DS 598
Introduction to RL

Xuezhou Zhang

Announcements

- Homework 1 is out. Due by the end of next week. Any questions?
- Course project details will be released next week.
- Pytorch tutorial + project environment installation Helpdesk in the discussion section next week.

Recap: Planning

- Value Iteration and Policy Iteration.

- Linear Programming with Occupancy Measures.

- None of these methods work directly for high-dimension/complicated MDPs 🥲 🥲 🥲

- In practice, people often apply some vanilla RL algorithms instead, which means a big room of improvement!

Chapter 3: Model-based RL

Online Reinforcement Learning

- Start by knowing nothing about the environment.
- Gather information while interacting with the environment.
- Gather reward / suffer costs along the way.


Online Reinforcement Learning

- The grand goal of RL/AGI is to **design the learning agent**, not the learned policy.

What is model-based RL (MBRL)?

- Step 1: Learn an estimated MDP using existing data.
- Step 2: Do planning in the learned MDP (simulator).
- Step 3: Try the new policy out in the real world.
- Go to Step 1 after collecting more data.

Why can MBRL work at all?

- Real world $\mathrm{M} = (P, r)$
- Learned model $\widehat{M} = (\hat{P}, \hat{r})$
- Find the optimal policy in $\widehat{M}$: $\hat{\pi} = \mathrm{argmax}_{\pi} V_{\widehat{M}}^{\pi}$.

- How well does $\hat{\pi}$ work in the real world? i.e. how good is $V_{M}^{\hat{\pi}}$?

Simulation Lemma

- Claim: $\max_s V_M^*(s) - V_M^{\hat{\pi}}(s) \leq \frac{\gamma}{1-\gamma} \max_{\mathrm{s,a}} \left|\left|\mathrm{P}(\cdot\,|s,a) - \hat{P}(\cdot\,|s,a)\right|\right|_1 \cdot \left|\left|\mathrm{V}^*\right|\right|_\infty$
- Not gonna bother you with proof this time.
- Check the textbook if you are interested.

🤔 Are Humans model-based learners?

Article | Published: 06 July 2020

Humans primarily use model-based inference in the two-stage task

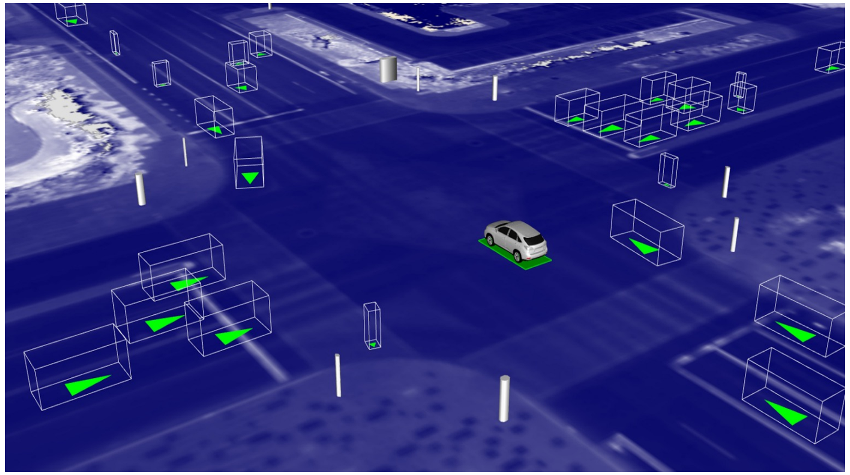
Carolina Feher da Silva & Todd A. Hare

Nature Human Behaviour **4**, 1053–1066 (2020) | Cite this article

Autonomous Driving is model-based.

WAYMO

- Test on the road and collect data



- Build and Train inside a Simulator

Does it work in practice?

- To some extent, yes.
- Heavily used in robotics.
- a.k.a. Model Predictive Control (MPC)

http://heli.stanford.edu 2008 Video

Model Predictive Control (MPC)

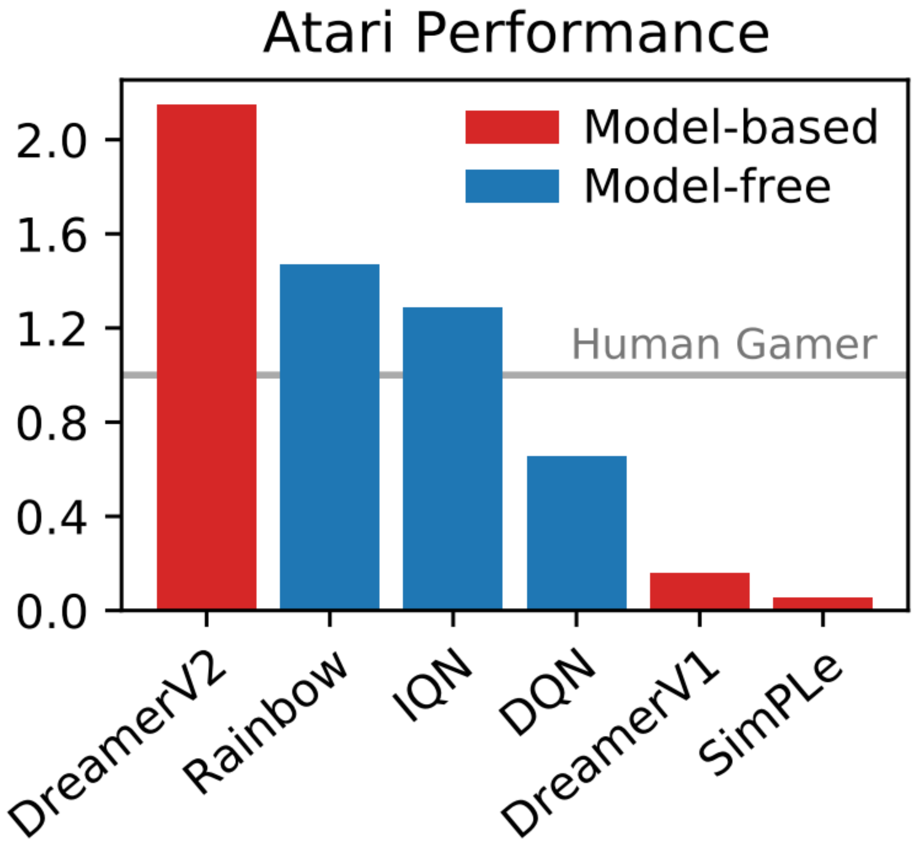
- This is used in “easy problems” like airplane/spaceship/rocket control.
- Linear Quadratic Regulator: $\dot{x} = Ax + Bu$
- e.g. x is (location, speed), u is force.

- Thanks to physics models, only need to learn a few parameters (independent from the number of states & actions, which are infinite).

- Planning: Solve differential equations.

Does it work?

- In other settings, model-based RL have been under-performing until very recently.



DQN: 2013
DreamerV2: 2021

Why does it work in robotics but not Atari Games

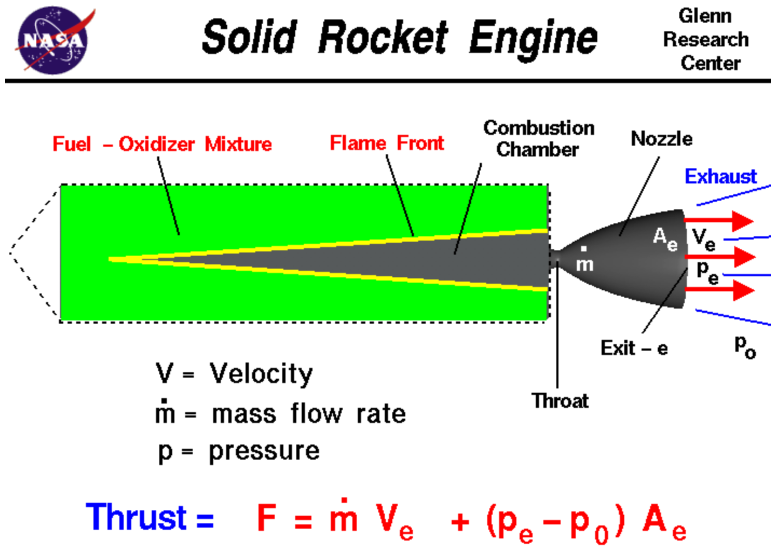
- Compact vs. High-dim representation of the state.

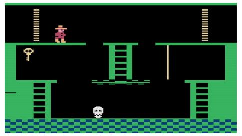
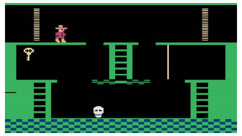
(Position, speed, acceleration) vs.

- $P \in \mathbb{R}^{S^2 A}$.

Why does it work in robotics but not Atari Games

- Strong prior knowledge on the physical model.
- # data needed scales polynomially with # of parameters



vs. P( |  , )

Summary of Challenges with MBRL

1. Transition model with high-dim observation is hard to learn.

2. How to plan in the learned model?

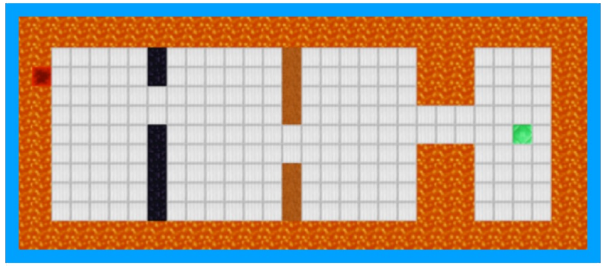
3. How to effectively encode domain knowledge?
   (This is a mega challenge for all of ML)

DreamerV2 [Hafner et al. 2021]

- Idea: learning a latent world model with **discrete states**.

high-dim observation

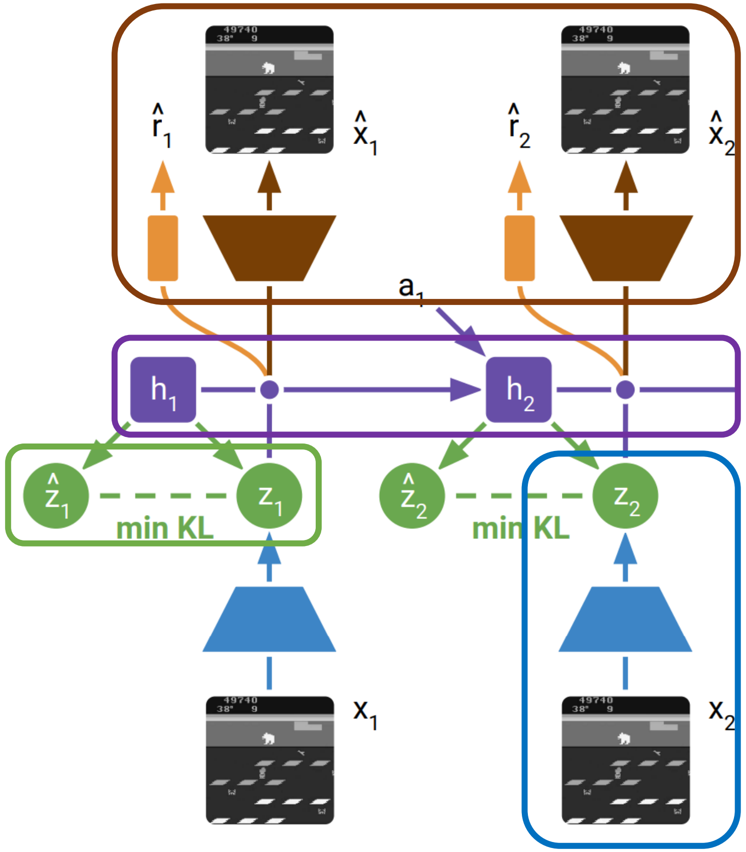
vs.



low-dim latent state

DreamerV2 [Hafner et al. 2021]

- DreamerV2 world model:



Loss Function:

$$\mathcal{L}(\phi) = \sum_{t=1}^{T} -\log o_\phi(x_t | h_t, z_t) + \beta KL\left(q_\phi(\cdot \mid h_t, x_t), p_\phi(\cdot \mid h_t, a_t)\right)$$

Reconstructor (RNN): $\hat{x}_t \sim o_\phi(\cdot \mid h_t, z_t)$

Recurrent Module (RNN): $h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1})$

Transition function (RNN): $\hat{z}_t \sim p_\phi(\cdot \mid h_t, a_t)$

Representation function: $z_t \sim q_\phi(\cdot \mid h_t, x_t)$

DreamerV2 [Hafner et al. 2021]

- Planning purely inside the world model: $\pi(a_t|z_t)$

- In the paper, they use Actor-Critic and PG methods but potentially you try many things here.

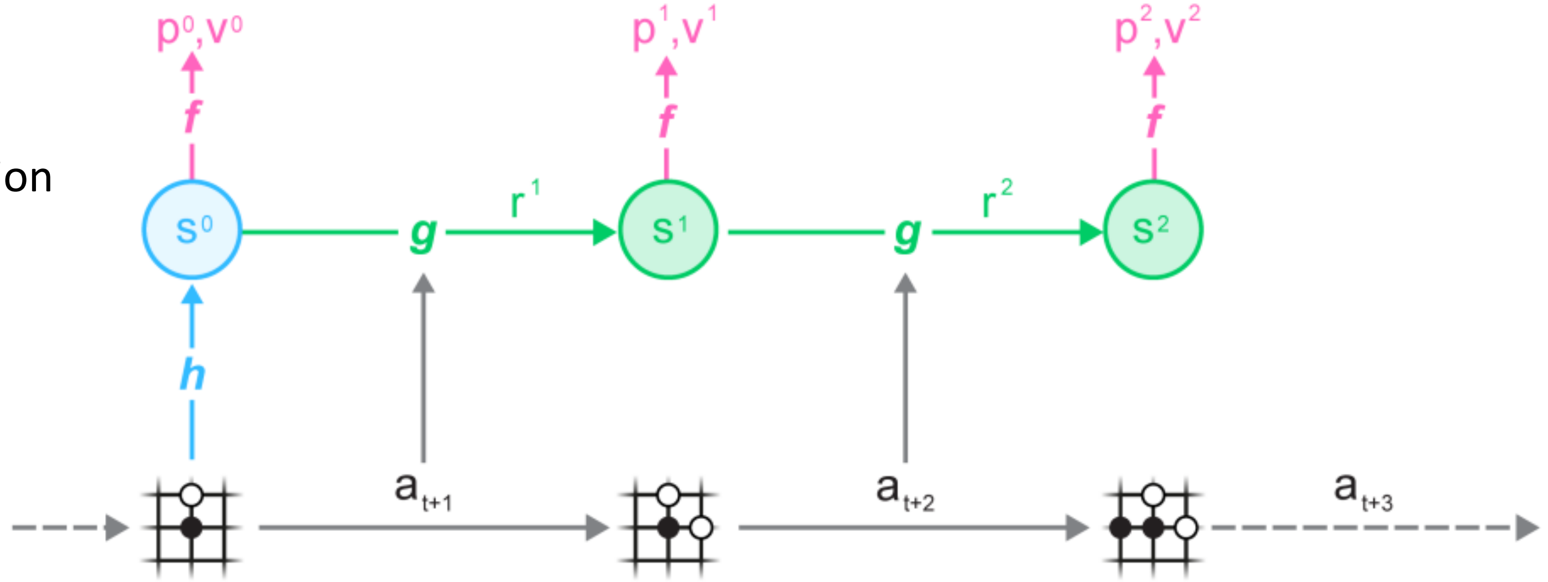
- DreamerV3 [2023]: https://arxiv.org/pdf/2301.04104.pdf

DreamerV2 [Hafner et al. 2021]

- Summary of DreamerV2:
  - Learn Latent world model with discrete latent states and image recovery.
  - Plan in the latent world model with some RL algorithms.

MuZero [Schrittwieser et al. 2020] DeepMind

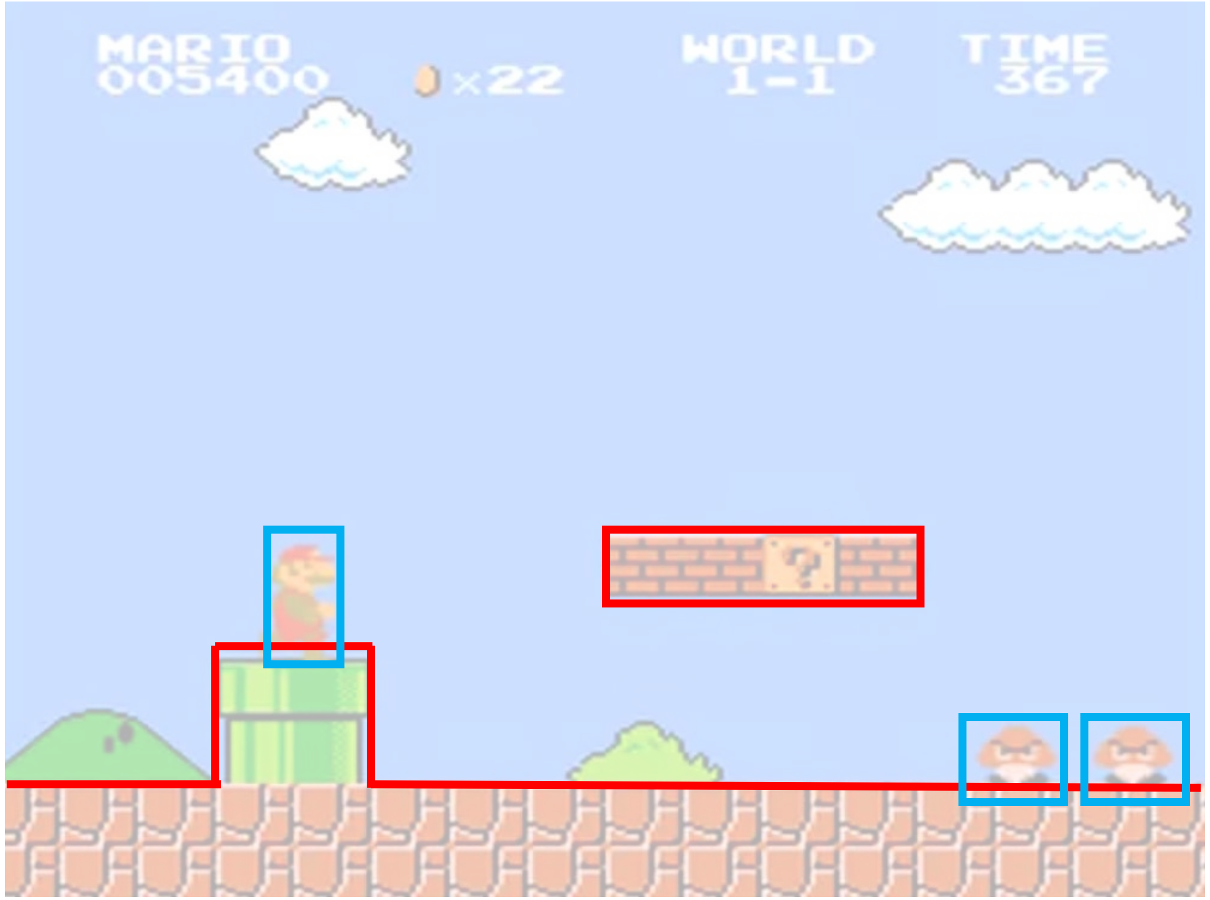
h = representation function
f = prediction function
g = transition function



- Model learning: predicting $\pi(s), V^{\pi}(s)$

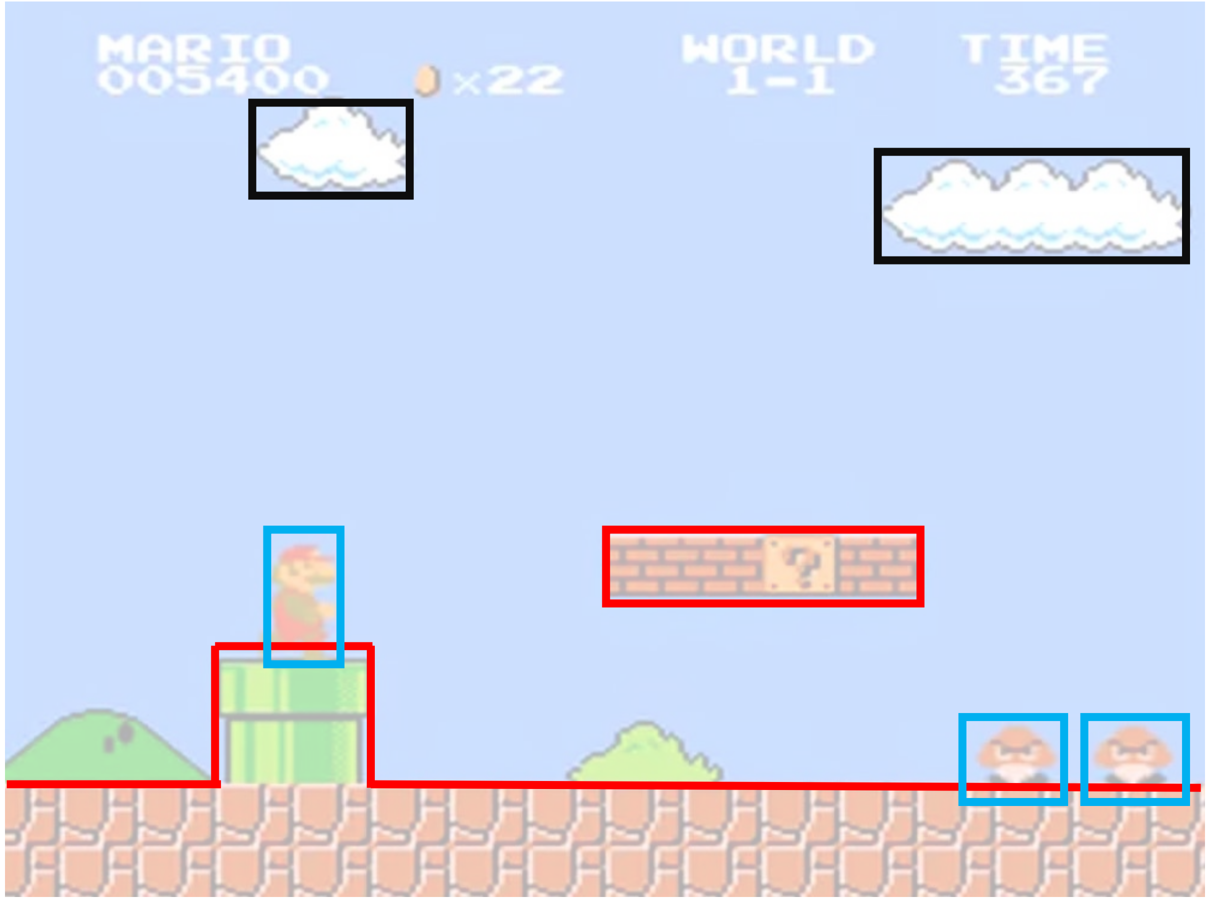
Recover the pixel observation or not?

Recover the pixel observation or not?



- A minimal representation

Recover the pixel observation or not?



- A minimal representation

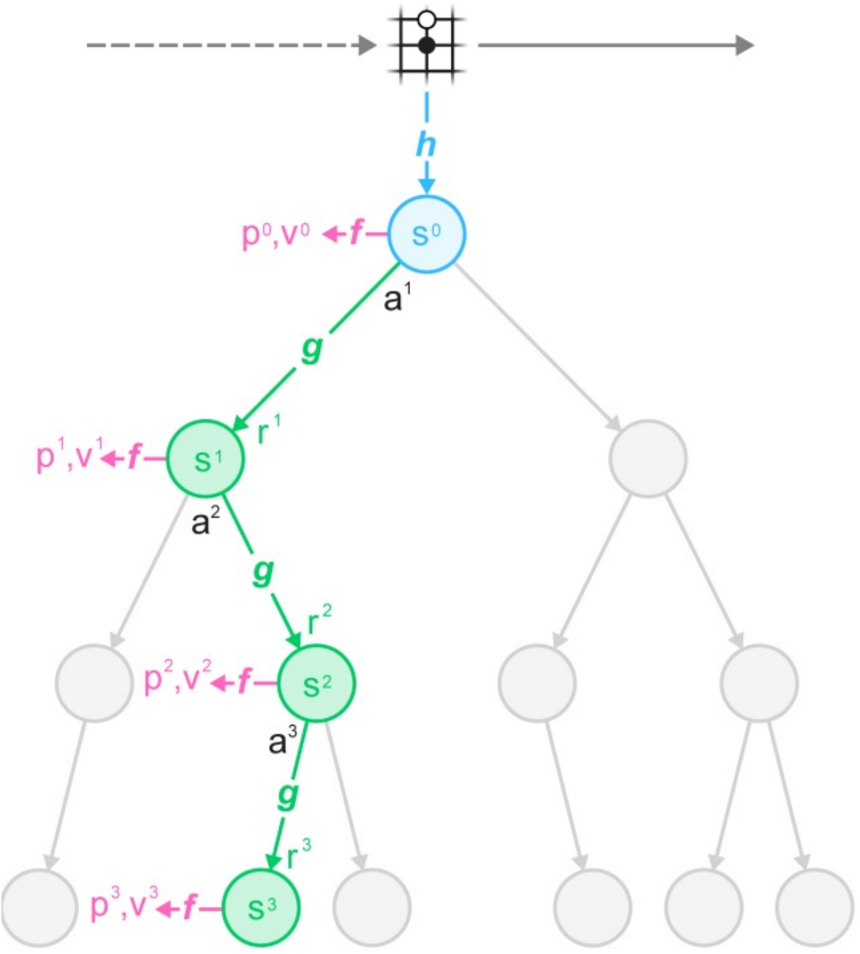
- Representation required to recover the pixels (redundant information)

Recover the pixel observation or not?

- Pros:
  - generalize well across tasks.
- Cons:
  - more redundant representation.
  - Harder to train.

MuZero [Schrittwieser et al. 2020] DeepMind

- Planning via Monte Carlo Tree Search (MCTS)
- Vanilla MCTS only works for deterministic transition.
- Variants exist that handles stochastic transitions.



MuZero [Schrittwieser et al. 2020] DeepMind

- Summary:
  - Learning Latent Model by predicting V, π.
  - Planning inside the learned model with MCTS.