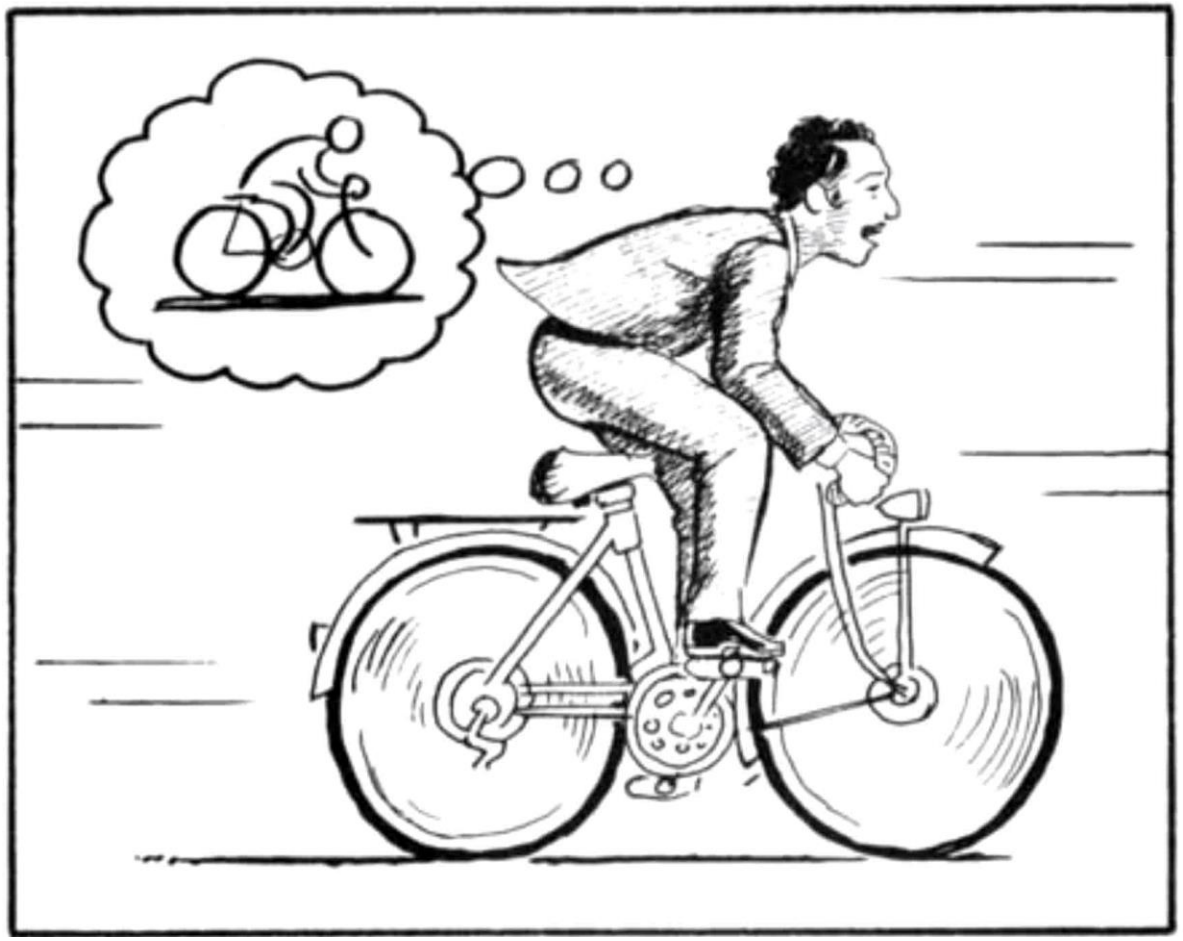
World Models



Instructor: Daniel Brown --- University of Utah

World Models

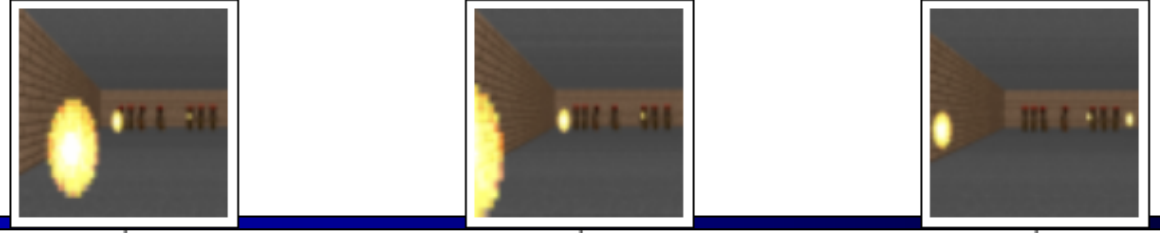
David Ha [1] **Jürgen Schmidhuber** [2 3]

Compare and Contrast with PETS, Dyna-Q

- Train inside model vs. outside model
- Explicit vs. Implicit Policy
- Modeling Uncertainty

Answer: C

At each time step, our agent receives an **observation** from the environment.



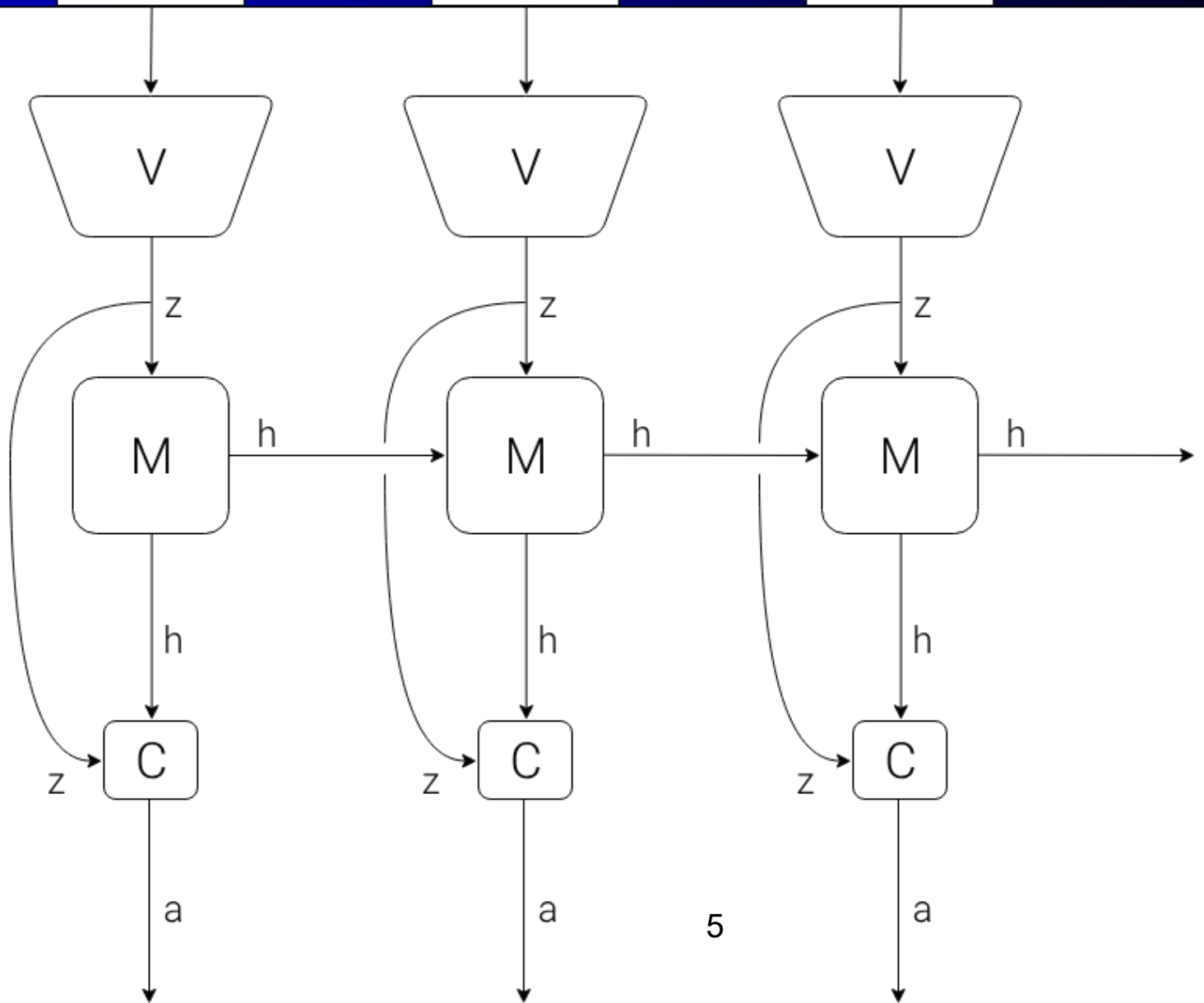
World Model

The **Vision Model (V)** encodes the high-dimensional observation into a low-dimensional latent vector.

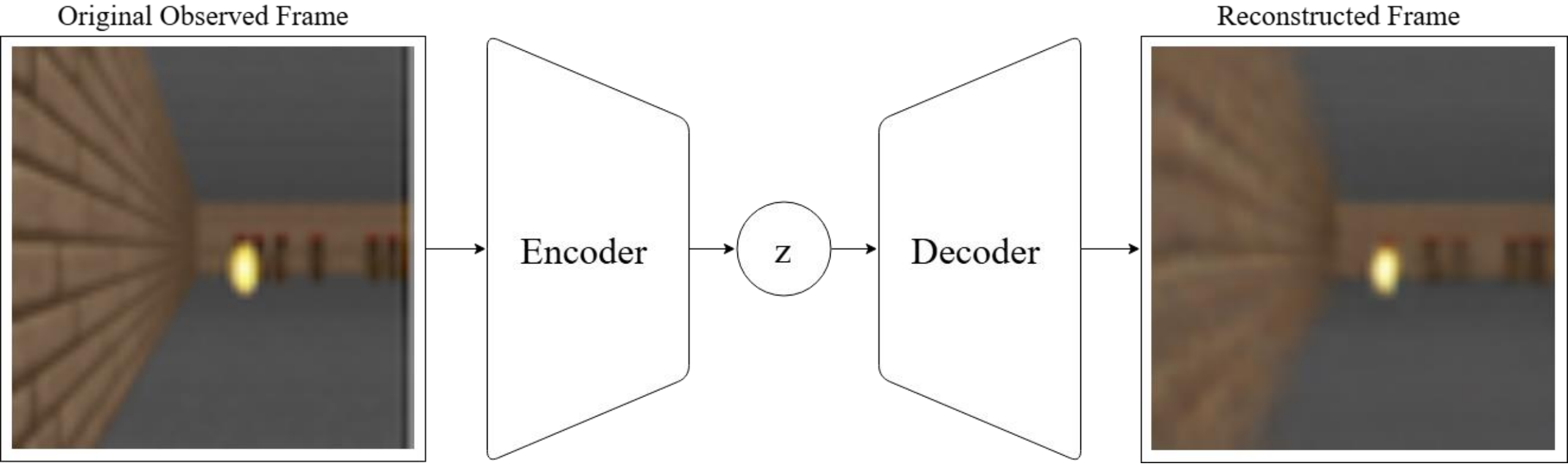
The **Memory RNN (M)** integrates the historical codes to create a representation that can predict future states.

A small **Controller (C)** uses the representations from both **V** and **M** to select good actions.

The agent performs **actions** that go back and affect the environment.



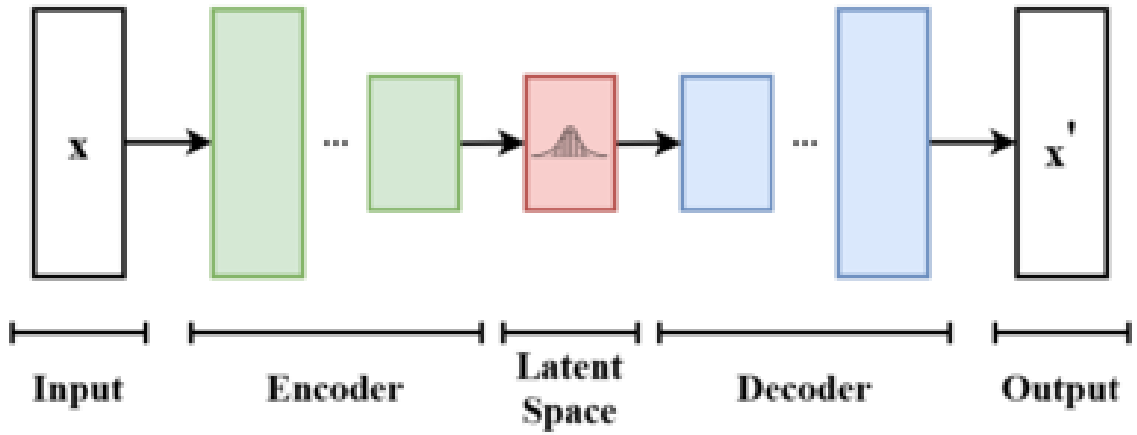
Vision Model



Demo: https://worldmodels.github.io/

Variational Autoencoders (VAEs)

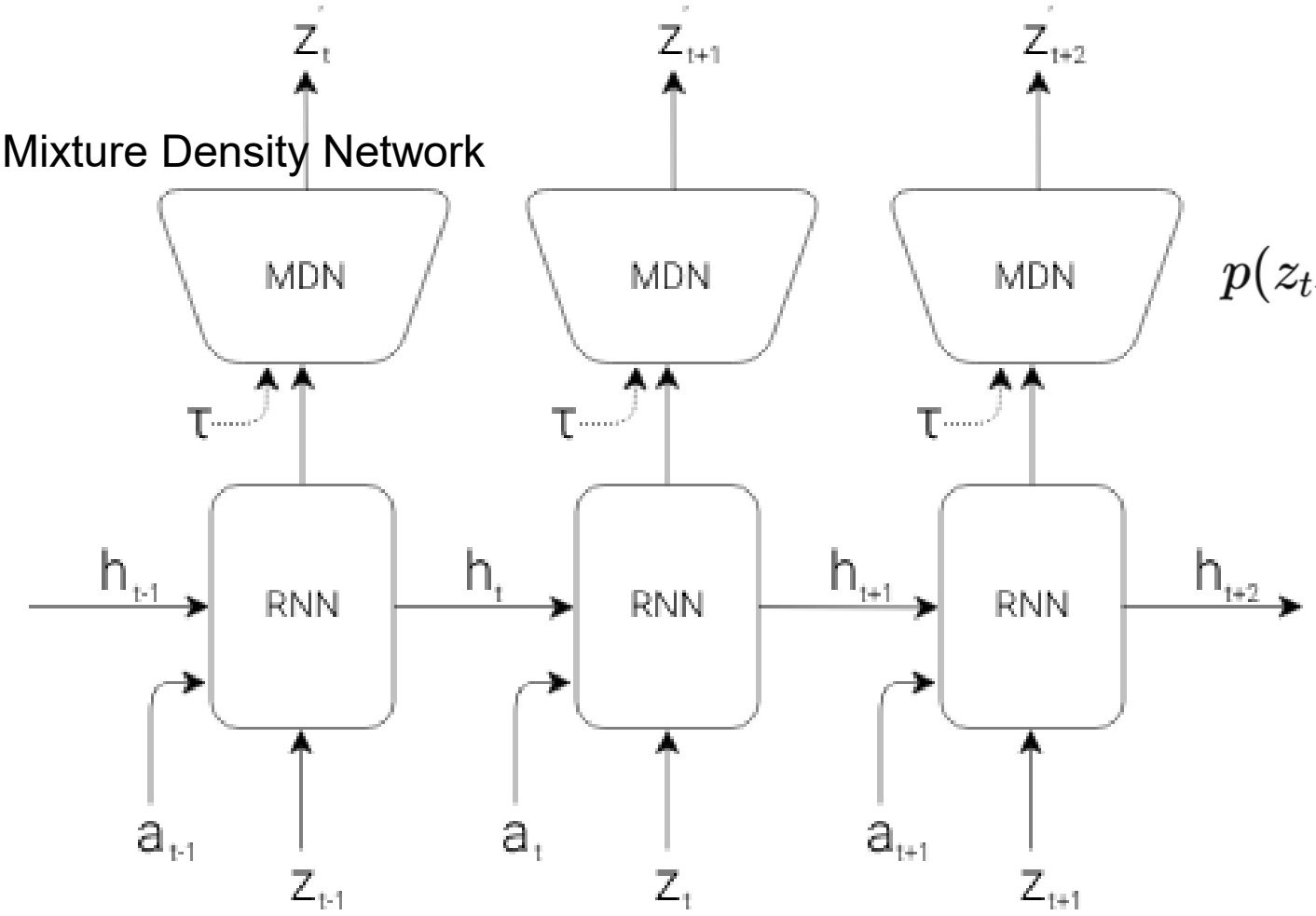
- Autoencoders learn latent representations
- VAEs map input into a distribution over latent variables z
- Loss function is reconstruction plus KL divergence



$$\mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{\mathrm{KL}}(q(z|x)||p(z))$$

Memory Model to Predict the Future

Mixture Density Network



the RNN will model $P(z_{t+1} \mid a_t, z_t, h_t)$

$$p(z_{t+1} \mid x_t) = \sum_{k=1}^{K} \pi_k(x_t)\, \mathcal{N}\big(z_{t+1}; \mu_k(x_t), \Sigma_k(x_t)\big)$$

https://worldmodels.github.io/ SketchRNN

Simple Controller

At each time step, our agent receives an **observation** from the environment.

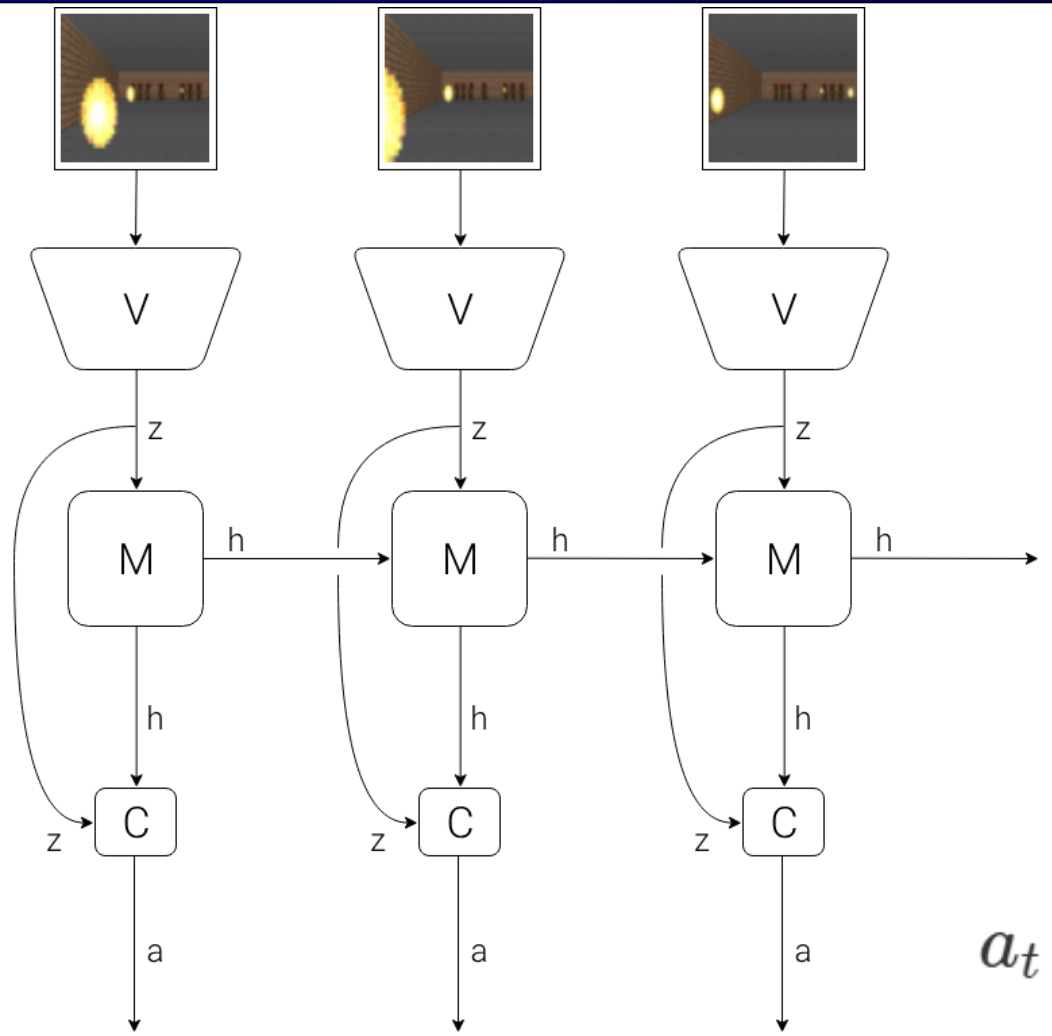
World Model

The **Vision Model (V)** encodes the high-dimensional observation into a low-dimensional latent vector.

The **Memory RNN (M)** integrates the historical codes to create a representation that can predict future states.

A small **Controller (C)** uses the representations from both **V** and **M** to select good actions.

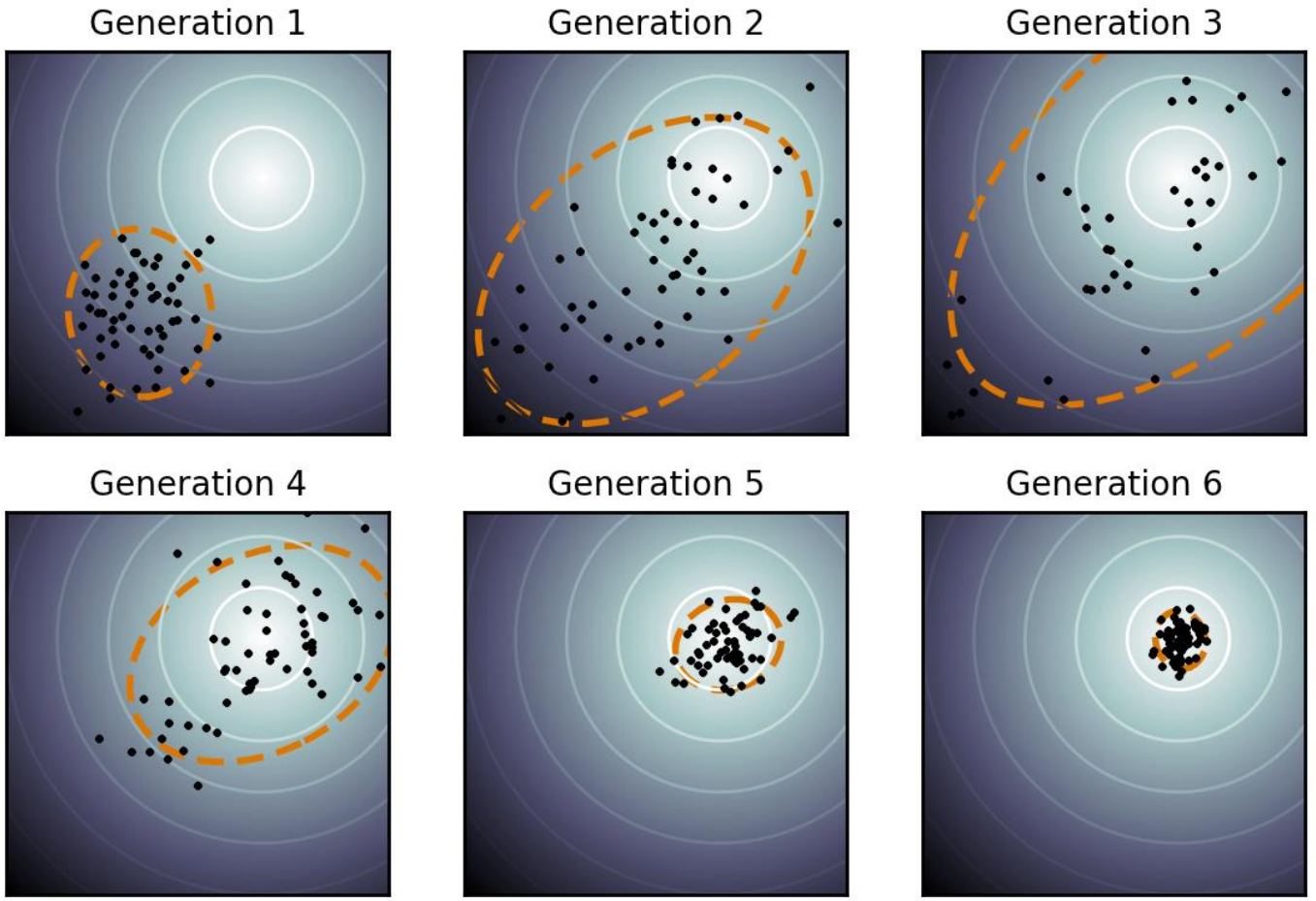
The agent performs **actions** that go back and affect the environment.



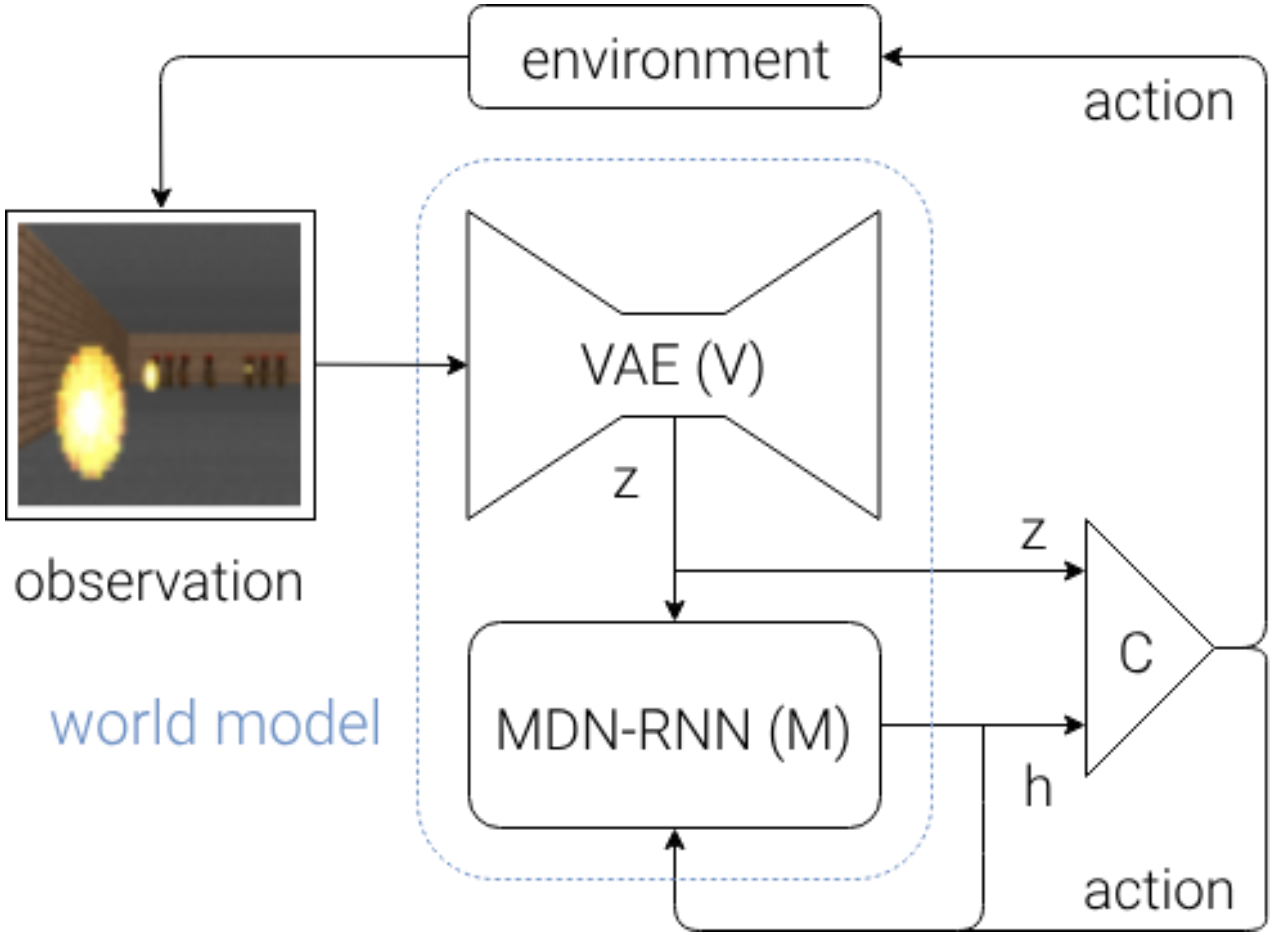
$$a_t = W_c \left[z_t \; h_t \right] + b_c$$

CMA-ES

- Covariance matrix adaptation evolution strategy



Putting everything together



Algorithm

- https://worldmodels.github.io/

Procedure

To summarize the Car Racing experiment, below are the steps taken:

1. Collect 10,000 rollouts from a random policy.
2. Train VAE (V) to encode each frame into a latent vector $z \in \mathcal{R}^{32}$.
3. Train MDN-RNN (M) to model $P(z_{t+1} \mid a_t, z_t, h_t)$.
4. Evolve Controller (C) to maximize the expected cumulative reward of a rollout.

Model	Parameter Count
VAE	4,348,547
MDN-RNN	422,368
Controller	867

METHOD	AVG. SCORE
DQN (PRIEUR, 2017)	343 ± 18
A3C (CONTINUOUS) (JANG ET AL., 2017)	591 ± 45
A3C (DISCRETE) (KHAN & ELIBOL, 2016)	652 ± 10
CEOBILLIONAIRE (GYM LEADERBOARD)	838 ± 11
V MODEL	632 ± 251
V MODEL WITH HIDDEN LAYER	788 ± 141
FULL WORLD MODEL	$\mathbf{906 \pm 21}$

Table 1. `CarRacing-v0` scores achieved using various methods.

VizDoom Experiments

Dreamer V1

Published as a conference paper at ICLR 2020

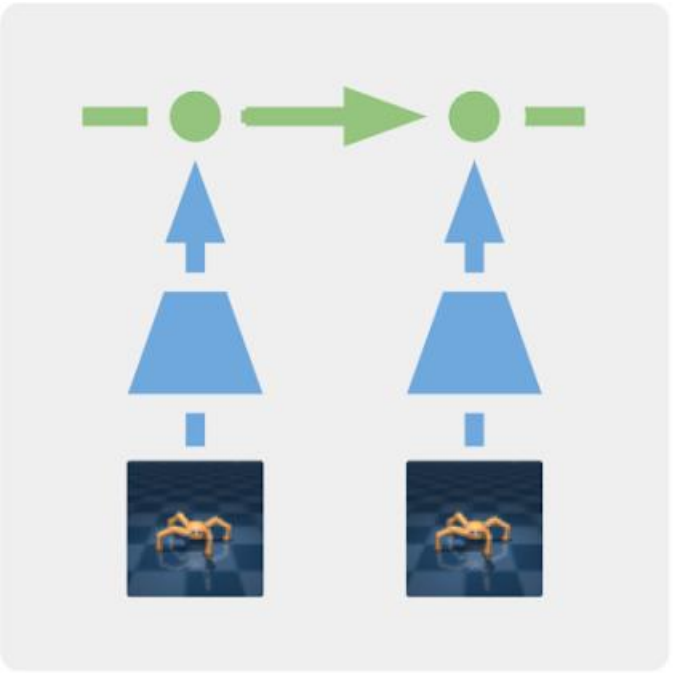
DREAM TO CONTROL: LEARNING BEHAVIORS BY LATENT IMAGINATION

Danijar Hafner *
University of Toronto
Google Brain

Timothy Lillicrap
DeepMind

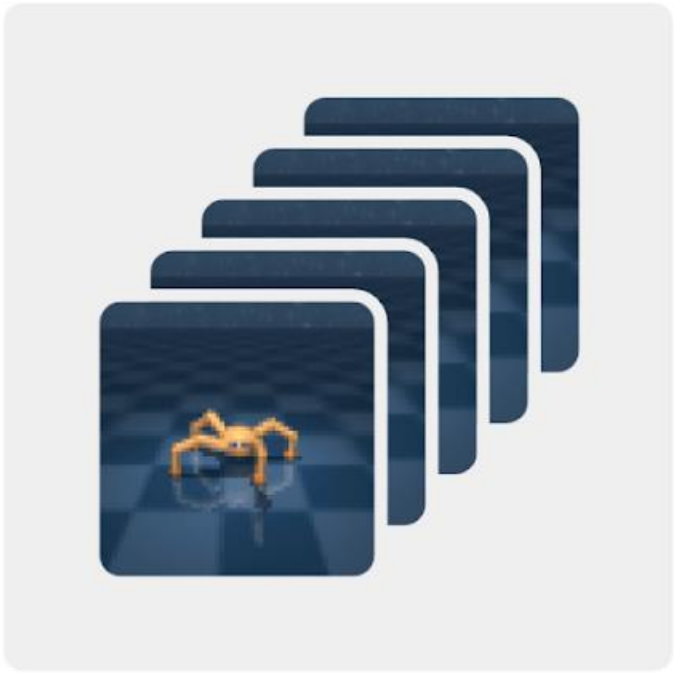
Jimmy Ba
University of Toronto

Mohammad Norouzi
Google Brain



World Model Learning

Learning Value and Actor Networks



Environment Interaction

What makes Dreamer different?

- Actor Critic Architecture
  - Learns value function to optimize Bellman error over imagined rewards
  - Actor gradients are computed through the dynamics

Dreamer V4

Google DeepMind

Training Agents Inside of Scalable World Models

Danijar Hafner* Wilson Yan* Timothy Lillicrap

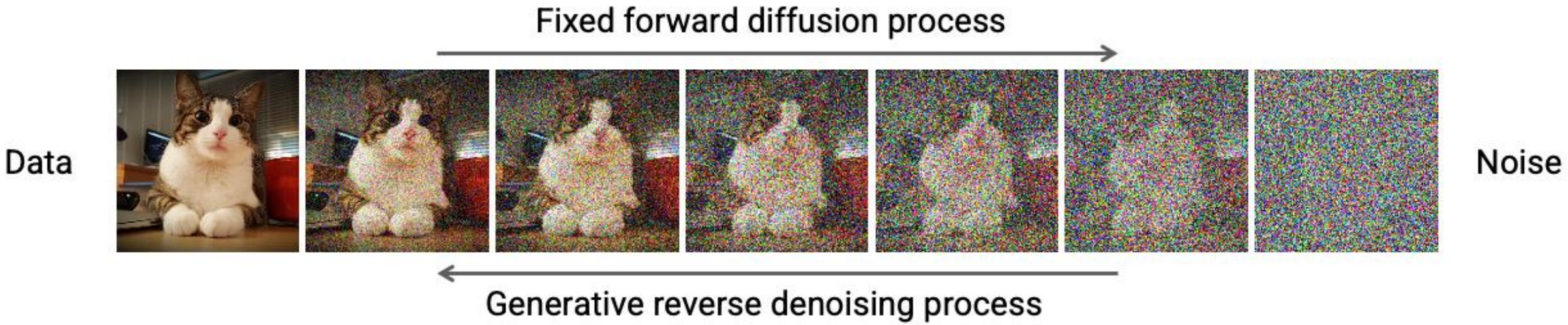
World models learn general knowledge from videos and simulate experience for training behaviors in imagination, offering a path towards intelligent agents. However, previous world models have been unable to accurately predict object interactions in complex environments. We introduce Dreamer 4, a scalable agent that learns to solve control tasks by reinforcement

Compare and Contrast with PETS and World Models

Main Idea

- Learn a powerful generative world model from video and then train an RL policy entirely inside the world model

Denoising Diffusion (high-level)



Flow Matching

- Training

$$x_0 \sim \mathrm{N}(0, \mathbb{I}) \qquad x_1 \sim \mathcal{D}$$

$$x_\tau = (1-\tau)\, x_0 + \tau\, x_1$$

$$\tau \sim p(\tau)$$

$$\mathcal{L}(\theta) = \| f_\theta(x_\tau, \tau) - (x_1 - x_0) \|^2$$

Learn to predict velocity vectors that point towards clean data!

Evaluation

$$x_0 \sim \mathrm{N}(0, \mathbb{I})$$

$$x_{\tau+d} = x_\tau + f_\theta(x_\tau, \tau)\, d$$

Iteratively denoise starting from a completely random sample.

But how many steps do we need?

Additional Stuff

- Shortcut Models
  - Condition on step size to generate good samples in fewer (2-4) steps.
- Masked Autoencoders
- Transformers
- Multi-task Behavior Cloning for Finetuning
- Imagination Training
- Applications to Minecraft and Robotics