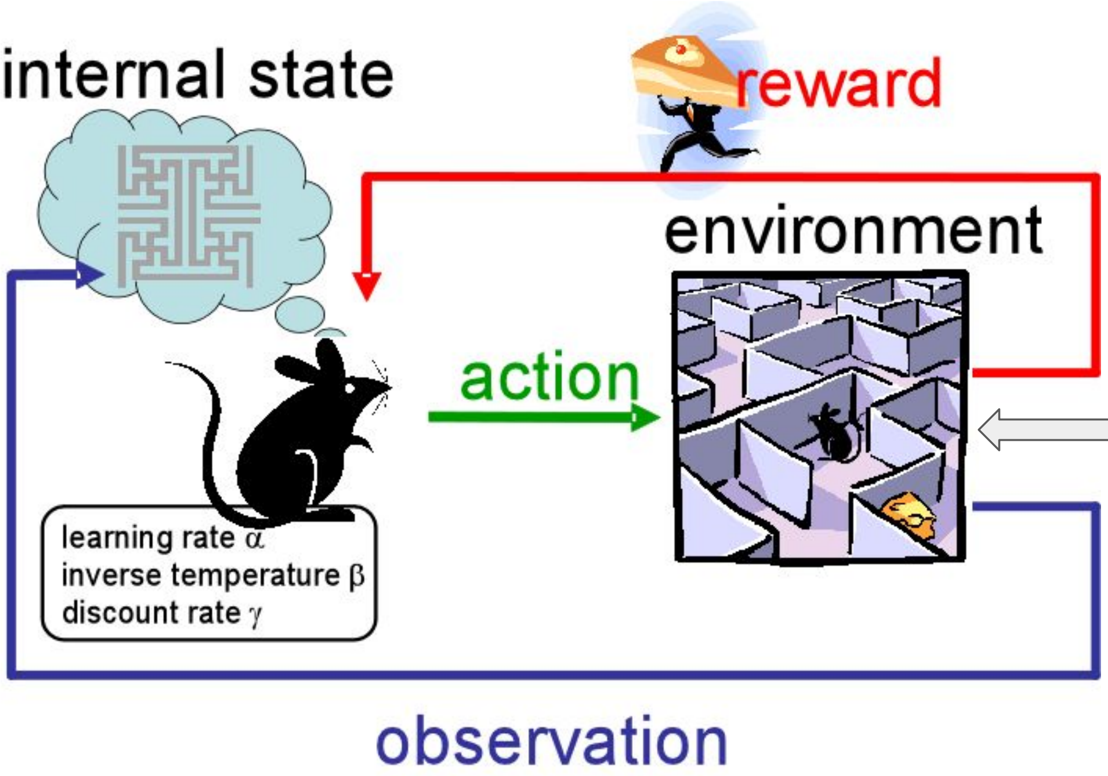
Dream to Control: Learning Behaviors by Latent Imagination

Presenter: Thomas Nathaniel Plaxton

10/04/2022

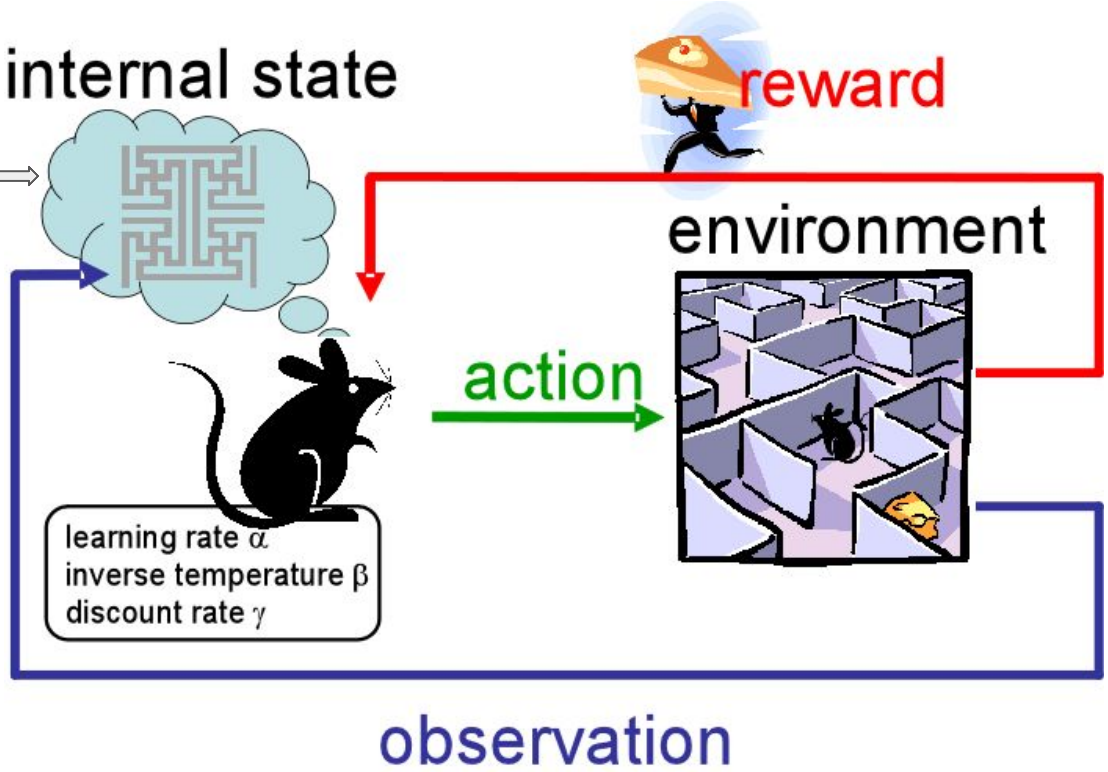
Traditional Reinforcement Learning



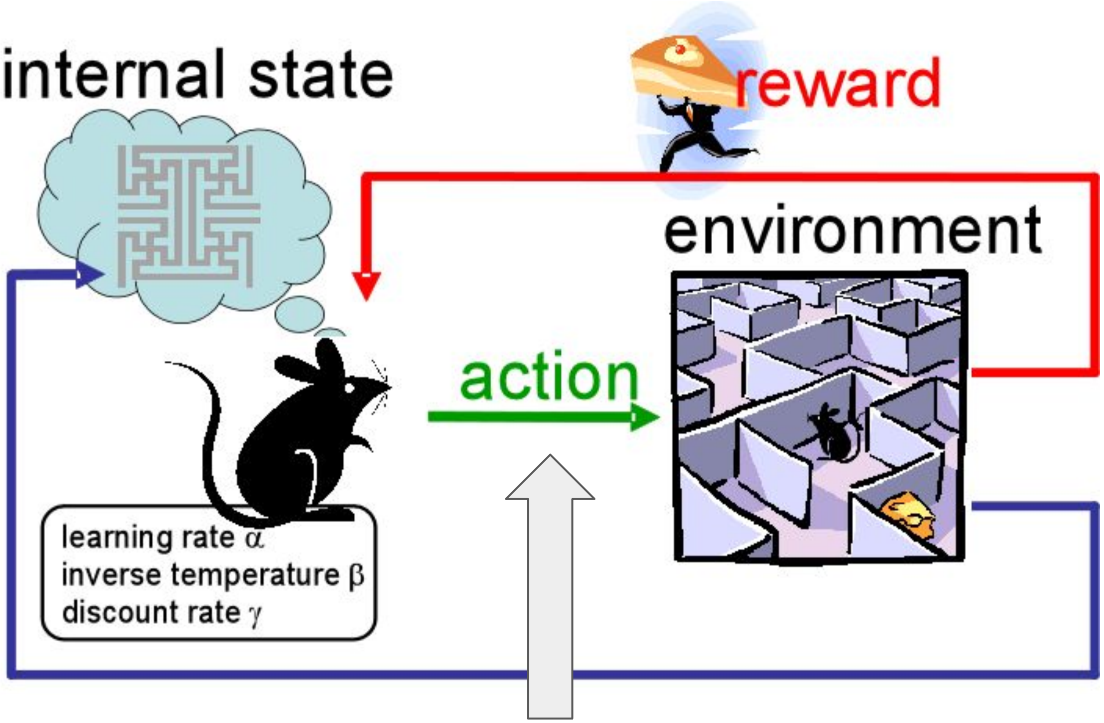
Agent makes an observation of its environment

Traditional Reinforcement Learning

Agent updates its internal state according to the observation

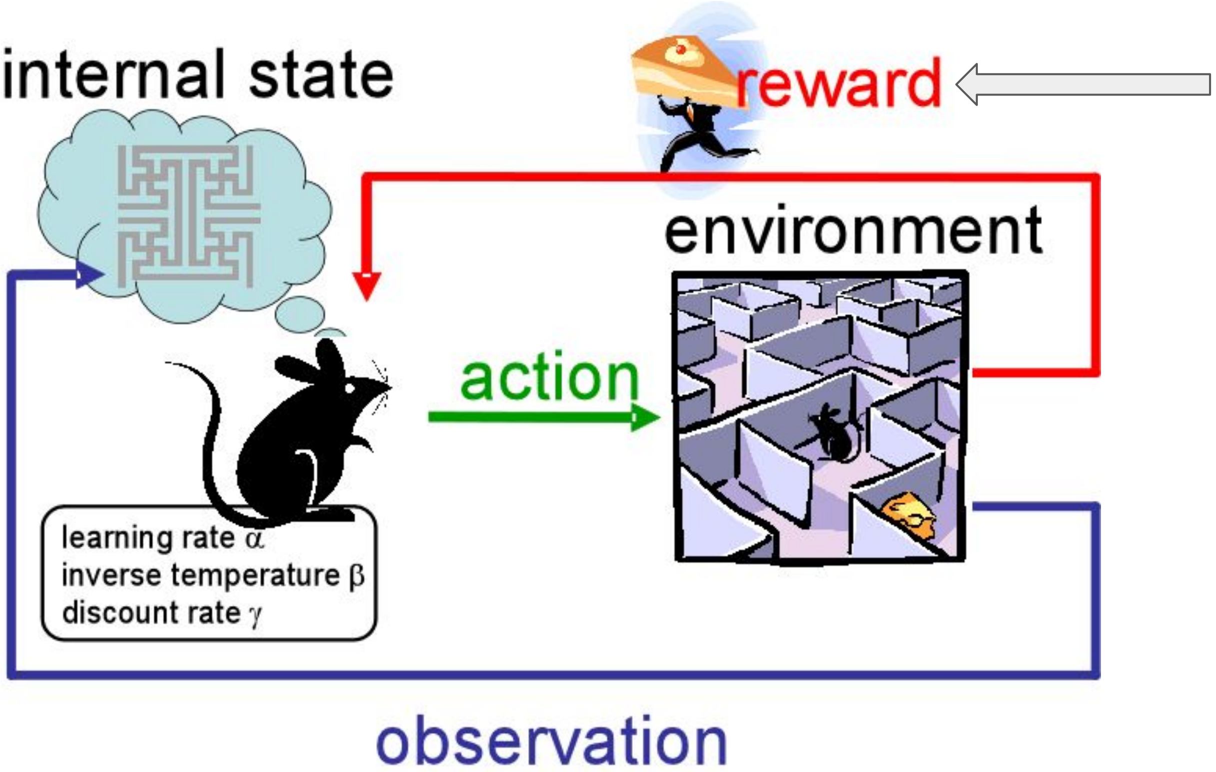


Traditional Reinforcement Learning



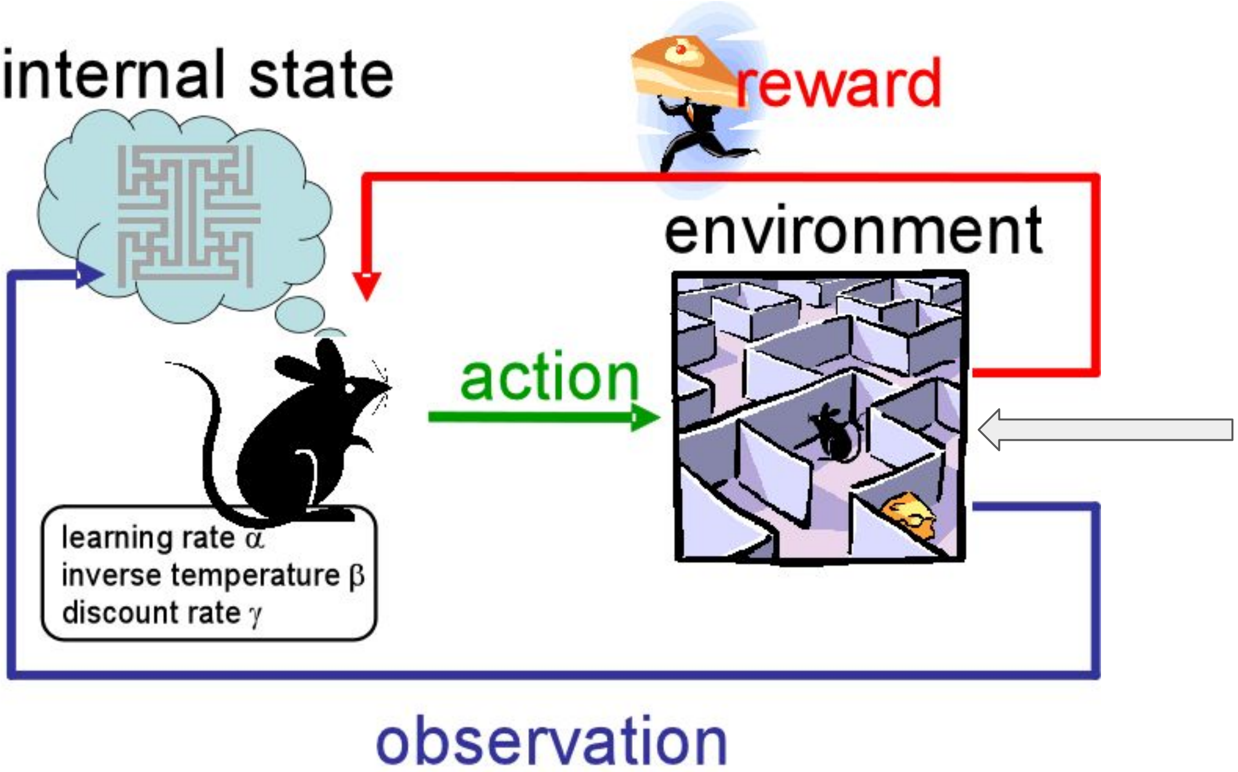
Agent acts according to its policy

Traditional Reinforcement Learning



Agent receives feedback (positive, negative, or neutral) for its action

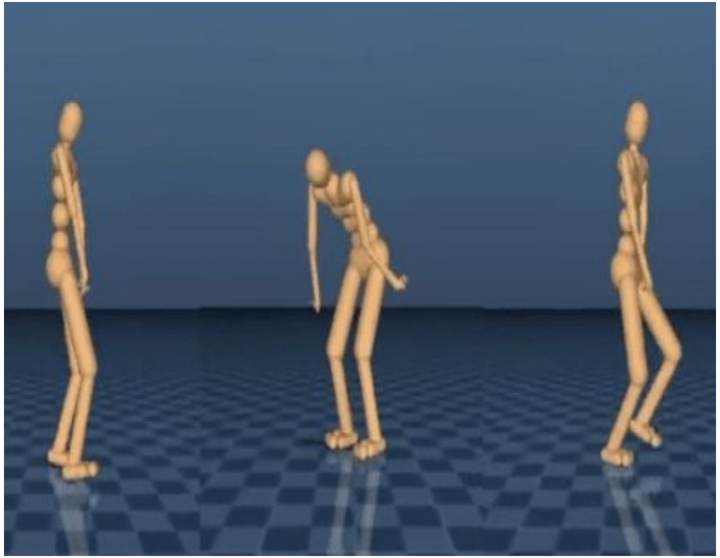
Traditional Reinforcement Learning



Cycle repeats…

Main Limitation of this Approach

- Interacting in the environment can be expensive!
- State-of-the-art can take hundreds of thousands of episodes to learn



Big Question:

- **Can we train our policies outside of the environment?**

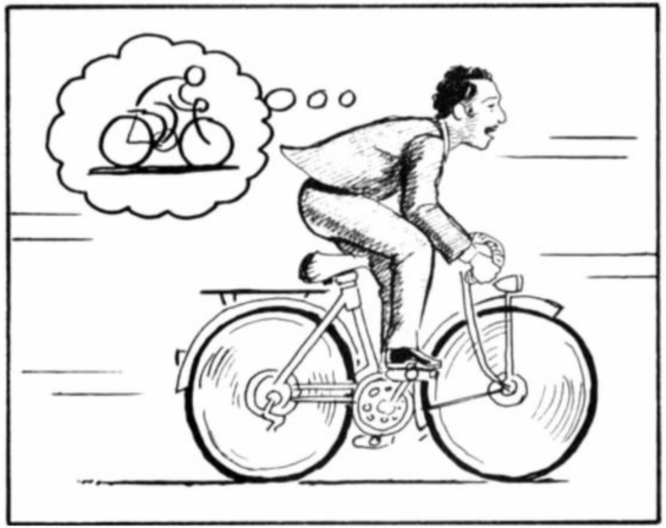


Figure 1. A World Model, from Scott McCloud's *Understanding Comics*. (McCloud, 1993; E, 2012)

Problem Setting

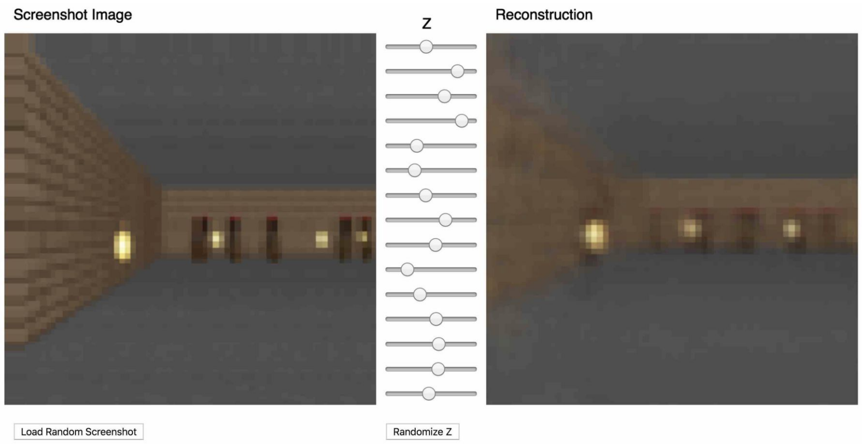
Working On Visual Control Problems (within DeepMind Control Suite)

- Formulate Visual Control as a Partially Observable Markov Decision Process (POMDP) with:
  - Discrete time-steps t ∈ [1; T]
  - Continuous vector-valued, agent-generated actions $a_t \sim p(a_t | o_{\leq t}, a_{<t})$
  - High-dimensional observations, scalar rewards generated by the environment $o_t, r_t \sim p(o_t, r_t | o_{<t}, a_{<t}))$

Prior Work

World Models (David Ha, Jurgen Schmidhuber (2018))

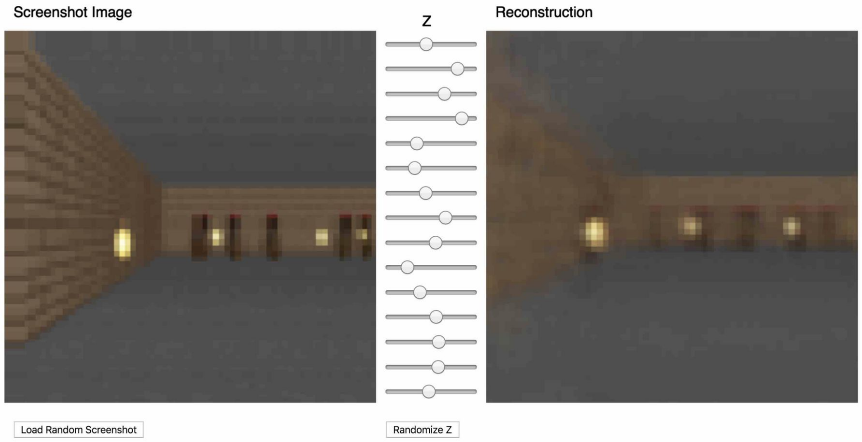
- ❖ Learn a dynamics model for a RL environment



Prior Work

World Models (David Ha, Jurgen Schmidhuber (2018))

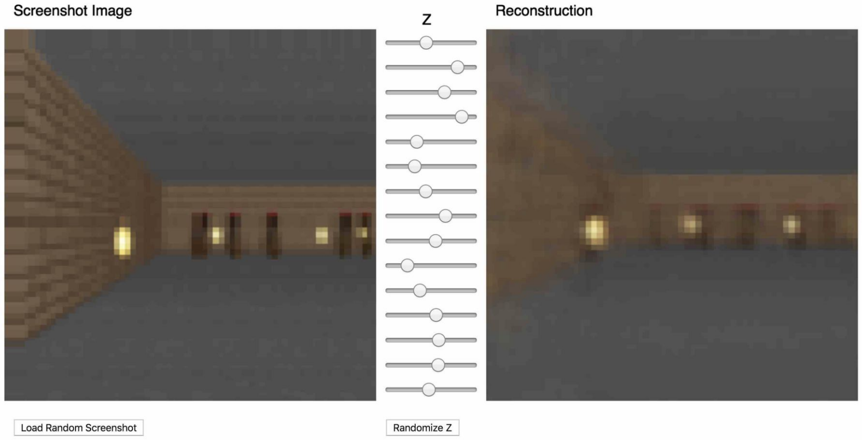
- ❖ Learn a dynamics model for a RL environment
- ❖ The model's latent space contains the key features, making learning an optimal policy easier



Prior Work

World Models (David Ha, Jurgen Schmidhuber (2018))

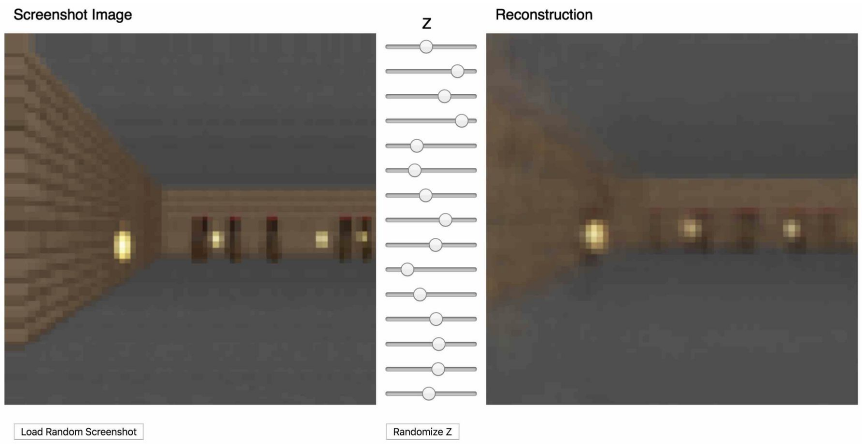
- Learn a dynamics model for a RL environment
- The model's latent space contains the key features, making learning an optimal policy easier
- Can train a model entirely in the latent space



Prior Work

World Models (David Ha, Jurgen Schmidhuber (2018))

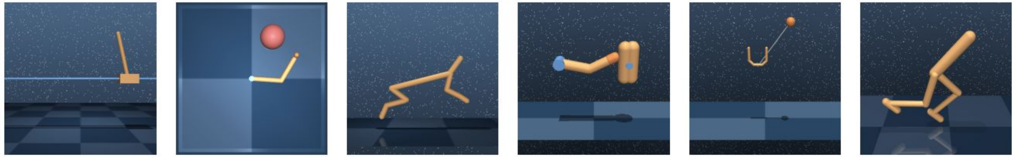
- ❖ Learn a dynamics model for a RL environment
- ❖ The model’s latent space contains the key features, making learning an optimal policy easier
- ❖ Can train a model entirely in the latent space
- ❖ Save training time and resources



Prior Work

Learning Latent Dynamics for Planning with Pixels (Hafner et. al (2019))

❖ Learn a dynamics model for different tasks in the DeepMind Control suite



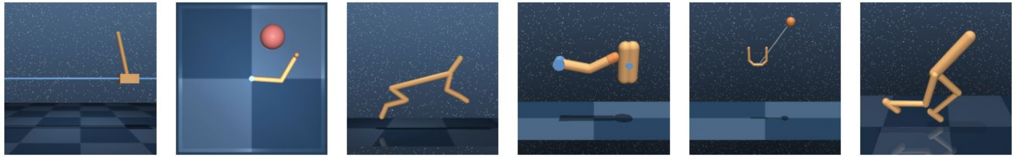
(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

Method	Modality	Episodes	Cartpole Swing Up	Reacher Easy	Cheetah Run	Finger Spin	Cup Catch	Walker Walk
A3C	proprioceptive	100,000	558	285	214	129	105	311
D4PG	pixels	100,000	862	967	524	985	980	968
PlaNet (ours)	pixels	1,000	821	832	662	700	930	951
CEM + true simulator	simulator state	0	850	964	656	825	993	994
Data efficiency gain PlaNet over D4PG (factor)			250	40	500+	300	100	90

Prior Work

Learning Latent Dynamics for Planning with Pixels (Hafner et. al (2019))

- ❖ Learn a dynamics model for different tasks in the DeepMind Control suite
- ❖ Plan only using the latent space of the dynamics model (PlaNet)



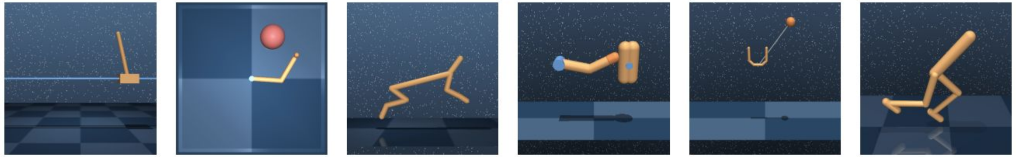
(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

Method	Modality	Episodes	Cartpole Swing Up	Reacher Easy	Cheetah Run	Finger Spin	Cup Catch	Walker Walk
A3C	proprioceptive	100,000	558	285	214	129	105	311
D4PG	pixels	100,000	862	967	524	985	980	968
PlaNet (ours)	pixels	1,000	821	832	662	700	930	951
CEM + true simulator	simulator state	0	850	964	656	825	993	994
Data efficiency gain PlaNet over D4PG (factor)			250	40	500+	300	100	90

Prior Work

Learning Latent Dynamics for Planning with Pixels (Hafner et. al (2019))

- ❖ Learn a dynamics model for different tasks in the DeepMind Control suite
- ❖ Plan only using the latent space of the dynamics model (PlaNet)
- ❖ Generalize to include multi-step predictions in latent space



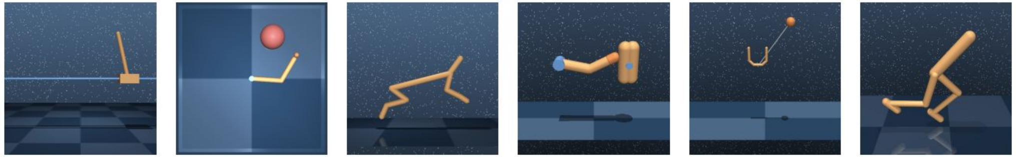
(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

Method	Modality	Episodes	Cartpole Swing Up	Reacher Easy	Cheetah Run	Finger Spin	Cup Catch	Walker Walk
A3C	proprioceptive	100,000	558	285	214	129	105	311
D4PG	pixels	100,000	862	967	524	985	980	968
PlaNet (ours)	pixels	1,000	821	832	662	700	930	951
CEM + true simulator	simulator state	0	850	964	656	825	993	994
Data efficiency gain PlaNet over D4PG (factor)			250	40	500+	300	100	90

Prior Work

Learning Latent Dynamics for Planning with Pixels (Hafner et. al (2019))

- ❖ Learn a dynamics model for different tasks in the DeepMind Control suite
- ❖ Plan only using the latent space of the dynamics model (PlaNet)
- ❖ Generalize to include multi-step predictions in latent space
- ❖ Performance on par with current state-of-the-art model-free approaches, with ~200x less environment interactions



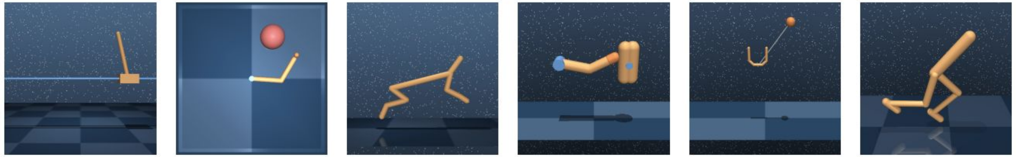
(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

Method	Modality	Episodes	Cartpole Swing Up	Reacher Easy	Cheetah Run	Finger Spin	Cup Catch	Walker Walk
A3C	proprioceptive	100,000	558	285	214	129	105	311
D4PG	pixels	100,000	862	967	524	985	980	968
PlaNet (ours)	pixels	1,000	821	832	662	700	930	951
CEM + true simulator	simulator state	0	850	964	656	825	993	994
Data efficiency gain PlaNet over D4PG (factor)			250	40	500+	300	100	90

Prior Work

Learning Latent Dynamics for Planning with Pixels (Hafner et. al (2019))

- Learn a dynamics model for different tasks in the DeepMind Control suite
- Plan only using the latent space of the dynamics model (PlaNet)
- Generalize to include multi-step predictions in latent space
- Performance on par with current state-of-the-art model-free approaches, with ~200x less environment interactions
- Has to used gradient-free planning



(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

Method	Modality	Episodes	Cartpole Swing Up	Reacher Easy	Cheetah Run	Finger Spin	Cup Catch	Walker Walk
A3C	proprioceptive	100,000	558	285	214	129	105	311
D4PG	pixels	100,000	862	967	524	985	980	968
PlaNet (ours)	pixels	1,000	821	832	662	700	930	951
CEM + true simulator	simulator state	0	850	964	656	825	993	994
Data efficiency gain PlaNet over D4PG (factor)			250	40	500+	300	100	90

Prior Work

Learning Latent Dynamics for Planning with Pixels (Hafner et. al (2019))

❖ Learn a dynamics model for different tasks in the DeepMind Control suite

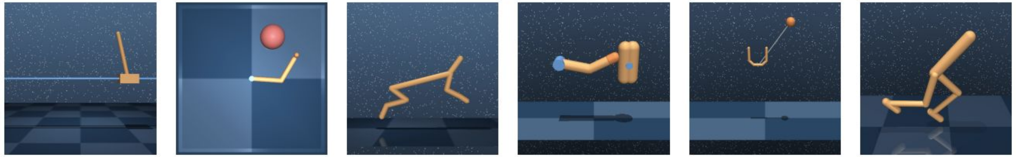
❖ Plan only using the latent space of the dynamics model (PlaNet)

❖ Generalize to include multi-step predictions in latent space

❖ Performance on par with current state-of-the-art model-free approaches, with ~200x less environment interactions

❖ Has to used gradient-free planning

❖ Cannot approximate sum of rewards beyond planning horizon



(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

Method	Modality	Episodes	Cartpole Swing Up	Reacher Easy	Cheetah Run	Finger Spin	Cup Catch	Walker Walk
A3C	proprioceptive	100,000	558	285	214	129	105	311
D4PG	pixels	100,000	862	967	524	985	980	968
PlaNet (ours)	pixels	1,000	821	832	662	700	930	951
CEM + true simulator	simulator state	0	850	964	656	825	993	994
Data efficiency gain PlaNet over D4PG (factor)			250	40	500+	300	100	90

Main Contributions

- **Iterative approach for exploring in the environment and gathering new observations**
  - World Models paper randomly explored in the environment to create the dynamics model
  - Instead, explore the environment according to our current policy

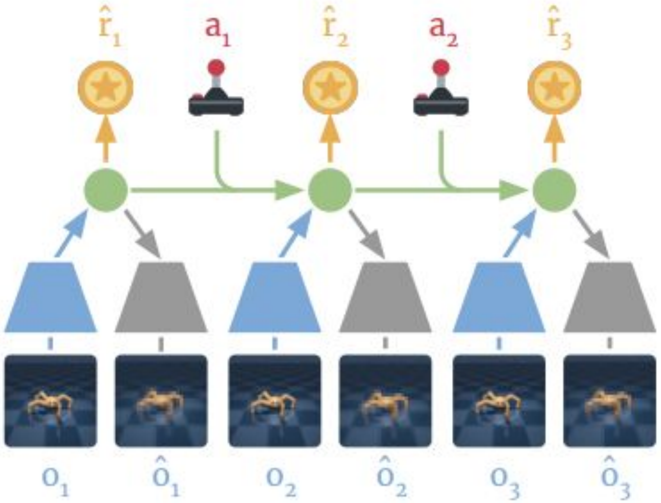
Main Contributions

- **Iterative approach for exploring in the environment and gathering new observations**
  - World Models paper randomly explored in the environment to create the dynamics model
  - Instead, explore the environment according to our current policy
- **Rather than just predict actions given a state, predict state values**
  - Allows for faster convergence to an optimal policy by learning long-horizon behaviors
  - Given value function setup allows for back propagation of value function through dynamics model's latent space

Main Contributions

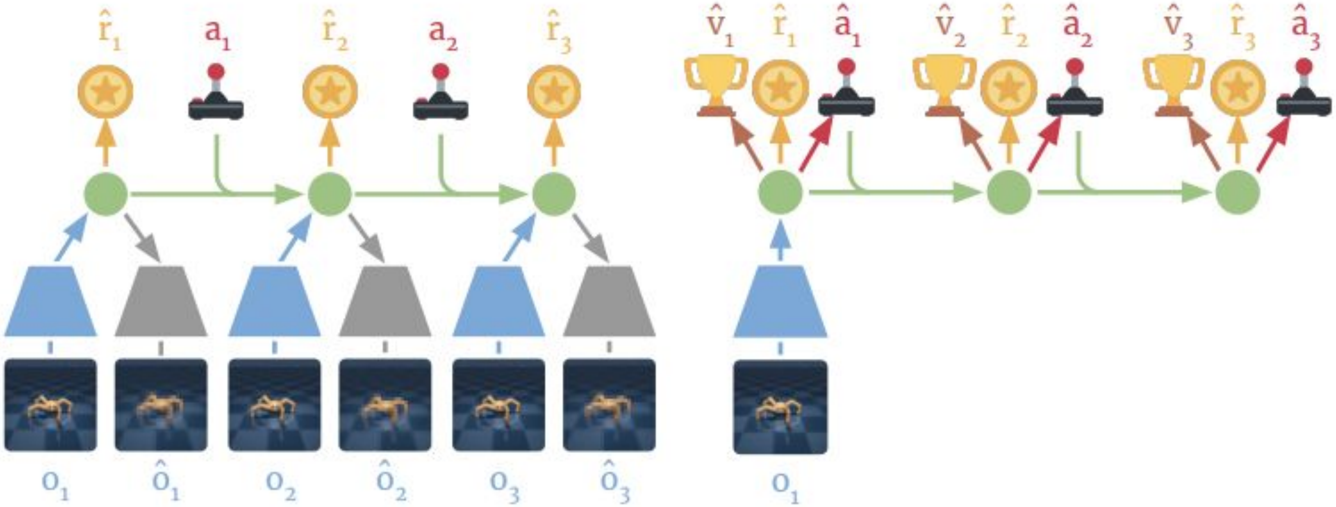
- **Iterative approach for exploring in the environment and gathering new observations**
  - World Models paper randomly explored in the environment to create the dynamics model
  - Instead, explore the environment according to our current policy
- **Rather than just predict actions given a state, predict state values**
  - Allows for faster convergence to an optimal policy by learning long-horizon behaviors
  - Given value function setup allows for back propagation of value function through dynamics model’s latent space
- **Demonstration of Efficacy of Approach**
  - Pair Dreamer with different representation learning approaches
  - Analyze performance in the DeepMind Control Suite
  - Exhibit state-of-the-art performance using the same hyperparameters for every task

Proposed Approach



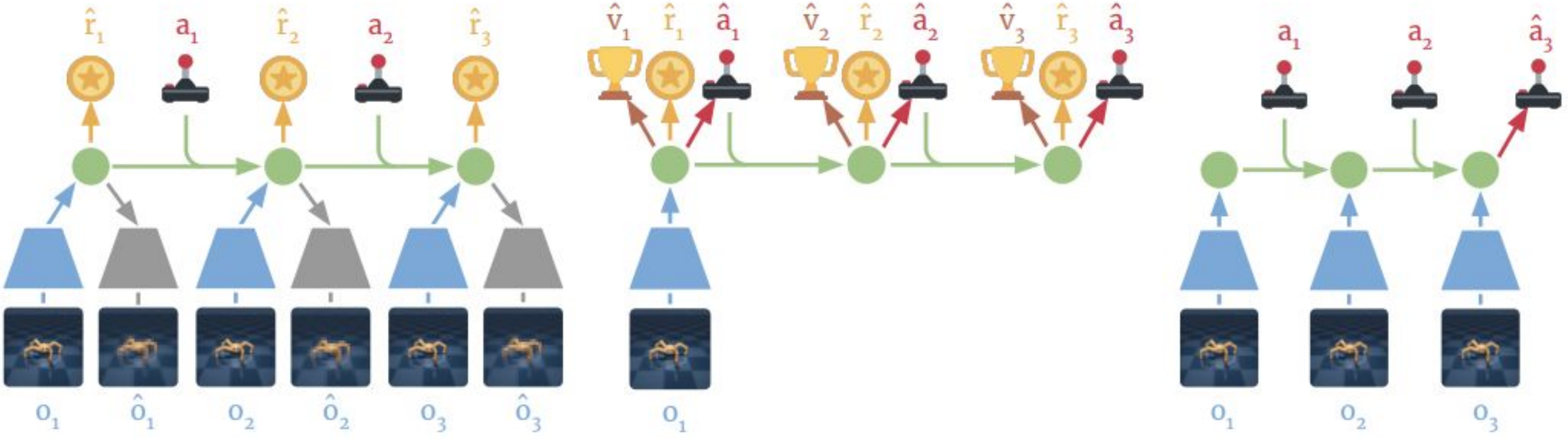
(a) Learn dynamics from experience

Proposed Approach



(a) Learn dynamics from experience (b) Learn behavior in imagination

Proposed Approach



(a) Learn dynamics from experience (b) Learn behavior in imagination (c) Act in the environment

Algorithm, formally

Algorithm 1: Dreamer

Initialize dataset $\mathcal{D}$ with S random seed episodes.
Initialize neural network parameters θ, ϕ, ψ randomly.
while *not converged* **do**
  for *update step* $c = 1..C$ **do**
    `// Dynamics learning`
    Draw B data sequences $\{(a_t, o_t, r_t)\}_{t=k}^{k+L} \sim \mathcal{D}$.
    Compute model states $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$.
    Update θ using representation learning.
    `// Behavior learning`
    Imagine trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^{t+H}$ from each s_t.
    Predict rewards $\mathrm{E}\big(q_\theta(r_\tau \mid s_\tau)\big)$ and values $v_\psi(s_\tau)$.
    Compute value estimates $\mathrm{V}_\lambda(s_\tau)$ via Equation 6.
    Update $\phi \leftarrow \phi + \alpha \nabla_\phi \sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)$.
    Update $\psi \leftarrow \psi - \alpha \nabla_\psi \sum_{\tau=t}^{t+H} \frac{1}{2} \big\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau) \big\|^2$.
  `// Environment interaction`
  $o_1 \leftarrow$ `env.reset()`
  for *time step* $t = 1..T$ **do**
    Compute $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$ from history.
    Compute $a_t \sim q_\phi(a_t \mid s_t)$ with the action model.
    Add exploration noise to action.
    $r_t, o_{t+1} \leftarrow$ `env.step(`a_t`)`.
  Add experience to dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o_t, a_t, r_t)_{t=1}^{T}\}$.

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

Algorithm, formally

Algorithm 1: Dreamer

Initialize dataset $\mathcal{D}$ with S random seed episodes.
Initialize neural network parameters θ, ϕ, ψ randomly.
while *not converged* **do**
  for *update step* $c = 1..C$ **do**
    `// Dynamics learning`
    Draw B data sequences $\{(a_t, o_t, r_t)\}_{t=k}^{k+L} \sim \mathcal{D}$.
    Compute model states $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$.
    Update θ using representation learning.
    `// Behavior learning`
    Imagine trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^{t+H}$ from each s_t.
    Predict rewards $\mathrm{E}\big(q_\theta(r_\tau \mid s_\tau)\big)$ and values $v_\psi(s_\tau)$.
    Compute value estimates $\mathrm{V}_\lambda(s_\tau)$ via Equation 6.
    Update $\phi \leftarrow \phi + \alpha \nabla_\phi \sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)$.
    Update $\psi \leftarrow \psi - \alpha \nabla_\psi \sum_{\tau=t}^{t+H} \frac{1}{2} \big\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau) \big\|^2$.
  `// Environment interaction`
  $o_1 \leftarrow$ `env.reset()`
  for *time step* $t = 1..T$ **do**
    Compute $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$ from history.
    Compute $a_t \sim q_\phi(a_t \mid s_t)$ with the action model.
    Add exploration noise to action.
    $r_t, o_{t+1} \leftarrow$ `env.step(`a_t`)`.
  Add experience to dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o_t, a_t, r_t)_{t=1}^{T}\}$.

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

Algorithm, formally

Algorithm 1: Dreamer

Initialize dataset $\mathcal{D}$ with S random seed episodes.
Initialize neural network parameters θ, ϕ, ψ randomly.
while *not converged* **do**
  for *update step* $c = 1..C$ **do**
    `// Dynamics learning`
    Draw B data sequences $\{(a_t, o_t, r_t)\}_{t=k}^{k+L} \sim \mathcal{D}$.
    Compute model states $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$.
    Update θ using representation learning.
    `// Behavior learning`
    Imagine trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^{t+H}$ from each s_t.
    Predict rewards $\mathrm{E}\big(q_\theta(r_\tau \mid s_\tau)\big)$ and values $v_\psi(s_\tau)$.
    Compute value estimates $\mathrm{V}_\lambda(s_\tau)$ via Equation 6.
    Update $\phi \leftarrow \phi + \alpha \nabla_\phi \sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)$.
    Update $\psi \leftarrow \psi - \alpha \nabla_\psi \sum_{\tau=t}^{t+H} \frac{1}{2} \big\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau) \big\|^2$.
  `// Environment interaction`
  $o_1 \leftarrow$ `env.reset()`
  for *time step* $t = 1..T$ **do**
    Compute $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$ from history.
    Compute $a_t \sim q_\phi(a_t \mid s_t)$ with the action model.
    Add exploration noise to action.
    $r_t, o_{t+1} \leftarrow$ `env.step(`a_t`)`.
  Add experience to dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o_t, a_t, r_t)_{t=1}^{T}\}$.

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

Algorithm, formally

Algorithm 1: Dreamer

Initialize dataset $\mathcal{D}$ with S random seed episodes.
Initialize neural network parameters θ, ϕ, ψ randomly.
while *not converged* **do**
  for *update step* $c = 1..C$ **do**
    `// Dynamics learning`
    Draw B data sequences $\{(a_t, o_t, r_t)\}_{t=k}^{k+L} \sim \mathcal{D}$.
    Compute model states $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$.
    Update θ using representation learning.
    `// Behavior learning`
    Imagine trajectories $\{(s_\tau, a_\tau)\}_{\tau=t}^{t+H}$ from each s_t.
    Predict rewards $\mathrm{E}\big(q_\theta(r_\tau \mid s_\tau)\big)$ and values $v_\psi(s_\tau)$.
    Compute value estimates $\mathrm{V}_\lambda(s_\tau)$ via Equation 6.
    Update $\phi \leftarrow \phi + \alpha \nabla_\phi \sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)$.
    Update $\psi \leftarrow \psi - \alpha \nabla_\psi \sum_{\tau=t}^{t+H} \frac{1}{2} \big\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau) \big\|^2$.
  `// Environment interaction`
  $o_1 \leftarrow$ `env.reset()`
  for *time step* $t = 1..T$ **do**
    Compute $s_t \sim p_\theta(s_t \mid s_{t-1}, a_{t-1}, o_t)$ from history.
    Compute $a_t \sim q_\phi(a_t \mid s_t)$ with the action model.
    Add exploration noise to action.
    $r_t, o_{t+1} \leftarrow$ `env.step(`a_t`)`.
  Add experience to dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o_t, a_t, r_t)_{t=1}^{T}\}$.

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

Formulation for Value Estimates

$$
\begin{aligned}
\mathrm{V_R}(s_\tau) &\doteq \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{n=\tau}^{t+H} r_n\right), \\
\mathrm{V_N^k}(s_\tau) &\doteq \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{n=\tau}^{h-1} \gamma^{n-\tau} r_n + \gamma^{h-\tau} v_\psi(s_h)\right) \quad \text{with} \quad h = \min(\tau + k, t + H), \\
\mathrm{V}_\lambda(s_\tau) &\doteq (1-\lambda) \sum_{n=1}^{H-1} \lambda^{n-1} \mathrm{V_N^n}(s_\tau) + \lambda^{H-1} \mathrm{V_N^H}(s_\tau),
\end{aligned}
$$

Learning Objective

$$\max_{\phi} \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)\right), \qquad (7)$$

$$\min_{\psi} \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{\tau=t}^{t+H} \frac{1}{2}\left\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau)) \right\|^2\right). \qquad (8)$$

Learning Objective

$$\max_{\phi} \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)\right), \qquad (7)$$

$$\min_{\psi} \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{\tau=t}^{t+H} \frac{1}{2}\left\| v_\psi(s_\tau) - \mathrm{V}_\lambda(s_\tau)) \right\|^2\right). \qquad (8)$$

Value estimates depend on reward and value predictions…

Reward and value predictions depend on imagined states…

Imagined states depend on imagined actions…

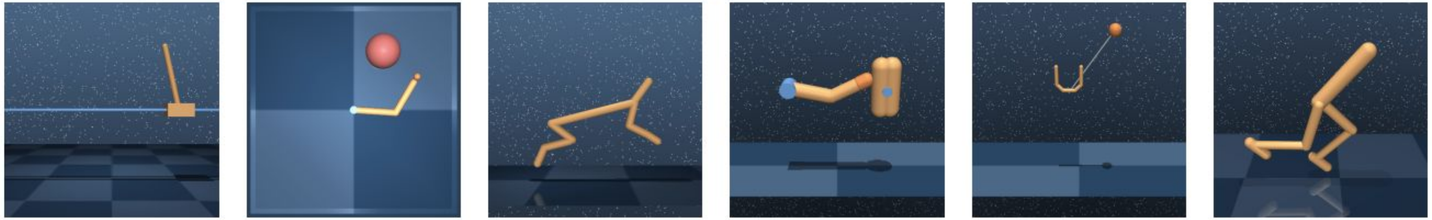
Model components

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	$v_\psi(s_t)$

We can use back propagation! $\nabla_\phi \mathrm{E}_{q_\theta, q_\phi}\left(\sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(s_\tau)\right)$

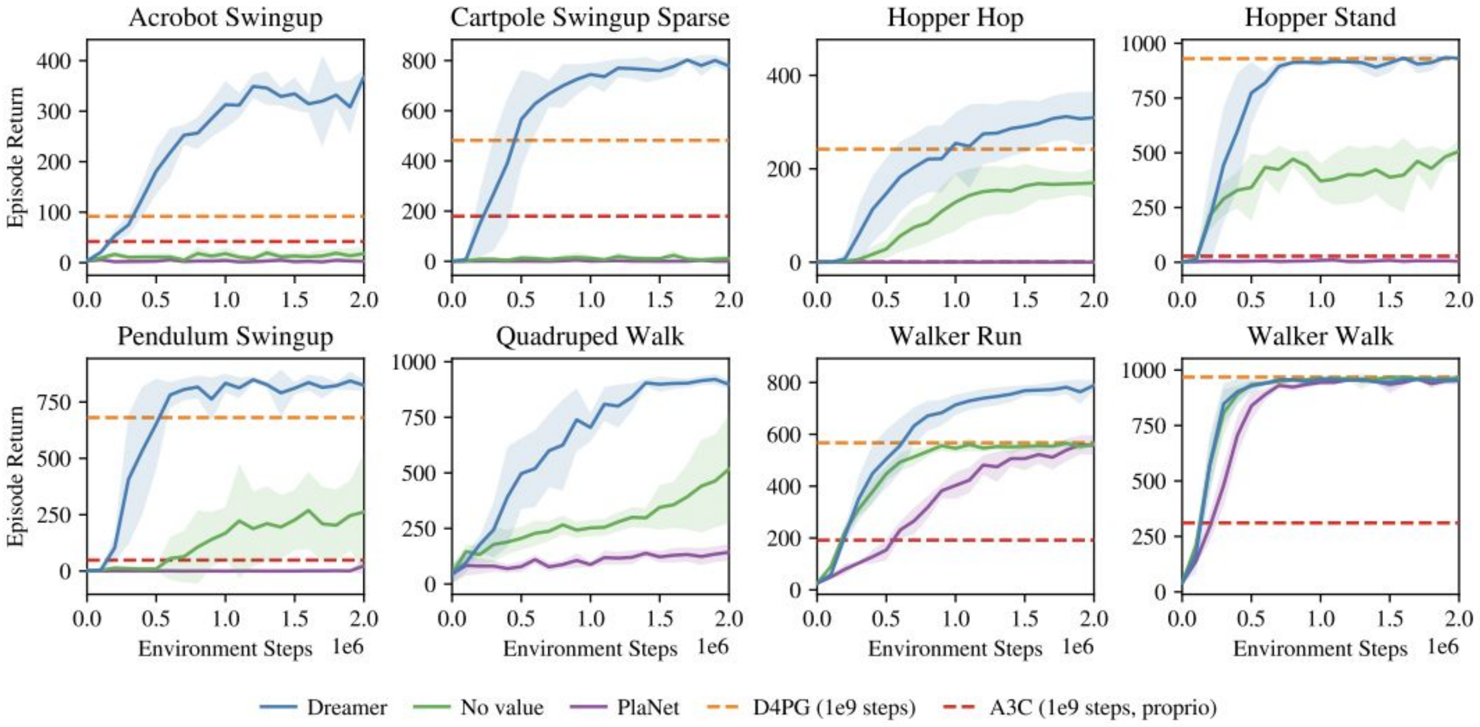
Experimental Setup

- Performance evaluated on visual control tasks in the DeepMind Control Suite
- Evaluated against:
  - PlaNet, previous latent imagination state-of-the-art
  - D4PG, top model-free agent
  - A3C, state-of-the-art actor-critic method

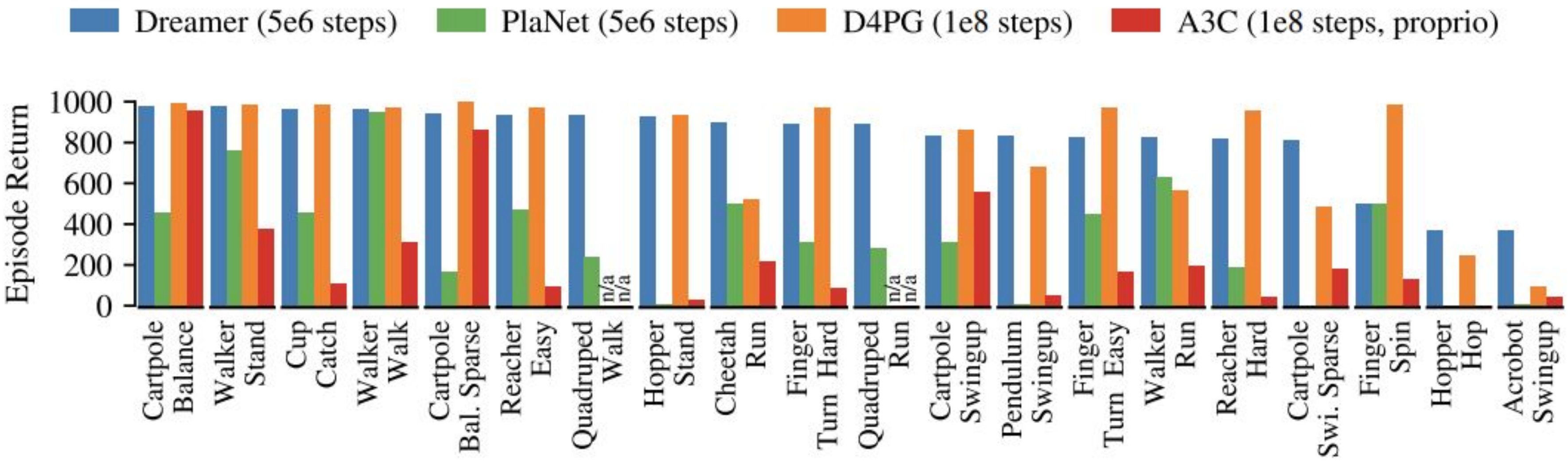


(a) Cartpole (b) Reacher (c) Cheetah (d) Finger (e) Cup (f) Walker

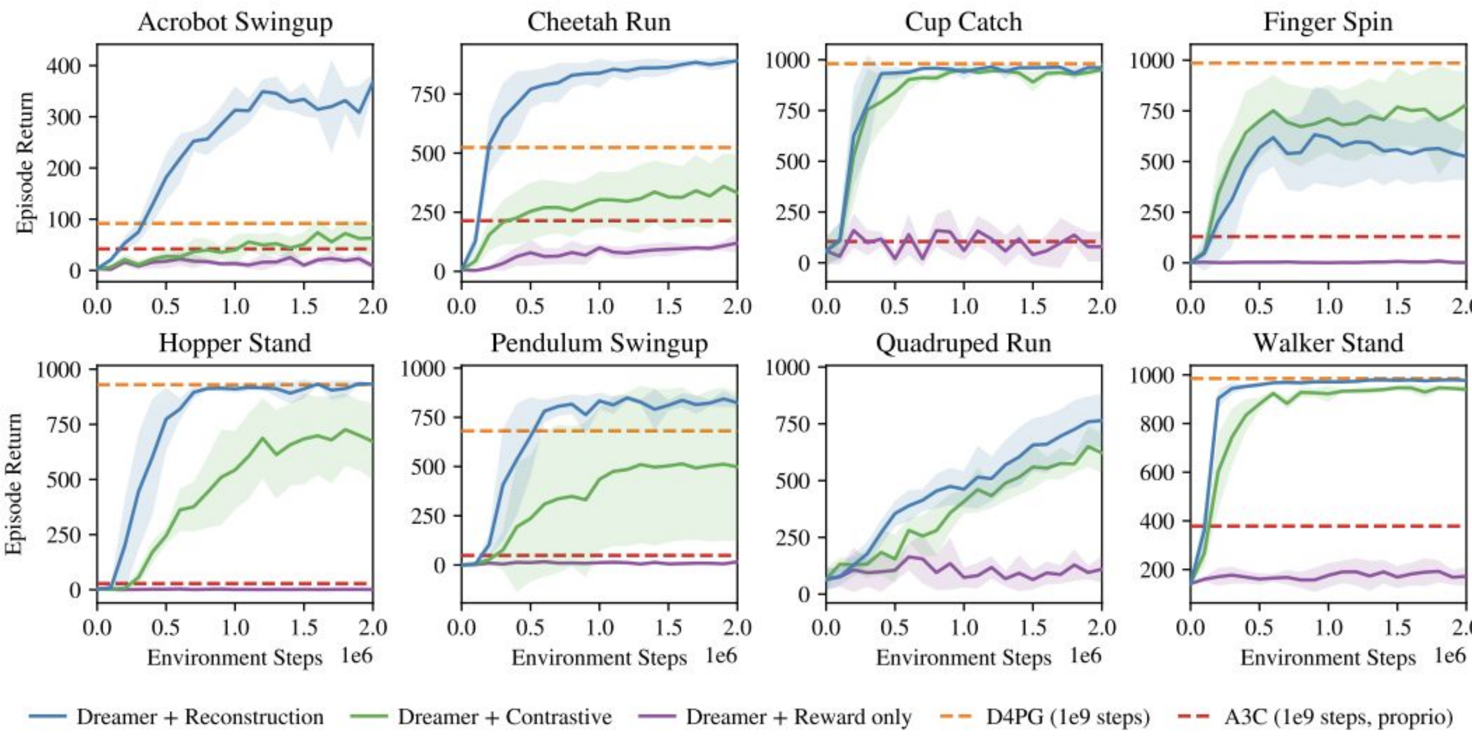
Experimental Results



Experimental Results



Experimental Results



Discussion of Results

- Demonstrate that Dreamer is able to be as efficient as PlaNet while matching or even outperforming state-of-the-art model-free agents
- Show that Dreamer is able to learn long-horizon behaviors from beyond the horizon, which outperforms more short-sighted approaches
- Performance of Dreamer is affected by the method of representation learning used
    - Better representation learning performance = Better Dreamer performance

Critique / Limitations / Open Issues

- **Ability to successfully utilize latent imagination depends on strength of representation learner**
  - Limits the breadth of tasks that this can be applied to rather than traditional reinforcement learning
- **Different Value estimation functions are not evaluated (besides the trivial one)**
  - To what extent can we improve on this equation, leading to faster learning?
  - This is the main insight of the paper, yet doesn't get very much discussion time

Future Work

- Learn more complex visual tasks with sparse rewards (e.g. Atari games, addressed by DreamerV2)
- Apply latent imagination to more input modalities, potentially getting us closer to real-world uses
- Could we experiment with different, more specialized representation learning approaches to perform more task-specific learning through imagination?

Extended Readings

- World Models
- Learning Latent Dynamics for Planning With Pixels (PlaNet)
- Dream to Explore: Adaptive Simulations for Autonomous Systems
- Mastering Atari with Discrete World Models (DreamerV2)

Summary

- Reinforcement learning traditionally involves many interactions with the environment

Summary

- Reinforcement learning traditionally involves many interactions with the environment
- Environment interactions can be computationally expensive

Summary

- Reinforcement learning traditionally involves many interactions with the environment
- Environment interactions can be computationally expensive
- We can instead train in a latent space, limiting need for interactions with the environment

Summary

- Reinforcement learning traditionally involves many interactions with the environment
- Environment interactions can be computationally expensive
- We can instead train in a latent space, limiting need for interactions with the environment
- Prior works used a fixed imagination horizon (short-sighted behaviors) and had to use derivative-free optimization

Summary

- Reinforcement learning traditionally involves many interactions with the environment
- Environment interactions can be computationally expensive
- We can instead train in a latent space, limiting need for interactions with the environment
- Prior works used a fixed imagination horizon (short-sighted behaviors) and had to use derivative-free optimization
- By computing an accurate value estimation, we can perform back-propagation

Summary

- Reinforcement learning traditionally involves many interactions with the environment
- Environment interactions can be computationally expensive
- We can instead train in a latent space, limiting need for interactions with the environment
- Prior works used a fixed imagination horizon (short-sighted behaviors) and had to use derivative-free optimization
- By computing an accurate value estimation, we can perform back-propagation
- Achieved state-of-the-art data efficiency, computational time, and performance

Questions For Discussion (slide hidden)

- So far, all the readings I have seen in this area have either been in environments for computer games (Tetris, Atari games, Doom, etc) or in task simulators (e.g DeepMind Control Suite). How can we apply these concepts towards learning to walk on a real robot? Would doing so reveal weaknesses of the approach?
- While Dreamer seems to perform remarkably well on most tasks in the DeepMind control suite, it really struggles on the “finger spin” task. Why is this? Could understanding this issue provide insight on limitations of the approach?
- More of an abstract question, but many times in machine learning we attempt to make artificial intelligence systems that model human behaviors. Is this “learning through imagination” idea something humans frequently do? If not, could we learn something from this different approach ourselves, perhaps to be better mentally prepared for upcoming challenges?