Systems Laboratory, Spring 2025

Damiano Varagnolo –

what is control

Contents map

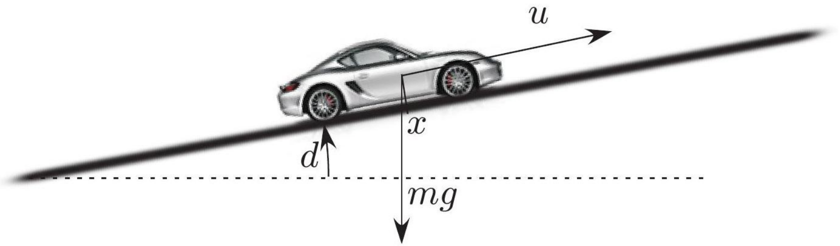
developed content units	**taxonomy levels**
feedforward	u1, e1
feedback	u1, e1
model based control	u1, e1
model free control	u1, e1

prerequisite content units	**taxonomy levels**
ODE	u1, e1

Main ILO of sub-module “what is control”

Interpret automatic control as an opportune operation on the dynamics of a system

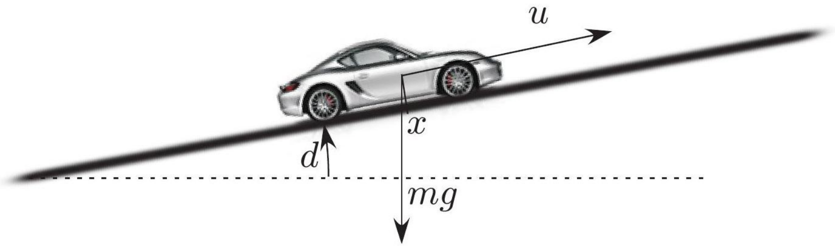
Example: speed control



main parameters:

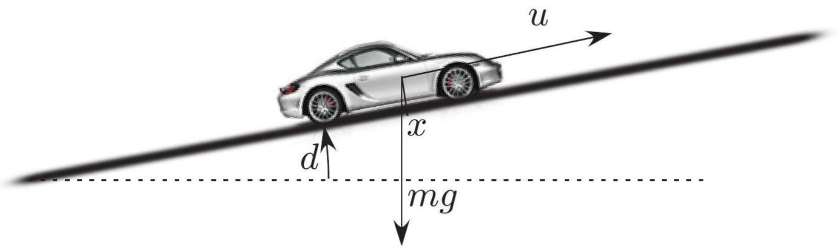
- m mass of the car
- b friction coefficient
- g gravity coefficient

A sufficiently accurate model for the purpose



from Newton laws: $m\ddot{x} = -b\dot{x} + u - mg\sin(d)$

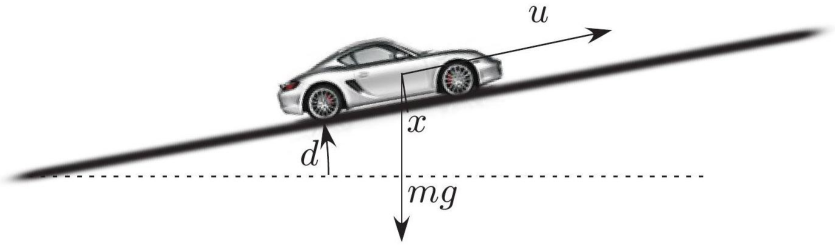
A sufficiently accurate model for the purpose

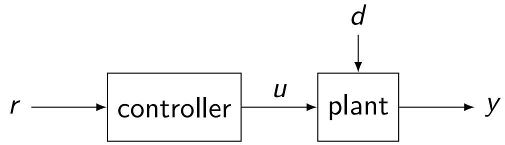


from Newton laws: $m\ddot{x} = -b\dot{x} + u - mg\sin(d)$

control objective: minimize $|y(t) - r(t)|$ with $r(t) =$ wished speed

Feedforward control: "I think I know which $d(t)$ will happen, and I compensate for that"





$u(t)$ = something that compensates that $\widehat{d}(t)$

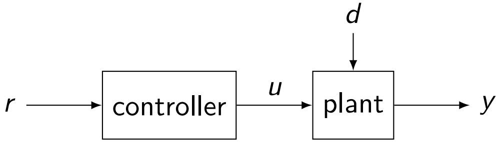
Open loop / feedforward control: "I think I know which $d(t)$ will happen, and I compensate for that"

$$u(t) = \text{something that compensates that } \widehat{d}(t)$$

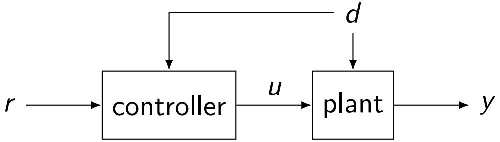
problem: if $\widehat{d} - d$ is big, then we expect $y - r$ to be big too, and we won't be able to note this!

Note that open loop $\neq$ feedforward

Open loop:



Feedforward:



Question 1

Open loop / feedforward control is so simple and naïve that no system in the world uses it

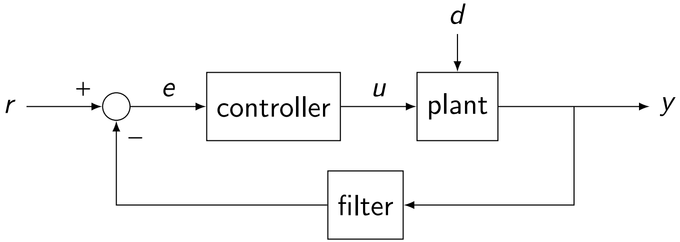
Potential answers:

I: true

II: false

III: I do not know

Feedback control: “I measure something, and depending on what I measure I take a decision”

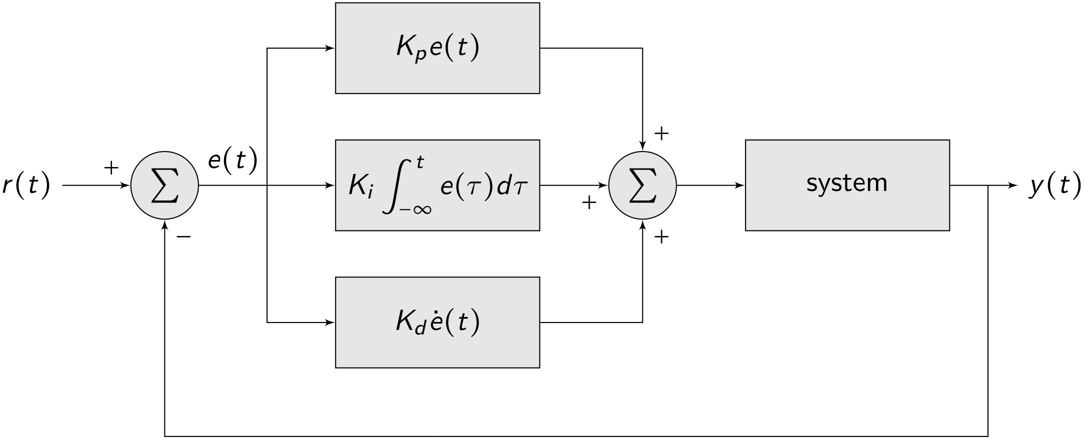


(= designing a controller, i.e., in this case designing a function that maps the signal e into the signal u)

Main dichotomy on how to build a feedback controller

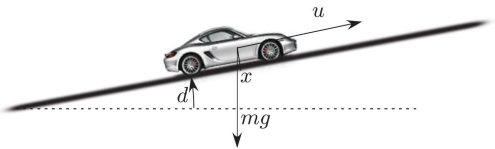
- model free *(e.g., PIDs)*
- model based *(e.g., MPCs)*

Crash-slide on PIDs



implicit assumption: we can measure $y(t)$*!* (see also `https://www.youtube.com/watch?v=UR0hOmjaHp0`!)

PID for the speed control task



$$
\begin{aligned}
u(t) = \quad & K_p \text{ error right now} \\
& + K_i \text{ sum of all past errors} \\
& + K_d \text{ current tendency of the error}
\end{aligned}
$$

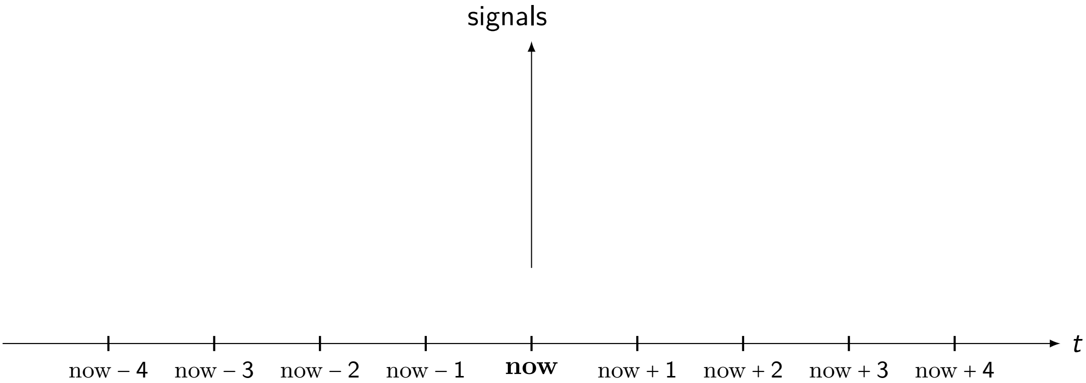
Question 2

A PID is guaranteed to work well

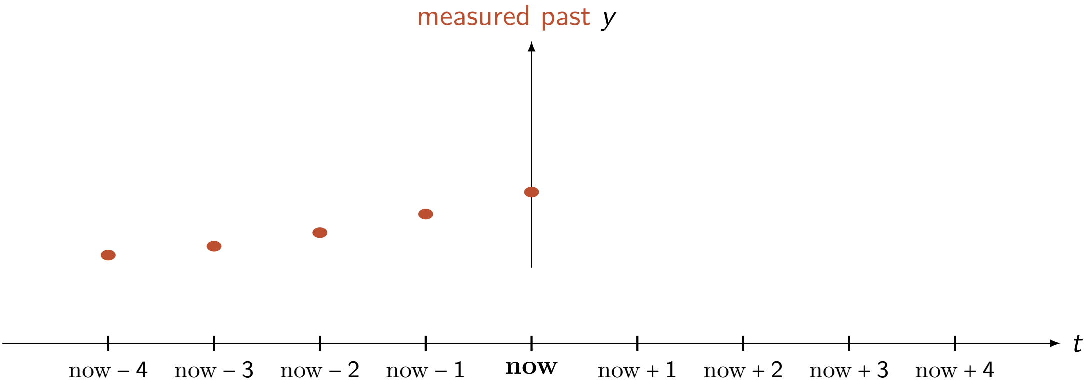
Potential answers:

I: yes, always
II: no, never
III: no, it depends on how well tuned it is
IV: I do not know

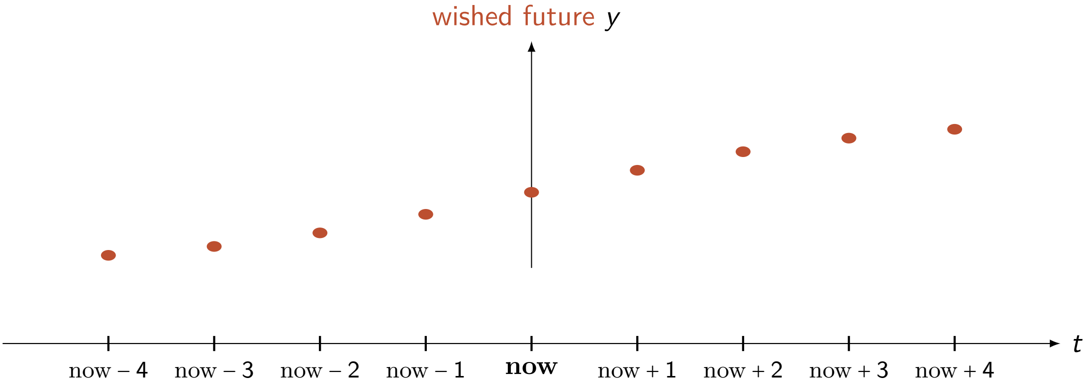
Crash-slide on MPCs



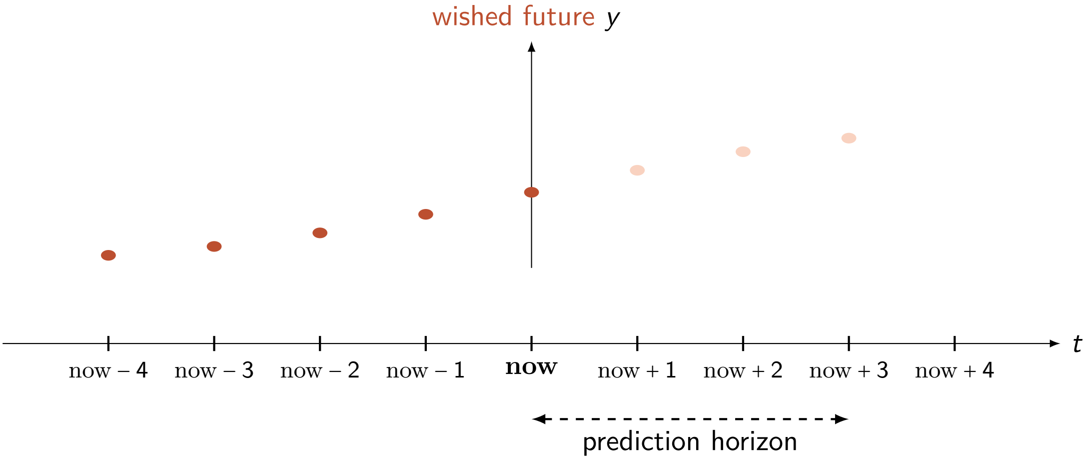
Crash-slide on MPCs



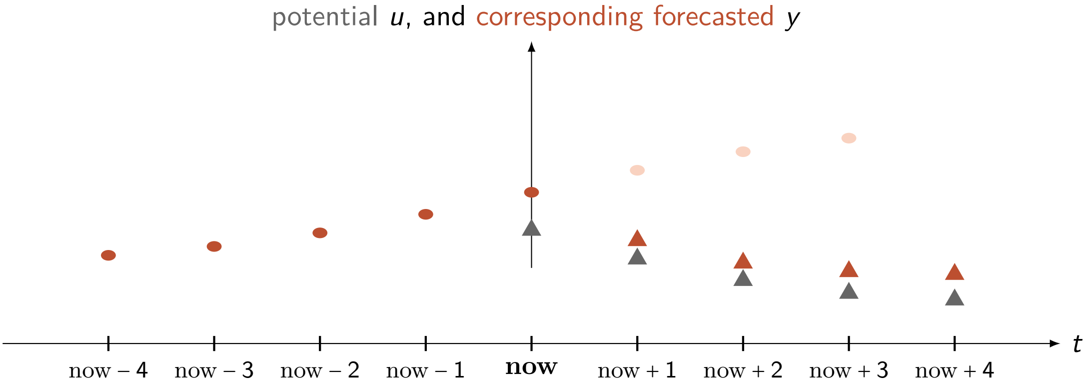
Crash-slide on MPCs



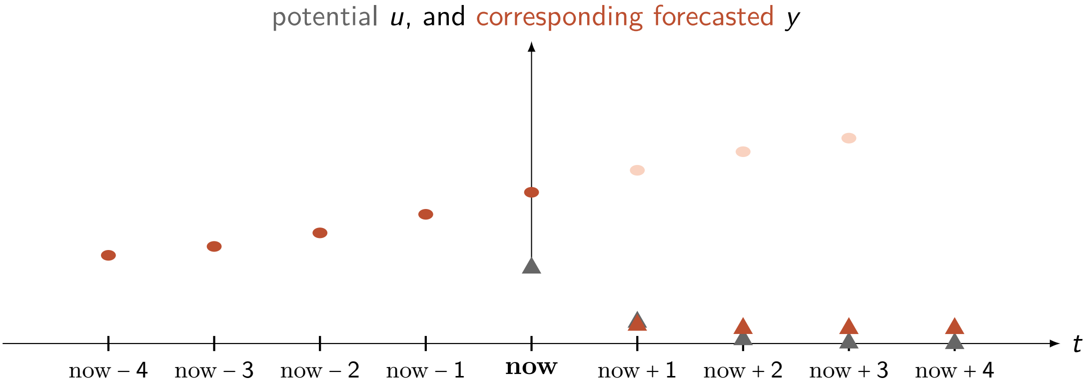
Crash-slide on MPCs



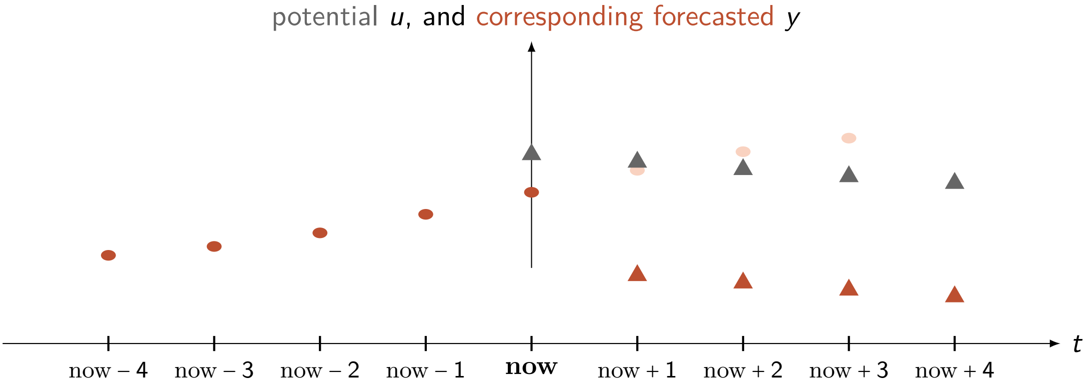
Crash-slide on MPCs



Crash-slide on MPCs

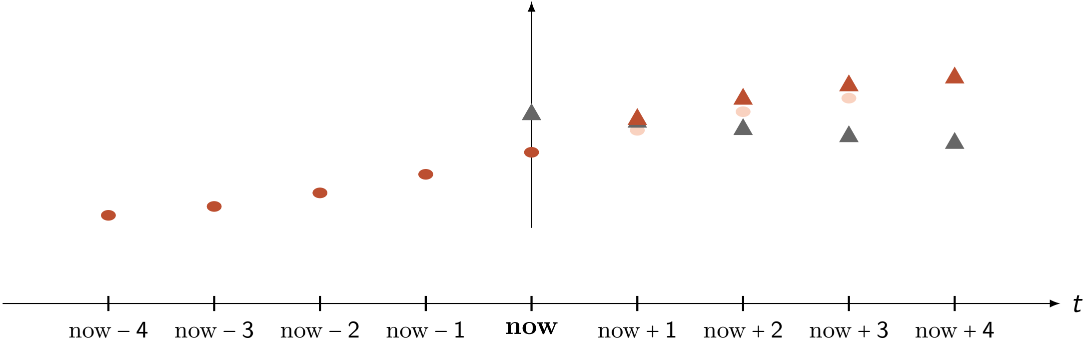


Crash-slide on MPCs

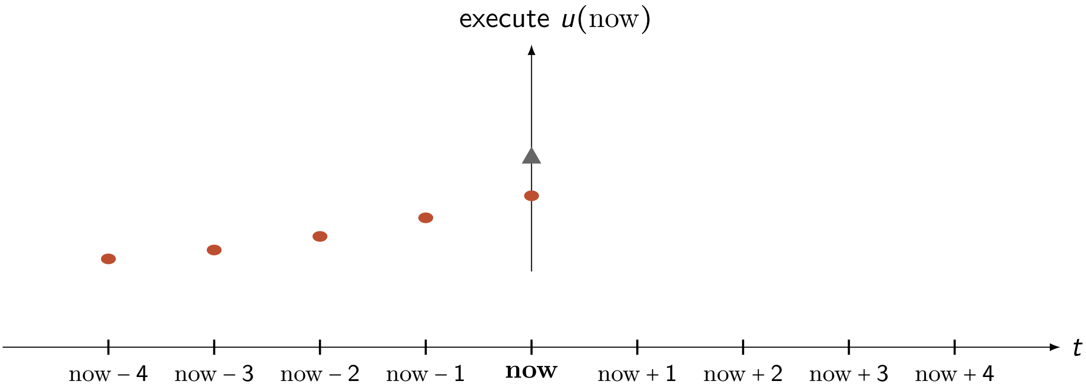


Crash-slide on MPCs

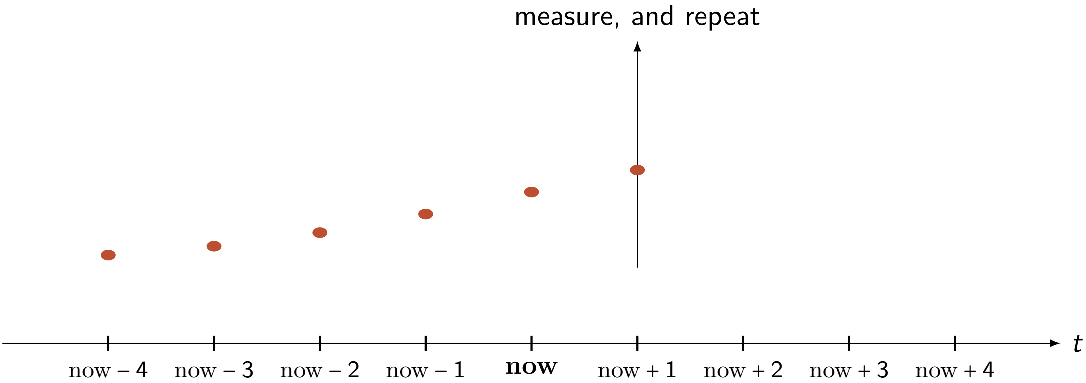
best u = "best" forecasted y



Crash-slide on MPCs



Crash-slide on MPCs



Question 3

A MPC is guaranteed to work well

Potential answers:

I: yes, always

II: no, never

III: no, it depends on how good the model is

IV: I do not know

Question 4

Is MPC guaranteed to work better than PID?

Potential answers:

I: yes, always
II: no, never
III: no, it actually depends on the situation
IV: I do not know

Question 5

Is closed loop control guaranteed to work better than open loop control?

Potential answers:

I: yes, always

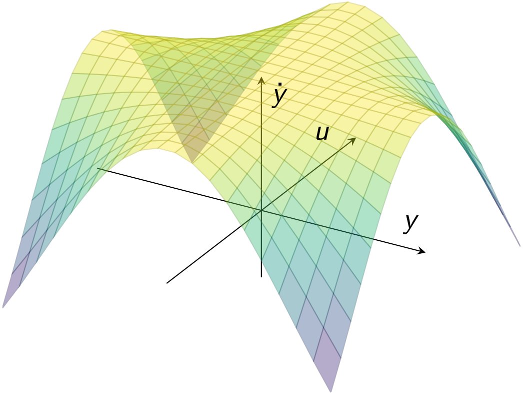
II: no, never

III: no, it actually depends on the situation

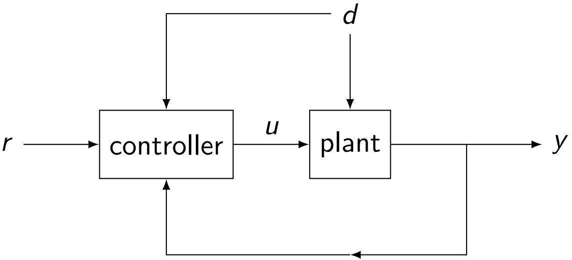
IV: I do not know

But eventually, what is control?

an algorithm to compute $u(t)$ starting from the available information



A final note: in practice it is a good choice to combine both feedback and feedforward actions



Summarizing

Interpret automatic control as an oppor-
tune operation on the dynamics of a system

- think at what feedforward and feedback mean
- think at the fact that essentially they are ways of computing u, and that that u enters the dynamics of the system

Most important python code for this sub-module

Note: going through everything here would take months - just be aware of their existence and start playing with them

- `https://python-control.readthedocs.io/en/0.10.1/`
- `https://pypi.org/project/simple-pid/`
- `https://www.do-mpc.com/en/latest/`

Self-assessment material

Question 6

PID control requires a model of the system to function correctly.

Potential answers:

I: yes, always
II: no, it works without a model
III: I do not know

Question 7

Model Predictive Control (MPC) can only be applied when the model is perfect.

Potential answers:

I: yes, the model must be perfect
II: no, it works with approximate models
III: I do not know

Question 8

Feedforward control is generally better than feedback control for handling disturbances.

Potential answers:

I: yes, feedforward is always better
II: no, feedback control is better for disturbances
III: I do not know

Question 9

Open-loop control is more reliable than closed-loop control in all situations.

Potential answers:

I: yes, open-loop is always more reliable
II: no, it depends on the system and application
III: I do not know

Question 10

PID controllers are always preferable to MPC in terms of performance.

Potential answers:

I: yes, PID always outperforms MPC
II: no, it depends on the system and objectives
III: I do not know

Recap of sub-module "what is control"

- designing a controller means designing an algorithm that transforms information into decision
- there are several types of controllers, each with pros and cons
- taking decisions (i.e., actuating u) means modifying the dynamics of the system

?