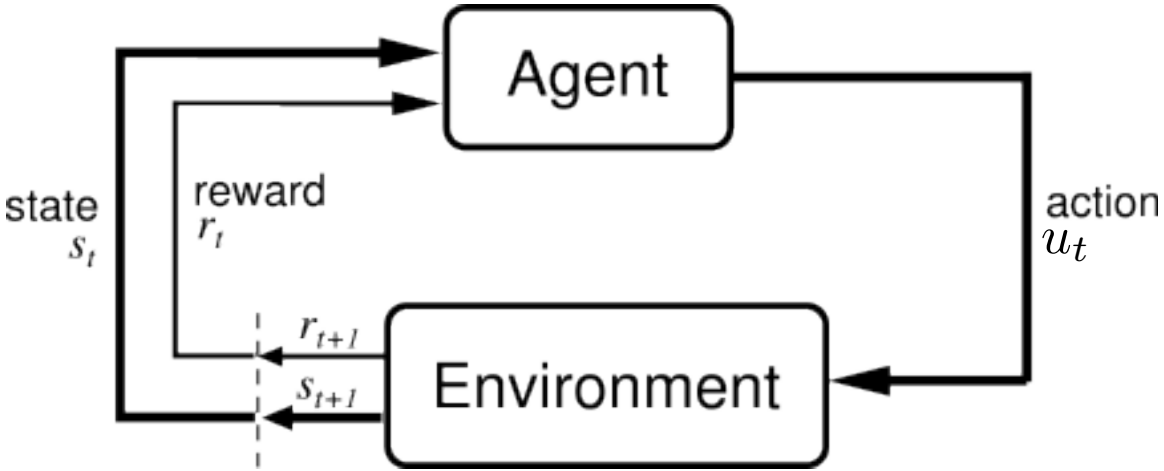
CS 287 Lecture 20 (Fall 2019)
Model-based RL

Pieter Abbeel
UC Berkeley EECS

Outline

- Model-based RL
- Ensemble Methods
  - Model-Ensemble Trust Region Policy Optimization
  - Model-based RL via Meta Policy Optimization
- Asynchronous Model-based RL
- Vision-based Model-based RL

Reinforcement Learning



[Figure source: Sutton & Barto, 1998]

“Algorithm”: Model-Based RL

- For iter = 1, 2, …
  - Collect data under current policy
  - Learn dynamics model from past data
  - Improve policy by using dynamics model
    - e.g SVG(k) requires dynamics model, but can also run TRPO/A3C in simulator

Why Model-Based RL?

- Anticipate data-efficiency
  - Get model out of data, which might allow for more significant policy updates than just a policy gradient
- Learning a model
  - Re-usable for other tasks [assuming general enough]

“Algorithm”: Model-Based RL

for iter = 1, 2, …

- Collect data under current policy
- Learn dynamics model from past data
- Improve policy by using dynamics model

Anticipated benefit?
– much better sample efficiency

So why not used all the time?
-- training instability → ME-TRPO
-- not achieving same asymptotic performance as model-free methods → MB-MPO

Overfitting in Model-based RL

- Standard overfitting (in supervised learning)
  - Neural network performs well on training data, but poorly on test data
    - E.g. on prediction of s_next from (s, a)
- New overfitting challenge in Model-based RL
  - policy optimization tends to exploit regions where insufficient data is available to train the model, leading to catastrophic failures
  - = “model-bias” (Deisenroth & Rasmussen, 2011; Schneider, 1997; Atkeson & Santamaria, 1997)
  - Proposed fix: Model-Ensemble Trust Region Policy Optimization (ME-TRPO)

Model-Ensemble Trust-Region Policy Optimization

Algorithm 1 Vanilla Model-Based Deep Reinforcement Learning

Initialize a policy π_θ and a model $\hat{f}_\phi$.
Initialize an empty dataset D.
repeat
  Collect samples from the real environment f using π_θ and add them to D.
  Train the model $\hat{f}_\phi$ using D.
  repeat
    Collect fictitious samples from $\hat{f}_\phi$ using π_θ.
    Update the policy using BPTT on the fictitious samples.
    Estimate the performance $\hat{\eta}(\theta; \phi)$.
  until the performance stop improving.
until the policy performs well in real environment f.

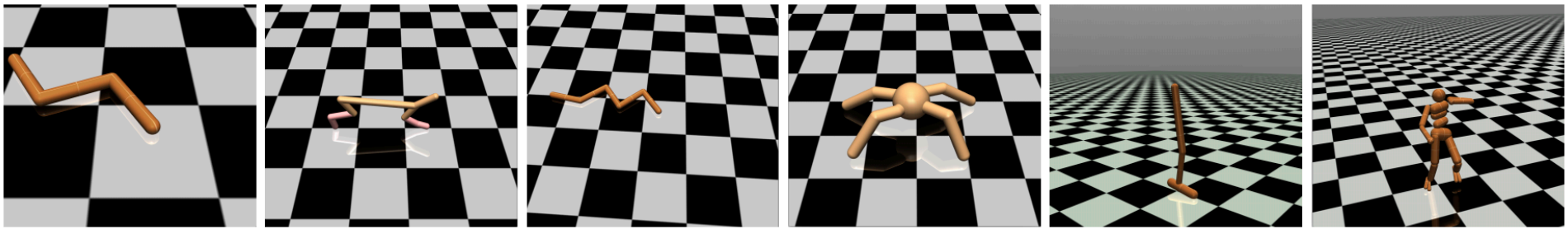
Algorithm 2 Model Ensemble Trust Region Policy Optimization (ME-TRPO)

Initialize a policy π_θ and all models $\hat{f}_{\phi_1}, \hat{f}_{\phi_2}, ..., \hat{f}_{\phi_K}$.
Initialize an empty dataset $\mathcal{D}$.
repeat
  Collect samples from the real system f using π_θ and add them to D.
  Train all models using $\mathcal{D}$.
  repeat ▷ Optimize π_θ using all models.
    Collect fictitious samples from $\{\hat{f}_{\phi_i}\}_{i=1}^{K}$ using π_θ.
    Update the policy using TRPO on the fictitious samples.
    Estimate the performances $\hat{\eta}(\theta; \phi_i)$ for $i = 1, ..., K$.
  until the performances stop improving.
until the policy performs well in real environment f.

[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO Evaluation

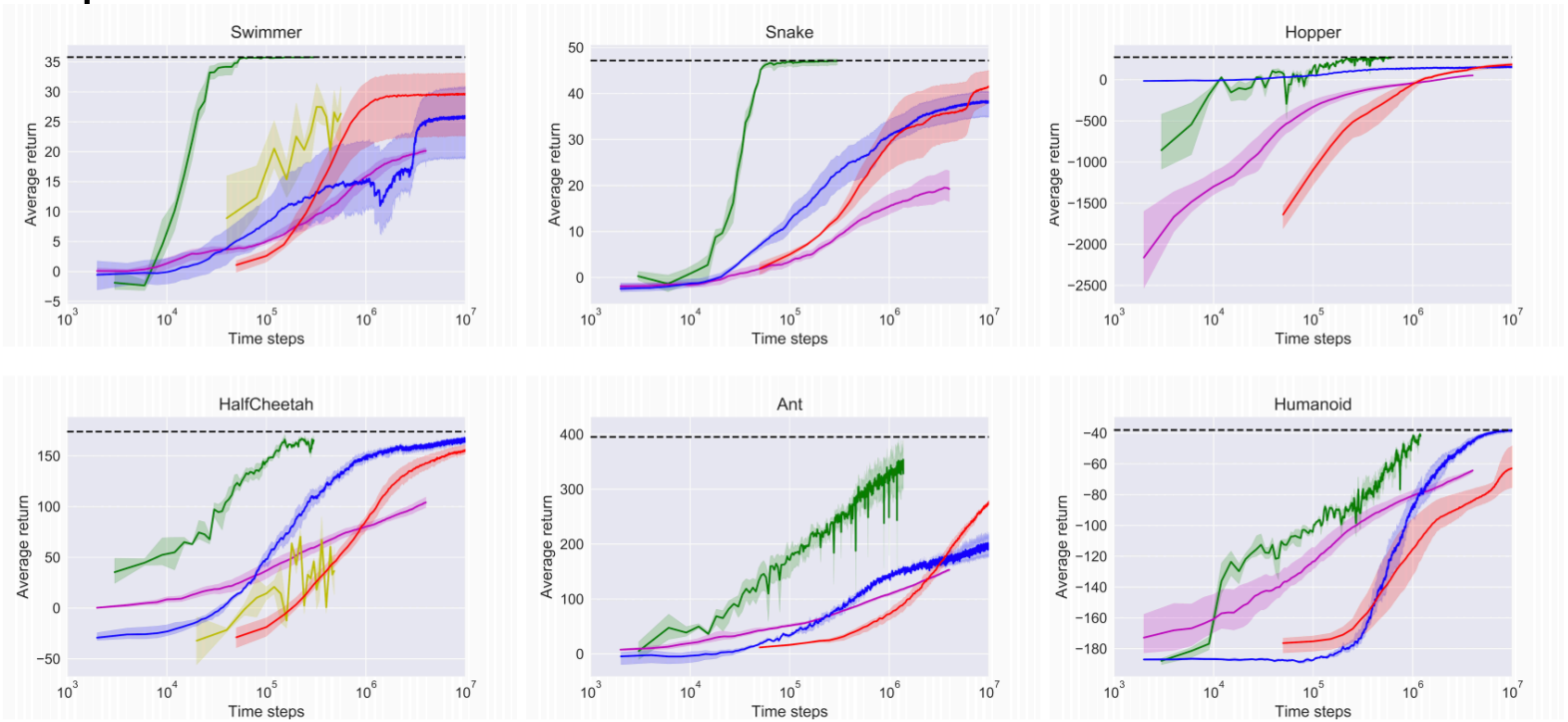
- Environments:



[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO Evaluation

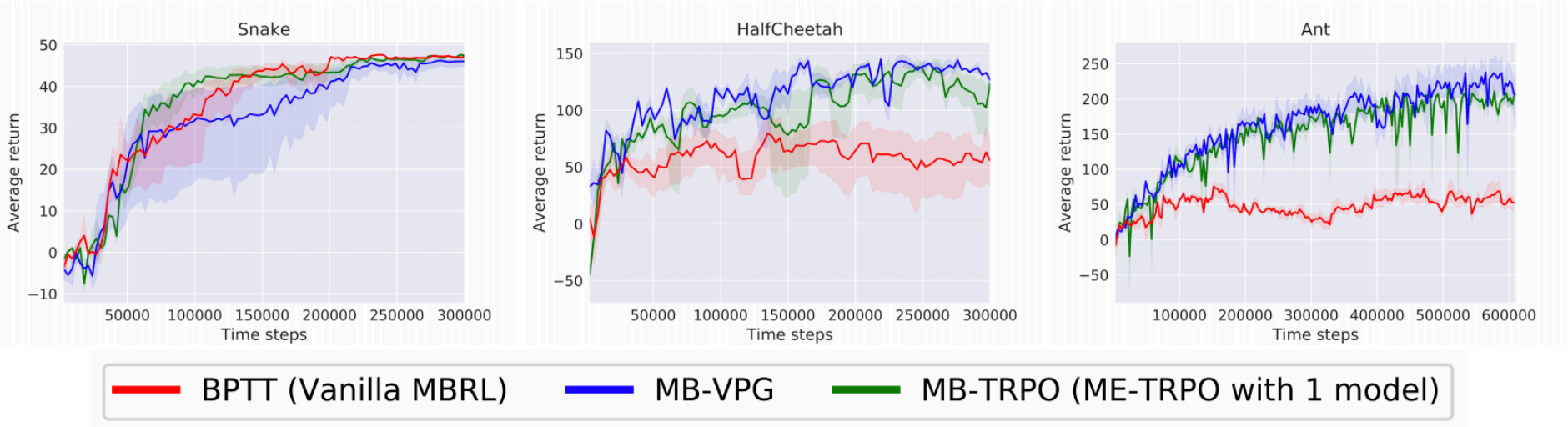
- Comparison with state of the art



[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO -- Ablation

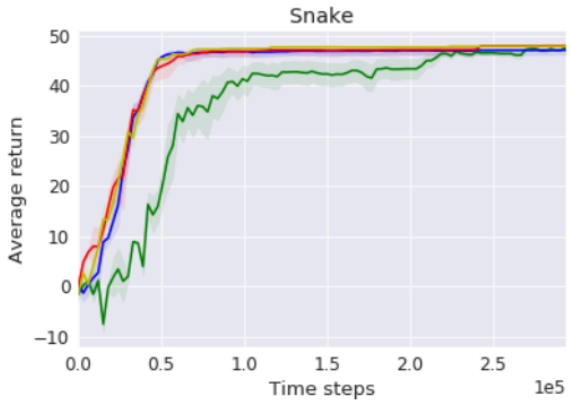
TRPO vs. BPTT in standard model-based RL

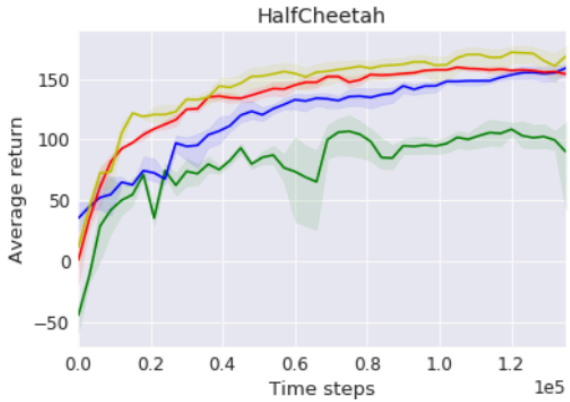


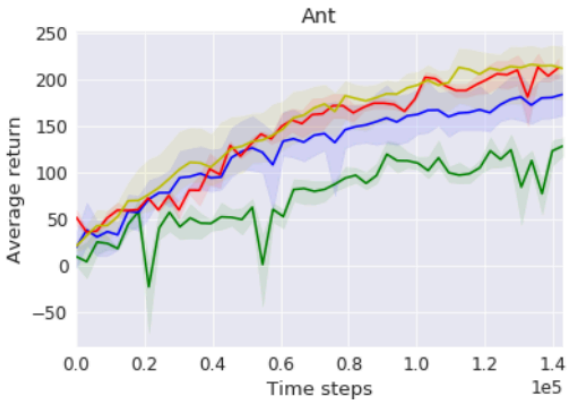
[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO -- Ablation

Number of learned dynamics models in the ensemble







ME-TRPO-1 ME-TRPO-5 ME-TRPO-10 ME-TRPO-20

[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

“Algorithm”: Model-Based RL

for iter = 1, 2, …

- Collect data under current policy
- Learn dynamics model from past data
- Improve policy by using dynamics model

Anticipated benefit?
– much better sample efficiency

So why not used all the time?
-- training instability → ME-TRPO
-- not achieving same asymptotic performance as model-free methods → MB-MPO

Model-based RL Asymptotic Performance

- Because learned (ensemble of) model imperfect
  - Resulting policy good in simulation(s), but not optimal in real world
- Attempted Fix 1: learn better dynamics model
  - Such efforts have so far proven insufficient
- Attempted Fix 2: model-based RL via meta-policy optimization (MB-MPO)
  - Key idea:
    - Learn ensemble of models representative of generally how the real world works
    - Learn an ***adaptive policy*** that can quickly adapt to any of the learned models
    - Such adaptive policy can quickly adapt to how the real world works

Model-Based RL via Meta Policy Optimization (MB-MPO)

for iter = 1, 2, …

- collect data under current adaptive policies $\pi_{\theta'_1}, \ldots, \pi_{\theta'_K}$
- learn **ENSEMBLE** of K simulators from all past data
- **meta-policy optimization over ENSEMBLE**
  - → **new meta-policy** π_θ
  - → **new adaptive policies** $\pi_{\theta'_1}, \ldots, \pi_{\theta'_K}$

[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

Model-based via Meta-Policy Optimization MB-MPO

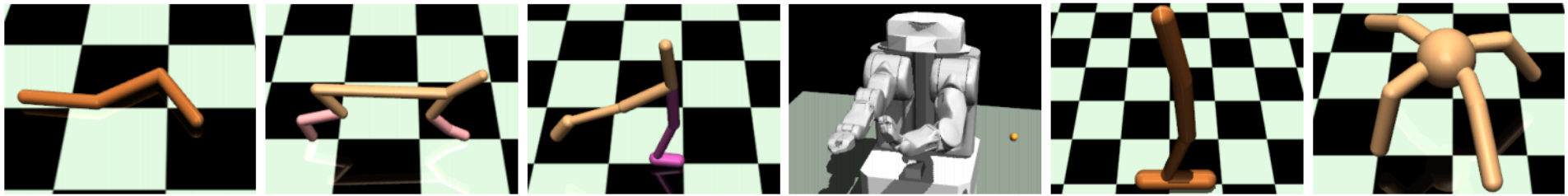
Algorithm 1 MB-MPO

Require: Inner and outer step size α, β

1: Initialize the policy $\pi_{\boldsymbol{\theta}}$, the models $\hat{f}_{\boldsymbol{\phi}_1}, \hat{f}_{\boldsymbol{\phi}_2}, ..., \hat{f}_{\boldsymbol{\phi}_K}$ and $\mathcal{D} \leftarrow \emptyset$
2: **repeat**
3: Sample trajectories from the real environment with the adapted policies $\pi_{\boldsymbol{\theta}'_1}, ..., \pi_{\boldsymbol{\theta}'_K}$. Add them to $\mathcal{D}$.
4: Train all models using $\mathcal{D}$.
5: **for all** models $\hat{f}_{\boldsymbol{\phi}_k}$ **do**
6: Sample imaginary trajectories $\mathcal{T}_k$ from $\hat{f}_{\boldsymbol{\phi}_k}$ using $\pi_{\boldsymbol{\theta}}$
7: Compute adapted parameters $\boldsymbol{\theta}'_k = \boldsymbol{\theta} + \alpha \nabla_{\boldsymbol{\theta}} J_k(\boldsymbol{\theta})$ using trajectories $\mathcal{T}_k$
8: Sample imaginary trajectories $\mathcal{T}'_k$ from $\hat{f}_{\boldsymbol{\phi}_k}$ using the adapted policy $\pi_{\boldsymbol{\theta}'_k}$
9: **end for**
10: Update $\boldsymbol{\theta} \rightarrow \boldsymbol{\theta} - \beta \frac{1}{K} \sum_k \nabla_{\boldsymbol{\theta}} J_k(\boldsymbol{\theta}'_k)$ using the trajectories $\mathcal{T}'_k$
11: **until** the policy performs well in the real environment
12: **return** Optimal pre-update parameters $\boldsymbol{\theta}^*$

[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation



[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

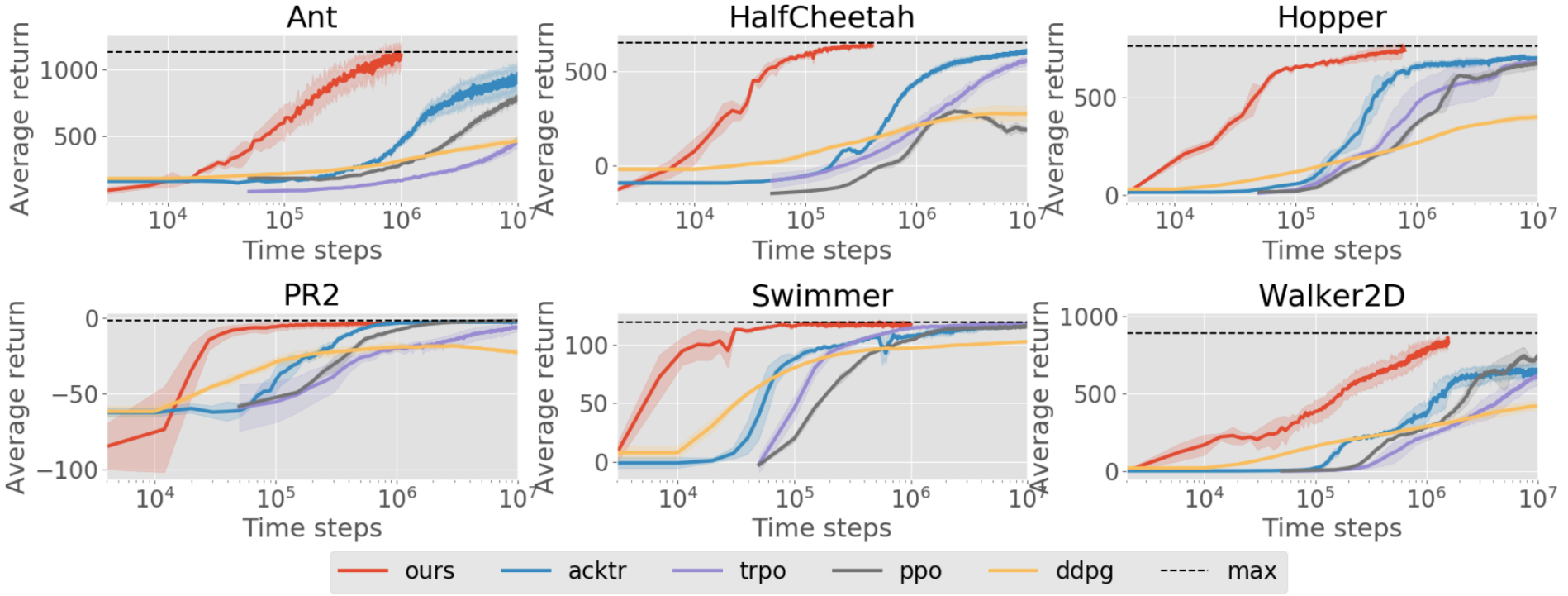
[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

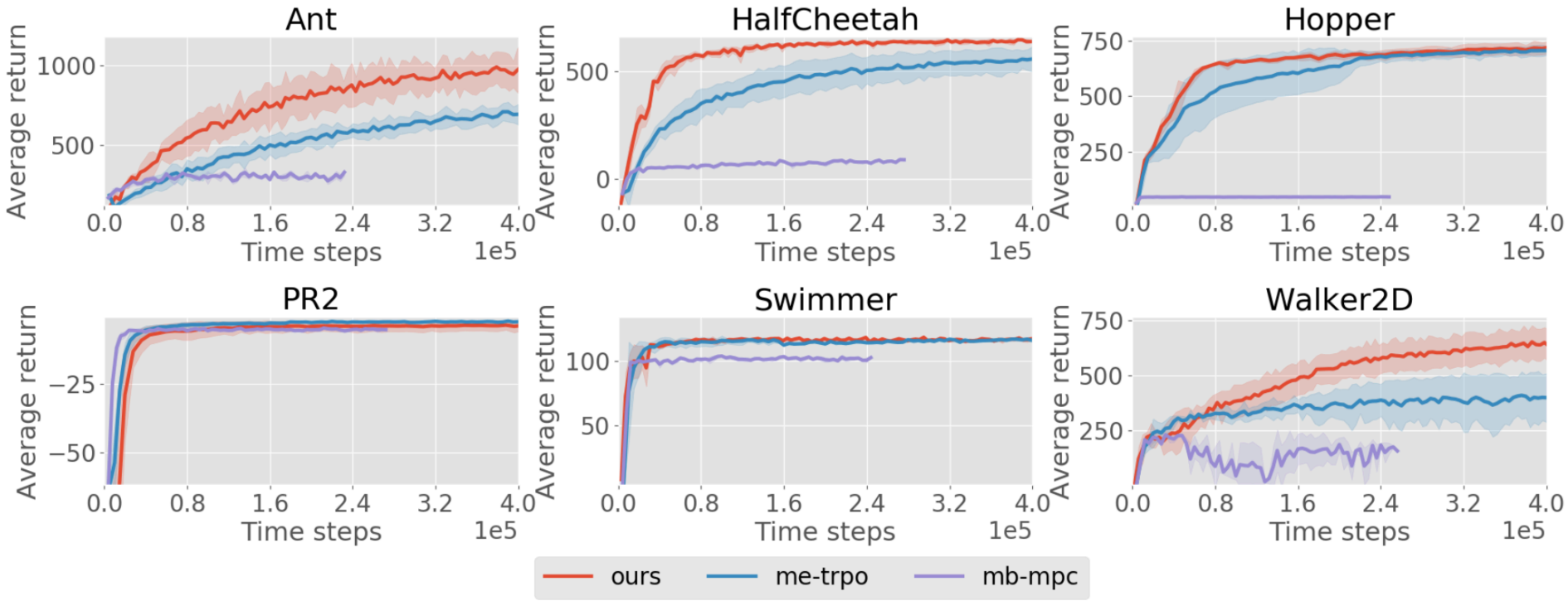
- Comparison with state of the art model-free



[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

- Comparison with state of the art model-based



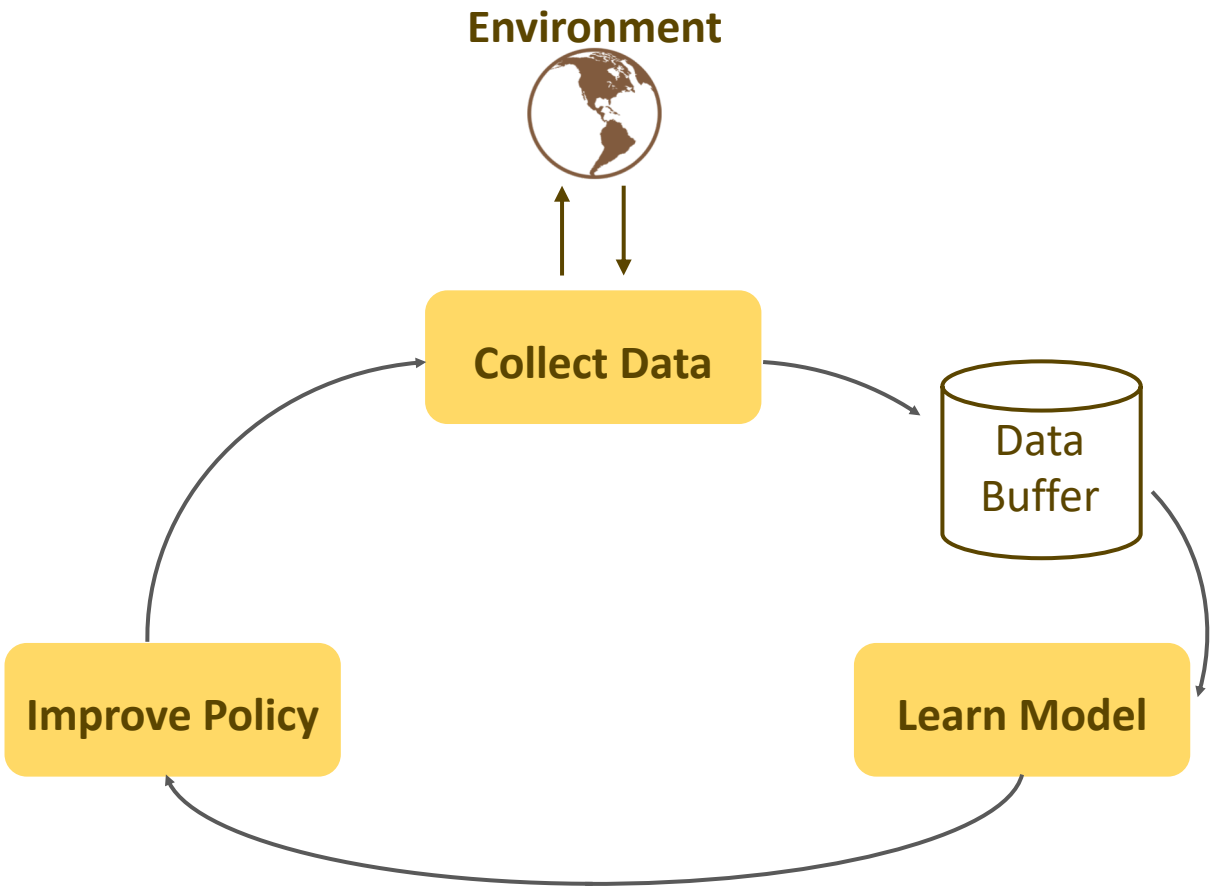
[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

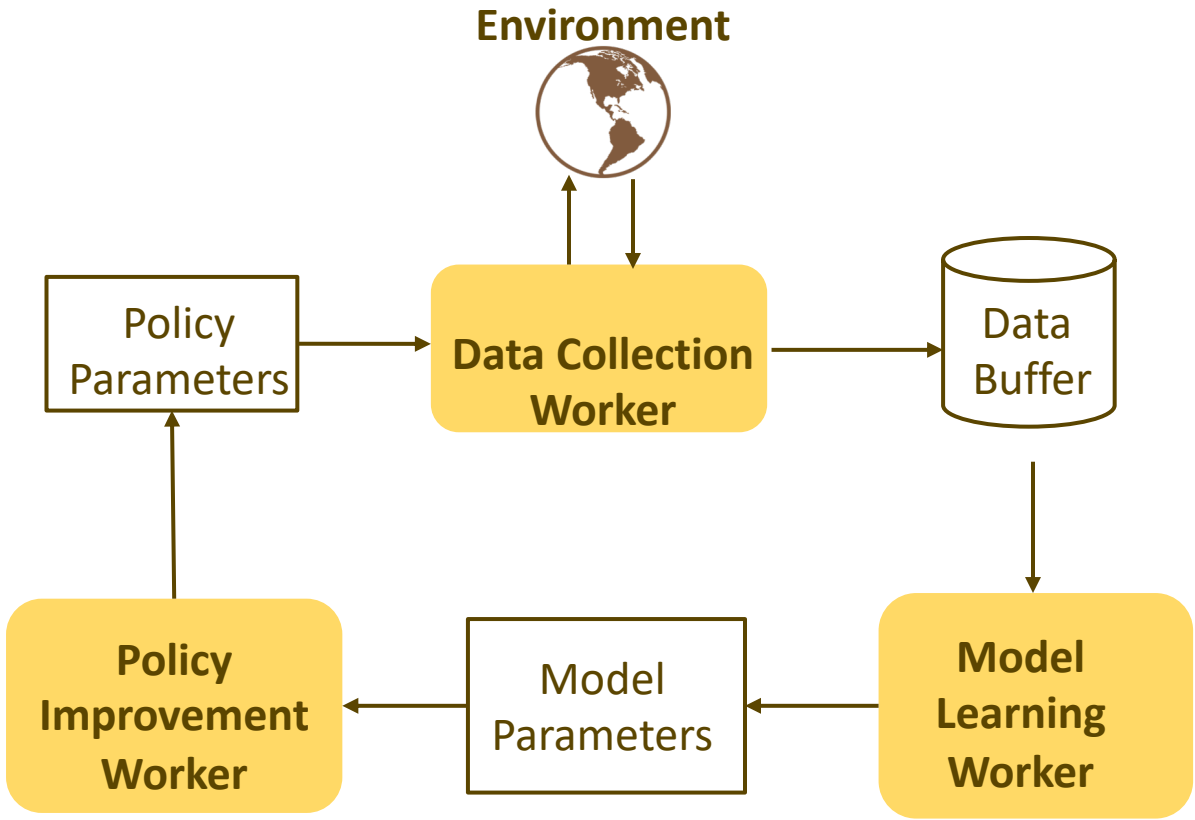
So are we done?

- No…
  - Not real-time --- exacerbated by need for extensive hyperparameter tuning
  - Limited to short horizon
  - From state (though some results have started to happen from images)

So are we done?

- No…
  - ***Not real-time --- exacerbated by need for extensive hyperparameter tuning***
  - Limited to short horizon
  - From state (though some results have started to happen from images)


Environment
Collect Data
Data
Buffer
Learn Model
Improve Policy


Environment
Policy Parameters
Data Collection Worker
Data Buffer
Policy Improvement Worker
Model Parameters
Model Learning Worker

Questions to be answered

1. Performance?

Questions to be answered

1. Performance?
2. Effect on policy regularization?

Questions to be answered

1. Performance?
2. Effect on policy regularization?
3. Effect on data exploration?

Questions to be answered

1. Performance?
2. Effect on policy regularization?
3. Effect on data exploration?
4. Robustness to hyperparameters?

Questions to be answered

1. Performance?
2. Effect on policy regularization?
3. Effect on data exploration?
4. Robustness to hyperparameters?
5. Robustness to data collection frequency?

Experiments

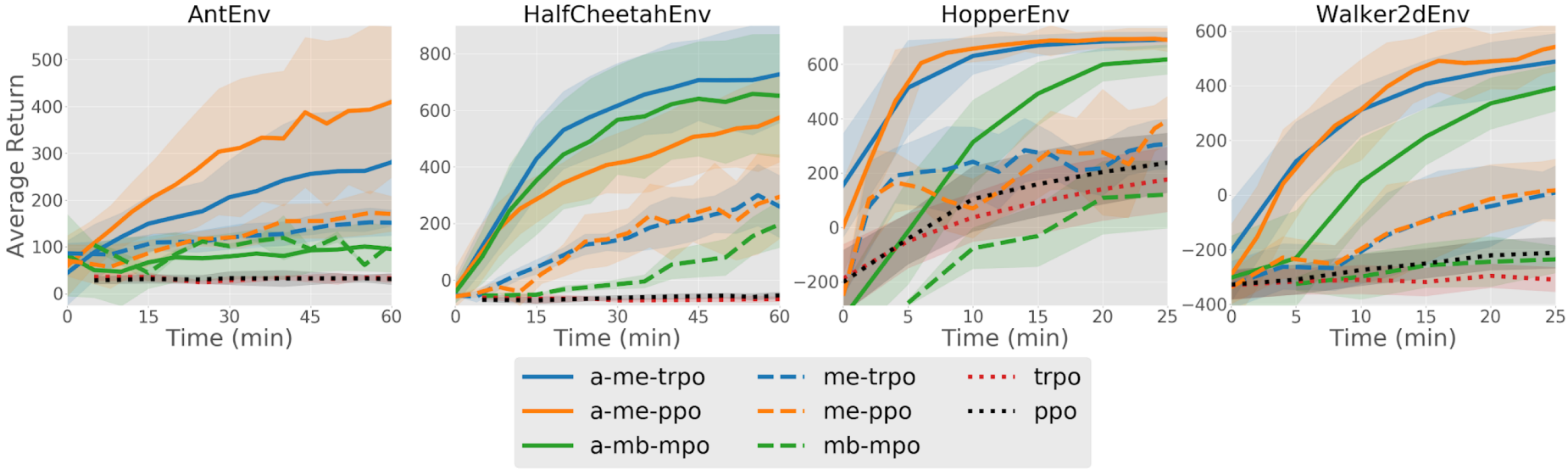
1. **How does the asynch-framework perform?**

   Asynch: ME-TRPO, ME-PPO, MB-MPO

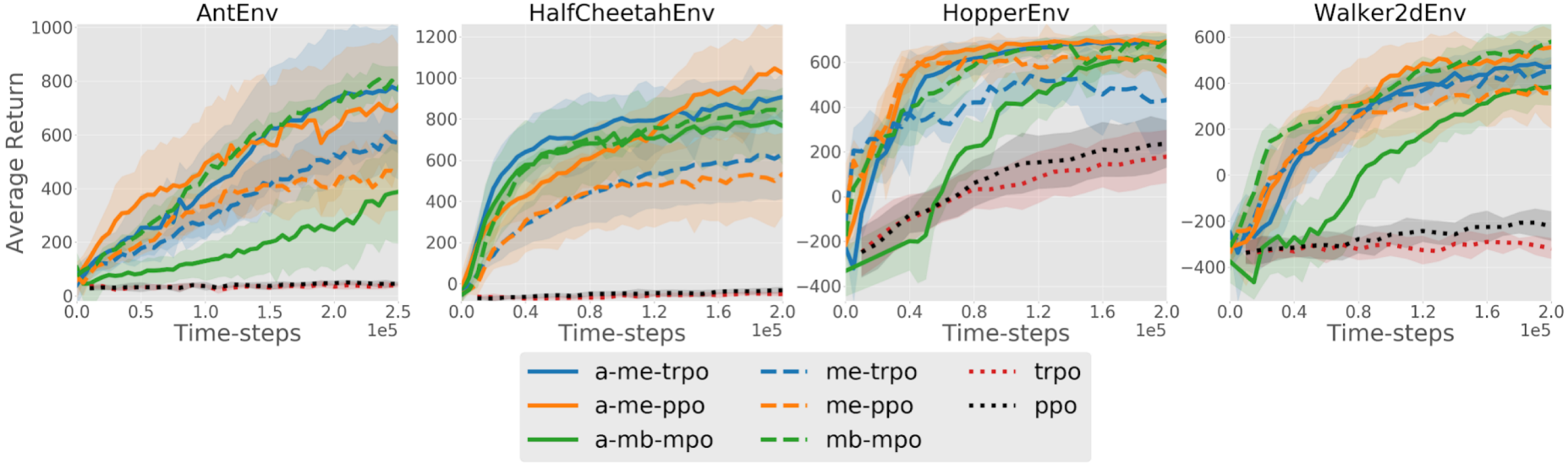
   Baselines: ME-TRPO, ME-PPO, MB-MPO; TRPO, PPO

   a. Average Return vs. Time
   b. Average Return vs. Sample complexity (Timesteps)

Performance Comparison: Wall-Clock Time



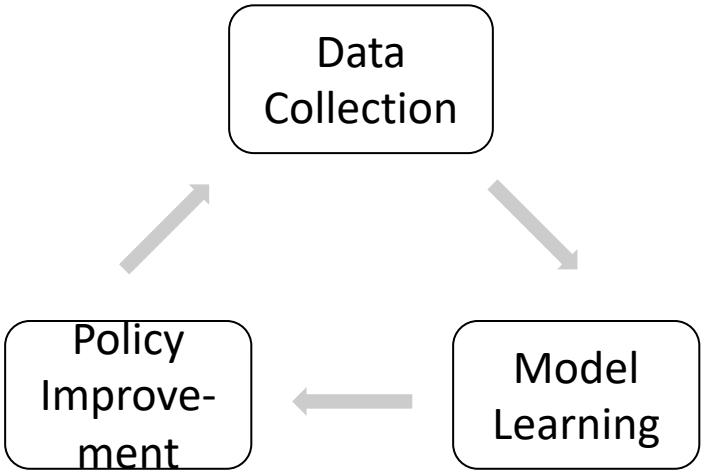
Performance Comparison: Sample Complexity



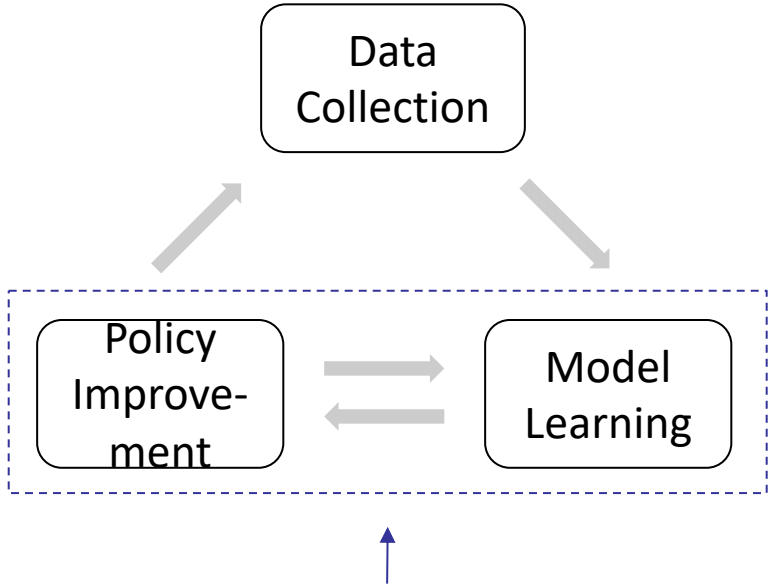
Experiments

1. Performance comparison
2. **Are there benefits of being asynchronous other than speed?**
    a. Policy learning regularization
    b. Exploration in data collection

Policy Learning Regularization

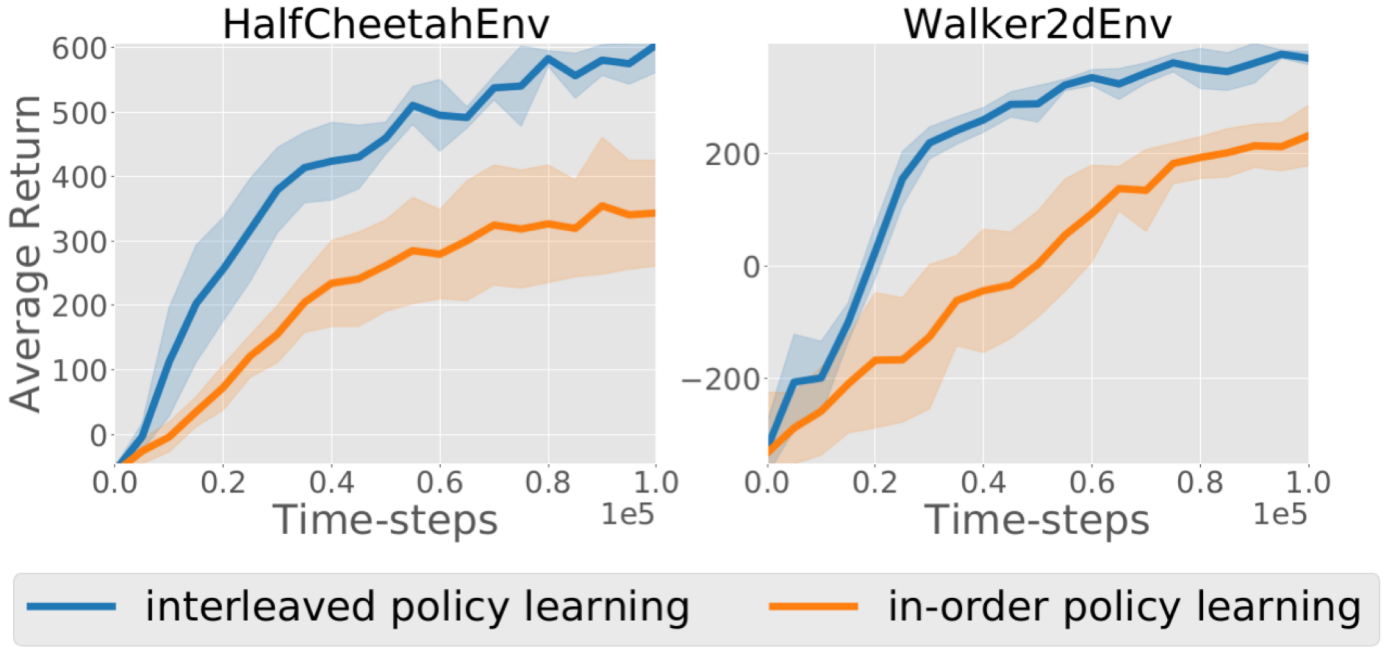


Synchronous

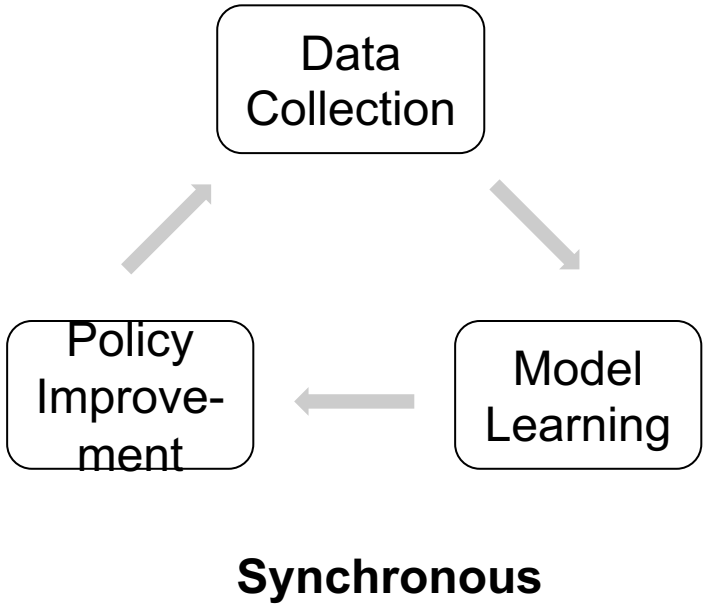


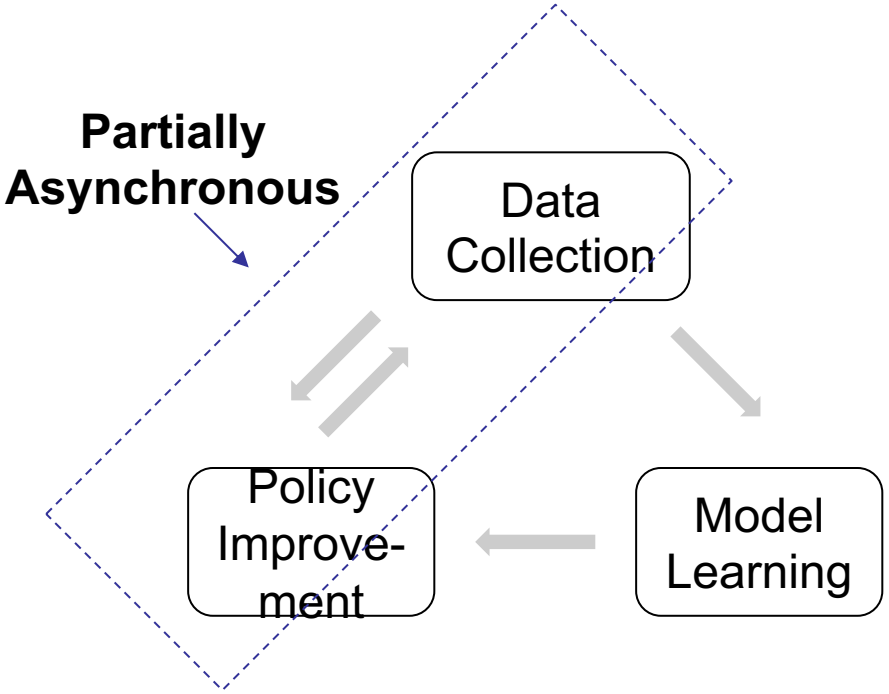
Partially Asynchronous

Policy Learning Regularization

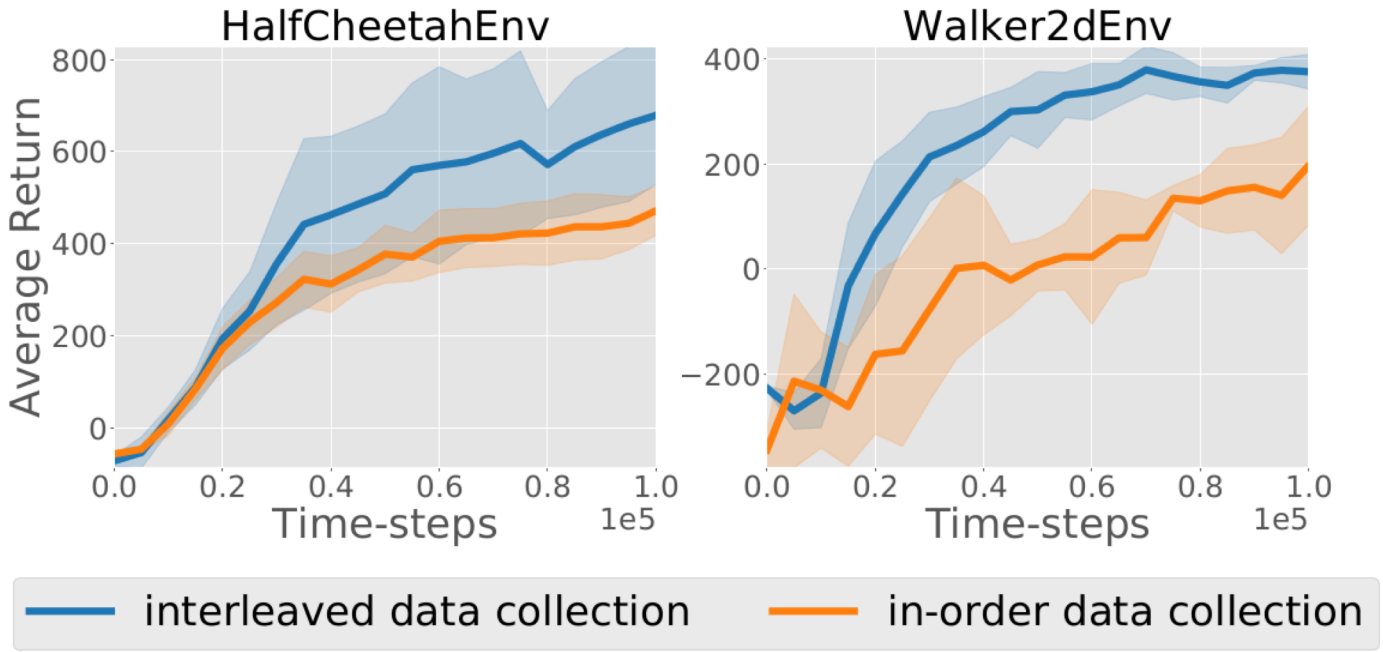


Improved Exploration for Data Collection





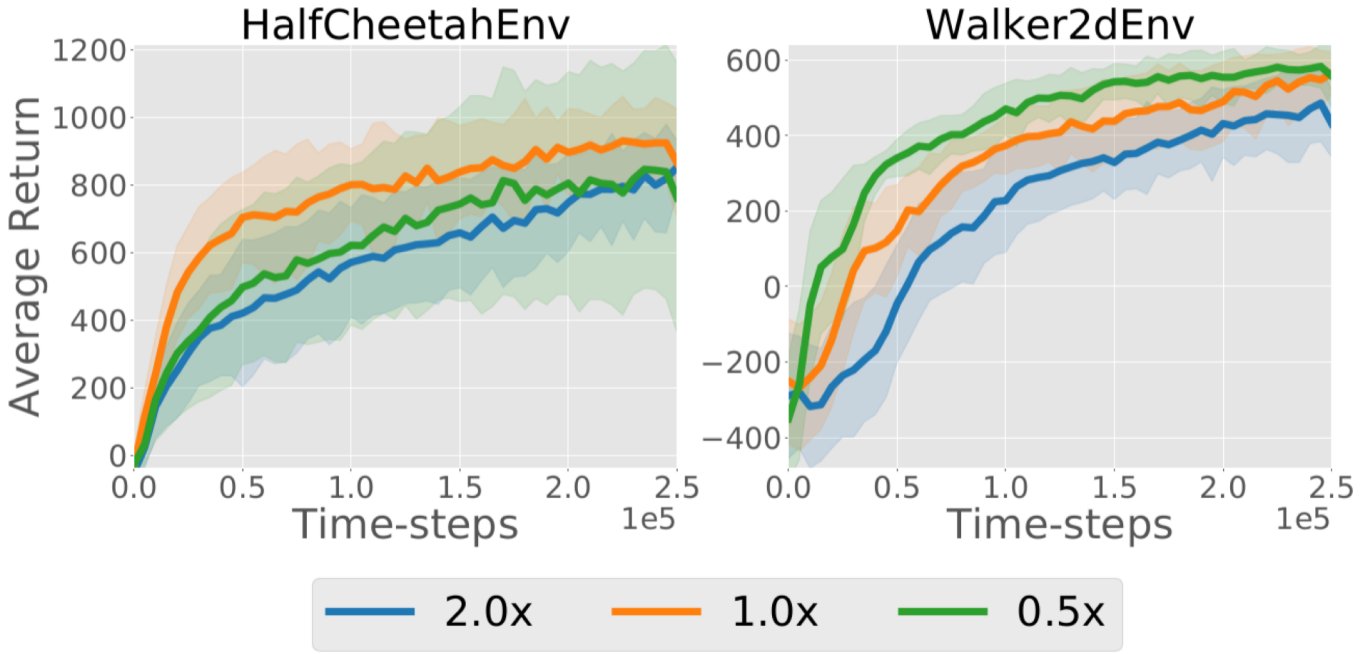
Improved Exploration for Data Collection



Experiments

1. Performance comparison
2. Asynchronous effects
3. **Is the asynch-framework robust to data collection frequency?**

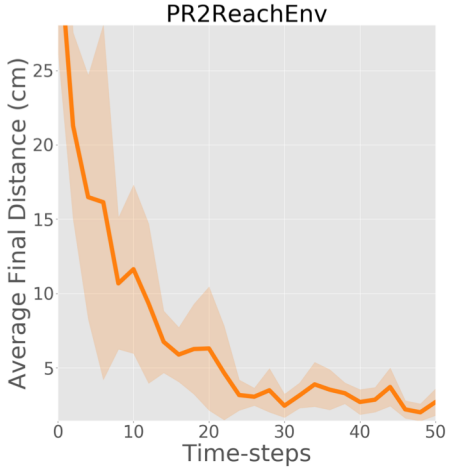
Ablations: Sampling Speed

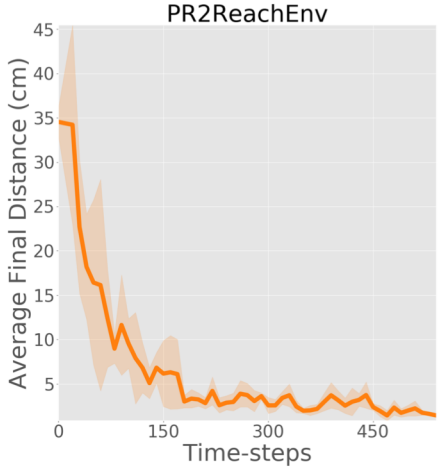


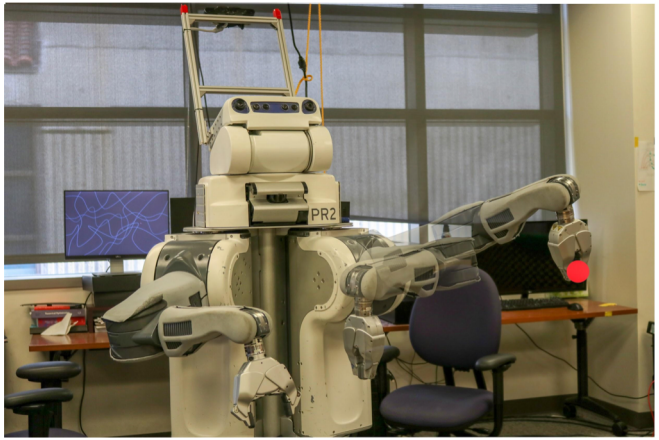
Experiments

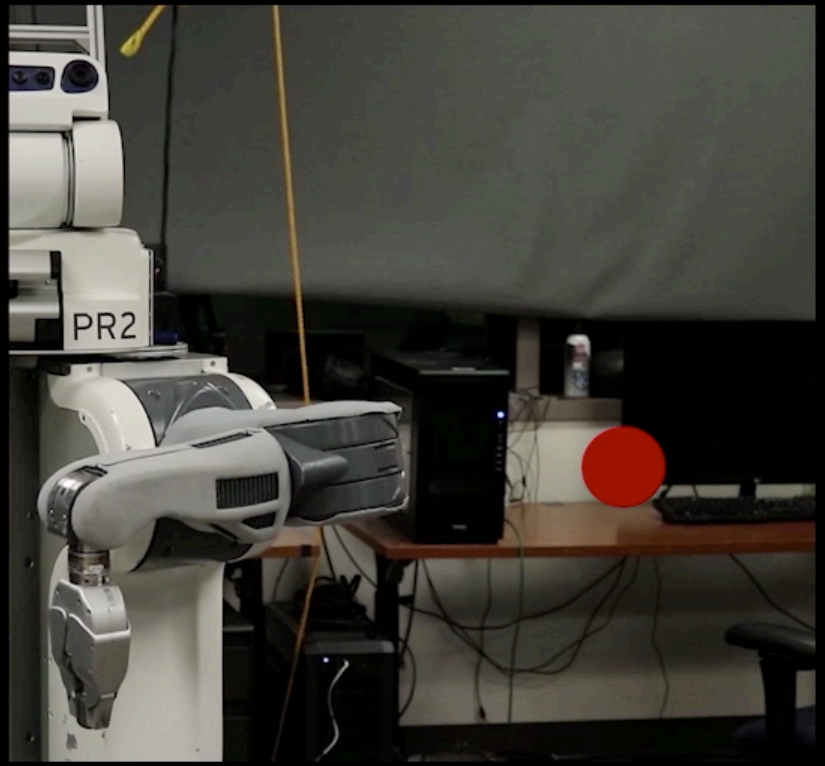
1. Performance comparison
2. Asynchronous effects
3. Ablations
4. **Does the aynch-framework work in real robotics tasks?**
    a. Reaching a position
    b. Inserting a unique shape into its matching hole in a box
    c. Stacking a modular block onto a fixed base

Real Robot Tasks: Reaching Position

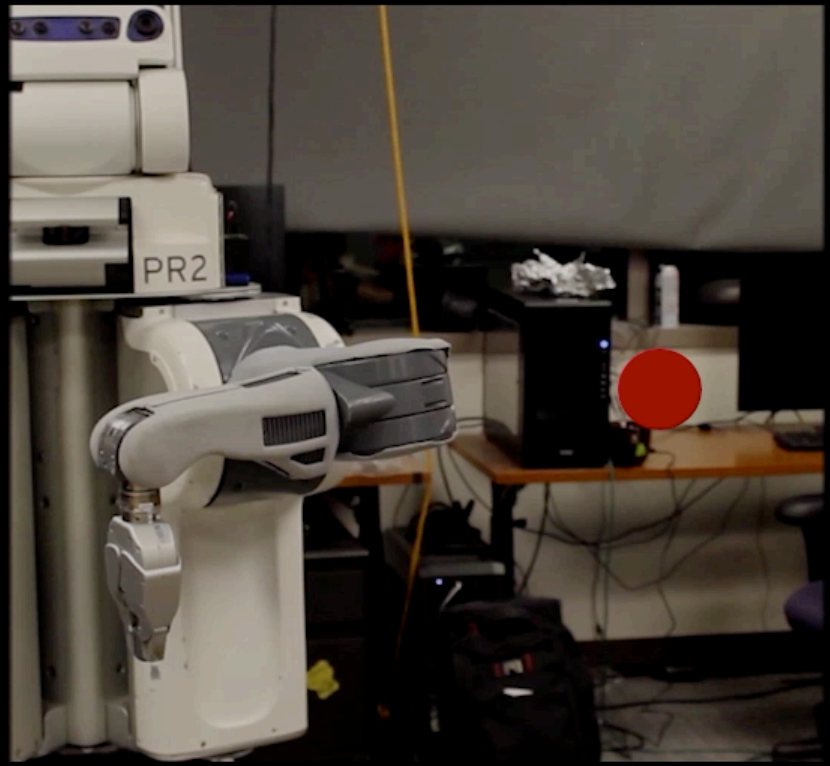








Synchronous

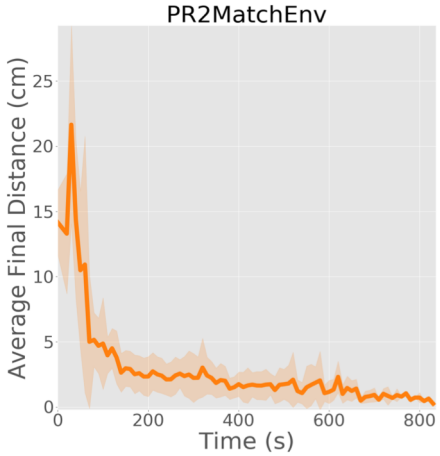


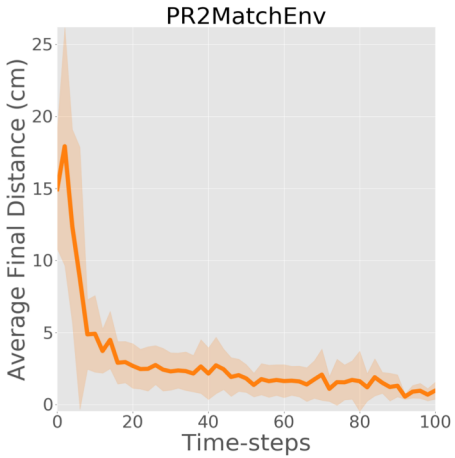
Asynchronous

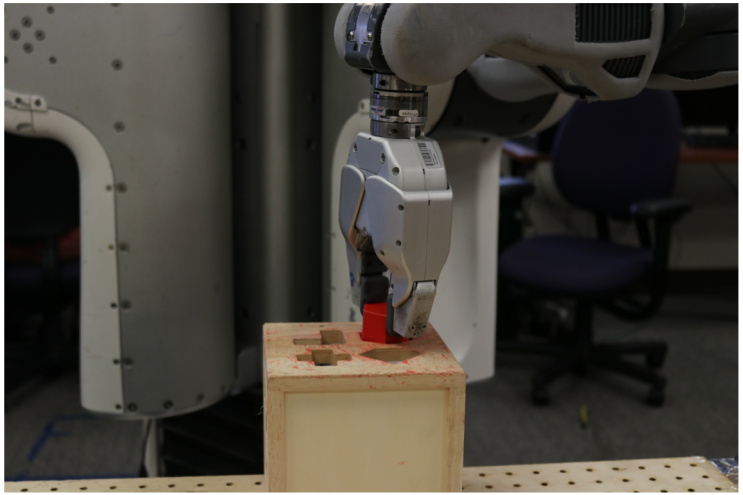
Wall Clock Time: 0 seconds

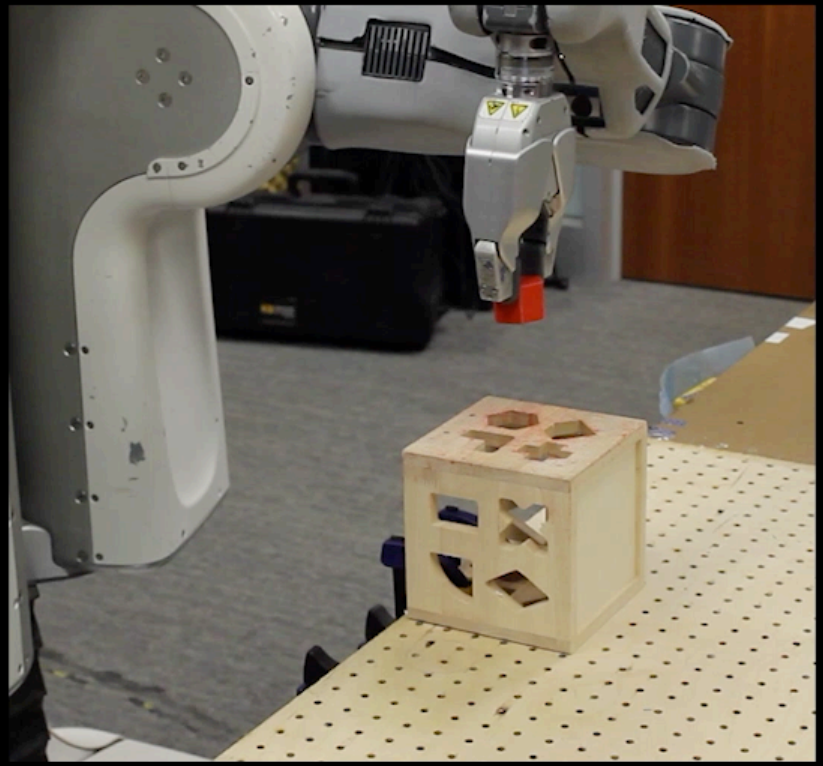
x2

Real Robot Tasks: Matching Shape

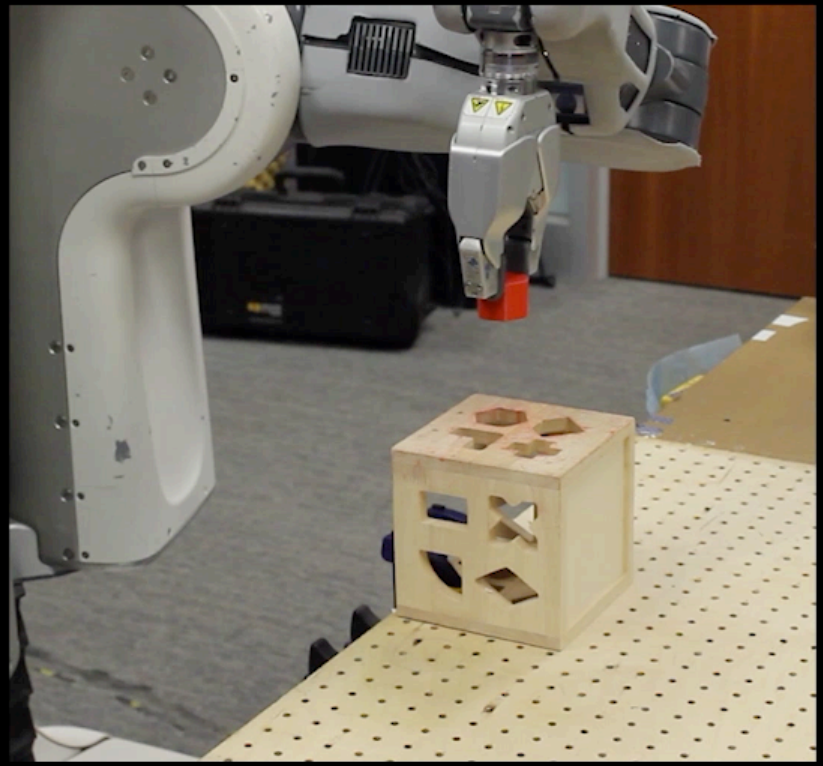








Synchronous

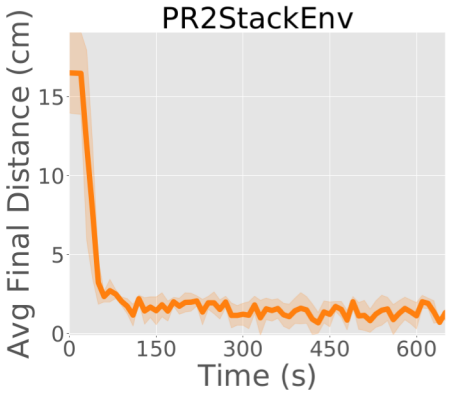


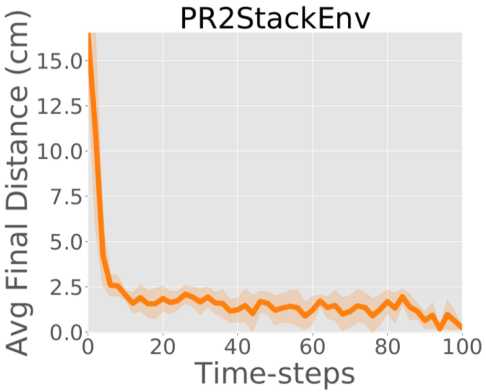
Asynchronous

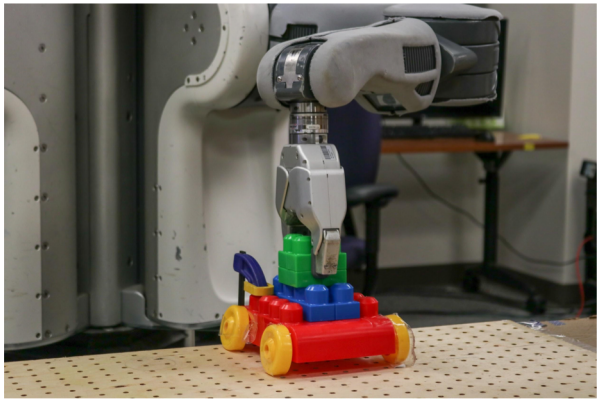
Wall Clock Time: 0 minutes

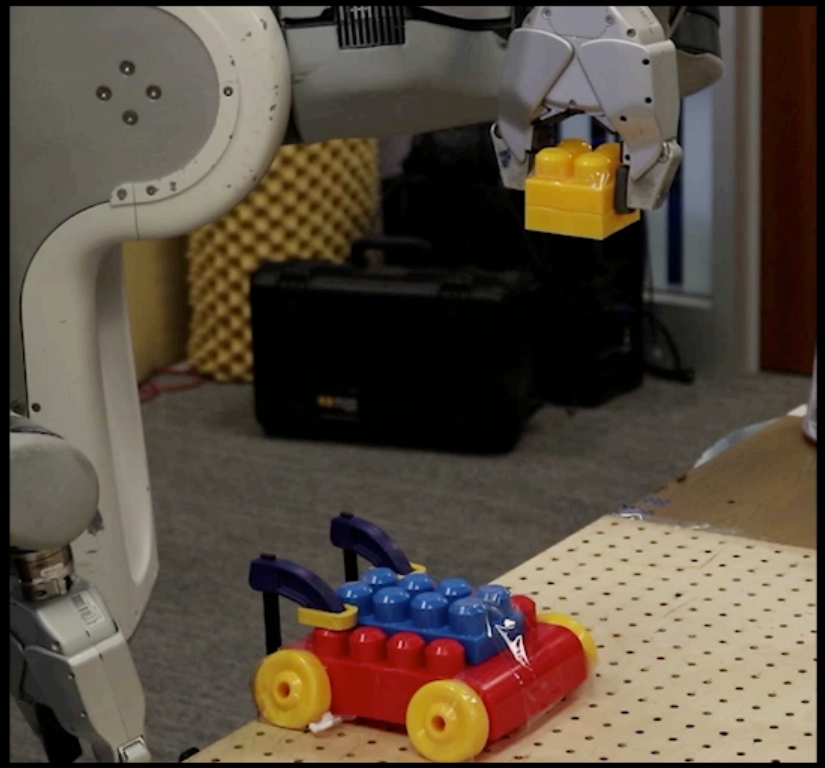
x2

Real Robot Tasks: Stacking Lego

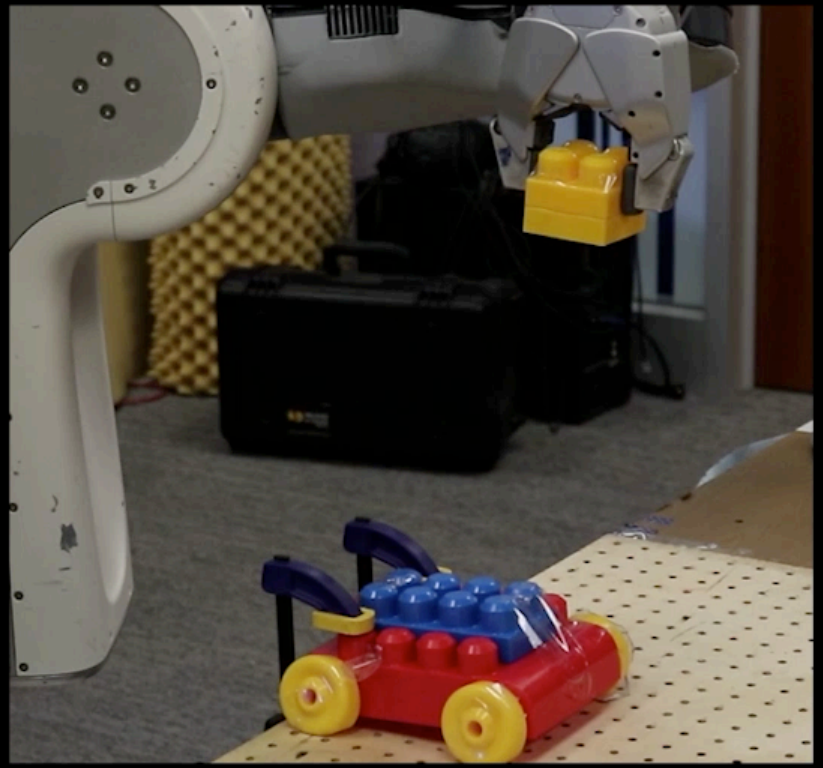








Synchronous



Asynchronous

Wall Clock Time: 0 minutes

x2

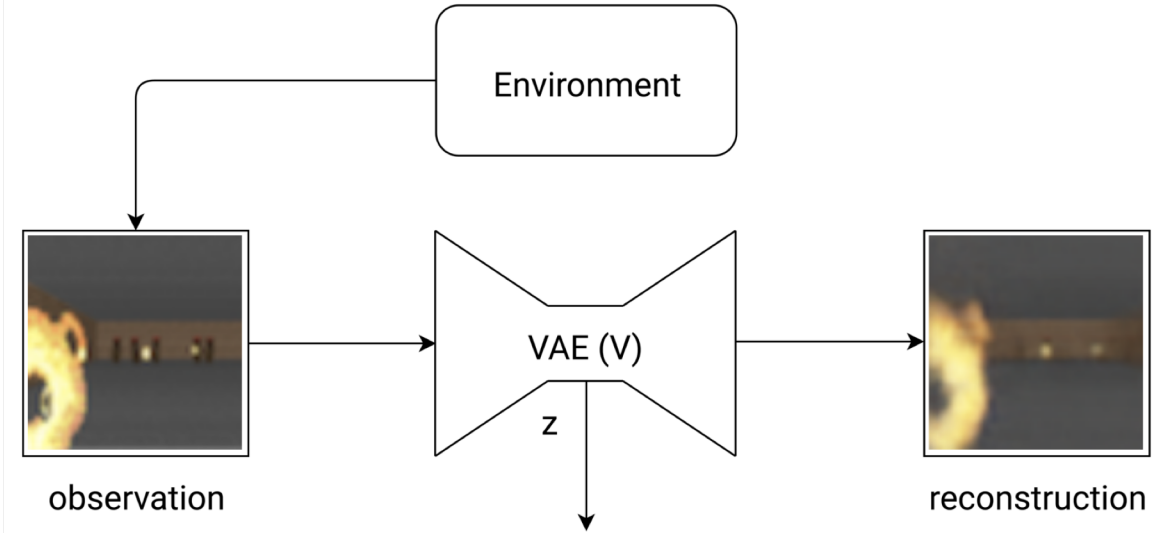
Summary of Asynchronous Model-based RL

- Problem
  - Need fast and data efficient methods for robotic tasks
- Contributions
  - General asynchronous model-based framework
  - Wall-clock time speed-up
  - Sample efficiency
  - Effect on policy regularization & data exploration
  - Effective on real robots

Outline

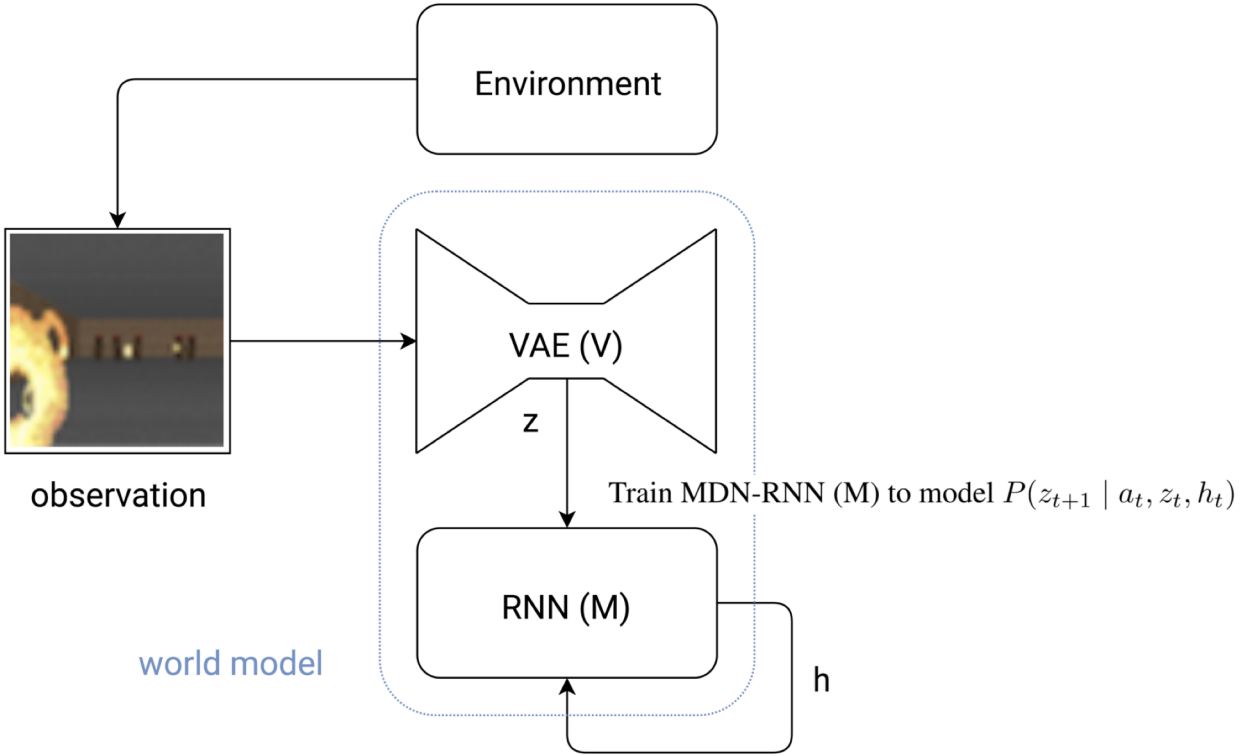
- Model-based RL
- Ensemble Methods
  - Model-Ensemble Trust Region Policy Optimization
  - Model-based RL via Meta Policy Optimization
- Asynchronous Model-based RL
- Vision-based Model-based RL

World Models

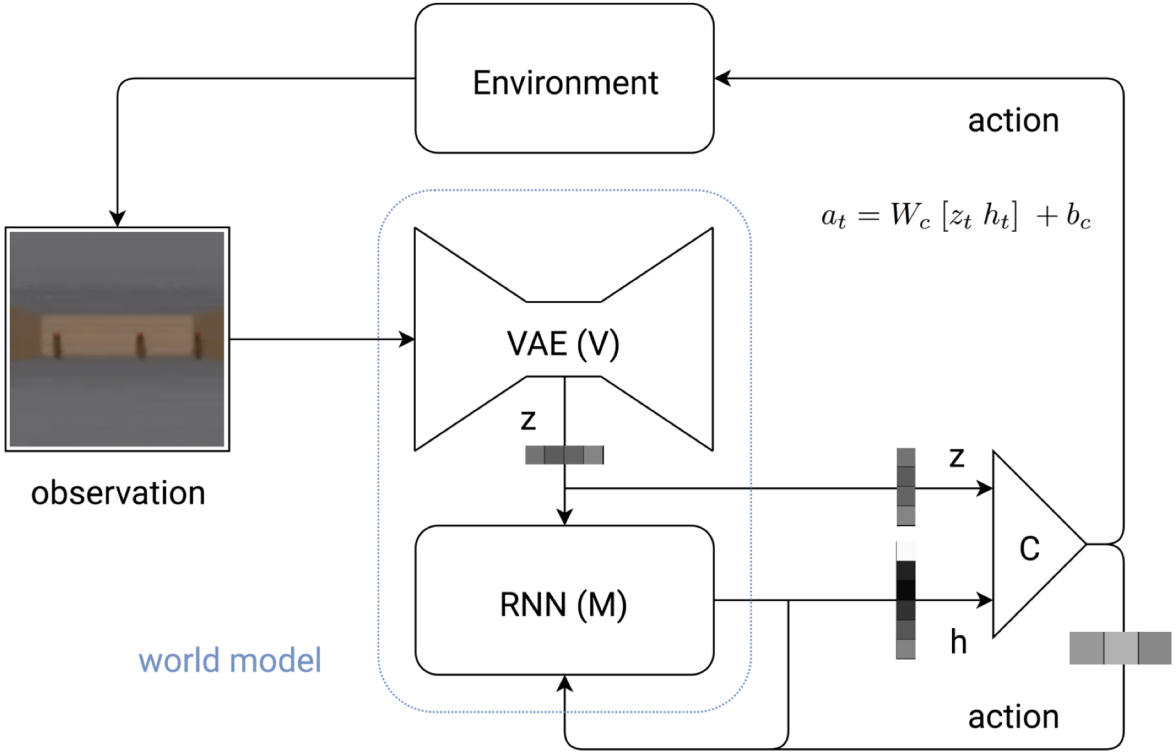


Train VAE (V) to encode frames into z

World Models

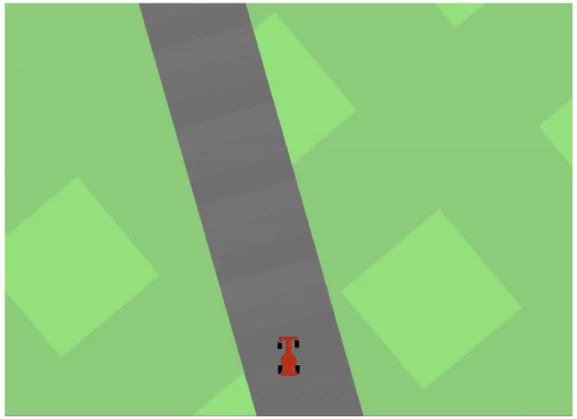


World Models



World Models

CarRacing-v0



- Randomly generated tracks
- Stay on tiles, travel around track
- Average Score > 900 (100 trials) to “solve” task

Procedure:

1. Collect 10,000 rollouts from a random policy.
2. Train VAE (V) to encode frames into $z \in \mathcal{R}^{32}$.
3. Train MDN-RNN (M) to model $P(z_{t+1} \mid a_t, z_t, h_t)$.
4. Evolve linear controller (C) to maximize the expected cumulative reward of a rollout. $a_t = W_c \, [z_t \; h_t] \; + b_c$

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

Source: Taste of Home (www.tasteofhome.com)

World Models

Action-Conditional Video Prediction using Deep Networks in Atari Games (2015)

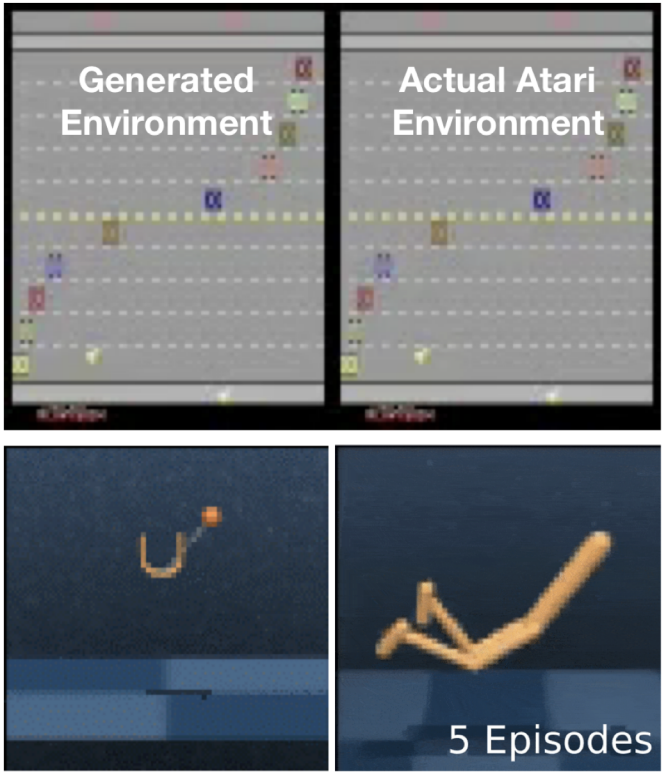
Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard Lewis and Satinder Singh

Model-Based Reinforcement Learning for Atari (2018)

Błażej Osiński, Łukasz Kaiser, Mohammad Babaeizadeh, George Tucker, Dumitru Erhan, Ryan Sepassi, Chelsea Finn, Sergey Levine, Piotr Kozakowski, Konrad Czechowski, Piotr Miłos and Henryk Michalewski

Learning Latent Dynamics for Planning from Pixels (2018)

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee and James Davidson



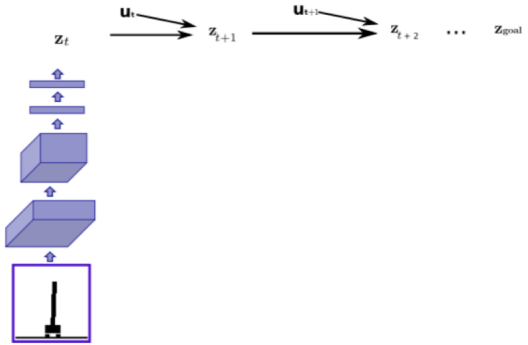
Embed to Control

Embed to Control: Model based RL from raw images

Can we perform model based RL starting from raw images ?

→ Standard algorithms would fail in pixel space

→ We want to **unsupervisedly** learn latent space $\mathbf{z}_t$ for control from images $\mathbf{x}_t$

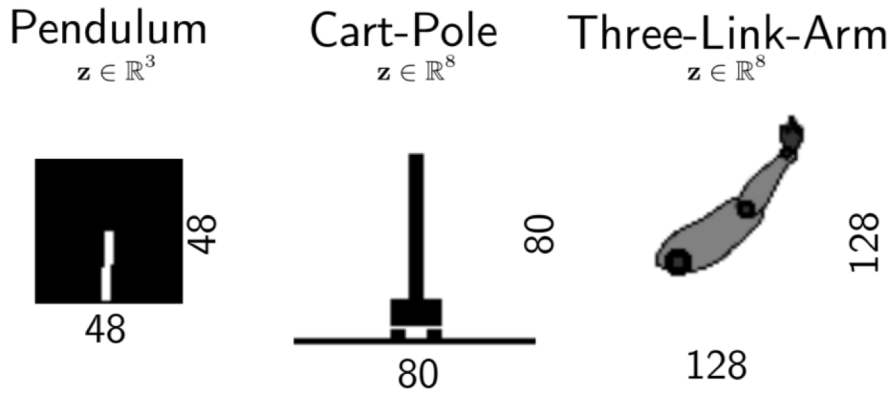


Related attempts:

- [Lange et al.:Deep Learning of Visual Control Policies, 2010]
- [Wahlstroem et al.: From Pixels to Torques, 2015]

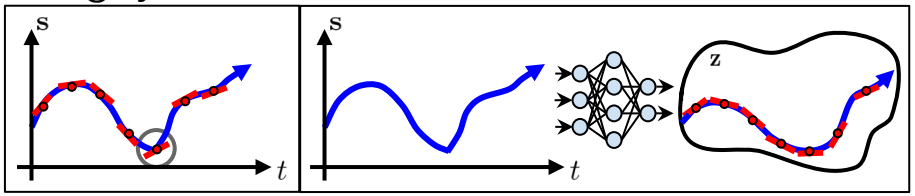
Embed to Control

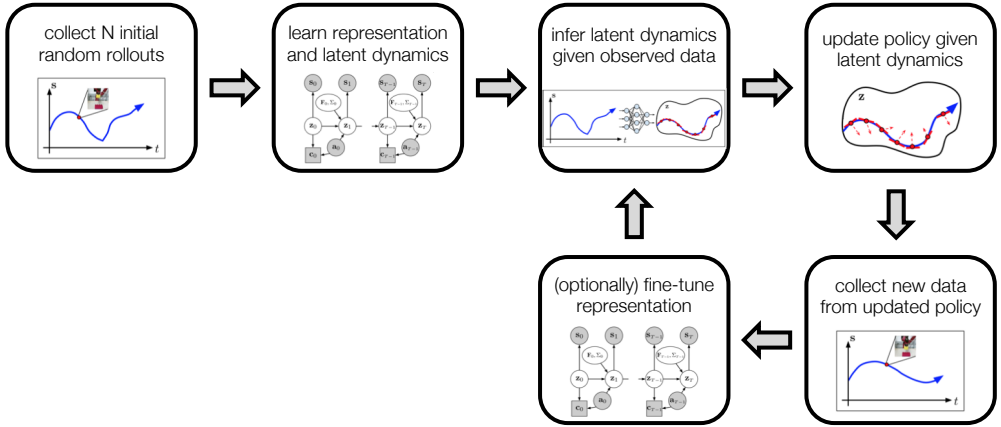
Systems we consider

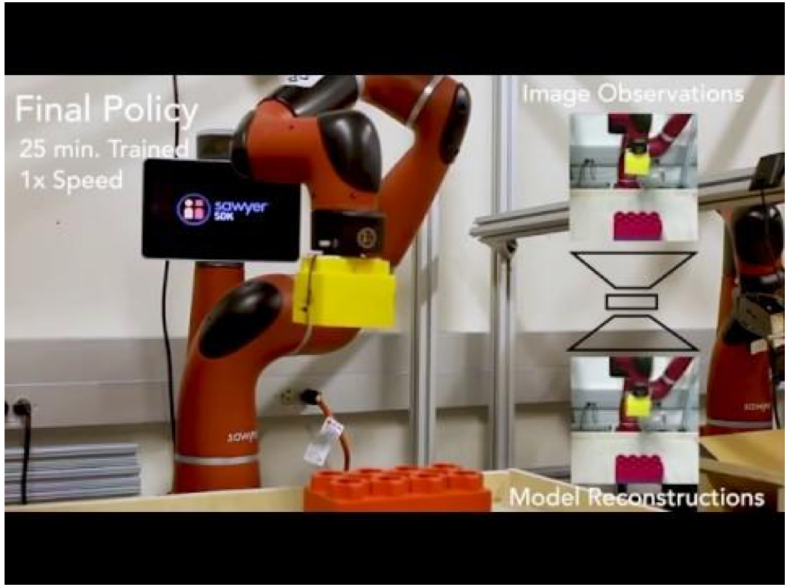


SOLAR: Deep Structured Representations for Model-Based Reinforcement Learning

Marvin Zhang*, Sharad Vikram*, Laura Smith, Pieter Abbeel, Matthew Johnson, Sergey Levine



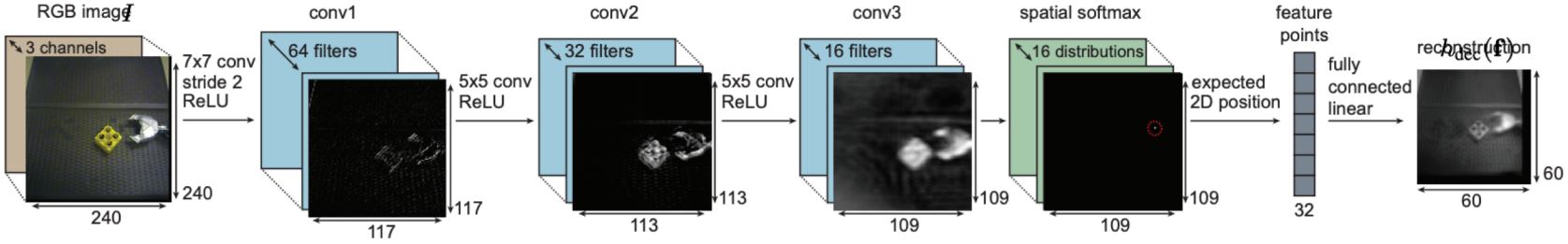




https://goo.gl/AJKocL

Deep Spatial Autoencoders

- Deep Spatial Autoencoders for Visuomotor Learning, Finn, Tan, Duan, D
  (https://arxiv.org/abs/1509.06113)
  - Train deep spatial autoencoder
  - Model-based RL through iLQR in the latent space



Robotic Priors / PVEs

- PVEs: Position-Velocity Encoders for Unsupervised Learning of Structured State Representations
  Rico Jonschkowski, Roland Hafner, Jonathan Scholz, and Martin Riedmiller (https://arxiv.org/pdf/1705.09805.pdf)
  - Learn an embedding without reconstruct

$$\mathbf{s}_t^{(p)} = \phi(\mathbf{o}_t)$$

$$\mathbf{s}_t^{(v)} = \alpha(\mathbf{s}_t^{(p)} - \mathbf{s}_{t-1}^{(p)})$$

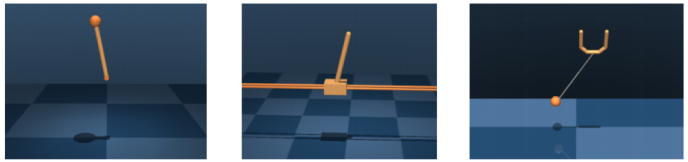
$$L_{\text{conservation}} = \mathbf{E}\left[\left(\|\mathbf{s}_t^{(v)}\| - \|\mathbf{s}_{t-1}^{(v)}\|\right)^2\right]$$

$$L_{\text{variation}} = \mathbf{E}\left[e^{-\|\mathbf{s}_a^{(p)} - \mathbf{s}_b^{(p)}\|}\right]$$

$$L_{\text{controlability (i)}} = e^{-\text{Cov}(\mathbf{a}_{t,i}, \mathbf{s}_{t+1,i}^{(a)})} = e^{-\mathbf{E}\left[\left(a_{t,i} - \mathbf{E}[a_{t,i}]\right)\left(s_{t+1,i}^{(a)} - \mathbf{E}[s_{t+1,i}^{(a)}]\right)\right]}$$

$$L_{\text{slowness}} = \mathbf{E}\left[\|\mathbf{s}_t^{(p)} - \mathbf{s}_{t-1}^{(p)}\|^2\right]$$

$$L_{\text{inertia}} = \mathbf{E}\left[\|\mathbf{s}_t^{(v)} - \mathbf{s}_{t-1}^{(v)}\|^2\right] = \mathbf{E}\left[\|\mathbf{s}_t^{(a)}\|^2\right]$$

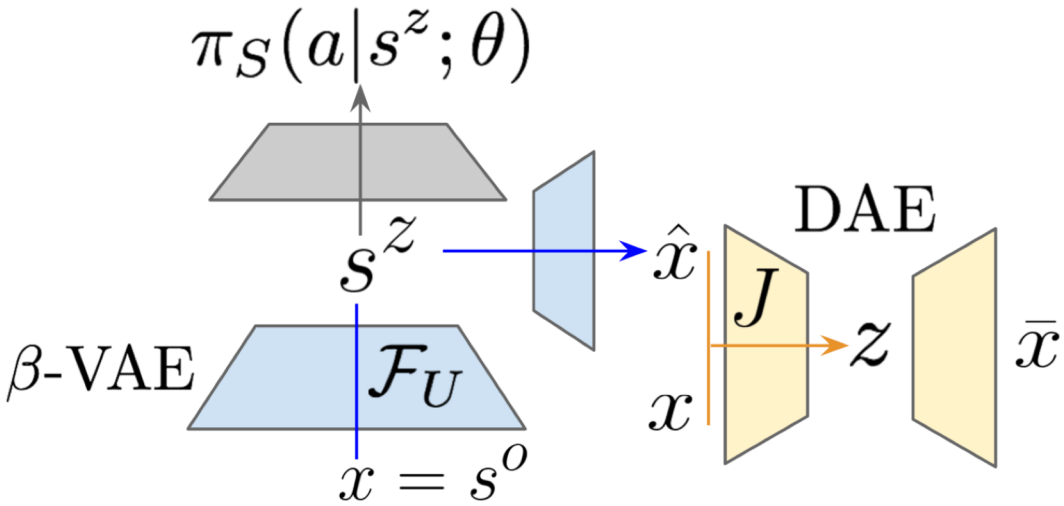


(a) Inverted pendulum (b) Cart-pole (c) Ball in cup

Disentangled Representation Learning Agent (Darla)

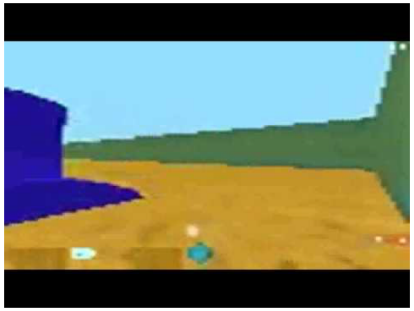
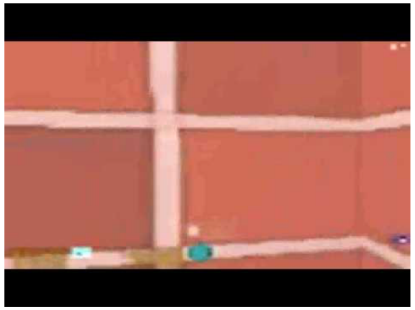
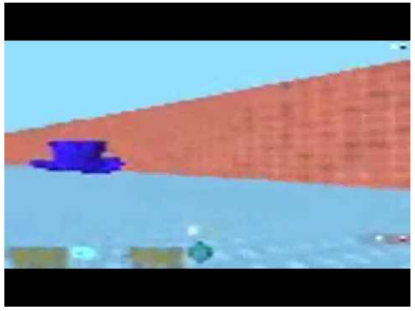
DARLA: Improving Zero-Shot Transfer in Reinforcement Learning

Irina Higgins, Arka Pal, Andrei A. Rusu, Loic Matthey, Christopher P Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, Alexander Lerchner (https://arxiv.org/abs/1707.08475)



DeepMind Lab Transfer

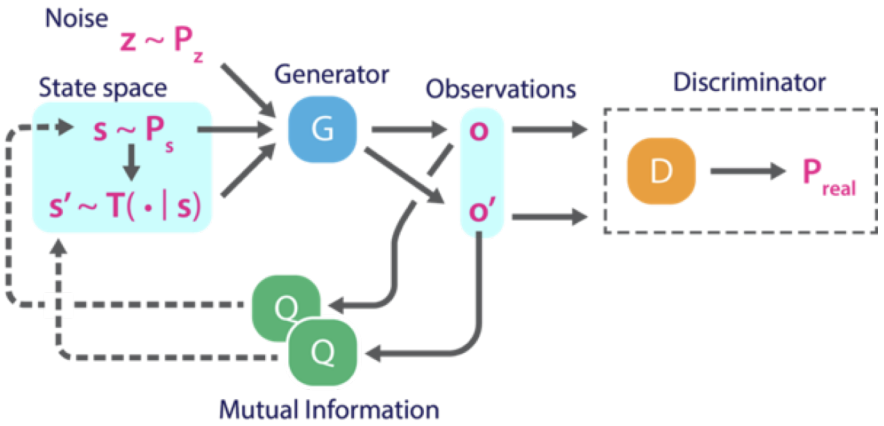
DARLA vs DQN baseline

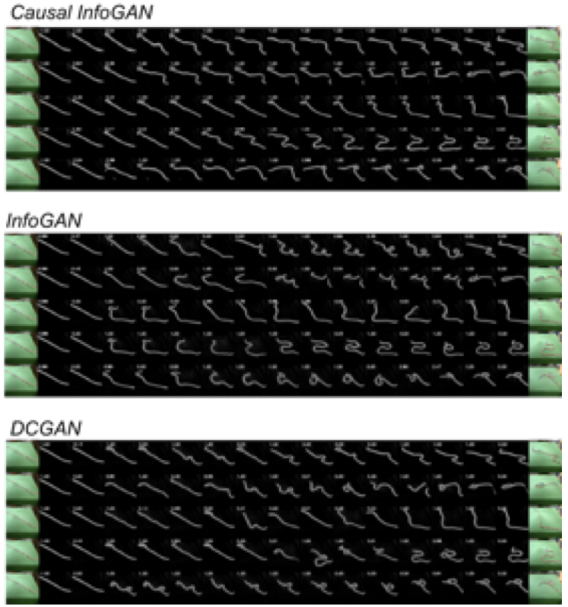
	Train	Transfer
DQN		
DARLA		

Causal InfoGAN

Learning Plannable Representations with Causal InfoGAN

Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart Russell, Pieter Abbeel (https://arxiv.org/pdf/1807.09341.pdf)



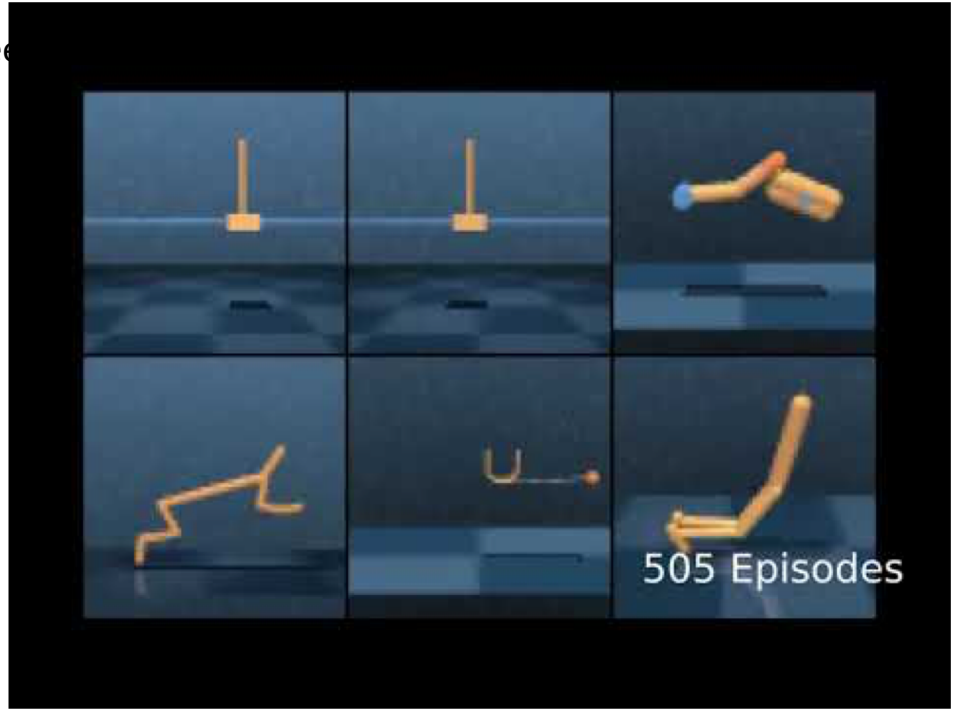


PlaNet

Learning latent dynamics for planning from pixels

Danijar Hafner, T. Lillicrap, I Fischer, R Villegas, D Ha, H Lee

- Learn latent space dynamics model
- Multi-step prediction
- Planning in latent space



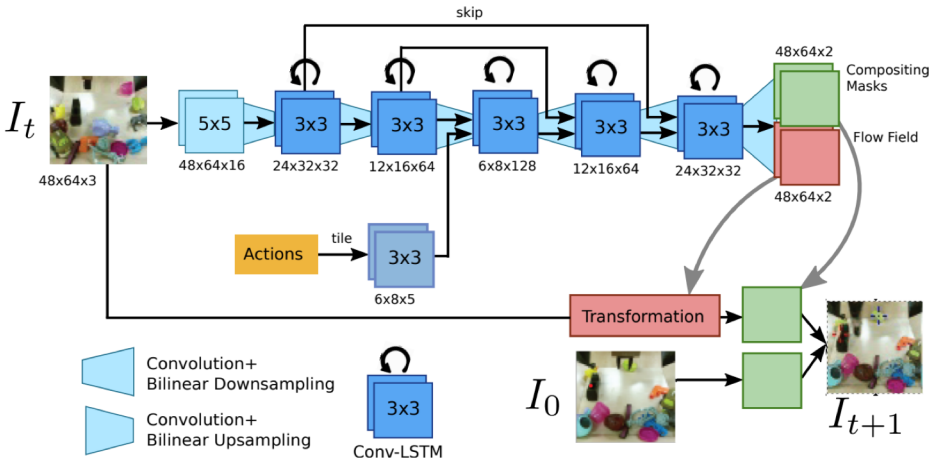
Visual Foresight

Deep Visual Foresight for Planning Robot Motion, Finn and Levine, ICRA 2017 http://arxiv.org/abs/1610.00696

Visual Foresight: Model-Based Deep Reinforcement Learning for Vision-Based Robotic Control, Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, Sergey Levine, https://arxiv.org/abs/1812.00568,

https://bair.berkeley.edu/blog/2018/11/30/visual-rl/

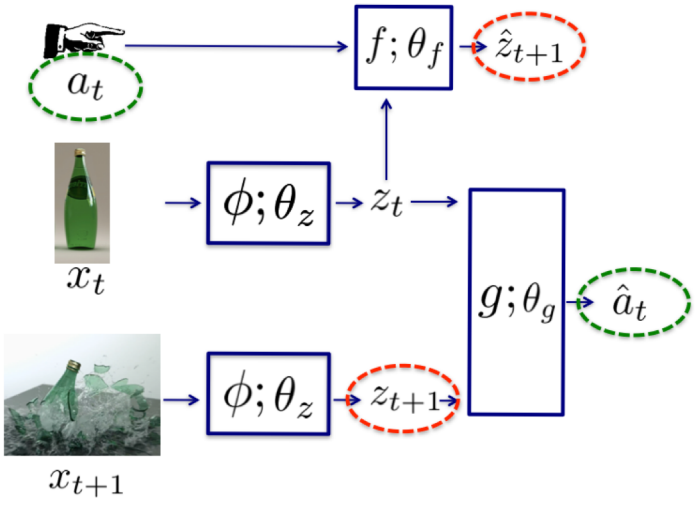
- Video prediction + Cross Entropy Maximization for MPC



Forward + Inverse Dynamics Models

Learning to Poke by Poking: Experiential Learning of Intuitive Physics, Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, Sergey Levine, https://arxiv.org/abs/1606.07419

- Learning a forward model in latent space
- BUT: couldn't the latent features always be zero?
- SOLUTION: require the features from t and t+1 to be sufficient to predict a_t



An alternative to making predictions in the pixel space!

Forward + Inverse Dynamics Models

Learning to Poke by Poking: Experiential Learning of Intuitive Physics, Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, Sergey Levine, https://arxiv.org/abs/1606.07419

- Learning a forward model in latent space
- BUT: couldn't the latent features always be zero?
- SOLUTION: require the features from t and t+1 to be sufficient to predict a_t

Predictron

The Predictron: End-To-End Learning and Planning

David Silver, Hado van Hasselt, Matteo Hessel, Tom Schaul, Arthur Guez, Tim Harley, Gabriel Dulac-Arnold, David Reichert, Neil Rabinowitz, Andre Barreto, Thomas Degris (https://arxiv.org/pdf/1612.08810.pdf)

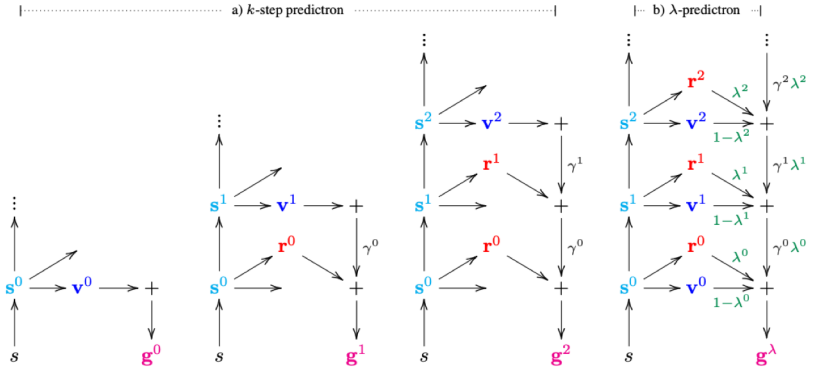
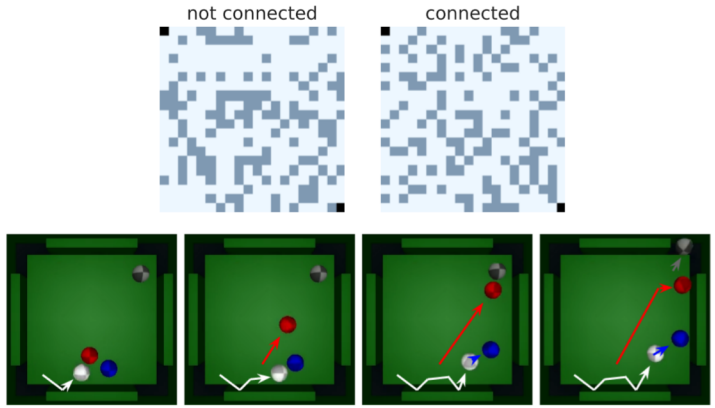


Figure 1. a) The k-step predictron architecture. The first three columns illustrate 0, 1 and 2-step pathways through the predictron. The 0-step preturn reduces to standard model-free value function approximation; other preturns "imagine" additional steps with an internal model. Each pathway outputs a k-step preturn $\mathbf{g}^k$ that accumulates discounted rewards along with a final value estimate. In practice all k-step preturns are computed in a single forward pass. b) The λ-predictron architecture. The λ-parameters gate between the different preturns. The output is a λ-preturn $\mathbf{g}^\lambda$ that is a mixture over the k-step preturns. For example, if $\boldsymbol{\lambda}^0 = \mathbf{1}, \boldsymbol{\lambda}^1 = \mathbf{1}, \boldsymbol{\lambda}^2 = \mathbf{0}$ then we recover the 2-step preturn, $\mathbf{g}^\lambda = \mathbf{g}^2$. Discount factors $\boldsymbol{\gamma}^k$ and λ-parameters $\boldsymbol{\lambda}^k$ are dependent on state $\mathbf{s}^k$; this dependence is not shown in the figure.

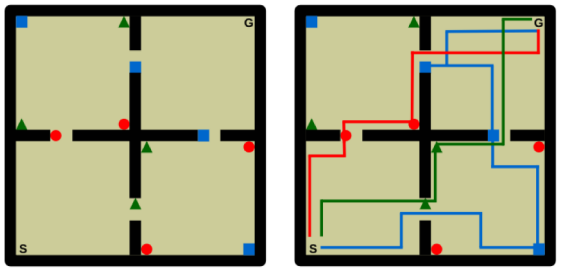


Successor Features

Successor Features for Transfer in Reinforcement Learning

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, Hado van Hasselt, David Silver (https://arxiv.org/abs/1606.05312)

$$
\begin{aligned}
Q^{\pi}(s,a) &= \mathrm{E}^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \ldots \,|\, S_t = s, A_t = a\right] \\
&= \mathrm{E}^{\pi}\left[\boldsymbol{\phi}_{t+1}^{\top}\mathbf{w} + \gamma \boldsymbol{\phi}_{t+2}^{\top}\mathbf{w} + \ldots \,|\, S_t = s, A_t = a\right] \\
&= \mathrm{E}^{\pi}\left[\textstyle\sum_{i=t}^{\infty} \gamma^{i-t}\boldsymbol{\phi}_{i+1} \,|\, S_t = s, A_t = a\right]^{\top}\mathbf{w} = \boldsymbol{\psi}^{\pi}(s,a)^{\top}\mathbf{w}
\end{aligned}
$$



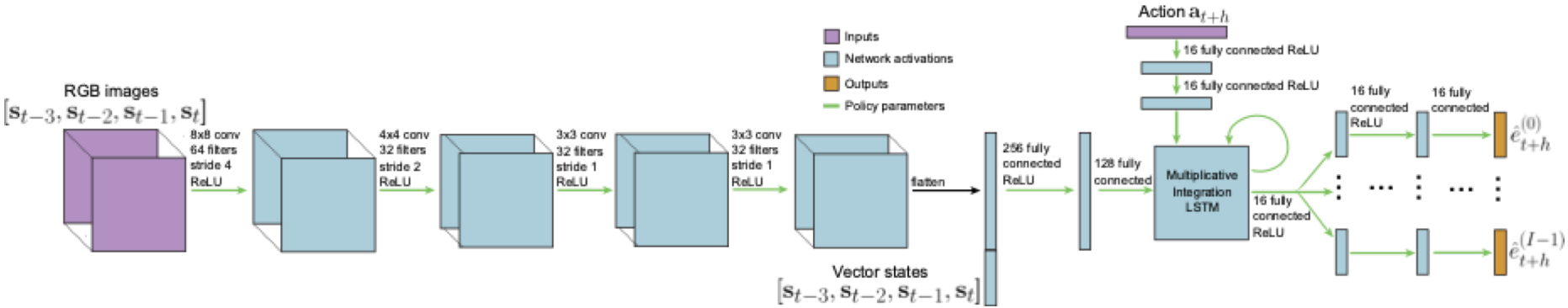
Kahn et al.

Composable Action-Conditioned Predictors: Flexible Off-Policy Learning for Robot Navigation

Gregory Kahn*, Adam Villaflor*, Pieter Abbeel, Sergey Levine, CoRL 2018 (https://arxiv.org/pdf/1810.07167.pdf)

Self-supervised Deep Reinforcement Learning with Generalized Computation Graphs for Robot Navigation

Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, Sergey Levine, ICRA 2018 (https://arxiv.org/pdf/1709.10489.pdf)



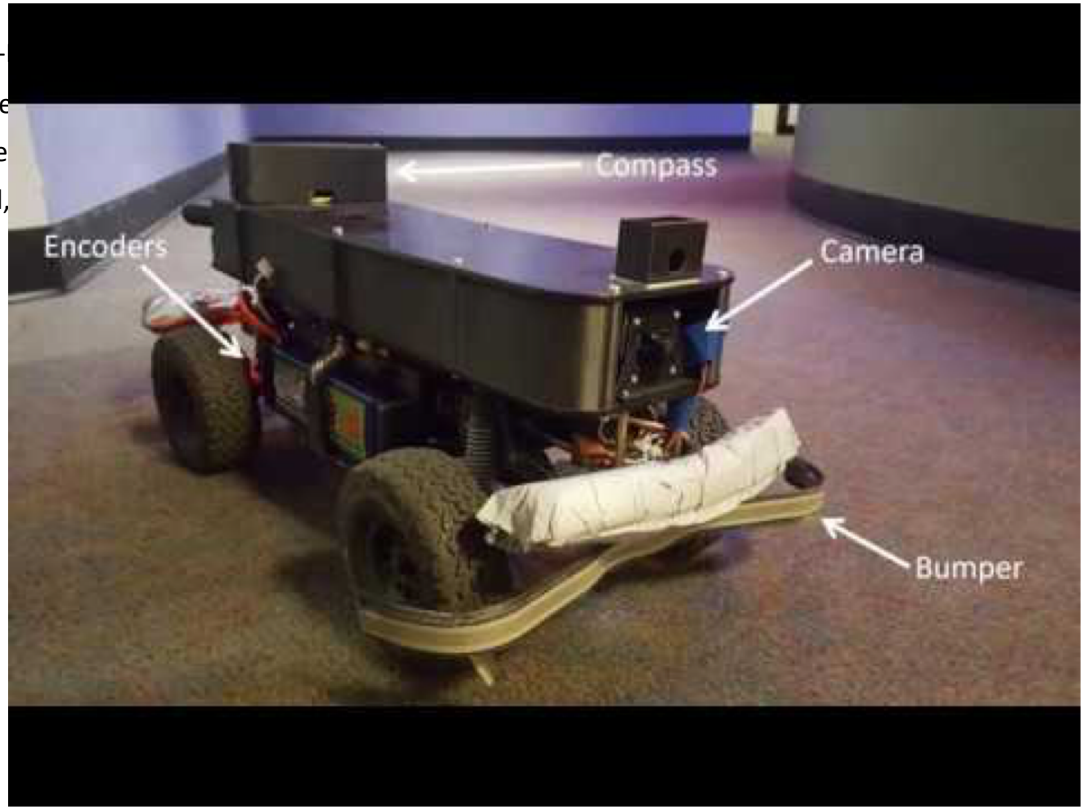
Kahn et al.

Composable Action-Conditioned Predictors: Flexible Off-

Gregory Kahn*, Adam Villaflor*, Pieter Abbeel, Sergey Le

Self-supervised Deep Reinforcement Learning with Gene

Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel,



Some Theory References on State Representations

- From skills to symbols: Learning symbolic representations for abstract high-level planning: https://jair.org/index.php/jair/article/view/11175
- Homomorphism: https://www.cse.iitm.ac.in/~ravi/papers/KBCS04.pdf
- Towards a unified theory of state abstraction for mdps: https://pdfs.semanticscholar.org/ca9a/2d326b9de48c095a6cb5912e1990d2c5ab46.pdf
- Model reduction techniques for computing approximately optimal solutions for markov decision processes.https://arxiv.org/abs/1302.1533
- Adaptive aggregation methods for infinite horizon dynamic programming
- Transfer via soft homomorphisms. http://www.ifaamas.org/Proceedings/aamas09/pdf/01_Full%20Papers/12_67_FP_0798.pdf
- Near optimal behavior via approximate state abstraction https://arxiv.org/abs/1701.04113
- Using PCA to Efficiently Represent State Spaces: http://irll.eecs.wsu.edu/wp-content/papercite-data/pdf/2015icml-curran.pdf

A Separation Principle for Control in the Age of Deep Learning

A Separation Principle for Control in the Age of Deep Learning

Alessandro Achille, Stefano Soatto (https://arxiv.org/abs/1711.03321)

We review the problem of defining and inferring a "state" for a control system based on complex, high-dimensional, highly uncertain measurement streams such as videos. Such a state, or representation, should contain all and only the information needed for control, and discount nuisance variability in the data.