From LQR to Nonlinear Control

Lucas Janson

CS/Stat 184(0): Introduction to Reinforcement Learning

Fall 2024

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

Feedback from feedback forms

1. Thank you to everyone who filled out the forms!
2. Positive definiteness of P_h at every step of induction

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

Recap: LQR

Problem Statement (finite horizon, time homogeneous):

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[x_H^\top Q x_H + \sum_{h=0}^{H-1} (x_h^\top Q x_h + u_h^\top R u_h) \right]$$

such that $x_{h+1} = Ax_h + Bu_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0, \sigma^2 I)$

Recap: LQR

Problem Statement (finite horizon, time homogeneous):

$$\arg\min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\left[x_H^\top Q x_H + \sum_{h=0}^{H-1}(x_h^\top Q x_h + u_h^\top R u_h)\right]$$

such that $\quad x_{h+1} = Ax_h + Bu_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0,\sigma^2 I)$

- States $x_h \in \mathbb{R}^d$
- Actions/controls $u_h \in \mathbb{R}^k$
- Additive noise $w_h \sim \mathcal{N}(0,\sigma^2 I)$
- Dynamics linear with state coefficient matrix $A \in \mathbb{R}^{d\times d}$ and action coefficient matrix $B \in \mathbb{R}^{d\times k}$
- Cost function quadratic with positive definite state coefficient matrix $Q \in \mathbb{R}^{d\times d}$ and positive definite action coefficient matrix $R \in \mathbb{R}^{k\times k}$

Recap: LQR Optimal Control

Recap: LQR Optimal Control

$$V_H^\star(x) = x^\top Q x, \text{ define } P_H = Q, p_H = 0,$$

Recap: LQR Optimal Control

$$V_H^\star(x) = x^\top Q x, \text{ define } P_H = Q, p_H = 0,$$

We showed that $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

$$p_h = \text{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Recap: LQR Optimal Control

$$V_H^\star(x) = x^\top Q x, \text{ define } P_H = Q, p_H = 0,$$

We showed that $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

$$p_h = \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Along the way, we also showed that $\pi_h^\star(x) = -K_h x$, where:

$$K_h = (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

Recap: LQR Optimal Control

$$V_H^\star(x) = x^\top Q x, \text{ define } P_H = Q, p_H = 0,$$

We showed that $V_h^\star(x) = x^\top P_h x + p_h$, where:

$$P_h = Q + A^\top P_{h+1} A - A^\top P_{h+1} B (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

$$p_h = \mathrm{tr}\left(\sigma^2 P_{h+1}\right) + p_{h+1}$$

Along the way, we also showed that $\pi_h^\star(x) = -K_h x$, where:

$$K_h = (R + B^\top P_{h+1} B)^{-1} B^\top P_{h+1} A$$

Optimal policy has nothing to do with initial distribution μ_0 or the noise σ^2!

Time-Dependent Costs and Dynamics

Time-Dependent Costs and Dynamics

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\left[x_H^\top Q_H x_H + \sum_{h=0}^{H-1}(x_h^\top Q_h x_h + u_h^\top R_h u_h)\right]$$

such that $x_{h+1} = A_h x_h + B_h u_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0,\sigma^2 I)$

Time-Dependent Costs and Dynamics

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\left[x_H^\top Q_H x_H + \sum_{h=0}^{H-1}(x_h^\top Q_h x_h + u_h^\top R_h u_h)\right]$$

such that $x_{h+1} = A_h x_h + B_h u_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0,\sigma^2 I)$

Exact same derivation, only thing that changes is the Ricatti equation:

$$P_h = Q_h + A_h^\top P_{h+1} A_h - A_h^\top P_{h+1} B_h (R_h + B_h^\top P_{h+1} B_h)^{-1} B_h^\top P_{h+1} A_h$$

More General Quadratic Cost Function

More General Quadratic Cost Function

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\Big[x_H^\top Q_H x_H + x_H^\top q_H + c_H$$

$$+ \sum_{h=0}^{H-1} \left(x_h^\top Q_h x_h + u_h^\top R_h u_h + u_h^\top M_h x_h + x_h^\top q_h + u_h^\top r_h + c_h \right)\Big]$$

such that $x_{h+1} = A_h x_h + B_h u_h + v_h + w_h, \quad x_0 \sim \mu_0, \quad u_h = \pi_h(x_h), \quad w_h \sim N(0, \sigma^2 I)$

More General Quadratic Cost Function

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\Big[x_H^\top Q_H x_H + x_H^\top q_H + c_H + \sum_{h=0}^{H-1}\left(x_h^\top Q_h x_h + u_h^\top R_h u_h + u_h^\top M_h x_h + x_h^\top q_h + u_h^\top r_h + c_h\right)\Big]$$

such that $x_{h+1} = A_h x_h + B_h u_h + v_h + w_h\,,\quad x_0 \sim \mu_0\,,\quad u_h = \pi_h(x_h)\,,\quad w_h \sim N(0, \sigma^2 I)$

Derivation is quite similar, just more algebra!

Tracking a Predefined Trajectory

Tracking a Predefined Trajectory

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\left[(x_H - x_H^\star)^\top Q_H (x_H - x_H^\star) + \sum_{h=0}^{H-1}\left((x_h - x_h^\star)^\top Q_h (x_h - x_h^\star) + (u_h - u_h^\star)^\top R_h (u_h - u_h^\star)\right)\right]$$

such that $x_{h+1} = A_h x_h + B_h u_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0,\sigma^2 I)$

Tracking a Predefined Trajectory

$$\arg\min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\Big[(x_H - x_H^\star)^\top Q_H (x_H - x_H^\star) + \sum_{h=0}^{H-1}\Big((x_h - x_h^\star)^\top Q_h (x_h - x_h^\star) + (u_h - u_h^\star)^\top R_h (u_h - u_h^\star)\Big)\Big]$$

such that $x_{h+1} = A_h x_h + B_h u_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0,\sigma^2 I)$

Can you see why we already know how to solve this?

Tracking a Predefined Trajectory

$$\arg\min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d\to\mathbb{R}^k} \mathbb{E}\Bigg[(x_H - x_H^\star)^\top Q_H (x_H - x_H^\star) + \sum_{h=0}^{H-1}\Big((x_h - x_h^\star)^\top Q_h (x_h - x_h^\star) + (u_h - u_h^\star)^\top R_h (u_h - u_h^\star)\Big)\Bigg]$$

such that $x_{h+1} = A_h x_h + B_h u_h + w_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)\,, \quad w_h \sim N(0,\sigma^2 I)$

Can you see why we already know how to solve this?

Expanding all the quadratic terms produces a special case of the previous slide!

Beyond LQR

Beyond LQR

So far: many extensions to LQR essentially reduce to the same problem

Beyond LQR

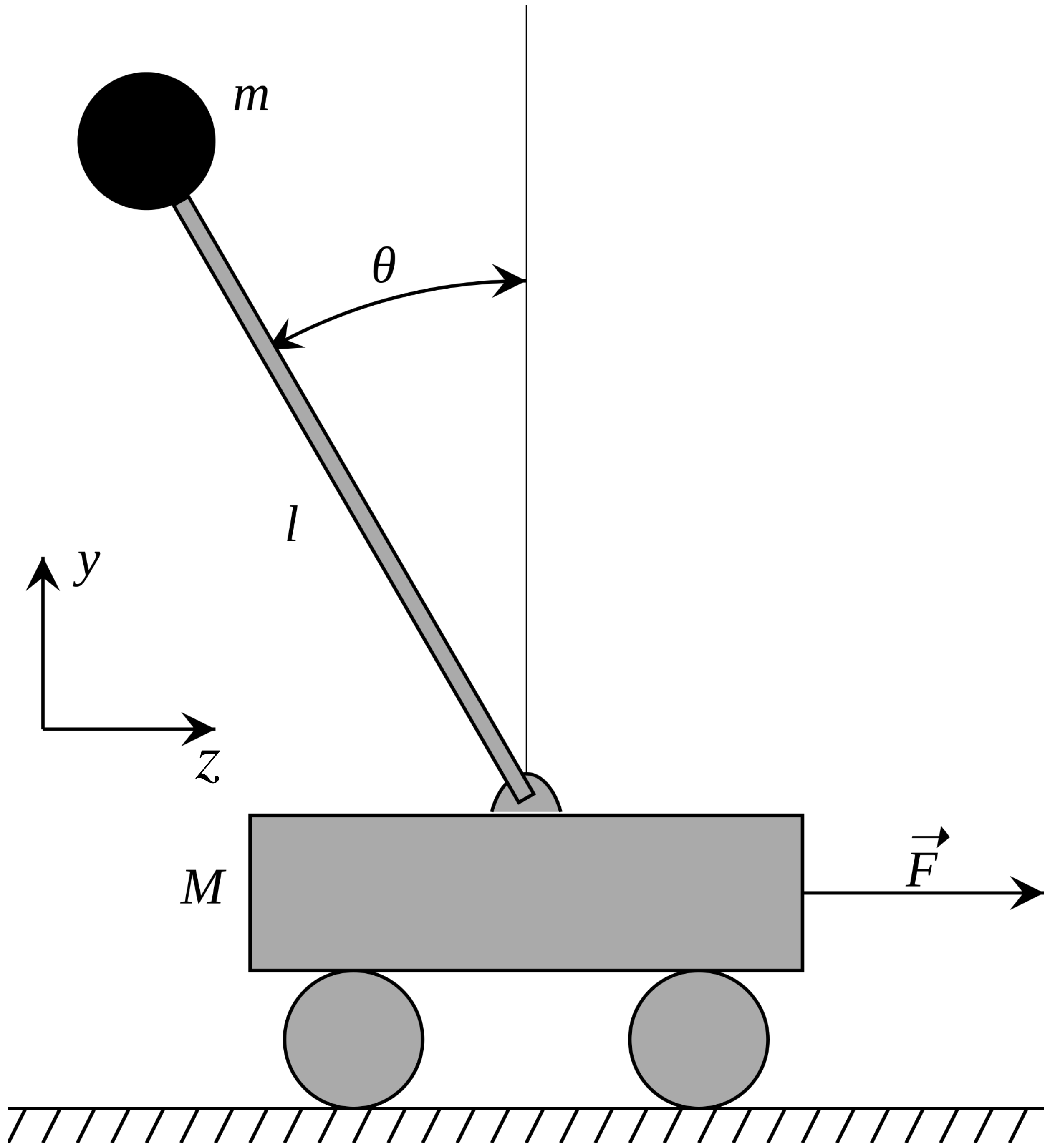
So far: many extensions to LQR essentially reduce to the same problem

But what about problems with nonlinear dynamics and/or nonquadratic costs?

Today

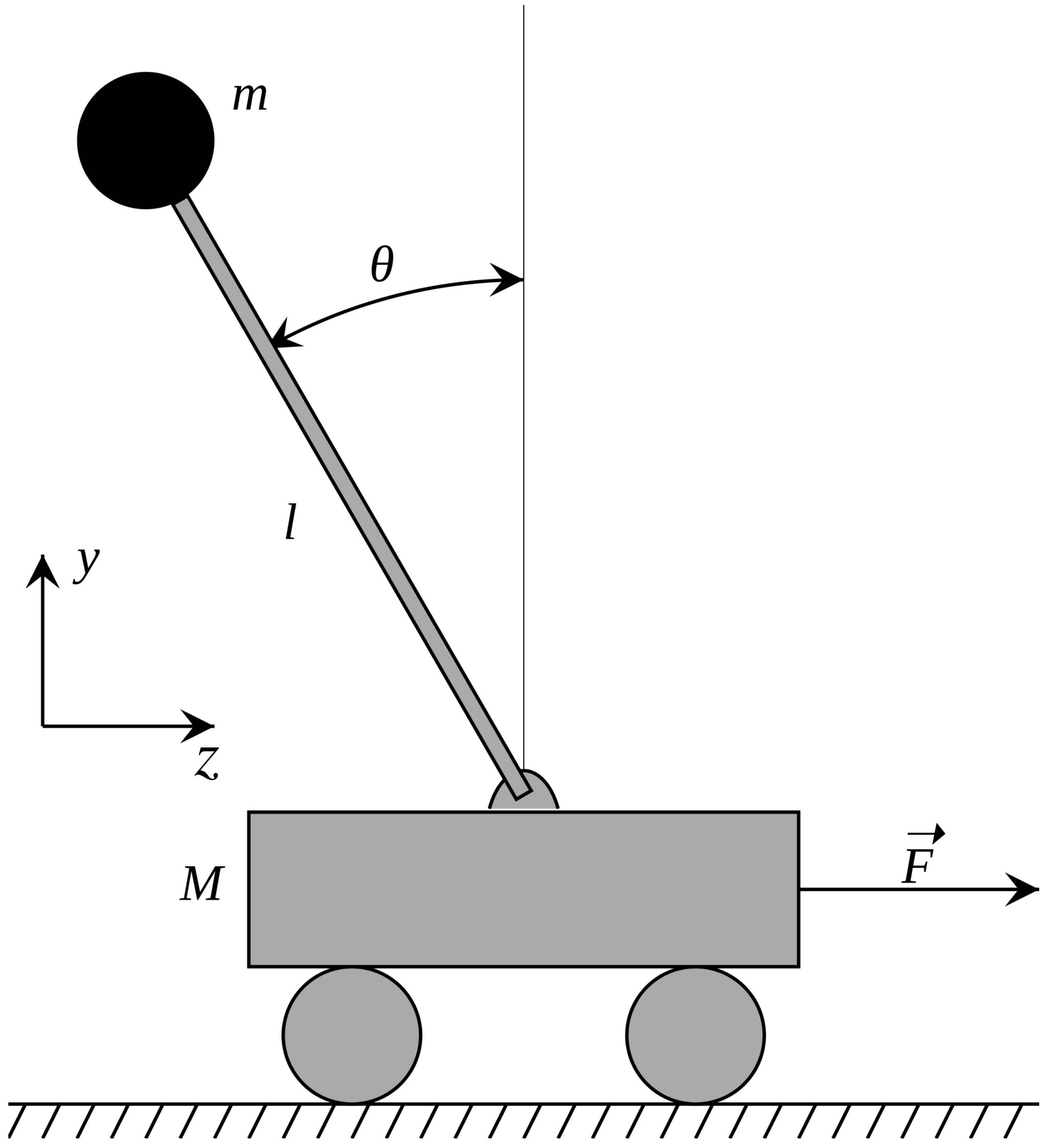
- ✓ Feedback from last lecture
- ✓ Recap
- Locally linearization
- Iterative LQR

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

Setting for Local Linearization Approach:

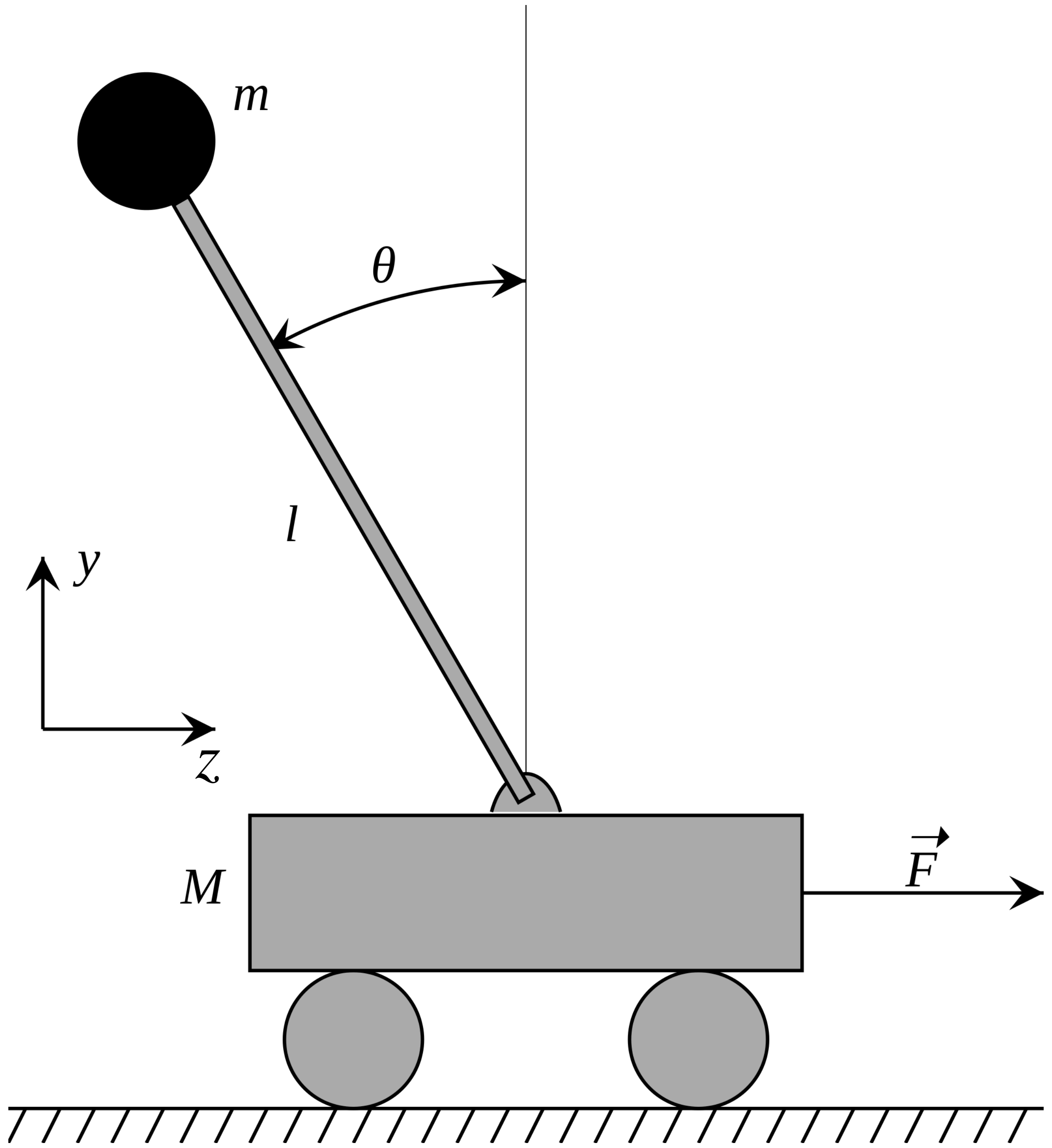


Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

$$\text{minimize } \mathbb{E}_\pi \Big[\sum_{h=0}^{H-1} c(x_h, u_h) \Big]$$

$$\text{s.t. } \; x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$$

Setting for Local Linearization Approach:



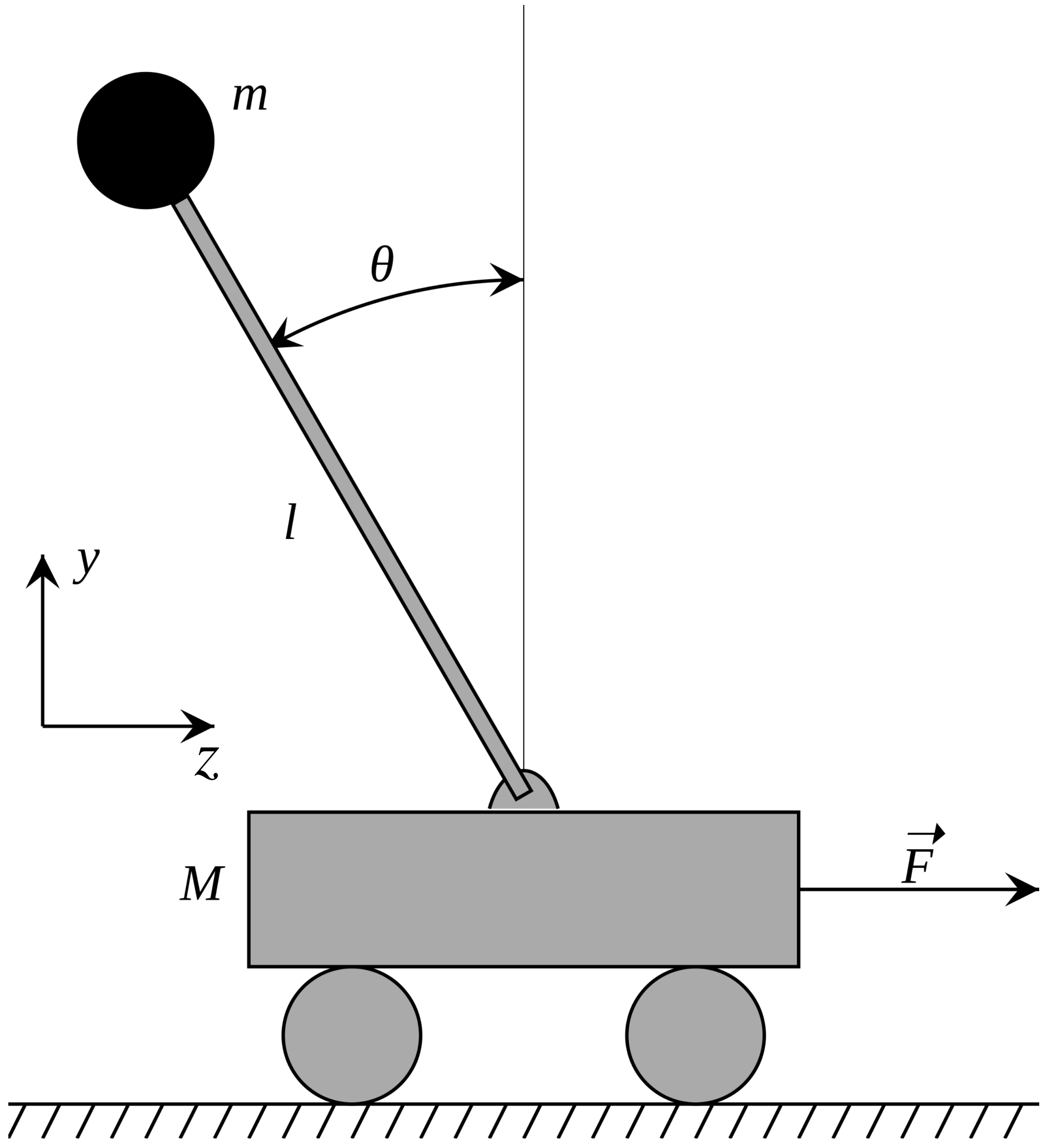
Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

minimize $\mathbb{E}_\pi\left[\sum_{h=0}^{H-1} c(x_h, u_h)\right]$

s.t. $x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$

No noise!

Setting for Local Linearization Approach:



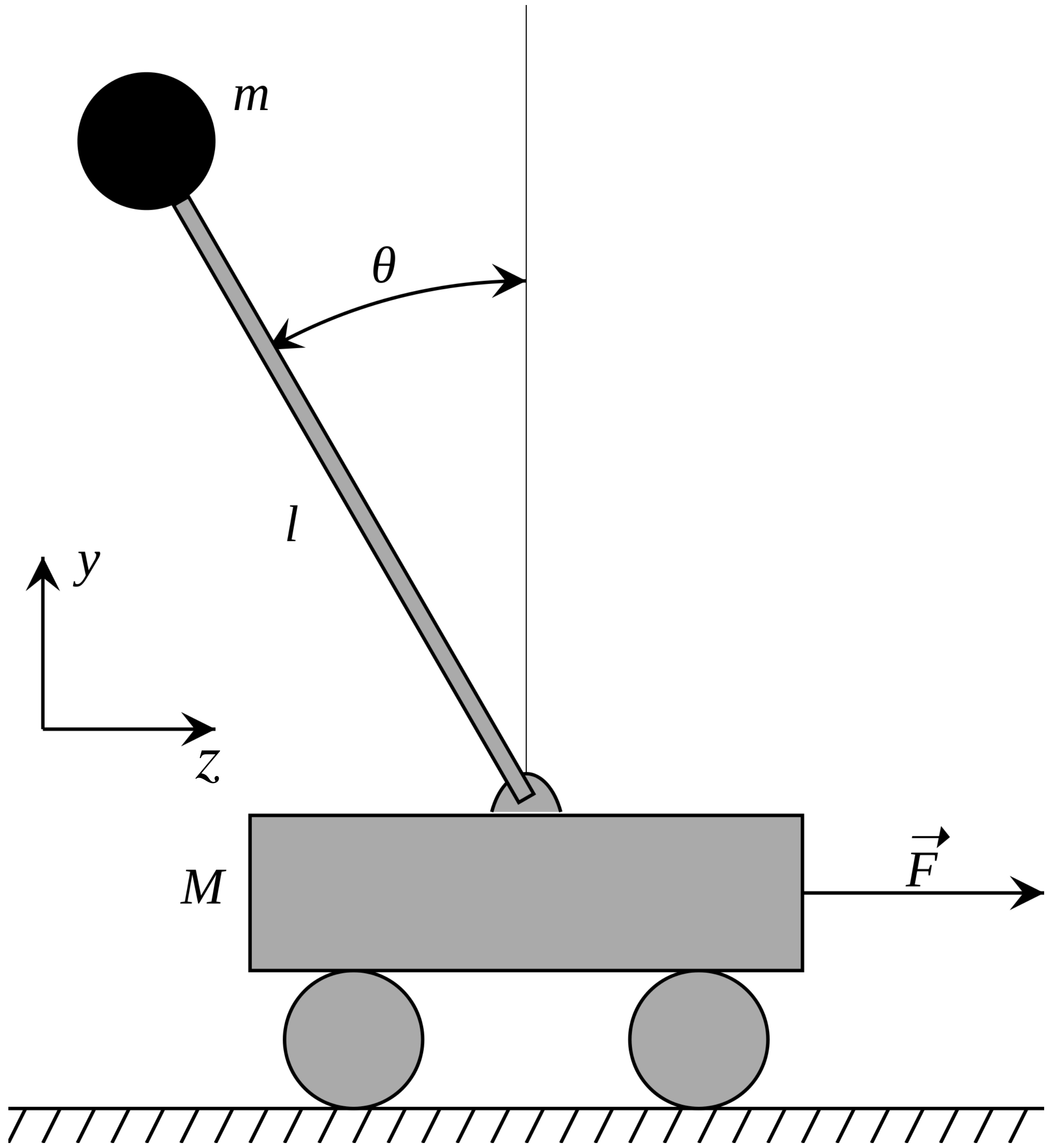
Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

minimize $\mathbb{E}_\pi\left[\sum_{h=0}^{H-1} c(x_h, u_h)\right]$

s.t. $x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$

No noise! No terminal cost $c_h(x_H)$!

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

minimize $\mathbb{E}_\pi\left[\sum_{h=0}^{H-1} c(x_h, u_h)\right]$

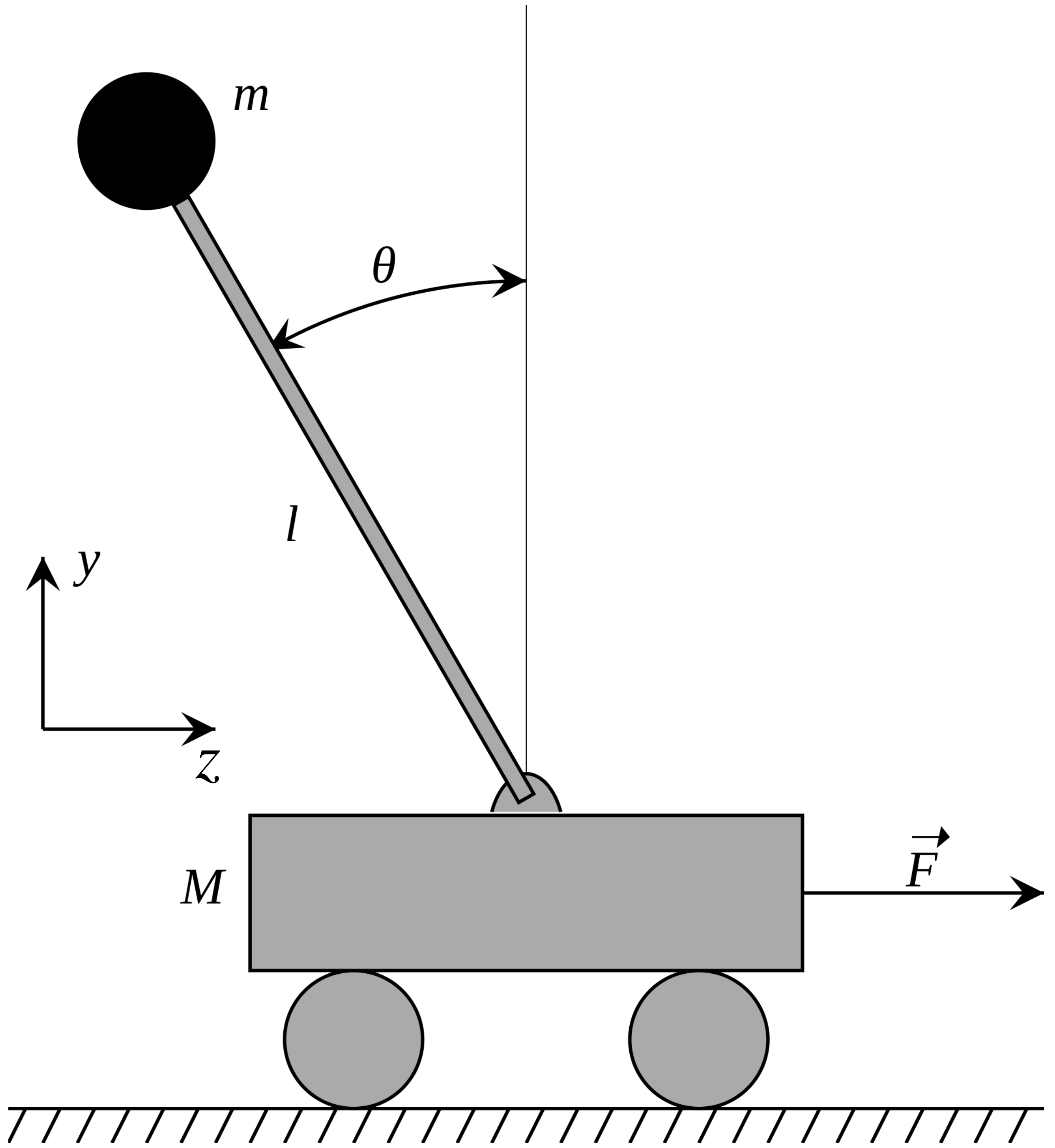
s.t. $x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$

No noise! No terminal cost $c_h(x_H)$!

Assumptions:

1. We have black-box access to f & c:

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

$$\text{minimize } \mathbb{E}_\pi\Big[\sum_{h=0}^{H-1} c(x_h, u_h)\Big]$$

s.t. $x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$

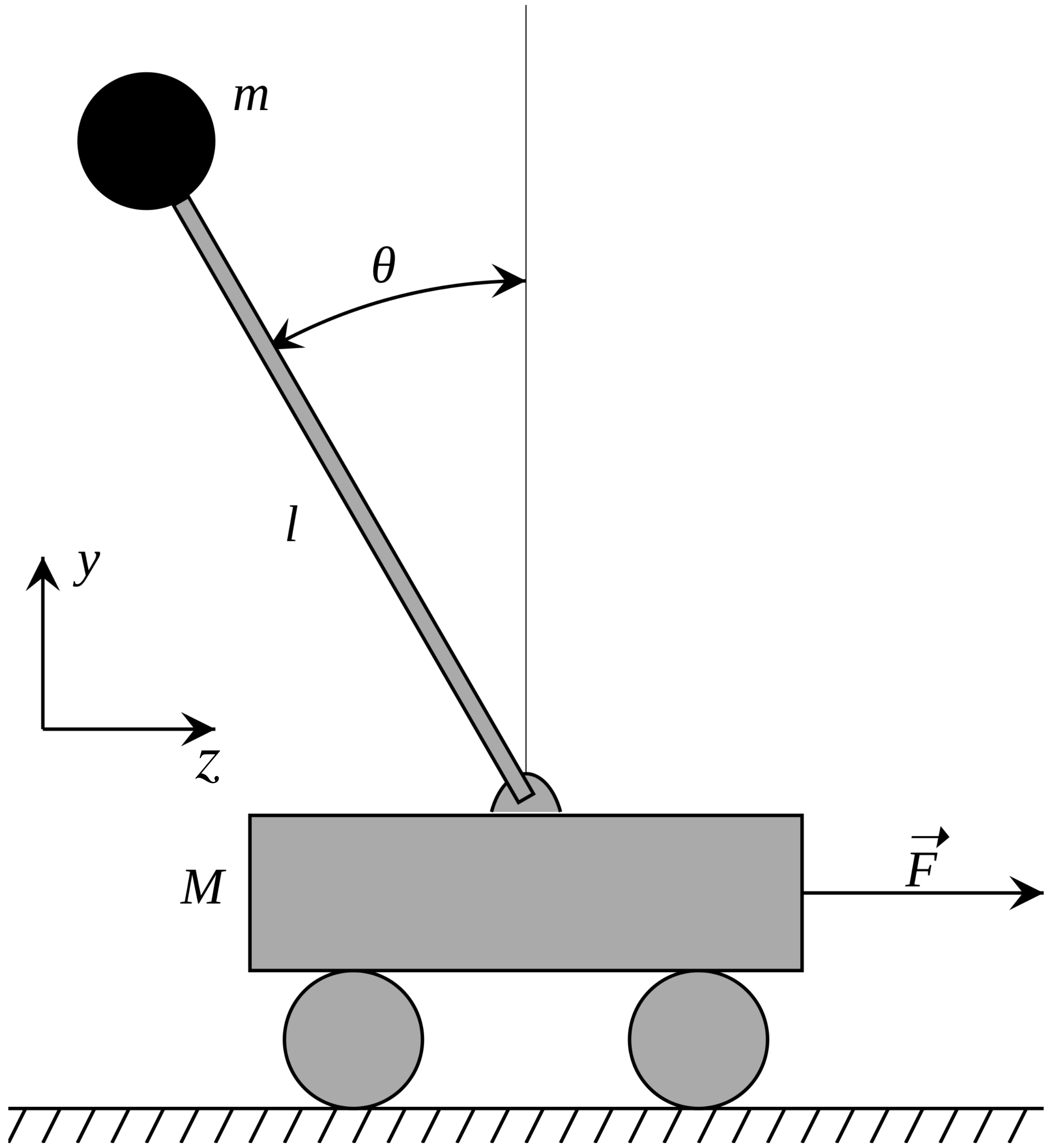
No noise! No terminal cost $c_h(x_H)$!

Assumptions:

1. We have black-box access to f & c:

f and c have unknown analytical form but can be queried at any (x, u) to give x', c, where $x' = f(x, u), c = c(x, u)$

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

minimize $\mathbb{E}_\pi \left[\sum_{h=0}^{H-1} c(x_h, u_h) \right]$

s.t. $x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$

No noise! No terminal cost $c_h(x_H)$!

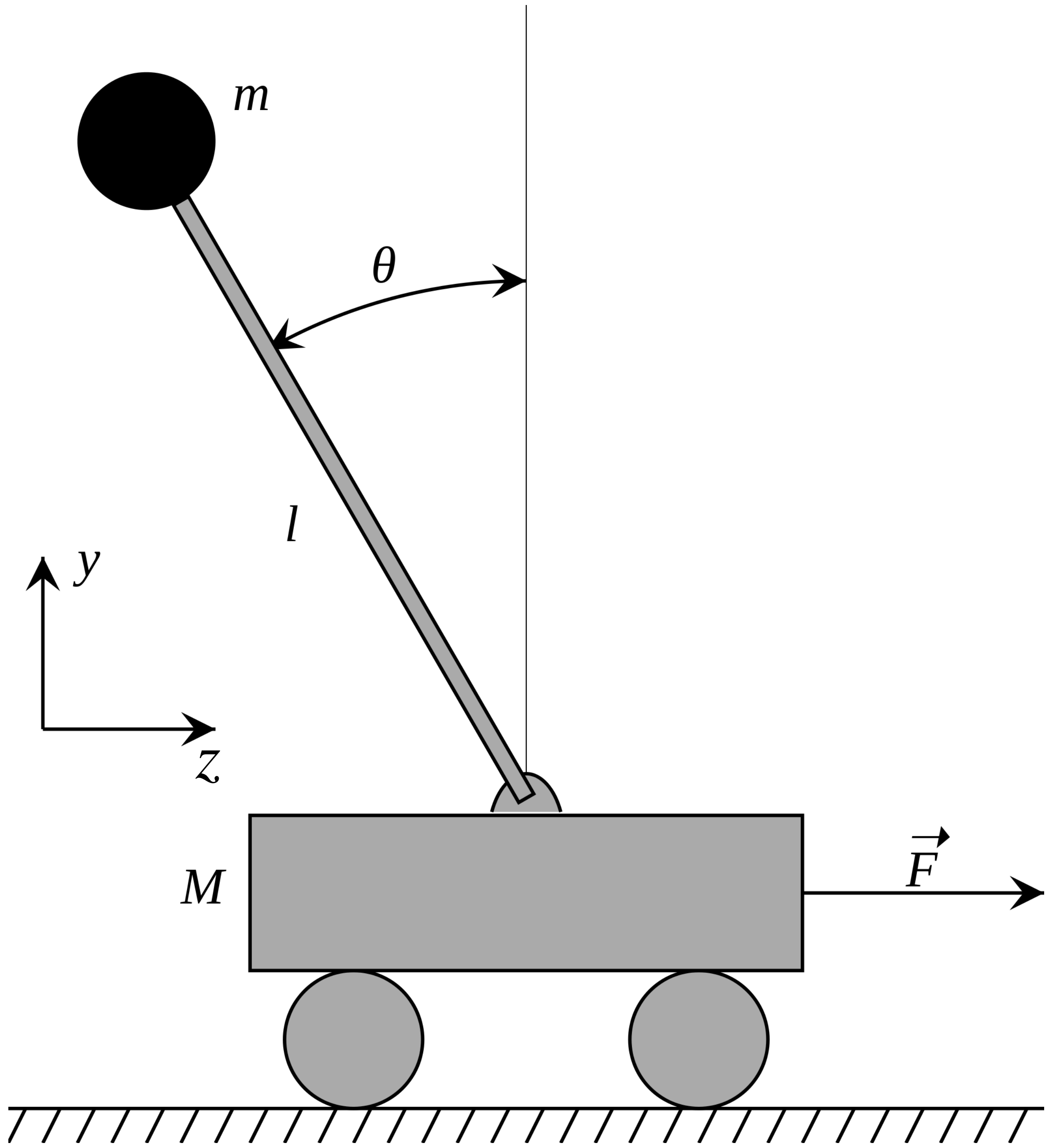
Assumptions:

1. We have black-box access to f & c:

f and c have unknown analytical form but can be queried at any (x, u) to give x', c, where $x' = f(x, u), c = c(x, u)$

2. f is differentiable and c is twice differentiable

Setting for Local Linearization Approach:



Goal: stabilizing around the goal $(x = x^\star, u = u^\star)$

minimize $\mathbb{E}_\pi \left[\sum_{h=0}^{H-1} c(x_h, u_h) \right]$

s.t. $x_{h+1} = f(x_h, u_h), \quad u_h = \pi(x_h), \quad x_0 \sim \mu_0$

No noise! No terminal cost $c_h(x_H)$!

Assumptions:

1. We have black-box access to f & c:

f and c have unknown analytical form but can be queried at any (x, u) to give x', c, where $x' = f(x, u), c = c(x, u)$

2. f is differentiable and c is twice differentiable

$\nabla_x f(x, u), \nabla_u f(x, u), \nabla_x c(x, u), \nabla_u c(x, u),$
$\nabla_x^2 c(x, u), \nabla_u^2 c(x, u), \nabla_{x,u}^2 c(x, u)$

Local Linearization of Dynamics

Local Linearization of Dynamics

Assume that all possible initial states x_0 are close to $x^\star$ and can be kept there with actions close to $u^\star$

Local Linearization of Dynamics

Assume that all possible initial states x_0 are close to $x^\star$ and can be kept there with actions close to $u^\star$

We can approximate $f(x, u)$ locally with a first-order Taylor expansion:

$$f(x,u) \approx f(x^\star, u^\star) + \nabla_x f(x^\star, u^\star)\left(x - x^\star\right) + \nabla_u f(x^\star, u^\star)(u - u^\star)$$

Local Linearization of Dynamics

Assume that all possible initial states x_0 are close to $x^\star$ and can be kept there with actions close to $u^\star$

We can approximate $f(x, u)$ locally with a first-order Taylor expansion:

$$f(x,u) \approx f(x^\star, u^\star) + \nabla_x f(x^\star, u^\star)\left(x - x^\star\right) + \nabla_u f(x^\star, u^\star)(u - u^\star)$$

where:

$$\nabla_x f(x,u) \in \mathbb{R}^{d\times d}, \quad \nabla_x f(x,u)[i,j] = \frac{\partial f[i]}{\partial x[j]}(x,u)$$

$$\nabla_u f(x,u) \in \mathbb{R}^{d\times k}, \quad \nabla_u f(x,u)[i,j] = \frac{\partial f[i]}{\partial u[j]}(x,u)$$

Local Quadratization of Cost Function

Local Quadratization of Cost Function

We can approximate $c(x, u)$ locally at $(x^\star, u^\star)$ with second-order Taylor expansion:

Local Quadratization of Cost Function

We can approximate $c(x, u)$ locally at $(x^\star, u^\star)$ with second-order Taylor expansion:

$$
\begin{aligned}
c(x,u) \approx\ & c(x^\star, u^\star) + \nabla_x c(x^\star, u^\star)^\top (x - x^\star) + \nabla_u c(x^\star, u^\star)^\top (u - u^\star) \\
& + \frac{1}{2}(x - x^\star)^\top \nabla_x^2 c(x^\star, u^\star)(x - x^\star) + \frac{1}{2}(u - u^\star)^\top \nabla_u^2 c(x^\star, u^\star)(u - u^\star) + (x - x^\star)^\top \nabla_{x,u}^2 c(x, u)(u - u^\star)
\end{aligned}
$$

Local Quadratization of Cost Function

We can approximate $c(x, u)$ locally at $(x^\star, u^\star)$ with second-order Taylor expansion:

$$c(x,u) \approx c(x^\star,u^\star) + \nabla_x c(x^\star,u^\star)^\top (x - x^\star) + \nabla_u c(x^\star,u^\star)^\top (u - u^\star)$$
$$+\frac{1}{2}(x - x^\star)^\top \nabla_x^2 c(x^\star,u^\star)(x - x^\star) + \frac{1}{2}(u - u^\star)^\top \nabla_u^2 c(x^\star,u^\star)(u - u^\star) + (x - x^\star)^\top \nabla_{x,u}^2 c(x,u)(u - u^\star)$$

$$\nabla_x c(x,u) \in \mathbb{R}^d, \quad \nabla_x c(x,u)[i] = \frac{\partial c}{\partial x[i]}(x,u),$$

$$\nabla_u c(x,u) \in \mathbb{R}^k, \quad \nabla_u c(x,u)[i] = \frac{\partial c}{\partial u[i]}(x,u),$$

$$\nabla_x^2 c(x,u) \in \mathbb{R}^{d\times d}, \quad \nabla_x^2 c(x,u)[i,j] = \frac{\partial^2 c}{\partial x[i]\partial x[j]}(x,u),$$

$$\nabla_{x,u}^2 c(x,u) \in \mathbb{R}^{d\times k}, \quad \nabla_{x,u}^2 c(x,u)[i,j] = \frac{\partial^2 c}{\partial x[i]\partial u[j]}(x,u)$$

Local Linearization: Putting it all Together

$$c(x,u) \approx c(x^\star,u^\star) + \nabla_x c(x^\star,u^\star)^\top (x - x^\star) + \nabla_u c(x^\star,u^\star)^\top (u - u^\star)$$
$$+\frac{1}{2}(x - x^\star)^\top \nabla_x^2 c(x^\star,u^\star)(x - x^\star) + \frac{1}{2}(u - u^\star)^\top \nabla_u^2 c(x^\star,u^\star)(u - u^\star) + (x - x^\star)^\top \nabla_{x,u}^2 c(x,u)(u - u^\star)$$

$$f(x,u) \approx f(x^\star,u^\star) + \nabla_x f(x^\star,u^\star)\left(x - x^\star\right) + \nabla_u f(x^\star,u^\star)(u - u^\star)$$

Local Linearization: Putting it all Together

$$c(x,u) \approx c(x^\star, u^\star) + \nabla_x c(x^\star, u^\star)^\top (x - x^\star) + \nabla_u c(x^\star, u^\star)^\top (u - u^\star)$$
$$+\frac{1}{2}(x - x^\star)^\top \nabla_x^2 c(x^\star, u^\star)(x - x^\star) + \frac{1}{2}(u - u^\star)^\top \nabla_u^2 c(x^\star, u^\star)(u - u^\star) + (x - x^\star)^\top \nabla_{x,u}^2 c(x,u)(u - u^\star)$$

$$f(x,u) \approx f(x^\star, u^\star) + \nabla_x f(x^\star, u^\star)\left(x - x^\star\right) + \nabla_u f(x^\star, u^\star)(u - u^\star)$$

Rearranging terms, we get back to the following formulation:

$$\arg \min_{\pi_0,\ldots,\pi_{H-1}:\mathbb{R}^d \to \mathbb{R}^k} \quad \mathbb{E}\left[\sum_{h=0}^{H-1} (x_h^\top Q x_h + u_h^\top R u_h + u_h^\top M x_h + x_h^\top q + u_h^\top r + c)\right]$$

such that $\quad x_{h+1} = A x_h + B u_h + v\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)$

Special case of one of the LQR extensions!

Summary of Local Linearization So Far:

For tasks such as balancing near goal state $(x^\star, u^\star)$,
we can perform **first order Taylor expansion on** $f(x, u)$,
and **second order Taylor expansion on** $c(x, u)$ around the balancing point $(x^\star, u^\star)$

$$\arg \min_{\pi_0, \ldots, \pi_{H-1} : \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{h=0}^{H-1} (x_h^\top Q x_h + u_h^\top R u_h + u_h^\top M x_h + x_h^\top q + u_h^\top r + c) \right]$$

such that $\quad x_{h+1} = A x_h + B u_h + v\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)$

Summary of Local Linearization So Far:

For tasks such as balancing near goal state $(x^\star, u^\star)$,
we can perform **first order Taylor expansion on** $f(x, u)$,
and **second order Taylor expansion on** $c(x, u)$ around the balancing point $(x^\star, u^\star)$

$$\arg \min_{\pi_0, \ldots, \pi_{H-1}: \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{h=0}^{H-1} (x_h^\top Q x_h + u_h^\top R u_h + u_h^\top M x_h + x_h^\top q + u_h^\top r + c) \right]$$

such that $x_{h+1} = A x_h + B u_h + v, \quad x_0 \sim \mu_0, \quad u_h = \pi_h(x_h)$

Last step: checking some practical issues

Locally Convexifying the Cost Function

Locally Convexifying the Cost Function

Note that $c(x, u)$ might not even be convex;

So, $\nabla_x^2 c(x^\star, u^\star)$ & $\nabla_u^2 c(x^\star, u^\star)$ may not be positive definite

Locally Convexifying the Cost Function

Note that $c(x, u)$ might not even be convex;

So, $\nabla_x^2 c(x^\star, u^\star)$ & $\nabla_u^2 c(x^\star, u^\star)$ may not be positive definite

What can we do?

Locally Convexifying the Cost Function

Note that $c(x, u)$ might not even be convex;

So, $\nabla_x^2 c(x^\star, u^\star)$ & $\nabla_u^2 c(x^\star, u^\star)$ may not be positive definite

What can we do?

In practice, we force them to be positive definite:

Locally Convexifying the Cost Function

Note that $c(x, u)$ might not even be convex;

So, $\nabla_x^2 c(x^\star, u^\star)$ & $\nabla_u^2 c(x^\star, u^\star)$ may not be positive definite

What can we do?

In practice, we force them to be positive definite:

Given a *symmetric matrix* $W \in \mathbb{R}^{d\times d}$,

we compute the eigen-decomposition $W = \sum_{i=1}^{d} \sigma_i z_i z_i^\top$, and we approximate W as

$$W \approx \sum_{i=1}^{d} \mathbf{1}(\sigma_i > 0)\sigma_i z_i z_i^\top + \lambda I,$$

for some small $\lambda > 0$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (x, u), the black boxes output x', c, where

$$x' = f(x, u), c = c(x, u)$$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (x, u), the black boxes output x', c, where

$$x' = f(x, u), c = c(x, u)$$

Compute gradient using finite differencing:

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (x, u), the black boxes output x', c, where

$$x' = f(x, u), c = c(x, u)$$

Compute gradient using finite differencing:

$$\frac{\partial f[i]}{\partial x[j]}(x, u) \approx \frac{f(x + \delta_j, u)[i] - f(x - \delta_j, u)[i]}{2\delta}, \text{ where } \delta_j = [0, \ldots, 0, \underbrace{\delta}_{j'th \text{ entry}}, 0, \ldots 0]^\top$$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (x, u), the black boxes output x', c, where

$$x' = f(x, u), c = c(x, u)$$

Compute gradient using finite differencing:

$$\frac{\partial f[i]}{\partial x[j]}(x, u) \approx \frac{f(x + \delta_j, u)[i] - f(x - \delta_j, u)[i]}{2\delta}, \text{ where } \delta_j = [0, \ldots, 0, \underbrace{\delta}_{j'th \text{ entry}}, 0, \ldots 0]^\top$$

To compute second derivative, e.g., $\frac{\partial^2 c}{\partial x[i] \partial u[j]}(x, u)$

Computing Approximate Derivatives

Recall our assumption: we only have black-box access to f & c:

i.e., unknown analytical form, but given any (x, u), the black boxes output x', c, where

$$x' = f(x, u), c = c(x, u)$$

Compute gradient using finite differencing:

$$\frac{\partial f[i]}{\partial x[j]}(x, u) \approx \frac{f(x + \delta_j, u)[i] - f(x - \delta_j, u)[i]}{2\delta}, \text{ where } \delta_j = [0,\ldots,0, \underbrace{\delta}_{j'th \text{ entry}}, 0, \ldots 0]^\top$$

To compute second derivative, e.g., $\frac{\partial^2 c}{\partial x[i] \partial u[j]}(x, u)$

First implement finite differencing procedure for $\partial c / \partial x[i]$, and then perform another finite differencing with respect to $u[j]$ on top of the first finite differencing procedure for $\partial c / \partial x[i]$

Summary for local linearization approach

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(x, u)$
and second order Taylor expansion on $c(x, u)$, both around the balancing point $(x^\star, u^\star)$

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(x, u)$
and second order Taylor expansion on $c(x, u)$, both around the balancing point $(x^\star, u^\star)$

2. Force Hessians $\nabla_x^2 c(x, u)$ & $\nabla_u^2 c(x, u)$ to be positive definite

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(x, u)$
and second order Taylor expansion on $c(x, u)$, both around the balancing point $(x^\star, u^\star)$

2. Force Hessians $\nabla_x^2 c(x, u)$ & $\nabla_u^2 c(x, u)$ to be positive definite

3. Leverage finite differences to approximate gradients and Hessians

Summary for local linearization approach

1. Perform first order Taylor expansion on $f(x, u)$
and second order Taylor expansion on $c(x, u)$, both around the balancing point $(x^\star, u^\star)$

2. Force Hessians $\nabla_x^2 c(x, u)$ & $\nabla_u^2 c(x, u)$ to be positive definite

3. Leverage finite differences to approximate gradients and Hessians

4. The approximation is a (direct extension of) LQR, so we know how to compute the optimal policy

Today

- ✓ Feedback from last lecture
- ✓ Recap
- ✓ Locally linearization
- Iterative LQR

Limits of Local Linearization

Limits of Local Linearization

Local linearization can work if x_0 is very close to $x^\star$ and stays there with near-optimal (i.e., near-$u^\star$) actions

Limits of Local Linearization

Local linearization can work if x_0 is very close to $x^\star$ and stays there with near-optimal (i.e., near-$u^\star$) actions

But when x_h is far away from $x^\star$ or u_h needs to be far from $u^\star$ for any h, first/second-order Taylor expansion is not accurate anymore

Idea of Iterative LQR

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(x^\star, u^\star)$, linearize/quadratize around some other $(\bar{x}, \bar{u})$

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(x^\star, u^\star)$, linearize/quadratize around some other $(\bar{x}, \bar{u})$

In fact, we can even linearize/quadratize around different points $(\bar{x}_h, \bar{u}_h)$ at each h

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(x^\star, u^\star)$, linearize/quadratize around some other $(\bar{x}, \bar{u})$

In fact, we can even linearize/quadratize around different points $(\bar{x}_h, \bar{u}_h)$ at each h

After linearization and quadratization at time h around waypoint $(\bar{x}_h, \bar{u}_h)$, $\forall h$, re-arranging terms gives:

$$\arg \min_{\pi_0, \ldots, \pi_{H-1}: \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{h=0}^{H-1} (x_h^\top Q_h x_h + u_h^\top R_h u_h + u_h^\top M_h x_h + x_h^\top q_h + u_h^\top r_h + c_h) \right]$$

such that $x_{h+1} = A_h x_h + B_h u_h + v_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)$

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(x^\star, u^\star)$, linearize/quadratize around some other $(\bar{x}, \bar{u})$

In fact, we can even linearize/quadratize around different points $(\bar{x}_h, \bar{u}_h)$ at each h

After linearization and quadratization at time h around waypoint $(\bar{x}_h, \bar{u}_h)$, $\forall h$, re-arranging terms gives:

$$\arg \min_{\pi_0, \ldots, \pi_{H-1}: \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{h=0}^{H-1} (x_h^\top Q_h x_h + u_h^\top R_h u_h + u_h^\top M_h x_h + x_h^\top q_h + u_h^\top r_h + c_h)\right]$$

such that $x_{h+1} = A_h x_h + B_h u_h + v_h\,, \quad x_0 \sim \mu_0\,, \quad u_h = \pi_h(x_h)$

Time-dependent LQR problem: we know the solution

Idea of Iterative LQR

Instead of linearizing/quadratizing around $(x^\star, u^\star)$, linearize/quadratize around some other $(\bar{x}, \bar{u})$

In fact, we can even linearize/quadratize around different points $(\bar{x}_h, \bar{u}_h)$ at each h

After linearization and quadratization at time h around waypoint $(\bar{x}_h, \bar{u}_h)$, $\forall h$, re-arranging terms gives:

$$\arg \min_{\pi_0, \ldots, \pi_{H-1} : \mathbb{R}^d \to \mathbb{R}^k} \mathbb{E}\left[\sum_{h=0}^{H-1} (x_h^\top Q_h x_h + u_h^\top R_h u_h + u_h^\top M_h x_h + x_h^\top q_h + u_h^\top r_h + c_h) \right]$$

such that $x_{h+1} = A_h x_h + B_h u_h + v_h \,, \quad x_0 \sim \mu_0 \,, \quad u_h = \pi_h(x_h)$

Time-dependent LQR problem: we know the solution

Question: how to choose the waypoints $(\bar{x}_h, \bar{u}_h)$ to get the best approximation/solution?

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

Note that although true f is stationary, its approximation f_h is not

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_h is not

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

For each h, quadratize $c_h(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$c_h(x, u) \approx \frac{1}{2} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_{x,u}^2 c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_{u,x}^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_u^2 c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}$$

$$+ \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_u c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} + c(\bar{x}_h^i, \bar{u}_h^i)$$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_h is not

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

For each h, quadratize $c_h(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$c_h(x, u) \approx \frac{1}{2} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_{x,u}^2 c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_{u,x}^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_u^2 c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}$$

$$+ \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_u c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} + c(\bar{x}_h^i, \bar{u}_h^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{H-1}^i$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

Note that although true f is stationary, its approximation f_h is not

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

For each h, quadratize $c_h(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$c_h(x, u) \approx \frac{1}{2} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_{x,u}^2 c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_{u,x}^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_u^2 c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}$$

$$+ \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_u c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} + c(\bar{x}_h^i, \bar{u}_h^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{H-1}^i$

Set new nominal trajectory: $\bar{x}_0^{i+1} = \bar{x}_0, \;\; \bar{u}_h^{i+1} = \pi_h^i(\bar{x}_h^{i+1}), \;$ and $\bar{x}_{h+1}^{i+1} = f(\bar{x}_h^{i+1}, \bar{u}_h^{i+1})$

Iterative LQR (iLQR)

Recall $x_0 \sim \mu_0$; denote $\mathbb{E}_{x_0 \sim \mu_0}[x_0] = \bar{x}_0$

Initialize $\bar{u}_0^0, \ldots, \bar{u}_{H-1}^0$, (how might we do this?)

Generate nominal trajectory: $\bar{x}_0^0 = \bar{x}_0, \bar{u}_0^0, \ldots, \bar{u}_h^0, \bar{x}_{h+1}^0 = f(\bar{x}_h^0, \bar{u}_h^0), \ldots, \bar{x}_{H-1}^0, \bar{u}_{H-1}^0$

For $i = 0,1,\ldots$

For each h, linearize $f(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

Note that although true f is stationary, its approximation f_h is not

$$f_h(x, u) \approx f(\bar{x}_h^i, \bar{u}_h^i) + \nabla_x f(\bar{x}_h^i, \bar{u}_h^i)(x - \bar{x}_h^i) + \nabla_u f(\bar{x}_h^i, \bar{u}_h^i)(u - \bar{u}_h^i)$$

For each h, quadratize $c_h(x, u)$ at $(\bar{x}_h^i, \bar{u}_h^i)$:

$$c_h(x, u) \approx \frac{1}{2} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_{x,u}^2 c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_{u,x}^2 c(\bar{x}_h^i, \bar{u}_h^i) & \nabla_u^2 c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}$$

$$+ \begin{bmatrix} x - \bar{x}_h^i \\ u - \bar{u}_h^i \end{bmatrix}^\top \begin{bmatrix} \nabla_x c(\bar{x}_h^i, \bar{u}_h^i) \\ \nabla_u c(\bar{x}_h^i, \bar{u}_h^i) \end{bmatrix} + c(\bar{x}_h^i, \bar{u}_h^i)$$

Formulate **time-dependent** LQR and compute its optimal control $\pi_0^i, \ldots, \pi_{H-1}^i$

Set new nominal trajectory: $\bar{x}_0^{i+1} = \bar{x}_0, \;\; \bar{u}_h^{i+1} = \pi_h^i(\bar{x}_h^{i+1}), \;$ and $\bar{x}_{h+1}^{i+1} = f(\bar{x}_h^{i+1}, \bar{u}_h^{i+1})$

Note this is true f, not approximation

Practical Considerations of Iterative LQR:

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}^i_0, \ldots, \bar{u}^i_{H-1}$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}_0^i, \ldots, \bar{u}_{H-1}^i$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}_h^{i+1} := \alpha\, \bar{u}_h^i + (1-\alpha)\bar{u}_h$ has the smallest cost,

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}^i_0, \ldots, \bar{u}^i_{H-1}$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}^{i+1}_h := \alpha\, \bar{u}^i_h + (1-\alpha)\bar{u}_h$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{h=0}^{H-1} c(x_h, \bar{u}^{i+1}_h)$$

$$\text{s.t.} \quad x_{h+1} = f(x_h, \bar{u}^{i+1}_h), \quad \bar{u}^{i+1}_h = \alpha \bar{u}^i_h + (1-\alpha)\bar{u}_h, \quad x_0 = \bar{x}_0$$

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}^i_0, \ldots, \bar{u}^i_{H-1}$, and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}^{i+1}_h := \alpha\, \bar{u}^i_h + (1-\alpha)\bar{u}_h$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{h=0}^{H-1} c(x_h, \bar{u}^{i+1}_h)$$

$$\text{s.t.} \quad x_{h+1} = f(x_h, \bar{u}^{i+1}_h), \quad \bar{u}^{i+1}_h = \alpha \bar{u}^i_h + (1-\alpha)\bar{u}_h, \quad x_0 = \bar{x}_0$$

Why is this tractable?

Practical Considerations of Iterative LQR:

1. We still want to use the eigen-decomposition trick to ensure positive definite Hessians

2. Still want to use finite differences to approximate derivatives

3. We want to use line-search to get monotonic improvement:

Given the previous nominal control $\bar{u}_0^i, \ldots, \bar{u}_{H-1}^i,$ and the latest computed controls $\bar{u}_0, \ldots, \bar{u}_{H-1}$

We want to find $\alpha \in [0,1]$ such that $\bar{u}_h^{i+1} := \alpha\, \bar{u}_h^i + (1-\alpha)\bar{u}_h$ has the smallest cost,

$$\min_{\alpha \in [0,1]} \sum_{h=0}^{H-1} c(x_h, \bar{u}_h^{i+1})$$

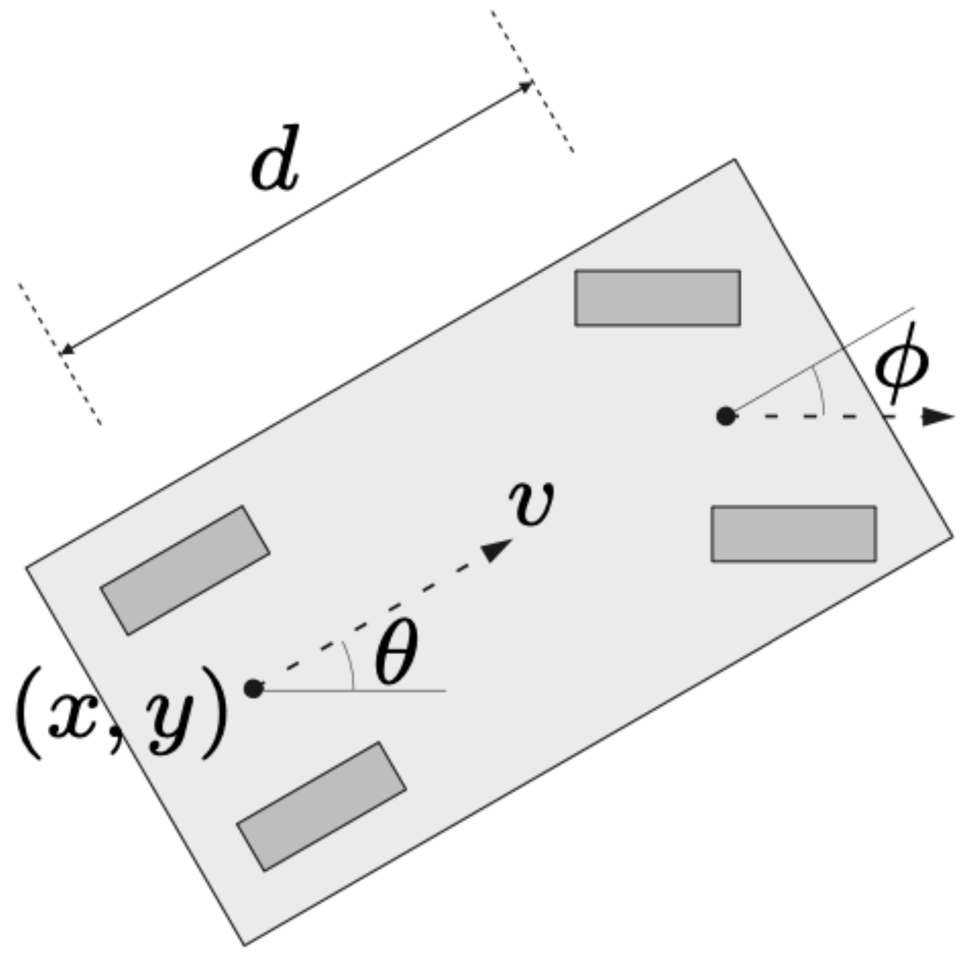
s.t. $x_{h+1} = f(x_h, \bar{u}_h^{i+1}), \quad \bar{u}_h^{i+1} = \alpha \bar{u}_h^i + (1-\alpha)\bar{u}_h, \quad x_0 = \bar{x}_0$

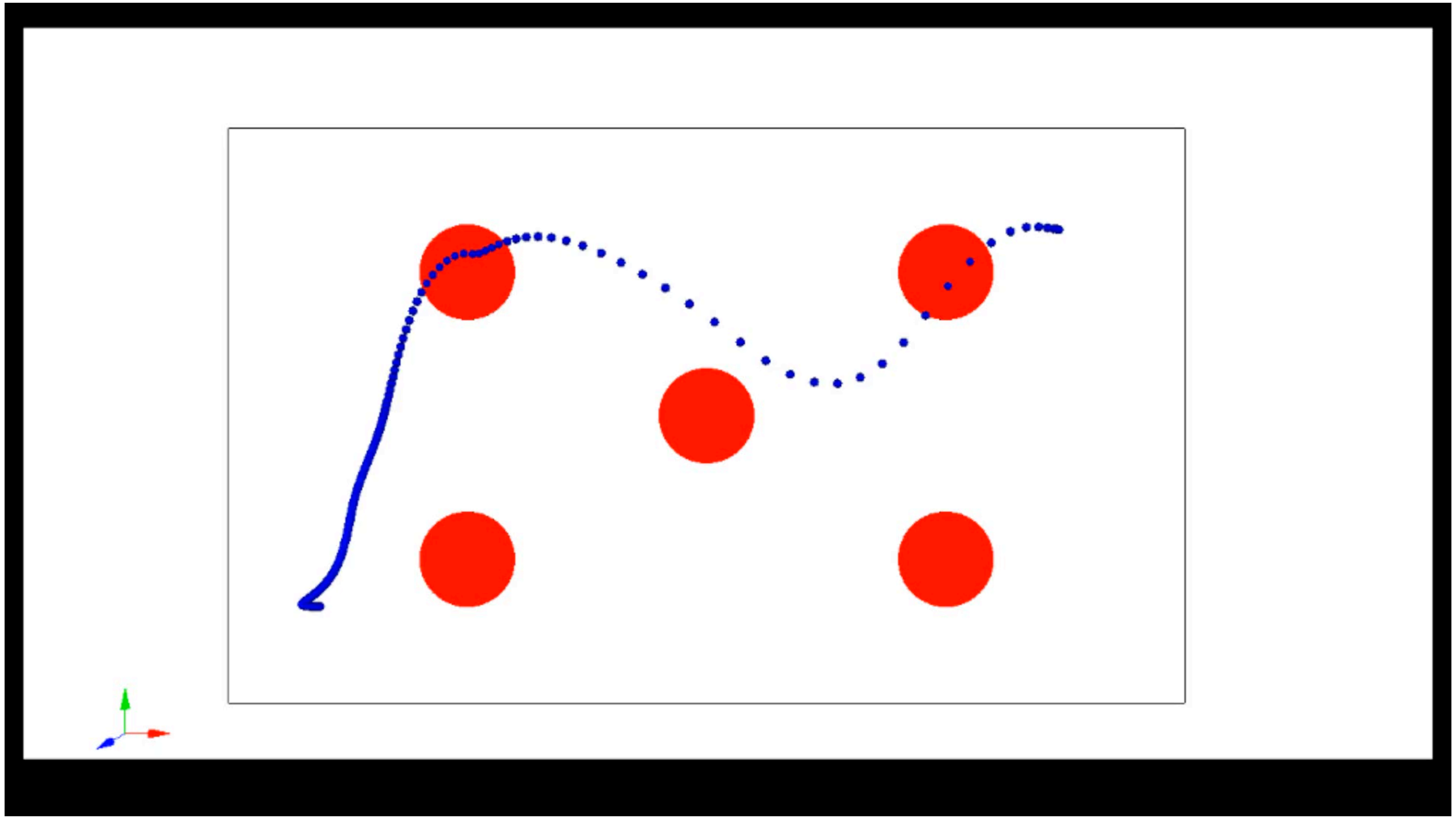
Why is this tractable? because it is 1-dimensional!

Example:

2-d car navigation

Cost function is designed such that it gets to the goal without colliding with obstacles (in red)

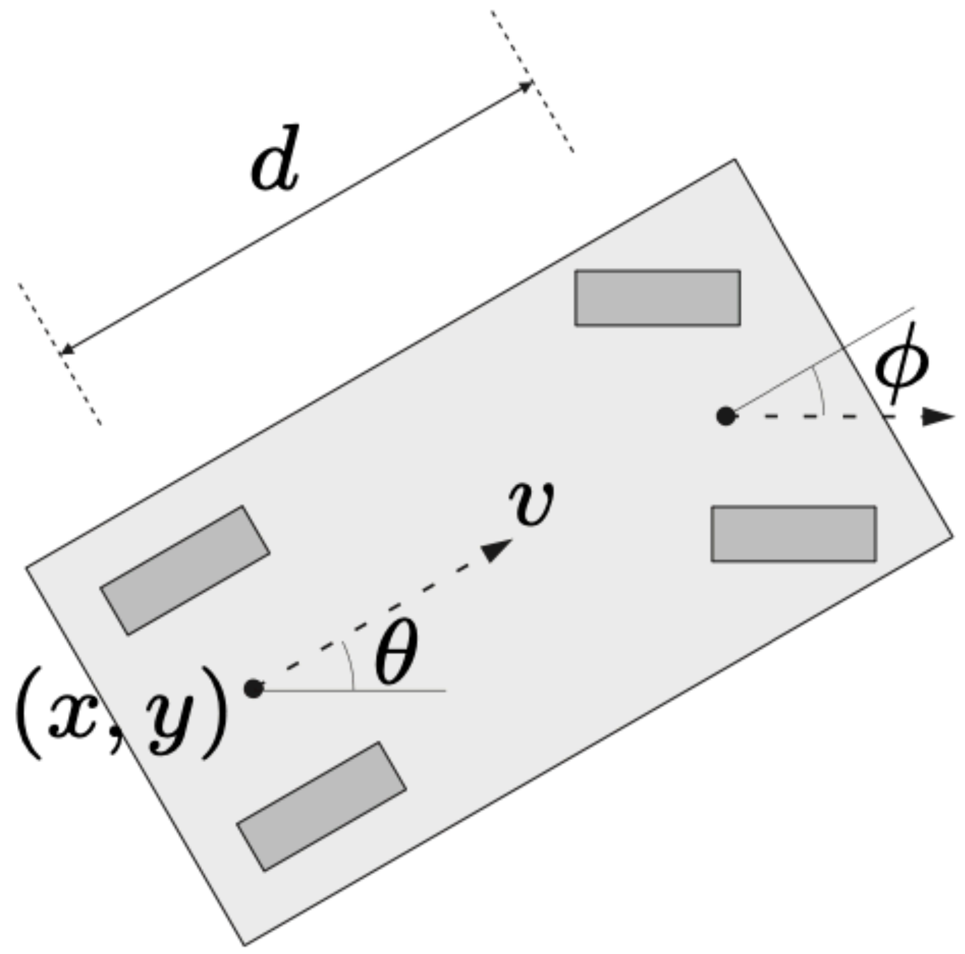


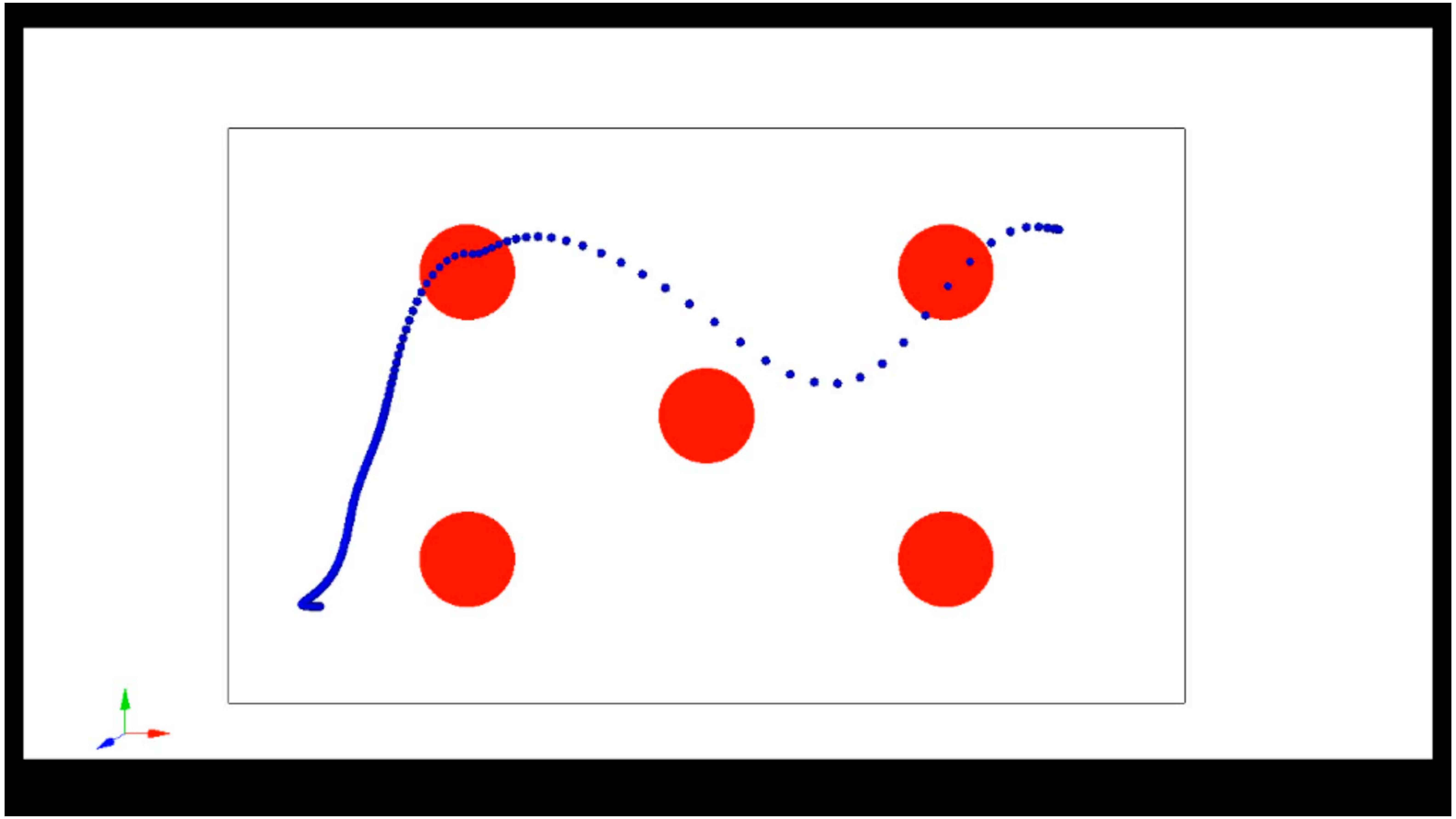


Example:

2-d car navigation

Cost function is designed such that it gets to the goal without colliding with obstacles (in red)





Summary of LQR extended to nonlinear control:

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Iterative LQR

Iterate between:

(1) forming an LQR around the current nominal trajectory,

(2) computing a new nominal trajectory using the optimal policy of the LQR

Summary of LQR extended to nonlinear control:

Local Linearization:

Approximate an LQR at the balance (goal) position $(x^\star, u^\star)$ and then solve the approximated LQR

Computes an approximately globally optimal solution for a small class of nonlinear control problems

Iterative LQR

Iterate between:

(1) forming an LQR around the current nominal trajectory,

(2) computing a new nominal trajectory using the optimal policy of the LQR

Computes a locally optimal (in policy space) solution for a large class of nonlinear control problems

Today

- Feedback from last lecture
- Recap
- Locally linearization
- Iterative LQR

Summary:

Local linearization

- Allows us to approximately optimally control any system near its optimum

Iterative LQR

- Uses LQR approximation to find locally optimal nonlinear control solution

Attendance:

bit.ly/3RcTC9T

Feedback:

bit.ly/3RHtlxy