1. No in-class course Thursday 10/29 due to unavailability of the classroom; Zoom link will be posted.
2. Class 11/24 will be an invited talk by Dr. Randall Balestriero



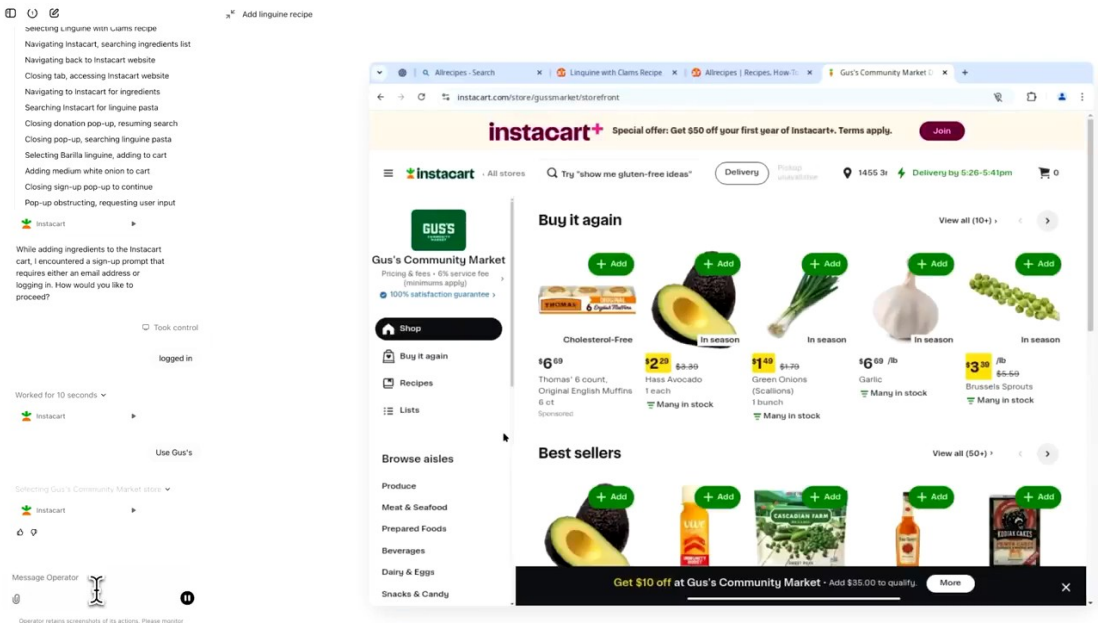
Are Normalizing Flows Good Candidates for World Models?

Jiatao Gu

University of Pennsylvania · AMI Labs

Agents are changing our world

Digital agents



Operator · Computer use

Physical agents



Gemini Robotics · Vision–language–action

A language model controls a real robot

Task: place the red block inside the bowl



GPT-6 Astra with camera observations and a robot tool interface

Robocurve, September 2026 / Motion only. Inference pauses omitted. Click to play.

What happens between two motions?

The visible output is a structured tool call

Camera images + robot state



Goal and recent interaction

GPT-6
Astra

Internal reasoning
is not exposed

One tool call

Public action note

Current assessment + reason for moving

Motion target

Position, orientation or gripper setting

IK + motion controller execute the target

New observations return for the next call

Robocurve, September 2026 / Interface simplified from the original transcript

Before the robot moves, what does it expect?

Step 37: the fingers have closed around the block



Current assessment

Partial closure suggests contact.

Reason for the action

Lift to test whether the grasp holds.

Expected observation

The block should rise with the gripper.

An action, an expected change, and a check

Expected consequence → Execute action → Observe the result

Before release: step 73



Intention: release
the block into the bowl

Open the gripper

left_gripper = 1

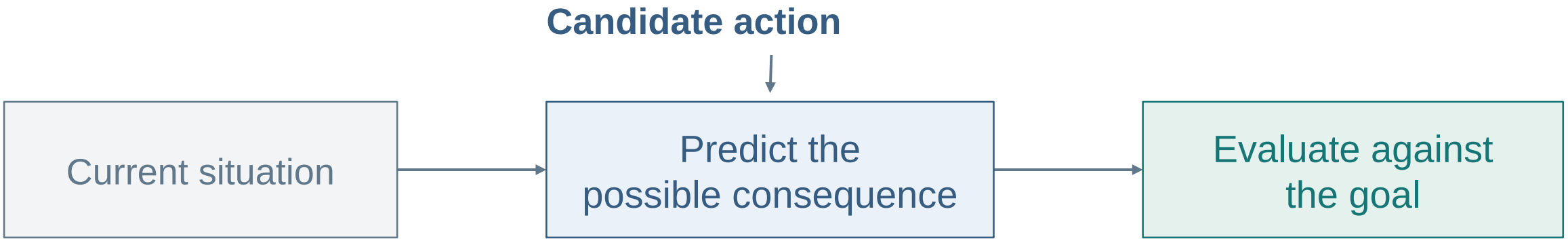
After release: step 79



Observation: block
rests inside the bowl

Can predictions help choose the next action?

“What should I do?” depends on “What might happen?”



World modeling

Anticipate what may change

Planning

Use predictions to help choose actions

The goal: learn predictions that are useful for decisions.

Functional decomposition for this talk. Feedback can check and revise expectations.

World models: three roles

Simulator for learning

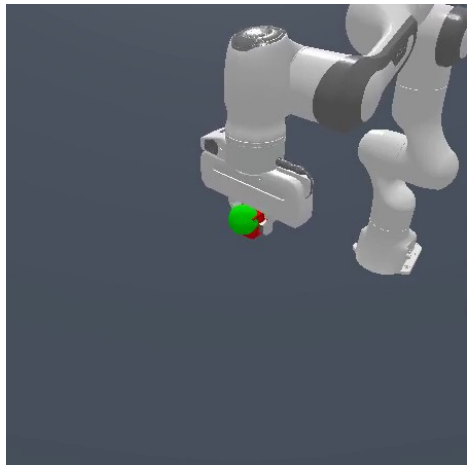
Learn a policy in the model



Policy deployed in VizDoom
Trained inside a learned model

Predictor for planning

Anticipate action consequences



TD-MPC2 · online planning
Simulation execution

World generator

Create an interactive world



Genie 3
Generated interactive world

Today’s focus

Ha & Schmidhuber, 2018 / Hansen et al., TD-MPC2, 2024 / Google DeepMind, Genie 3, 2025. Click to play.

Anticipating consequences before acting

Kenneth Craik · 1943



“a ‘small-scale model’ of external reality and of its own possible actions within its head”

Yann LeCun · 2026

12-second video

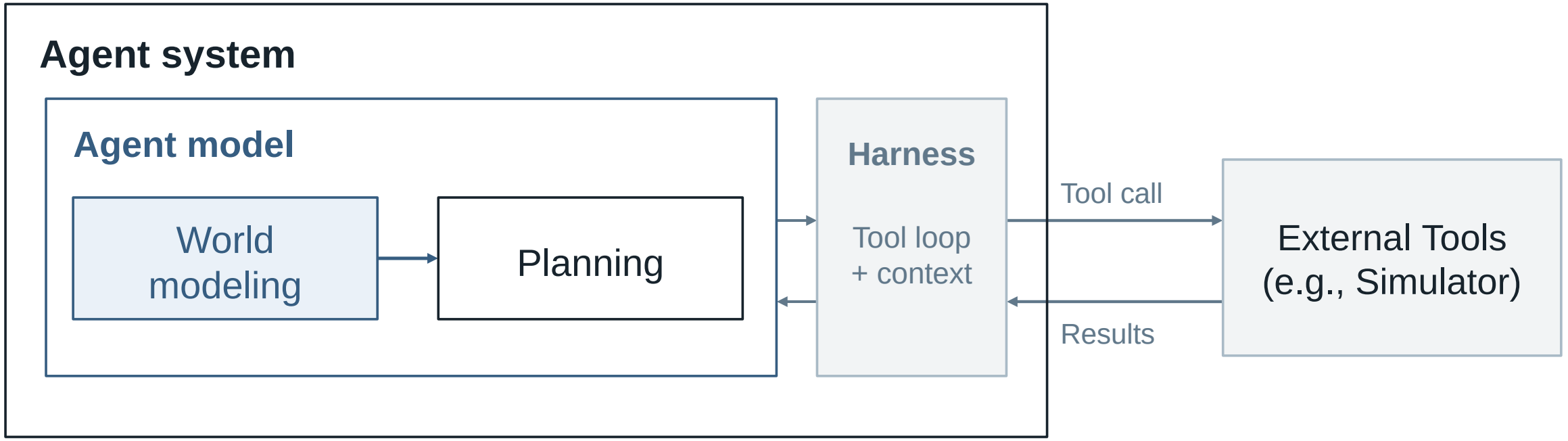


“…allows an agentic system to anticipate the consequences of its own actions.”

Craik, 1943, p. 61 / LeCun, Unsupervised Learning, May 2026, 09:10–09:22

What is a world model?

A world model predicts how the world may evolve.
Planning uses these predictions to choose actions.



World modeling as dynamics prediction

Predict the next observation under a candidate action (if given)

$$p_\theta(x_{t+1} \mid x_{\leq t}, a_{\leq t})$$



x: observation or learned representation a: action

Planning compares futures under different action choices.

A language model can predict symbolic consequences

Goal: orange on red, and red on blue.

Initial state: orange on yellow; red and blue on the table.

PREDICTED STATES

Explicit text-state prediction + tree search

Orange onto red first → red becomes blocked.

Red onto blue first → red stays clear.

Then orange onto red → both goals hold.

SELECTED PLAN

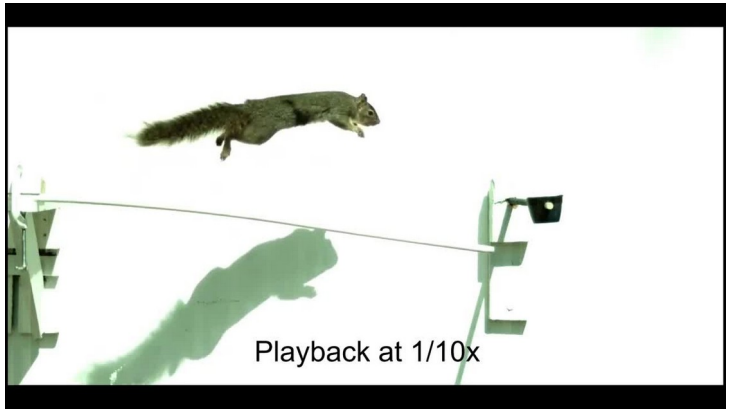
Actions chosen by the planner

1. Pick up red
2. Stack red on blue
3. Unstack orange from yellow
4. Stack orange on red

RAP system: LLM action proposals + state predictions + MCTS

Hao et al., RAP, 2023, Fig. 4

What do animals understand about the world?







How do animals learn consequences from observation and action?

Hunt et al. 2021 / UC Berkeley / BBC Earth / Adolph / Science Friday

What should prediction learn from these examples?

1. Learn from natural signals

Video: motion and persistence
Audio: impact and material cues
Interaction: actions and changes

2. Represent more than words

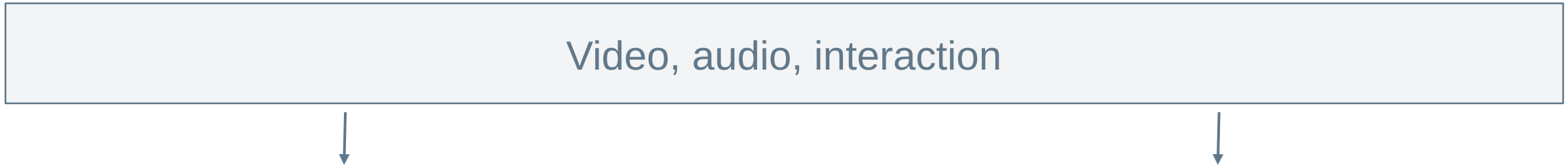
Pose and velocity
Contact and occlusion
Deformation and geometry

These quantities need not be described in language.

How to encode them is a separate modeling choice.

LeCun, 2022 / Owens et al., Visually Indicated Sounds, 2016.

Non-textual prediction: two valid modeling choices



Quantized tokens

Finite vocabulary
Categorical prediction

Autoregressive token models

Numerical representations

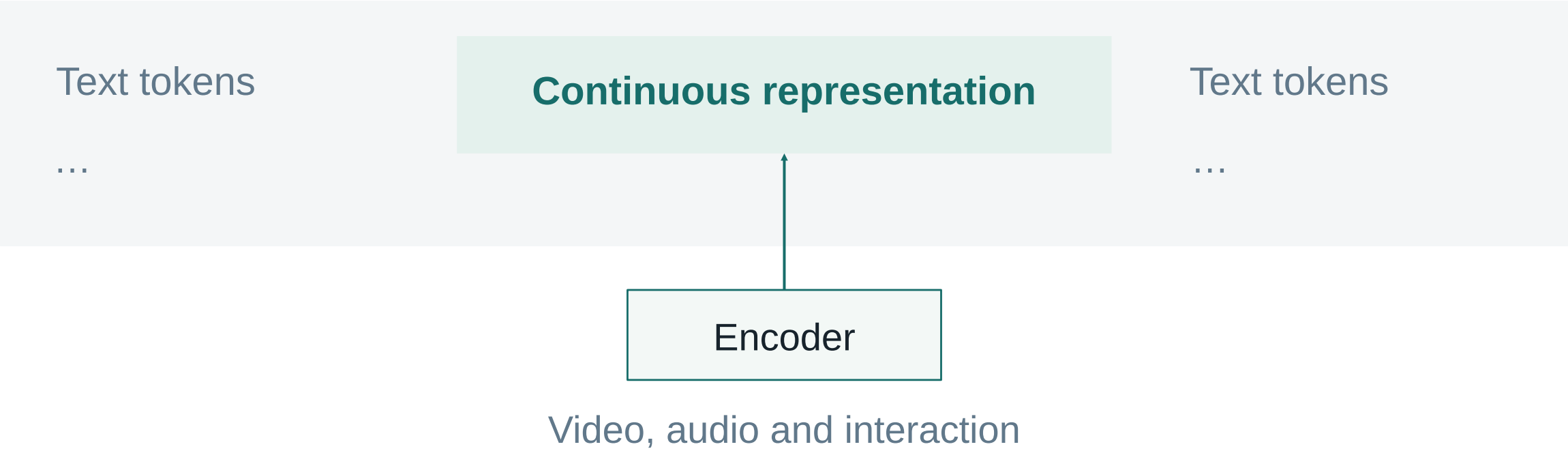
Finite precision
Density prediction at a chosen scale

Diffusion / flow matching / normalizing flows

This talk explores the second route. It is a choice, not a necessity.

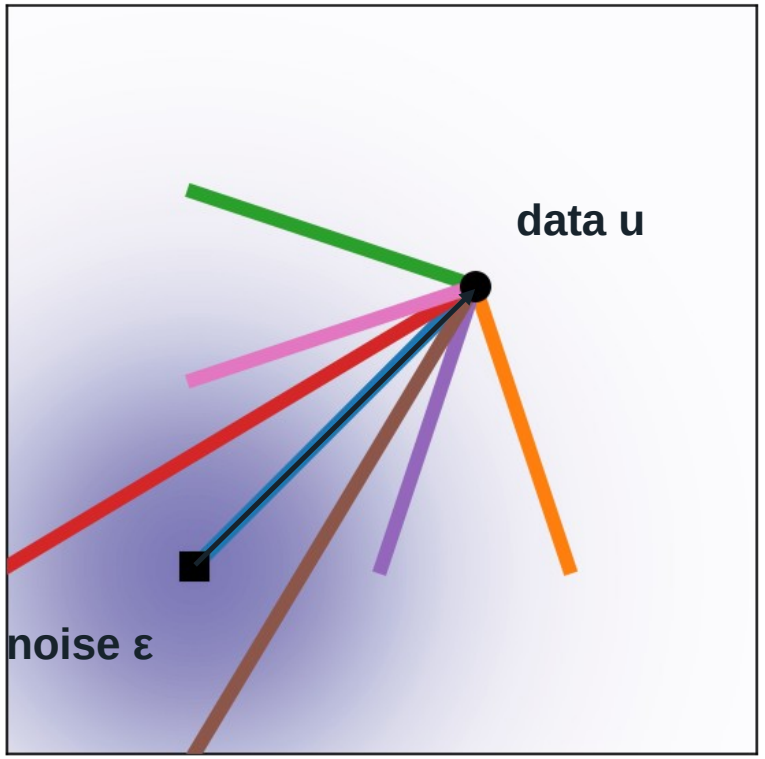
Discrete tokens and numerical latents are alternative interfaces to non-textual prediction.

Could thinking use continuous representations?



How should we predict the continuous part?

A widely used route: diffusion and flow matching



Conditional transport paths

Diffusion learns to denoise.

Flow matching learns a velocity field.

$$x_\tau = (1-\tau)\epsilon + \tau u$$

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}\,\|v_\theta(x_\tau, \tau, c) - (u-\epsilon)\|^2$$

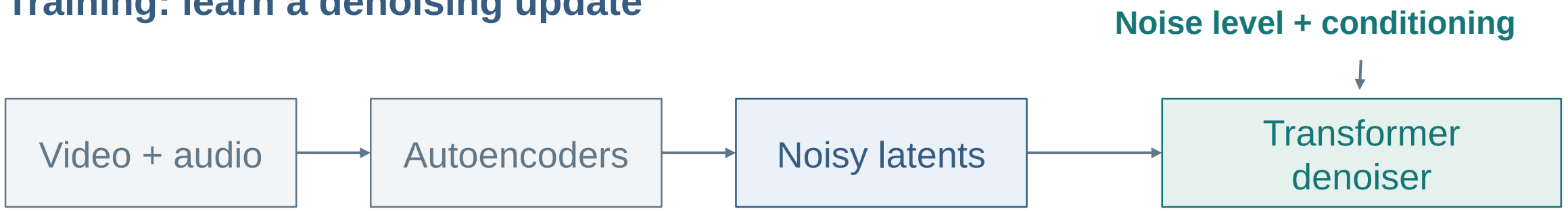
Generate by integrating the learned field

Flow matching trains a continuous normalizing flow.

Lipman et al., Flow Matching, ICLR 2023

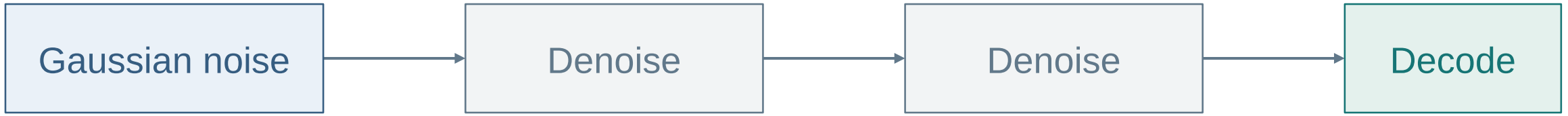
Video diffusion in a learned latent space

Training: learn a denoising update



Conditioning: prompt, observed frames or actions, depending on the model.

Sampling: repeat the update from noise



Video time: events in the clip. Denoising time: generation updates.

Veo 3 technical report, p. 1. Generic conditioning interface; not every model supports every condition.

Video generation: Veo 3

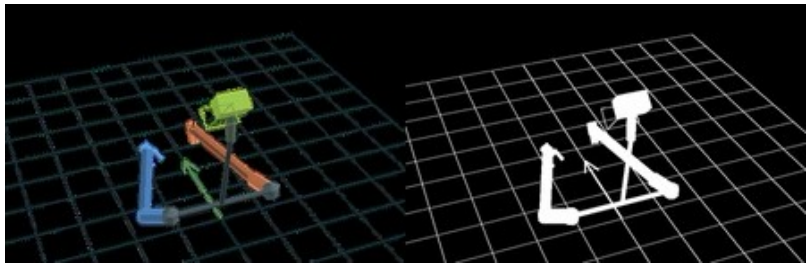


Google DeepMind, Veo 3 generated video + audio. Original eight-second demo.

Action-conditioned video prediction

Given: history + actions → Generate: future video

Input robot actions



3D action visualization

Generated RGB

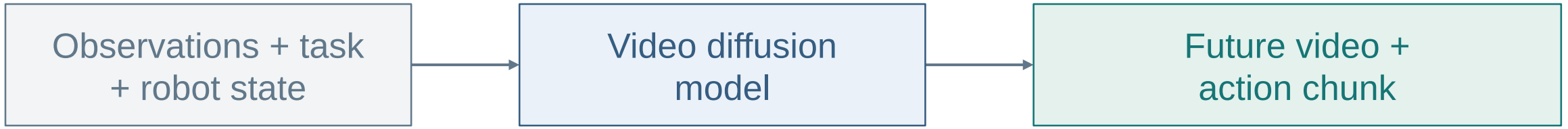


History and new actions condition each new video chunk

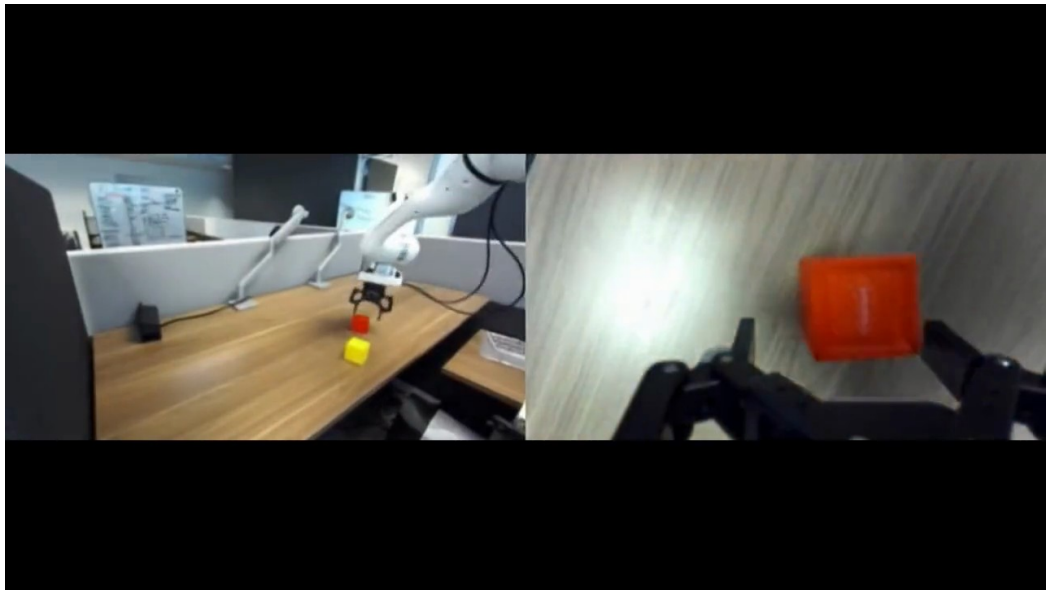
NVIDIA, Interactive World Simulation with Cosmos, 2026. Generated video. Click each clip to play.

World-action models: DreamZero

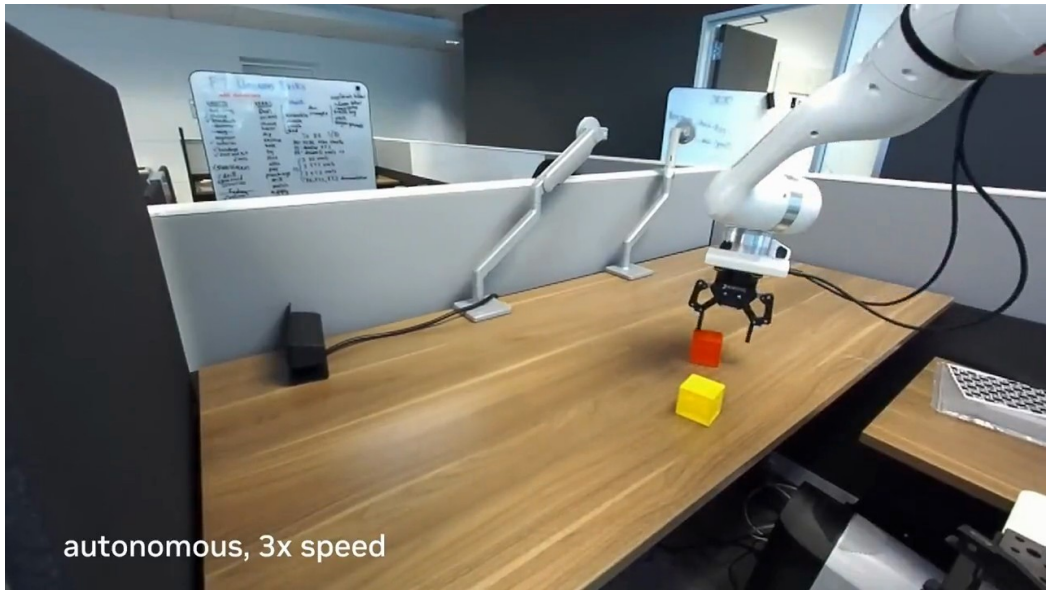
Given: observations + task + state → Generate: video + actions



Predicted video



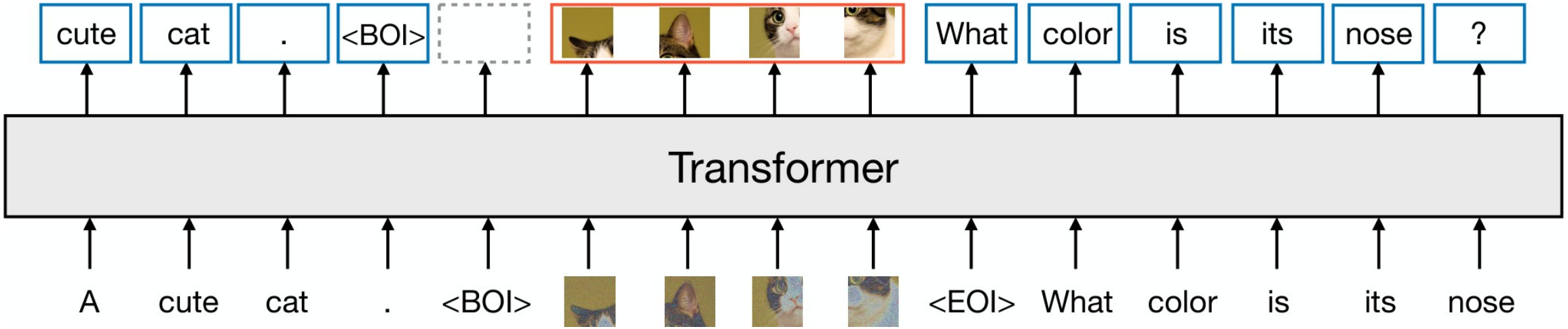
Real robot execution



Ye et al., DreamZero, 2026. Official demo pair. Real execution shown at 3× speed.

Transfusion: one model, two prediction objectives

Text tokens **Continuous image patches**



$$\mathcal{L} = \mathcal{L}_{\text{text}} + \lambda \mathcal{L}_{\text{diffusion}}$$

Image patches are denoised together; text continues token by token.

Zhou et al., Transfusion, 2024, Fig. 1

What choices and costs remain?

Different objectives

Text NLL + denoising loss
Different scales and loss weights

Time / noise weighting

$$\mathbb{E}_{\tau}\left[w(\tau)\,\ell_{\tau}\right]$$

Where does finite capacity go?

Mode coverage

Are all valid futures retained,
including low-frequency branches?

Two inference loops

Next sequence block

Denoise → denoise → …

Transfusion, 2024 / Kingma & Gao, 2023 / MAR, 2024

One likelihood principle for mixed sequences

Text tokens

...

Continuous representation

Text tokens

...

$$\mathcal{L} = -\sum_{\text{positions}} \log p_\theta(\text{next element} \mid \text{preceding context})$$

Text: probability mass

Continuous representation: density

Representation and noise scale remain modeling choices.

Continuous likelihood via change of variables

An invertible map connects representations to a simple base density

$$z = f_\theta(x)$$

For a fixed preceding context

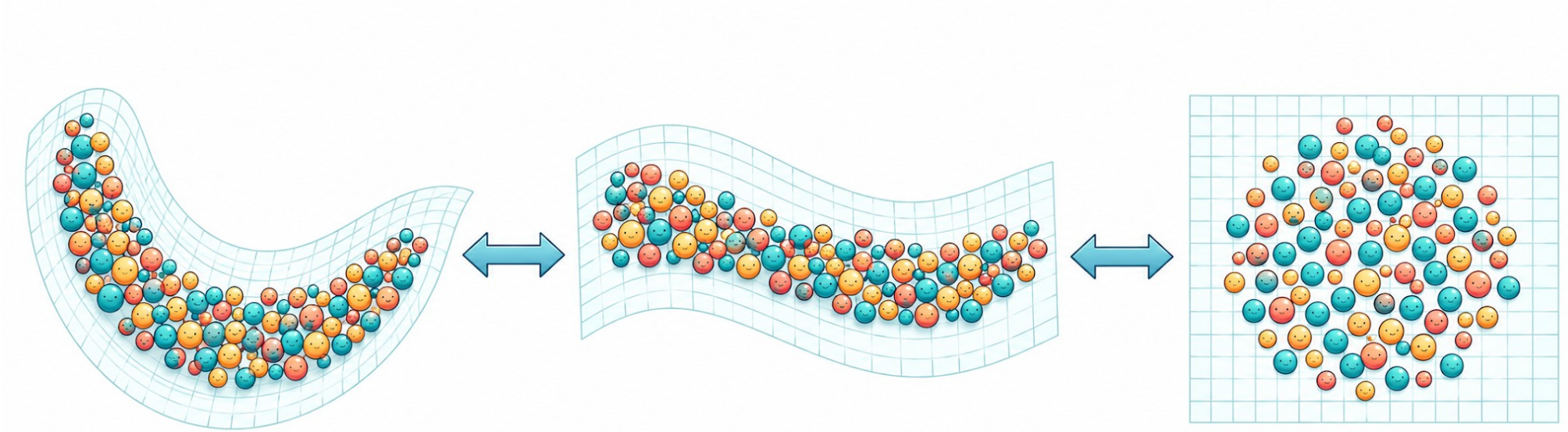
$$\log p_\theta(x) = \underbrace{\log p_0(f_\theta(x))}_{\text{simple base density}} + \underbrace{\log \left| \det \frac{\partial f_\theta(x)}{\partial x} \right|}_{\text{volume correction}}$$

Train by maximizing this log-density.

Normalizing flows: Papamakarios et al., 2021

Change of variables: the geometric picture

Representation density **Intermediate coordinates** **Simple base distribution**



Likelihood: map the observation to the base

Base log-density + a correction for local volume.

Sampling: draw from the base, then invert

The same map supports both operations.

Why structure the invertible transform?

Density evaluation: x → z **Sampling: z → x**

$$J_f(x) = \frac{\partial z}{\partial x}, \qquad [J_f(x)]_{ij} = \frac{\partial z_i}{\partial x_j}$$

Rows describe output coordinates. Columns describe input coordinates.

$$\log p_X(x) = \log p_Z(f_\theta(x)) + \log \left|\det J_f(x)\right|$$

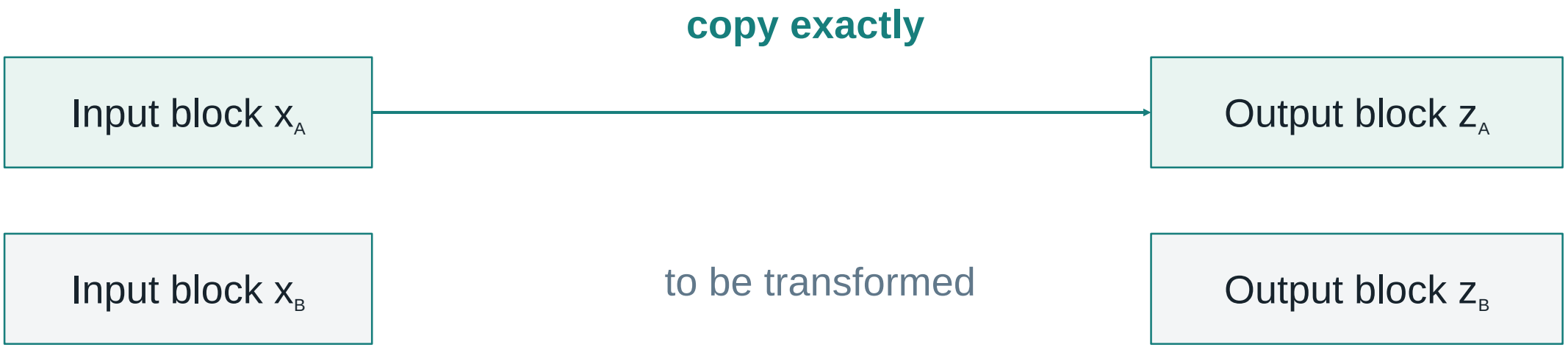
We want an explicit inverse and a tractable log-determinant.

The structure of the layer makes those operations simple.

Dinh et al., Real NVP (2017), §3

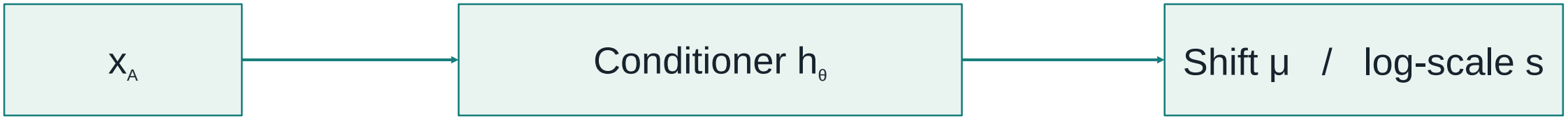
Split the coordinates and copy one block

$$x = \begin{bmatrix} x_A \\ x_B \end{bmatrix}, \qquad z = \begin{bmatrix} z_A \\ z_B \end{bmatrix}, \qquad z_A = x_A$$



The copied block provides the conditioning information.

The conditioner predicts shift and scale



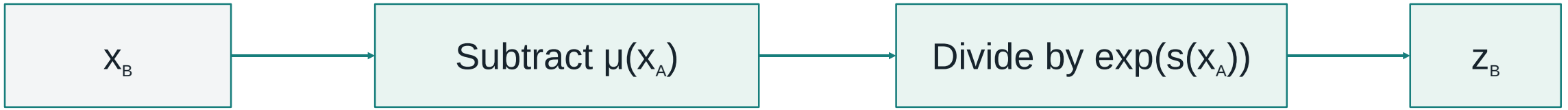
$$(\mu, s) = h_\theta(x_A), \qquad e^s > 0$$

One shift and scale per coordinate in block B.

The conditioner reads x_A. It does not read x_B.

Dinh et al., Real NVP (2017), §3

Normalize the other block



$$z_B = (x_B - \mu(x_A)) \odot e^{-s(x_A)}$$

The same affine relation, rearranged:

$$x_B = \mu(x_A) + e^{s(x_A)} \odot z_B$$

For fixed x_A, each coordinate of B is shifted and rescaled.

Dinh et al., Real NVP (2017), §3

Write the Jacobian in four blocks

$$z_A = x_A,$$
$$z_B = (x_B - \mu(x_A)) \odot e^{-s(x_A)}$$

$$J_f(x) = \frac{\partial(z_A, z_B)}{\partial(x_A, x_B)} = \begin{bmatrix} \frac{\partial z_A}{\partial x_A} & \frac{\partial z_A}{\partial x_B} \\ \frac{\partial z_B}{\partial x_A} & \frac{\partial z_B}{\partial x_B} \end{bmatrix}$$

Block rows: z_A, z_B. Block columns: x_A, x_B.

Compute how each output block depends on each input block.

The copied block gives I and zero

Since $z_A = x_A$:

$$\frac{\partial z_A}{\partial x_A} = I, \qquad \frac{\partial z_A}{\partial x_B} = 0$$

$$J_f(x) = \begin{bmatrix} I & 0 \\ ? & ? \end{bmatrix}$$

z_A does not depend on x_B, so the upper-right block is exactly zero.

Dinh et al., Real NVP (2017), §3

Different coordinates in B do not mix directly

$$z_{B,j} = (x_{B,j} - \mu_j(x_A))\, e^{-s_j(x_A)}$$

Hold x_A fixed: $\mu_j(x_A)$ and $s_j(x_A)$ are constants.

$$\frac{\partial z_{B,j}}{\partial x_{B,k}} = \begin{cases} e^{-s_j(x_A)} & j = k, \\ 0 & j \neq k. \end{cases}$$

The lower-right block is diagonal, with entries $\exp(-s_j(x_A))$.

The lower-left block can be dense

$$z_{B,j} = \left(x_{B,j} - \mu_j(x_A)\right) e^{-s_j(x_A)}$$

Apply the product rule with respect to $x_{A,k}$:

$$\frac{\partial z_{B,j}}{\partial x_{A,k}} = -\frac{\partial \mu_j}{\partial x_{A,k}} e^{-s_j} - (x_{B,j} - \mu_j) e^{-s_j} \frac{\partial s_j}{\partial x_{A,k}}$$

$$\frac{\partial z_{B,j}}{\partial x_{A,k}} = -e^{-s_j} \frac{\partial \mu_j}{\partial x_{A,k}} - z_{B,j} \frac{\partial s_j}{\partial x_{A,k}}$$

This block may be dense. The determinant will not require its entries.

Assemble the block-lower-triangular Jacobian

$$J_f(x) = \begin{bmatrix} \dfrac{\partial z_A}{\partial x_A} & \dfrac{\partial z_A}{\partial x_B} \\ \dfrac{\partial z_B}{\partial x_A} & \dfrac{\partial z_B}{\partial x_B} \end{bmatrix} = \begin{bmatrix} I & 0 \\ * & \operatorname{diag}\left(e^{-s(x_A)}\right) \end{bmatrix}$$

The upper-right zero makes the matrix block lower triangular.

$$\det J_f = \det(I)\, \det\left(\operatorname{diag}(e^{-s(x_A)})\right)$$

The determinant depends only on the diagonal blocks.

Dinh et al., Real NVP (2017), §3

The log-determinant becomes a sum

$$\det J_f = \det(I)\ \det\Big(\mathrm{diag}(e^{-s(x_A)})\Big)$$

$$\det J_f = \prod_{j\in B} e^{-s_j(x_A)} = \exp\left(-\sum_{j\in B} s_j(x_A)\right)$$

$$\log|\det J_f(x)| = -\sum_{j\in B} s_j(x_A)$$

For x → z, the log-determinant is −Σ $s_j(x_A)$.

Dinh et al., Real NVP (2017), §3

Insert the result into the likelihood

$$\log p_X(x) = \log p_Z(f_\theta(x)) + \log \left|\det J_f(x)\right|$$

$$\log \left|\det J_f(x)\right| = -\sum_{j \in B} s_j(x_A)$$

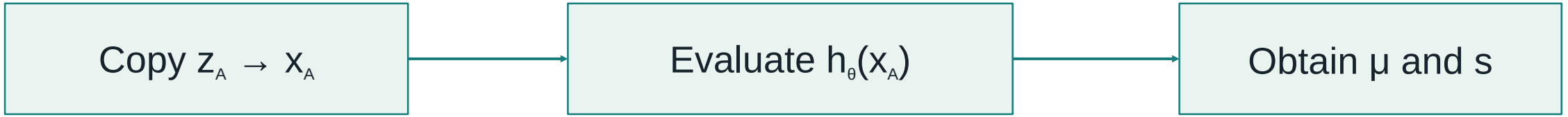
For a standard Gaussian base:

$$-\log p_X(x) = \frac{1}{2}\|f_\theta(x)\|^2 + \sum_{j \in B} s_j(x_A) + \frac{D}{2}\log(2\pi)$$

The volume correction is part of exact maximum-likelihood training.

Dinh et al., Real NVP (2017), §3

Sample by undoing scale and shift

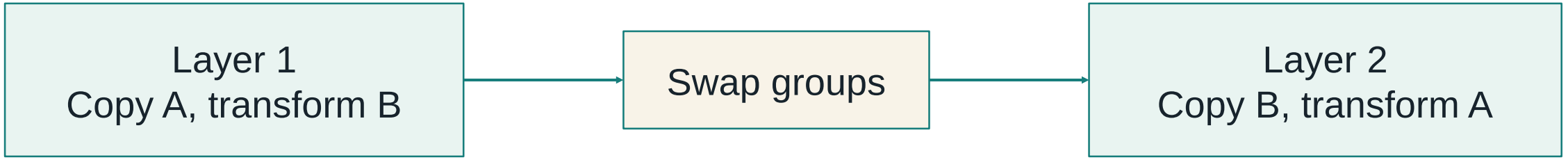


$$x_A = z_A,$$

$$x_B = \mu(x_A) + e^{s(x_A)} \odot z_B$$

Evaluate the conditioner, then undo the affine transformation.

Stack layers to transform every coordinate



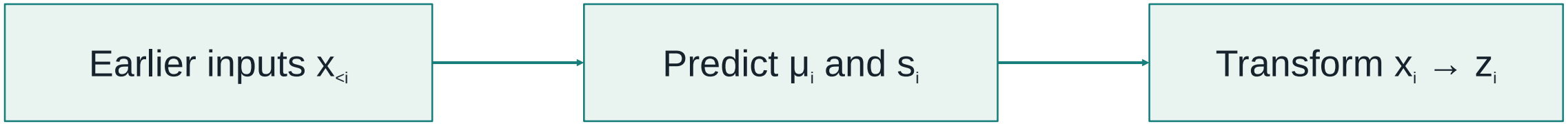
Each layer has its own shift and scale network.

All coordinates can change across the stack.

Alternate the transformed group; add each layer’s log-determinant.

Autoregressive flow: condition on earlier inputs

$$x_{<i} = (x_1, \ldots, x_{i-1})$$



$$z_i = \left(x_i - \mu_i(x_{<i})\right) e^{-s_i(x_{<i})}$$

Only earlier inputs determine the shift and scale for x_i.

The Jacobian is triangular again

$$J_f(x) = \begin{bmatrix} e^{-s_1} & 0 & \cdots & 0 \\ * & e^{-s_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & e^{-s_D} \end{bmatrix}$$

No dependence on later inputs

→ zeros above the diagonal

The diagonal uses the affine derivative already derived.

$$\log \left| \det J_f(x) \right| = -\sum_{i=1}^{D} s_i(x_{<i})$$

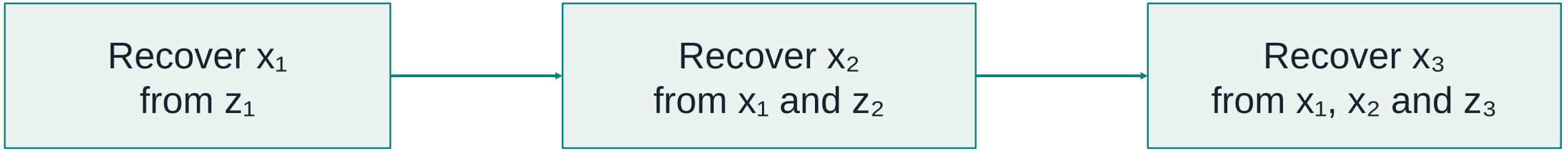
Reuse the affine derivative and the triangular-determinant rule.

Training is parallel; sampling is sequential

Training: the entire input x is observed.

All shifts and scales can be computed in one masked pass.

$$x_i = \mu_i(x_{<i}) + e^{s_i(x_{<i})} z_i$$



Sampling conditions on the x-prefix already reconstructed.

Discrete likelihood vs continuous density

Discrete: probability mass

$$0 \leq P(x) \leq 1$$

Probability assigned to a symbol

Continuous: density

$$p(x) \approx \frac{\text{mass in a small region}}{\text{volume of that region}}$$

Same mass, smaller volume, higher density

Density can exceed 1. With finite data, fitting ever-narrower peaks can inflate it.

Choose a finite resolution for the representation.

Fit the noise-smoothed distribution at that scale.

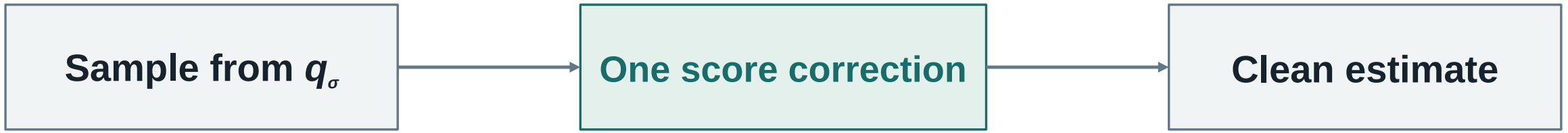
Ho et al., Flow++, 2019, §3.1 · Loaiza-Ganem et al., 2022

Likelihood at a chosen resolution

Gaussian noise sets the modeling resolution.

$$q_\sigma = p_{\text{data}} * \mathcal{N}(0, \sigma^2 I)$$

At scale σ, learn the density of the perturbed variable.



$$x \sim q_\sigma, \qquad \hat{x}_{\text{clean}} \approx x + \sigma^2 \nabla_x \log q_\sigma(x)$$

Tweedie posterior-mean estimate

Scaling an invertible model

Capacity

Rich dependencies in
high-dimensional visual data

Structure

An invertible map with
a tractable log-determinant

The challenge is an architecture that scales while providing both.

TARFlow revisits this design with causal Transformers.

Zhai et al., TARFlow, ICML 2025, §§2.1–2.3, 4

TARFlow: Transformer autoregressive flows

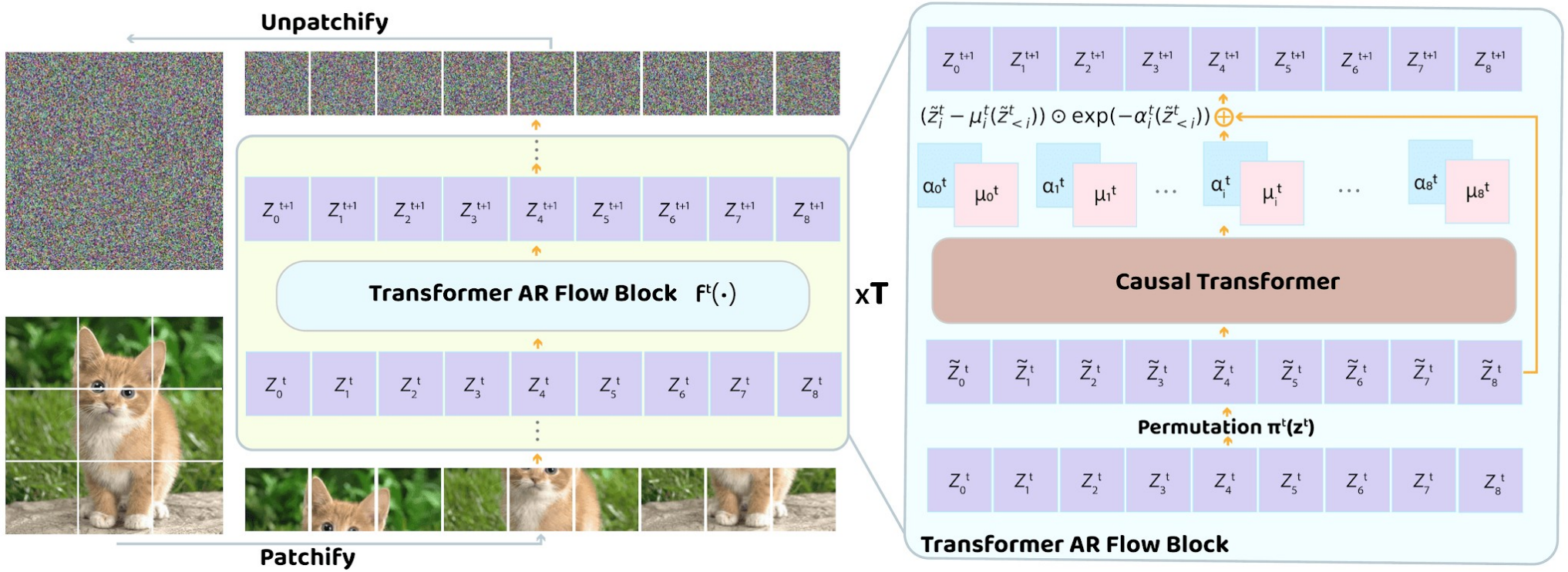


Image-patch autoregression

Causal Transformers predict affine parameters.

Noise-augmented likelihood

Sample a noisy image; denoise with the model score.

Why stacked AR flows are more expressive

Proposition 1: alternating orderings with 3+ blocks give universal approximation.

1 block

Single Gaussian per conditional

2 opposite orders

Gaussian mixtures, except the final coordinate

3+ blocks

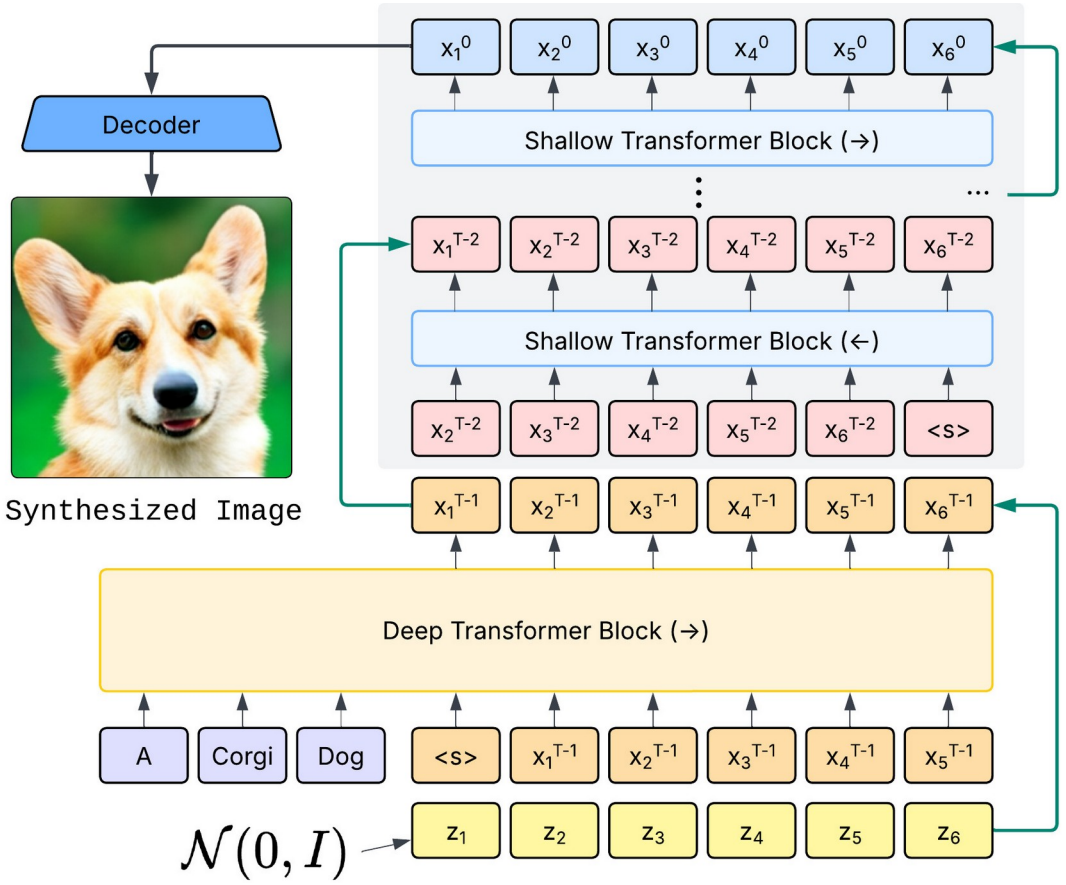
Removes the remaining coordinate restriction

$$p_\theta(x_d \mid x_{<d}) = \int \mathcal{N}(x_d; \hat{\mu}, \hat{\sigma}^2)\, p(y_{>d} \mid x_{<d})\, \mathrm{d}y_{>d}$$

Intermediate coordinates act as mixing variables (two-block case, d < D).

> Expressivity with sufficient capacity; successful optimization is a separate question.

STARFlow: two equivalent views of the same model



$$u = f_S(x), \quad z = f_D(u)$$

1. Simple base prior

The full flow maps codec latents x to a fixed Gaussian base z.

2. Learned AR prior

Shallow blocks map x to u.
The deep block defines the AR prior.

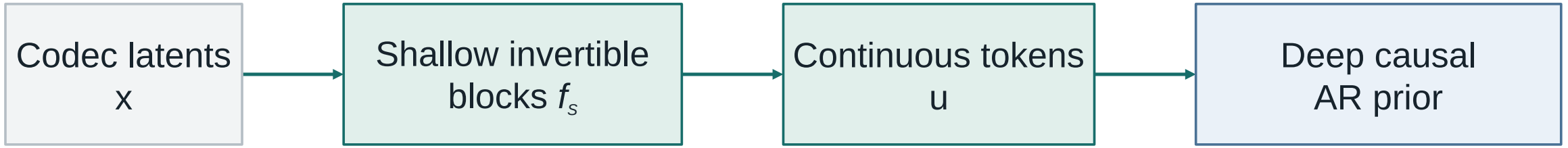
$$p_\theta(x) = \pi_0(z)\,|\det J_f(x)|$$
$$= p_{\mathrm{AR}}(u)\,|\det J_{f_S}(x)|$$

Score: x → u → z Sample: z → u → x

Gu et al., STARFlow, NeurIPS 2025, §3.2 & Fig. 4

An invertible tokenizer + next-Gaussian prediction

A useful view of the model



$$p_{\mathrm{AR}}(u_i \mid u_{<i}) = \mathcal{N}\big(\mu_i(u_{<i}),\, \operatorname{diag}\sigma_i^2(u_{<i})\big)$$

$$\mathcal{L}_{\mathrm{NLL}} = \underbrace{-\sum_i \log p_{\mathrm{AR}}(u_i \mid u_{<i})}_{\text{Fit predictable conditionals}} - \underbrace{\log\left|\det J_{f_S}(x)\right|}_{\text{Correct for tokenizer volume}}$$

Reversible continuous tokenization, without quantization.

Learn the coordinates and their conditional density together.

Gu et al. · STARFlow · NeurIPS 2025 · Fig. 4 / §3.2

STARFlow: high-resolution image synthesis

Deep–shallow blocks and pretrained codec latents



ImageNet 256², FID-50K ↓

TARFlow, pixels

5.56 · 1.3B

Deep–shallow, pixels

4.69 · 1.4B

STARFlow, latents

2.40 · 1.4B

Left: 3.8B text-conditioned samples. Right: class-conditioned ImageNet models.

Video prediction brings in temporal dynamics

Image density

$$p_\theta(x)$$

A rich visual distribution

Next-frame density

$$p_\theta(x_t \mid x_{<t})$$

A rich visual distribution
conditioned on the observed past

Predict how the scene can evolve over time.

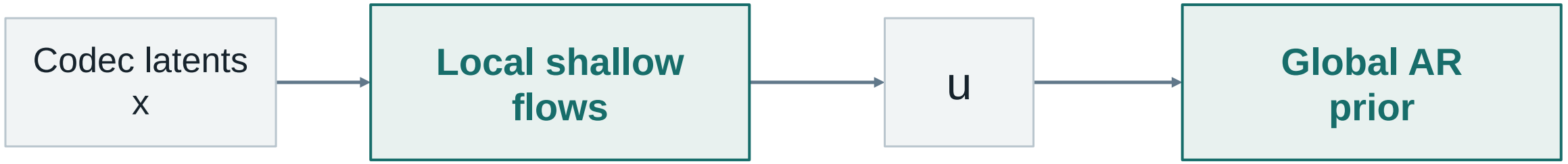
Action-conditioned prediction will return in NFWM.

Gu et al., STARFlow-V, CVPR 2026

STARFlow-V: local transforms, global autoregression

$$u = f_{\text{local}}(x)$$

The same density decomposition, now for video



Attention restricted to each frame

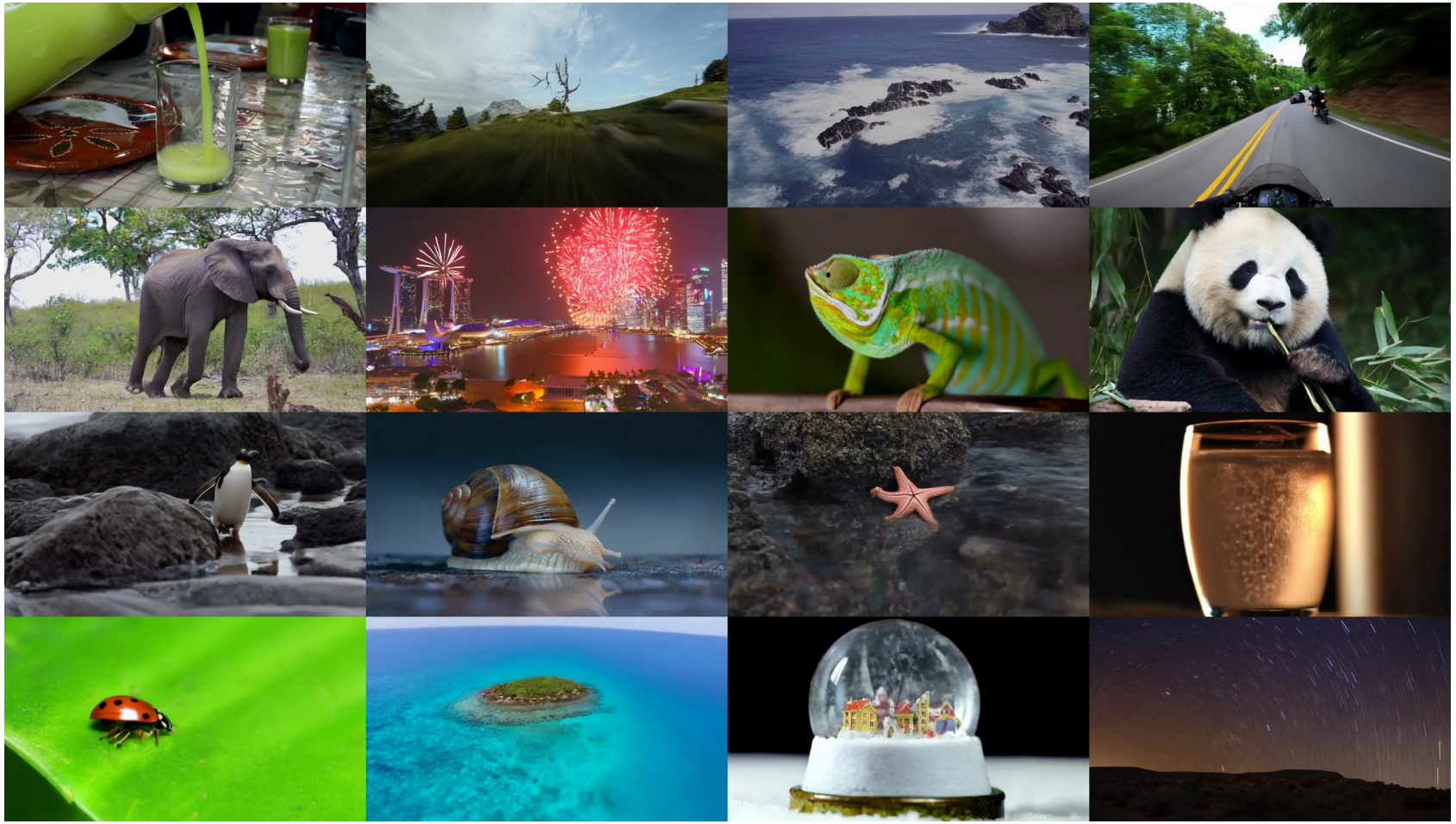
Past frames + earlier within-frame tokens

$$p_\theta(x) = p_{\text{global AR}}(u)\ \left|\det J_{f_{\text{local}}}(x)\right|$$

Videos also use a learned causal denoiser; exact density applies before correction.

Gu et al., STARFlow-V, CVPR 2026, §3.1 & Eq. 3.2

STARFlow-V samples



Model-generated video montage

One video model, three conditioning modes

Shared architecture for text, image and video conditioning

Text-to-video



Sample a video from text.

Image-to-video



Encode a frame, then continue.

Video-to-video



Input clip Edited clip

V2V editing uses a fine-tuned checkpoint.

Click an example to open the source demonstrations.

STARFlow-V: video-generation evidence

Reported VBench text-to-video results

MODEL	TOTAL	CAUSAL	FAMILY
Wan2.1	83.69	No	Diffusion
CogVideoX	80.91	No	Diffusion
STARFlow-V	78.67	Yes	NF

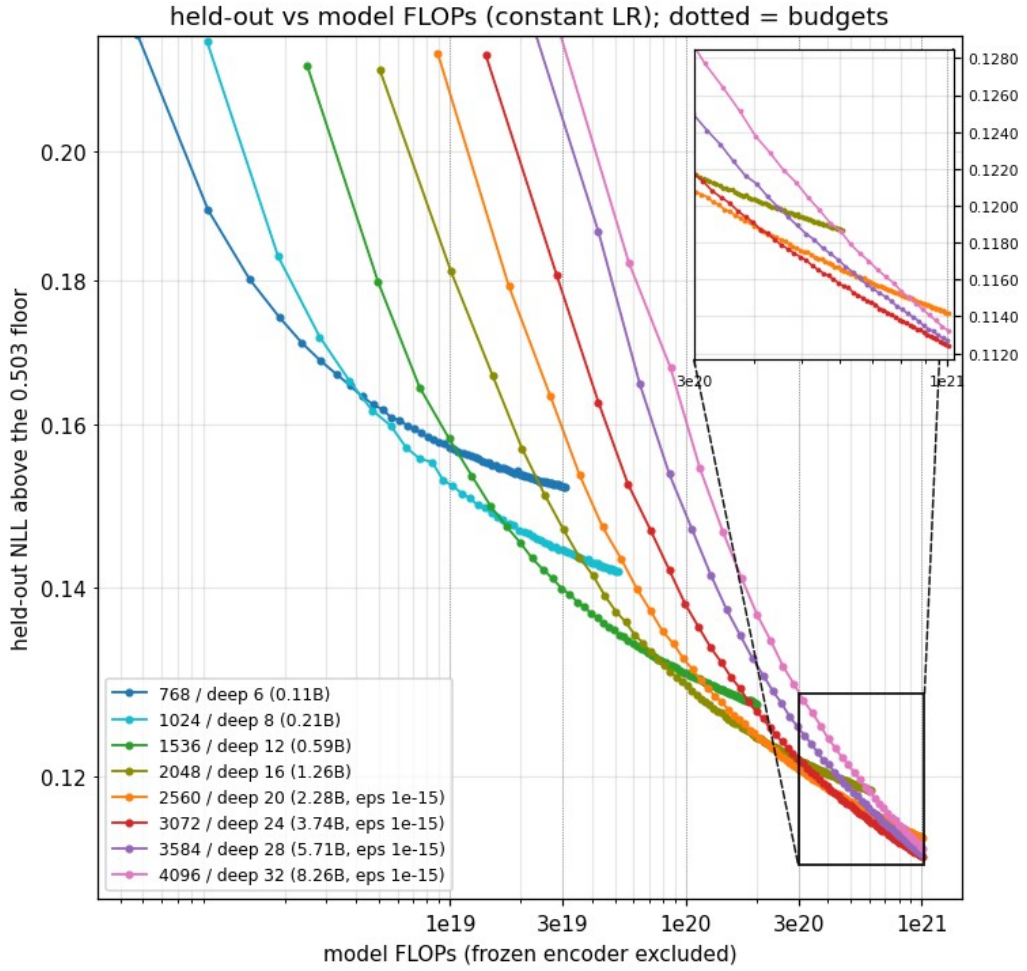
STARFlow-V with a GPT prompt rewriter: 79.70.

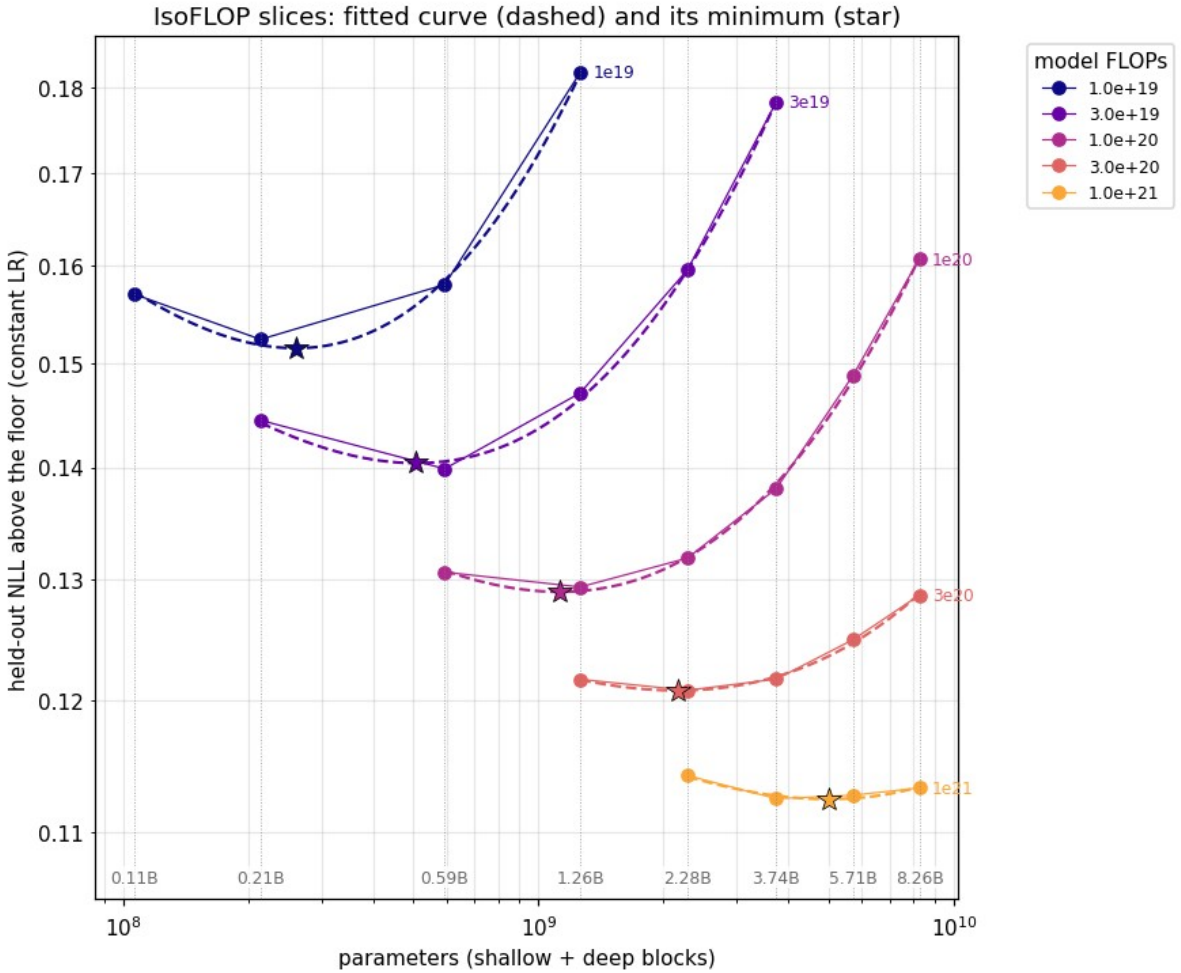
A causal NF generates competitive videos; quality still has room to improve.

Model sizes and training budgets differ.

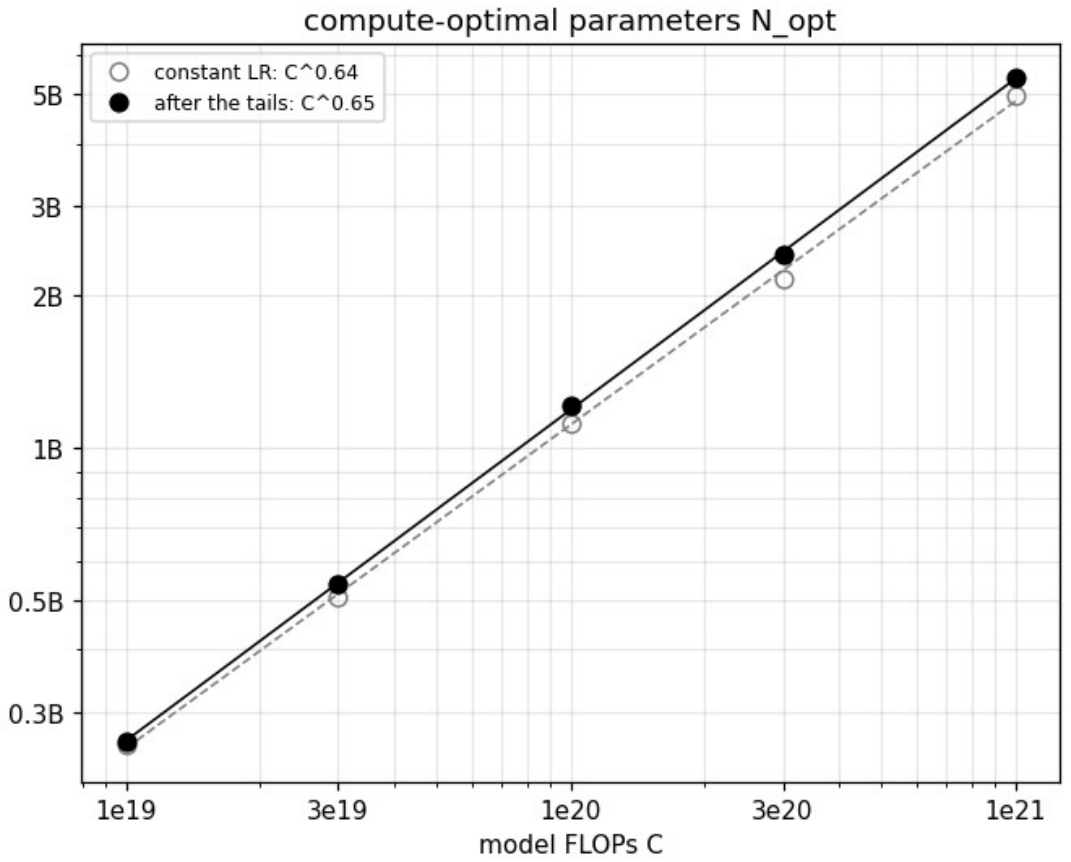
Recent result: STARFlow-V scales smoothly

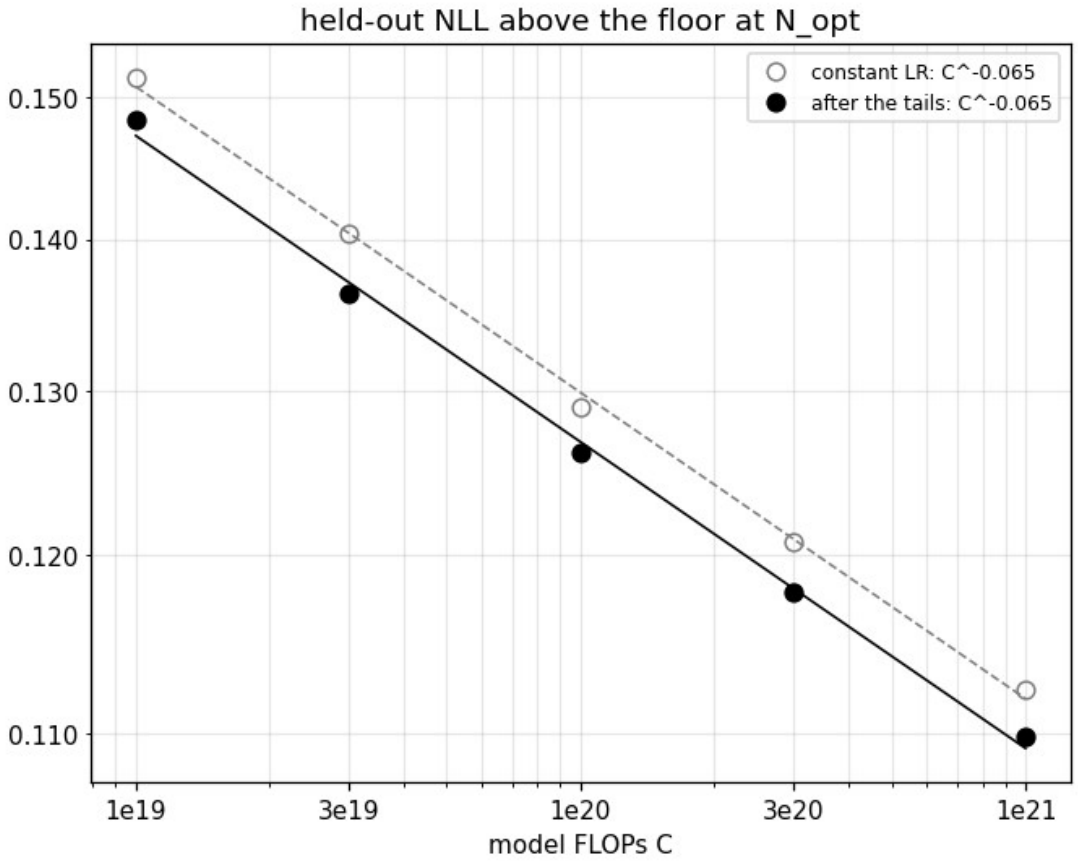
More compute lowers NLL. Larger budgets favor larger models.





STARFlow-V: empirical compute-optimal scaling



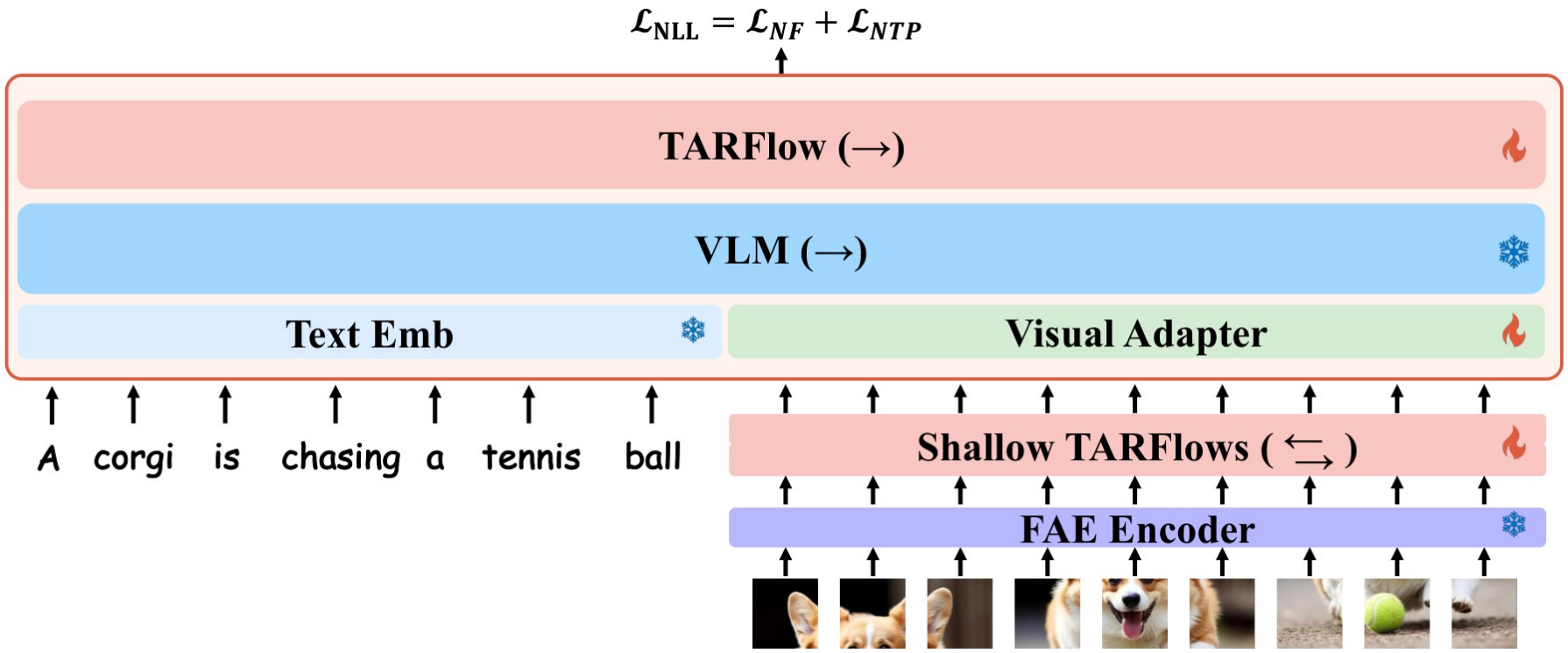


$N_{opt} \propto C^{0.64–0.65}$

$(L – 0.503) \propto C^{-0.065}$

STARFlow-V · recent experiments · C: model FLOPs (frozen encoder excluded)

STARFlow2: multimodal generation



Frozen VLM + trainable TARFlow

Cross-stream connections align the two models.

Text prediction + visual density modeling

A shared FAE latent space; token-wise visual decoding.

STARFlow2: image generation and visual questions

Text to image

“A tiny astronaut explores a garden of oversized glowing mushrooms…”



Image to text



“How many dogs are there in the image?”

“There are two dogs in the image.”

STARFlow2: image editing from instructions

Input image



“Add a tiny captain’s hat…”



“Add a tropical island…”



“Remove the sunglasses…”



Three separate edits conditioned on the same input

STARFlow2: interleaved visual thinking

“If the yellow figurine on the far right is removed, how many cat-shaped figurines are left?”

INPUT



THINKING

“By visually inspecting the image, we can confirm the presence of several cat-shaped figurines…”



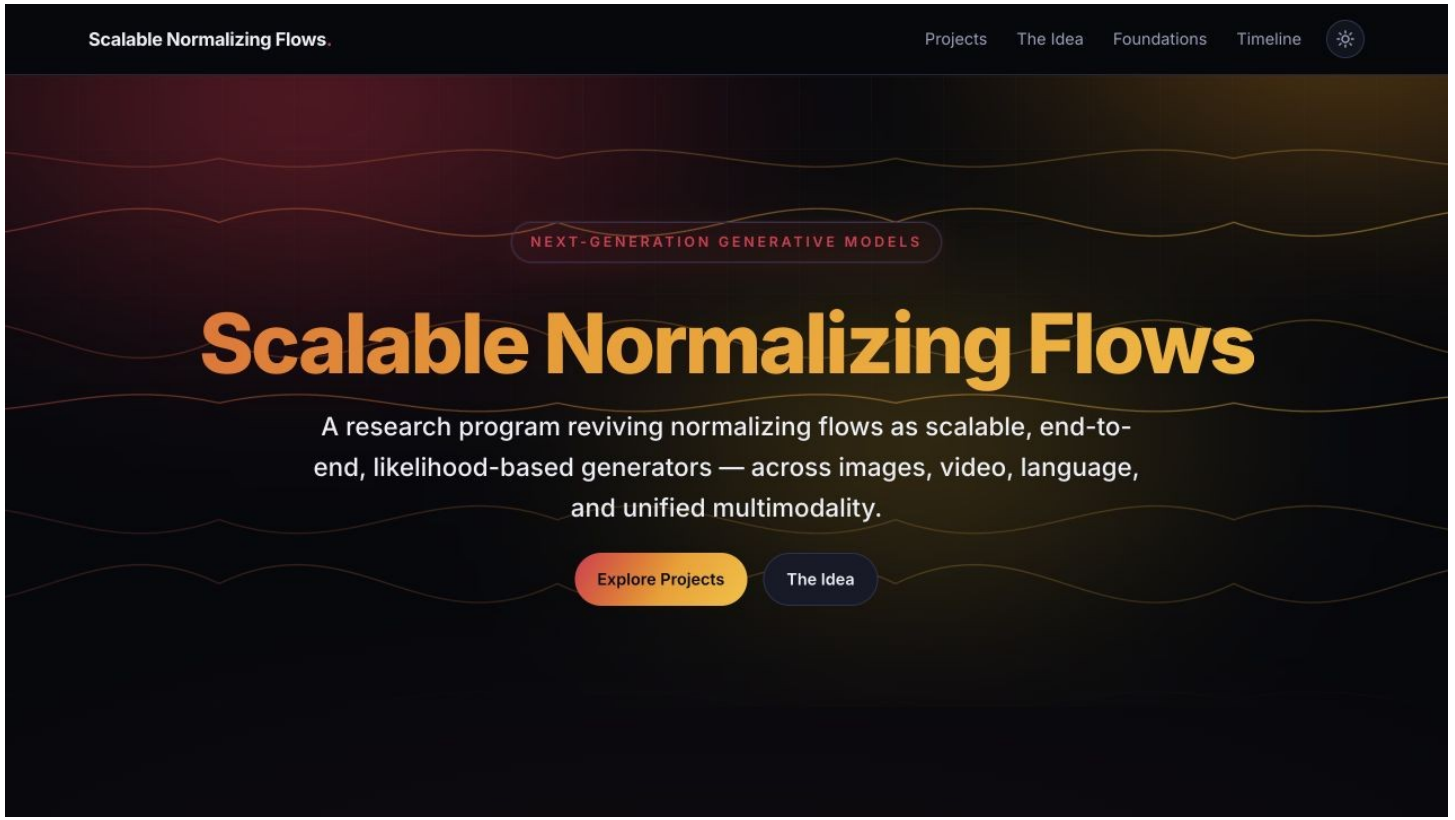
Generated image

ANSWER

6

The TARFlow family

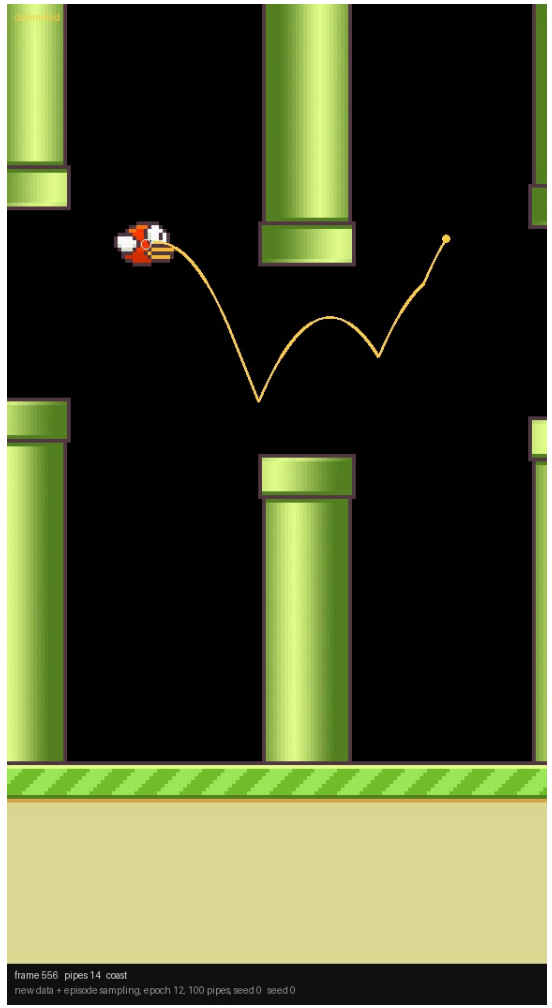
Images, video, and interleaved multimodal generation



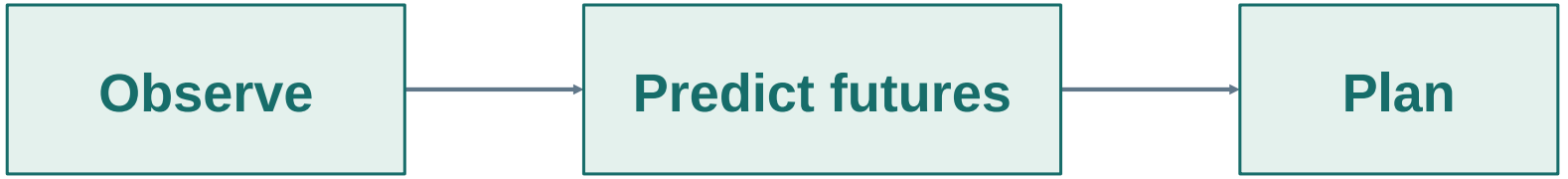
Can likelihood-based prediction improve decisions?

Project website: starflow-v.github.io

NFWM: predict, plan, and play



Imagine before acting



Candidate actions: flap or coast

Score imagined trajectories.
Execute the first action, then replan.

A flow model can be used inside a control loop.

From video prediction to action-conditioned prediction

Video prediction

$$p_\theta(x_{t+1} \mid x_{\leq t})$$

What happens next?

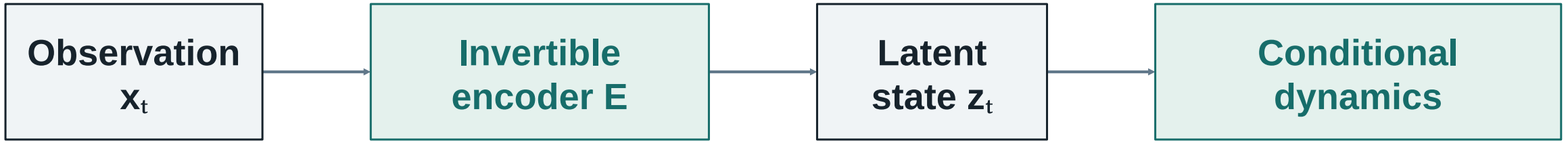
Action-conditioned prediction

$$p_\theta(x_{t+1} \mid x_{\leq t}, a_{\leq t})$$

What happens if I take this action?

Compare candidate actions by their predicted consequences.

NFWM: invertible encoding and stochastic latent dynamics



$$z_t = E_\theta(x_t), \qquad x_t = E_\theta^{-1}(z_t)$$

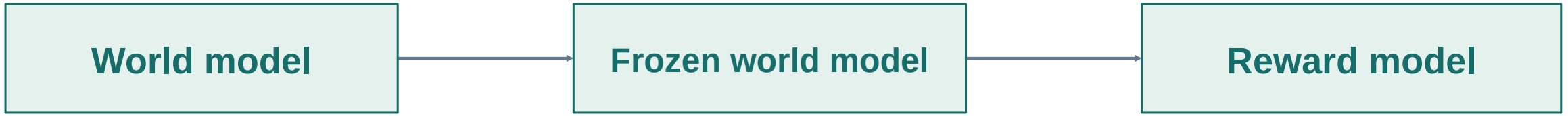
$$p_\phi(z_{t+1} \mid z_{\le t}, a_{\le t}) = \mathcal{N}(\mu_{t+1}, \operatorname{diag} \sigma_{t+1}^2)$$

$$\mathcal{L}_{\mathrm{NLL}} = -\log p_\phi(E_\theta(x_{t+1}) \mid z_{\le t}, a_{\le t}) - \log \left|\det J_{E_\theta}(x_{t+1})\right|$$

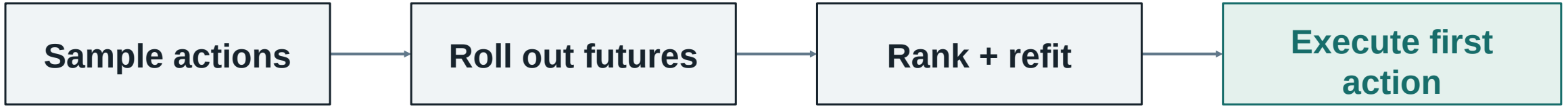
Train the encoder and dynamics jointly.

A simple control recipe: freeze, read out reward, plan

Learn the model → **Freeze it** → **Learn a reward readout**



$$a^*_{t:t+H-1} = \arg\max_{a_{t:t+H-1}} \mathbb{E}\left[\sum_{k=1}^{H} \hat{r}_\psi(z_{t+k})\right]$$



CEM repeats the first three steps; control repeats after each observation.

Planning with a learned model: one CEM iteration

Start with the current observation and a distribution over action sequences

1. Sample sequences

flap → coast → coast
coast → flap → coast
coast → coast → flap

2. Predict and score

Frozen world model
+ reward readout

Compare imagined returns

3. Keep elites + refit

Shift the action distribution toward high-return sequences

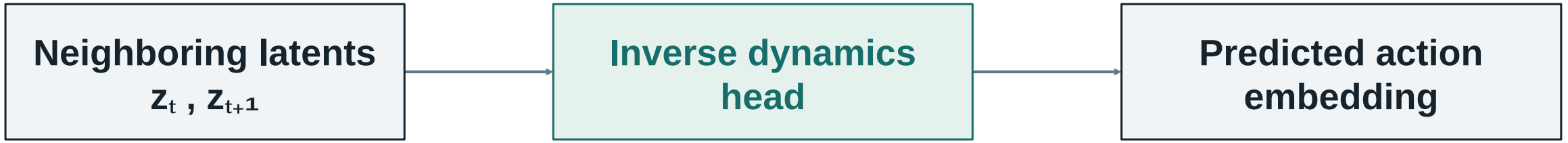
Repeat, then execute only the first action.

Receive a new observation and plan again.

CEM = cross-entropy method. Illustrative action sequences; no experimental scores are shown.

Shape the latent space for control

Recovering an observation does not define distance to a goal.



Training signal shapes the encoder

$$\mathcal{L}_{\text{inv}} = \|\hat{e}_{a_t} - \text{sg}(e_{a_t})\|_2^2$$

$$\mathcal{L} = \mathcal{L}_{\text{NLL}} + \lambda_{\text{inv}}\mathcal{L}_{\text{inv}}$$

Make action-induced changes accessible in the representation.

Plan actions with distance to an encoded goal



CEM refines actions

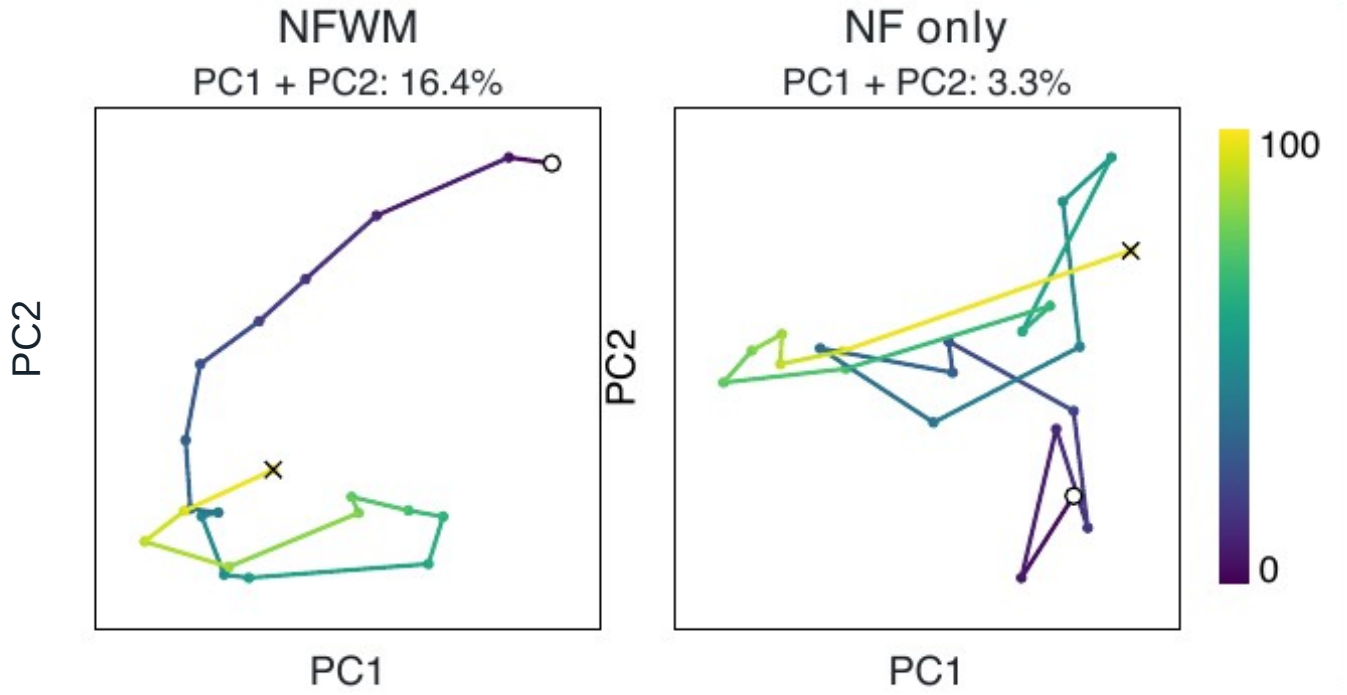
Sample intermediate futures

Compare to encoded goal

$$a^{*}_{t:t+H-1} = \arg\min_{a_{t:t+H-1}} \left\| \mu_{t+H}(a_{t:t+H-1}) - E_\theta(x_g) \right\|_2^2$$

The planner searches actions; the representation supplies the cost.

Both density learning and geometry shaping matter



Push-T planning success

Full objective

86%

Without inverse dynamics

34%

Without NLL

28%

PCA is illustrative; planning success tests whether the geometry is useful.

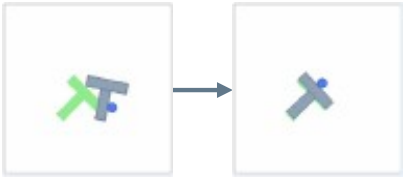
One world-model framework, different uses

NFWM
imagined futures

Goal planning

CEM searches action sequences

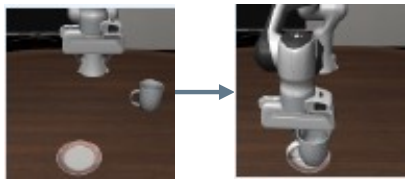
84.5% average on four deterministic tasks



Push-T

Action ranking

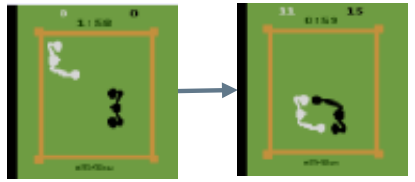
Compare candidate outcomes



LIBERO

Policy learning in imagination

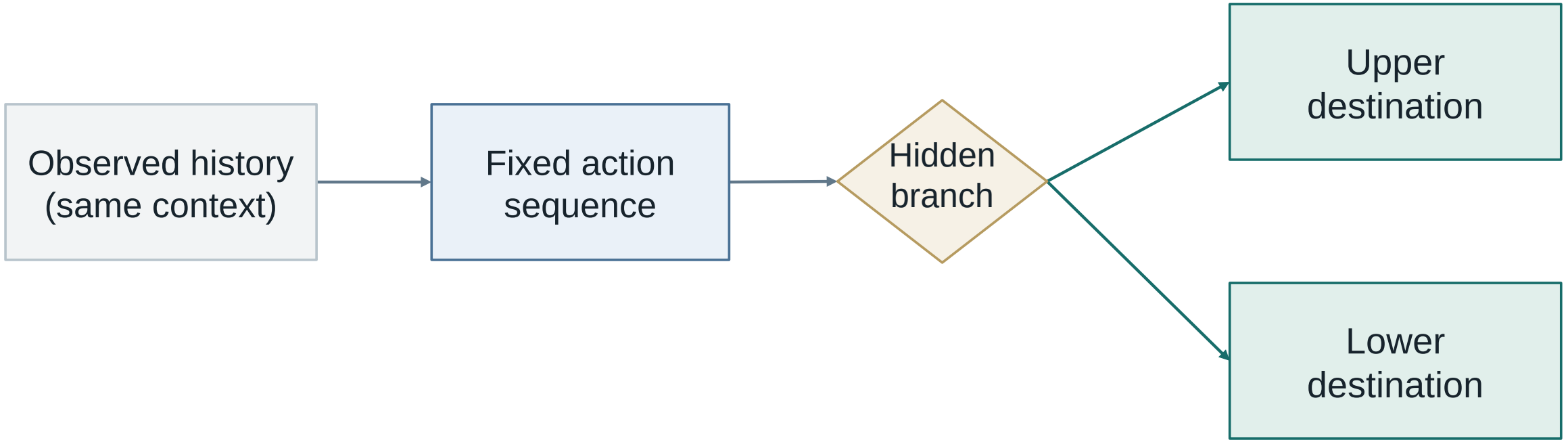
Update the policy using model rollouts



Atari Boxing

Teleport: identical context, different futures

The hidden branch is chosen after the action sequence is fixed.

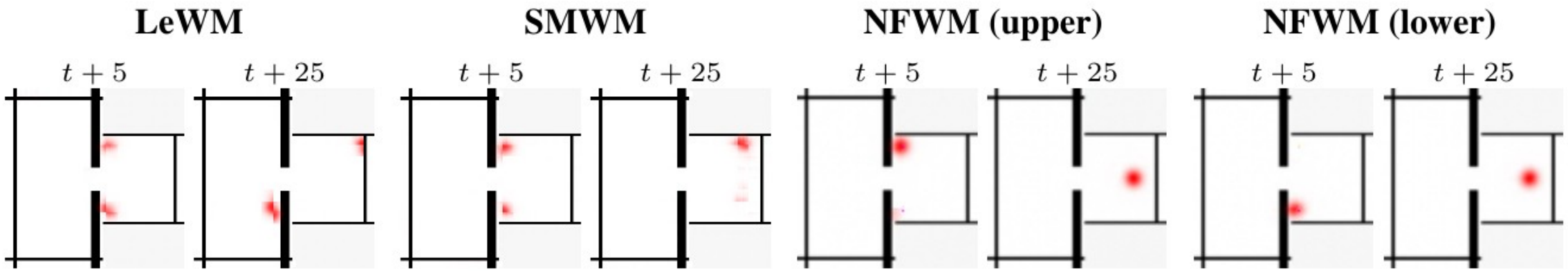


One point prediction cannot represent both outcomes.

An MSE predictor may average them.

Teleport: distinct sampled future branches

Gaussian in learned coordinates → nonlinear inverse → richer observed futures

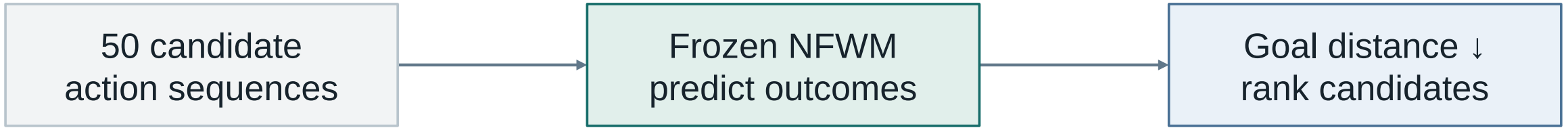


Same history + actions; distinct sampled branches.

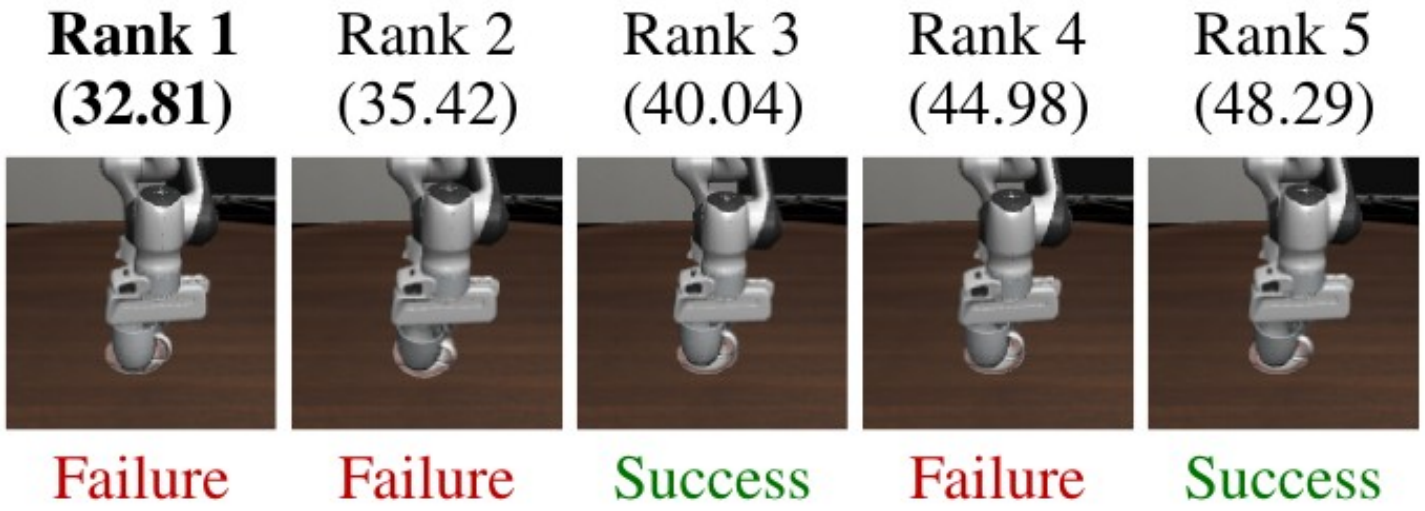
LeWM / SMWM: latent world-model baselines

Fidelity remains limited: 22/50 samples assigned to a valid branch.

Use imagined futures to rank actions

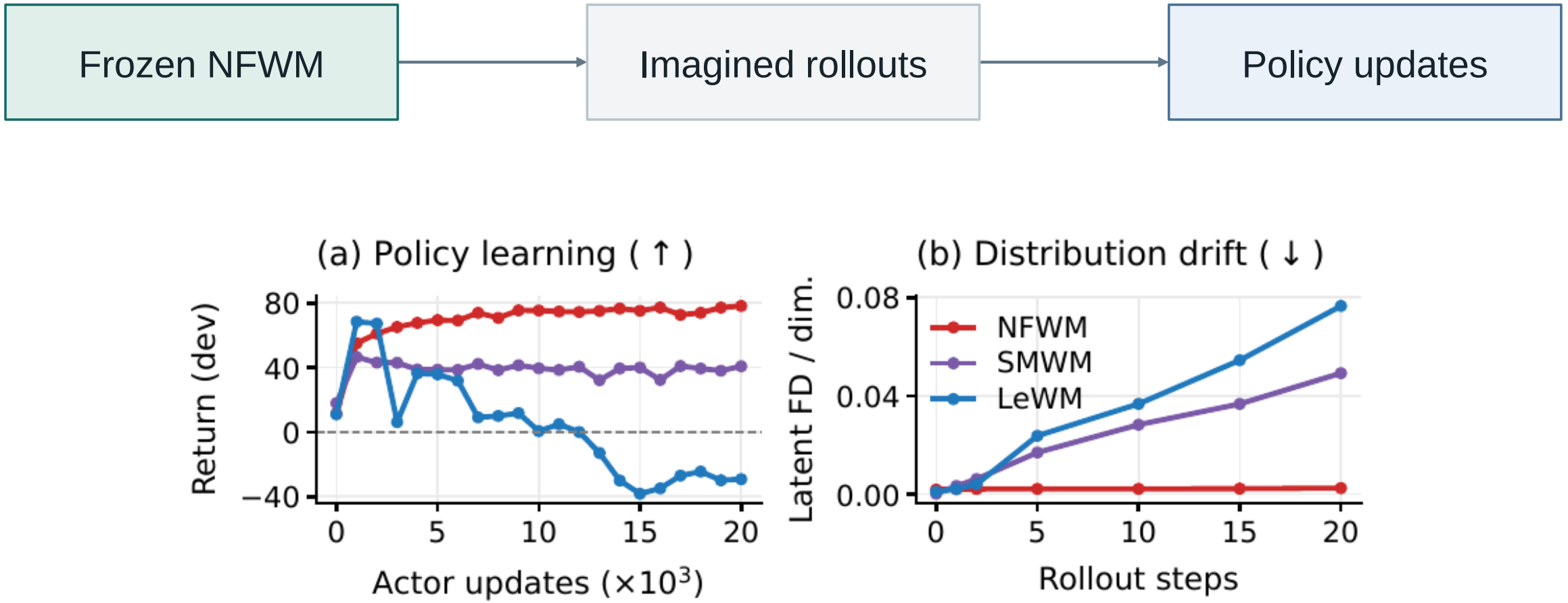


Goal: place the mug on the plate.
Photos: actual outcomes. Numbers: predicted goal distance ↓.



Top-1: NFWM 36.0% vs random 26.04% (same candidate pool)

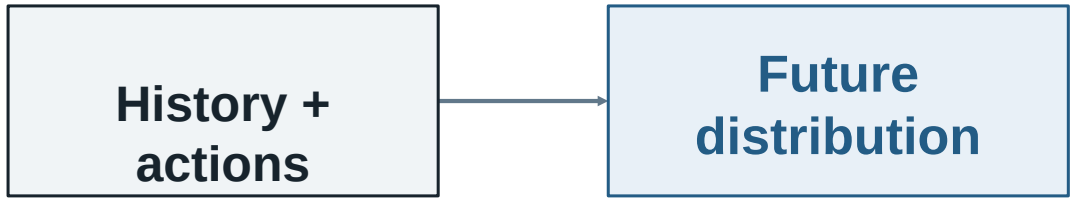
Use imagined futures to train a policy



NFWM improves in all 3 seeds. Return: 77.37 ± 9.28 (mean ± SD).

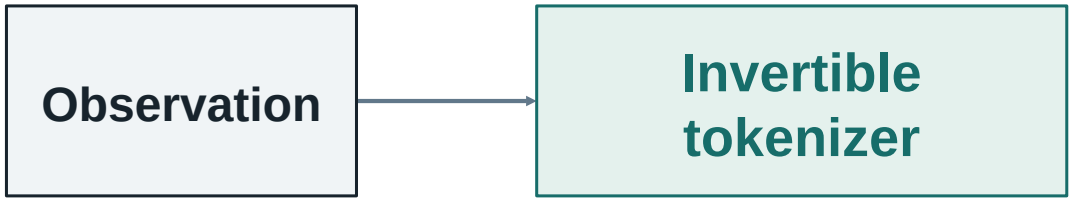
Two views of NFWM

1. A conditional predictor



Model continuous futures
in the chosen input space.

2. A learned space for prediction

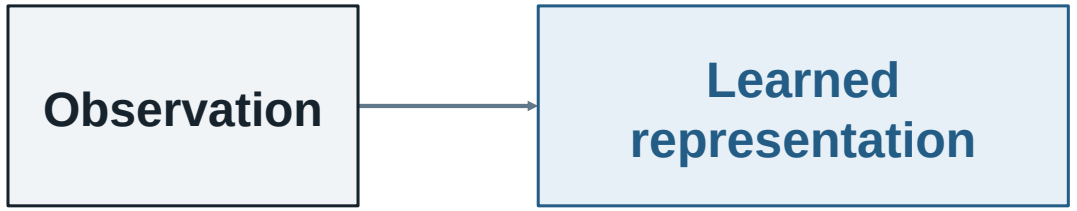


Learn coordinates for stochastic
latent dynamics and planning.

Tested uses: goal planning, action ranking and policy learning.

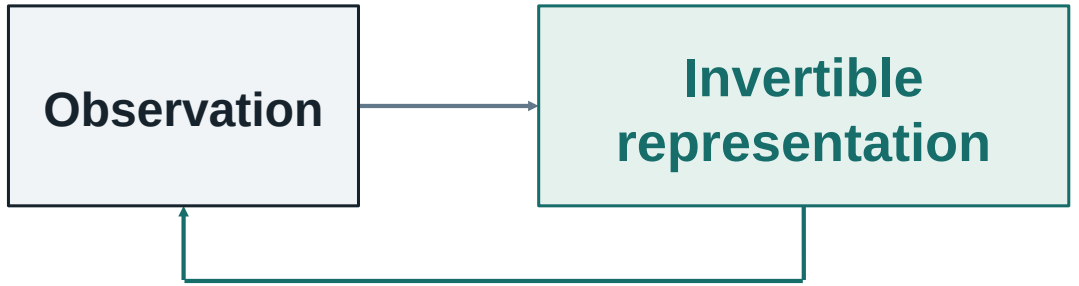
What changes when the encoder is invertible?

JEPA-style latent prediction



Predict a learned target.
Recoverability is not required.

NFWM



Predict a conditional density.
The input remains recoverable.

Keep details recoverable; learn coordinates that simplify prediction.

Reward and goal readouts choose which details matter for the task.

NFWM discussion · JEPA: Assran et al., 2023; V-JEPA 2, 2025

Does the death signal survive imagination?

Side experiment · frozen readout: P(death next), before collision

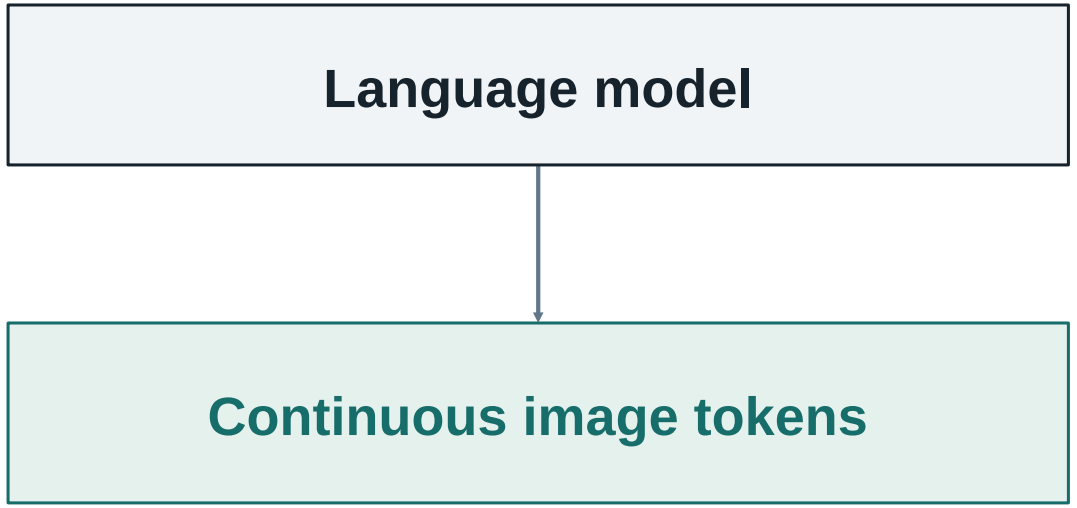
World model	Real frame	6-step imagined rollout
NFWM	**0.99**	**0.97**
MSE + SIGReg · invertible encoder	1.00	0.07
MSE + SIGReg · ViT encoder	1.00	0.05

Past first pipe: NFWM 10/10 · SIGReg + invertible 1/10 · SIGReg + ViT 0/10

All readouts detect the real-frame signal; the gap appears after rollout.

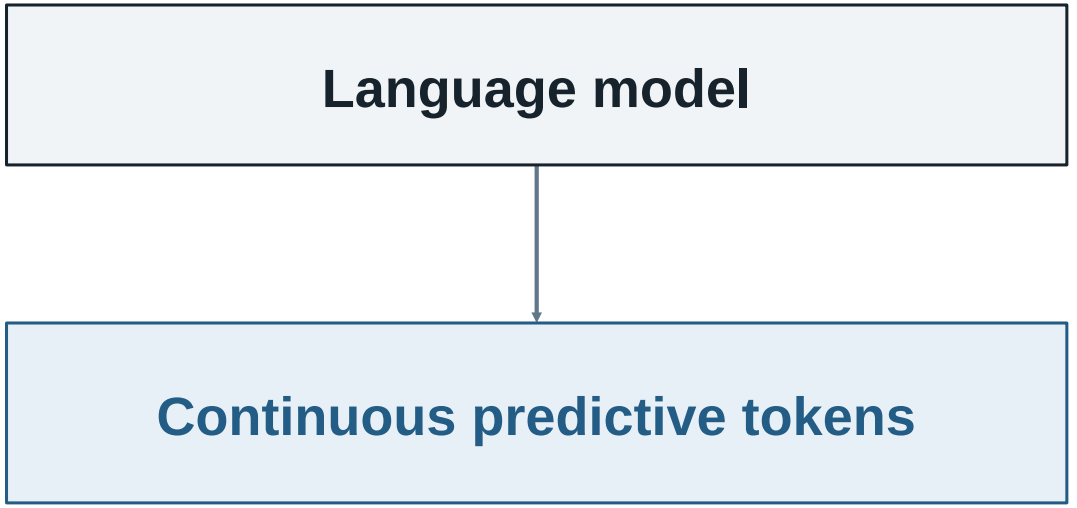
A continuous token space can connect to an LLM

Established: STARFlow2



Interleaved text and images

Direction: NFWM + language



Action-conditioned latent futures

Shared principle: conditional likelihood, with the continuous volume correction.

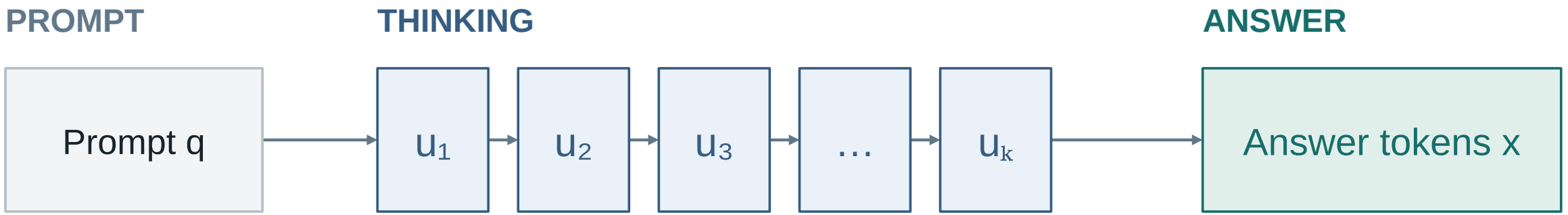
One framework, different predicted variables

Model	Input context	Predicted output
Video model	Past video	Future video
Action-conditioned world model	Past video + actions	Future video
World-action model (conceptual)	Past video	Actions + future video

State clearly what is conditioned on and what the model predicts.

Continuous thoughts before the answer

NF-CoT: the same density framework, now for continuous thoughts.



One causal backbone · Gaussian thought head + categorical answer head

$$p(u_i \mid q, u_{<i}) = \mathcal{N}\left(\mu_i, \operatorname{diag} \sigma_i^2\right)$$

Thoughts are learned from text rationales; answers remain text.

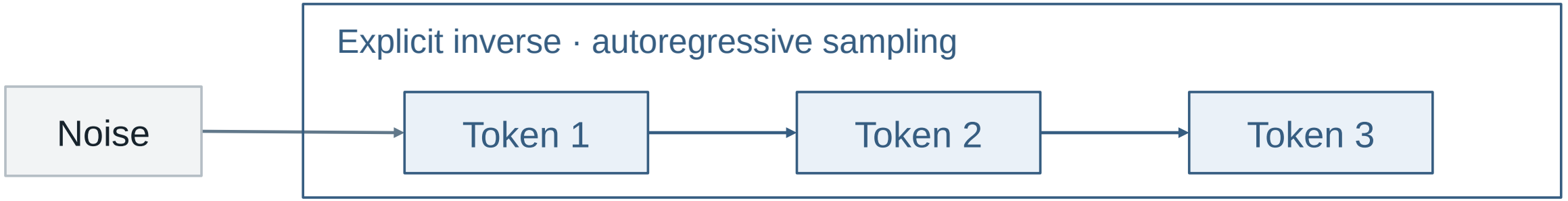
Are normalizing flows good candidates?

USEFUL STRUCTURE	CURRENT EVIDENCE	OPEN QUESTIONS
Exact likelihood	Generation and	Scaling, latency and calibration
Invertible coordinates	simulated planning	Closed-loop decision quality

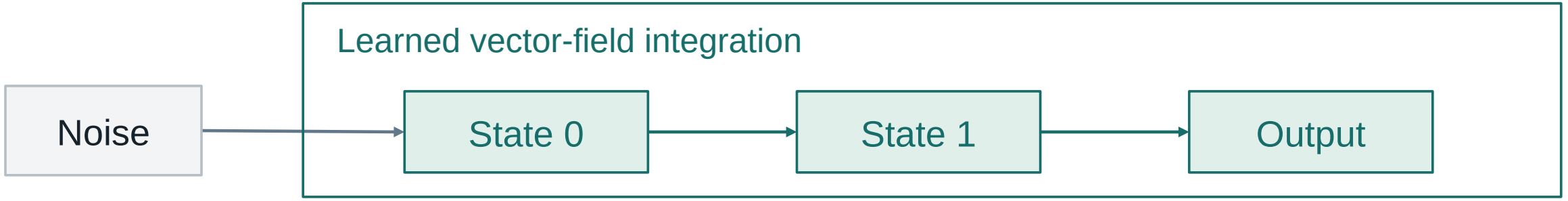
Promising candidates, with important empirical questions still open.

Where should the computation go?

STARFlow-style flow



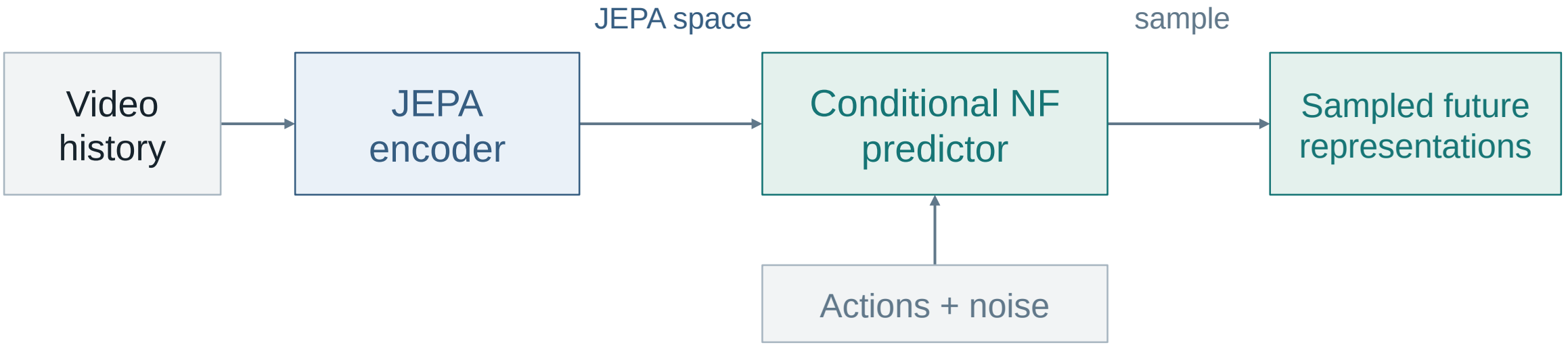
FM-trained continuous flow



At the same budget, which predicts and decides better?

NF prediction in a JEPA space

An open direction: learned abstractions with stochastic dynamics



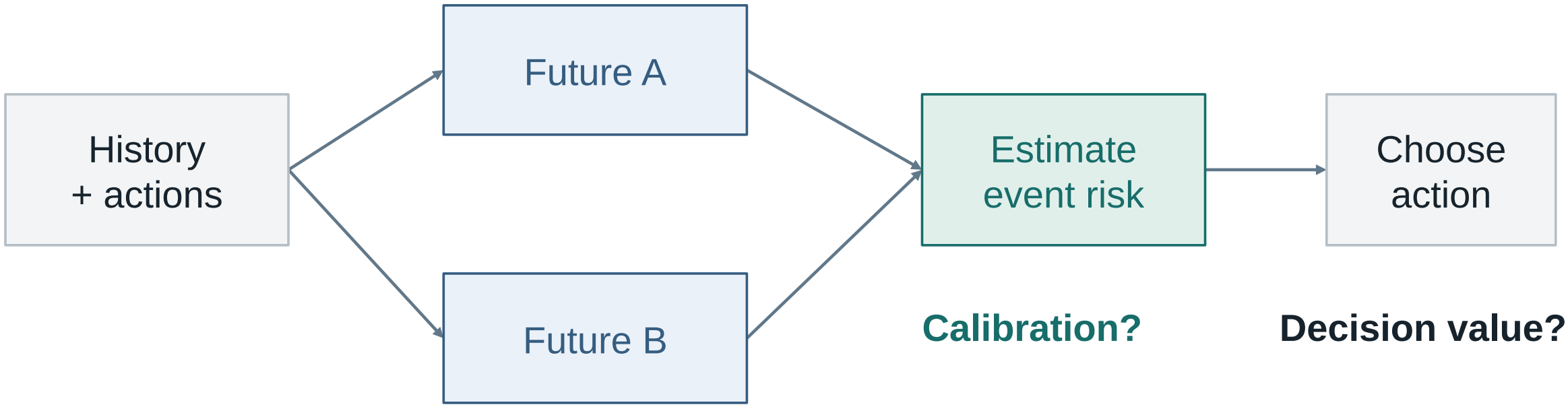
JEPA learns the representation. NF models its possible futures.

Planning still depends on what the encoder preserves.

JEPA: Assran et al., 2023 / 2025 · Proposed hybrid: likelihood in feature space

What do probabilities buy an agent?

Event risk aggregates probability across possible futures



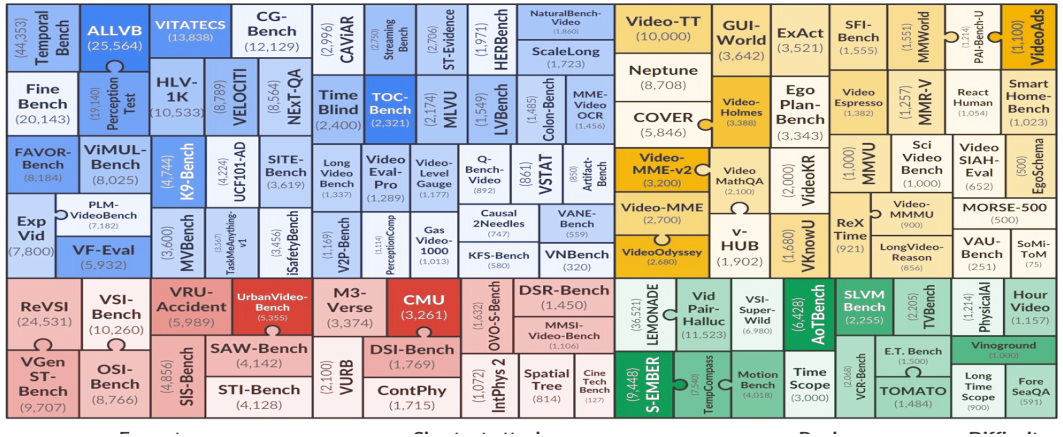
Schematic futures

Our recent work on evaluation

How do we test world understanding?

Video-Index

Does the score require video understanding?



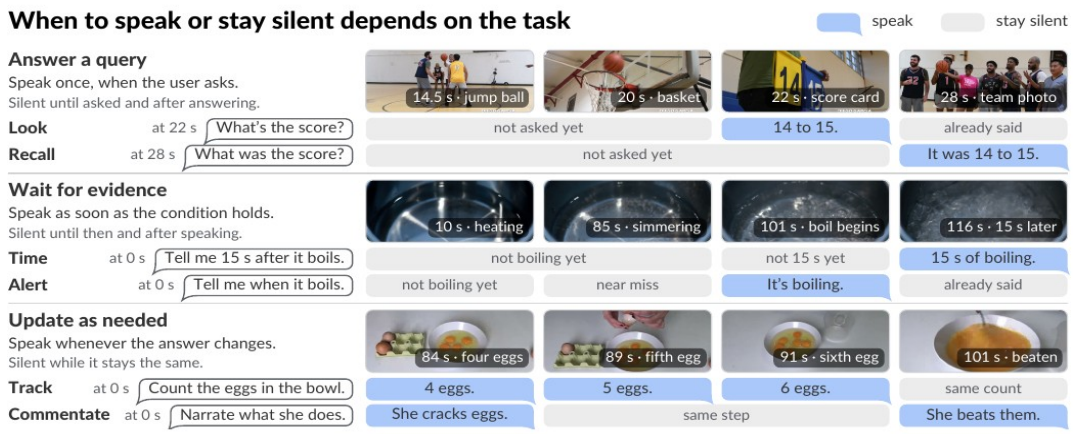
96% retained after frame shuffling

Median of full-video accuracy across
51 benchmarks with temporal probes.

Song et al. · Video-Index · 2026

InteractionBench

Can a video assistant respond at the right time?

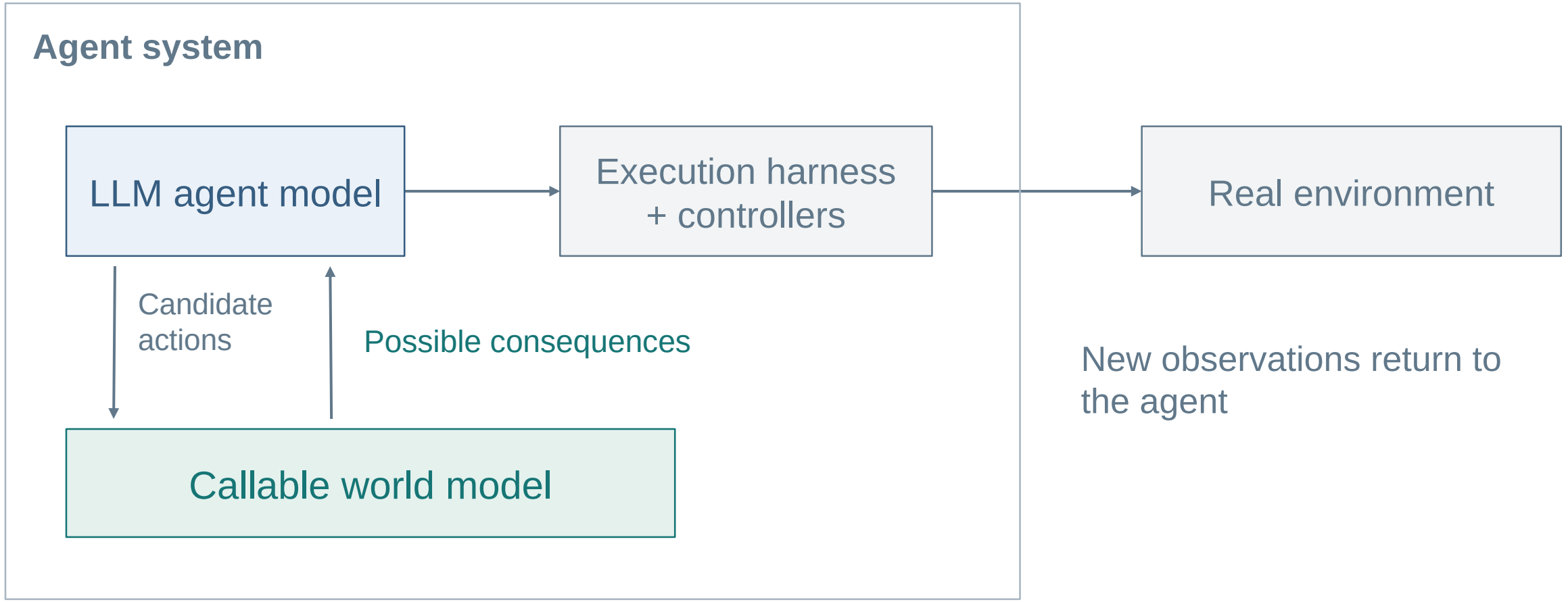


Content, timing and silence

Evaluate the complete streaming system:
model, memory and response controller.

Song et al. · InteractionBench · 2026

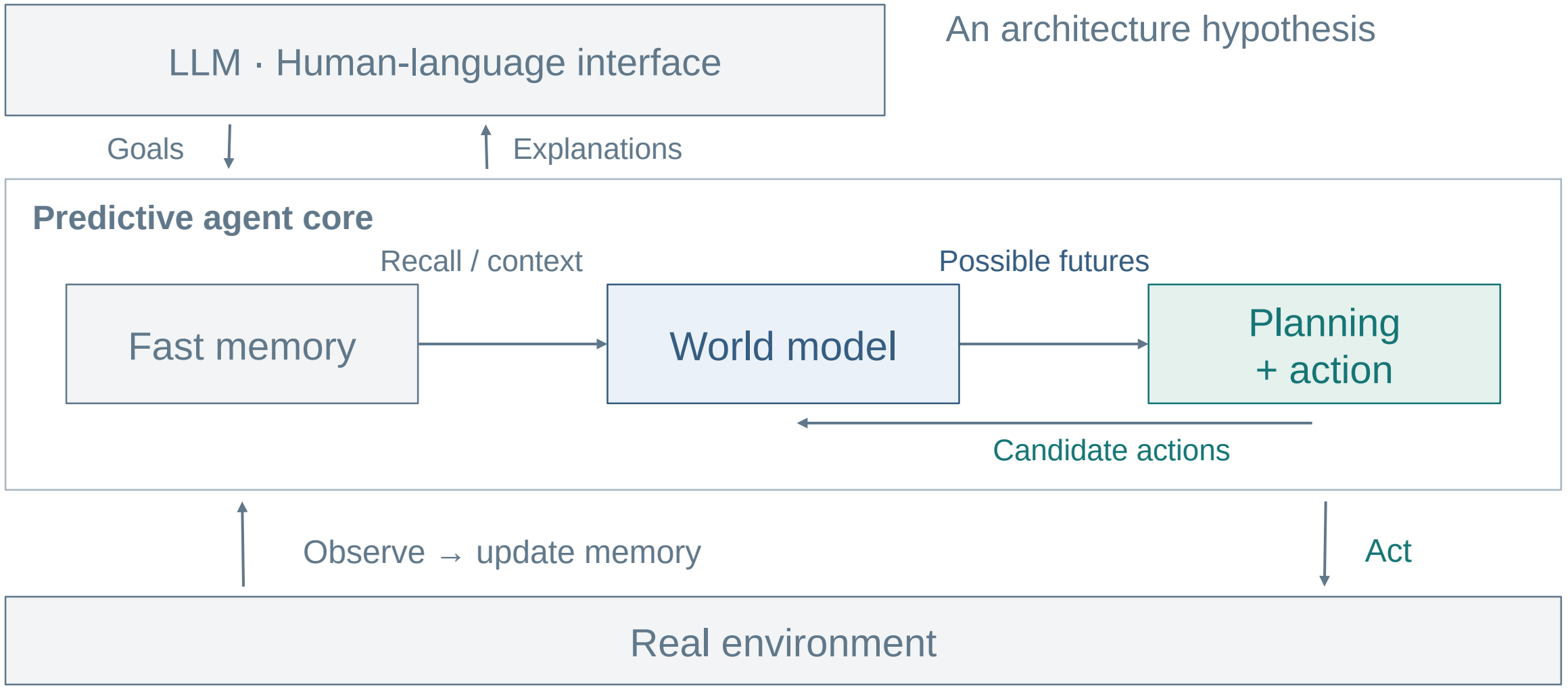
Add a world model to an LLM-centered agent



Can better predictions improve the decisions of an existing agent?

Future direction: a callable predictor within the agent system. Not a claim about Astra's internals.

LLMs as the interface, world models at the core



Interaction updates state and memory; learning from accumulated experience is a separate process.

Thank you