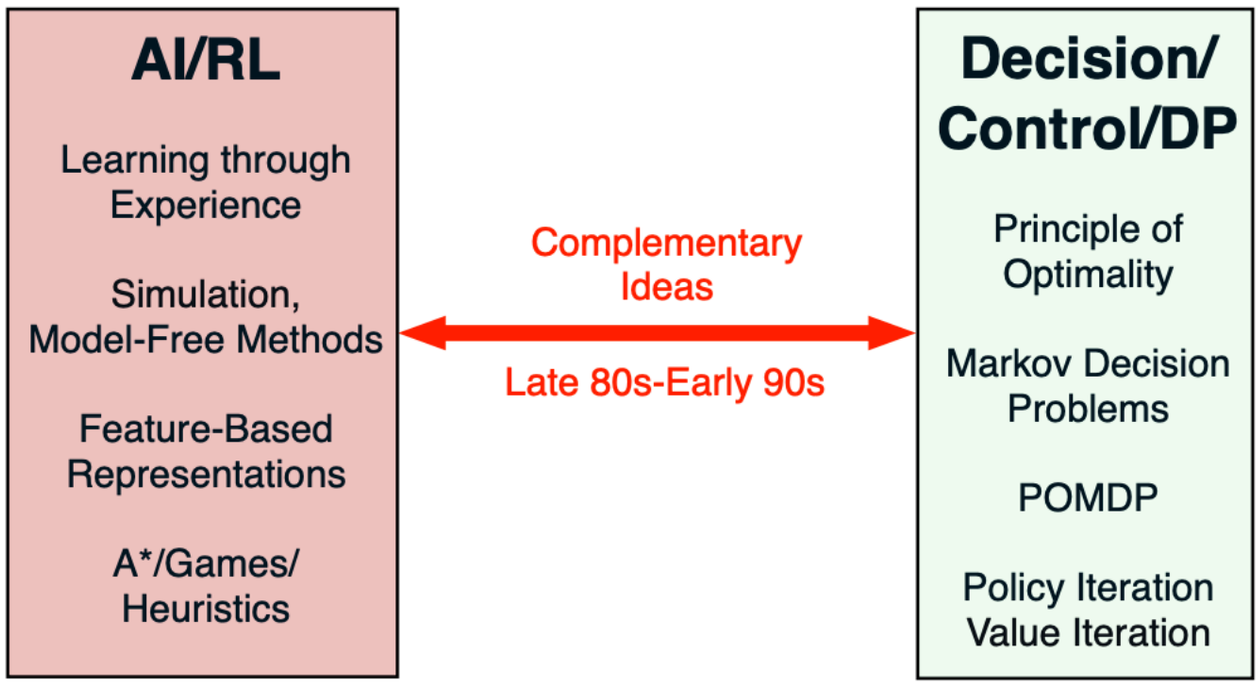
An introduction to optimal control for reinforcement learners

January, 2019
Lecture at the University of Tübingen

Jia-Jie Zhu
Max Planck Institute for Intelligent Systems
jzhu@tuebingen.mpg.de


KEEP AWAY
DANGER
CONSTRUCTION AREA
KEEP OUT
DANGER
HIGH
VOLTAGE
ROBOTICS CHALLENGE
FINALS 2015
DARPA
Safety

Reinforcement Learning (RL): A Happy Union of AI and Decision/Control Ideas

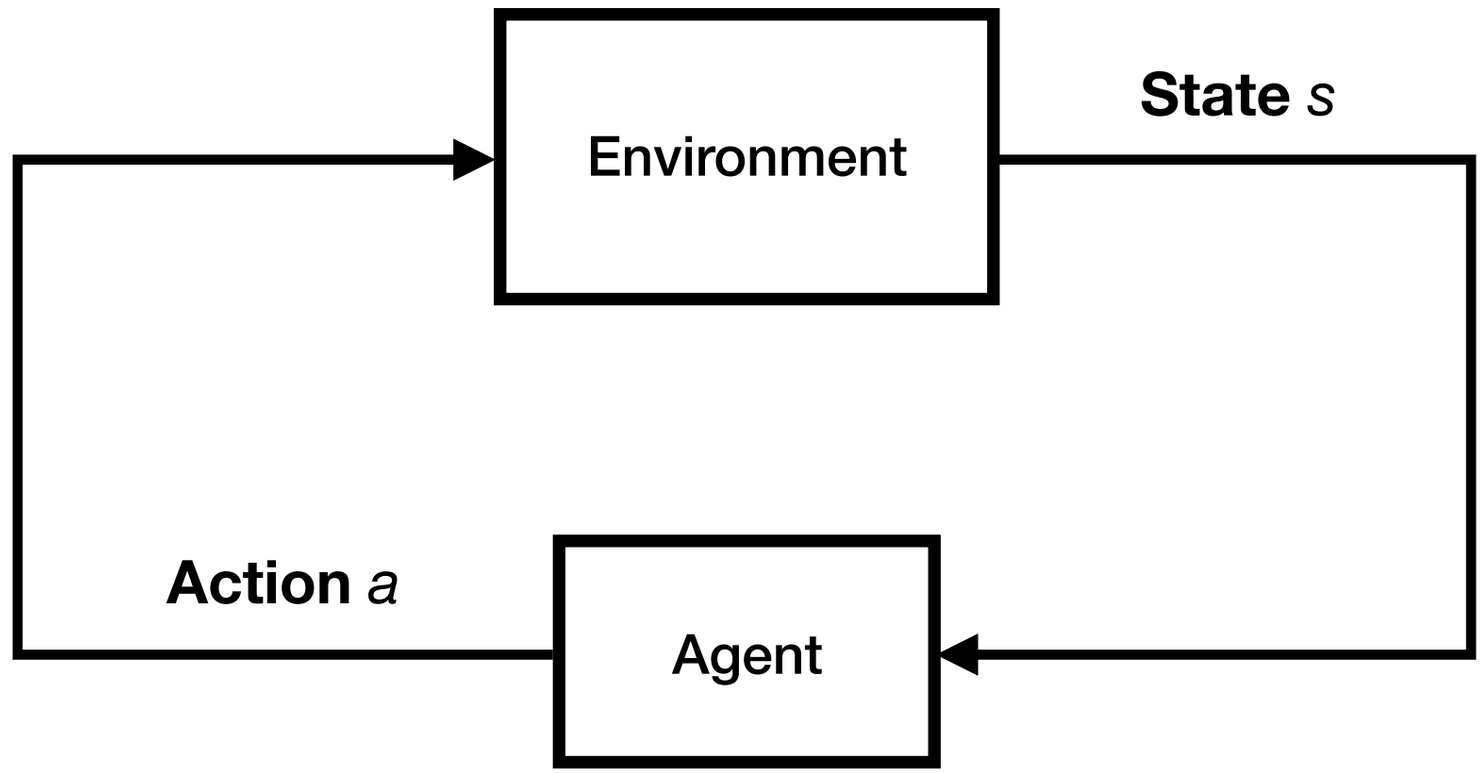


Historical highlights

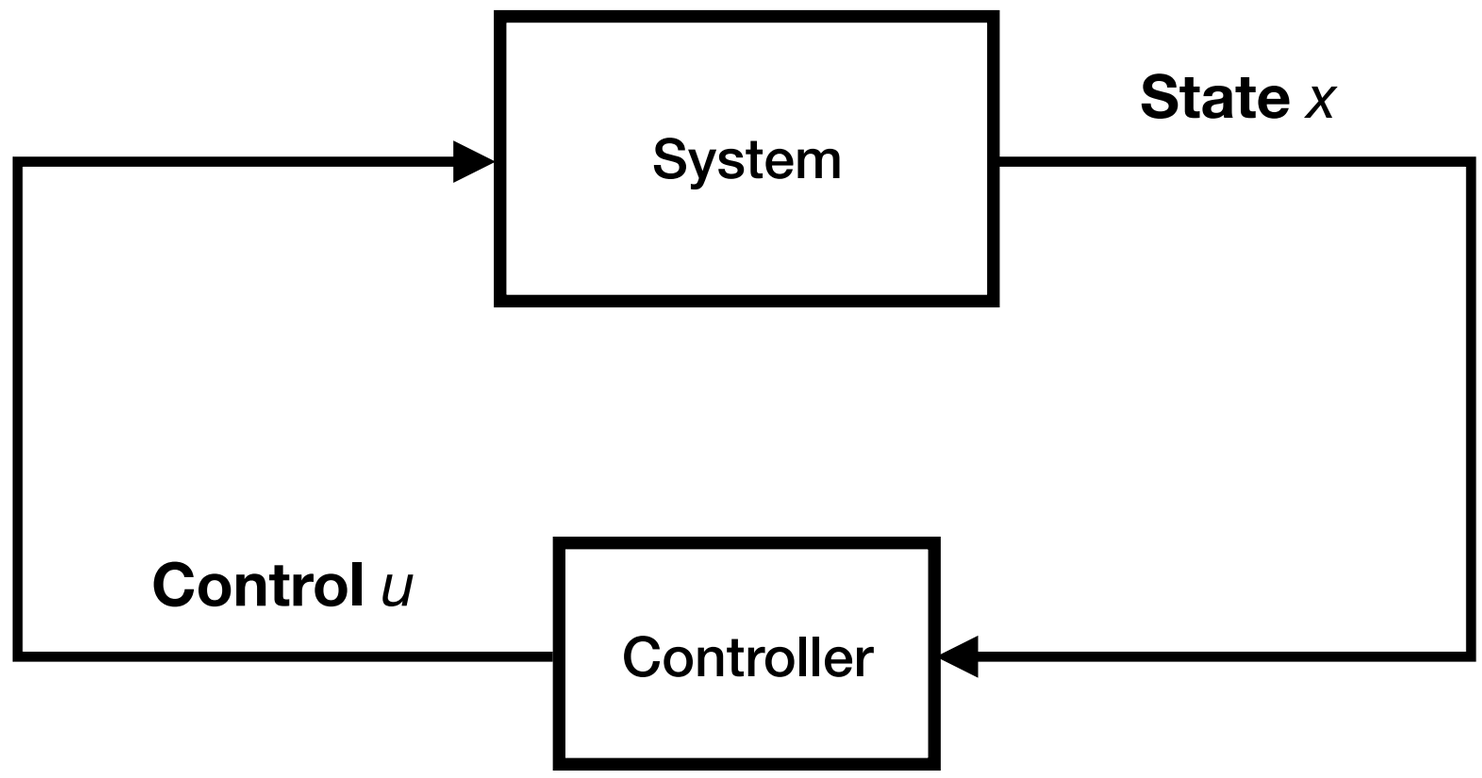
- Exact DP, optimal control (Bellman, Shannon, 1950s ...)
- First major successes: Backgammon programs (Tesauro, 1992, 1996)
- Algorithmic progress, analysis, applications, first books (mid 90s ...)
- Machine Learning, BIG Data, Robotics, Deep Neural Networks (mid 2000s ...)
- AlphaGo and Alphazero (DeepMind, 2016, 2017)

Slide adapted from D. Bertsekas

RL



Control (OC)



Terminology

RL uses Max/Value, DP uses Min/Cost

- Reward of a stage = (Opposite of) Cost of a stage.
- State value = (Opposite of) State cost.
- Value (or state-value) function = (Opposite of) Cost function.

Controlled system terminology

- Agent = Decision maker or controller.
- Action = Control.
- Environment = Dynamic system.

Methods terminology

- Learning = Solving a DP-related problem using simulation.
- Self-learning (or self-play in the context of games) = Solving a DP problem using simulation-based policy iteration.
- Planning vs Learning distinction = Solving a DP problem with model-based vs model-free simulation.

Slide adapted from D. Bertsekas

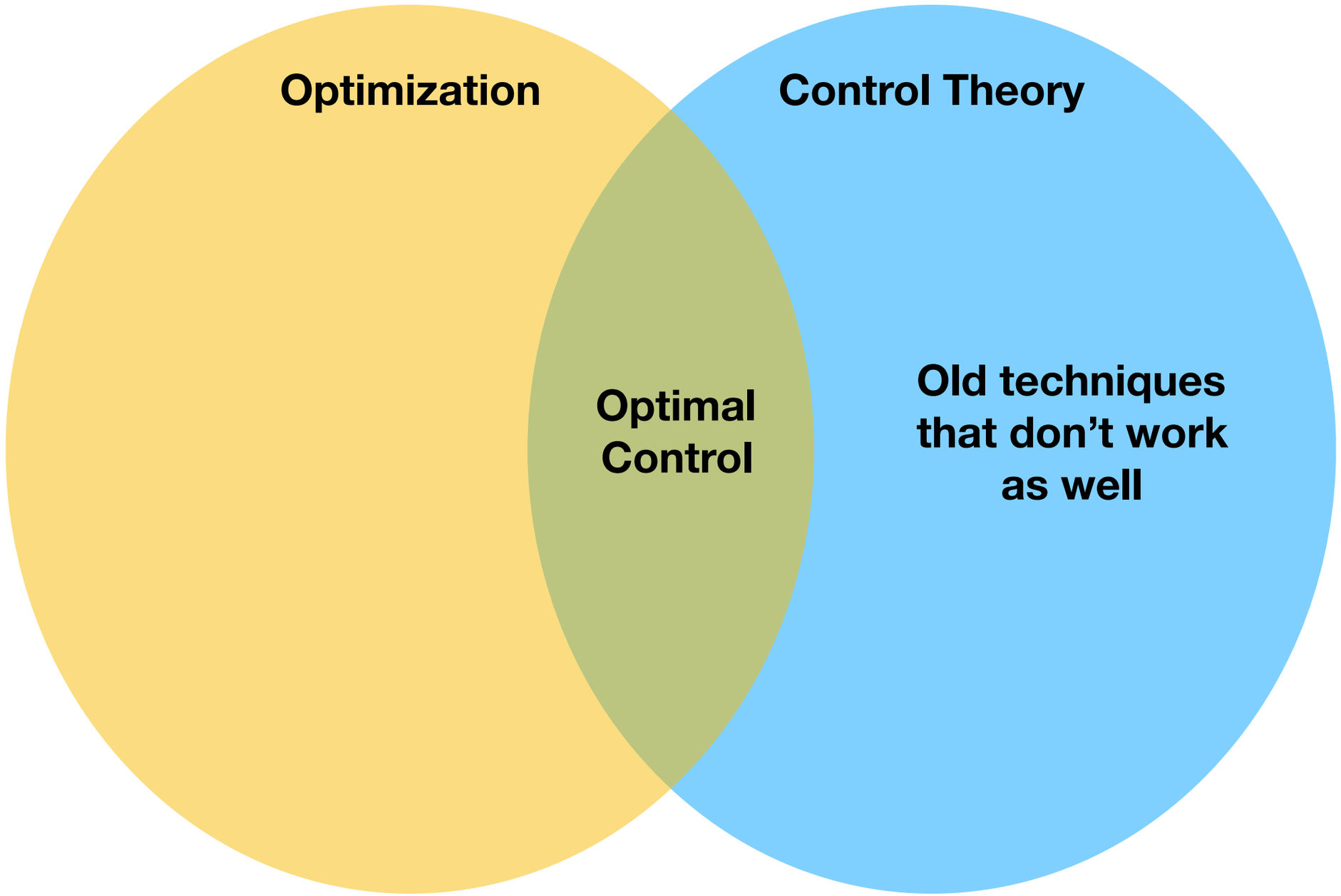
The RL problem

$$\max_{\mu} \quad \mathbb{E} \sum_{t=1}^{T} R(x_t, \mu(x_t))$$

$$\min_{u_{1:T}} \quad \sum_{t=1}^{T} - R(x_t, u_t)$$

The optimal control problem

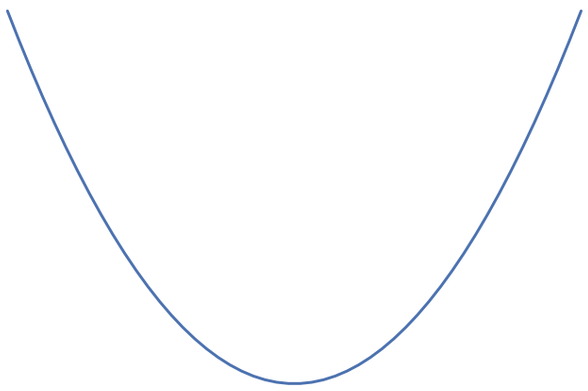
OC


Optimization
Control Theory
Optimal Control
Old techniques that don’t work as well

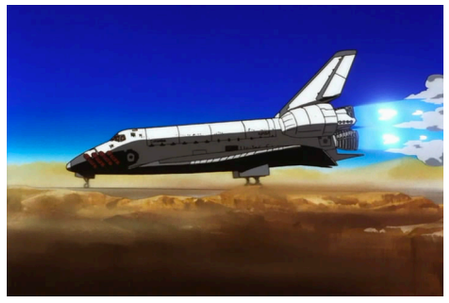
Recall
Optimization

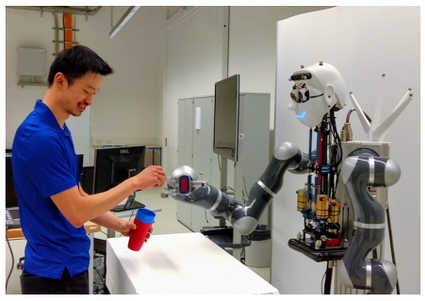
$$\min f(x)$$

Decision

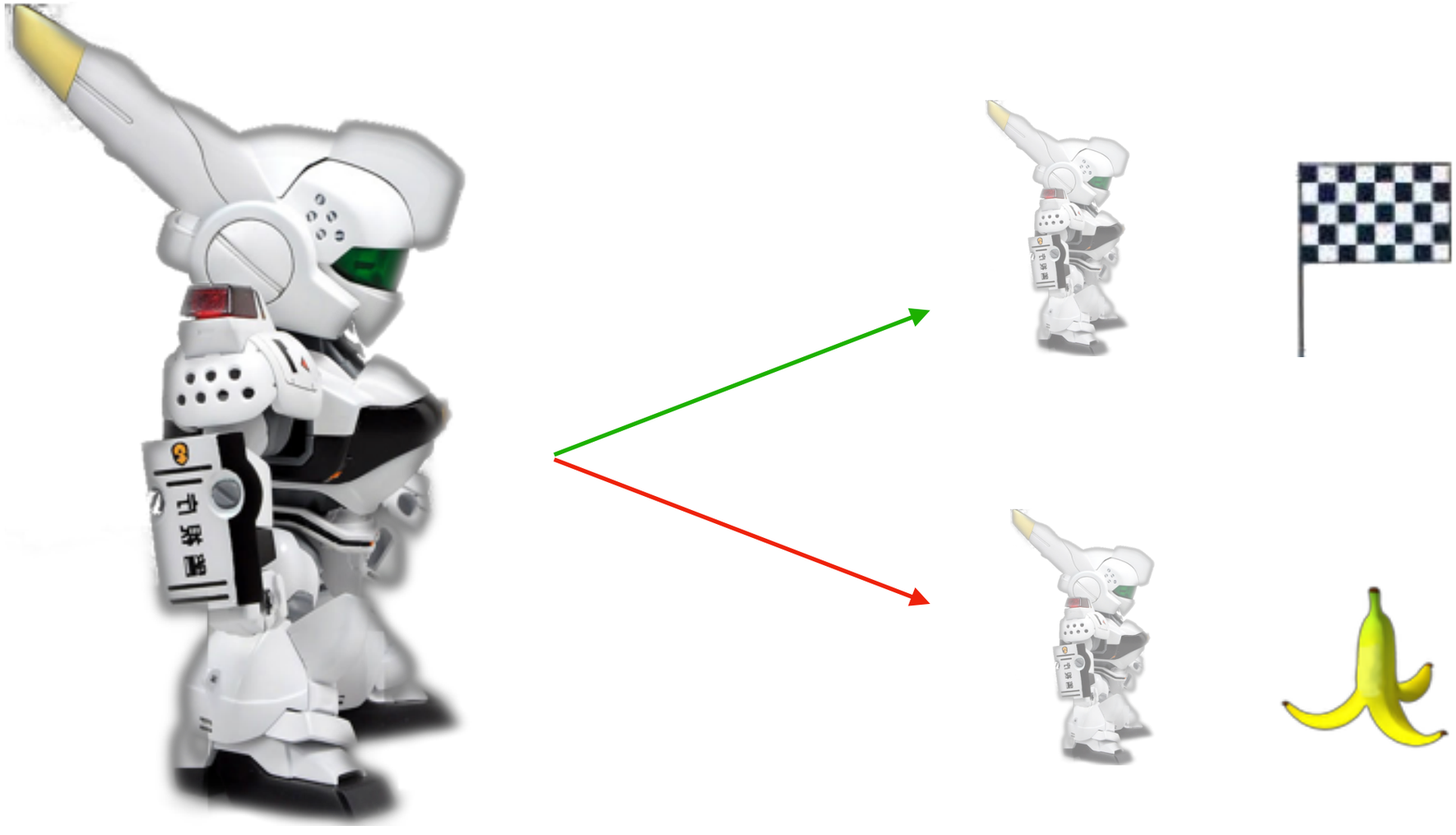


ective



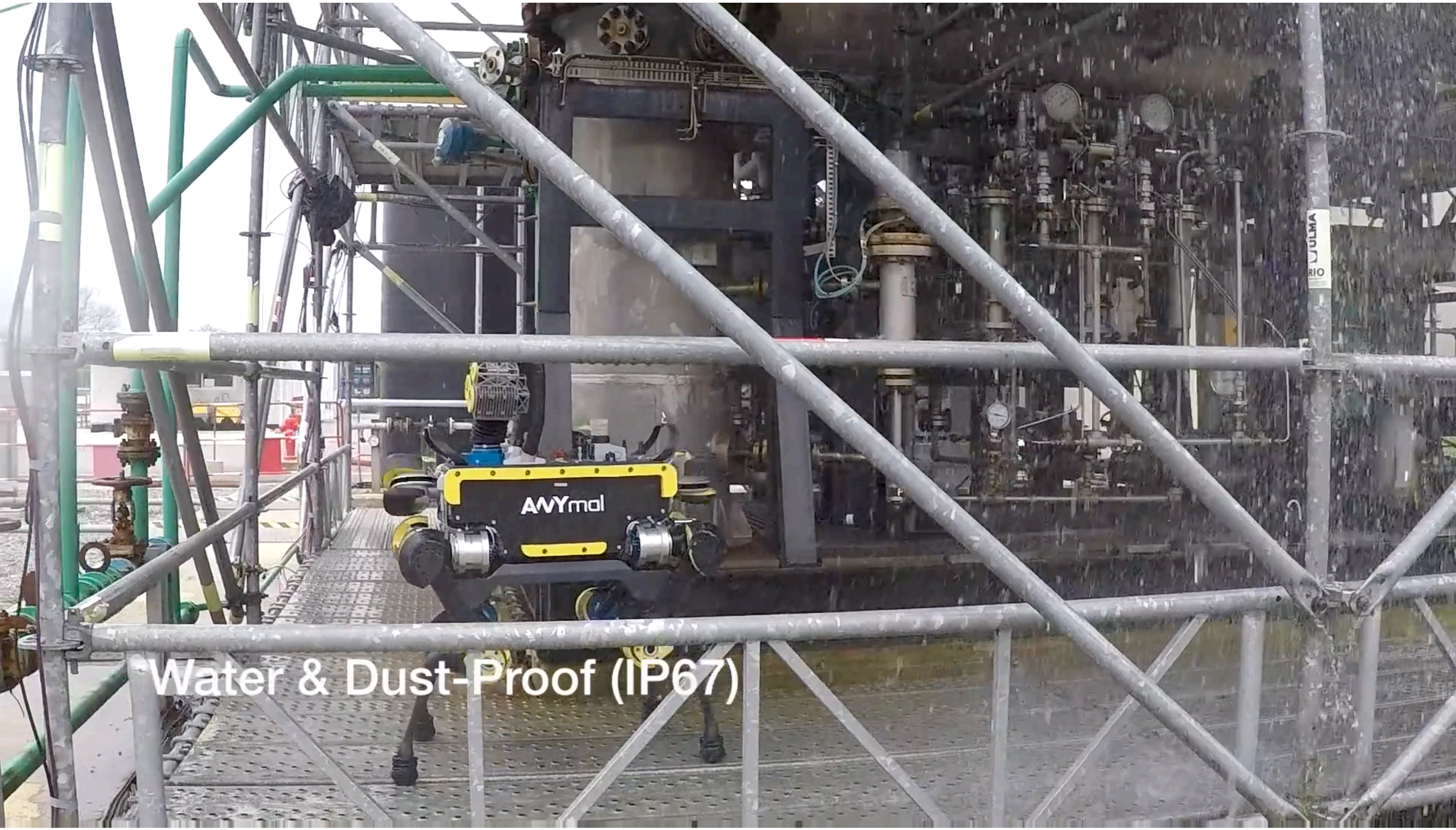


$$\min f(x)$$



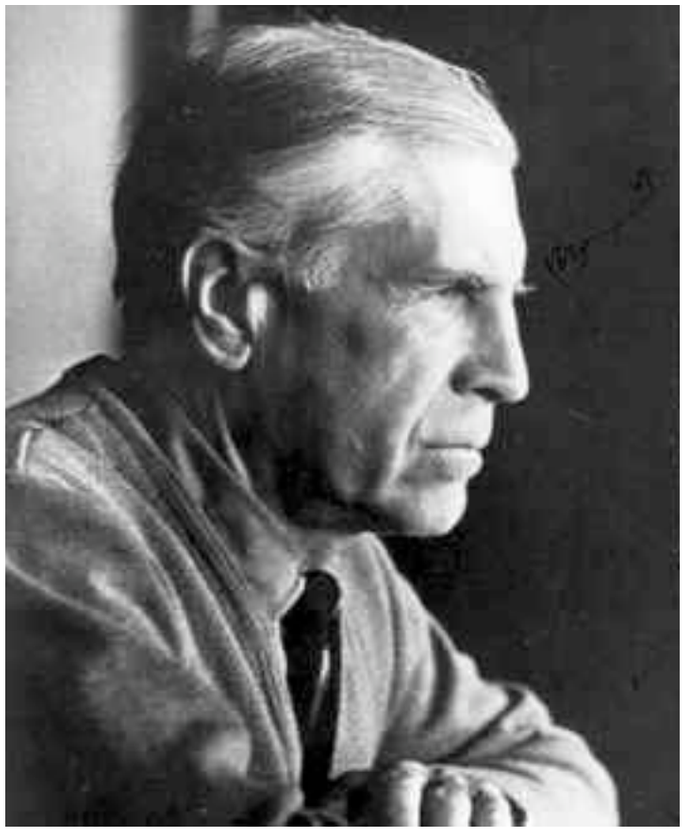
x :

take green action OR take red action



Water & Dust-Proof (IP67)

OC



Pontryagin
Minimum principle

Bellman
Principle of optimality

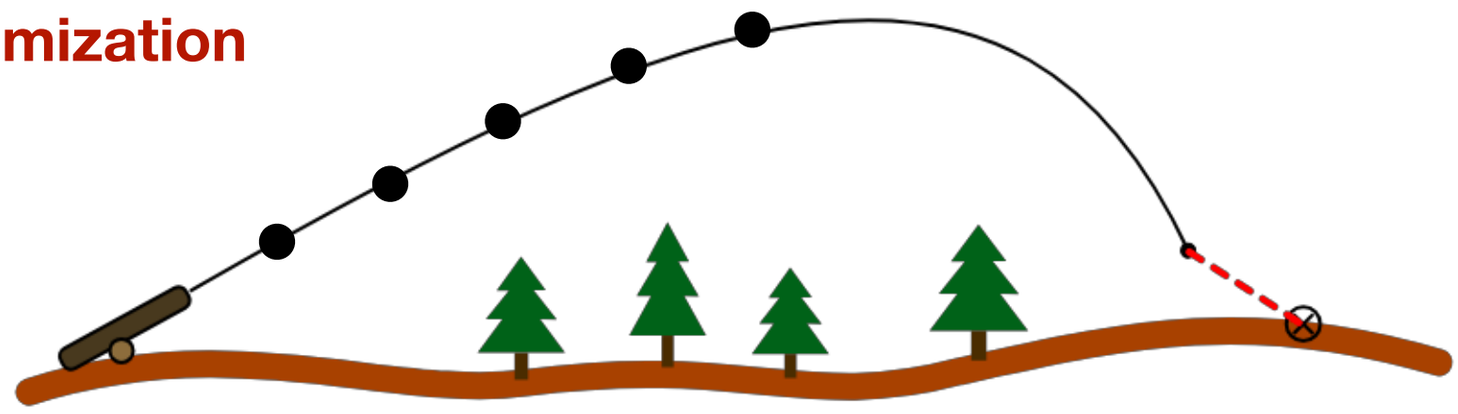
The optimal control problem

$$\min_{u_{1:T}} \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t), t = 1,2,\ldots, T$

$$x_0 = x[0]$$

This is called
constrained optimization



The optimal control problem

$$\min_{u_{1:2}} \quad g(x_1, u_1) + g(x_2, u_2) + \dots$$

subject to

$$x_2 = F(x_1, u_1)$$

$$x_3 = F(x_2, u_2)$$

$$\dots$$

$$x_0 = x[0]$$

But what is F?

What is a model?

- A (dynamics) model is a function that maps the current state action pair to the next state

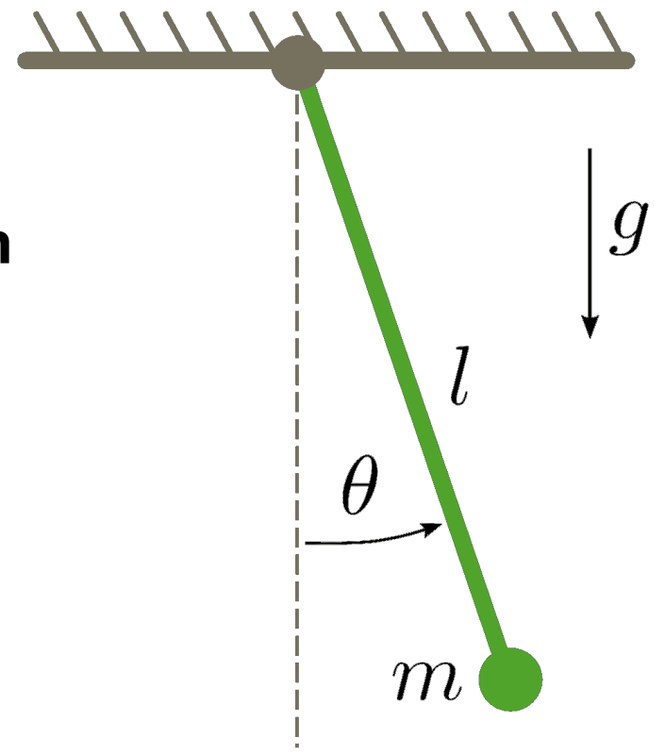
$$F : (x_t, u_t) \mapsto x_{t+1}$$

- e.g. Newton's law

- It describe how the system (environment) evolves over time

Example: pendulum

$$ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u\,.$$



We use a new variable to write the system in first-order form

$$\dot{\theta} = \omega$$

$$\dot{\omega} = -\frac{g}{l}\sin\theta - \frac{b}{ml^2}\omega + u$$

Discretize

$$\theta_{t+1} = \theta_t + \Delta t \cdot \omega$$

$$\omega_{t+1} = \omega_t + \Delta t \cdot (-\frac{g}{l}\sin\theta - \frac{b}{ml^2}\omega + u)$$

The optimal control problem

$$\min_{u_{1:T}} \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t), t = 1,2,\ldots,T$

$$x_0 = x[0]$$

This is called
constrained optimization

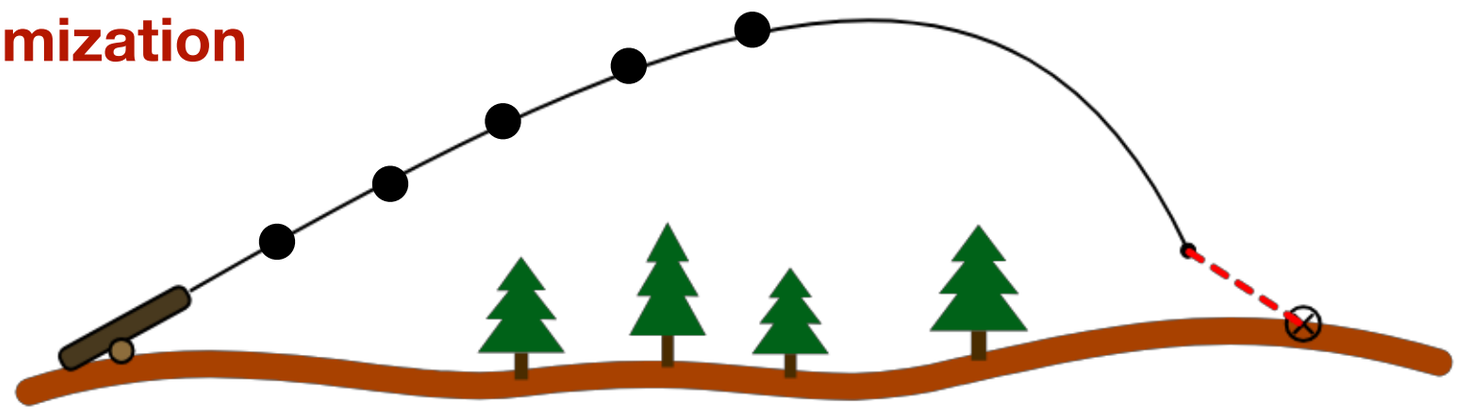
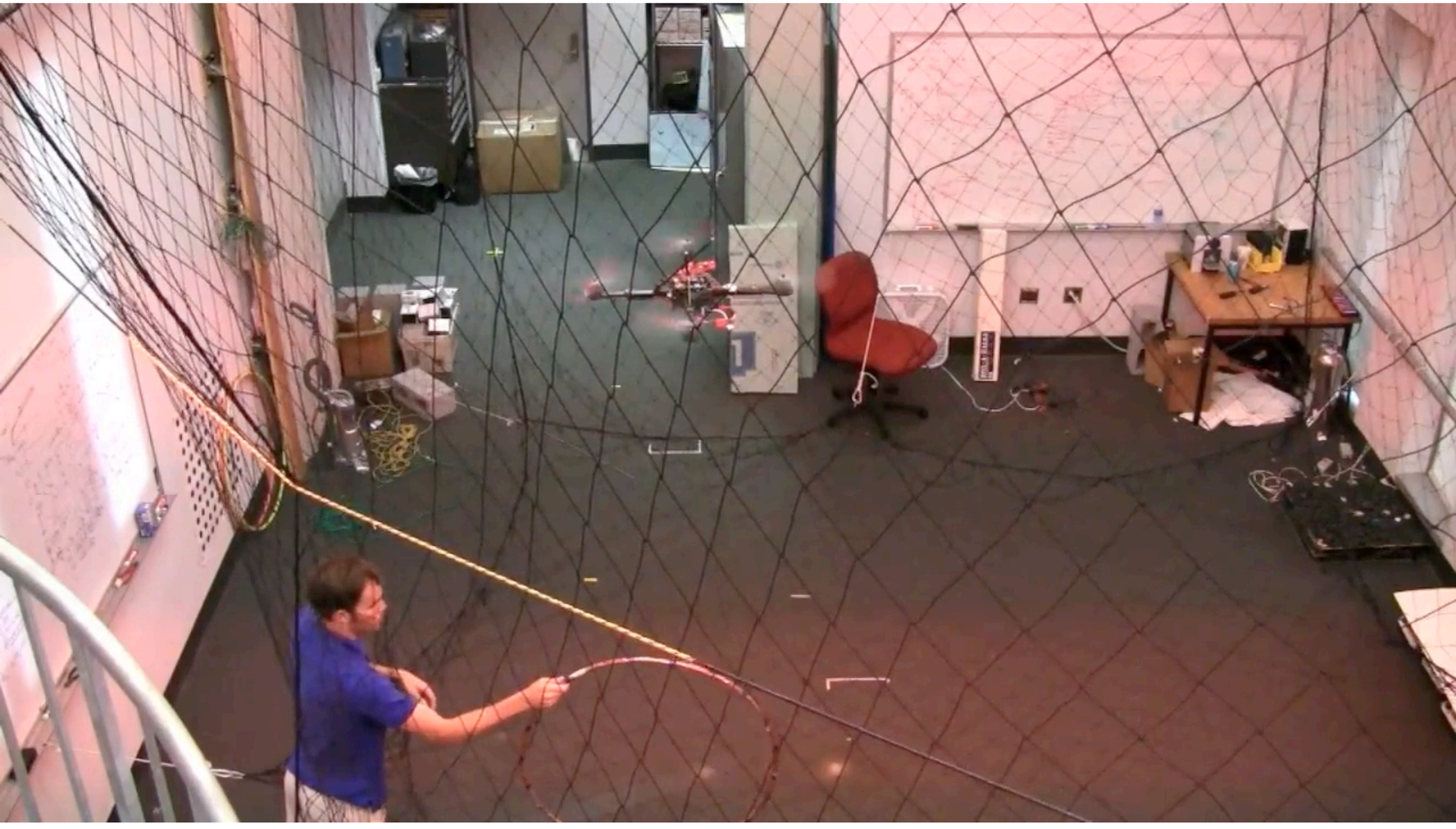


Figure adapted from M. Kelly



Linear quadratic regulator (LQR)

Kalman

$$\min_{u_{1:T}} \quad \sum_{t=1}^{T} x_t^\top Q x_t + u_t^\top R u_t$$

subject to

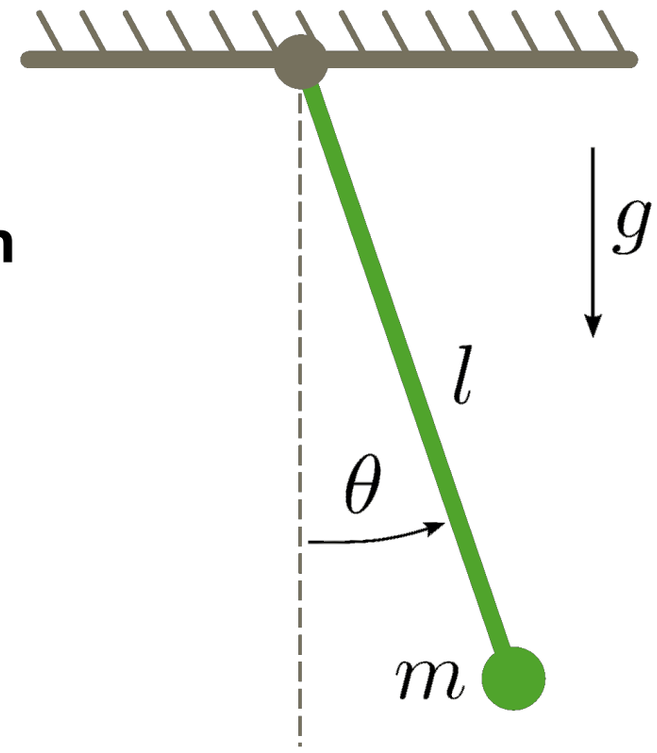
$$x_{t+1} = Ax_t + Bu_t, t = 1,2,\ldots,T$$

$$x_0 = c_0$$

MATLAB function: lqr()

Example of a LQ-system
Simple pendulum

$$ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u\,.$$



We use a new variable to write the system in first-order form

$$\dot{\theta} = \omega$$

$$\dot{\omega} = -\frac{g}{l}\sin\theta - \frac{b}{ml^2}\dot{\theta} + u$$

Discretize

$$\sin\theta \approx \theta$$

$$\theta_{t+1} = \theta_t + \Delta t \cdot \omega_t$$

$$\omega_{t+1} = \omega_t + \Delta t \cdot (-\frac{g}{l}\sin\theta_t - \frac{b}{ml^2}\omega_t + u_t)$$

$$\begin{bmatrix}\theta_{t+1}\\ \omega_{t+1}\end{bmatrix} = A\begin{bmatrix}\theta_t\\ \omega_t\end{bmatrix} + B\,[u_t]$$

Figure adapted from M. Kelly

- Remember the principle of optimality?

$$J_t(x_t) = \min_{u_t} g(x_t, u_t) + J_{t+1}(x_{t+1})$$

- In DP, we make decision backwards, so

$$J_T(x_T) = J(Ax_{T-1} + Bu_{T-1}) = (Ax_{T-1} + Bu_{T-1})'Q(Ax_{T-1} + Bu_{T-1})$$

$$J_{T-1}(x_{T-1}) = \min_{u_t} x_{T-1}^\top Q x_{T-1} + u_{T-1}^\top R u_{T-1} + J_T(x_T)$$

$$J_{T-1}(x_{T-1}) = \min_{u_t} x_{T-1}^\top Q x_{T-1} + u_{T-1}^\top R u_{T-1} + (Ax_{T-1} + Bu_{T-1})'Q(Ax_{T-1} + Bu_{T-1})$$

Rhs is a minimization of a quadratic form

$$u_{T-1}^* = -(R + B'QB)^{-1}B'QAx_{T-1}$$

$$= L_{T-1}x_{T-1}$$

To make decision for T-2, Plug into the RHS of $J_{T-1}(x_{T-1})$

$$J_{T-1}(x_{T-1}) = x'_{T-1}K_{T-1}x_{T-1}$$

$$K_{T-1} = A'(Q - QB(BQB + R)^{-1}BQ)A + Q$$

Repeat for step T-1, T-2, till we figure all the u's

Summary

The optimal control policy of LQ-system is given by:

$$u_t^*(x_t) = L_t x_t$$

$$L_t = -(R + B'K_{t+1}B)^{-1}B'K_{t+1}A$$

$$K_T = Q$$

$$K_t = A'(K_{t+1} - K_{t+1}B(B'K_{t+1}B + R)^{-1}B'K_{t+1})A + K_{t+1}$$

This is the (discrete-time) Riccati equation

MATLAB function: lqr()

Linear quadratic regulator (LQR)

Kalman

$$\min_{u_{1:T}} \sum_{t=1}^{T} x_t^\top Q x_t + u_t^\top R u_t$$

subject to $x_{t+1} = Ax_t + Bu_t, t = 1,2,\ldots,T$

$$x_0 = c_0$$

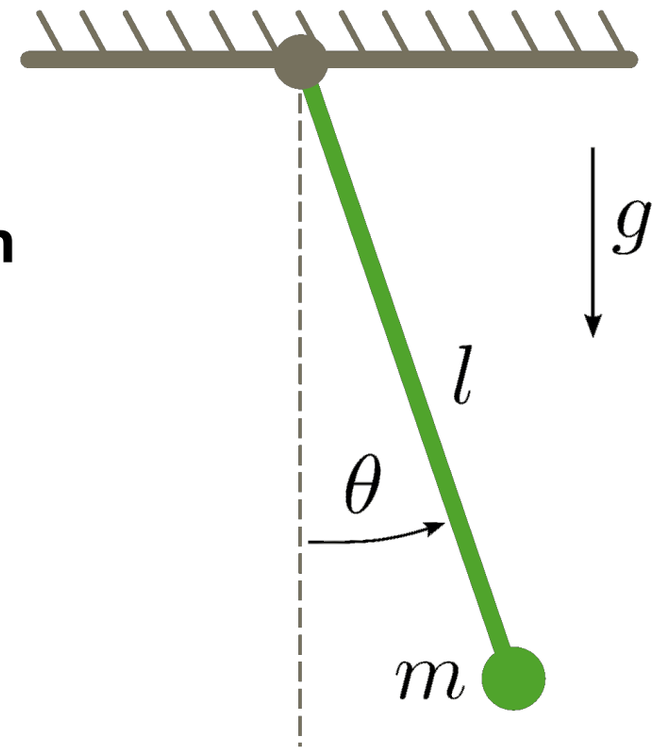
The optimal control is given in the closed form

$$u_t^*(x_t) = L_t x_t$$

MATLAB function: lqr()

Example of a LQ-system
Simple pendulum

$$ml^2\ddot{\theta} + b\dot{\theta} + mgl\sin\theta = u\,.$$



We use a new variable to write the system in first-order form

$$\dot{\theta} = \omega$$

$$\dot{\omega} = -\frac{g}{l}\sin\theta - \frac{b}{ml^2}\dot{\theta} + u$$

Discretize

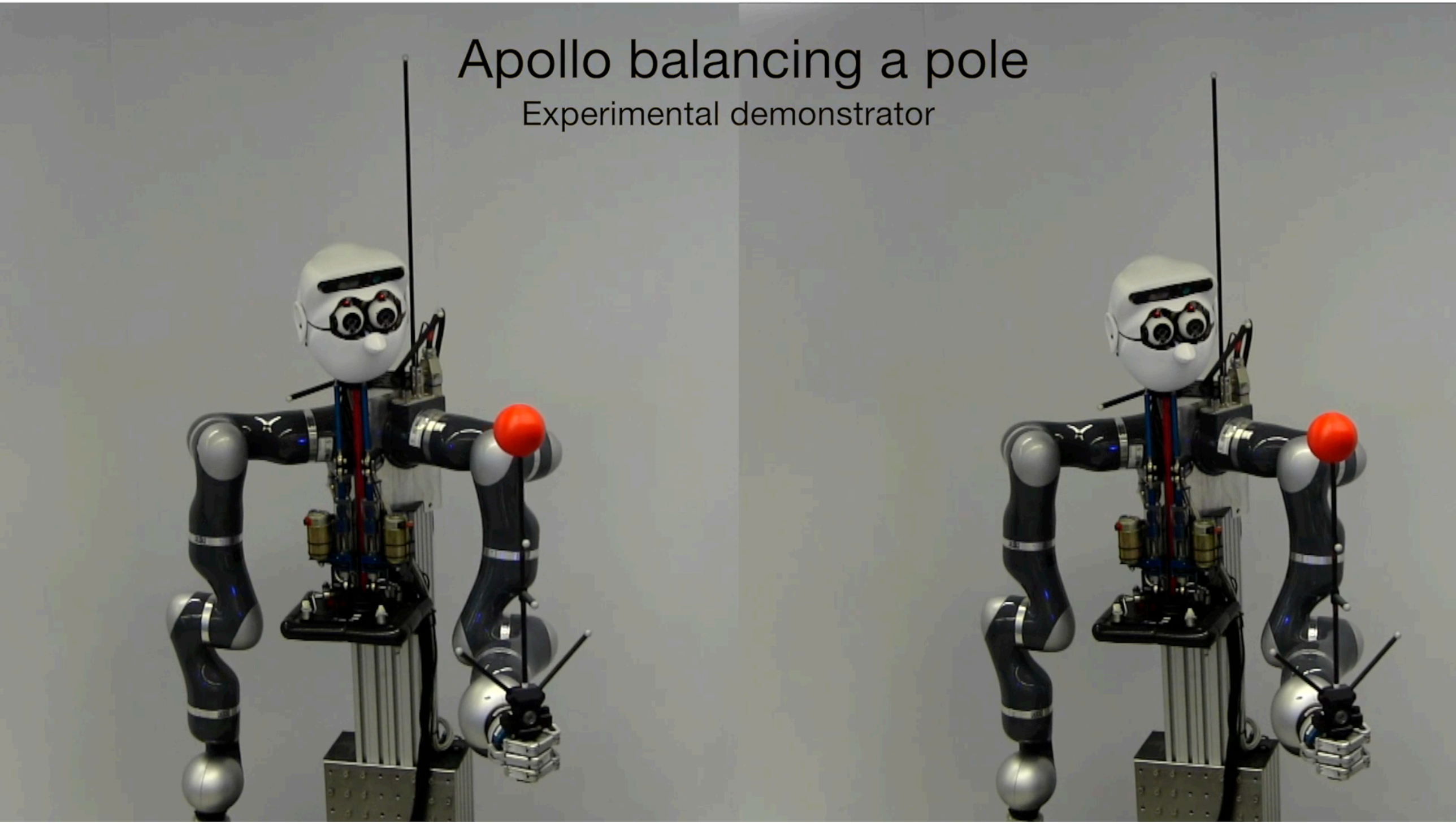
$$\sin\theta \approx \theta$$

$$\theta_{t+1} = \theta_t + \Delta t \cdot \omega_t$$

$$\omega_{t+1} = \omega_t + \Delta t \cdot (-\frac{g}{l}\sin\theta_t - \frac{b}{ml^2}\omega_t + u_t)$$

$$\begin{bmatrix}\theta_{t+1}\\ \omega_{t+1}\end{bmatrix} = A\begin{bmatrix}\theta_t\\ \omega_t\end{bmatrix} + B\,[u_t]$$

Figure adapted from M. Kelly


Apollo balancing a pole
Experimental demonstrator

In general, OC needs to be solved by numerical optimization

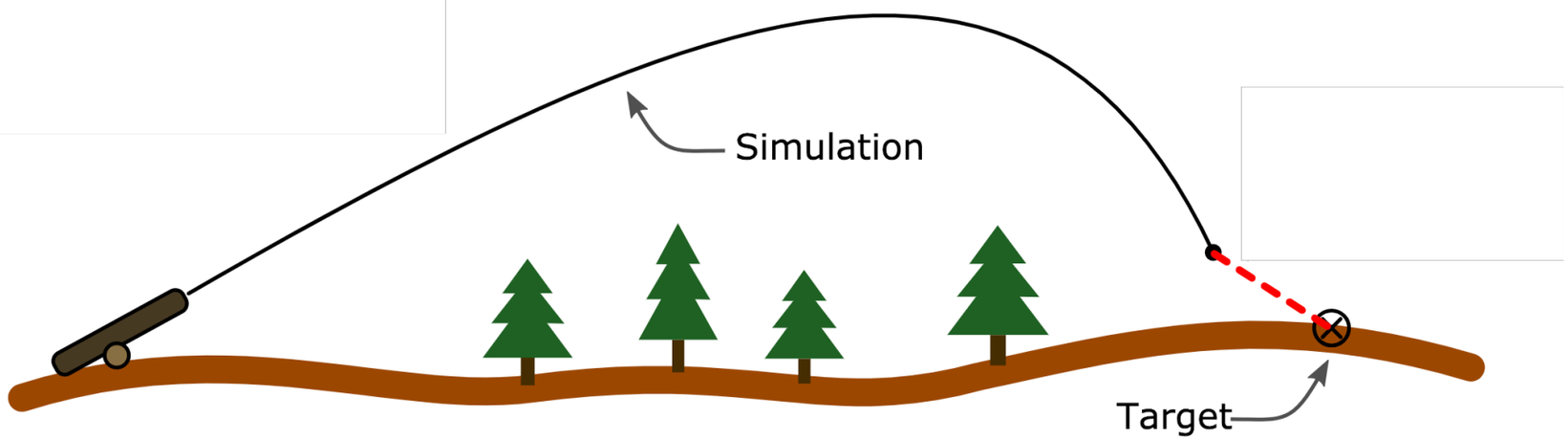
Trajectory optimization: Shooting method

$$\min_{u_{1:T}} \quad \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t),$

$t = 1,2,\ldots,T$

$x_0 = x[0]$



Simultaneous method

$$\min_{u_{1:T},\ x_{1:T}} \quad \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t),$

$t = 1,2,\ldots,T$

$x_0 = x[0]$

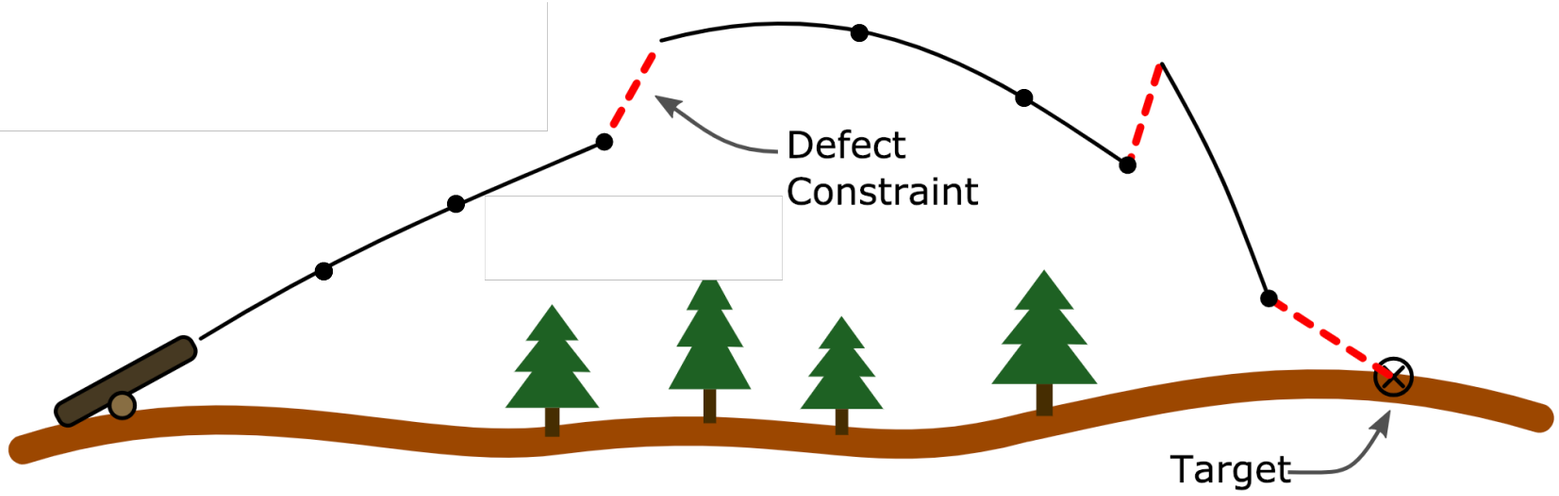


Figure adapted from M. Kelly

The optimal control problem

$$\min_{u_{1:T}} \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t), t = 1,2,\ldots,T$

$$x_0 = x[0]$$

This is called
constrained optimization

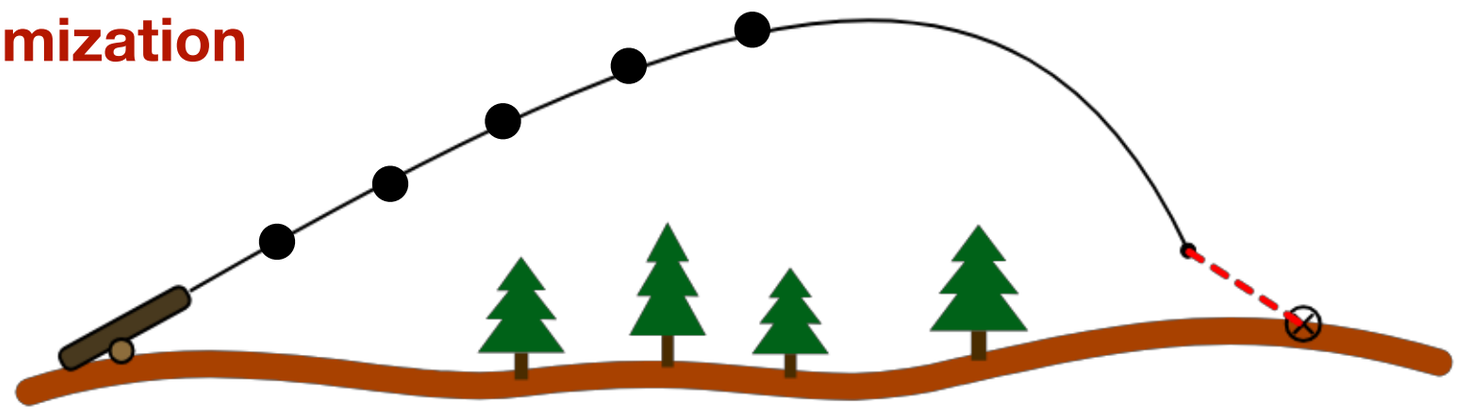
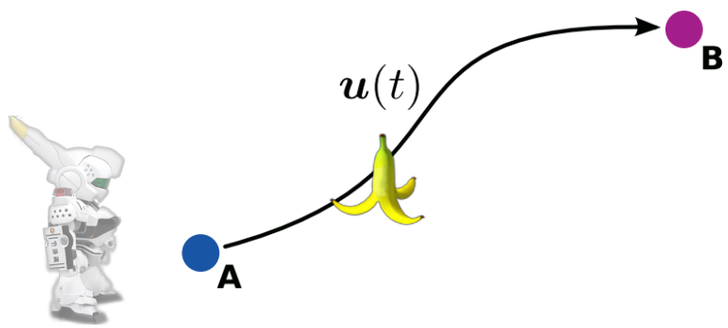
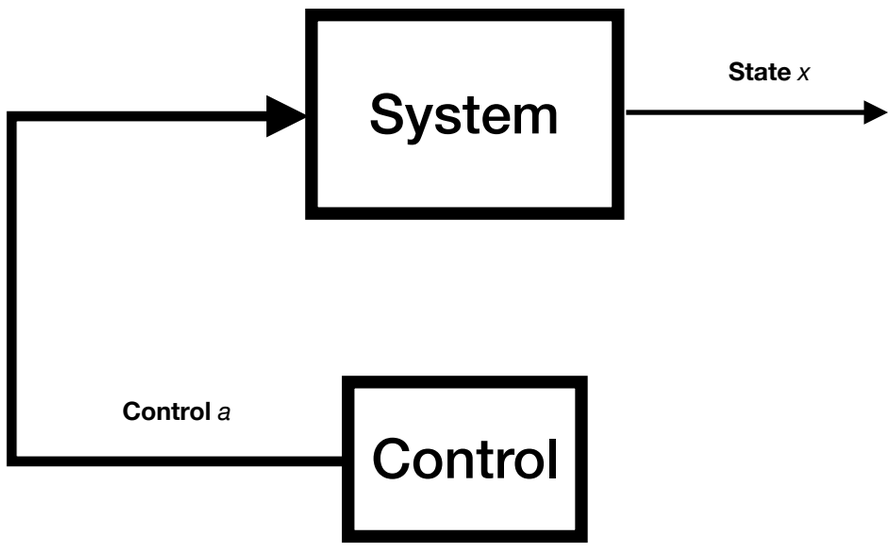


Figure adapted from M. Kelly

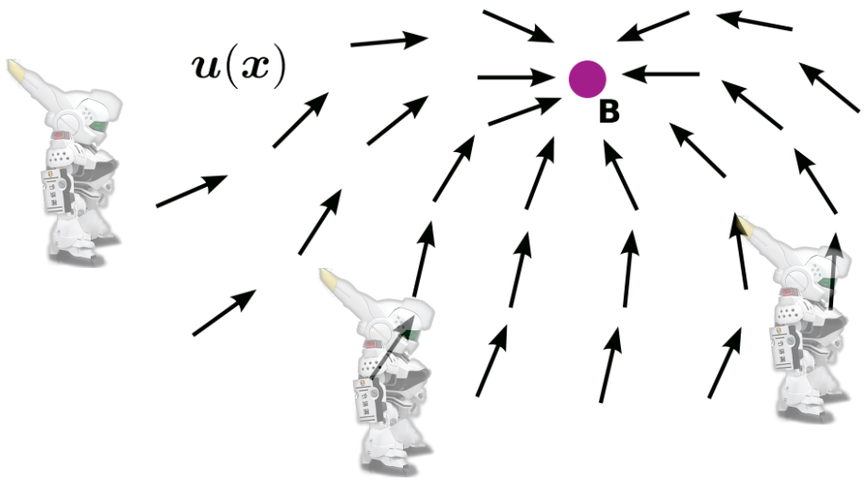
Open vs closed-loop control

Open-loop
Find trajectory





Closed-loop
Find policy



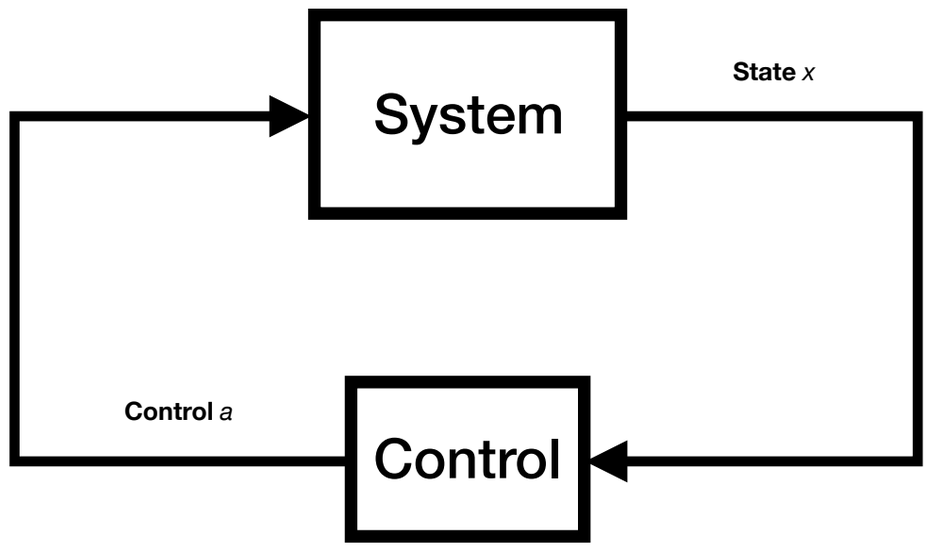


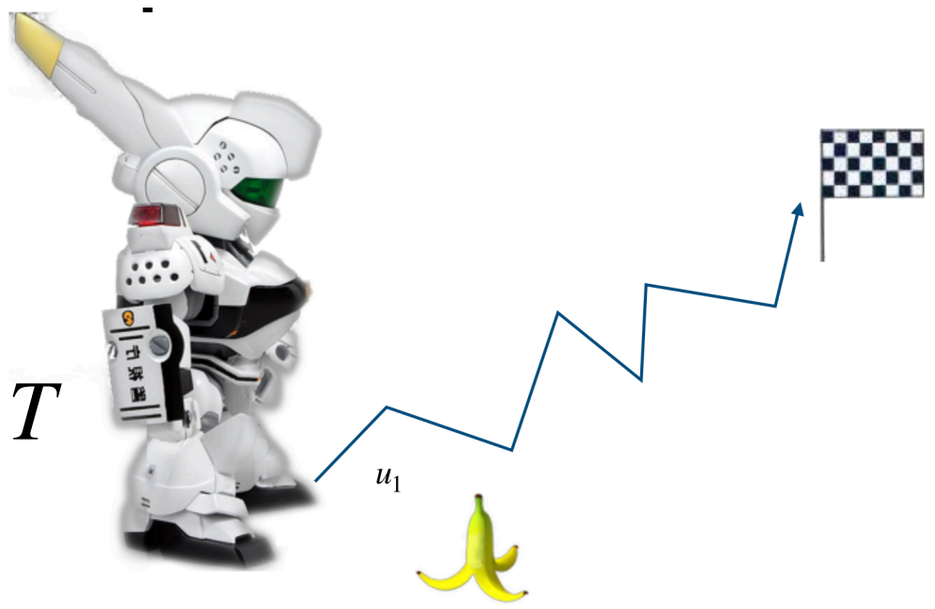
Figure adapted from M. Kelly

Model predictive control Formulation

$$\min_{u_{1:T}} \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t), t = 1,2,\ldots,T$

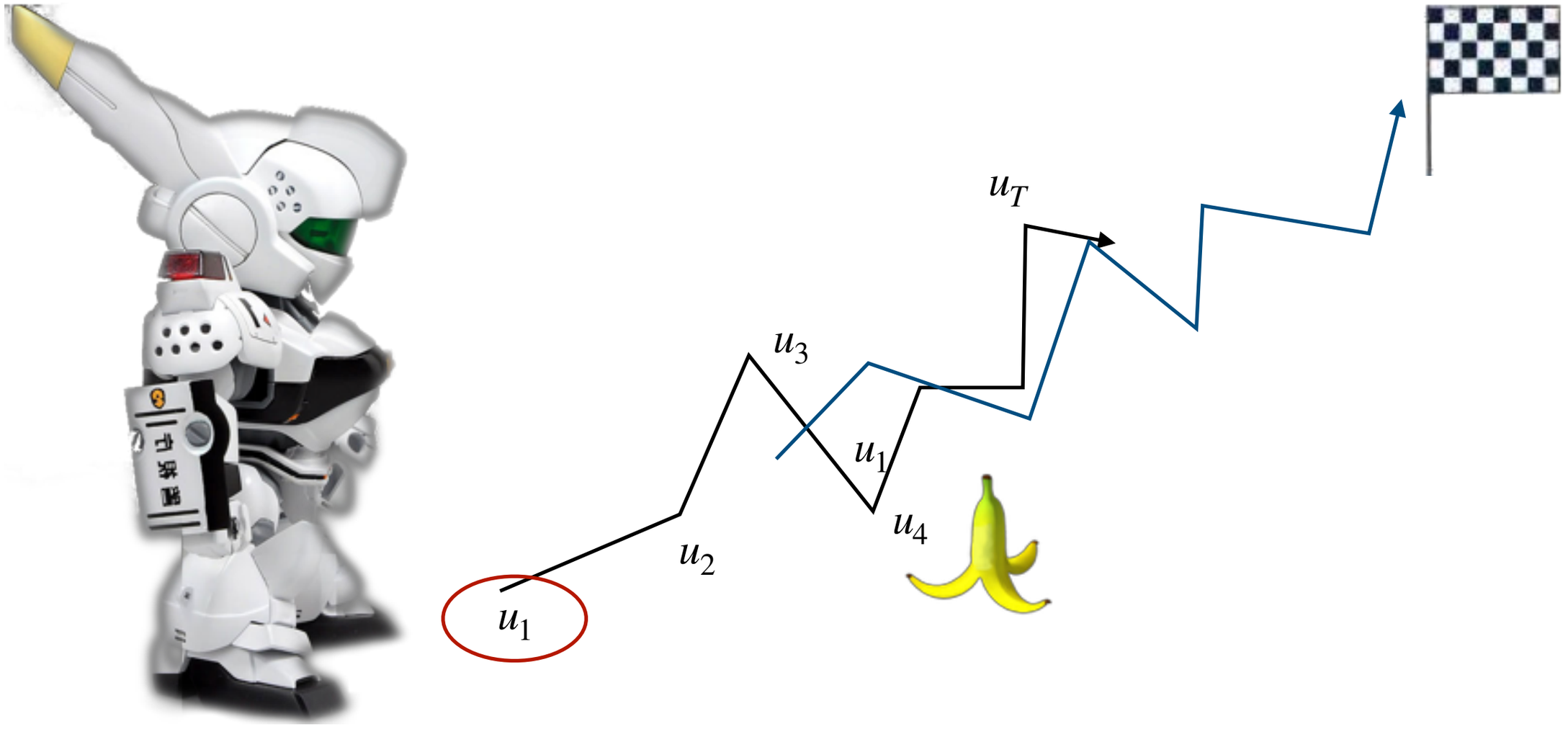
$x_1 = x[t]$



Only implement the first component of solution u_1^*

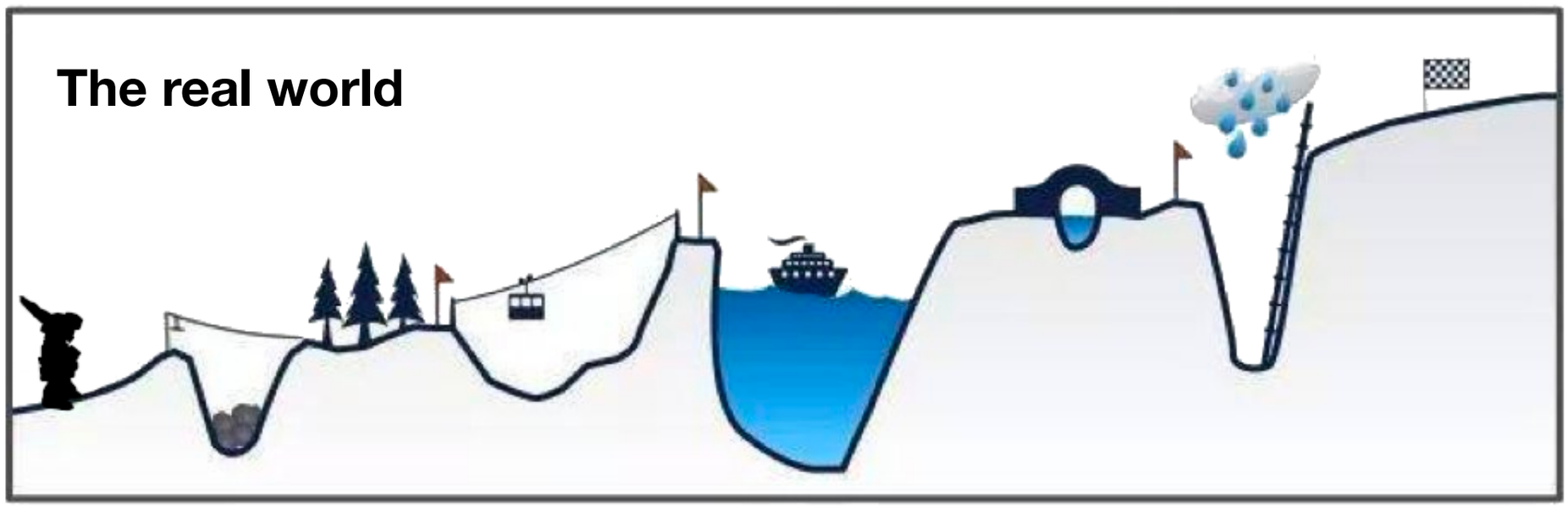
Move on to t+1, solve the optimization problem again

A dominant control method
Model predictive control



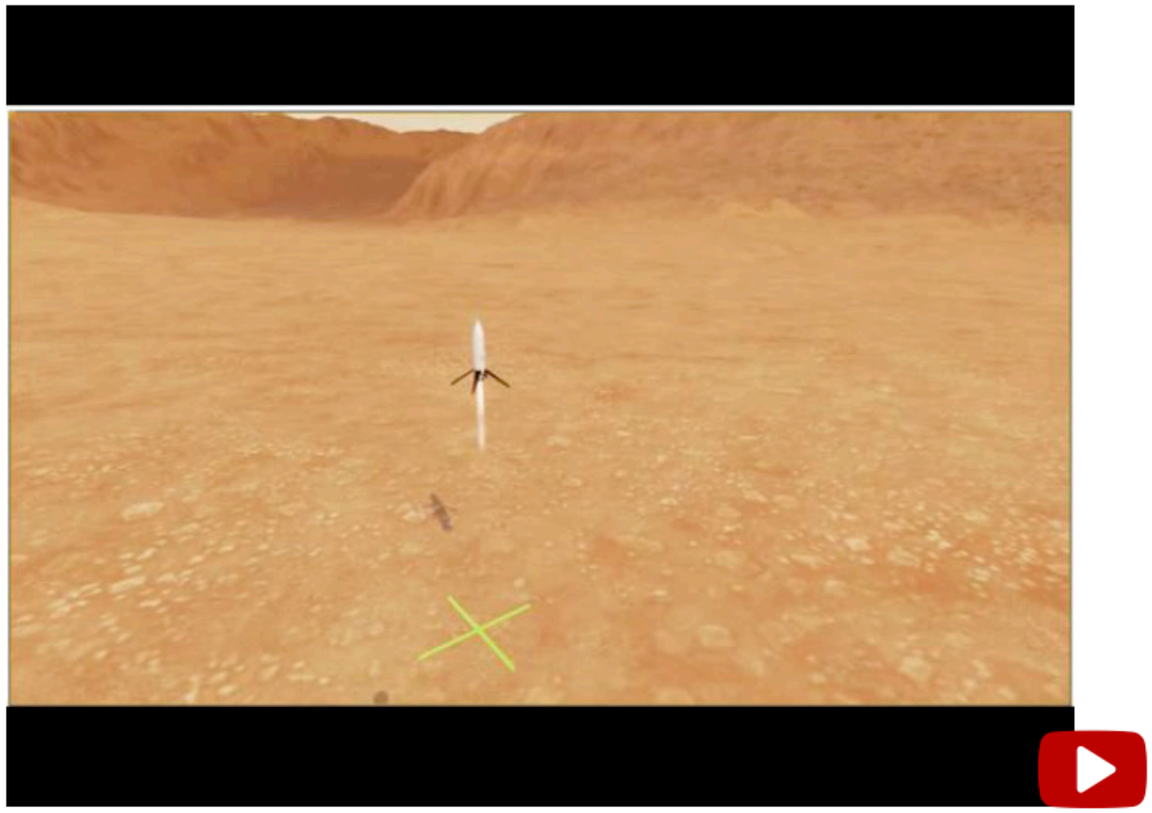
Plan control input using OC

Why replan?
Predicting the future is hard!



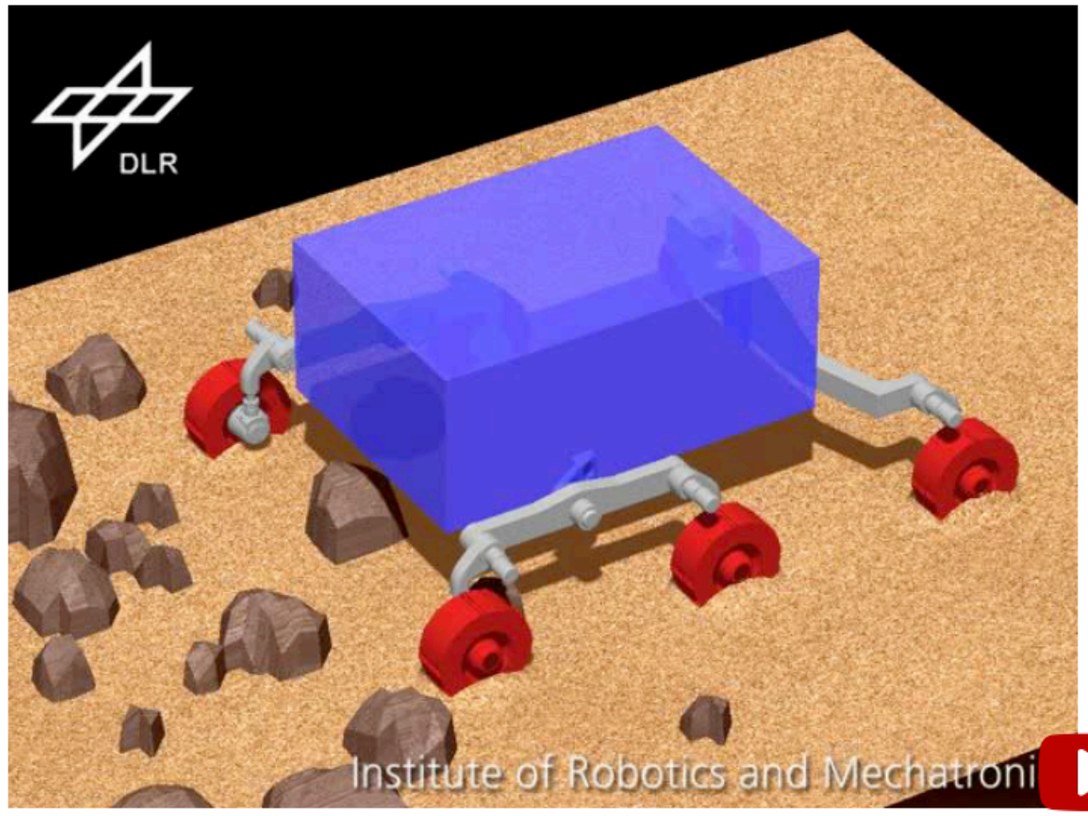
Model predictive control

powered descent



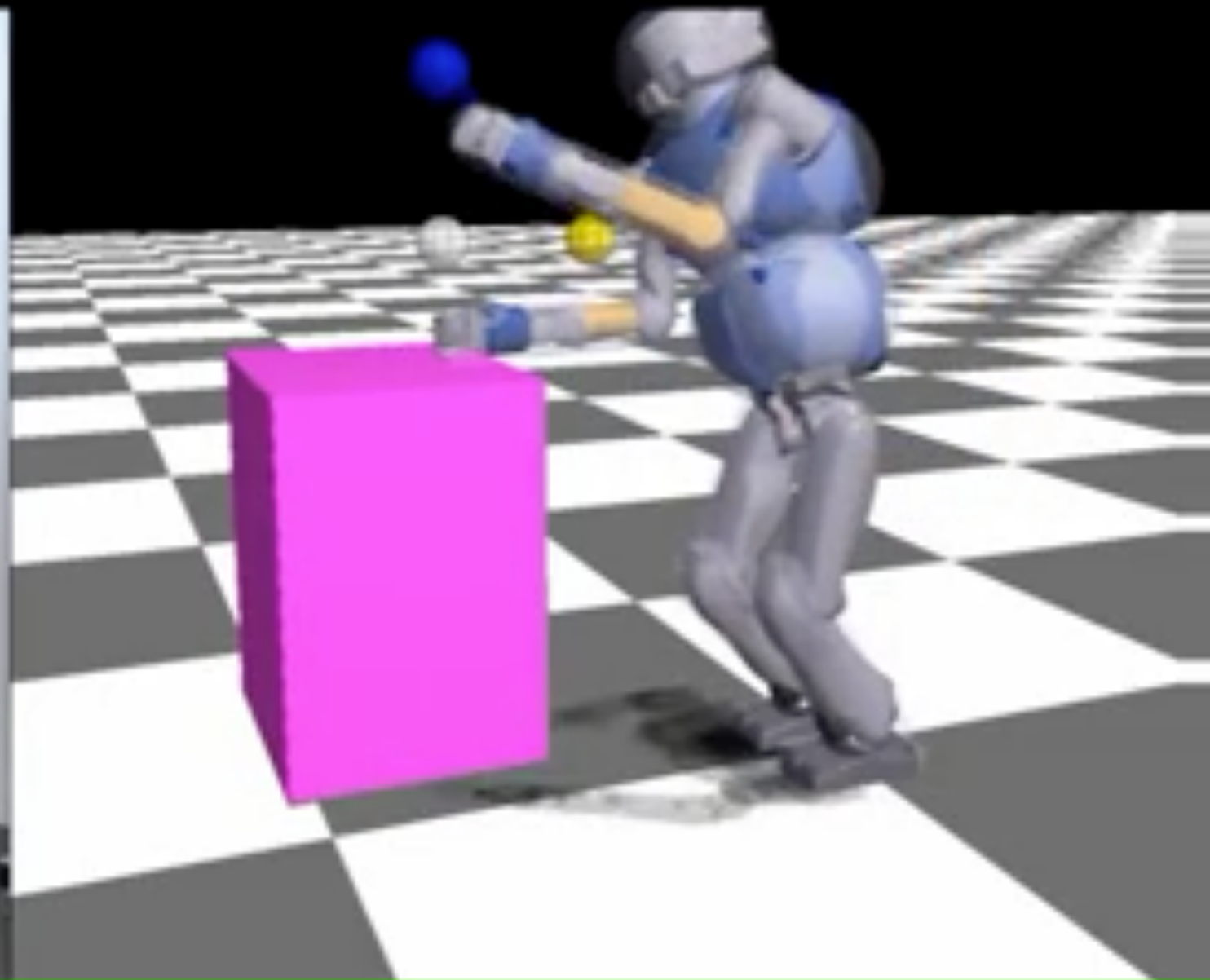
(Pascucci, Bennani, Bemporad, 2016)

planetary rover



(Krenn et. al., 2012)

The robot exploits the additional support in order to maintain balance

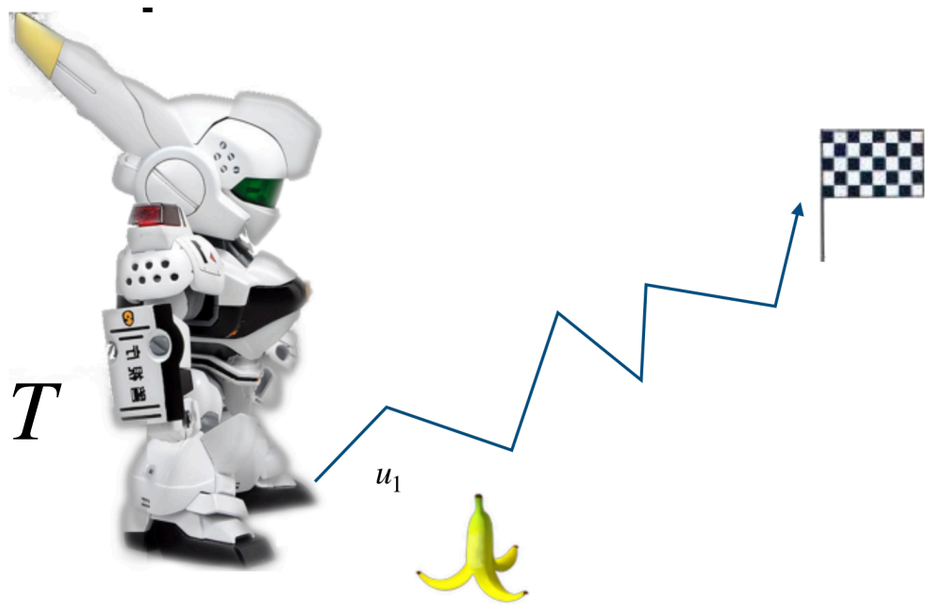


Model predictive control Formulation

$$\min_{u_{1:T}} \sum_{t=1}^{T} g(x_t, u_t)$$

subject to $x_{t+1} = F(x_t, u_t), t = 1,2,\ldots,T$

$x_1 = x[t]$

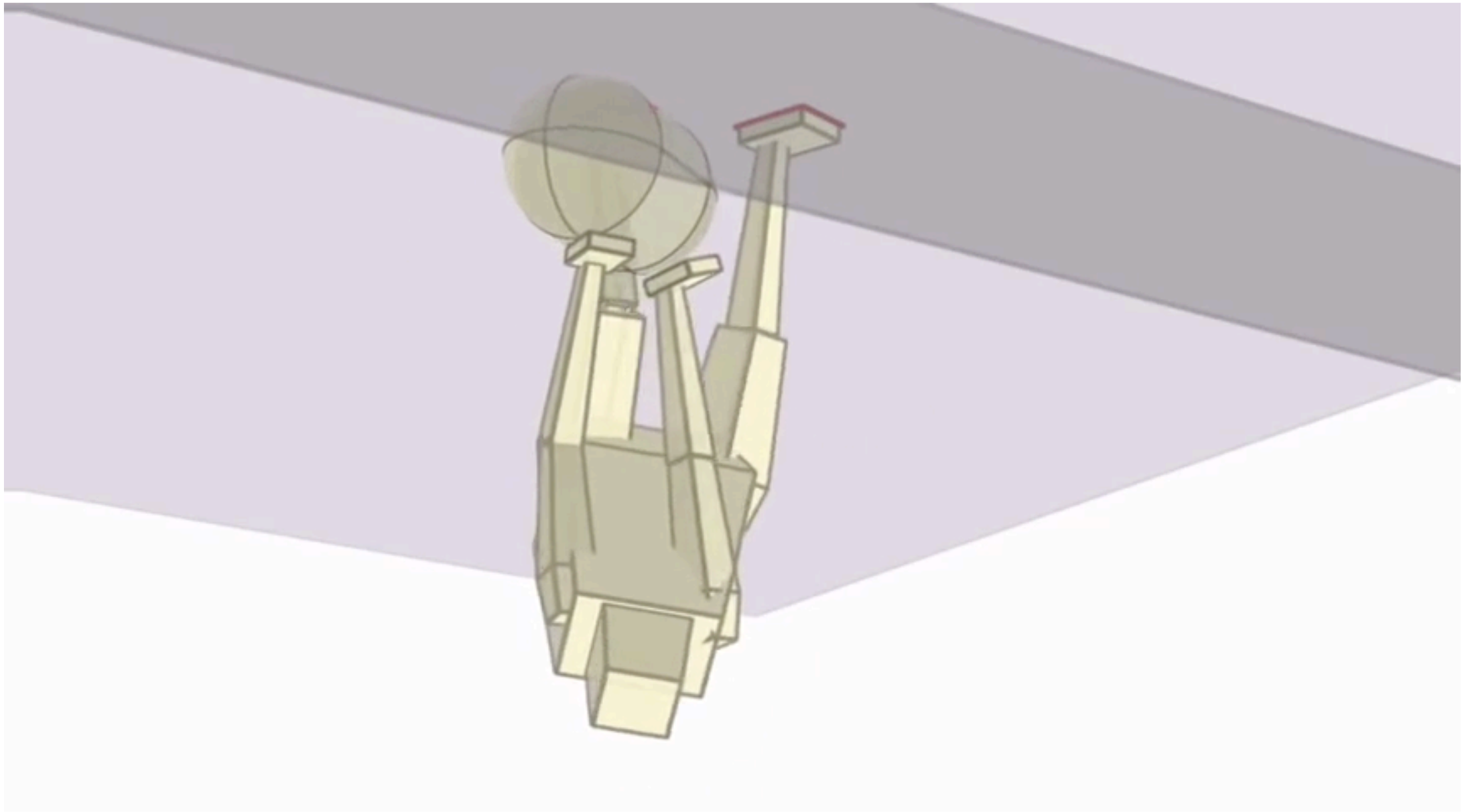


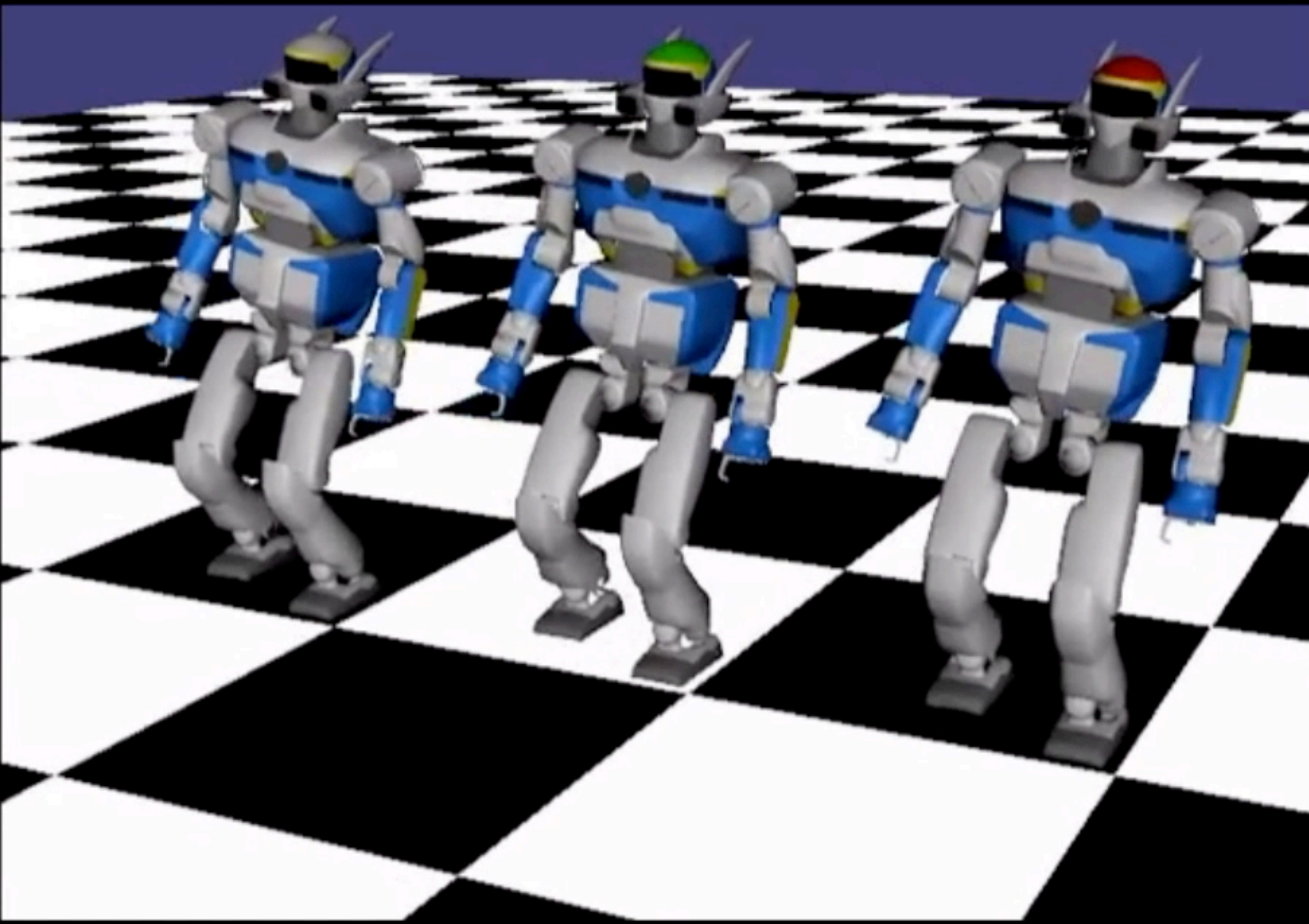
Only implement the first component of solution u_1^*

Move on to t+1, solve the optimization problem again

Challenge of Control

- Real world is stochastic
- Robustness under unknown model regime is hard
  - Need better system identification
- Traditional method does not use (large amounts of) data
- Non-linear systems (not $Ax+Bu$) is hard





Model-based RL

- Learning a model: $F : (x_t, u_t) \mapsto x_{t+1}$ using ML methods
- Plug it into your favorite optimal control method
  - e.g. MPC with learned model

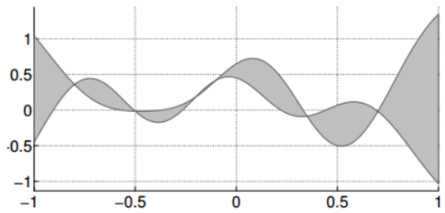
Model-based RL

- More sample-efficient (compared with model-free methods), if model is chosen appropriately
- Still benefits from ML methods
  - e.g., input may be images/video

How to learn (fit) a model?

$$F : (x_t, u_t) \mapsto x_{t+1}$$

Gaussian process



GP with input $(\mathbf{s}, \mathbf{a})$ and output $\mathbf{s}'$

Pro: very data-efficient

Con: not great with non-smooth dynamics

Con: very slow when dataset is big

neural network

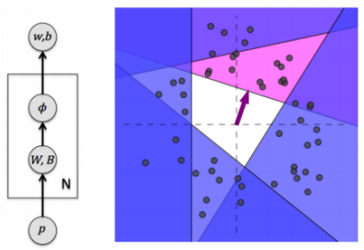


image: Punjani & Abbeel '14

Input is $(\mathbf{s}, \mathbf{a})$, output is $\mathbf{s}'$

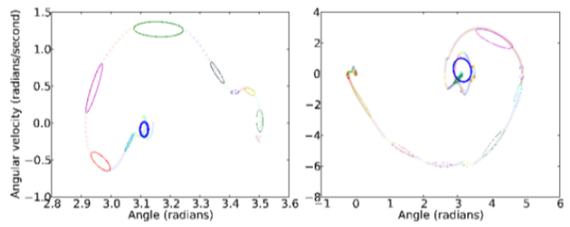
Euclidean training loss corresponds to Gaussian $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$

More complex losses, e.g. output parameters of Gaussian mixture

Pro: very expressive, can use lots of data

Con: not so great in low data regimes

other



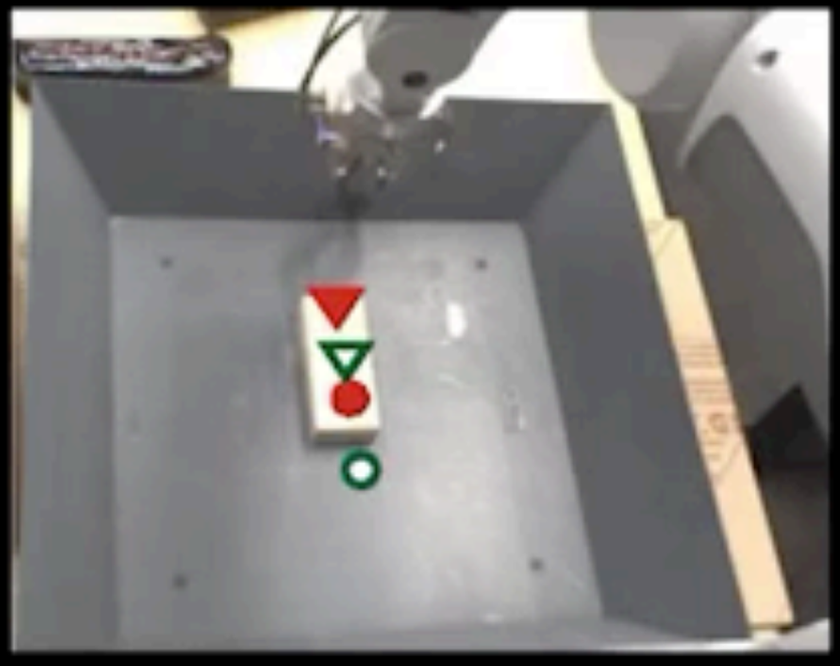
GMM over $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$ tuples

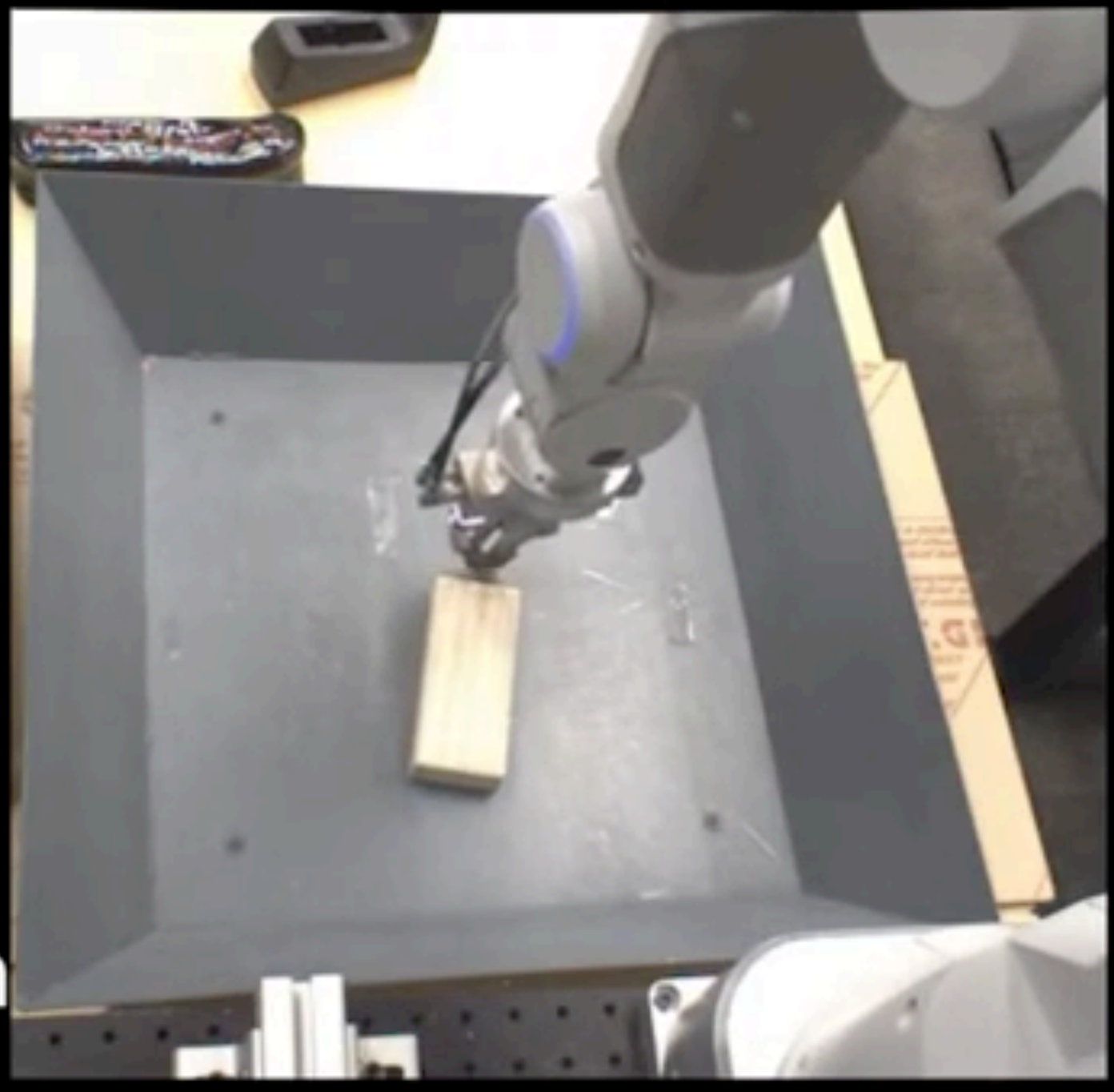
Train on $(\mathbf{s}, \mathbf{a}, \mathbf{s}')$, condition to get $p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$

For i^{th} mixture element, $p_i(\mathbf{s}, \mathbf{a})$ gives region where the mode $p_i(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ holds

other classes: domain-specific models (e.g. physics parameters)

Slide adapted from S. Levine





object translation

Sample complexity

gradient-free methods
(e.g. NES, CMA, etc.)

10x

fully online methods
(e.g. A3C)

10x

policy gradient methods
(e.g. TRPO)

10x

replay buffer value estimation methods
(Q-learning, DDPG, NAF, etc.)

10x

model-based deep RL
(e.g. guided policy search)

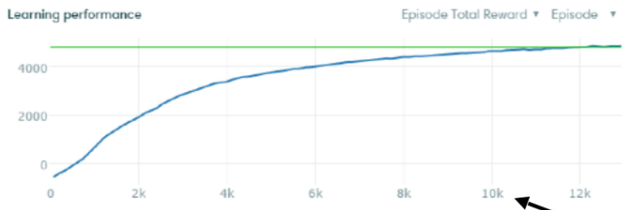
10x

model-based "shallow" RL
(e.g. PILCO)

Evolution Strategies as a Scalable Alternative to Reinforcement Learning

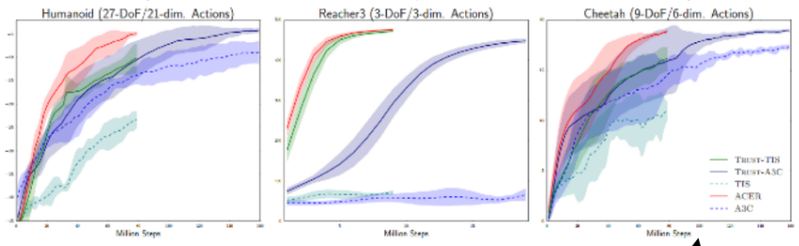
Tim Salimans [1] Jonathan Ho [1] Xi Chen [1] Ilya Sutskever [1]

half-cheetah (slightly different version)



TRPO+GAE (Schulman et al. '16)

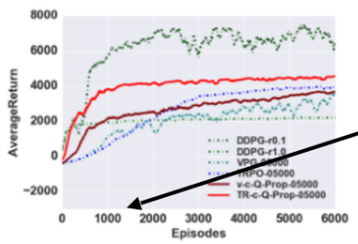
10,000,000 steps
(10,000 episodes)
(~ 1.5 days real time)



Wang et al. '17

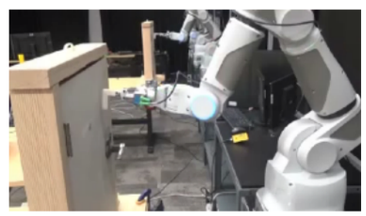
100,000,000 steps
(100,000 episodes)
(~ 15 days real time)

half-cheetah

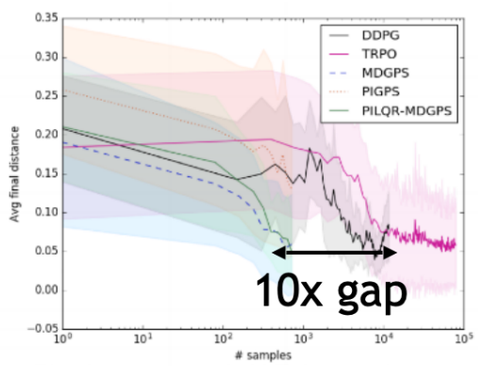


Gu et al. '16

1,000,000 steps
(1,000 episodes)
(~ 3 hours real time)



	cart-pole	cart-double-pole	unicycle
state space	$\mathbb{R}^4$	$\mathbb{R}^6$	$\mathbb{R}^{12}$
# trials	≤ 10	20–30	≈ 20
experience	≈ 20 s	≈ 60 s–90 s	≈ 20 s–30 s
parameter space	$\mathbb{R}^{305}$	$\mathbb{R}^{1816}$	$\mathbb{R}^{28}$



Chebotar et al. '17 (note log scale)

about 20 minutes of experience on a real robot

Slide adapted from S. Levine

References for studying optimal control and model-based approaches

- Russ Tedrake's online text "Underactuated Robotics"
    - http://underactuated.csail.mit.edu/underactuated.html?chapter=dp
    - I like this one a lot. The text is really accessible. Talked quite a bit about the basics of optimal control and dynamics.
- Sergey Levine's deep RL lecture on MB-RL; Chelsea Finn's lecture on MB-RL
    - http://rail.eecs.berkeley.edu/deeprlcourse/
    - https://sites.google.com/view/deep-rl-bootcamp/lectures
    - Sergey and Chelsea are very active in the space of MBRL. They approach are very relevant to the ML audience.
- Dimitri Bertsekas's new text (online)
    - http://web.mit.edu/dimitrib/www/RLbook.html
    - Dimitri Bertsekas has been a leading figure in optimization and control. He is particually interested in the recent development of RL as well. He has several classic text, for example, the Dynamic Programming and Optimal Control 2 Volume referenced on our course page. This is a new text he took the perspective of comparing RL and optimal control.
- Matthew Kelly's tutorial on trajectory optimization
    - http://www.matthewpeterkelly.com/tutorials/trajectoryOptimization/index.html
- Good MPC courses:
    - http://cse.lab.imtlucca.it/~bemporad/teaching.html
    - http://www.mpc.berkeley.edu/mpc-course-material
- And many more. The field of optimal control is vast. Feel free to shoot me an email if you have questions: jzhu@tuebingen.mpg.de