Dream to Control: Learning Behaviors By Latent Imagination

Presenter: Zayne Sprague

10/14/21

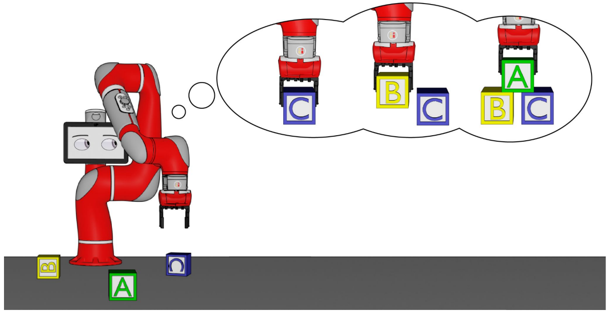
Why Do Humans Model The World?

Modeling the world helps us, as humans, avoid similar outcomes for future situations.

- That argument you lost
- Choosing the wrong class because you thought it sounded good
- Releasing test code into a production server

❖ We are able to learn from these situations because we can model the world and invision, to some degree, what could've changed or been a better outcome.

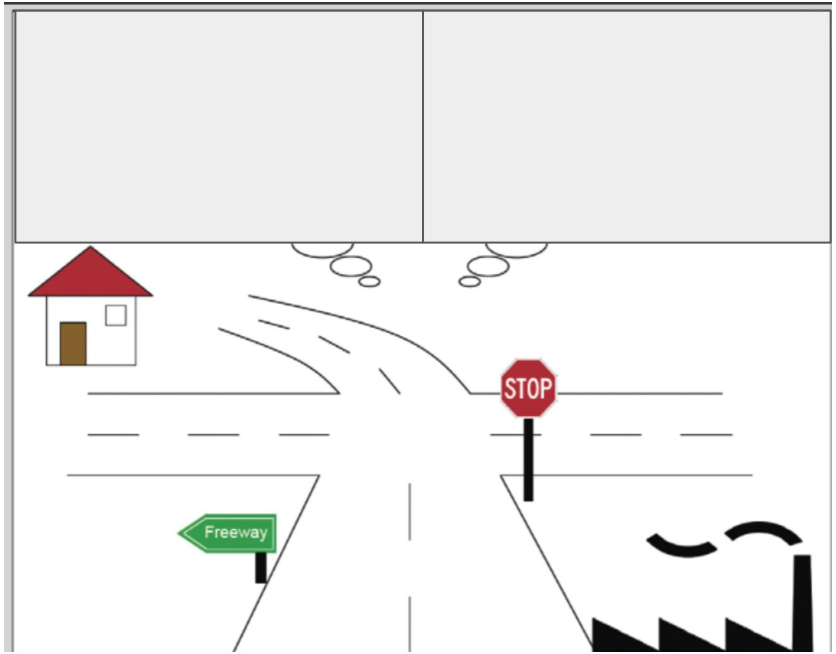
❖ Dream To Control is a paper on giving Robots that power, modeling the world (or environment) its trained in for long-horizons.



How Do Robots See The World?

There are two main ways we can have an Agent learn to interact with the world:

- **Model Based:** Build out a model of the world and use it for predicting rewards and actions



- **Model Free**: The Agent looks at its inputs (vision/sound/etc.) and predicts the world from those, then the corresponding action that follows (Currently getting the best results)

Why stick with Model Based?

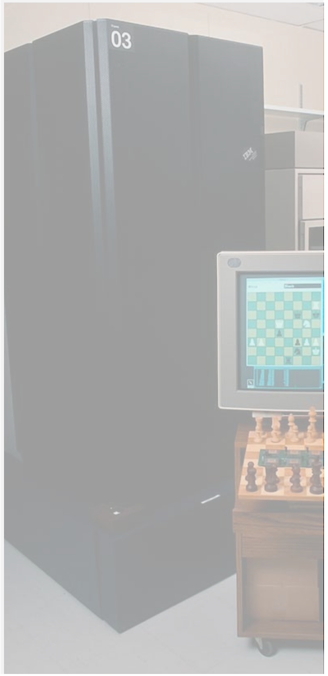
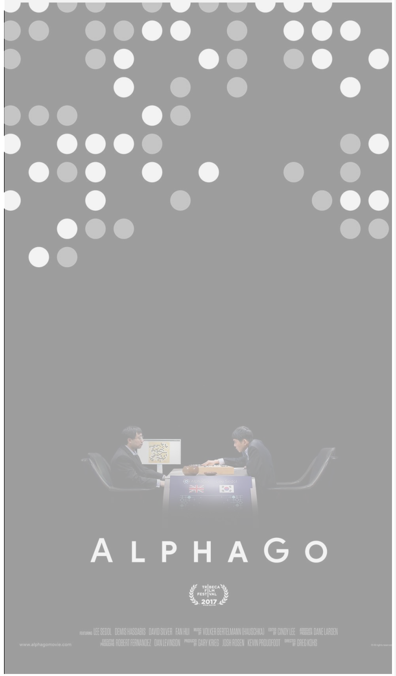
A lot of the benefits of Model Based come from learning Latent Dynamics (an embedding of world you the agent learns to model).

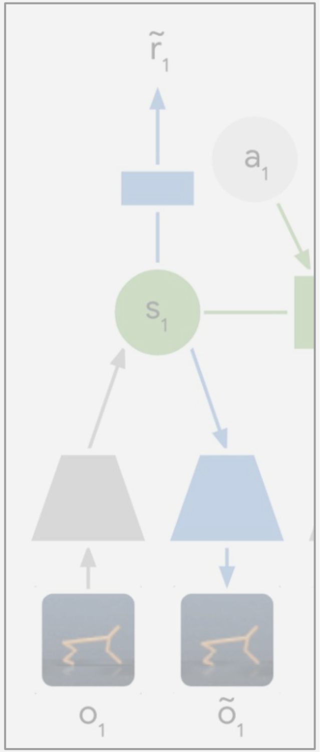
- **Easier State Manipulation**: You no longer have to interact with the environment, capture the state via input modalities (cameras etc.)-- with Latent Dynamics you just mutate a vector
- **Less Training Data**: If the Latent Dynamics is a good model of the world, you can easily create new scenarios for training by manipulating the state vector in novel ways.

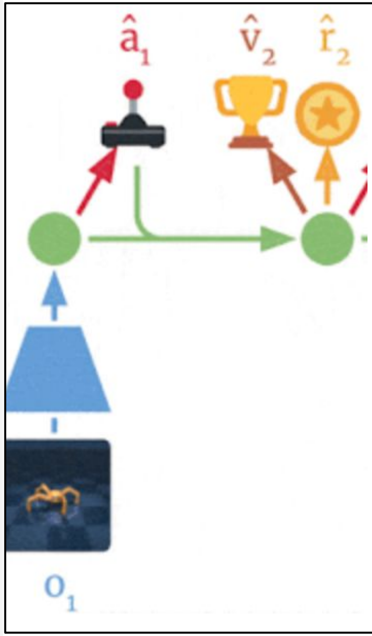
❖ When using Latent Dynamics during training, we say the model is “Imagining” the environment

Progress of Model Based models

- Deep Blue, 1997
- AlphaGo, 2015
- PlaNet, 2019
- Dreamer, 2020





Standard RL Problem Setting

Reinforcement Learning Setup:

- with $t \in [1;T]$ timsteps
- we generate an action $a_t \sim p(a_t \,|\, o_{\leq t}, a_{<t})$
- which then produces a reward and observation $o_t,\, r_t \sim p(\, o_t, r_t \,|\, o_{<t}, a_{<t})$
- with the goal of maximizing the expected rewards $E_p\left(\sum_{t=1}^{T} r_t\right)$

Model-Based Formulation

With Latent Dynamics, you need models predicting states, rewards, and actions

- Starting with real state vectors drawn from experience $p(s_t \mid s_{t-1}, a_{t-1}, o_t)$
    - where p is the distribution where we sample real experiences or "memories"
- We can build a Transition Model without the observation $q(s_\tau \mid s_{t-1},\ a_{t-1})$
    - where q is the distribution where we sample state approximations in latent imagination
    - and τ is a timestep in latent imagination

- From the imagined state we train a Reward Model $q(r_\tau \mid s_\tau)$
- as well as to an Action model $q(a_\tau \mid s_\tau)$
- Ultimately to achieve high imagined rewards $E_q\left(\sum_{\tau=t}^{\infty} \gamma^{\tau-t} r_\tau\right)$

Time Horizons and Related Work

- **What is the time horizon really?**
  - When we restrict the number of rewards the agent can look at
- **What has been done to extend it?**
  - DeepBlue iterated through the moves via manually made rules & algorithms
  - AlphaGo uses an imaginary board with a Neural Net but still defined rules to plan ahead
  - PlaNet created Latent Dynamics (embedding vectors) which made it more resource efficient and allowed the model to learn the rules rather than have them hardcoded
- **What about Model-Free Agents?**
  - Time Horizons are still a major issue, however they are tethered to their input observations

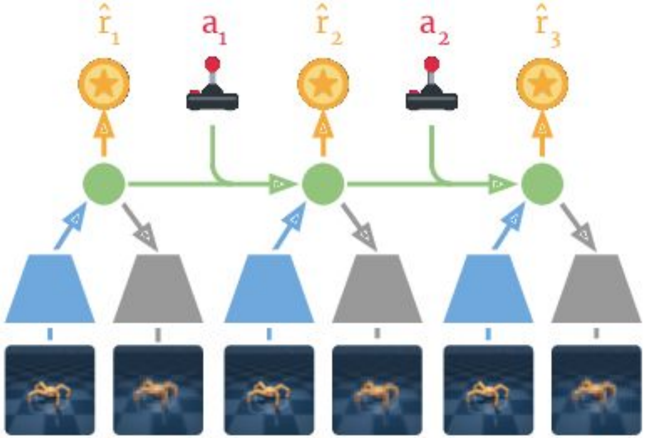
Enter the Dreamer

Model-Based | Latent Dynamics | Long Time Horizons

The Dreamer

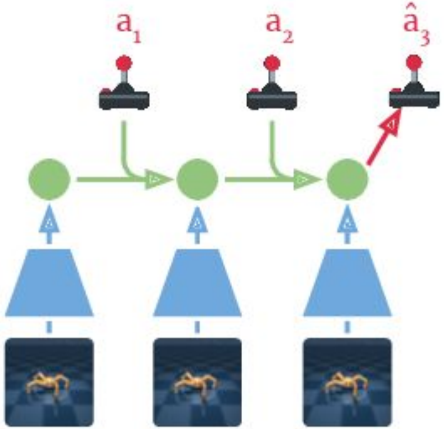
Dream has three main phases during its training

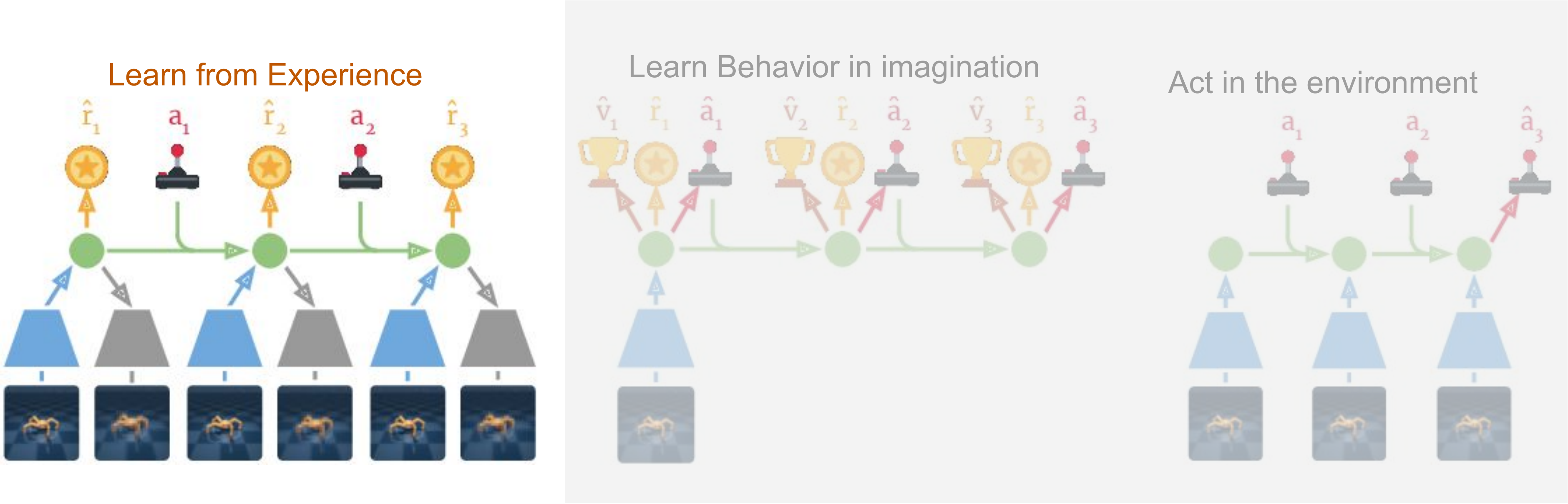
Learn from Experience



Learn Behavior in imagination

Act in the environment




Learn from Experience
$\hat{r}_1$ a_1 $\hat{r}_2$ a_2 $\hat{r}_3$
Learn Behavior in imagination
$\hat{v}_1$ $\hat{r}_1$ $\hat{a}_1$ $\hat{v}_2$ $\hat{r}_2$ $\hat{a}_2$ $\hat{v}_3$ $\hat{r}_3$ $\hat{a}_3$
Act in the environment
a_1 a_2 $\hat{a}_3$

Learning From Experience

Learning From Experience

Dreamer optimizes 4 models Jointly via shared parameters θ

- The Representation Model $p_\theta(s_t \mid s_{t-1},\, a_{t-1},\, o_t)$
- The Reward Model $q_\theta(r_t \mid s_t)$
- The Transition Model $q_\theta(s_t \mid s_{t-1},\, a_{t-1})$
- And a new model, the State Model $q_\theta(s_t \mid o_t)$

Each model is responsible for an Objective Task

$$j_D^t = -\beta KL(p(s_t \mid s_{t-1}, a_{t-1}, o_{t-1}) || q(s_t \mid s_{t-1},\, a_{t-1}))$$

Can we predict the state without the observation

$$j_r^t = \ln q(r_t \mid s_t)$$

Can we predict rewards

$$j_S^t \doteq \ln \left(q(s_t \mid o_t) - \ln \left(\sum_{o'} q(s_t \mid o') \right) \right)$$

Are the states unique given different observations

Bringing it all together

$$\jmath_D^t = -\beta KL(p(s_t \mid s_{t-1}, a_{t-1}, o_{t-1})||q(s_t \mid s_{t-1},\, a_{t-1}))$$

Can we predict the state without the observation

$$\jmath_r^t \ln q(r_t|s_t|)$$

Can we predict rewards

$$\jmath_S^t \doteq \ln\left(q(s_t \mid o_t) \;-\; \ln\left(\sum_{o'} q(s_t \mid o')\right)\right)$$

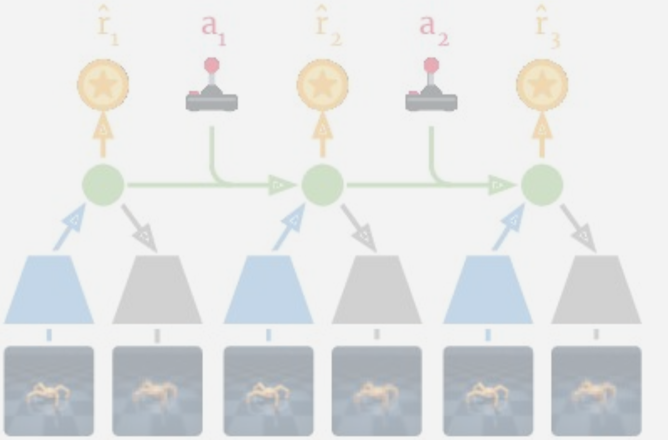
Are the states unique given different observations

Training is done via Noise Contrastive Estimation

$$\jmath_{NCE} \doteq E\left(\sum_t (\jmath_S^t + \jmath_R^t + \jmath_D^t)\right)$$

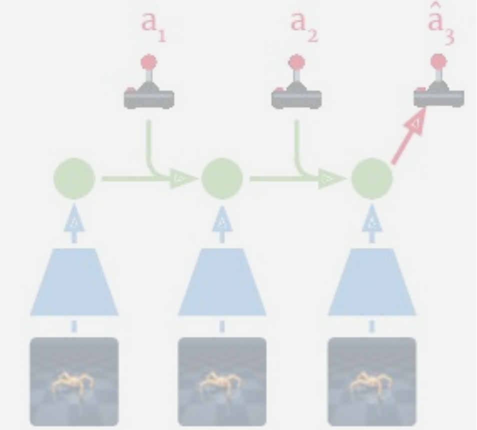
In human words - we want to train our model to be able to predict states without observations, rewards given states, and create unique state vectors when given observations.

Learn from Experience



Learn Behavior in imagination

Act in the environment



Learning Behavior in Imagination

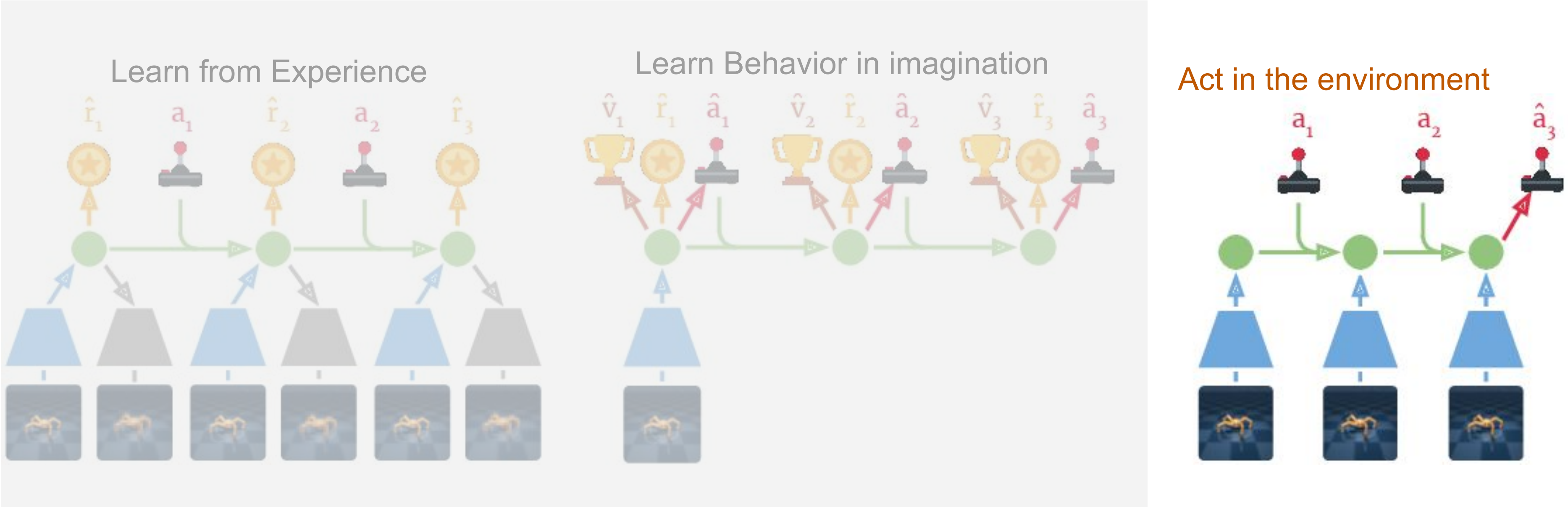
Learning Behavior and the Long Horizon

Dreamer overcomes the Time Horizon problem through an Actor Critic approach

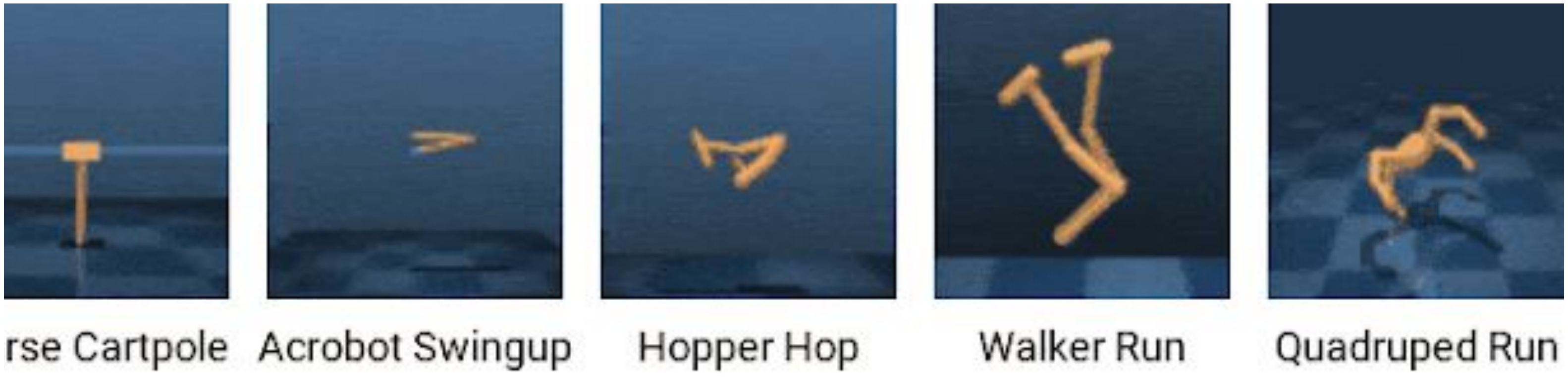
- The action model (actor) $a_t \sim q(a_\tau \mid s_\tau)$
  - Estimates actions based on value
- The value model (critic) $v_\psi(s_\tau) \approx E_{q(\cdot \mid s_\tau)}\left(\sum_{\tau=t}^{t+H} \gamma^{\tau-t} r_\tau\right)$
  - estimates rewards from the changing action model

The Training Objective (how do we update these models)

- Action Model $\max_{\phi} E_{q\theta,q\phi}\left(\sum_{\tau=t}^{t+H} V_\lambda(s_\tau)\right)$
- Value Model $\min_{\psi} E_{q\theta,q\phi}\left(\sum_{\tau=t}^{t+H} \frac{1}{2}||v_\psi(s_\tau) - V_\lambda(s_\tau)||^2\right)$


Learn from Experience
$\hat{r}_1$
a_1
$\hat{r}_2$
a_2
$\hat{r}_3$
Learn Behavior in imagination
$\hat{v}_1$
$\hat{r}_1$
$\hat{a}_1$
$\hat{v}_2$
$\hat{r}_2$
$\hat{a}_2$
$\hat{v}_3$
$\hat{r}_3$
$\hat{a}_3$
Act in the environment
a_1
a_2
$\hat{a}_3$

Do the Stuff!


rse Cartpole
Acrobot Swingup
Hopper Hop
Walker Run
Quadruped Run

Training Life Cycle

Algorithm 1: Dreamer

Initialize dataset $\mathcal{D}$ with S random seed episodes.
Initialize neural network parameters θ, ϕ, ψ randomly.

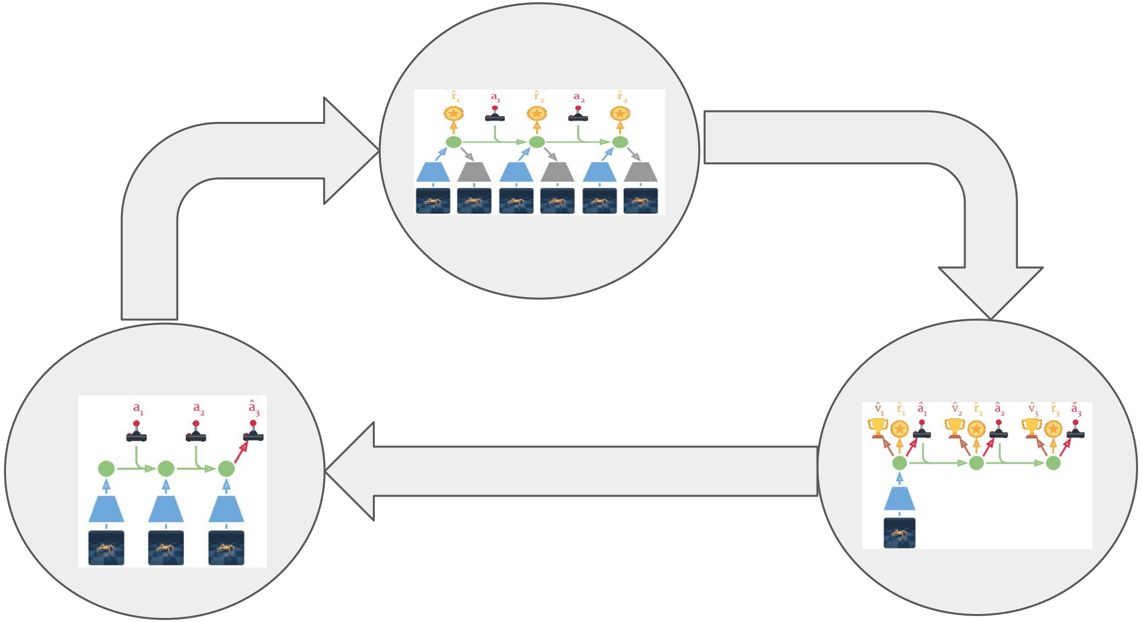
for *update step* $c = 1..C$ **do**

Model components

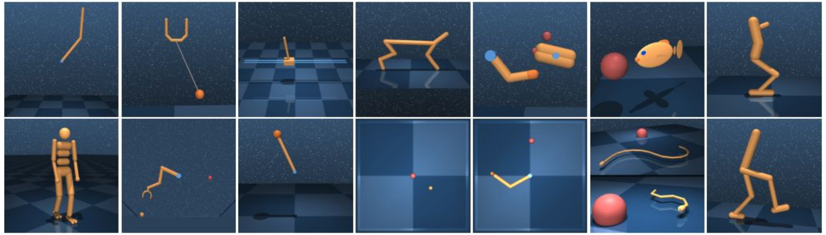
Representation	$p_\theta(s_t \mid s_{t\text{-}1}, a_{t\text{-}1}, o_t)$
Transition	$q_\theta(s_t \mid s_{t\text{-}1}, a_{t\text{-}1})$
Reward	$q_\theta(r_t \mid s_t)$
Action	$q_\phi(a_t \mid s_t)$
Value	$v_\psi(s_t)$

Hyper parameters

Seed episodes	S
Collect interval	C
Batch size	B
Sequence length	L
Imagination horizon	H
Learning rate	α



Testing Dreamer’s ability to Control



- **Tests were ran on the DeepMind Control Suite**
  - Image Observations of the environment were 64x64x3
  - Each env has up to 12 discrete actions
  - Episodes have 1000 steps and random initial states
- **Results are compared against 3 other models**
  - PlaNet (model-based)
  - D4PG (model-free)
  - A3C (model-free)

What are we testing

Dreamer is compared to other models through various metrics

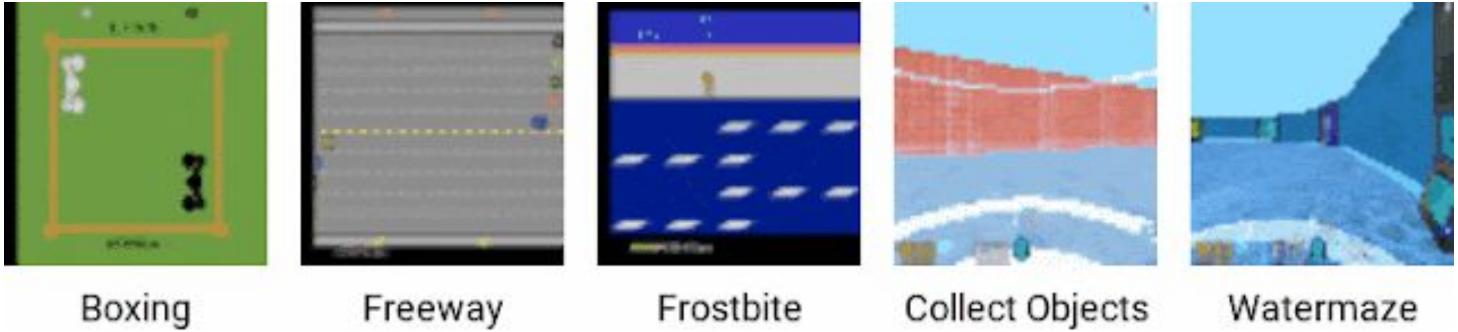
- Episode Return
  - The main objective in these tasks
  - Higher is better

- Data Efficiency
  - How much data is needed
  - How long do we have to train

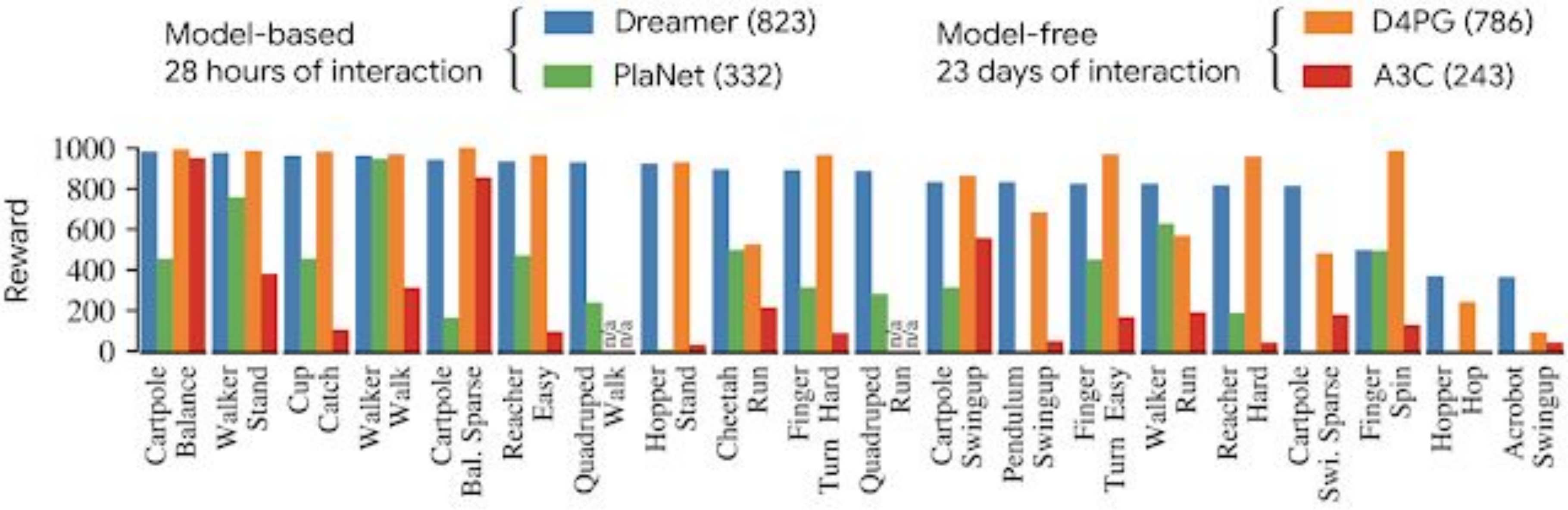
- Time Horizons
  - How robust is Dreamer to long time horizons

Atari Examples

- **Dreamer learns to play games that have complexity (long term tasks) in atari 2D environments**



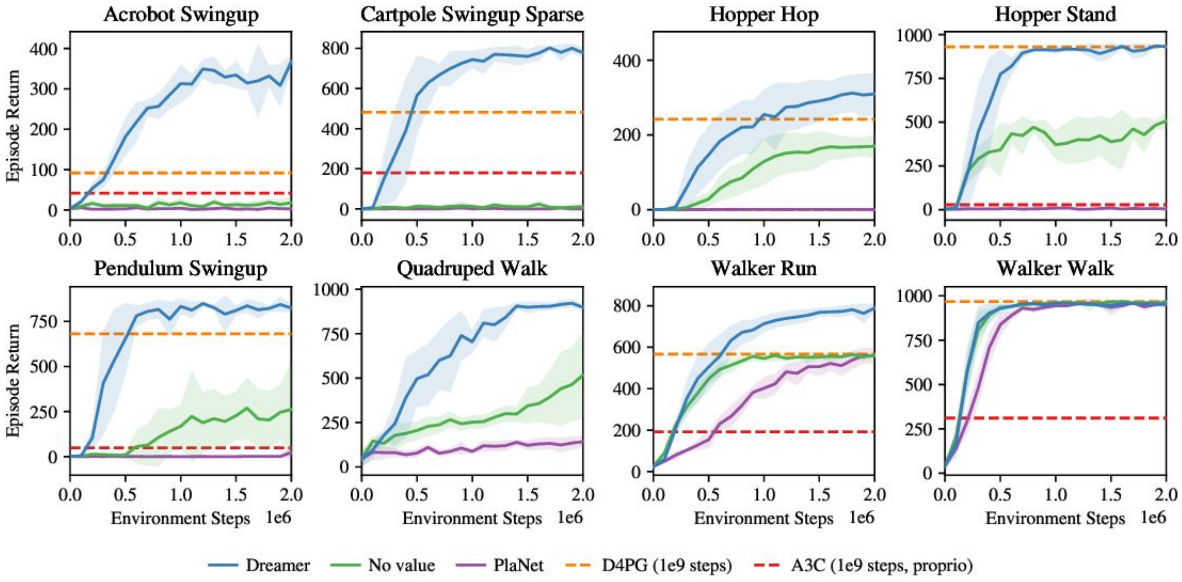
Episode Return Comparisons: Dreaming Pays Off



- **Dreamer Outperforms the Model-Based Model as well as the Model-Free on some tasks!**
- **On average, Dreamer outperforms all models (numbers next to model indicate avg return)**

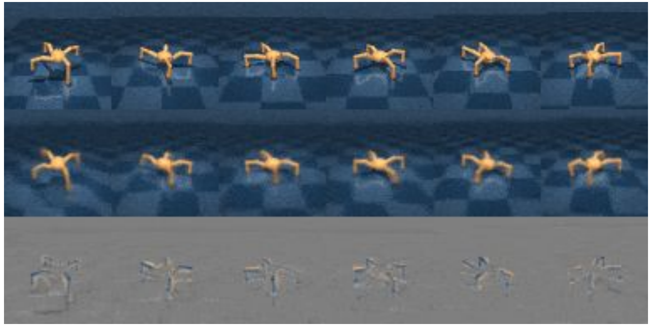
Dreaming is Efficient and Long Horizons Matter

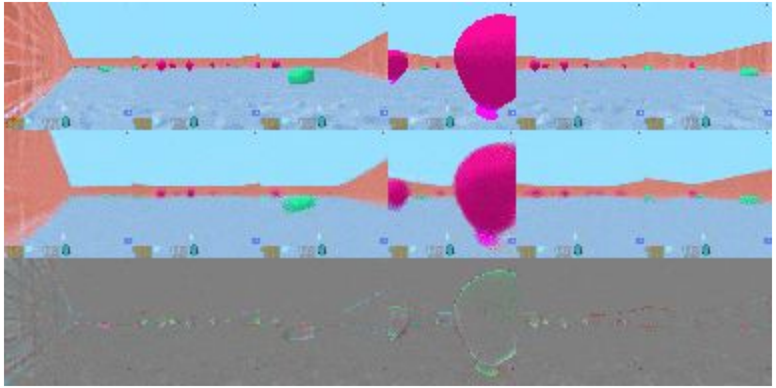
- **Dreamer does not require as long to train as model-free agents**
- **Dreamer performs best in environments that require long range planning (Acrobot Swingup)**

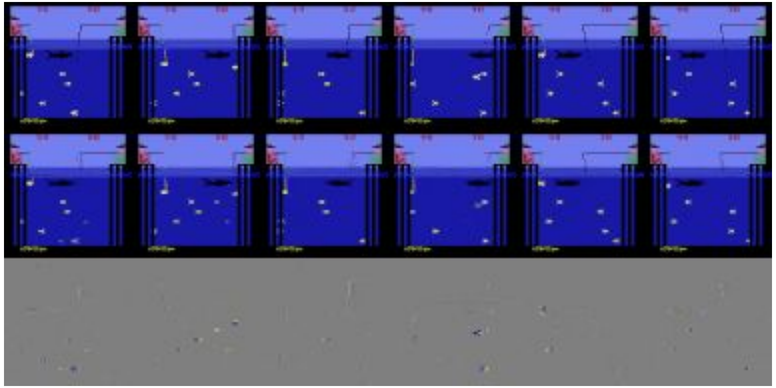


What is the dreamer dreaming?

- Top: What Happened
- Middle: What Dreamer imagined
- Bottom: Differences







Where can Dreamer Improve & Critiques

- **The paper only compares against one Model-Based agent, PlaNet (a model the authors built prior to Dreamer)**
- **The actions taken are discrete only, which is not applicable to the real world**
- **The evaluation and interpretation of the World Model (the Latent Dynamics) in Dreamer are not well defined and could be a source for future work.**

How can Dreamer improve

- **Representational Learning**
  - How can Dreamer learn the model more efficiently (pixel wise reconstruction is expensive and not all details matter)
- **The World Model**
  - How can Dreamer better encode the model, learn concepts, and capture more data efficiently
- **Better Actions**
  - How can Dreamer be used for Continuous Actions instead of Discrete.
- **Real World Training**
  - How does Dreamer work with real world training

Dreamer V2 and Extended Readings

- Mastering Atari With Discrete World Models
  - DreamerV2, encodes the world state better, applicable to continuous action, beats Model-Free & Human Level Performance in Atari Games!
- CURL: Contrastive Unsupervised Representations for Reinforcement Learning
  - A better way to encode representational learning via pixel observations
- Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning From Pixels
  - A way to train model-free latent dynamics quicker and with less data from pixels
- Latent Skill Planning for Exploration and Transfer
  - How can Dreamers learned knowledge of the environment transfer to new tasks

Summary of Dreamer

- Latent Dynamics can learn world models and be used to train Behavior efficiently
- Dreamer uses a State Model to quickly learn from observations with compact models
- Prior work has struggled to overcome the Time Horizon
- The Long Time Horizon problem can be reduced via Actor Critic methods (the value model)
- Where you are matters -- what’s the value of being in the current state (the value model)
- On average, dreamer outperformed model based and some model free algorithms