Machine Learning Department
School of Computer Science
Carnegie Mellon University

Interactive World Models

Matt Gormley & Aran Nayebi
Lecture 26
Dec. 3, 2025

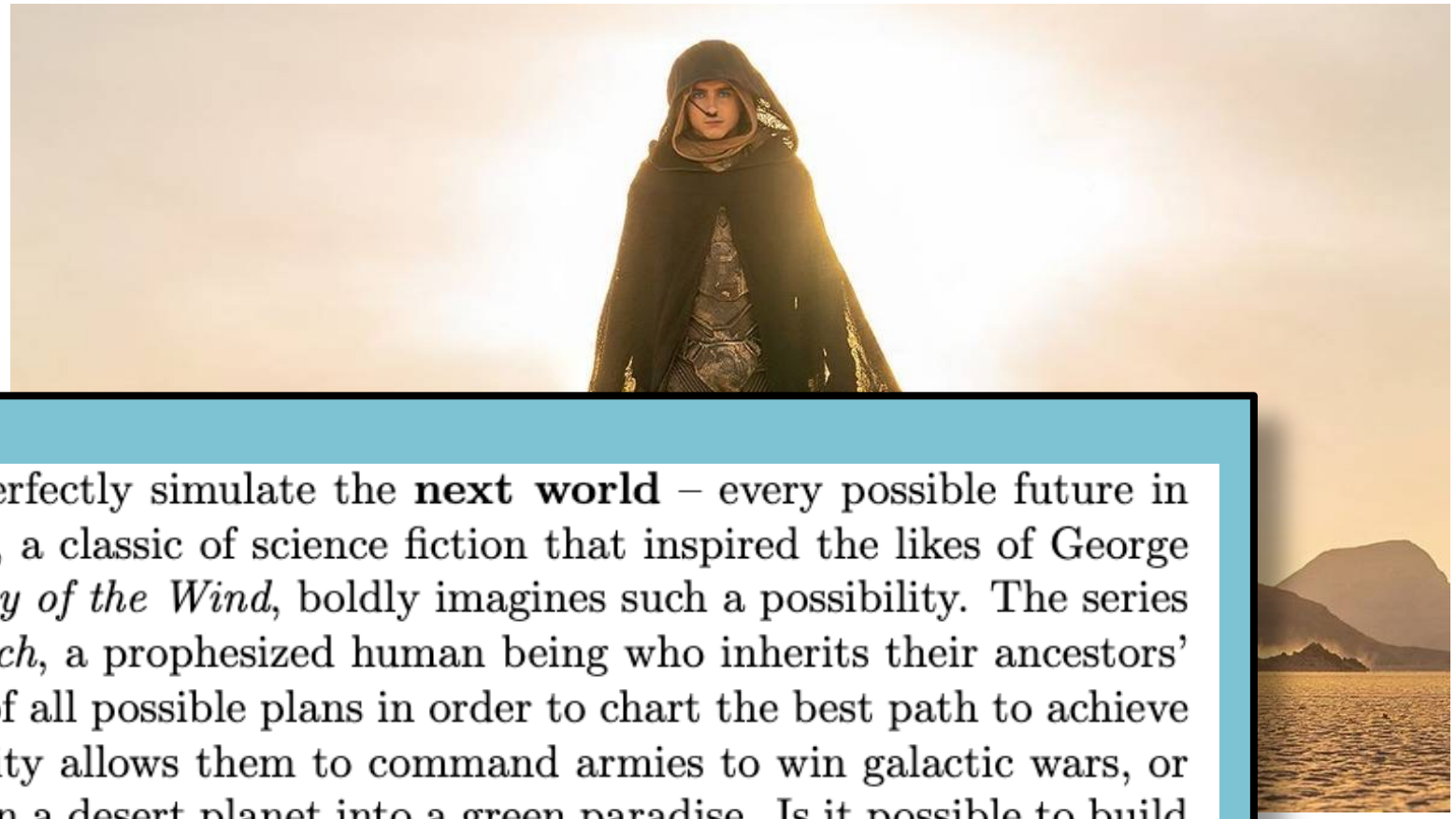
Reminders

- Project “Poster”
  - Upload Due: Sun, Dec-07 at 11:59pm
  - Presentations: Tue, Dec-09 at 8:30am-11:30am
- Project Final Report
  - Due: Thu, Dec-11 at 11:59pm
- Project Code Upload
  - Due: Thu, Dec-11 at 11:59pm

WORLD MODELS

What is a World Model (WM)?

Dune offers one notion of a world model…



What would you do if you could perfectly simulate the **next world** – every possible future in the environment that we reside? *Dune*, a classic of science fiction that inspired the likes of George Lucas' *Star Wars* and Miyazaki's *Valley of the Wind*, boldly imagines such a possibility. The series is centered around the *Kwisatz Haderach*, a prophesized human being who inherits their ancestors' memories and simulates the outcomes of all possible plans in order to chart the best path to achieve their goals [25]. Such superhuman ability allows them to command armies to win galactic wars, or to oversee global-scale projects that turn a desert planet into a green paradise. Is it possible to build towards computer systems with similar functionalities using a similar approach?

Text from http://arxiv.org/abs/2507.05169

What is a World Model (WM)?

the psychology literature has long defined our notion of a world model as “hypothetical thinking”

informally, we speak about this as a “thought experiment”

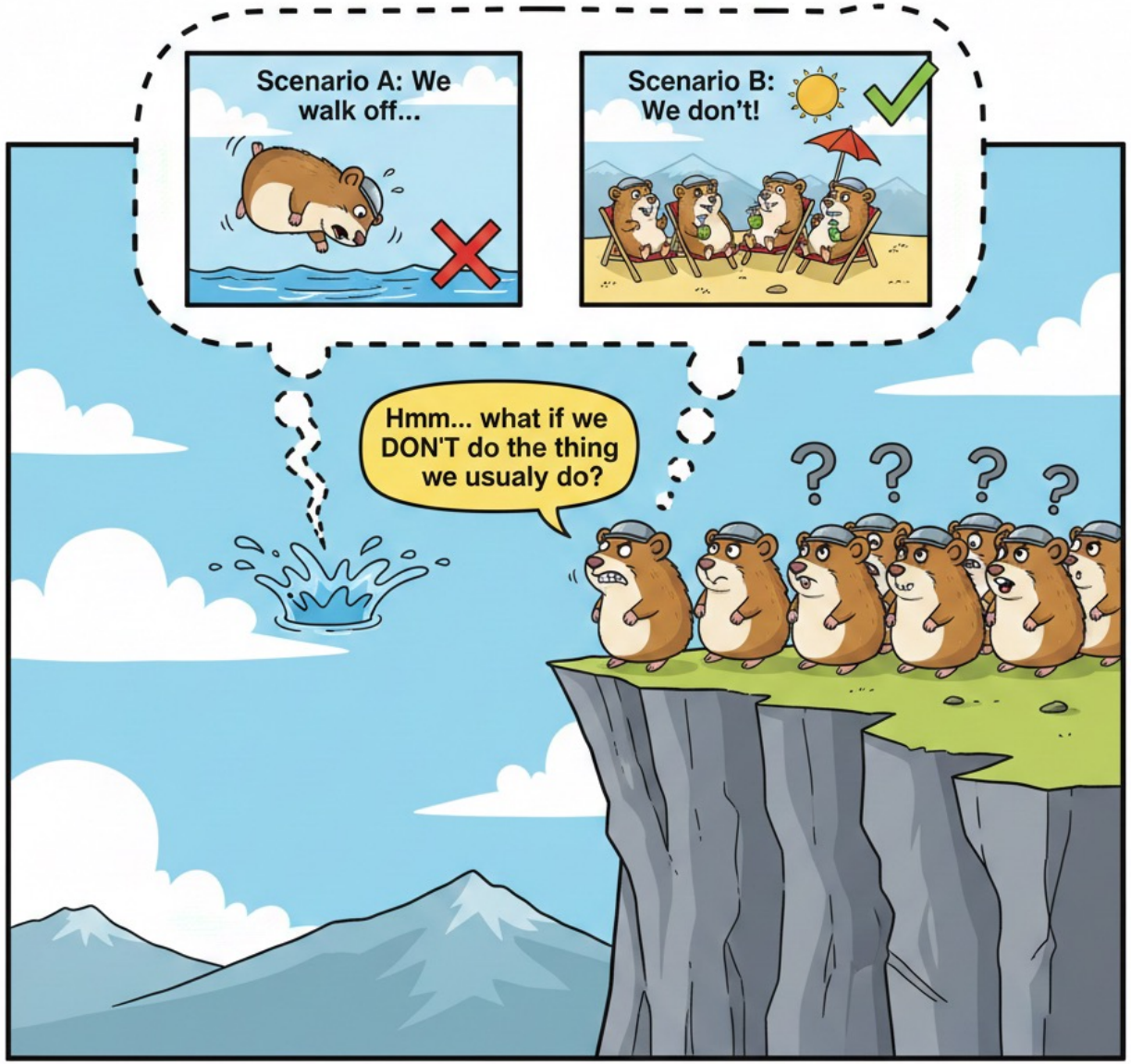


Figure generated by Nano Banana

What is a World Model (WM)?

Question: What does a world state hold?

Definition: A *world model* takes a previous world state s and an action a, and samples (or predicts) the next world state s' through a distribution (or function):

$$s' \sim p(s' \mid s, a)$$

How could we use a World Model (WM)?

1. Generate (simulated) training data for robotics
2. Generate (simulated) training data for autonomous driving
3. Create interactive worlds (aka. computer games) through prompting
4. Enable automatic generation / editing / refinement of 3D animation or graphics
5. Allow a thinking model to hypothesize about physical interactions in the real world
6. Enable faster creation of environments for virtual reality or augmented reality

Data for World Models

- Many different datasets can be used for modeling the world
- These include:
  - large scale image datasets (millions)
  - large scale video datasets (millions of hours)
  - smaller scale 3D scene datasets (see examples to the right)
  - text data
  - audio data
  - multimodal data

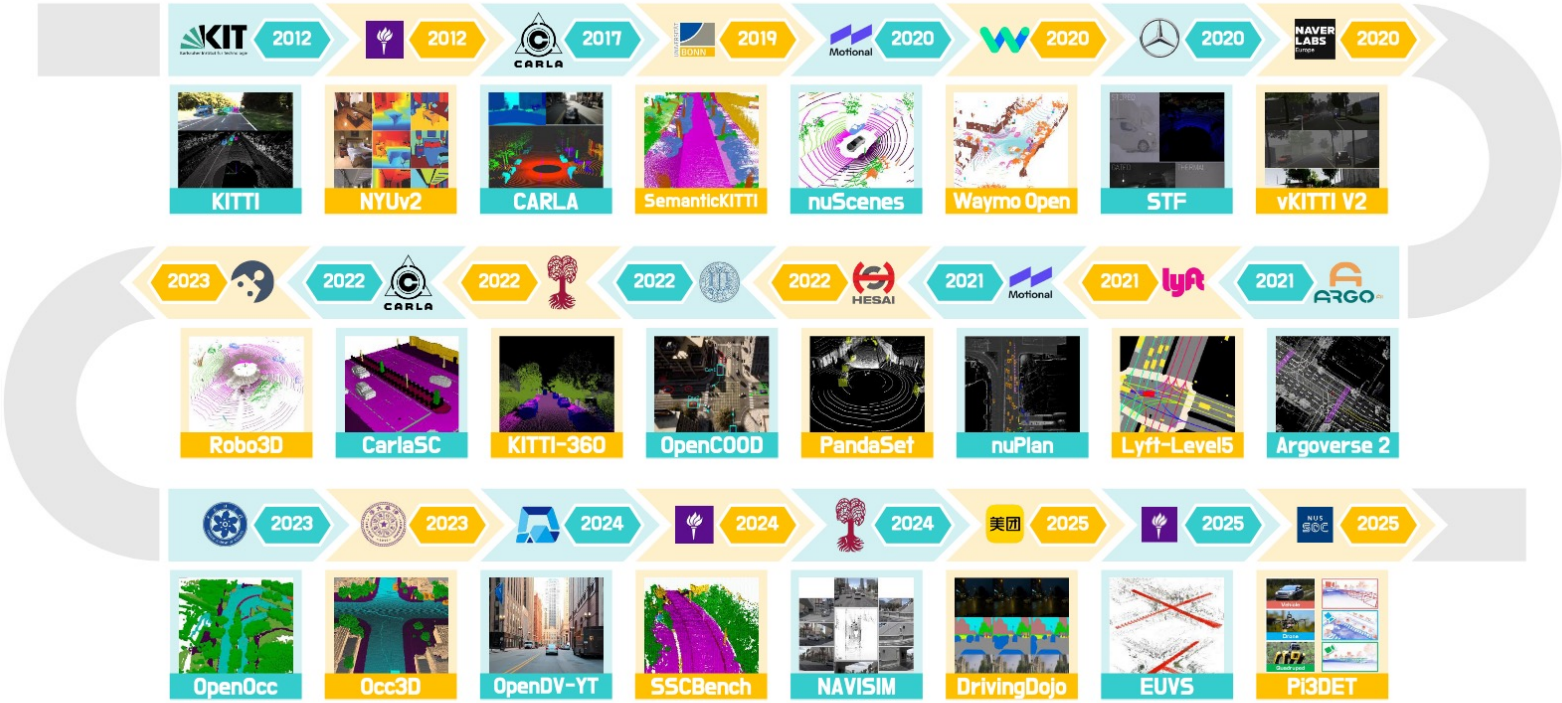


Fig. 3: Summary of existing **datasets and benchmarks** used for training and evaluating **VideoGen**, **OccGen**, and **LiDARGen** models. For detailed dataset configurations and statistics, kindly refer to Table 5. Images adopted from the original papers.

Figure from http://arxiv.org/abs/2509.07996

Three Paradigms for World Models

1. Generate a 3D Scene, then Render/Simulate

Examples:

- Marble (World Labs)
- GSGen (BNRist)
- DreamFusion (Google Research)
- NeRF-VAE (DeepMind)

Creates a representation that can be rendered using an existing renderer

2. Interactive Videos

Examples:

- Genie 1-3 (DeepMind)
- GameNGen
- Muse (Microsoft)
- Oasis (Decart AI)
- GAIA 1-2 (Wayve)

The feel of interacting with a game engine

3. Latent World Representations

Examples:

- PAN (MBZUAI)
- V-JEPA 1-2 (Meta)

Represent the world state in a low dimensional latent space (continuous, discrete, or both)

Three Paradigms for World Models

2. Interactive Videos

Examples:

- Genie 1-3 (DeepMind)
- GameNGen
- Muse (Microsoft)
- Oasis (Decart AI)
- GAIA 1-2 (Wayve)

The feel of interacting with a game engine

4. Text-to-video

Examples:

- Gen 1-3 (Runway)
- Sora (OpenAI)
- Veo (Google)

Prompt for a video

These represent a fourth class of world models, but we set them aside for the purposes of our discussion here

Three Paradigms for World Models

1. Generate a 3D Scene, then Render/Simulate

2. Interactive Videos

3. Latent World Representations

GENERATING 3D SCENES

NeRF and Gaussian Splatting

- Goals for both…
  - synthesize 3D scene from 2D images
  - define a **differentiable** renderer
- **NeRF**
  - Continuous neural field mapping {3D position, view direction} to {density, emitted radiance}
  - *Drawback:* volumetric neural rendering is slow and not ideal for real-time / interactive use
- **Gaussian Splatting**
  - Represent scene as a cloud of 3D Gaussians (aka. splats), each with position, shape, opacity, color
  - Enables real-time or near-real-time rendering via efficient rasterization of splats

NeRF:

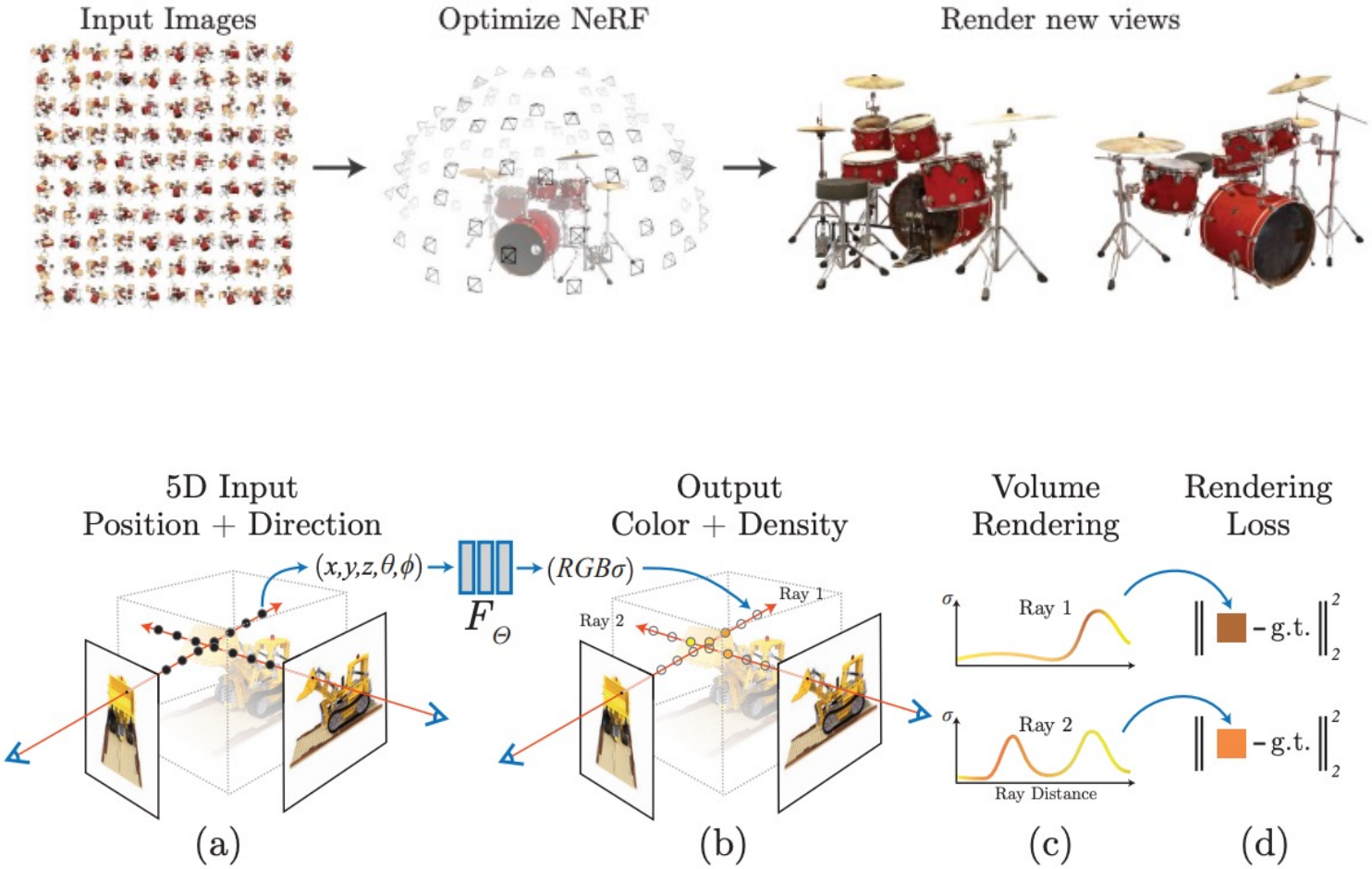


Figure from https://arxiv.org/pdf/2003.08934

NeRF and Gaussian Splatting

- Goals for both…
  - synthesize 3D scene from 2D images
  - define a **differentiable** renderer
- **NeRF**
  - Continuous neural field mapping {3D position, view direction} to {density, emitted radiance}
  - *Drawback:* volumetric neural rendering is slow and not ideal for real-time / interactive use
- **Gaussian Splatting**
  - Represent scene as a cloud of 3D Gaussians (aka. splats), each with position, shape, opacity, color
  - Enables real-time or near-real-time rendering via efficient rasterization of splats

Gaussian Splatting:

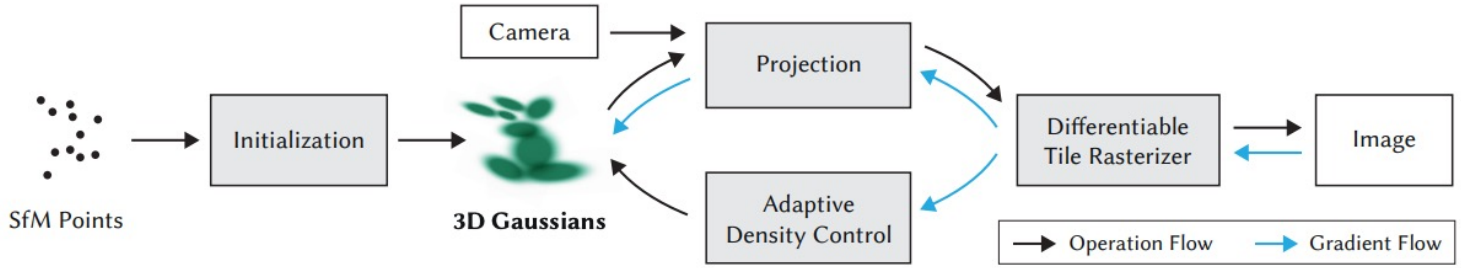


Figure from https://arxiv.org/pdf/2308.04079

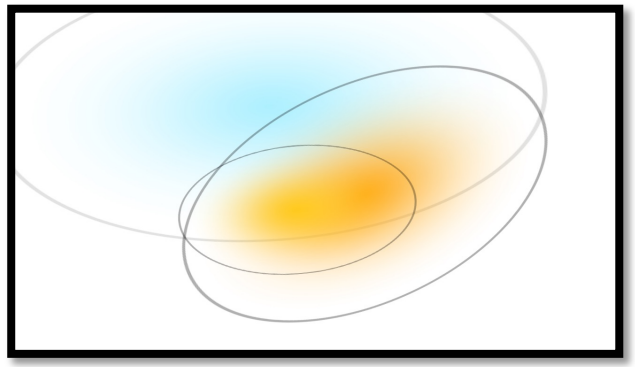


Figure from https://huggingface.co/blog/gaussian-splatting

DreamFusion

- DreamFusion is a text-to-3D-scene model
- Key idea: directly generate the NeRF parameterization and train it from scratch for each caption
- Because NeRF is differentiable we can get away with this

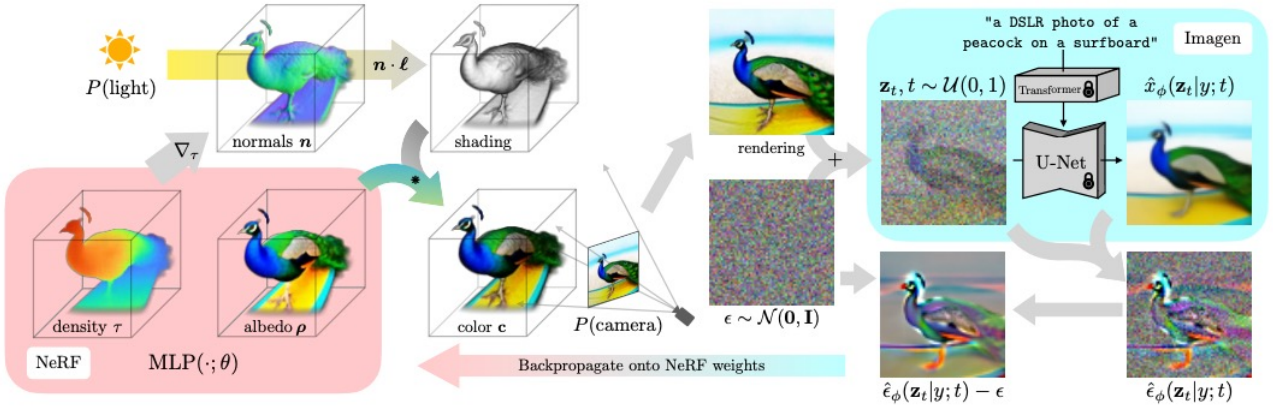
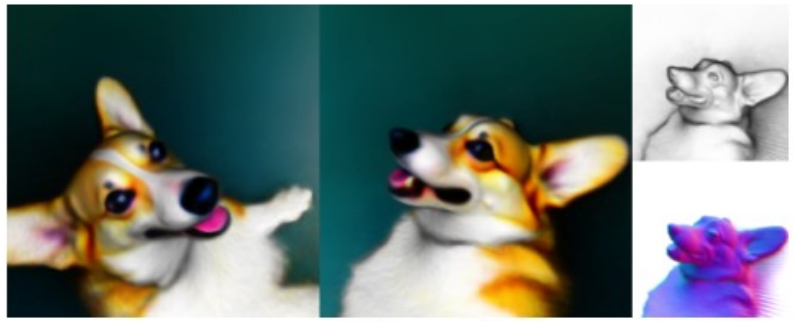


Figure 3: DreamFusion generates 3D objects from a natural language caption such as *"a DSLR photo of a peacock on a surfboard."* The scene is represented by a Neural Radiance Field that is randomly initialized and trained from scratch for each caption. Our NeRF parameterizes volumetric density and albedo (color) with an MLP. We render the NeRF from a random camera, using normals computed from gradients of the density to shade the scene with a random lighting direction. Shading reveals geometric details that are ambiguous from a single viewpoint. To compute parameter updates, DreamFusion diffuses the rendering and reconstructs it with a (frozen) conditional Imagen model to predict the injected noise $\hat{\epsilon}_\phi(\mathbf{z}_t|y;t)$. This contains structure that should improve fidelity, but is high variance. Subtracting the injected noise produces a low variance update direction $\text{stopgrad}[\hat{\epsilon}_\phi - \epsilon]$ that is backpropagated through the rendering process to update the NeRF MLP parameters.

an orangutan making a clay bowl on a throwing wheel*



a corgi taking a selfie*

Demos: https://dreamfusion3d.github.io/

Figure from https://arxiv.org/pdf/2209.14988

GSGen

- **GSGen** is a text-to-3D scene generator, but unlike DreamFusion it generates the Gaussian splatting instead of NeRF

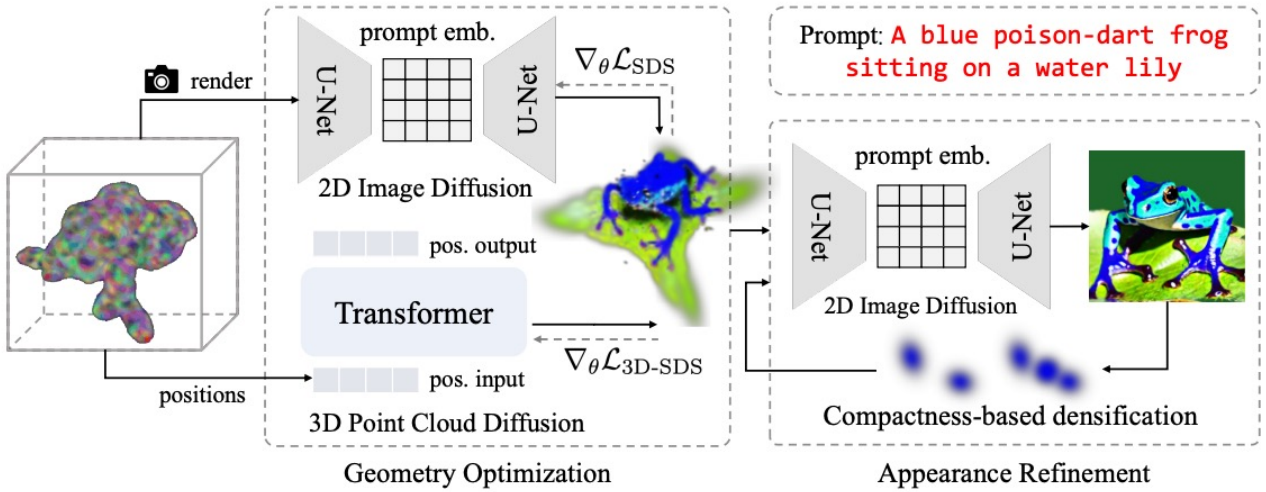


Figure 3. **Overview of the proposed GSGEN.** Our approach aims at generating 3D assets with accurate geometry and delicate appearance. GSGEN starts by utilizing Point-E to initialize the positions of the Gaussians (Sec 4.3). The optimization is grouped into geometry optimization (Sec 4.1) and appearance refinement (Sec 4.2) to meet a balance between coherent geometry structure and detailed texture.

Figure from https://arxiv.org/pdf/2309.16585

Marble

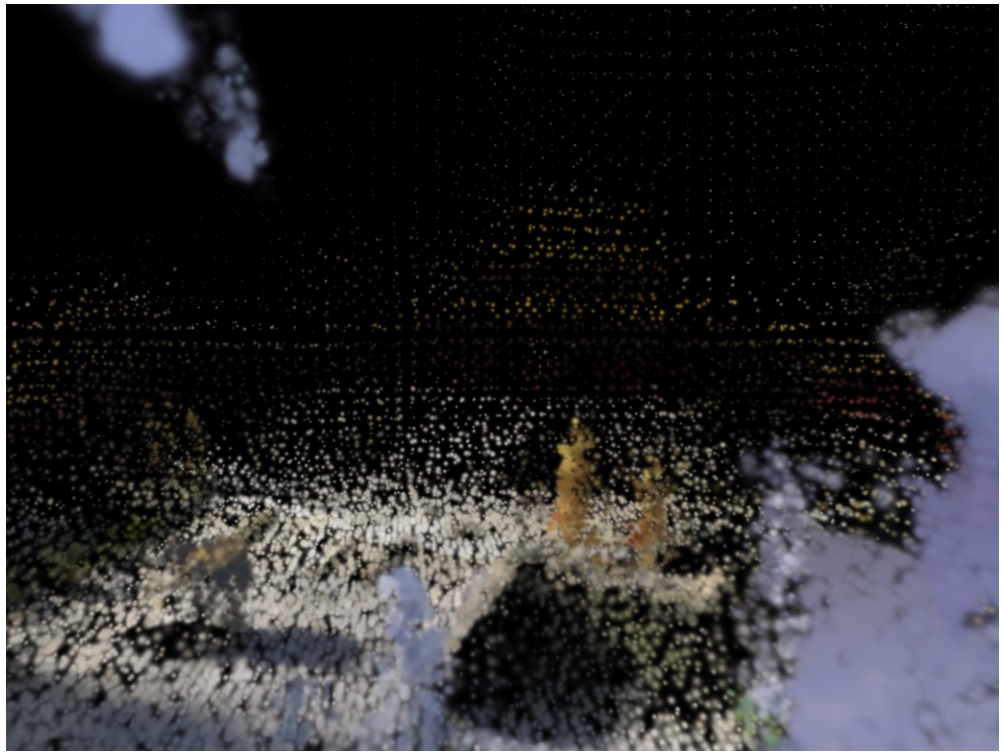
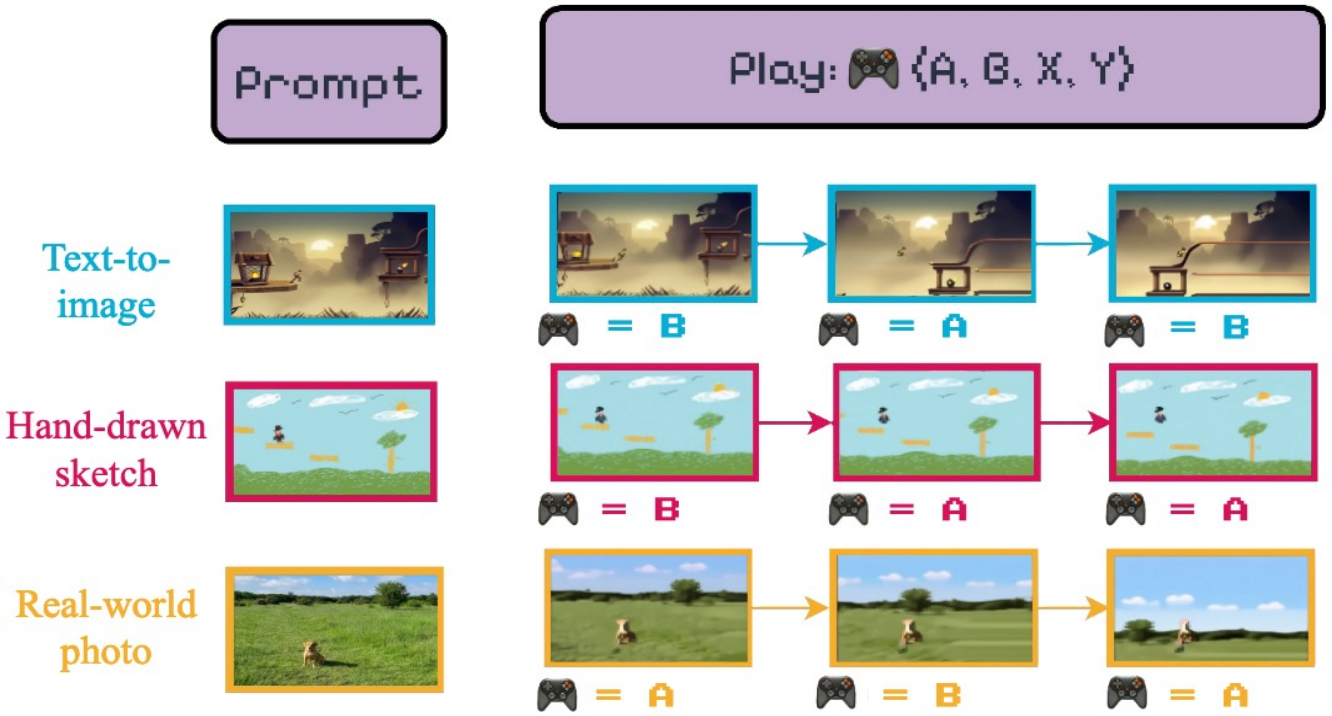


Figure from https://marble.worldlabs.ai/

INTERACTIVE GENERATIVE VIDEO MODELS

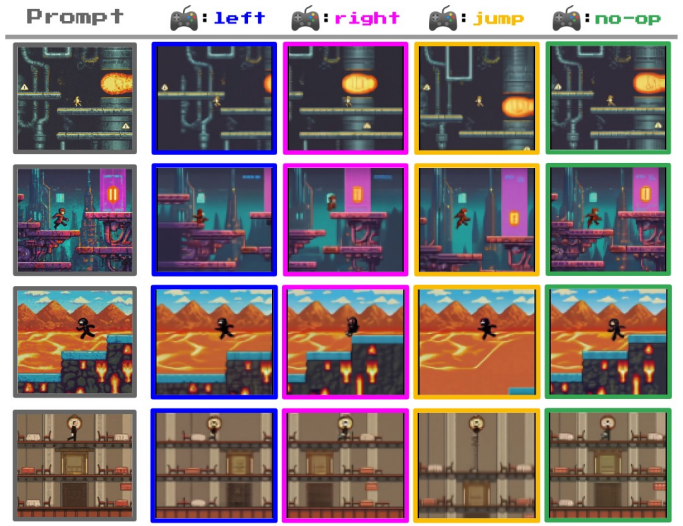
Genie-1



- Genie is an interactive video generation model
- At test time:
  - It takes a prompt (image) as input
  - At each time step it accepts an action given by the user
  - Given the current frame and the action, it generates the next frame
- This simulates the style of interaction one would have with a video game

Figure from http://arxiv.org/abs/2402.15391

Genie-1



- **Primary training data:** 6.8M 16s videos (30k hours) of 2D platformer games
- **Model:** three components
  1. Video tokenizer/detokenizer
  2. Latent action model
  3. Dynamics model
- **Size:** 11B parameters

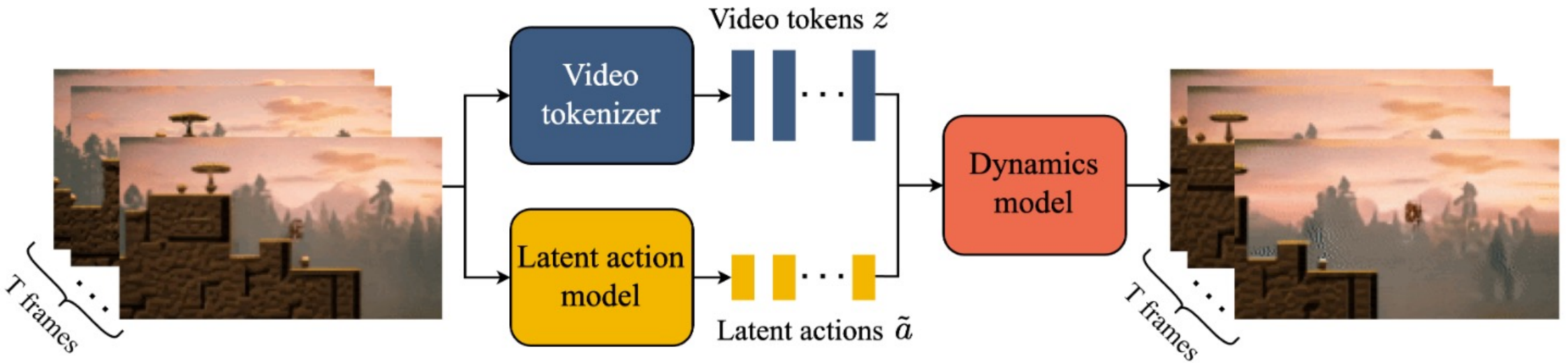
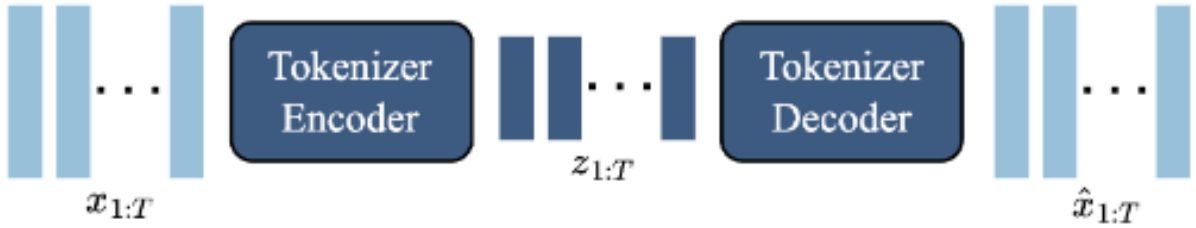


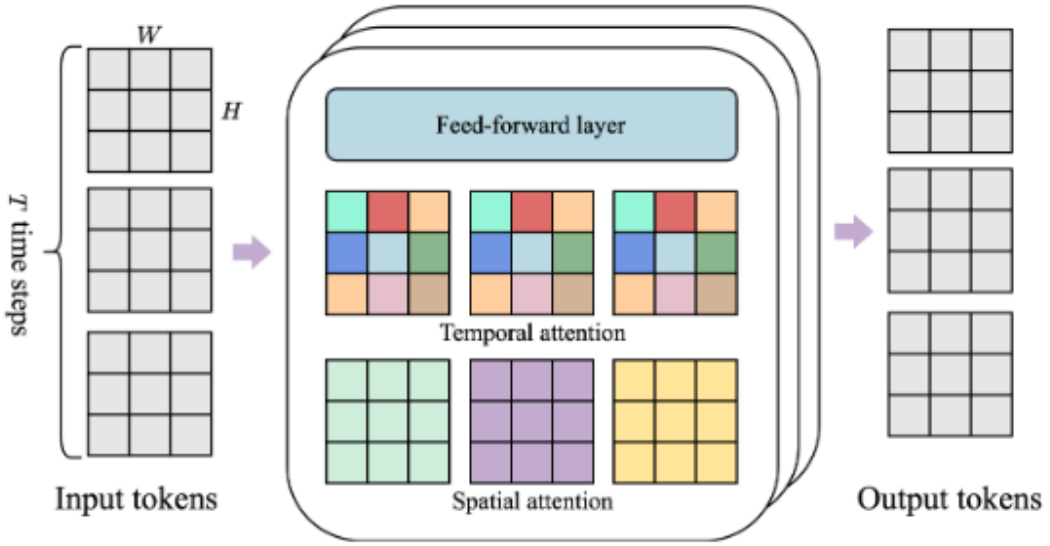
Figure from http://arxiv.org/abs/2402.15391

Genie-1 Model

Video tokenizer / detokenizer:



ST-Transformer:



Video tokenizer / detokenizer:

- VQ-VAE that encodes *T* frames of video:

$$\boldsymbol{x}_{1:T} = (x_1, x_2, \cdots, x_T) \in \mathbb{R}^{T \times H \times W \times C}$$

- into *T* discrete tokens (from vocab of size D)

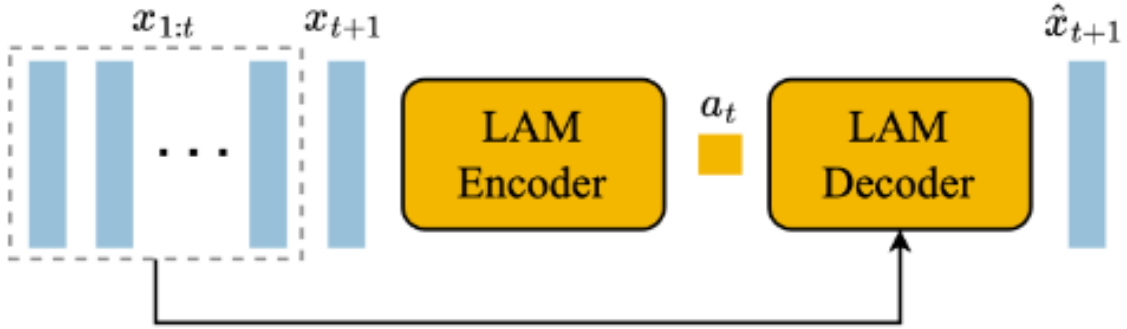
$$\boldsymbol{z}_{1:T} = (z_1, z_2, \cdots, z_T) \in \mathbb{I}^{T \times D}$$

- Trained via standard VQ-VAE objective over video dataset
- *Backbone model:*
  - ST-Transformer, which scales well for longer sequences

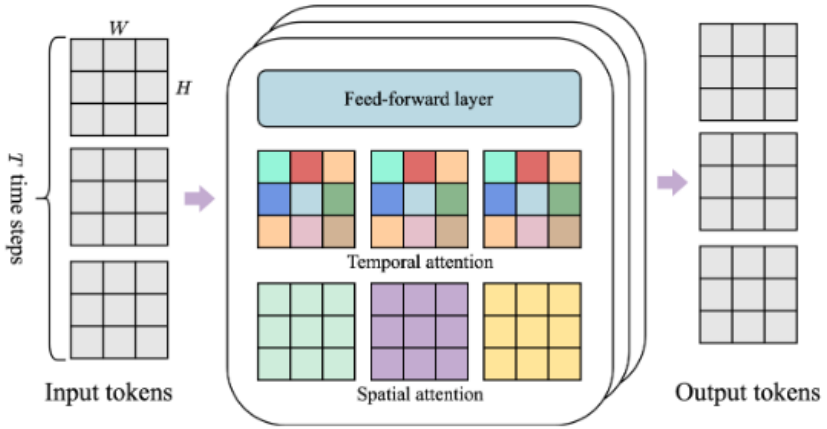
Figure from http://arxiv.org/abs/2402.15391

Genie-1 Model

Latent Action Model:



ST-Transformer:



Latent Action Model (LAM):

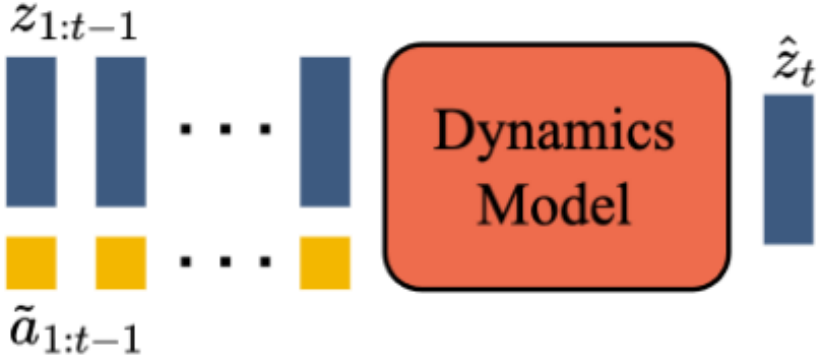
- An unsupervised model for learning *latent* actions from unlabled videos
- Another VQ-VAE model
  - Encoder: converts t frames of video plus the t+1 frame ($\mathbf{x}_{1:t+1}$) into sequence of t+1 actions ($\mathbf{a}_{1:t+1}$)
  - Decoder: converts t frames ($\mathbf{x}_{1:t}$) and t actions ($\mathbf{a}_{1:t}$) into the t+1 frame ($\mathbf{x}_{t+1}$)
- At test time, this model is discarded since actions come from a human user

- *Backbone model:* ST-Transformer

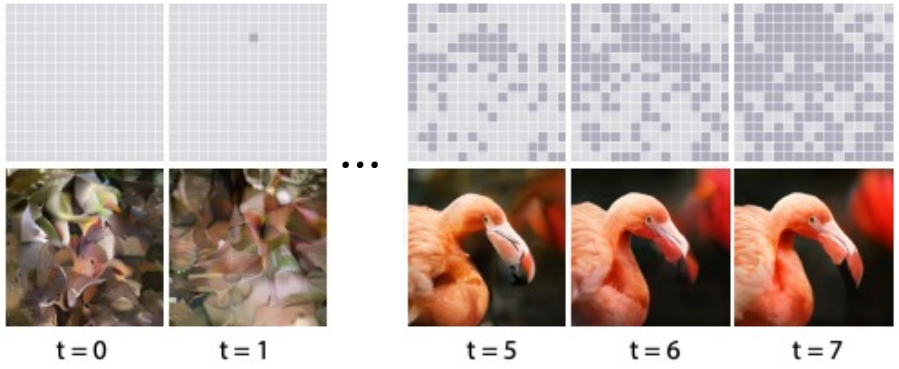
Figure from http://arxiv.org/abs/2402.15391

Genie-1 Model

Latent Action Model:



Decoder-only MaskGIT:



Dynamics Model

- Decoder-only MaskGIT
- Conditions on the sequence of previous video tokens ($z_{1:t}$) and actions ($\mathbf{a}_{1:t}$) to generate the next video token (z_{t+1}), which can be detokenized into a video frame ($\mathbf{x}_{t+1}$)
- Trained via cross-entropy for next token prediction
- *Note*: Action embeddings are incorporated **additively** in the dynamics model (and LAM), rather than via **concatenation**
- *Backbone model:* ST-Transformer

Figure from https://arxiv.org/pdf/2202.04200

Figure from http://arxiv.org/abs/2402.15391

Genie-1 at Test Time

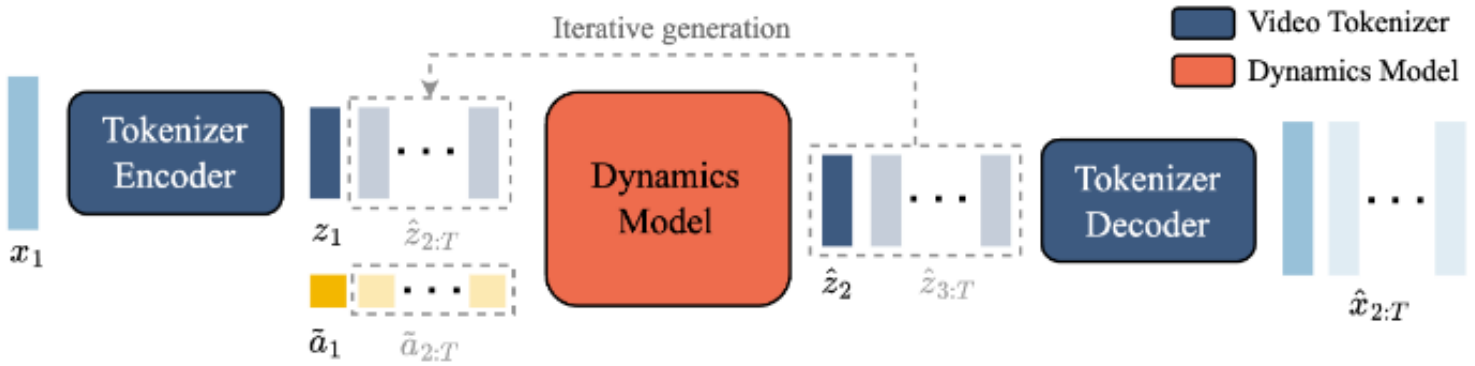


Figure 8 | **Genie Inference**: the prompt frame is tokenized, combined with the latent action taken by the user, and passed to the dynamics model for iterative generation. The predicted frame tokens are then decoded back to image space via the tokenizer's decoder.

Figure from http://arxiv.org/abs/2402.15391

Genie-1 Qualitative Results

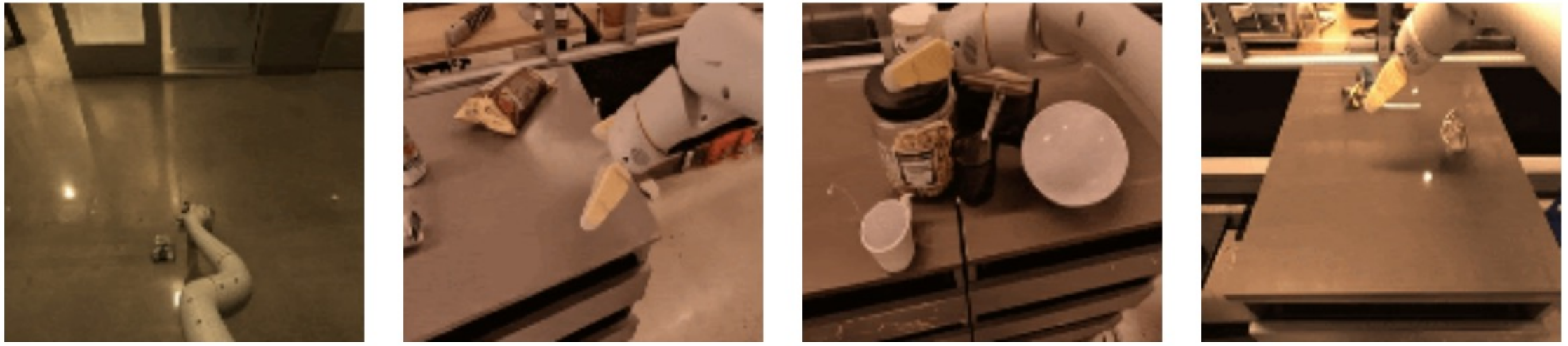
- Images created with a text-to-image model can serve as a prompt.
- Then Genie-1 enables a user to interact directly with the environment.
- The latent actions are mapped to actual keys (a, s, d, f, etc.) so that the user can explore what they do.

Figure from https://sites.google.com/view/genie-2024/home

Genie-1 Application to Robotics

- Robotics dataset:
  - 130k robotics demonstrations
  - a simulation dataset (size unspecified)
  - 209k episodes of real robot data
- No actions included, just videos

- Training Genie on the robotics datasets learns a new model that can simulate "game" style play with a robotic arm as shown below



latent actions: 0, 0, 1, 2, 6, 1, 1, 4, 0, 0

Figure from https://sites.google.com/view/genie-2024/home

Genie-1: Are the Latent Actions Transferable?

- CoinRun is a procedurally generated platformer
- Models:
  1. **Behavior Cloning:** train a policy given a dataset of each frame paired with the expert action (skyline)
  2. **Random Agent:** always take a random action (baseline)
  3. **LAM-based Policy:** train a policy given a dataset of each frame paired with the Genie latent action; separately learn a mapping from latent actions to expert actions from a few examples

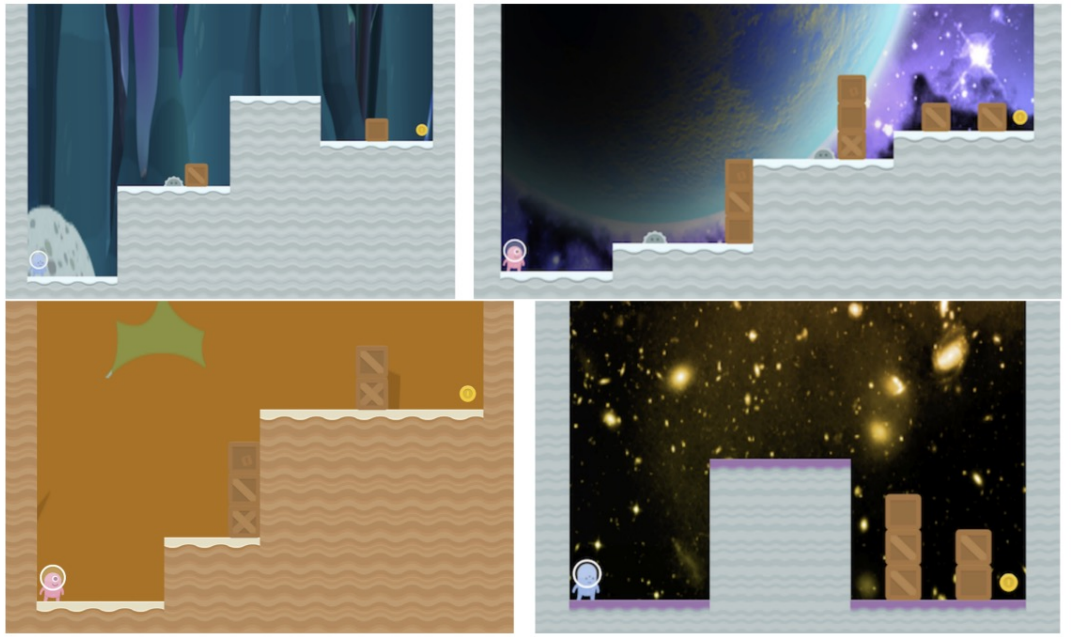


Fig. 1: Four generated CoinRun levels with the coin on the right.

Figure from https://sites.google.com/view/genie-2024/home

Genie-1: Are the Latent Actions Transferable?

- CoinRun is a procedurally generated platformer
- Models:
  1. **Behavior Cloning:** train a policy given a dataset of each frame paired with the expert action (skyline)
  2. **Random Agent:** always take a random action (baseline)
  3. **LAM-based Policy:** train a policy given a dataset of each frame paired with the Genie latent action; separately learn a mapping from latent actions to expert actions from a few examples

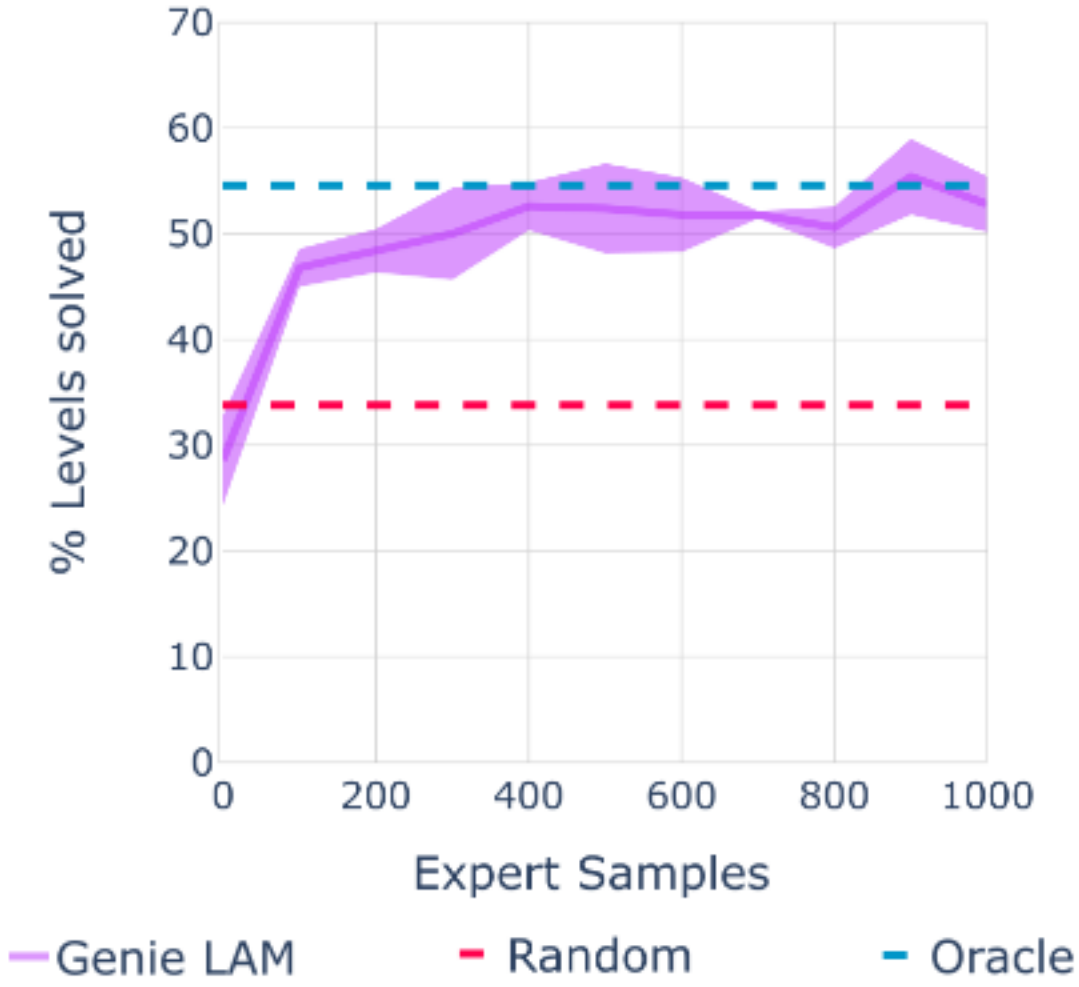
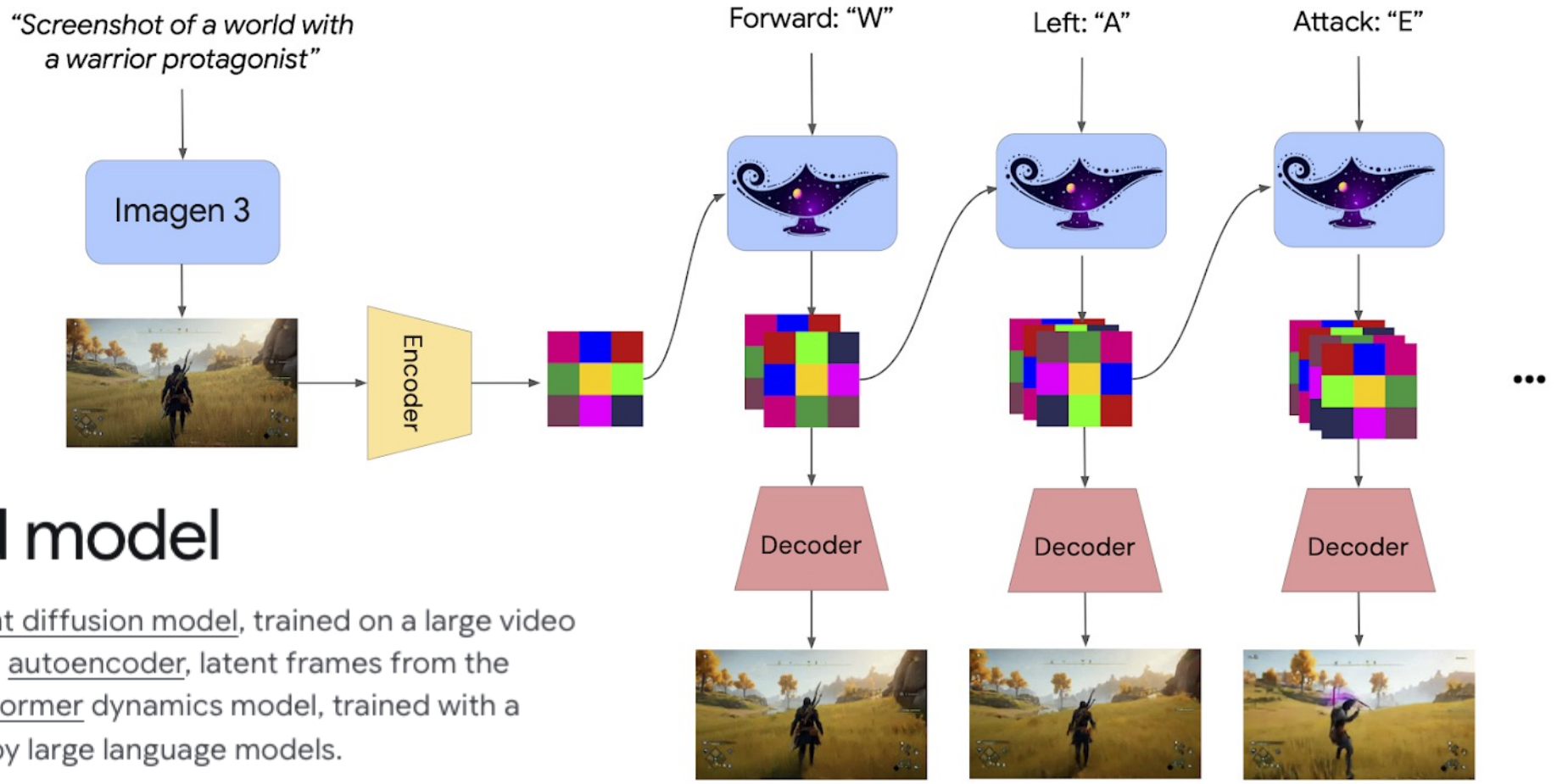


Figure from https://sites.google.com/view/genie-2024/home

Genie-2



Diffusion world model

Genie 2 is an autoregressive latent diffusion model, trained on a large video dataset. After passing through an autoencoder, latent frames from the video are passed to a large transformer dynamics model, trained with a causal mask similar to that used by large language models.

At inference time, Genie 2 can be sampled in an autoregressive fashion, taking individual actions and past latent frames on a frame-by-frame basis. We use classifier-free guidance to improve action controllability.

The samples in this blog post are generated by an undistilled base model, to show what is possible. We can play a distilled version in real-time with a reduction in quality of the outputs.

Figure from https://deepmind.google/blog/genie-2-a-large-scale-foundation-world-model/

Genie-2

- The technical description of Genie-2 is rather vague
- This could indicate that the primary change is **scale**
  - **more parameters**
  - **more training data**

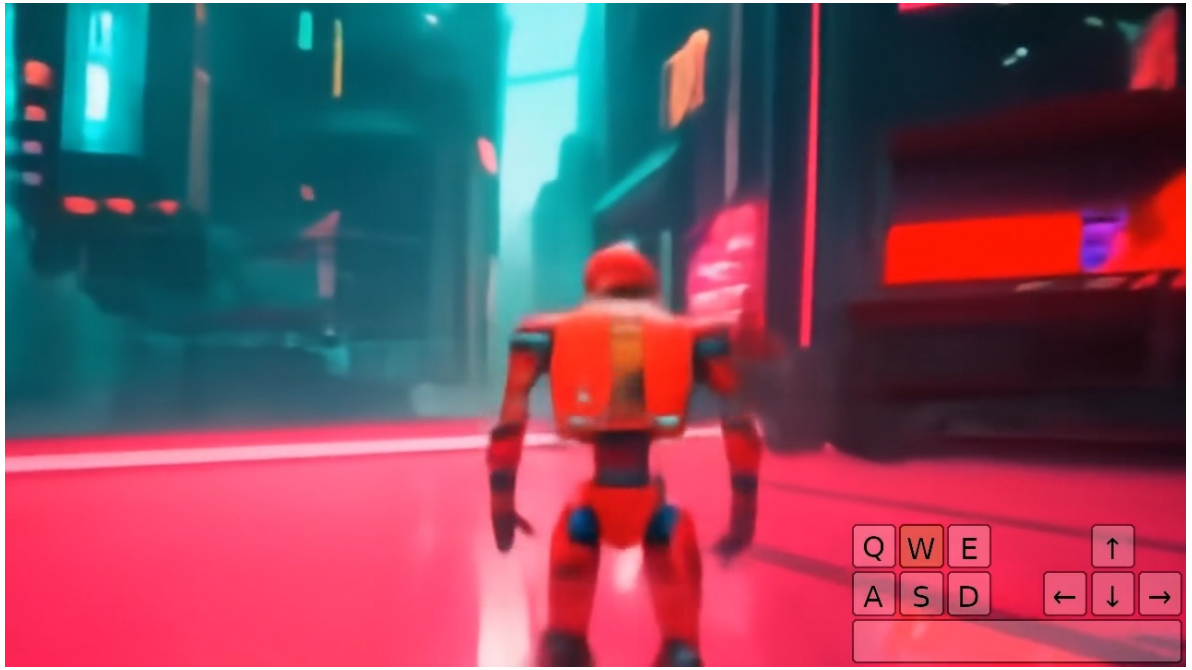


Figure from https://deepmind.google/blog/genie-2-a-large-scale-foundation-world-model/

Genie-3

	GameNGen	Genie 2	Veo	Genie 3
Resolution	320p	360p	720p to 4K	720p
Domain	Game-specific	3D Environments	General	General
Control	Game-specific	Limited keyboard / mouse actions	Video-level description*	Navigation; Promptable world events
Interaction Horizon	A few seconds	10-20 seconds	8 seconds	Multiple minutes
Interaction Latency	Real time	Not real time	N/A	Real time

*Additional controls such as references, style and camera are available.

Figure from https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/

Genie-3

Figure from https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/

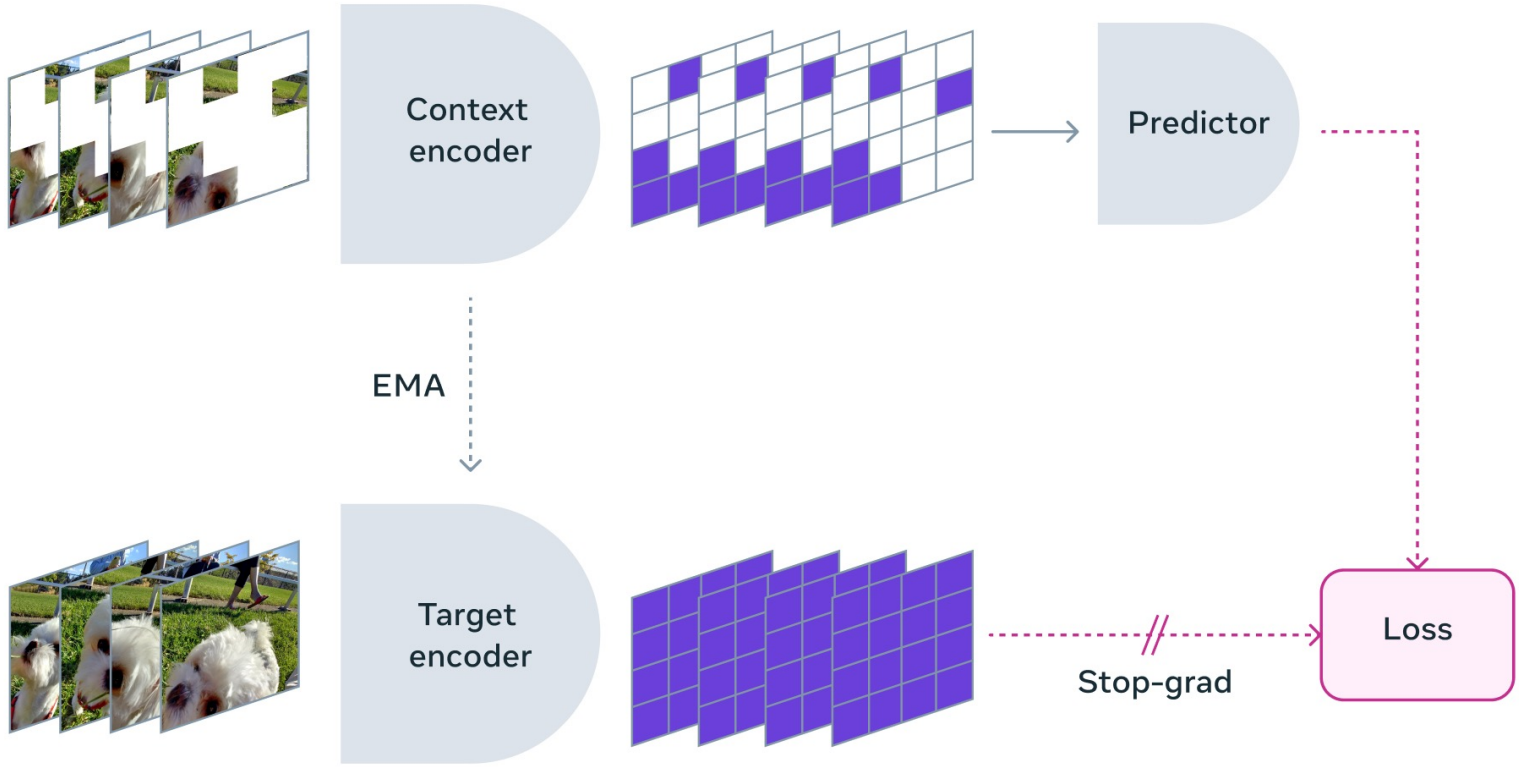
Genie 3 Limitations

- Interaction window is limited to a short duration (a few minutes)
- Actions (are likely) latent and come from a fixed set; defining new actions is not possible
- Prompting for changes in the world is possible and represents an open set; thus representing an asymmetry between the closed action set and the open world change set
- Limited to one character (at least that seems to be the case)
- Fundamentally just a video model: no game-engine-interpretable representation of the world (e.g. point cloud, mesh, Gaussian splatting) is provided
- Super high computational demands

LATENT WORLD REPRESENTATIONS

V-JEPA 1-2

- A **non-generative** world model
- **Key idea:**
  - always work in **latent space**
  - minimize loss between:
    - the latent representation of true masked regions from raw video AND
    - the latent representation of the predicted masked regions from masked video
  - the loss itself is between the latents E(x) and E(y)
- **Goal:** to transfer latent representation to other tasks



$$\text{minimize}_{\theta,\phi,\Delta_y} \quad \|P_\phi(\Delta_y, E_\theta(x)) - \text{sg}(E_{\bar{\theta}}(y))\|_1,$$

Figure from https://arxiv.org/abs/2506.09985

Figure from https://ai.meta.com/blog/v-jepa-yann-lecun-ai-model-video-joint-embedding-predictive-architecture/

V-JEPA 1-2

- A **non-generative** world model
- **Key idea:**
  - always work in **latent space**
  - minimize loss between:
    - the latent representation of true masked regions from raw video AND
    - the latent representation of the predicted masked regions from masked video
  - the loss itself is between the latents E(x) and E(y)
- **Goal:** to transfer latent representation to other tasks

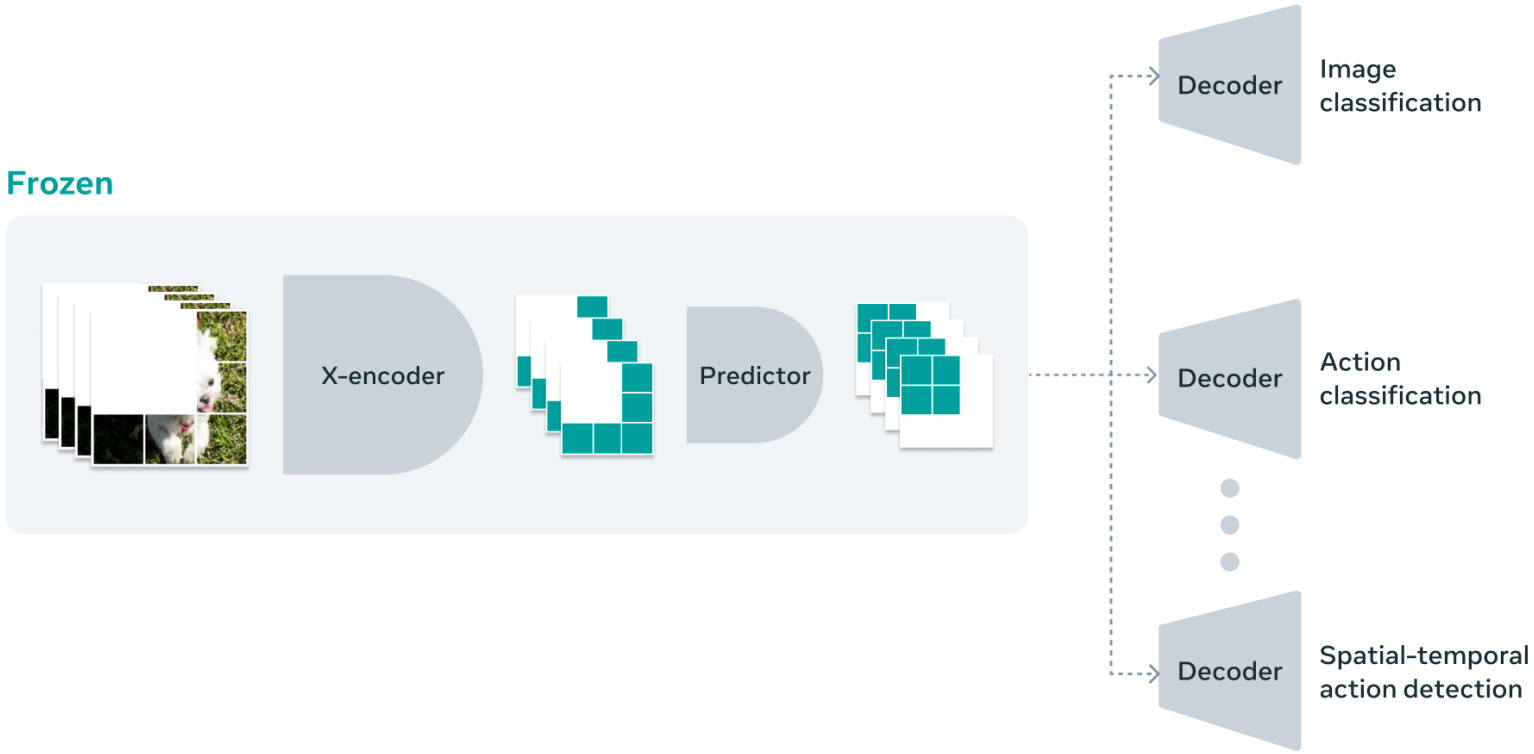
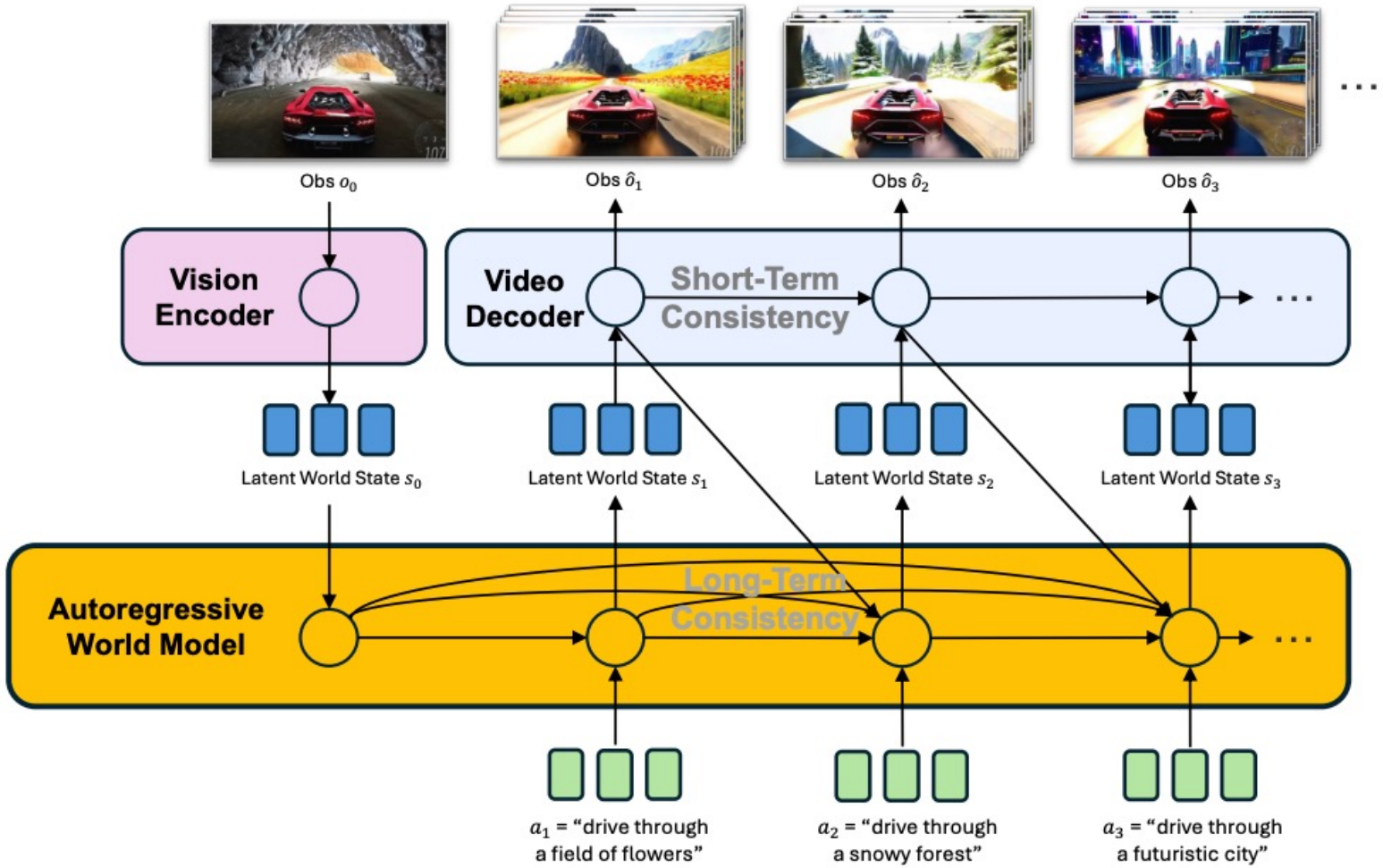


Figure from https://ai.meta.com/blog/v-jepa-yann-lecun-ai-model-video-joint-embedding-predictive-architecture/

PAN

- Goal: interactive world model for long-horizon, conditioned simulation
- Model: three parts
  - *Vision Encoder (h):* maps observations into structured latent states
  - *Autoregressive World Model (f):* predicts next latent state conditioned on actions & history (**long horizon**)
  - *Video Diffusion Decoder (g):* reconstructs high-quality visual observations from latent states (**short horizon**)



$$p_{\text{PAN}}(o_{t+1} \mid o_t, a_t) = \sum_{\hat{s}_t, \hat{s}_{t+1}} \underbrace{p_h(\hat{s}_t \mid o_t)}_{\text{encoder}} \underbrace{p_f(\hat{s}_{t+1} \mid \hat{s}_t, a_t)}_{\text{world model}} \underbrace{p_g(o_{t+1} \mid \hat{s}_{t+1})}_{\text{decoder}}$$

Figure from https://arxiv.org/pdf/2511.09057

PAN

- Training: unlike JEPA, the training objective remains in observation space, not latent space
  - the JEPA objective is prone to collapse and requires careful regularization to avoid this
  - the PAN objective forces the model to be grounded in the real-world observations in the training data
- Recall:
  - h = *Vision Encoder*
  - *f = Autoregressive World Model*
  - *g = Video Diffusion Decoder*

- predicted next observation:

$$\hat{o}_{t+1} = g \circ f(h(o_t), a_t)$$

- ground truth next observation

$$o_{t+1}$$

$$\mathcal{L}_{\text{PAN}} = \mathbb{E}_{(o_t, a_t, o_{t+1}) \sim \mathcal{D}} \left[\text{disc}\left(\hat{o}_{t+1},\, o_{t+1}\right) \right].$$

$$\mathcal{L}_{\text{JEPA}} = \mathbb{E}_{(o_t, a_t, o_{t+1}) \sim \mathcal{D}} \left[\| f(h(o_t), a_t) - h(o_{t+1}) \| \right]$$

Figure from https://arxiv.org/pdf/2511.09057

PAN Results

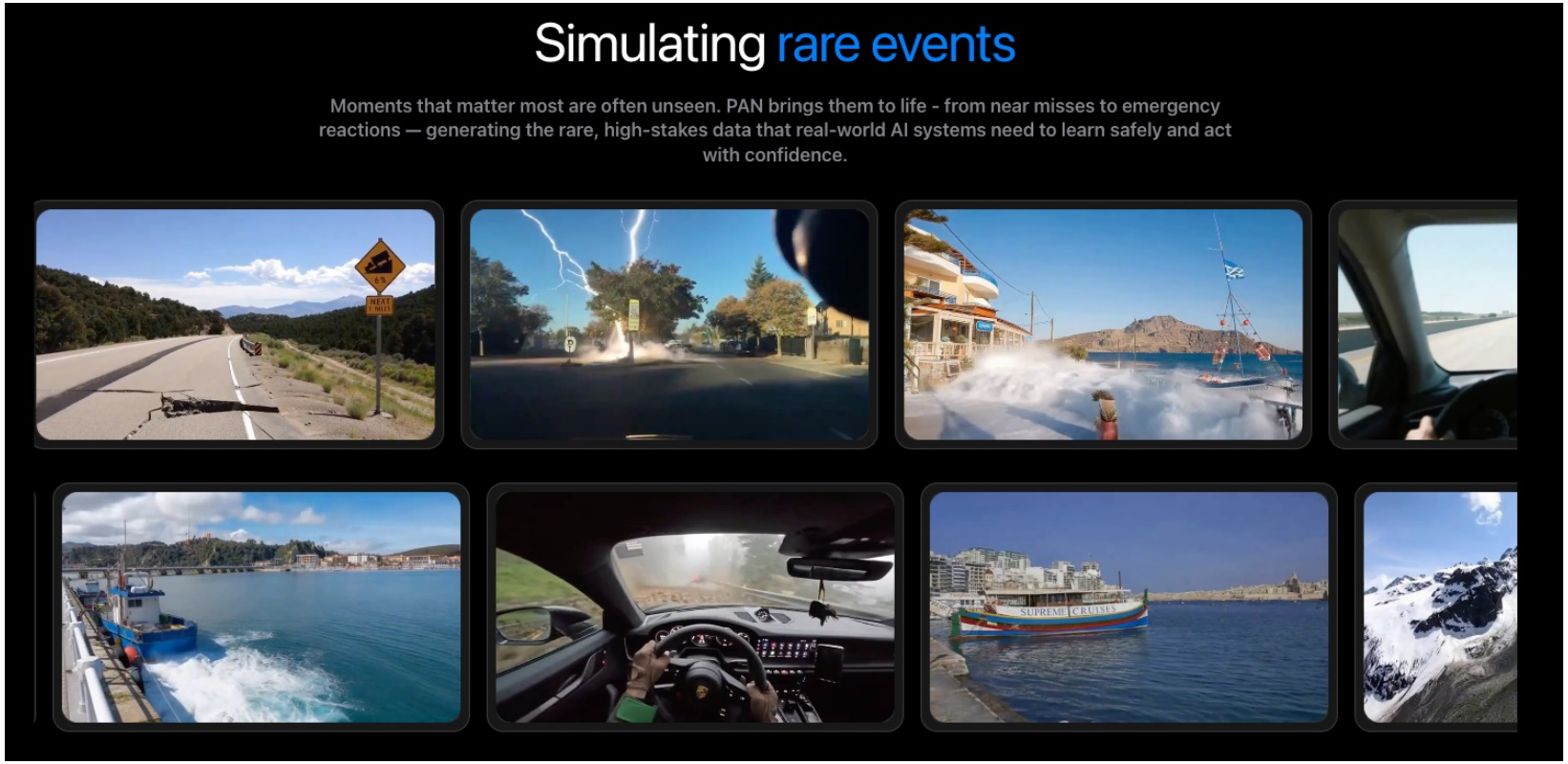


Figure from https://panworld.ai/

PAN Results

Figure from https://panworld.ai/