to execute actions and complete motion. In Fig. 11 we show

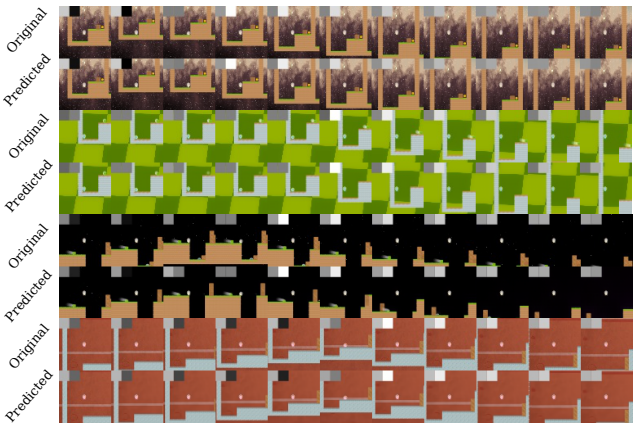


Figure 10. **GenieRedux-TA Qualitative Comparison.** We present a few samples from the test set with various actions. We demonstrate that GenieRedux-TA performs the actions correctly.

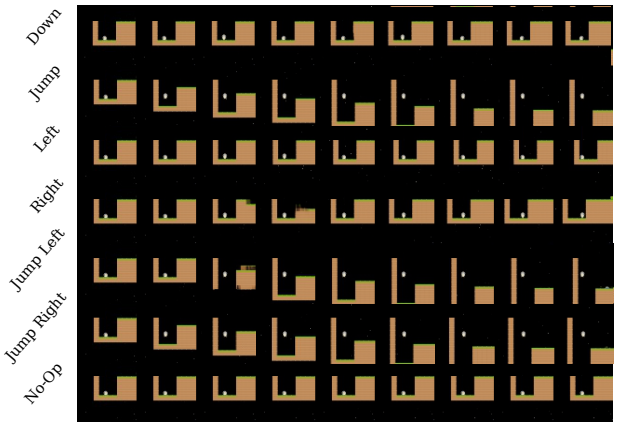


Figure 11. **GenieRedux-TA Controllability.** We show predictions for all environment actions of GenieRedux-TA.

that the model is capable of executing all actions of the environment.

F.6. Jafar Qualitative Comparison

We compare with Jafar Willi et al. [1] - a concurrent with ours implementation of Genie (in JAX). We obtain and train their model as instructed. We train GenieRedux with Jafar's model parameters and like them separate LAM from Dynamics in training. The latter significantly worsened GenieRedux's action representation. Despite that, GenieRedux shows significantly better visual fidelity metrics, achieving 17.91 PSNR (46.12 FID), compared to Jafar's 12.66 PSNR (154.12 FID). GenieRedux does not exhibit Jafar's artifacts or the reported problematic "hole digging" behavior. Moreover, we observe that Jafar lacks causality, which we find problematic.

In Fig. 12 we show Jafar's reconstruction of 10 frames into the future, given the first frame and a sequence of actions. The results are on the validation set after training. We observe an abundance of artifacts. We note that if we provide the images instead of providing the first frame, we get much less artifacts. This seems to hint that Jafar relies on future images to make predictions for the current frame, which might be an inherent problem of the model not being causal.

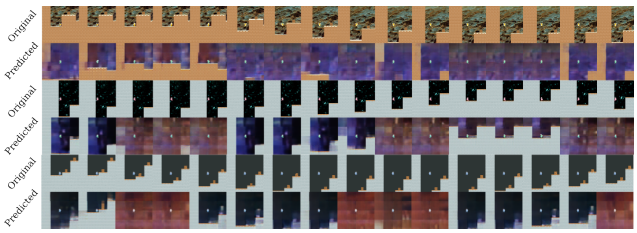


Figure 12. **Jafar Qualitative Results.** The results are on the validation set. We give only a single image and actions and predict 15 frames in the future.

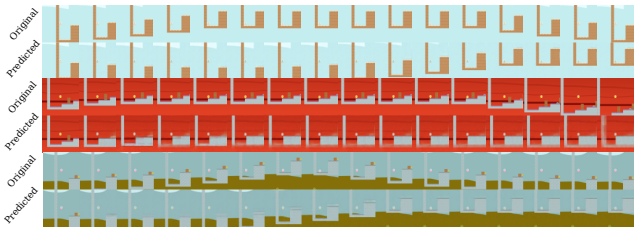


Figure 13. **GenieRedux with Jafar's Parameters Qualitative Results.** We show 15 frames into the future given actions and an initial frame of our model.

We additionally report test set results for Jafar - 0.48 SSIM and for GenieRedux (with Jafar parameters) - 0.62 SSIM.

In addition, we show the version of GenieRedux that we trained to match Jafar in Fig. 13. While it can be noticed that the model prefers inaction when encountering actions, it successfully progresses motion - e.g. moving a character through the air. We also notice fairly good visual quality.

F.7. Additional GenieRedux-G-TA Qualitative Results and Limitations

We provide additional visuals of our best performing GenieRedux-G-TA in Fig. 14 and Fig. 15. We see that our model performs well under different actions and scenarios.

Next, we discuss the limitations of GenieRedux-G-TA and visualize the known cases in Fig. 16. One possible failure case occurs whenever the environment state or the actions suggest that a major exploration of the environment will unfold - for example, when falling down from mid-jump. As the agent is only given a single frame and cannot possibly know the layout of the level, it attempts to reconstruct something that is not guaranteed to be the actual

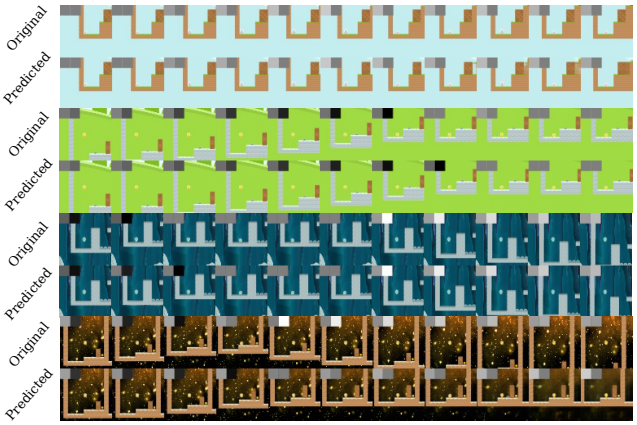


Figure 14. **GenieRedux-G-TA Extra Qualitative Results.** More sampled sequences from the test set, showing good match with the ground truth when enacting actions.

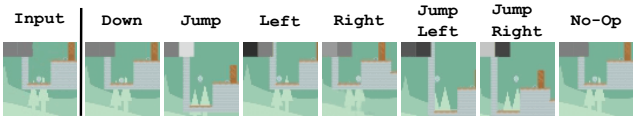


Figure 15. **GenieRedux-G-TA Controllability Demonstration.** We show that GenieRedux-G is able to perform all Coinrun environment actions.

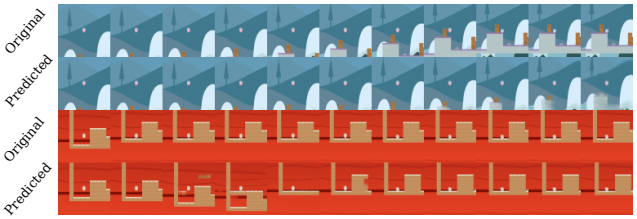


Figure 16. **GenieRedux-G-TA Limitations.** Two failure cases of GenieRedux-G-TA - whenever a sizeable new unknown part of the environment is revealed; whenever an in-progress motion is ambiguous.

level. Often, the agent exhibits uncertainty in these cases, as shown in the results.

Another possible weakness occurs whenever on the first frame a motion is already in progress - for example, in progress of jumping. In that case the model observes a single frame with the agent in the air and has no information about which direction the agent is heading - going up or going down. In that case, the model could exhibit uncertainty in the form of artifacts suggesting that the agent is both landing and jumping up, or alternatively not perform an action at all. This is a state from which the agent often recovers in a few steps. Still, we find that it can be avoided by providing more input frames to the model that can give motion information.

References

[1] Timon Willi, Matthew Thomas Jackson, and Jakob Nicolaus Foerster. Jafar: An open-source genie reimplemention in jax. In *First Workshop on Controllable Video Generation @ ICML 2024*, 2024. 7, 8