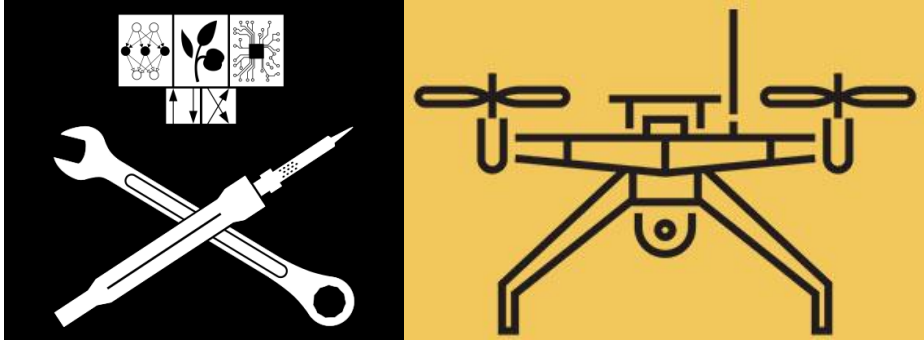


Controllability and open-loop control

Lecture 10

Jerome Jouffroy, Professor
jerome@sdu.dk

SDU

Today's lecture

- Controllability concept and criterion
- Basics of open-loop control for linear systems
- Polynomial trajectories
- Open-loop control for general linear systems

Controllability in a nutshell (1/2)

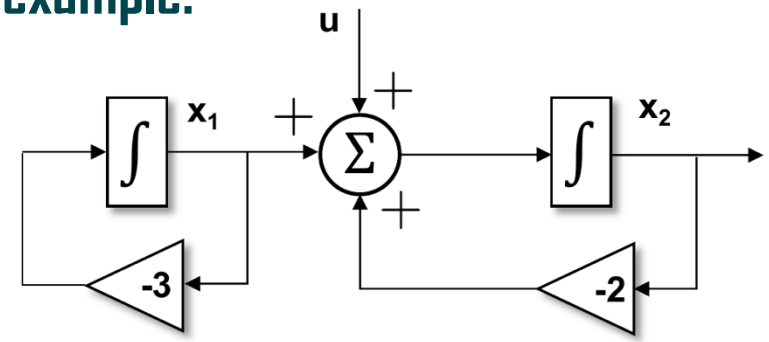
Basic controllability notion: Starting from the origin, can we use u(t) to bring the state to any target state x_T?

➡ u(t) needs to be able to influence all state components

(directly or indirectly)

Graphical interpretation with block diagrams:

First example:



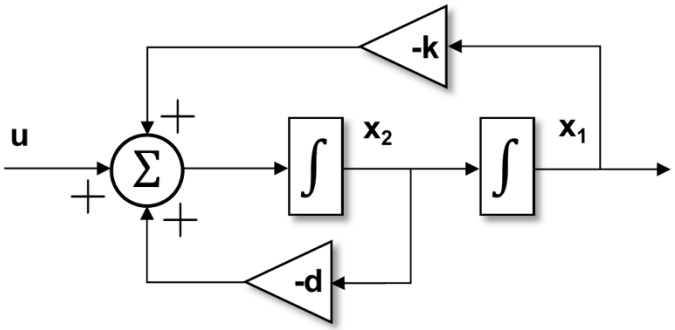
$$\begin{cases} \dot{x}_1 = -3x_1 \\ \dot{x}_2 = -2x_2 + x_1 + u \end{cases}$$

follow the flow of the arrows: u influences x_2 but not x_1

➡ this system is NOT controllable

Controllability in a nutshell (2/2)

Second example:



$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{bmatrix} = \begin{bmatrix}0 & 1\\ -\frac{k}{m} & -\frac{d}{m}\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{m}\end{bmatrix}u(t)$$

u influences directly x_2 and indirectly x_1 (through x_2)

this system is controllable

Definition: controllability

The system represented by: $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u$ is controllable if, for $\mathbf{x}(0) = 0$ and for any point $\mathbf{x}_T$ of the state-space, there exists a finite time $T > 0$ and a control input $u(t)$, $t \in [0, T]$, such that $\mathbf{x}(T) = \mathbf{x}_T$. □

The Kalman criterion for checking controllability

Systematic way to check controllability: Kalman criterion

consider the controllability matrix defined as

$$\mathbf{W}_c := \left[\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}, ..., \mathbf{A}^{n-1}\mathbf{B}\right]$$

Main result

The system $\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{B}u$ is controllable if, and only if the controllability matrix $\mathbf{W}_c$ as given by equation (4.16) is **invertible**! □

Example:

$$\dot{\mathbf{x}} = \begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 1 \\ 0 & 1 & -1 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} u$$

compute

$$\mathbf{W}_c = \left[\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}\right] = \begin{bmatrix} 0 & 1 & -2 \\ 1 & -1 & 3 \\ 0 & 1 & -2 \end{bmatrix}$$

$$\Rightarrow \det(\mathbf{W}_c) = 0$$

this system is NOT controllable

Controllability and the Controllability Canonical Form (CCF)

Throughout this course, we have often seen this form:

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \\ -a_0 & -a_1 & -a_2 & -a_3 & \dots & -a_{n-1} \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ b \end{bmatrix} u(t)$$

It is called a Controllability Canonical Form (CCF)

why?

$$\mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ b \end{bmatrix} \quad \mathbf{AB} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ b \\ * \end{bmatrix} \quad \mathbf{A}^2\mathbf{B} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ b \\ * \\ * \end{bmatrix}$$

and so on until $\mathbf{A}^{n-1}\mathbf{B} = \begin{bmatrix} b \\ * \\ * \\ \vdots \\ * \\ * \end{bmatrix}$

S → **W_c shows that this system is <u>always controllable</u> as long as $b \neq 0$**

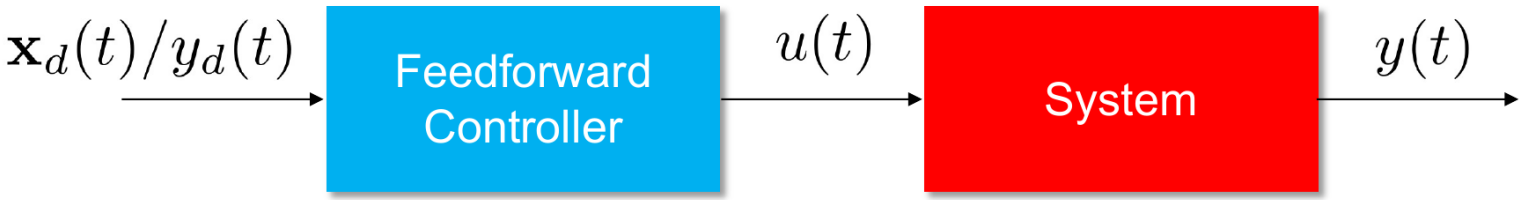
The open-loop control problem

control is not always only about feedback

open-loop control objective:

Make the system follow a pre-defined trajectory $y_d(t)/x_{1d}(t)$

by computing the corresponding $u(t) = u_d(t)$



advantages:

- no sensor is needed to do real-time control (lower budget on sensing)
- computationally more effective (everything is pre-computed, no need to sensing-computing-actuation loop)

alternative names: feedforward control, motion planning

Open-loop control for CCFs (1/2)

Take the CCF

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = x_4 \\ \vdots \\ \dot{x}_{n-1} = x_n \\ \dot{x}_n = -a_0 x_1 - a_1 x_2 - \ldots - a_{n-1} x_n + bu \end{cases} \quad \text{with} \quad y = x_1$$

Assume that $y_d(t)$ is a polynomial desired trajectory. If y(t) follows $y_d(t)$, then

- $x_1(t) = x_{1d}(t) = y_d(t)$ is a polynomial
- $x_2(t) = \dot{x}_1(t) = \dot{x}_{1d}(t) = \dot{y}_d(t)$ is a polynomial (ie derivative of polynomial)
- $x_3(t) = \dot{x}_2(t) = \ddot{x}_1(t) = \ddot{x}_{1d}(t) = \ddot{y}_d(t)$ is a polynomial
- $x_n(t) = \dot{x}_{n-1}(t) = x_1^{(n-1)}(t) = y_d^{(n-1)}(t)$ is a polynomial
- $\dot{x}_n(t) = x_1^{(n)}(t) = y_d^{(n)}(t)$ is a polynomial

finally, isolating u(t) in the CCF:

$$u(t) = u_d(t) = \frac{1}{b}\left[a_0 y_d(t) + a_1 \dot{y}_d(t) + \ldots + a_{n-1} y_d^{(n-1)}(t) + y_d^{(n)}(t)\right]$$

(sum of polynomials)

Open-Loop controller for a CCF

Open-loop control: the MSD example

Example:

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\end{bmatrix} = \begin{bmatrix}0 & 1\\ -k/m & -d/m\end{bmatrix}\begin{bmatrix}x_1(t)\\ x_2(t)\end{bmatrix} + \begin{bmatrix}0\\ 1/m\end{bmatrix}u(t)$$

with $y(t) = x_1(t)$

Take the polynomial desired trajectory $y_d(t)$

If the output follows the desired trajectory, then

$$x_1(t) = y_d(t) \quad \text{is a polynomial}$$

$$x_2(t) = \dot{x}_1(t) = \dot{x}_{1d}(t) = \dot{y}_d(t) \quad \text{is a polynomial}$$

$$\dot{x}_2(t) = \ddot{x}_1(t) = \ddot{y}_d(t) \quad \text{is a polynomial}$$

So that we have the open-loop controller

$$u_d(t) = ky_d(t) + d\dot{y}_d(t) + m\ddot{y}_d(t)$$

which will make the output follow the desired trajectory

Remark: the above reasoning also works for other functions, eg. sine/cosine, exponential = differentiable functions

Polynomial trajectories (1/2)

Let us see how to construct a polynomial trajectory (2nd order case).

we want to go from $\mathbf{x}(0) = \begin{bmatrix} x_{10} \\ x_{20} \end{bmatrix} =: \mathbf{x}_0$ to $\mathbf{x}(T) = \begin{bmatrix} x_1(T) \\ x_2(T) \end{bmatrix} = \begin{bmatrix} x_{1T} \\ x_{2T} \end{bmatrix} =: \mathbf{x}_T$

(x(t) is the state of a CCF)

in time T

a polynomial trajectory is described by

$$y_d(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3 + \dots$$

2 initial and 2 final conditions → we need 4 coef. = poly of order 3

we have

$$x_1(t) = y_d(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3 \quad (1)$$

and its derivative

$$x_2(t) = \dot{x}_1(t) = \dot{y}_d(t) = \alpha_1 + 2\alpha_2 t + 3\alpha_3 t^2 \quad (2)$$

at t=0, (1) gives $x_1(0) = y_d(0) = x_{10} = \alpha_0$

and (2) gives $x_2(0) = \dot{y}_d(0) = x_{20} = \dot{x}_1(0) = \alpha_1$

→ we have the first 2 coefficients!

Polynomial trajectories (2/2)

$$\begin{cases} x_1(t) = y_d(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3 & (1) \\ x_2(t) = \dot{x}_1(t) = \dot{y}_d(t) = \alpha_1 + 2\alpha_2 t + 3\alpha_3 t^2 & (2) \end{cases}$$

At time $t = T$

(1) and (2) give

$$\begin{cases} x_1(T) = y_d(T) = \alpha_0 + \alpha_1 T + \alpha_2 T^2 + \alpha_3 T^3 = x_{1T} \\ x_2(T) = \dot{y}_d(T) = \alpha_1 + 2\alpha_2 T + 3\alpha_3 T^2 = x_{2T} \end{cases}$$

separate known from unknown terms

$$\begin{cases} x_{1T} - (x_{10} + x_{20}T) = \alpha_2 T^2 + \alpha_3 T^3 \\ x_{2T} - x_{20} = 2\alpha_2 T + 3\alpha_3 T^2 \end{cases}$$

rewrite in vectorial form

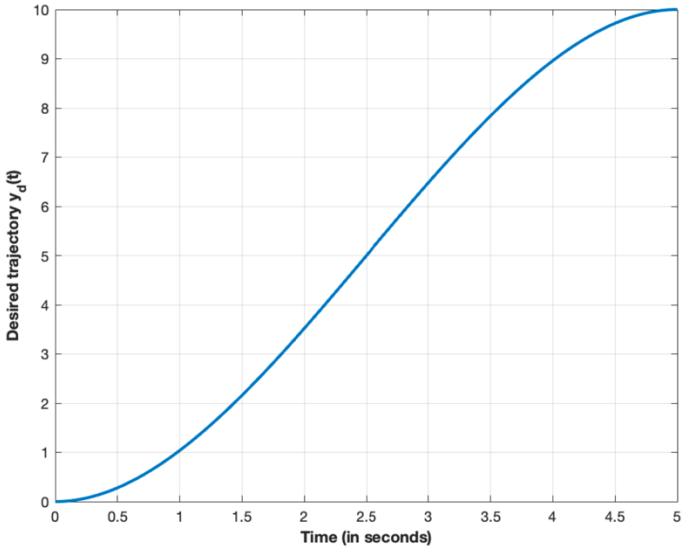
$$\begin{bmatrix} x_{1T} - (x_{10} + x_{20}T) \\ x_{2T} - x_{20} \end{bmatrix} = \begin{bmatrix} T^2 & T^3 \\ 2T & 3T^2 \end{bmatrix} \begin{bmatrix} \alpha_2 \\ \alpha_3 \end{bmatrix}$$

invert the time matrix to get the last two coefficients

$$\begin{bmatrix} \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} T^2 & T^3 \\ 2T & 3T^2 \end{bmatrix}^{-1} \begin{bmatrix} x_{1T} - (x_{10} + x_{20}T) \\ x_{2T} - x_{20} \end{bmatrix}$$

(implementation trick:)

$$\begin{bmatrix} x_{1T} - (x_{10} + x_{20}T) \\ (x_{2T} - x_{20})\,T \end{bmatrix} = \begin{bmatrix} T^2 & T^3 \\ 2T^2 & 3T^3 \end{bmatrix} \begin{bmatrix} \alpha_2 \\ \alpha_3 \end{bmatrix}$$



$$y_d(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3$$

example:

$$y_d(0) = 0 \text{ to } y_d(T) = 10$$
$$\text{in } T = 5 \text{ seconds}$$
$$\text{with } \dot{y}_d(0) = \dot{y}_d(T) = 0$$

SDU

Back to the MSD

Example: Previously (slide 9), we obtained the open-loop controller

$$u_d(t) = ky_d(t) + d\dot{y}_d(t) + m\ddot{y}_d(t)$$

since $y_d(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3$

we have $\dot{y}_d(t) = \alpha_1 + 2\alpha_2 t + 3\alpha_3 t^2$

and $\ddot{y}_d(t) = 2\alpha_2 + 6\alpha_3 t$

so that we have the polynomial expression

$$u_d(t) = k(\alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3) + d(\alpha_1 + 2\alpha_2 t + 3\alpha_3 t^2) + m(2\alpha_2 + 6\alpha_3 t)$$

Remark: do not implement things like this! This way of programming is too tedious ☺

State-coordinate transformation

Our OL algorithm works for a CCF. What about more general forms?

→ trick: transform a SS rep. back into a CCF to do OL control

Relation between 2 different SS rep.?

start with (1) $\begin{cases} \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u \\ y = \mathbf{C}\mathbf{x} \end{cases}$ and define the invertible change of coordinates $\mathbf{z}(t) = \mathbf{T}\mathbf{x}(t)$ with constant $\mathbf{T} \in \mathbb{R}^{n \times n}$

implies → $\mathbf{x} = \mathbf{T}^{-1}\mathbf{z}$ and $\dot{\mathbf{z}} = \mathbf{T}\dot{\mathbf{x}}$

so that (1) is transformed into $\begin{cases} \dot{\mathbf{z}} = \mathbf{T}(\mathbf{A}\mathbf{x} + \mathbf{B}u) \\ y = \mathbf{C}\mathbf{T}^{-1}\mathbf{z} \end{cases}$ → $\begin{cases} \dot{\mathbf{z}} = \mathbf{T}\mathbf{A}\mathbf{T}^{-1}\mathbf{z} + \mathbf{T}\mathbf{B}u \\ y = \mathbf{C}\mathbf{T}^{-1}\mathbf{z} \end{cases}$

→ $\begin{cases} \dot{\mathbf{z}} = \tilde{\mathbf{A}}\mathbf{z} + \tilde{\mathbf{B}}u \\ y = \tilde{\mathbf{C}}\mathbf{z} \end{cases}$

$\tilde{\mathbf{A}} = \mathbf{T}\mathbf{A}\mathbf{T}^{-1}$ $\quad \tilde{\mathbf{C}} = \mathbf{C}\mathbf{T}^{-1}$ $\quad \tilde{\mathbf{B}} = \mathbf{T}\mathbf{B}$

A second-order example

Example: start with ODE $\ddot{y} + 3\dot{y} = 2y + u$ (ODE)

a CCF of (ODE) is $\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 2 & -3 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$

Define a change of coordinate z=T x where $\mathbf{T} = \begin{bmatrix} 3 & 1 \\ 1 & 0 \end{bmatrix}$

so that we have $\dot{\mathbf{z}} = \tilde{\mathbf{A}}\mathbf{z} + \tilde{\mathbf{B}}u$ (SS rep. after transform, in the z-space)

with $\tilde{\mathbf{A}} = \mathbf{T}\mathbf{A}\mathbf{T}^{-1} = \begin{bmatrix} 3 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 2 & -3 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -3 \end{bmatrix} = \begin{bmatrix} 0 & 2 \\ 1 & -3 \end{bmatrix}$

and $\tilde{\mathbf{B}} = \mathbf{T}\mathbf{B} = \begin{bmatrix} 3 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

ie we have the new SS $\dot{\mathbf{z}} = \begin{bmatrix} 0 & 2 \\ 1 & -3 \end{bmatrix} \mathbf{z} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$ (not a CCF!)

rep.

Transforming a SS-rep. into a CCF (1/2)

Objective: we want to find the change of coordinates $\mathbf{z} = \mathbf{T}\mathbf{x}$

so that $\dot{\mathbf{z}} = \tilde{\mathbf{A}}\mathbf{z} + \tilde{\mathbf{B}}u$ is a CCF

Recall that

$$\tilde{\mathbf{A}} = \mathbf{T}\mathbf{A}\mathbf{T}^{-1} \iff \boxed{\tilde{\mathbf{A}}\mathbf{T} = \mathbf{T}\mathbf{A}} \quad \text{and} \quad \boxed{\tilde{\mathbf{B}} = \mathbf{T}\mathbf{B}}$$

Let

$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \end{bmatrix}$$

(we split T into n row vectors)

$$\begin{bmatrix} 0 & 1 & \dots & 0 \\ \vdots & 0 & \ddots & \vdots \\ 0 & \dots & 0 & 1 \\ -a_0 & -a_1 & \dots & -a_{n-1} \end{bmatrix} \begin{bmatrix} \mathbf{T}_1 \\ \mathbf{T}_2 \\ \vdots \\ \mathbf{T}_n \end{bmatrix} = \begin{bmatrix} \mathbf{T}_1\mathbf{A} \\ \mathbf{T}_2\mathbf{A} \\ \vdots \\ \mathbf{T}_n\mathbf{A} \end{bmatrix}$$

SDU

$$\begin{bmatrix} \mathbf{T}_2 \\ \mathbf{T}_3 \\ \vdots \\ -a_0\mathbf{T}_1 - a_1\mathbf{T}_2 - \dots - a_{n-1}\mathbf{T}_n \end{bmatrix} = \begin{bmatrix} \mathbf{T}_1\mathbf{A} \\ \mathbf{T}_2\mathbf{A} \\ \vdots \\ \mathbf{T}_n\mathbf{A} \end{bmatrix}$$

ie $\boxed{\begin{aligned} \mathbf{T}_2 &= \mathbf{T}_1\mathbf{A} \\ \mathbf{T}_3 &= \mathbf{T}_2\mathbf{A} \end{aligned}}$ etc...

if we have T_1, we have all the other T_i's!

Transforming a SS-rep. into a CCF (2/2)

Remaining question: we need to find $\mathbf{T}_1$

let us use $\tilde{\mathbf{B}} = \mathbf{T}\mathbf{B}$ with $\tilde{\mathbf{B}}^T = \begin{bmatrix} 0 & \dots & 0 & 1 \end{bmatrix}$

(CCF with b=1)

this gives

$$\tilde{\mathbf{B}} = \begin{bmatrix} \mathbf{T}_1\mathbf{B} \\ \mathbf{T}_2\mathbf{B} \\ \mathbf{T}_3\mathbf{B} \\ \vdots \\ \mathbf{T}_n\mathbf{B} \end{bmatrix} = \begin{bmatrix} \mathbf{T}_1\mathbf{B} \\ \mathbf{T}_1\mathbf{A}\mathbf{B} \\ \mathbf{T}_1\mathbf{A}^2\mathbf{B} \\ \vdots \\ \mathbf{T}_1\mathbf{A}^{n-1}\mathbf{B} \end{bmatrix}$$

transpose

$$\tilde{\mathbf{B}}^T = \begin{bmatrix} 0 & \dots & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{T}_1\mathbf{B} & \mathbf{T}_1\mathbf{A}\mathbf{B} & \mathbf{T}_1\mathbf{A}^2\mathbf{B} & \dots & \mathbf{T}_1\mathbf{A}^{n-1}\mathbf{B} \end{bmatrix}$$

so that we have $\tilde{\mathbf{B}}^T = \mathbf{T}_1\mathbf{W}_c$

inverting W_c, we get T_1 as $\mathbf{T}_1 = \tilde{\mathbf{B}}^T\mathbf{W}_c^{-1}$

Back to the second-order example

Example: start with $\dot{\mathbf{x}} = \begin{bmatrix} 0 & 2 \\ 1 & -3 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$ (not a CCF)

Transform this into a CCF?

compute the controllability matrix $\mathbf{W}_c = \begin{bmatrix} \mathbf{B} & \mathbf{AB} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

and set $\tilde{\mathbf{B}}^T = \begin{bmatrix} 0 & 1 \end{bmatrix}$

Calculate $\mathbf{T}_1 = \tilde{\mathbf{B}}^T \mathbf{W}_c^{-1} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \end{bmatrix}$

which gives $\mathbf{T}_2 = \mathbf{T}_1 \mathbf{A} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 2 \\ 1 & -3 \end{bmatrix} = \begin{bmatrix} 1 & -3 \end{bmatrix}$

so that we have the coordinate transform $\mathbf{T} = \begin{bmatrix} 0 & 1 \\ 1 & -3 \end{bmatrix}$

giving the new SS rep. $\dot{\mathbf{z}} = \begin{bmatrix} 0 & 1 \\ 2 & -3 \end{bmatrix} \mathbf{z} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$ (CCF)

Open-control for general linear systems

Algorithm - Open-Loop control for general LTI systems

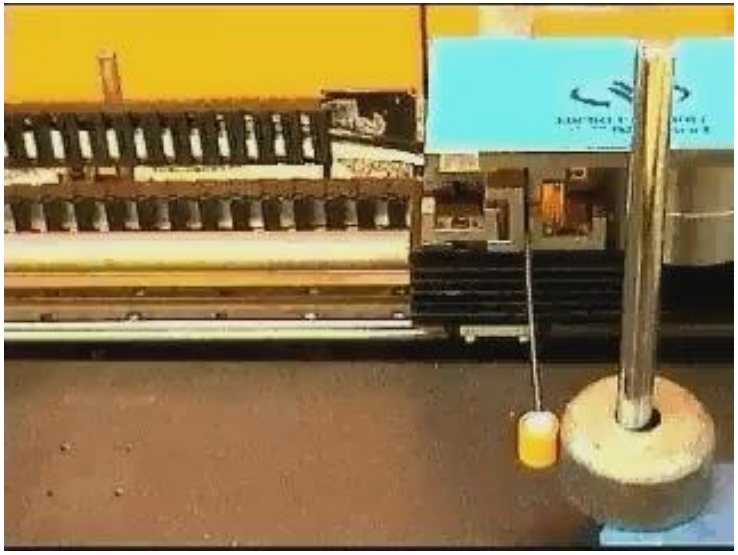
Given the following state-space representation

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u \\ y = \mathbf{C}\mathbf{x} \end{cases}$$

- After checking the controllability of the system, compute the transform $\mathbf{T}$ and obtain a CCF.
- We want the state $\mathbf{x}(t)$ to go from $\mathbf{x}_0$ to $\mathbf{x}_T$. Translate this in the z-space: we want the state $\mathbf{z}(t)$ to go from $\mathbf{z}_0 = \mathbf{T}\mathbf{x}_0$ to $\mathbf{z}_T = \mathbf{T}\mathbf{x}_T$.
- Construct polynomial trajectory $y_d(t)$ using $\mathbf{z}_0$ and $\mathbf{z}_T$ as boundary conditions (ie we want to have $y_d(0) = z_{1,0}$, $\dot{y}_d(0) = z_{2,0}$, ..., for the initial conditions and $y_d(T) = z_{1,T}$, $\dot{y}_d(T) = z_{2,T}$, ..., for the final conditions).
- Implement feedforward controller

$$u_d(t) = a_0 y_d(t) + a_1 \dot{y}_d(t) + \ldots + a_{n-1} y_d^{(n-1)}(t) + y_d^{(n)}(t)$$

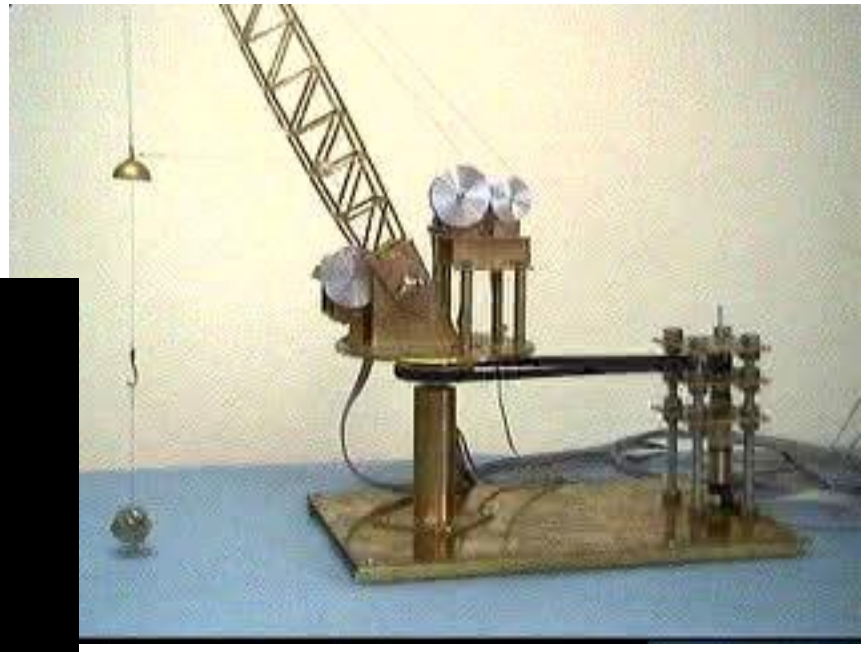
Application

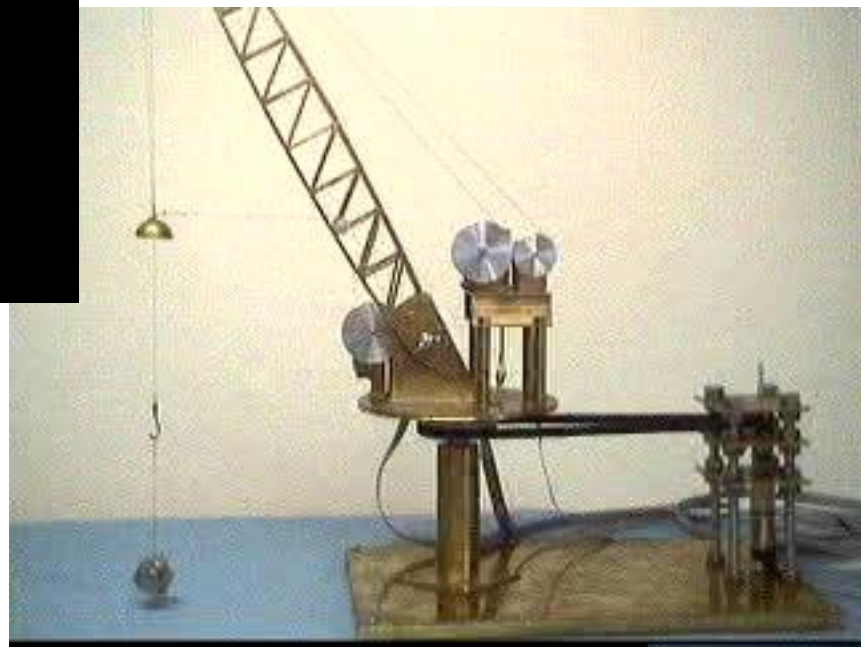


SDU

Beyond linear systems (1/2)

Beyond linear systems (2/2)





SDU