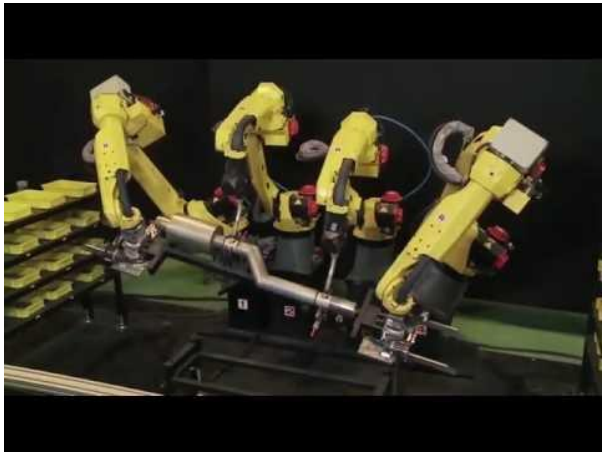
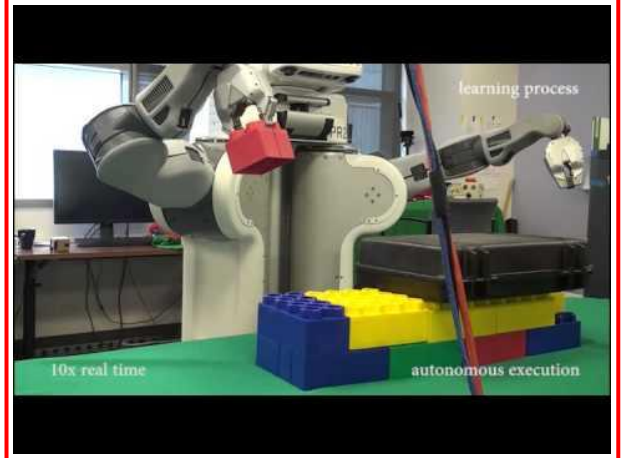
Model-Based Reinforcement Learning and Representation Learning for Robots

Marvin Zhang
marvin@eecs

What can (and can’t) robots do?





Outline

Review

Model-based RL

Representation learning

Putting it all together

Outline

Review

Model-based RL

Representation learning

Putting it all together

Markov decision processes

A Markov decision process (MDP) is characterized by:

$\mathcal{S}$ – state space — states $\mathbf{s} \in \mathcal{S}$ (discrete or continuous)

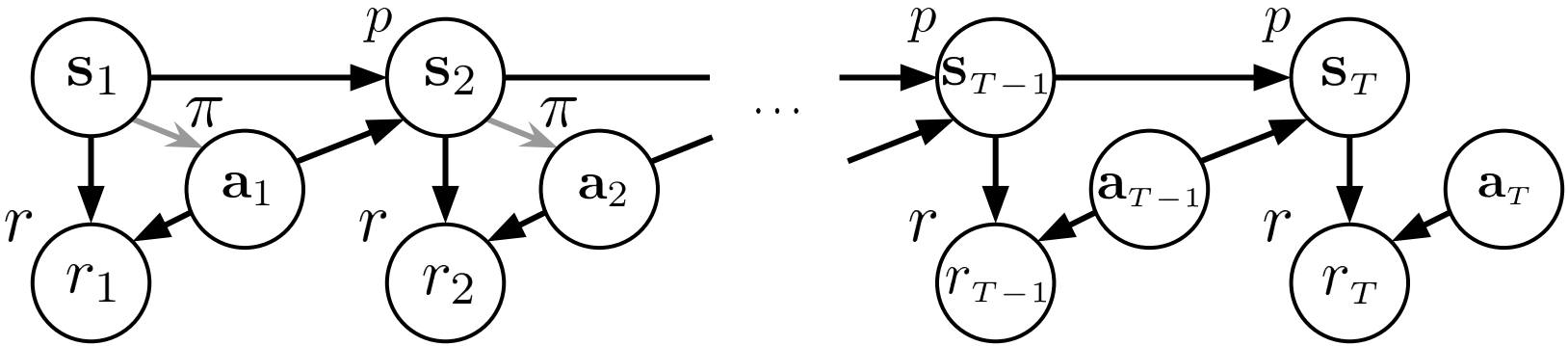
$\mathcal{A}$ – action space — actions $\mathbf{a} \in \mathcal{A}$ (discrete or continuous)

$p(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t)$ – transition probabilities

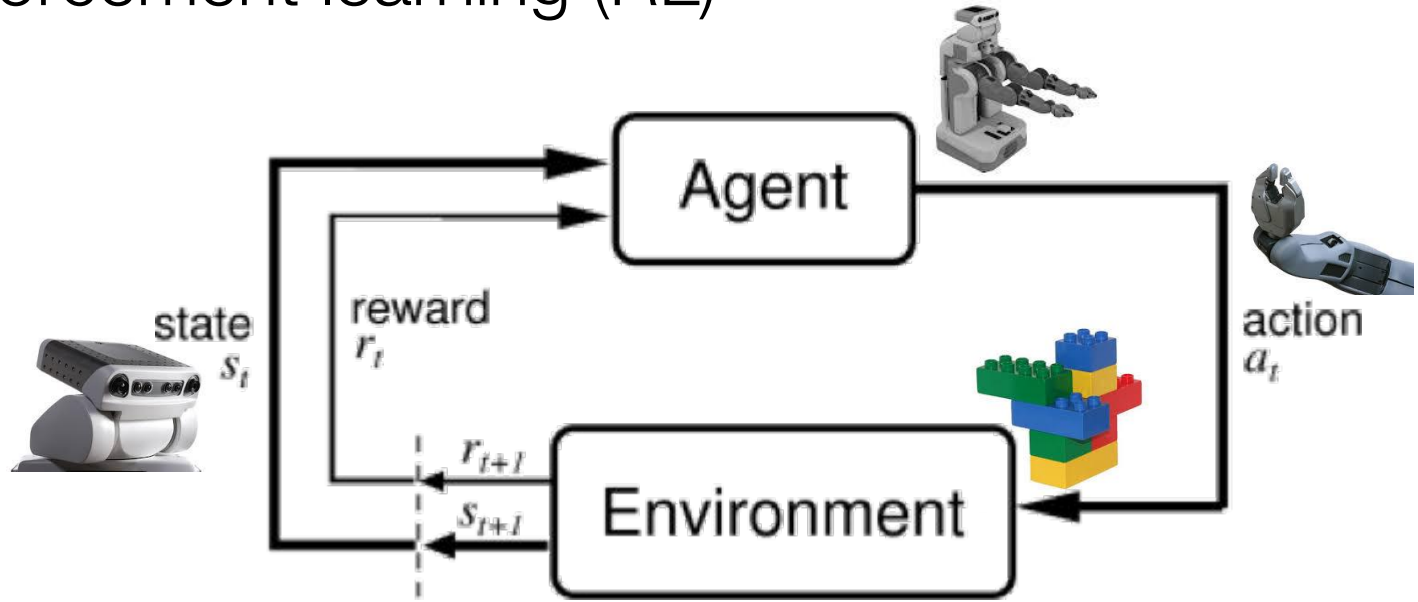
$r(\mathbf{s}_t, \mathbf{a}_t)$ – reward function

Goal: find optimal policy $\pi^\star(\mathbf{s}_t) \rightarrow \mathbf{a}_t$ that maximizes expected rewards

Markov decision processes

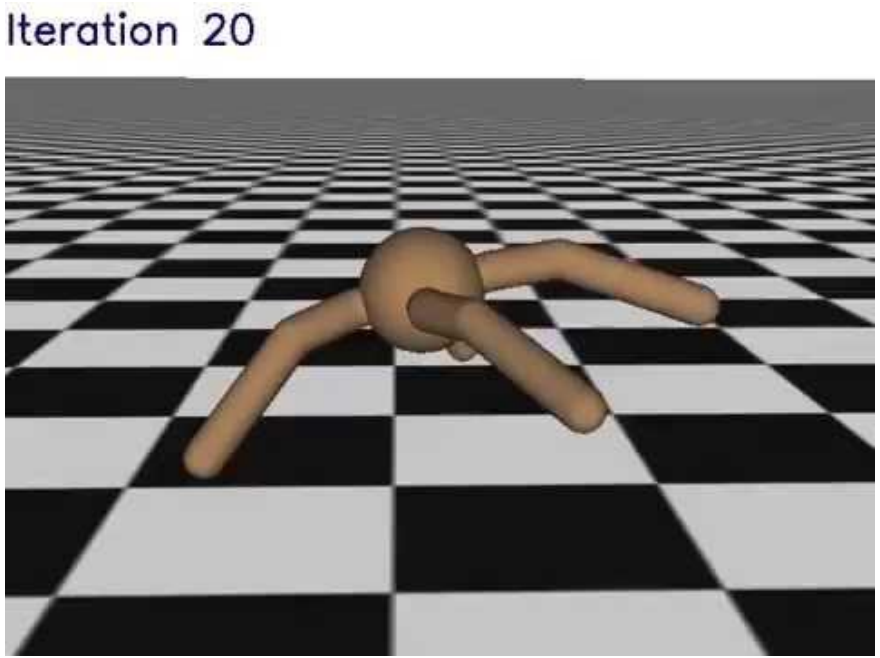


Reinforcement learning (RL)



Maximize rewards through *trial and error*

Model-free RL



Schulman et al, “High-dimensional continuous control using generalized advantage estimation”. ICLR 2016.

Outline

Review

Model-based RL

Representation learning

Putting it all together

Model-free RL is slow

High-Dimensional Continuous Control Using Generalized Advantage Estimation

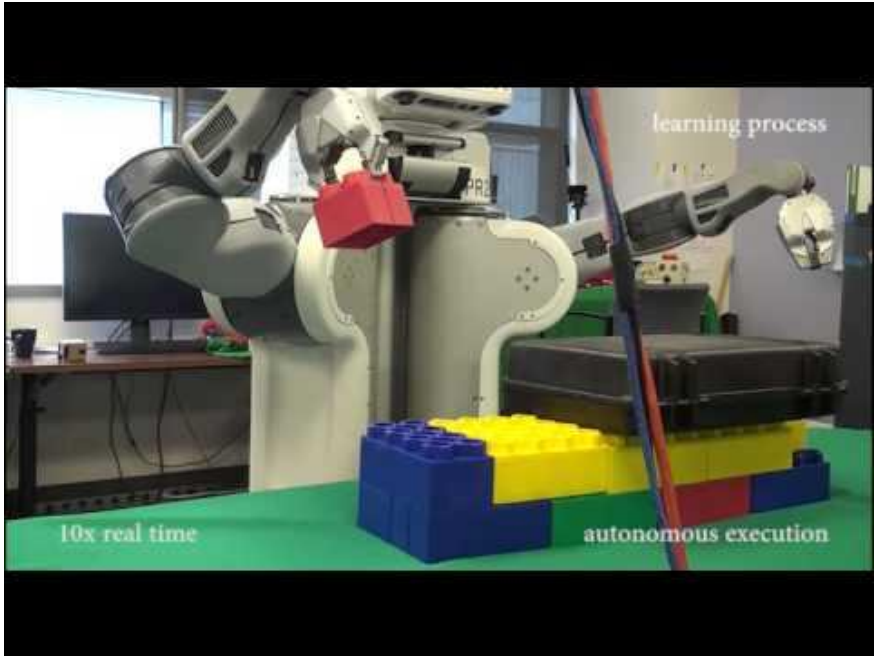
John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, Pieter Abbeel

(Submitted on 8 Jun 2015 (v1), last revised 20 Oct 2018 (this version, v6))

Policy gradient methods are an appealing approach in reinforcement learning because they directly optimize the cumulative reward and can straightforwardly be used with nonlinear function approximators such as neural networks. The two main challenges are the large number of samples typically required, and the difficulty of obtaining stable and steady improvement despite the nonstationarity of the incoming data. We address the first challenge by using value functions to substantially reduce the variance of policy gradient estimates at the cost of some bias, with an exponentially-weighted estimator of the advantage function that is analogous to TD(lambda). We address the second challenge by using trust region optimization procedure for both the policy and the value function, which are represented by neural networks.

Our approach yields strong empirical results on highly challenging 3D locomotion tasks, learning running gaits for bipedal and quadrupedal simulated robots, and learning a policy for getting the biped to stand up from starting out lying on the ground. In contrast to a body of prior work that uses hand-crafted policy representations, our neural network policies map directly from raw kinematics to joint torques. Our algorithm is fully model-free, and the amount of simulated experience required for the learning tasks on 3D bipeds corresponds to 1-2 weeks of real time.

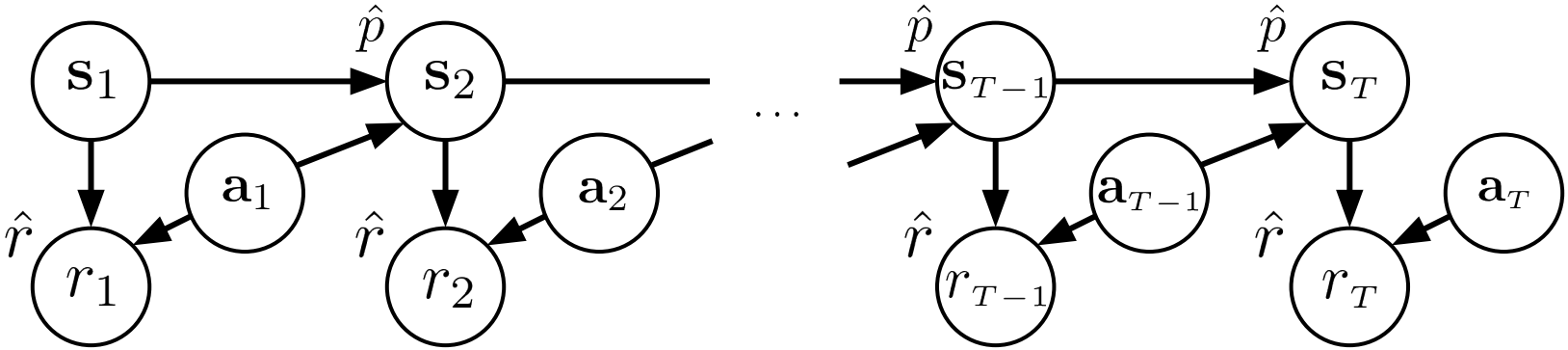
Model-based RL is fast



Levine*, Finn* et al, “End-to-end training of deep visuomotor policies”. JMLR 2016.

Why is model-based RL fast?

Intuition: the learned model can be used to find a better policy, or even $\pi^\star$,
without having to generate any additional data



Model-based RL for robotics

Discrete MDPs are easier to work with, but many robotics settings are continuous

What are some easy continuous MDPs to work with?

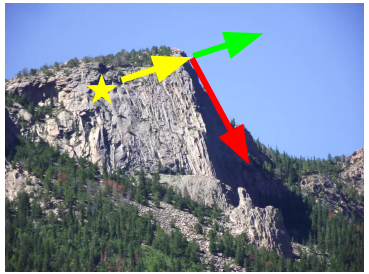
Perhaps the easiest: model transitions as linear (Gaussian) and rewards as quadratic

$$\hat{p}(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t) = \mathcal{N}\left(\mathbf{F}_t \mathbf{s}_t + \mathbf{H}_t \mathbf{a}_t, \Sigma_t\right)$$
$$\hat{r}(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{s}_t^\top \mathbf{Q}_t \mathbf{s}_t + \mathbf{a}_t^\top \mathbf{R}_t \mathbf{a}_t$$

Why is this easy? $\pi^\star$ can be computed exactly! (with the *LQR equations*)

LQR

If the MDP actually has linear transitions and quadratic reward, we can compute the optimal policy with no data needed except to learn the models



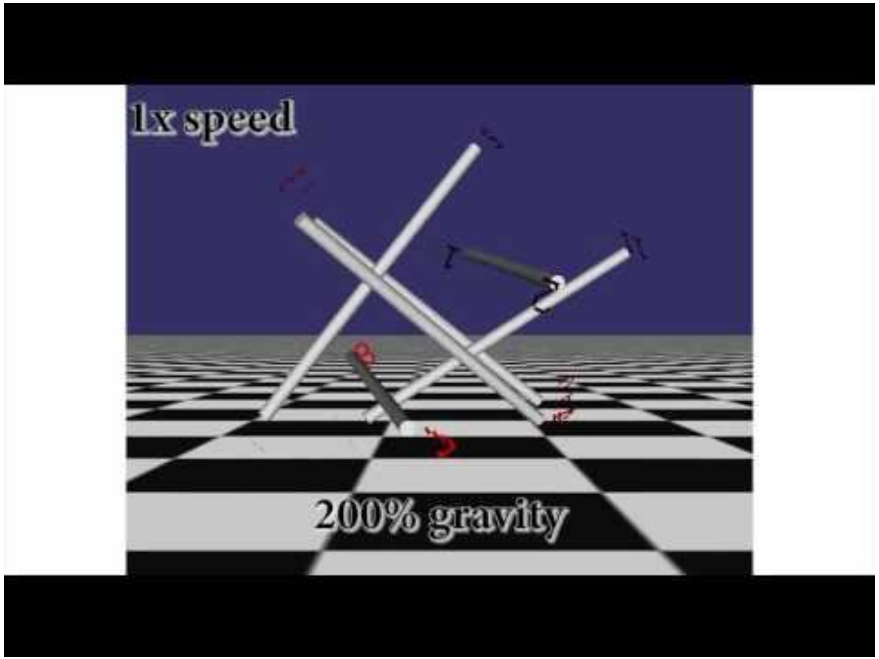
This can be way less data than what model-free RL needs

But if the MDP is not linear-quadratic, we might get a bad policy

Solution: approximate LQR to try and get a *better* policy, not the optimal policy

Then, re-learn models and repeat, to find better and better policies

Approximating LQR



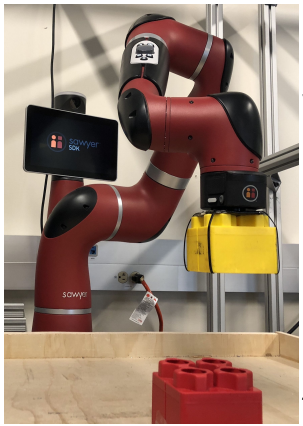
Zhang*, Geng*, Bruce* et al, “Deep reinforcement learning for tensegrity robot locomotion”. ICRA 2017.

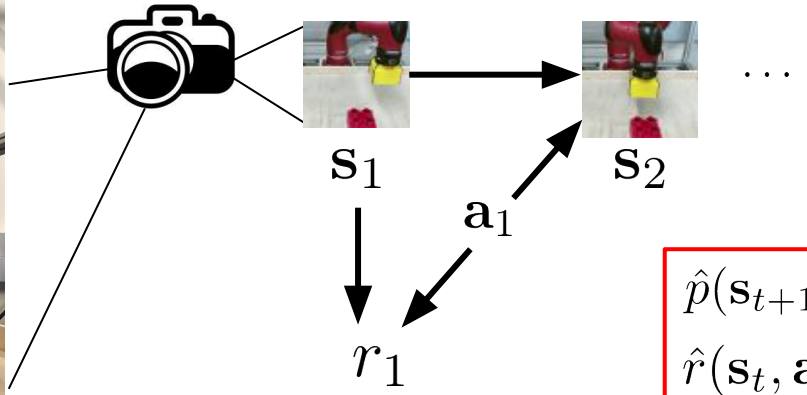
Chebotar*, Hausman*, Zhang* et al, “Combining model-based and model-free updates for trajectory-centric reinforcement learning”. ICML 2017.

Does it work?

Approximate LQR is fundamentally limited by how well linear transition models and quadratic reward models can fit the MDP

So it won't work for very complex state spaces such as images from a camera feed





$$\hat{p}(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t) = \mathcal{N}\left(\mathbf{F}_t \mathbf{s}_t + \mathbf{H}_t \mathbf{a}_t, \Sigma_t\right)$$

$$\hat{r}(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{s}_t^\top \mathbf{Q}_t \mathbf{s}_t + \mathbf{a}_t^\top \mathbf{R}_t \mathbf{a}_t \quad ?$$

Outline

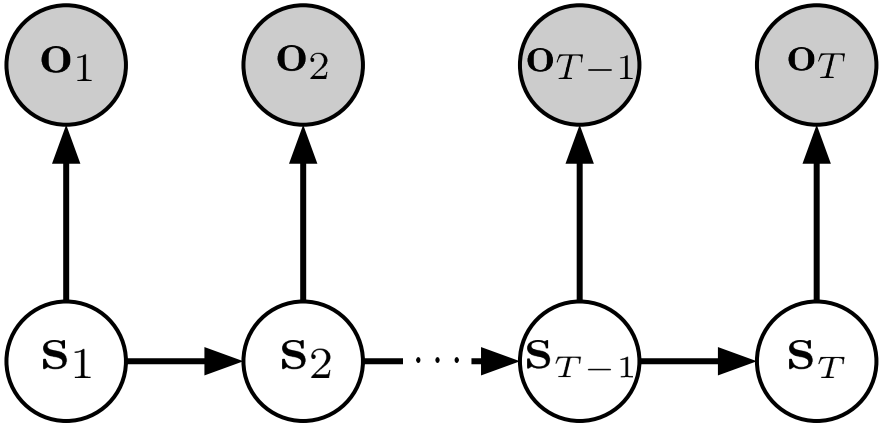
Review

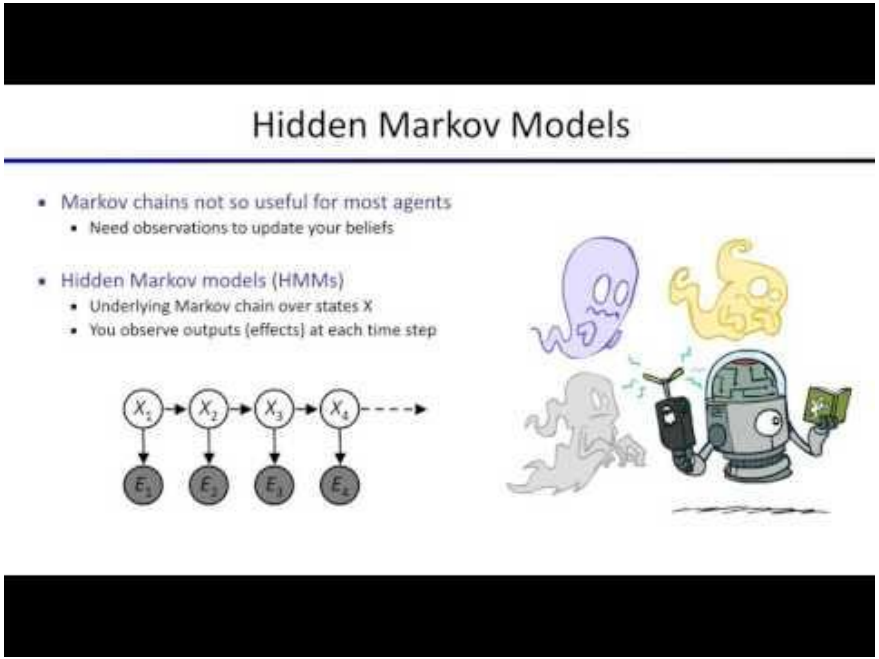
Model-based RL

Representation learning

Putting it all together

Hidden Markov models

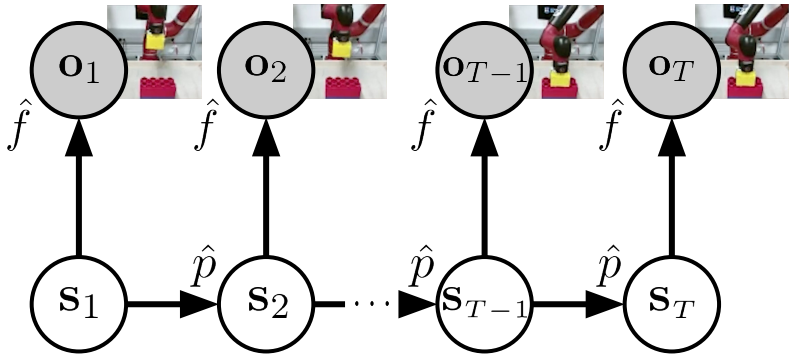




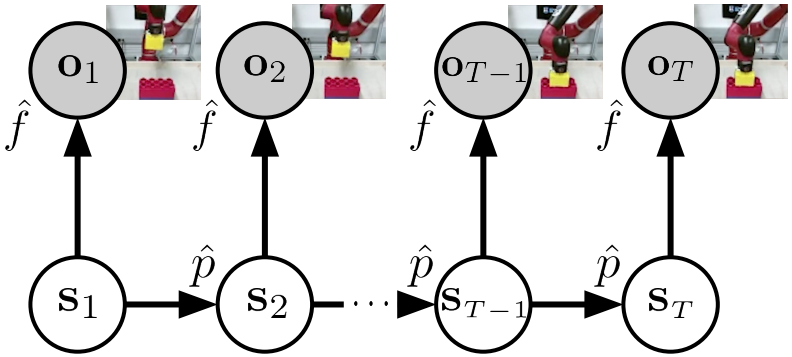
Hidden Markov models

If images are too complicated to treat as states, can we treat them as observations?

In particular, let's say there is a *hidden state* that has a linear transition model, and the state generates the image through the *observation model* $\hat{f}(\mathbf{s}_t) \rightarrow \mathbf{o}_t$



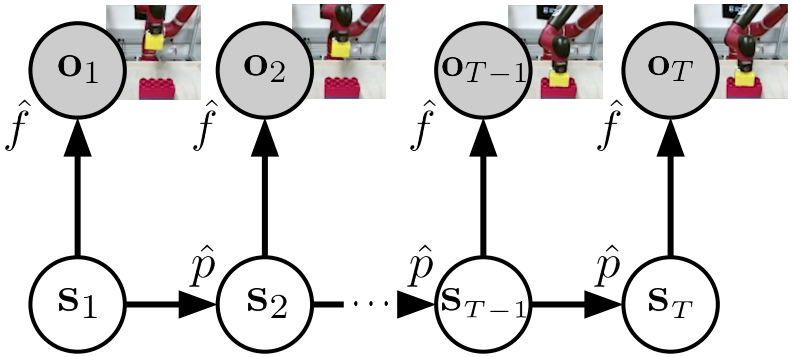
Extending HMMs



This is a good start, but we're still missing some things:

1. How do we figure out the hidden state?
2. How do we incorporate actions and costs?

Extending HMMs



This is a good start, but we're still missing some things:

1. **How do we figure out the hidden state?**
2. How do we incorporate actions and costs?

Representation learning

The previous assumption for HMMs was that we knew the representation of the hidden state, such as the (unknown) positions of the ghosts

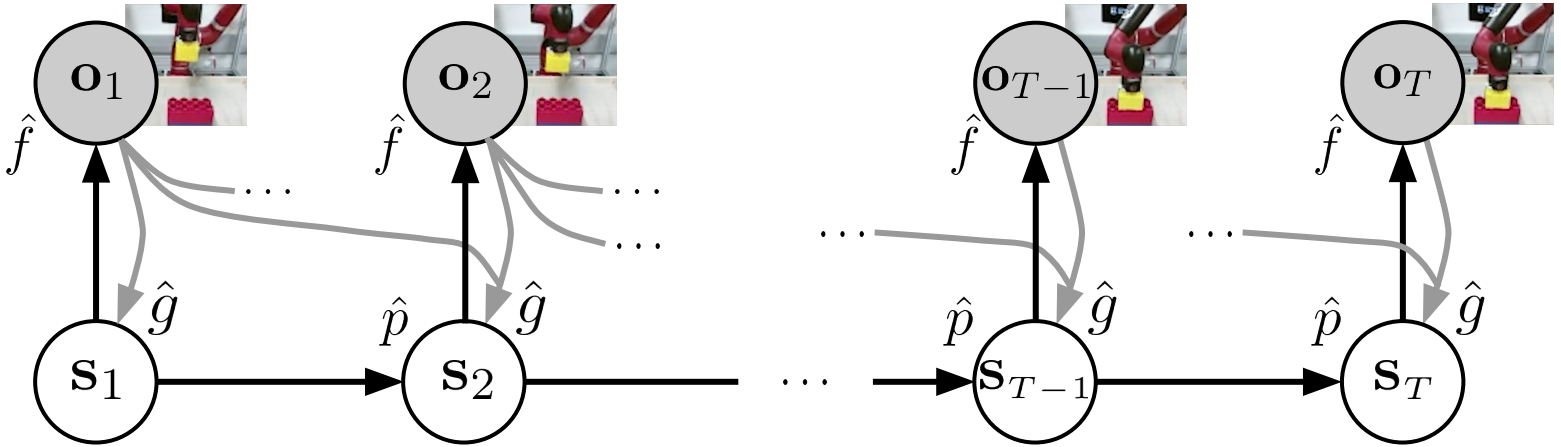
But in real applications such as robotics, we don't always know this representation

Even if we do, it might not be a good fit for a linear transition model!

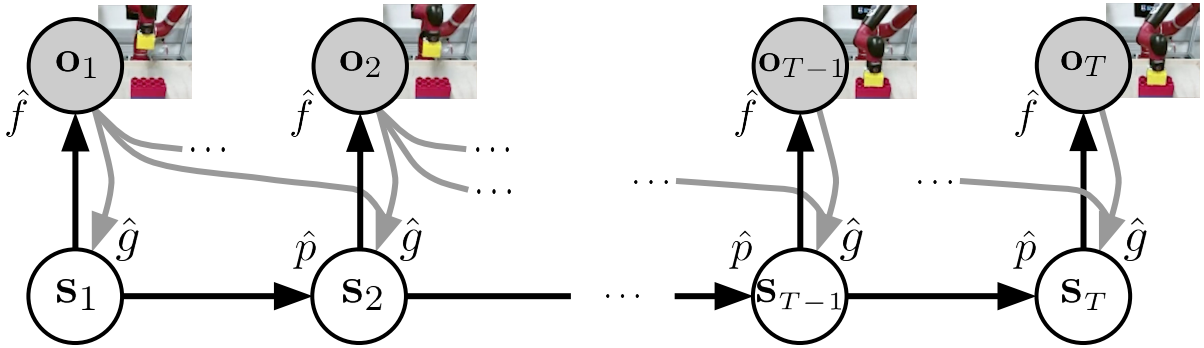
Solution: learn a state representation model $\hat{g}(\mathbf{o}_1, \ldots, \mathbf{o}_t) \rightarrow \mathbf{s}_t$

Representation learning

How do we learn $\hat{g}$? We want it to be consistent with the other models



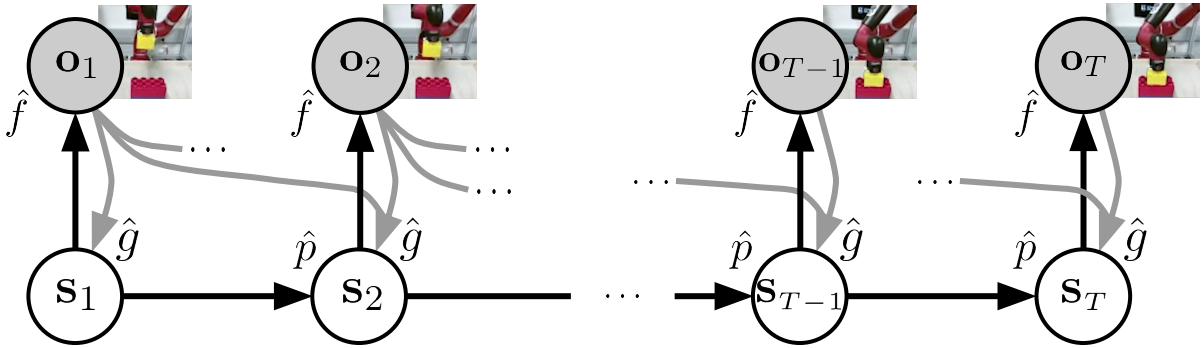
Extending HMMs



This is a good start, but we're still missing some things:

1. **How do we figure out the hidden state?**
2. How do we incorporate actions and costs?

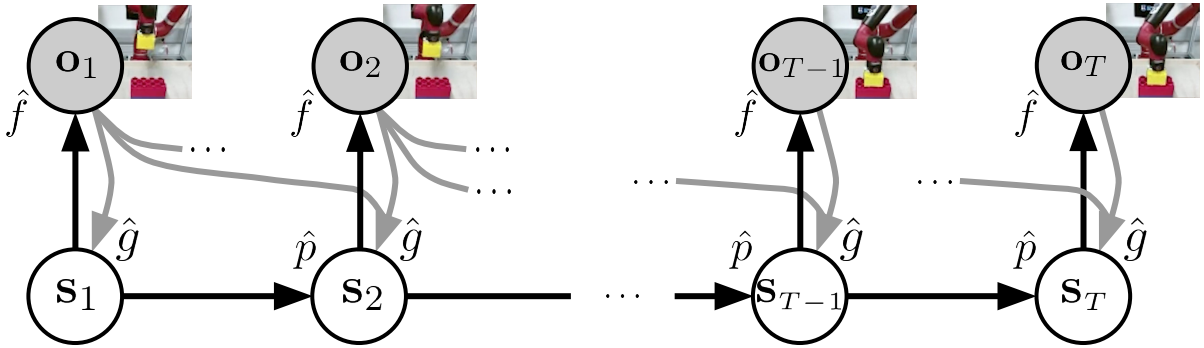
Extending HMMs



This is a good start, but we're still missing some things:

1. **How do we figure out the hidden state?** *Representation learning*
2. How do we incorporate actions and costs?

Extending HMMs



This is a good start, but we're still missing some things:

1. How do we figure out the hidden state? *Representation learning*
2. **How do we incorporate actions and costs?**

Outline

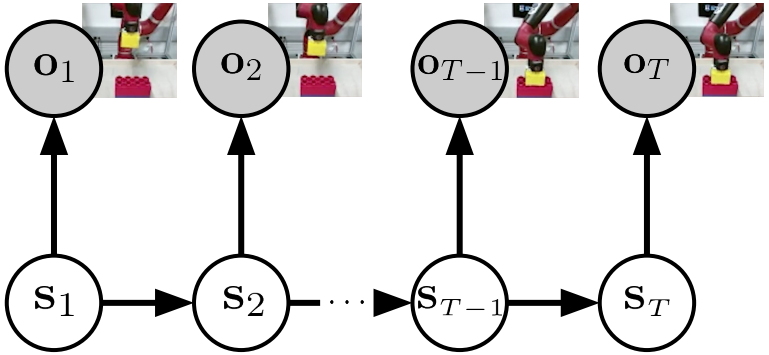
Review

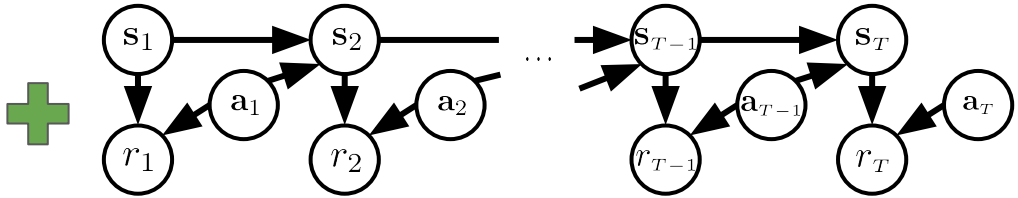
Model-based RL

Representation learning

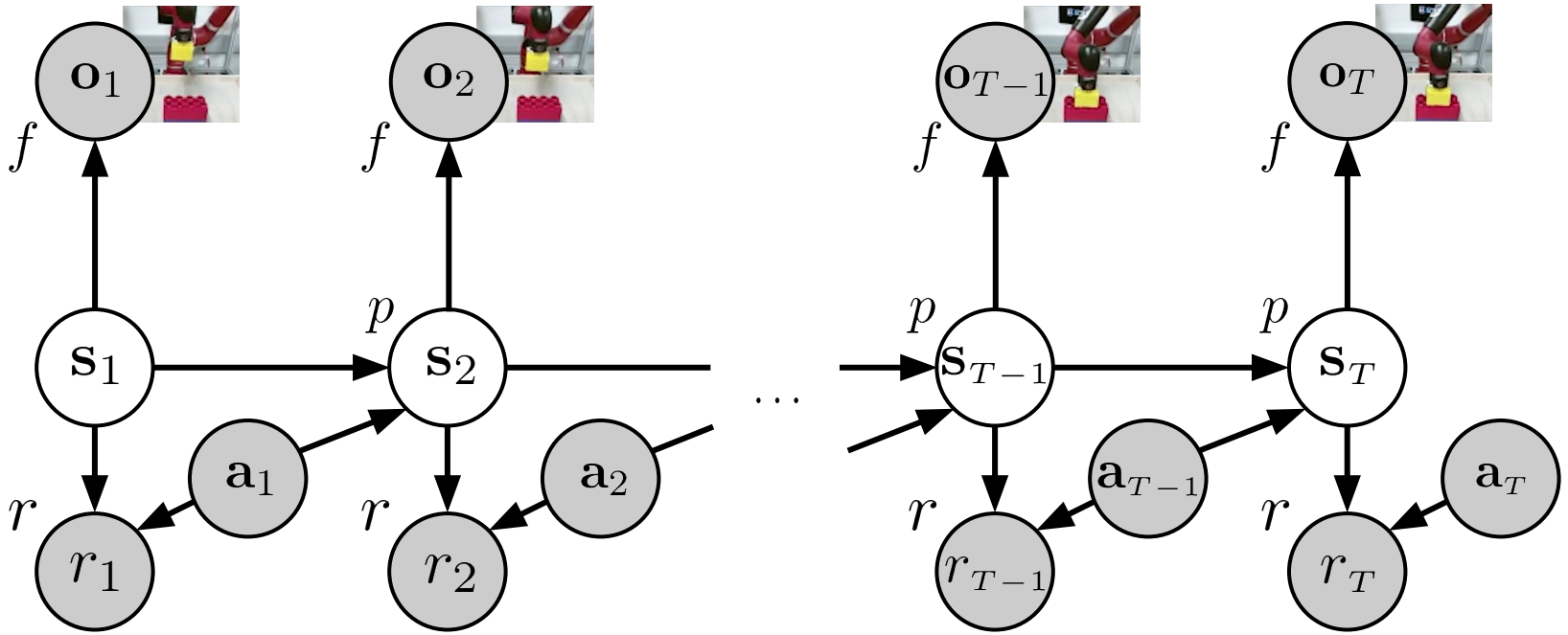
Putting it all together

HMM + MDP = ?

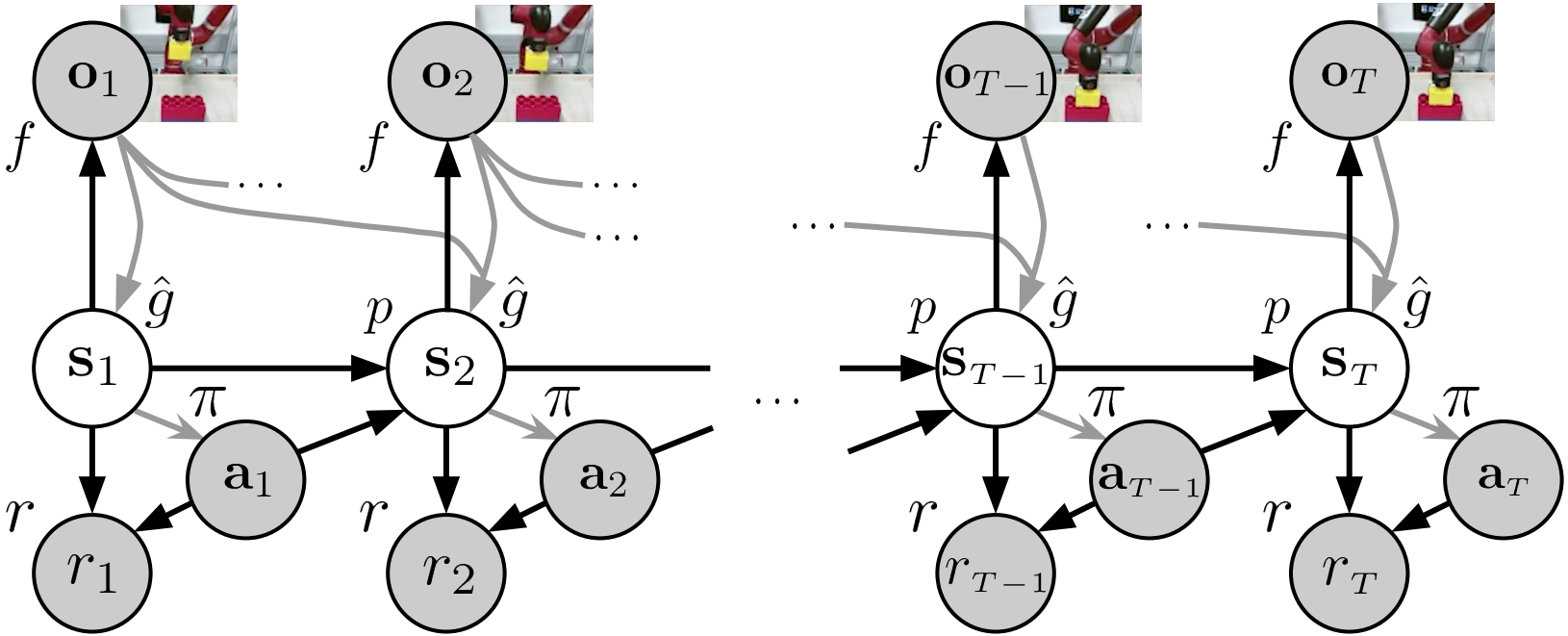

$\mathbf{o}_1$
$\mathbf{o}_2$
$\mathbf{o}_{T-1}$
$\mathbf{o}_T$
$\mathbf{s}_1$
$\mathbf{s}_2$
$\cdots$
$\mathbf{s}_{T-1}$
$\mathbf{s}_T$


$\mathbf{s}_1$
$\mathbf{s}_2$
$\cdots$
$\mathbf{s}_{T-1}$
$\mathbf{s}_T$
$\mathbf{a}_1$
$\mathbf{a}_2$
$\mathbf{a}_{T-1}$
$\mathbf{a}_T$
r_1
r_2
r_{T-1}
r_T

Partially observed MDP (POMDP)



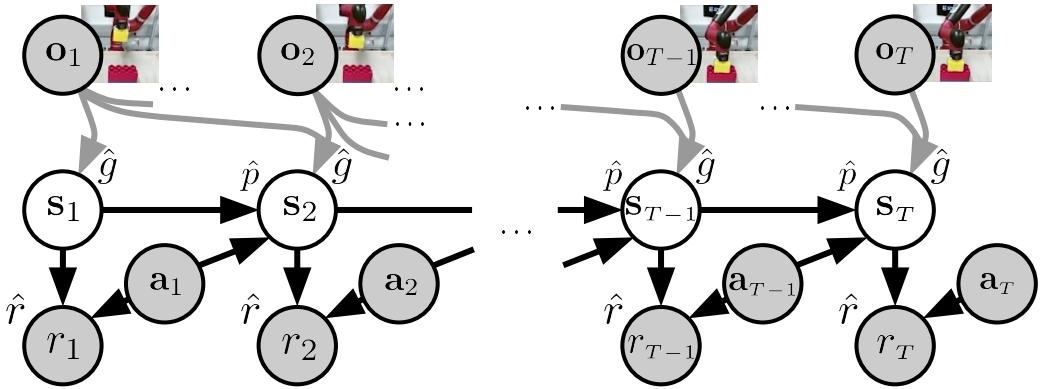
RL in POMDPs


$\mathbf{o}_1$
$\mathbf{o}_2$
$\mathbf{o}_{T-1}$
$\mathbf{o}_T$
f
$\hat{g}$
p
$\mathbf{s}_1$
$\mathbf{s}_2$
$\mathbf{s}_{T-1}$
$\mathbf{s}_T$
π
$\mathbf{a}_1$
$\mathbf{a}_2$
$\mathbf{a}_{T-1}$
$\mathbf{a}_T$
r
r_1
r_2
r_{T-1}
r_T

Model-based RL in POMDPs

Just like before, if we learn a linear transition and quadratic reward model, we can approximate LQR to get a better policy

The difference is that these models are now for a hidden state, which we estimate with our state representation model



Model-based RL from images

Zhang*, Vikram* et al, “SOLAR: Deep structured representations for model-based reinforcement learning”. ICML 2019.

Questions?

marvin@eecs

Referenced papers (*not* representative of the larger research community):

J. Schulman, P. Moritz, S. Levine, M. Jordan, P. Abbeel. “High-dimensional continuous control using generalized advantage estimation”. ICLR 2016, https://arxiv.org/abs/1506.02438.

S. Levine*, C. Finn*, T. Darrell, P. Abbeel. “End-to-end training of deep visuomotor policies”. JMLR 2016, https://arxiv.org/abs/1504.00702.

M. Zhang*, X. Geng, J. Bruce*, K. Caluwaerts, M. Vespignani, V. SunSpiral, P. Abbeel, S. Levine. “Deep Reinforcement Learning for Tensegrity Robot Locomotion”. ICRA 2017, https://arxiv.org/abs/1609.09049.

Y. Chebotar*, K. Hausman*, M. Zhang*, G. Sukhatme, S. Schaal, S. Levine. “Combining Model-Based and Model-Free Updates for Trajectory-Centric Reinforcement Learning”. ICML 2017, https://arxiv.org/abs/1703.03078.

M. Zhang*, S. Vikram*, L. Smith, P. Abbeel, M. Johnson, S. Levine. “SOLAR: Deep Structured Representations for Model-Based Reinforcement Learning”. ICML 2019, https://arxiv.org/abs/1808.09105.