World Models

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

August 9, 2026

Table of Contents

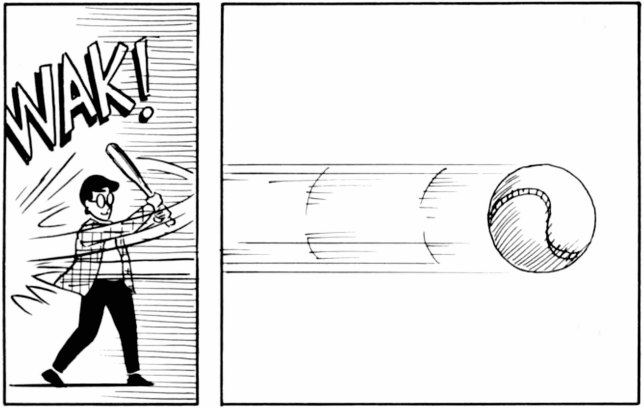
1. Recap and background: world models so far, RL vocabulary, VAEs
2. World models: motivation, definition, applications, and modelling paradigms
3. Ha & Schmidhuber (2018): V–M–C architecture and training inside the dream
4. RSSM / PlaNet: stochastic and deterministic latents, ELBO, planning with CEM
5. Dreamer $\rightarrow$ DreamerV2 $\rightarrow$ DreamerV3: behaviors learned in imagination
6. Generative simulators (Sora, GAIA, Genie) vs. JEPA-style latent prediction
7. LeWorldModel (LeWM): end-to-end JEPA from pixels, SIGReg, and latent planning
8. The wider landscape: MuZero, interactive worlds (Genie), embodied simulators (Cosmos), and spatial intelligence (World Labs)
9. Limitations, summary, hands-on resources, and references

Recap: Where World Models Have Already Appeared

- **Covered earlier:** a world model as a *predictor in representation space*; multistep and hierarchical world models; V-JEPA 2-AC planning with CEM over imagined latent futures.
- **Also covered earlier:** the large pretrained backbones (DINOv2, V-JEPA 2) that such systems build on.
- **Today, the control side of the story:** agents that learn a model of their environment, imagine futures inside it, and use those imagined futures to plan or to train a policy. This is **model-based reinforcement learning**.
- Route: Ha & Schmidhuber (2018) $\rightarrow$ PlaNet / RSSM $\rightarrow$ Dreamer v1–v3 $\rightarrow$ generative simulators vs. JEPA $\rightarrow$ LeWorldModel.

World Models: Motivation

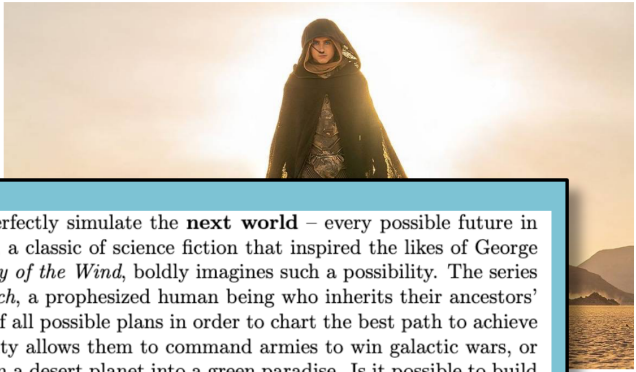
Intuition: a baseball batter reacts in milliseconds; the swing relies on a **predictive**, action-conditioned internal model, not on slow deliberate search.



Scott McCloud excerpt (time–space in comics).

World Models: Motivation (cont. I)

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?

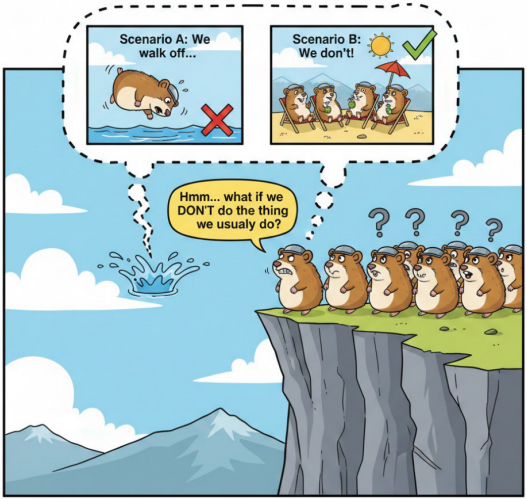
Text from http://arxiv.org/abs/2507.05169

CMU 10-423, Lecture 26.

World Models: Motivation (cont. II)

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

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



CMU 10-423, Lecture 26.

World Models: The Core Idea

- Humans develop a mental model of the world based on what they are able to perceive with their limited senses. Our decisions and actions are based on this internal model.
- Train a generative world model in an unsupervised way to learn a compact spatio-temporal representation of the environment.
- Use features from the world model to train a simple policy (agent) to solve the required task.

World Models: Definition

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):

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

What Does a World State Hold?

Question: What does a world state hold?

Answer: It depends on what you're trying to model. Consider the contrast of:

A) All geopolitical entities, their leaders, and decisions they make

B) Physical interactions in a 3D world

World Models: Prediction, Imagination, and Latents

- A **world model** is a learned map that summarizes experience so we can predict (or sample) **what happens next** and sometimes **counterfactuals**: "if I had pushed left . . . "
- An **agent** chooses actions a_t and receives observations o_t and rewards r_t. Planning and learning are easier if we can **imagine** plausible futures without acting in the real environment.
- Core tension: predict **everything in pixels** (high-dimensional, noisy) vs. predict in a **compact latent space** (lose detail but gain stability and generalization).

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 (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

Some paradigms of World Models

Generate a 3D Scene, then Render/Simulate

Interactive Videos

Latent World Representations

CMU 10-423, Lecture 26.

Paradigms: Generated 3D Worlds (World Labs)

IMAGE CREDITS: WORLD LABS

Fei-Fei Li's World Labs speeds up the world model race with Marble, its first commercial product

Rebecca Bellan 5:44 AM PST · November 12, 2025

World Labs, the startup founded by AI pioneer Fei-Fei Li, is launching its first commercial world model product. Marble is now available via freemium and paid tiers that let users turn text prompts, photos, videos, 3D layouts, or panoramas into editable, downloadable 3D environments.

StrictlyVC vancouver
May 27 | Athens, Greece

World Labs.

Minimal Reinforcement Learning Vocabulary

This is a computer vision course, so here is the RL vocabulary used today:

- **Agent–environment loop:** at each step the agent receives an observation o_t, picks an action a_t, and gets a reward r_t from the environment.
- **Return:** cumulative (discounted) reward over time; the quantity the agent maximizes.
- **Policy** $\pi(a \mid s)$: the agent's action rule. **Value function** $V(s)$: expected return from state s; a network estimating it is called a **critic**, and the policy network an **actor**.
- **Model-free RL:** learn the policy and value directly from real interactions; no model of the environment.
- **Model-based RL:** first learn a model of the environment (the **world model**), then plan or train the policy inside it. Typically needs far fewer real interactions.
- **Replay buffer:** a store of past real interactions (observations, actions, rewards) that training keeps drawing from.

Recap: Variational Autoencoders (VAE)

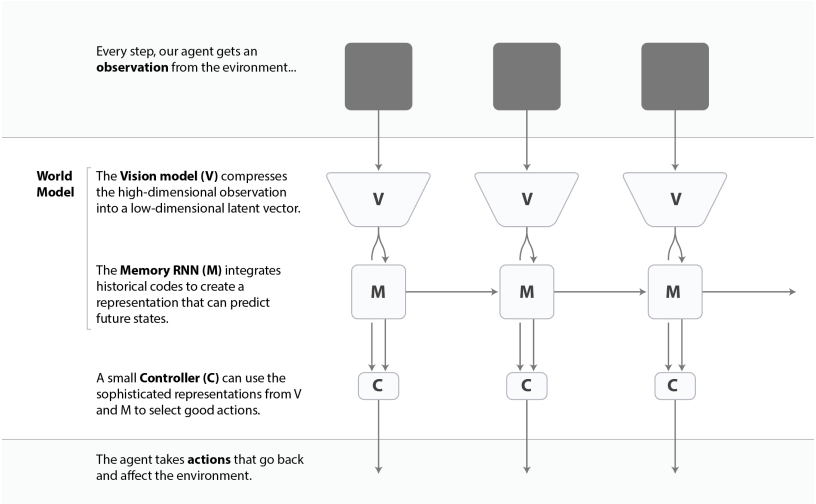
As covered earlier in the course, in brief:

- An **autoencoder** compresses an input x to a latent code z (encoder) and reconstructs x from z (decoder).
- A **VAE** (Kingma & Welling, 2014) makes the latent *probabilistic*: the encoder outputs a distribution $q(z \mid x) = \mathcal{N}(\mu, \sigma^2)$ and z is **sampled** from it before decoding.
- Training maximizes the **ELBO** (evidence lower bound):

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(z|x)}[\log p(x \mid z)]}_{\text{reconstruction}} - \underbrace{\mathrm{KL}(q(z \mid x) \,\|\, \mathcal{N}(0, I))}_{\text{keep latents near a prior}}$$

- The sampling step stays differentiable via the **reparameterization trick**: $z = \mu + \sigma \cdot \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I)$.
- Why it matters today: Ha & Schmidhuber's vision module *is* this VAE, and RSSM training uses the same ELBO idea unrolled through time.

World Model Overview

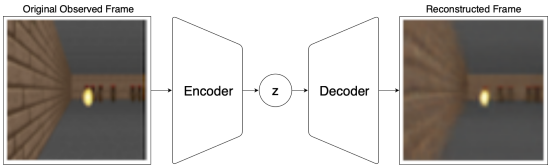


World model overview.

Source: Ha & Schmidhuber, 2018

Vision module: variational autoencoder (V)

- **Goal:** learn an abstract, compressed representation of each observed input frame.
- **VAE:** first stage of a two-stage sequence model; similar in spirit to modern tokenizers, but trained with the classic VAE objective (see the recap slide).

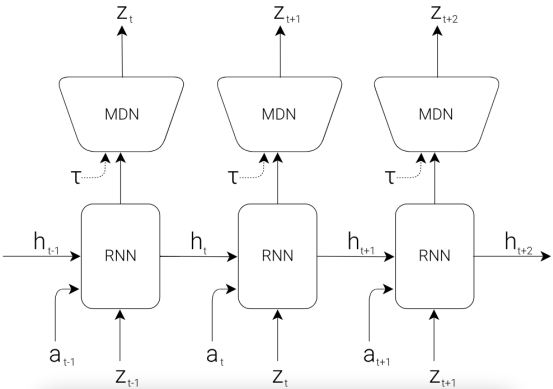


VAE diagram.

Source: Ha & Schmidhuber, 2018

Memory module: MDN-RNN (M)

- Model $p(z_{t+1} \mid z_t, a_t, h_t)$ where z_t is the latent state at time t, a_t is the action at time t, and h_t is the hidden state at time t.
- **MDN-RNN:** an RNN (an LSTM, covered earlier) whose output parameterizes a **mixture of Gaussians** over the next latent z_{t+1}; a mixture can represent several plausible futures at once.
- A temperature parameter τ scales the sampling uncertainty; it becomes important when training inside the dream.



MDN-RNN architecture.

Source: Ha & Schmidhuber, 2018

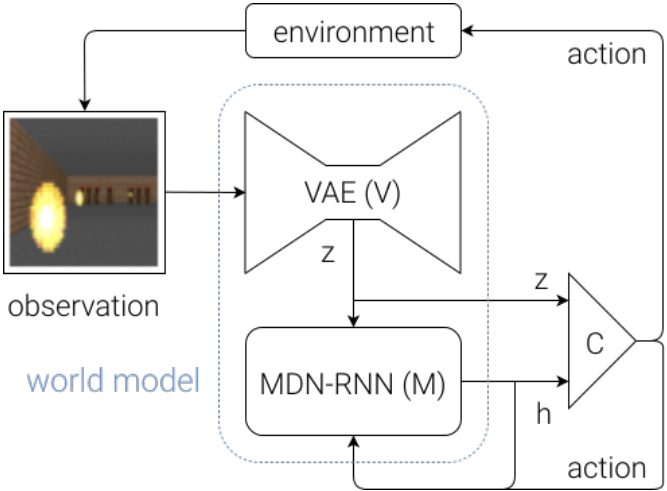
Controller module: decision-making (C)

- The **Controller (C)** selects actions to maximize expected cumulative reward during agent rollout.
- Designed to be as simple and small as possible; trained **separately** from the Vision (V) and Memory (M) modules.
- Most agent complexity is thus concentrated in the world model (V and M).
- C is a **single-layer linear** model that maps the latent vector z_t and RNN hidden state h_t directly to the action a_t at each time step:

$$a_t = W_c[z_t; h_t] + b_c$$

- W_c and b_c are the weight matrix and bias; $[z_t; h_t]$ is the concatenated latent and memory state, and a_t the resulting action.
- Because C has only a few hundred parameters, it is trained with **CMA-ES**, an evolution strategy that searches weight space directly; no gradients through the agent loop are needed.

World model: Putting Everything Together



Agent-environment flow (observation $\rightarrow z_t$, RNN state h_t, action a_t, next step).

Source: Ha & Schmidhuber, 2018

Training Inside the Dream

- The memory model M can generate its own environment: replace real observations with imagined latents and train the controller **entirely inside the hallucinated rollout**.
- In the VizDoom experiment, a policy trained purely in the dream **transfers back to the real game** (as reported in the paper).
- The failure mode: the controller learns to **exploit model errors**, steering into states the model predicts wrongly (for example, imagined worlds where enemies never fire).
- Countermeasure: raise the temperature τ so the dream is noisier and harder to exploit; the resulting policy is more robust in the real environment.
- This exploit-the-model problem returns in Dreamer, mitigated there by short imagination horizons and continual retraining on fresh real experience.

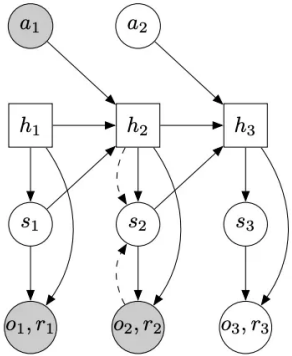
Takeaways

- **Separation of concerns:** perception (VAE), dynamics (RNN), decision (controller). Easy to prototype, but the modules do **not** share gradients end-to-end through the full agent loop at scale.
- **Stochastic latents** matter: the world is multi-modal; Gaussian or MDN heads encode "many possible futures".
- Next step (PlaNet, Dreamer): unify training with **RSSM**-style latent dynamics and **model-based RL** losses.

RSSM: Motivation

- Purely deterministic RNNs (e.g., GRUs, LSTMs) struggle to represent multiple plausible futures in complex, stochastic environments.
- **Recurrent State Space Models (RSSM)** address this by combining:
  - **Deterministic hidden state** h_t (e.g., GRU): maintains long-term memory and context.
  - **Stochastic state** z_t (e.g., Gaussian): captures uncertainty and enables diverse possible futures.
- This split allows RSSM to unify stable memory (for causality) and flexible modeling (for multi-modal, uncertain environments).
- Without the stochastic component, agents often **fail to learn** in partially observable or noisy tasks.
- Result: a flexible temporal representation for RL agents capable of handling uncertainty and representing many possible outcomes.

RSSM: Architecture



Recurrent State Space Model (RSSM). Circles are stochastic variables (filled when observed) and squares are deterministic variables; solid lines denote the generative process, dashed lines the inference model. A mixed deterministic–stochastic model.

Source: PlaNet, arXiv:1811.04551.

RSSM: Dynamics

- **Deterministic recurrence:** the GRU carries memory and the **action** forward,

$$h_t = f(h_{t-1}, z_{t-1}, a_{t-1})$$

- **Posterior transition (training):** Updates stochastic state using real observation

$$z_t \sim q_\phi(z_t \mid h_t, x_t)$$

- **Prior transition (imagination):** Predicts next latent state without observation,

$$z_t \sim p_\theta(z_t \mid h_t)$$

- **Training objective:** the model learns by reconstructing images, predicting rewards, and keeping its imagined predictions close to what really happened, using an ELBO-style loss (next slide).
- This ensures that planning in latent space is meaningful: predictions made during "imagination" are grounded in real transitions.

RSSM: Training Objective

The same ELBO as the VAE, unrolled through time, with a reward head added:

$$\max \; \mathbb{E}_q \left[\sum_t \Big(\underbrace{\ln p(x_t \mid h_t, z_t)}_{\text{reconstruction}} + \underbrace{\ln p(r_t \mid h_t, z_t)}_{\text{reward prediction}} - \beta \underbrace{\mathrm{KL}\big(q(z_t \mid h_t, x_t) \,\|\, p(z_t \mid h_t)\big)}_{\text{imagination matches reality}} \Big) \right]$$

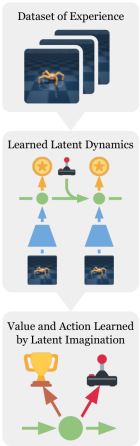
- **Reconstruction** grounds the latent state in observations.
- The **reward head** makes latent rollouts useful for planning and RL.
- The **KL term** keeps the prior (used during imagination) and the posterior (informed by real observations) close together, so imagined futures stay realistic; how to weight its two directions is revisited by DreamerV2's KL balancing.

PlaNet: Planning in Latent Space

- **PlaNet** (Hafner et al., 2019) pairs the RSSM with **online planning**; there is no policy network at all.
- At each step: sample candidate action sequences, roll them out in **latent space**, score them with the predicted rewards, and refine with the **cross-entropy method (CEM)**; execute the first action and replan.
- This is the same CEM / model-predictive control (**MPC**) loop that V-JEPA 2-AC uses (covered earlier); PlaNet did it with an ELBO-trained generative model and a reward head instead of goal images.
- Learns image-based control tasks with far fewer environment interactions than model-free baselines.
- Two costs: planning at every step is expensive, and CEM has no value estimate beyond its horizon. **Dreamer** fixes both.

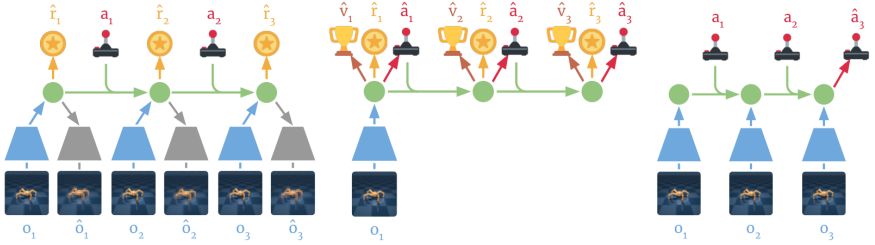
Dreamer: motivation

Model-based RL agent with an RSSM core that reaches state-of-the-art performance on image-based control tasks while requiring fewer samples.



Hafner et al., *Dream to Control*. Chen, Towards AI (2020).

Dreamer: architecture



Dreamer architecture. Diagram credit: Hafner et al. (ICLR 2020), PDF

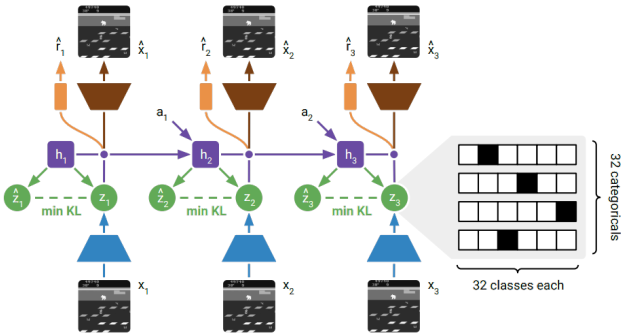
Dreamer is composed of three main parts:

- **(a) Dynamics Learning:** learns representation, transition, reconstruction, reward models.
- **(b) Behavior Learning:** based on the dynamics model, learns actor–critic architecture to maximize rewards.
- **(c) Environment Interaction:** using action model for data collection.

Dreamer: Learning Behaviors in Imagination

- Instead of planning at act time (PlaNet), Dreamer trains an **actor** and a **critic** on **imagined latent trajectories**.
- Rollouts never touch pixels: start from latent states of real experience drawn from the replay buffer, then unroll the RSSM **prior** for a short horizon.
- The **critic** estimates value beyond the horizon; the **actor** is improved by backpropagating value gradients **through the learned dynamics**, which is possible because the world model is differentiable.
- Result: acting is a single forward pass (no per-step search), and the critic gives long-horizon credit assignment that CEM planning lacked.

DreamerV2: discrete world model & V1 → V2



World-model overview. Diagram credit: Belotti (2023)

- **Key upgrade:** The world model now uses **discrete categorical latents** instead of continuous Gaussian variables. Each latent consists of 32 variables, each taking one of 32 categories, enabling a richer and more expressive latent space.
- Discrete latents are optimized using **straight-through gradients** for effective backpropagation despite non-differentiability. This allows for more precise representation and imagination.
- **KL balancing** weights the prior and posterior gradient paths of the KL term differently (the prior is trained faster), preventing the posterior from collapsing onto an untrained prior.

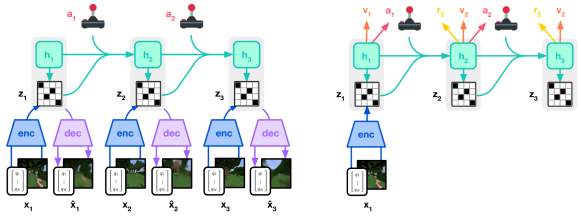
DreamerV3: motivation

- Achieves state-of-the-art performance across 150+ diverse tasks with a single configuration.
- Learns a world model to imagine future trajectories. Successfully solves challenging open-world environments such as collecting diamonds in Minecraft without human data.
- Architecturally similar to DreamerV2. Robustness and scalability stem from careful normalization, balancing, and mathematical tricks that enable stability across domains.

DreamerV3: architecture

Learning algorithm consists of three neural networks:

- **World model:** RSSM model that predicts the outcomes of potential actions, modeling future states of the environment.
- **Critic:** Judges the value of each predicted outcome, estimating how beneficial a state is for achieving the agent's goals.
- **Actor:** Selects actions intended to achieve the most valuable outcomes by maximizing the evaluated value.



(a) World Model Learning **(b)** Actor Critic Learning

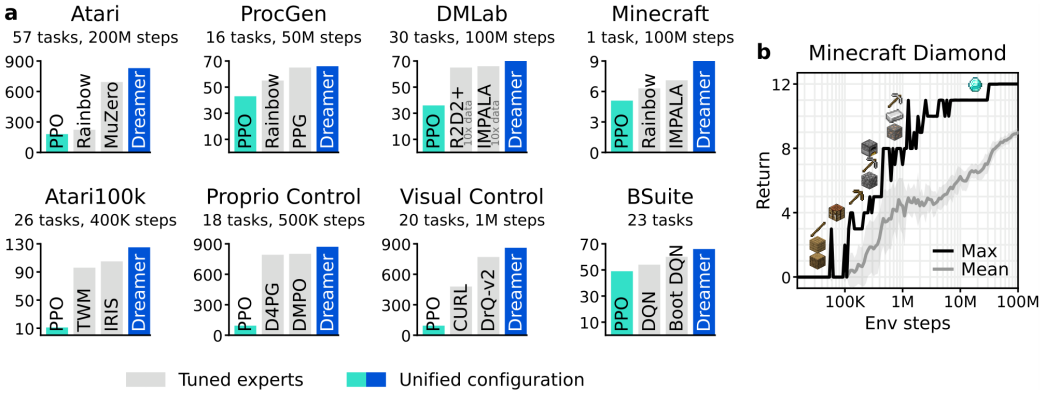
Hafner et al., arXiv:2301.04104.

DreamerV3: What Makes It Stable

The "mathematical tricks" behind one configuration for 150+ tasks:

- **Symlog predictions:** inputs and regression targets pass through a signed-log transform, taming value scales that differ by orders of magnitude across domains.
- **Two-hot return targets:** the critic predicts a distribution over discretized returns instead of a single scalar, which is more robust to outliers.
- **KL balancing and free bits:** the world-model KL term is split and clipped so neither prior nor posterior collapses, and small KL values are not over-optimized.
- **Percentile return normalization:** returns are divided by an EMA of their 5th–95th percentile range, so exploration behaves consistently across dense and sparse rewards.
- Published in *Nature* (2025); the headline demo is collecting diamonds in Minecraft from scratch, without human data.

DreamerV3: results



Benchmark summary: with fixed hyperparameters across all domains, Dreamer outperforms tuned expert algorithms across a wide range of benchmarks and data budgets, and learns to obtain diamonds in Minecraft from scratch given sparse rewards.

Hafner et al., arXiv:2301.04104.

Generative Simulators: Video Generation as a World Model

- Large video generators are increasingly pitched as world simulators; most build on diffusion or autoregressive generation (covered earlier in the course).
- **Sora** (OpenAI, 2024) frames video generation explicitly as "world simulation"; **GAIA-1** and its successors (Wayve) generate driving scenarios for autonomous-vehicle development.
- **Genie** (Google DeepMind) makes **playable** worlds: the model predicts the next frames *given user actions*, which is exactly the world-model interface, realized in pixel space. Genie 3 (2025) is reported to run interactive worlds in real time at 720p.
- The debate: photorealistic video is not the same as physically consistent prediction. Failures in object permanence and physics are common, and evaluating a simulator's "correctness" is an open problem.

Generative world models

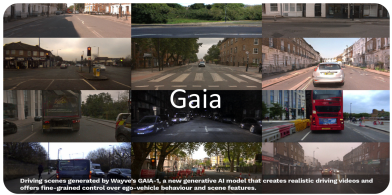


Image credit: CS25.

Architectures: Generative vs. Joint Embedding

- **Generative: predicts y** (with all the details, including irrelevant ones)
- **Joint Embedding: predicts an abstract representation of y**

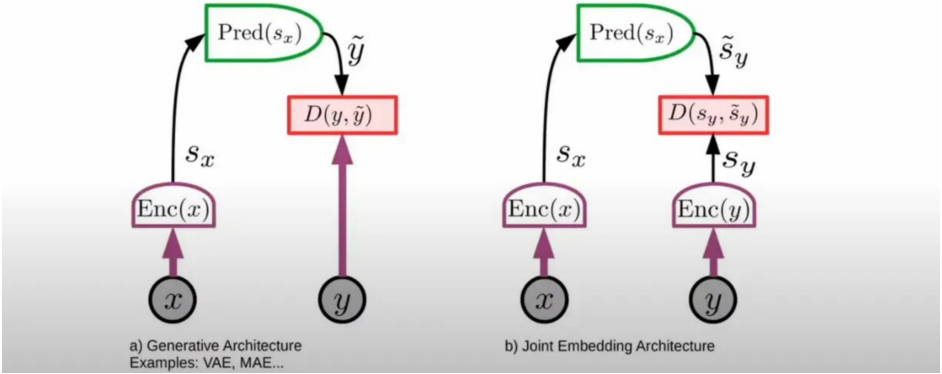
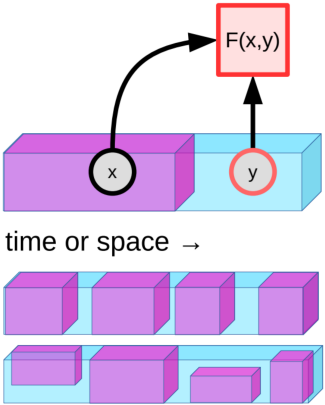


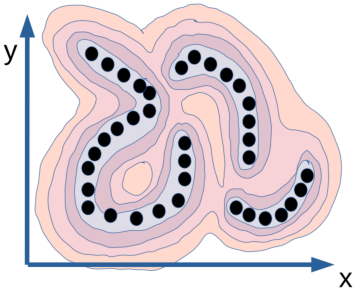
Image credit: CS25.

JEPA & energy-based view



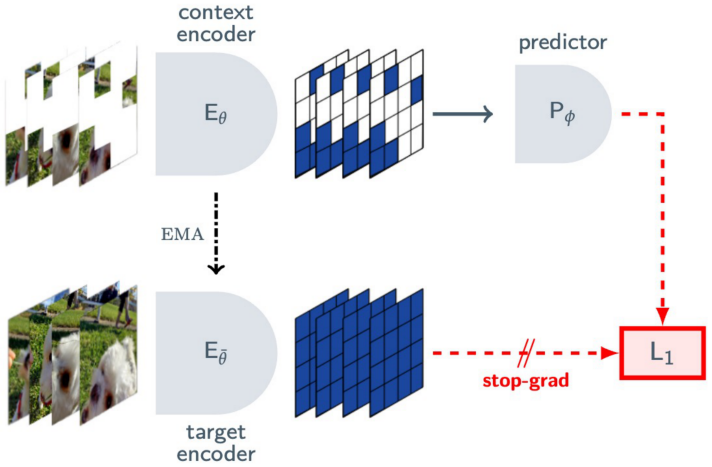
Energy Function

Energy Based Model



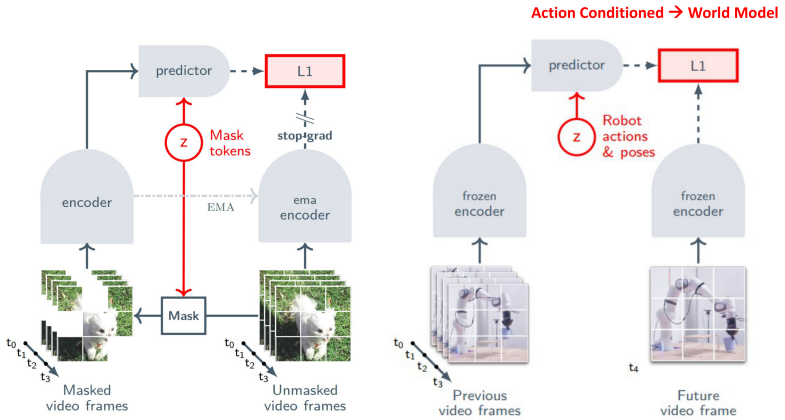
Paper: OpenReview, Y. LeCun (2022).

V-JEPA: video JEPA



Paper: Bardes et al., arXiv:2404.08471 (V-JEPA).

V-JEPA 2: action-conditioned latent prediction



Paper: Assran et al., arXiv:2506.09985 (V-JEPA 2).

Generative world models vs. JEPA-style prediction

Generative (reconstruction) path

- Learn latents that **decode** back to pixels (VAE/RSSM).
- Strength: full **observation model** for planning, denoising, and imagined rollouts with explicit images.
- Cost: optimization can emphasize pixels over semantics.

Joint-embedding predictive (JEPA) path

- Predict **embeddings** of future or masked regions in **representation space**.
- Strength: can learn useful **video/robotics** features without needing to generate every pixel.
- Cost: no pixel rollouts to inspect, and control requires adding an action pathway (V-JEPA 2-AC, LeWM).

LeWorldModel (LeWM): Motivation

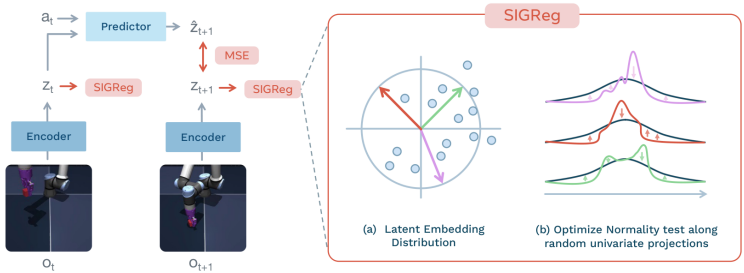


Image credit: Maes et al., LeWorldModel (arXiv:2603.19312).

- **LeWorldModel (LeWM):** a JEPA-style world model trained **end-to-end from raw pixels**, presented as the first to do so with only two loss terms.
- The two terms: a next-latent **prediction loss** and **SIGReg**, a regularizer that pushes latents toward a Gaussian distribution.
- This replaces the EMA / stop-gradient machinery of earlier JEPA recipes, with a stability and anti-collapse argument (as claimed in the paper).

LeWorldModel (LeWM): Architecture

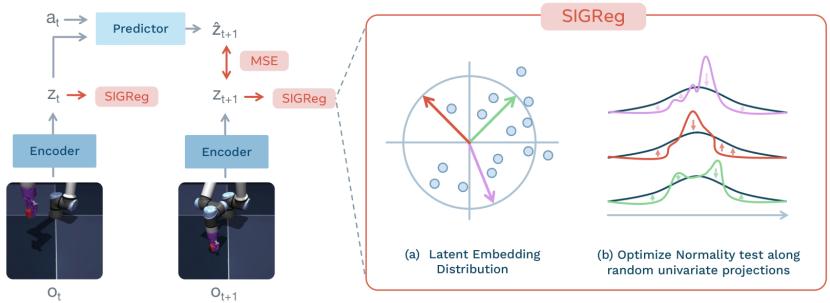
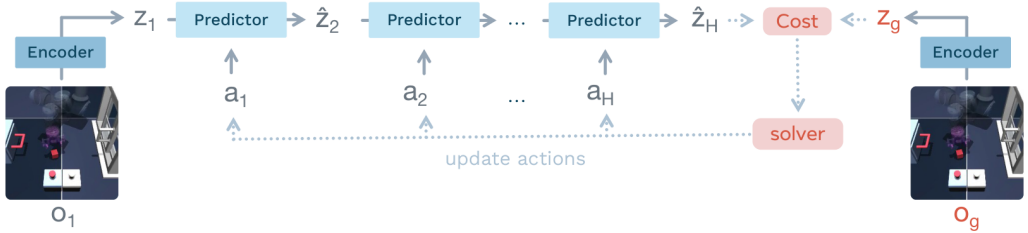


Image credit: Maes et al., LeWorldModel (arXiv:2603.19312).

- **Model Architecture.** LeWM consists of an encoder (mapping frames to latent codes) and a predictor (forecasting the next latent given the current latent and action).
- **Training Objective.** The model is trained end-to-end with a prediction loss plus a regularizer that encourages Gaussian-distributed latents:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda \, \text{SIGReg}(Z)$$

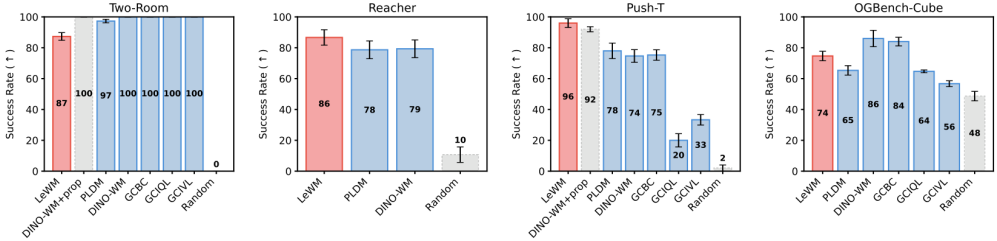
LeWorldModel (LeWM): Latent Planning



LeWorldModel Latent Planning. Given an initial observation o_1 and a goal o_g, the world model learned performs planning in the LeWM latent space. The initial state embedding z_1 and the goal embedding z_g are obtained from the encoder. The predictor then rolls out future latent states up to a horizon H. A latent cost between the final predicted state and the goal embedding guides a solver to optimize the action sequence. This prediction–optimization loop is repeated until convergence to a good plan candidate.

Image credit: Maes et al., LeWorldModel (arXiv:2603.19312), Fig. 4.

LeWorldModel (LeWM): Results



Planning performance across environments (Two-Room, Reacher, PushT, OGBench-Cube). LeWM outperforms PLDM and DINO-WM on PushT and Reacher. DINO-WM leads slightly on the visually complex 3D OGBench-Cube, and both baselines lead on the simple Two-Room task, where the paper suggests SIGReg's Gaussian pressure is a poor fit for a low-dimensional environment.

Image credit: Maes et al., LeWorldModel (arXiv:2603.19312), Fig. 6.

Synthesis: Four Design Points

- **Ha & Schmidhuber (V–M–C):** modular VAE + MDN-RNN + tiny controller; training inside the dream; a generative model with an evolution-trained policy.
- **PlaNet / Dreamer v1–v3:** ELBO-trained RSSM latents; CEM planning (PlaNet), then actor-critic learned in imagination (Dreamer); categorical latents and normalization for stability at scale.
- **V-JEPA 2 (+AC):** non-generative video SSL at scale; an action-conditioned head plus CEM planning brings it to robotics (covered earlier).
- **LeWM:** JEPA from pixels for control, with SIGReg replacing EMA recipes; a small model with fast latent planning, and a good code-first reference for course projects.

Zooming Out: The World-Model Landscape

We have followed two lineages in depth (RSSM/Dreamer and JEPA). The wider field sorts by two questions: **what does the model predict**, and **what is it for**?

- **Latent dynamics for control:** PlaNet, Dreamer v1–v3, TD-MPC2. Predict compact latents plus reward; train policies in imagination (Dreamer) or plan at act time.
- **Value-equivalent planning:** MuZero. Predict only what planning needs (reward, value, policy), nothing about appearance.
- **Latent prediction (JEPA):** V-JEPA 2-AC, DINO-WM, PLDM, LeWM. Predict representations; plan by latent search.
- **Video simulators:** Sora, GAIA-1/2, NVIDIA Cosmos. Generate future video, often for data and evaluation.
- **Interactive generative worlds:** Genie 1–3, GameNGen, Oasis. Generate the next frame *conditioned on user actions*; the latest systems run in real time.
- **Spatial intelligence / 3D worlds:** World Labs (Fei-Fei Li). Generate explicit, persistent 3D scenes rather than video streams.

Roots: Schmidhuber's Controller–Model Systems and Dyna

- **Schmidhuber (1990):** a recurrent **controller** network paired with a recurrent **model** network that learns to predict the consequences of actions; the controller can be trained through the differentiable learned model. Essentially the 2018 V–M–C blueprint, three decades earlier.
- **Artificial curiosity (1991):** reward the agent where the model's *prediction error* is high, so it explores what it does not yet understand; the world model doubles as an exploration signal.
- **Sutton's Dyna (1991):** in parallel, RL work framed planning as *learning from imagined experience* generated by a learned model, the same idea Dreamer scales up.
- Takeaway for the exam: "training a policy inside a learned model" is not a 2018 invention; 2018–2025 made it **work at scale** with deep networks.

MuZero: Planning with a Value-Equivalent Model

- **MuZero** (Schrittwieser et al., Nature 2020) learns a model that predicts only three things: **reward**, **value**, and **policy**. It never reconstructs observations at all.
- The model is **value-equivalent**: it only has to be right about quantities that matter for decisions, not about how the world looks.
- Planning: Monte Carlo tree search unrolled in the learned latent space; reached superhuman play in Go, chess, shogi, and strong Atari results without being given the rules.
- Contrast with today's other families: Dreamer keeps a decoder (reconstruction grounds the latent), JEPA drops the decoder but still predicts *representations*; MuZero is the extreme end where the "world" is reduced to its decision-relevant summary.

Interactive Generative Worlds: GameNGen, Oasis, Genie

- **GameNGen** (Google, 2024): a diffusion model that runs *DOOM* at interactive frame rates; the game engine is replaced entirely by next-frame prediction conditioned on player actions.
- **Oasis** (Decart and Etched, 2024): a playable Minecraft-style world generated frame by frame in real time from keyboard and mouse input.
- **Genie** (Google DeepMind): a foundation world model trained on internet gameplay video; it learns **latent actions** without any action labels, so generated worlds become playable. Genie 2 (2024) produced 3D playable scenes from one image; **Genie 3** (2025) generates interactive worlds from a text prompt in real time at 720p, staying consistent for minutes (as reported by DeepMind).
- These systems answer the world-model question $p(s' \mid s, a)$ directly **in pixel space**, with the user supplying a live.

Embodied and Driving Simulators: GAIA and Cosmos

- **GAIA-1/GAIA-2** (Wayve): generative world models for driving; produce controllable, multi-camera driving scenarios (weather, agents, rare events) used to train and evaluate autonomous-driving stacks.
- **NVIDIA Cosmos** (2025): a *world foundation model* platform for "Physical AI"; pretrained video world models released with open weights, meant to be post-trained into robot- or vehicle-specific simulators for synthetic data generation and policy evaluation.
- The economics: real robot and driving data is expensive and dangerous to collect at the tails; a world model that generates rare, safety-critical scenarios earns its keep even if it is never used for planning directly.
- Same generative-simulator recipe as Sora, pointed at embodiment rather than media.

Spatial Intelligence: Fei-Fei Li and World Labs

- **Thesis (Fei-Fei Li):** language is not enough; **spatial intelligence**, i.e. perceiving, reasoning about, and generating 3D space, is the next frontier for AI. World Labs (founded 2024) builds toward it.
- **RTFM** (2025): a real-time frame model that renders explorable views of a scene interactively on a single GPU (as reported by World Labs).
- **Marble** (2025): generates **persistent, editable 3D worlds** (Gaussian-splat scenes) from text, images, or video; worlds can be exported into standard graphics and game pipelines.
- Key difference from video world models: the world is an **explicit 3D state**, so it does not drift or forget; geometry persists when you look away. The cost is that dynamics and agents must be added on top.
- For robotics and AR, this line argues the world model should *be* a 3D scene, not a video stream.

The Landscape at a Glance

Family	Predicts	Representatives	Primary use
Latent dynamics (RSSM)	latents + reward (pixels optional; TD-MPC2 drops them)	PlaNet, Dreamer v1–v3, TD-MPC2	imagination RL or online planning
Value-equivalent	reward, value, policy only	MuZero	search-based planning
Latent prediction (JEPA)	representations	V-JEPA 2-AC, DINO-WM, PLDM, LeWM	latent MPC, robotics
Video simulators	future video	Sora, GAIA-1/2, Cosmos	synthetic data, evaluation
Interactive worlds	next frame given action	Genie 3, GameNGen, Oasis	playable environments
3D scene generation	persistent 3D scenes	World Labs Marble, RTFM	AR, games, robotics

The families increasingly borrow from each other: interactive worlds add memory, latent planners add dense features, and simulators add action conditioning.

Limitations and Open Problems

- **Compounding errors:** imagined rollouts drift from reality as the horizon grows; short horizons with value bootstrapping (Dreamer) or frequent replanning (MPC) are workarounds, not solutions.
- **Model exploitation:** policies find states where the model is wrong. Ha's temperature trick and retraining on fresh real experience both target versions of this problem.
- **Pixels vs. latents, unresolved:** generative simulators produce impressive video with weak physics guarantees; latent models plan well but have no decoder to inspect, so evaluating what they know is indirect.
- **Goal specification:** reward heads or goal images are the norm; language-conditioned goals for latent world models remain largely open.
- **Benchmarks are narrow:** games, simulated control, and tabletop robotics dominate; long-horizon, open-world transfer is not demonstrated.
- **Compute asymmetry:** video-generator simulators are expensive at inference time; latent planners are cheap but lower fidelity. Where the field settles between them is open.

Summary

- World models compress experience so an agent can roll imagined futures and choose actions; the main design split is predict pixels (generative simulators) vs. predict latents (JEPA-style), with hybrid systems in between.
- Dreamer $\rightarrow$ DreamerV2 $\rightarrow$ DreamerV3: a single RSSM-style world model (representation, dynamics, often decode/reward) plus actor–critic training on imagined trajectories; V2 switches to discrete categorical latents; V3 keeps the recipe and stresses stability across many domains with one configuration.
- JEPA: predict in embedding space, not full observations; V-JEPA scales masked video prediction; V-JEPA 2 adds action-conditioned latent prediction as a bridge to planning and robotics.
- LeWorldModel: End-to-end from pixels with just next-embedding loss + SIGReg. Fast latent MPC planning with a small model.

Hands-On: Code and Exercise

- **Interactive article:** worldmodels.github.io (Ha & Schmidhuber) lets you play with the VAE latents and the dream environment in the browser.
- **DreamerV3:** the first author's reference reimplementation at danijar/dreamerv3; small control tasks train on a single GPU.
- **LeWM:** code at lucas-maes/le-wm; small enough to read end to end, and a good base for course projects.
- **V-JEPA 2:** checkpoints on Hugging Face (e.g. `facebook/vjepa2-vitl-fpc64-256`, as covered earlier) for the latent-planning side.
- **Suggested exercise:** train DreamerV3 on a DeepMind Control task and log imagined vs. real rollouts side by side; then run LeWM's latent planner on PushT and compare how each system fails as the horizon grows.

References

1. Schmidhuber, "Making the World Differentiable," TR FKI-126-90, TU Munich, 1990 (recurrent controller–model systems); "A Possibility for Implementing Curiosity and Boredom in Model-Building Neural Controllers," SAB 1991.
2. Sutton, "Dyna, an Integrated Architecture for Learning, Planning, and Reacting," SIGART Bulletin, 1991.
3. Ha & Schmidhuber, *Recurrent World Models Facilitate Policy Evolution* (NeurIPS 2018), arXiv:1803.10122; interactive write-up worldmodels.github.io.
4. Gormley, M., CMU 10-423 / 10-623, **Lecture 26: World Models**. PDF; F25 mirror PDF (10423-F25).
5. Stanford CS25 deck excerpt (**JEPA / Joint Embedding Predictive World Models** slides raster): JEPA_cs25.pdf (Google Drive); course CS25.
6. Cambridge teaching archive (Ha 2018 PDF). cl.cam.ac.uk PDF.
7. Hafner et al., "Learning Latent Dynamics for Planning from Pixels" (PlaNet), ICML 2019, arXiv:1811.04551.
8. Kingma & Welling, "Auto-Encoding Variational Bayes" (VAE), ICLR 2014, arXiv:1312.6114.
9. Hafner et al., *Dream to Control: Learning Behaviors by Latent Imagination* (ICLR 2020), arXiv:1912.01603 (**Dreamer**). *Slides:* Figs. 1,3,4,8 cropped from PDF.
10. Hafner et al., "Mastering Atari with Discrete World Models" (**DreamerV2**), ICLR 2021, arXiv:2010.02193. Pedagogical diagrams: Belotti, *Dreamer V2* (EclecticSheep blog, 2023).

References (cont.)

11. Hafner et al., “Mastering Diverse Domains through World Models” (DreamerV3), arXiv:2301.04104; published as “Mastering Diverse Control Tasks through World Models,” Nature 2025.
12. Bardes et al., “Revisiting Feature Prediction for Learning Visual Representations from Video” (V-JEPA), arXiv:2404.08471.
13. Assran et al., “V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning”, arXiv:2506.09985.
14. Mur-Labadia et al., “V-JEPA 2.1: Unlocking Dense Features in Video Self-Supervised Learning”, arXiv:2603.14482.
15. Maes et al., “LeWorldModel: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels”, arXiv:2603.19312. PDF `leworldmodel_arxiv2603_19312v1.pdf`. Code: github.com/lucas-maes/le-wm.
16. Bruce et al., “Genie: Generative Interactive Environments”, ICML 2024, arXiv:2402.15391; Google DeepMind, “Genie 3: A New Frontier for World Models”, 2025 (blog).
17. OpenAI, “Video Generation Models as World Simulators” (Sora technical report), 2024.
18. Hu et al. (Wayve), “GAIA-1: A Generative World Model for Autonomous Driving”, 2023, arXiv:2309.17080.
19. Zhou et al., “DINO-WM: World Models on Pre-trained Visual Features Enable Zero-shot Planning”, 2024, arXiv:2411.04983.
20. Sobal et al., “Learning from Reward-Free Offline Data: A Case for Planning with Latent Dynamics Models” (PLDM), 2025, arXiv:2502.14819.

References (cont.)

21. Schrittwieser et al., “Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model” (MuZero), *Nature*, 2020. arXiv:1911.08265.
22. Hansen et al., “TD-MPC2: Scalable, Robust World Models for Continuous Control,” ICLR 2024. arXiv:2310.16828.
23. Valevski et al. (Google), “Diffusion Models Are Real-Time Game Engines” (GameNGen), 2024. arXiv:2408.14837.
24. Decart and Etched, “Oasis: A Universe in a Transformer,” 2024 (blog and playable demo).
25. NVIDIA, “Cosmos World Foundation Model Platform for Physical AI,” 2025. arXiv:2501.03575.
26. Russell et al. (Wayve), “GAIA-2: A Controllable Multi-View Generative World Model for Autonomous Driving,” 2025. arXiv:2503.20523.
27. World Labs (Fei-Fei Li et al.), generative interactive 3D worlds: **RTFM** and **Marble** (2025). `https://www.worldlabs.ai/`; Li, “From Words to Worlds: Spatial Intelligence Is AI’s Next Frontier,” 2025 (essay).