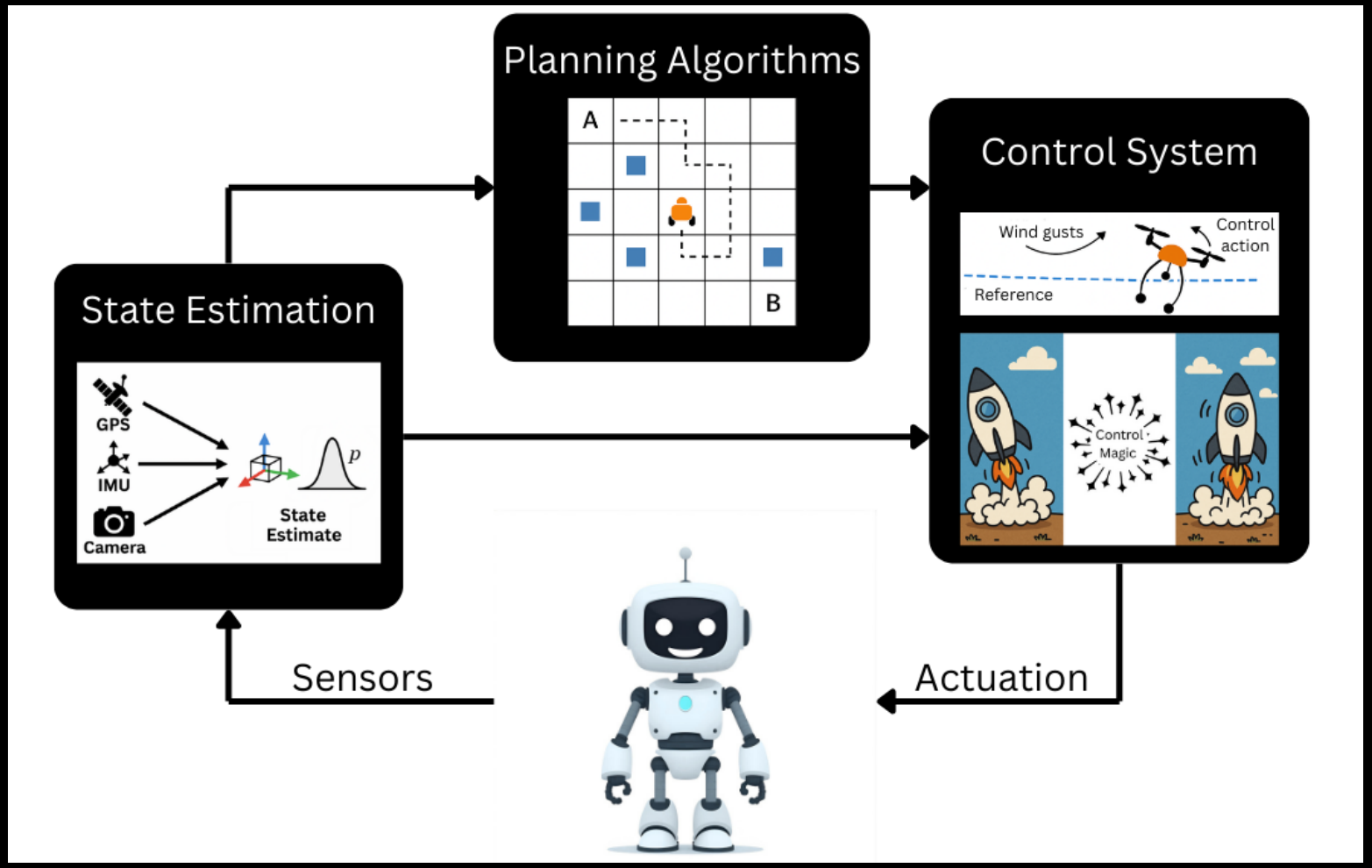


Control Stack Workshop 1

08. November 2025

Joschua Wüthrich

About Me

- PhD Student in Learning-enabled Control at the Intelligent Control Systems Group
- MSc and BSc in Mechanical Engineering
- Project experiences: SpaceHopper, Digital Twins of Surgical Procedures, Formula 1 Powertrain
- Why did I specialize in controls? It‘s a combination of applied math and engineering where abstract math concepts become tangible

Contents

- Intro to Control
- PID Control
- LQR Control
- Optimization Fundamentals
- MPC Basics

- Assumptions: perfect states and given reference trajectories
- Next week: State estimation and planning

Today‘s Case Study: Inverted Pendulum

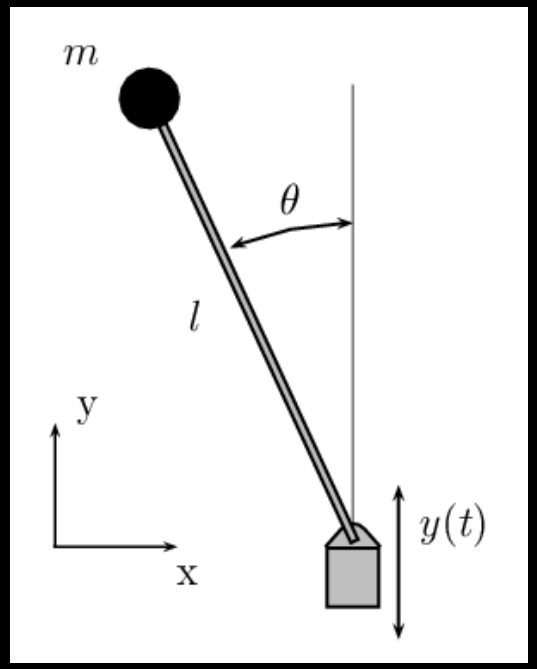
- Might seem boring, but many systems are approximated like that (such as vertical rocket landing)

- Two settings:
  - Rotating inverted pendulum (1 input, 1 output)
  - Cart-Pole system (1 input, 2 outputs)

Rotating Inverted Pendulum – Model

Dynamics:

$$x_1 = \theta, x_2 = \dot{\theta}$$
$$\dot{x_1} = x_2$$
$$\mathrm{I}\dot{x_2} = u + mgL \sin x_1 - bx_2$$



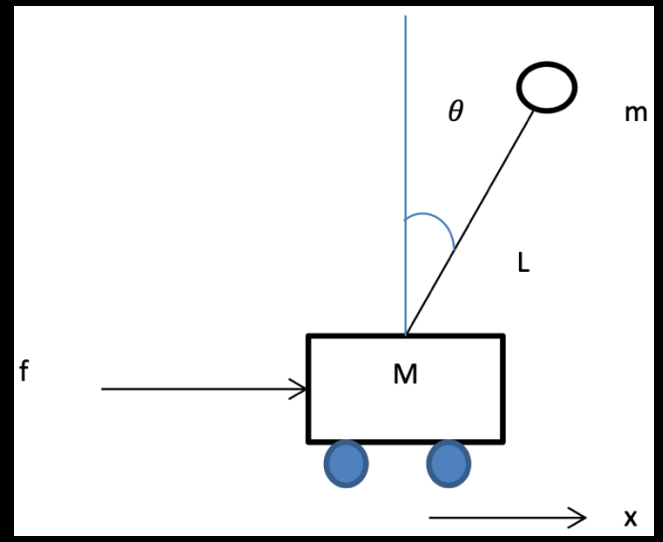
Input: Torque at base

Output: Angle θ with $\theta = 0$ meaning that the pendulum is at the top

Cart-Pole System – Model

Dynamics:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dfrac{f + m\sin\theta(L\dot{\theta}^2 - g\cos\theta)}{M + m\sin^2\theta} \\ \dot{\theta} \\ \dfrac{-f\cos\theta - mL\dot{\theta}^2\sin\theta\cos\theta + (M+m)g\sin\theta}{L(M + m\sin^2\theta)} \end{bmatrix}$$



Input: Force in x-direction acting on cart

Output: Angle θ with $\theta = 0$ meaning that the pendulum is at the top as well as the x-position of the cart

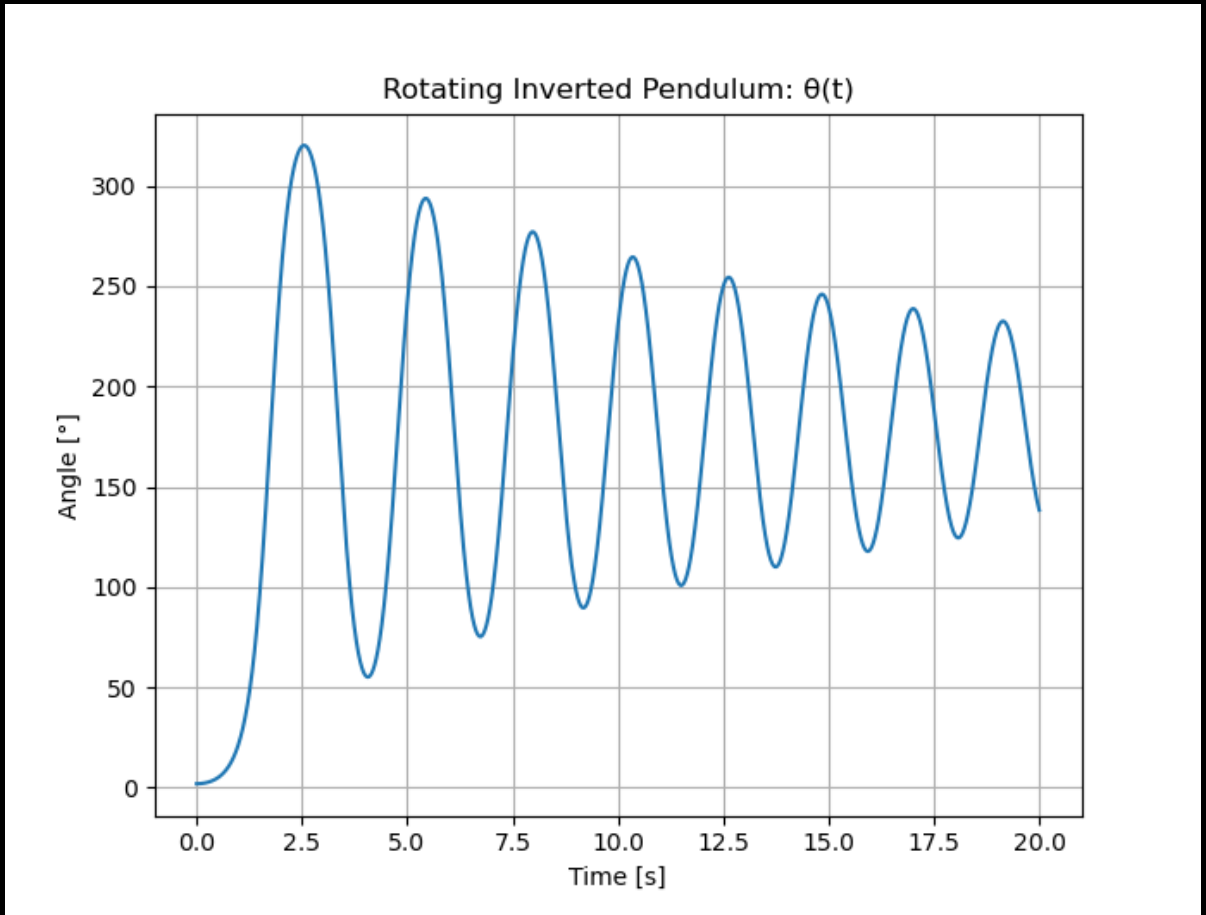
Why do we need (feedback) control?

- Let‘s simulate the systems with a non-zero initial condition and let‘s see what happens.

- How to simulate?
  - Discretize (simplest version: Euler Forward) and assume zero-order hold for input (constant input over sample interval)

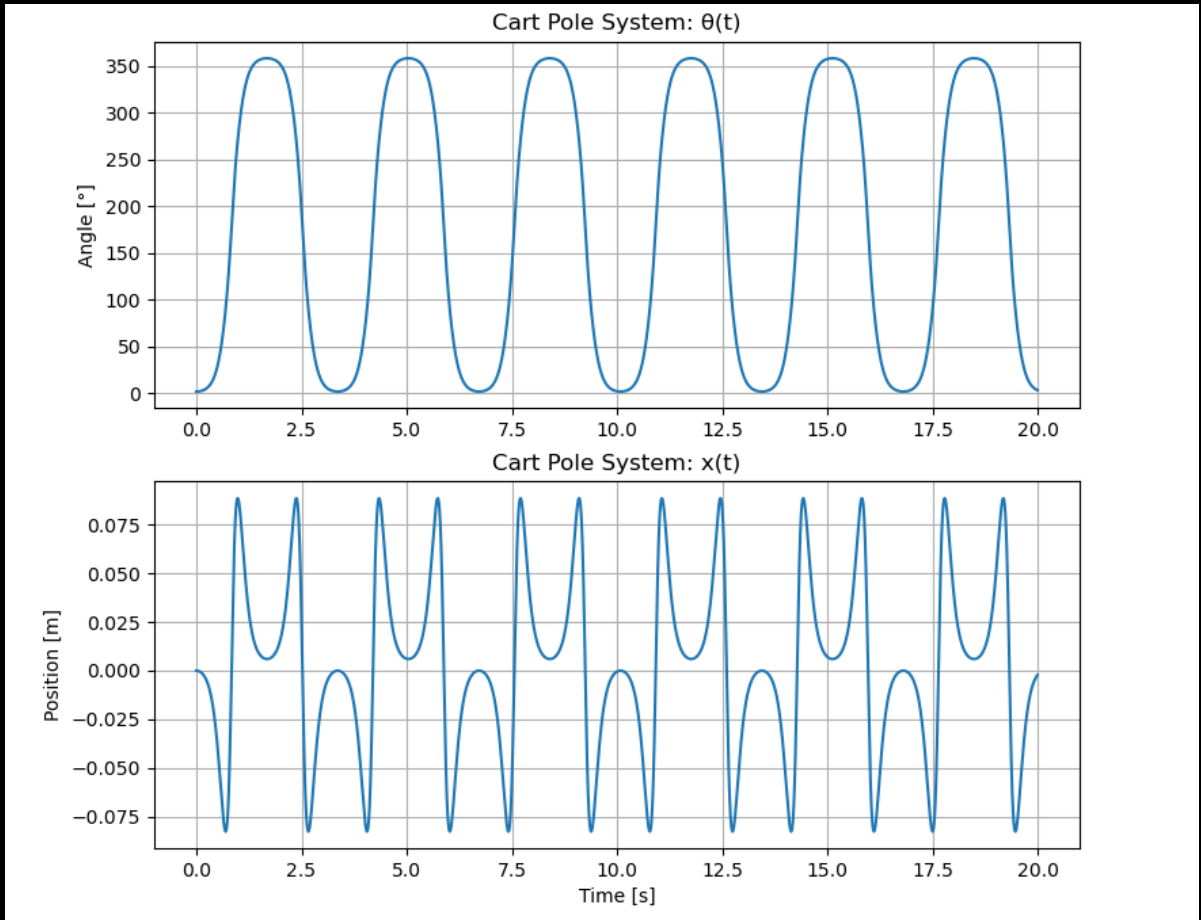
$$\frac{dx}{dt} \approx \frac{x[k+1] - x[k]}{\Delta t} = f(x,u) \rightarrow x[k+1] = x[k] + \Delta t \cdot f(x[k], u[k])$$

Why do we need (feedback) control? $\theta(0) = 2°$



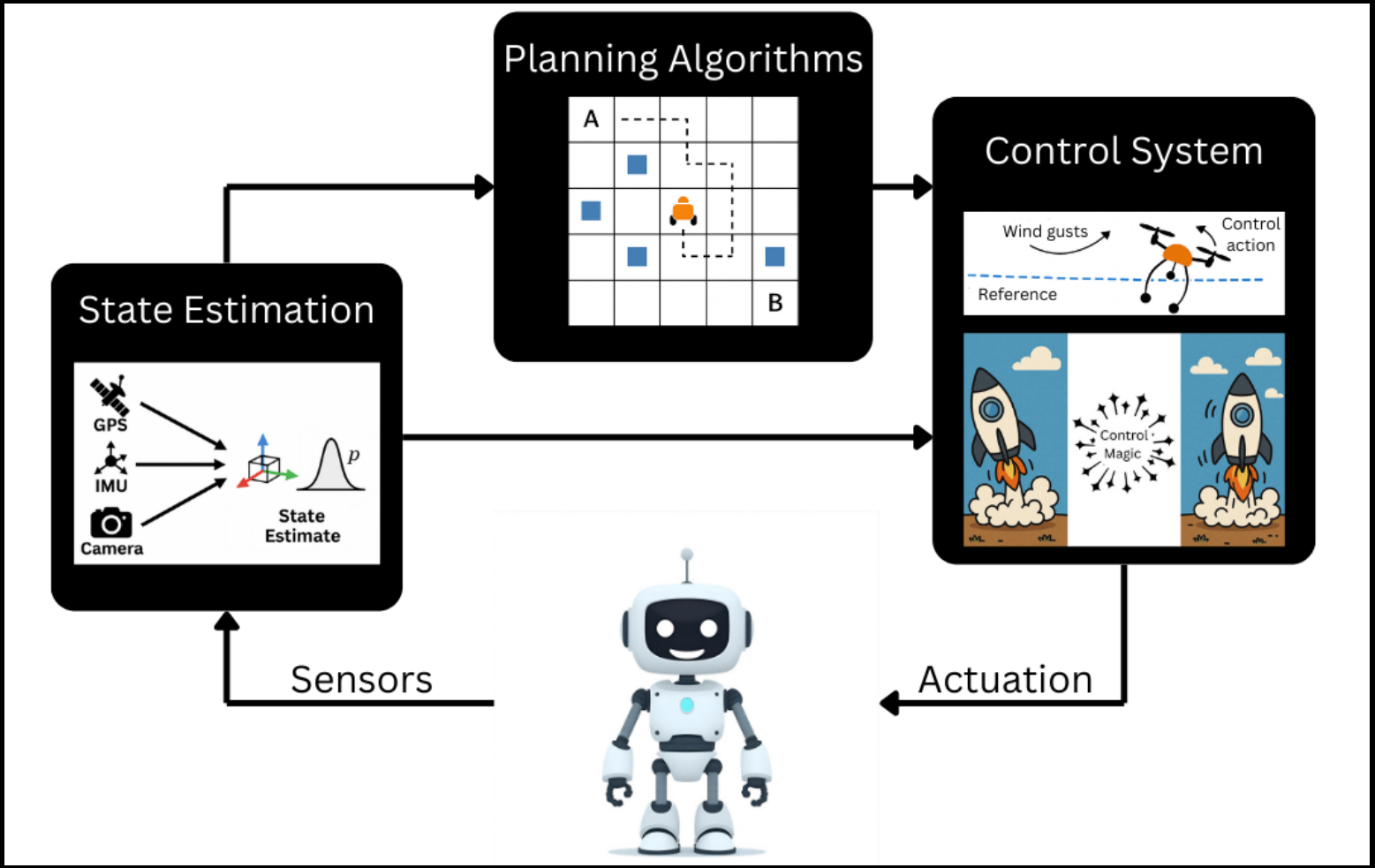
Even though it looks like it stabilizes, it actually tips over and then swings around the lower stable equilibrium point at $\theta = 180°$

Why do we need (feedback) control? $\theta(0) = 2°$



Since we didn't include a friction term in the model for the cart-pole system a small initial deviation from the upper equilibrium point leads to a limit-cycle behavior.

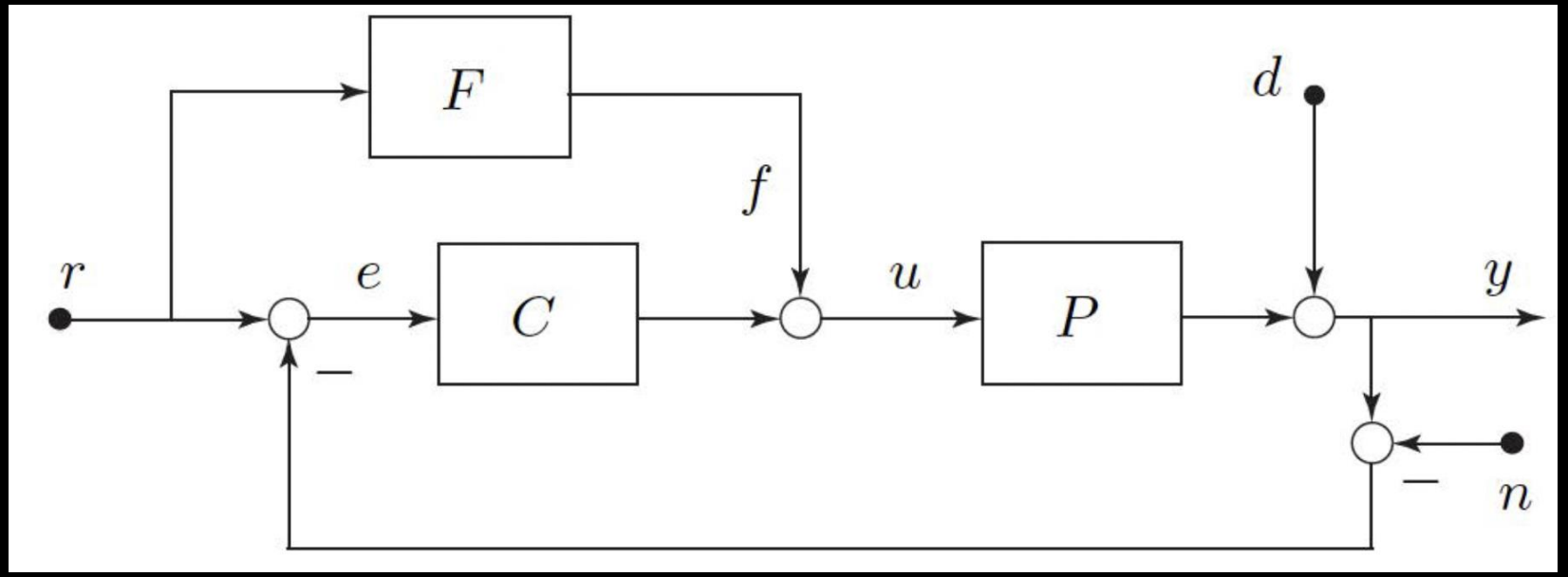
What Is Control?



To control a system we need to make sensor information useful (state estimation)and we need a plan of what we want to control (planning algorithms).

But what is control? Calculation of actuation signals to follow a given plan based on sensor signals

What Is Control?



r: Reference

e: Reference tracking error

C: Controller

F: Feedforward controller

f: Feedforward signal

u: Control input

P: Plant (Model)

d: disturbance

y: output / measurement

n: noise

PID Control

- Except for hardcoding control signals, this is probably one of the most simplest ways to control a system
- 90% of industry applications use PID controllers
- Where you will use them (if at all) in your projects: motor driver control, i.e., to control the current to reach the desired torques

What is a PID Controller?

P: proportional

I: integral

D: derivative

- These terms refer to the reference tracking error

$$u \approx K_p e + K_i \int e\, dt + K_d \frac{de}{dt}$$

What is a PID Controller?

K_P results in a more aggressive response to current error, which can:

- Increase the speed of response and reduce rise time.
- Improve the steady-state error.
- Increase the risk of overshooting the target and causing oscillations.

K_I accumulates past errors, which can:

- Eliminate steady-state error more quickly.
- Increase overshoot and settling time.

K_D acts as a predictive measure based on the rate of error change, which can:

- Decrease overshoot and settling time by damping the response.
- Slow down the transient response.
- Amplify high-frequency noise in the signal

PID in Practice – Add Feedforward Terms

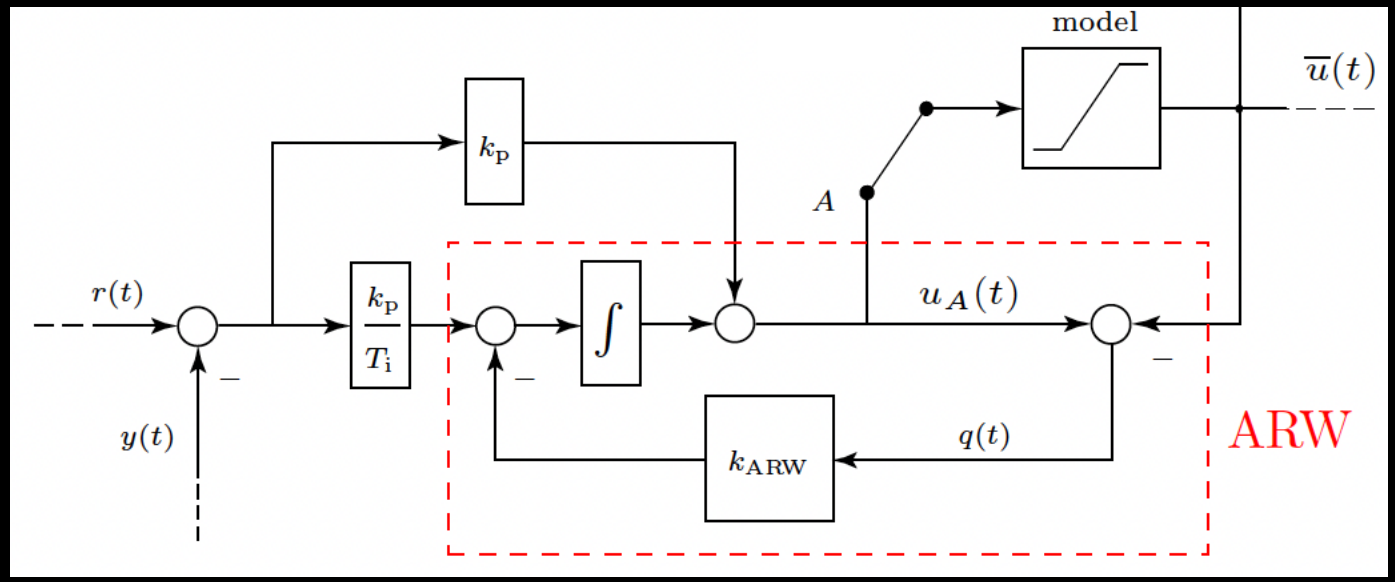
- Augment with things you can estimate → for the rotating inverted pendulum this means that we can compensate for the gravity and the friction (our model should be reasonable close to the real system)

$$u = u_{PID} \pm u_{gravity} \pm u_{friction}$$

  → choose the signs of the single terms opposite of how they are used in the model

- Due to the integral term of the PID controller, we might run into controller saturation

PID in Practice – Anti-Reset Windup

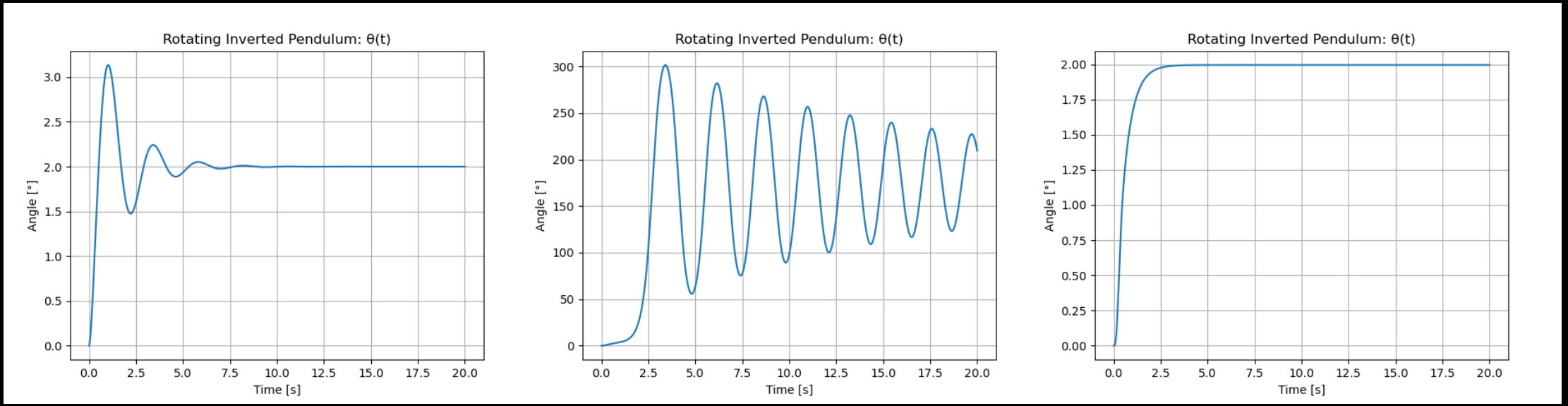


If saturation occurs, the integral part is reduced via a basic feedback signal with tuning parameter k_{ARW}

PID Control – Task 1

- Your boss needs a demo for inverted pendulum control at the end of the day to impress some investors.
- Implement a PID controller on the nonlinear inverted pendulum system (add gravity and friction compensation)
- If you are fast, you can also implement the ARW
- Play around with the tuning parameters and get a feeling.

PID Control – Task: Equal gains, effect of ARW



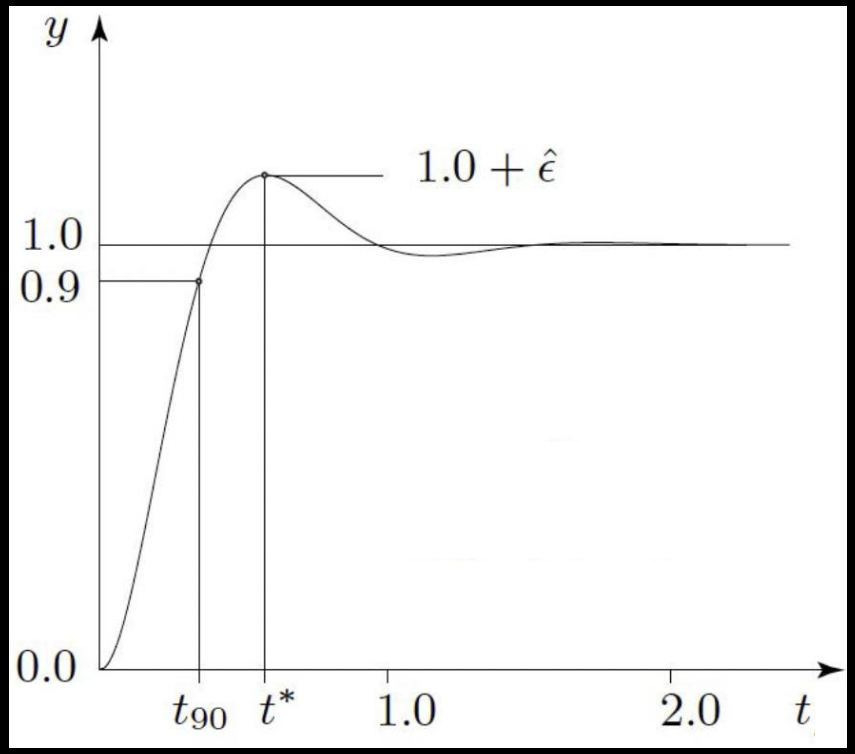
Without saturation, we converge to the reference but the input is at 0.8 Nm

Our motor can however only generate 0.5 Nm torque, let's simply saturate → system unstable (either tune again or use ARW)

ARW takes care of keeping the integrator empty enough to avoid saturation issues but also large enough to get rid of steady-state errors

PID Control – Specs

- Now your boss comes and asks for specs of the controller on the system. Based on your step response you can identify:



Well, it takes xy seconds to achieve 90% of our desired reference and we overshoot by only $\hat{\epsilon}$

→ Not really convincing, right?

Linear System Theory for SISO Systems

- Since we anyways focus at the moment to stabilize the system around a given operating point → linearize the system around this point and use linear system theory
  - Transfer functions (capturing I/O behavior)
  - Stability Analysis
  - Visualizations
  - Robustness Metrics
  - PID Design Methods

Linearization

0 if $\bar{x}$ equilibrium $\rightarrow \dot{x} = f(x)$

$$f(x) \approx \cancel{f(\bar{x})} + \left.\frac{\partial f}{\partial x^\top}\right|_{x=\bar{x}} (x - \bar{x}), \text{ with } \frac{\partial f}{\partial x^\top} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\delta x(t) = \frac{\partial f_0}{\partial x}|_{x=x_\mathrm{e},u=u_\mathrm{e}} \cdot \delta x(t) + \frac{\partial f_0}{\partial u}|_{x=x_\mathrm{e},u=u_\mathrm{e}} \cdot \delta u(t)$$

$$\delta y(t) = \frac{\partial g_0}{\partial x}|_{x=x_\mathrm{e},u=u_\mathrm{e}} \cdot \delta x(t) + \frac{\partial g_0}{\partial u}|_{x=x_\mathrm{e},u=u_\mathrm{e}} \cdot \delta u(t)$$

$\begin{bmatrix} \frac{\partial f_{0,1}}{\partial x_1}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} & \cdots & \frac{\partial f_{0,1}}{\partial x_n}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{0,n}}{\partial x_1}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} & \cdots & \frac{\partial f_{0,n}}{\partial x_n}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} \end{bmatrix} \in \mathbb{R}^{n\times n}$	$\begin{bmatrix} \frac{\partial f_{0,1}}{\partial u}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} \\ \vdots \\ \frac{\partial f_{0,n}}{\partial u}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} \end{bmatrix} \in \mathbb{R}^{n\times 1}$
$\begin{bmatrix} \frac{\partial g_0}{\partial x_1}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} & \cdots & \frac{\partial g_0}{\partial x_n}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} \end{bmatrix} \in \mathbb{R}^{1\times n}$	$\begin{bmatrix} \frac{\partial g_0}{\partial u}\Big\|_{x=x_\mathrm{e},u=u_\mathrm{e}} \end{bmatrix} \in \mathbb{R}^{1\times 1}$

Time-Domain to Frequency-Domain

- General solution for $\dot{x} = Ax + bu$ is difficult to solve, especially if we have feedback and u is a function of x.

$$x(t) = e^{A \cdot t} \cdot x(0) + \int_0^t e^{A \cdot (t-\rho)} \cdot b \cdot u(\rho)\, d\rho$$

- Way out: Laplace Transformation

Time-Domain to Frequency-Domain

linearity	:	$\mathcal{L}\{a \cdot x_1(t) + b \cdot x_2(t)\} = a \cdot X_1(s) + b \cdot X_2(s)$
similarity	:	$\mathcal{L}\left\{\frac{1}{a} \cdot x\left(\frac{t}{a}\right)\right\} = X(s \cdot a)$
shift	:	$\mathcal{L}\{x(t-T)\} = \mathrm{e}^{-T \cdot s} \cdot X(s)$
damping	:	$\mathcal{L}\left\{x(t) \cdot \mathrm{e}^{a \cdot t}\right\} = X(s-a)$
derivative t	:	$\mathcal{L}\left\{\frac{\mathrm{d}}{\mathrm{d}t}x(t)\right\} = s \cdot X(s) - x(0)$
derivative s	:	$\mathcal{L}\{t \cdot x(t)\} = -\frac{\mathrm{d}}{\mathrm{d}s}X(s)$
integration t	:	$\mathcal{L}\left\{\int_0^t x(\tau)\,\mathrm{d}\tau\right\} = \frac{1}{s} \cdot X(s)$
integration s	:	$\mathcal{L}\left\{\frac{1}{t} \cdot x(t)\right\} = \int_s^\infty X(\sigma)\,\mathrm{d}\sigma$
convolution t	:	$\mathcal{L}\{x_1(t) * x_2(t)\} = X_1(s) \cdot X_2(s)$
convolution s	:	$\mathcal{L}\{x_1(t) \cdot x_2(t)\} = X_1(s) * X_2(s)$
initial value	:	$\lim_{t \to 0_+} x(t) = \lim_{s \to \infty} s \cdot X(s)$
final value	:	$\lim_{t \to \infty} x(t) = \lim_{s \to 0_+} s \cdot X(s)$

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = A \cdot x(t) + b \cdot u(t), \qquad x(0) = x_0$$

$$y(t) = c \cdot x(t)$$

$$s \cdot X(s) - x_0 = A \cdot X(s) + b \cdot U(s)$$

$$X(s) = (s\,I - A)^{-1} \cdot b \cdot U(s) + (s\,I - A)^{-1} \cdot x_0$$

$$Y(s) = c \cdot (s\,I - A)^{-1} \cdot b \cdot U(s) + c \cdot (s\,I - A)^{-1} \cdot x_0$$

P(s) captures the input/output behavior of our model!

often $x_0 = 0$ can be assumed

Stability Analysis & Co.

- To analyse the stability of the equilibrium point around which the model was linearized: look at the poles of the system (transfer function is in the SISO case a fraction and poles are the zeros of the denominator) which is the same as the eigenvalues of A.

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = A \cdot x(t), \quad x(0) = x_0, \quad 0 < ||x_0|| < \infty \longrightarrow x(t) = \mathrm{e}^{A \cdot t} \cdot x(0)$$

if A is diagonalizable, there is a coordinate transformation $x = V\tilde{x}$ such that $\tilde{A} = V^{-1}AV$ is diagonal and $\tilde{x}(t) = V^{-1} \cdot \mathrm{e}^{A \cdot t} \cdot V \cdot \tilde{x}(0) = \mathrm{e}^{\tilde{A} \cdot t} \cdot \tilde{x}(0)$

Stability Analysis & Co.

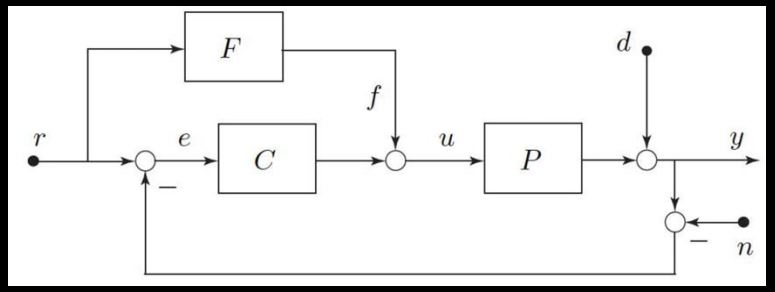
- Then it follows that ($\lambda_i = \sigma_i + j \cdot \omega_i$)

$$\tilde{x}(t) = \begin{bmatrix} e^{\lambda_1 \cdot t} \cdot \tilde{x}_1(0) \\ e^{\lambda_2 \cdot t} \cdot \tilde{x}_2(0) \\ \cdots \\ e^{\lambda_n \cdot t} \cdot \tilde{x}_n(0) \end{bmatrix} = \begin{bmatrix} e^{\sigma_1 \cdot t} \cdot \left(\cos(\omega_1 \cdot t) + j \cdot \sin(\omega_1 \cdot t)\right) \cdot \tilde{x}_1(0) \\ e^{\sigma_2 \cdot t} \cdot \left(\cos(\omega_2 \cdot t) + j \cdot \sin(\omega_2 \cdot t)\right) \cdot \tilde{x}_2(0) \\ \cdots \\ e^{\sigma_n \cdot t} \cdot \left(\cos(\omega_n \cdot t) + j \cdot \sin(\omega_n \cdot t)\right) \cdot \tilde{x}_n(0) \end{bmatrix}$$

$\sigma_i < 0$	Asymp. stable
$\sigma_i = 0$	stable
$\sigma_i > 0$	unstable
$j \cdot \omega_i > 0$	oscillations

- Controllability: unstable modes must be controllable, else add an actuator
- Observability: unstable modes must be observable, else add a sensor

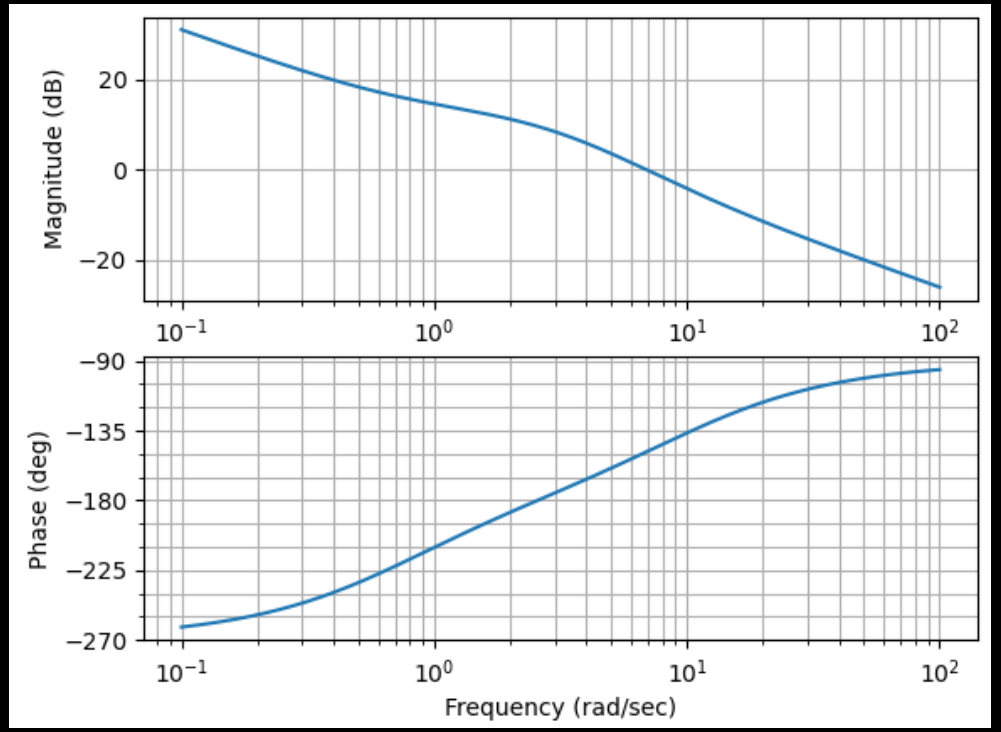
Other Transfer Functions



- L is called open loop (r → y) and $L(s) = \frac{Y(s)}{R(s)} = P(s)C(s)$
- T is called closed loop (r → y) and (n → y) and $\mathrm{T}(s) = \frac{Y(s)}{R(s)} = \frac{L(s)}{1+L(s)}$
- S is called sensitivity of the closed loop (d → y) and $\mathrm{S}(s) = \frac{Y(s)}{R(s)} = \frac{1}{1+L(s)}$
- Look at frequency responses → $s = j \cdot \omega$ and disturbances occur usually at low frequencies, we want to track around low and mid frequencies and noise occurs at high frequencies

Loop Shaping

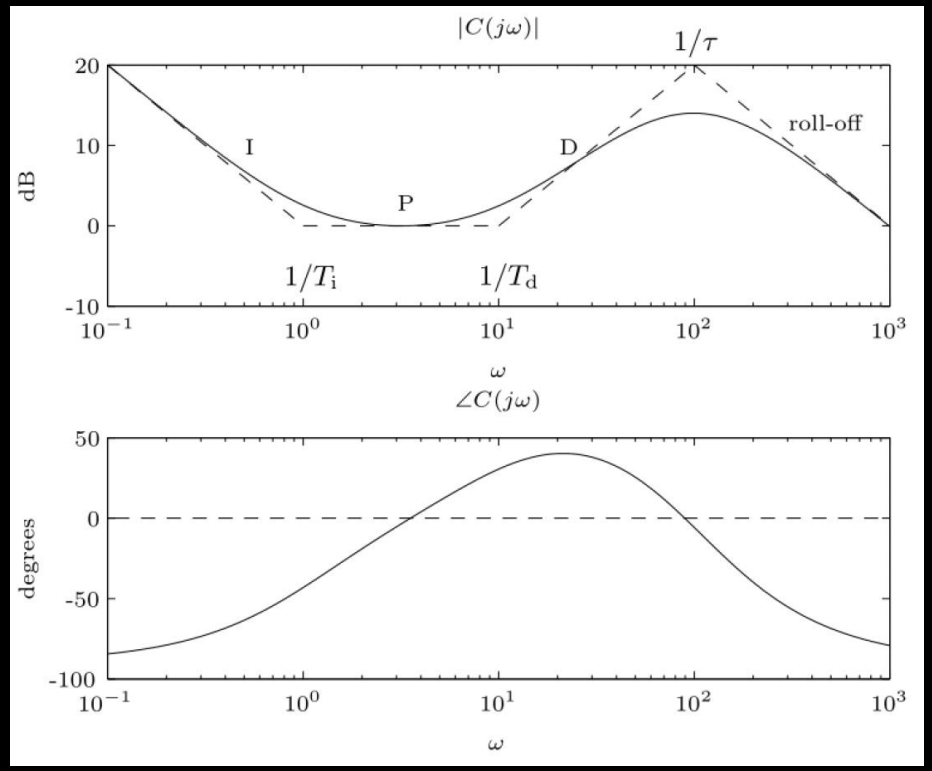
- We want that L is high at low and moderate frequencies and that it is low at high frequencies.



Bode Plot:
Look at the harmonic response (amplitude and phase) of the system for different frequencies. This is possible as a property of linear systems is that for a harmonic input, a harmonic output will result with the same frequency.

PID Controller in Frequency Domain

- If you know how the Bode Plot of your plant and your desired open loop transfer function looks like, design the controller accordingly.

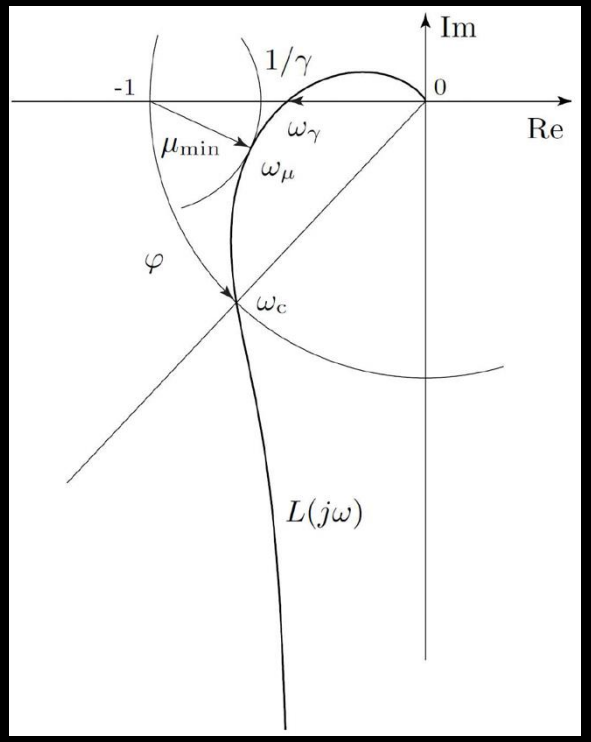


$$C(s) = k_p \cdot \left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s\right) \cdot \underbrace{\frac{1}{(s \cdot \tau + 1)^2}}$$

roll-off term to attenuate high-frequency noise

Robustness Metrics

- From S and T, we note that L = -1 is a dangerous point, as S and T will be infinite.



Nyquist Plot:
Plot the transfer function as a function of the frequency in the complex plane (magnitude = distance to origin, phase = angle between transfer function and real axis)

ω_c: crossover-frequency	Where the transfer function crosses the 0 dB line for the first time
φ: phase margin	Angle at ω_c between L and -1 (allows for uncertainty in delays)
γ: gain margin	Inverse of magnitude of L where the phase is -180° (allows for model uncertainties, compensate actuator drift)
μ_{min}: minimum return difference	Minimum distance between -1 and L

PID Control – Task 2

- Implement a PID controller in the frequency domain for the linearized system
- *Use:* ctrl.ss, s = ctrl.tf('s'), ctrl.ss2tf, ctrl.pole, ctrl.margin, ctrl.bode_plot, ctrl.feedback, ctrl.step_response

- Study different transfer functions and their step responses and give your boss some better specs such that he can impress those investors

PID Control – Tuning Methods & Limitations

- Tuning Methods:
  - Ziegler Nichols for first-order systems
  - Åström – Hägglund: specify μ_{min} and get directly PID gains

But:

- Tuning is cumbersome
- Constraints? Performance? Flexibility?
- Analysis Design tools work mainly for SISO systems

→Your boss thinks the pendulum is too simple, so let‘s switch to the cart-pole system

Multiple Input and Output Systems x PID

- For general systems, $P(s) = C \cdot (sI - A)^{-1} \cdot B$ is a matrix where $P_{i,j}$ describes the effect of input j to output i.
- For the cart-pole system we have one input and two outputs, therefore, P is a 2x1 matrix.
- This means C is a 1x2 matrix as Y = PCE where Y and E are row vectors of length 2.
- Try to find gains for two separate PID controllers to stabilize the system (only low gains are valid, of course with infinite control input everything is stabilizable)

LQR Control

L: Linear (linear system $\dot{x} = Ax + Bu$)

Q: Quadratic (quadratic cost $J = x^T Qx + u^T Ru$)

R: Regulator (regulates the state back to zero, i.e., stabilization)

We want to minimize:

$$J(x(t), u(t)) = \int_{t=0}^{\infty} x^T(t) \cdot Q \cdot x(t) + u(t)^T \cdot R \cdot u(t)\, dt$$

$$Q = Q^T \geq 0, \quad \text{and} \quad R = R^T > 0$$

LQR Control – Control Law

- Through recursion it can be shown that a state-feedback law is minimizing this cost

$$u^\star(k) = -(B^\top P_\infty B + R)^{-1} B^\top P_\infty A x(k) := F_\infty x(k)$$

- P_∞ is the solution of the algebraic Riccati equation (also can be shown by recursively solving the problem)

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

How to Implement an LQR Controller?

- Define your linear system matrices
- Define your cost matrices
- Plug into discrete algebraic Riccati equation → gives you matrix P from which you can find the linear control law K
- Simulate $\dot{x} = (A + BK)x$

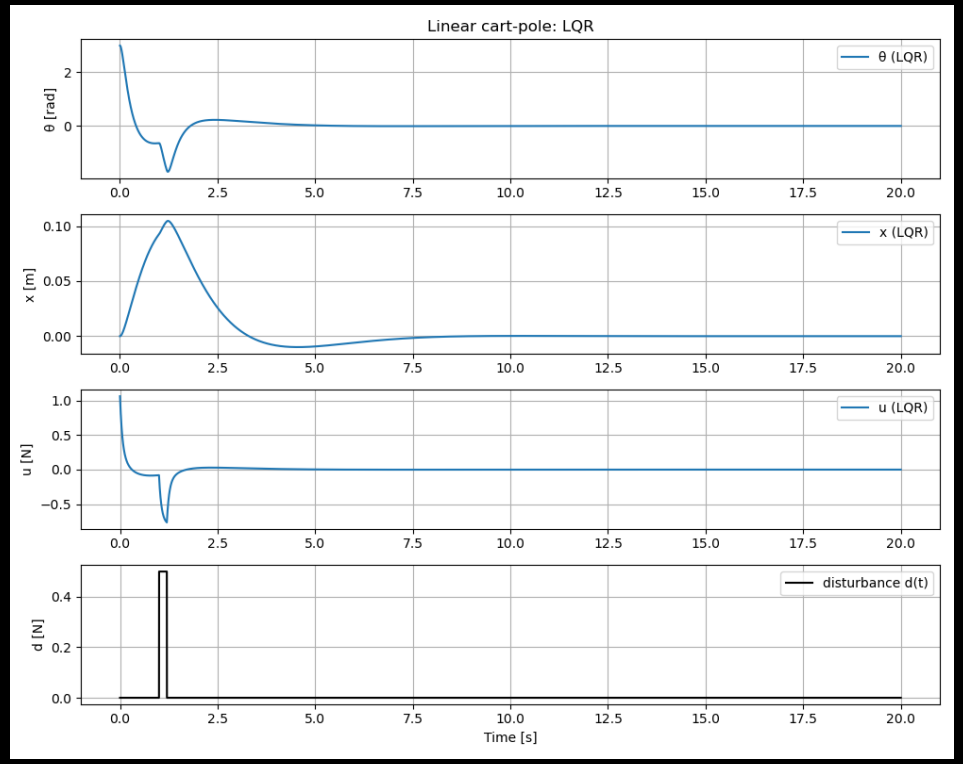
LQR Control – Task 3

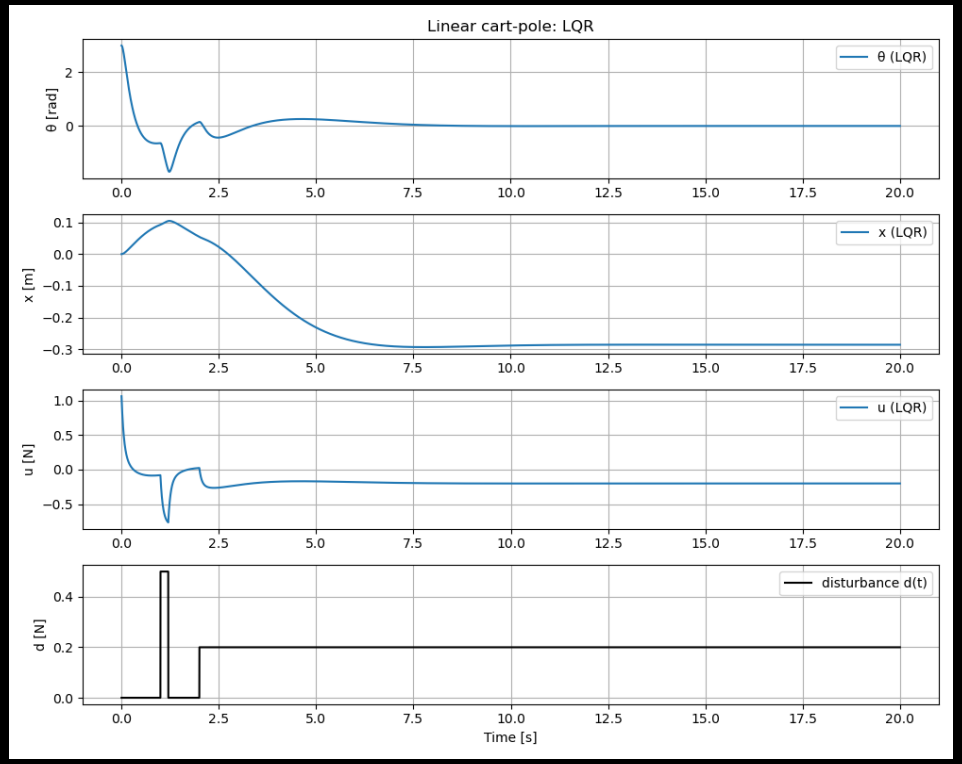
- Implement the LQR controller and show your boss that even the more complex model can be controlled.

Use: solve_discrete_are, np.linalg.inv, np.diag

- Play around with Q and R to see how the resulting control behavior changes.

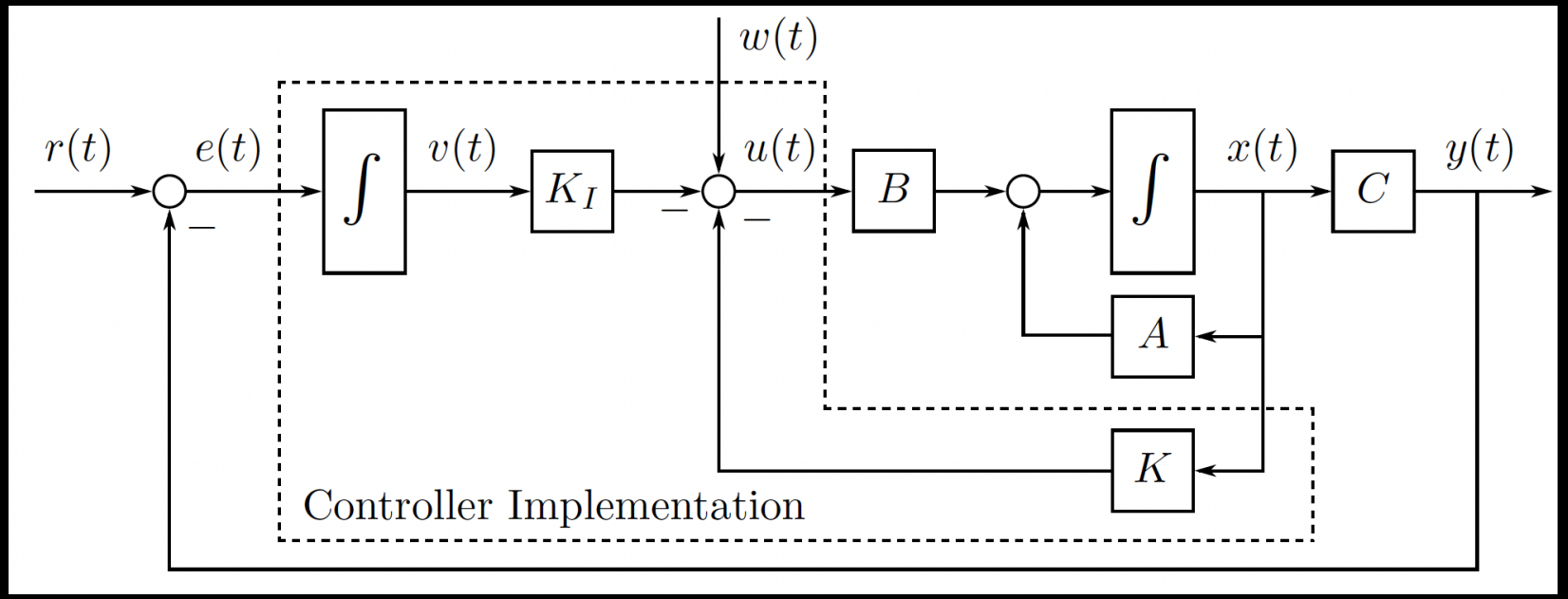
LQR Control – Persisting Disturbance





We observe that we have a steady-state error in the x-position if we have a persisting disturbance (annoying wind where we present the system to the investors). But from the PID theory we know that we can get rid of steady-state errors by introducing an integrator → LQRI

LQRI



LQRI

$$\widetilde{x}(t) = \begin{bmatrix} x(t) \\ v(t) \end{bmatrix}$$

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\widetilde{x}(t) &= \begin{bmatrix} A \cdot x(t) + B \cdot (u(t) + w(t)) \\ r(t) - y(t) \end{bmatrix} \\ &= \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \cdot \begin{bmatrix} x(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \cdot (u(t) + w(t)) + \begin{bmatrix} 0 \\ I \end{bmatrix} \cdot r(t) \end{aligned}$$

$$\widetilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & Q_I \end{bmatrix},$$

$$Q_I = \begin{bmatrix} \gamma_1 & 0 & \dots & 0 \\ 0 & \gamma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \gamma_m \end{bmatrix}$$

Same procedure as before but with update A, B, Q matrices and then you can extract the two controllers

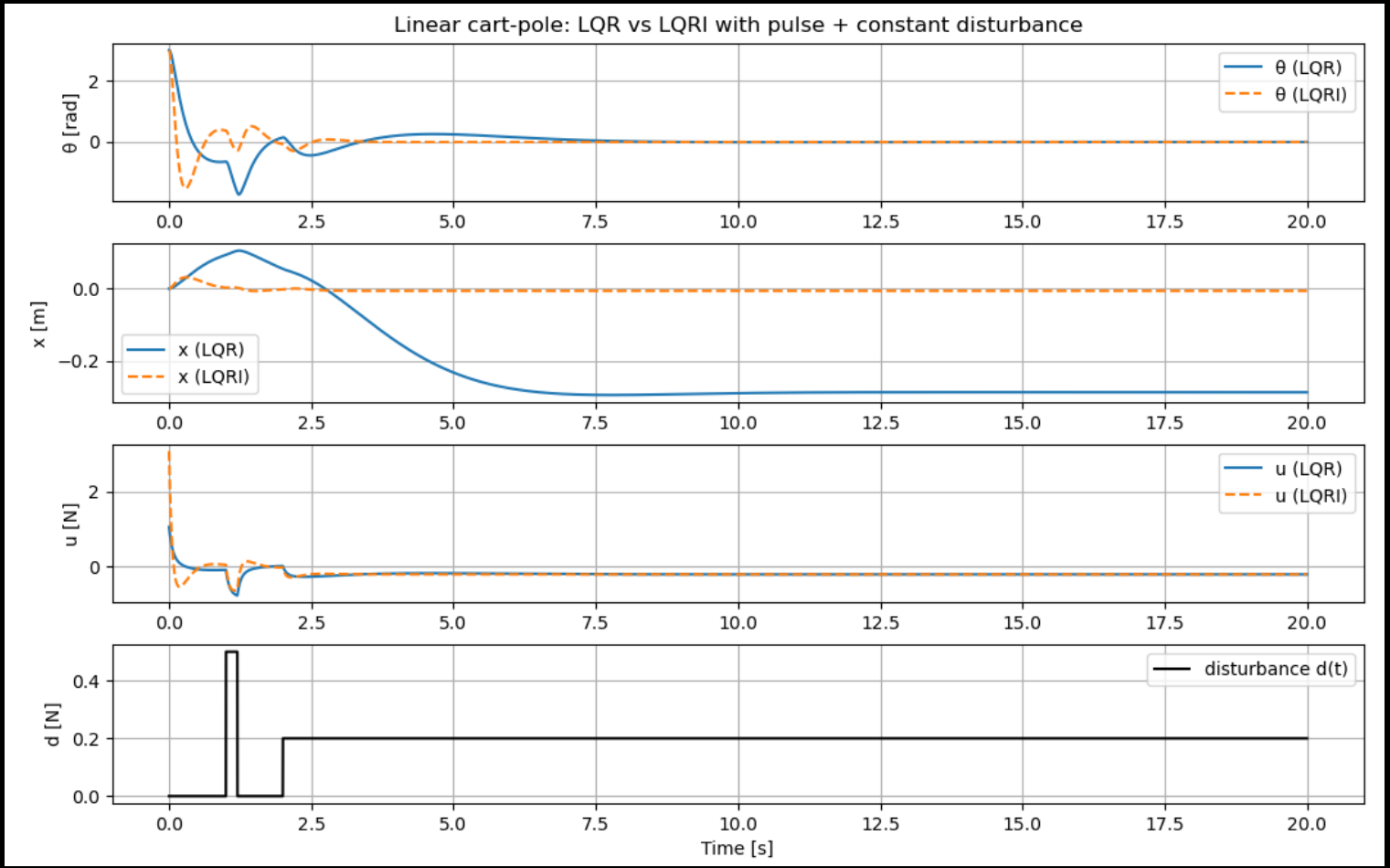
$$\widetilde{K} = \begin{bmatrix} K, & K_I \end{bmatrix}$$

LQRI Control – Task 4

- Augment your existing LQR controller with integrating behavior to get rid of the steady state error such that you can impress the investors even if there‘s an annoying wind disturbing your system the whole time

Use: solve_discrete_are, np.linalg.inv, np.diag, np.block

LQRI Control – Task 4



Augmenting the existing LQR controller with an integrator helps us remove the steady-state error!

LQR(I) Control – Add-Ons & Limitations

- It‘s also possible to track simple references with the LQR(I) controller (identify steady state and input, shift your dynamics and regulate the shifted system to the origin)

$$\begin{aligned}\Delta x(t) &= x(t) - x_\infty \\ \Delta u(t) &= u(t) - u_\infty\end{aligned}$$

$$u(t) = -K \cdot (x(t) - x_\infty) + u_\infty = -K \cdot x(t) + K \cdot x_\infty + u_\infty$$

But:

- Cannot track arbitrarily complex references
- What if we are looking at finite horizons?
- Your boss wants to include constraints as the investors want to deploy the system with cheap actuators and in a confined space.

How to Include constraints?

We need to solve an optimization problem where we find the optimal input by minimizing an objective subject to state and input constraints

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{s.t.} & g_i(x) \leq 0, \quad i = 1, \ldots, n_g \\ & h_i(x) = 0, \quad i = 1, \ldots, n_h \end{array}$$

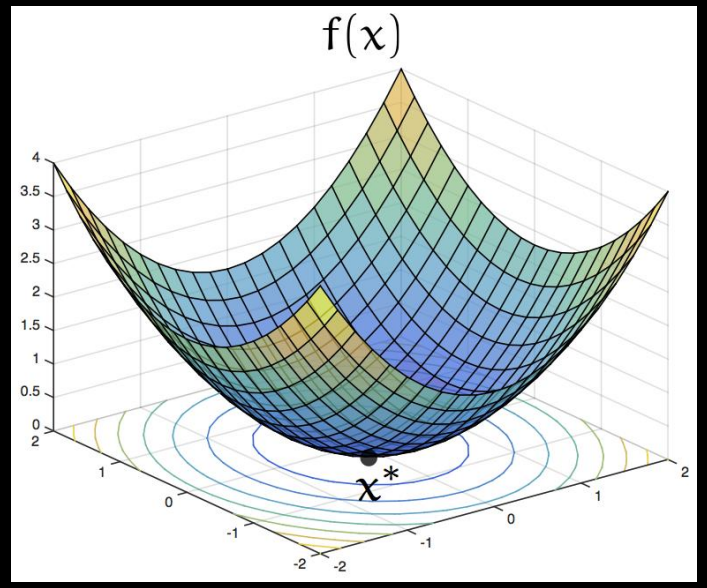
x: optimization variables	In the context of control, these are usually state and input variables
f(x): objective function	If we want to track a reference, the objective would be to minimize the reference tracking error
h(x): equality constraints	Usually the dynamics
g(x): inequality constraints	State and input constraints

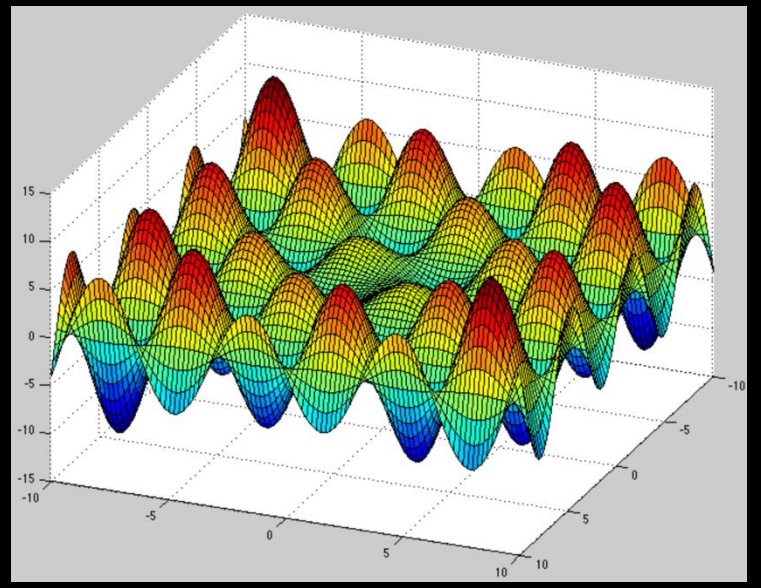
Optimization Fundamentals

Two classes are distinguished – Convex and Non-convex problems

Convex: an optimization problem is convex if f and g are convex and h is affine

Non-convex: else





Optimization Fundamentals – Convex

- No matter how you initialize the optimization, the optimizer will lead to a global optimum of the objective function

Types

- Linear Program: $\text{minimize } c^\mathsf{T} x \quad \text{s.t. } Ax - b \geqslant 0, x \geqslant 0$
- Quadratic Program (P positive semi-definite): $\underset{x \in \mathbb{R}^n}{\text{minimize}} \ \frac{1}{2} x^\mathsf{T} P x + q^\mathsf{T} x \quad \text{s.t. } Gx \leqslant h, Ax = b$
- Second-Order Cone Program:

$$\begin{aligned} \text{minimize} \ & f^\mathsf{T} x \\ \text{s.t.} \ & |A_i x + b| \leqslant c_i^\mathsf{T} x + d_i, \quad i = 1, \ldots, m \\ & Fx = g. \end{aligned}$$

- Semidefinite Program (F_i are symmetric):

$$\begin{aligned} \text{min.} \ & c^\mathsf{T} x \\ \text{s.t.} \ & x_1 F_1 + x_2 F_2 + \ldots + x_n F_n + G \preceq 0 \\ & Ax = b, \end{aligned}$$

Optimization Fundamentals – Convex

Linear Program: time-optimal control, support function calculations

Quadratic Program: LQR with constraints, tracking problems

Second-Order Cone Program: many convex relaxations lead to SOC constraints $\sqrt{x_1 \cdot x_2} \geq x_3, \quad x_1 \cdot x_2 \geq 0,$ $\left\| \begin{matrix} 2 \cdot x_3 \\ x_1 - x_2 \end{matrix} \right\|_2 \leq x_1 + x_2$

Semidefinite Program: LQR and H_∞ controller design via LMIs, terminal set calculations for MPC

→ These problems can be solved using CVXPY

Optimization Fundamentals – Non-Convex

There are two main problem types – those that are inherently non-convex due to nonlinear dynamics, costs, constraints and those that are mixed-integer problems where the integer relaxation is still convex.

→ Initial guess matters!

$$
\begin{aligned}
&\text{minimize } f(x) \\
&\text{subject to } g_i(x) \leq 0 \text{ for each } i \in \{1, \ldots, m\} \\
&\qquad\qquad h_j(x) = 0 \text{ for each } j \in \{1, \ldots, p\} \\
&\qquad\qquad x \in X.
\end{aligned}
$$

Solve with CasADi and use the IPOPT solver
No global optimality guarantees
Usually larger computational times

$$
\begin{aligned}
\min_x f(x) \quad \text{s.t.} \quad & L_i \leq x_i \leq U_i, \quad i \in \{1, \ldots, n\} \\
& g_i(x) \leq 0, \quad i \in I \\
& h_j(x) = 0, \quad j \in J \\
& x_i \in \mathbb{Z}, \quad i \in \mathcal{I}.
\end{aligned}
$$

If the integer relaxation is convex, can use CVXPY with MOSEK or Gurobi
Close to globally optimal
If it‘s a MINLP then use CasADi and Bonmin

Optimization Solver Survival Guide

- Find a tailored solver for your optimization problem
- Even if the solver documentations are cumbersome to read, understand the solver options as they can improve performance drastically
- Normalize your optimization problem (improves numerical stability)
- Remove redundant constraints
- If your problem is infeasible, remove constraints and add them one by one to identify the issue. Then double check your implementation.
- Warmstart (especially for NLPs) and / or do multiple initializations
- Try different solvers
- Regularize the objective function $\min f(x) + \lambda^T x$

Optimization Fundamentals – Task 5

Implement the following two optimization problems

QP

$$\begin{aligned} \min_{x} \quad & \frac{1}{2} x^\top P x + q^\top x \\ \text{s.t.} \quad & x \geq 0, \\ & x_1 + x_2 = 1 \end{aligned}$$

Use: cp.Variable, cp.Minimize, cp.Problem

NLP

$$\begin{aligned} \min_{x,y} \quad & (x-1)^2 + (y-2.5)^2 \\ \text{s.t.} \quad & x^2 y \geq 1, \\ & x + y^2 = 4 \end{aligned}$$

Use: ca.MX.sym, define constraints as g1 >= 0 and g2 == 0 where g is a vector and you have a lower bound and upper bound vector lbg and ubg, such bounds are also needed for the variables

What we want to solve vs. what we can solve

$$
\begin{aligned}
J^{\star}_{\infty}(x(0)) = \min_{u(\cdot)} \; & \sum_{i=0}^{\infty} l(x_i, u_i) \\
\text{subj. to } & x_{i+1} = Ax_i + Bu_i, \;\; i = 0, \ldots, \infty \\
& x_i \in \mathcal{X}, u_i \in \mathcal{U}, i = 0, \ldots, \infty \\
& x_0 = x(0)
\end{aligned}
$$

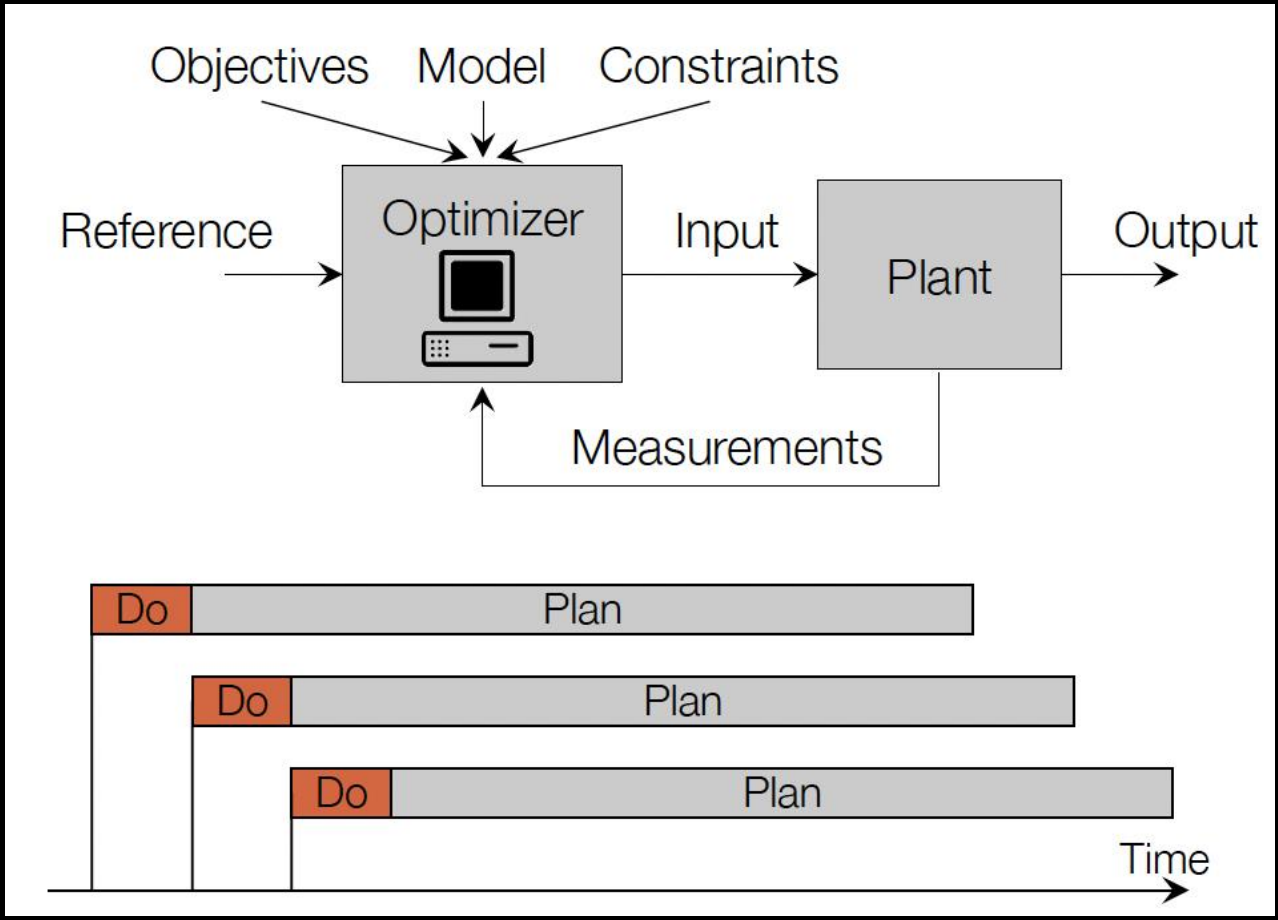
Optimizing over a trajectory provides a tradeoff between short- and long-term benefits of actions

But: can‘t compute the control law as we have infinite optimization variables

$$
\begin{aligned}
J^{\star}_{k \to k+N|k}(x(k)) = \min_{U_{k \to k+N|k}} \; & l_f(x_{k+N|k}) + \sum_{i=0}^{N-1} l(x_{k+i|k}, u_{k+i|k}) \\
\text{subj. to } & x_{k+i+1|k} = Ax_{k+i|k} + Bu_{k+i|k}, \;\; i = 0, \ldots, N-1 \\
& x_{k+i|k} \in \mathcal{X}, u_{k+i|k} \in \mathcal{U}, i = 0, \ldots, N-1 \\
& x_{k+N|k} \in \mathcal{X}_f \\
& x_{k|k} = x(k)
\end{aligned}
$$

Solve the truncated problem and choose a terminal cost to approximate the “tail“ of the cost and a terminal constraint set to approximate the “tail“ of the constraints.

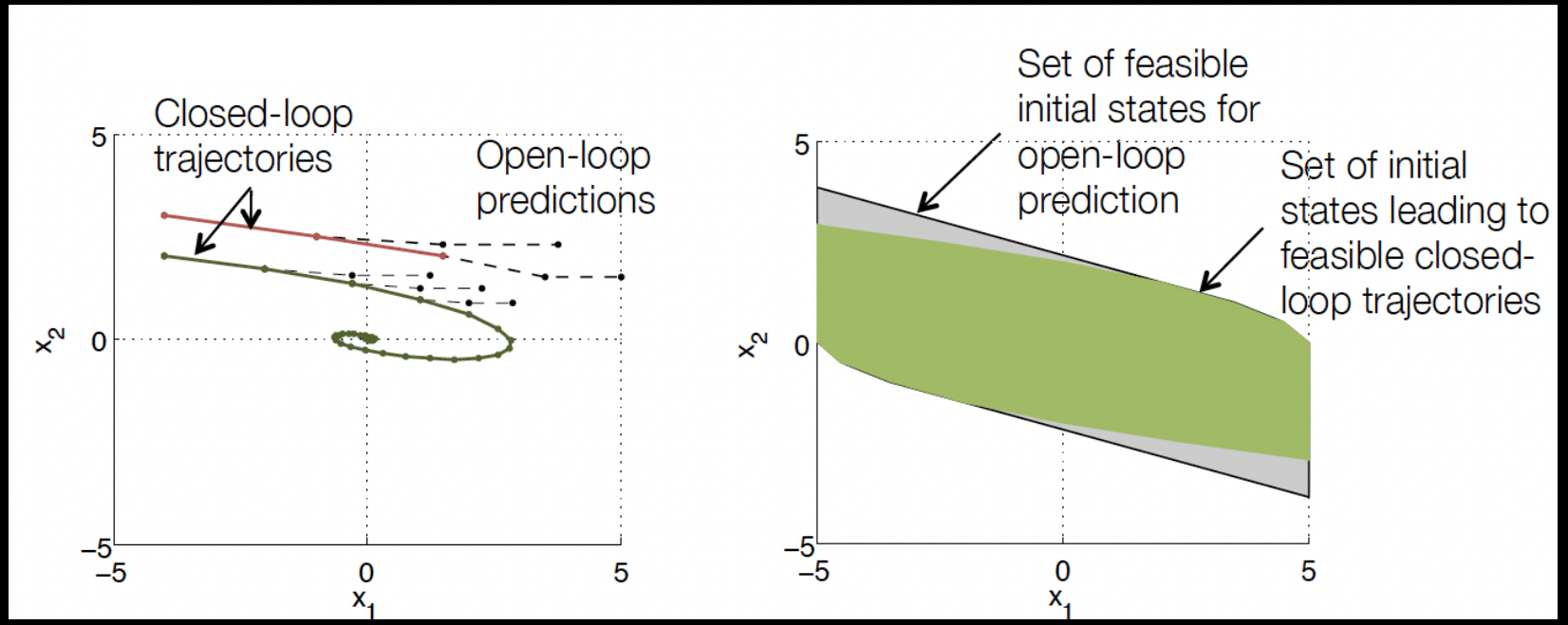
Receding Horizon MPC



Receding Horizon:
Always solve the optimization problem for the same horizon length and only apply the first input of the optimal input variables. Measure the new state and initialize the next optimization with it.
This introduces feedback!

Counterpart: Shrinking horizon but theory and implementation not as straight-forward

Open Loop vs. Closed-Loop



Open loop predictions differ from the closed-loop behavior (the larger N, the less prominent the effect, however, larger optimization problem).
Due to this gap, we face two issues if we would blindly apply MPC:
Feasibility (even without model mismatch and disturbances), Stability

Invariance

Positively Invariant set

A set $\mathcal{O}$ is said to be a positively invariant set for the autonomous system $x(k+1) = g(x(k))$ if

$$x(k) \in \mathcal{O} \;\Rightarrow\; x(k+1) \in \mathcal{O} \;,\; \forall k \in \{0, 1, \dots\}$$

Maximal Positively Invariant Set $\mathcal{O}_\infty$

The set $\mathcal{O}_\infty \subset \mathcal{X}$ is the maximal positively invariant set with respect to $\mathcal{X}$ if $\mathcal{O}_\infty$ is positively invariant and $\mathcal{O}_\infty$ contains all positively invariant sets.

Pre Set

Given a set S and the dynamic system $x(k+1) = g(x(k))$, the **pre-set** of S is the set of states that evolve into the target set S in one time step:

$$\mathrm{pre}(S) := \{x \mid g(x) \in S\}$$

Input: g, $\mathcal{X}$
Output: $\mathcal{O}_\infty$

$\Omega_0 \leftarrow \mathcal{X}$
loop
  $\Omega_{i+1} \leftarrow \mathrm{pre}(\Omega_i) \cap \Omega_i$
  if $\Omega_{i+1} = \Omega_i$ **then**
    return $\mathcal{O}_\infty = \Omega_i$
  end if
end loop

Lyapunov Stability Theory

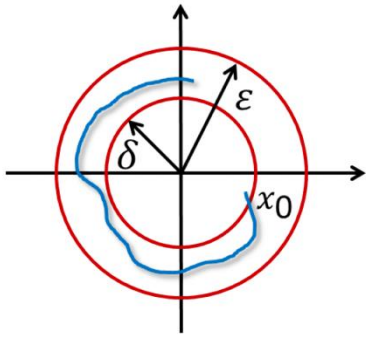
Lyapunov stability

Formally, the equilibrium point $\bar{x}$ of a system (1) is

- **Lyapunov stable** if for every $\epsilon > 0$ there exists a $\delta(\epsilon)$ such that

$$\|x(0) - \bar{x}\| < \delta(\epsilon) \rightarrow \|x(k) - \bar{x}\| < \epsilon, \forall k \geq 0$$

- **unstable** otherwise.



Asymptotic stability

An equilibrium point $\bar{x} \in \Omega$ of system (2) is

- **asymptotically stable** in the positive invariant set $\Omega \subseteq \mathbb{R}^n$ if it is Lyapunov stable and **attractive**, i.e.

$$\lim_{k \to \infty} \|x(k) - \bar{x}\| = 0, \ \forall x(0) \in \Omega$$

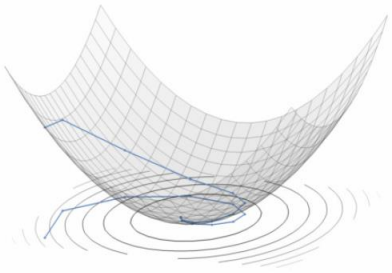
- **globally asymptotically stable** if it is asymptotically stable and $\Omega = \mathbb{R}^n$

Lyapunov Stability Theory

Definition: Global Lyapunov function

Consider the equilibrium point $\bar{x} = 0$ of system (1). A function $V : \mathbb{R}^n \to \mathbb{R}$, continuous at the origin, finite for every $x \in \mathbb{R}^n$, and such that

$$\|x\| \to \infty \Rightarrow V(x) \to \infty$$
$$V(0) = 0 \text{ and } V(x) > 0, \ \forall x \in \mathbb{R}^n \setminus \{0\}$$
$$V(g(x)) - V(x) \leq -\alpha(x) \ \forall x \in \mathbb{R}^n$$



where $\alpha : \mathbb{R}^n \to \mathbb{R}$ is continuous positive definite,

is called a **Lyapunov function**.

Theorem: Global Lyapunov stability (global asymptotic stability)

If a system (1) admits a Lyapunov function $V(x)$, then $x = 0$ is **globally asymptotically stable**.

Think of a Lyapunov function as a system theoretic generalization of energy.

Example:
A mechanical system is asymptotically stable if the total mechanical energy is decreasing over time (friction)

MPC Core Theory

1. Stage cost is positive definite, i.e. it is strictly positive and only zero at the origin
2. Terminal set is **invariant** under the local control law $\kappa_f(x_i)$:

$$x_{i+1} = Ax_i + B\kappa_f(x_i) \in \mathcal{X}_f, \quad \text{for all } x_i \in \mathcal{X}_f$$

   All state and input **constraints are satisfied** in $\mathcal{X}_f$:

$$\mathcal{X}_f \subseteq \mathcal{X},\ \kappa_f(x_i) \in \mathcal{U}, \quad \text{for all } x_i \in \mathcal{X}_f$$

3. Terminal cost is a continuous **Lyapunov function** in the terminal set $\mathcal{X}_f$ and satisfies:

$$l_f(x_{i+1}) - l_f(x_i) \leq -l(x_i, \kappa_f(x_i)), \quad \text{for all } x_i \in \mathcal{X}_f$$

Theorem

The closed-loop system under the MPC control law $u_0^\star(x)$ is asymptotically stable and the set $\mathcal{X}_N$ is positive invariant for the system

$$x(k+1) = Ax(k) + Bu_0^\star(x(k)).$$

Conditions on Terminal Constraint – Applied

- For a linear system with quadratic stage cost (assumption 1 holds):
    - Design the unconstrained LQR control law
    - Choose the terminal set to be the maximum invariant set under the LQR control law for the system: $x^+ = (A + BK)x$
    - While designing the maximum invariant set, make sure that all state and input constraints are respected (given by shown algorithm)
    - Or use as a terminal law u = 0 → $x^+ = Ax + Bu \rightarrow X_f = 0$, but in that case, the feasible set of initial states is a lot smaller than under the LQR law

→ By design, the terminal set is invariant and therefore assumption 2 holds!

Conditions on Terminal Cost – Applied

- Again for a linear system with quadratic stage cost:

- Simply use as a terminal cost, the matrix P, which is the solution of the algebraic Riccati equation. Here, the cost decrease criterion is shown that is a requirement for a continuous Lyapunov function in the terminal set

$$
\begin{aligned}
& x_{k+1}^{\top} P x_{k+1} - x_k^{\top} P x_k \\
& = x_k^{\top} (-P_\infty + A^{\top} P_\infty A + F_\infty^{\top} B^{\top} P_\infty A - F_\infty^{\top} R F_\infty) x_k \\
& = x_k^{\top} (-P_\infty + A^{\top} P_\infty A - A^{\top} P_\infty B (B^{\top} P_\infty B + R)^{-1} B^{\top} P_\infty A - F_\infty^{\top} R F_\infty) x_k \\
& = -x_k^{\top} (Q + F_\infty^{\top} R F_\infty) x_k
\end{aligned}
$$

Linear MPC Setup

Standard formulation for stabilization:

$$
\begin{aligned}
\min_{\{x_k,u_k\}_{k=0}^{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) \; + \; x_N^\top P_\infty x_N \\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k, \quad k = 0, \dots, N-1, \\
& x_0 \text{ given}, \\
& x_k \in \mathcal{X}, \quad u_k \in \mathcal{U}, \quad k = 0, \dots, N-1, \\
& x_N \in \mathcal{X}_f.
\end{aligned}
$$

$$x_k \in \mathcal{X} := \{x \mid H_x x \le h_x\}, \qquad u_k \in \mathcal{U} := \{u \mid H_u u \le h_u\}$$

$$Q \succcurlyeq 0, R \succ 0$$

$$
\begin{aligned}
\mathcal{X}_f^{(0)} &= \{x \in \mathcal{X} \mid Kx \in \mathcal{U}\}, \\
\mathcal{X}_f^{(i+1)} &= \{x \in \mathcal{X}_f^{(i)} \mid A_K x \in \mathcal{X}_f^{(i)}\}, \\
\mathcal{X}_f &= \lim_{i \to \infty} \mathcal{X}_f^{(i)}.
\end{aligned}
$$

Solve the optimization problem and apply u_0^* to the real system, obtain x(k+1) and reinitialize the optimization problem

Reference Tracking with MPC

Need to identify steady state and input which gives us the following target condition (in our case H=C)

$$x_s = Ax_s + Bu_s$$
$$Hx_s = r$$

In the presence of constraints, this leads us to the following optimization problem

$$\begin{aligned} &\min\ u_s^T R_s u_s \\ &\text{subj. to } \begin{bmatrix} I - A & -B \\ H & 0 \end{bmatrix} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} 0 \\ r \end{bmatrix} \\ &\qquad x_s \in \mathcal{X}, \quad u_s \in \mathcal{U}. \end{aligned}$$

$$\begin{aligned} &\min\ (Hx_s - r)^T Q_s (Hx_s - r) \\ &\text{subj. to } x_s = Ax_s + Bu_s \\ &\qquad x_s \in \mathcal{X}, \quad u_s \in \mathcal{U}. \end{aligned}$$

If target is not reachable, do this as a backup, find the closest state to the target

Reference Tracking with MPC

Use the same idea as presented quickly for the LQR. Do a coordinate transformation and shift the variables (and here also the constraints) to recover a regulation problem (instead to zero, we regulate now to the reference).

$$\begin{aligned} \Delta x &= x - x_s \\ \Delta u &= u - u_s \end{aligned} \quad \Rightarrow \quad \begin{aligned} \Delta x_{k+1} &= x_{k+1} - x_s \\ &= Ax_k + Bu_k - (Ax_s + Bu_s) \\ &= A\Delta x_k + B\Delta u_k \end{aligned}$$

$$\begin{aligned} G_x x \leq h_x \Rightarrow \quad & G_x \Delta x \leq h_x - G_x x_s \\ G_u u \leq h_u \Rightarrow \quad & G_u \Delta u \leq h_u - G_u u_s \end{aligned}$$

Reference Tracking with MPC

$$\min \sum_{i=0}^{N-1} \Delta x_i^T Q \Delta x_i + \Delta u_i^T R \Delta u_i + V_f(\Delta x_N)$$

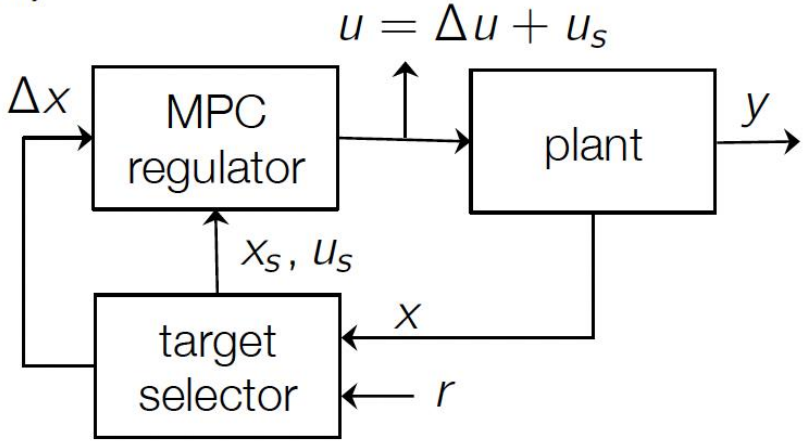
$$\text{s.t. } \Delta x_0 = \Delta x(k)$$

$$\Delta x_{i+1} = A\Delta x_i + B\Delta u_i$$

$$G_x \Delta x_i \leq h_x - G_x x_s$$

$$G_u \Delta u_i \leq h_u - G_u u_s$$

$$\Delta x_N \in \mathcal{X}_f$$



Initialize with $\Delta x(k) = x(k) - x_s$ and apply $u_0^* = \Delta u_0^* + u_s$ to the real system

Linear MPC – Task 6

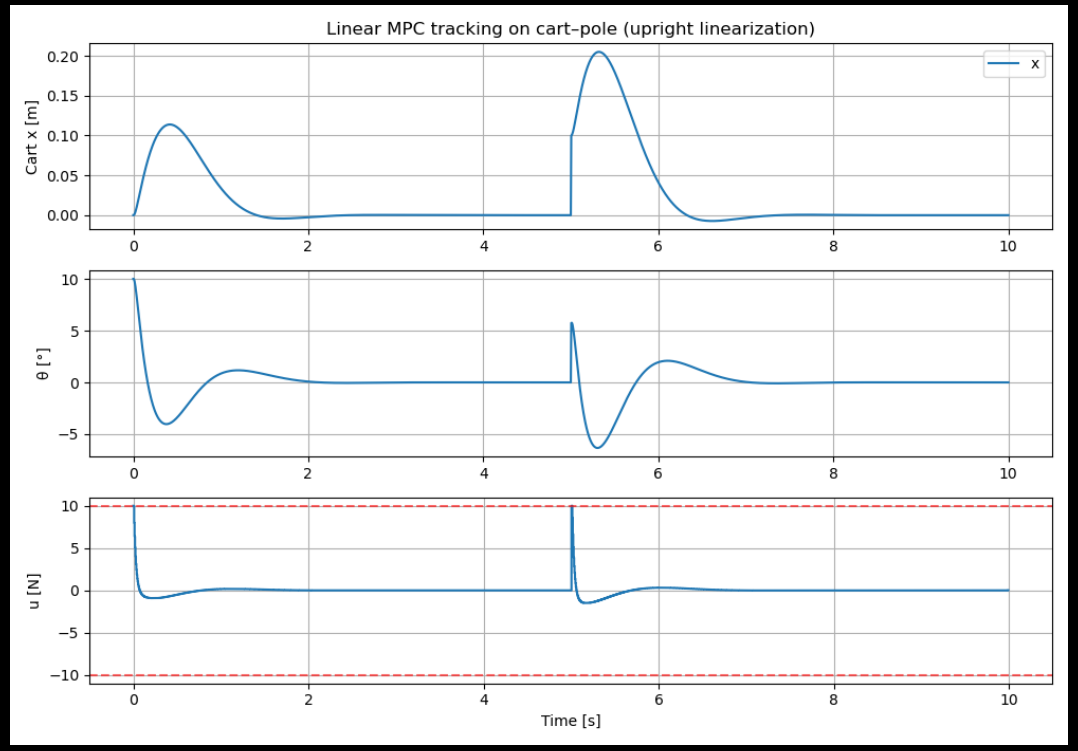
- To comply with the constraints of the investors, implement a linear MPC to stabilize the cart-pole system.

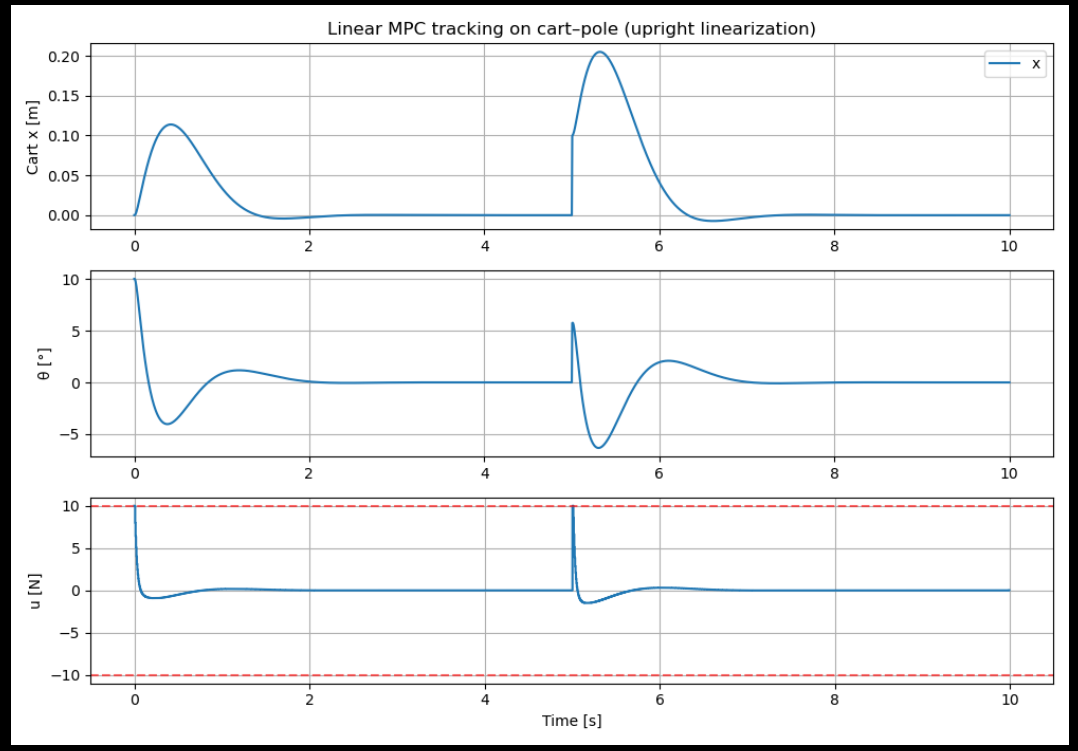
  Constraints: $u \in [-10, 10]\mathrm{N}, x \in [-0.45, 0.45]$ (position)

- If that works, include the reference estimator and let it track a reference (first the easy step function, then the nonlinear reference)

Use: solve_discrete_are, np.linalg.inv, np.vstack, cp.Variable, cp.Parameter, cp.quad_form, np.block

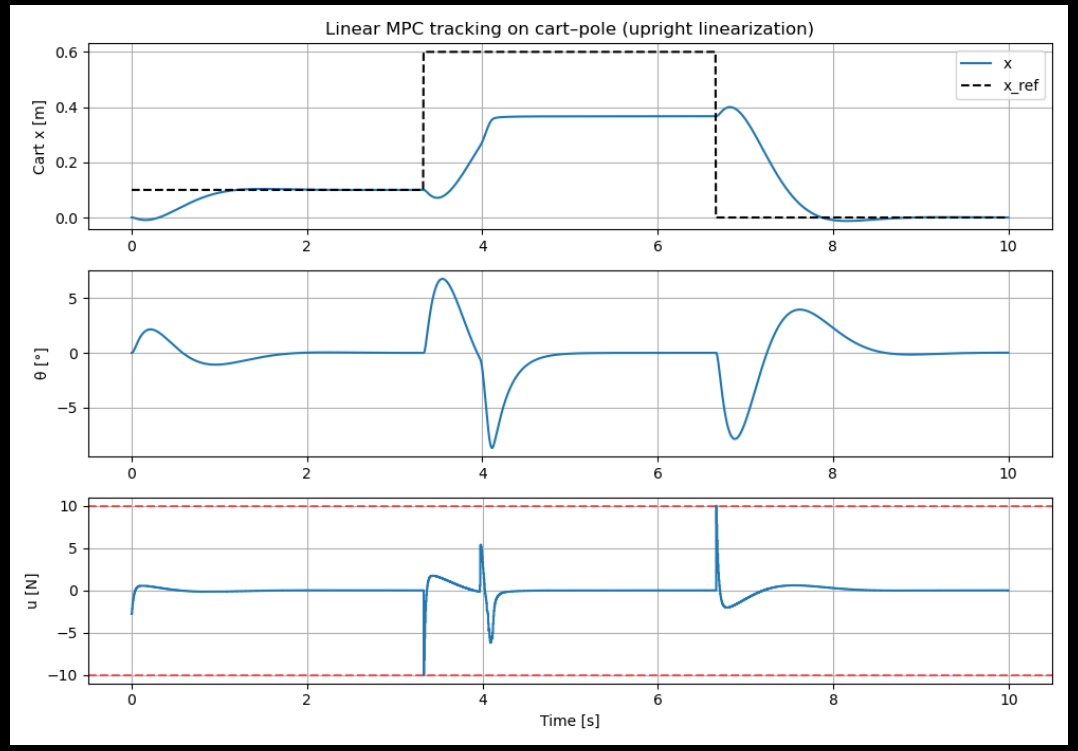
Linear MPC – Task 6: Stabilization (N=5 vs. 15)

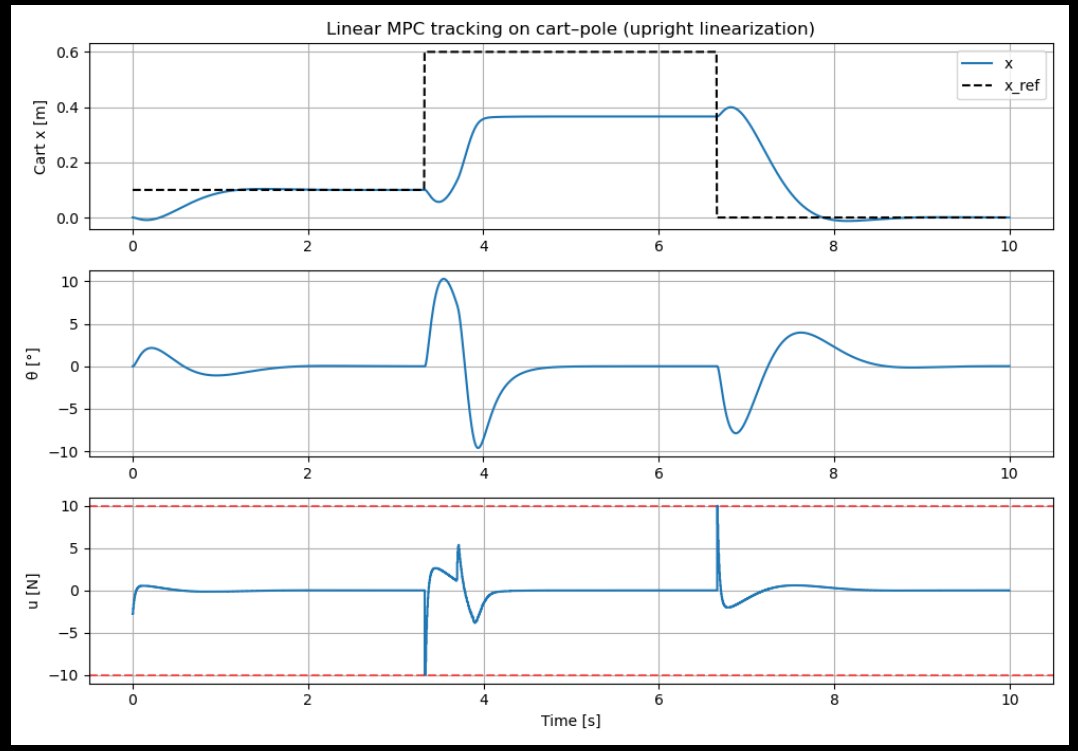




Hardly any difference between N=5 and N=15. Stabilization can be achieved with low horizons and the disturbance is anyways not predictable, therefore, a longer horizon offers no benefit.

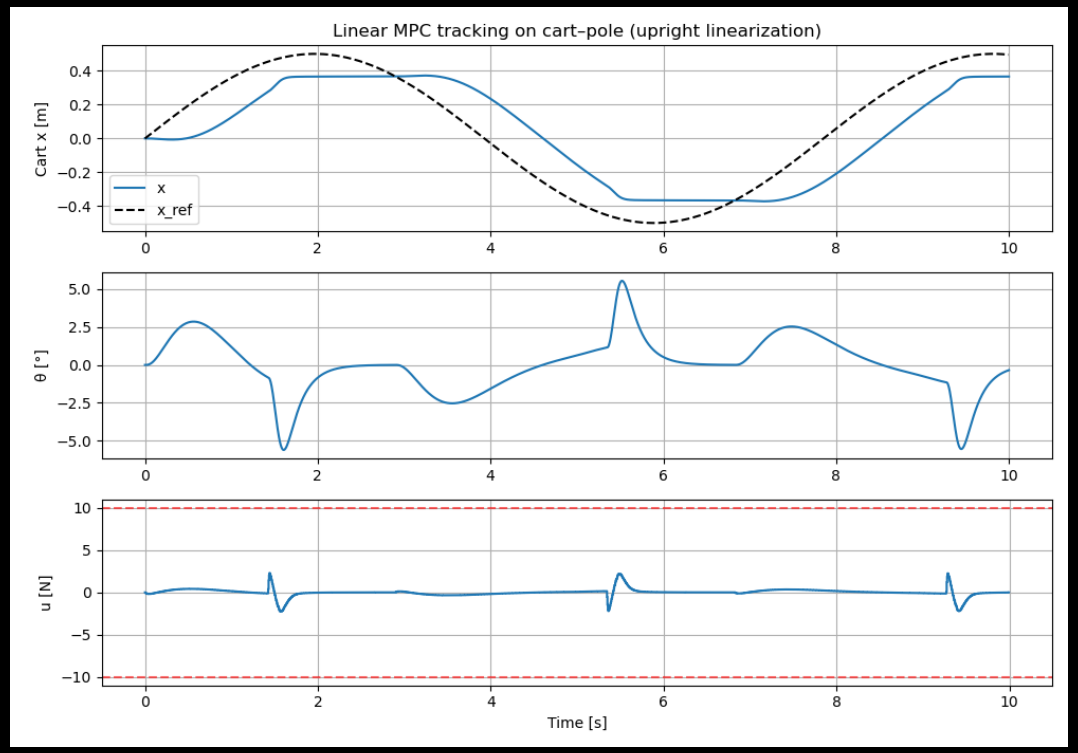
Linear MPC – Task 6: Reference (N=5 vs. 15)

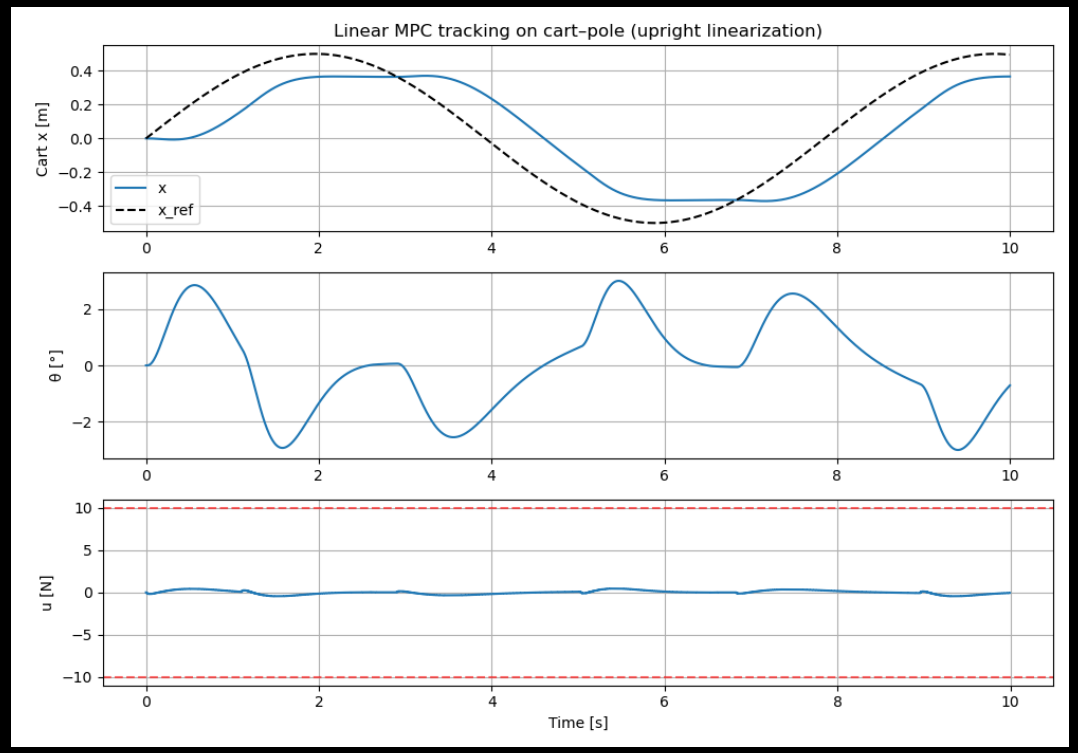




In the case of reference tracking, a longer horizon can be beneficial, as the actions can be planned based on a longer reference signal. We can see that with N=15 the reaction is faster for the jump at 4 seconds.

Linear MPC – Task 6: Nonlinear Reference





For a nonlinear reference, the effect of different horizon lengths is a bit better visable (left N=15, right N = 15). The overall control on a longer horizon is a lot smoother and leads to less deviations in the angle.

What about Nonlinear References?

- Your boss is not yet happy with simple steps that your system can track, he wants to see nice smooth references.

- You don‘t have much time left, so you blindly apply MPC to the nonlinear system without thinking about the terminal set and terminal cost.

- In practice this can work, especially for high update frequencies and large horizons (but then computation might take longer than an update, which is an issue). Moreover, all guarantees are out of the window.

Nonlinear MPC

Looking closely at the MPC core theory, we did never require that the system is linear.

Therefore, if you can fulfill these assumptions, you still get guarantees!

But, unfortunately, for general nonlinear systems, it‘s super hard to find the terminal set and Lyapunov functions

Active research area and usually other concepts are used (ISS stability, incremental Lyapunov stability, etc.)

Nonlinear MPC

There is still some light at the end of the tunnel:

- In practice, under large enough horizons, you can linearize your system around your reference and still use the LQR for terminal set and cost

- If your optimization problem is differentiable and it has QP nature, can use special SQP (sequential QP) solvers that perform well in practice.

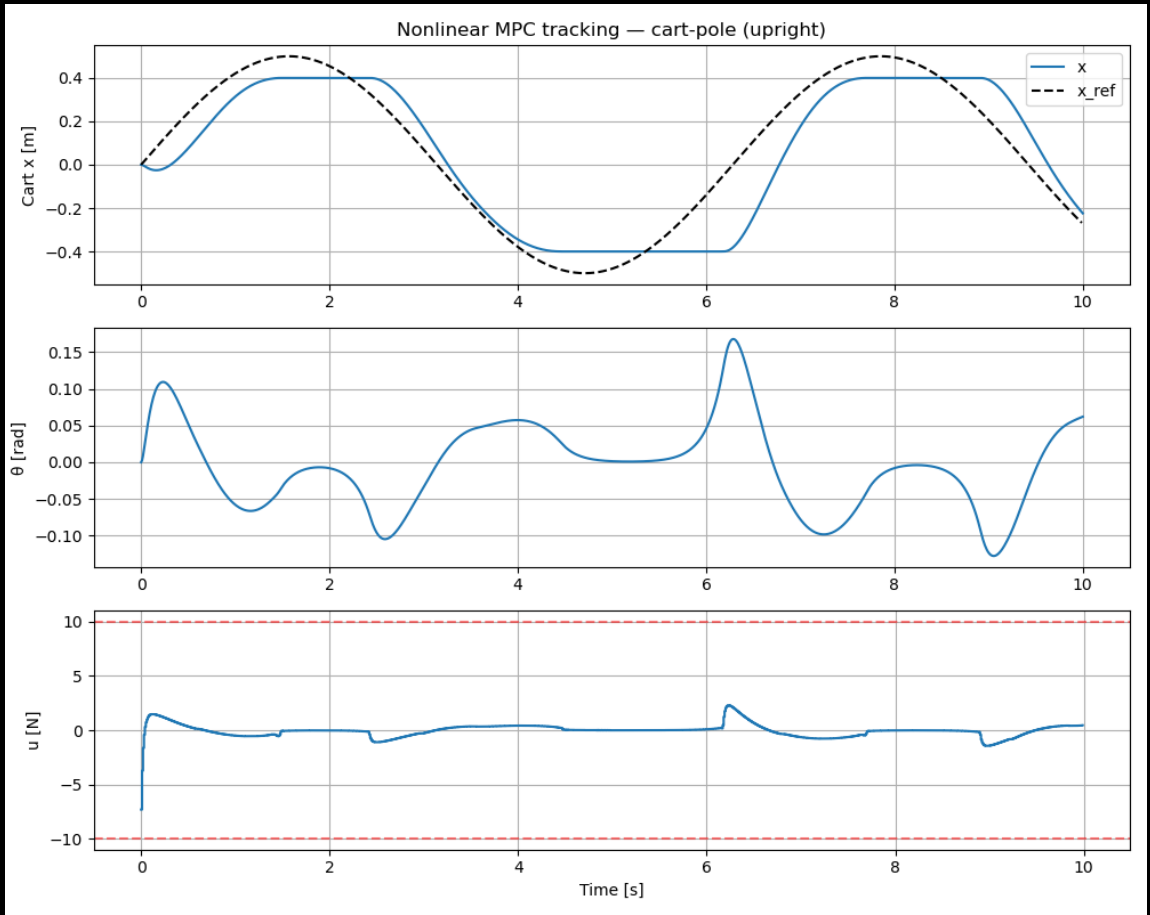
Nonlinear MPC – Task 7

Since the investors arrive in 10 minutes and your boss wants to have some fancy reference tracked by the system, you quickly implement a Nonlinear MPC without considering assumptions on the terminal constraint and cost.

You choose a large enough horizon, and hope for the best!

Use: ca.SX.sym, ca.Function, ca.mtimes

Nonlinear MPC – Task 7



Nonlinear MPC (with N = 100) manages to follow the reference quite closely even in the presence of constraints

But what‘s also important, compared to linear MPC, the nonlinear MPC reacts better to the reference changes and the angle is a lot closer to zero.

MPC in Practice

Explicit MPC

- Precompute the control law as a function of initial state
- For linear MPC + quadratic cost, the controller is a piecewise affine function
- Online evaluation of a piecewise affine function is in the ns range (very fast)
- Can only be done for small systems (3-6 states)

Soft-Constrained MPC

$$\min_{\mathbf{u}} \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i + l_\epsilon(\epsilon_i) + x_N^T P x_N + l_\epsilon(\epsilon_N)$$

$$\text{s.t. } x_{i+1} = A x_i + B u_i$$
$$H_x x_i \leq k_x + \epsilon_i,$$
$$H_u u_i \leq k_u,$$
$$\epsilon_i \geq 0$$

Theorem: Exact Penalty Function

$l_\epsilon(\epsilon) = v \cdot \epsilon$ satisfies the requirement for any $v > \lambda^\star \geq 0$, where $\lambda^\star$ is the optimal Lagrange multiplier for the original problem.

Advanced MPC Topics – Robustness

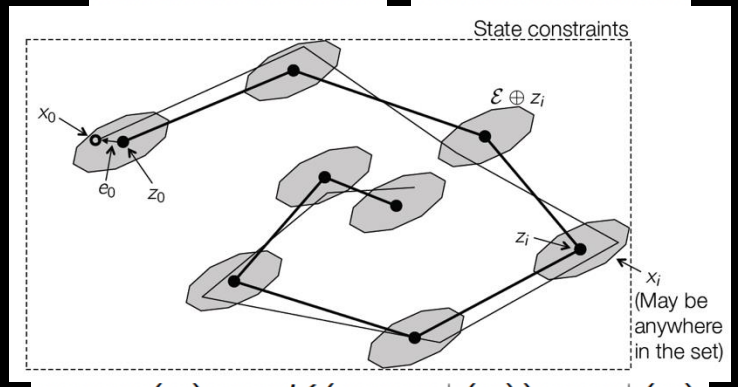
Bounded Disturbance

Tube MPC: do MPC on the nominal system, but tighten the constraints based on minimum invariant set of the error dynamics that are stabilized by an LQR law

$$z_i \in \mathcal{X} \ominus \mathcal{E}$$

$$v_i \in \mathcal{U} \ominus K\mathcal{E}$$

$$x_0 \in z_0 \oplus \mathcal{E}$$



$$\mu_{\text{tube}}(x) := K(x - z_0^\star(x)) + v_0^\star(x)$$

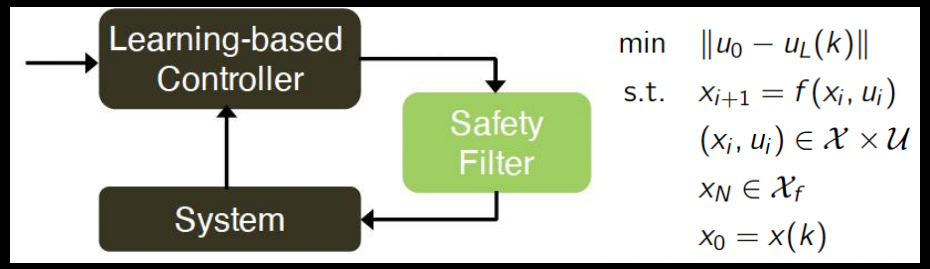
Stochastic Disturbance

- Quite experimental, not so many guarantees at the moment, still works quite well in practice for some systems

Key words:

- Asymptotic Average Performance
- Chance Constraints

Advanced MPC Topics – Learning



$$\begin{aligned} \min \quad & \|u_0 - u_L(k)\| \\ \text{s.t.} \quad & x_{i+1} = f(x_i, u_i) \\ & (x_i, u_i) \in \mathcal{X} \times \mathcal{U} \\ & x_N \in \mathcal{X}_f \\ & x_0 = x(k) \end{aligned}$$

Safety filter: $\pi_f(u_L, x) = \begin{cases} u_L & \text{if } f(x, u_L) \in \mathcal{O} \text{ and } u_L \in \mathcal{U} \\ \pi_s(x) & \text{otherwise (maintains invariance property of } \pi_s) \end{cases}$

Learning-based MPC

$$x(k+1) = \underbrace{Ax(k) + Bu(k)}_{\text{nominal dynamics}} + \underbrace{g(x(k), u(k)) + w(k)}_{:= \bar{w}(k) \text{ model error}}$$

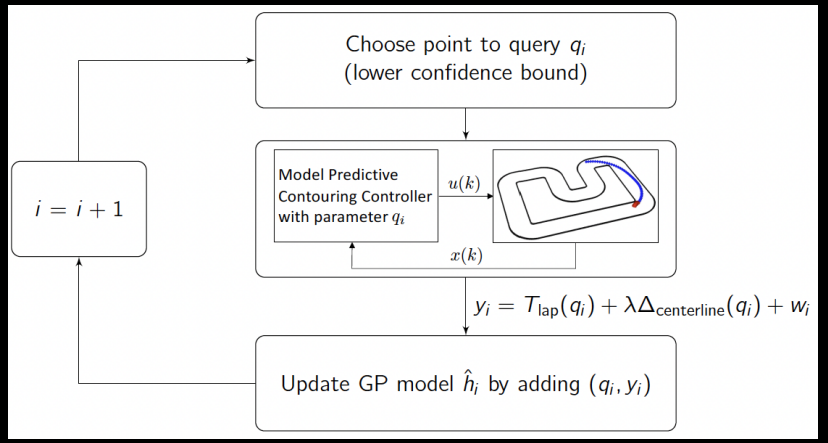
$$J_k^\star(x(k)) = \min_u \; l_f(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i)$$

$$\begin{aligned} \text{s.t.} \quad & x_0 = x(k), \quad z_0 = x(k) \\ & x_{i+1} = Ax_i + Bu_i + \mathcal{O}_k(x_i, u_i) \\ & z_{i+1} = Az_i + Bu_i \\ & z_i \in \mathcal{X} \ominus \mathcal{F}_i, \quad u_i \in \mathcal{U} \ominus K\mathcal{F}_i \\ & z_N \in \mathcal{X}_f \ominus \mathcal{F}_N \end{aligned}$$

learned-model nominal model

tighten constraints on nominal system based on bound of model mismatch

Bayesian Optimization



Summary

- Start simple
- The more time you invest in a good model, the easier the control task becomes
- Simple steady-state control for SISO systems → PID
- Stabilization / Reference tracking close to equilibrium points without any constraints → LQR
- Constraints but slowly varying references → Linear MPC
- Non-linearizable system, fast-changing, nonlinear references → Nonlinear MPC x Google Scholar

What‘s missing?

Today, we assumed that we have perfect state knowledge and that all references are given.

In the second edition of the workshop, we will see how we can estimate states and develop a plan we want to follow

State estimation / Sensor Fusion:

- For linear systems → Luenberg Observers, Kalman Filter
- For general systems → Extended Kalman Filter, Particle Filter

Planning:

- Online Planning → racing line optimization
- Discrete decisions → Dynamic Programming
- Complex Tasks → RL (basically develop a plan with RL and track it with an MPC)

Credits

Lots of material taken from the following courses:

- Control Systems I and II by Lino Guzzella
- Large-Scale Convex Optimization by Michael Mühlebach
- (Advanced) Model Predictive Control by Melanie Zeilinger, Johannes Köhler, Kim Wabersich, Andrea Carron