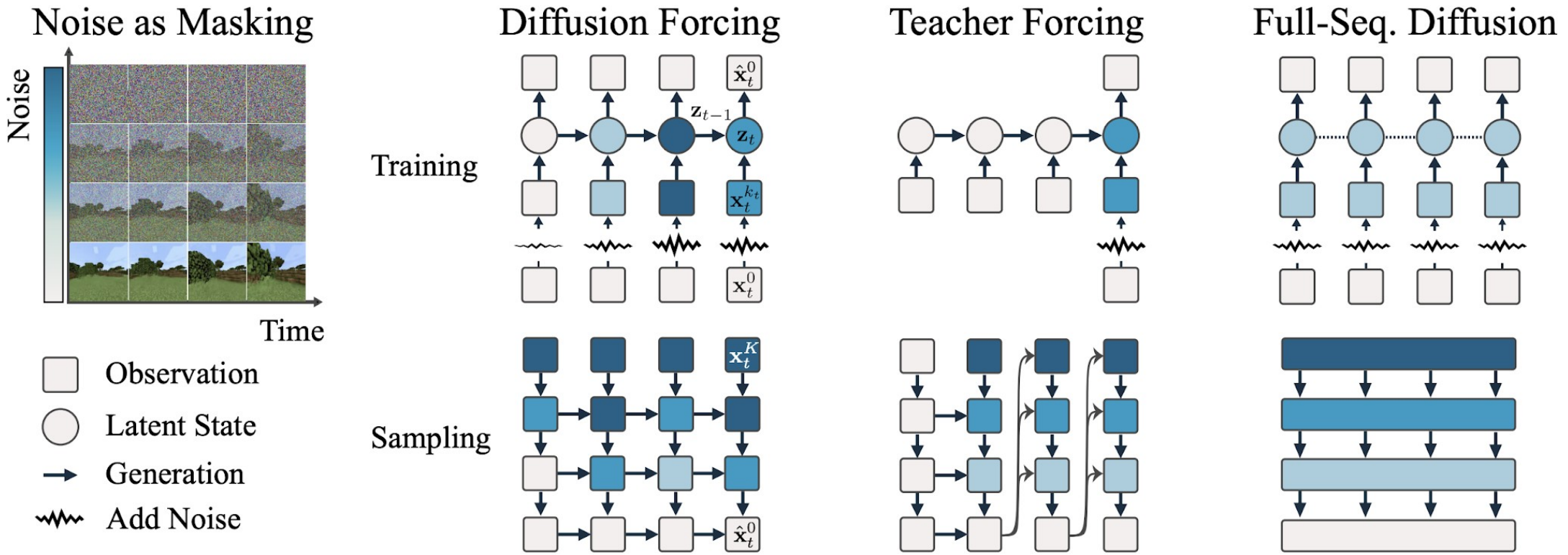
Diffusion Forcing

Next-token Prediction Meets Full-Sequence Diffusion



Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, Vincent Sitzmann

Presenter: Yidong Huang

The high-level problem: streaming vs. global consistency

We want a world model that can generate the next frame now — without losing consistency over a long future.

Autoregressive models

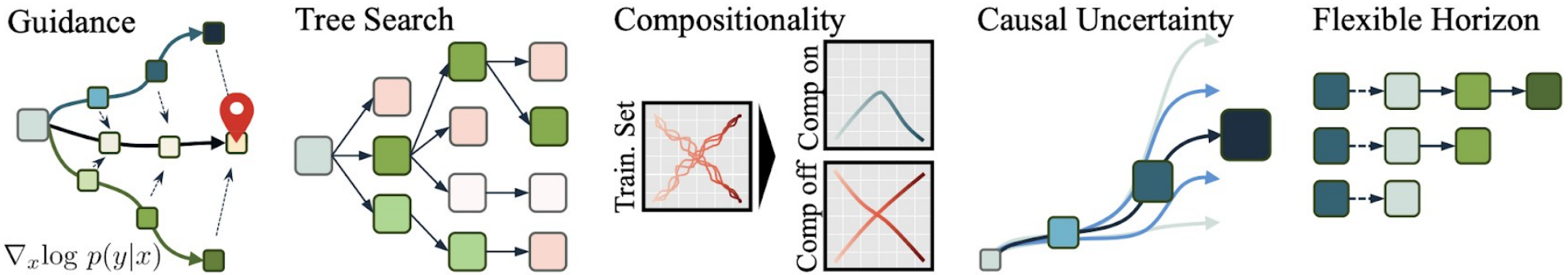
Can stream and react to new observations, but small prediction errors accumulate.

Full-sequence diffusion

Samples a coherent trajectory jointly, but usually commits to a fixed horizon.

Diffusion Forcing: independently noise each time step → choose the schedule at inference.

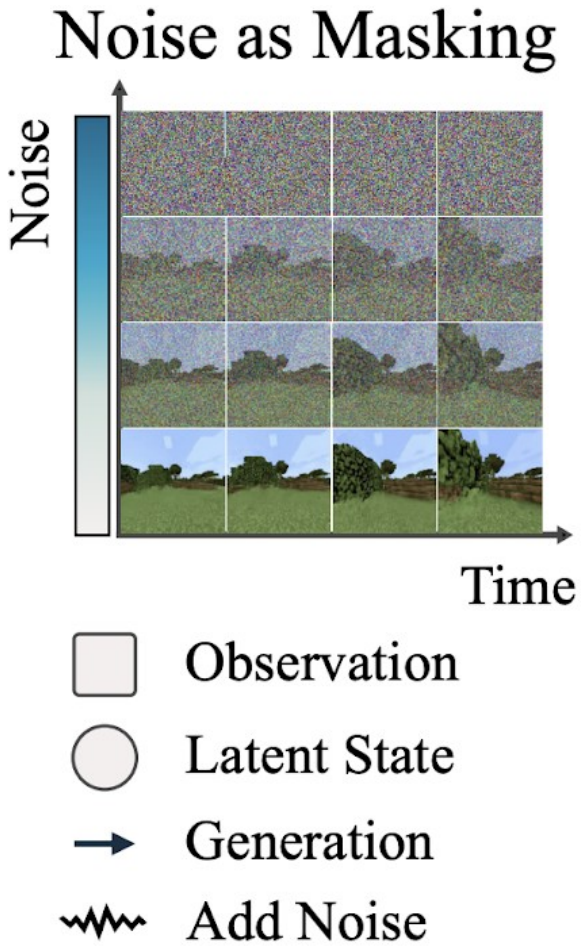
Motivation: neither paradigm has all the desired properties



	Guidance	Tree Search	Compositionality	Causal Uncertainty	Flexible Horizon
Teacher Forcing	✗	✓	✓	✓	✓
Full-Seq. Diffusion	✓	✗	✗	✗	✗
Diffusion Forcing	✓	✓	✓	✓	✓

- Autoregressive: flexible rollout, but errors can compound.
- Full-sequence diffusion: coherent joint sampling, but less naturally streaming.

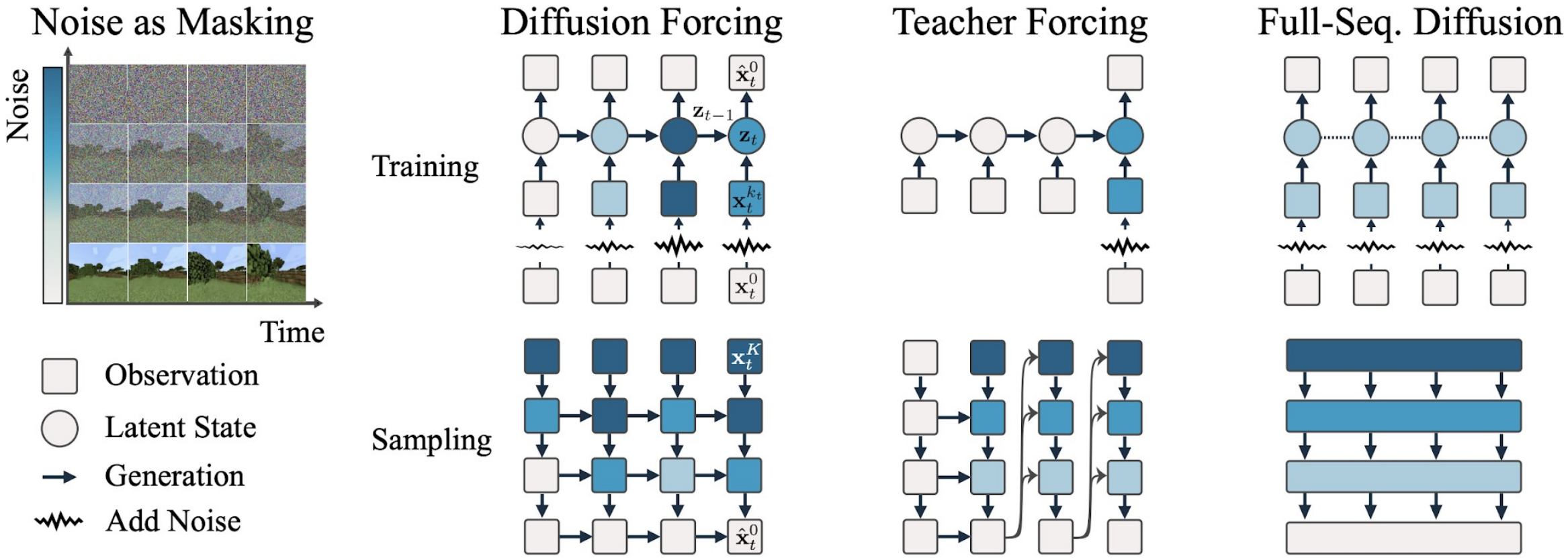
Core intuition: diffusion noise is a soft mask



- Noise level k = 0: token is fully observed.
- Intermediate k: token is partially corrupted.
- Maximal k = K: token is nearly pure noise.
- Give each time step its own independent kt.

Chen et al., NeurIPS 2024 • Fig. 2 (left)

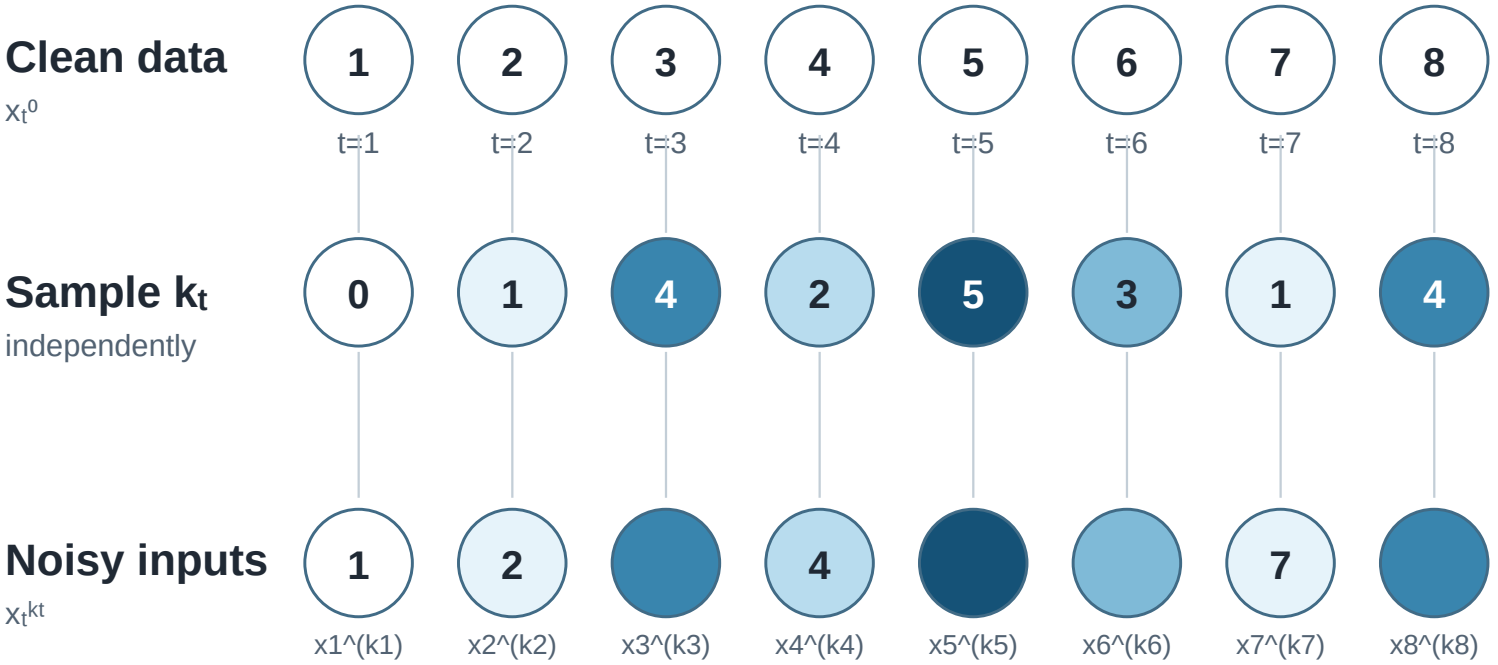
Method: causal dynamics with per-token denoising



One trained model can emulate AR, full-sequence diffusion, or hybrids at inference.

Training (1/2): independently noise each time step

One sequence, many noise levels



Different noise amounts at different temporal positions.

NOISE LEVEL ○ clean ◔ partially noisy ● near random

1. Sample a sequence

Draw x_1:T from the training dataset.

2. Choose noise levels

For each t, independently sample $k_t \in \{0, \ldots, K\}$.

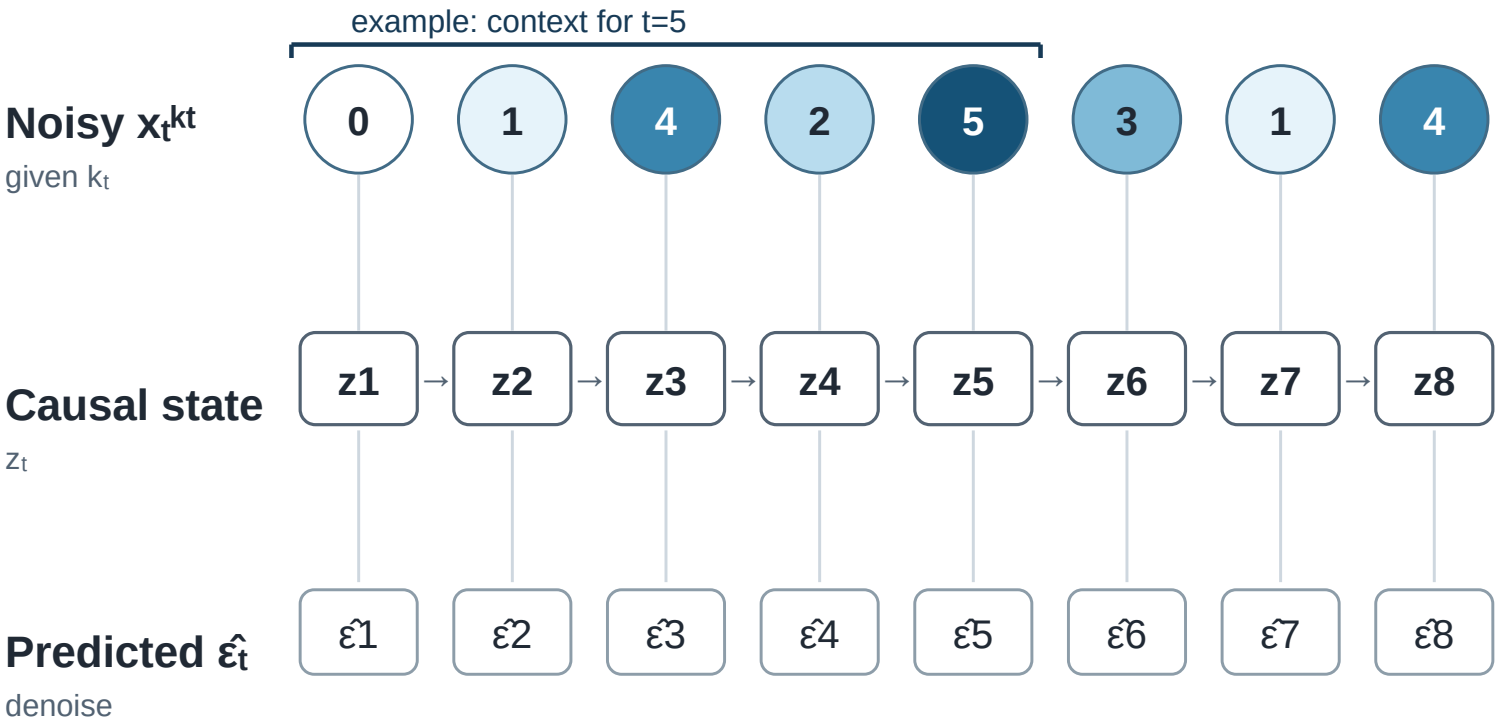
3. Forward diffuse

Generate x_t^{kt} from x_t^0 using its sampled k_t.

Illustrative dots adapted from Chen et al., NeurIPS 2024 • Fig. 2 and Algorithm 1

Training (2/2): causal denoiser and one gradient update

The same noisy sequence now enters a causal recurrent model



$$\text{Loss} = \Sigma_t \, \| \, \varepsilon_t - \hat{\varepsilon}_t \, \|^2$$

→ one backprop update to θ

NOISE LEVEL ○ clean ◔ partially noisy ● near random

1. Update memory

z_t uses previous state z_{t-1} and current noisy x_t^{kt}.

2. Predict noise

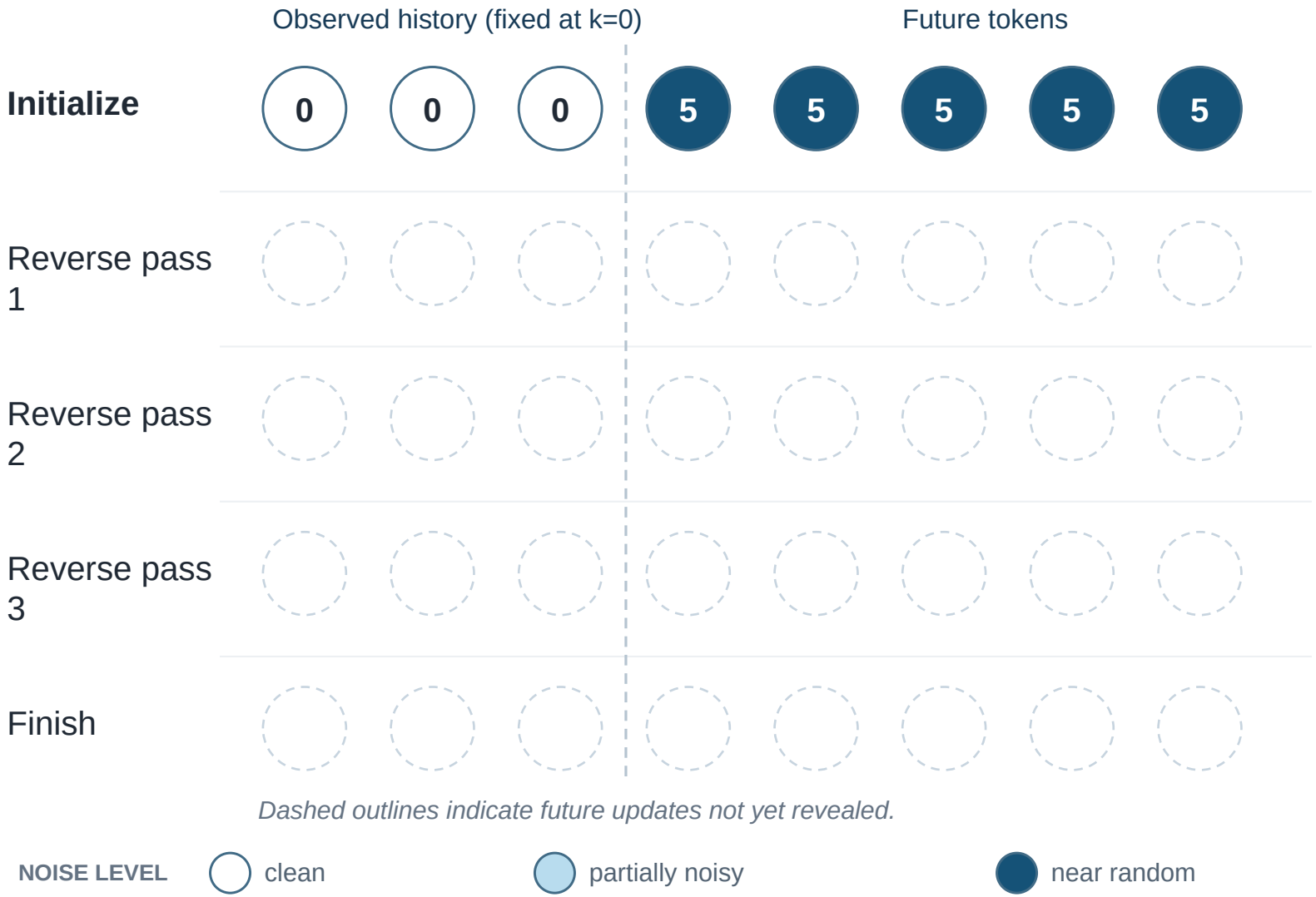
The denoising head estimates $\hat{\varepsilon}_t$, conditioned on the causal prefix.

3. Compute MSE

Compare $\hat{\varepsilon}_t$ against actual ε_t, sum over t, backprop.

Illustrative dots adapted from Chen et al., NeurIPS 2024 • Fig. 2 and Algorithm 1

Sampling (1/2): initialize a noise-level schedule



Dashed outlines indicate future updates not yet revealed.

NOISE LEVEL clean · partially noisy · near random

1. Fix observations

History is already known (k=0, white).

2. Start future noisy

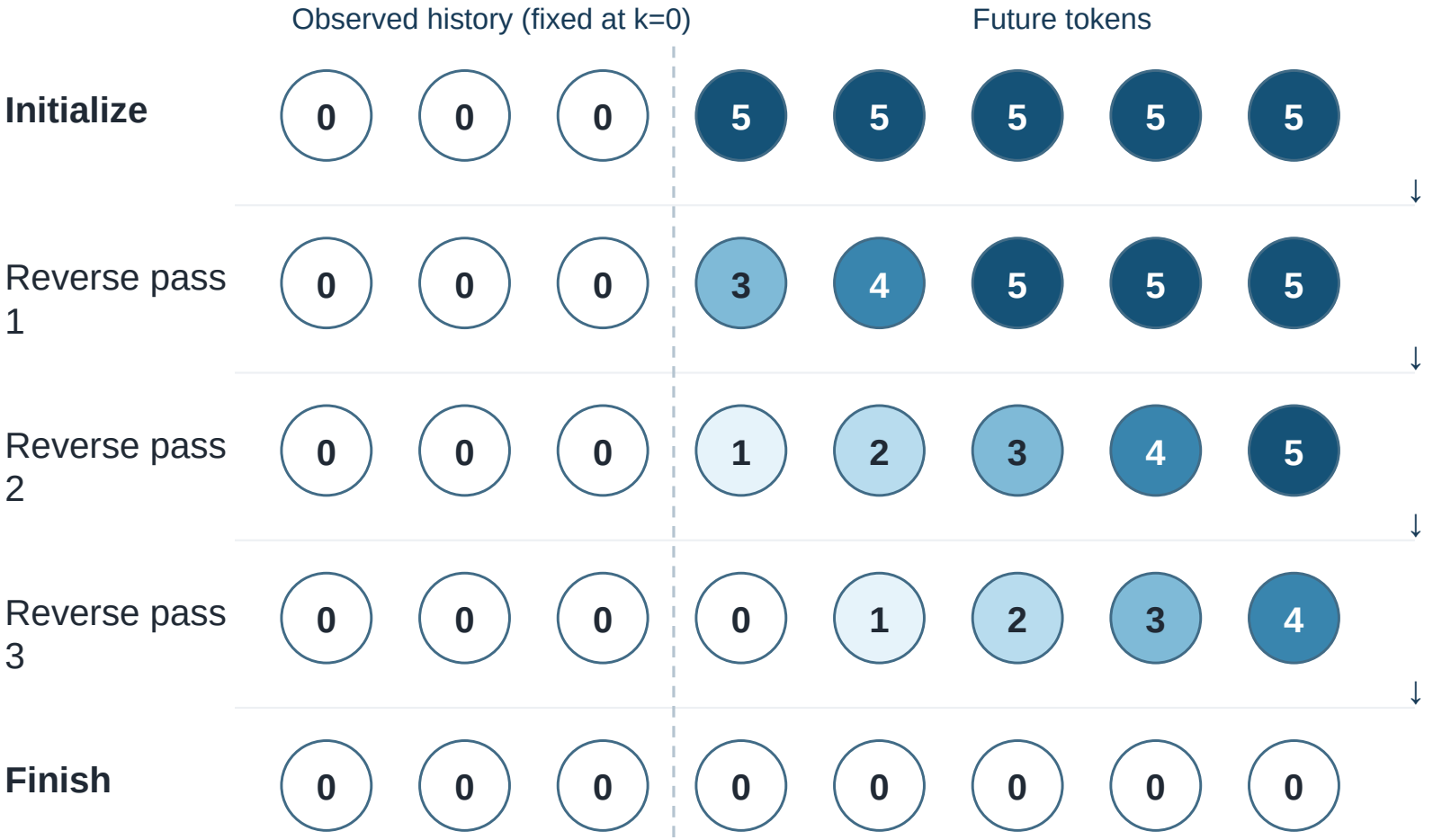
Unknown future positions begin at k=K (dark blue).

3. Choose schedule

A 2D matrix sets the noise level per position after each reverse pass.

Schematic interpretation of Chen et al., NeurIPS 2024 • Algorithm 2

Sampling (2/2): denoise left-to-right, then repeat



Illustrative schedule (k = 0,…,5); the paper permits other matrices.

NOISE LEVEL ○ clean ◔ partially noisy ● near random

1. Causal sweep

For each schedule row, update tokens from left to right.

2. Refine gradually

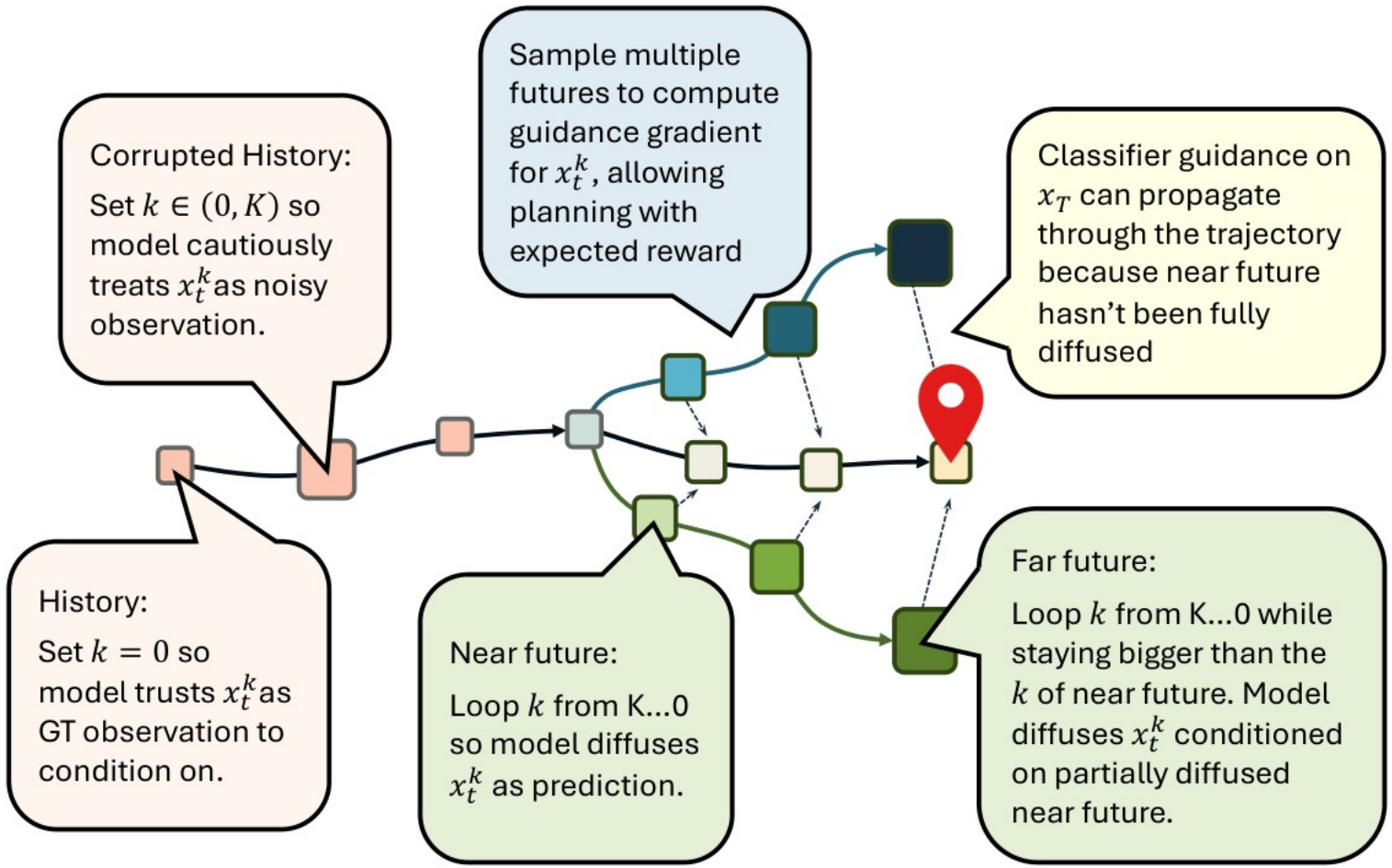
Near-future positions become clean before distant ones.

3. Finish at k=0

Repeat until all generated tokens are clean.

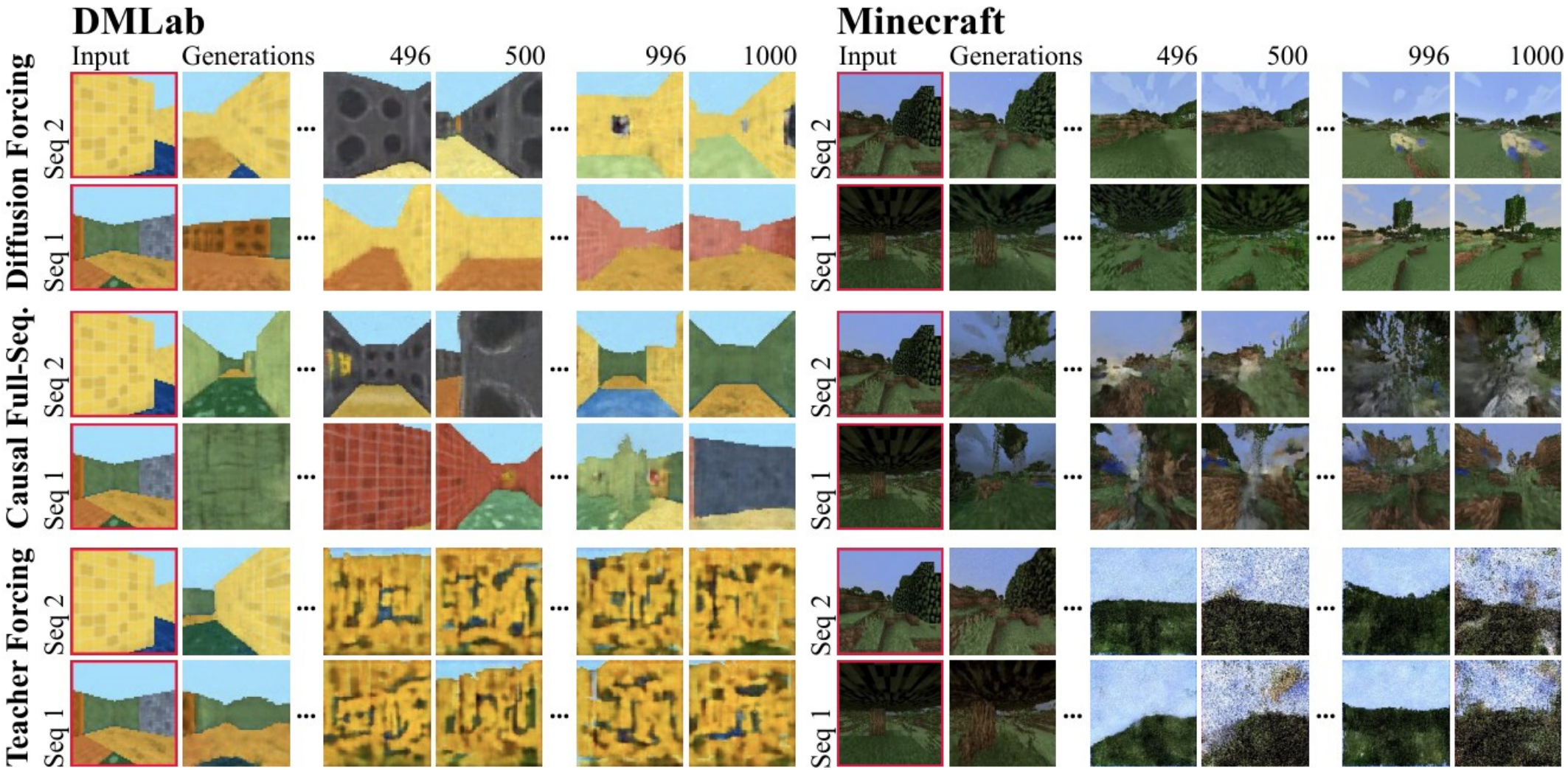
Schematic interpretation of Chen et al., NeurIPS 2024 • Algorithm 2

Why a causal schedule helps: uncertainty grows into the future

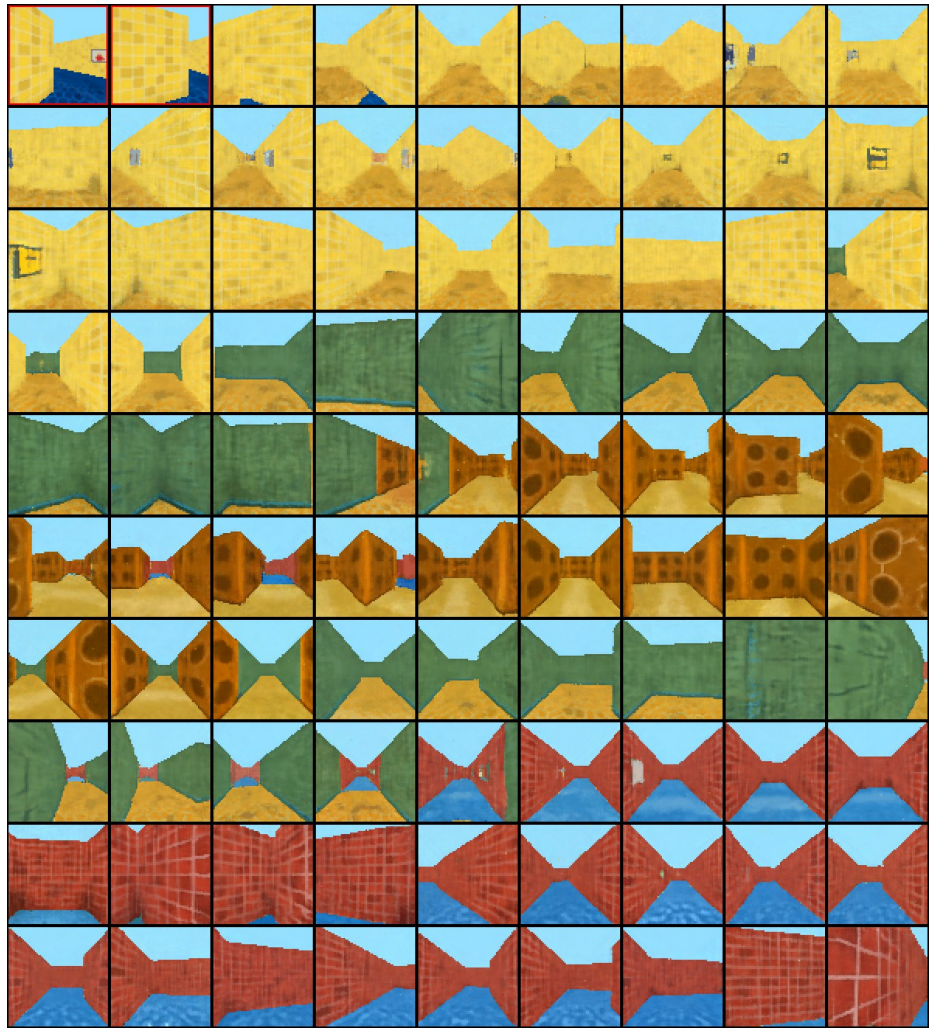


- Near future can be denoised more precisely.
- Far future stays uncertain and is sampled repeatedly.
- Monte Carlo Guidance (MCG) estimates expected future reward.

Video prediction: much more stable than the baselines



Long-horizon rollout: 2,000 frames from short training clips

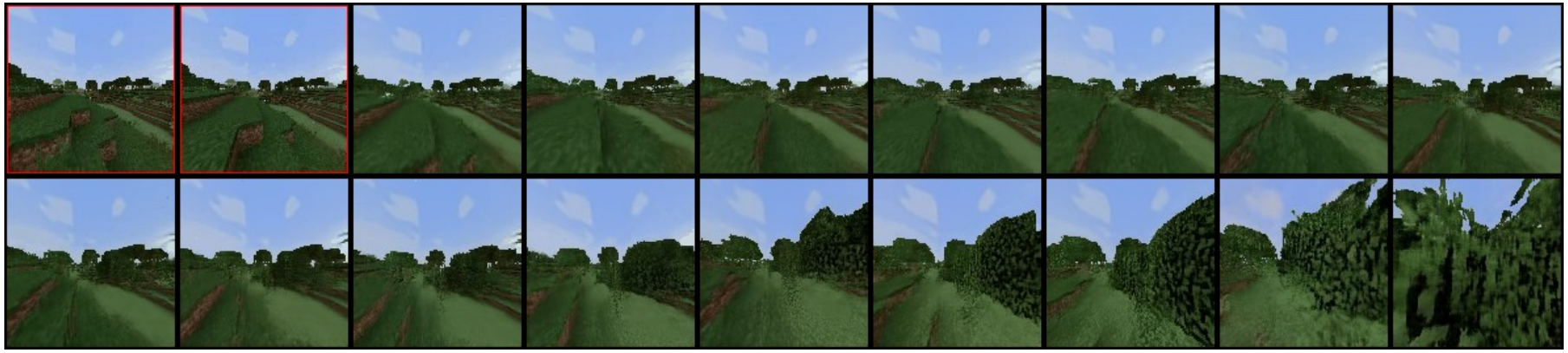


- DMLab: trained on 36-frame sequences.
- Rolls out 2,000+ frames without RNN-state reset or sliding window.
- Original DMLab image resolution: 64 × 64.

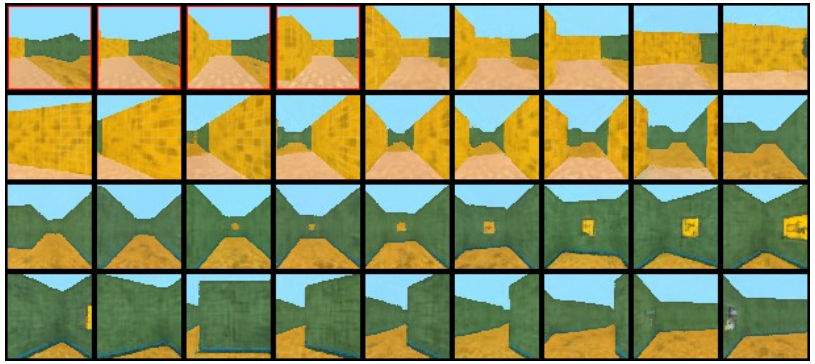
Chen et al., NeurIPS 2024 • original author project visual

Additional qualitative evidence: Minecraft and DMLab

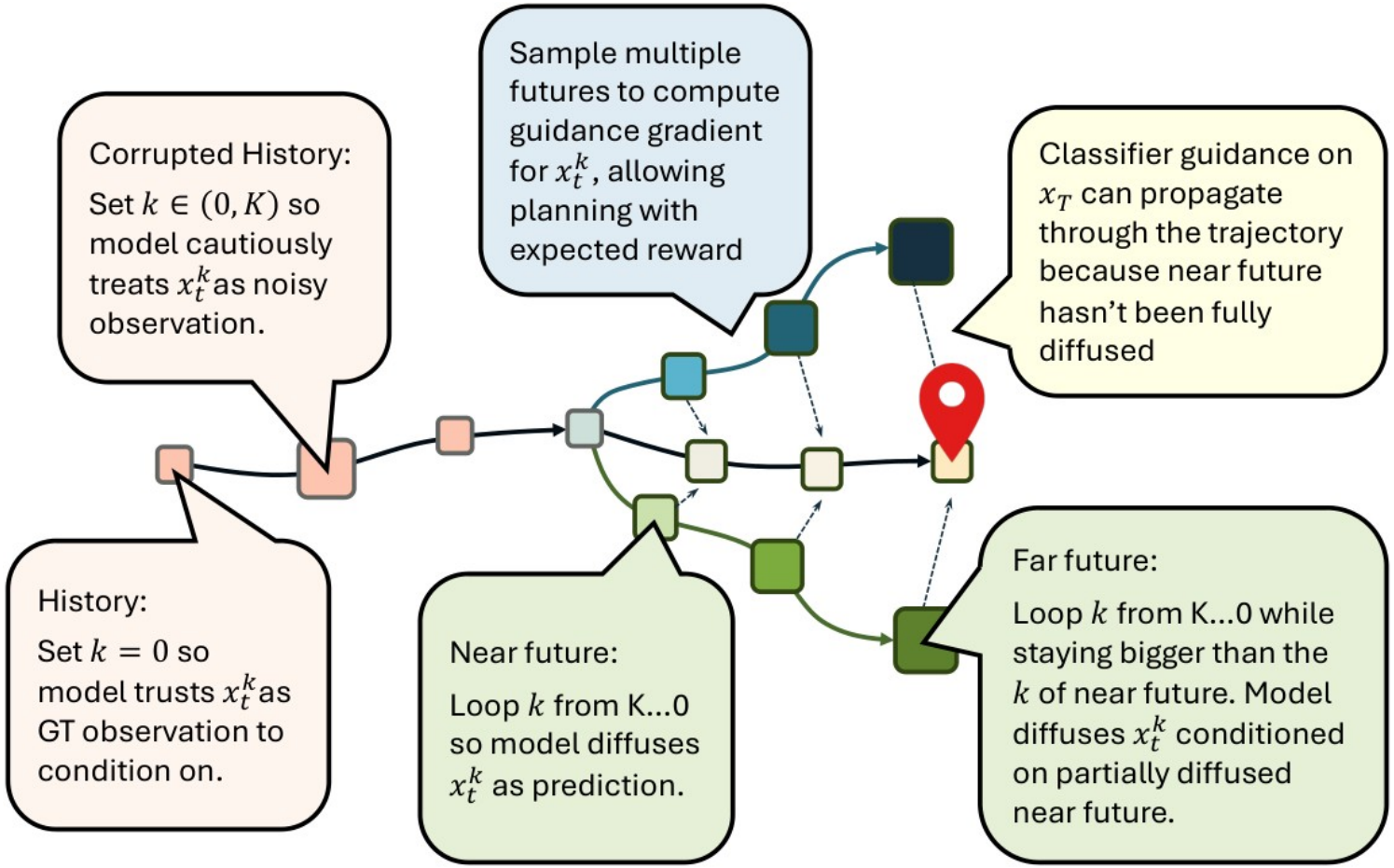
Minecraft



DMLab

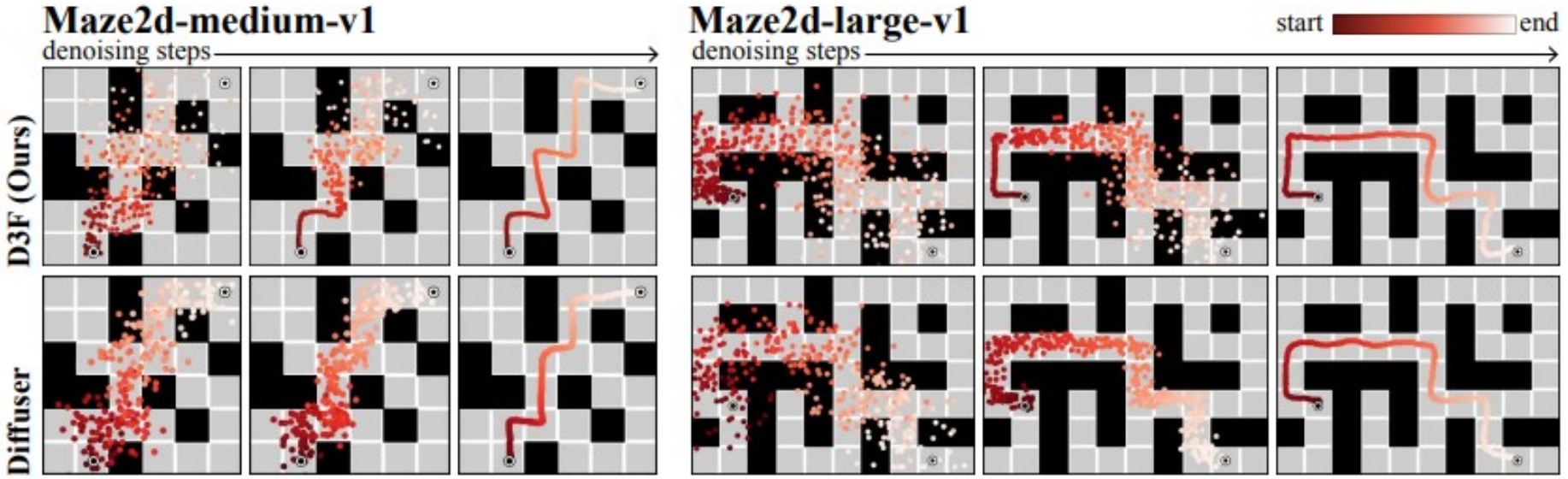


Planning: MCG operates on expected future reward



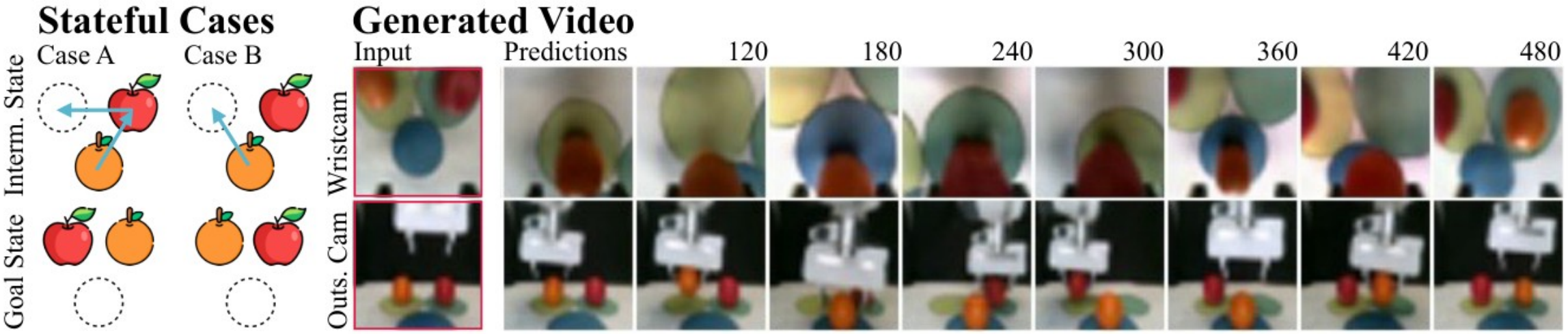
- Condition on history and plausible futures.
- Branch to sample multiple stochastic outcomes.
- Average the guidance signal over future samples.

Planning evaluation: a benefit beyond prettier videos



Environment		MPPI	CQL	IQL	Diffuser*	Diffuser w/ diffused action	Ours wo/ MCG	Ours
Maze2D	U-Maze	33.2	5.7	47.4	113.9 ± 3.1	6.3 ± 2.1	110.1 ± 3.9	**116.7 ± 2.0**
Maze2D	Medium	10.2	5.0	34.9	121.5 ± 2.7	13.5±2.3	136.1 ± 10.2	**149.4 ± 7.5**
Maze2D	Large	5.1	12.5	58.6	123.0 ± 6.4	6.3 ±2.1	142.8 ± 5.6	**159.0 ± 2.7**
Single-task Average		16.2	7.7	47.0	119.5	8.7	129.67	**141.7**
Multi2D	U-Maze	41.2	-	24.8	128.9 ± 1.8	32.8±1.7	107.7 ± 4.9	**119.1 ± 4.0**
Multi2D	Medium	15.4	-	12.1	127.2 ± 3.4	22.0±2.7	145.6 ± 6.5	**152.3 ± 9.9**
Multi2D	Large	8.0	-	13.9	132.1 ± 5.8	6.9 ±1.7	129.8 ± 1.5	**167.1 ±2.7**
Multi-task Average		21.5	-	16.9	129.4	20.6	127.7	**146.2**

Real robot: long-horizon memory + robustness



Franka fruit-swapping task

- 80% success in the base task, where history is required.
- 76% with noisy / obstructed observations (4-point drop).
- Next-frame diffusion baseline: 48% under distraction.

Strengths and limitations

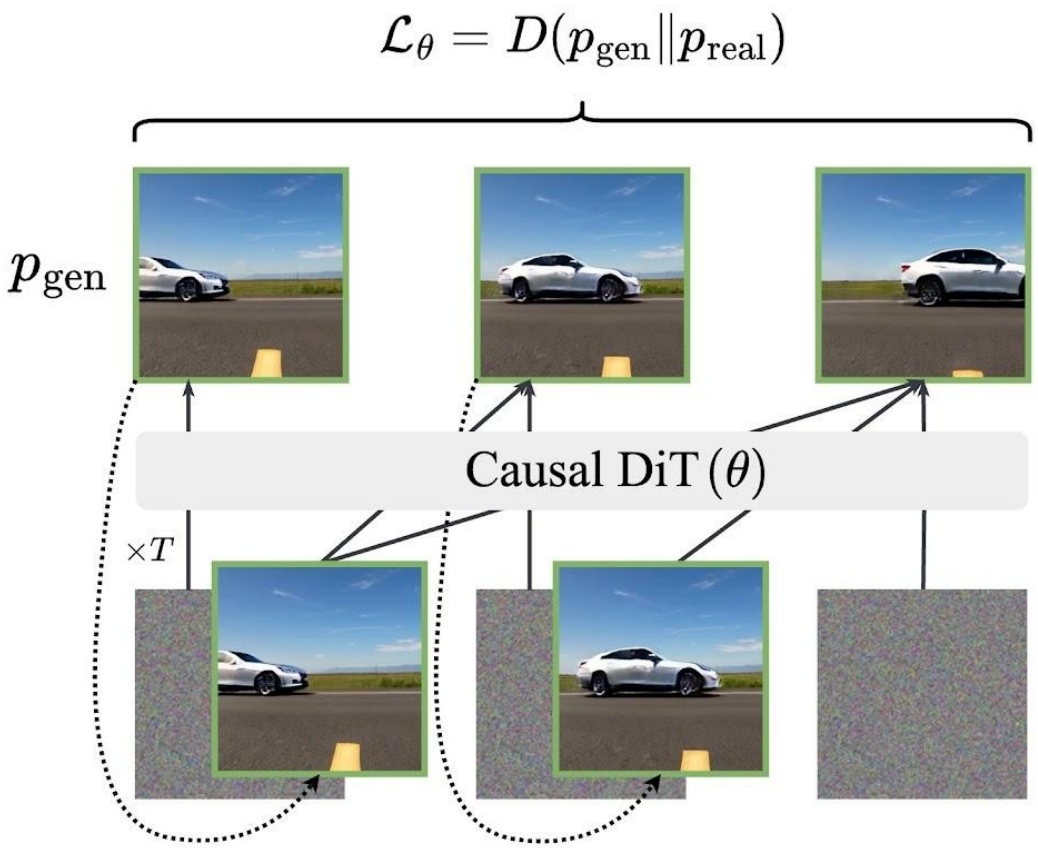
Strengths

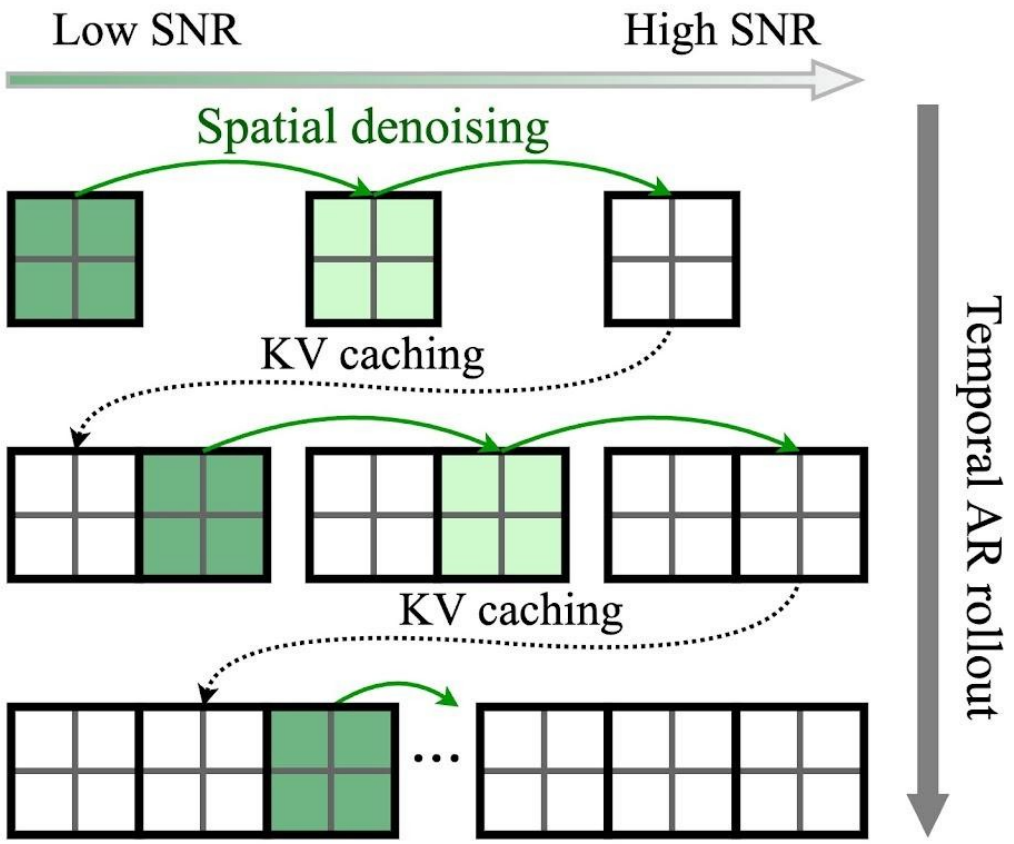
- Independent per-frame noise levels offer flexible inference schedules.
- Causal generation supports long-horizon video, planning, and feedback.

Limitations

- Training uses noise-corrupted data histories, not self-generated rollouts.
- Long trajectories can still accumulate identity and appearance drift.

Follow-up: Self Forcing closes the train–test gap

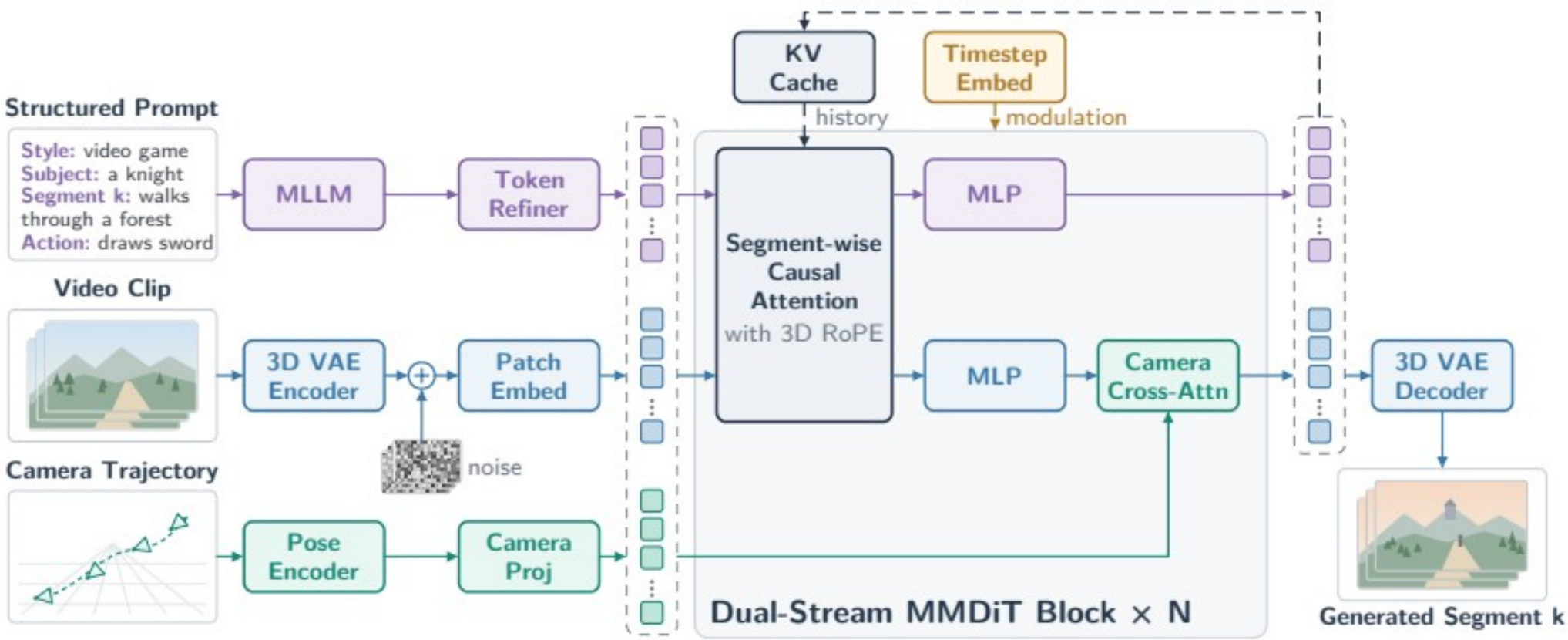




Diffusion Forcing: noisy data-derived history

Self Forcing: self-generated rollout + video-level matching

Follow-up: Scaling Forcing to Large Diffusion Models



HY Nomas: 8B World Model

Discussion: what should the next model learn?

- Long-horizon consistency: How can video world models maintain persistent memory of objects and scenes, even after they disappear from view for hundreds of frames?
- Error Accumulation: Can self-generated rollout training truly eliminate long-horizon drift, or do we need explicit mechanisms to correct accumulated errors?
- Causal vs. Bidirectional Generation: Can we combine the global consistency of bidirectional diffusion with the low latency of autoregressive generation in a single model?
- Learning vs. Inference: Should long-horizon consistency be learned during training, or enforced through memory, guidance, and verification at inference time?