Optimal Control with Learned Forward Models

Pieter Abbeel
UC Berkeley

Jan Peters
TU Darmstadt

Where we are?

Reinforcement Learning Data

$\mathscr{D} = \{(\mathbf{x}_i, \mathbf{u}_i, \mathbf{x}_{i+1}, r_i)\}\}$

$\mathbf{x}' \sim \mathscr{P}^{\mathbf{u}}_{\mathbf{x}\mathbf{x}'}$

$r \approx \mathscr{R}^{\mathbf{u}}_{\mathbf{x}\mathbf{x}'}$

$V^*(\mathbf{x})$

$\pi^*(\mathbf{u}|\mathbf{x})$

Optimal Control with Model Learning

Now

$V^*(\mathbf{x})$

$\pi^*(\mathbf{u}|\mathbf{x})$

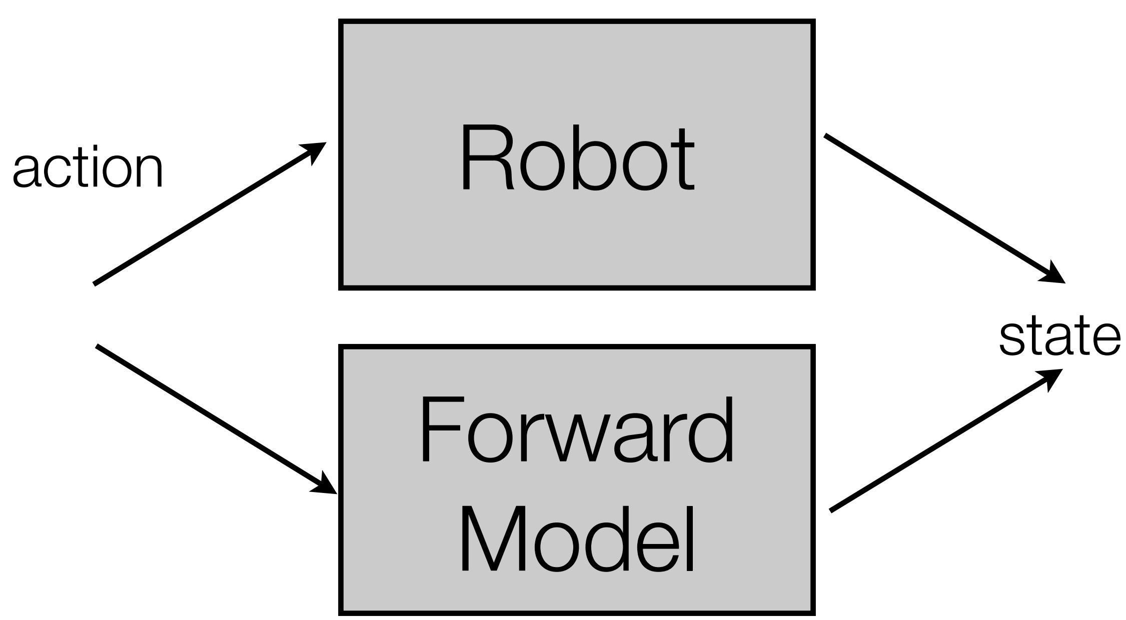
Value Function Methods

$\pi^*(\mathbf{u}|\mathbf{x})$

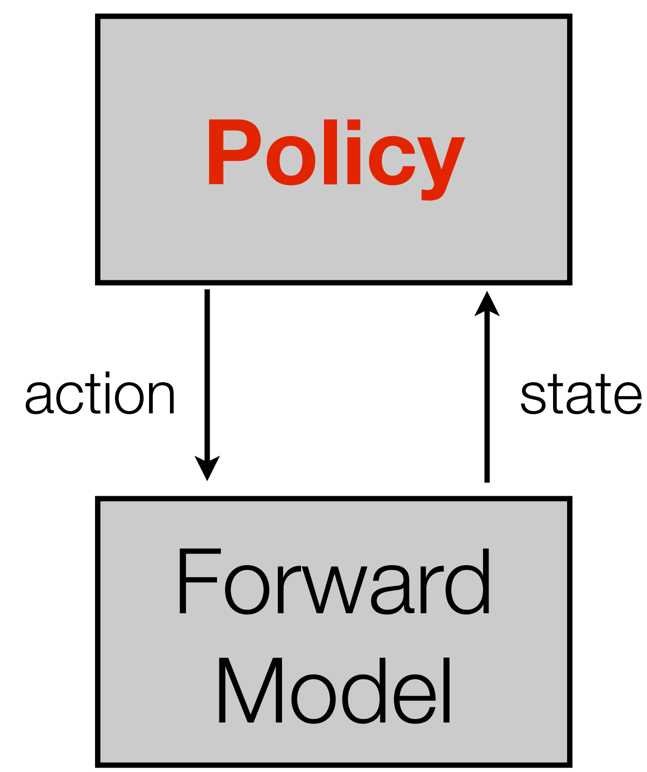
Policy Search

Goal of this Lecture

1. Step: Learn an Forward Model



2. Step: Use your favorite *Optimal Control Method* to get an optimal policy

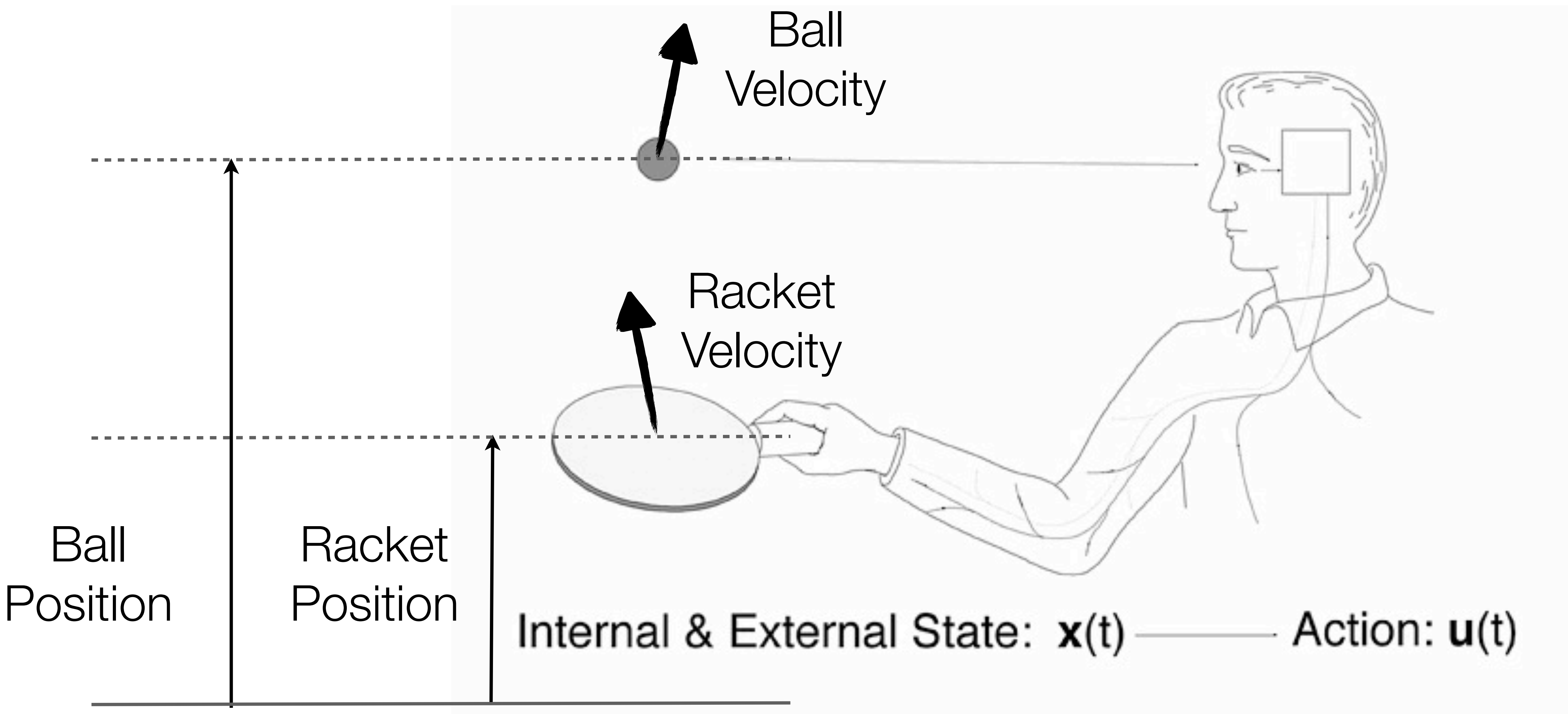


Outline of the Lecture

1. Introduction to Optimal Control
2. Solving Linear-Quadratic Optimal Control Problems
3. Optimal Control with Learned Models
4. Hot story: Marc Deisenroth’s PILCO Approach
5. Final Remarks

Example: Ball Paddling

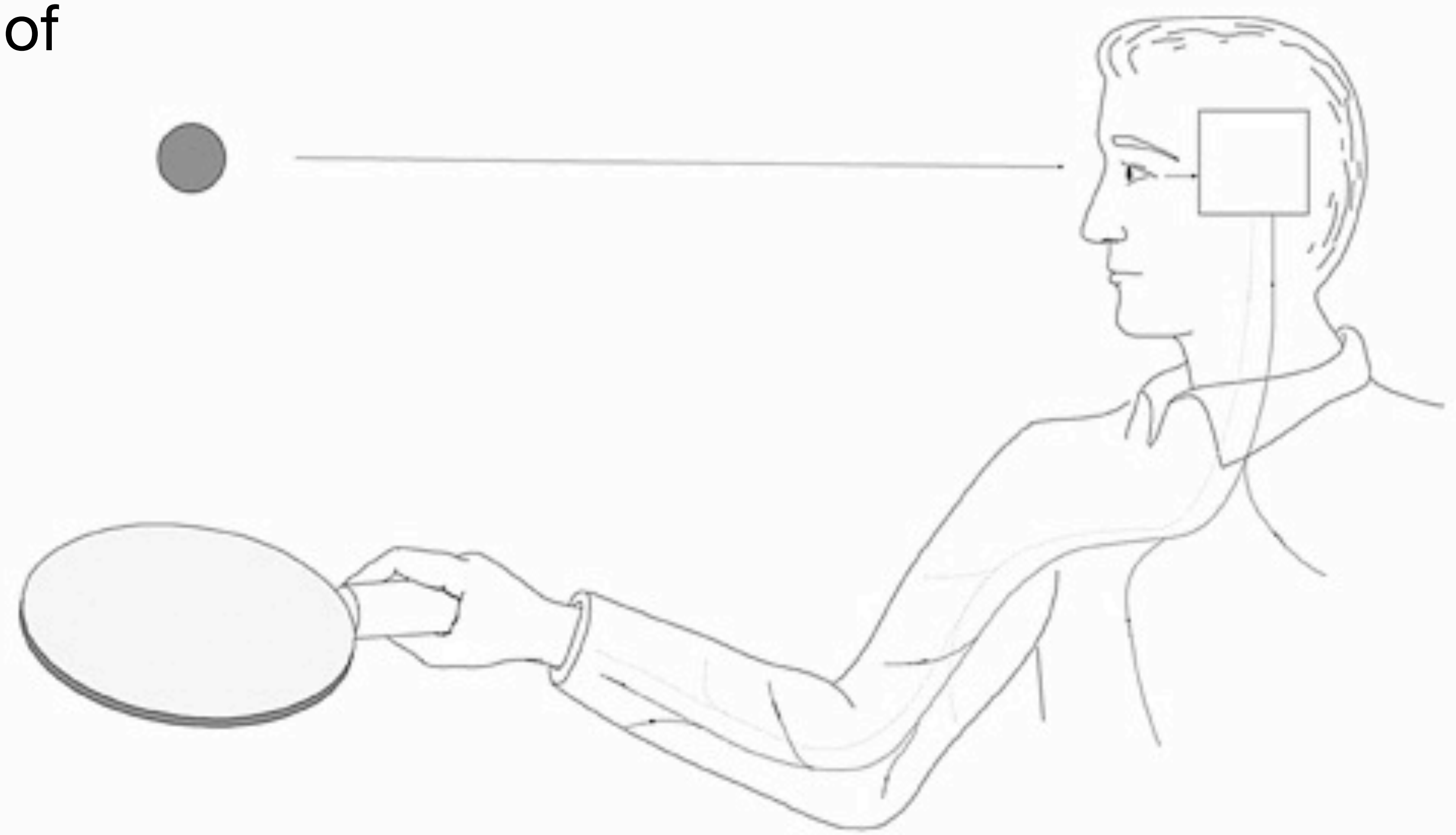
What are the states **x**?



Example: Ball Paddling

What are the actions **u**?

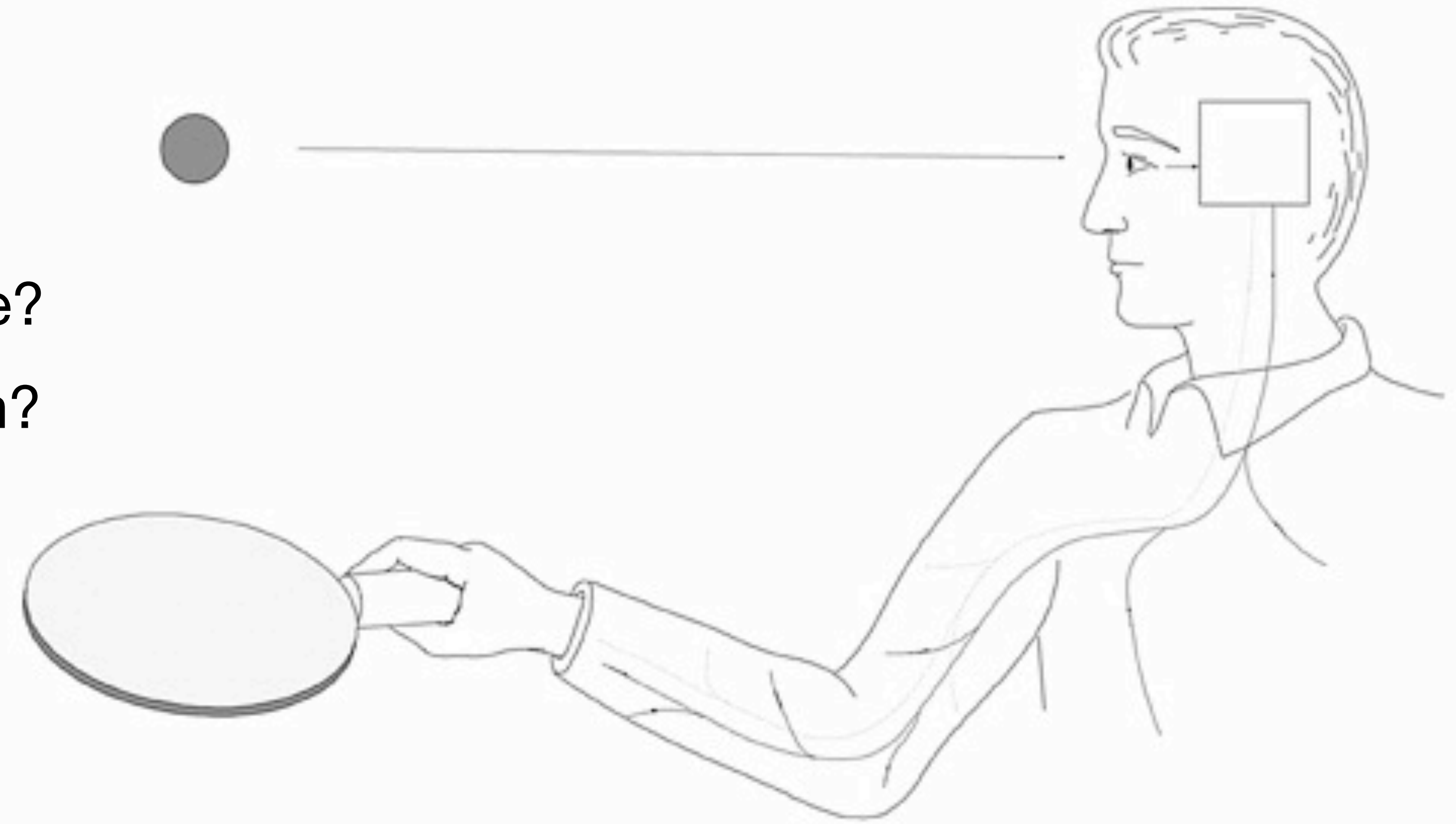
- **All motor torques?**
  - If you do not have a model ...
- **Joint Accelerations?**
  - Perfect, if you have a good model ...
  - Maybe identify the proper degrees of freedom?
- **Accelerations in Operational Space?**
  - **Ideally!**
  - ... but only if you have a good operational space control law!



Example: Ball Paddling

What are good rewards *r*?

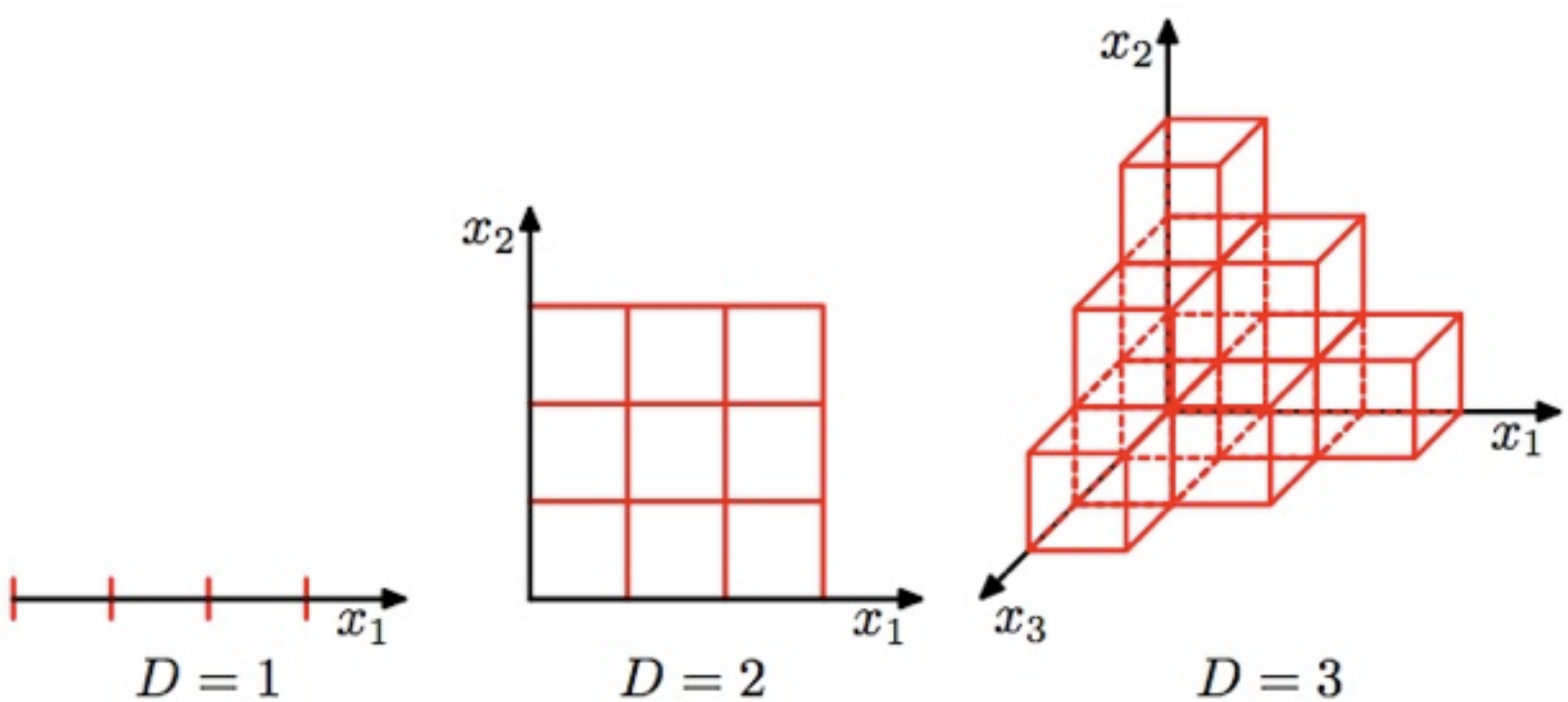
- **Task knowledge or success/failure?**
  - For some algorithms rewards in {1,0} are perfect ...
  - Real problems often require *reward shaping*...
- **What's a good reward for our problem?**
  - Height of the ball?
  - Distance between ball and the paddle?
  - Ball needs to move in a certain region?
  - All of the above?
  - Additional punishments?



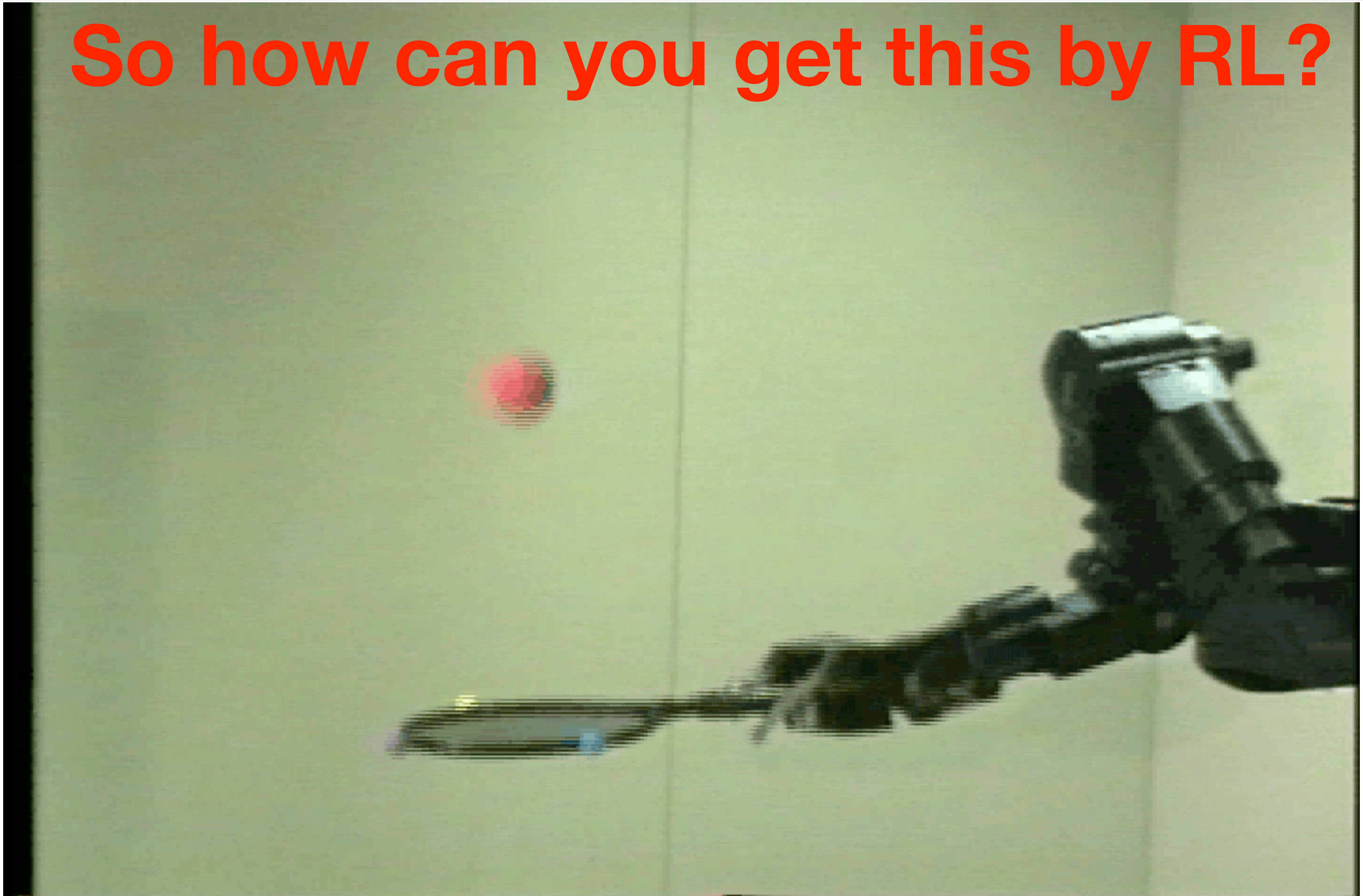
➡All of these together do the job!

So can we get this to work?

- The state space has at least 12 dimensions.
- The action space has at least 3 dimensions.
- Can discretizations deal with such spaces?
- No! Finding an Optimal Value Function is limited by the curse of dimensionality.



Example: Real world application...



Outline of the Lecture

1. Introduction to Optimal Control
2. Solving Linear-Quadratic Optimal Control Problems
3. Optimal Control with Learned Models
4. Hot story: Marc Deisenroth's PILCO Approach
5. Final Remarks

Optimal Control Goal

- The goal of optimal control is find a policy ***u*** ~ π(***x***) such that

$$J(\pi) = \lim_{T\to\infty} \frac{1}{T} E\left\{\sum_{k=0}^{T} r(\mathbf{x}_t, \mathbf{u}_t)\right\}$$

is maximal for a given

reward function such as $r(\mathbf{x}, \mathbf{u}) = -\mathbf{x}^T \mathbf{Q}\mathbf{x} - \mathbf{u}^T \mathbf{R}\mathbf{u}$

system: $\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ ~~$+ \mathbf{c}$~~ For Simplicity of Derivation! Nothing Changes!

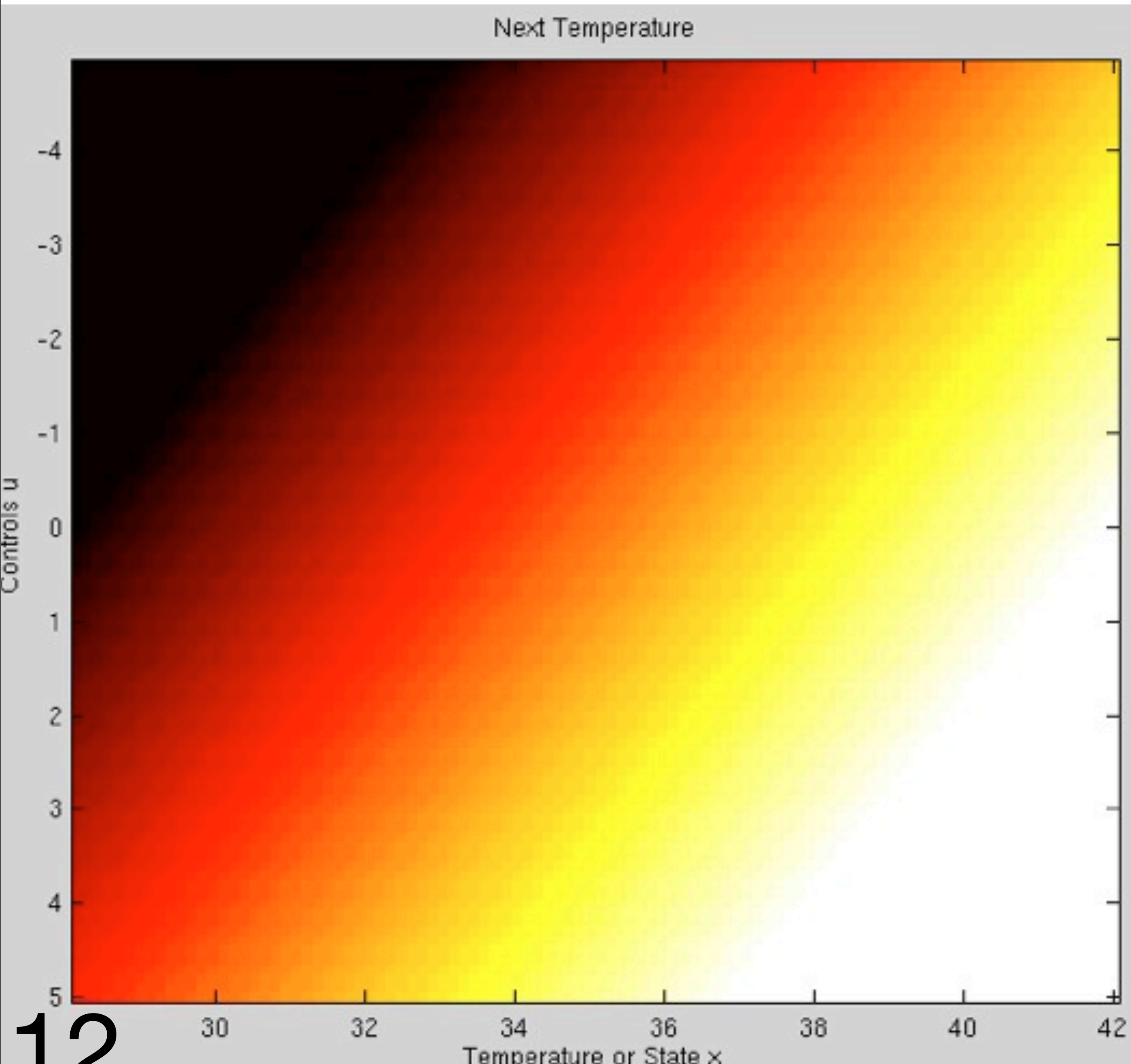
...so how do we solve this?

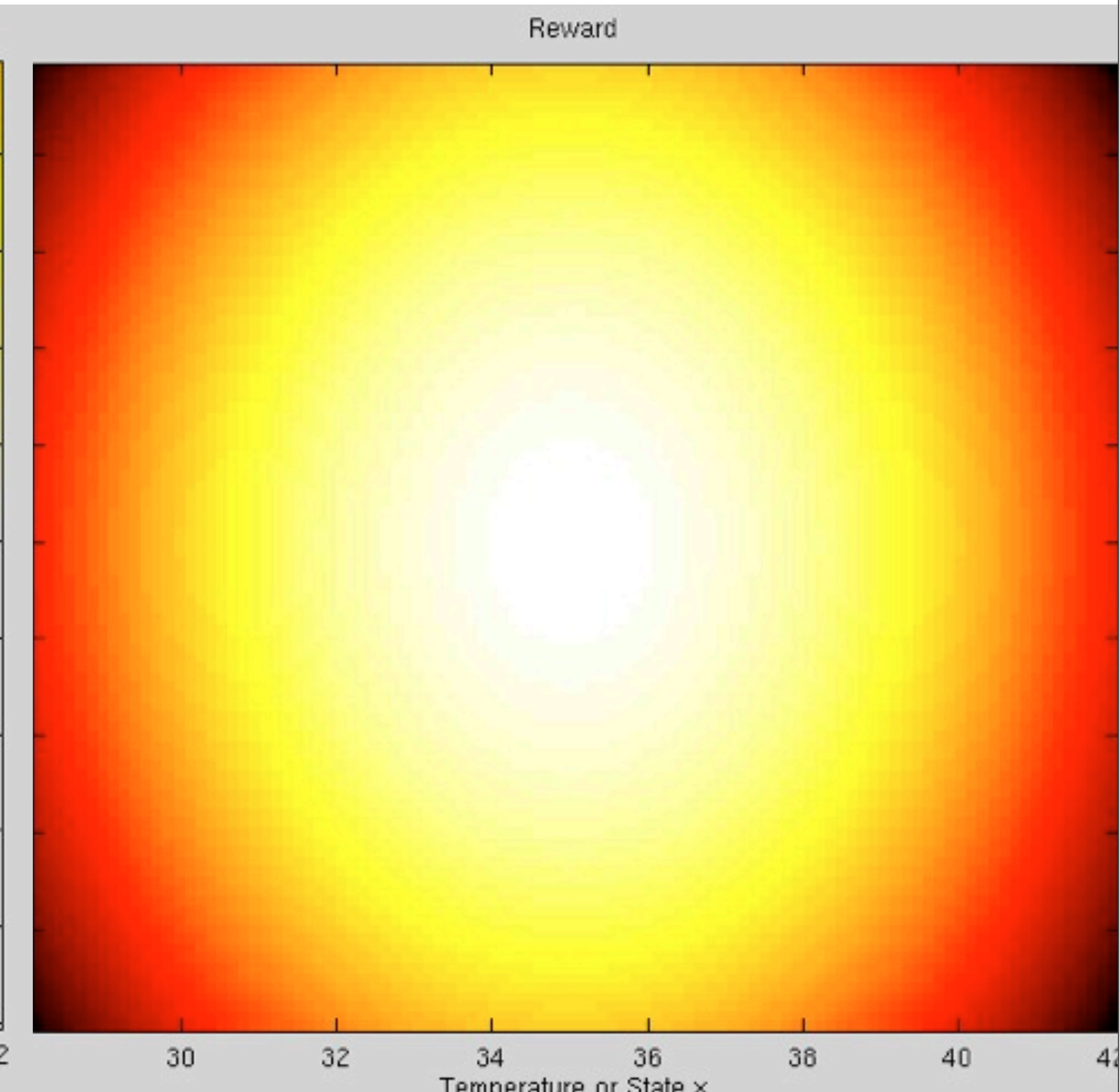
!

Plot the Problem

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \cancel{\mathbf{c}}$$

$$r(\mathbf{x}, \mathbf{u}) = -\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{u}^T\mathbf{R}\mathbf{u}$$





Bellman' Recipe: Steps 1+2

1. At the last step, we have the value function

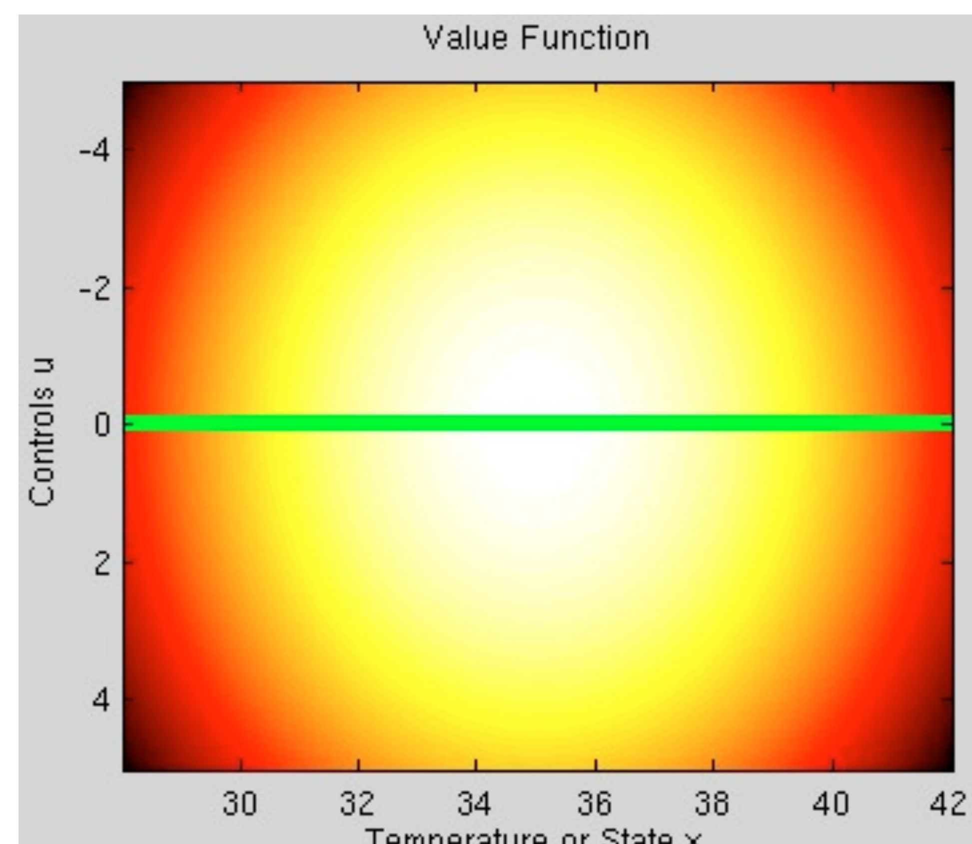
$$V_T^*(\mathbf{x}) = 0$$

2. For *t=T-1*, compute optimal policy such that

$$\pi_t^*(u|x) = \operatorname{argmax}_\pi \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\}$$

determined by

$$\begin{aligned} \frac{d}{d\mathbf{u}} \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\} &= 0 \\ \frac{d}{d\mathbf{u}} \left\{ -\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{u}^T\mathbf{R}\mathbf{u}) \right\} &= 0 \\ \mathbf{u}^* &= 0 \end{aligned}$$



Bellman' Recipe: Step 3+4

3.Obtain next value function $V_t^*(x) = \max_\pi \left\{ r(\mathbf{x}, \mathbf{u}) + V_{t+1}^*(f(\mathbf{x}, \mathbf{u})) \right\}$

$$
\begin{aligned}
V_{t+1}^*(\mathbf{x}) &= r(\mathbf{x}, \mathbf{u}^*) + V_{t+1}^*(f(\mathbf{x}, \mathbf{u}^*)) \\
&= -\mathbf{x}^T \mathbf{Q} \mathbf{x} - \mathbf{u}^{*T} \mathbf{R} \mathbf{u}^* \\
&= -\mathbf{x}^T \mathbf{Q} \mathbf{x}
\end{aligned}
$$

4.**As not converged, go back to Step 2.**

Bellman' Recipe: Step 2 *again*!

2.For *t<T-1*, compute optimal policy such that

$$\pi_t^*(u|x) = \text{argmax}_\pi \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\}$$

determined by

$$\begin{aligned}
\frac{d}{d\mathbf{u}} \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\} &= 0 \\
\frac{d}{d\mathbf{u}} \left\{ -\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{u}^T\mathbf{R}\mathbf{u} - f(\mathbf{x},\mathbf{u})^T\mathbf{P}_{t+1}f(\mathbf{x},\mathbf{u}) \right\} &= 0 \\
\frac{d}{d\mathbf{u}} \left\{ -\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{u}^T\mathbf{R}\mathbf{u} - (\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u})^T\mathbf{P}_{t+1}(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u}) \right\} &= 0 \\
\frac{d}{d\mathbf{u}} \left\{ -\mathbf{R}\mathbf{u} - \mathbf{B}^T\mathbf{P}_{t+1}(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u}) \right\} &= 0
\end{aligned}$$

which implies

Policy Parameters!

$$\mathbf{u}^* = -(\mathbf{R}+\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A}\mathbf{x} = \boldsymbol{\theta}_t\mathbf{x}$$

!

Bellman’ Recipe: Step 3+4

3.Obtain next value function $V_t^*(x) = \max_\pi \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\}$

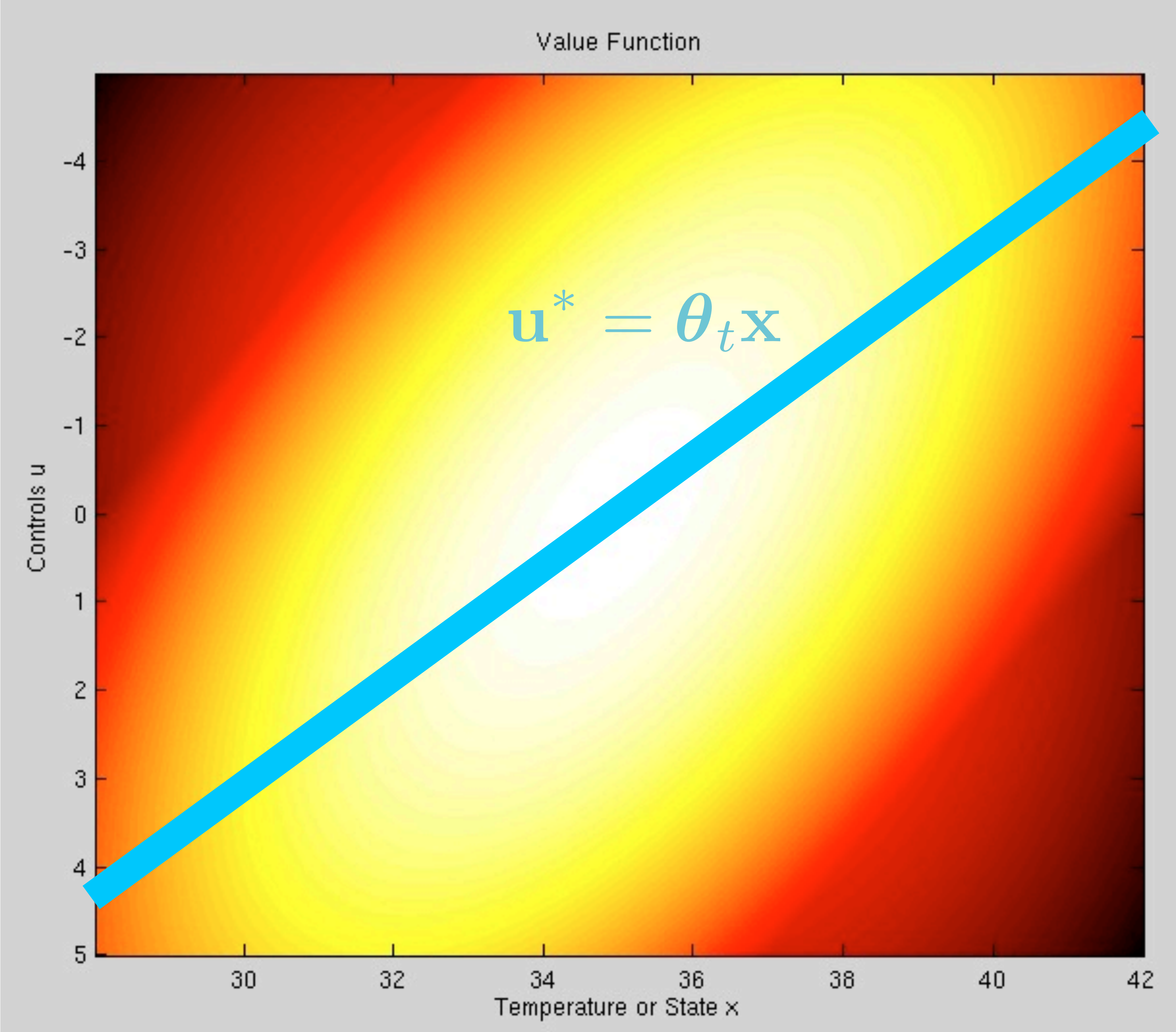
$$\begin{aligned} V_t^*(\mathbf{x}) &= r(\mathbf{x},\mathbf{u}^*) + V_{t+1}^*(\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}^*) \\ &= -\mathbf{x}^T\mathbf{Q}\mathbf{x} - (\boldsymbol{\theta}_t\mathbf{x})^T\mathbf{R}(\boldsymbol{\theta}_t\mathbf{x}) \\ &\quad -(\mathbf{A}\mathbf{x} + \mathbf{B}\boldsymbol{\theta}_{t+1}\mathbf{x})^T\mathbf{P}_{t+1} + (\mathbf{A}\mathbf{x} + \mathbf{B}\boldsymbol{\theta}_t\mathbf{x}) \\ &= -\mathbf{x}^T[\mathbf{Q} - \boldsymbol{\theta}_t^T\mathbf{R}\boldsymbol{\theta}_t \\ &\quad -(\mathbf{A} + \mathbf{B}\boldsymbol{\theta}_t)^T\mathbf{P}_{t+1} + (\mathbf{A} + \mathbf{B}\boldsymbol{\theta}_t)]\mathbf{x} \\ &= -\mathbf{x}^T\mathbf{P}_t\mathbf{x} \end{aligned}$$

4. **We have a converged to *Recursion*:**

$$\mathbf{P}_t = -\mathbf{Q} - \boldsymbol{\theta}_{t+1}^T\mathbf{R}\boldsymbol{\theta}_{t+1} - (\mathbf{A} + \mathbf{B}\boldsymbol{\theta}_{t+1})^T\mathbf{P}_{t+1} + (\mathbf{A} + \mathbf{B}\boldsymbol{\theta}_{t+1})$$

$$\boldsymbol{\theta}_t = -(\mathbf{R} + \mathbf{B}^T\mathbf{P}_{t+1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{P}_{t+1}\mathbf{A}$$

Optimal Solution



Outline of the Lecture

1. Introduction to Optimal Control

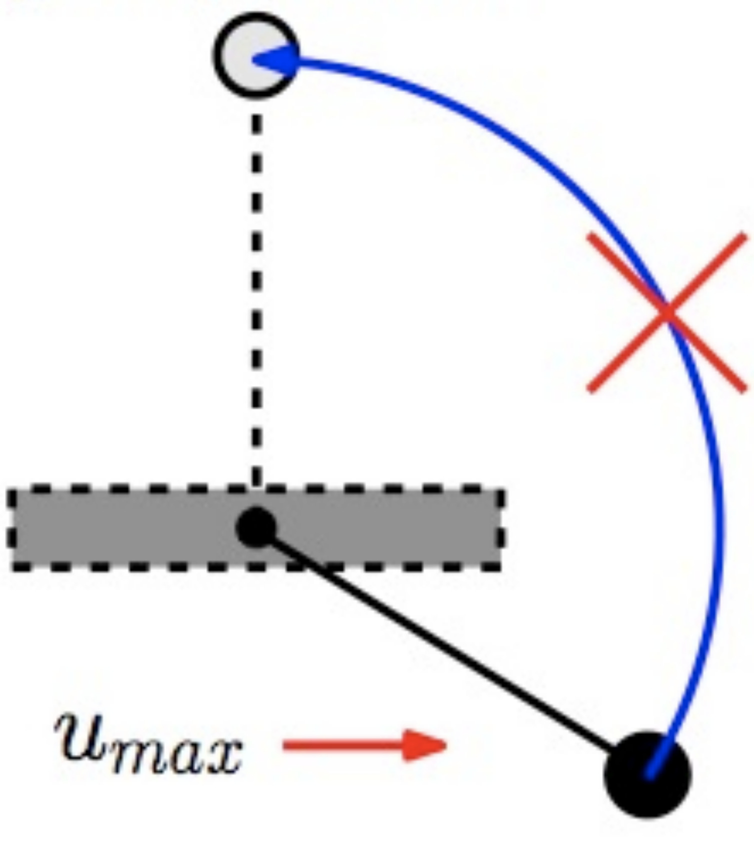
2. Solving Linear-Quadratic Optimal Control Problems

3. Optimal Control with Learned Models

4. Hot story: Marc Deisenroth's PILCO Approach

5. Final Remarks

Example: Swing-Up



System

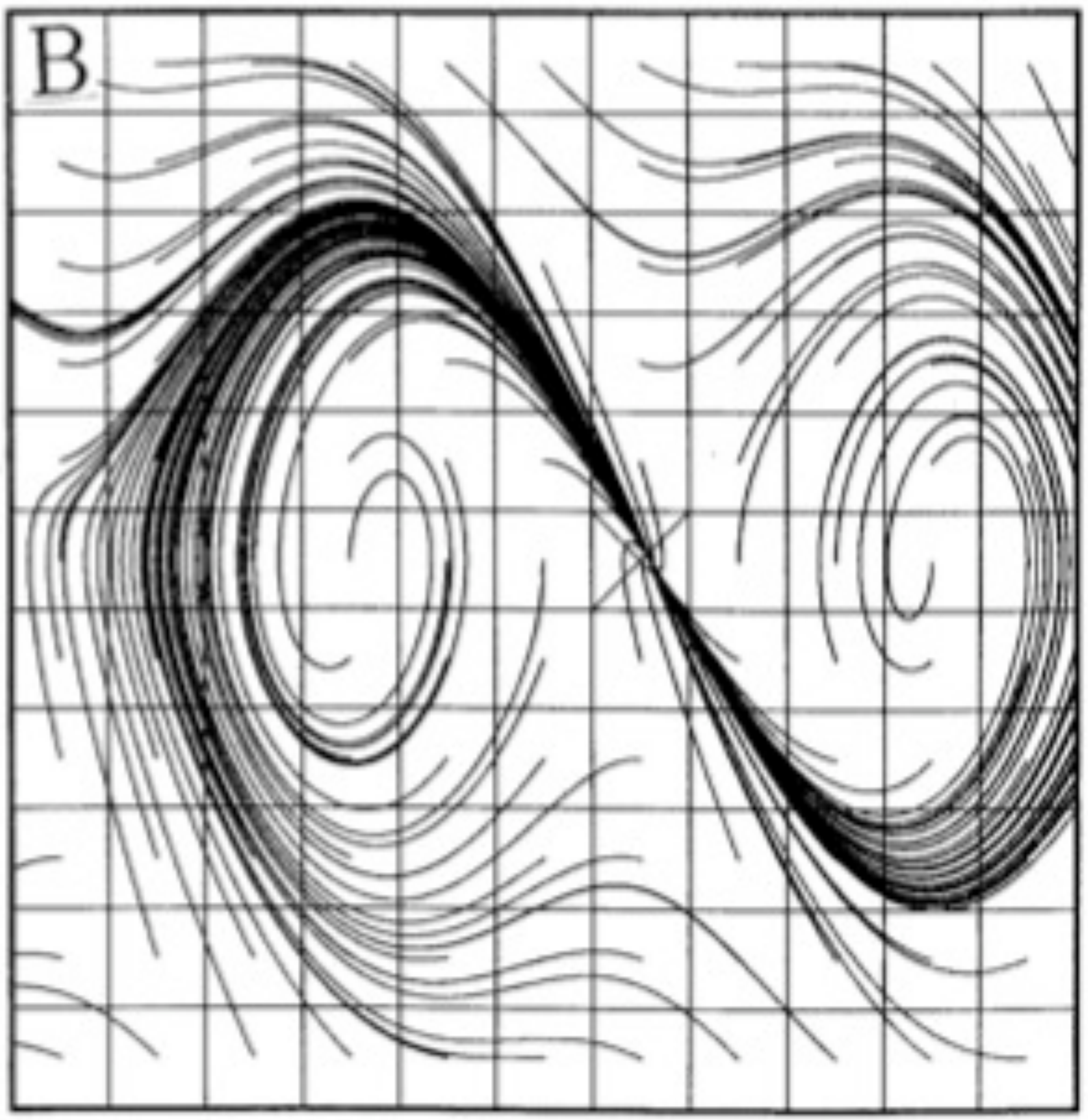
$$\ddot{\varphi}(t) = \frac{-\mu\dot{\varphi}(t) + mgl\sin(\varphi(t)) + u(t)}{ml^2}$$

$$\mathbf{x}_{k+1} := \begin{bmatrix} \varphi_{k+1} \\ \dot{\varphi}_{k+1} \end{bmatrix} = \begin{bmatrix} \varphi_k + \Delta_t\dot{\varphi}_k + \frac{\Delta_t^2}{2}\ddot{\varphi}_k \\ \dot{\varphi}_k + \Delta_t\ddot{\varphi}_k \end{bmatrix}$$

Reward

$$r(\mathbf{x}, \mathbf{u}) = -\mathbf{x}_k^T \operatorname{diag}(1, 0.1)\mathbf{x}_k - 0.2\, u_k^2$$

Possible: Learn Solutions only where needed!



If you know places where we start...

... we can just look ahead!

Atkeson & Schaal, 1995

Local Solutions

- Every smooth function can be modeled with a Taylor expansion

$$f(\mathbf{x}) = f(\mathbf{a}) + \left.\frac{df}{d\mathbf{x}}\right|_{\mathbf{x}=\mathbf{a}} (\mathbf{x}-\mathbf{a}) + \frac{1}{2}(\mathbf{x}-\mathbf{a})^T \left.\frac{d^2 f}{d\mathbf{x}^2}\right|_{\mathbf{x}=\mathbf{a}} (\mathbf{x}-\mathbf{a}) + \dots$$

- Hence, we can also approximate:

$$\begin{aligned} \mathbf{x}' &\approx f(\hat{\mathbf{x}}, \hat{\mathbf{u}}) + \left.\frac{df}{d\mathbf{x}}\right|_{\hat{\mathbf{x}},\hat{\mathbf{u}}} (\mathbf{x}-\hat{\mathbf{x}}) + \left.\frac{df}{d\mathbf{u}}\right|_{\hat{\mathbf{x}},\hat{\mathbf{u}}} (\mathbf{u}-\hat{\mathbf{u}}) \\ &= \mathbf{a}_t^0 + \mathbf{A}_t(\mathbf{x}-\hat{\mathbf{x}}) + \mathbf{B}_t(\mathbf{u}-\hat{\mathbf{u}}) \end{aligned}$$

$$r \approx r(\hat{\mathbf{x}}, \hat{\mathbf{u}}) + \begin{bmatrix} \frac{dr}{d\mathbf{x}} \\ \frac{dr}{d\mathbf{u}} \end{bmatrix}^T \begin{bmatrix} \mathbf{x}-\hat{\mathbf{x}} \\ \mathbf{u}-\hat{\mathbf{u}} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}-\hat{\mathbf{x}} \\ \mathbf{u}-\hat{\mathbf{u}} \end{bmatrix}^T \begin{bmatrix} \frac{d^2 r}{d\mathbf{x}^2} & \frac{d^2 r}{d\mathbf{x}d\mathbf{u}} \\ \frac{d^2 r}{d\mathbf{x}d\mathbf{u}} & \frac{d^2 r}{d\mathbf{u}^2} \end{bmatrix} \begin{bmatrix} \mathbf{x}-\hat{\mathbf{x}} \\ \mathbf{u}-\hat{\mathbf{u}} \end{bmatrix}$$

Atkeson & Schaal, 1995

Bellman' Recipe: Steps 1-4

1.At the last step, we have the value function

$$V_T^*(\mathbf{x}) = 0$$

2.For *t=T-1*, compute optimal policy such that

$$\pi_t^*(u|x) = \text{argmax}_\pi \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\}$$

gives $\mathbf{u} = \hat{\mathbf{u}}$.

3.Obtain next value function $V_t^*(x) = \max_\pi \left\{ r(\mathbf{x},\mathbf{u}) + V_{t+1}^*(f(\mathbf{x},\mathbf{u})) \right\}$

$$V_t^*(\mathbf{x}) \quad = \quad -r_{t+1} - (\mathbf{x} - \hat{\mathbf{x}}_t)^T \mathbf{Q}_{t+1} (\mathbf{x} - \hat{\mathbf{x}}_t)$$

4.**As not converged, go back to Step 2.**

Bellman’ Recipe: Step 2-4 *again*!

2.For *t<T-1*, compute optimal policy such that

$$\pi_t^*(u|x) = \text{argmax}_\pi \left\{ r(\mathbf{x}, \mathbf{u}) + V_{t+1}^*(f(\mathbf{x}, \mathbf{u})) \right\}$$

determined by

$$\mathbf{u}^* = \hat{\mathbf{u}}_t - (\mathbf{R}_t + \mathbf{B}_t^T \mathbf{P}_t \mathbf{B}_t)^{-1} \mathbf{B}_t^T \mathbf{P}_{t+1} (\mathbf{a}_t^0 + \mathbf{A}_t (\mathbf{x} - \hat{\mathbf{x}}_t)) = \boldsymbol{\theta}_t^1 (\mathbf{x} - \hat{\mathbf{x}}_t) + \boldsymbol{\theta}_t^0$$

3.Obtain the recursions

$$\begin{aligned} \boldsymbol{\theta}_t^1 &= \hat{\mathbf{u}}_t - (\mathbf{R}_t + \mathbf{B}_t^T \mathbf{P}_{t+1} \mathbf{B}_t)^{-1} \mathbf{B}_t^T \mathbf{P}_{t+1} \mathbf{A}_t (\mathbf{x} - \hat{\mathbf{x}}_t) \\ \boldsymbol{\theta}_t^0 &= \hat{\mathbf{u}}_t - (\mathbf{R}_t + \mathbf{B}_t^T \mathbf{P}_{t+1} \mathbf{B}_t)^{-1} \mathbf{B}_t^T \mathbf{P}_{t+1} \mathbf{a}_t^0 \\ \mathbf{P}_t &= -\mathbf{Q}_t - \boldsymbol{\theta}_t^{T} \mathbf{R}_t \boldsymbol{\theta}_t^1 + (\mathbf{A} + \mathbf{B}_t \boldsymbol{\theta}_t^1) \mathbf{P}_{t+1} (\mathbf{A} + \mathbf{B}_t \boldsymbol{\theta}_t^1) \end{aligned}$$

How do we get the Optimal Policy

1. Forward Propagation: Run Simulator to Obtain Linearizations

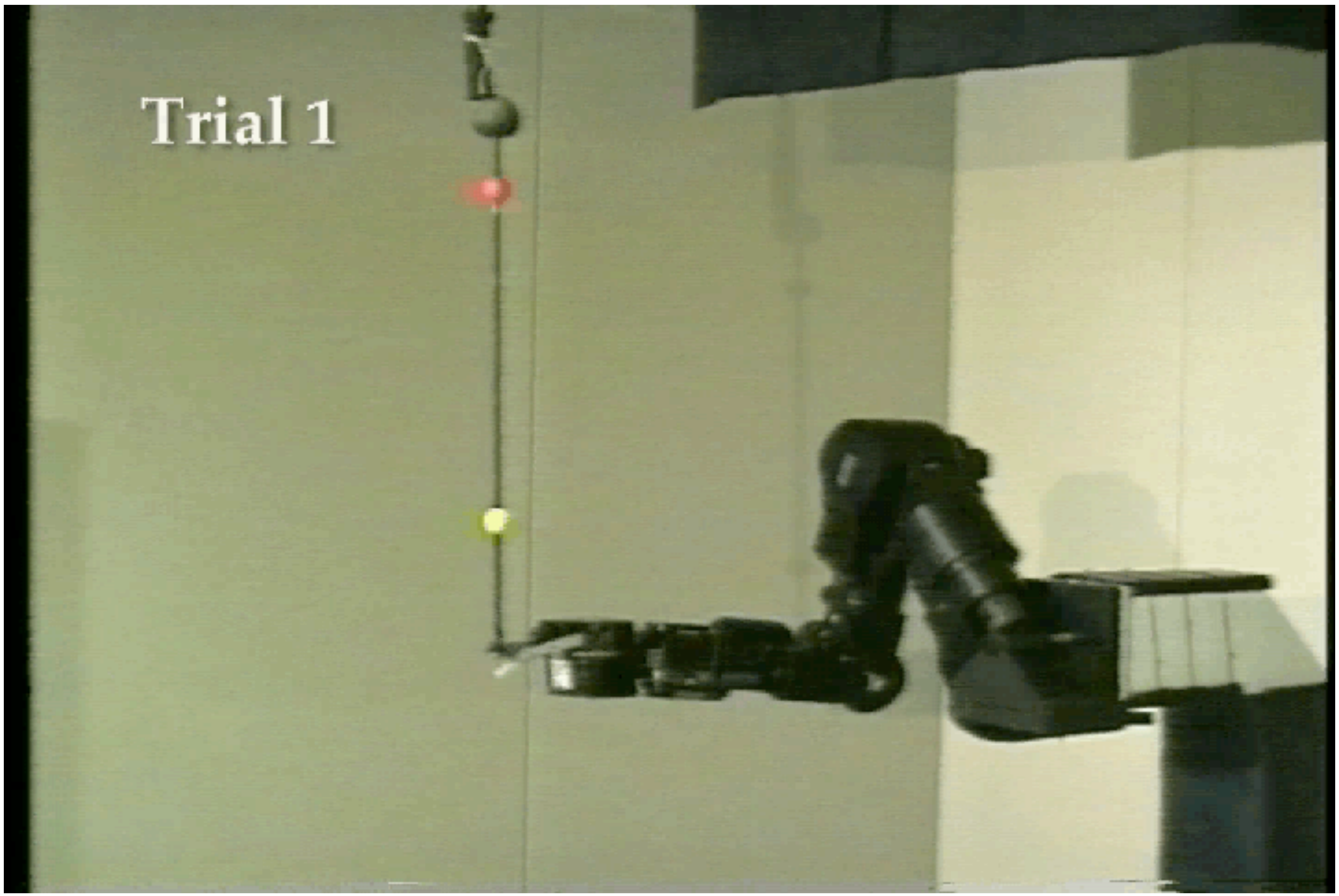
2. Backward Solution: Compute Optimal Control Law

3. If not converged, go to 1.

Model Learning with subsequent Policy Optimization

Atkeson & Schaal, 1996; Schaal, 1997

Model Learning with subsequent Policy Optimization



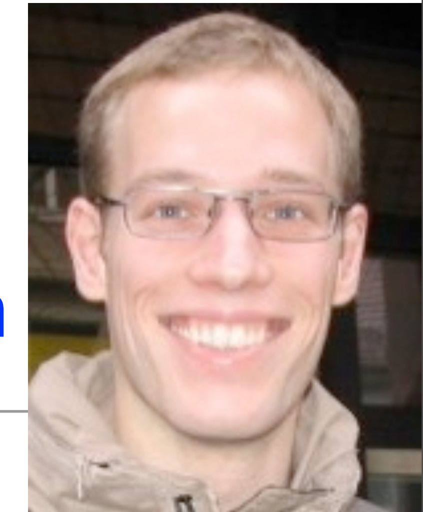
Atkeson & Schaal, 1996; Schaal, 1997

Outline of the Lecture

1. Introduction to Optimal Control
2. Solving Linear-Quadratic Optimal Control Problems
3. Optimal Control with Learned Models
4. Hot story: Marc Deisenroth’s PILCO Approach
5. Final Remarks

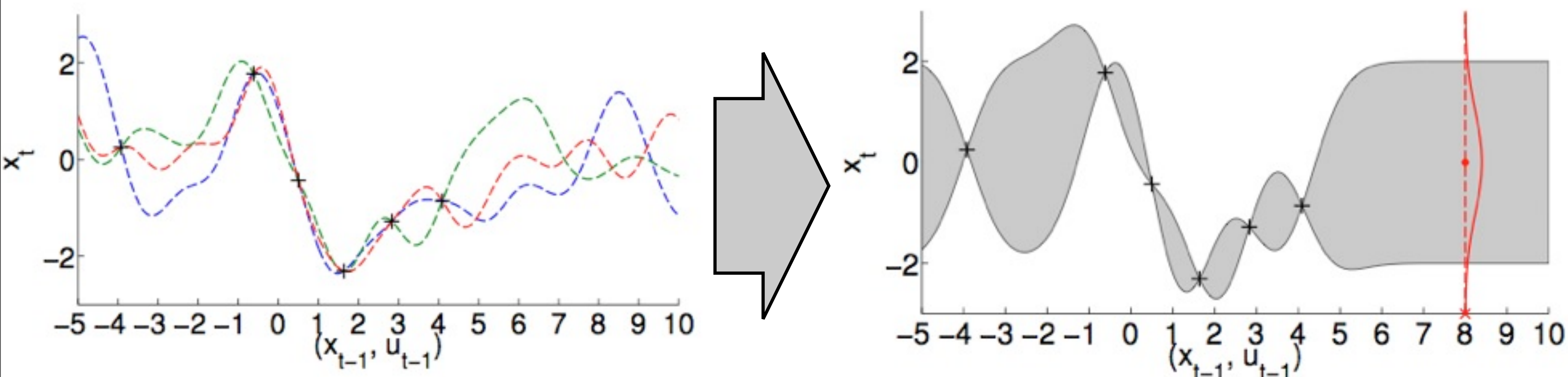
Probabilistic Forward Models

Marc
Deisenroth



- Many forward models explain measured data.
- Choosing a bad model destroys will cause an optimization bias.
- Can we average ensure robustness towards bad approximations?

➡ Yes! Even in a Bayesian way with Gaussian Process Regression!



Deisenroth, Fox, Rasmussen, R:SS 2011

Basic Idea

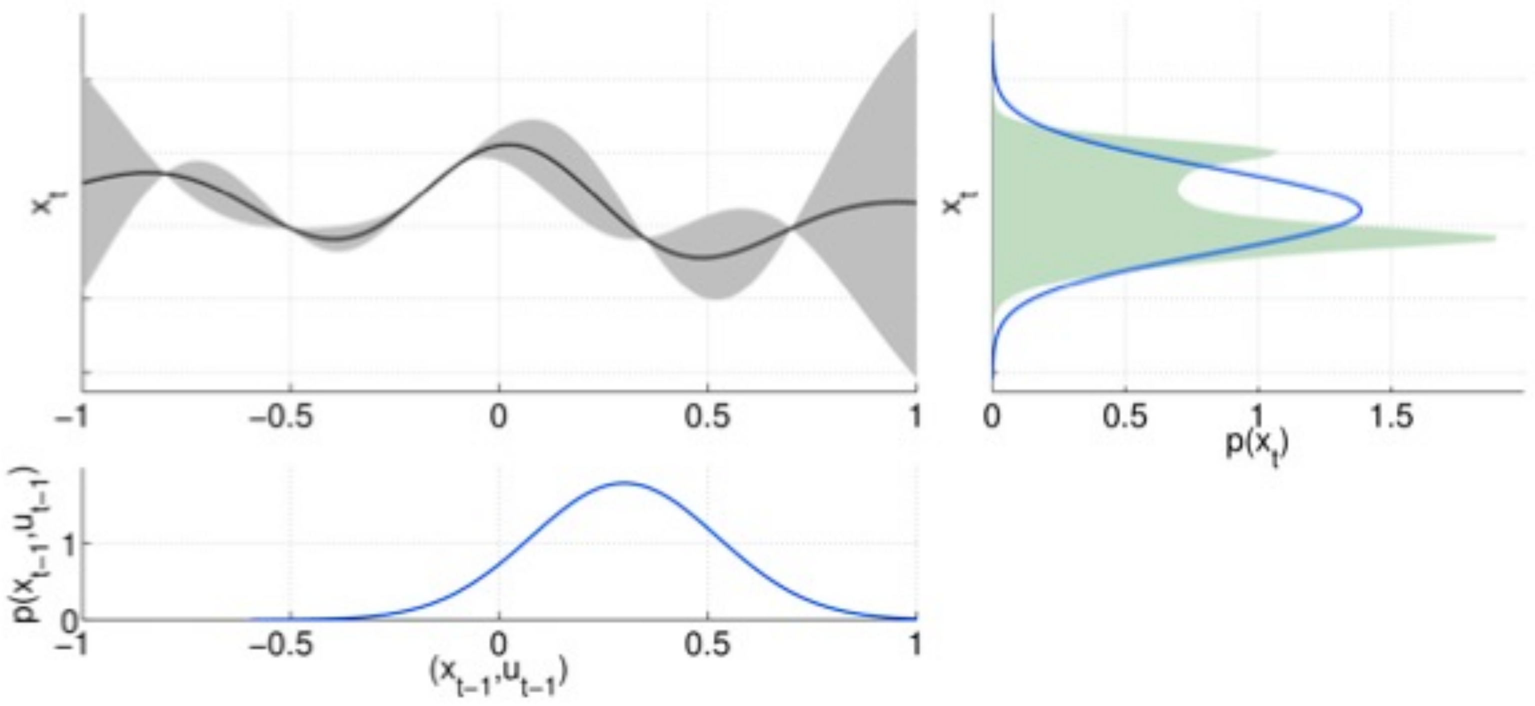
Marc Deisenroth

- With a GP, you can compute the distributions over all future states based on all forward models weighted by their likelihood.

$$p(\mathbf{x}_1), \ldots, p(\mathbf{x}_T)$$

- The propagation is still approximate and done by moment matching
- From the distribution:
  - Expected Return: $J^\pi(\boldsymbol{\theta})$
  - Gradient: $\mathrm{d}J^\pi(\boldsymbol{\theta})/\mathrm{d}\boldsymbol{\theta}$

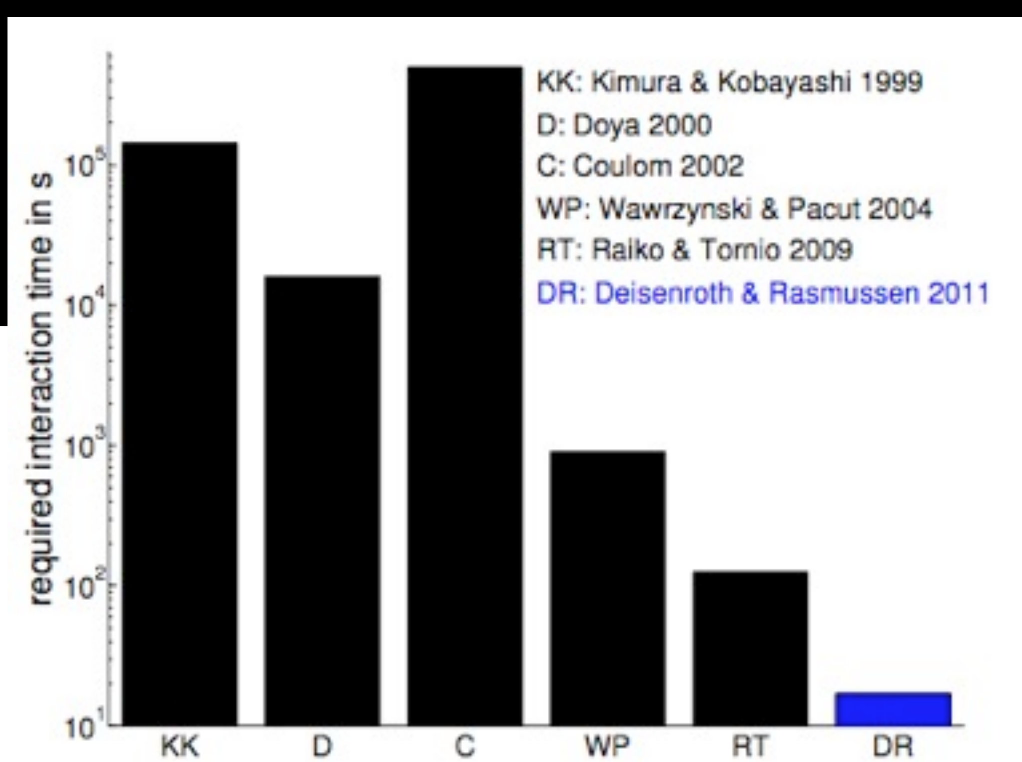
➡ We can do policy updates!

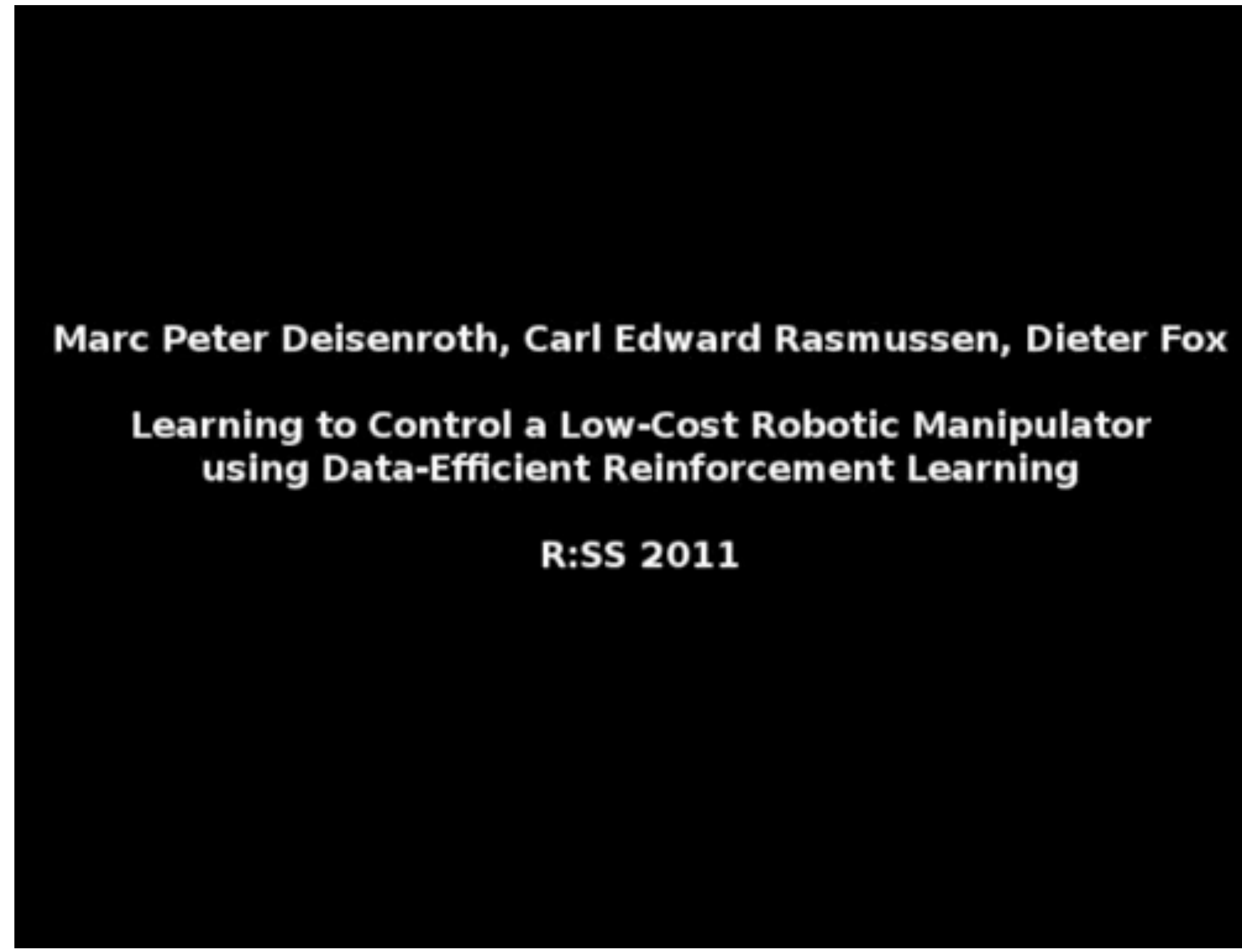


Deisenroth, Fox, Rasmussen, R:SS 2011

Applications

Marc Deisenroth





Deisenroth, Fox, Rasmussen, R:SS 2011

Outline of the Lecture

1. Introduction to Optimal Control

2. Solving Linear-Quadratic Optimal Control Problems

3. Optimal Control with Learned Models

4. Hot story: Marc Deisenroth's PILCO Approach

5. Final Remarks

Conclusions

- You have learned about optimal control today!

- Only two cased are solvable: linear & discrete!
  - Linear scales but does not generalize.
  - Discrete generalizes but does not scale.

- Using Learned Models, you can compute at least optimal “policy tubes”.

- If you have many many tubes, in good regions, you have a policy.

- We will continue with Value Function and Policy Search Methods.

Further Reading

- C. G. Atkeson (1994), Using Local Trajectory Optimizers to Speed Up Global Optimization in Dynamic Programming, Proceedings, Neural Information Processing Systems, Denver, Colorado, December, 1993, In: Neural Information Processing Systems 6, J. D. Cowan, G. Tesauro, and J. Alspector, eds. Morgan Kaufmann, 1994.
- Schaal, S. (1997). “Learning from demonstration”. In: M.C. Mozer, M. Jordan, & T. Petsche (eds.), Advances in Neural Information Processing Systems 9, pp.1040-1046. Cambridge, MA: MIT Press
- Marc P. Deisenroth, Carl E. Rasmussen, Dieter Fox (2011). Learning to Control a Low-Cost Robotic Manipulator Using Data-Efficient Reinforcement Learning, Robotics: Science & Systems (RSS 2011)