Sequence Generative Modeling II

Prof. Vincent Sitzmann

Task: Sequence Generation

x_1 x_2 x_3 x_4

• • •

Task: Sequence Generation

x_1 x_2 x_3 x_4

• • •

Model the **joint distribution of all frames**

$$p(x_1, x_2, x_3, \dots)$$

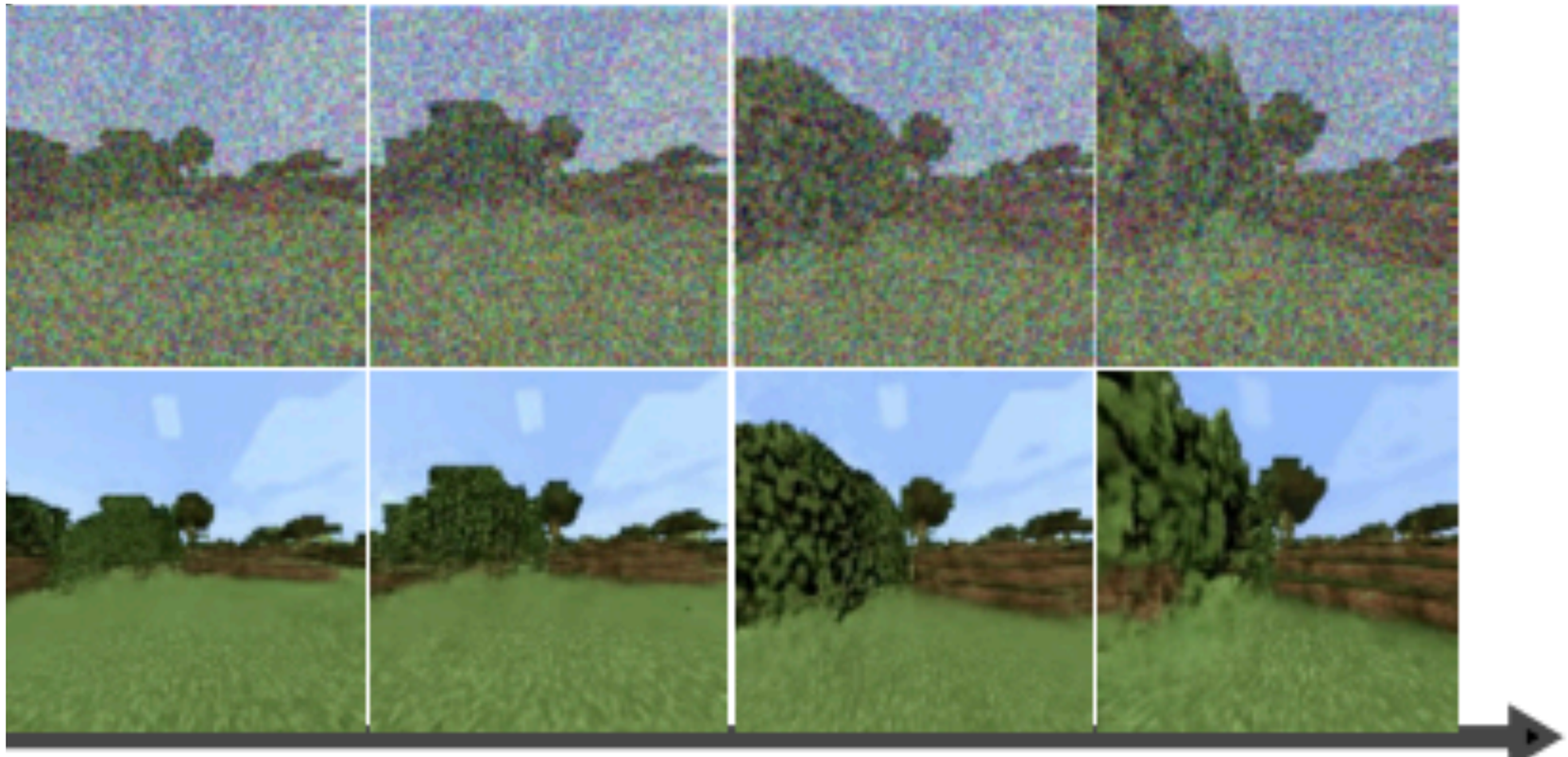
Preliminaries: Noise as Masking

Unmasked

Preliminaries: Noise as Masking

Slightly Masked

Unmasked

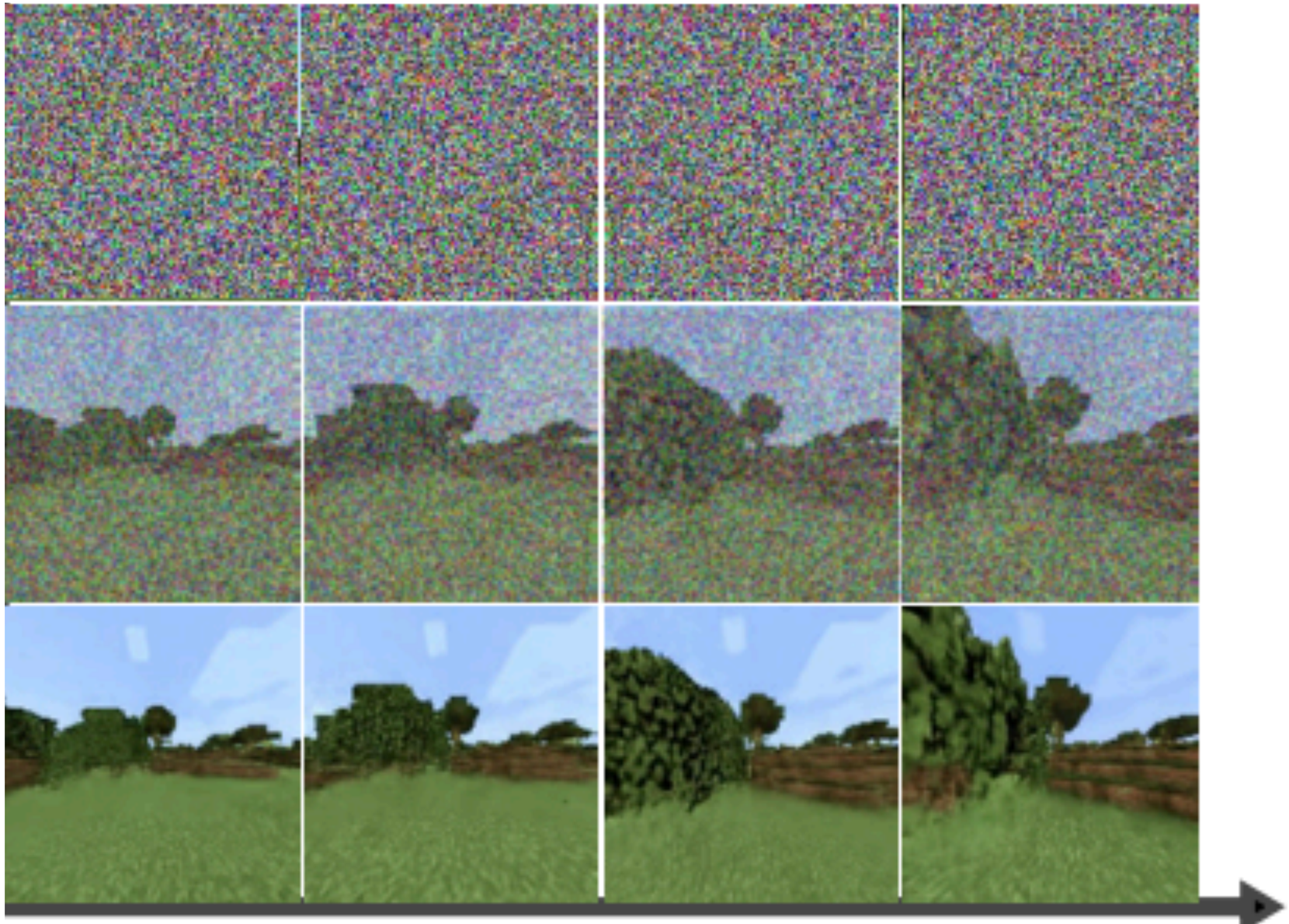


Preliminaries: Noise as Masking

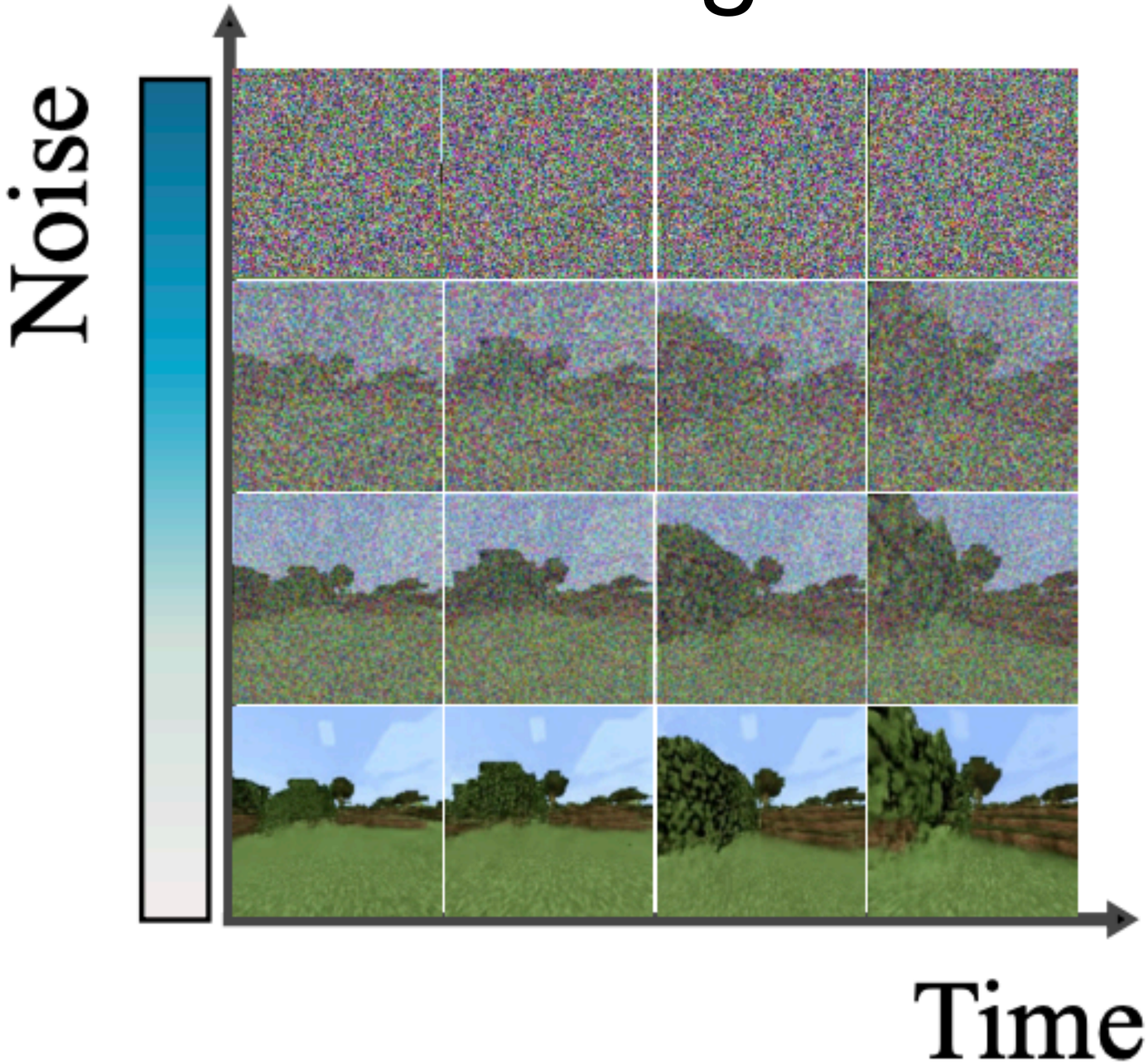
Fully Masked

Slightly Masked

Unmasked

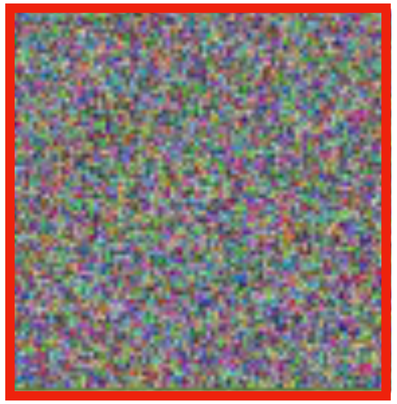


Preliminaries: Noise as Masking



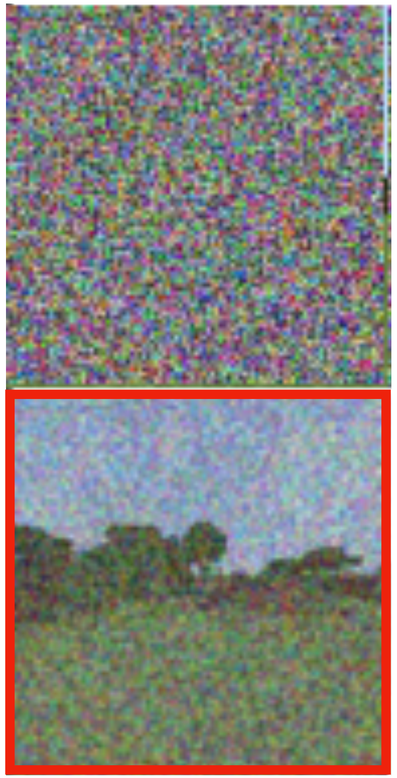
Two axes of
sequence generation

Auto-Regressive Generation (Next-Token Prediction)



Sampling from $p(x_1)$:
Denoising x_1

Auto-Regressive Generation (Next-Token Prediction)



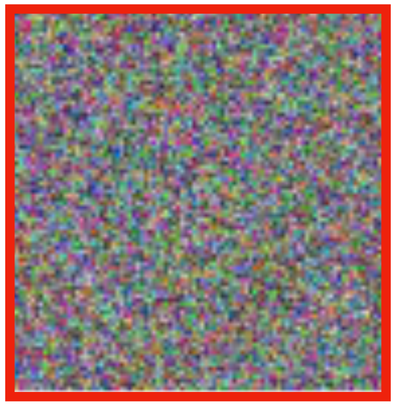
Sampling from $p(x_1)$:
Denoising x_1

Auto-Regressive Generation (Next-Token Prediction)

Sampling from $p(x_1)$:
Denoising x_1

Just image diffusion.

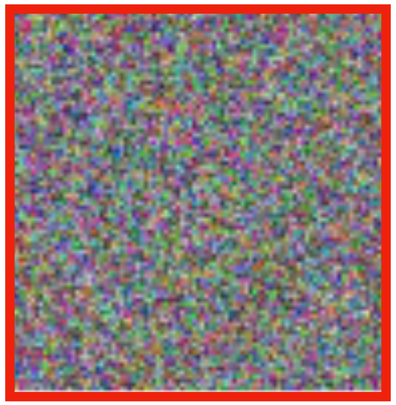
Auto-Regressive Generation (Next-Token Prediction)



Modeling $p(x_2 \mid x_1)$:
Denoising x_2
conditional on x_1

Context

Auto-Regressive Generation (Next-Token Prediction)

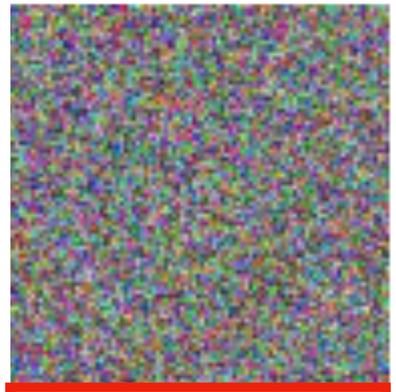


Modeling $p(x_2 \mid x_1)$:
Denoising x_2
conditional on x_1

Context

Like *conditional* image diffusion:
Give NN *both* the noisy next frame
and the previous frame

Auto-Regressive Generation (Next-Token Prediction)



Modeling $p(x_2 \mid x_1)$:
Denoising x_2
conditional on x_1

Context

Like *conditional* image diffusion:
Give NN *both* the noisy next frame
and the previous frame

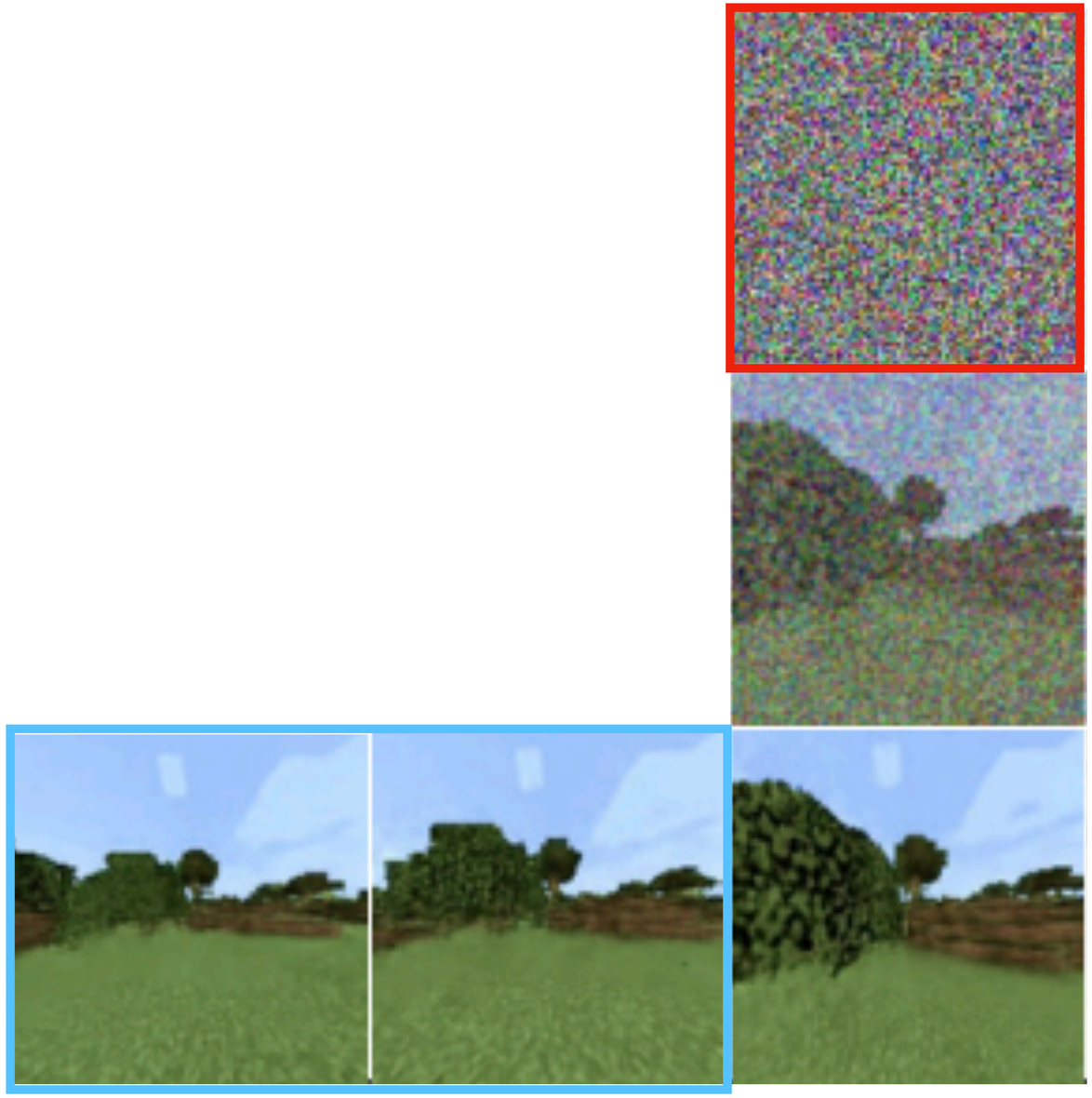
Auto-Regressive Generation (Next-Token Prediction)

Modeling $p(x_2 \mid x_1)$:
Denoising x_2
conditional on x_1

Context

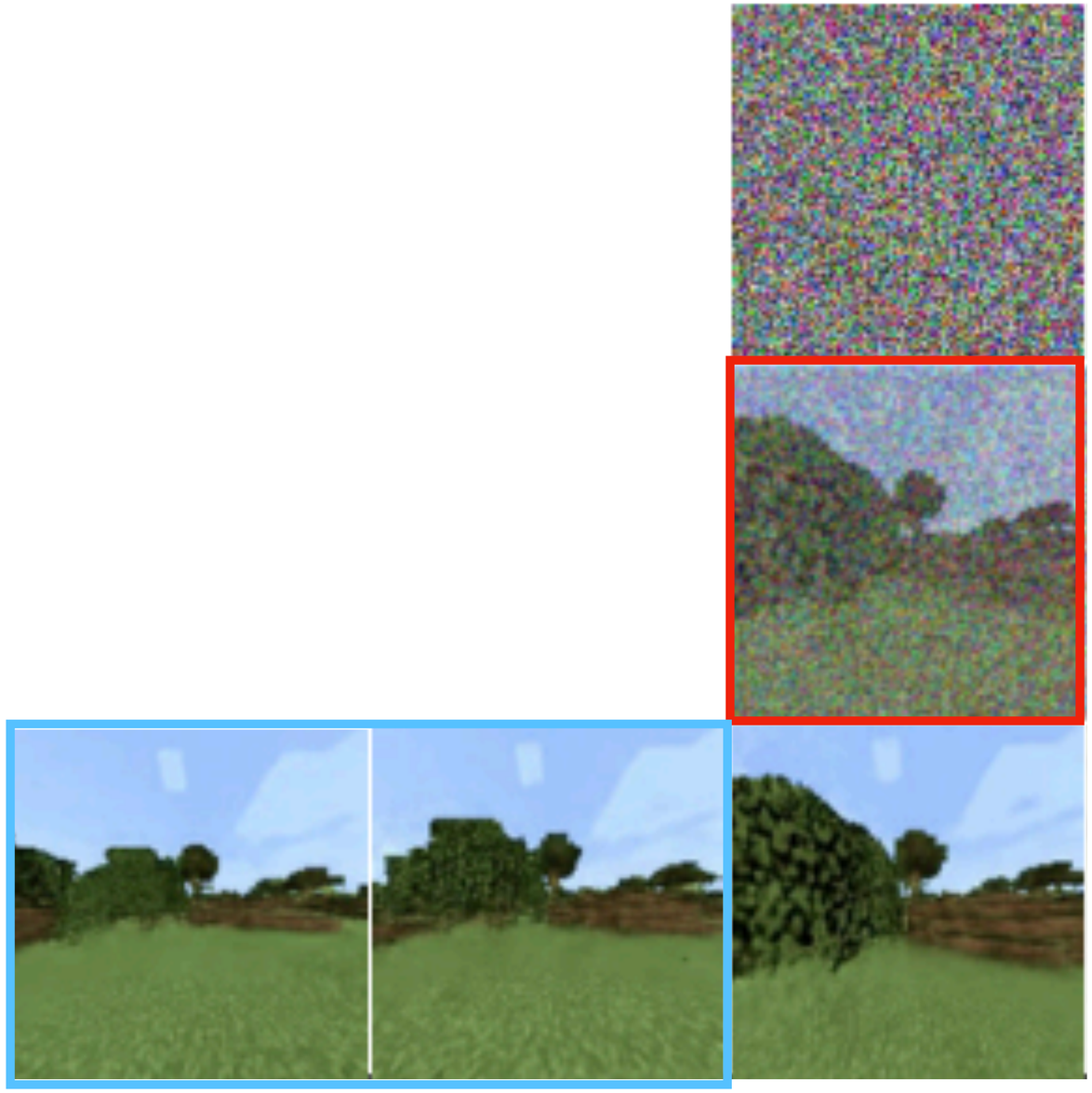
Like *conditional* image diffusion:
Give NN *both* the noisy next frame
and the previous frame

Auto-Regressive Generation (Next-Token Prediction)



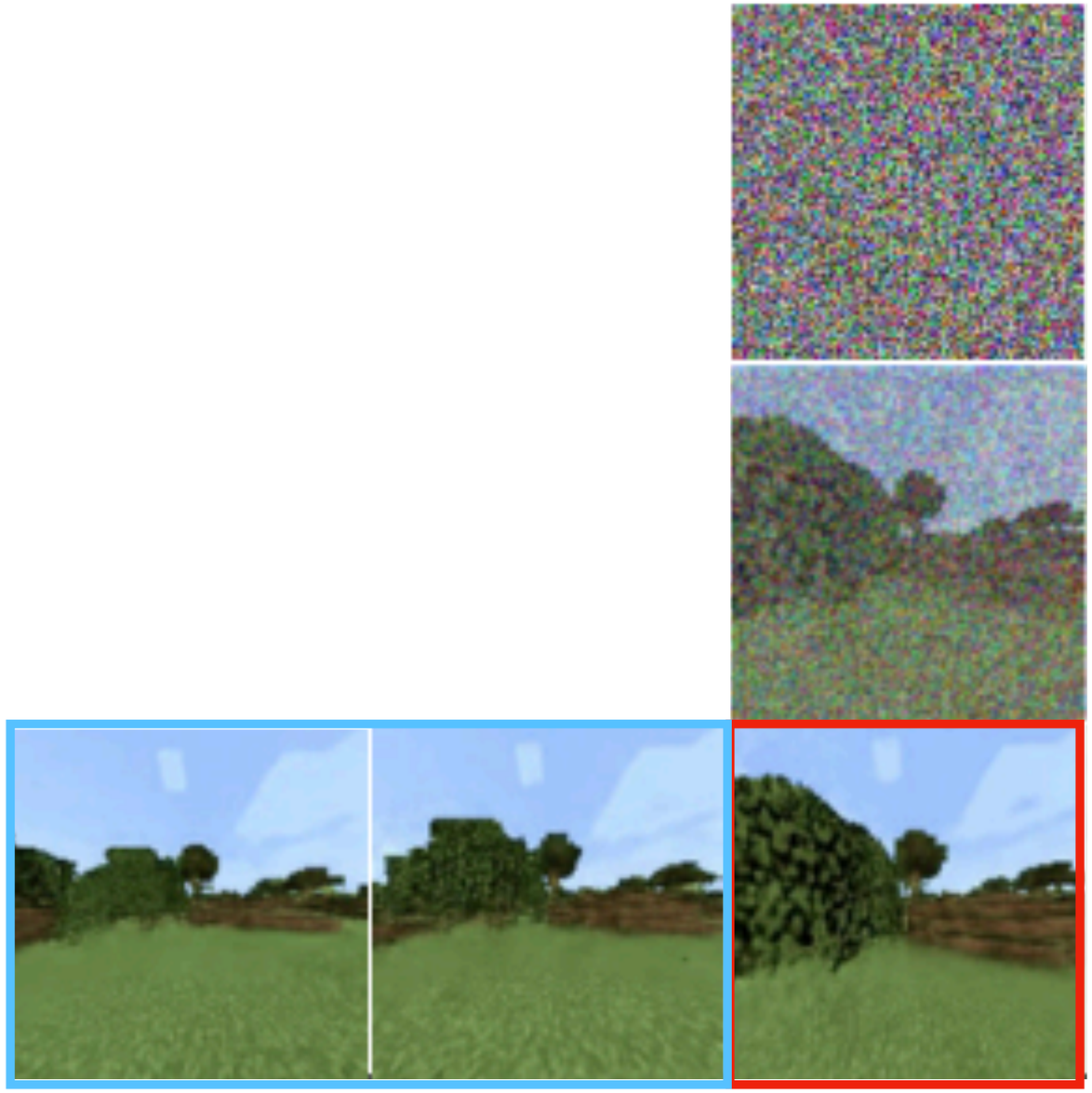
Context

Auto-Regressive Generation (Next-Token Prediction)



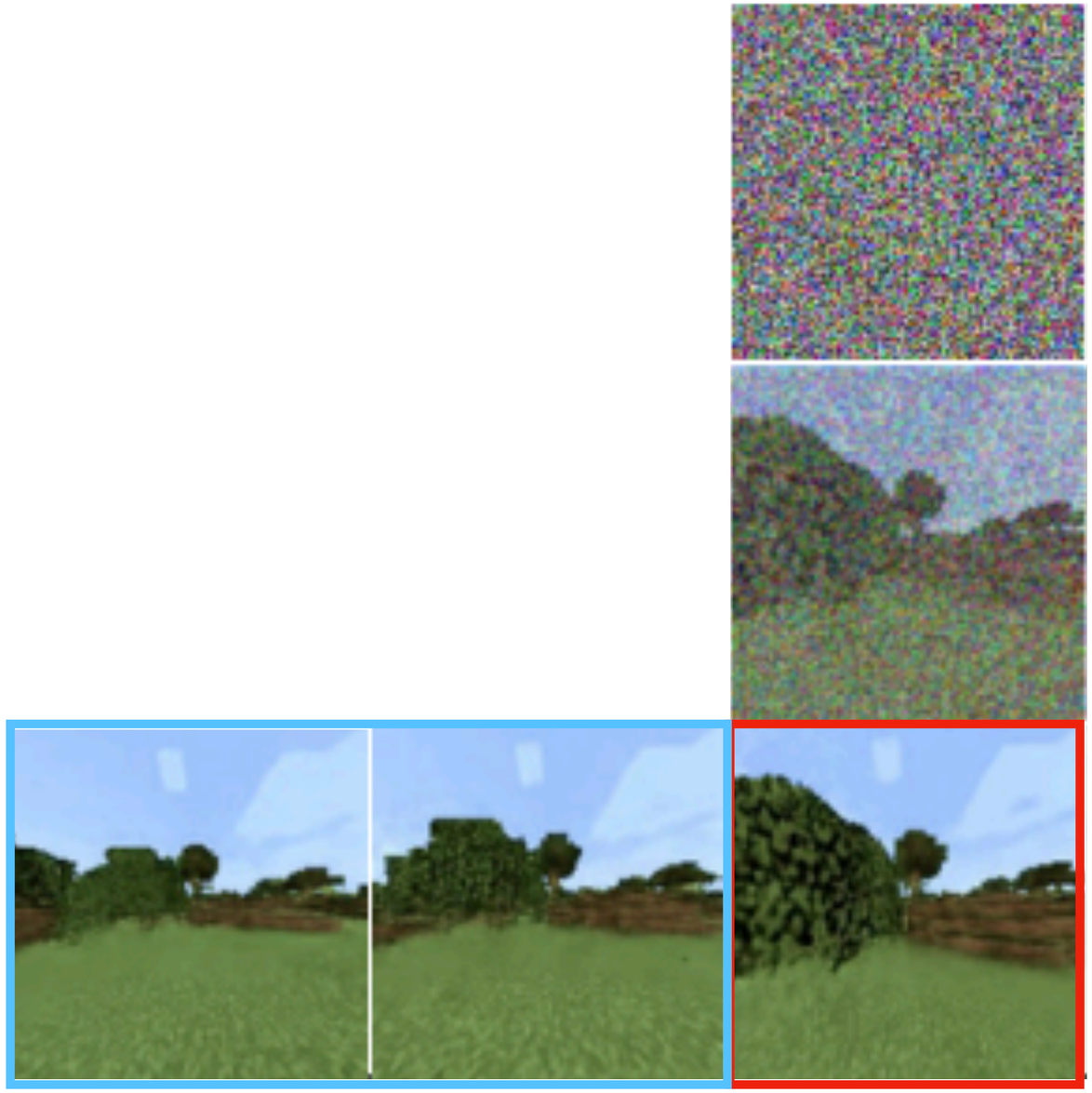
Context

Auto-Regressive Generation (Next-Token Prediction)



Context

Auto-Regressive Generation (Next-Token Prediction)

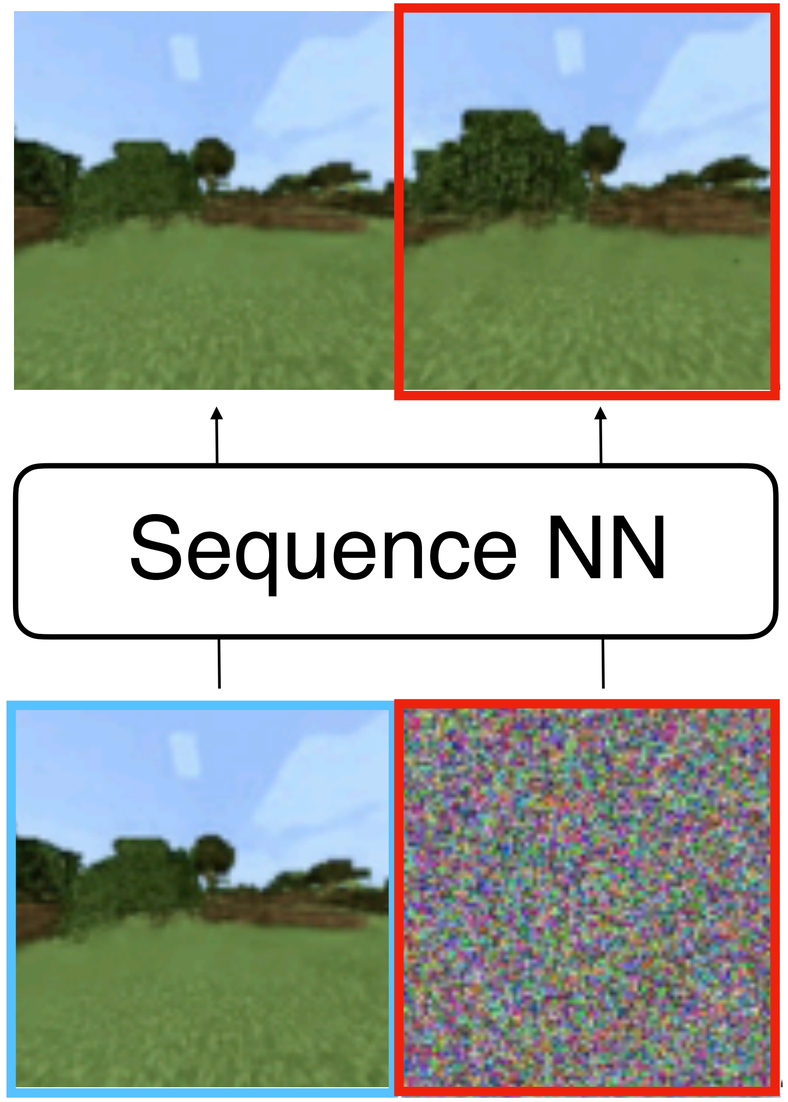


Context

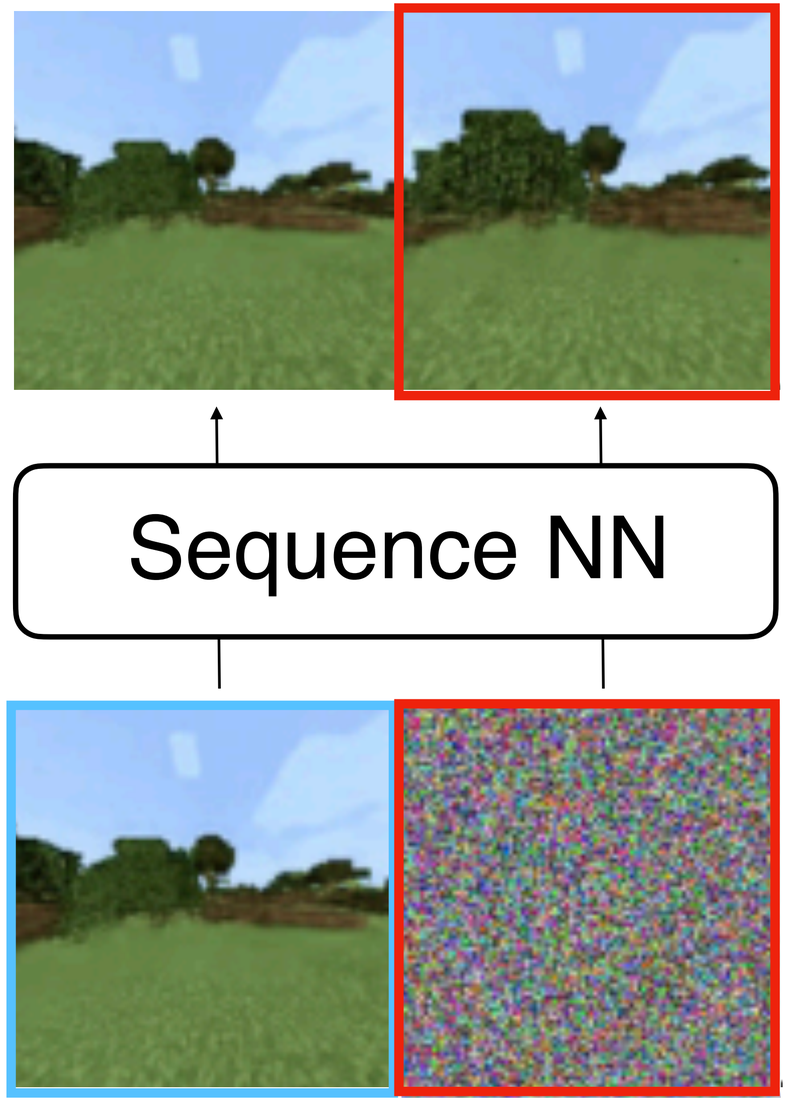
Generally: Auto-Regressive factorization of joint

$$p(x_1, x_2, x_3, \dots) = p(x_1) \prod_{i=2}^{N} p(x_i \mid x_{<i})$$

Teacher Forcing for Training Next-Token Pred.

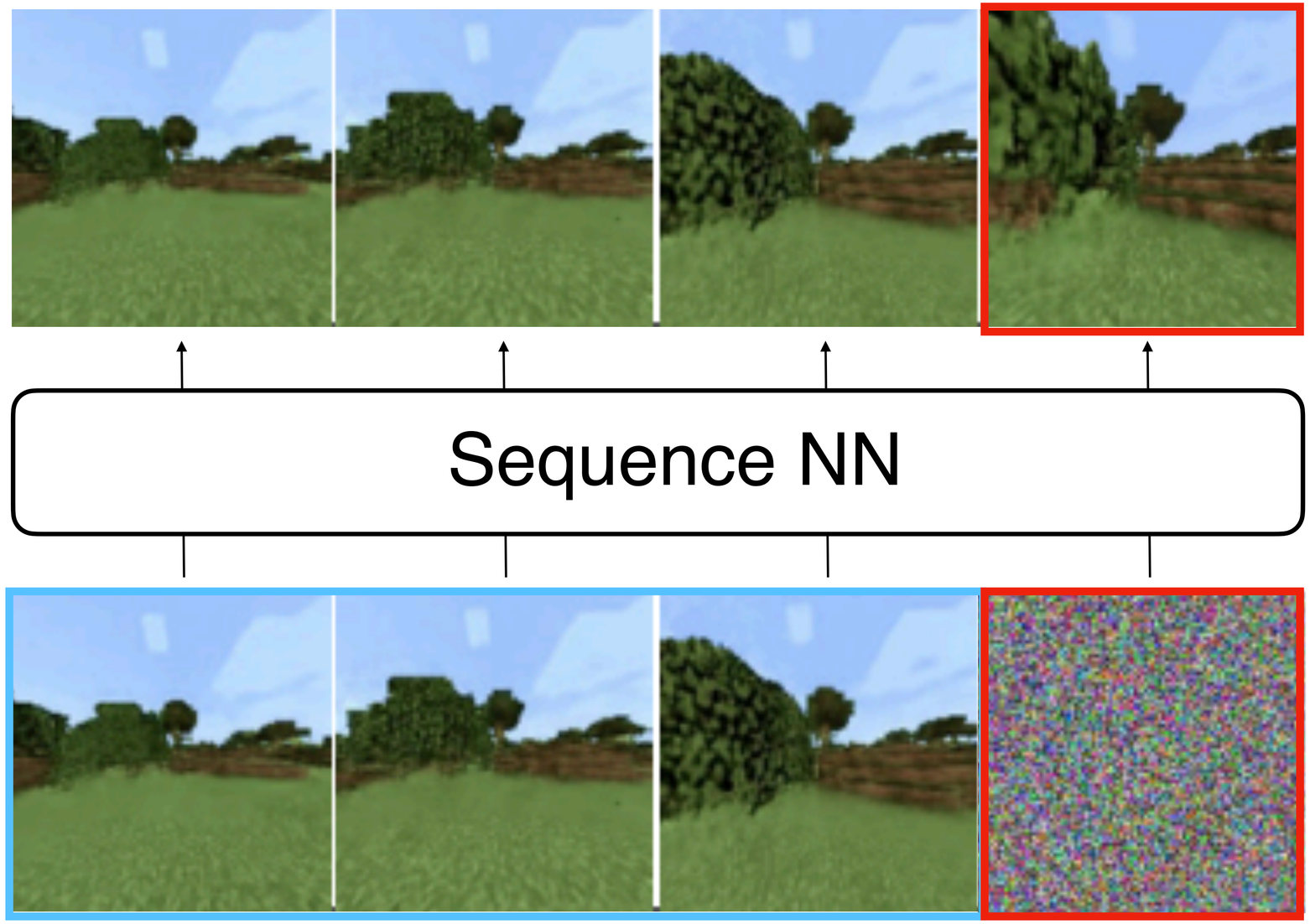


Teacher Forcing for Training Next-Token Pred.



Idea: Given Ground-Truth history, only predict next frame

Teacher Forcing for Training Next-Token Pred.



Idea: Given Ground-Truth history, only predict next frame

Problem with Teacher Forcing: Test-Time is OOD!

Next-token prediction

compounding error

Problem with Teacher Forcing: Test-Time is OOD!

Next-token prediction

compounding error

Problem with Teacher Forcing: Test-Time is OOD!

Next-token prediction

↑

compounding error

Problem: at test time, history is generated as well.

Problem with Teacher Forcing: Test-Time is OOD!

Next-token prediction

compounding error

Problem: at test time, history is generated as well.

This goes OOD quickly and diverges.

Problem with Teacher Forcing: Test-Time is OOD!

Next-token prediction

↑
compounding error

Problem: at test time, history is generated as well.

This goes OOD quickly and diverges.

Various ways of fixing this have been proposed:
Professor Forcing (Goyal, Lamb et al. 2016)
Scheduled Sampling (Bengio et al. 2015, Huszar et al. 2015)

Problem with Teacher Forcing: Test-Time is OOD!

Next-token prediction

↑
compounding error

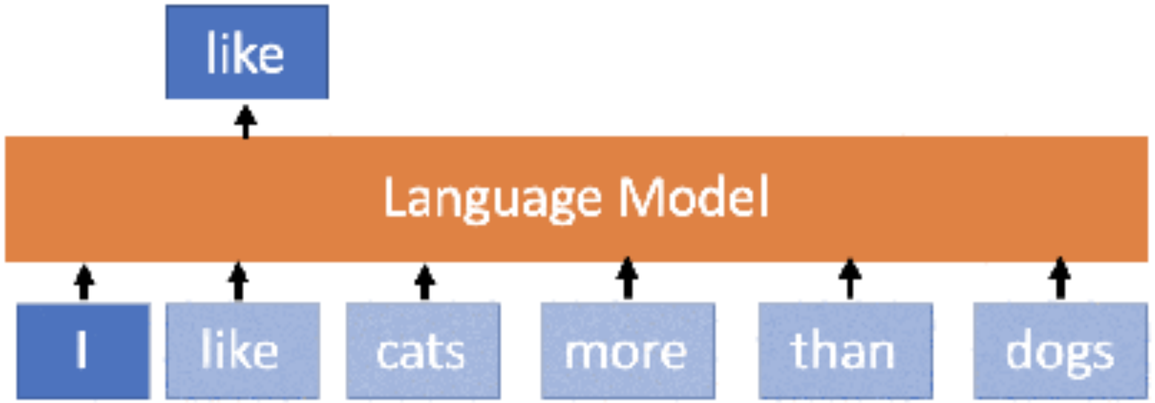
Problem: at test time, history is generated as well.

This goes OOD quickly and diverges.

Various ways of fixing this have been proposed:
Professor Forcing (Goyal, Lamb et al. 2016)
Scheduled Sampling (Bengio et al. 2015, Huszar et al. 2015)

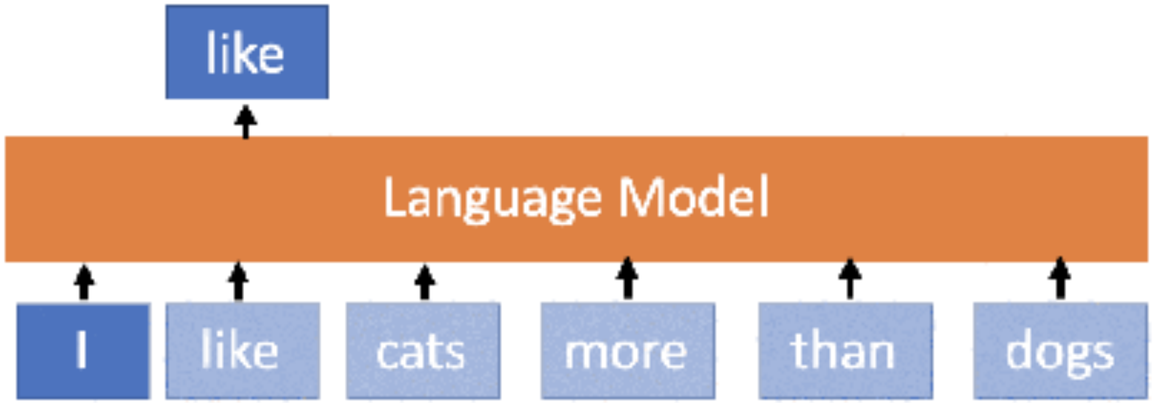
But never really fixed (though not aware of a modern attempt!)

Why does it work for language?



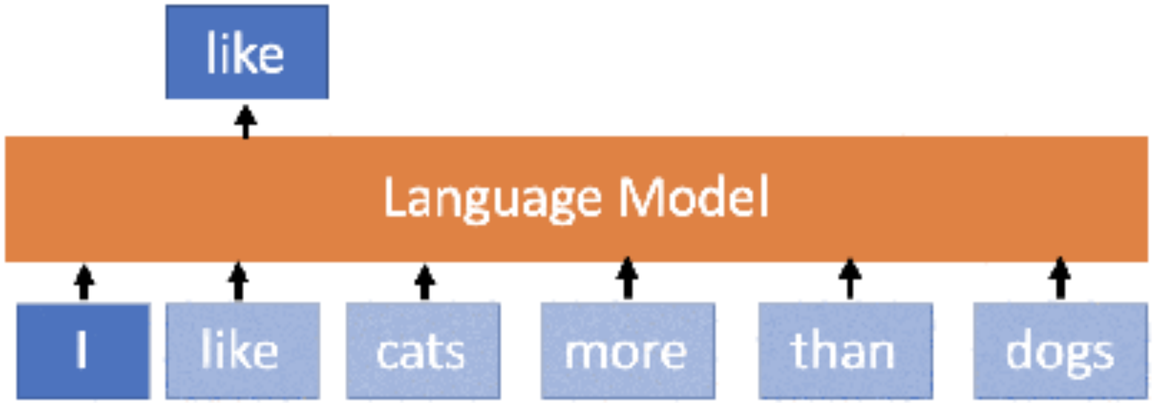
LLMs generally trained with
Next-Token Prediction

Why does it work for language?



LLMs generally trained with
Next-Token Prediction

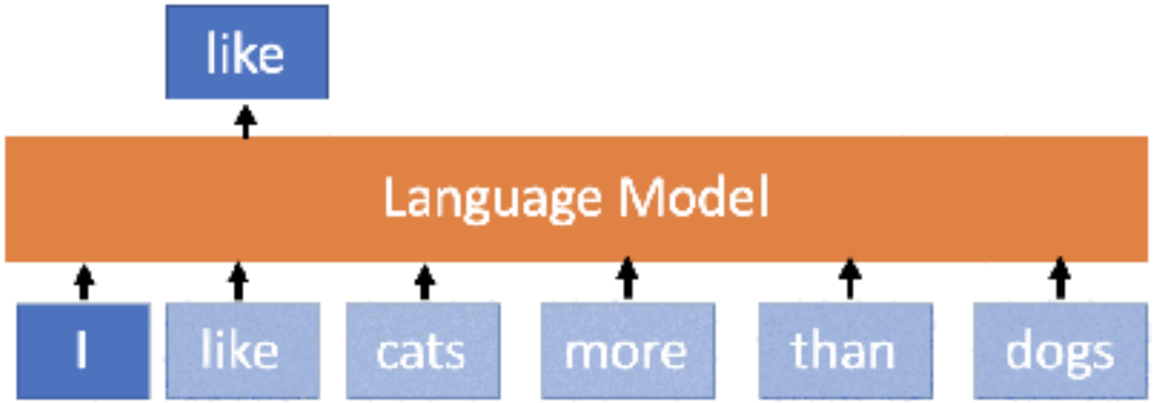
Why does it work for language?



Text has *discrete tokens!* Much lower-dimensional, and cannot "go OOD" on the per-token level!

LLMs generally trained with
Next-Token Prediction

Why does it work for language?

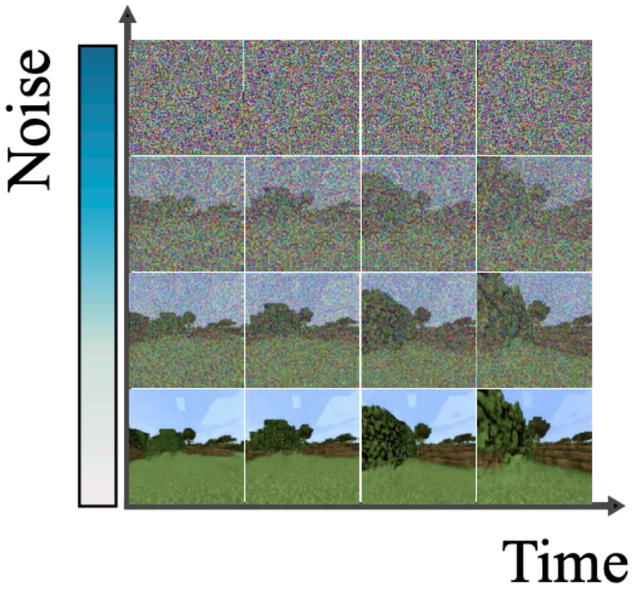


LLMs generally trained with **Next-Token Prediction**

Text has *discrete tokens!* Much lower-dimensional, and cannot "go OOD" on the per-token level!

Some attempts at Video Generative Models with next-token predictions have been made (GAIA-1, Hu et al. 2023): These work with VQ-VAE latents!

Auto-Regressive Generation (Next-Token Prediction)



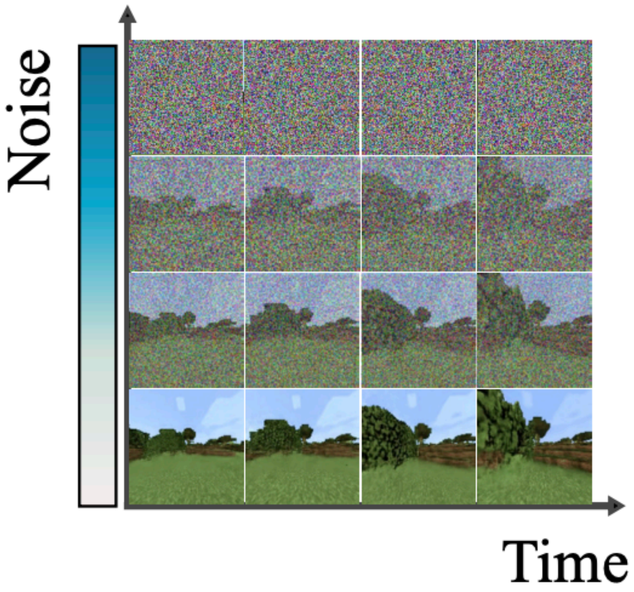
Observation

Latent State

Generation

Add Noise

Auto-Regressive Generation (Next-Token Prediction)



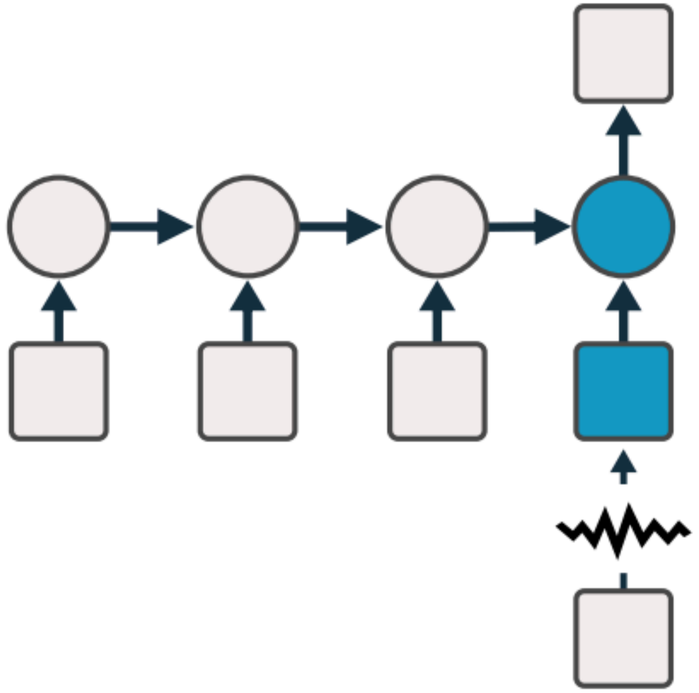
Observation

Latent State

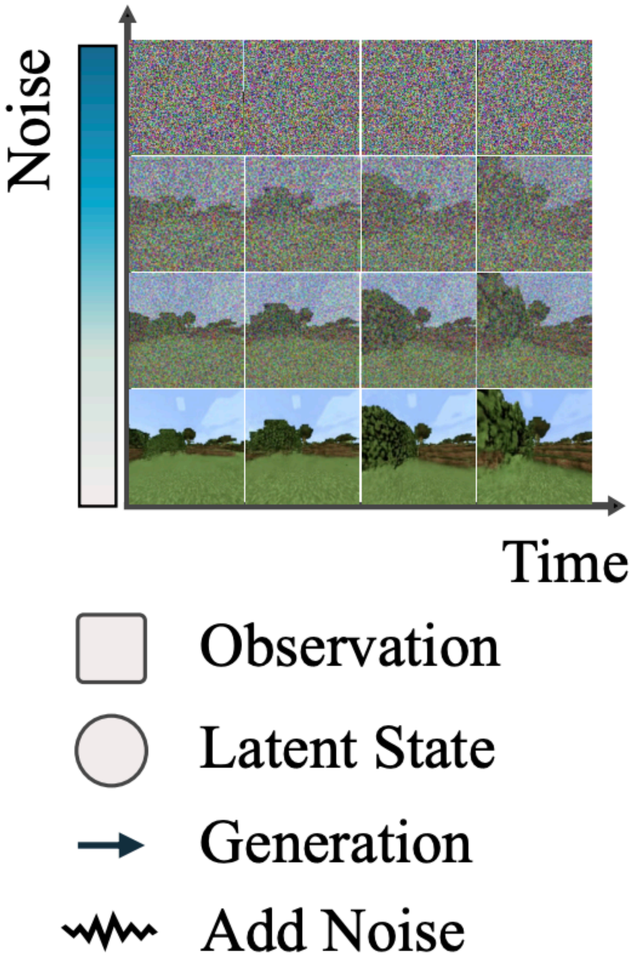
Generation

Add Noise

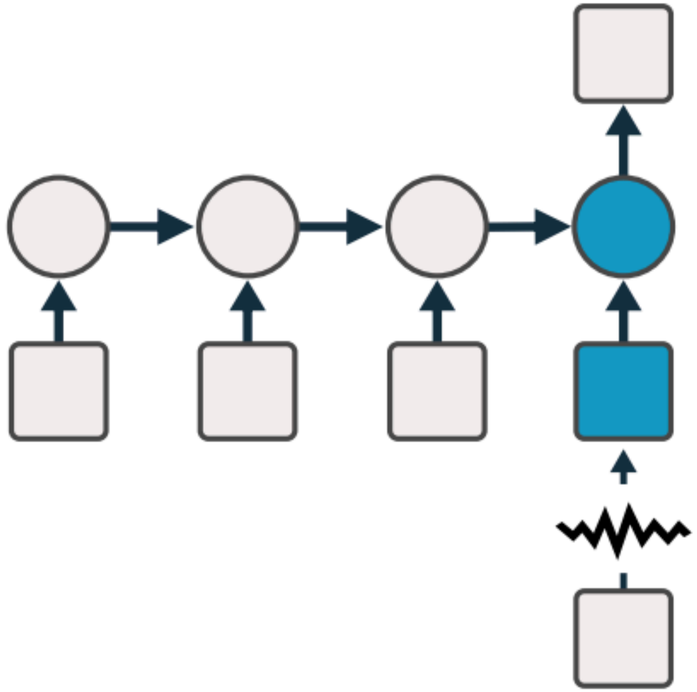
Training



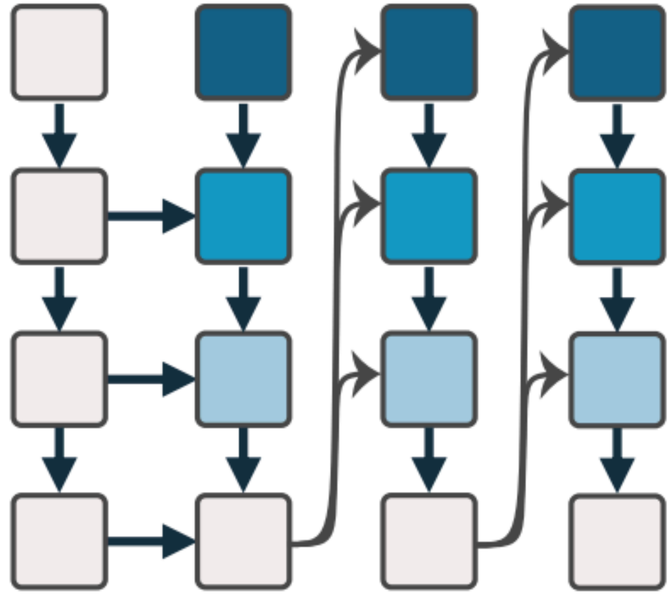
Auto-Regressive Generation (Next-Token Prediction)



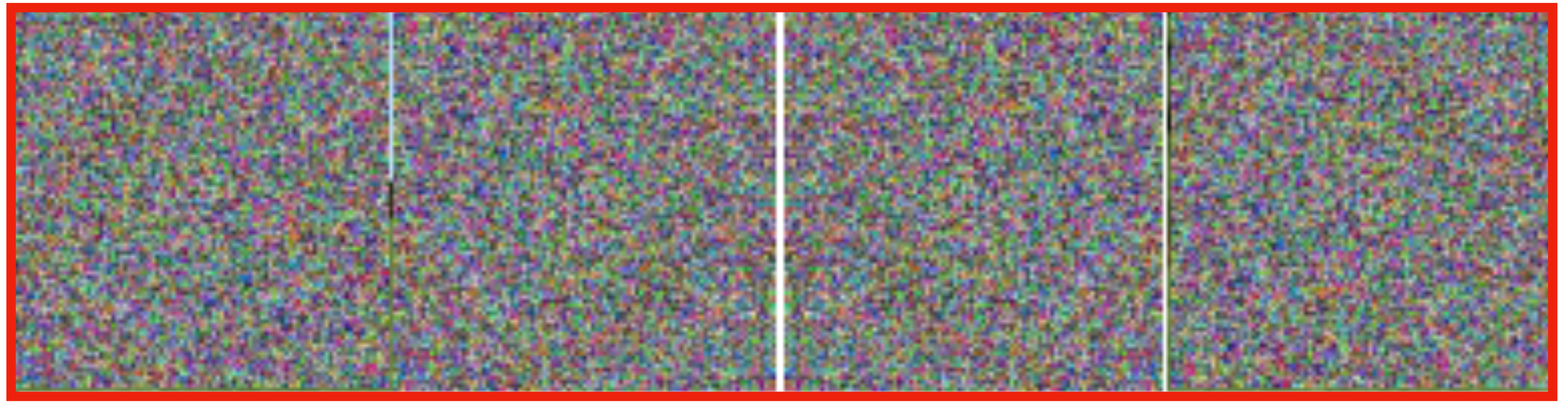
Training



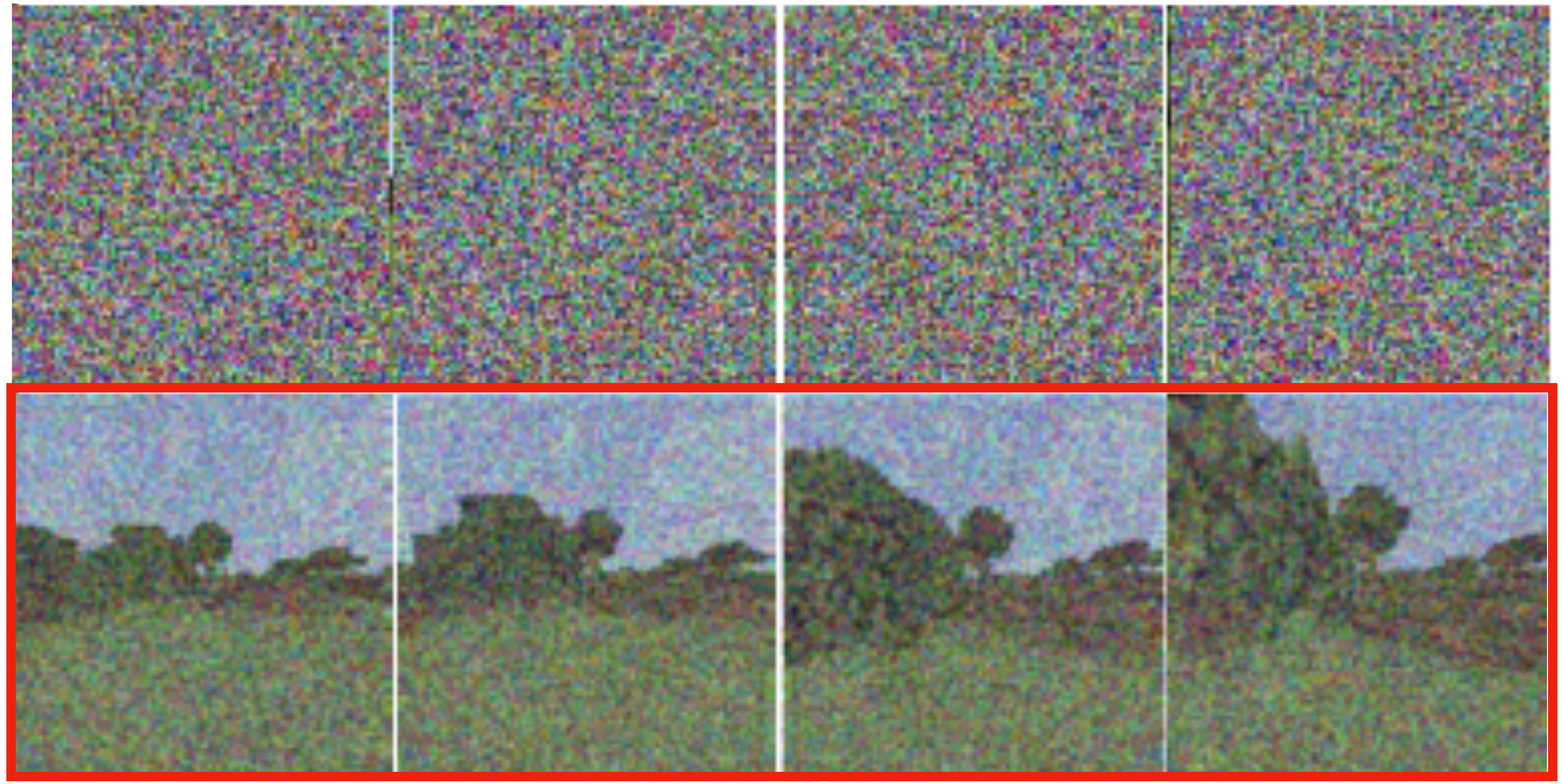
Inference



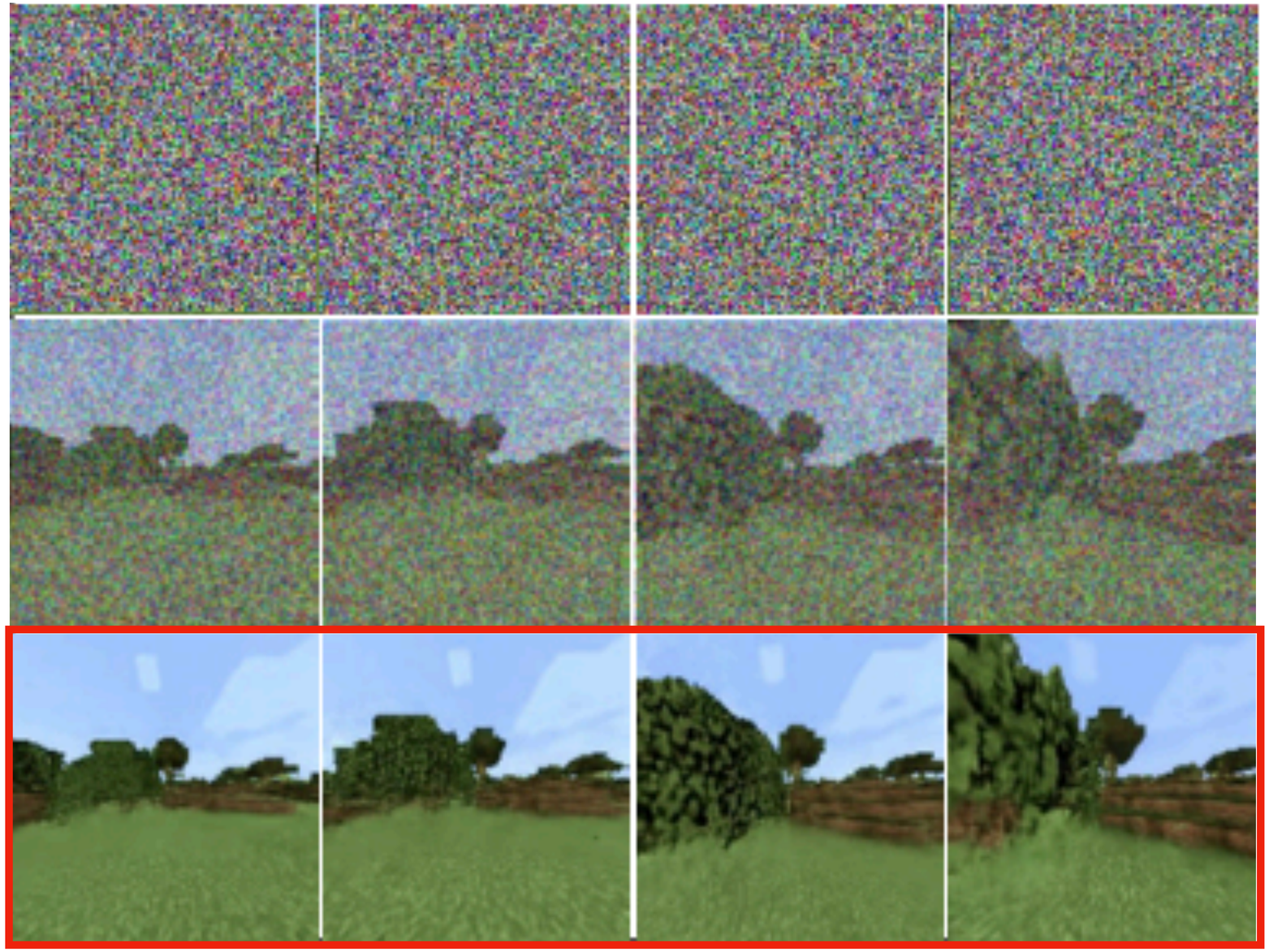
Direct Modeling of the Joint (Full-Sequence Diffusion)



Direct Modeling of the Joint (Full-Sequence Diffusion)

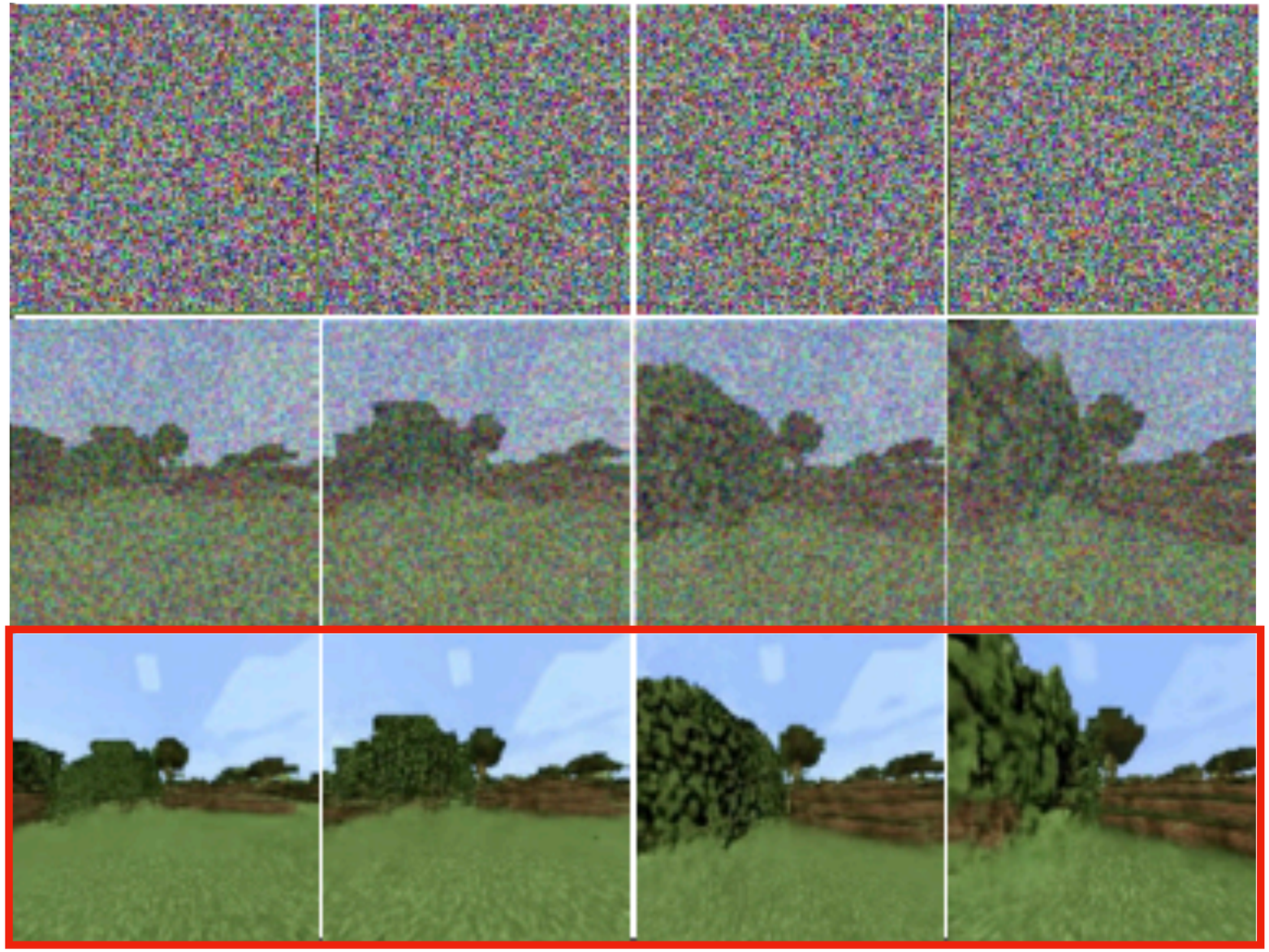


Direct Modeling of the Joint (Full-Sequence Diffusion)



$$p(x_1, x_2, x_3, \dots)$$

Direct Modeling of the Joint (Full-Sequence Diffusion)



Directly models joint!

$$p(x_1, x_2, x_3, \dots)$$

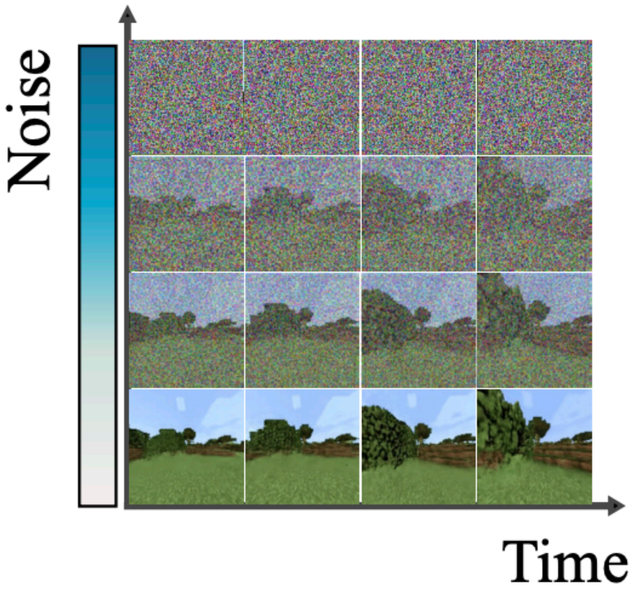
Current-Gen Video Generative Models are Trained with Full-Sequence Diffusion

Current-Gen Video Generative Models are Trained with Full-Sequence Diffusion

Current-Gen Video Generative Models are Trained with Full-Sequence Diffusion

Sources: Runway, OpenAI

Full-Sequence Diffusion



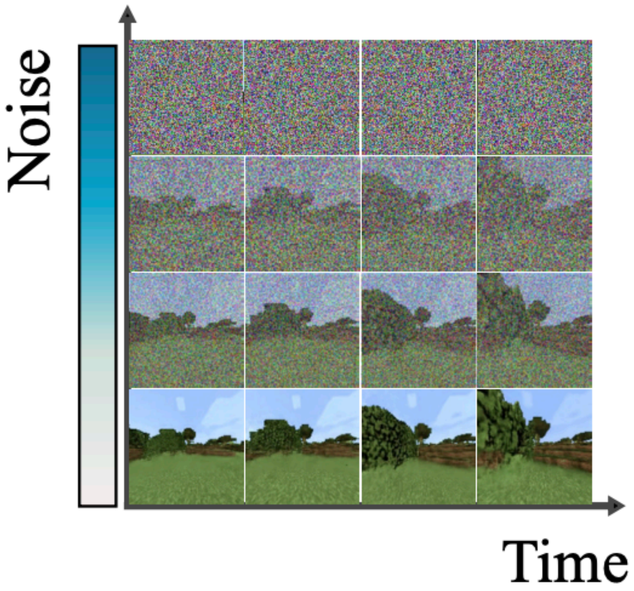
Observation

Latent State

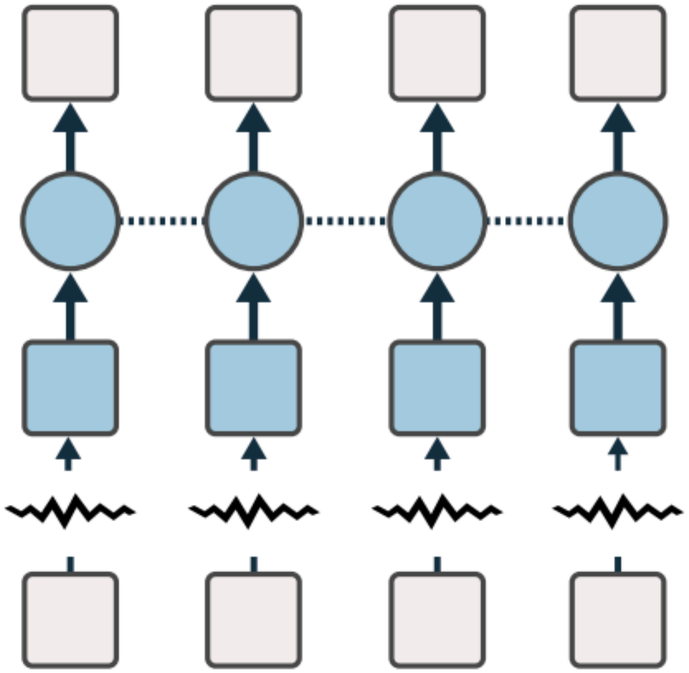
Generation

Add Noise

Full-Sequence Diffusion



Training



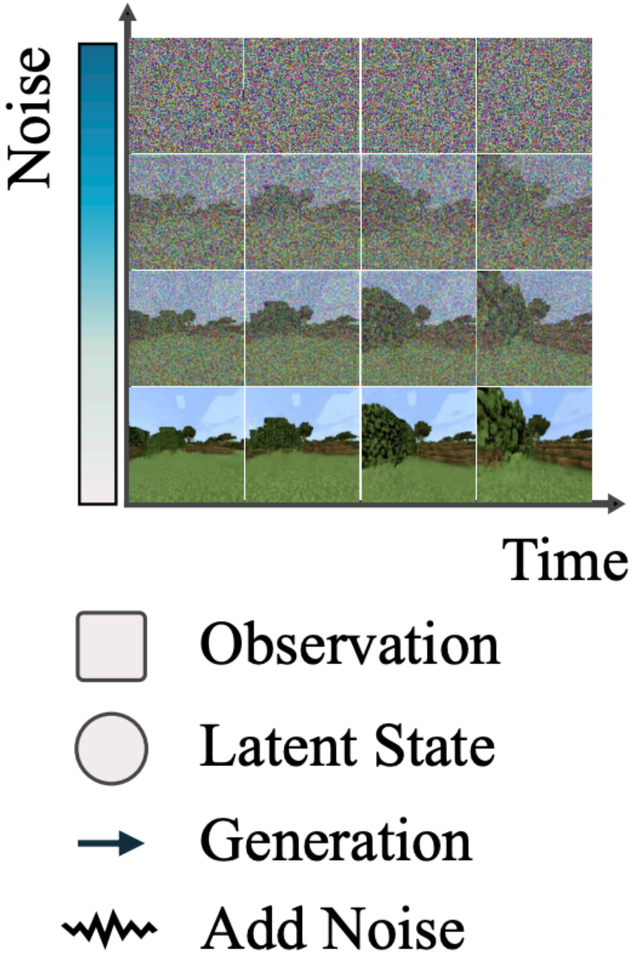
Observation

Latent State

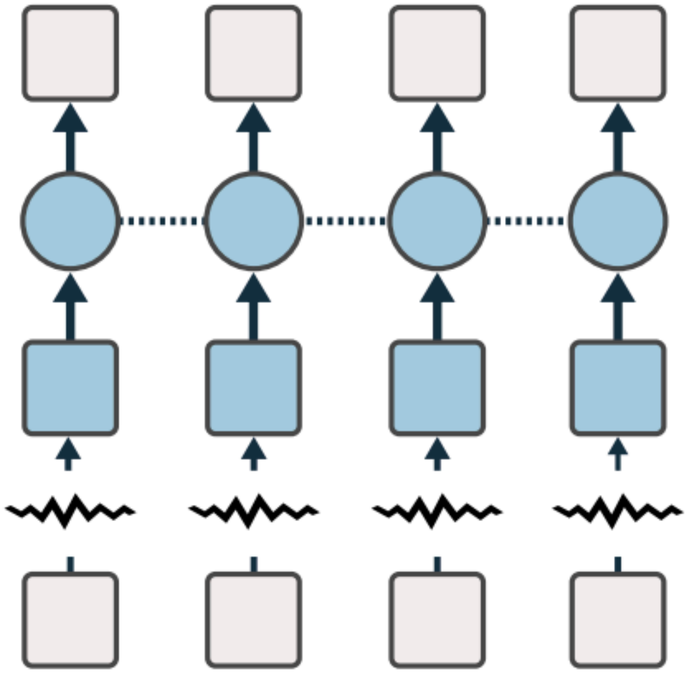
Generation

Add Noise

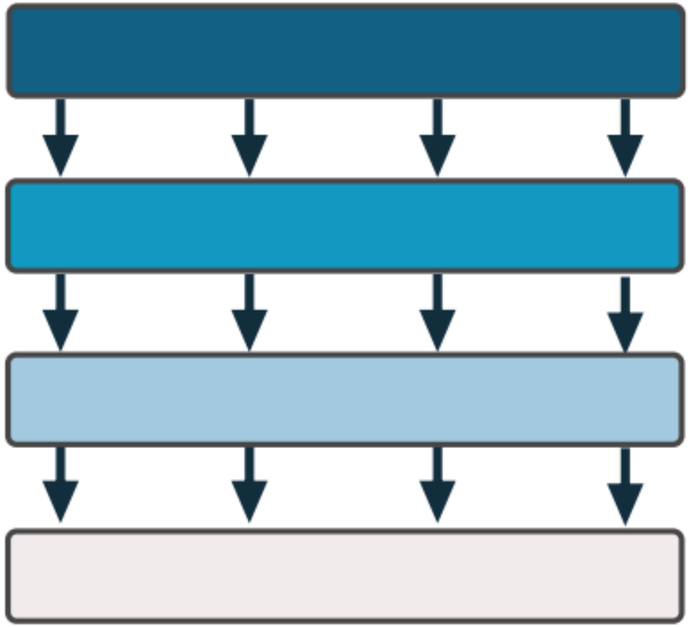
Full-Sequence Diffusion



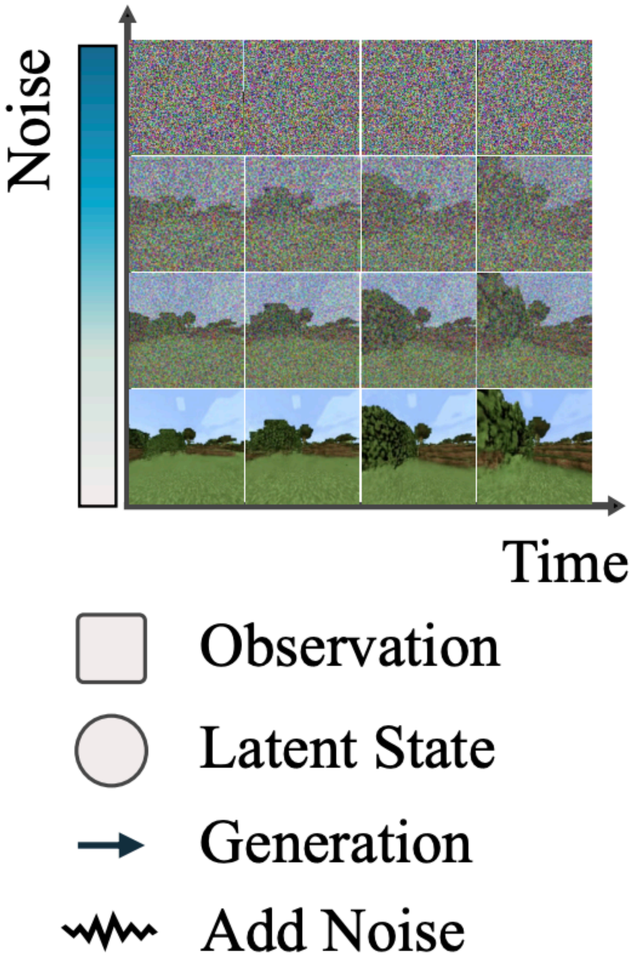
Training



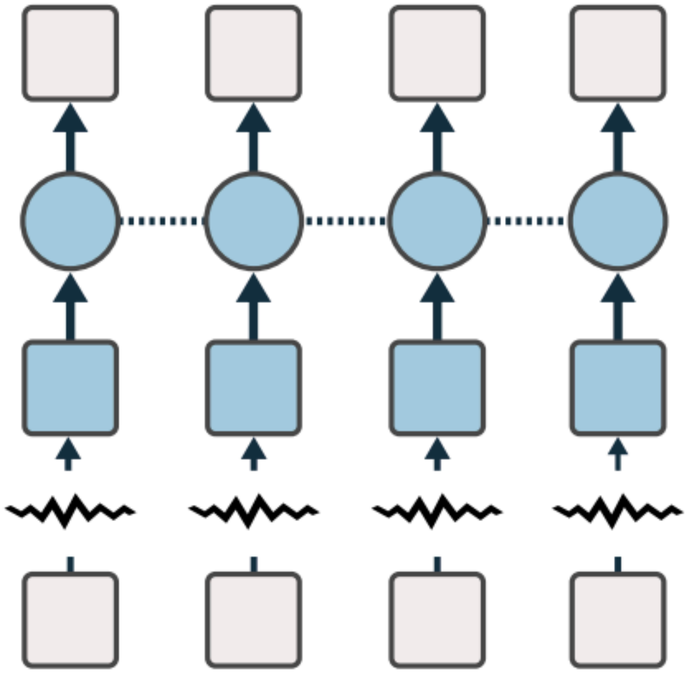
Inference



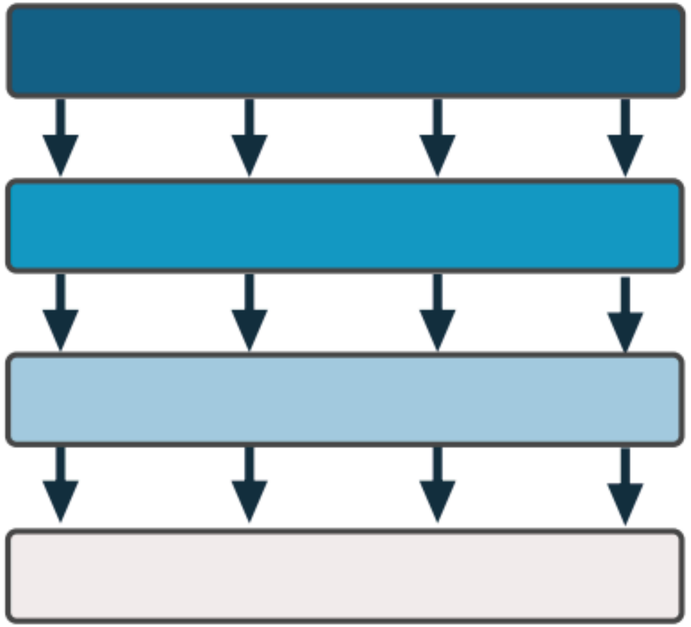
Full-Sequence Diffusion



Training

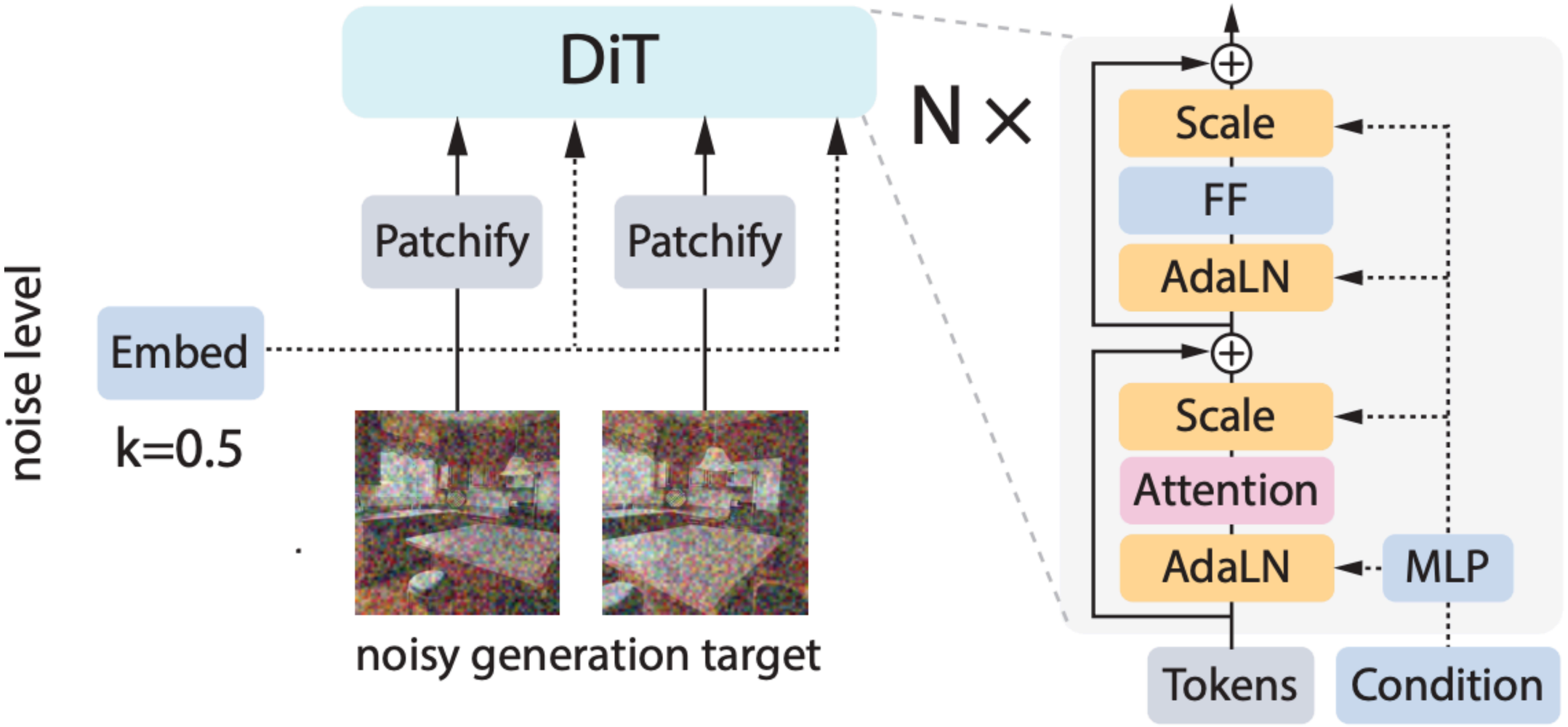


Inference

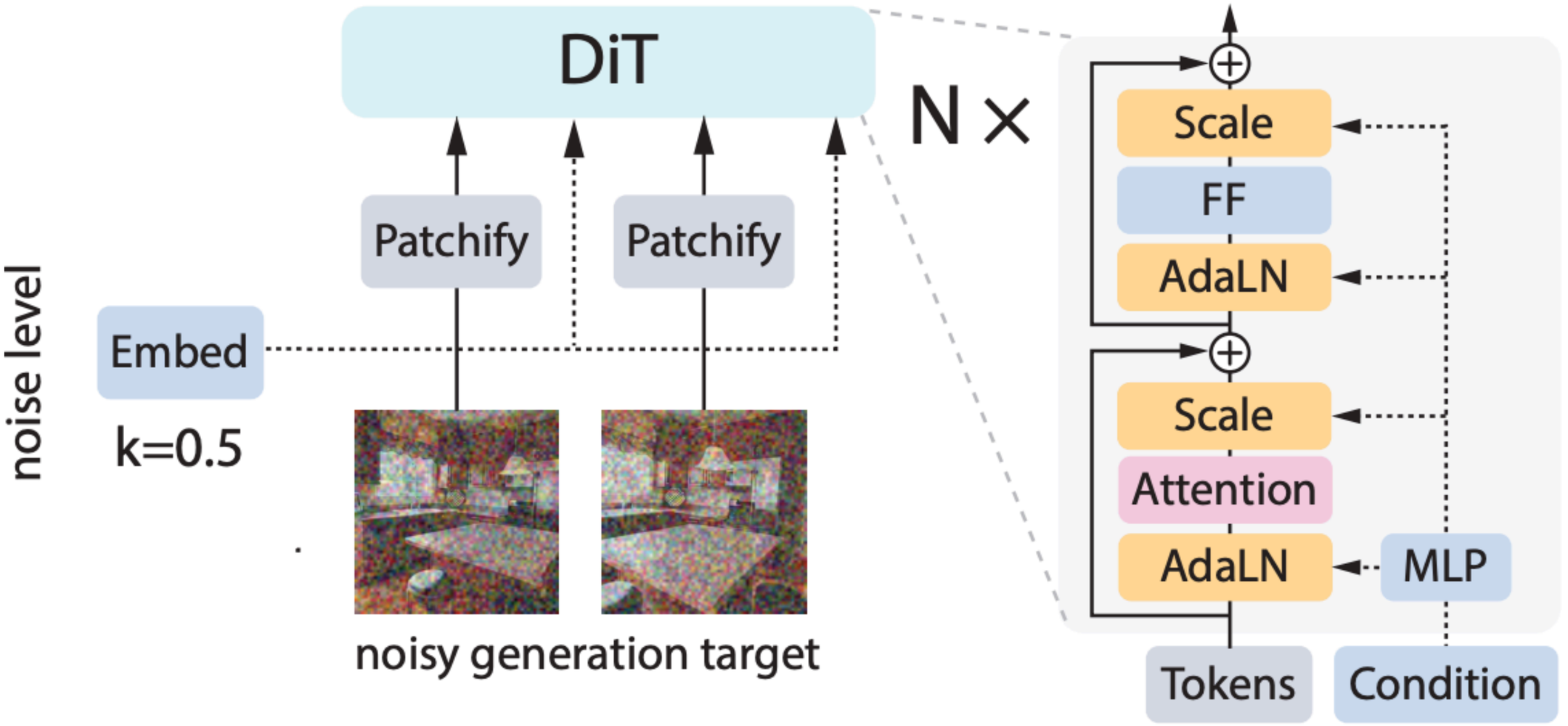


Examples: SORA, Meta Movie Gen, Diffusion Probabilistic Modeling for Video Generation, [Yang et al. 2022], Video Diffusion Models [Ho et al. 2022]

Dominant Paradigm: Full-Sequence Diffusion with DiT

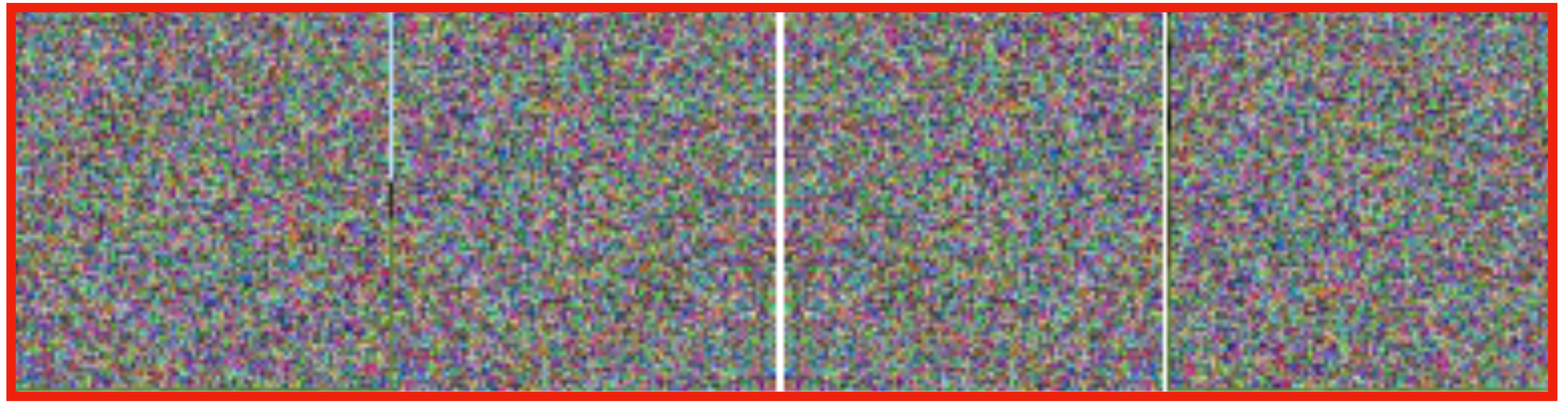


Dominant Paradigm: Full-Sequence Diffusion with DiT

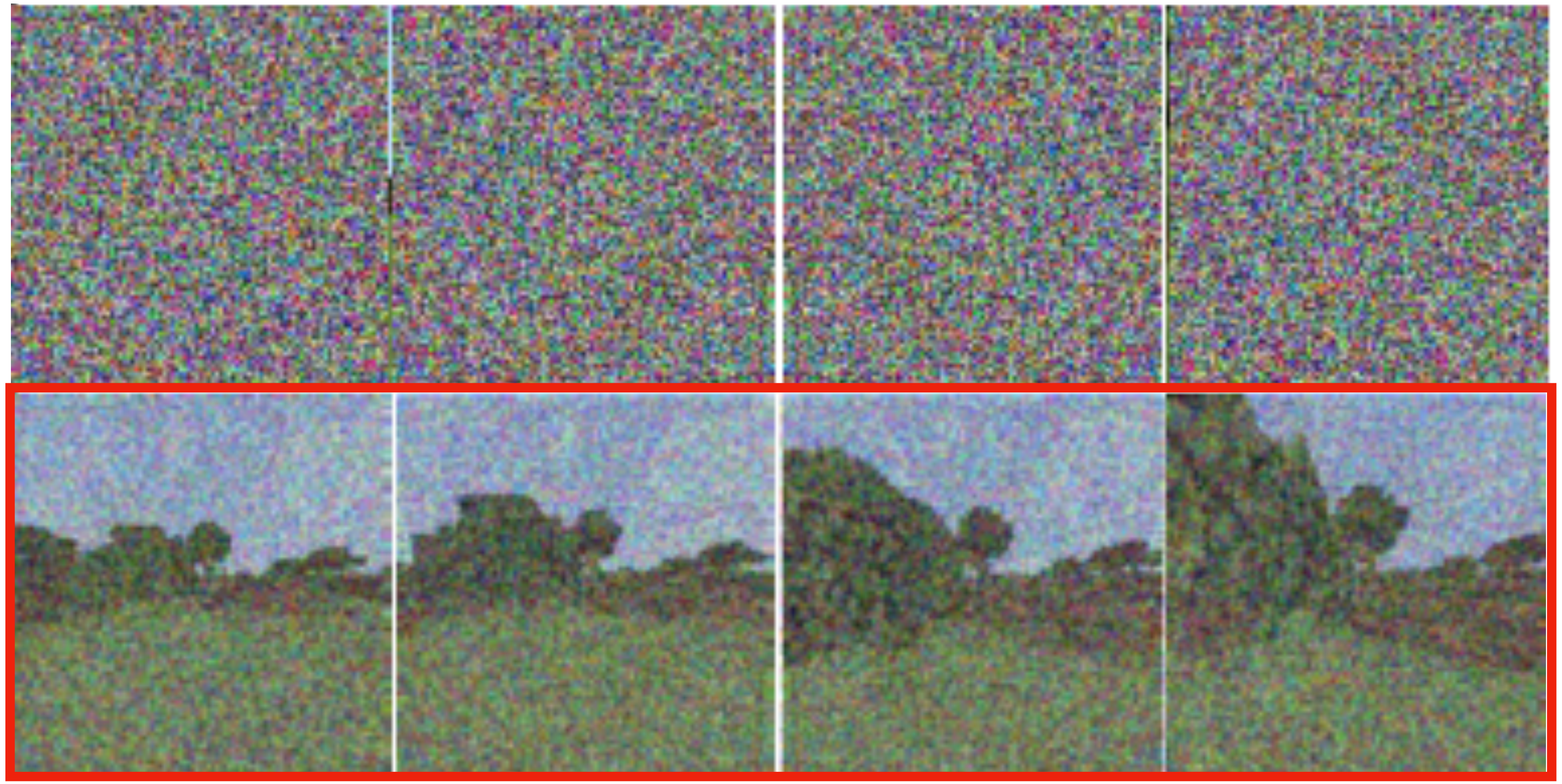


DiT: Peebles et al. 2022, Figure: History-Guided Video Diffusion

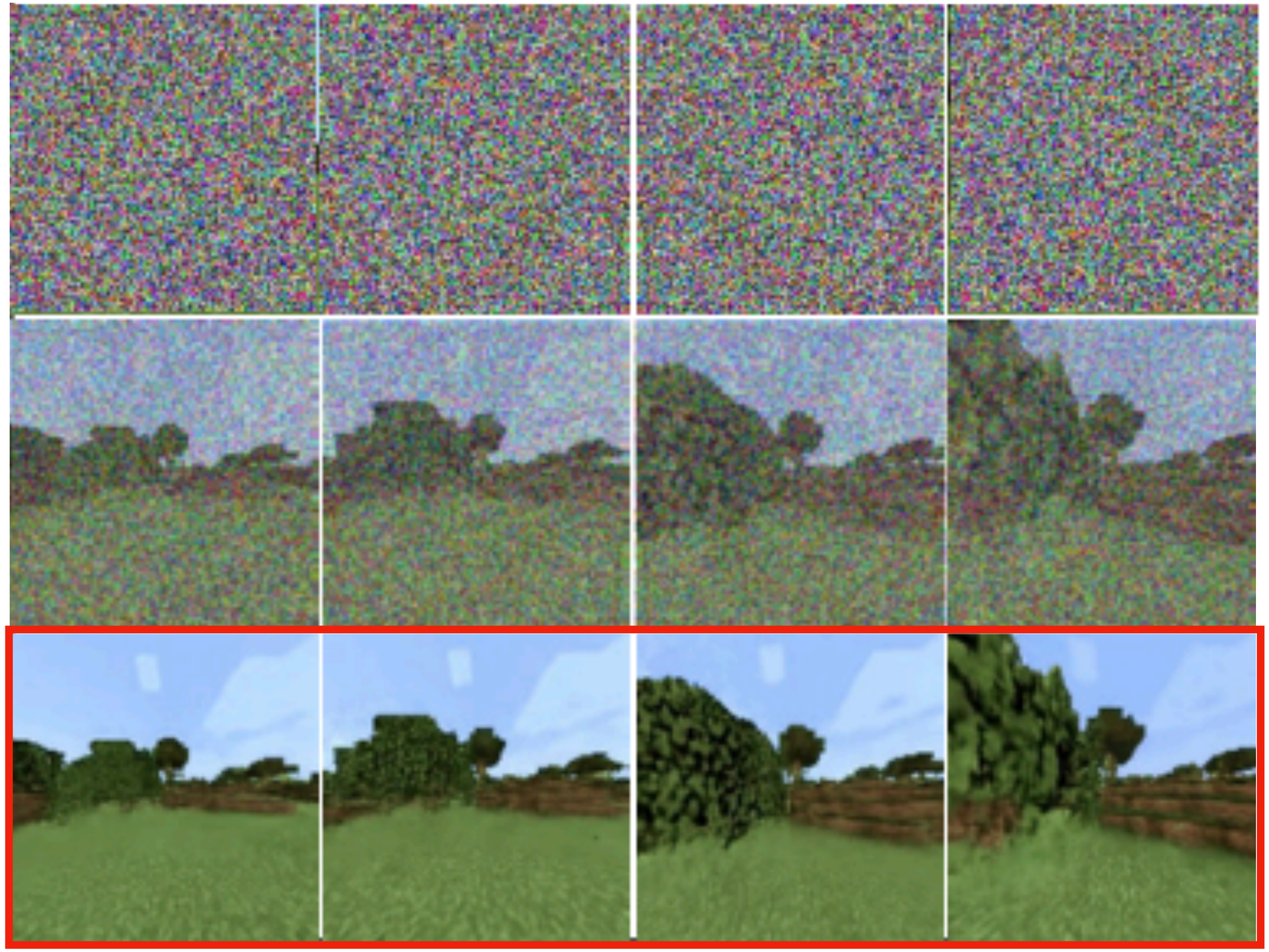
Problem: How to extend video?



Problem: How to extend video?

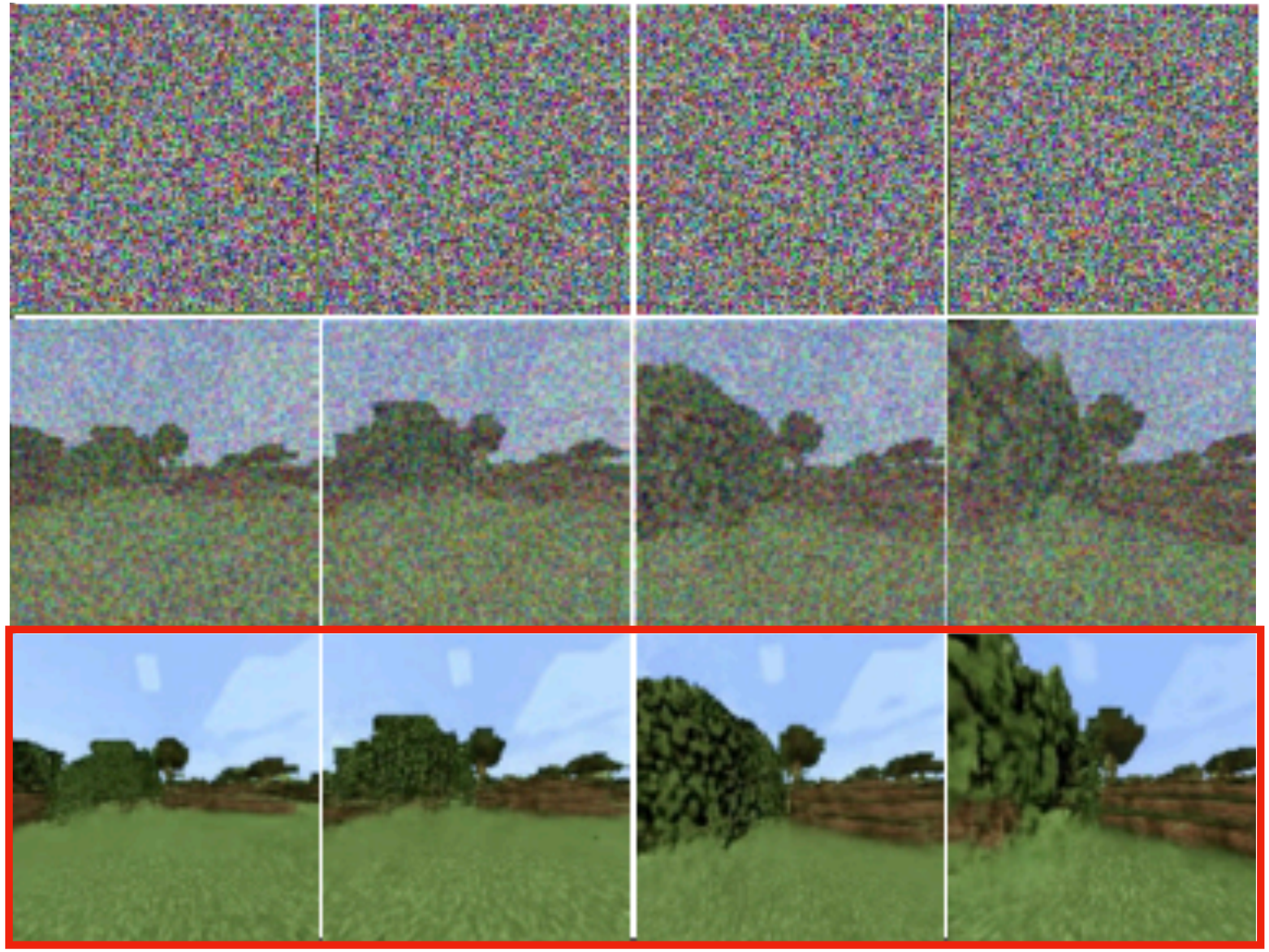


Problem: How to extend video?



$$p(x_1, x_2, x_3, \dots)$$

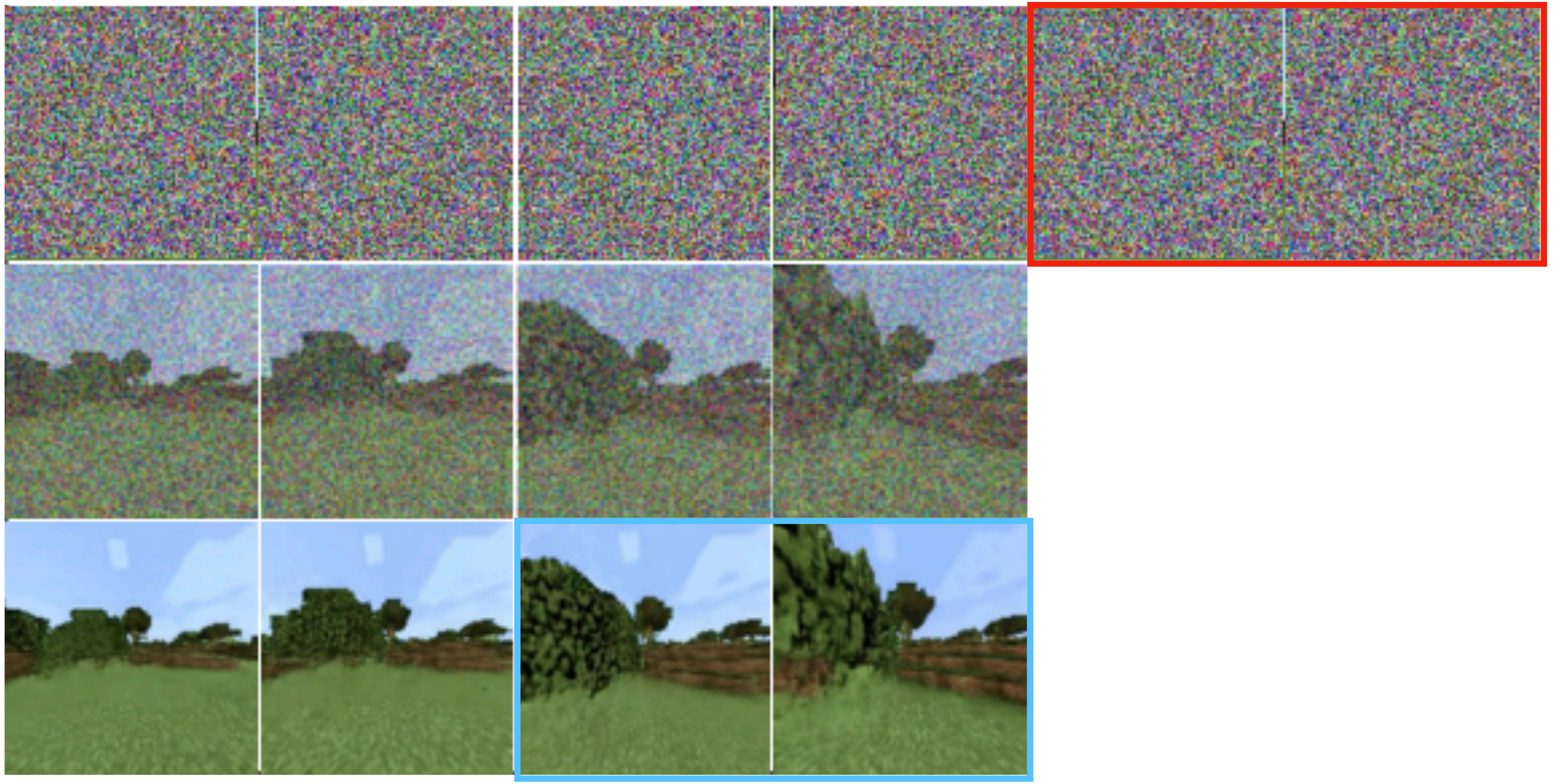
Problem: How to extend video?



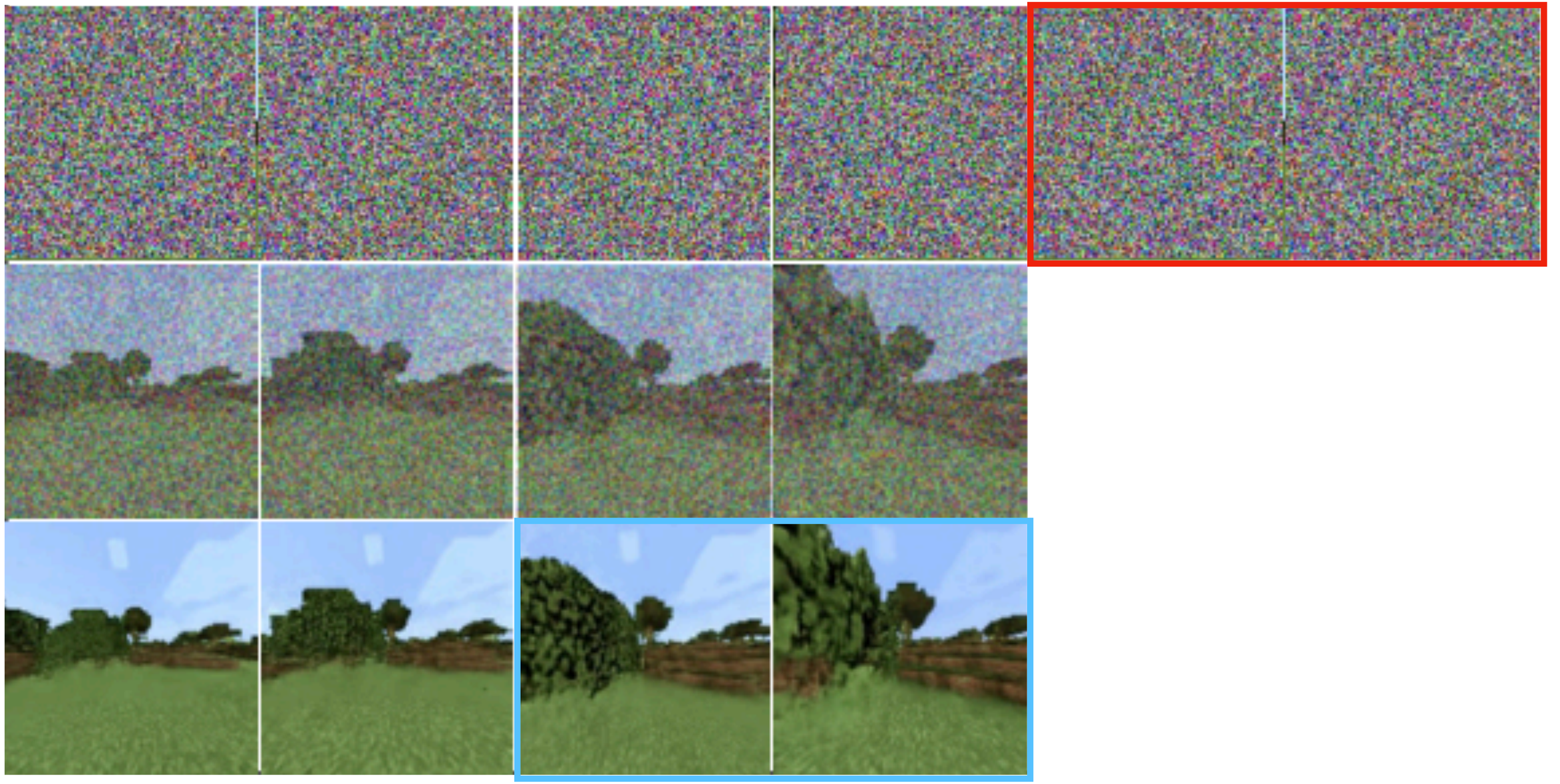
Directly models joint!

$$p(x_1, x_2, x_3, \dots)$$

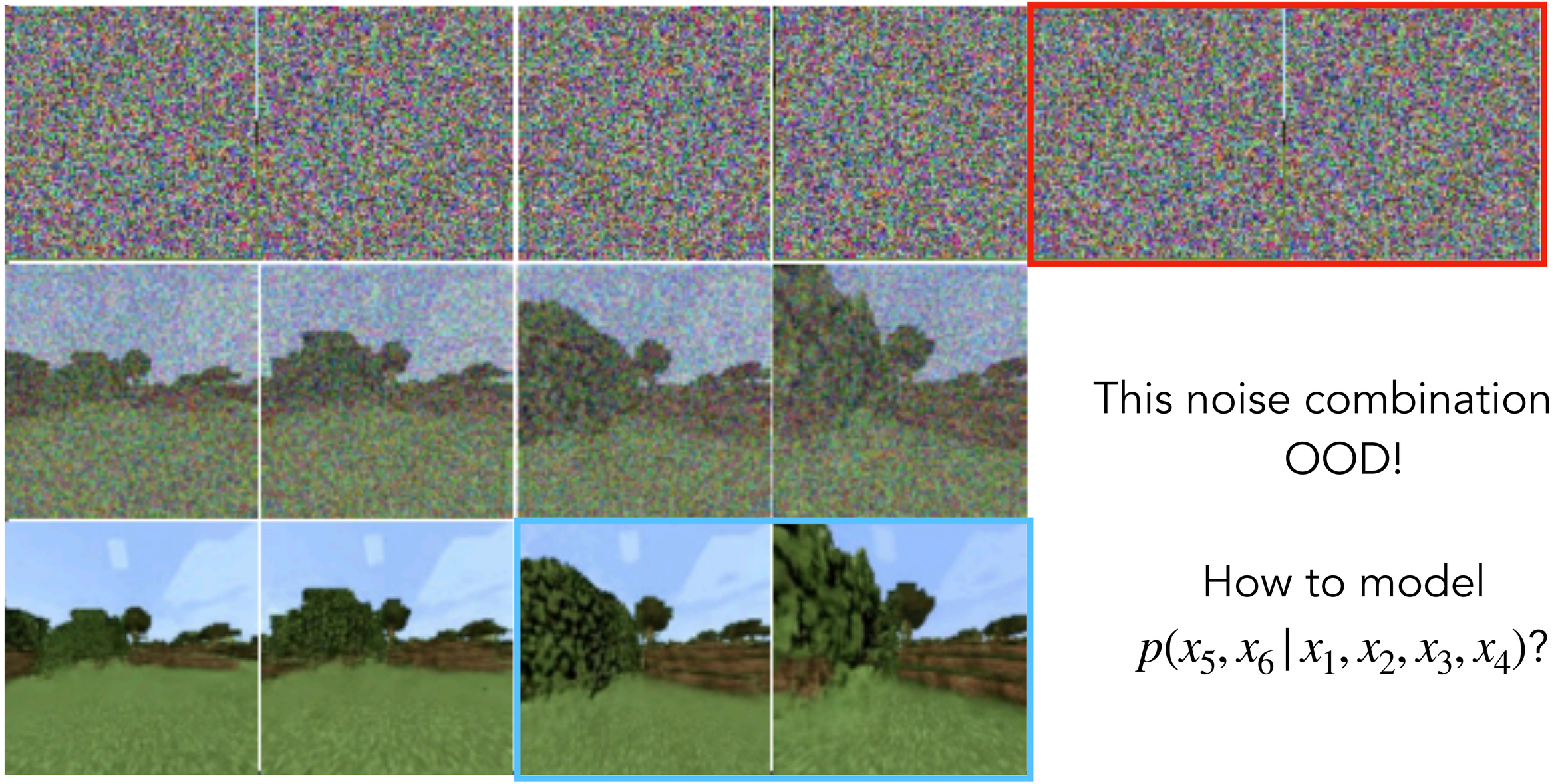
Problem: How to extend video?



Problem: How to extend video?



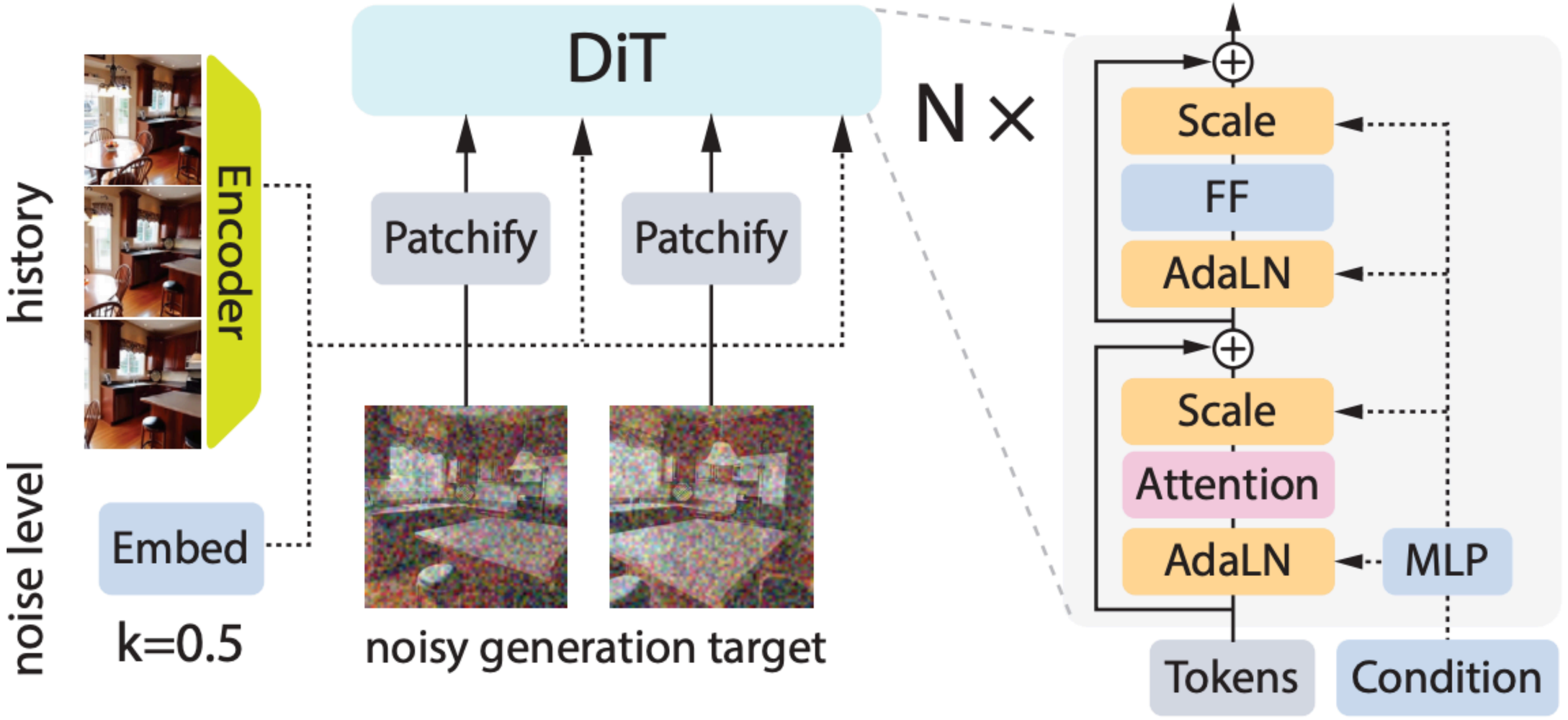
Problem: How to extend video?



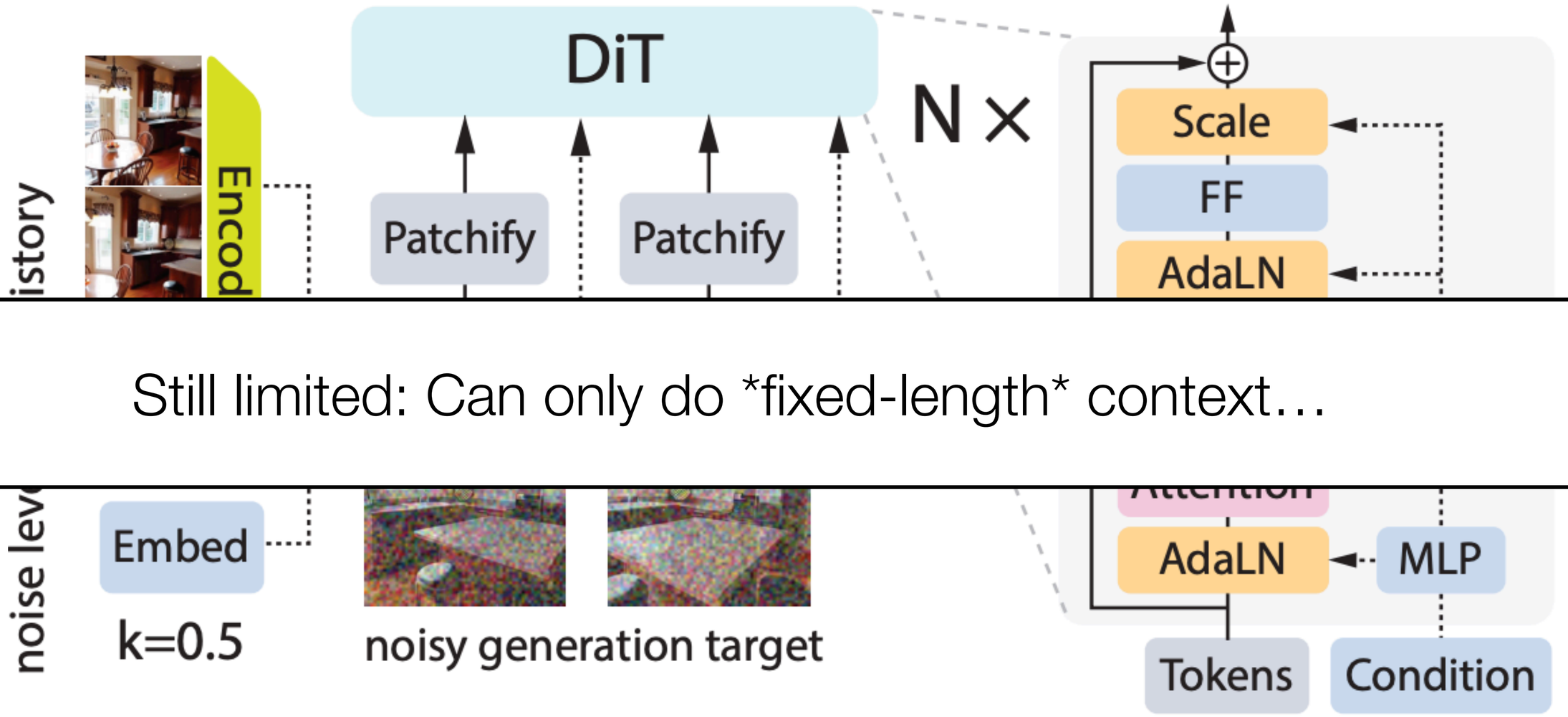
This noise combination is OOD!

How to model $p(x_5, x_6 \mid x_1, x_2, x_3, x_4)$?

Possible Solution: Feed in History Frames via Encoder

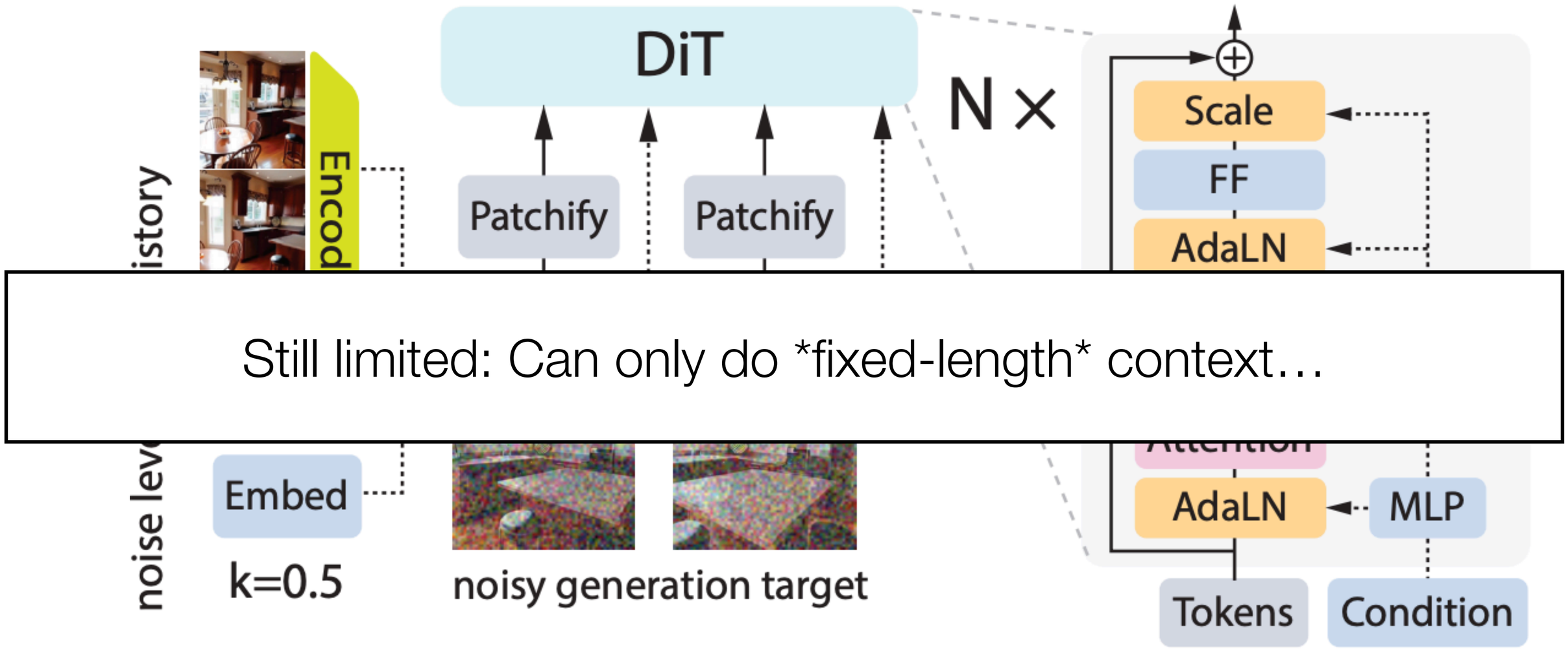


Possible Solution: Feed in History Frames via Encoder



Still limited: Can only do *fixed-length* context…

Possible Solution: Feed in History Frames via Encoder



Still limited: Can only do *fixed-length* context…

[Peebles & Xie 2023, Bao et al. 2023, Rombach et al. 2022]

Diffusion Forcing: Next-token Prediction Meets Full-Sequence Diffusion

Boyuan Chen
MIT CSAIL
boyuanc@mit.edu

Diego Marti Monso*
Technical University of Munich
diego.marti@tum.de

Yilun Du
MIT CSAIL
yilundu@mit.edu

Max Simchowitz
MIT CSAIL
msimchow@mit.edu

Russ Tedrake
MIT CSAIL
russt@mit.edu

Vincent Sitzmann
MIT CSAIL
sitzmann@mit.edu

Abstract

This paper presents Diffusion Forcing, a new training paradigm where a diffusion model is trained to denoise a set of tokens with *independent* per-token noise levels. We apply Diffusion Forcing to sequence generative modeling by training a causal next-token prediction model to generate one or several future tokens without fully diffusing past ones. Our approach is shown to combine the strengths of next-token prediction models, such as variable-length generation, with the strengths of full-sequence diffusion models, such as the ability to guide sampling to desirable trajectories. Our method offers a range of additional capabilities, such as (1) rolling-out sequences of continuous tokens, such as video, with lengths past the training horizon, where baselines diverge and (2) new sampling and guiding schemes that uniquely profit from Diffusion Forcing's variable-horizon and causal architecture, and which lead to marked performance gains in decision-making and planning tasks. In addition to its empirical success, our method is proven to optimize a variational lower bound on the likelihoods of all subsequences of tokens drawn from the true joint distribution. Project website: https://boyuan.space/diffusion-forcing

1 Introduction

Probabilistic sequence modeling plays a crucial role in diverse machine learning applications including natural language processing [6, 48], video prediction [32, 70, 26] and decision making [3, 22]. Next-token prediction models in particular have a number of desirable properties. They enable the generation of sequences with varying length [33, 21, 38] (generating only a single token or an "infinite" number of tokens via auto-regressive sampling), can be conditioned on varying amounts of history [21, 38], support efficient tree search[71, 23, 25], and can be used for online feedback

History-Guided Video Diffusion

Kiwhan Song* [1] **Boyuan Chen*** [1] **Max Simchowitz** [1] **Yilun Du** [1] **Russ Tedrake** [1] **Vincent Sitzmann** [1]

Abstract

Classifier-free guidance (CFG) is a key technique for improving conditional generation in diffusion models, enabling more accurate control while enhancing sample quality. It is natural to extend this technique to video diffusion, which generates video conditioned on a variable number of context frames, collectively referred to as history. However, we find two key challenges to guiding with variable-length history: architectures that only support fixed-size conditioning, and the empirical observation that CFG-style history dropout performs poorly. To address this, we propose the Diffusion Forcing Transformer (DFoT), a video diffusion architecture and theoretically grounded training objective that jointly enable conditioning on a flexible number of history frames. We then introduce *History Guidance*, a family of guidance methods uniquely enabled by DFoT. We show that its simplest form, *vanilla history guidance*, already significantly improves video generation quality and temporal consistency. A more advanced method, *history guidance across time and frequency* further enhances motion dynamics, enables compositional generalization to out-of-distribution history, and can stably roll out extremely long videos. Website: this URL

Introduction

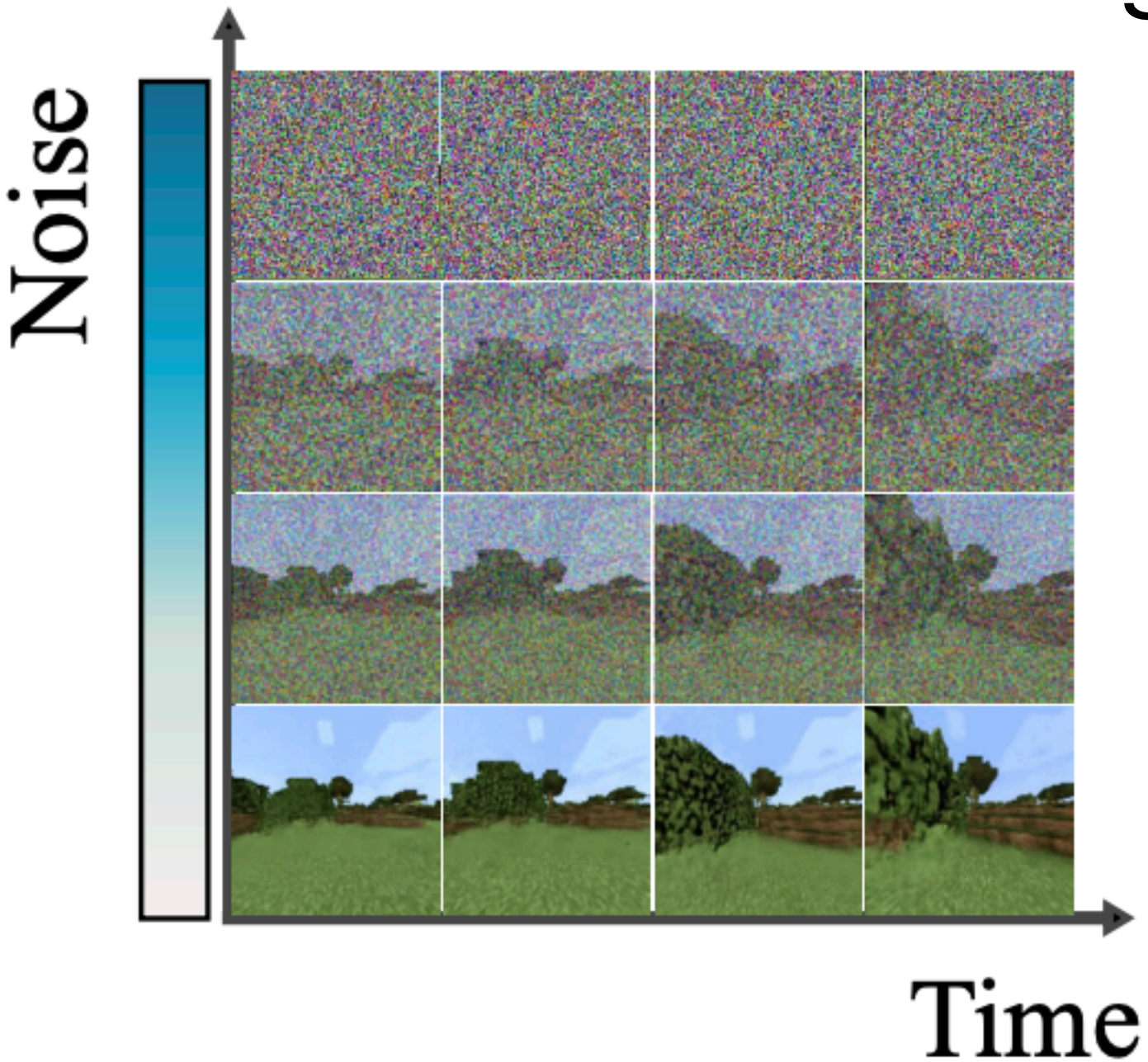
iffusion models are effective generative models in domains ch as image, sound, and video. Critical to their success classifier-free guidance (CFG) (Ho & Salimans, 2022),

question: Can we use different portions of history - variable lengths, subsets of frames, and even different image-domain frequencies - as a form of guidance for video generation? Importantly, CFG with flexible history is incompatible with existing diffusion model architectures and the most obvious fix significantly degrades sample quality (see Section 3).

To address these limitations, we propose the Diffusion Forcing Transformer (DFoT), a video diffusion framework that enables flexible conditioning on any portion of the input history. Extending the "noising-as-masking" paradigm in Diffusion Forcing (Chen et al., 2024) to non-causal transformers, DFoT trains video diffusion models by applying independent noise levels to each frame. During sampling, portions of the history can be selectively masked with noise, enabling flexible conditioning and guidance. For instance, in CFG, the unconditional score corresponds to our model with the entire history masked out. Notably, DFoT is compatible with existing architectures such as DiT (Peebles & Xie, 2023) and U-ViT (Hoogeboom et al., 2023; 2024) and can be efficiently implemented through fine-tuning of pre-trained video diffusion models.

At sampling time, the DFoT facilitates a family of history-conditioned guidance methods, collectively referred to as *History Guidance* (HG). The simplest of these, *Vanilla History Guidance* (HG-v), uses an arbitrary length of history as the conditioning variable for CFG. Notably, even this simple method significantly enhances video quality. We further introduce two advanced methods enabled by the DFoT: *Temporal History Guidance* (HG-t) and *Fractional History Guidance* (HG-f) . These extend history guidance beyond a special case of CFG. Temporal History Guidance combines scores from different history windows. Fractional History Guidance conditions on history windows corrupted by vary-

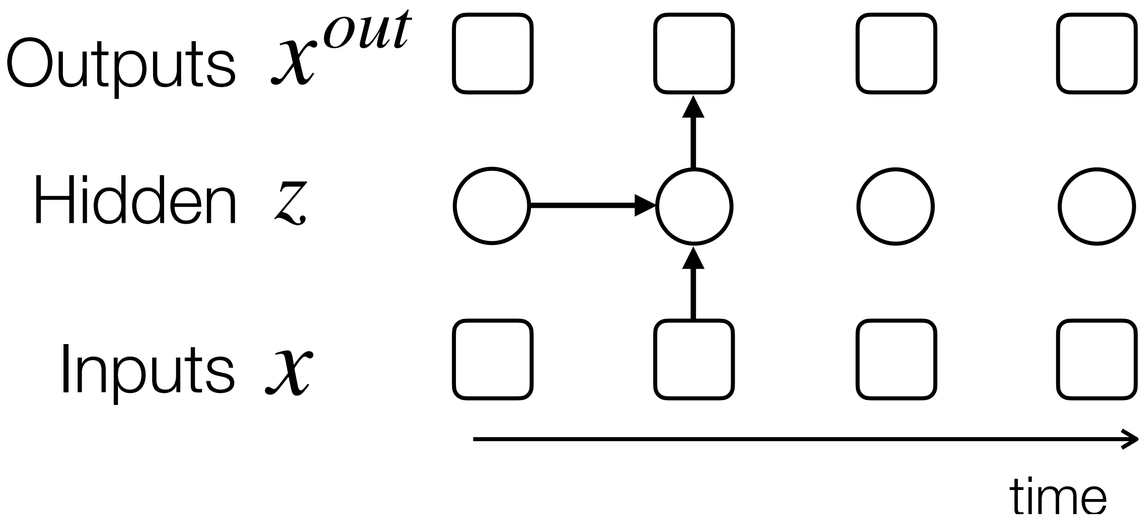
Let’s really *think* about noise as masking



Two axes of
sequence generation

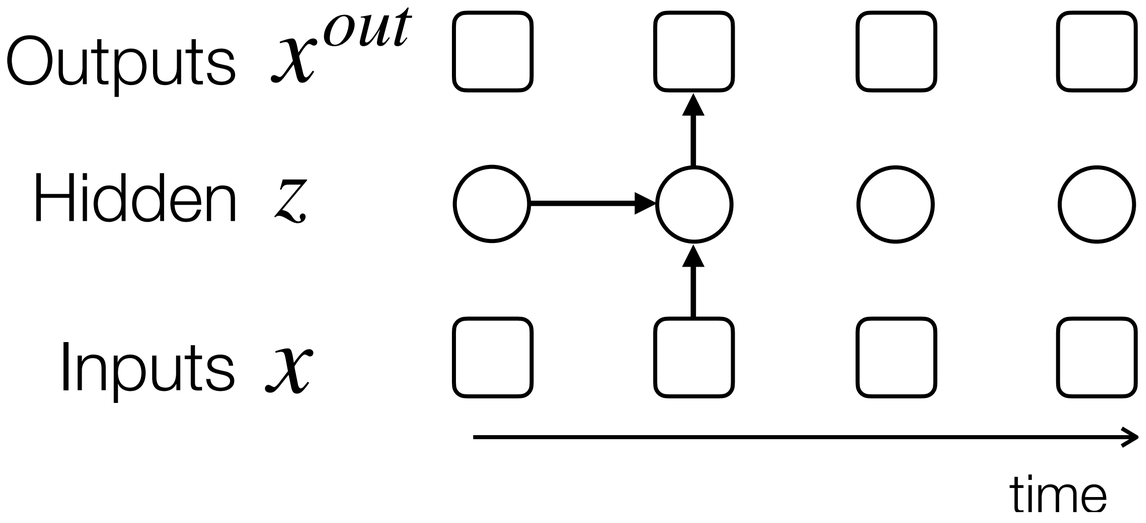
Sequence NN Architectures

Recurrent Neural Networks (RNNs)



Sequence NN Architectures

Recurrent Neural Networks (RNNs)

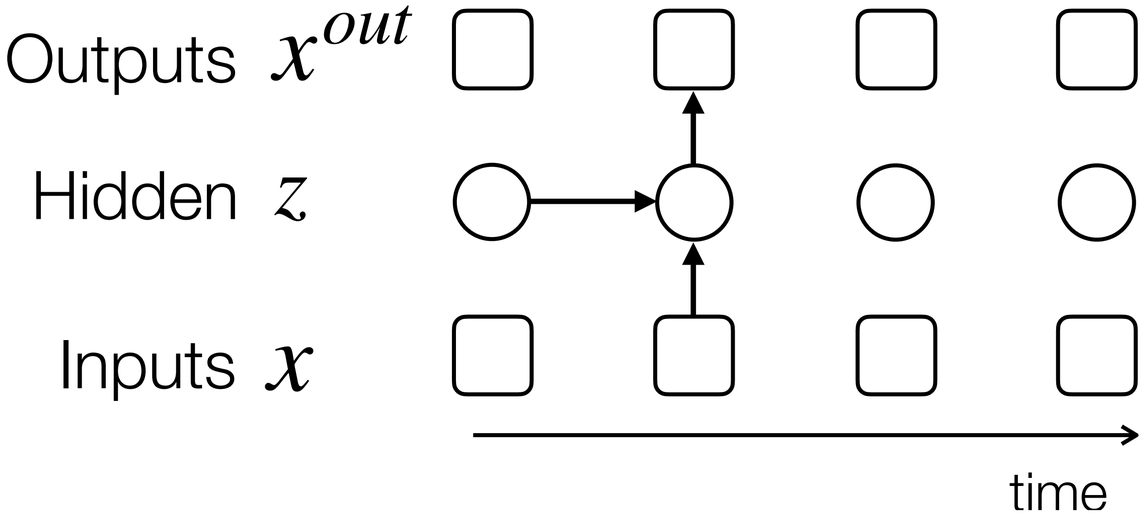


$$z_t = f(z_{t-1}, x_t)$$

$$x_t^{out} = g(z_t)$$

Sequence NN Architectures

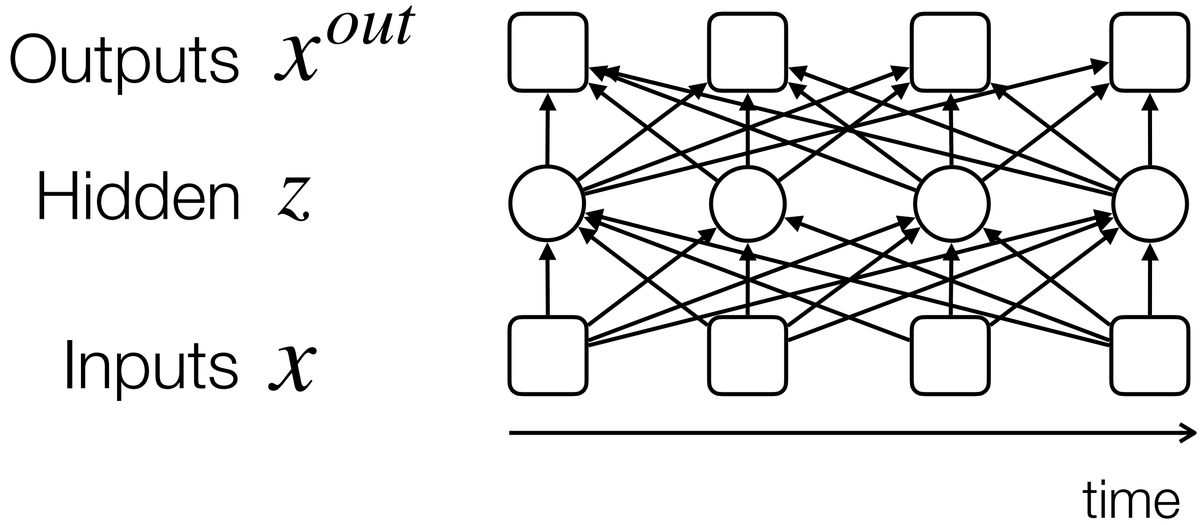
Recurrent Neural Networks (RNNs)



$$z_t = f(z_{t-1}, x_t)$$

$$x_t^{out} = g(z_t)$$

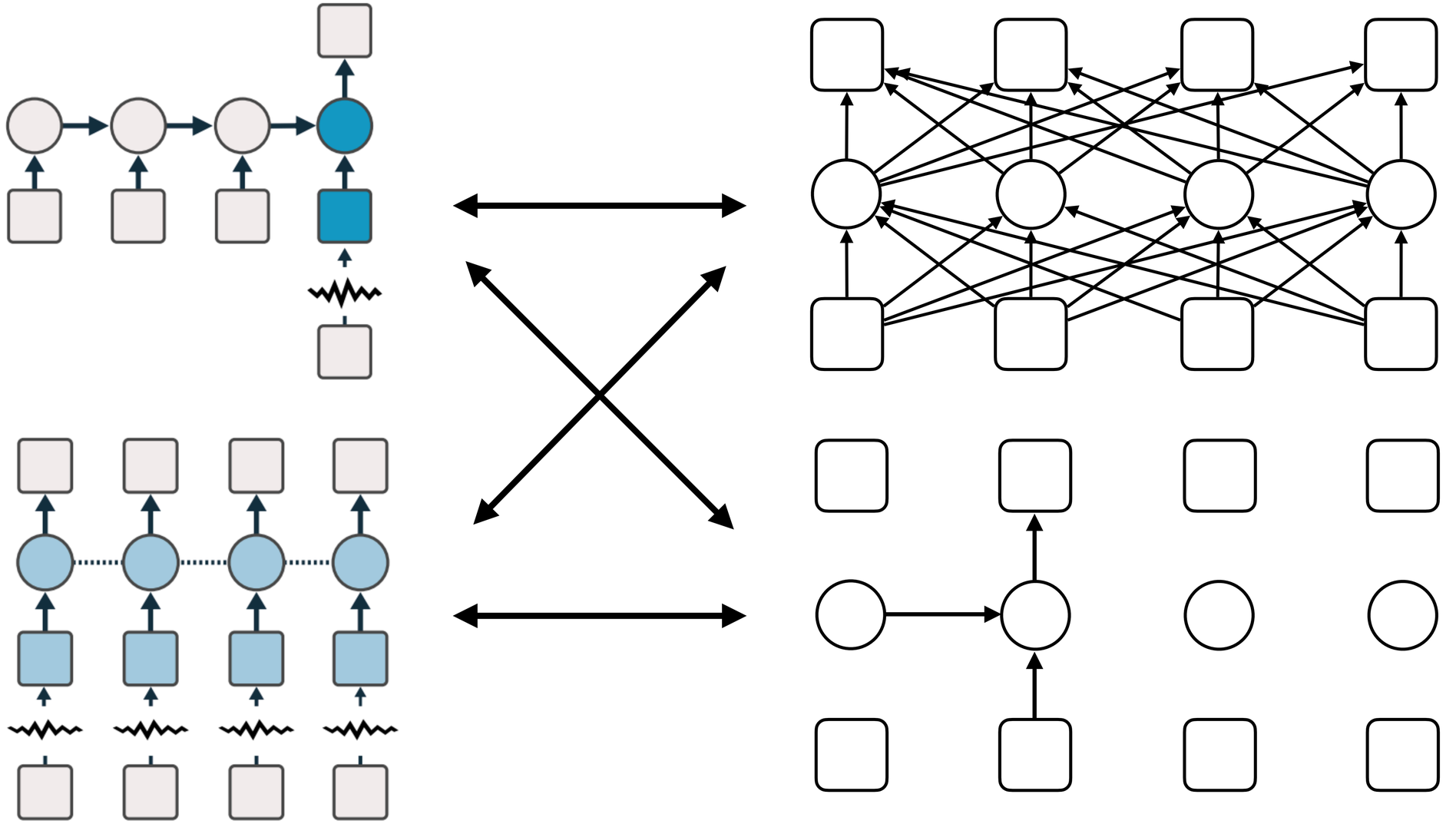
Transformers



Insight: Either sampling scheme can be done by either architecture

Next-Token

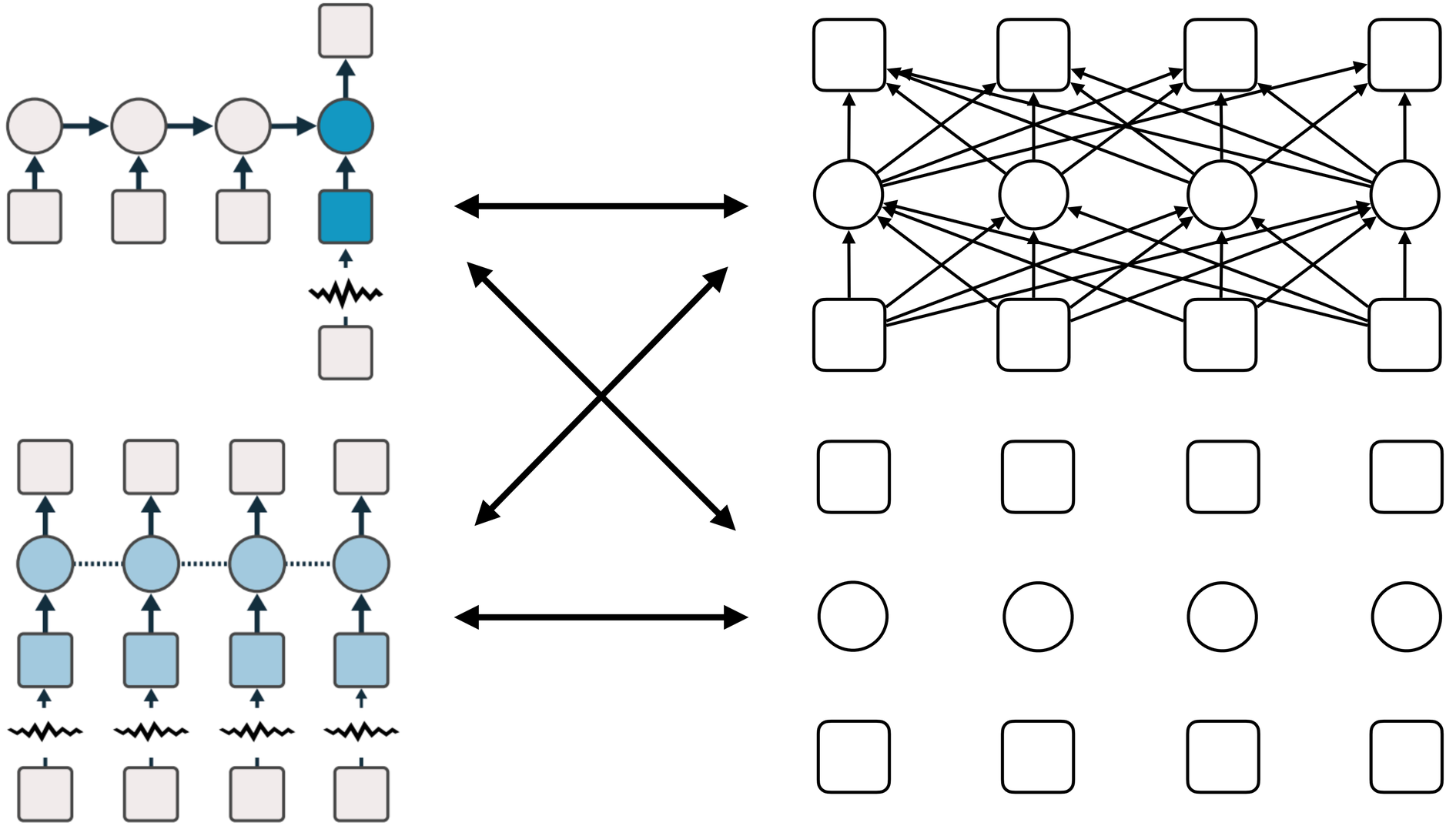
Full-Seq. Diffusion



Insight: Either sampling scheme can be done by either architecture

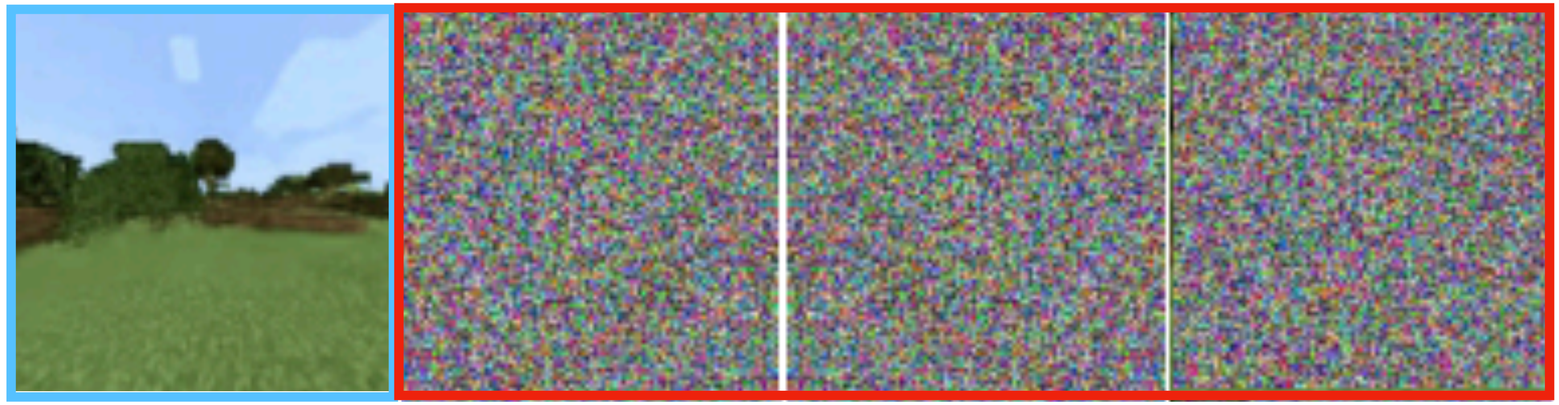
Next-Token

Full-Seq. Diffusion

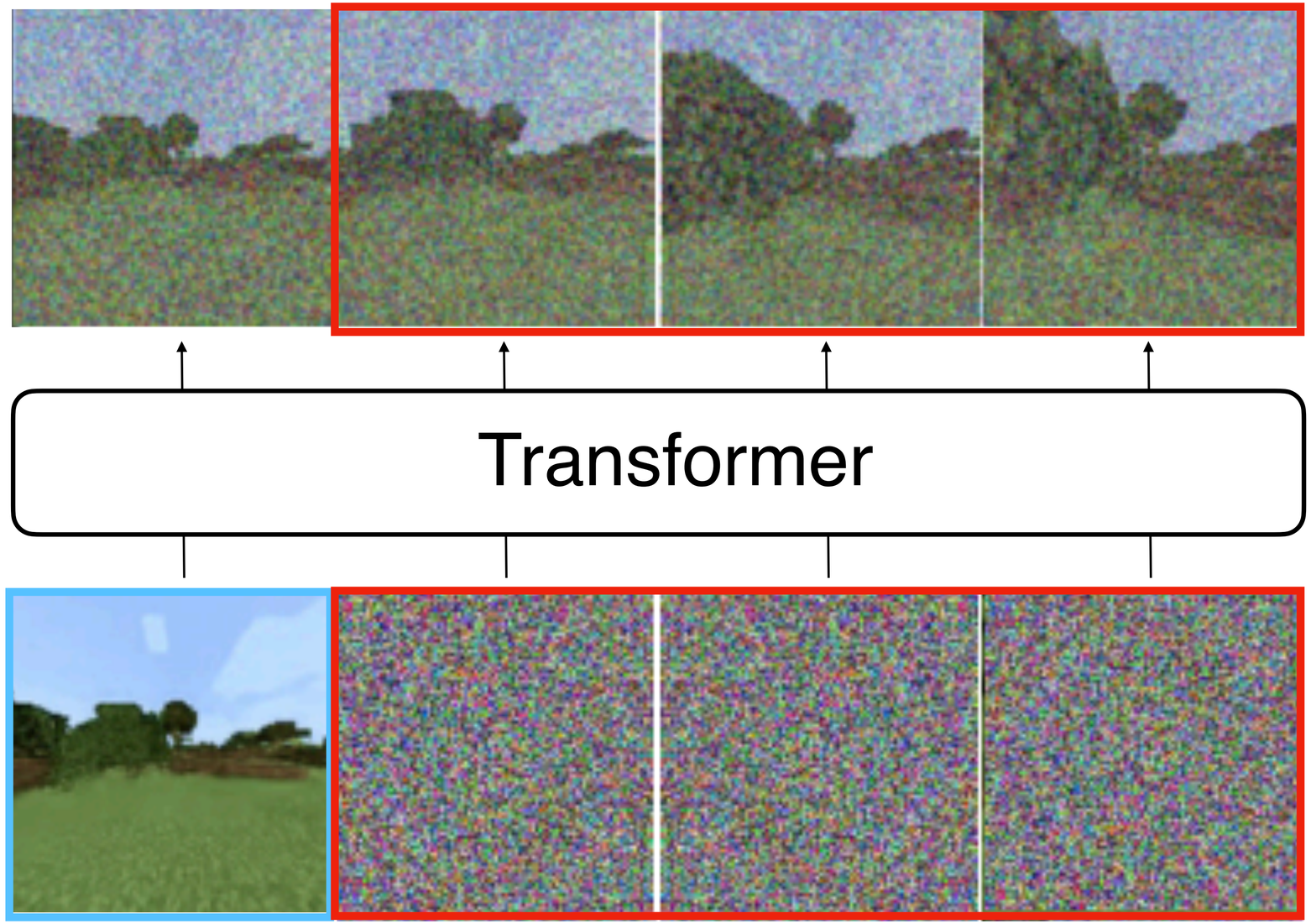


Example: Transformer Implements Auto-Regressive Sampling

Transformer

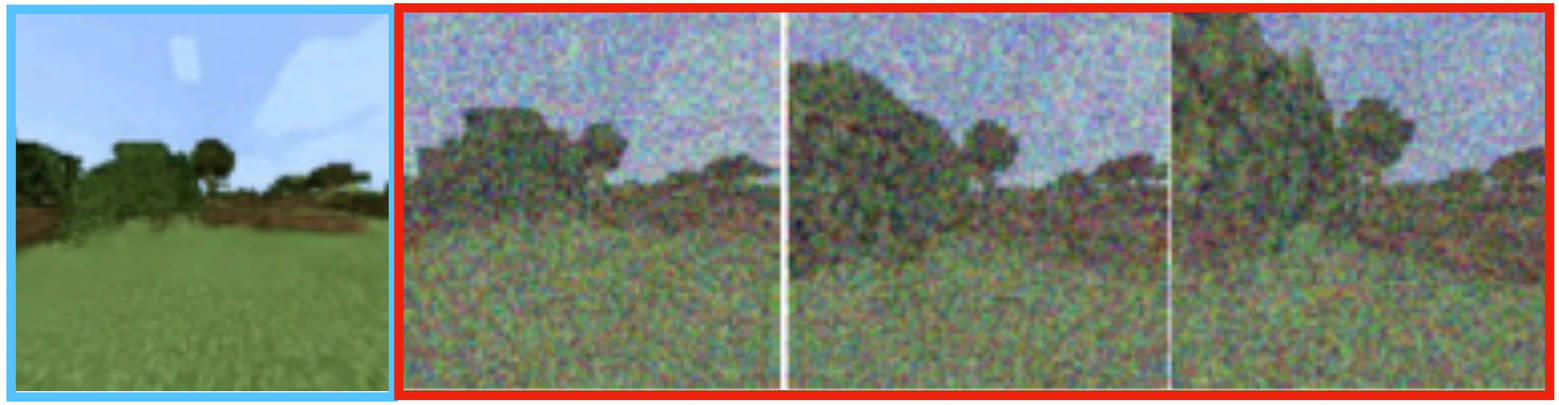


Example: Transformer Implements Auto-Regressive Sampling



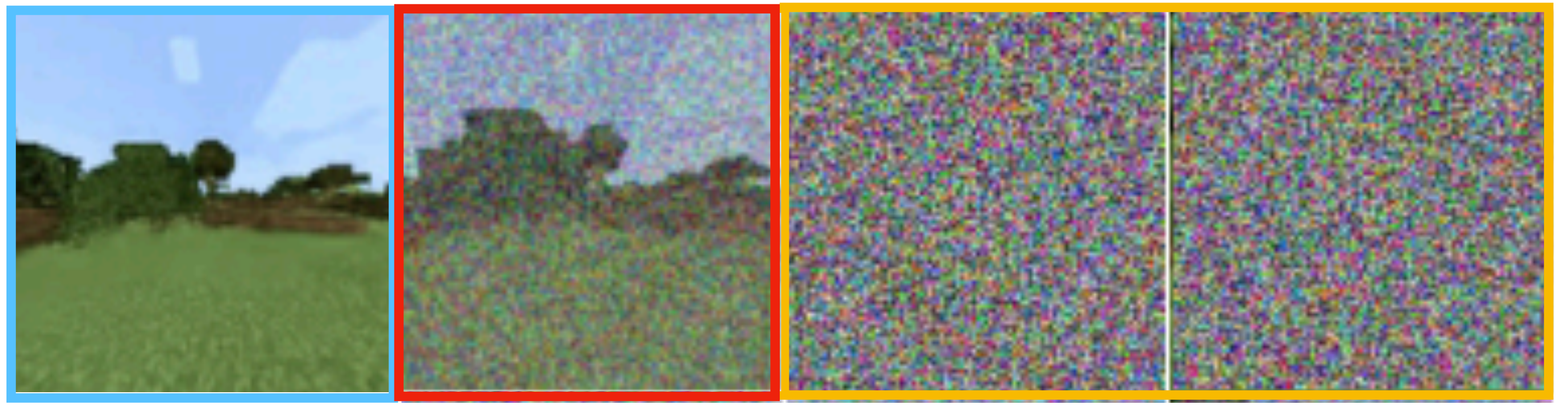
Example: Transformer Implements Auto-Regressive Sampling

Transformer



Example: Transformer Implements Auto-Regressive Sampling

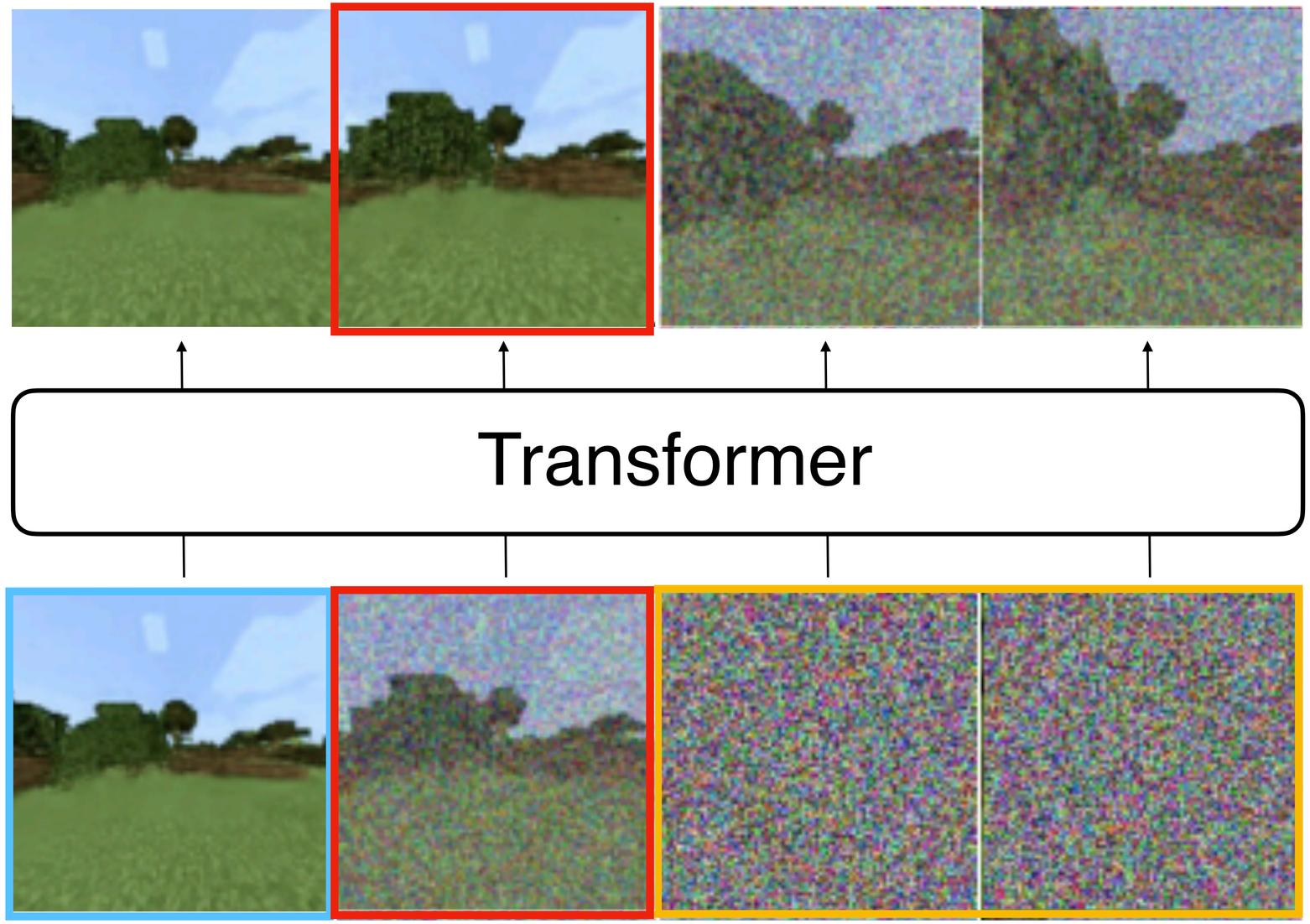
Transformer



We *mask out* future frames by keeping them noisy throughout diffusion process!

We are *still* sampling from $p(x_2 \mid x_1)$!

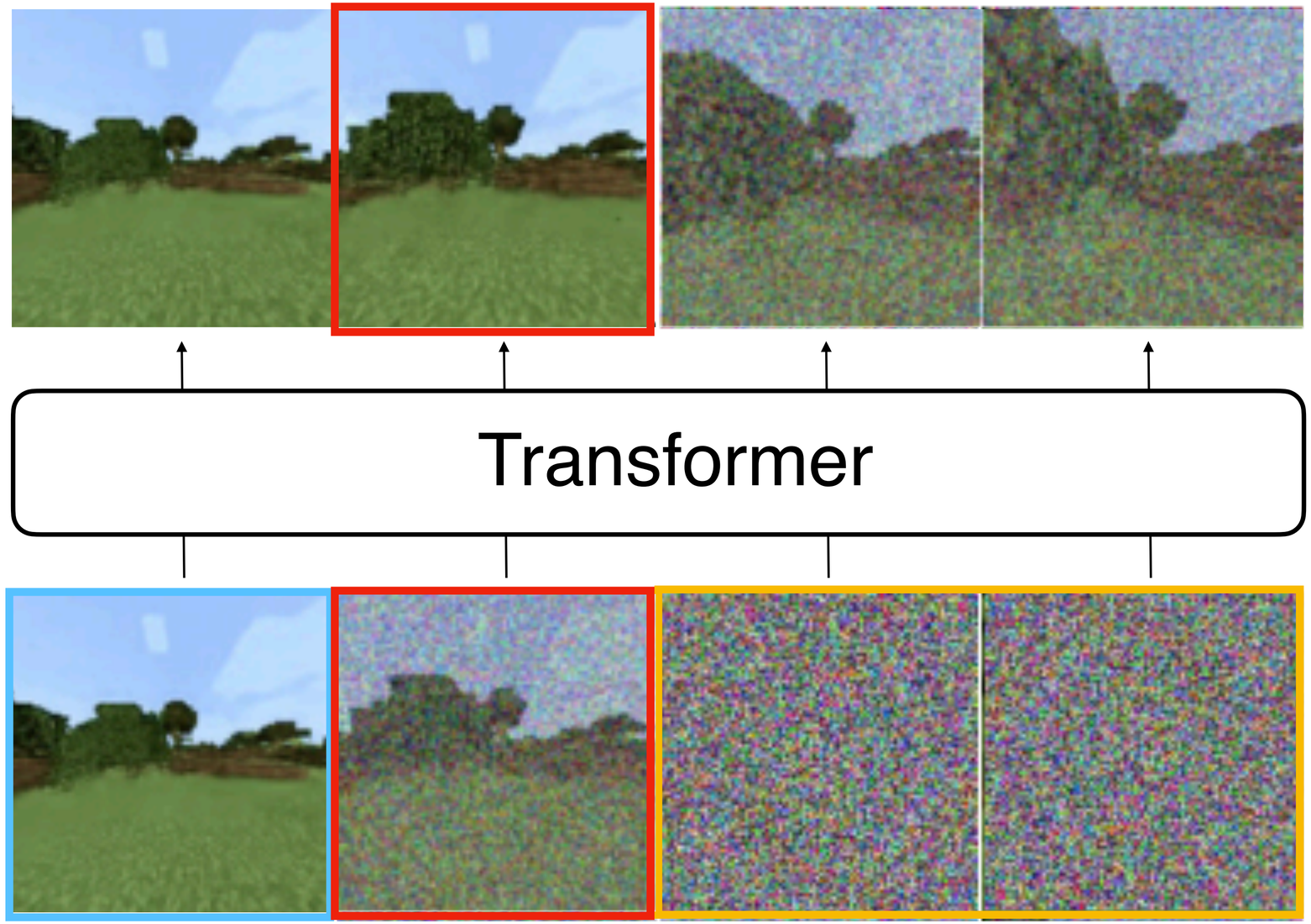
Example: Transformer Implements Auto-Regressive Sampling



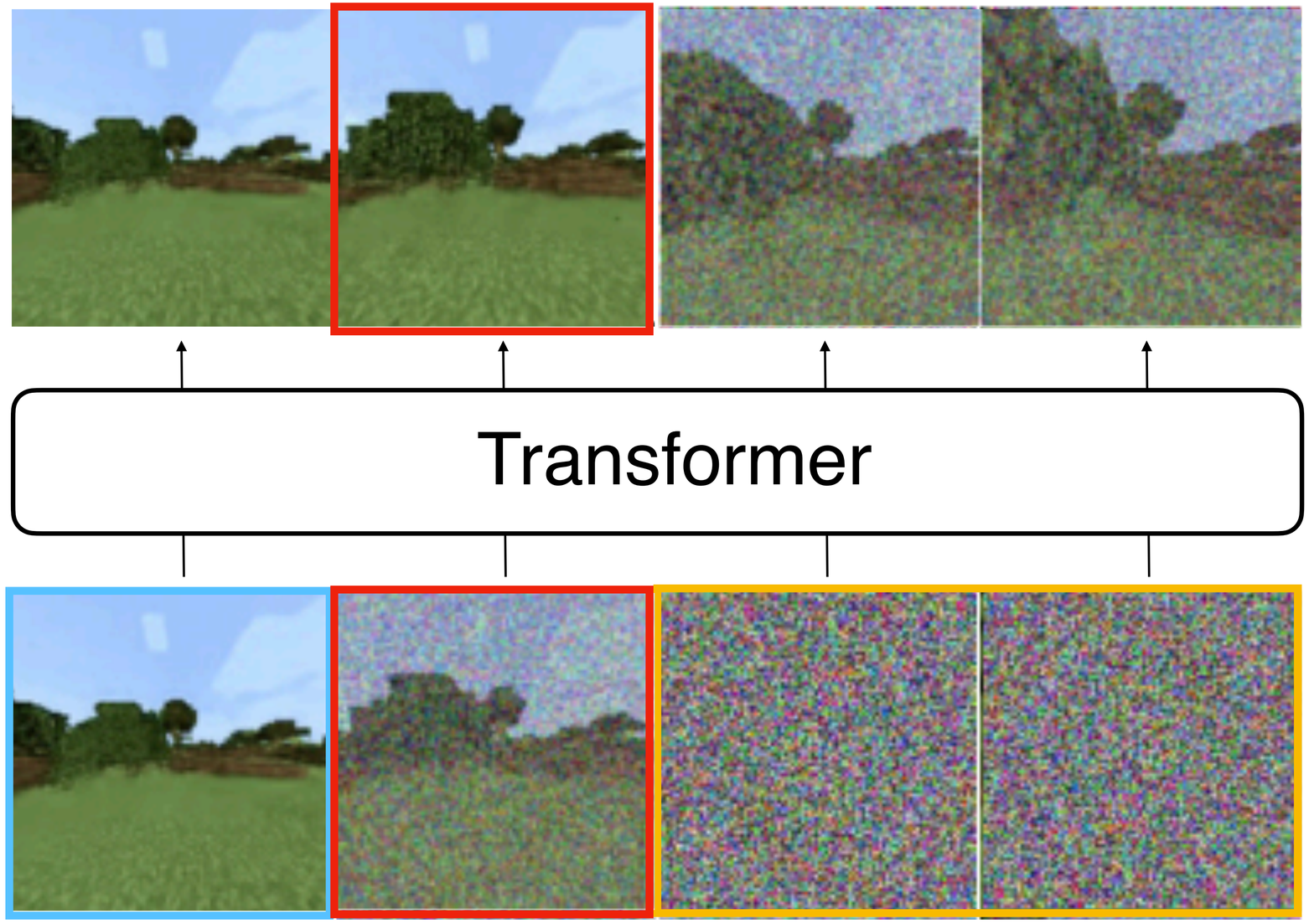
We *mask out* future frames by keeping them noisy throughout diffusion process!

We are *still* sampling from $p(x_2 \mid x_1)$!

Example: Transformer Implements Auto-Regressive Sampling

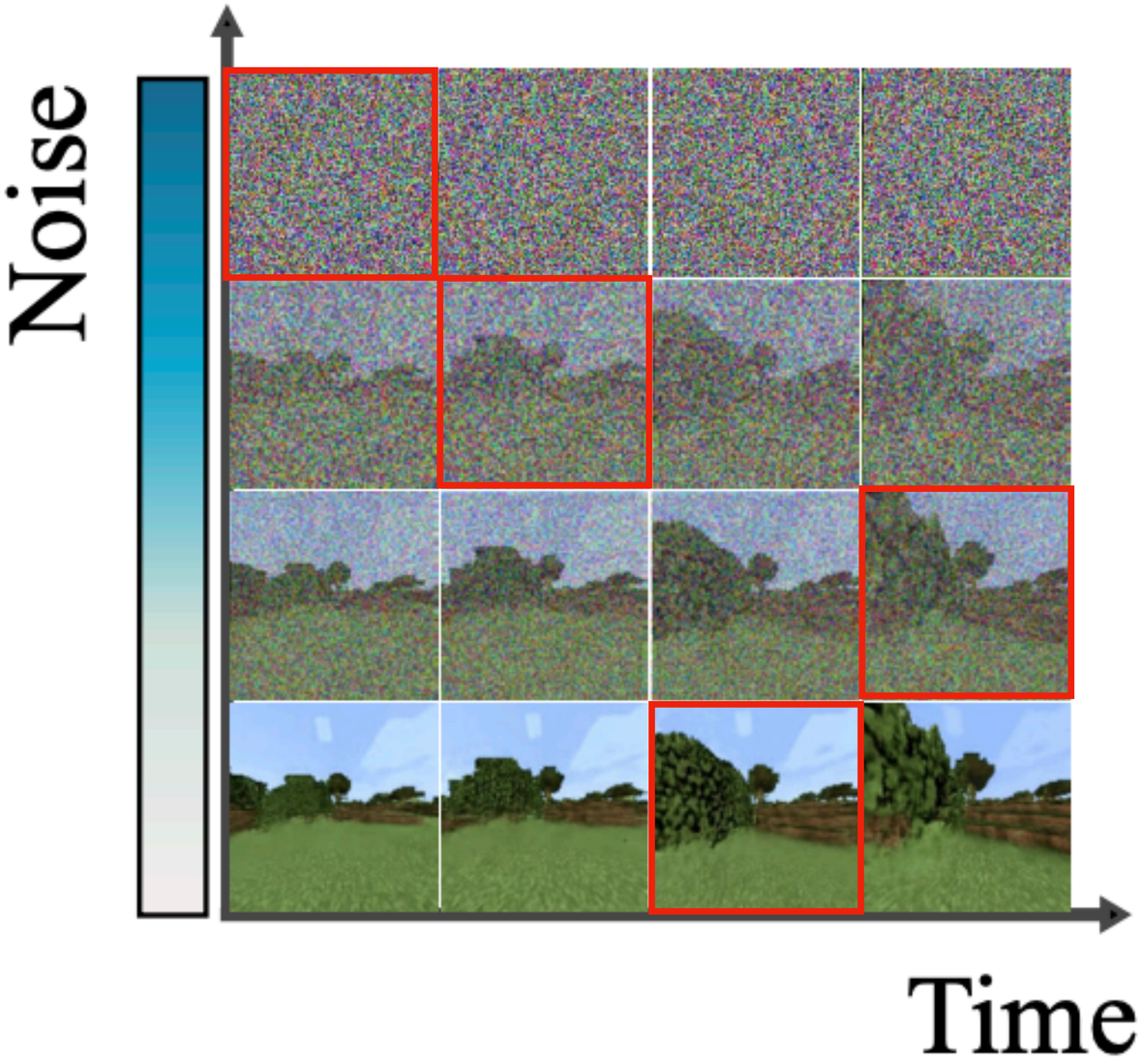


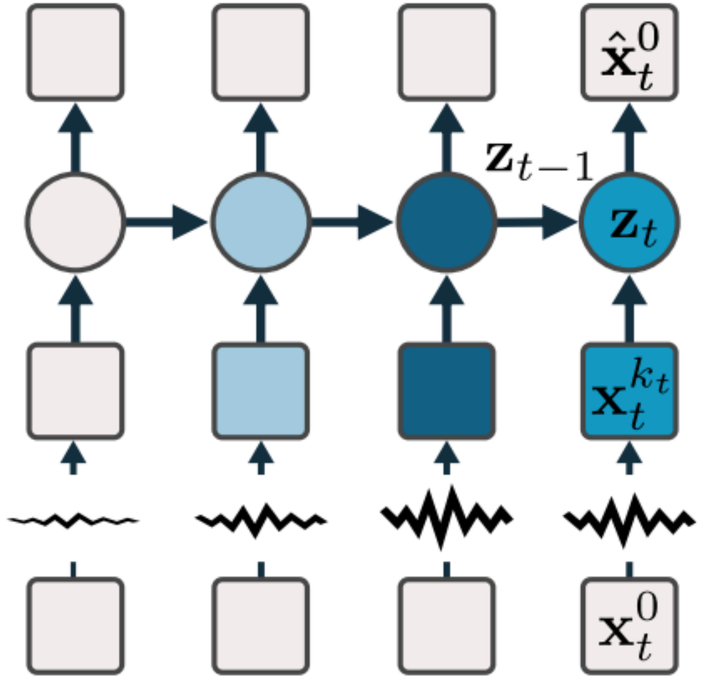
Example: Transformer Implements Auto-Regressive Sampling



However, as before, this combination of noise levels is OOD for a full-sequence diffusion model…

Diffusion Forcing

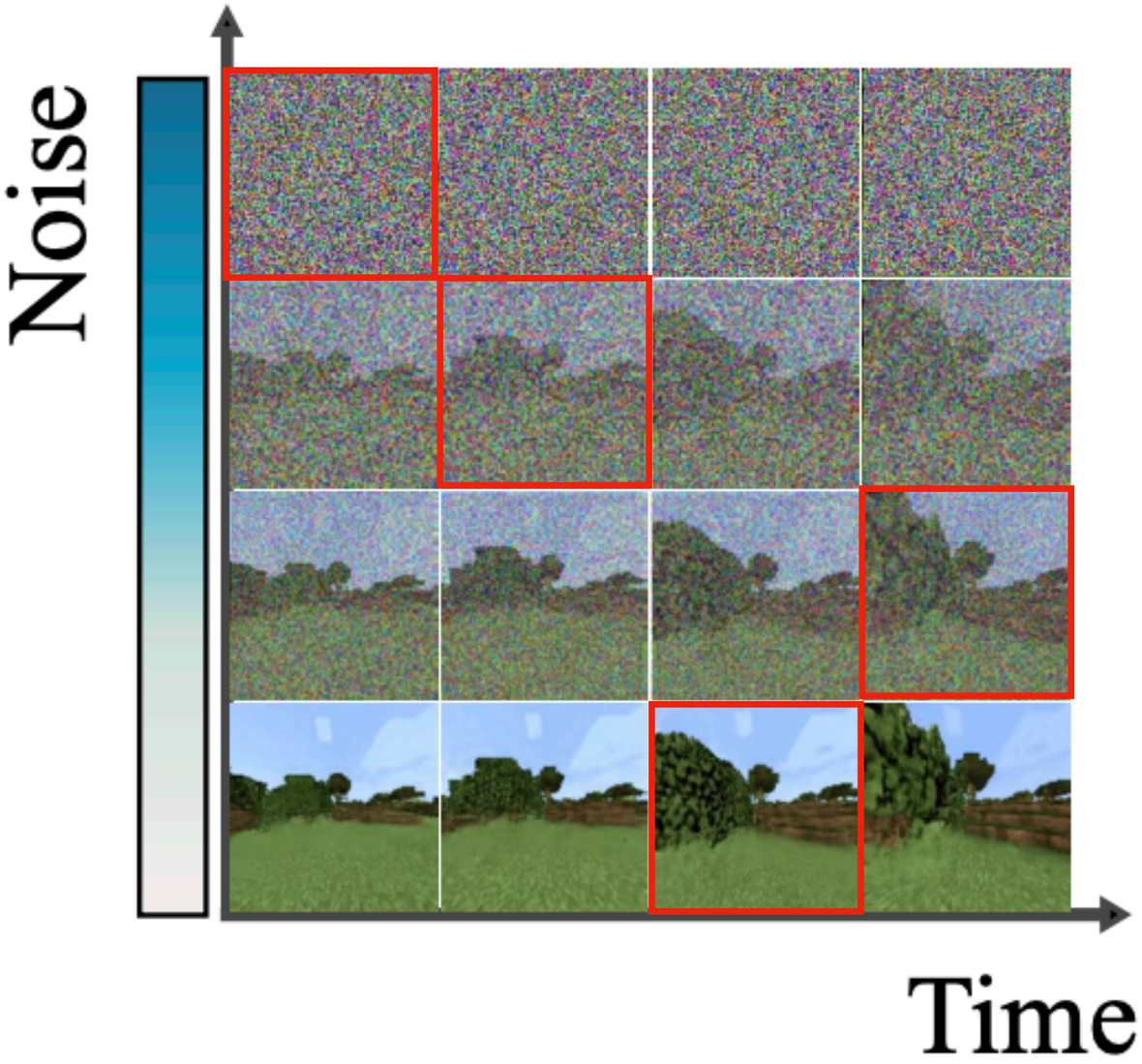




Key Idea: Train the model to denoise frames with *random combinations of noise levels!*

[Chen et al. 2024, Song and Chen et al. 2024]

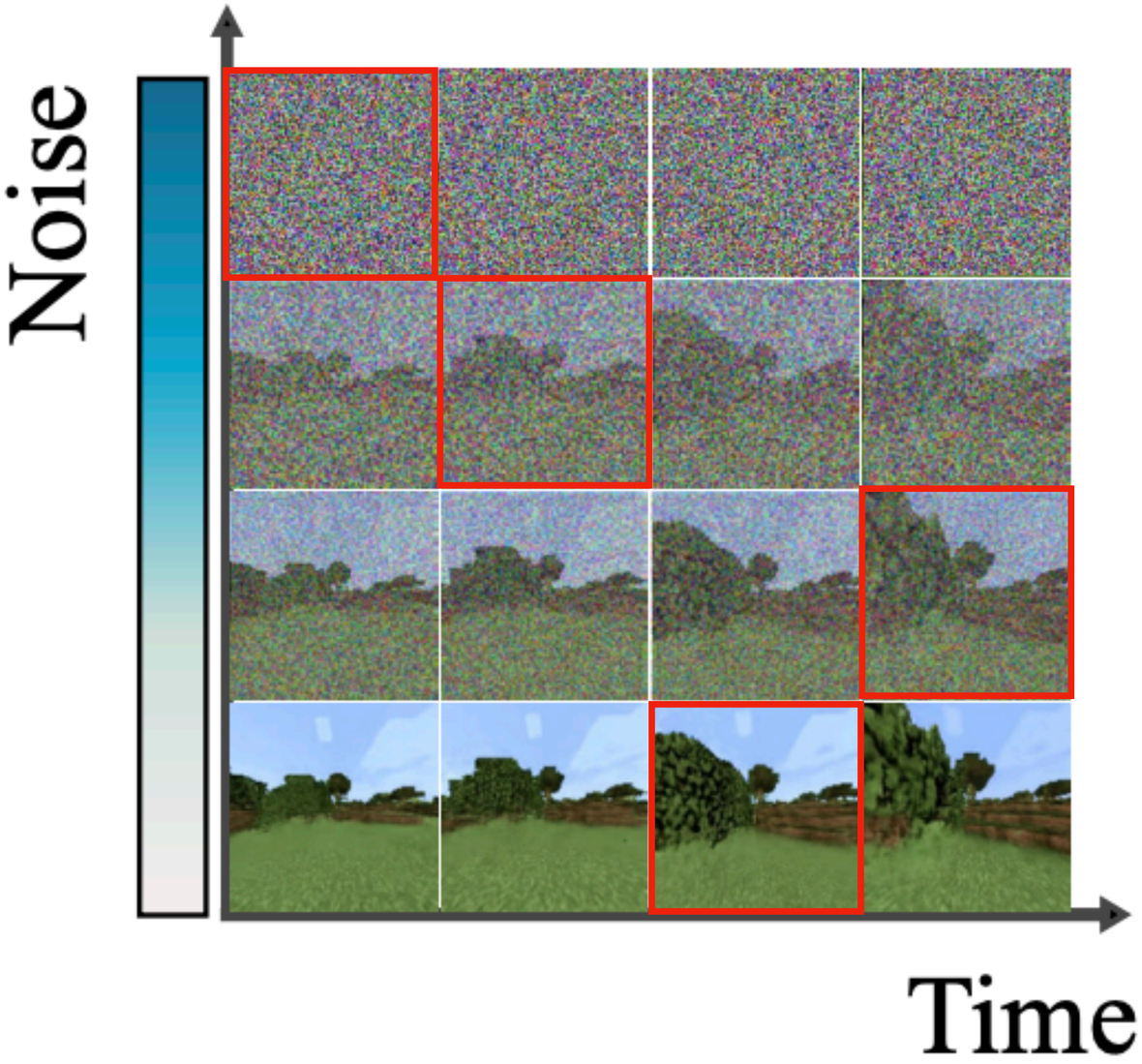
Diffusion Forcing



Key Idea: Train the model to denoise frames with *random combinations of noise levels!*

[Chen et al. 2024, Song and Chen et al. 2024]

Diffusion Forcing

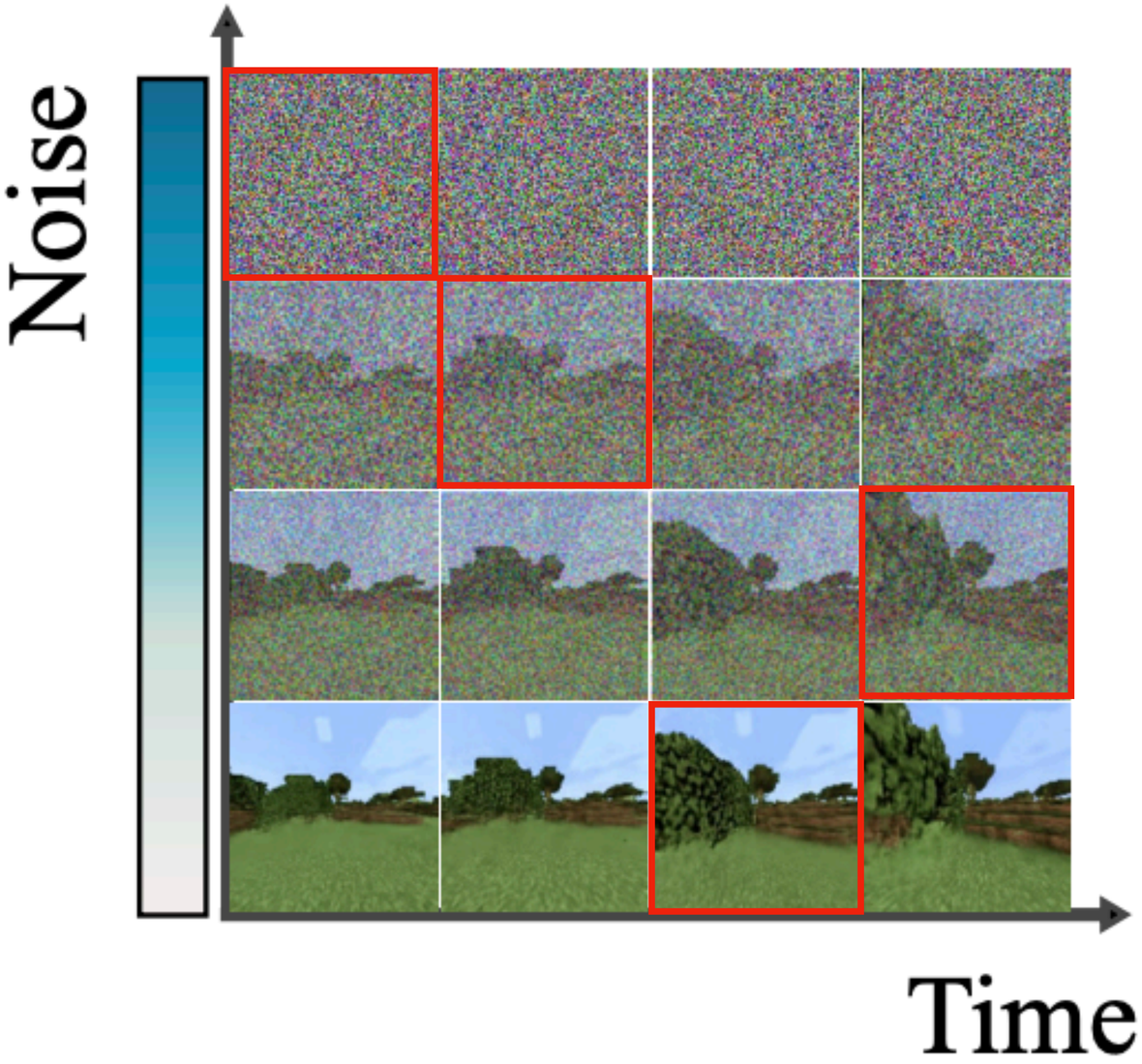


Key Idea: Train the model to denoise frames with *random combinations of noise levels!*

Over time, the model sees *every possible combination of noise levels*

[Chen et al. 2024, Song and Chen et al. 2024]

Diffusion Forcing



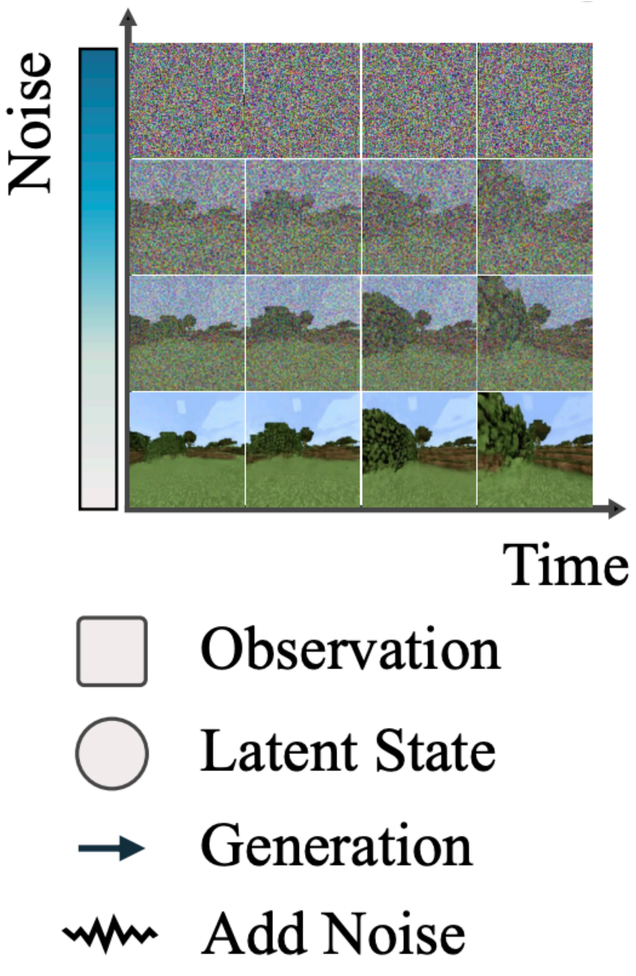
Key Idea: Train the model to denoise frames with *random combinations of noise levels!*

Over time, the model sees *every possible combination of noise levels*

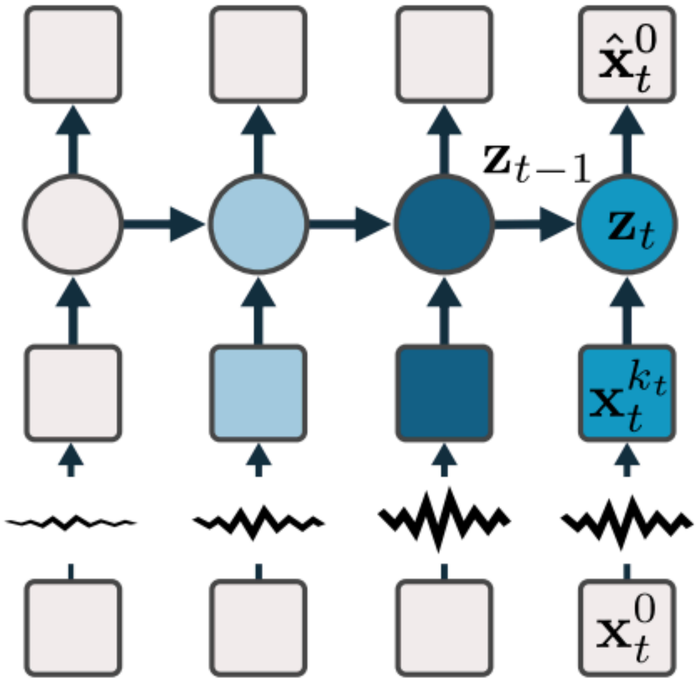
This generalizes both auto-regressive sampling and full-sequence diffusion

[Chen et al. 2024, Song and Chen et al. 2024]

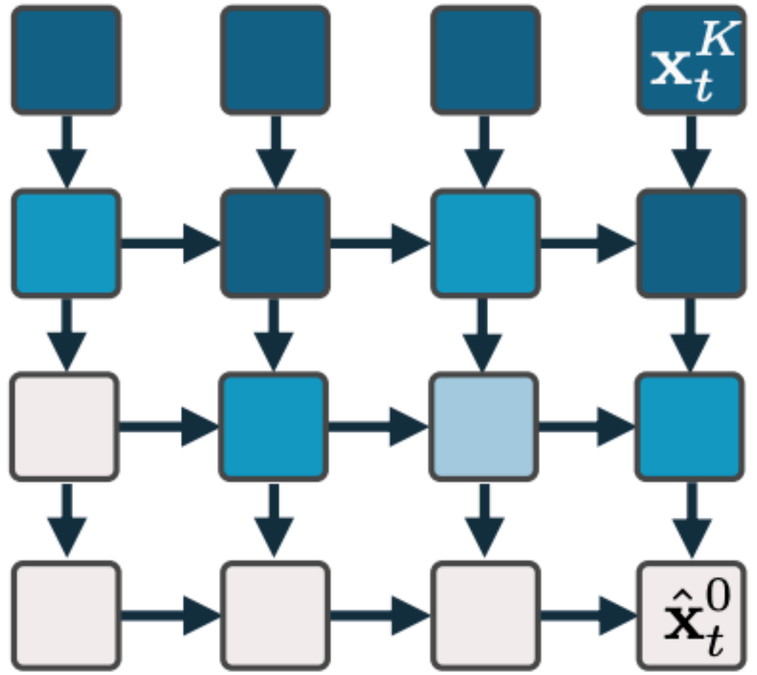
Diffusion Forcing



Training

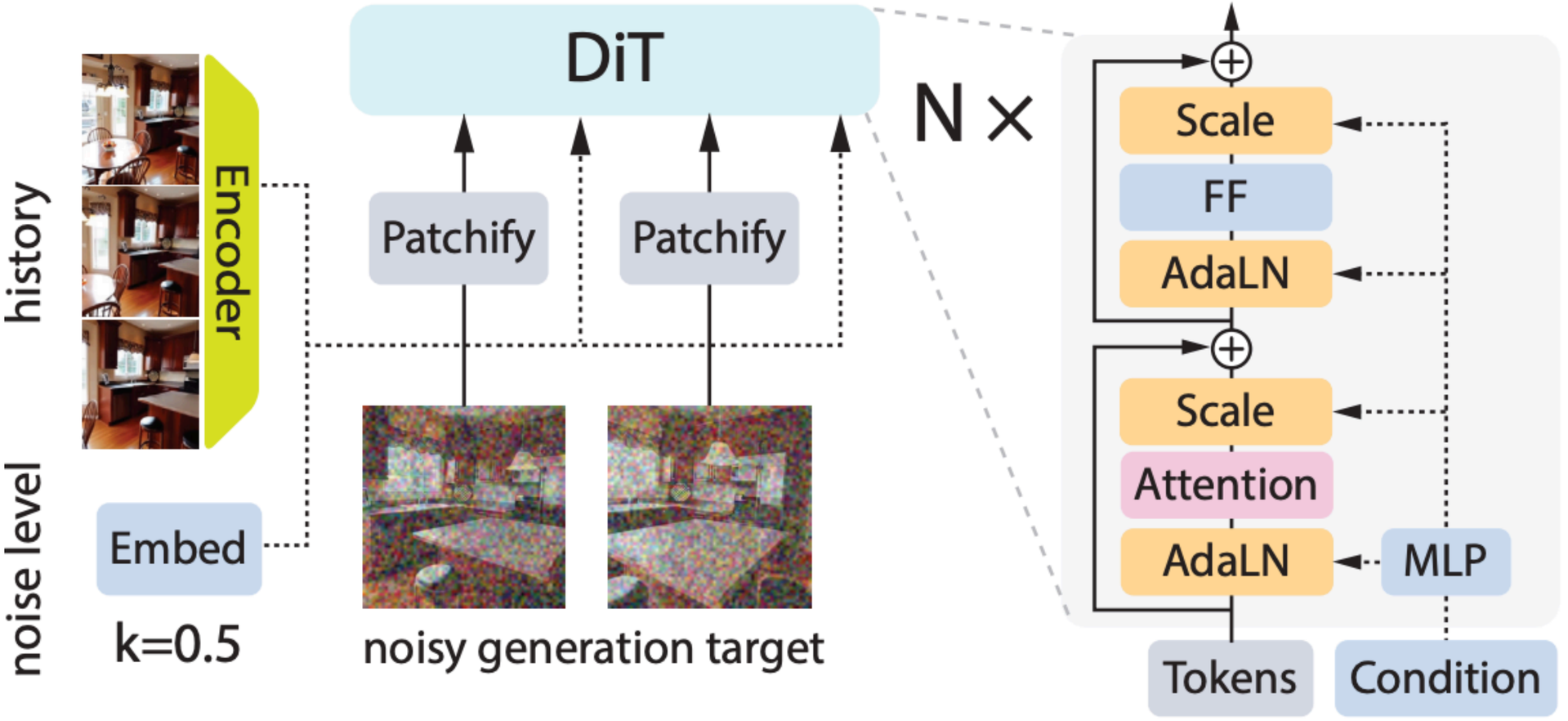


Prediction

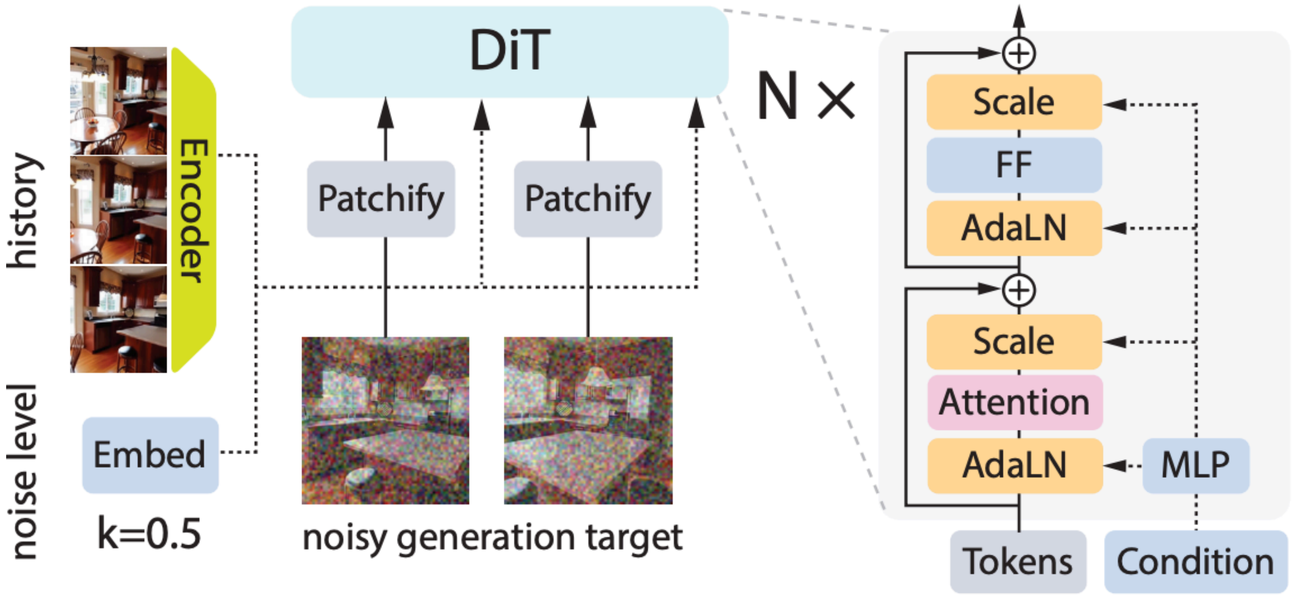


Note: Can again be trained with *either* transformers or RNNs…

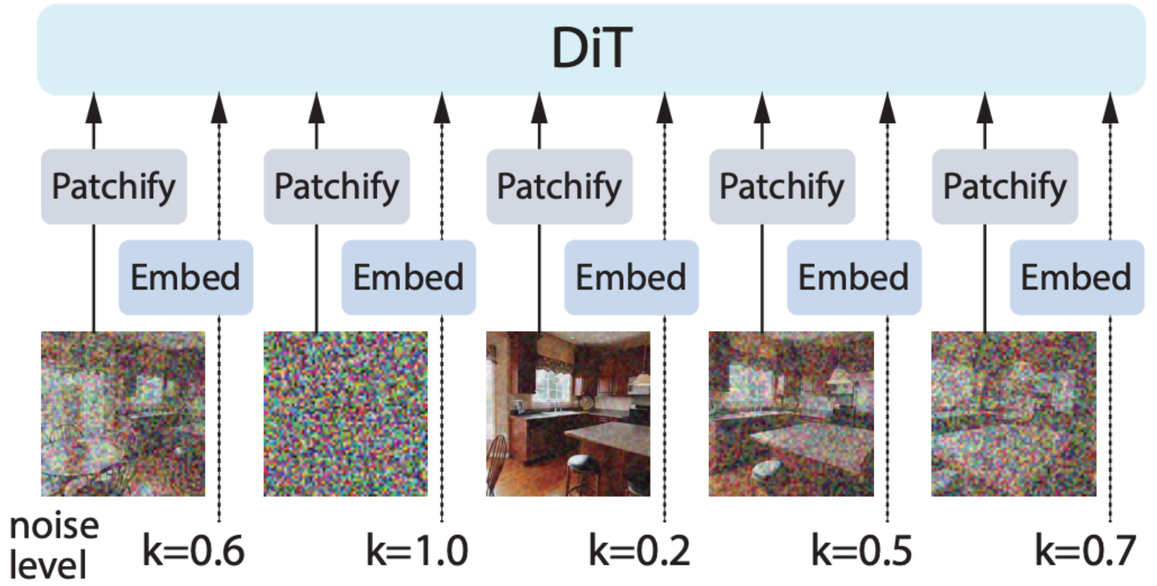
Before: Conditioning Frames Fed in Separately



Now: Single “Diffusion Forcing Transformer” (DFoT)

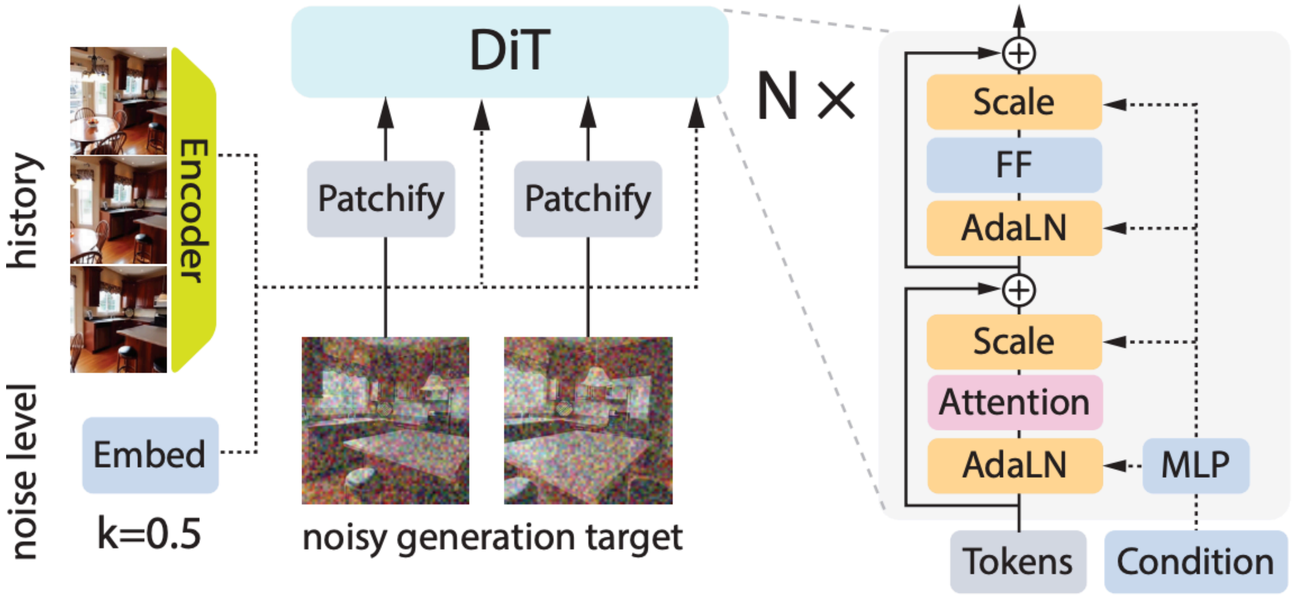


(a) Conventional Video Diffusion

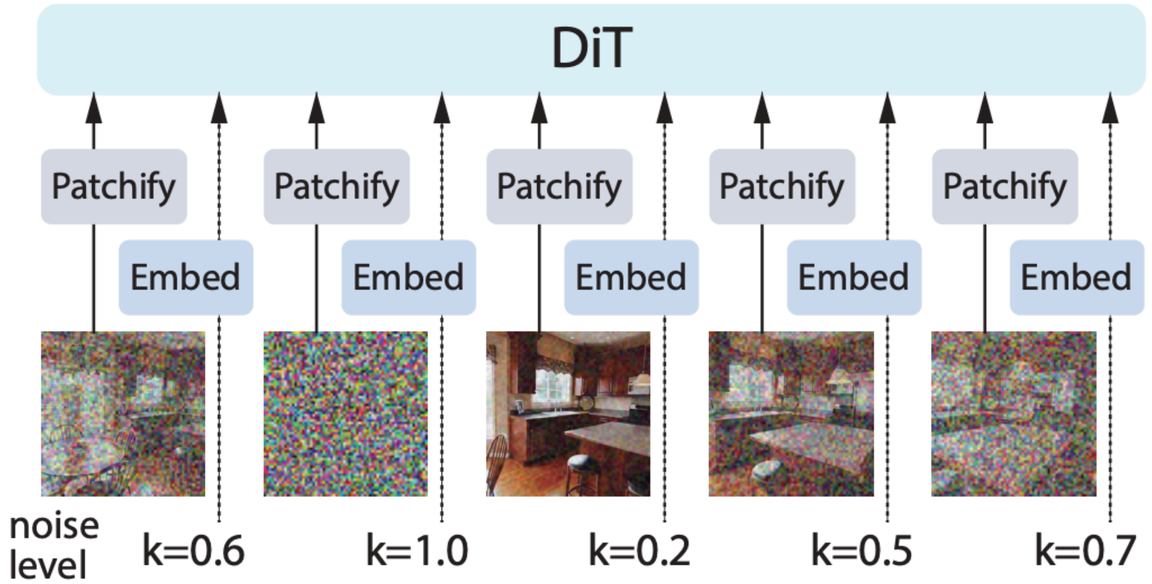


(b) Diffusion Forcing Transformer

Now: Single “Diffusion Forcing Transformer” (DFoT)



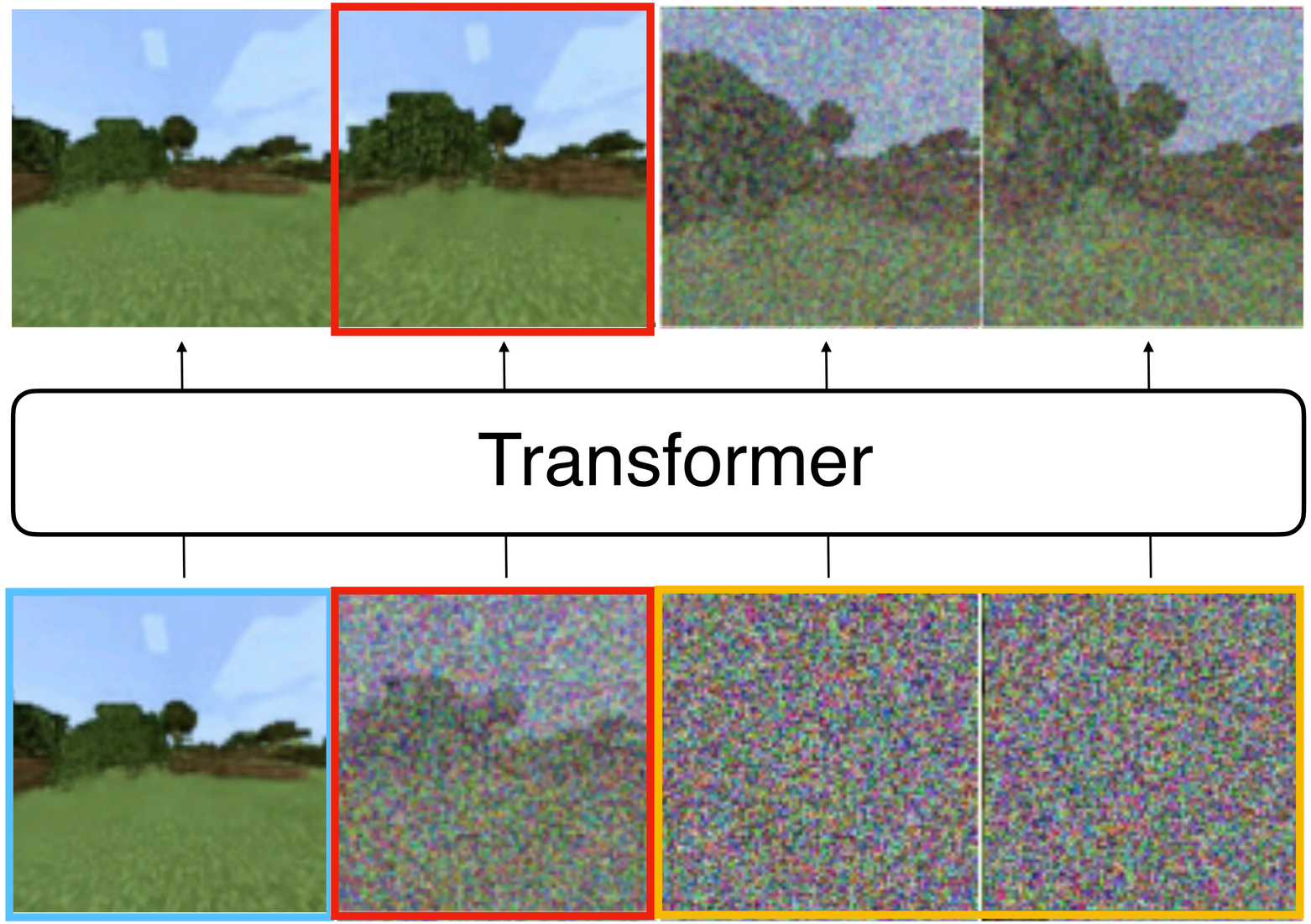
(a) Conventional Video Diffusion



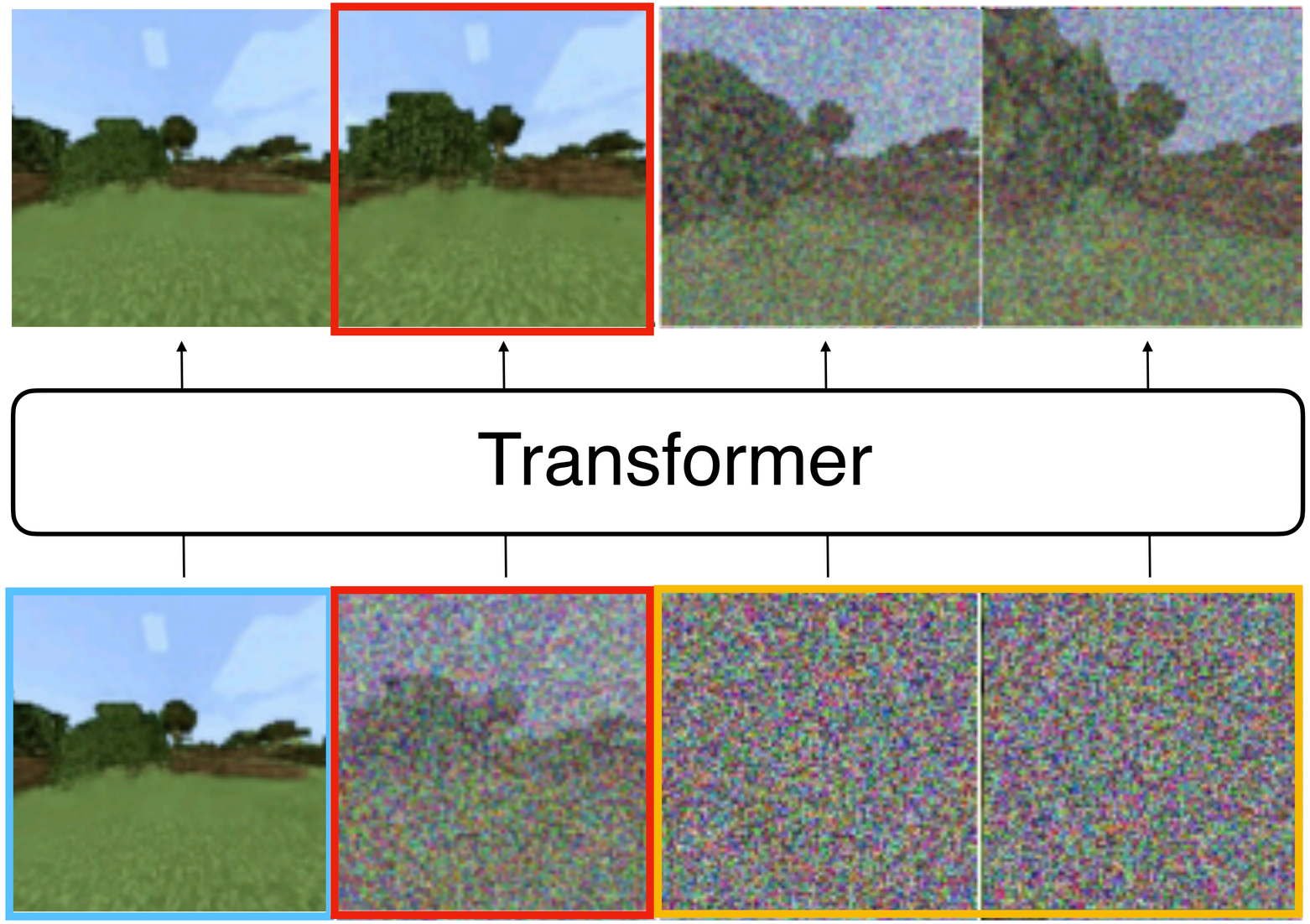
(b) Diffusion Forcing Transformer

Just a transformer where each frame gets different noise levels at training time

Diffusion Forcing enables straight-forward Auto-Regressive Sampling

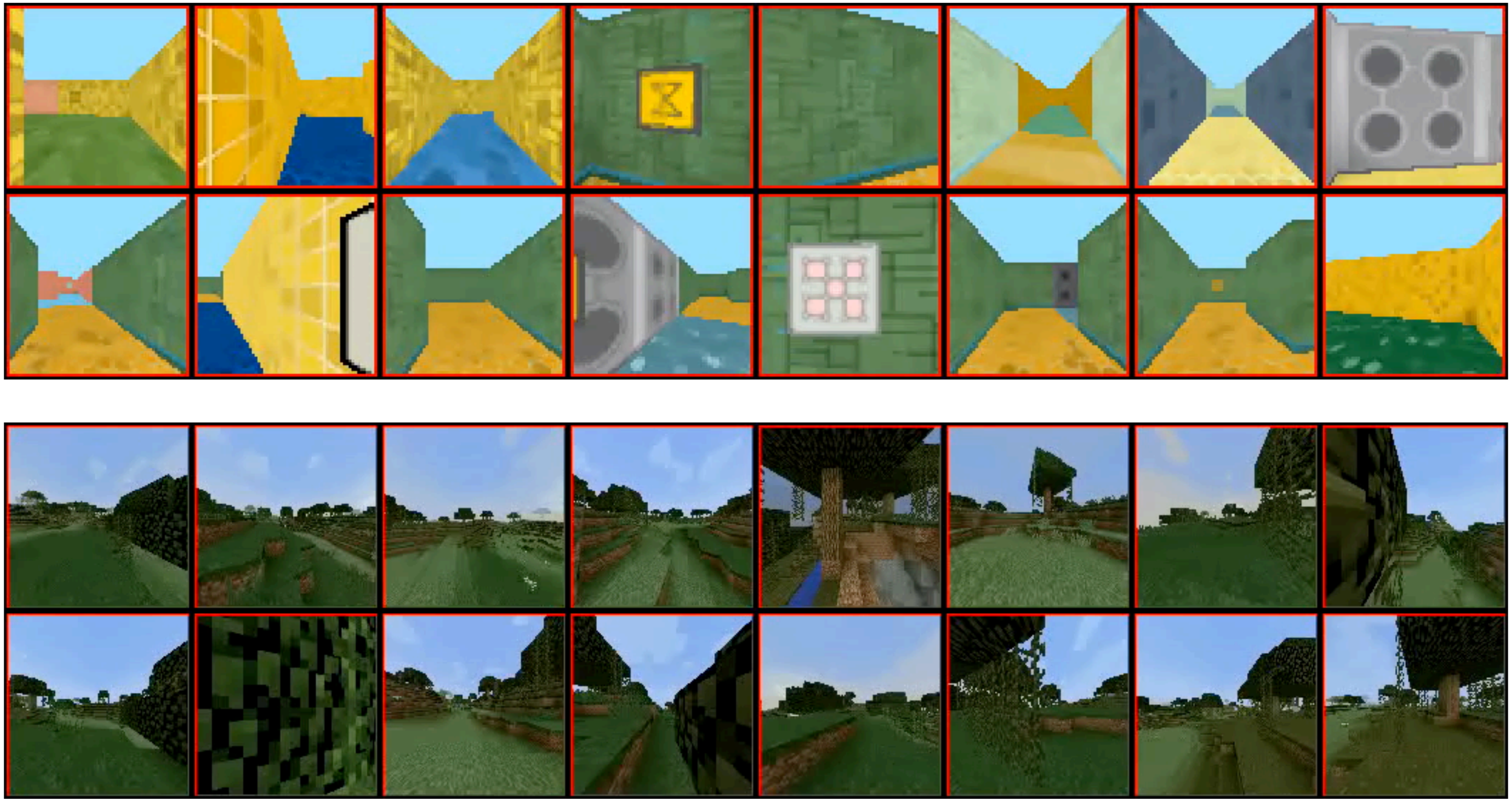


Diffusion Forcing enables straight-forward Auto-Regressive Sampling

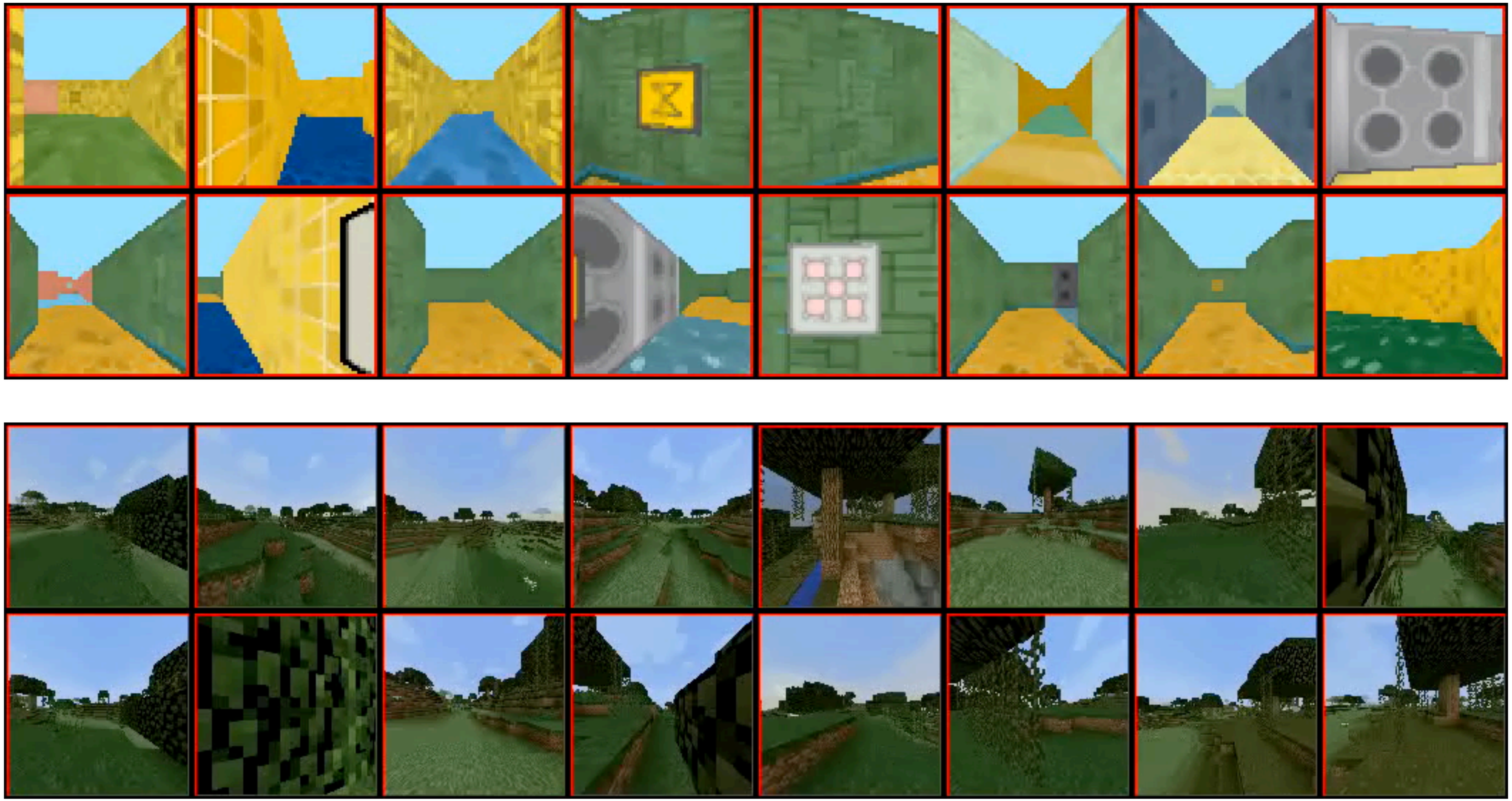


This combination of noise levels is not OOD anymore!

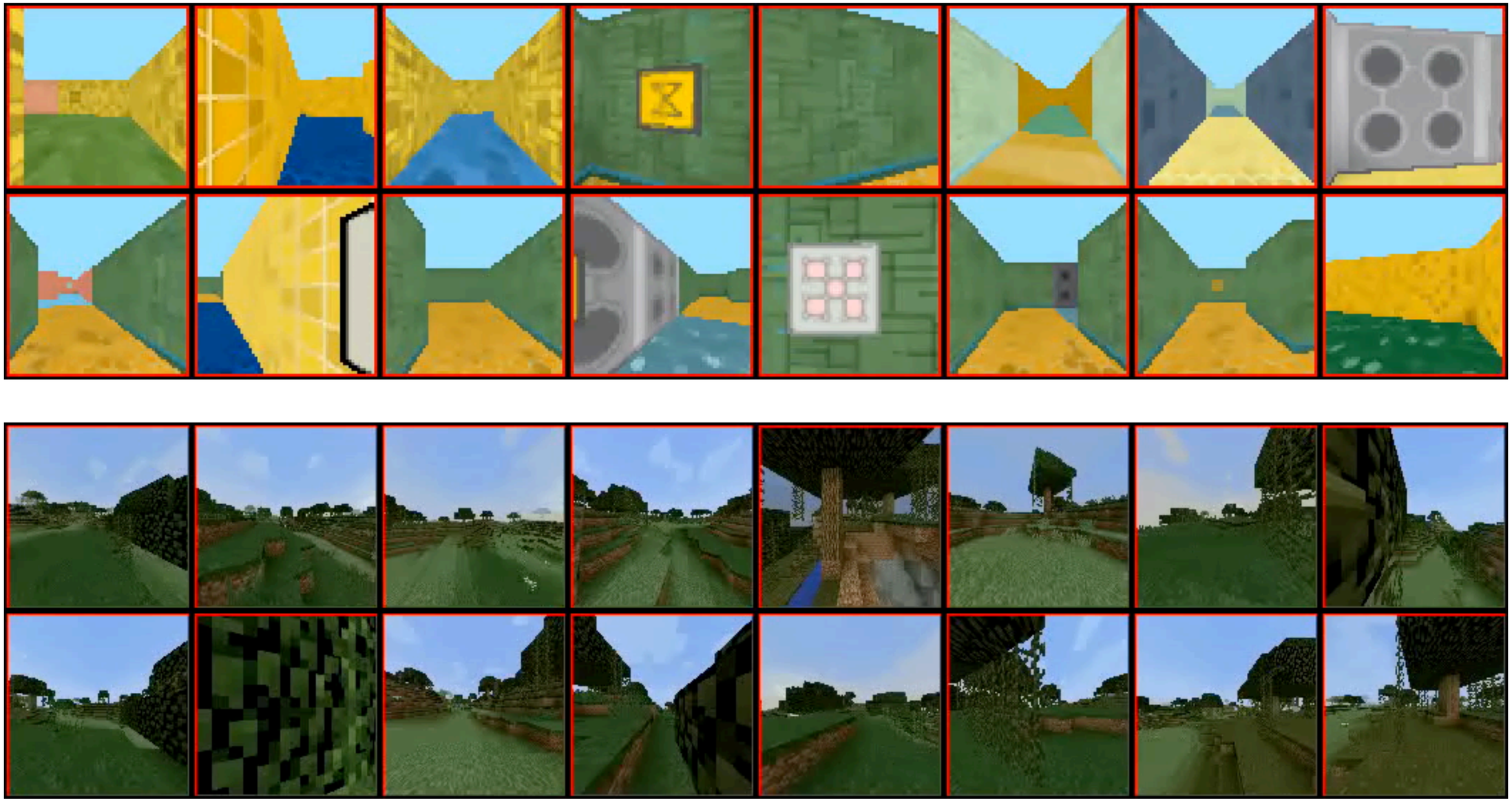
Naive auto-reg rollout diverges: Accumulates Error



Naive auto-reg rollout diverges: Accumulates Error



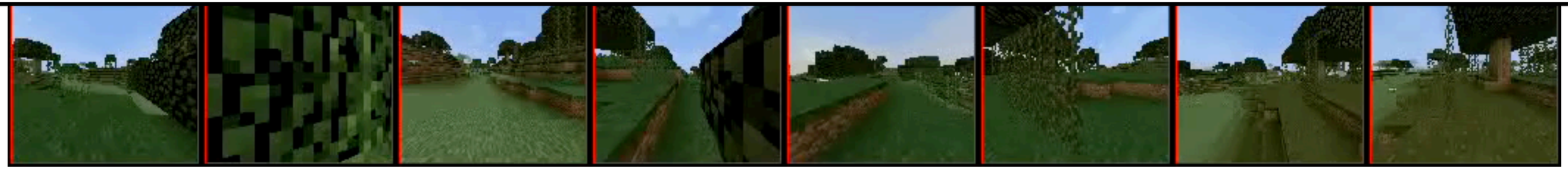
Naive auto-reg rollout diverges: Accumulates Error



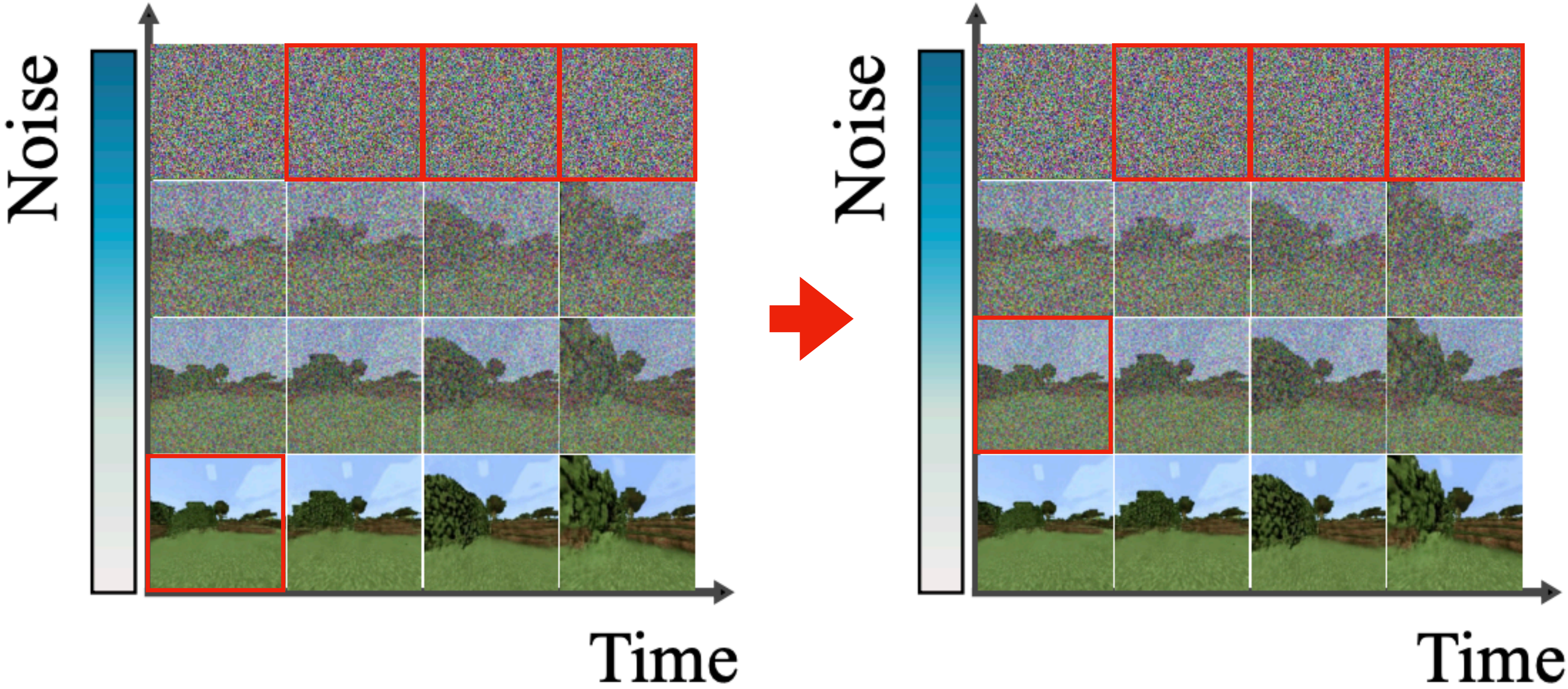
Naive auto-reg rollout diverges: Accumulates Error

Autoregressive Sampling compounds error

Why? Model “trusts” context as GT (b/c that’s how it’s trained) but at sampling time, context is generated, too!



Solution: Feed in *slightly noisy* past



Stable Auto-Regressive Rollout

Rollouts with an RNN

Diffusion Forcing

Next-token prediction

↑
compounding error

Full-Sequence Diffusion
(unconditional model)

↑
consistency issues

Stable Auto-Regressive Rollout

Rollouts with an RNN

Diffusion Forcing

Next-token prediction

↑
compounding error

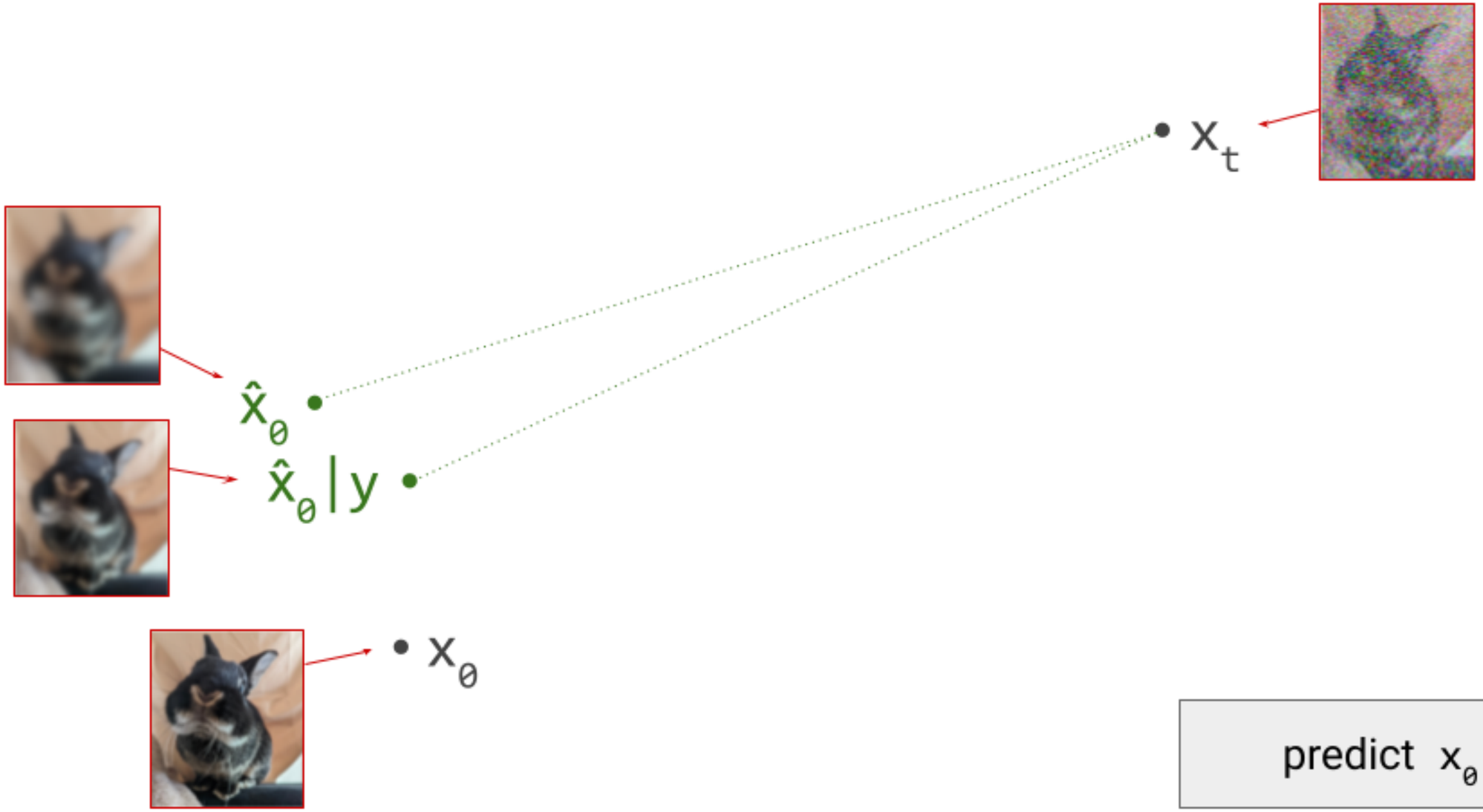
Full-Sequence Diffusion
(unconditional model)

↑
consistency issues

Classifier-Free Guidance

[Ho & Salimans, 2023]

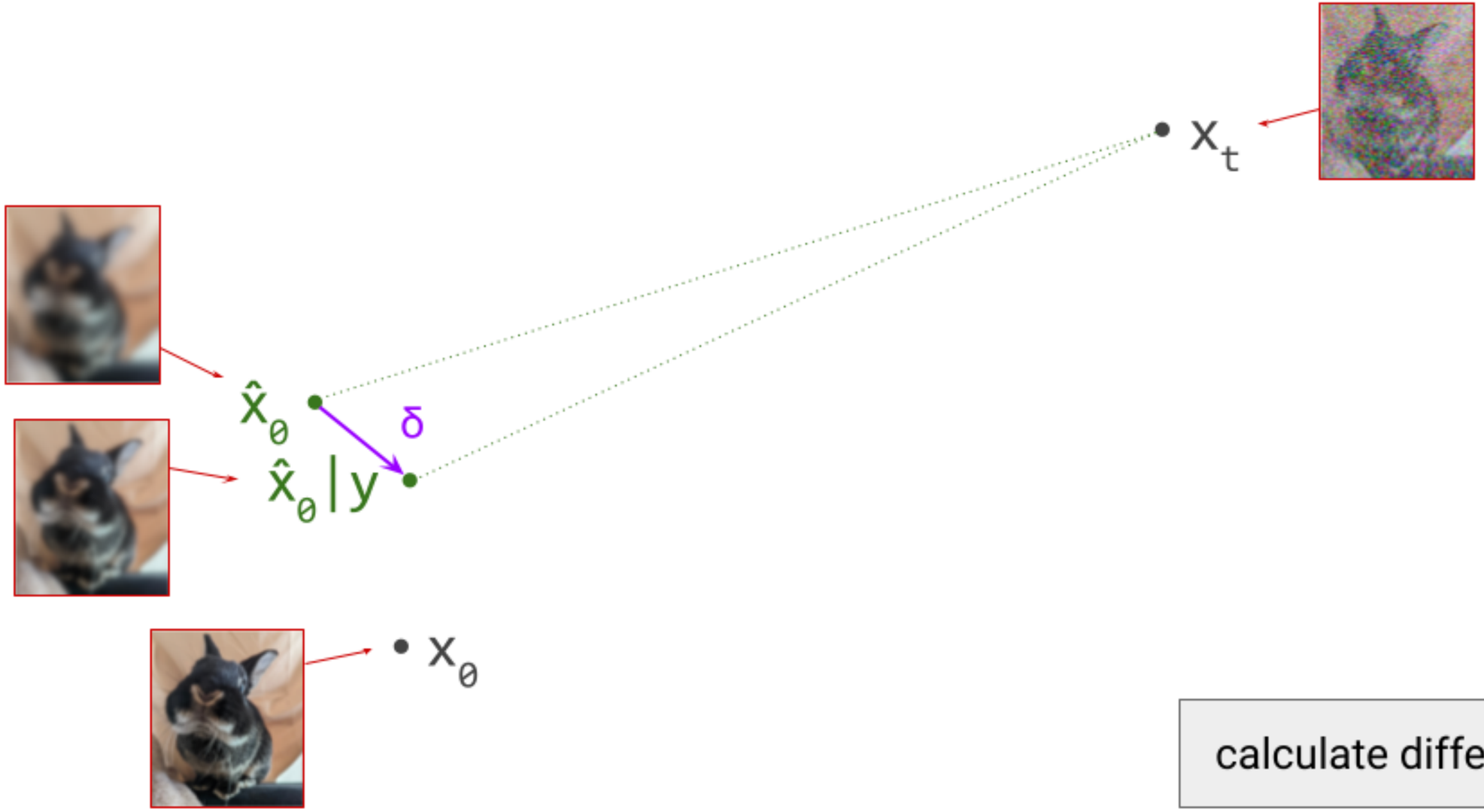
Illustration from Sander Dieleman, “Guidance: A Cheat Code for Diffusion Models”



Classifier-Free Guidance

[Ho & Salimans, 2023]

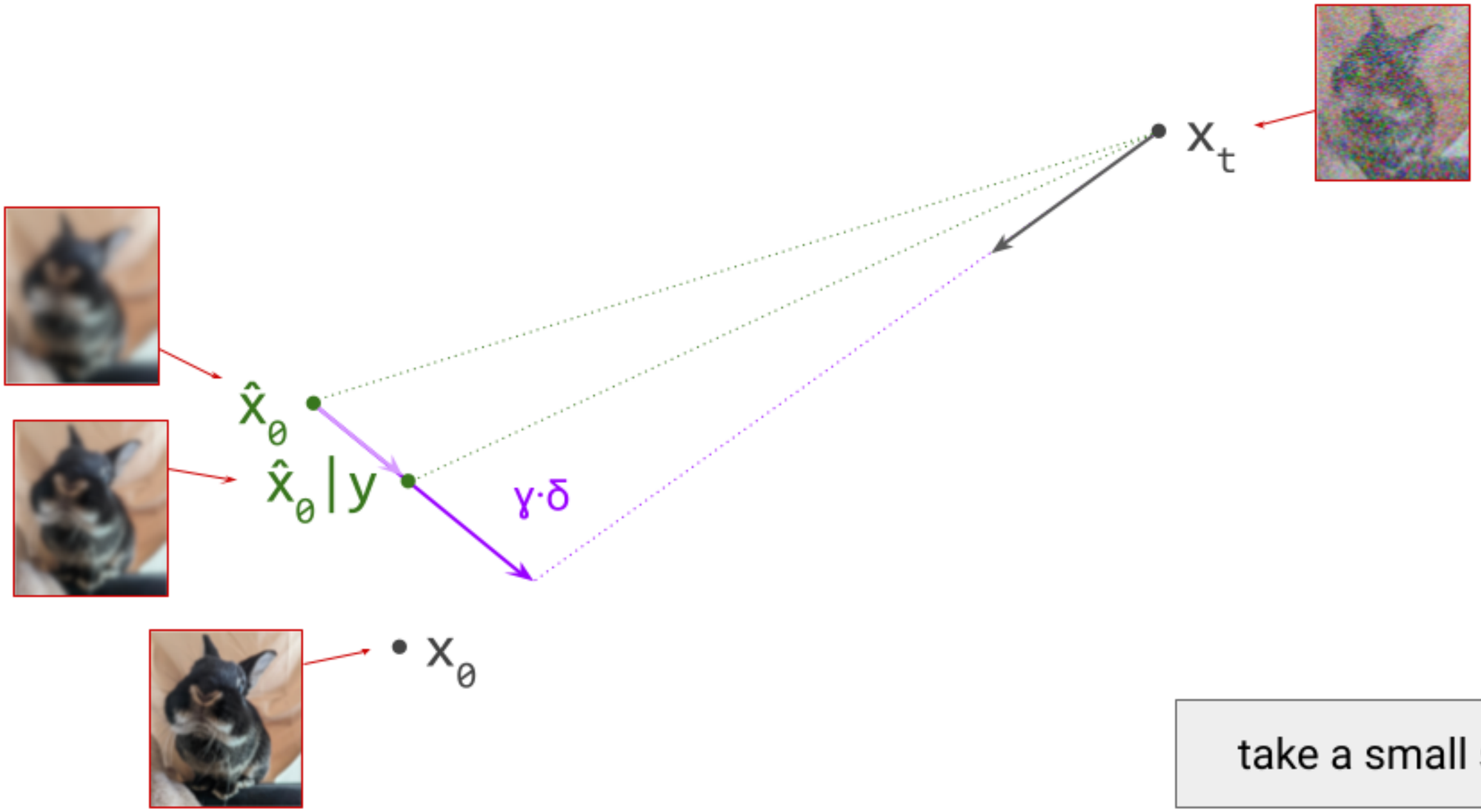
Illustration from Sander Dieleman, “Guidance: A Cheat Code for Diffusion Models”



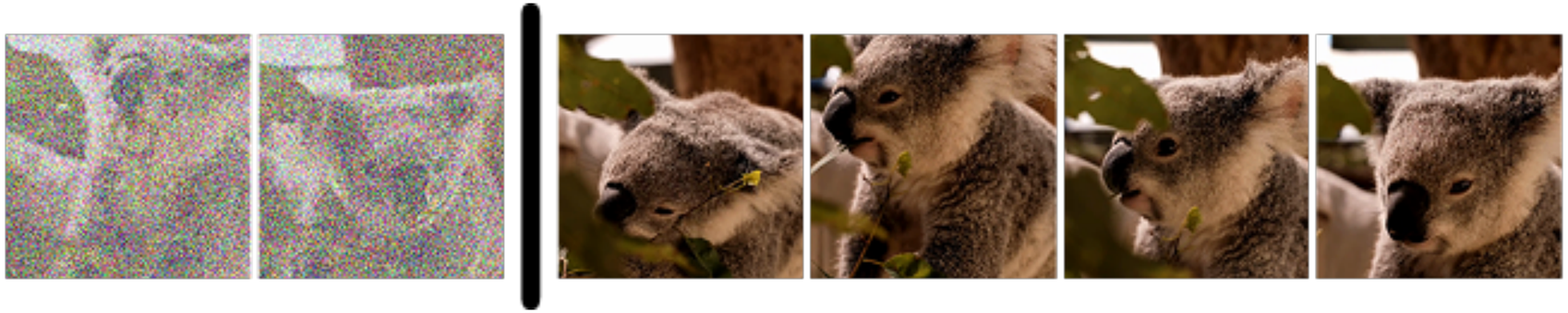
Classifier-Free Guidance

[Ho & Salimans, 2023]

Illustration from Sander Dieleman, “Guidance: A Cheat Code for Diffusion Models”



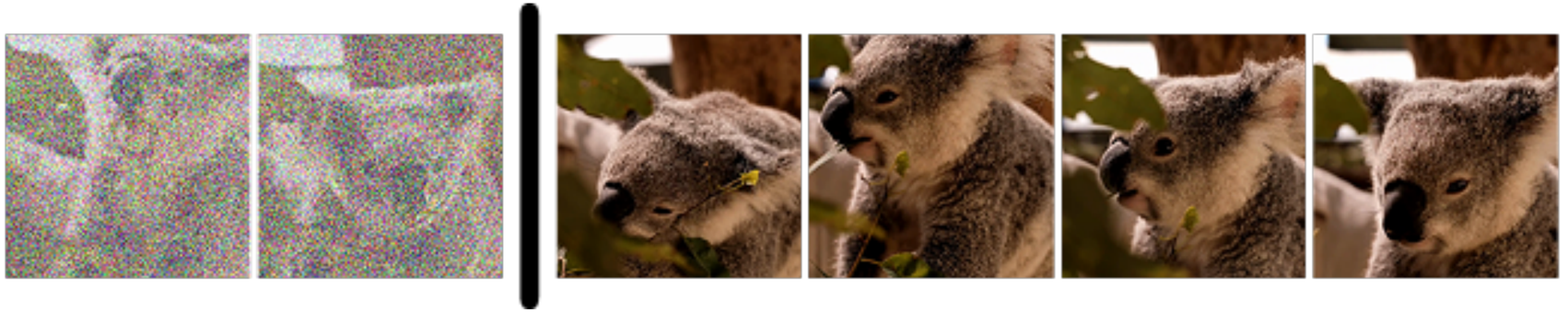
Insight: “History Frames” are Conditioning Variables for Target Frames!



Target Frames

History Frames

Insight: “History Frames” are Conditioning Variables for Target Frames!



Target Frames

History Frames

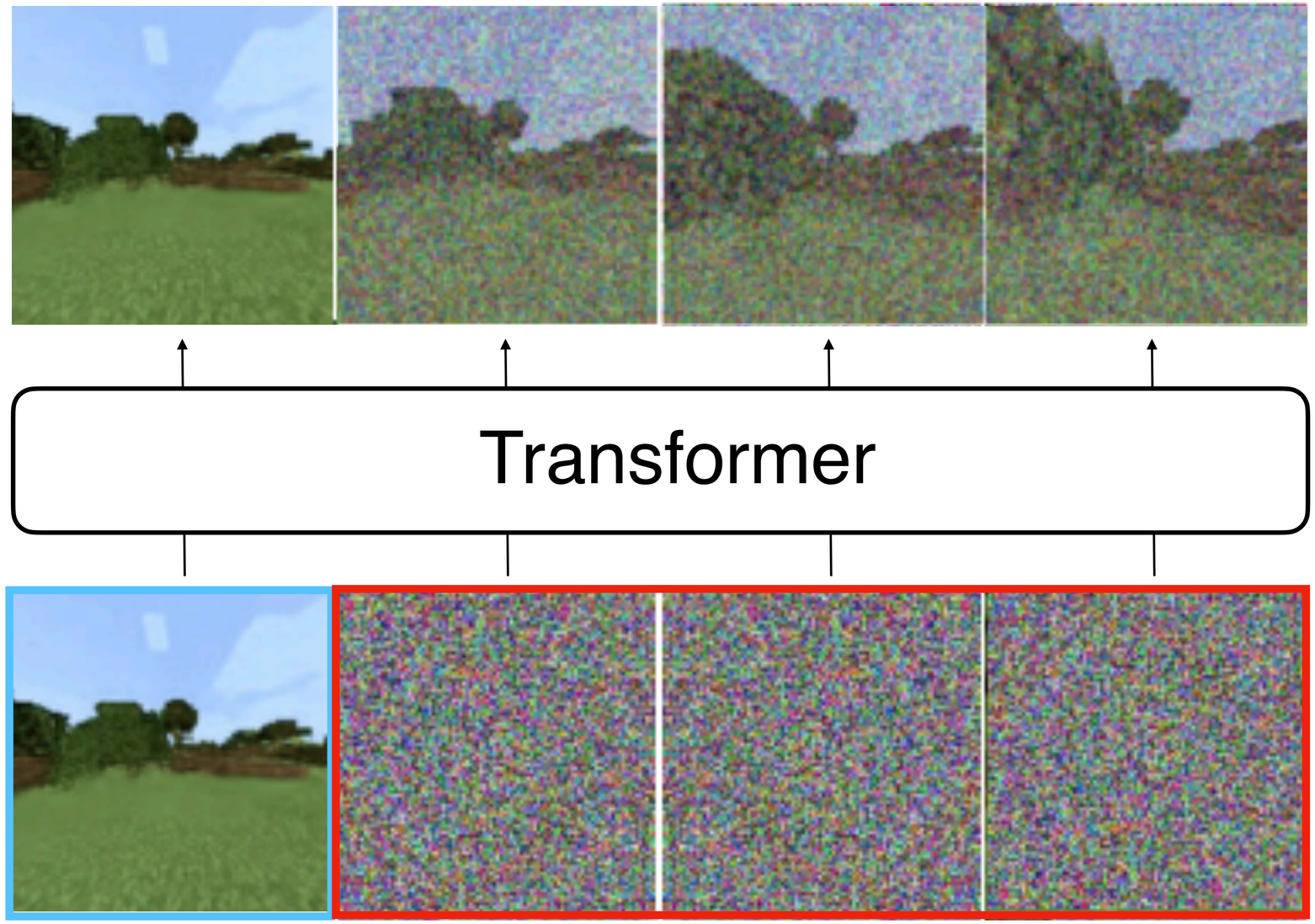
Special situation:
History is *variable-length*

Insight: “History Frames” are Conditioning Variables for Target Frames!

Can we do guidance with history frames?

Special situation:
History is *variable-length*

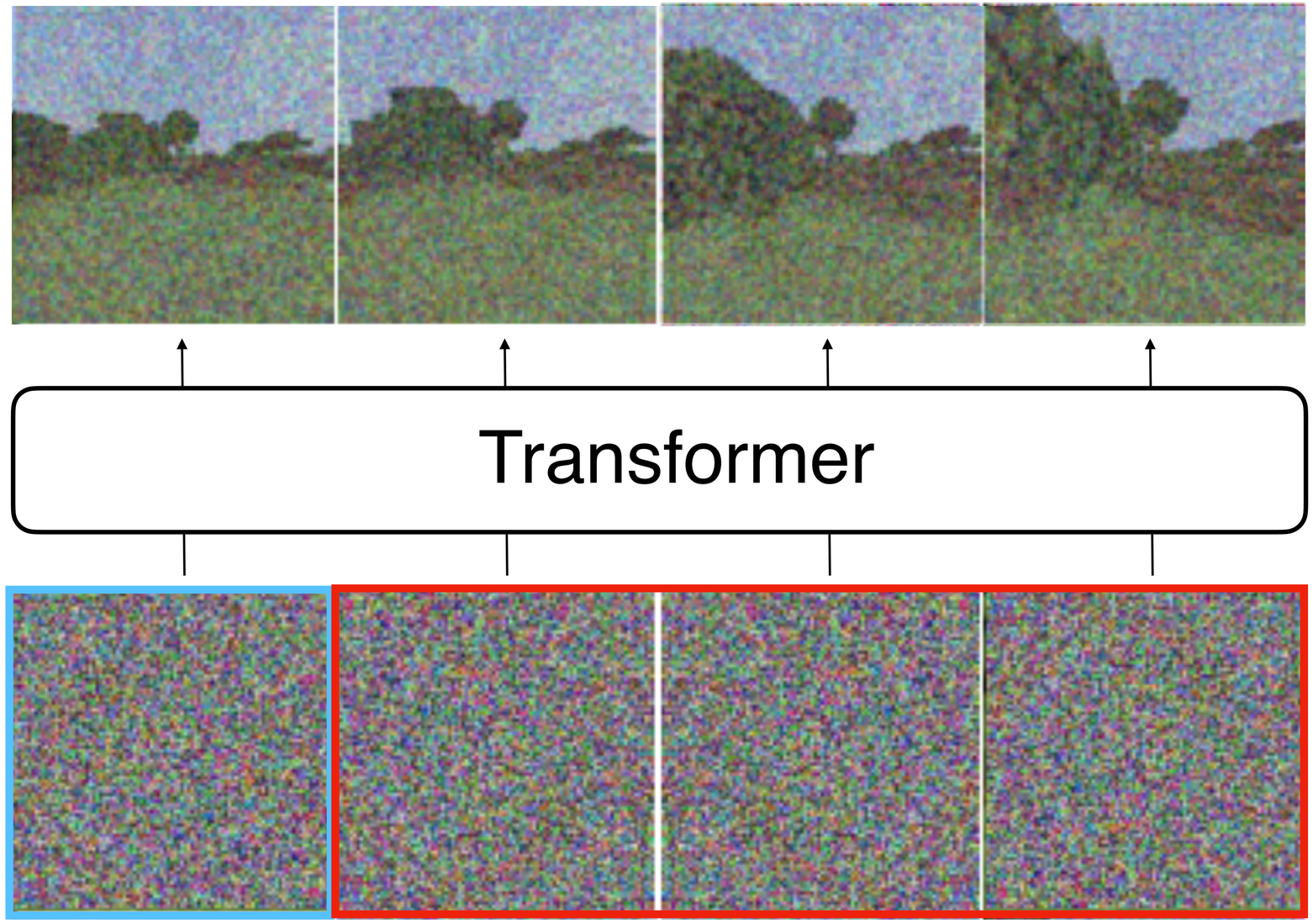
Insight: As Before, we can “switch off” conditioning simply via adding noise!



Sampling from

$$p(x_2, x_3, x_4 \mid x_1)$$

Insight: As Before, we can “switch off” conditioning simply via adding noise!

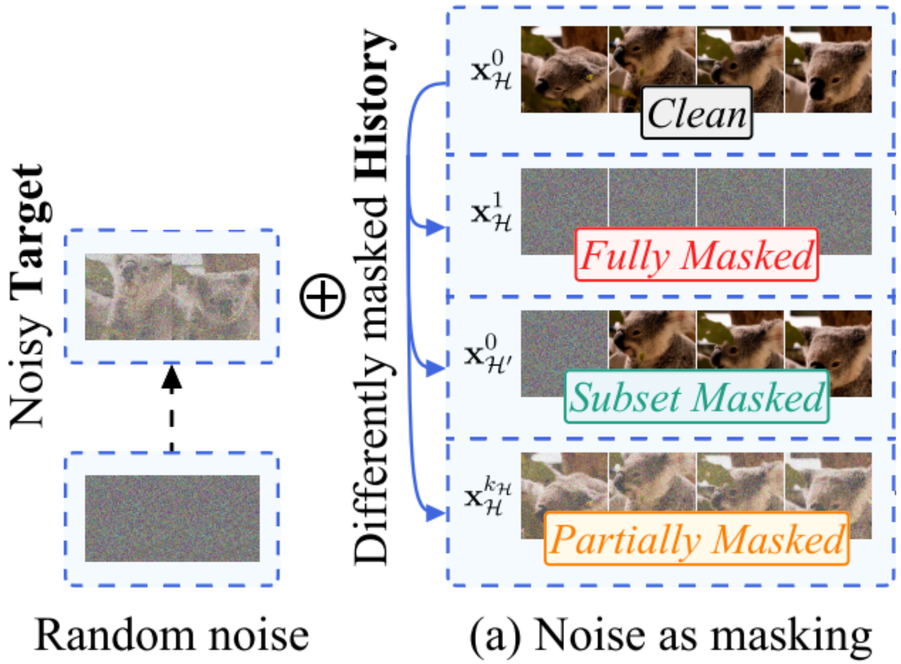


Sampling from

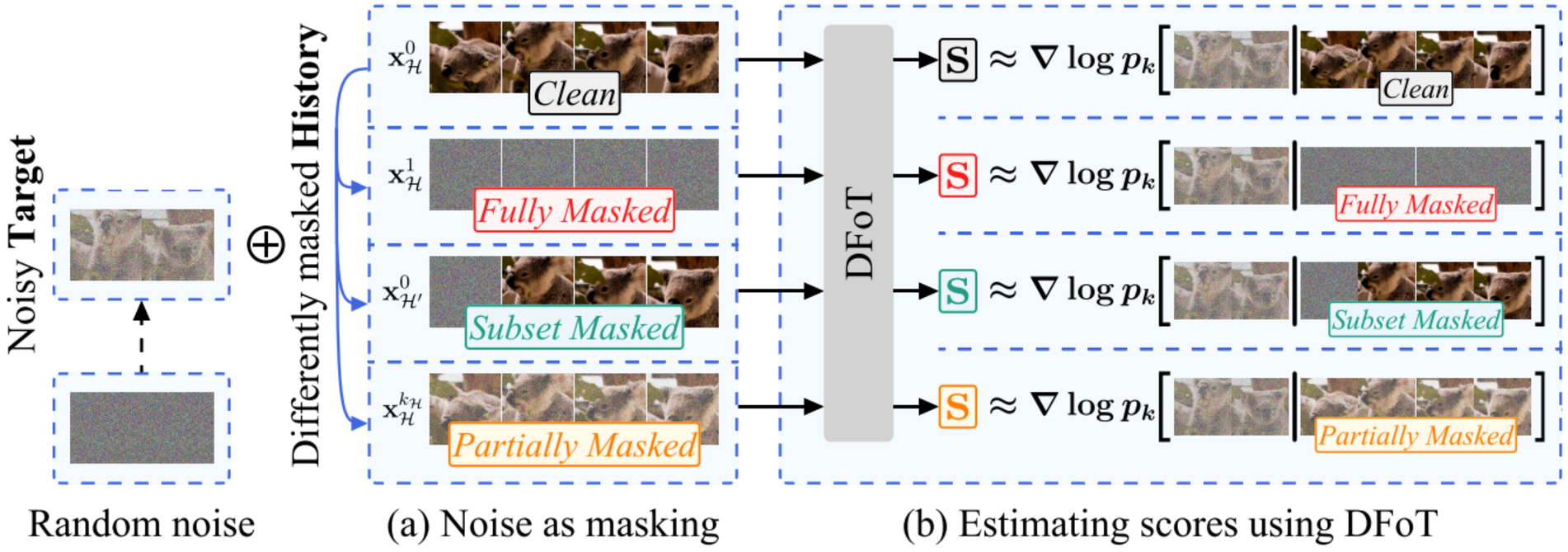
$$p(x_1, x_2, x_3, x_4)$$

Unconditional model!

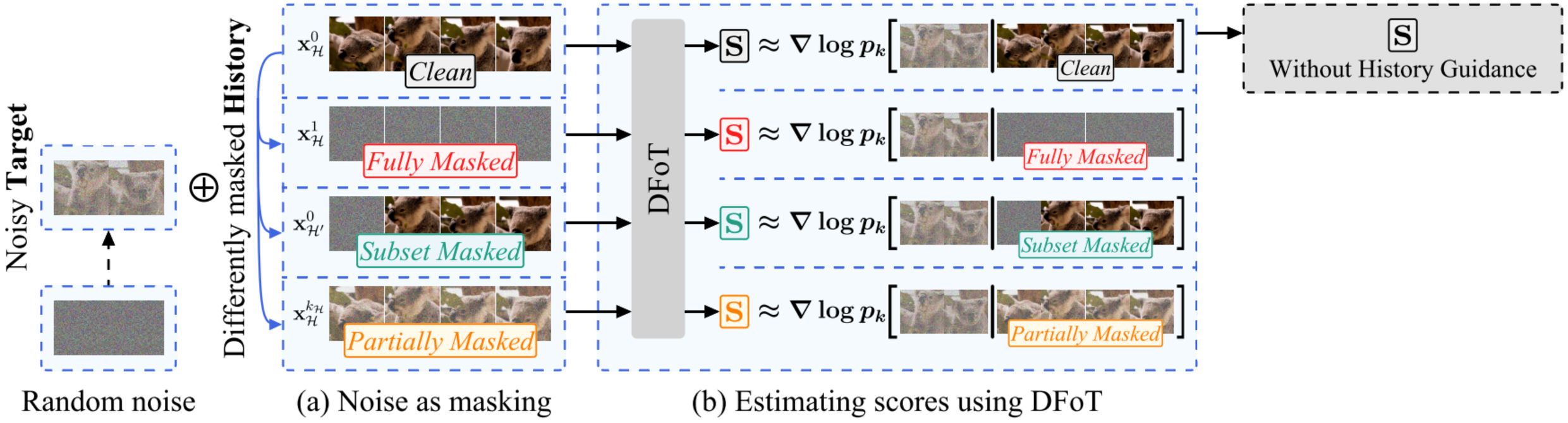
History-Guided Video Diffusion



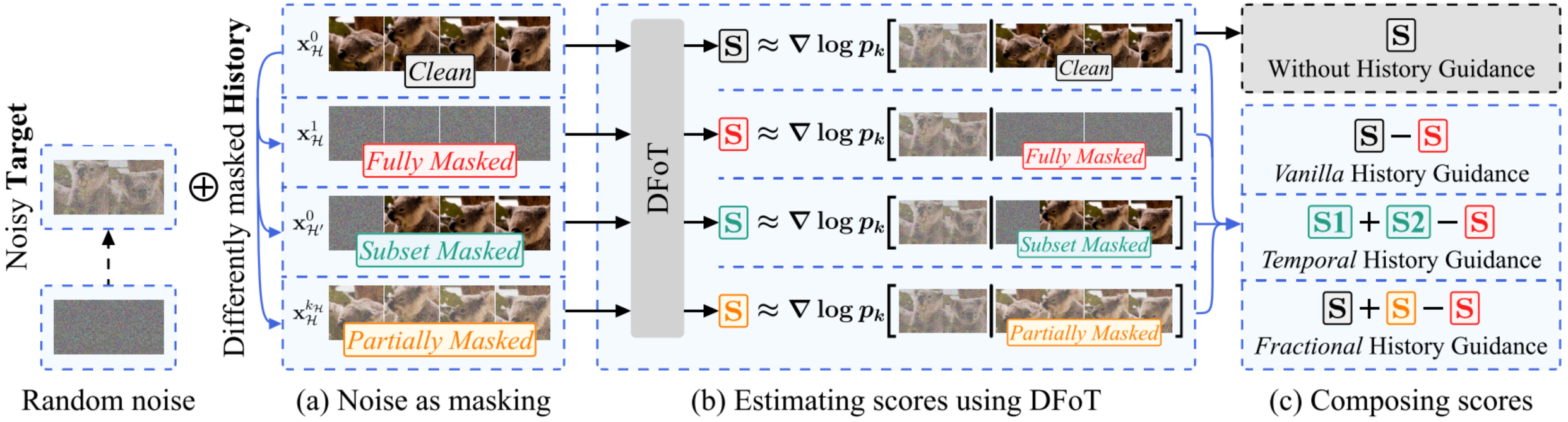
History-Guided Video Diffusion



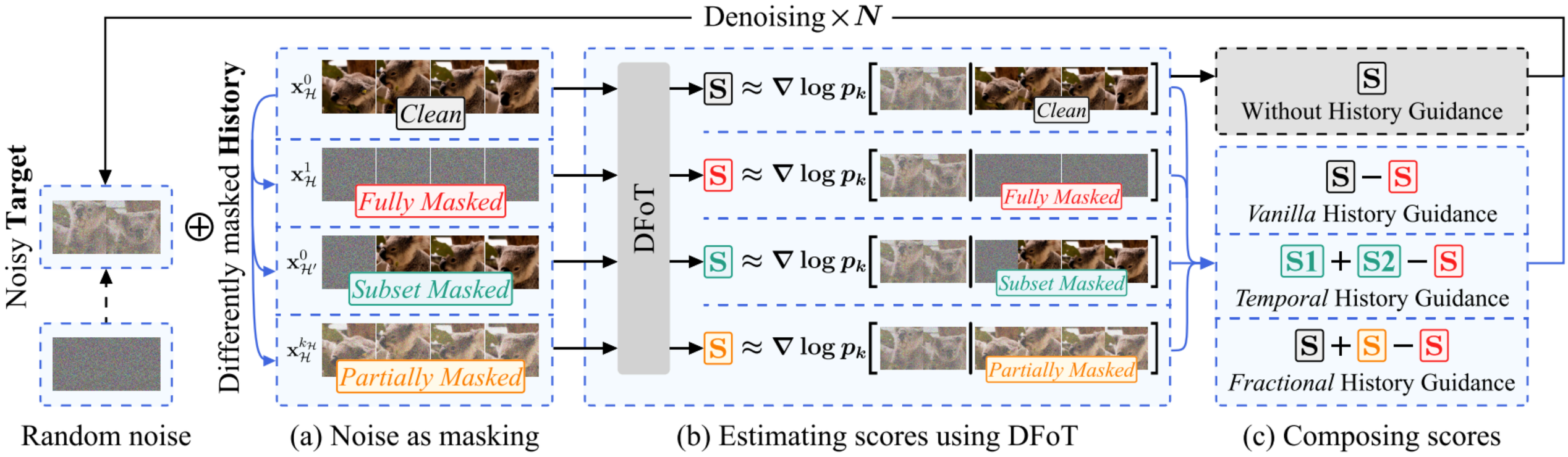
History-Guided Video Diffusion



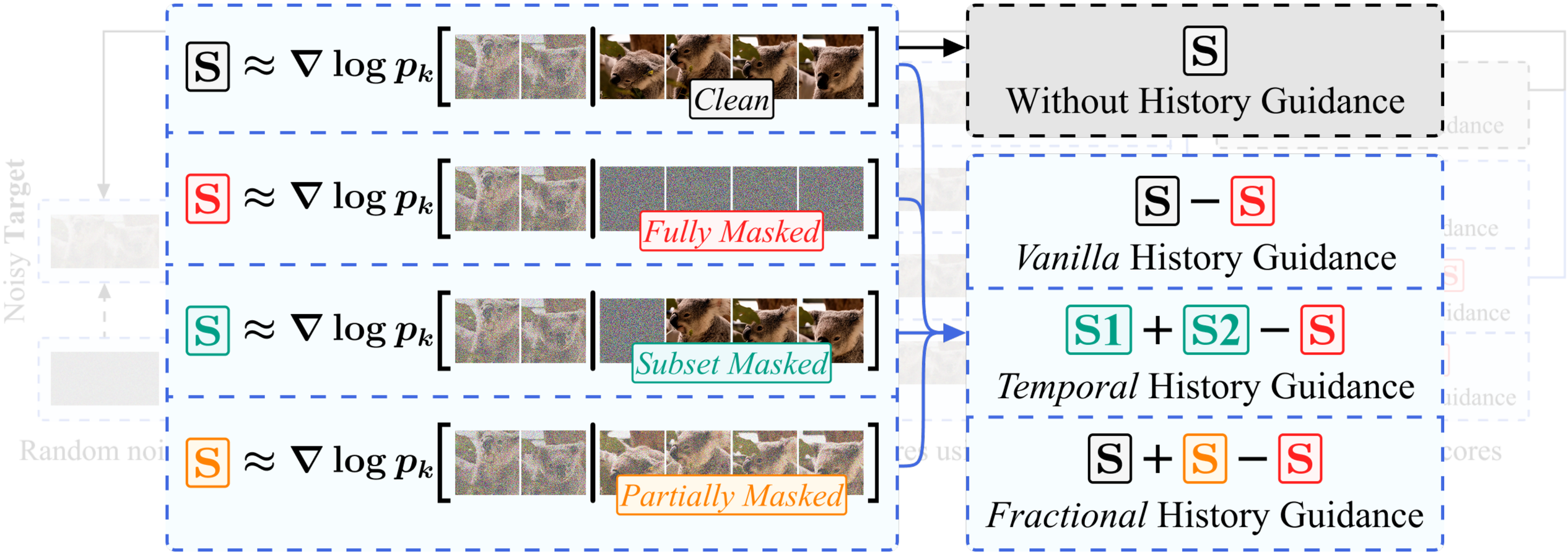
History-Guided Video Diffusion



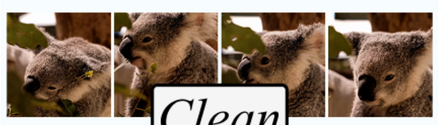
History-Guided Video Diffusion



History-Guided Video Diffusion



Fractional History Guidance

$$\boxed{\mathbf{S}} \approx \nabla \log p_k \Big[\; \text{  } \; \Big| \; \text{  } \; \text{Clean} \; \Big]$$

$$\boxed{\mathbf{S}} \approx \nabla \log p_k \Big[\; \text{  } \; \Big| \; \text{  } \; \text{Fully Masked} \; \Big]$$

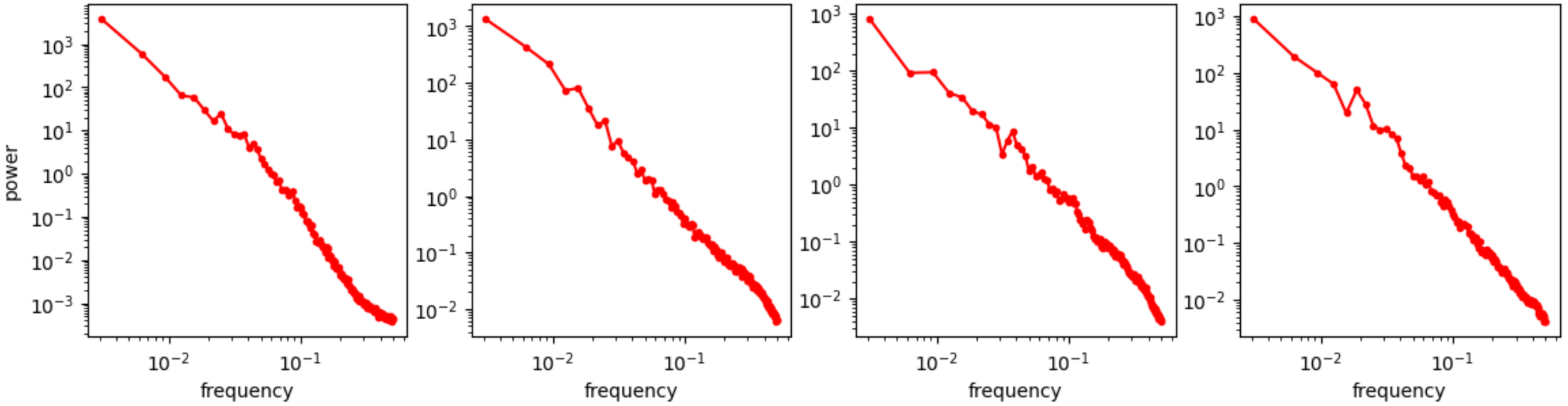
$$\boxed{\mathbf{S}} \approx \nabla \log p_k \Big[\; \text{  } \; \Big| \; \text{  } \; \text{Partially Masked} \; \Big]$$

$$\boxed{\mathbf{S}} + \boxed{\mathbf{S}} - \boxed{\mathbf{S}}$$

Fractional History Guidance

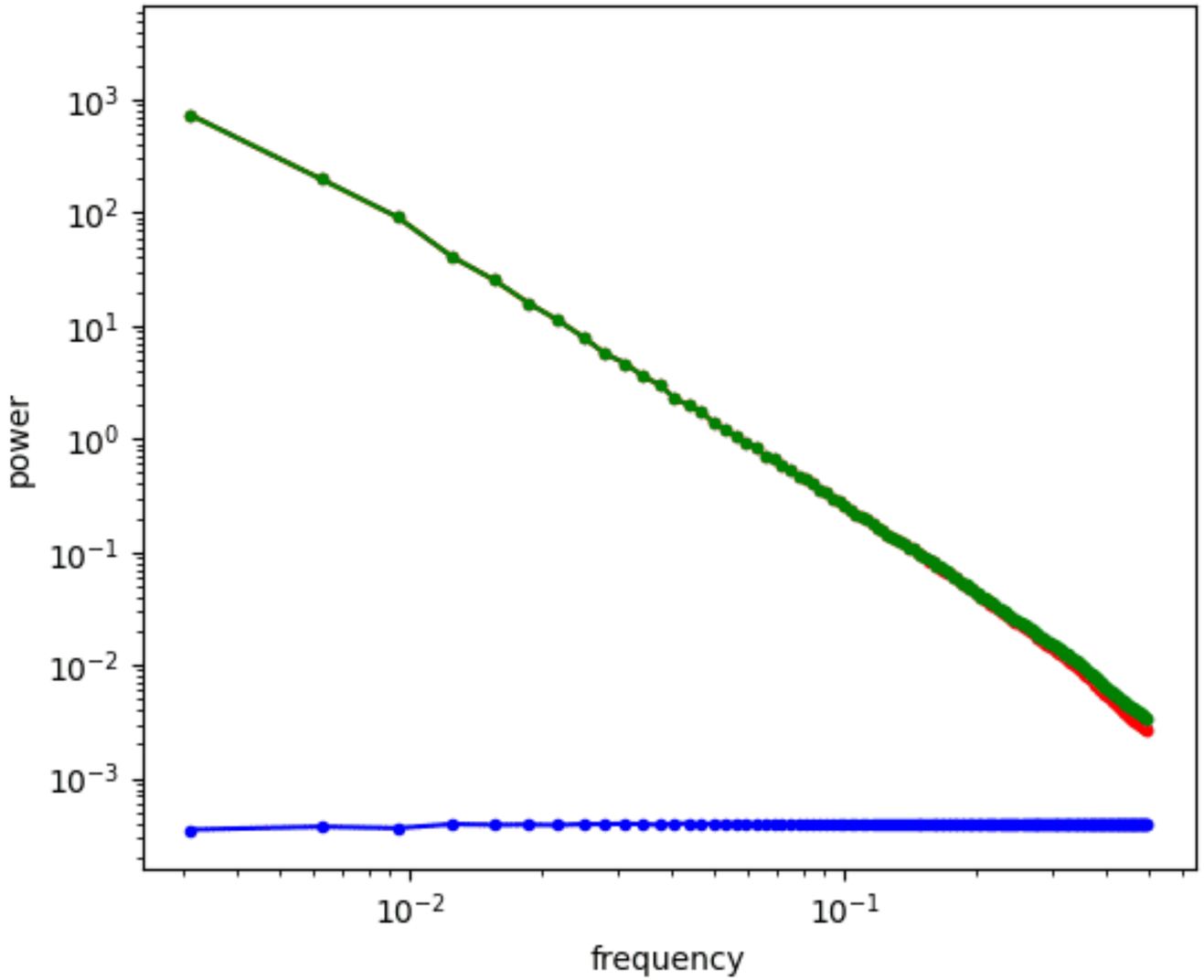
Insight: Image Power Spectra Follow Power-Law

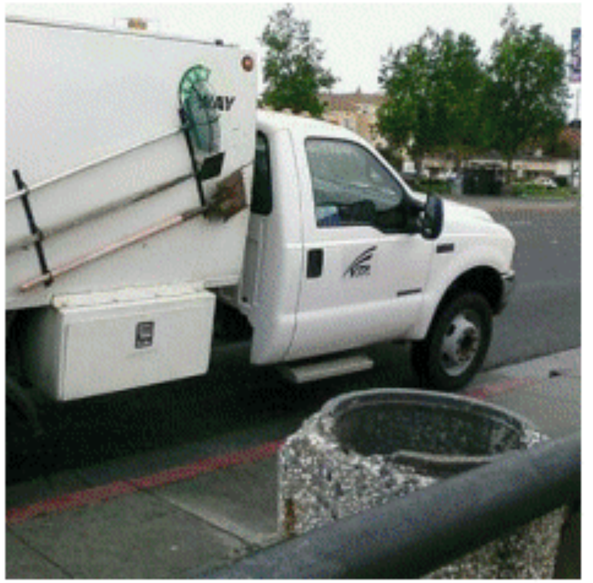
Sander Dieleman: Diffusion is spectral autoregression



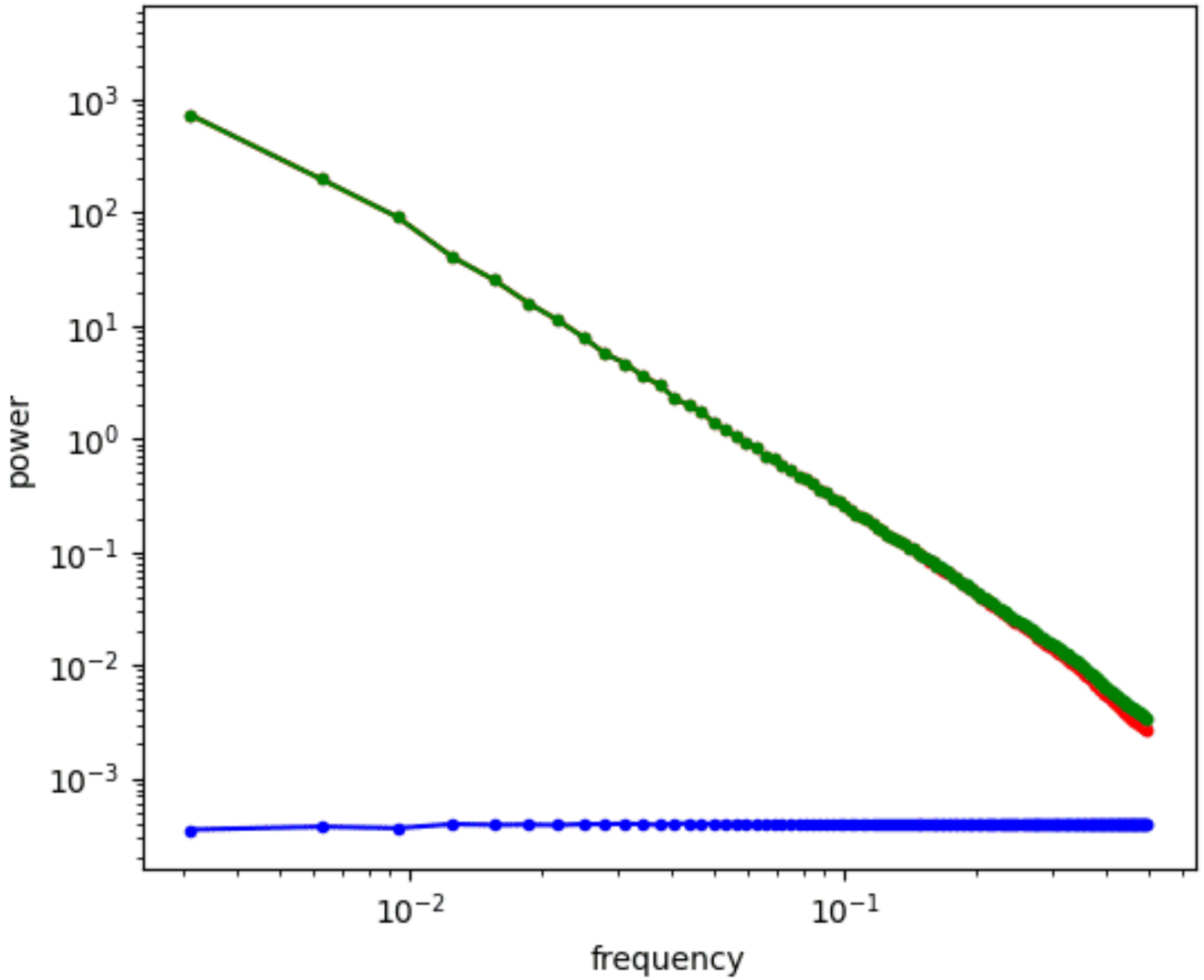
https://sander.ai/2024/09/02/spectral-autoregression.html

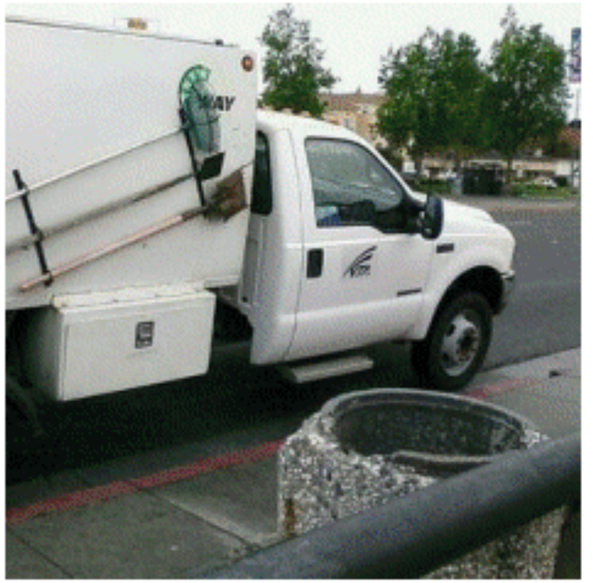
What happens when we add noise?



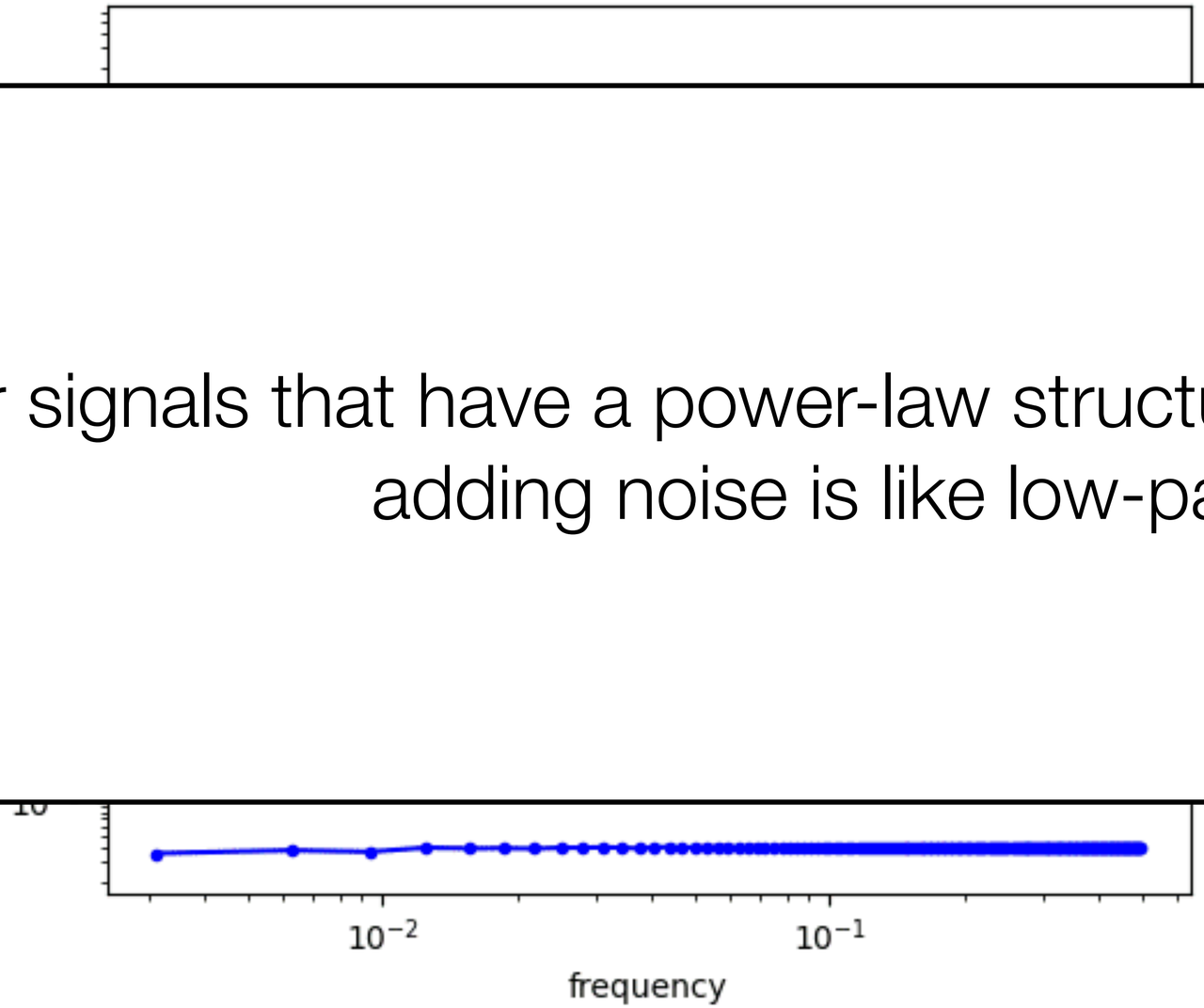


What happens when we add noise?



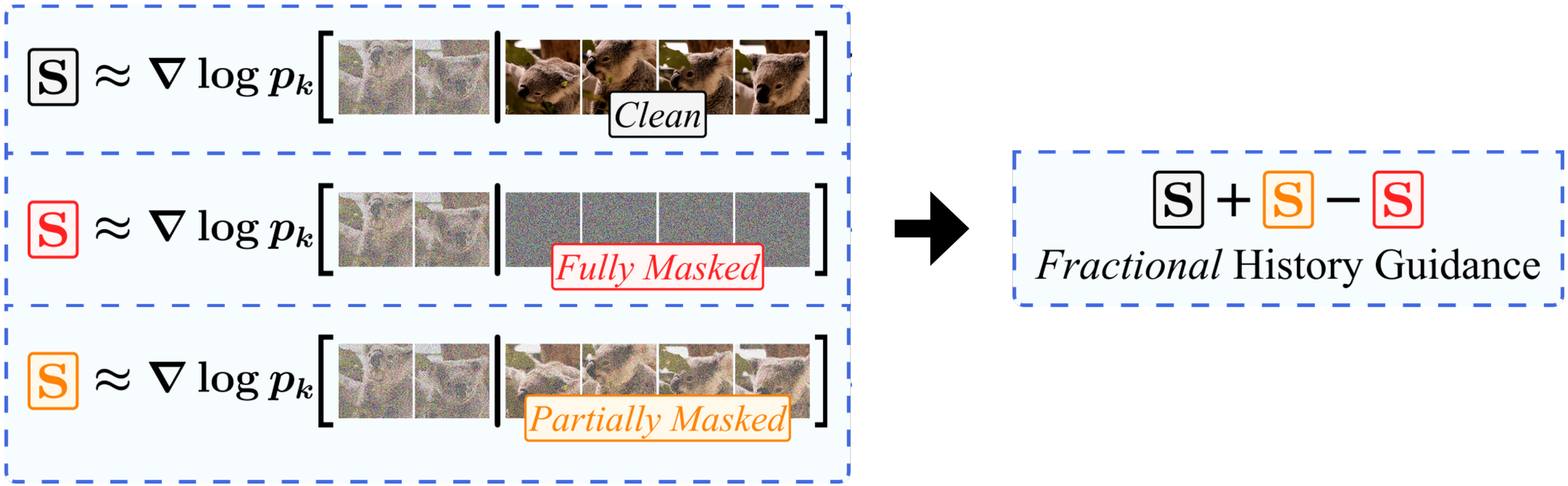


What happens when we add noise?



For signals that have a power-law structure in the spectral density, adding noise is like low-pass filtering!

Fractional History Guidance



Essentially performs guidance with “blurred” history!

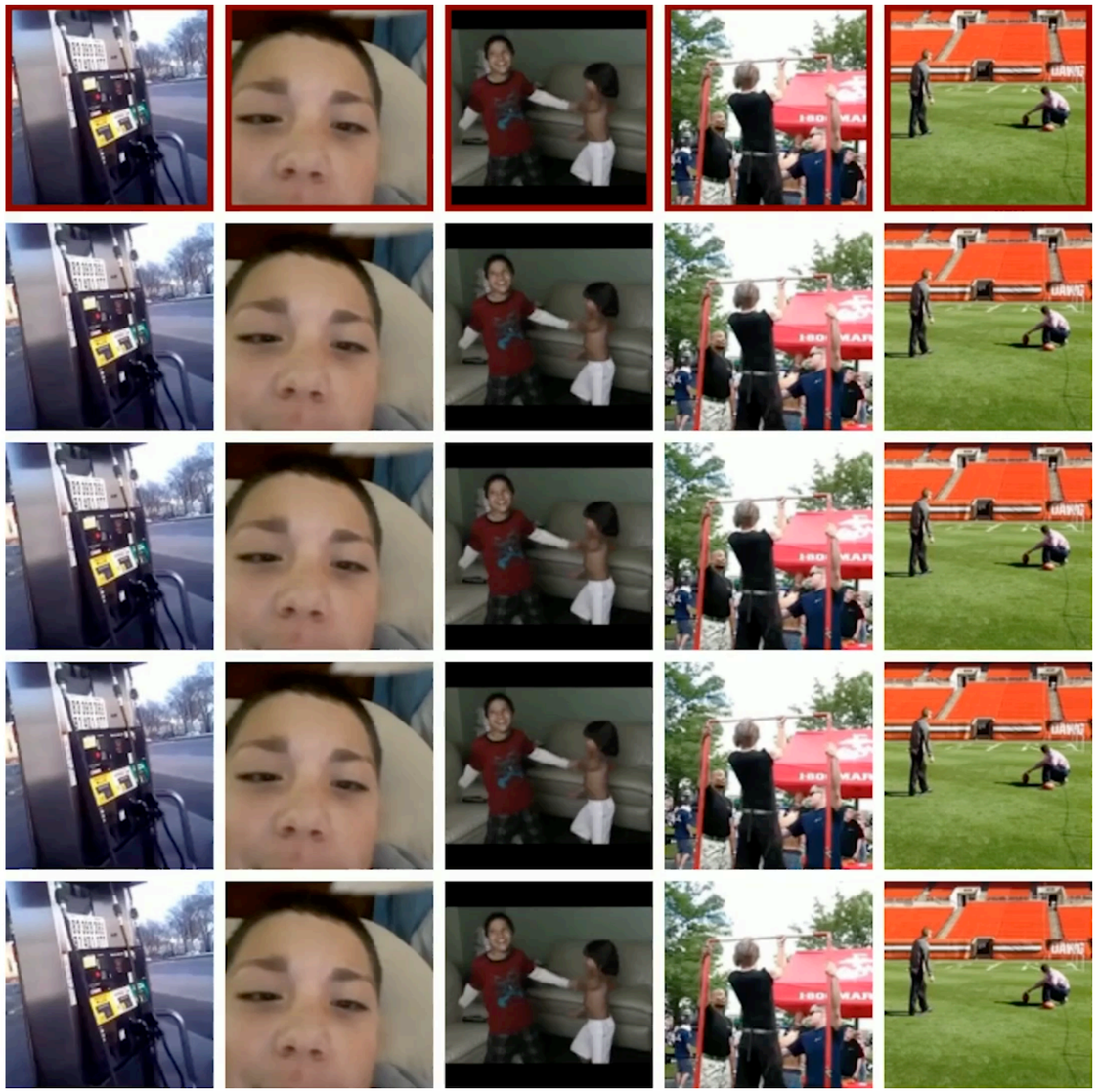
Helps with the curse of dimensionality: The more history frames, the less support in training data. But more support for *blurry* frames!

History-Guided Video Diffusion:
Improved Consistency & Motion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

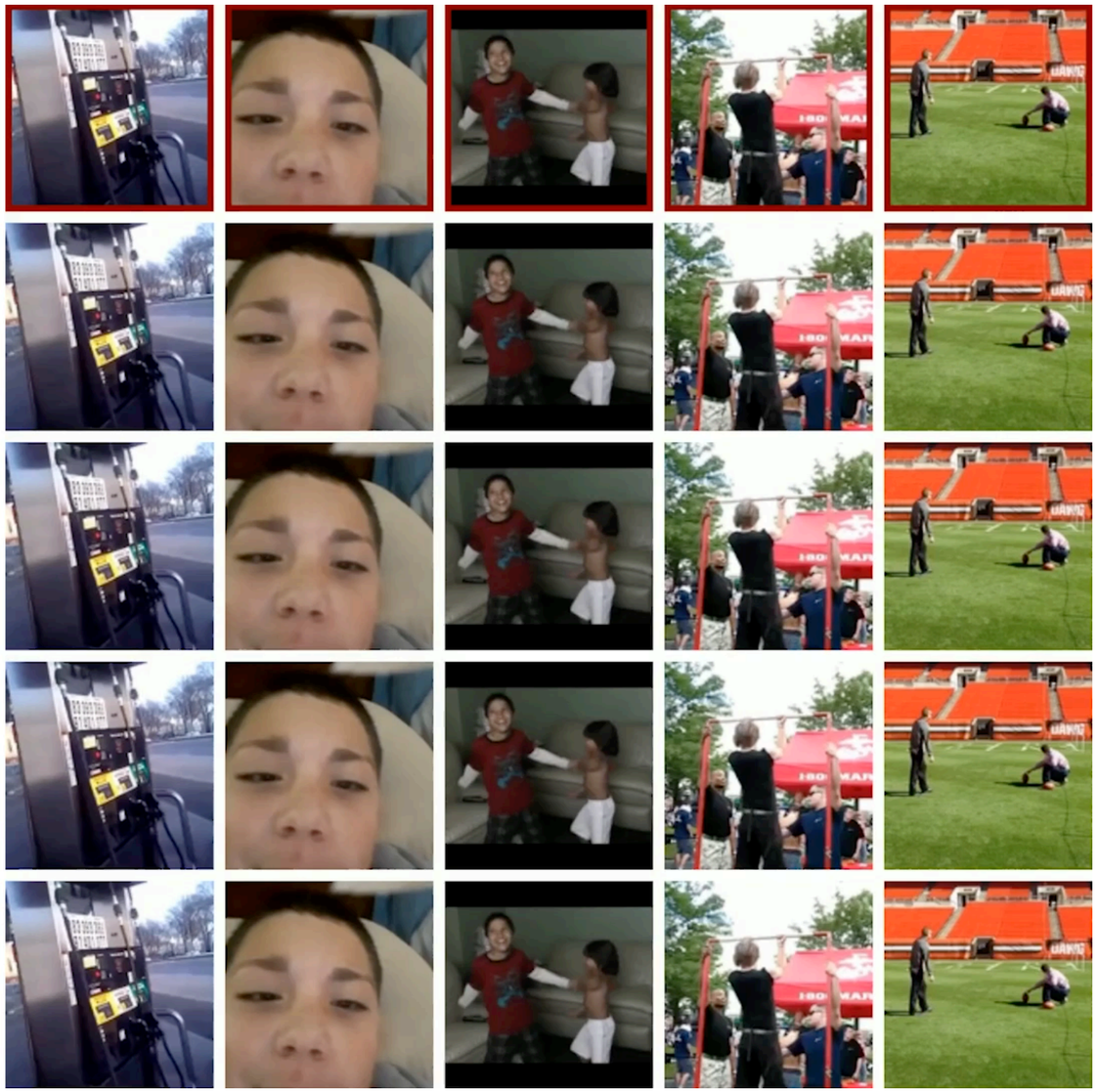


History-Guided Video Diffusion:
Improved Consistency & Motion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

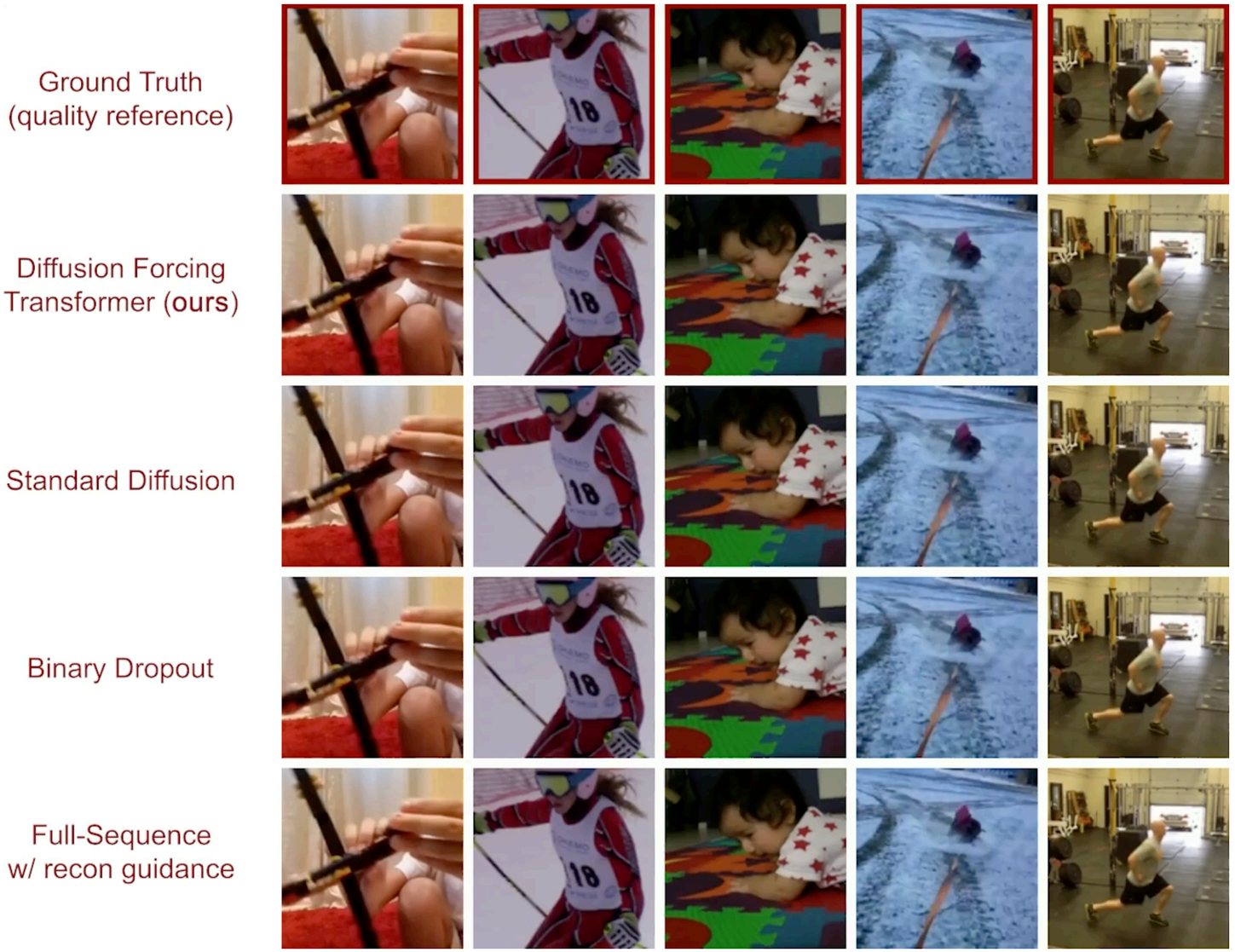


History-Guided Video Diffusion:
Improved Consistency & Motion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

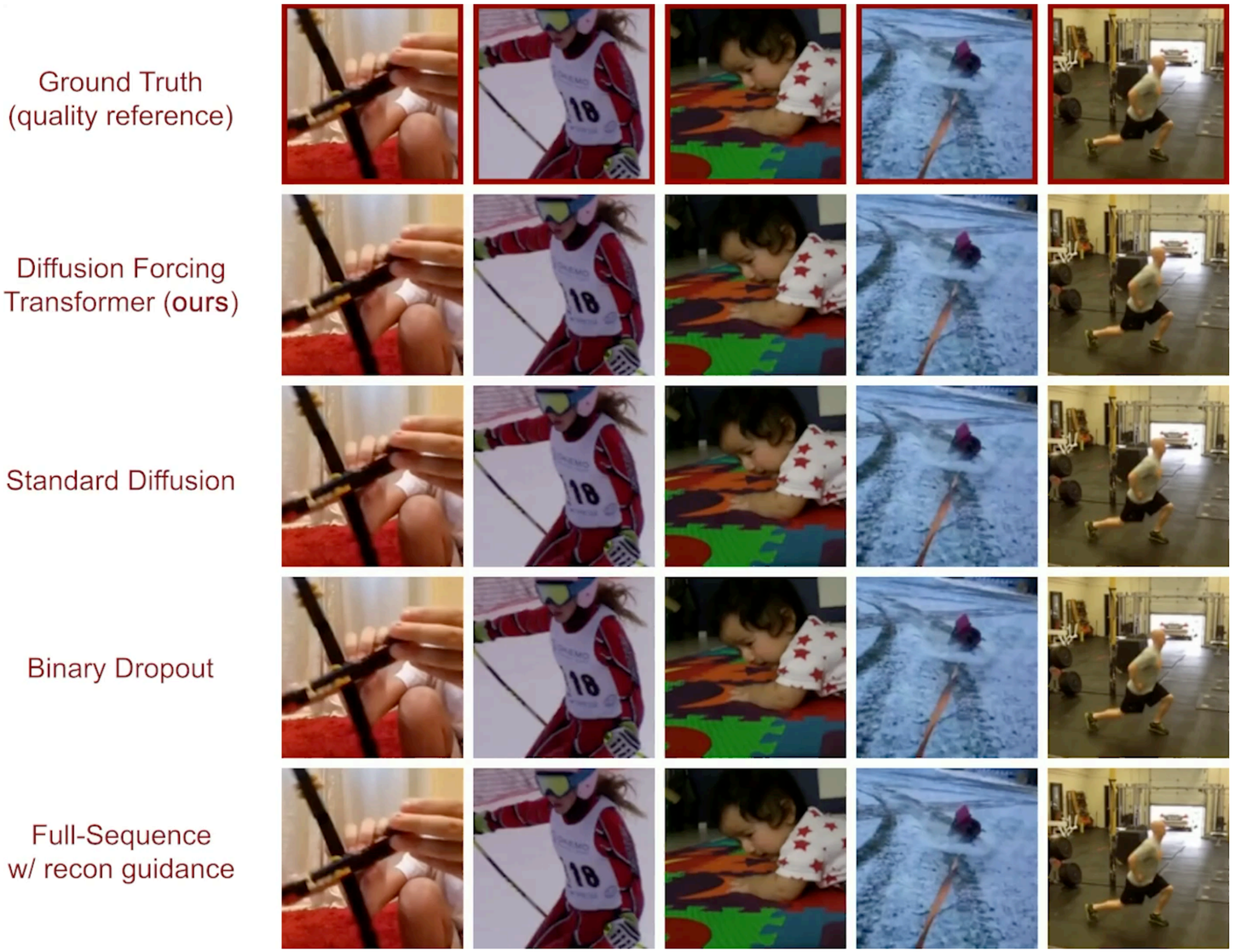


History-Guided Video Diffusion:
Improved Consistency & Motion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance



History-Guided Video Diffusion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

With HG

Without HG

History-Guided Video Diffusion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

With HG

Without HG

History-Guided Video Diffusion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

With HG

Without HG

History-Guided Video Diffusion

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

With HG

Without HG

History-Guided Video Diffusion:
Diverse, High-Quality Videos

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

History-Guided Video Diffusion:
Diverse, High-Quality Videos

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

History-Guided Video Diffusion:
Ultra-Long Video Generation

S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

S + S − S
Fractional History Guidance

Input: **Two starting frames**
Output: **Long-rollout, 3D-consistent novel views, controlled by camera pose!**

History-Guided Video Diffusion:
Ultra-Long Video Generation

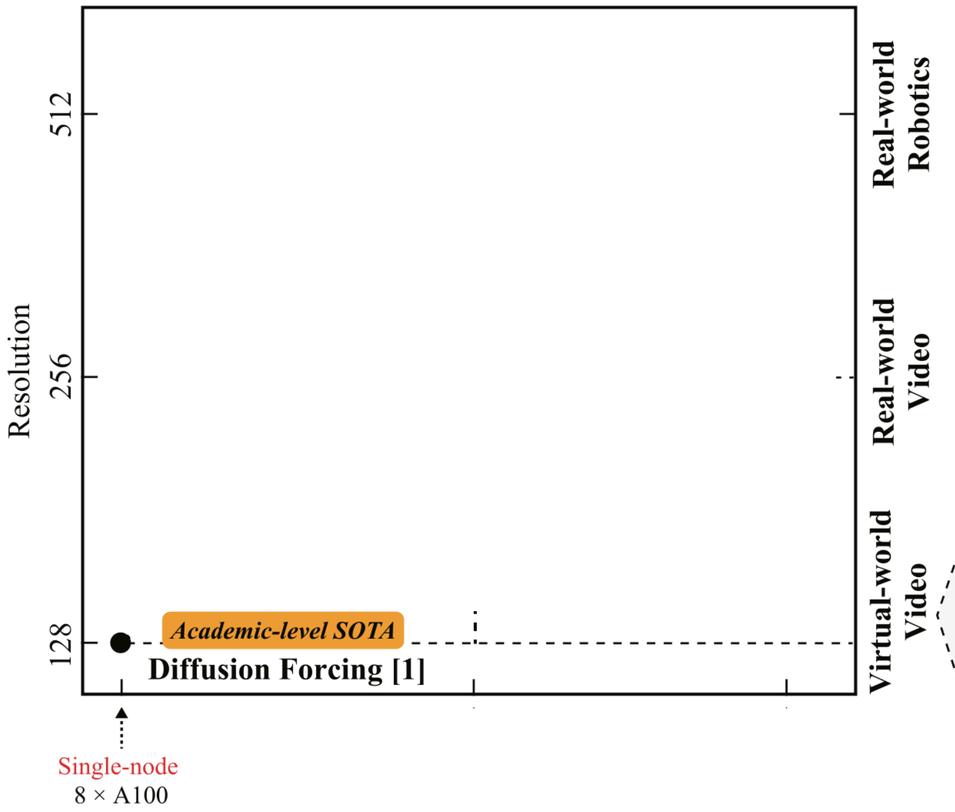
S − S
Vanilla History Guidance

S1 + S2 − S
Temporal History Guidance

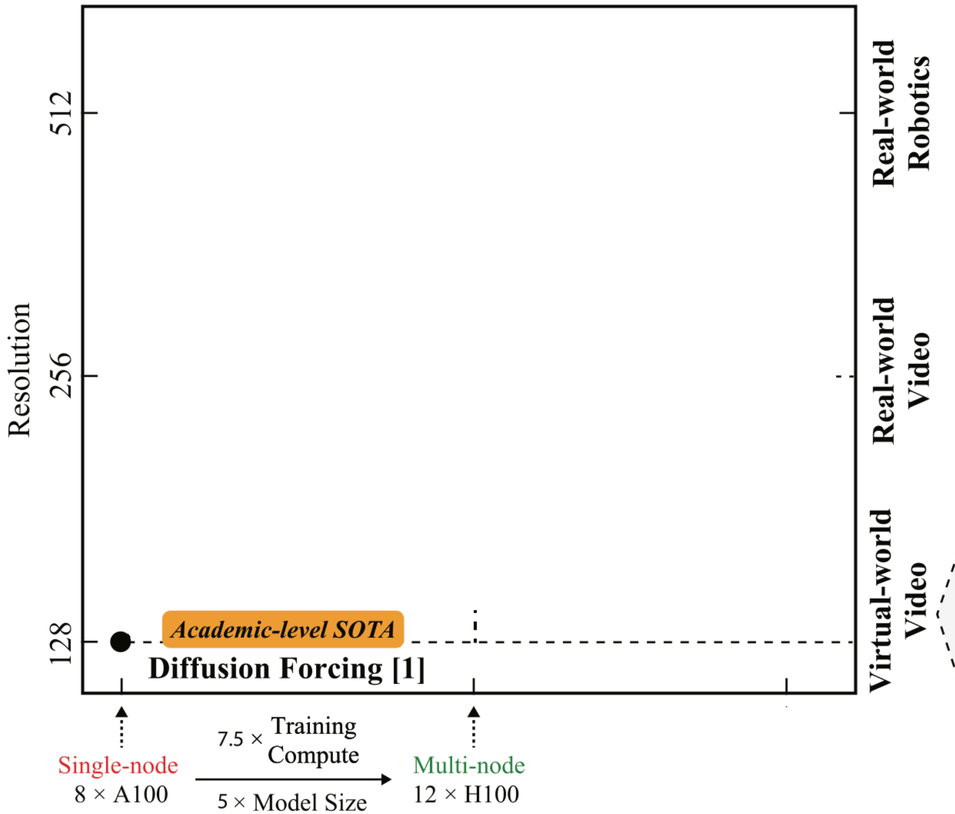
S + S − S
Fractional History Guidance

Input: **Two starting frames**
Output: **Long-rollout, 3D-consistent novel views, controlled by camera pose!**

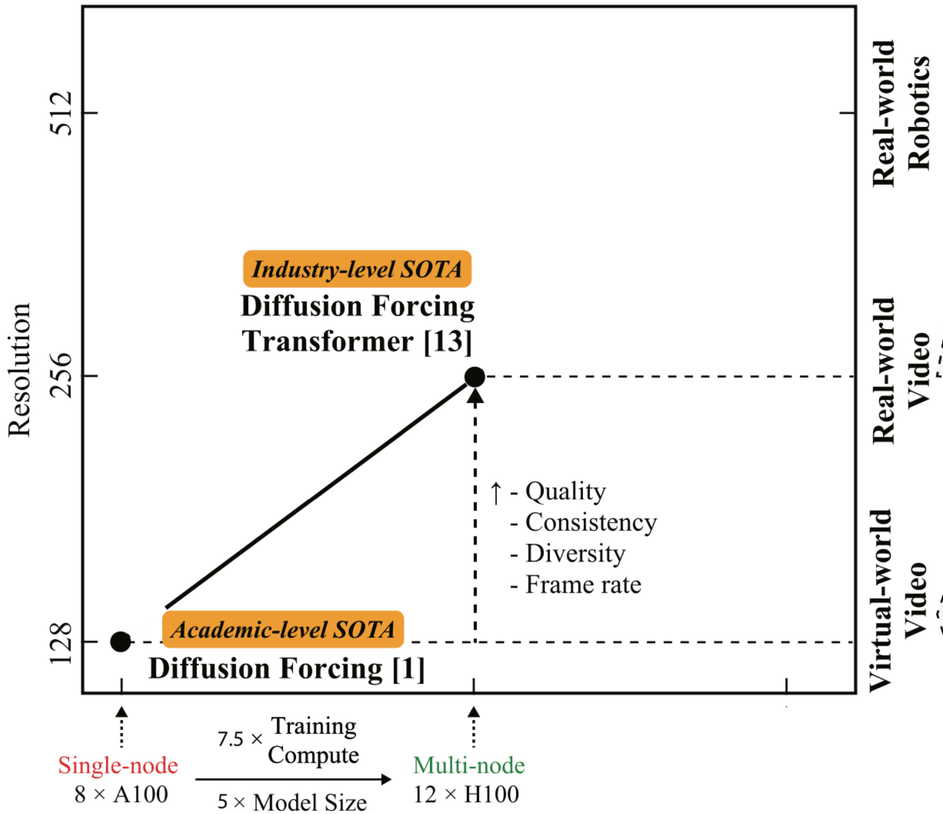
Scaling Helps

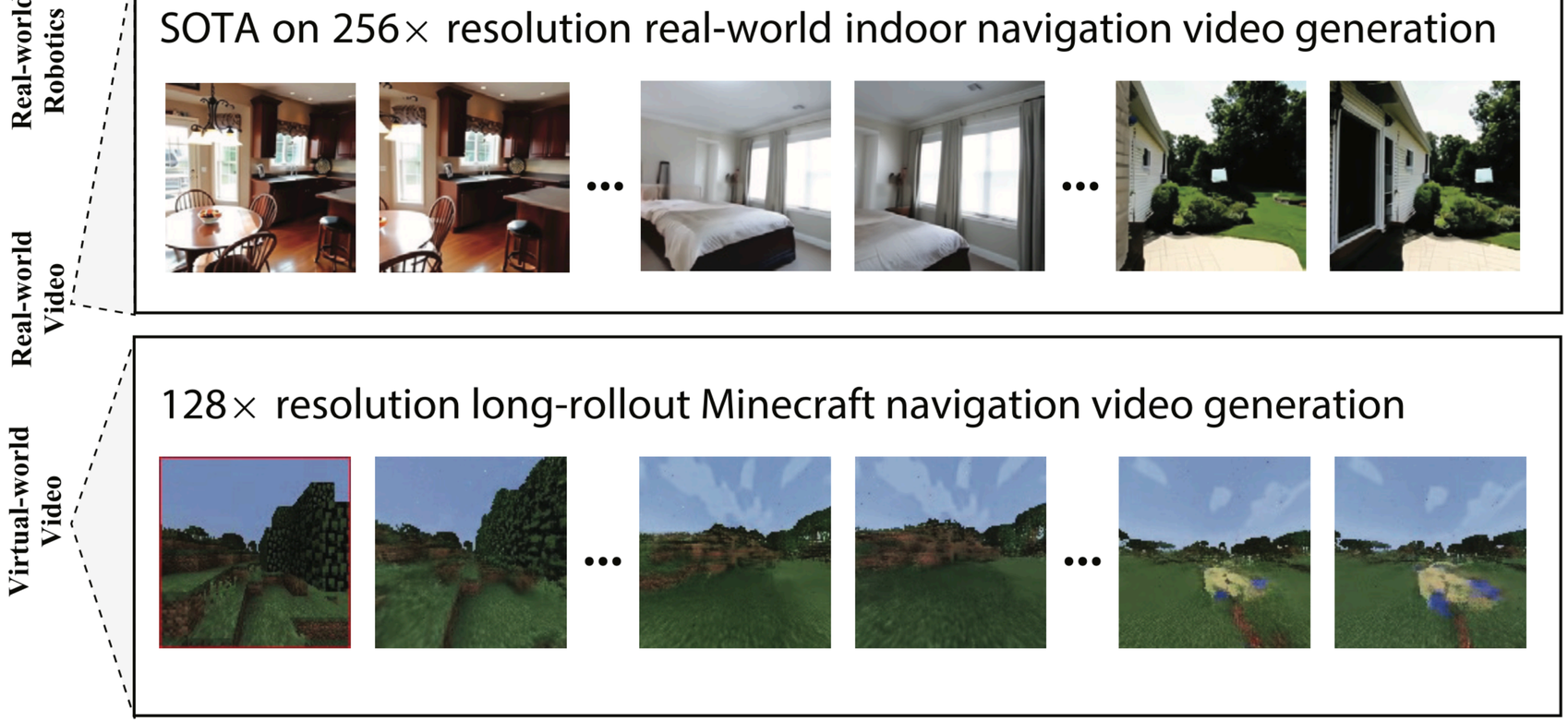


Scaling Helps



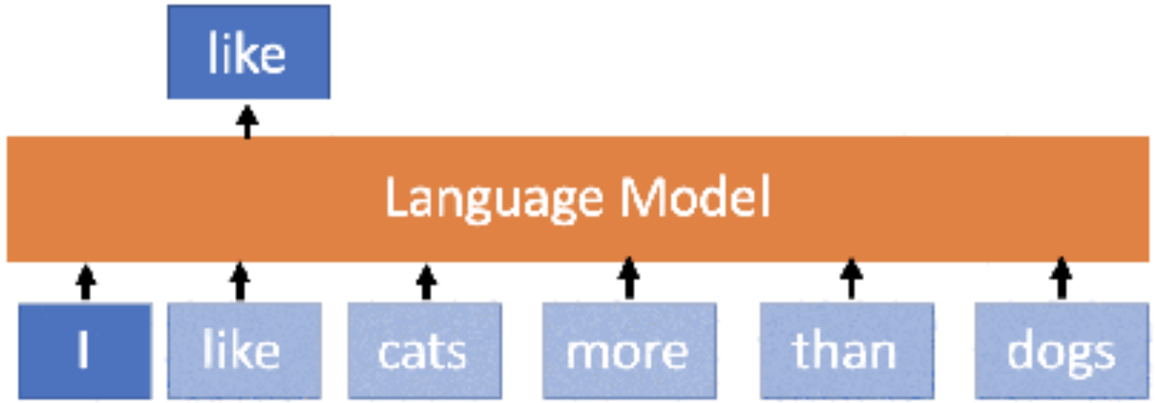
Scaling Helps





Two major paradigms for sequence generation

Video generation generally trained with **Full-Sequence Diffusion**

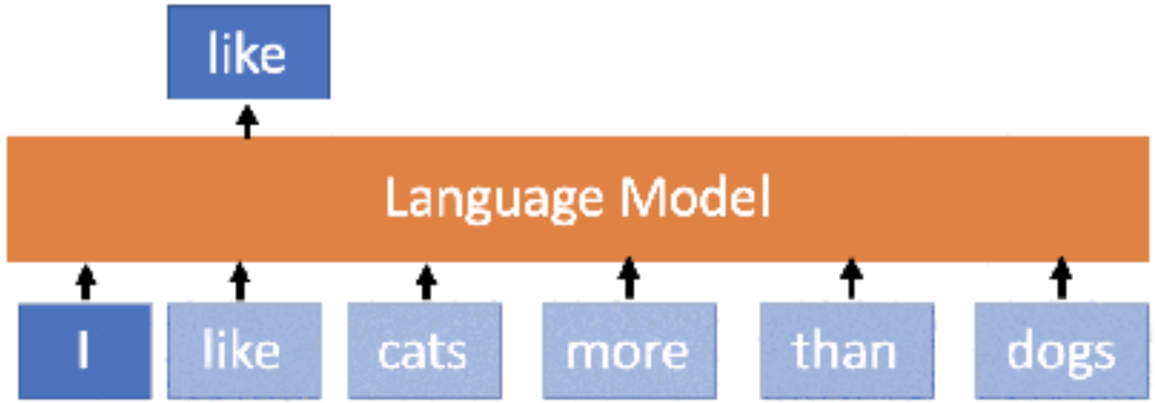


LLMs generally trained with **Next-Token Prediction**

Diffusion Probabilistic Modeling for Video Generation, Yang et al. 2022
Video Diffusion Models, Ho et al. 2022

Two major paradigms for sequence generation

Video generation generally trained with **Full-Sequence Diffusion**



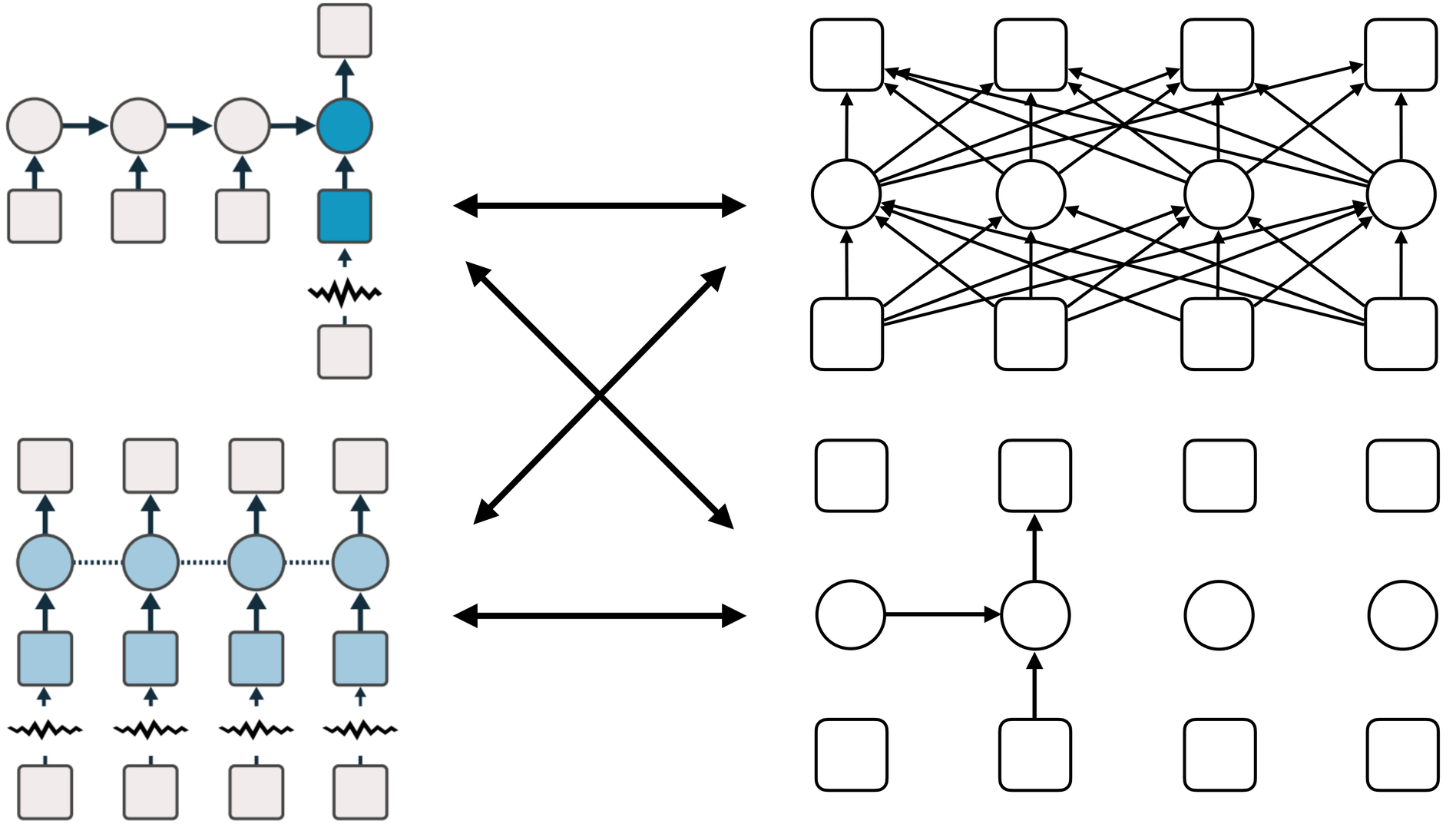
LLMs generally trained with **Next-Token Prediction**

Diffusion Probabilistic Modeling for Video Generation, Yang et al. 2022
Video Diffusion Models, Ho et al. 2022

Questions of Architecture…

Next-Token

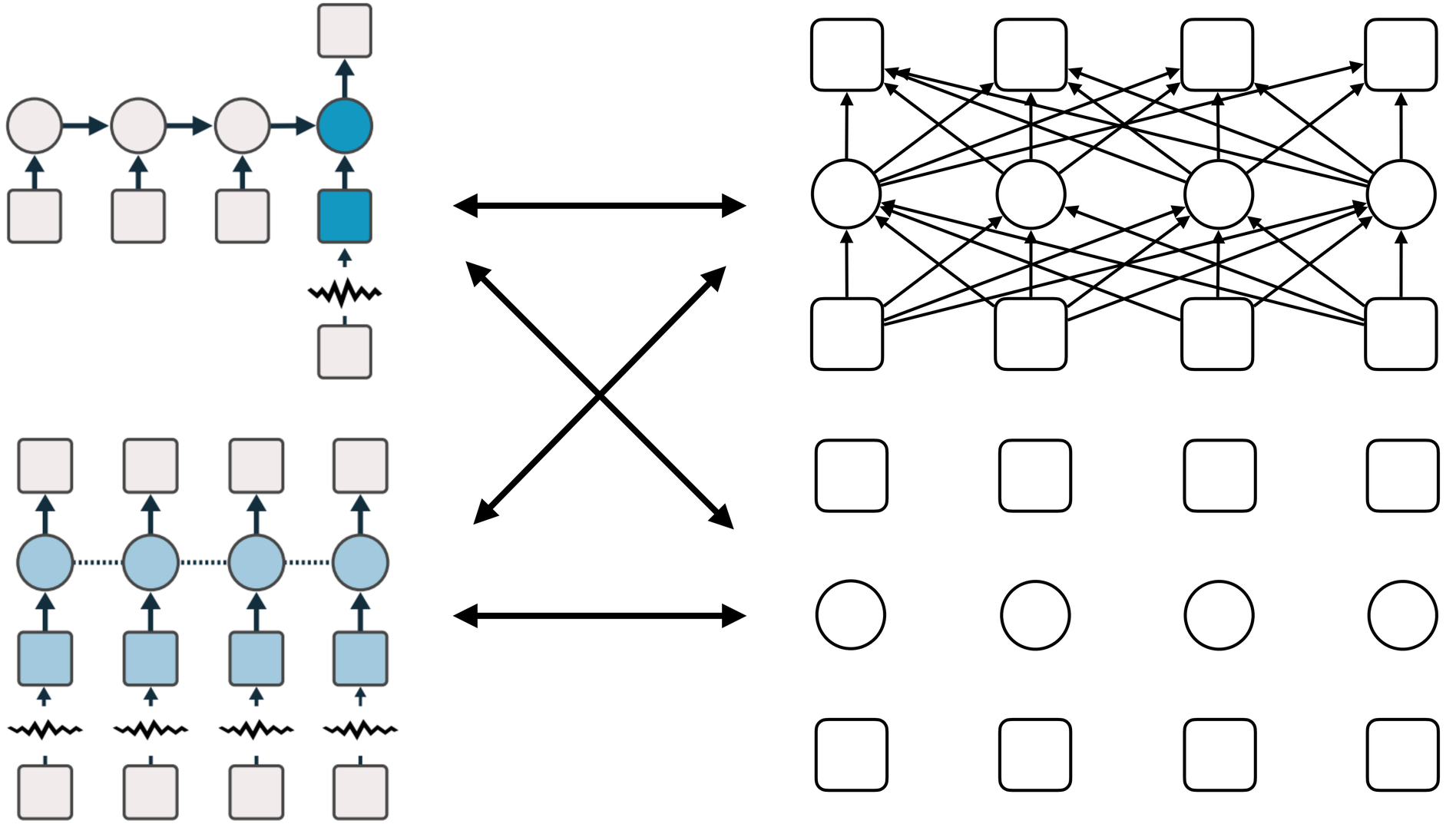
Full-Seq. Diffusion



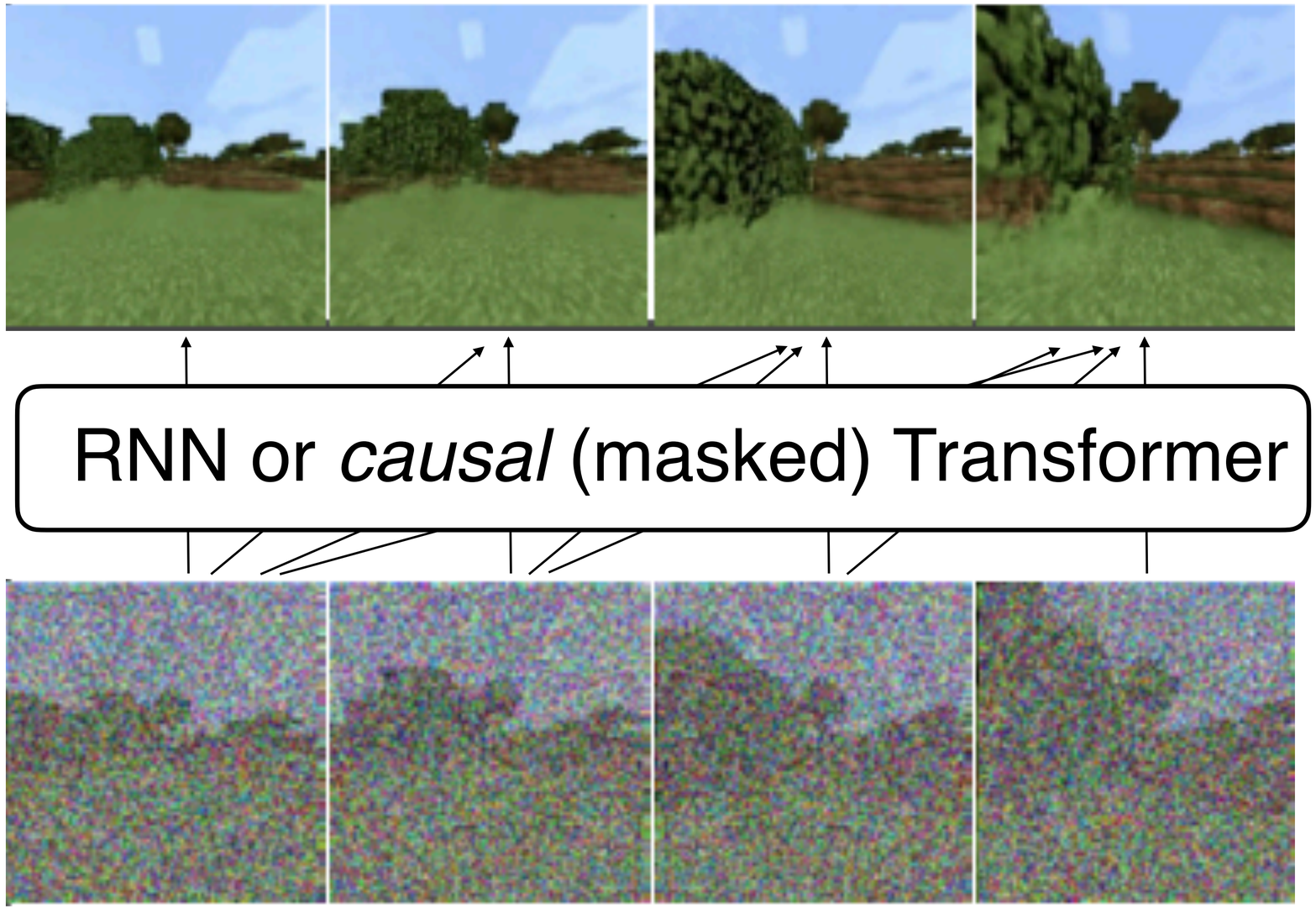
Questions of Architecture…

Next-Token

Full-Seq. Diffusion

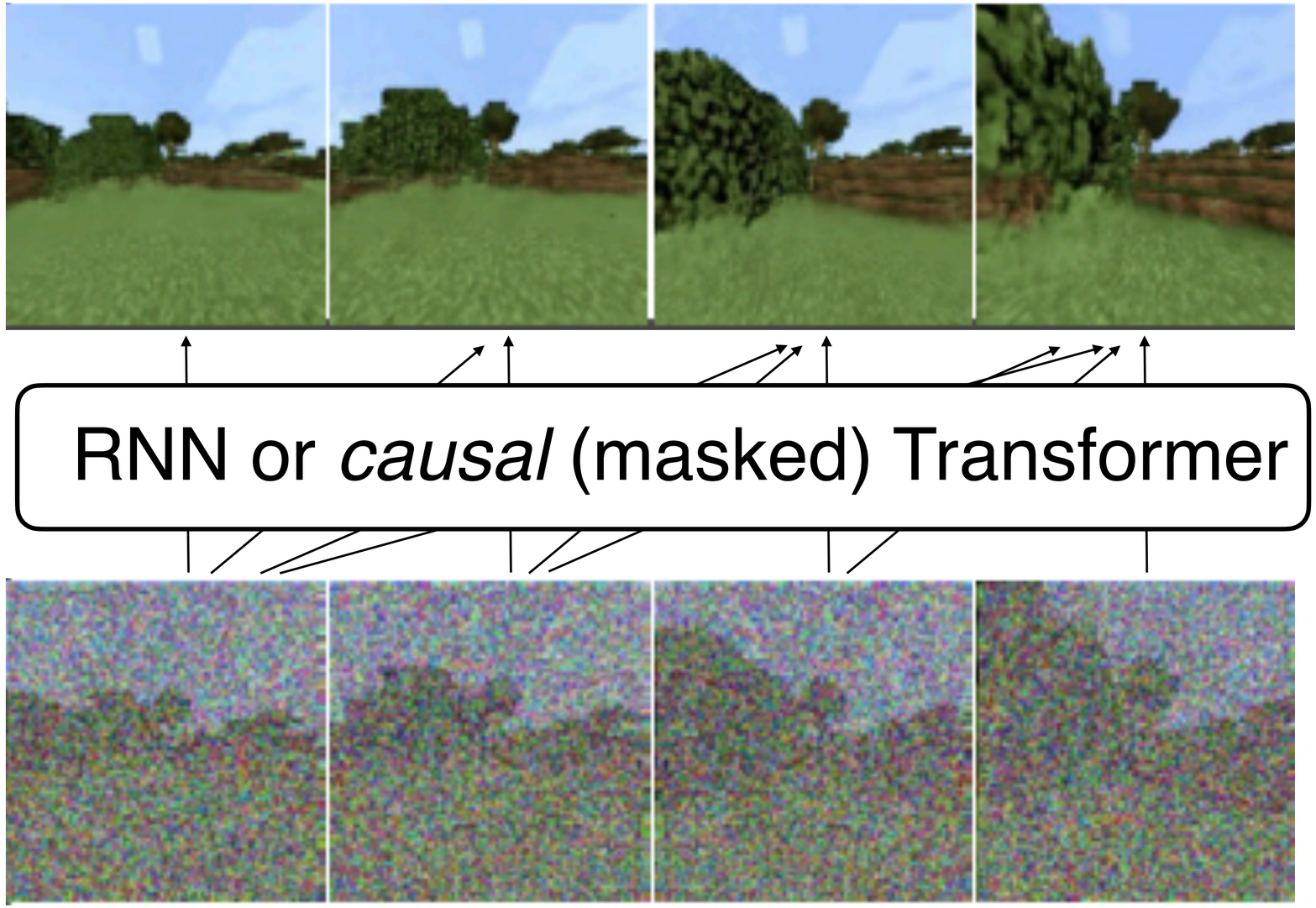


Example: RNN Implements Full-Seq. Diffusion

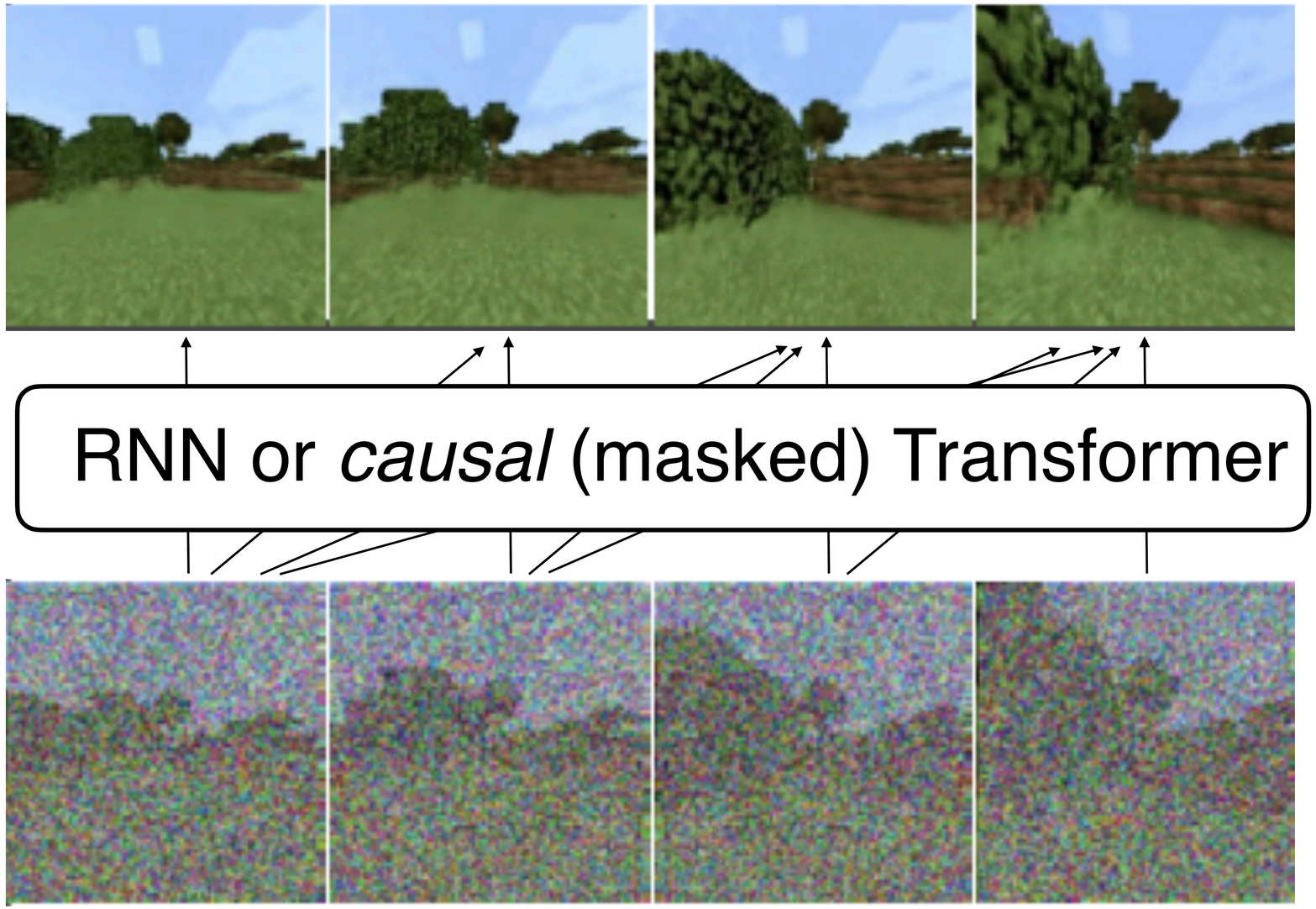


Example: RNN Implements Full-Seq. Diffusion

Models $p(x_1, x_2, x_3, x_4)$



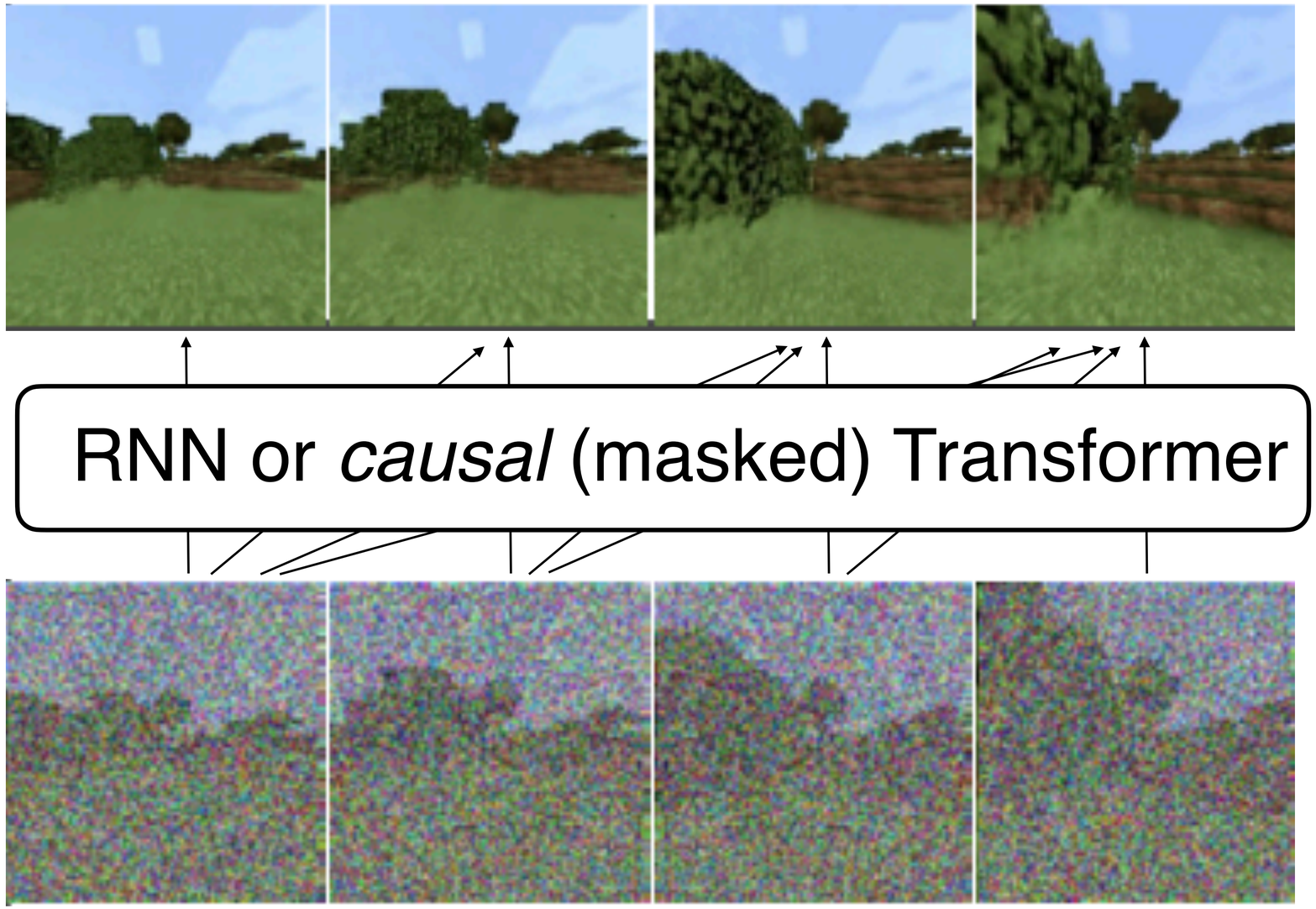
Example: RNN Implements Full-Seq. Diffusion



Models $p(x_1, x_2, x_3, x_4)$

By design a causal factorization

Example: RNN Implements Full-Seq. Diffusion

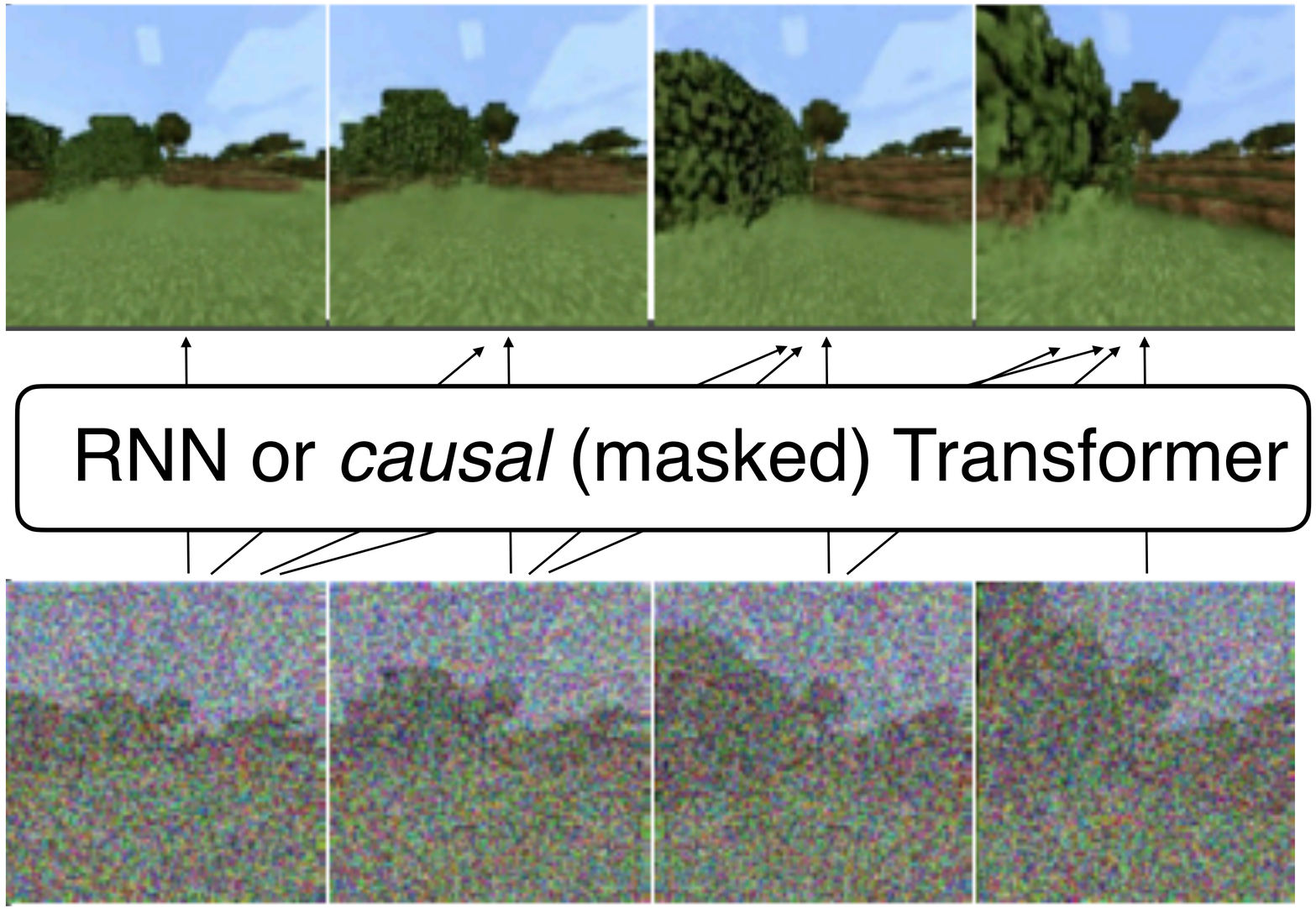


Models $p(x_1, x_2, x_3, x_4)$

By design a causal factorization

Benefit: Can be faster & cheaper (fewer flops per denoising step)

Example: RNN Implements Full-Seq. Diffusion



Models $p(x_1, x_2, x_3, x_4)$

By design a causal factorization

Benefit: Can be faster & cheaper (fewer flops per denoising step)

But hard to train in practice…

From Slow Bidirectional to Fast Autoregressive Video Diffusion Models

Tianwei Yin[1*] Qiang Zhang[2*] Richard Zhang[2]
William T. Freeman[1] Frédo Durand[1] Eli Shechtman[2] Xun Huang[2]

[1]MIT [2]Adobe

https://causvid.github.io/

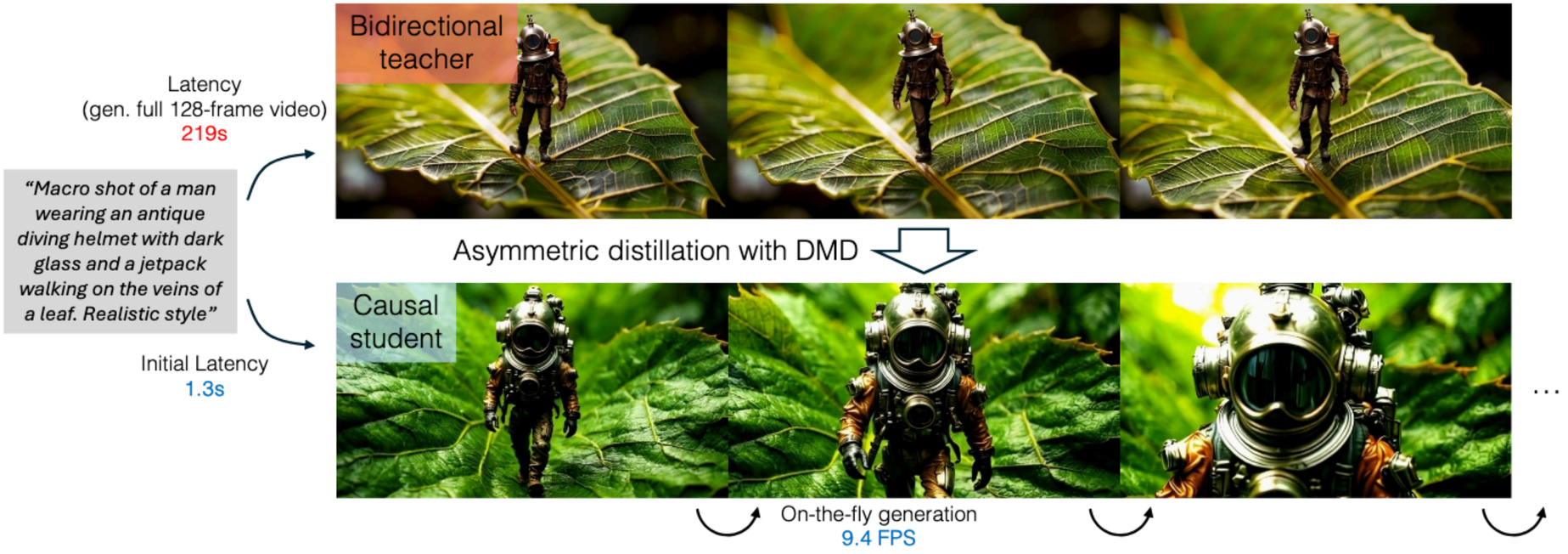
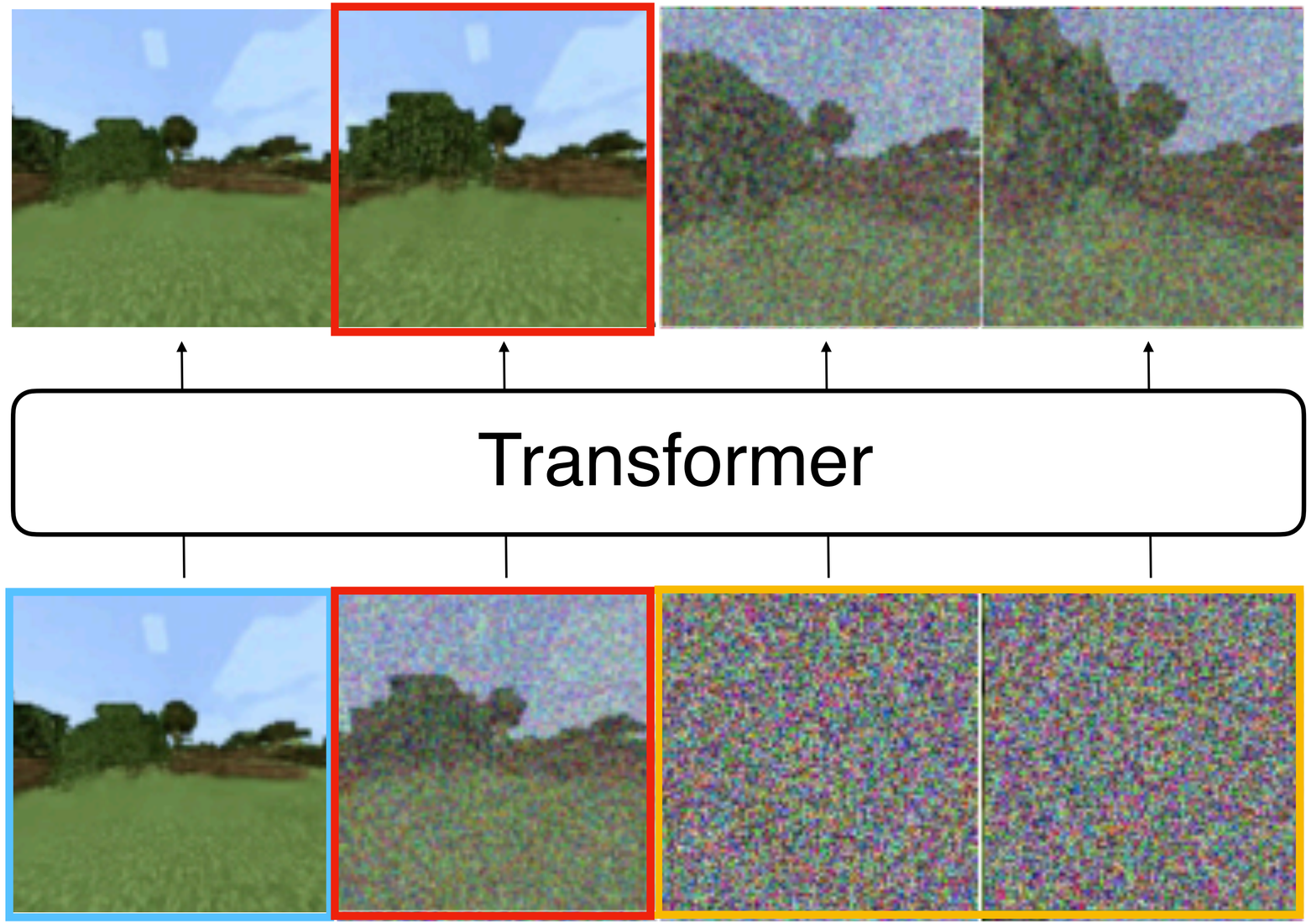


Figure 1. Traditional bidirectional diffusion models (top) deliver high-quality outputs but suffer from significant latency, taking 219 seconds to generate a 128-frame video. Users must wait for the entire sequence to complete before viewing any results. In contrast, we distill the bidirectional diffusion model into a few-step autoregressive generator (bottom), dramatically reducing computational overhead. Our model (**CausVid**) achieves an initial latency of only 1.3 seconds, after which frames are generated continuously in a streaming fashion at approximately 9.4 FPS, facilitating interactive workflows for video content creation.

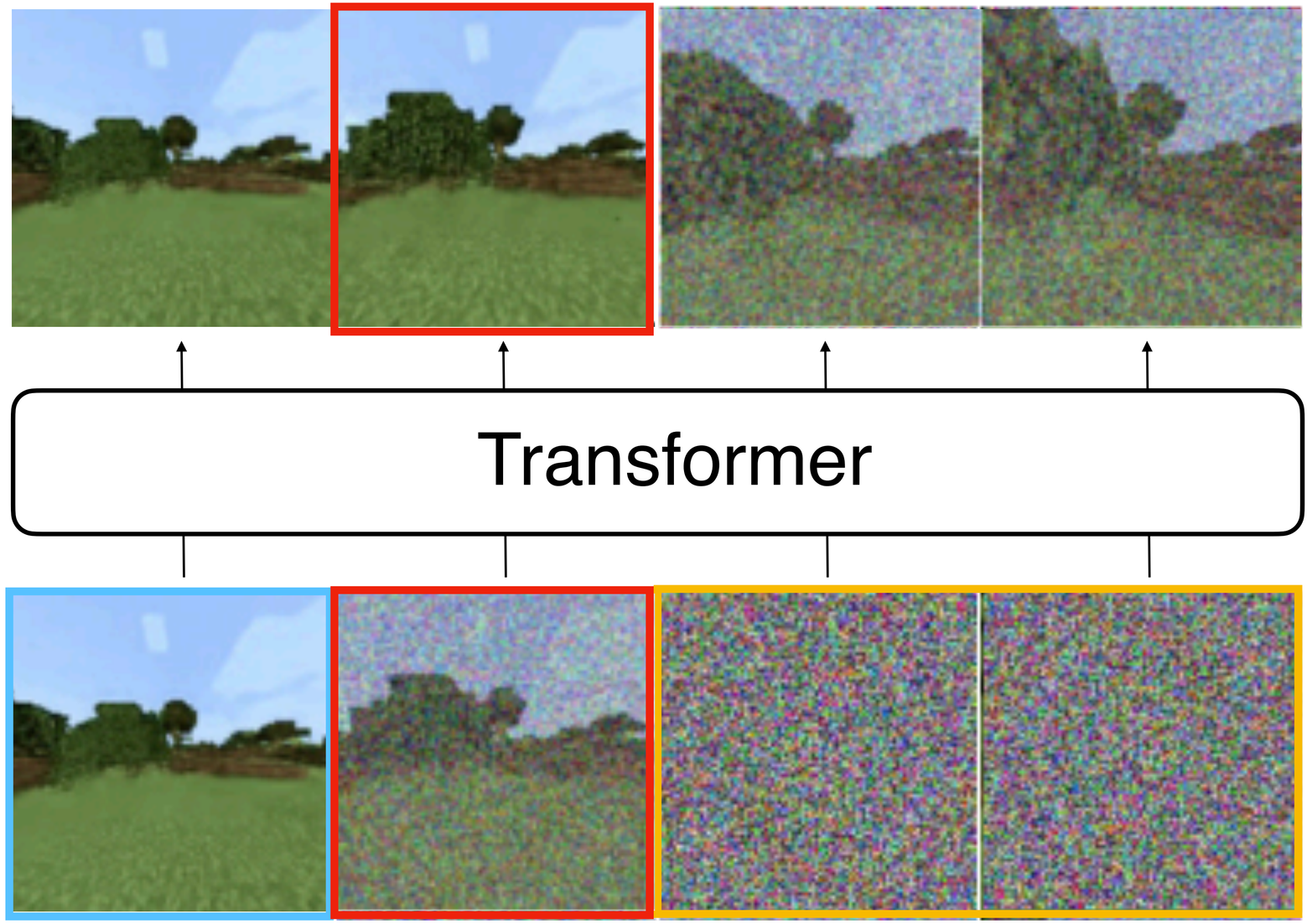
Diffusion Language Models - the next generation?

Diffusion Language Models - the next generation?

Key Takeaway: Controlling conditionals via masking in sequence generation

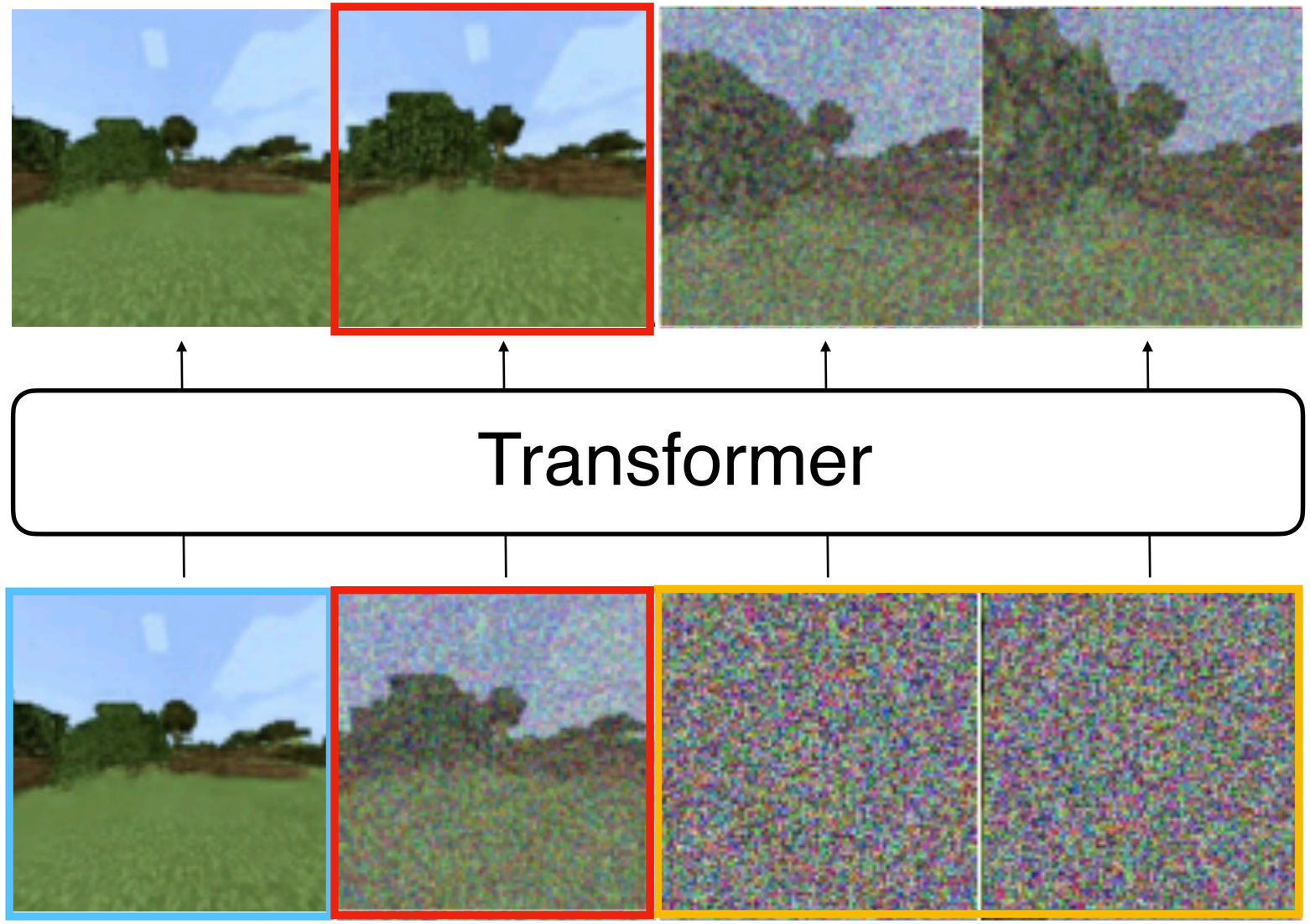


Key Takeaway: Controlling conditionals via masking in sequence generation



Time dimension
is unlike spatial ones

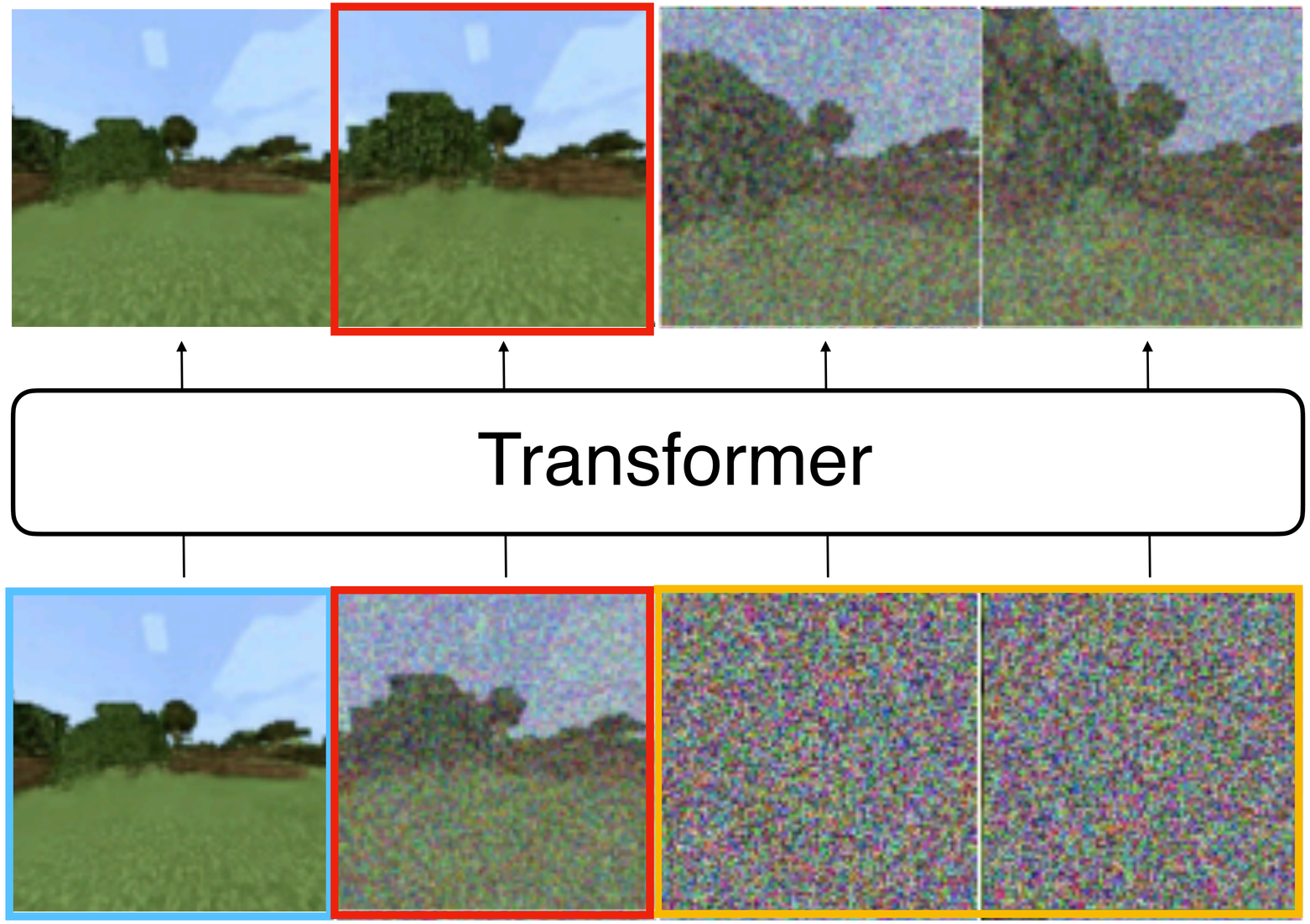
Key Takeaway: Controlling conditionals via masking in sequence generation



Time dimension
is unlike spatial ones

Generating sequences
requires carefully
controlling what we
condition on

Key Takeaway: Controlling conditionals via masking in sequence generation

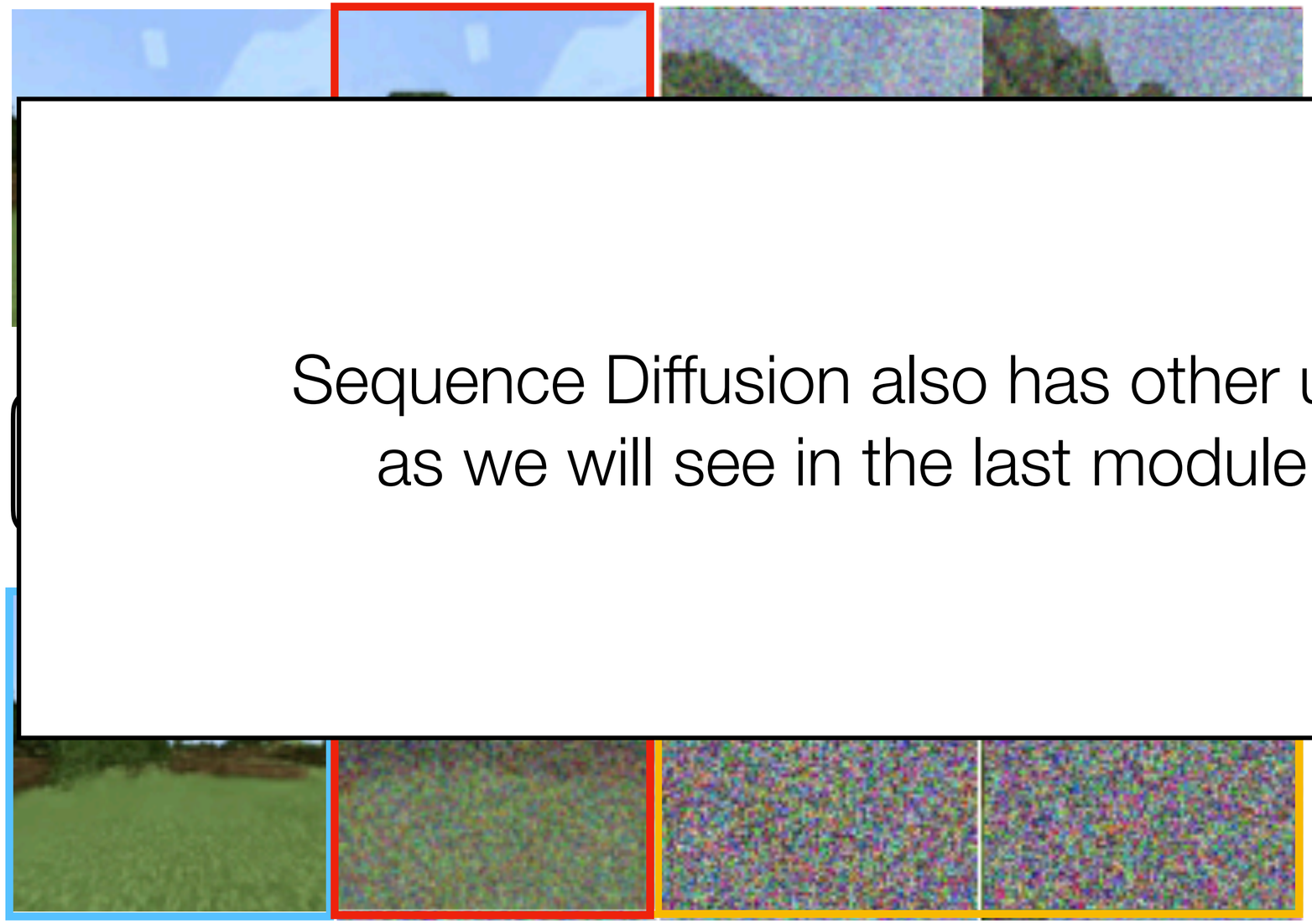


Time dimension is unlike spatial ones

Generating sequences requires carefully controlling what we condition on

“Noise as masking” + Diffusion is a straightforward way of doing so

Key Takeaway: Controlling conditionals via masking in sequence generation



Time dimension
is unlike spatial ones

s

straightforward way of
doing so

Sequence Diffusion also has other unique benefits,
as we will see in the last module of the class!