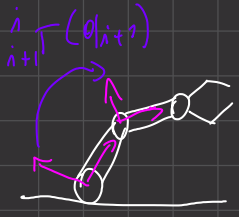
19/02/24

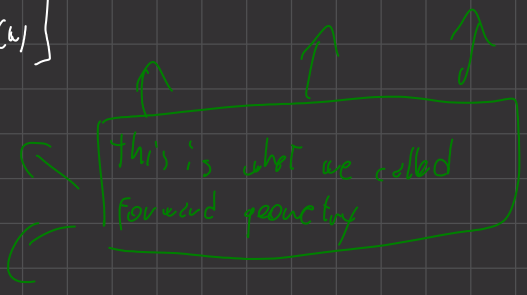
In this course we will mainly focus on manipulators



We saw that we could assign variables to joints

$$\underline{q}(t) = \begin{bmatrix} q_1(t) \\ \vdots \\ q_n(t) \end{bmatrix}$$

in order to control the robot.



There are two important entities:

$$\begin{cases} {}^{i}_{i}R\,(q_{i+1}) \\ r_{i+1/i}(q_{i+1}) \end{cases}$$

In order to control the robot we needed to use the inverse geometric model.

There is always a solution to the forward geometric model.
In the inverse problem we are in trouble if the robot is redundant.
We can have infinite many solutions.

We also have to keep into consideration what happens when time passes

Typically we have an end effector attached to the end-effector of the robot.

q variables change in time so I might be interested by the trajectories generated by the end effector.

So we have two steps:

- What happens if for a given configuration at a given time instant I assign a set of joint velocities:

$$\dot{q}(t) = \begin{bmatrix} \dot{q}_1(t) \\ \vdots \\ \dot{q}_n(t) \end{bmatrix}$$

If I can compute the velocities then I can compute the system velocity of the ee wrt any frame of interest.
We can also compute also the linear velocity

Forward kinematic problem

$$\begin{bmatrix} {}^{\phi}\omega_{e/\phi}(t) \\ {}^{\phi}v_{e/\phi}(t) \end{bmatrix} = {}^{\phi}J_{e/\phi}(q)\,\dot{q}(t)$$

input

output

linear transformation

Computing this solves the forward Kinematic Problem

N.B These are all function of time

Inverse Kinematic Problem

Here control comes directly into play.

Now we have to reverse the problem.

In this case I want to achieve a particular angular velocity at the EE.

$$X^* = \begin{bmatrix} \omega^*(t) \\ v^*(t) \end{bmatrix} = J(q)\,\dot{q}(t)$$

this is the forward model of the robot. Not software

the output is not explicitely expressed.

output

Input

this expression falls in this type of formulae: $Ax = y$

Now we have to follow rules depending on the shape of J

1) square → there can be one solution (excluding sing.)
2) flat (more columns than 6) → infinite solutions
3) Tall (less columns than 6) → You might not find the solution
   (is this is the typical case of mobile robots)

In the case of square problems we can compute:

> If square matrix = $\dot{q}(t) = J^{-1}x(t)$

> a flat matrix = We have more solution (the difference between rows and cols number)

the simplest way of solving this is using the least square solution.

$$\dot{q}(t) = \underbrace{J^T(J\,J^T)^{-1}}\,x^*$$ → Left pseudoinverse (this is software)

There are variations because the motors of some joints may not be as powerfull as required

This is a closed form formula.

The pseudoinverse has some properties: $J^{\#} J = I$

So for square and flat we can find only one solution.

> In tall configuration you have a best effort solution

$$\dot{q} = J^T (J J^T)^{-1} \dot{x}^*$$

In every case we still have problems with singularities.

In this case you get something close to the movement that you desire.

You go from J to $J^{\#}$ by using SVD

$$A = U \Sigma V^T$$

$$\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ 0 & & 0 \end{bmatrix}$$

$$\sigma_1 > \dots > \sigma_n > 0$$

CHI:

$$\chi = \frac{\sigma_1}{\sigma_n}$$

CONDITION NUMBER OF THE MATRIX

If χ is big then the system will be noisy and the result will be big. This effect gets bigger when you are close to singularities.

In practice you need to adjust the singular values in order to bound the results, and not have jerky motions.

The solution to this problem is to compute $J^{\#}$ like this:

$$J^{\#} = J^T (J J^T + \lambda I)^{-1}$$

λ must be added when $\det J$ is getting critical. λ Generates a wrong solution but keeps everything bounded.

This is called regularization.

The pseudoinverse of A (or J) is iteratively computed as:

$$A^{\#} = U \Sigma^{\#} V^T$$

Orthonormal matrices $\rightarrow$ $\begin{cases} U U^T = I_n \\ V^T V = I_m \end{cases}$

$$\Sigma^{\#} = \begin{bmatrix} \frac{1}{\sigma_1} & & & \\ & \ddots & & \\ & & \frac{1}{\sigma_n} & \\ & & & \emptyset \end{bmatrix}$$

When the critical singular values not critical you can play with lambda.

Every inverse should be done with the pseudoinverse

Inverse geometry problem

I want to find two values:

* means desired.

$^{\emptyset}_{e}R^{*}(t)$ $\quad$ $^{\emptyset}_{e}R(q)$

$\underline{r}^{*}_{e/\emptyset}$ $\quad$ $\underline{r}_{e/\emptyset}(q)$

(displacement is a vector)

$$\underline{e}_L(t) = \underline{r}_{e/\emptyset} - \underline{r}^{*}_{e/\emptyset}$$

linear error

$\underline{r}_{e/\emptyset}$ → actual position

$\underline{r}^{*}_{e/\emptyset}$ → desired position

misalignment between the two frames

$\rho = v\,\theta$

↳ angular error.

My control objective is to $\begin{cases} \rho \to \emptyset \\ e_L \to \emptyset \end{cases}$ Here we want to reduce the distance

N.B DISPLACEMENT (vector) ≠ DISTANCE (scalar)

$$\frac{1}{2}|e_L|^2 = \frac{1}{2} e_L \cdot e_L$$

↳ the derivative of this $\sqrt{\text{negative}}$ We want

$$\frac{1}{2}\frac{d}{dt}(e_L \cdot e_L) = \frac{1}{2} e_L \cdot \dot{e}_L + \frac{1}{2}\dot{e}_L \cdot e_L = e_L \cdot \dot{e}_L = e_L \cdot [v_{e/\emptyset} - v^{*}]$$

This is where I define my control structure.

this is the control signal

Note that e_L is given

(We want to choose $v_{e/\emptyset}$ so that the scalar product is negative)

This process is called control synthesis

We can express:

$$e_L \cdot [v_{e/\phi} - \underbrace{v^*}_{-\gamma e_L}] = \underbrace{-\gamma (e_L \cdot e_L)}_{<0}$$

If I choose this for the term inside the matrix this will be guaranteed

We want to guarantee that:

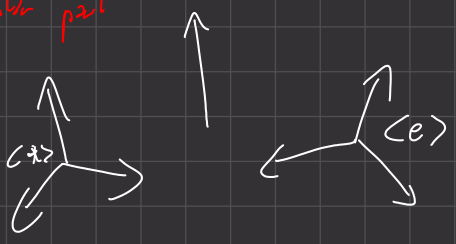
$$v_{e/\phi} - v^* = -\gamma e_L$$

output ← $v_{e/\phi}$; "input" ← v^* ; control parameter ← γ ; input ← e_L

$$v_{e/\phi} = -\gamma e_L + v^*$$

$[\dot{x}^*]$; feedback control term. ; feed forward term providing velocity correction to keep track of the goal.

Angular part



$$\frac{1}{2} |p \cdot p|^2 = \frac{1}{2} \theta^2$$

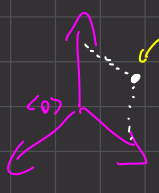
This because v is unit vector

$$\frac{d}{dt} \frac{1}{2} (p \cdot p) = \theta v \cdot (\underbrace{w_{e/\phi} - w^*}_{-\gamma_A v \theta})$$

I want this to be opposite of v because I want to turn the other way round.

$$w_{e/\phi} = -\gamma_A v \theta + w^*$$

21/02/24



$$r_{p/\phi} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

$$v_{p/\phi} = \begin{bmatrix} \phi_{1_1} & \phi_{J_1} & \phi_{K_1} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}$$

This is how we wrote this in MCM → $\begin{bmatrix} \phi_{n_1} & \phi_{n_2} & \phi_{n_3} \end{bmatrix}$

I want to move P toward P*

The course will be divided in two steps:

1) Review of fundamentals of mechanics
2) Computational Mechanics. (Newton Euler recursive equations)
3) Dynamic models of holonomic robots.

At the end the robot will be:

$$A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) = M + D$$

- $A(q)\ddot{q}$: inertia
- $B(q,\dot{q})\dot{q}$: coriolis and centripetal
- $C(q)$: gravity
- M: torques
- D: disturbances

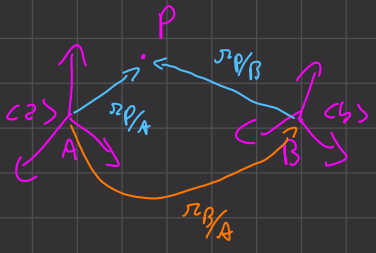
REMARK

$$\dot{x}^* = J^{\#}\dot{q}$$

4) Control Algorithms. / Fundamentals of Robot dynamics contd.
5) Less standard control algorithms. (Dynamic Motion Primitives)

$$\frac{d_2}{dt} u(t) = \frac{d_1}{dt} u(t) + \omega_{1/2} \times u$$

revision of the matter of all the formulas.



$$\frac{d_2}{dt}\left(r_{P/A}\right) = \frac{d_2}{dt}\left(r_{P/B} + r_{B/A}\right) = \underbrace{\frac{d_2}{dt} r_{B/A}}_{v_{B/A}} + \underbrace{\frac{d_1}{dt} r_{P/B}}_{v_{P/B}} + \omega_{B/A} \times r_{P/B}$$

$$\boxed{v_{P/A} = v_{B/A} + v_{P/B} + \omega_{B/A} \times r_{P/B}} \quad (7)$$

$$\frac{d_2}{d_0}\left(v_{P/A}\right) = \frac{d_2}{dt}\left[v_{B/A} + v_{P/B} + \omega_{B/A} \times r_{P/B}\right] =$$

$a_{P/A}$

linear operator

$$\rightarrow \frac{d_2}{dt} v_{B/A} + \frac{d_b}{dt} v_{P/B} + \omega_{B/A} \times v_{P/B} + \left(\frac{d_2}{dt}\omega_{B/A}\right) \times r_{P/B} + \omega_{B/A} \times \left(\frac{d_2}{dt} r_{P/B}\right) =$$

$$= a_{B/A} + a_{P/B} + \omega_{B/A} \times v_{P/B} + \left(\frac{d_2}{dt}\omega_{B/A}\right) \times r_{P/B} + \omega_{B/A} \times \left(v_{P/B} + \omega_{B/A} \times r_{P/B}\right) =$$

$$= a_{P/A} = a_{B/A} + a_{P/B} + 2\cdot(\omega_{B/A} \times v_{P/B}) + \left(\frac{d_2}{dt}\omega_{B/A}\right) \times r_{P/B} + \omega_{B/A} \times (\omega_{B/A} \times r_{P/B}) \quad (2)$$

Coriolis

centripetal accel.

Proof that $\omega_{A/B} = \omega_{B/A}$

$$\frac{d_2}{dt}\omega_{B/2} = \frac{d_b}{dt}\omega_{B/A} + \omega_{B/A} \times \omega_{B/A}$$

So we can always write $\omega_{A/B}$

Point Mass

Ideal mechanical entity characterized by scalar quantity which is the mass. ≥ 0

It's a point in euclidean sense

Isolated point

A point which is not interacting with any other point in the universe.

Force

Mechanical entity responsible the interaction between points mass

Inertial reference frames

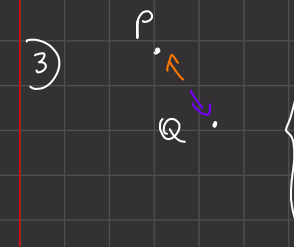
Reference frame which is not rotating wrt. the stars.

Newton Laws

1) An isolated point mass has constant velocity wrt. inertial reference frames.

2) Given a point mass $m \Rightarrow ma = F$

Inertial quantities

3) 

$$\begin{cases} F_{Q\to P} = -F_{P\to Q} \\ F_{Q\to P} \times F_{P\to Q} = \emptyset \end{cases}$$

$m\,\ddot{r}_{P/\emptyset} = F(r_{P/\emptyset}, \dot{r}_{P/\emptyset}, *(t))$

↳ can be seen as a kind of control action.

↳ is something in function of time.

1) Forward Dynamic Problem

Computing $r_{P/\emptyset}$ and $\dot{r}_{P/\emptyset}$ given $F(\cdot,\cdot,\cdot)$

$r_{P/\emptyset}, \dot{r}_{P/\emptyset}\Big|_{t=t_0}$

2) Inverse Dynamic Problem ("Control Problem")

I want the point to move with a given trajectory and I want to know the forces to move the point with that trajectory.

Given $r^*_{P/\emptyset}$, $\dot{r}^*_{P/\emptyset}$, $\ddot{r}^*_{P/\emptyset}$ compute F.

Compute the forward dynamic model:

Input:
- $\underline{F}$
- m

Parameter:
- h (sample time)

$\ddot{r}_{P/\emptyset} = \frac{1}{m} \cdot F$ ← This is valid for a given time instant.

initialization:

$r_{P/\emptyset}(1) = *$

$\dot{r}_{P/\emptyset}(1) = **$

for $i = 1:N$:

$\ddot{x}_{P/\emptyset}(t_{i+1}) = \frac{1}{m}\,\underline{F}(i+1)$ →

$\dot{r}_{P/\emptyset}(i+1) = \dot{r}_{P/\emptyset}(i) + h\,\ddot{r}_{P}(i+1)$

$r_{P/\emptyset}(i+1) = r_{P/\emptyset}(i) + h\,\dot{r}_{P/\emptyset}(i+1)$

end

$\dot{x} = f(x)$

$\frac{x(t+h) - x(t)}{h} = f(x(t))$

Explicit euler formula.

1) $F = -m * g.d * \left[zeros(2, N+1);\ 1:1:N+1 \right]$???

2) Function F = force ($r_{P/O}$)

EQUILIBRIUM (POINT)

$r^{E}_{P/O}$ = constant (w.r.t some inertial reference frame) $\forall t$

$\dot{r}^{E}_{P/O} \equiv 0$ The resultant forces are equal to 0.

$\ddot{r}_{P/O} \equiv 0$

Energy (scalar)

It's related to the capability of some system to generate work.

$E = T + V$

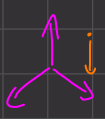
T: kinetic $\frac{1}{2}\left(\dot{r}_{P/O} \cdot \dot{r}_{P/O}\right) m$

V: Potential $F_c = \frac{\partial V}{\partial r}$

F_c: conservative (they do not dissipate energy)

Examples of conservative forces

1) Gravity



$F_g = -mg\hat{k}$

$V_g = mg(r_{P/O} \cdot \hat{k})$

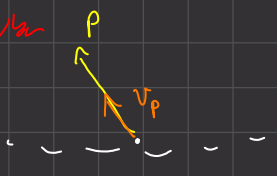
2) Elastic Force

$F_{Q \to P} = -K_E \underbrace{(Q-P)}_{r_{Q/P}}$ (Constant)

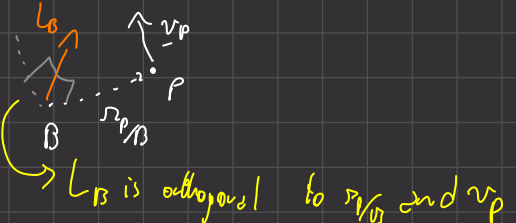
$V_E = \frac{1}{2} K_e \left(r_{Q/P} \cdot r_{Q/P}\right)$

LINEAR MOMENTUM and angular

Linear: $\underline{p} = m \cdot \underline{v}_P$

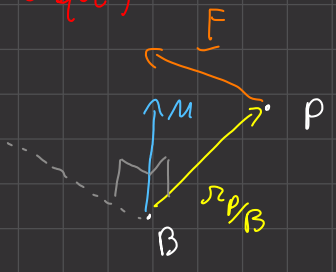


Angular: $\underline{L}_B = m\, r_{P/B} \times v_P$



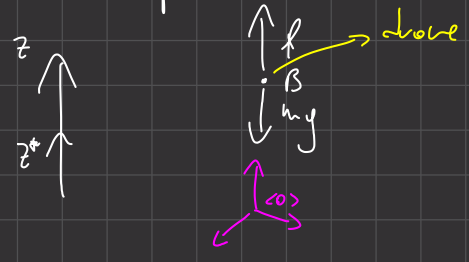
Moment of a force (Torque)

$\underline{M} = r_{P/B} \times F$



26/02/24

Another example



I build a model

$${}^{\phi} m\, \ddot{r}_{B/\phi} = {}^{\phi} F \quad \text{(result of external forces)}$$

$${}^{\phi} r_{B/\phi} = \begin{bmatrix} 0 \\ 0 \\ z \end{bmatrix} \qquad {}^{\phi} F = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix}$$

So I collapse my model in:

$$m \ddot{z} = F_z$$

Forces in this example:

- gravity: mg
- the propellers: f

Can I control the model?

$$\not{f} = \begin{bmatrix} 0 \\ 0 \\ \not{f}_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \not{f} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}$$

EXTERNAL FORCES: forces acting on the body

I want to drive my mass with $\not{f}$ in my desired position (z^*)

I look $m\ddot{z} = F_z$ and to change z I have to change $\ddot{z}$

How to change $\ddot{z}$? Changing F_z

F_z is computed by 2 terms $\not{f}$ and mg ($\begin{cases} m \text{ fixed} \\ g \text{ fixed} \end{cases}$)

So I should change $\not{f}$

Law: $\not{f} = K_{\not{f}}\, \omega$

$K_{\not{f}}$: constant

ω: Turning speed of the propellers

Propellers tend to push up the point, how to control ω?

$u(t) \rightarrow$ [motor + propeller] $\rightarrow \omega$

$u(t)$: input, what I control,
it's reasonable that $\omega(t) = K_\omega\, u(t)$

I rewrite the equation:

$$\ddot{z} = \frac{1}{m}\left[-mg + \not{f}\right] = \frac{1}{m}\left[-mg + (K_{\not{f}} + K_\omega)u\right] =$$

$$= \frac{1}{m}\left[-mg + K u\right]$$

So we are in this condition:

$u(t)$ → [S] → $z(t)$, with d entering S

The gravity for example is a constant disturbance

My control -> my control law : specification to implement, your software, typically digital

let's assume your system is in the desired position:

$$z(t) = z^*$$

Then we can define an error quantity:

$$e(t) = z(t) - z^* = \emptyset$$

this is what I want

So this means:

$$\dot{e}(t) = \dot{z}(t) - \dot{z}^* = \emptyset$$

$$\ddot{e}(t) = \ddot{z}(t) - \ddot{z}^* = \emptyset$$

This is valid under this assumption:

$$\ddot{z}(t) = \emptyset$$

So z^* is constant

We can claim:

$$\ddot{z} = \emptyset = \frac{1}{m}\left[-mg + K u\right]$$

$$\frac{1}{m} K u = g \Rightarrow \boxed{u_0(t) = \frac{1}{K} mg}$$ -> this is control

To know K -> dynamometer
m is another parameter, known with a scale.
g is known
u_0 is the control

What's wrong with this?

To maintain the height of my system my computer must generate the

Example:

$m = 0{,}1\,[kg]$

$K = 10^{-3}\,\frac{60}{2\pi}\left[\frac{N}{rad/s}\right]$

$g = 9{,}8\,\left[m/s^2\right]$

But generally the point is not in that z^*

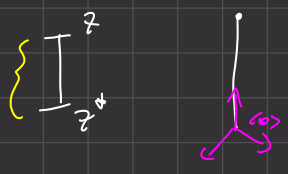
u_0 is still controllable.

By computing in this way this is an open loop signal. It's not good to cancel out disturbances.

That's due to the fact that $z(\varphi) \neq z^*$ so $u(t) = u_0(t) + \dots$

Not sufficient: this only cancels out gravity so the point is fixed.
But I want to control the height!

$z - z^*$
$e(t)$

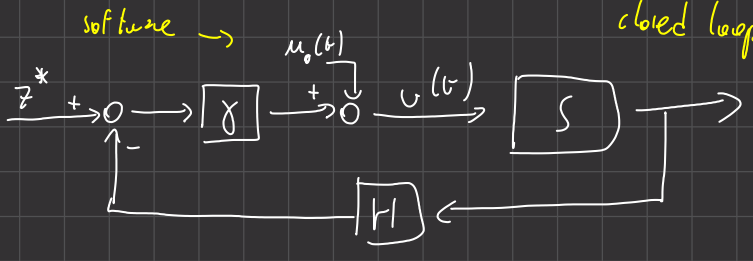


To slow down:

$$u_0(t) = K_W\left[v_0(t) - \gamma e(t)\right]$$
$$= K_W\, v(t)$$

If $e(t) > 0$ we are slowing down.

So what we do is this:

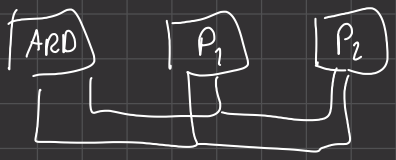


It's still not working because we are only analysing the physics

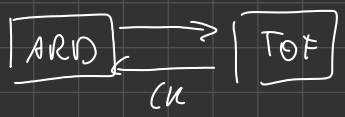
To measure what is the output of S you add the block H.

They are sensors. How to implement it?

BUS: I^2C



?

SPI 

?

For your software

```
for(;;)
    read_12c()
    e = z - z*
    u = u0 - γe   // this could be computed out of the loop to save counters
    write_i2c();
    <sync>
```

$e(t) = z - z^*$

$\dot{e}(t) = \dot{z} - \dot{z}^*$

$\ddot{e}(t) = \ddot{z} - \ddot{z}^* = \frac{1}{m}[-mg + u_{tot}] - \ddot{z}^*$ ← this is the error dynamics

How the control error is going with my action control signal!

I substitute what is u

$= \frac{1}{m}[\gamma e(t)] = -\frac{\gamma}{m} e(t)$

$f_0: \ddot{e}(t) + \frac{\gamma}{m} e(t) = 0$ 2° order linear equation

eq point.

Is this eq. point esymptotically stable?

I need to compute the poles associated with the system:

$\lambda^2 + \frac{\gamma}{m} = 0$ strictly negative real parts of poles is required to be esym. stable eq. point.

solutions:

$\lambda_{1,2} = \pm j\sqrt{\frac{\gamma}{m}}$ This is not the case

It means if I apply this control law there is no way to reach undefinetly the pool.

We need to dissipate energy: add another term to check when it arrive at the target. I have the robot velocity:

So: $u(t) = u_0(t) - \gamma e(t) - \alpha \dot{e}(t)$ → dumping term

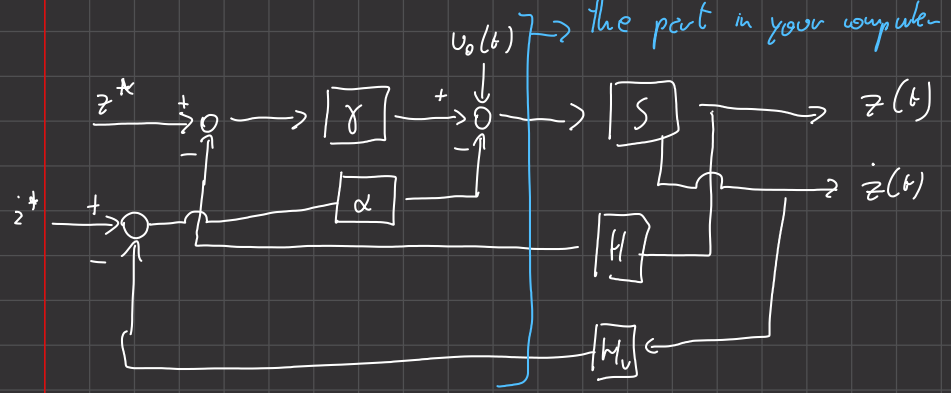
they are not real forces, the physics finishes with $u_0(t)$

Now we are considering control forces, they are virtual, but they have physical effects.

The sense of $-\alpha e(t)$ is to decrease $e(t)$ when you are moving at the goal

But we have to estimate $\dot{e}(t)$: the relative velocity.

You end up with this algorithm:



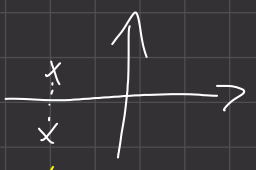
We add a control of velocity

So I find: $\ddot{e}(t) + \frac{\alpha}{m}\dot{e}(t) + \frac{\gamma}{m}e(t) = 0$

↳ closed loop error dynamics

$\lambda^2 + \frac{\alpha}{m}\lambda + \frac{\gamma}{m} = 0$

α and $\gamma \geq 0$ } permanent of signs } cartesio theorem



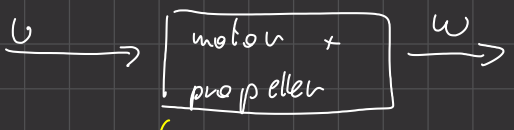
Strictly negative poles.

$\Rightarrow e(t)$ is an asym. stable equilibrium point. ✓

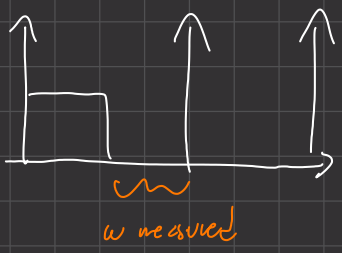
You must find a trade off between α and γ, because if one is greater than the other you can have oscillation or slowed down behaviours.

You should make experiments

$\begin{cases} \gamma \gg \alpha & \text{oscillations} \\ \alpha \gg \gamma & \text{slow motion} \end{cases}$



↳ Inside here there is closed loop control



Sensor-less feedback control

$w_{measured}$ is a voltage and it is used in the back.

For us v is a digital number.

28/02/24

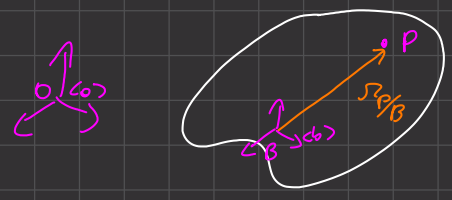
Study motion caused by external events. We need models to develop software.

model → physical property of model

model → control → software → simulation → reality

A robot is a connection of bodies

↳ In this course only rigid bodies



We consider a point B and its reference frame $\langle B \rangle$ equal for all points because of rigid body.

Rigid Body constraint

$$|P - B| = \text{constant} \quad \forall t$$

using our convention $|\Omega_{P/B}| = \text{constant} \quad \forall t$

We have some considerations derived from mother of formulas.

$$v_{P/\phi} = v_{B/\phi} + \omega_{\omega/\phi} \times \Omega_{P/B}$$

(because $v_{P/B} = \phi$ for rigid body constraint)

the velocity of the body is the sum of velocities of B wrt. ϕ and cross product between angular velocity of frame $\langle \omega \rangle$ wrt. $\langle \omega \rangle$ and the displacement between P and B.

Also for the acceleration

$$a_{P/O} = a_{B/O} + \omega_{b/\phi} \times r_{P/B} + \omega_{B/\phi} \times (\omega_{b/\phi} \times r_{P/B})$$

these expressions hold for any point of the body and for any point which is fixed w.r.t. the body.

Center of Mass

The body is made of stuff connected to each other. If you cut a part of the body -> it has a mass.

We can assume to have $\rho = \rho(p)$ TODO

(ρ : density?)

$$dm = \rho \, dV$$

ρ is a positive function

If I sum all the small parts I have the mass:

$$m = \int_V \rho(p) \, dV \qquad \text{TODO}$$

$$(C - O) = \frac{\int_V (P-O) \rho \, dV}{\int_V \rho \, dV} = \frac{\int_V (P-O) \rho \, dV}{m}$$

Formula for center of mass

The numerator is a sum of infinite vectors, the sum is weighted by the density.

In our notation:

$$r_{C/\phi} = \frac{\int_V r_{P/\phi} \, \rho \, dV}{m}$$

point ϕ is an arbitrary point ?

TODO

Properties:

1) $r_{C/\phi} = \dfrac{\int_V (r_{P/B} + r_{B/\phi}) \, \rho \, dV}{m}$ ($r_{P/\phi} = r_{P/B} + r_{B/\phi}$)

$$= \frac{\int_V r_{B/B}\,\rho\,dV}{m} + \frac{\int_V r_{B/\phi}\,\rho\,dV}{m}$$

(not depending on ρ so out)

TODO

$$= \frac{\int_V r_{B/B}\,\varphi\,dV}{m} + \frac{r_{B/\phi}\int_V \varphi\,dV}{m}$$

$$r_{C/\phi} = r_{C/B} + r_{B/\phi}$$

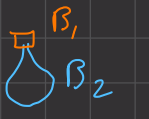
We can express the center of mass wrt. any inertial reference frame (or) by a translation ($r_{B/\phi}$)

2) the body is the union of 2 distinct bodies.

Each one will have its own center of mass

$$B = B_1 \cup B_2$$

Example of a bottle



$$B_1 \cap B_2 = \phi$$

So $$r_{C/\phi} = \frac{\int_V r_{P/\phi}\,\rho\,dV}{m}$$

$$= \frac{\int_{V_1} r_{P/\phi}\,\rho\,dV + \int_{V_2} r_{P/\phi}\,\rho\,dV}{m}$$

$$= \frac{m_1\, r_{C_1/\phi}}{m} + \frac{m_2\, r_{C_2/\phi}}{m}$$

Total center of mass is a linear combination of 2 center of masses

$$m\, r_{C/\phi} = m_1\, r_{C_1/\phi} + m_2\, r_{C_2/\phi}$$

$m \downarrow (m_1+m_2)$

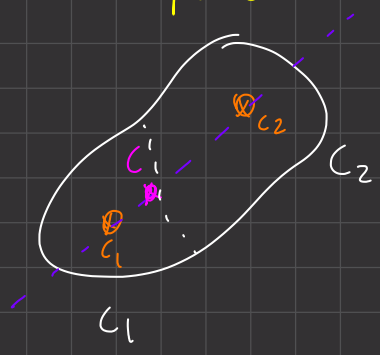
$$(m_1\, r_{C/\phi} - m_1\, r_{C_1/\phi}) + (m_2\, r_{C/\phi} - m_2\, r_{C_2/\phi}) = \phi$$

$$m_1\, r_{C/C_1} + m_2\, r_{C/C_2} = \phi$$

m_1 and m_2 are positive scalars and this means r_{C/C_1} and r_{C/C_2} must be parallel.

- So the center of mass lies on the line joining both both center of masses and it's closer to the bigger mass

On the picture:



So we summarize:

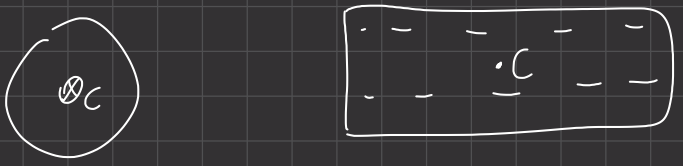
$$r_{C/\phi} = \frac{m_1 r_{C_1/\phi} + m_2 r_{C_2/\phi}}{m}$$

and $r_{C/C_1} \parallel r_{C/C_2}$

We can generalize this formula to n bodies:

$$r_{C/\phi} = \frac{1}{m} \sum_{i=1}^{n} m_i r_{C_i/\phi}$$

REMARK



⊗ C is not a physical point in the body.

⊗ C is the point where we consider all the mass to be concentrated.

For point mass:

$$p = m v_{P/\phi}$$

For linear movement:

$$L_B = m r_{P/B} \times v_{P/\phi}$$

I want to define these quantities for the whole body:

$$\begin{cases} dp = v_{P/\phi} \, dm \\ dL_B = (r_{P/B} \times v_{P/\phi}) \, dm \end{cases}$$

$$L_B = \int_V dL_B = \int (r_{P/B} \times v_{P/\phi}) \rho \, dv$$

$$= \int_V r_{P/B} \times \left[v_{B/\mathcal{O}} + (\omega_{b/\mathcal{O}} \times r_{P/B}) \right] \rho \, dV$$

$$= \int_V (r_{P/B} \times v_{B/\mathcal{O}}) \rho \, dV + \int_V r_{P/B} \times (\omega_{b/\mathcal{O}} \times r_{P/B}) \rho \, dV$$

$$= m (r_{C/B} \times v_{B/\mathcal{O}}) + I_B(\omega_{b/\mathcal{O}})$$

It's a vector depending on the position of the center of mass w.r.t. B, which is an arbitrary point, and the velocity of point w.r.t. $\mathcal{O}$

B is arbitrary so I can choose it as C so $r_{C/C} = \emptyset$

Inertia operator

$$I_B(\omega_{b/\mathcal{O}}) \triangleq \int_V r_{P/B} \times (\omega_{b/\mathcal{O}} \times r_{P/B}) \rho \, dV$$

It's a vector, precisely an operator
↳ transforms input into output

In this case: sum of infinite vectors derived from that operation

the input is $\omega_{b/\mathcal{O}}$, so I can see it as:

$$\omega_{b/\mathcal{O}} \longrightarrow \boxed{} \longrightarrow I_B(\cdot)$$

So I can write:

$$I_B(v) = \int_V r_{P/B} \times (\mu \times r_{P/B}) \rho \, dV$$

↳ geometric vector

properties

1) The definition has arbitrary choice of B

$$I_B(\mu) = I_C(\mu) + m \, r_{C/B} \times (\mu \times r_{C/B})$$

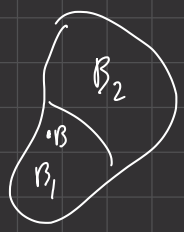
C: center of mass

You condense the formula of inertia operator and the second term is considered as if the whole mass is concentrated in a single point, which is the center of mass.

↳ the proof is in the notes

$$r_{P/B} = r_{P/C} + r_{C/B}$$

2)



$$B = B_1 + B_2$$

$$B_1 \cap B_2 = \emptyset$$

We choose point B and we can write:

$$I_B(u) = I_B^{(1)}(u) + I_B^{(2)}(u)$$

We notice the inertia operation is additive

3) $u \cdot I_B(u) \geq 0 \quad \forall u \neq 0 \quad \forall_B$ TODO

↑ ?

This holds for physical objects

↳ a line is not
↳ a paper sheet depends

there might be some vectors for which the scalar mod is very small, so similar to zero.

DEMONSTRATION

$$u \cdot I_B(u) = u \cdot \int_V r_{P/B} \times (u \times r_{P/B}) \rho \, dV$$

$$= \int_V u \left[r_{P/B} \times (u \times r_{P/B}) \right] \rho \, dV =$$

NOTE

$$a \cdot (b \times c) = c \cdot (a \times b) = b \cdot (c \times a)$$

$$= \int_V (u \times r_{P/B}) \cdot (u \times r_{P/B}) \rho \, dV$$

we have scalar prod of the same 2 vectors

$$= \int_V |u \times r_{P/B}|^2 \rho \, dV \geq 0$$

squared mod so non negative quantity

It's $= 0$ when u is aligned with $(P-B)$ but notice.

- u is arbitrary
- we are considering not pathological objects. For those reasons > 0 holds

How do I compute those things?

Algebric form of "I_B"

$I_B(\mu)$ is a geometric vector

I know it's an integral, integral = sum of bits.

$$ {}^{\phi} I_B(\mu) = \int_V {}^{\phi}\left[r_{P/B} \times (\mu \times r_{P/B}) \rho \, dV \right] $$

↳ Project the operation on a frame for sake of simpl, coz.

$$ = \int_V \left[{}^{\phi} r_{P/B} \times \right] \left\{ -\left[r_{P/B} \times \right] {}^{\phi}\mu \right\} \rho \, dV = $$

NOTE

$$ {}^{\phi} a = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad {}^{\phi}[a\times] = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} $$

μ does not depend on ρ so out of the integral

the $-$ goes at the beginning and

$$ -\left[r_{P/B} \times \right] = \left[r_{P/B} \times \right]^T $$

$$ = \left[\int_V {}^{\phi}\left[r_{P/B} \times \right]^T {}^{\phi}\left[r_{P/B} \times \right] \rho \, dV \right] {}^{\phi}\mu $$

this is a 3x3 frame expressed in frame ϕ

$$ {}^{\phi} I_B \in \mathbb{R}^{3\times 3} $$

So:

$$ {}^{\phi} I_B = \int_V {}^{\phi}\left[r_{P/B} \times \right]^T {}^{\phi}\left[r_{P/B} \times \right] \rho \, dV \in \mathbb{R}^{3\times 3} $$

and $ {}^{\phi} I_B = {}^{\phi} I_B^T $

This matrix changes depending on the position of the body.

↳ It's time varying in general

But I can choose a frame $\mathcal{B}$ fixed with body:

$$ {}^{b} I_b = \int_V \left[{}^{b} r_{P/B} \times \right]^T \left[r_{P/B} \times \right] \rho \, dV $$

if I consider $\{b\}$ fixed with body the matrix does not change because $r_{P/B}$ is constant.

So ${}^{b}I_B$ is constant.

How to link ${}^{\phi}I_B$ and ${}^{b}I_B$? Rotation Matrix

$$ {}^{\phi}I_B = {}^{\phi}_{b}R \, {}^{b}I_B \, {}^{b}_{\phi}R $$

You will never compute the integral but the important thing is

Property 3 implies:

$$ u \cdot I_B(u) > 0 \iff {}^{b}u^T \, {}^{b}I_B \, {}^{b}u > 0 $$
$$ \forall u \qquad\qquad \forall {}^{b}u $$

${}^{b}I_b$ is strictly positive defined matrix. It means:

${}^{b}I_B = {}^{b}I_B^T \rightarrow$ symmetric

${}^{b}I_B$ has 3 real positive eigenvalues

${}^{b}I_B$ can be diagonalized: $\exists V \in \mathbb{R}^{3\times 3}$ s.t. $V^T \, {}^{b}I_B \, V = \mathrm{diag}(I_1, I_2, I_3)$ (Eigenvalues)

$V V^T = \mathbb{1}$ (exemp. $V \in \mathbb{R}$)

the eigenvalues: they describe principle directions and how the mass is distributed along the axis.

the smaller ε the more the mass is not distributed.

TODO???

Principle axis of inertia.

Note: $\forall Q = Q^T \geq 0$

We have the property $\rightarrow$ symmetry
$\Rightarrow$ strictly real positive eigenvalues $v_1 \dots v_m$
$\Rightarrow$ $v_1, \dots v_m$ are orthonormal vectors.

$$ V = [v_1 \dots v_m] \in \mathbb{R}^{n\times n} $$

this V diagonalizes Q, that means:

$$V^T Q V = \mathrm{diag}\begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{bmatrix}$$

Definition of eigenvalue

the number which solves the equation

$$A v = \lambda v$$

v: eigenvector; λ: eigenvalue

So ${}^b I_B \in \mathbb{R}^{3\times 3}$ and ${}^b I_B = I_B^T$ $\Rightarrow$ $\lambda_1 = I_1 > 0$, $\lambda_2 = I_2 > 0$, $\lambda_3 = I_3 > 0$

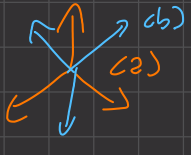
They are called principle inertia moments and are a property of the body.

$$V = [v_1, v_2, v_3] \in \mathbb{R}^{3\times 3}$$

$$V V^T = \mathbb{1}_{3\times 3}$$

There exists a set of axis (e) for which I have an orthogonal matrix.

graphically:



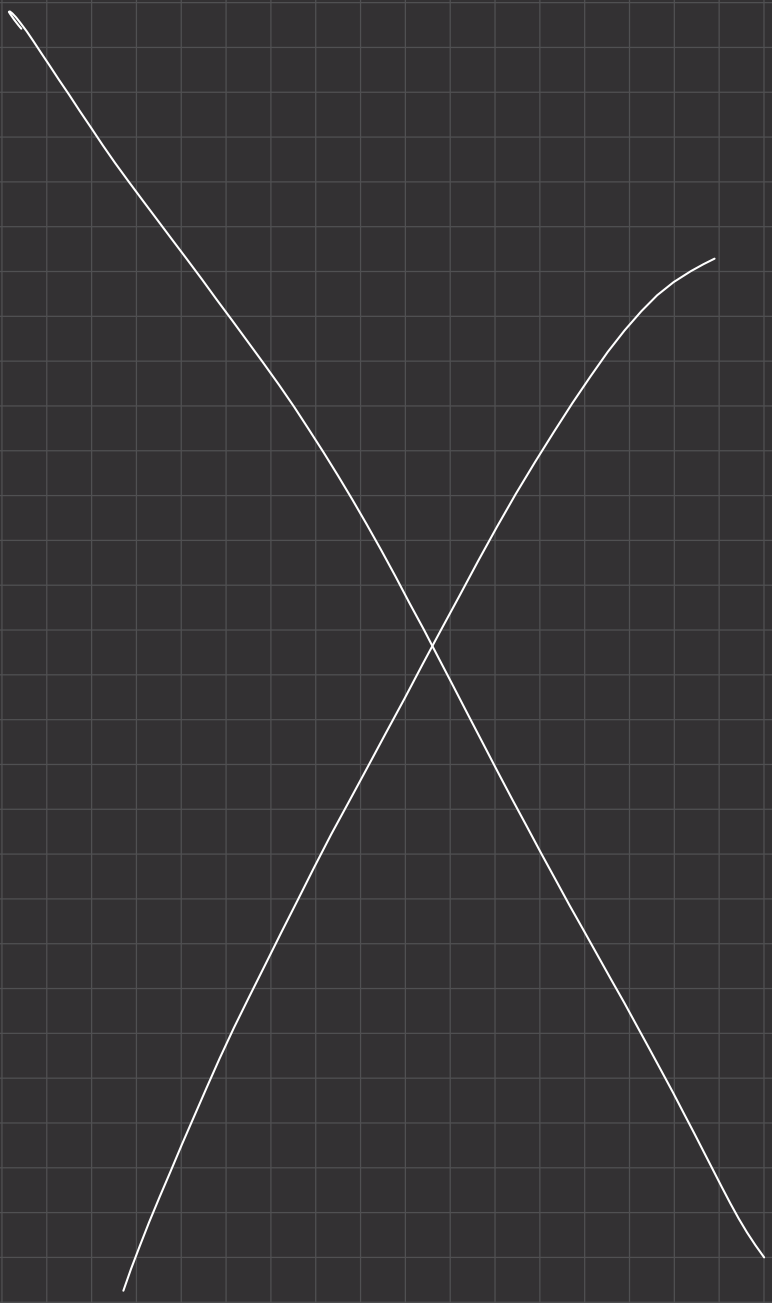
these are the principle inertia direction.

Note:

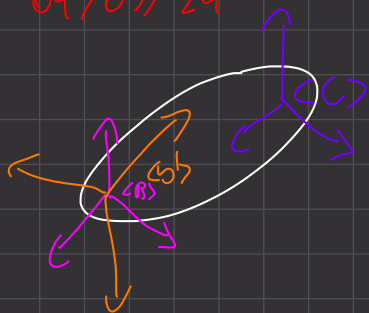
$$I_B(m) = I_C(m) + m\, r_{C/B} \times (m \times r_{C/B})$$

$$\boxed{{}^b I_B = {}^b I_C + m\left[{}^b r_{C/B} \times\right]^T \left[{}^b r_{C/B} \times\right]}$$

stessa cosa delle precedenti.



04/03/24



If the inertial reference frame is fixed and the frame of the body then the inertial operator is constant

$I_B(\cdot) \rightarrow {}^b I_B$ -> 3x3 matrix and strictly positive, and symmetric

$$\begin{bmatrix} I_{11} & I_{12} & I_{13} \\ I_{12} & I_{22} & I_{23} \\ I_{13} & I_{23} & I_{33} \end{bmatrix} \geq 0$$

Positive definite.

the way think about these come out the c.o.d. and they define the inertial

this has 3 eigenvalues and all strictly positive.

There exists a transformation orthonormal matrix that turns from the inertial into diagonal. The terms on the diagonal are the eigenvalues.

Every strictly positive matrix can be diagonalized ???

there exists as many eigenvectors as real eigenvalues.

$${}^{b}I_B \, v_i = \lambda_i \, v_i \qquad i = 1,3$$

$$\left[{}^{b}I_B v_1 \,;\, {}^{b}I_B v_2 \,;\, {}^{b}I_B v_3 \right] = \left[v_1 \lambda_1 \,;\, v_2 \lambda_2 \,;\, v_3 \lambda_3 \right]$$

$${}^{b}I_B \underbrace{\left[{}^{b}v_1 \,;\, {}^{b}v_2 \,;\, {}^{b}v_3 \right]}_{V} = \underbrace{\left[{}^{b}v_1 \,;\, {}^{b}v_2 \,;\, {}^{b}v_3 \right]}_{V} \underbrace{\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}}_{\Delta}$$

M.R

$$v_i^T \cdot v_j = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \triangleq \delta_{ij}$$

$${}^{b}I_B \, V = V \Delta$$

↑ orthonormal basis (matrix)

($V^T V = \mathbb{1}$)

(I multiply for V^T)

$$\left(\begin{bmatrix} v_1^T \\ v_2^T \\ v_3^T \end{bmatrix} \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} = \mathbb{1} \right)$$

$$V^T \, {}^{b}I_B \, V = \Delta$$

There is a linear transformation to transform ${}^{b}I_B$ into diagonal form.

V is an orthonormal matrix and can be seen in our case as a change of coordinates (change of space)

We changed $\langle B \rangle$ to $\langle b' \rangle$ but is still fixed in frame B.

$$V^T \, {}^{b}I_B \, V = {}^{b'}I_B(\Delta)$$

To compute V I use SVD.

$V = {}^{b}_{b'}R$ → You want this to be right hand so $\det(V) = +1$

If you compute V and $\det \neq 1$ just swap two eigenvectors

We can now give physical meaning to eigenvalues

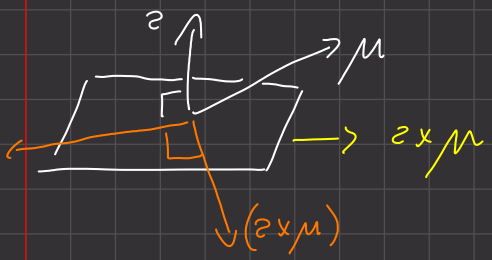
$$\Delta = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} = \begin{bmatrix} {}^b I_{11} & 0 & 0 \\ 0 & {}^b I_{22} & 0 \\ 0 & 0 & {}^b I_{33} \end{bmatrix}$$

↳ 3 moments of inertia.

$$[\,{}^b I_B = {}^b I_C + m[\,{}^b r_{C/b} \times]^T [\,{}^b r_{C/b} \times]$$

↳ strictly positive definite ↳ semipositive definite

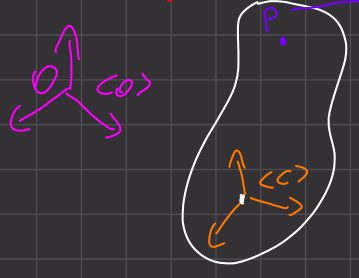
→ real 2 in general because cross product part



$$[z\times]^T [z\times]\,\mu$$

↳ b belongs to the plane and are velocity $-(+)$ 90° on that plane.

Newton equations



P will move following newton second law:

$$\frac{d}{dt} v_{P/O} \cdot dm = f_P^{Ext} + f_P^{INT}$$

↳ force acting on P due to external forces.

↳ resultant of all the forces which influence the considered by all the other points of the body acting on P

$dm = \rho\, dv$ (density, volume)

To explain this think of two balls connected by a spring and then put the balls in a piece of plastic

The internal force would be the spring and the plastic

Let's apply this law to all points of the body. So lets do an integral:

$$\int_V \frac{d\rho}{dt} v_{p/\phi}\, \rho\, dV = \int_V \not{F}_p^{ext}\, dV + \int_V \not{F}_p^{int}\, dV$$

→ the integral is always equal to 0. It's a function for not rigid bodies.

$v_{c/\phi} + \omega_{b/\phi} \times r_{p/c}$

EXT

F

resultant of all external forces.

All this must be compensated otherwise the object would explode

$$\frac{d}{dt} v_{c/\phi} + \omega_{b/\phi} \times r_{p/c} = a_{c/\phi} + \dot{\omega}_{b/\phi} \times r_{p/c} + \omega_{b/\phi} \times \left[\cancel{\frac{d\rho}{dt} r_{p/c}} \times \omega_{b/\phi} \times r_{p/c} \right]$$

$$\int_V a_{c/\phi}\, \rho\, dV + \int_V \dot{\omega}_{b/\phi} \times r_{p/c}\, \rho\, dV + \int_V \omega_{b/\phi} \times (\omega_{b/\phi} \times r_{p/c})\, \rho\, dV$$

$$m\, a_{c/\phi} + \dot{\omega}_{b/\phi} \times \underbrace{\int_V r_{p/c}\, \rho\, dV}_{m\, r_{c/c} = 0} + \omega_{b/\phi} \times \left(\omega_{b/\phi} \times \underbrace{\int_V r_{p/c}\, \rho\, dV}_{m\, r_{c/c} = 0} \right)$$

$$m\, a_{c/\phi} = F^{ext}$$

Formulation of the third motion laws for fundamental forces. Forces on the body act as if they act on the center of mass.

$$m\, {}^{\phi}a_{c/\phi} = {}^{\phi}\underline{F}^{EXT}$$

$$m \frac{d}{dt}{}^{\phi} v_{c/\phi} = {}^{\phi}F_{EXT} \quad \Rightarrow \quad X = \begin{bmatrix} {}^{\phi}r_{c/\phi} \\ {}^{\phi}v_{c/\phi} \end{bmatrix}$$

$$\frac{d\, {}^{\phi}r_{c/\phi}}{dt}$$

This extends $V_{C/\phi}$

$$\frac{d}{dt}x(t) = \dot{x}(t) = \begin{bmatrix} \phi_3 & \Pi_3 \\ \phi_3 & \phi_2 \end{bmatrix} x(t) + \begin{bmatrix} \phi_3 \\ \frac{1}{m} \Pi_3 \end{bmatrix} u(t)$$

note: $\dot{x}(t) = Ax + Bu$

Remark #1

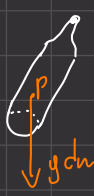
If $\begin{cases} F^{EXT} = \phi \quad \forall t \\ \phi V_{C/\phi}(t_0) = \phi \\ \phi \Pi_{C/\phi}(t_0) = \phi_{\Omega} \phi \end{cases}$

What is the solution of the previous formulas.

$\Rightarrow \begin{cases} V_{C/\phi}(t) = \phi \\ \Pi_{C/\phi}(t) = \Omega_{\phi} \end{cases} \forall t$

↳ Equilibrium condition.

Example:



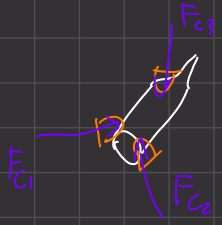
Bottle of water

Each point is feeling gravity: $(g\,dm)$

$$\int_V f_P^{EXT}\,dV = \int_V g\,dm = \int_V g\,\rho\,dV = g\int_V \rho\,dV = gm$$

Gravity acts as a pure force on the center of mass.

I Know that on the bottle is acting one external force due to gravity and then I decide to grasp it



Let's assume that I can generate three forces

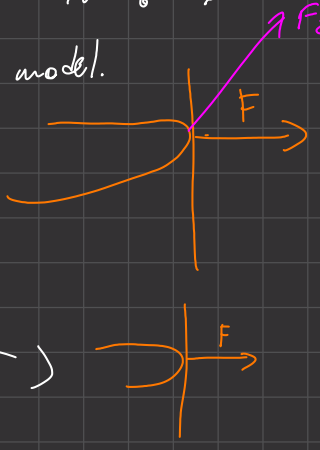
$$F_G = f_{C1} + f_{C2} + f_{C3}$$

↓ grasping force

To keep the bottle still I must guarantee that $F_{n1} + F_{f} = 0$

for these kind of forces you need a contact model.

- Point contact
  - w/ friction
  - w/o friction.

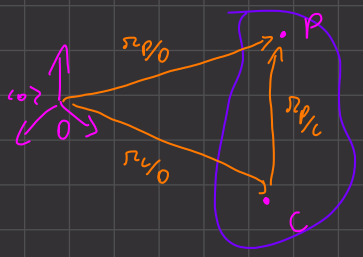


So if finger contact

there is also some torque that the finger applies to the object not allowing rotation

Euler Equations 06/03/24

they describe the motion of the robot.



$$\frac{d}{dt} v_{P/\emptyset}\, dm = f_P^{EXT} + f_P^{INT} \quad \text{and} \quad m_C = r_{P/C} \times f$$

So the moment is:

$$m_C = r_{P/C} \times \frac{d}{dt} v_{P/\emptyset}\, dm$$

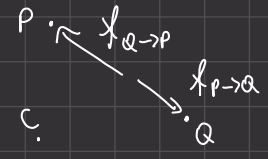
$$= \underline{v}_{P/C} \times f_P^{EXT} + r_{P/C} \times f_P^{INT}$$

$$= m_{P/C}^{EXT} + m_{P/C}^{INT}$$

To have the moment for the whole body while keeping generality, we integrate:

$$\int_V \left(r_{P/C} \times \frac{d}{dt} v_{P/\emptyset} \right) \rho\, dV = \int_V m_{P/C}^{EXT}\, dV + \int_V m_{P/C}^{INT}\, dV$$

NOTE:



$$(P-C) \times f_{Q\to P} + (Q-C) \times f_{P\to Q} = [(P-C) - (Q-C)] \times f_{Q\to P} = (P-Q) \times f_{Q\to P} = 0$$

(newton 3 law) (newton 3 law)

Valid for every $f_{q\to p}$, so for every point of a rigid body

$$\frac{d_0}{dt}\left(r_{P/C} \times v_{P/O}\right) = \frac{d_0}{dt} r_{P/C} \times v_{P/O} + r_{P/C} \times \frac{d_0}{dt} v_{P/O}$$

$$\left(r_{P/C} \times \frac{d_0}{dt} v_{P/\phi}\right) = \frac{d_0}{dt}\left(r_{P/C} \times v_{P/O}\right) - \frac{d_0}{dt}\left[r_{P/\phi} - r_{C/\phi}\right] \times v_{P/\phi}$$

$$\left(v_{P/\phi} - v_{C/O}\right) \times v_{P/O}$$

$$= \frac{d_0}{dt}\left(r_{P/C} \times v_{P/\phi}\right) + v_{C/\phi} \times v_{P/O}$$

$$v_{P/C} + \omega_{B/\phi} \times r_{P/C}$$

$\phi \rightarrow$ Not moving

$$= \frac{d_0}{dt}\left[r_{P/C} \times \left(v_{C/\phi} + \omega_{B/\phi} \times r_{P/C}\right)\right] + v_{C/\phi} \times \left(\omega_{B/\phi} \times r_{P/C}\right)$$

$$\int_V \left(r_{P/C} \times \frac{d_0}{dt} v_{P/\phi}\right) \rho \, dV = \int_V \frac{d_0}{dt}\left[*\right] \rho \, dV + \int_V v_{e/O} \times \left(\omega_{B/\phi} \times r_{P/C}\right) \rho \, dV$$

$$= \frac{d_0}{dt} \int_V r_{P/C} \times \left[v_{C/O} + \omega_{B/\phi} \times r_{P/C}\right] \rho \, dV + v_{C/\phi} \times \left[\omega_{B/\phi} \times \int_V r_{P/C} \rho \, dV\right]$$

$m\, r_{C/} \leftarrow$ from def of cent of mass and so always $= \phi$

$$= \frac{d_0}{dt} \int_V r_{P/C} \times v_{C/\phi} \, \rho \, dV + \frac{d_0}{dt} \int_V r_{P/C} \times \left[\omega_{B/\phi} \times r_{P/C}\right] \rho \, dV$$

$v_{C/\phi}$: Not depending on ρ

$$= \frac{d_0}{dt}\left[\left(\int_V r_{P/C} \rho \, dV\right) \times v_{C/O}\right] + \frac{d_0}{dt} I_C\left(\omega_{B/\phi}\right)$$

$\int_V r_{P/C} \rho \, dV = \phi$

$$\frac{d_0}{dt} I_C\left(\omega_{B/\phi}\right) = M^{ext}$$

↳ resultant of external Moments.

$$\frac{d_0}{dt} I_C\left(\omega_{B/\phi}\right) = \frac{d_b}{dt} I_C\left(\omega_{B/\phi}\right) + \omega_{B/\phi} \times I_C\left(\omega_{B/O}\right)$$

$$\frac{d_b}{dt} I_C\left(\omega_{B/\phi}\right) = \frac{d_b}{dt} \int_V r_{P/C} \times \left(\omega_{B/\phi} \times r_{P/C}\right) \rho \, dV$$

$$= \int_V \frac{d_b}{dt}\left[r_{P/C} \times \left(\omega_{B/\phi} \times r_{P/C}\right)\right] \rho \, dV$$

$r_{P/C}$: always ϕ in b ←

$$= \int_V r_{B/C} \times \left[\left(\frac{db}{dt} \omega_{B/\phi} \right) \times r_{B/C} \right] \rho \, dV = I_C(\dot{\omega}_{B/\phi})$$

but $\boxed{\frac{db}{dt} \omega_{B/\phi} = \frac{do}{dt} \omega_{B/\phi}}$ so $\triangleq \dot{\omega}_{B/\phi}$

Euler equation of motion

$$I_C(\dot{\omega}_{B/\phi}) + \omega_{B/\phi} \times I_C(\omega_B) = M^{EXT}$$

We need to numerically compute these equations so let's see in algebraic form

Assume to project this equation in Frame ϕ?

You get ${}^{\phi}I_C \dot{\omega}_{B/\phi} + [{}^{\phi}\omega_{B/\phi} \times] + {}^{\phi}\dot{I}_C {}^{\phi}\omega_{B/\phi} = {}^{\phi}M^{EXT}$

- ${}^{\phi}I_C$: 3x3 time varying matrix (depends on body attitude)
- $\dot{\omega}_{B/\phi}$: unknown
- First order diff equation.

We need to couple this equation with the kinematics:

$${}^{\phi}_{b}\dot{R} = [{}^{\phi}\omega_{B/\phi} \times] \, {}^{\phi}_{b}R$$

can be computed offline.

$${}^{b}I_C \, {}^{b}\dot{\omega}_{B/\phi} + [{}^{b}\omega_{B/\phi} \times] \, {}^{b}I_C \, {}^{b}\omega_{B/\phi} = {}^{b}M^{EXT}$$

- ${}^{b}I_C$: comes out from cad
- Same as the one before, but computationally much easier.

$${}^{o}_{b}\dot{R} = {}^{o}_{b}R \, [{}^{b}\omega_{b/\phi} \times]$$

VERY NICE

NOTE

$$\boxed{\begin{aligned} {}^{a}A \, {}^{a}u &= {}^{a}[A(u)] \\ {}^{b}A \, {}^{b}u &= {}^{b}[A(u)] \end{aligned}}$$

$${}^{a}_{b}R \, {}^{b}A \, {}^{b}u = \underbrace{{}^{a}_{b}R \, {}^{b}A \, {}^{b}_{a}R}_{{}^{a}A} \, \underbrace{{}^{a}u}_{{}^{b}u}$$

$$\begin{cases} {}^b\dot{\omega}_b = {}^bI_c^{-1}\left[\left[{}^b\omega_{b/0}\times\right]{}^bI_c\,{}^b\omega_{b/\phi} + {}^0M^{EXT}\right] \\ {}^\phi_b\dot{R} = {}^0_bR\left[{}^b\omega_{b/\phi}\times\right] \Rightarrow \begin{matrix} {}^\phi\hat{\imath}_B = \text{col}_1(\ ) \\ {}^\phi\hat{k}_B = \text{col}_3(\ ) \end{matrix} \end{cases}$$

You have to check that the vectors are orthonormal $\hat{\imath}_B = \text{col}_1({}^0_BR\,[{}^b\omega_{b/\phi}\times])$

(is not a function of linear or cyclic cell)

(You could also use quaternions)

These are non linear differential equations

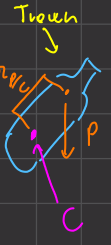
Conceptually there are xx first order diff equation because you need 3 parameters for orientation.
Angular velocity and orientation can be done with Euler angles.

Summarize Newton-Eulero for the rigid body

They are two sets of equations describing the evolution of the body looking at its center of mass.

$$\left.\begin{aligned} m\frac{d_0}{dt}v_{c/\phi} &= F^{ext} \\ I_c(\dot{\omega}_{c/\phi}) + \omega_{c/\phi}\times I_c(\omega_{b/\phi}) &= M^{EXT} \end{aligned}\right\}$$ Forward dynamics

REMARK



All the external forces acting on the bottle are only $F^{EXT} = mg$

Now let's see for any. vol.

integrate for every point

$$m_{p/c}^{EXT} \rightarrow M^{EXT} = \int_V r_{p/c}\times g\,dm = \int_V (r_{p/c}\times g)\,\rho\,dV$$

$$m_{p/c}^{EXT} = r_{p/c}\times f_p^{EXT}$$

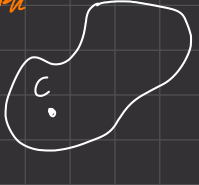
$$= \left[\int_V r_{p/c}\,\rho\,dV\right]\times g = \emptyset$$

$\int_V r_{p/c}\,\rho\,dV = 0$

The gravity is not generally momentum.

The inertia matrix can be chosen in smart way:

REMARK



Assume $\mathcal{B}$ such that ${}^{\mathcal{B}}I_c = \begin{bmatrix} I_1 & \emptyset & \emptyset \\ \emptyset & I_2 & \\ \emptyset & & I_3 \end{bmatrix}$

$I_i > \emptyset$ for rigid bodies. $i\in(1..3)$

We have 3 conditions:

① General structure $I_1 > I_2 > I_3$ (principal axis)

rotations on the intermediate axis are unstable

Rotation around the largest inertia moment are stable rotations and locally asymptotically stable.

② Spherical structure when $I_1 = I_2 = I_3$ Any frame is a principle inertia

Note that geometrically could be not a sphere if made of different density material

③ Gyroscopic structure $I_1 = I_2 \neq I_3$

$${}^b I_C = \begin{bmatrix} I_1 & \emptyset & \emptyset \\ \emptyset & I_2 & \emptyset \\ \emptyset & \emptyset & I_3 \end{bmatrix} \qquad {}^b\omega_{b/\emptyset} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \Big\} \text{ scalar functions.} \qquad M^{EXT} = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix}$$

$${}^b I_C \, {}^b\dot{\omega}_{b/\emptyset} + \left[{}^b\omega_{b/\emptyset} \times \right] {}^b I_C \, \omega_{b/\emptyset} = {}^b M^{EXT}$$

$$\begin{bmatrix} I_1 & \emptyset & \emptyset \\ \emptyset & I_2 & \emptyset \\ \emptyset & \emptyset & I_3 \end{bmatrix} \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} + \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & \emptyset & -\omega_x \\ -\omega_y & \omega_x & \emptyset \end{bmatrix} \underbrace{\begin{bmatrix} I_1 & \emptyset & \emptyset \\ \emptyset & I_2 & \emptyset \\ \emptyset & \emptyset & I_3 \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}}_{\begin{bmatrix} I_1\omega_x \\ I_2\omega_y \\ I_3\omega_z \end{bmatrix}} = {}^b M^{EXT}$$

$$I_1 \dot{\omega}_x + \left[-\omega_z I_2 \omega_y + \omega_y I_3 \omega_z \right] = M_x^{EXT}$$

$$I_2 \dot{\omega}_y + \left[\omega_z I_1 \omega_x - \omega_x I_3 \omega_z \right] = M_y^{EXT}$$

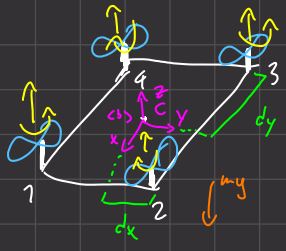
$$I_3 \dot{\omega}_z + \left[-\omega_y I_1 \omega_x + \omega_x I_2 \omega_y \right] = M_z^{EXT}$$

$$\begin{cases} I_1 \dot{\omega}_x + (I_3 - I_2)\,\omega_z \omega_y = M_x \\ I_2 \dot{\omega}_y + (I_3 - I_1)\,\omega_x \omega_z = M_y \\ I_3 \dot{\omega}_z + (I_3 - I_1)\,\omega_x \omega_y = M_z \end{cases}$$

the signs depend on the type.

11/03/24

Drone dynamics



$Mass = m > 0$

$I_x = I_y \neq I_z > 0$

$${}^bI_C = \begin{bmatrix} I_x & & \\ & I_y & \\ & & I_z \end{bmatrix}$$

$\dot{q}_i$ = ang vel of the propellers.

$f_i = \hat{k}_b \gamma_p \dot{q}$

f_i: forces generated by motor.

γ_p: constant related to the slope of the propeller

${}^b\hat{k}_b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

$$F^{EXT} = mg + f_1 + f_2 + f_3 + f_4$$

$${}^{\phi}F^{EXT} = m\,{}^{\phi}\ddot{p} + {}^{\phi}_bR\left[{}^bk_b\,\gamma_p\,\dot{q}_1 + {}^bk_b\,\gamma_p\,\dot{q}_2 + {}^bk_b\,\gamma_p\,\dot{q}_3 + {}^bk_b\,\gamma_p\,\dot{q}_4\right] =$$

$${}^{\phi}\ddot{p} = \begin{bmatrix} 0 \\ 0 \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}$$

$$\gamma_p \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{{}^bk_b} \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix}}_{\dot{q}}$$

$${}^{\phi}_bR \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{\phi}i_b & {}^{\phi}j_b & {}^{\phi}k_b \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = {}^{\phi}k_b$$

$$\Rightarrow : \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \gamma_p\,{}^{\phi}k_b \underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}}_{1} \dot{q} = {}^{\phi}F^{EXT}$$

N.B.

${}^br_{IC} = \begin{bmatrix} d_x \\ -d_x \\ 0 \end{bmatrix}$

$m_1 = r_{IC} \times f_1$

${}^bf_1 = \begin{bmatrix} 0 \\ 0 \\ f_1 \end{bmatrix}$

$${}^bm_1 = \begin{bmatrix} 0 & 0 & -d_x \\ 0 & 0 & -d_y \\ +d_x & +d_y & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ f_1 \end{bmatrix} = \begin{bmatrix} -d_x \\ -d_y \\ 0 \end{bmatrix} f_1$$

$$M^{EXT} = \left(-\tau_1 - \hat{\tau}_2 - \tau_3 - \tau_4\right) + \hat{i}_b\left(f_2 + f_3 - f_1 - f_4\right) d_x + \hat{j}_b\left(f_3 + f_4 - f_1 - f_2\right) d_y$$

(proportional)

$${}^bM^{EXT} = -\gamma_m \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{{}^bk_b} \underbrace{\begin{pmatrix} 1, -1, 1, -1 \end{pmatrix}}_{2} \dot{q} + \gamma_p \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{\text{propeller}} \underbrace{\begin{pmatrix} -1, 1, 1, -1 \end{pmatrix}}_{3} \dot{q} + \gamma_p \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \underbrace{\begin{pmatrix} -1, -1, 1, 1 \end{pmatrix}}_{4} \dot{q}$$

Reaction

M_x

M_y

The motor has to generate torque to move the propeller. The 3rd Newton law can be applied to torque to the motor generates torque and the propeller generates counter torque on the motor.

We can assume that τ_i is proportional to the spinning of the propeller.

$$\tau_i = \hat{k}_b \gamma_m \dot{q}_i$$

If you put these on top of each other the matrix is non singular so you can choose a speed for the motors that produce correct thrust and torque. This is invertible $N \neq 0$.

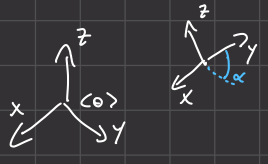
$$ {}^{b}M^{EXT} = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{cases} M_x = \gamma_p [-1\ 1\ 1\ -1]\, \dot{q}\, dx \\ M_y = \gamma_p [-1\ -1\ 1\ 1]\, \dot{q}\, dy \\ M_z = \gamma_m [1\ -1\ 1\ -1]\, \dot{q} \end{cases} $$

Set of linear equation coupled with (1)

Calculate moments wrt the body cb> and Forces wrt the origin <o

$$ {}^{\phi}_{b}\dot{R} = {}^{\phi}_{b}R^{T} \left[{}^{b}\omega_{b/\phi} \times \right] $$

Remember the step down.
By integrating this you get R



$$ {}^{\phi}_{b}R = {}^{\phi}_{1}R\, {}^{1}_{2}R\, {}^{2}_{3}R = (*) $$

$$ {}^{\phi}_{1}R = R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & -s\alpha \\ 0 & s\alpha & c\alpha \end{bmatrix} $$

$$ {}^{1}_{2}R = R_y(\beta) = \begin{bmatrix} c\beta & 0 & s\beta \\ 0 & 1 & 0 \\ -s\beta & 0 & c\beta \end{bmatrix} $$

$$ {}^{2}_{3}R = R_z(\gamma) = \begin{bmatrix} c\gamma & -s\gamma & 0 \\ s\gamma & c\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

$$ (*) = \begin{bmatrix} * & * & s\beta \\ * & * & -s\alpha\, c\beta \\ * & * & c\alpha\, c\beta \end{bmatrix} $$

Odd out

$$ F_x = \gamma_p\, s\beta\, \mu_T $$

$$ F_y = \gamma_p (-s\alpha\, c\beta)\, \mu_T $$

$$ F_z = \gamma_p (c\alpha\, c\beta)\, \mu_T $$

Dynamic equations

$$ m \frac{do}{dt} \dot{\Omega}_{c/\phi} = F^{EXT} $$

$$
\begin{cases}
m\ddot{x}(t) = F_x = \gamma_p (s\beta)\mu^T \\
m\ddot{y}(t) = F_y = \gamma_p (s\alpha c\beta)\mu^T \\
m\ddot{z}(t) = F_z = -mg + \gamma_p (s\alpha c\beta)\mu^T
\end{cases}
$$

μ^T is always different from 0 so the change force can change the condition of the drone on always.

$$
{}^b I_C \, {}^s\dot{\omega}_b + [{}^b\omega_{b/\phi}]_\times \, {}^b I_C \, {}^b\omega_{3/\phi} = {}^b M^{EXT}
$$

$$
{}^b\omega_{b/\phi} = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}
$$

$$
\begin{cases}
I_x \dot{\omega}_x + (I_z - I_y)\omega_y \omega_z = M_x \\
I_y \dot{\omega}_y + (I_x - I_z)\omega_x \omega_z = M_y \\
I_z \dot{\omega}_z = M_z
\end{cases}
$$

■ Compute $F_x, F_y, F_z, M_x, M_y, M_z$ for a steady drone and compute u_T

* from here $u_T = \frac{1}{\gamma_p} mg$ $\quad \begin{cases} \alpha = 0 \\ \beta = 0 \\ m\ddot{z}(t) = 0 \end{cases}$

Another exercise: ■ Velocity in euler form

$${}^2_b R = R_x(\alpha)\, R_y(\beta)\, R_x(\gamma)$$

Represent angular velocity of two frames with $(\dot{\alpha}, \dot{\beta}, \dot{\gamma})$

$$\omega_{1/\phi} = \hat{\imath}_1 \cdot \dot{\alpha} = \hat{\imath}_0 \dot{\alpha}$$

$$\omega_{2/1} = \hat{\jmath}_2 \cdot \dot{\beta} = \hat{\jmath}_1 \cdot \dot{\beta}$$

$$\omega_{3/2} = \hat{k}_3 \cdot \dot{\gamma} = \hat{k}_2 \dot{\gamma}$$

$$\omega_{3/\phi} + \omega_{3/0} = \omega_{1/\phi} + \omega_{2/1} + \omega_{3/2} = \hat{\imath}_0 \dot{\alpha} + \hat{\jmath}_1 \dot{\beta} + \hat{k}_2 \dot{\gamma}$$

$$
{}^b\omega_{3/\phi} = \left[{}^b J^A(\alpha, \beta, \gamma) \right] \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix}
$$

Do the calculations

It's a Jacobian matrix (angular part)

If you compute the derivative of J you get:

$${}^b\dot{\omega}_{b/\phi} = {}^b J^A \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \\ \ddot{\gamma} \end{bmatrix} + \dot{J}^A \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{bmatrix}$$

→ YOU DO compute this term

If you use the formula found in the euler formula you get 3 diff equations that allow you to not use the integration of the simplex.

$$ {}^{b}I_C\, {}^{b}\dot{\omega}_b + [{}^{b}\omega_{b/\phi}] \times {}^{b}I_C\, {}^{b}\omega_{b/\phi} = {}^{b}M^{EXT} $$

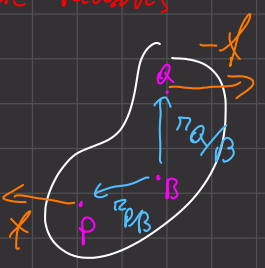
IS

$$ J^T I_C J^A \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \\ \ddot{\gamma} \end{bmatrix} + J^T B(\alpha,\beta,\gamma,\dot{\alpha},\dot{\beta},\dot{\gamma}) = J^T\, {}^{b}M^{EXT} $$

Inertia

23/09/24

Basic Remarks



P and Q are arbitrary

A pure moment can be expressed as a couple of forces

Parallel => equal magnitude, but opposite, they act at different lines

B arbitrary

$F_{tot} = f_1 + f_2 = 0$ -> the object is not subject to forces

Torques

$$ M_{tot} = r_{P/B} \times f + r_{Q/B} \times (-f) = [r_{P/B} - r_{Q/B}] \times f = r_{P/Q} \times f $$

M_B invariant to the choice of B

$$ F_{tot} = \sum_{i=1}^{N} f_i = f_1 + f_2 + \dots + f_N $$

(You can change the application point)

↳ EXT forces (because int = 0 because of rigid body)

$$ M_{tot}^{CRI} = \sum_{i=1}^{C} m_i + \sum_{i=1}^{N} r_{P_i/B} \times f_i $$

Remark 2

center of mass

Take force f acting on body: express it with C.

In this case $F_{tot} = f$

$M_{tot} = r_{P/C} \times f$

Project this equation in the reference frame you like:

I choose frame (0)

$$^{0}M_{tot} = \mathbb{1}_3\, 0 + [\,^{0}r_{PC}\times]\, ^{0}f$$

$$^{0}f_{tot} = 0 + \mathbb{1}_3\, ^{0}f$$

Adding m now

$$^{0}M_{tot} = \mathbb{1}_3\, m + [\,^{0}r_{PC}\times]\, ^{0}f$$

$$^{0}F_{tot} = 0 + \mathbb{1}_3\, ^{0}f$$

In matrix form

$$\begin{bmatrix} ^{0}M_{tot} \\ ^{0}F_{tot} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbb{1}_3 & [\,^{0}r_{PC}\times] \\ 0_3 & \mathbb{1}_3 \end{bmatrix}}_{\text{Always invertible}} \underbrace{\begin{bmatrix} ^{0}m \\ ^{0}f \end{bmatrix}}_{\text{Pure forces and torques}}$$

Remark

The jacobian is used to transform joint velocities (linearly)

$$\begin{bmatrix} ^{0}v_{e/0} \\ ^{0}\omega_{e/0} \end{bmatrix} = {}^{0}J_{e/0}\,\dot{q}\,, \quad {}^{0}J_{e/0} \in \mathbb{R}^{6\times n}$$

Basic jacobian: $^{0}J_{v_{i/0}}$

$$\begin{bmatrix} ^{0}v_{i/0} \\ ^{0}\omega_{i/0} \end{bmatrix} = {}^{0}J_{v_{i/0}}\,\dot{q}$$

The relationship between the previous formulas:

$$\begin{cases} \omega_{e/0} = \omega_{i/0} \rightarrow \text{no relation} \\ v_{e/0} = v_{i/0} + \omega_{i/0} \times r_{ei} \end{cases}$$

So we have:

$$\begin{bmatrix} {}^{\phi}\omega_{e/\phi} \\ {}^{\phi}v_{e/\phi} \end{bmatrix} = \begin{bmatrix} \mathbb{1}_3 & \phi_3 \\ [{}^{\phi}r_{e/n}\times]^T & \mathbb{1}_3 \end{bmatrix} \begin{bmatrix} {}^{\phi}\omega_{n/\phi} \\ {}^{\phi}v_{n/\phi} \end{bmatrix} =$$

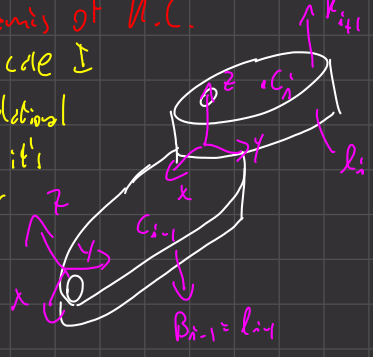
$$= {}^{\phi}\mathcal{J}_{e/n} \begin{bmatrix} {}^{\phi}\omega_{n/\phi} \\ {}^{\phi}v_{n/\phi} \end{bmatrix}$$

Nice looking :)

$$\begin{bmatrix} {}^{\phi}M_{rot} \\ {}^{\phi}F_{rot} \end{bmatrix} = \begin{bmatrix} \mathbb{1}_3 & [{}^{\phi}r_{e_C/e}\times] \\ \phi_3 & \mathbb{1}_3 \end{bmatrix} \begin{bmatrix} {}^{\phi}I_m \\ {}^{\phi}f \end{bmatrix}$$

Dynamics of N.C.

In this case I consider rotational joints but it's the same for trans. axes



Part of open kinematic

Problem: Motion of this bodies when it's subjected to forces

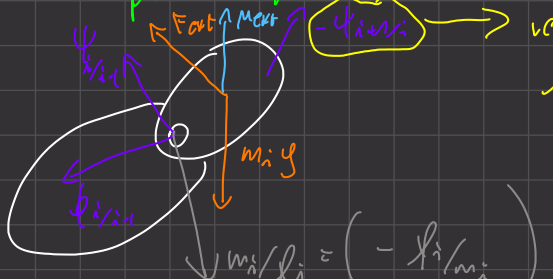
1) Specify the type of forces and torques that arrive from internal links

2) N bodies => 6n differential equations.

3) Interconnections: constraints => there are joints that limitate the motion,

we want a method to consider less equations and arrive to our purpose.

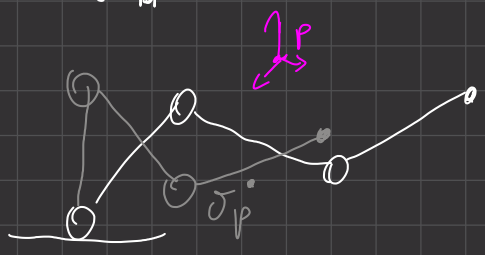
Consider a planar pendulum:



reaction of torque from l_{i+1} on l_i

The internal forces are now important because they keep everything together.

Couple $\begin{cases} F_{tot} \\ M_{tot} \end{cases}$



What happens assigning to any arbitrary variation to each joint variables?

$\delta p \rightarrow ??? \quad$ TODO

$\delta \Theta \rightarrow$ virtual displacement (rotation)

$\delta q_i \rightarrow$ virtual joint displacement

For open Kinematic chains the virtual joint displacements are arbitrary and independent

If I consider closed loop NO:

the movement is constrained.

You can't choose all the angles as you want.

How do I put in relation δP and $\delta \Theta$?

I consider generic link i:

$$\phi \omega_{i/\phi} = \phi J^{A}_{i/\phi} \dot{q}$$

$$\phi v_{P/\phi} = \phi J^{L}_{P/\phi} \cdot \dot{q}$$

Very tiny interval δt: $\dot{q} = \frac{dq}{dt} \approx \frac{\delta q}{\delta t}$

And I get:

$$\phi \omega_{i/\phi} \delta t = \delta \Theta = \phi J^{A}_{i/\phi} \delta q$$

$\Rightarrow$ angle integrated over an infinitesimal amount of time

$$\phi v_{P/\phi} \delta t = \delta \underbrace{\phi r_{P/\phi}}_{\delta P} = \phi J^{L}_{P/\phi} \delta q$$

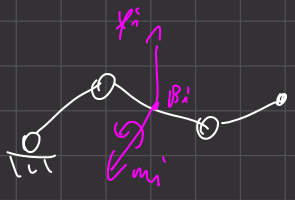
Virtual Work

Remark:

Work of a force is $W = f \cdot \Delta P$

We consider 2 types of forces with association with

- forces $\delta \Lambda = f \cdot \delta P$
- torques $\delta \Lambda = m \cdot \delta \Theta$



Torques:

$$\delta \Lambda = m_i \delta \Theta_i = {}^{\phi} m_i^T \, {}^{\phi} J^A_{B_i/\phi} \, \delta q = \delta q^T \left({}^{\phi} J^A_{B_i/\phi} \right)^T {}^{\phi} m_i$$

Point B_i — Remember the scalar product in a different way

Forces:

$$\delta \Lambda_i = f_i \cdot \delta P_i = {}^{\phi} f_i^T \, {}^{\phi} J^L_{B_i/\phi} \, \delta q = \delta q^T \left({}^{\phi} J_{B_i/\phi} \right)^T f_i$$

Remember: set of 3 dim vectors with ϕ and 1 vect not. or something

$${}^{\phi} J^A_{B_n/\phi} = \left[{}^{\phi} J^A_1 \; ; \; {}^{\phi} J^A_2 \; ; \; \dots \; ; \; {}^{\phi} J^A_n \; ; \; \phi \dots \phi \right]$$

$$J^A_i = \begin{cases} h_i & r_i = R \\ \phi & r_i = P \end{cases}$$

So: $${}^{A^T} J_{B_i/\phi} \cdot m_i = \begin{bmatrix} J^A_1 \cdot m_1 \\ J^A_2 \cdot m_2 \\ \vdots \\ J^A_i \cdot m_i \\ \phi \\ \vdots \\ \phi \end{bmatrix}$$

$$J^A_i \cdot m_i = \begin{cases} h_i \cdot m_i & r_i = R \\ \phi & r_i = P \end{cases}$$

Some considerations for lower part:

$${}^{\phi} J^L_{B_n/\phi} = \left[{}^{\phi} J^L_{B_n/1} \; ; \; {}^{\phi} J^L_{B_n/2} \; ; \; \dots \; ; \; {}^{\phi} J^L_{B_n/i} \; ; \; \phi \dots \right]$$

$$J^{L}_{B_i/\ell} = \begin{cases} (k_\ell \times r_{B_i/\ell}) & \Gamma_i = R \\ k_\ell & \Gamma_i = P \end{cases}$$

$$J_{D_i}: \quad {}^{\varphi}J^{L}_{B_i/\varphi} \cdot f_i = \begin{bmatrix} J^{L}_{B_i/\ell} f_i \\ \phi \\ \phi \\ \phi \end{bmatrix} \in \ell$$

$$J^{L}_{B_i/\ell} \cdot f_i \begin{cases} (k_\ell \times r_{B_i/\ell}) f_i & \Gamma_i = R \\ k_\ell \times f_i & \Gamma_i = P \end{cases}$$

$$(k_\ell \times r_{B_i/\ell}) f_i = \underbrace{(r_{B_i/\ell} \times f_i) \cdot k_\ell}$$

cyclic permutation

torque generated by forces and projected on ℓ

Let's consider this set of forces and torques:

M_i

F_{C_i}

$$\delta\Lambda^{(tot)} = \sum_{i=1}^{m} \delta\Lambda^{(m)}_{C_i} + \delta\Lambda^{(k)}_{C_i} = \sum_{i=1}^{m} \delta q^T \, {}^{\varphi}J^{A\,T}_{C_i/\varphi} M_i + \delta q^T \, {}^{\varphi}J^{L\,T}_{C_i/\varphi} F_{C_i}$$

δq does not depend on i:

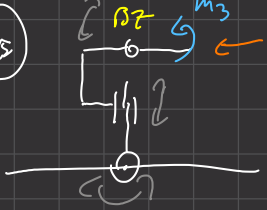
$$\delta q^T \sum_{i=1}^{m} \begin{bmatrix} {}^{\varphi}J^{A\,T}_{C_i/\varphi} & ; & {}^{\varphi}J^{L\,T}_{C_i/\varphi} \end{bmatrix} \begin{bmatrix} \varphi_{M_i} \\ \varphi_{F_{C_i}} \end{bmatrix}$$ (Matrix form)

$\delta\Lambda^{(tot)}$ is the same for M_i + ... and you get the following formula:

multi-call

$$\delta\Lambda^{(tot)} = \delta q^T \sum_{i=1}^{m} {}^{\varphi}J^{T}_{A_i/\varphi} \begin{bmatrix} M_i \\ F_i \end{bmatrix}$$

Example



$${}^{\varphi}J^{A}_{B_3/\varphi} = \begin{bmatrix} {}^{\varphi}k_1 & \phi & {}^{\varphi}k_2 \end{bmatrix}$$

$${}^{\phi}\mathcal{J}^{A^T}_{B_3/\phi}\, m_3 = \begin{bmatrix} k_1 \cdot m_3 \\ \phi \\ k_3\, m_3 \end{bmatrix}$$

if: $m_3 = \begin{bmatrix} \phi \\ \phi \\ * \end{bmatrix}$ $\qquad k_1 \cdot m_3 = [0\ 0\ 1] \begin{bmatrix} 0 \\ 0 \\ * \end{bmatrix} = *$

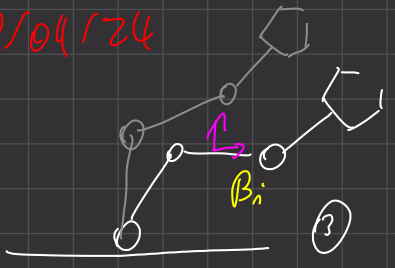
$$k_3 \cdot m_3 = [0\ 0\ 1] \begin{bmatrix} 0 \\ 0 \\ * \end{bmatrix} = *$$

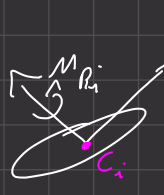
$${}^{\phi}\mathcal{J}^{L^T}_{B_3/\phi} \cdot f_3 = \begin{bmatrix} (k_1 \times r_{O_3/1}) \cdot f_3 \\ k_2 \times f_3 \\ (k_3 \times r_{O_3/3}) \cdot f_3 \end{bmatrix}$$

→ Dependency of direction of f_3 (by the cross)

// so the result is ϕ

18/04/24





δq_i

δB_i

$\delta \Theta_i$

$\delta B_i = {}^{\phi}\mathcal{J}_{B_i/\phi}\, \delta q$

$\delta \Theta_i = {}^{\phi}\mathcal{J}^{A}_{A/\phi}\, \delta q$

$$\delta L_i = \begin{cases} F_{C_i} \cdot \delta C_i = \delta q^T \left({}^{\phi}\mathcal{J}^{L}_{C_i/\phi} \cdot {}^{\phi}F^{EXT}_{C_i} \right) \\ M_{C_i}\, \delta \Theta_i = \delta q^T \left({}^{\phi}\mathcal{J}^{A}_{C_i/\phi}\, {}^{\phi}M_{C_i} \right) \end{cases}$$

virtual work — force acting on body on CM

N.B this is very important

It's the projection of the force acting on i and projected on the joint spaces. This is a geometric projection.

sum for each body

$$\delta L = \sum_{i=1}^{n} \delta q^T\, {}^{\phi}\mathcal{J}^{T}_{C_i/\phi} \begin{pmatrix} {}^{\phi}M_{C_i} \\ {}^{\phi}F_{C_i} \end{pmatrix} = \delta q^T \left[\sum_{i=1}^{n} {}^{\phi}\mathcal{J}^{T}_{C_i/\phi} \begin{pmatrix} {}^{\phi}M_{C_i} \\ {}^{\phi}F_{C_i} \end{pmatrix} \right]$$

→ transform forces acting on body into joint coordinates

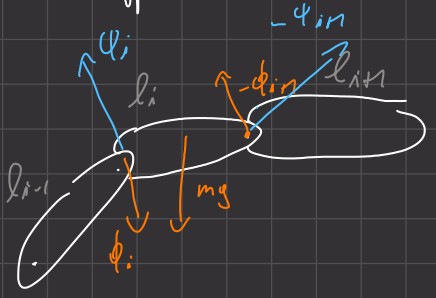
Wrench

n-dimensional vectors

$6 \left[\underbrace{[\mathcal{J}]}_{n} \right]$

$${}^{\phi}\mathcal{X}_{i/\phi} = {}^{\phi}\mathcal{S}_{C_i/i}\, {}^{\phi}\mathcal{J}_{i/\phi}$$

What happens if we connect many bodies:



The external forces are those who are not depending on the interconnection of the bodies. my will always be felt on the robot arm.

l_{i-1} is producing forces acting on i

$$\phi_i + (\phi_{i+1}) = 0 \qquad \psi_i + (\psi_{i+1}) = 0$$

they compensate

there are also actuation forces and torques

there exist actuation torques m_i which are producing the relative motion between body i and $i-1$.

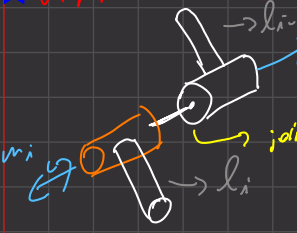
Our software will have to send this numbers to the motors. the magnitude of the vector is what we send.

there will also be a m_{i+1} counteracting on the joint. (Extension of 3rd newton law)

As an external force there is also the gravity.

Now it's important to compute the virtual work for every interaction of every body on the other.

VIRTUAL WORK OF ACTUATION F/T

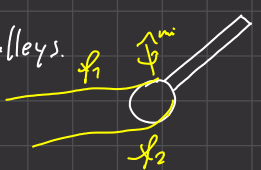


This is very typical

Joint congruent actuation (the actuation (force/torque) is actuated by motors on joints)

We can think of the connection of two bodies as a rot motor. One of the virtual things in what is the routing of the cable.

there is also actuation expressed by tendons and pulleys.



It would happen that the motor is easy back in the chain.

If we have to handle this situation the action of the forces is as if they are external forces.

Let's split between rotational and translational joints:

Rotational joints

this can be reduced to a scalar

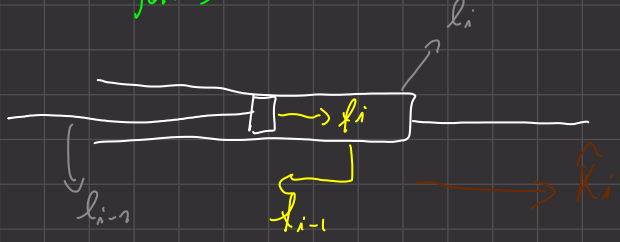
$$\delta L_i^{(act)} \rightarrow \text{actuation} = m_i \cdot \delta\Theta_i - m_i \cdot \delta\Theta_{i-1} = m_i \underbrace{[\delta\Theta_i - \delta\Theta_{i-1}]}_{\hat{K}_i \delta q_i} = \underbrace{(m_i \hat{K}_i)}_{\text{scalar}} \delta q_i$$

$\delta\Theta_i$: ang. displacement

$= \hat{T}_i \delta q_i$

geometrically this is the projection of the actuation on the rot axis so this is a scalar.

Translational joints



$$\delta L_i^{(act)} = f_i \delta P_i - f_i \delta Q_{i-1} = f_i \cdot \underbrace{[\delta P_i - \delta Q_{i-1}]}_{K_i \delta q_i \text{ scalar}} = \underbrace{(f_i \hat{K}_i)}_{f_i} \delta q_i$$

f_i: projection of the force on the axis of actuation.

$= \hat{T}_i \delta q_i$

$\hat{T}_i$ is the generalized actuation force or torque:

$$\hat{T}_i = \begin{cases} \underline{m_i} \cdot \underline{\hat{K}_i} \triangleq m_i & \hat{T}_i = R \\ \underline{f_i} \cdot \underline{\hat{K}_i} \triangleq f_i & \hat{T}_i = P \end{cases}$$

N.B.

$$\delta L^{(act)} = \sum_{i=1}^{n} \delta L_i^{(act)} \overset{(f+m)}{=} \sum_{i=1}^{n} \hat{T}_i \delta q_i = \delta q^T T$$

this is a scalar product between two vectors

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_n \end{bmatrix} \rightarrow \text{general } f/t \text{ actuation vector}$$

If all the components of this vector can be generated by independent motions the robot is fully actuated.

If there are less actuators then is under actuated

The drone is under actuated $\begin{cases} 4 \text{ motors} \\ 6 \text{ parameters.} \end{cases}$

Virtual work of reaction F/T

Ideal joints

One for which the virtual work of the reaction force or torque is equal to 0.

$$\delta \mathcal{W}_i = \phi_i \cdot \delta P_i - \phi_i \cdot \delta Q_{i-1} = \phi_i \cdot \begin{cases} \phi & \Gamma = R \\ \hat{K}_i \, \delta q_i & \Gamma = P \end{cases}$$

↓ the effect of reaction forces and torques at joint i.

$$\delta \mathcal{W}_i = \Psi_i \cdot \delta \theta_i - \Psi_i \cdot \delta \theta_{i-1} = \Psi_i \cdot \begin{cases} \hat{K}_i \, \delta q_i & \Gamma = R \\ \phi & \Gamma = P \end{cases}$$

$$\delta \mathcal{W}_i^{TOT} = \begin{cases} (\Psi_i \cdot \hat{K}_i) \, \delta q_i & \Gamma_i = R \\ (\phi_i \cdot K_i) \, \delta q_i & \Gamma_i = P \end{cases} \underbrace{= \phi \quad \forall \delta q_i}$$

If ideal this is true

Physically means that the reaction torque keeping the bodies together do not have any components on the rot axis and the same goes for traslation motion. We are supposing the joints to be frictionless.

What happens if we have to consider friction? We will see later.
What's going to happen is the total reaction of forces and torques which appear in the KC:

$$\boxed{\delta \mathcal{W}_i = \sum_{i=1}^{n} \delta_i \mathcal{W}_i^{TOT} = \phi}$$

Our joints are modelled as frictionless joints.

We have 3 classes of forces (external, actuation, reaction)

ROBOT Equilibrium equations.

We are doing the same reasoning of the base for a general chain.

$$\delta L^{(EXT)} = \delta q^T \sum_{i=1}^{n} {}^{\phi}J_{C_i/q}^T \begin{pmatrix} {}^{\phi}M_{C_i} \\ {}^{\phi}F_{C_i} \end{pmatrix} = \delta q^T \sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} {}^{\phi}\mu_i \\ {}^{\phi}f_i \end{pmatrix} \leftarrow \text{TODO prove this}$$

$$\delta L^{(ACT)} = \delta q^T \tau$$

$$\delta L^{(R)} = \phi$$

The robot is in equilibrium with a mental reference frame where it has ϕ cupola lin vel. SO'.

① $\begin{cases} F_{C_i}^{(tot)} = \phi \\ M_{C_i} = \phi \end{cases}$ $\forall_i = 1..n$ on each body on the chain

Here I have (6xn equations)

The unknowns are $\begin{cases} \text{the velocities } (R/t) \\ \text{the reaction } (R/t) \end{cases}$

We know that the only reaction values are the ones along the relative R axis.

REMARK

$F_{C_i} \longrightarrow F_i$

$M_{C_i} \longrightarrow M_i = M_{C_i} + r_{C_i} \times F_i$

$\eta_{ext} = \begin{cases} F_i = \phi \\ M_i = \phi \end{cases}$ (6xn) equation

We want from 6xn equations to n equations

② $\delta P_i \cdot F_i + \delta \Theta_i \cdot M_i = \phi \quad \forall_i = 1..n$

We can now compute the total virtual work:

③ $\delta L = \sum_{i=1}^{n} \delta P_i F_i + \delta \Theta_i M_i = \phi$ single scalar equation which depends on virtual displacement which depends on relative displacement.

$F_i = F_i^{(A)} + F_i^{(R)}$

$M_i = M_i^{(A)} + M_i^{(R)}$

④ $\sum_{i=1}^{n} \delta P_i F_i^{(A)} + \delta \Theta_i \cdot M_i^{(A)} + \left(\sum_{i=1}^{n} \delta P_i F_i^{(R)} + \delta_i \Theta_i M_i^{(R)} \right) = \phi$ Equilibrium

Virtual work due to all reaction forces and torques.

$\delta L^{(R)} = \phi$ if ideal

We get to a point where reaction forces and torque disappear.

REMARK

$F_i^{(A)} = F_i^{(ENT)} + f_i$ ← actuation force

$M_i^{(A)} = M_i^{(ENT)} + m_i$ ↳ actuation torque

If you substitute this on the previous one you get some other equations.

20/04/24

Robot equilibrium conditions

N.E eq at eq

(1) $\begin{cases} M_{C_i} = \phi \\ F_{C_i} = \phi \end{cases} \quad \forall i \qquad \begin{cases} M_i = \phi \\ F_i = \phi \end{cases}$ wrt point P_i (simpler)

Let's think about the ideas behind these formulae.

When you see a robot in a fixed position is because there are bounds. We are seeking the torques to keep the robot in eq. condition

(2) $\delta L_i = \delta\Theta_i \cdot M_i + \delta P_i \cdot F_i = \phi$ → virtual work

wrt work

(3) $\delta L^{TOT} = \sum_{i=1}^{n} \delta L_i \qquad \begin{pmatrix} \phi_{\delta\Theta} \\ \phi_{\delta P_i} \end{pmatrix} = {}^{\phi}J_{i/\phi}\, \delta q$

Tot virtual works.

(4) $\delta L^{TOT} = \delta q^T \cdot \left[\sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} M_i \\ F_i \end{pmatrix} \right] = \phi$

(5) $\begin{cases} M_i = M_i^{(A)} + M_i^{(R)} \\ F_i = F_i^{(A)} + F_i^{(R)} \end{cases} \qquad \begin{aligned} M_i^{(A)} &= M^{EXT} + M^{(A)} \\ F_i^{(A)} &= F^{EXT} + F^{(A)} \end{aligned}$ → actuation

(6) $\delta L^{TOT} = \delta q^T \left[\sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} {}^{\phi}M_i^{(A)} \\ {}^{\phi}F_i^{(A)} \end{pmatrix} \right] + \underbrace{\delta q^T \left[\sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} {}^{\phi}M_i^{(R)} \\ {}^{\phi}F_i^{(R)} \end{pmatrix} \right]}_{\text{Total virtual work of reactive forces and torques}}$

$\delta L^{TOT} = \phi$

The assumption is always that joints are frictionless.

(6) $\delta L^{TOT} = \underbrace{\delta q^T \left[\sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} {}^{\phi}M_i^{(EXT)} \\ {}^{\phi}F_i^{(EXT)} \end{pmatrix} \right]}_{\delta L^{TOT}_{ext}} + \underbrace{\delta q^T \left[\sum_{i=0}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} M_i^{(A)} \\ F_i^{(A)} \end{pmatrix} \right]}_{\delta L^{TOT}_{ACT}}$

$\delta q^T \tau$ → joint torques actuation

If the robot is under-actuated some components of τ might be ϕ.

(7) $\delta L^{TOT} = \delta q^T \left[\sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^T \begin{pmatrix} M_i^{EXT} \\ F_i^{EXT} \end{pmatrix} + \tau \right] = \phi$ → set of generalized forces and torques at the joints. This is what the software generates. This is the true control vector of the robot.

LEMMA

if $z^T \cdot x = \phi \ \forall x \Rightarrow z = \phi$

Proof

$x = [0 \ 0 \ \underset{i^{th}}{1} \ 0 \dots 0]$ ← let's assume x in this way

then $z^T \cdot x = z_i = \phi$ for the hypothesis

$\forall i$

δL is a scalar product of this kind.

If for any possible δq the scalar product is ϕ the term in bracket must be ϕ.

All the components of the bracket must be ϕ.

$$\tau + \sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^{T} \begin{pmatrix} M_i^{EXT} \\ F_i^{EXT} \end{pmatrix} = \phi \quad (n \text{ equations})$$

Effect of forces and torques applied to every joint

This is basically a model because let's assume (the robot still) $\forall i$ $\begin{cases} \omega_{i/\phi} = \phi \\ v_{i/\phi} = \phi \end{cases}$ at time t_ϕ

The question is what is the torque that I have to send to keep the robot still?

We can specify $\tau^{eq} = -\sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^{T} \begin{pmatrix} {}^{\phi}M_i^{EXT} \\ {}^{\phi}F_i^{EXT} \end{pmatrix}$

This is our software specification starting from ϕ vel

Now the question is how to get $M_i^{(EXT)}$ and $F_i^{(EXT)}$

REMARK

$\tau + \sum_{i=1}^{n} {}^{\phi}J_{i/\phi}^{T} \begin{pmatrix} {}^{\phi}M_i^{EXT} \\ {}^{\phi}F_i^{EXT} \end{pmatrix}$ Let's check if this can be applied to c_i

${}^{\phi}J_{c_i/\phi} = {}^{\phi}S_{c_i/_i} \, {}^{\phi}J_{i/\phi}$ $\quad {}^{\phi}S_{c_i} = \begin{bmatrix} \mathbb{1} & \phi \\ {}^{\phi}[r_{c_i/\phi,i}]^T & \mathbb{1} \end{bmatrix}$ lower triangular always invertible matrix

$$\downarrow$$

$$\phi J_{i/\phi} = \phi S_{C_i/i}^{-1} \, \phi J_{C_i/\phi}$$

$$\rightarrow \begin{bmatrix} \mathbb{1} & \phi \\ -\left[\phi \Omega_{C_i/i} \times\right]^T & \mathbb{1} \end{bmatrix} = \begin{bmatrix} \mathbb{1} & \phi \\ \left[\phi \Omega_{i/C_i} \times\right]^T & \mathbb{1} \end{bmatrix} = S_{i/C_i}$$

EXAMPLE

$$M = \begin{bmatrix} A & \phi \\ B & C \end{bmatrix}$$

this is invertible only if A and C are invertible.

$$\Rightarrow \begin{bmatrix} A & \phi \\ B & C \end{bmatrix} \begin{bmatrix} X & \phi \\ Y & Z \end{bmatrix} = \mathbb{1}$$

$$AX = \mathbb{1} \Rightarrow X = A^{-1}$$

$$BX + CY = \phi \qquad Y = -C^{-1} BX$$

$$CZ = \mathbb{1} \Rightarrow Z = C^{-1} \qquad Y = -C^{-1} B A^{-1}$$

Inverses for us.

Getting back to the computations:

$$\Upsilon + \sum_{i=\phi}^{n} \phi J_{i/\phi}^T \begin{pmatrix} \phi M_i^{(EXT)} \\ \phi F_i^{(EXT)} \end{pmatrix} = \Upsilon + \sum_{i=\phi}^{n} \left(S_{i/C_i} \, \phi J_{C_i/\phi} \right)^T \begin{pmatrix} \phi M_i^{(EXT)} \\ \phi F_i^{(EXT)} \end{pmatrix} =$$

$$= \Upsilon + \sum_{i=\phi}^{n} \phi J_{C_i/\phi}^T \, S_{i/C_i}^T \begin{pmatrix} \phi M_i^{(EXT)} \\ \phi F_i^{(EXT)} \end{pmatrix} =$$

this is the transportation from point P_i to C_i

$$f_i = f_{C_i}$$

$$m_{C_i} = m_i + \Omega_{i/C_i} \times f_i$$

$$= \Upsilon + \sum_{i=1}^{n} \phi J_{C_i/\phi}^T \, S_{i/C_i}^T \begin{pmatrix} \phi M_{C_i}^{(EXT)} \\ \phi F_{C_i}^{(EXT)} \end{pmatrix}$$

Always equivalent and equal to ϕ

You can express forces and torques with the C.M. or any other point

this is important because we need to take into account gravity. Note also that the force in the center of mass does not produce any torques.

Reformulation

$$F_{C_i}^{(EXT)} \longrightarrow F_{C_i}^{(EXT)} + m_i g$$

$$\tau^{EQ} = -\sum_{i=1}^{n} J_{C_i/\phi}^{T} \begin{pmatrix} M_{C_i}^{(EXT)} \\ F_{C_i}^{(EXT)} \end{pmatrix} - \sum_{i=1}^{n} J_{C_i/\phi}^{T} \begin{pmatrix} \phi \\ \phi_g \end{pmatrix} m_i$$

$F_{C_i}^{(EXT)}$ → resultant of all external forces except gravity.

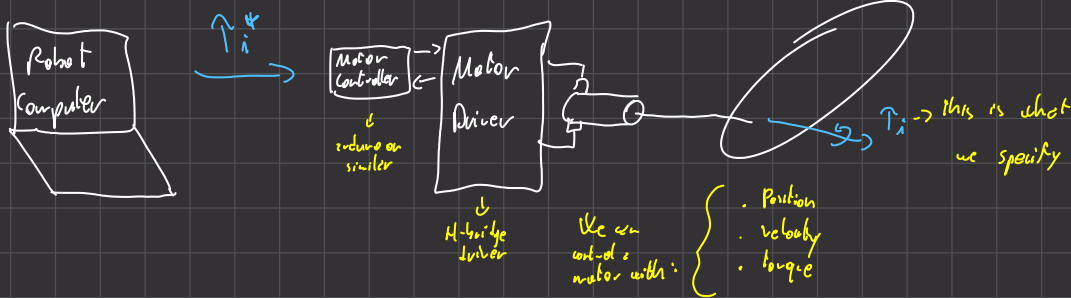
$$= -\sum_{i=1}^{n} J_{C_i/\phi}^{T} \begin{pmatrix} M_{C_i}^{(EXT)} \\ F_{C_i}^{(EXT)} \end{pmatrix} - \left(\sum_{i=1}^{n} m_i \phi_J^T{}_{C_i/\phi} \right) \begin{pmatrix} \phi \\ p_g \end{pmatrix}$$

gravity compensation terms → $G(q)$

function of the position of the robot.

REMARK

What is a robot?



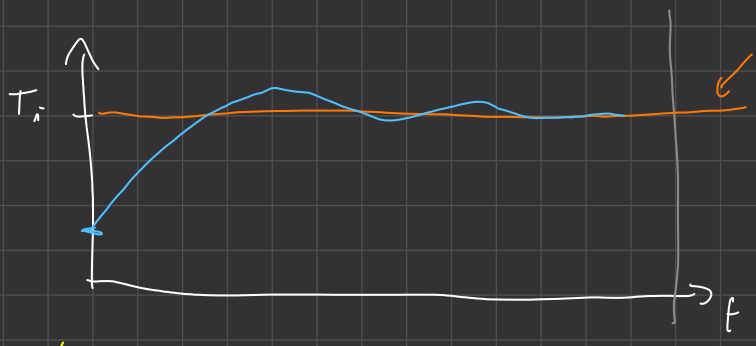
The red box must try to ensure $\tau_i = \tau_i^*$

- Ethernet
  T200/T26/T206 — UDP — TCP/IP
- Ethercat
  T200

the jitter here is in nanoseconds

- CAN / CAN-FD
  1 Mb/s — 8 Mb/s

I am assuming that $T_i \simeq T_i^*$ most of the time



We want the current response to be much slower than the temporal deadline

EXAMPLES of relevant cases

① No external forces and torques.

$$\underset{\sim}{\tau}^{EQ} = C(q) \rightarrow \text{gravity compensation term.}$$

② Robot holding something

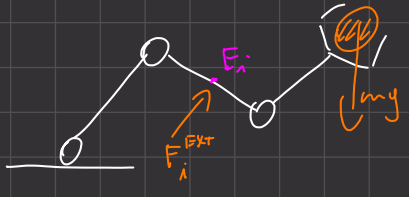
minus because $C(q)$ has the minus

$$\tau^{EQ} = C(q) - {}^{\phi}J^{T}_{E/\phi} \begin{pmatrix} \phi \\ g \end{pmatrix} m_b$$

$${}^{\phi}J_{F/\phi} = \begin{pmatrix} {}^{\phi}S_{E/n} & {}^{\phi}J_{w/\phi} \end{pmatrix}$$

Depending on what your grip screw will change S and m

③

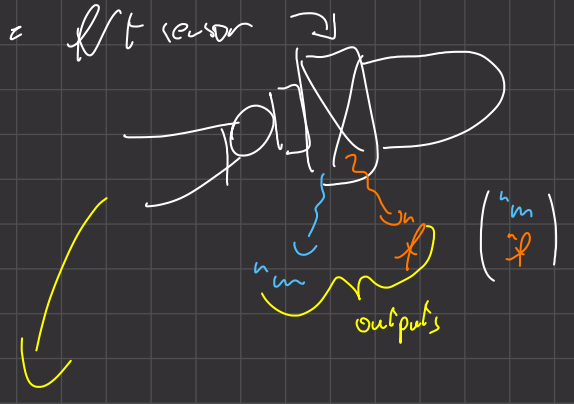


$$\tau^{EQ} = C(q) - {}^{\phi}J^{T}_{w/\phi} S^{T}_{e/n} \begin{pmatrix} \phi \\ g \end{pmatrix} m_b - {}^{\phi}J^{T}_{i/\phi} S_{e_i/i} {}^{\phi}J^{T}_{i/\phi} \begin{pmatrix} \phi \\ \phi_{F_i^{EXT}} \end{pmatrix}$$

We usually don't know position and magnitude of the force.

One way for estimating the force is using robot skin.

To measure forces and torques at the E.E usually it's mounted



They use strain gages or deformation sensors.

Here I don't really know what the force is applied with the shim I can move above.

03/04/24 Robot dynamics model

Recall

$$\begin{cases} M_{Ci} = \phi \\ F_{Ci} = \phi \end{cases} \quad \forall i \text{ at Equilibrium.}$$

$$\tau + \sum_{i=1}^{n} {}^{\phi}J_{Ci/\phi}^{T} \begin{pmatrix} {}^{\phi}M_{Ci}^{EXT} \\ {}^{\phi}F_{Ci}^{EXT} \end{pmatrix} - G(q) = \phi$$

τ: torque at every joint

$\sum$: $\tilde{\tau}^{EXT}$

$G(q)$: takes into account gravity

Behaviour of the robot at equilibrium.

$$\tau = -\tilde{\tau}^{EXT} + G(q)$$

Lesson

Modelling of the centripetal effect

$$\text{①} \quad \begin{cases} I_{Ci}(\omega_{i/\phi}) + \underline{\omega}_{i/0} \times I_{Ci}(\omega_i) = M_{Ci} \\ m_i \, \dot{v}_{Ci/\phi} = F_{Ci} \end{cases} \quad \forall i \in 1..n$$

These are equalities: if what's on the right it's force then what's on the left is force.

$$\text{②} \quad \begin{cases} \overbrace{I_{Ci}(\omega_{i/\phi}) + \underline{\omega}_{i/0} \times I_{Ci}(\omega_i)}^{M_i^D} - M_{Ci} = \phi \\ \underbrace{m_i \, \dot{v}_{Ci/\phi}}_{F_{Ci}^D} - F_{Ci} = \phi \end{cases}$$

Dynamic forces and torques

"physical forces and torques"

Dynamic equilibrium condition

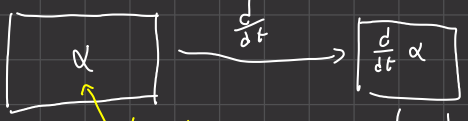
NOTE:

$$* \quad \begin{pmatrix} {}^{\phi}\omega_{C_i/\phi} \\ {}^{\phi}v_{C_i/\phi} \end{pmatrix} = {}^{\phi}J_{C_i/\phi}(q)\,\dot{q}$$

$$\begin{pmatrix} {}^{\phi}\omega_{i/\phi} \\ {}^{\phi}v_{C_i/\phi} \end{pmatrix} = {}^{\phi}J_{C_i/\phi}\,\ddot{q} + {}^{\phi}\dot{J}_{C_i/\phi}\,\dot{q}$$

"quadratic in $\dot{q}$" — elements linear in $\dot{q}$ · $\dot{q}$

each entry is the derivative of the original



J_{C_i}

$\Rightarrow \frac{d}{dt}\alpha(q) = \frac{\partial \alpha}{\partial q}\dot{q}$?

NOTE

$${}^{\phi}\left[\omega_{i/\phi} \times I_{C_i}(\omega_{i/\phi})\right]$$

linear in ω

omega is a linear function of $\dot{q}$ from *

Quadratic function in $\dot{q}$

Note

$$\begin{pmatrix} \delta\theta_i \\ \delta C_i \end{pmatrix} = {}^{q}J_{C_i/\phi}\,\delta q$$

(3) Compute the virtual work:

$$\delta L_{C_i} = \delta\theta_i \cdot \left[M^{D}_{C_i} - M_{C_i}\right] + \delta C_i \left[F^{D}_{C_i} - F_{C_i}\right] = \phi$$

Virtual work is ϕ

$$\delta L_{C_i} = \begin{bmatrix} {}^{\phi}\delta\theta_i \\ {}^{\phi}\delta C_i \end{bmatrix}^{T} \left\{ \begin{pmatrix} {}^{\phi}M^{D}_{C_i} \\ {}^{\phi}F^{D}_{C_i} \end{pmatrix} - \begin{pmatrix} {}^{\phi}M_{C_i} \\ {}^{\phi}F_{C_i} \end{pmatrix} \right\} = \phi$$

Work is always scalar.

Recalling step 2

$$\begin{pmatrix} {}^{\phi}M_{C_i} \\ {}^{\phi}F_{C_i} \end{pmatrix} = \begin{pmatrix} {}^{\phi}I_{C_i} & {}^{\phi}\dot{\omega}_{i/\phi} \\ m_i & \dot{v}_{C_i/\phi} \end{pmatrix} + \begin{pmatrix} \omega_{i/\phi} \times I_{C_i}(\omega_{i/\phi}) \\ \phi \end{pmatrix} =$$

$$\bar{A}_i(q) = \begin{pmatrix} {}^{\phi}I_{C_i} & \phi \\ \phi & m_i\,\mathbb{1}_3 \end{pmatrix}$$

$$= \begin{pmatrix} {}^{\phi}I_{C_i} & \phi \\ \phi & m_i\,\mathbb{1}_3 \end{pmatrix} \begin{pmatrix} \dot{\omega}_{i/\phi} \\ \dot{v}_{C_i/\phi} \end{pmatrix} + \begin{pmatrix} \omega_{i/\phi} \times I_{C_i}(\omega_{i/\phi}) \\ \phi \end{pmatrix}$$

6x6 invertible function of $q \rightarrow \bar{A}_i(q)$

$$\bar{A}_i(q)\left[{}^{\phi}J_{C_i/\phi}\,\ddot{q} + {}^{\phi}\dot{J}_{C_i/\phi}\,\dot{q}\right]$$

linear function of joint vel

some quadratic term related to velocity

???

$$\begin{pmatrix} M^D_{C_i} \\ F^D_{C_i} \end{pmatrix} = \underbrace{\bar{A}_i\,{}^{\phi}J_{C_i/\phi}\,\ddot{q}}_{\text{linear wrt joint accell}} + \underbrace{\left[\bar{A}_i\,\dot{J}_{C_i/\phi}\,\dot{q} + \begin{pmatrix} \phi \\ \omega_{i/\phi} \times I_{C_i}(\omega_{i/\phi}) \end{pmatrix}\right]}_{\bar{B}_i(q,\dot{q})\,\dot{q}}$$

NOTE

$$\begin{pmatrix} \delta\Theta_i \\ \delta C_i \end{pmatrix}^T = \left({}^{\phi}J_{C_i/\phi}\,\delta q\right)^T = \delta q^T\,J^T_{C_i/\phi}$$

$$\delta L_{C_i} = \delta q^T\,{}^{\phi}J^T_{C_i/\phi}\left[\underbrace{\bar{A}_i\,{}^{\phi}J_{C_i/\phi}}_{6\times 6}\,\ddot{q} + \bar{B}_i\,\dot{q} - \begin{pmatrix} M_{C_i} \\ F_{C_i} \end{pmatrix}\right] = \phi$$

$$\left({}^{\phi}J^T_{C_i/\phi}\,\bar{A}_i\,{}^{\phi}J_{C_i/\phi}\right) \triangleq A_i(q)$$

Positive definite symmetric matrix. Except for singular configurations of the robot this is always invertible

$$\left({}^{\phi}J^T_{C_i/\phi}\,\bar{B}_i\right) \triangleq B_i(q,\dot{q})$$

$$\begin{pmatrix} M_{C_i} \\ F_{C_i} \end{pmatrix} = \begin{pmatrix} M^A_{C_i} \\ F^A_{C_i} \end{pmatrix} + \underbrace{\begin{pmatrix} M^R_{C_i} \\ F^R_{C_i} \end{pmatrix}}_{\text{this disappear by considering the joint frictionless}}$$

$$\delta L_{C_i} = \delta q^T\left[A_i\,\ddot{q} + B_i\,\dot{q} - {}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} M^A_{C_i} \\ F^A_{C_i} \end{pmatrix} - {}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} M^R_{C_i} \\ F^R_{C_i} \end{pmatrix}\right]$$

$$= \delta q^T\left[A_i\,\ddot{q} + B_i\,\dot{q} - {}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} {}^{\phi}M^A_{C_i} \\ {}^{\phi}F^A_{C_i} \end{pmatrix}\right] - \underbrace{\delta q^T\,{}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} M^R_{C_i} \\ F^R_{C_i} \end{pmatrix}}_{\delta\Lambda_{C_i}} = \phi$$

④ $$\delta L^{TOT} = \sum_{i=1}^{n}\delta L_{C_i} = \sum_{i=1}^{n}\delta q^T\left[A_i\,\ddot{q} + B_i\,\dot{q} - {}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} {}^{\phi}M^A_{C_i} \\ {}^{\phi}F^A_{C_i} \end{pmatrix}\right] - \delta q^T\left[{}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} M^R_{C_i} \\ F^R_{C_i} \end{pmatrix}\right]$$

$$= \delta q^T\sum_{i=1}^{n}\left[A_i\,\ddot{q} + B_i\,\dot{q} - {}^{\phi}J^T_{C_i/\phi}\begin{pmatrix} M^A_{C_i} \\ F^A_{C_i} \end{pmatrix}\right] - \underbrace{\sum_{i=1}^{n}\delta\Lambda_i}_{=\phi} \quad \text{vanish of forces and torques}$$

(virtual work principle) → total virtual work and we proved that this is ϕ

$$\delta L^{TOT} = \delta q^T \left[\left(\sum_{i=1}^{n} A_i \right) \ddot{q} + \left(\sum_{i=1}^{n} B_i(q, \dot{q}) \right) \dot{q} - \sum_{i=1}^{n} \phi J_{C_i}^T \begin{pmatrix} M_{C_i}^A \\ F_{C_i}^A \end{pmatrix} \right] = \phi$$

$n \times n$

- Assume reference coincident to joint
- Assume we can neglect gravity out of the term.

$$\tau + \hat{\tau}^{EXT} - C(q) \qquad \sum_{i=1}^{n} \phi J_{C_i}^T \begin{pmatrix} M_{C_i}^{EXT} \\ F_{C_i}^{EXT} \end{pmatrix}$$

$$\sum_{i=1}^{n} \phi J_{C_i}^T \begin{pmatrix} \phi \\ m_i g \end{pmatrix}$$

$$\begin{cases} M_{C_i}^A = M_{C_i}^{EXT} \\ F_{C_i}^A = F_{C_i}^{EXT} + m_i g \end{cases}$$

$A(q) \triangleq \sum_{i=1}^{n} A_i(q)$

$n \times n$ symmetric positive definite

$B(q, \dot{q}) \triangleq \sum_{i=1}^{n} B_i(q, \dot{q})$

$n \times n$ symmetric matrix

(5)

$$\delta L^{TOT} = \delta q^T \left[A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) - \tau - \sum_{i=1}^{n} \phi J_{C_i}^T \begin{pmatrix} M_{C_i}^{EXT} \\ F_{C_i}^{EXT} \end{pmatrix} \right] = \phi$$

δq: virtual joint displacement vector

Since δq^T is arbitrary we have that the bracket must be equal to ϕ.

(6) $A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) = \tau + \tau^{EXT}$

non linear, not in normal form

NOTE

$n \times 1$

$$\ddot{q} = A^{-1}(q)\left[\tau + \tau^{EXT} - B(q,\dot{q})\dot{q} - C(q)\right]$$

Somebody writes this as $\ddot{q} = f(q, \dot{q}, \tau, \tau^{EXT})$

Direct Dynamic Problem and Inverse Dynamic Problem

Direct is the simulation, meaning that you set the velocity and initial configuration and fix the inputs (τ) and compute the overall joint movement

Dynamics can be seen as transforming

$$\tau(t), \tau^{ext}(t), q(t_0), \dot{q}(t_0) \rightarrow \begin{cases} q(t) \\ \dot{q}(t) \\ \ddot{q}(t) \end{cases} \forall t$$

Another instance is going to solve the forward dynamic model

the inverse uses given the motion in cartesian space, I want to know the joint torques giving that motion.

Let's assume that we know $q(t)$, $\dot{q}(t)$, $\ddot{q}(t)$ and τ^{Ext} we can calculate τ

We can compute

$$\ddot{q} = A^{-1}\left[\tau + \tau^{Ext} + B(q,\dot{q})\dot{q} - C(q)\right]$$

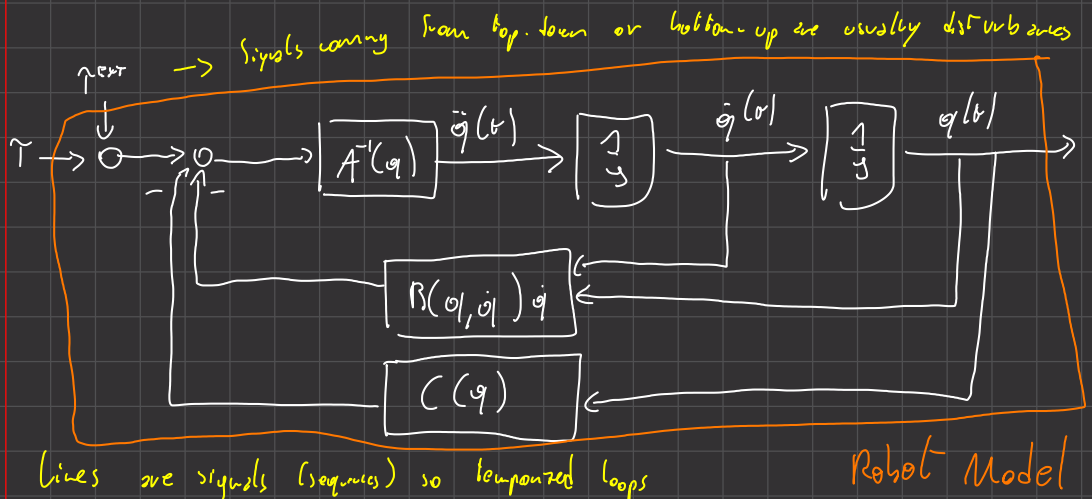
Now we can integrate

Getting $\dot{q}$ and q

$$\dot{q}(t) = \int_0^t \ddot{q}(\ell)\, d\ell + \dot{q}(0)$$

$$q(t) = \int_0^t \dot{q}(\ell)\, d\ell + q(0)$$

Block diagram

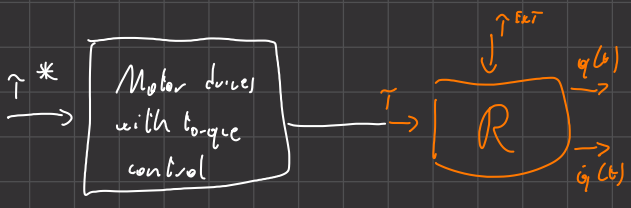


Lines are signals (sequences) so temporized loops

Robot Model

Play and Friction are not taken into account but usually described at control level.

The Robot Model can be described as



Fundamental properties

- $A(q) \in \mathbb{R}^{n \times n}$
- $A = A^T$
- $A > 0$

$\Rightarrow$

- A^{-1} exists
- $\exists P : P^T A P = diag(\lambda_1 \dots \lambda_n) \quad \lambda_i > 0$
- $P \cdot P^T = \mathbb{1}_n$ → This is a kind of nxn rotation matrix

- $\lambda_{min} \|x\|^2 \leq \underbrace{x^T A x}_{\text{quadratic form}} \leq \lambda_{max} \|x\|^2$ → way of giving the dim of the matrix
- $\frac{1}{2} \dot{A}(q) - B(q, \dot{q}) = N$ and $N = -N^T$ (skew symmetric)
- $\|B(q, \dot{q})\| \leq b_0 \|\dot{q}\|$ (b_0 constant) or $\|B(q, \dot{q})\dot{q}\| \leq b_0 \|\dot{q}\|^2$
  this is not growing faster than a quadratic
- $\|C(q)\| = \sqrt{C^T C} \leq c_0$
  → constant (May be not true for long joints)

Note

$F \in \mathbb{R}^{m \times n}$ (induced Norm)?

$\|F\| = ?$

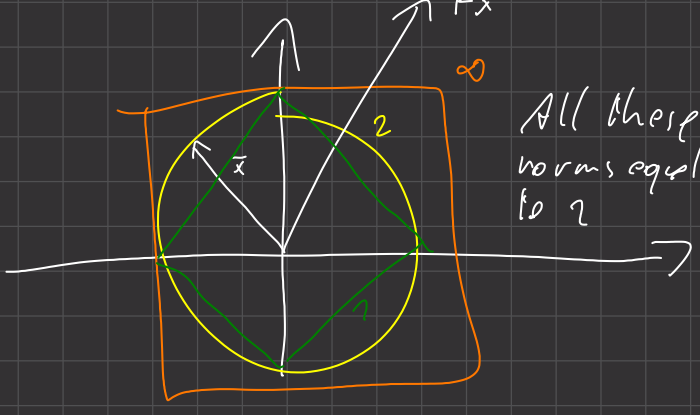
Let's start from a vector: $\underline{x} \in \mathbb{R}^{m \times 1}$

- $\|\underline{x}\|_2 = \sqrt{x^T x}$ $\quad \|x\|_p = \sqrt[p]{\sum_{i=1}^{m} x_i^p}$
- $\|x\|_1 = \sum_{i=1}^{m} |x_i|$
- $\|x\|_\infty = \max_i |x_i|$

$\|x\|_\infty \leq \|x\|_p \leq \|x\|_2 \leq \|x\|_1$

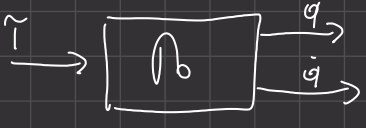
$\|F\| = \max_{\|\underline{x}\| = 1} \|F\underline{x}\|$

unit vectors



All these norms equal to 2

8/04/24 Computational Robotics



$$A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) = \tau$$

τ: quadratic vector

$\ddot{q}$: 2nd order differential equation

A: non symmetric positive definite matrix

Remember

$\tau \Leftrightarrow u(t)$ → it's the control vector

$$\begin{bmatrix} q \\ \dot{q} \end{bmatrix} \Leftrightarrow x(t) \qquad \dot{x}(t) = f(x,u)$$

$$\boxed{\dot{x}(t) = g(x) + H(x)u}$$ → forma that you find in some books

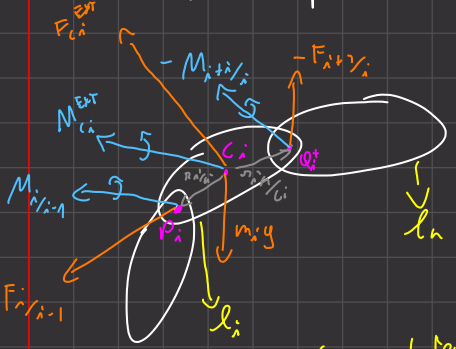
$$\dot{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \dot{x}_1(t) = x_2(t)$$

$$\dot{x}_2(t) = \dots + A^{-1}u$$

Equilibrium conditions

We want to solve the inverse dynamics problem.

Inverse dynamic problem such that $\dot{q}(t) = \ddot{q}(t) = 0$



We must have that:

$$\begin{cases} M_{ci} = 0 \\ F_{ci} = 0 \end{cases}$$ resultant of all forces and torques expressed wrt the cm of the robot.

counter forces and torques

$$\begin{cases} F_{i/i-1} \ominus F_{i+1/i} + m_i g + F_{ci}^{EXT} = 0 \\ M_{i/i-1} \ominus M_{i+1/i} + M_{ci}^{EXT} + r_{i/ci} \times F_{i/i-1} - \left(r_{i+1/ci} \times F_{i+1/i}\right) = 0 \end{cases}$$

This is a set of recursive equations

↓ ↓ ↓

NOTE

$$\underline{f}_{i/i-1} = \underline{f}_{i+1/i} - m_i \underline{g} - \underline{F}_{C_i}^{EXT}$$

$$\underline{m}_{i/i-1} = \underline{m}_{i+1/i} - \underline{m}_{C_i}^{EXT} - \left(\underline{r}_{i/C_i} \times \underline{f}_{i/i-1}\right) + \underline{r}_{i+1/C_i} \times \underline{F}_{i+1/i}$$

} input output relations

Think of them as lines of code

NOTE:

let's assume $i = n$

$$\underline{F}_{n/n-1} = -m_n g - F_{C_n}^{EXT}$$ → force of the object while I am gripping

$$M_{n/n-1} = -M_{C_n}^{EXT} - \left(r_{n/C_n} \times F_{n/n-1}\right)$$

All of this note is a backward recursion

$$F_{i/i-1} = \phi_{i/i-1} + \begin{cases} \phi & \Gamma = R \\ \text{scalar} \cdot \hat{K}_i & \Gamma = P \end{cases}$$

$\phi_{i/i-1}$: reaction force keeping the robot together

$$M_{i/i-1} = \psi_{i/i-1} + \begin{cases} \tau_i \hat{K}_i & \Gamma_i = R \\ \phi & \Gamma_i = P \end{cases}$$

$\psi_{i/i-1}$: reaction torque

We assumed no friction so ϕ / ψ has no component on invariant axis

thus we can say:

$$\tau_i = \begin{cases} F_{i/i-1} \hat{K}_i & \Gamma = P \\ M_{i/i-1} \hat{K}_i & \Gamma = R \end{cases}$$

$$\phi_{i/i-1} = f_{i/i-1} - \begin{cases} \phi & \Gamma_i = R \\ \tau_i K_i & \Gamma_i = P \end{cases}$$

$$\psi_{i/i-1} = M_{i/i-1} - \begin{cases} \tau_i K_i & \Gamma_i = R \\ \phi & \Gamma_i = P \end{cases}$$

these equations can be iteratively projected on local reference frames (i)

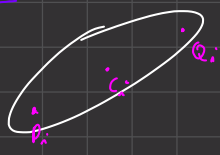
$$\underline{F}_{i/i-1} = \underline{F}_{i+1/i} - m_i \underline{g} - \underline{F}^{Ext}_{Ci}$$

$$\underline{M}_{i/i-1} = \underline{M}_{i+1/i} - \underline{M}^{Ext}_{Ci} - \left(\underline{R}_{i/Ci} \times \underline{F}_{i/i-1}\right) + \left(\underline{r}_{i-1/Ci} \times \underline{F}_{i+1/i}\right)$$

$$\tau_i = \begin{cases} \underline{F}_{i/i-1} \cdot \hat{K}_i & \Gamma = P \\ \underline{M}_{i/i-1} \cdot \hat{K}_i & \Gamma = R \end{cases}$$

↳ in i is a constant and always $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

NOTE



$$\left(Q_i^T - 0\right) = \begin{cases} \left(Q_i - 0\right) & \Gamma_i = R \\ \left(Q_i - 0\right) + R_{i+1} q_{i+1} & \Gamma_i = P \end{cases}$$

Model of generalized forces and torques at the equilibrium ($\ddot{q} = 0$):

$$\tau^{EQ} = c(q)$$

$$\tau^{EQ} = c(q) - \sum_{i=1}^{n} \varphi_i J^T_{Ci/0} \begin{bmatrix} M^{Ext}_{Ci} \\ F^{Ext}_{Ci} \end{bmatrix}$$

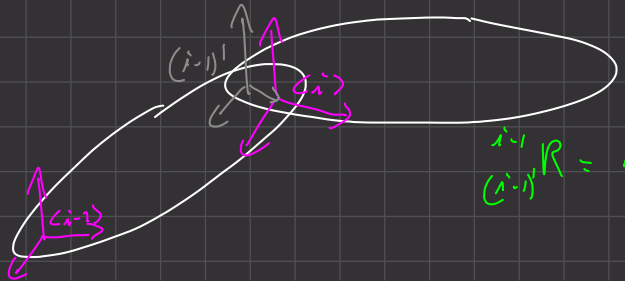
↳ the components of this vector are these components

Inverse dynamic problem (2)

Inputs: $q(t)$, $\dot{q}(t)$, $\ddot{q}(t)$, $\begin{pmatrix} M^{Ext}_{Ci} \\ F^{Ext}_{Ci} \end{pmatrix}$

Output: $\tau(t)$ → control vector

$$\begin{cases} \underbrace{\left(I_{Ci}(\omega_{i/0}) + \omega_{i/0} \times I_{Ci}(\omega_{i/0})\right)}_{\Delta} = M_{Ci} \\ \underbrace{m_i \dot{v}_i}_{D_i} = F_{Ci} \end{cases} \Leftrightarrow \begin{cases} M_{Ci} - \Delta_i = 0 \\ F_{Ci} - D_i = 0 \end{cases}$$



$${}^{i-1}_{(i-1)}R = R_{i-1} \quad \text{const}$$

Software:

// pos. current computation

$$ {}^{i-1}_{\ i}R(q_i) = \begin{cases} R_{i-1}\, R_z(q_i) & \Gamma = R \\ R_{i-1} & \Gamma = P \end{cases} $$

$$ r_{i/i-1} = \left(Q_i^+ - P_i\right) : \begin{cases} (Q_{i-1} - P_{i-1}) & \Gamma_i = R \\ (Q_{i-1} - P_{i-1}) + \hat{K}_i q_i & \Gamma = P \end{cases} $$

$$ r_{i/0} = r_{i-1/0} + r_{i/i-1} $$

$$ r_{ci/0} = \underbrace{r_{i/0}}_{\text{computed}} + \underbrace{r_{ci/i}}_{\text{const in frame } i} $$

// velocity computation

$$ \omega_{i/0} = \omega_{i-1/0} + \begin{cases} K_i \dot{q}_i & \Gamma = R \\ \emptyset & \Gamma = P \end{cases} $$

$$ v_{i/0} = v_{i-1/0} + \begin{cases} (\omega_{i-1/0} \times r_{i/i-1}) & \Gamma = R \\ (\omega_{i-1/0} \times r_{i/i-1}) + \hat{K}_i \dot{q}_i & \Gamma = P \end{cases} $$

$$ v_{ci/0} = v_{i/0} + \omega_{i/0} \times r_{ci/i} $$

// Forward recursion

for $i = 1 \dots n$ do

$$ \dot{\omega}_{i/0} = \dot{\omega}_{i-1/0} + \begin{cases} \underbrace{(\omega_{i-1/0} \times \hat{K}_i)\, \dot{q}_i} + \underbrace{\hat{K}_i \ddot{q}_i} & \Gamma_i = R \\ \emptyset & \Gamma_i = P \end{cases} $$

$$ \dot{v}_{i/0} = \dot{v}_{i-1/0} + \begin{cases} \left(\dot{\omega}_{i-1/0} \times r_{i/i-1}\right) + \omega_{i-1/0} \times \left(\omega_{i-1/0} \times r_{i/i-1}\right) & \Gamma_i = R \\ \left(\dot{\omega}_{i-1/0} \times r_{i/i-1}\right) + \left(\omega_{i-1/0} \times \left(v_{i/i-1} + \omega_{i-1/0} \times r_{i/i-1}\right)\right) + \\ \hat{K}_i \ddot{\theta}_i + \dot{K}_i + \left(\omega_{i/0} \times K_i\right) \dot{q}_i \end{cases} $$

10/04/24

// Forward recursion

for $i = 1 \dots n$ do

// geometry

$${}^{i-1}_{i}R = \dots$$

$${}^{0}_{i}R = {}^{0}_{1}R\,{}^{1}_{2}R \dots {}^{i-1}_{i}R$$

$$r_{i/i-1} = \dots$$

$$r_{i/i-1} = r_{O_{i-1}/i-1} + \begin{cases} 0 & \Gamma = R \\ k_i q_i & \Gamma = P \end{cases}$$

$$r_{i/0} = r_{i-1/0} + r_{i/i-1}$$

$$r_{c_i/0} = r_{i/0} + r_{c_i/i}$$

$$\omega_{i/0} = \omega_{i-1/0} + \begin{cases} k_i \dot{q}_i & \Gamma = R \\ 0 & \Gamma = P \end{cases}$$

$$v_{i/0} = v_{i-1/0} + \left(\omega_{i-1/i} \times r_{i/i-1}\right) + \begin{cases} 0 & \Gamma = R \\ k_i \dot{q}_i & \Gamma = P \end{cases}$$

$$v_{c_i/0} = v_{i/0} + \omega_{i/0} \times r_{c_i/i}$$

// acceleration

$$\dot{\omega}_{i/0} = \dot{\omega}_{i-1/0} + \begin{cases} (\omega_{i-1/0} \times k_i)\,\dot{q}_i + k_i \ddot{q} & \Gamma = R \\ 0 & \Gamma = P \end{cases}$$

$$\dot{v}_{i/0} = \dot{v}_{i-1/0} + \left(\dot{\omega}_{i-1/0} \times r_{i/i-1}\right) + \omega_{i-1/0} \times \begin{cases} \omega_{i-1/0} \times r_{i/i-1} & \Gamma = R \\ \omega_{i-1/0} \times r_{i/i-1} + k_i \dot{q}_i & \end{cases}$$

$$+ \begin{cases} 0 & \Gamma = R \\ \left(\omega_{i/0} \times k_i\right)\dot{q} + k_i \ddot{q}_i & \Gamma = P \end{cases}$$

$$\dot{v}_{c_i/0} = \dot{v}_{i/0} + \dot{\omega}_{i/0} \times r_{c_i/0} + \omega_{i/0} \times \left(\omega_{i/0} \times r_{c_i/i}\right)$$

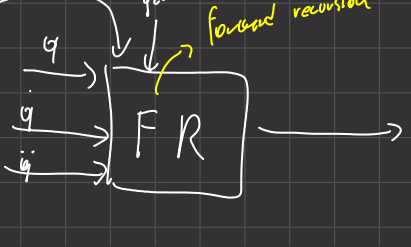
Remember that at the end we want to compute the torques for the IDP

IDP

$$q(t),\ \dot{q}(t),\ \ddot{q}(t),\ \begin{pmatrix} M_{c_i}^{ext} \\ f_{c_i}^{ext} \end{pmatrix},\ \left\{ \begin{array}{l} \text{dynamic} \\ \text{geometric} \\ \text{problem} \end{array} \right\} \longrightarrow \tau$$

the recursion is valid for each time instant.

$t = kT$



N.B.

$$\begin{cases} \omega_{i/0} \\ \dot{\omega}_{i/0} \\ v_{c_i/0} \\ \dot{v}_{c_i/0} \end{cases} \quad \forall i = 1 \dots n$$

$$ {}^{i}I_{Ci}\,{}^{i}\dot{\omega}_{i/\phi} + {}^{i}\left(\omega_{i/\phi} \times I_{Ci}(\omega_{i/\phi})\right) = {}^{i}M_{Ci} $$

$$ m_i\,{}^{i}\dot{v}_{Ci/\phi} = {}^{i}F_{Ci} $$

Backward recursion

$$ \begin{cases} I_{Ci}(\dot{\omega}_{i/\phi}) + \omega_{i/\phi} \times I_{Ci}(\omega_{i/\phi}) = M_{Ci} \\ m_i \dot{v}_{Ci/\phi} = F_{Ci} \end{cases} \qquad \forall i = 1 \dots n $$

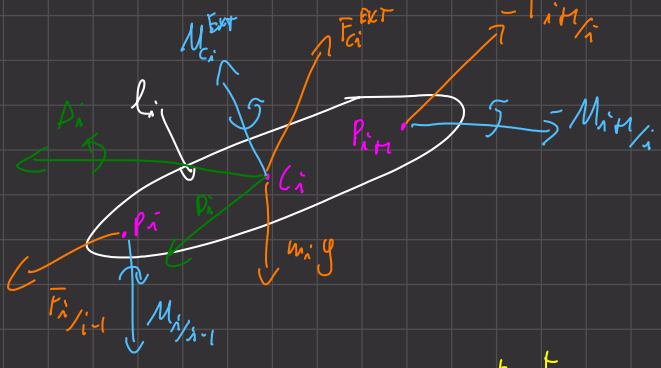
$$ \left.\begin{aligned} \Delta_i &\triangleq I_{Ci}(\dot{\omega}_{i/\phi}) + \omega_{i/\phi} \times I_{Ci}(\omega_{i/\phi}) \\ D_i &\triangleq m_i \dot{v}_{Ci/\phi} \end{aligned}\right\} \text{these are known components} $$

$$ {}^{i}\Delta_i = {}^{i}I_{Ci}\,{}^{i}\dot{\omega}_{i/\phi} + {}^{i}\left[\omega_{i/\phi} \times\right]\,{}^{i}I_{Ci}\,{}^{i}\omega_{i/\phi} $$

$$ {}^{i}D_i = m_i\,{}^{i}\dot{v}_{Ci/\phi} $$

N-E equations

$$ \begin{cases} M_{Ci} - \Delta_i = \phi \\ F_{Ci} - D_i = \phi \end{cases} \quad \text{Dynamic equilibrium conditions.} $$



This uses the static cycle.

Now we have two additional terms.

output

$$ F_{Ci} - D_i = \phi $$

$$ F_{Ci} \triangleq F_{i/i-1} + F_{Ci}^{EXT} - F_{i+1/i} + m_i g \Rightarrow \boxed{F_{i/i-1} + F_{Ci}^{EXT} - F_{i+1/i} + m_i g - D_i = \phi} $$

$$ M_{Ci} - \Delta_i = \phi $$

$$ \Omega_{i+1/Ci} = \Omega_{i+1/i} - \Omega_{Ci/i} $$

$$ M_{i/i-1} - M_{i+1/i} + M_{Ci}^{Ext} + \left(R_{i/Ci} \times F_{i/i-1}\right) - \left(\Omega_{i+1/Ci} \times F_{i+1/i}\right) - \Delta_i = \phi $$

For $i = n, \dots, 1$

① $D_i = \dots$

$\Delta_i = \dots$

$^{i}F_{i+1/i} = {}^{i}_{i+1}R \; {}^{i+1}F_{i+1/i}$ from the forward computation

② $F_{i/i-1} = -F_{ci}^{EXT} + F_{i+1/i} - m_i g + D_i$ (there are n in body n)

$M_{i/i-1} = M_{i+1/i} - M_{ci}^{EXT} - (r_{i/ci} \times F_{i/i-1}) + (r_{i+1/ci} \times F_{i+1/i}) + \Delta_i$

$^{i}M_{i+1/i} = {}^{i}_{i+1}R \; {}^{i+1}M_{i+1/i}$

We assume that $^{i}k_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

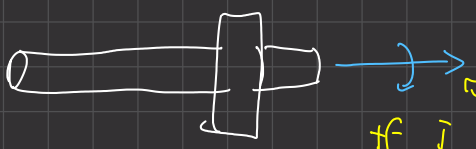
③ $\tau_i = \begin{cases} M_{i/i-1} k_i & r_i = R \\ F_{i/i-1} k_i & r_i = P \end{cases} = \begin{cases} (M_{i/i-1})_z & r_i = R \\ (F_{i/i-1})_z & r_i = P \end{cases}$

④ Optional (where you can compute reaction forces and torques)

$\underline{\phi}_{i/i-1} = \underline{F}_{i/i-1} - \tau_i \underline{\hat{K}}_i$

$\underline{\psi}_{i/i-1} = \underline{M}_{i/i-1} - \tau_i \underline{\hat{K}}_i$

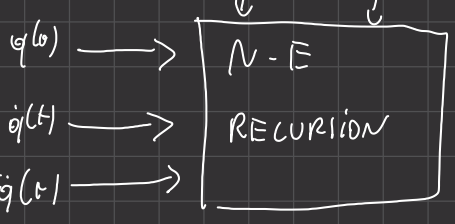
Thumb aboard a joint



Using the model I am only computing this force

If I grasp the object other forces will come into play. The optional is needed to compute those extra forces.

$\{ M_{ci}^{EXT}, F_{ci}^{EXT} \}$ $\{$ geom/dynamic parameterization $\}$



I. D. P.

D.D.P.

$$\tau, \begin{Bmatrix} M_{ci}^{EXT} \\ F_{ci}^{EXT} \end{Bmatrix} \rightarrow \ddot{q}(t) \rightarrow \dot{q}(t) \rightarrow q(t)$$

τ^{EXT}

$$A(q)\,\ddot{q}(t) + B(q,\dot{q})\,\dot{q} + C(q) = \tau + \tau^{EXT}$$

Algorithm to compute A, B, C

(A) Computation of $\boxed{A(q)}$

$q \rightarrow$ current robot configuration

$\dot{q}$ set to $\emptyset \Rightarrow C(q) = \emptyset$

$\tau^{EXT} = \emptyset$

$\ddot{q} = \emptyset \rightarrow B$ goes to $\emptyset$

Now I set $\rightarrow \ddot{q} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

Doing the computation I get:

$$A(q) \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \tau$$

τ in this condition is the first column of A

$\tau = a_1(q)$ given $A(q) = [a_1, a_2 \dots a_n]$

I repeat the computation with $\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ and so on, so forth

Concluding I set 1 by 1 the components of q equal to 1. Let the items that I need $= \emptyset$, then I get:

$$a_i(q) = \tau = A(q) \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ \emptyset \end{bmatrix} \rightarrow \text{i-th component.}$$

(B) Computation of $\boxed{B(q,\dot{q})\,\dot{q}}$

$\dot{q} = \emptyset$

$q = \emptyset$

$\tau^{EXT} = \emptyset$

$$\tau = B(q,\dot{q})\,\dot{q}$$

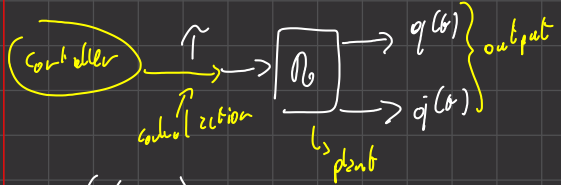
(C) Computation of $\tau(q)$

$$\begin{cases} \dot{q} = 0 \\ \ddot{q} = 0 \\ \tau^{EXT} = 0 \end{cases} \qquad \tau = \tau(q)$$

With these you can solve the direct dynamic problem

28/04/24

Elements of feedback control



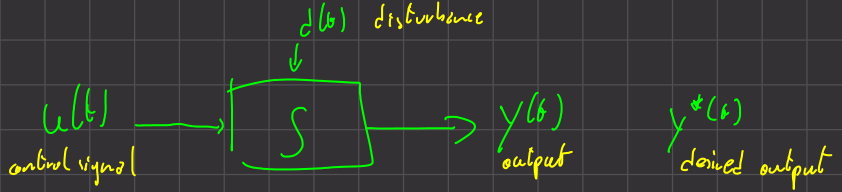
$${}^0T = {}^0T_e(q)$$

$$\begin{pmatrix} {}^0v_{e/0} \\ {}^0\omega_{e/0} \end{pmatrix} = {}^0J_{e/0}(q)\,\dot{q}$$

$q, \dot{q}$ } can be acquired by software.

T is the software output

the model in the robot can be described as: $A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) = \tau$



Control problem

We want to generate a signal $u(t)$ s.t. the output $y(t) \approx y^*(t)$

This because of { disturbances, uncertainties }

We need 4 commitments

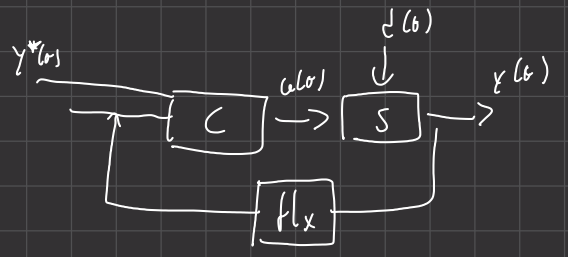
1) Relative stability → settling time. $t_s \leq t_{s_{max}}$

2) Precision $|y - y^*| = e(t) \leq E_{max}$

3) Disturbance Rejection

4) Robustness

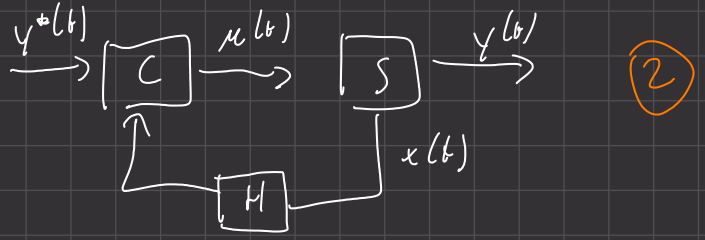
To achieve all of this you need some algorithm



1

What does the controller need to know in order to compute the control signal?

$$\sum: \begin{cases} \dot{x}(t) = f(x, u) \\ y(t) = g(x) \end{cases}$$

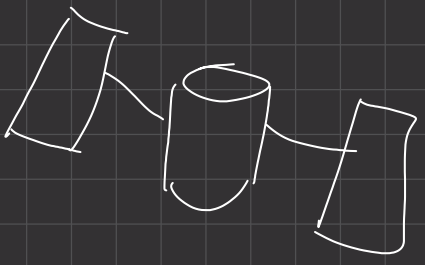


2

1 Is cheaper and simpler than 2

You can move from 1 to 2 by approximation and that is called state space estimation.

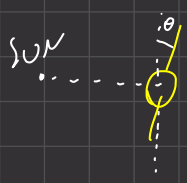
- S.S. observers
- Luenberger observer
- Kalman filter



$y(t) = \theta(t)$

$y^*(t) = \varphi = \theta^*$

$J_z \omega_z = \tau_z$



$\begin{cases} \omega_z = \dot{\theta} \\ \dot{\omega}_z = \ddot{\theta} \end{cases}$ $\quad J_z \ddot{\theta} = \hat{\tau}_z$

$$\hat{\tau}_z = J_z \left[-K_v \dot{\theta} - K_p (\theta - \theta^*) \right]$$

(K_v: feedback gain)

for (;;)

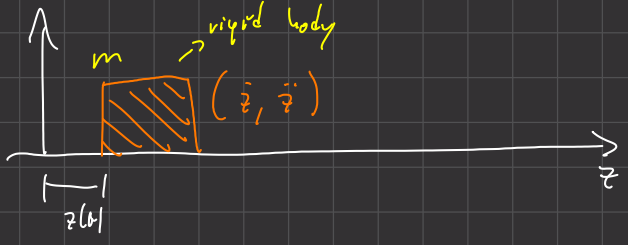
θ = read (orientation_sensor);

$\dot{\theta}$ = read (rate_vel_sensor); → less critical

$\tau_z = \underbrace{J_z}_{(\sim m \cdot l^2)} \left[-K_v \dot{\theta} - K_p \theta \right]$;

write (τ_z, actuator);

<suspend until next sampling time>



$m \ddot{z}(t) = f$

$f \uparrow$

$\tau(t) \leftarrow u(t)$

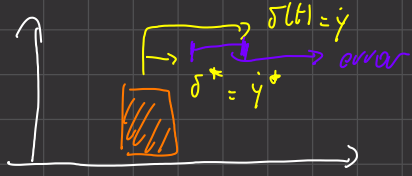
→ control problem #1 (I want to assign a velocity to the rigid body)

$\dot{y}^*(t) = \dot{y}^*$ const (e.g. $20^{-2} m/s$)

$m \ddot{y}(t) = u(t) \Rightarrow \ddot{y}(t) = \underbrace{\frac{1}{m}}\, u(t)$

Open loop dynamic gain of the plant

$e(t) = \dot{y}(t) - \dot{y}^*(t)$



$$\dot{y} = \frac{1}{m} u(t) = \frac{1}{m} \cdot [-K e(t)]$$

$$u(t) = -K e(t)$$

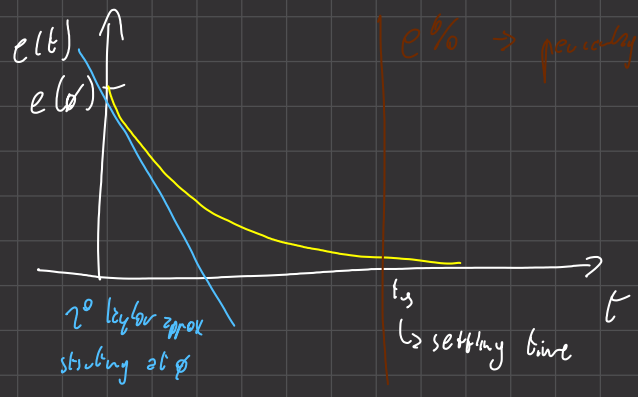
being $\dot{y}^*$ constant $\Rightarrow \ddot{y}^*(t) = 0$

$$\ddot{y}(t) - \ddot{y}(t)^* = \frac{1}{m} [-K e(t)] - \ddot{y}^*(t)$$

$$\dot{e}(t) = -\frac{K}{m} e(t)$$

$$\dot{e}(t) + \frac{K}{m} e(t) = 0$$ Closed loop error dynamics (first order linear diff equation)

$$e(t) = e^{-\frac{K}{m} t} e(0)$$



$e\% \rightarrow$ percentage error $= \frac{e(t)}{e(0)}$

I also define some tolerance.

$$TOL = \begin{cases} 5\% \text{ of } e_{\%}(0) & \text{or} \\ 2\% \text{ of } e_{\%}(0) \\ \vdots \end{cases}$$

$e_{\%}(0) = 1$

$$e_{\%} = e^{-\frac{K}{m} t_s} = TOL$$

$$-\frac{K}{m} t_s = \log(TOL)$$

negative because < 1

$$t_s = -\left(\frac{1}{\frac{K}{m}}\right) \log(TOL)$$

$-\ln(tol) := \begin{cases} 3 & @\ tol := 0.05 \\ 4 & @\ tol := 0.02 \\ & \vdots\ 0.01 \end{cases}$

$e_{\%}(t) \simeq \underset{\uparrow}{1} - \frac{k}{m} t$ Taylor approx

$e_{\%}(0) + \dot{e}_{\%}(0)\, t$

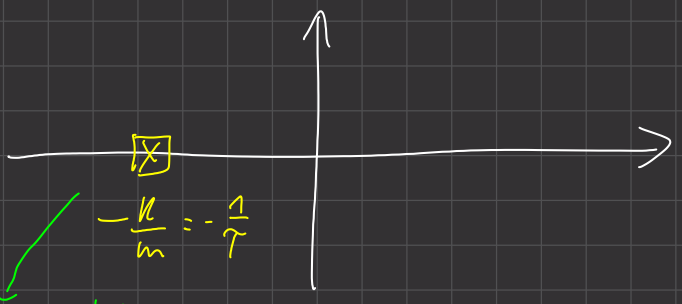
$1 - \frac{k}{m}\tau = 0$ $\boxed{\tau := \frac{1}{\frac{k}{m}}}$ Time constant of the system. (rate of decrease of the error)

$t_y := \begin{cases} 3\tau & @\ tol\ 5\% \\ 4\tau & @\ tol\ 3\% \end{cases}$

I can set the settling time by using the feedback gain.

$\frac{k}{m}$ is the eigenvalue associated with the system

This is also called pole of the system



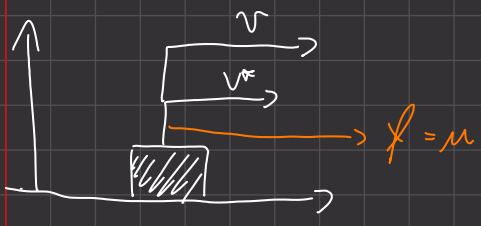
box = closed loop

Closed loop poles we want real system real part.

Rules:

- the further the pole from the origin the faster the transient response
- the larger the k_p the larger the control action.

24/04/24



This system is not isolated but has external interferences such as friction

$$m\ddot{y} = \mu(t)$$

I want to design my control signal

$\delta\dot{y}$ is the error between desired velocity and actual velocity

$$\delta\dot{y} = \dot{y}(t) - \dot{y}^*$$

last lesson this was e

$$u(t) = -K\,\delta\dot{y}$$

I need to understand the effect of the control signal on the system so I apply the control to the system:

Open loop plant model: $m\ddot{y} = u(t)$ $\quad u(t) = -K\delta\dot{y}$

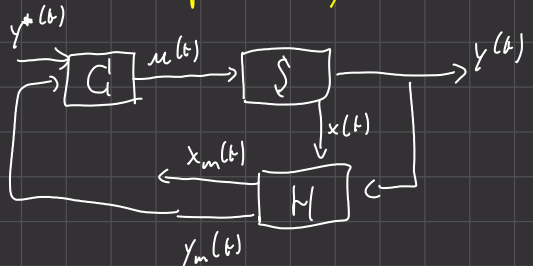
$$\ddot{y}(t) - \ddot{y}(t)^* = -\frac{K}{m}\delta\dot{y}(t) - \ddot{y}^*(t)$$

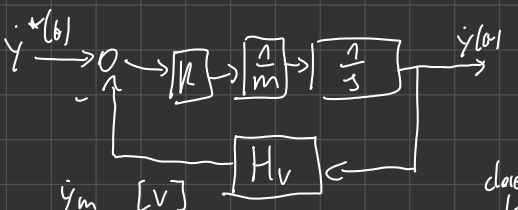
$$\delta\ddot{y}(t) + \frac{K}{m}\delta\dot{y}(t) - \ddot{y}(t) = 0$$

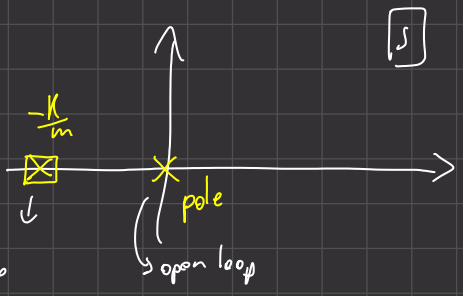
$\ddot{y} = 0$ because constant

$$\delta\ddot{y}(t) + K\,\delta\dot{y}(t) = 0$$

Closed loop error dynamics

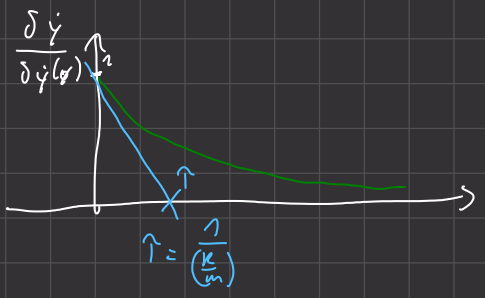






$$\delta\dot{y}(t) = e^{-\frac{K}{m}t}\,\delta\dot{y}(0)$$

error dynamics



Provided that we set $K > 0$ the error will exponentially decay to 0

The larger the K the faster the error will go to 0.

τ represents the linear approximation starting from $t=0$ where the line reaches 0

This function is going to 0 asymptotically. This means at $t \to \infty$.
In practice we set a threshold under which the error will be considered as 0

This parameter is TOL, the tolerance.

$$\lim_{t\to\infty} \delta\dot{y}(t) = 0 \qquad \forall \varepsilon > 0 \quad \exists\, t_\varepsilon > 0 \text{ s.t. } |\delta\dot{y}(t)| < \varepsilon \quad \forall t \geq t_\varepsilon$$

If we now fix $\varepsilon = TOL$ there will be a t_ε that we call:
t_s (settling time). PAGE 3

t_s is proportional to τ: $t_s = \tilde{\tau} \cdot \begin{cases} 3 & @\ TOL = 0.05 \\ 4 & @\ TOL = 0.02 \end{cases}$

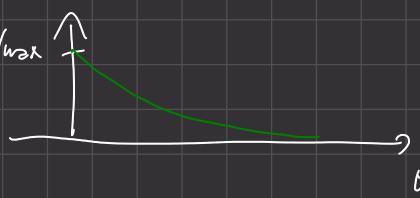
PAGE 1

Assume you have a simple mobile robot and we have simple operation conditions.

EXAMPLE:

$$V_{max} = 1\ m/sec = \frac{20^{-3}\,Km}{\frac{1}{360}\,h} = 3.6\ Km/h$$

Now let's say I am walking and I have want to stop



I have some requirements:

To be considered "stationary" we will have: $V_{safe} \leq 2\ cm/s = 0.02\ m/s$

requirement

$$\frac{v_{sole}}{v_{max}} = \frac{0.02\ m/s}{1\ m/s} = 0.02\ \text{rot}$$

Other requirement will be $t_s \leq 2\ sec$ (requirement)

$$t_s = 4\tau \quad \Rightarrow \quad \tau = \frac{t_s}{4} \leq \frac{2}{4} = \frac{1}{2} \qquad \tau = \frac{1}{\left(\frac{k}{m}\right)} \leq \frac{1}{2}$$

Assuming the mass to be known we now just have to solve for k.

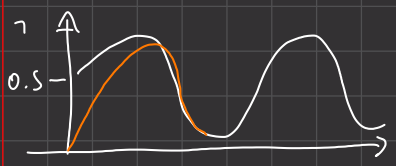
$$k \geq 2m$$

Now the synthesis is complete

Generic rules of thumb

- If you increase k you have
  - Faster response
  - higher accuracy
  - better disturbance rejection
  - better robustness

In case of digital controllers this will cause instability



I may decide that $\dot{y}^*(t) = 0.5\left[1 + \sin(\omega_0 t)\right]$

Note

$0.5 = 0.5\ \underbrace{xxxx}_{\text{unknown numbers}}$

$\frac{1}{2} = 0.5000\ldots$ is guaranteed to be 0s

I wish I could remember pi exactly

3,1415926

$\ddot{y}(t) = \frac{1}{m} u(t)$ ultra-simplification for simplicity (this is our "true" model of the system)

This would be valid $\ddot{y}^*(t) = \frac{1}{m} u^*(t) \Rightarrow u^*(t) = m\,\ddot{y}^*(t)$ (definition of the inverse dynamic problem)

Here we are introducing uncertainties. For example how the uncertainty is the mass.

So I redefine the error:

$\delta\dot{y}(t) = \dot{y}(t) - \dot{y}^*(t)$

$\delta \ddot{y}(t) = \ddot{y}(t) - \ddot{y}^*(t)$ → this can be computed from $y^*(t)$

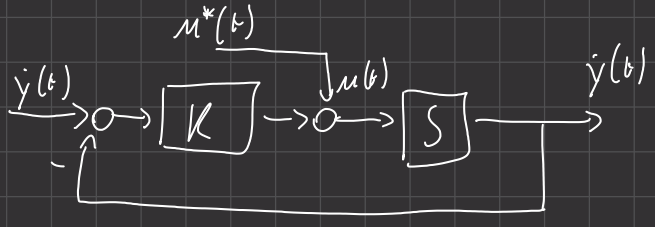
$\delta \ddot{y}(t) = \frac{1}{m} u(t) - \frac{1}{m} u^*(t) = \frac{1}{m} [u(t) - u^*(t)]$

$u(t) = -K \delta \dot{y}(t) + u^*(t)$ (In the previous example was $u(t) = -K \delta \dot{y}$)

$\delta \ddot{y}(t) = \frac{1}{m} [-K \delta \dot{y}(t) + u^*(t) - u^*(t)]$

$\delta \ddot{y}(t) + \frac{K}{m} \delta \dot{y}(t) = 0$

So I can track any differentiable control signal $y^*(t)$



This is a feedback loop with feed forward compensation.

The feed forward is the key element to ensure the following of the response.

— — — — — — — —

Going back to the mass consideration we can say:

$\hat{m} = m - \delta m$

$\hat{m}$: nominal parameter; m: real; δm: uncertainty

$\delta \ddot{y}(t) = \frac{1}{m} [-K \delta \dot{y}(t) + \underbrace{\hat{m} \ddot{y}^*(t)}_{u^* \text{ that I am using in my software.}} - \underbrace{m \ddot{y}^*(t)}_{\text{true } u^*}]$, with $\hat{m} = m - \delta m$

$\delta \ddot{y}(t) = -\frac{K}{m} \delta \dot{y}(t) - \frac{\delta m}{m} \delta \ddot{y}^*(t)$

$\delta \ddot{y}(t) + \frac{K}{m} \delta \dot{y}(t) = -\frac{\delta m}{m} \delta \ddot{y}^*(t)$ → if $\delta m = 0$ then we get the same equation as before

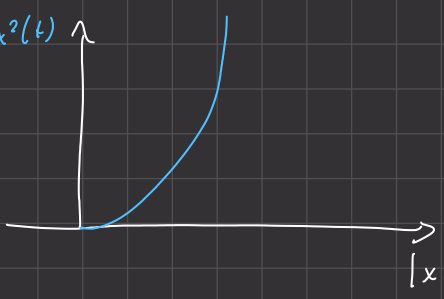
Theorem (BIBO stability)

If a LTI system is asymptotically stable then it's BIBO

Lyapunov based analysis

$\dot{x}(t) = ax(t) + bu(t)$ -> these are scalar but could be extended to vectors.

$V(x) = \frac{1}{2} x^2(t)$



The origin is asymptotically stable if any solution of $V(x)$ will go to 0

Let's consider the first derivative of V:

$$\dot{V}(t) = x(t)\dot{x}(t) = x(t)\left[ax(t) + bu(t)\right] \leq 0$$

If I can guarantee that then this will go to 0

$$x(t)\left[ax(t) + bu(t)\right] = ax^2(t) + bu(t)x(t)$$

Let's assume b or $u = 0$, then if $a < 0$ then the system will be asymptotically stable.

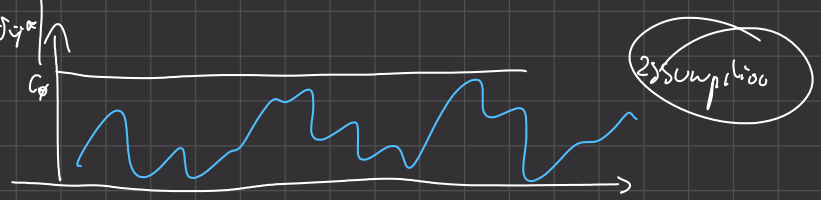
If $a \geq 0$ then I will have to manage b in order to give stability

Let's try to use this method in the previous example:

$$V(\delta\dot{y}) = \frac{1}{2}\delta\dot{y}^2(t)$$

$$\dot{V} = \frac{1}{2} 2\,\delta\dot{y}(t)\cdot\delta\ddot{y}(t) = \delta\dot{y}\left[-\frac{k}{m}\delta\dot{y} - \frac{\delta m}{m}\delta\ddot{y}^*\right]$$

$$\dot{V} = -\frac{k}{m}(\delta\dot{y})^2 - \delta\dot{y}\left(\frac{\delta m}{m}\delta\ddot{y}^*\right)$$

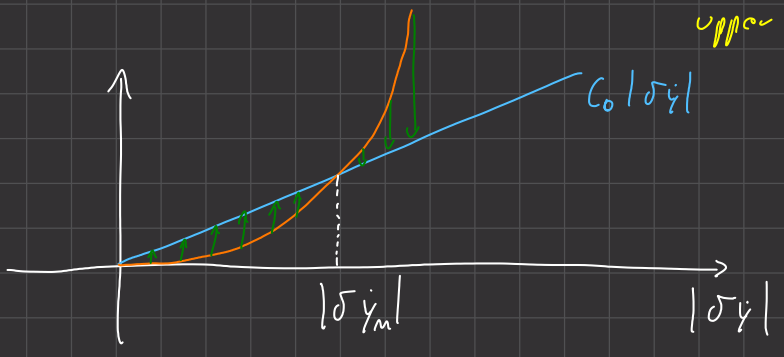


Remember that we want $\dot{V}$ to be <0

I put myself in the worst possible case: $\begin{cases} \left(\frac{\delta m}{m}\,\delta\ddot{y}^*\right) < 0 \\ \delta\dot{y} \gtrless 0 \end{cases}$

$$\dot{V} = -\frac{k}{m}(\delta y)\dot{\delta y} - \dot{\delta y}\left(\frac{\delta m}{m}\,\delta\ddot{y}^*\right) \leq \underbrace{-\frac{k}{m}(\delta\dot{y})^2 + |\delta\dot{y}|\,c < 0}_{\text{upper bound}}$$

parabola, line



If $|\delta\dot{y}| > |\delta\dot{y}_M|$ then V will decrease.

If V decreases then $\delta\dot{y}$ will decrease.

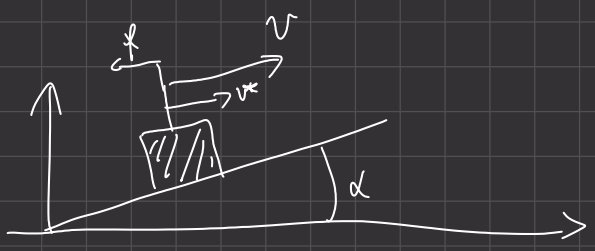
If $|\delta\dot{y}| < |\delta\dot{y}_M|$ the trajectories of the system might have a growing behaviour in absolute value

We can guarantee that if $|\delta\dot{y}| > |\delta\dot{y}_M|$ then the error will decrease.

- $|\delta\dot{y}| = |\delta\dot{y}_M| \Rightarrow -\frac{k}{m}|\delta\dot{y}|^2 + c_0|\delta\dot{y}| = 0$

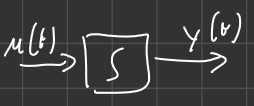
$$|\delta\dot{y}_M| = \frac{c_0}{\frac{k}{m}} = \frac{c_0 m}{k}$$

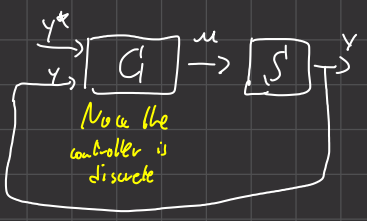
Let's assume now that:



Elements of discrete time control

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$





T is the sample period of the computer

Every T time you get inputted and send out output.

We can say that $u(t)$ has these characteristics:

$$u(t) = u(kT) \triangleq u(k) \qquad \forall t \in [kT, (k+1)T]$$

$$x(t) = e^{A(t-t_0)} x(t_0) + \int_{t_0}^{t} e^{A(t-\tau)} B u(\tau)\, d\tau$$

→ convolution integral

Assume $t = (k+1)T$

$$t_0 = kT$$

$$x(k+1) = e^{AT} x(k) + \int_{kT}^{(k+1)T} e^{A[(k+1)T-\tau]}\, d\tau \; B u(k)$$

$Bu(k)$ out because constant in $[kT, (k+1)T]$

$\int_0^T e^{A\tau}\, d\tau$

$$\Sigma_D = \begin{cases} x(k+1) = A_0 x(k) + B_0 u(k) \\ y(k) = C_0 x(k) \end{cases}$$

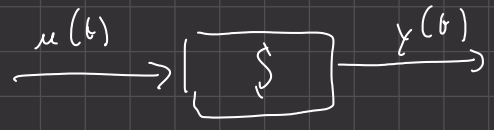
$$\begin{cases} A_0 = e^{AT} \\ B_0 = \int_0^T e^{A\tau}\, d\tau \; B \\ C_0 = C \end{cases}$$

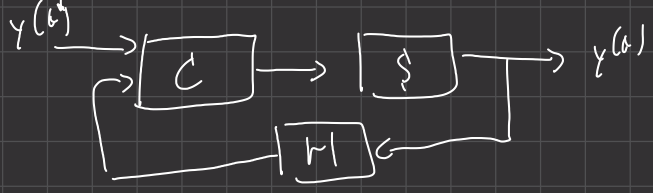
If $u(t)$ in the continuous form is the same as $u_d(t)$ in the sampling instants then the solution are the same.

MATLAB:

$$[A_D, B_D, C_D, D_D] = c2d(A, B, C, D, T, 'ZOH')$$

06/05/24 Control





4 Elements to always be considered

1) (Relative) stability
2) Precision difference between signal and derived one. $\left[\delta\dot{y} = e^{-\frac{\pi}{\omega_n} t}\,\delta y_0\right]$
3) Disturbance rejection
4) Robustness.

If you have a LTI continuous system you can transform it to a discrete system.

S_D LSI (Linear shift invariant)

sampling time, "zero order hold"

$LTI \rightarrow (A, B, C)$
$LSI \rightarrow (A_D, B_D, C_D)$
} methods c2d(..., T, "zoh")

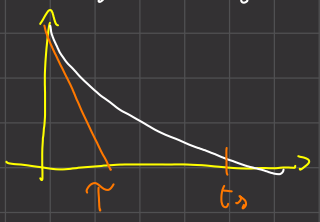
S_C: $\dot{x}(t) = A\,x(t) + B\,u(t)$ differential equations
S_D: $x(k+1) = A_D\,x(k) + B_D\,u(k)$ difference equation
} these are equivalent given that T is respected.

$f_s = \frac{1}{T}\,[Hz]$ (sampling frequency)

NOTE:

$m\ddot{y} = f$ → first order dynamic equation because: $m\frac{d\dot{y}}{dt} = f$

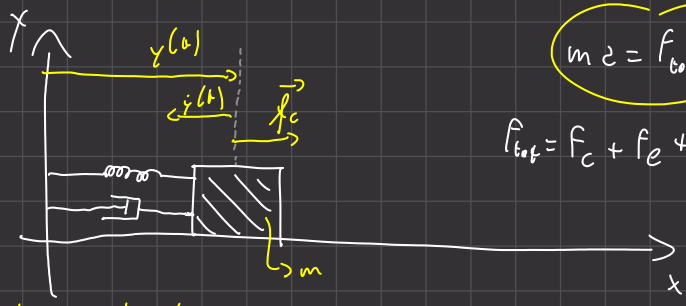
$\lambda = \frac{1}{\tau} = \frac{\gamma}{m}$



$e^{-\frac{\gamma}{m}t}$ is the RESPONSE MODE

A system has as many response modes as order of the system.

Second order LTI systems



$m\ddot{z} = f_{tot}$ this is always valid.

$f_{tot} = f_c + f_e + f_{viscous}$

The inertial mass will be only horizontal.

$f_E = -K_E\, y(t)$ elastic force $K_E > 0$

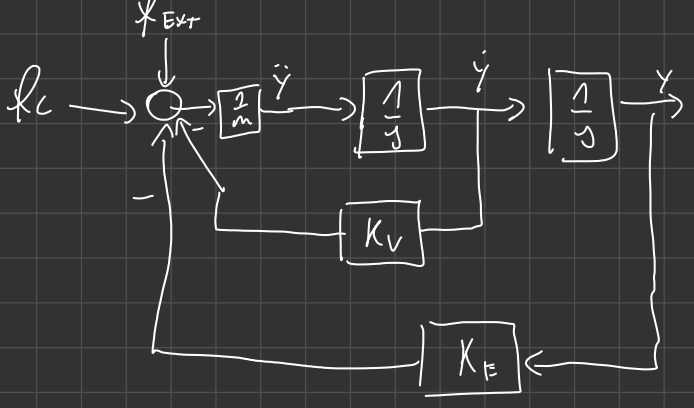
The viscous force goes against the motion of the inertia:

$f_v = -K_V\, \dot{y}(t)$ $K_V > 0$

$m\ddot{y}(t) = f_c - K_E\, y(t) - K_V\, \dot{y}(t)$

$m\ddot{y}(t) + K_V\dot{y}(t) + K_E\, y(t) = f_c(t)$

The action of the damping depends to the position, etc, etc.



$$\ddot{y}(t) + \frac{K_V}{m}\dot{y}(t) + \frac{K_E}{m}y(t) = \underbrace{\frac{1}{m}K_C(t)}_{\omega_n^2\, u(t)}$$ This corresponds on the model above

$$\Sigma_C : x(t) \triangleq \begin{bmatrix} y(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$ 2 state variables – 2 order system ✱

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = -\frac{K_V}{m}x_2(t) - \frac{K_E}{m}x_1(t) + \omega_n^2\, u(t) \end{cases}$$

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -\frac{K_E}{m} & -\frac{K_V}{m} \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \omega^2 m \end{bmatrix} u(t) \\ \qquad\qquad A \text{ (2x2 matrix)} \qquad B \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} x(t) \end{cases}$$

$$\ddot{y}(t) + 2\xi\omega_n\dot{y}(t) + \omega_n^2 y(t) = \omega_n^2 u(t)$$ Standard form for 2nd order LTI

$$\omega_n^2 = \frac{K_E}{m} \Rightarrow \omega_n = \sqrt{\frac{K_E}{m}}$$ → natural frequency of the system

$$2\xi\omega_n = \frac{K_V}{m} \Rightarrow \xi = \frac{K_V}{m}\frac{1}{2\omega_n} = \frac{K_V}{2m\sqrt{mK_E}}$$

↳ this is called Damping coefficient

Stability is related to the eigenvalues of A.

You can compute them with the matrix A or with $\Psi_A(\lambda)$ which is the characteristic polinomial of A.

This can be derived from: $\Psi_A(\lambda) = \lambda^2 + 2\xi\omega_n\lambda + \omega_n^2 = 0$

The roots are the eigenvalues:

$$\lambda_{1,2} = \frac{1}{2}\left[-2\xi\omega_n \pm \sqrt{4\xi^2\omega_n^2 - 4\omega_n^2}\right] =$$

$$= \omega_n\left[-\xi \pm \sqrt{(\xi^2 - 1)}\right]$$

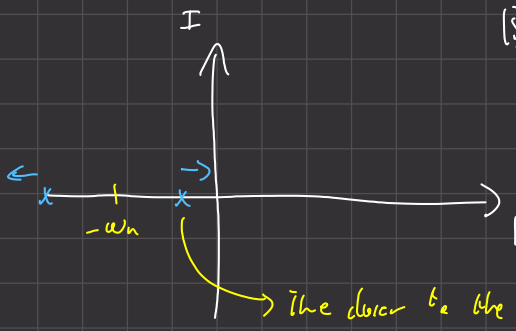
The system is stable if the poles are in the negative of the plane.

Considerations:

1) $\xi > 1 \Rightarrow \lambda_1 < 0, \lambda_2 < 0$ (overdamped system)

2) $0 < \xi < 1 \Rightarrow \lambda_{1,2} = \omega_n \left[-\xi \pm j\sqrt{1-\xi^2} \right]$ (underdamped system)

3) $\xi = 1 \Rightarrow \lambda_1 = \lambda_2 = -\omega_n$ (critically damped system)



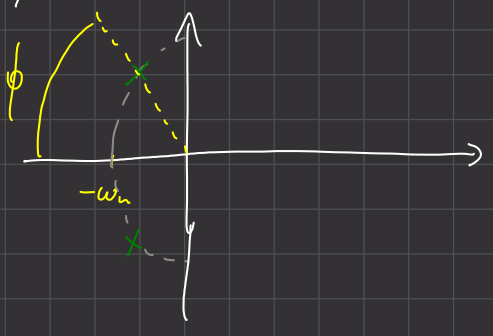
[5]

the closer to the origin the slower the response

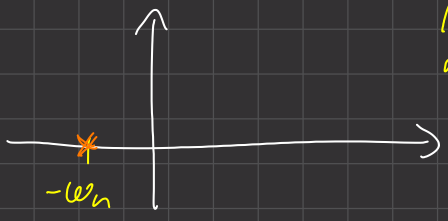
the arrows refer to the increase of ω_n

the relation between the two must be 5 to apply this.

When the poles are very far then you can approximate to 1st order system, called first order dominant.

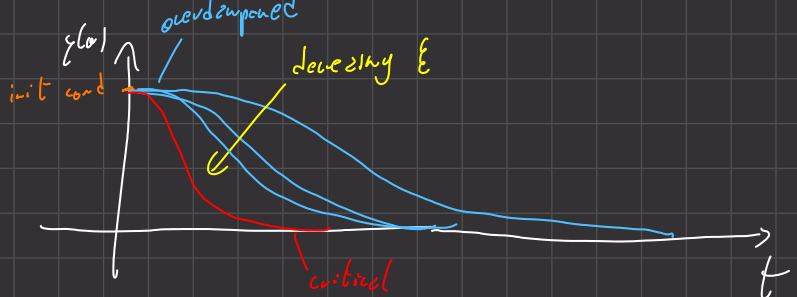


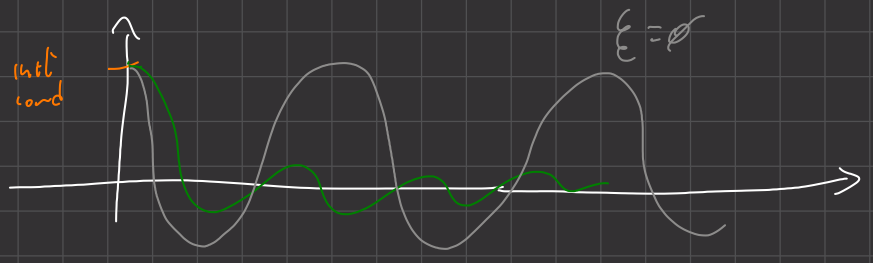
If ϕ is no more than 45° then the overshoot is no more than 5%



Fastest response of the system with real response mode

Step response





$\xi = 1$ may not be robust even if is the best

08/05/24

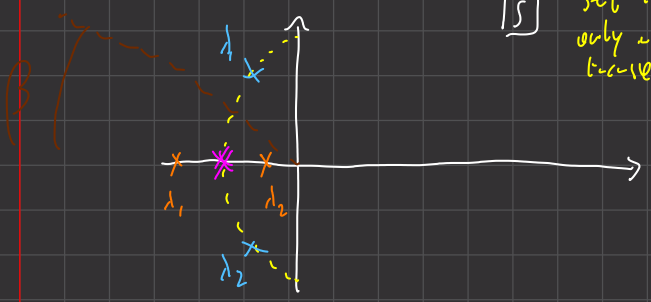
$$\ddot{y}(t) + 2\xi\omega_n \dot{y}(t) + y(t) = \omega_n^2 u(t)$$

$$\ddot{y}(t) + \frac{K_V}{m}\dot{y}(t) + \frac{K_E}{m} y(t) = \frac{1}{m} F_c(t)$$

set to 0 only interested in transient

ω_n natural frequency
ξ damping coefficient

$$\begin{cases} \omega_n = \sqrt{\frac{K_E}{m}} \\ \xi = \frac{1}{2}\frac{K_V}{\omega_0} \end{cases}$$



↳ ω_n define a region to have an idea of where the eigenvalues are

1. $\xi > 1 \Rightarrow \lambda_1, \lambda_2$
2. $\omega c \xi < 1 \Rightarrow \lambda_1 = \bar{\lambda}_2$
3. $\xi = 1 \Rightarrow \lambda_1 = \lambda_2$

↳ This is a limit because any small change causes oscillations.

$\xi = \cos\beta$

$\xi \geq \frac{\sqrt{2}}{2}$ → rule of thumb for good robot actuation.

NOTE

$K_E \Leftrightarrow \omega_n$ $\quad e^{-\xi \omega_n t}$ is the rate at which the response goes to 0 more or less.

Consider the total energy of the system.

remember $x = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$

potential

$$V = \underbrace{\frac{1}{2} m \dot{y}^2(t)}_{K_E} + \underbrace{\frac{1}{2} K_e y^2(t)}_{E_p}$$

→ this is a candidate function

- Always positive defined
- $V(x) > 0 \quad \forall x \neq 0$
- $\dot{V}(x) \leq 0$

$$\dot{V} = \frac{1}{2} m 2\dot{y}(t)\cdot\ddot{y} + \frac{1}{2} K_e 2 y \dot{y}(t) =$$

m here is simplified

$$= \dot{y}(t)\left[-K_V \dot{y} - K_E y\right] + K_E \dot{y} y =$$

$= -K_V \dot{y}^2 \cancel{- K_E y\dot{y}} \cancel{+ K_E y\dot{y}}$

$$\dot{V} = -K_V \dot{y}^2 \leq 0$$ Semidefinite positive function

K_V is the coefficient which contribute to the dissipation of the total energy in the system because the bigger K_V, the bigger $\dot{V}$, the faster the decrease.

Assume $\dot{y}(t)\big|_{t=\bar{t}} = 0$ then $\dot{V}=0$ then the system may be in { equilibrium / some thing else.

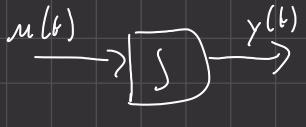
Let's check if the system may be ϕ and to check that substitute that to the system equations.

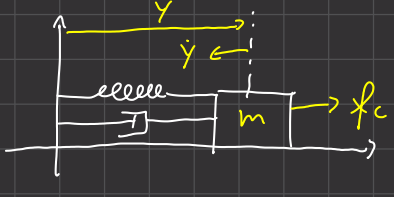
$$m\ddot{y}(\bar{t}) + K_E y(\bar{t}) = 0$$ this is not equilibrium because:

A system is in equilibrium if $\dot{y}$ and $\ddot{y}$ are identically $=0$

The only way of it to be 0 is y to be 0.

In practice this is going to be valid for asymptotically stable systems.





- this system is LTI
- this is asymptotically stable

$u(t) \cong \underbrace{\sum \gamma_i \sin(\omega_i t + \varphi_i)}_{\text{Fourier}}$ the input can be approximated by Fourier.

$y(t) = \sum \gamma_i y_i(t)$ $\quad \sin(\omega_i t) \rightarrow \boxed{\quad} \rightarrow y_i(t)$

Let's assume the input to be just:

$$u(t) = \sin(\omega_0 t)$$

Now we want to know y

There are two terms for y: { transient response (which is 0 because asymptotic stable) / regime response

$$y(t) = \underbrace{y_{TRANS}(t)}_{} + y_{REG}(t)$$

this contributes while the system "is starting". think of the transient

$y_{trans}(t) \to 0$ as $t \to +\infty$ In practice y_{trans} can be neglected once we reached the t_s

fraction of the tolerance ⇒ settling time

$t_s \approx 4\tau$ @ 6% 0.05

let's now consider the Regime Response.

$y_{reg}(t) = A \sin(\omega_0 t + \varphi)$ this is related to the freq response of the system.

a good value for A is 1 because the output tracks well the input

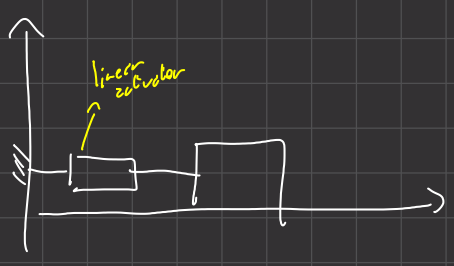
If $A \simeq 1$ then $y_{REG}(t) \sim u(t)$

$A \simeq 1$ holds $\forall \omega_0 \in [0, \omega_B]$ → Band width

$1 \leq \xi$

↳ important role for the digital implementation

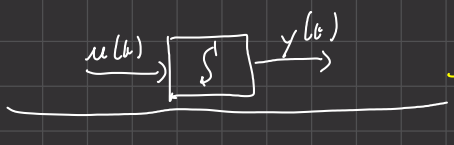
the system can copy any signal almost robustically provided that freq comp of u is smaller the ω_B



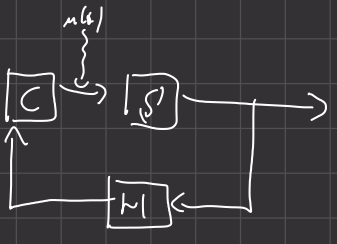
$m_2 = x_c = u(t)$

$m\ddot{y}(t) = u(t) \Leftrightarrow \ddot{y}(t) = \frac{1}{m} u(t)$

Open loop system



→ controlled by →



$u(t)$ → the goal is to design this signal

the control objective is to track the following reference signal:

$$\begin{cases} y^* = y^*(t) \\ \dot{y}^* = \dot{y}^*(t) \\ \ddot{y}^* = \ddot{y}^*(t) \end{cases}$$

Real control problem

let's introduce some important quantities

- $\delta y \triangleq \underbrace{y(t)}_{\text{real}} - \underbrace{y^*(t)}_{\text{desired}}$ ⇒ tracking error

- $\delta \dot{y} \triangleq \dot{y}(t) - \dot{y}^*(t)$
- $\delta \ddot{y} \triangleq \ddot{y}(t) - \ddot{y}^*(t)$

$$u(t) = \underbrace{m}_{\text{param}}\left[\ddot{y}^*(t) - \underbrace{K_V}_{\text{param (damping)}}(\dot{y}(t) - \dot{y}^*(t)) - \underbrace{K_E}_{\text{param (stiffness)}}(y(t) - y^*(t))\right]$$

← the implement this

Now we take u and put it in the state model and you know the closed loop system will behave.

$$\ddot{y}(t) = \frac{1}{\underbrace{m}_{\text{param}}}\left[\ddot{y}^*(t) - \underbrace{K_V}_{\text{param (damping)}}(\dot{y}(t) - \dot{y}^*(t)) - \underbrace{K_E}_{\text{param (stiffness)}}(y(t) - y^*(t))\right]$$

→ this is what we get from the system

$$\left(\ddot{y}(t) - \ddot{y}^*(t)\right) = -\underbrace{K_V}_{\text{param (damping)}}(\dot{y}(t) - \dot{y}^*(t)) - \underbrace{K_E}_{\text{param (stiffness)}}(y(t) - y^*(t))$$

2nd LTI homogeneous Diff equation

$$\delta\ddot{y}(t) + K_V\,\delta\dot{y} + K_E\,\delta y = 0$$

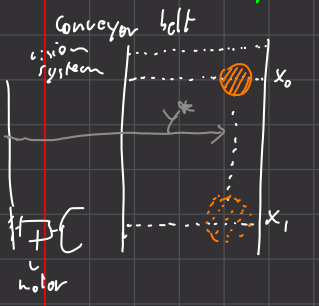
this is equivalent to the previous example equation

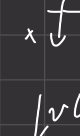
Closed Loop error dynamics

this equation tells me how I choose K_E and K_V

We now can safely set K_E and K_V

Exercise example





$v(t) = \text{const}$

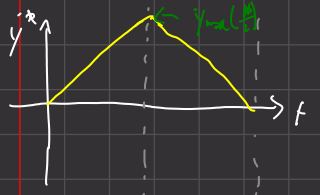
$d = x_1 - x_0$

$d = vt \Rightarrow t^* = \frac{d}{v}$

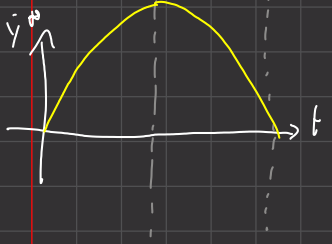
the actual problem is that at time t_0 the system is triggered and I know that at time t^* I will have to pick up the thing

Let's define the reference signals

$$\ddot{y}^*(t) = \begin{cases} +\alpha & 0 < t < \frac{1}{2}t^* \\ -\alpha & \frac{1}{2}t^* < t < t^* \\ 0 & t \geq t^* \end{cases}$$



$$\dot{y}^*(t) = \begin{cases} +\alpha t & 0 < t < \frac{1}{2} t^* \\ -\alpha (t - t^*) & \frac{1}{2} t^* < t < t^* \\ 0 & t \geq t^* \end{cases}$$



$$y^*(t) = \begin{cases} \frac{1}{2} \alpha t^2 & 0 < t < \frac{1}{2} t^* \\ -\frac{1}{2} \alpha \frac{t^{*2}}{4} + \frac{1}{2} \alpha t^{*2} & \frac{1}{2} t^* < t < t^* \\ 0 & t \geq t^* \end{cases}$$

→ We want $\frac{1}{2} \alpha \frac{t^{*2}}{4}$

We need to define a safe settling time.

$t^* = 1$ second

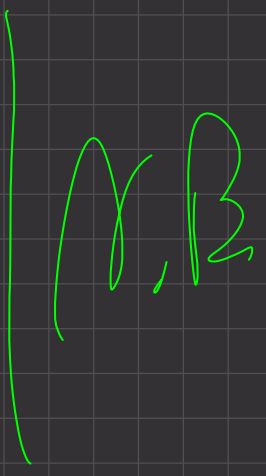
$t_s = 0.75$ @ tol 0.05

$t_s = 4 \tilde{\tau} = 4 / \omega_n$

$\omega_n = \frac{4}{t_s} = \frac{16}{3} \simeq 5.5$

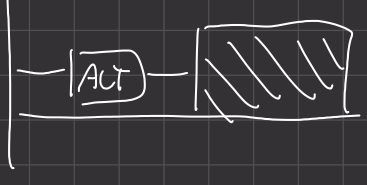
$K_E = \omega_n^2 \Rightarrow (5.5)^2$

$K_V = 2 \xi \omega_n \Big|_{\xi = 1} = 11$



15/05/24

REMEMBER



$u(t)$

$* \quad m\ddot{y} = f_c \Leftrightarrow \ddot{y} = \frac{1}{m} f_c$

$u(t) = m [\ddot{y}^* + K_V \delta \dot{y} - K_E \delta y]$

→ important numbers

$\delta y = y(t) - y^*(t)$

$\delta \dot{y} = \dot{y}(t) - \dot{y}^*(t)$

$\ddot{y}^*(t)$

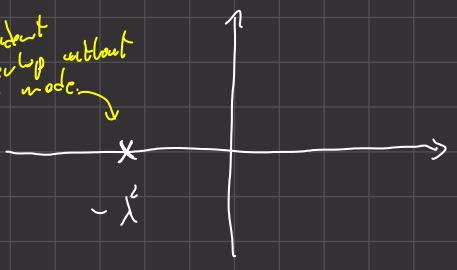
m → mass which is as close as possible to the actual value of the mass.

K_V, K_E } feedback gains of the controller

* the closed formula is $\ddot{\delta y} + K_v \dot{\delta y} + K_p \delta y = 0$

K_v and $K_p > 0$ N.B.

Them being coincident makes them overlap without oscillatory mode.



$$\begin{cases} K_p = \omega_n^2 \\ K_v = 2\omega_n \end{cases} \quad @\ \xi = 1$$

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \triangleq \begin{bmatrix} y \\ \dot{y} \end{bmatrix} \quad \Sigma: \dot{X}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u(t)$$

Σ eigenvalues $= 0$
Not s.t.e. because eigens not < 0

State feedback control law

$$u(t) = -K\,\underline{x}(t) + u^*(t) \qquad K = [K_1, K_2]$$

→ this is what you implement in your software.

If the system is fully controllable then we can always find K_1, K_2 that set the system to reach the objective position.

$$\Sigma: \dot{x}(t) = Ax + bu(t)$$

$$\dot{x}(t) = Ax - bKx + bu^*(t)$$

$$\dot{x}(t) = (A - bK)x + bu^*(t)$$

Our feedback allowed us to transform A into $A - bK$

In matlab you can

$$K = \text{assign}(A, b, [\lambda_1, \lambda_2])$$

In this way you get K needed to have the given eigenvalues with A and b.

proving that the two control laws are the same

A) $u(t) = -K_1 x_1 - K_2 x_2 + u^*(t) = -K_1 y - K_2 \dot{y} + u^*(t)$

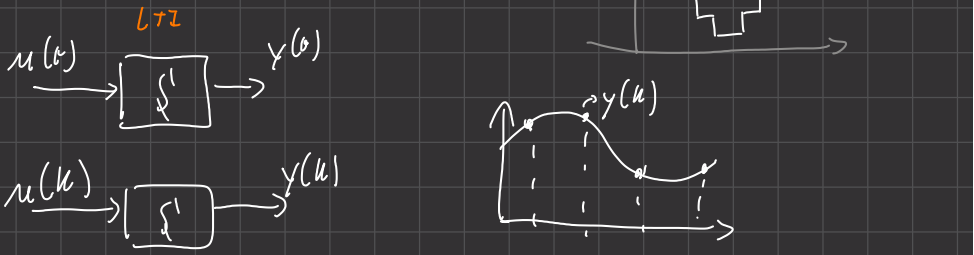
B) $u(t) = -(m K_e) y - m K_v \dot{y} + m \ddot{y}^* + m K_e y^* + m K_v \dot{y}^*$

$K_1 = m K_E$

$K_2 = m K_v$

$u^*(t) = m[\ddot{y}^* + K_v \dot{y}^* + K_E y^*]$

DIGITAL IMPLEMENTATION



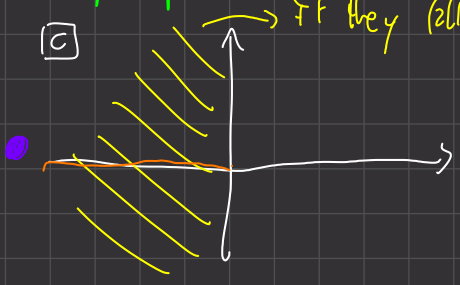
the digital controller is sending u at time k

Remember that there is a direct mapping from continuous to discrete

$$\begin{cases} \dot{S}_C : \dot{x}(t) = A x(t) + B \bar{u}(t) & y(t) = C \underline{x}(t) \\ S_D : x(k+1) = A_D x(k) + B_D u(k) & y(k) = C_D x(x_k) \end{cases}$$

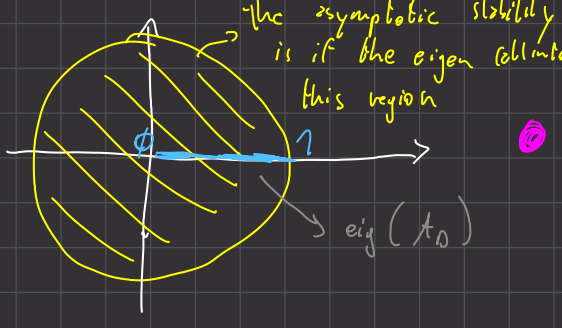
↳ these are equivalent

STability is always what we want and is related by the eigenvalues position in the complex plane



We asked more: relative stability so they should also stay on the x axis

In the discrete case instead of the complex plane s we use z



Having Nyquist then:

$A \to A_D = e^{AT}$

Assume now that the eigenvalues of A are distinct

$\lambda \to \lambda_D = e^{\lambda T}$

these two equations mean that the orange line in the z plane corresponds to the line

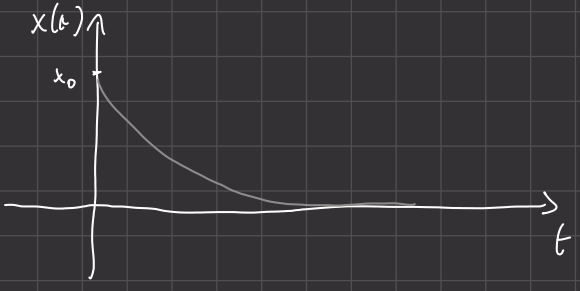
EXAMPLE

$\dot{x}(t) = \lambda x(t) \Rightarrow x(t) = e^{\lambda t} x_0$

$x(k+1) = e^{\lambda T} x(k)$

(eigen) λ assumed negative

I can try to compute this to check if the two match



$x(0) = x_0$

$x(1) = \lambda_D x(0)$

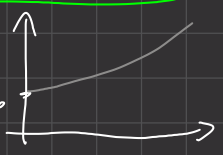
$x(2) = \lambda_D x(1) = \lambda_D^2 x_0$

$x(k) = \lambda_D^k x_0 = e^{\lambda(kT)} x_0$

If λ negative then the system converges to 0

$\lambda < 0 \Rightarrow 0 < \lambda_D < 1$

$\lambda > 0 \Rightarrow$



$\lambda > 0 \Rightarrow \lambda_D > 1$ definitely outside of the unit circle

EXAMPLE

$\dot{x}(t) = u(t)$ open loop $\quad x(t) = x_0$ @ rest

This is an integrator and needs closed loop stabilization

$u(t) = -K x(t)$

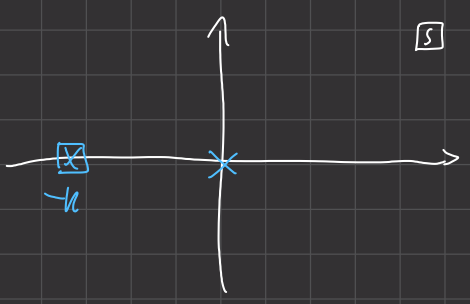
control law

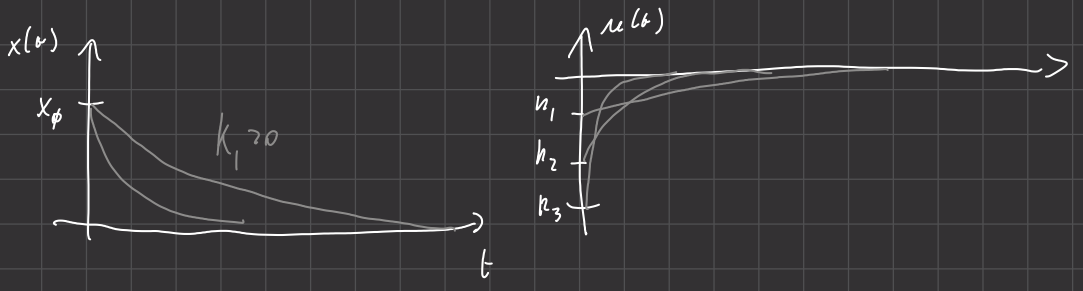
Now I substitute and check if it will mean K

$\dot{x}(t) = -K x(t)$ → closed loop

Provided that $K > 0$ then $-K < 0$

$-K = \lambda$ here





By taking k more negative the response will be faster, but the control signal must be physically feasible.

— — — — — — — — —

Now let's work in discrete time

$\dot{x}(t) = u$

$x(k+1) = x(k) + T\,u(k)$

$\dot{x}(t) = u$ $\quad$ $T = 10^{-2}$ sec (sampling rate)

$u(k) = -\gamma\, x(k)$

↳ this should be a scalar k matrix

Code

```
for (;;)
    x = read_io();
    u = -Γ * x ;
    write_io();
    wait_next_period();
```

To check if it works let's go closed loop.

$x(k+1) = x(k) - T\gamma\, x(k)$

$x(k+1) = \underbrace{(1-\gamma T)}\, x(k)$

↳ this is the eigenvalue

$\lambda_0 = 1 - \gamma T$

$\gamma_1 = 1 \quad \Rightarrow \quad 1 - 10^{-2} = 0.99$ → good => in the segment

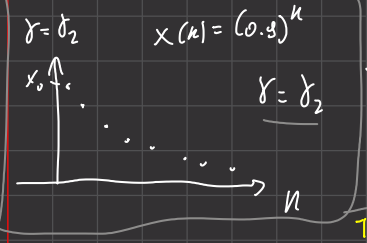
$\gamma_2 = 10 \quad \Rightarrow \quad 1 - 10 \cdot 10^{-2} = 0.9$

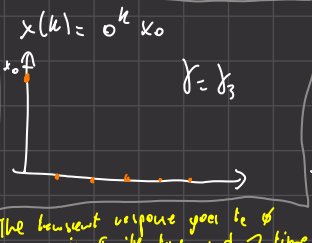
$\gamma_3 = 100 \quad \Rightarrow \quad 1 - 10^2 \cdot 10^{-2} = 0$ → The discrete control diverges from the behaviour of the continuous control. This is called OBAD BRAT CONTROLLER

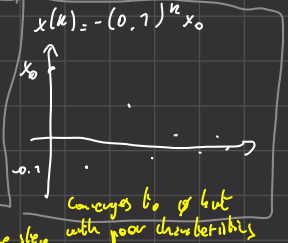
$\gamma_4 = 1000$ (unstable, wrong value) $\Rightarrow \quad 1 - 10^3 \cdot 10^{-2} = (-0.1)^k x_0$

$\gamma_5 = 1000 \quad \Rightarrow \quad = (-9)^k x_0$

Ringing sequence





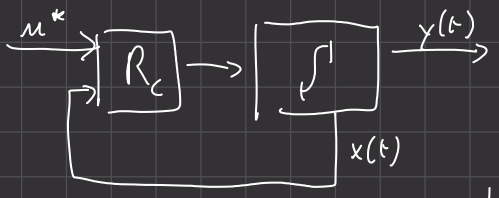


The transient response goes to 0 in finite time and 1 time step

Converges to 0 but with poor controllability

If you thrill again check the software then you see what is happening in 3 because you have a period to wait so until you get that period you are doing nothing. this means that you are overshooting the goal in some sense.

How to chose T



1) Specify the close loop dominant poles (close to positive) $\left(\frac{1}{\tau_{max}}\right)$
   Specify the longest time constant in closed loop

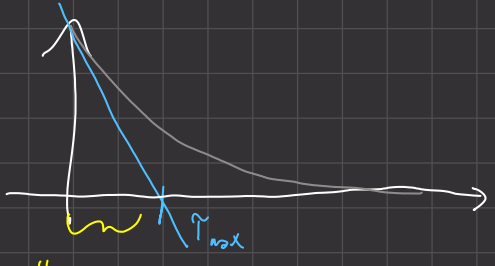
2) Set closed loop bandwidth $\omega_B \simeq \frac{1}{\tau_{max}}$ (Note this is not related to Nyquist)

3) Set the sampling frequency $\omega_s \simeq \frac{2\pi}{T}$ $\omega_s \geq \underbrace{S}_{\text{const}} \omega_B$ $S = 25/30$

$$\frac{2\pi}{T} \geq S \frac{1}{\tau_{max}}$$

$$\tau_{max} \geq \frac{S}{2\pi} T$$

Assume to have this continuous time response:



Here we are expecting 5/6 samples, not too much

4) $\lambda_D = e^{\lambda T}$

5) $\gamma = \text{zssign}(A_D, B_D, [\lambda'_D, \lambda''_D])$

EXAMPLE we want to do a robot drummer capable of hitting 4 times a second

$\omega_B = 4 \times 2\pi$ rad/s

$\omega_s = \frac{2\pi}{T} \geq 30 \times \omega_B = 120 \times 2\pi$ $\quad T \leq \frac{1}{120}$

CLOSED LOOP ROBOT CONTROL

1) Define open loop plant

$$x(t) = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \qquad S: A(q)\,\ddot{q}(t) + B(q,\dot{q})\,\dot{q} + C(q) = \tau$$

GOAL: Tracking of $q^*(t)$, $\dot{q}^*(t)$, $\ddot{q}^*(t)$

SOFTWARE IN ORBEN

τ is the output of the software

$$\tau = A(q)\left[\ddot{q}^*(t) + K_v\,\delta\dot{q}(t) + K_p\,\delta q(t)\right] + B(q,\dot{q})\,\dot{q} + C(q)$$

$$\delta q = q(t) - q^*(t)$$

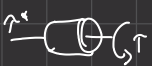
20/09/24

Robot Dynamic Control

Green for generated quantities
Blue is measured

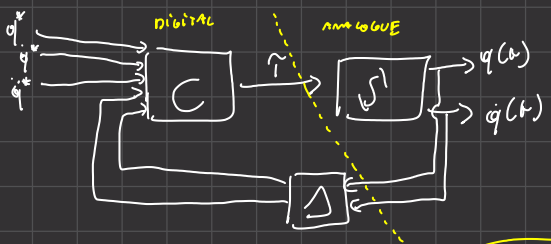
1) Decide which is the system: $S: A(q)\,\ddot{q}(t) + B(q,\dot{q})\,\dot{q} + C(q) = \tau$

2) $q^* = q^*(t)$, $\dot{q}^* = \dot{q}^*(t)$, $\ddot{q}^* = \ddot{q}^*(t)$ (specified by the user)



3) $\tau = A(q)\left[\ddot{q}^* - K_v(\dot{q} - \dot{q}^*) - K_p(q - q^*)\right] + B(q,\dot{q})\,\dot{q} + C(q)$

4)



5) ~~$A(q)\,\ddot{q}(t) + B(q,\dot{q})\,\dot{q} + C(q) =$~~ $A(q)\left[\ddot{q}^*(t) - K_v(\dot{q} - \dot{q}^*) - K_p(q - q^*)\right] + $ ~~$B(q,\dot{q})\,\dot{q} + C(q)$~~

Models of the left one

closed loop dynamic model of the system.

Note that the semplified quantities are not the same thing but they have the same value. The right side is computed, the left is used.
If the models are good enough they can be simplified.

$$\ddot{q}(t) = \ddot{q}(t) - K_v\,\delta\dot{q} - K_p\,\delta q(t)$$

$$\delta\ddot{q}(t) = -K_v\,\delta\dot{q} - K_p\,\delta q(t) \qquad \text{(same as the next one)}$$

$$\ddot{\delta q}(t) + K_v \dot{\delta q} + K_p \delta q(t) = 0$$

K_v and K_p

$K_v, K_p \in \mathbb{R}^{n \times n}$

$K_v = \text{diag}(K_{v_1}, K_{v_2} \dots K_{v_n})$

$K_p = \text{diag}(K_{p_1}, K_{p_2} \dots K_{p_n})$

Set of non coupled diff equation of the form $\ddot{\delta q_i} + K_{v_i} \dot{q}_i + K_{p_i} \delta q_i = 0 \ \forall i$ (IF $K_p, K_v > 0$)

We only need that the involved quantities are bounded and differentiable all the time.

NOTE

$$\tau = A(q)\underbrace{u(t)}_{\text{Generic control signal}} + B(q,\dot{q})\dot{q} + C(q)$$

$$A(q)\ddot{q}(t) + B(q,\dot{q})\dot{q} + C(q) = A(q)u + B(q,\dot{q})\dot{q} + C(q)$$

$$\ddot{q}(t) = u(t)$$ Double integrator (the simplest)

Computing A, B, C for a robot is quite demanding

Let's say that we want to track $q^*(t) = q^* = \text{constant}$.

Let's design a Lyapunov function that is associated with energy.

$$V(q,\dot{q}) = \frac{1}{2}\dot{q}^T A(q)\dot{q} + \frac{1}{2}(q-q^*)^T K_E (q-q^*)$$

↓ Kinetic energy of the Robot ; ↓ Potential energy

$$\dot{V}(q,\dot{q}) = \frac{1}{2}\ddot{q}^T A \dot{q} + \frac{1}{2}\dot{q}^T A \ddot{q} + \frac{1}{2}\dot{q}^T \dot{A} \dot{q} + \frac{1}{2}(q-q^*) K_E (\dot{q}-\dot{q}^*) + \frac{1}{2}(\dot{q}-\dot{q}^*) K_E (q-q^*) =$$

$$= \dot{q}^T A \ddot{q}(t) + \frac{1}{2}\dot{q}^T \dot{A} \dot{q} + \dot{q}^T K_E (q-q^*)$$

$$= \dot{q}^T [\tau - B(q,\dot{q})\dot{q} - C(q)] + \frac{1}{2}\dot{q}^T \dot{A} \dot{q} + \dot{q}^T K_E (q-q^*)$$

$$= \dot{q}^T [\tau - C(q)] + \dot{q}^T \left[\frac{1}{2}\dot{A} - B(q,\dot{q})\right]\dot{q} + \dot{q}^T K_E (q-q^*)$$

Always 0 for robots

$A = A^T > 0$

$K_E = K_E^T > 0$

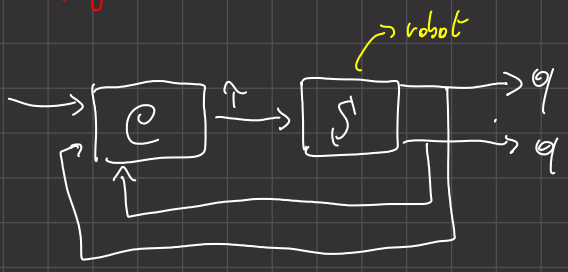
$A\ddot{q} = -B(q,\dot{q})\dot{q} - C(q) + \tau$

$\frac{1}{2}\dot{A} - B(q,\dot{q}) = S$ skew symmetric

$\forall x \in \mathbb{R}^n \ x^T S x = 0$

22/05/24

Set Point Regulation



1) $A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) = \tau$

2) $q^*(t) = q^*$ const

3) $\tau = -K_V \dot{q} - K_E \delta q + C(q)$ (2)

4) Check if z_1 is stable in closed loop

We introduce a Lyapunov function that describes the energy in the system.

$$V(q,\dot{q}) = \frac{1}{2}\dot{q}^T A(q)\dot{q} + \frac{1}{2}(q-q^*)^T K_E (q-q^*) > 0$$

$$\dot{V}(q,\dot{q}) = \dot{q}^T\left[A(q)\ddot{q} + \frac{1}{2}\dot{A}\dot{q}\right] + \dot{q}^T K_E \delta q =$$

$$A(q)\ddot{q} = -B\dot{q} - C + \tau$$

$$= \dot{q}^T\left[\tau - C(q) + K_E \delta q\right] + \dot{q}^T\left[\frac{1}{2}\dot{A} - B(q,\dot{q})\right]\dot{q}^T$$

$S = -S^T \rightarrow = 0$

$\tau - C(q) + K_E \delta q = -K_V \dot{q}$ (we want this to be true)

$\tau = C(q) - K_E \delta q - K_V \dot{q}$

$$= \dot{q}^T\left[-K_V\dot{q} - K_E\delta q + C(q) - C(q) + K_E\delta q\right] = -\dot{q}^T K_V \dot{q}$$

If this > 0 then $V < 0$

$V(q,\dot{q}) > 0$

$\dot{V}(q,\dot{q}) = -\dot{q}^T K_V \dot{q} \leq 0$

Per Assurdo Suppose $\exists\, t_{inf} \bar{t} : \dot{q}(\bar{t}) = 0 \quad q(\bar{t}) = \bar{q} \neq q^*$ (vector of numbers)

$$A(\bar{q})\ddot{q} + B(\bar{q},0)0 + C(\bar{q}) = -K_V 0 - K_E(\bar{q} - q^*) + C(\bar{q})$$

$$A(\bar{q})\ddot{q} = -K_V(\bar{q} - q^*)$$

$\delta q \triangleq q - q^*$

$K_E = K_E^T > 0 \in \mathbb{R}^{n \times n}$

$A(q) = A^T(q) < \mathbb{R}^{n \times n}$

$\frac{1}{2}\dot{A} - B(q,\dot{q}) \triangleq S = -S^T$

$K_V = K_V^T > 0 \in \mathbb{R}^{n \times n}$

$\dot{\bar{q}} = -A^{-1}(\bar{q})\, K_V(\bar{q} - q^*)$

Full Rank square matrix

If will remain stable only if q^* is the eq position so $\delta q = 0$ is the only possibility of stability.

---

(1) $\hat{\tau} = A(q)\left[\ddot{q}^* - K_V \delta\dot{q} - K_E \delta q\right] + B(q,\dot{q})\dot{q} + C(q)$

(2) Is more computational intensive than (1)

Remembering that the closed loop error dynamics for (1) was:

$$\delta\ddot{q} + K_V \delta\dot{q} + K_E \delta q = 0$$

the closed loop error dynamics of (2) is:

$$A(q)\ddot{q} + B(q,\dot{q})\dot{q} + C(q) = -K_V\dot{q} - K_E \delta q + C(q)$$

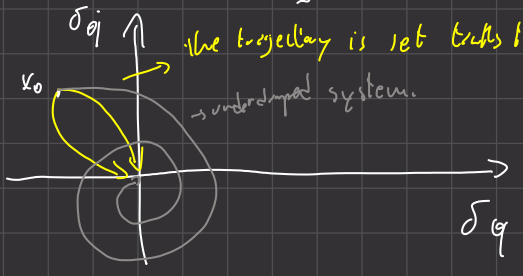
If $q(t) = \text{const}$ then $A(q)\ddot{q} = A(q)\delta\ddot{q}$

$$A(q)\delta\ddot{q} + K_V\dot{q} + K_E \delta q + B(q,\dot{q})\dot{q} = 0$$

This is closed loop error dynamics.

This is very harder than (2) form.

EXAMPLE $\quad x = \begin{bmatrix} \delta q \\ \delta\dot{q} \end{bmatrix}$ (1)



the trajectory is set thanks to K_E or K_V

→ underdamped system.

(2) We cannot say something by simply looking at K_E and K_V

(2) this is very good for control but I have no control over the transient response

(1) We have decoupled differential equations.

Exercise

$$V(q,\dot{q}) = \frac{1}{2}\dot{q}^T A(q)\dot{q}(t) + \frac{1}{2}\delta q^T K_E \delta q + U(q)$$

$$C(q) \triangleq \left(\frac{\partial U}{\partial q}\right)^T$$

this is lower bounded but wants go to ∞ at q^a

$$\dot{V}(q,\dot{q}) = \dot{q}^T\left[\hat{\tau} + K_E \delta q - C(q)\right] + \dot{q}^T C(q) \qquad \frac{d}{dt}U(q) = \frac{\partial U}{\partial q}\dot{q} =$$

$= \dot{q}^T [\tau + K_E \delta q] + \dot{q}^T C(q) - \dot{q}^T C(q)$ $\quad = \dot{q}^T \left(\frac{\partial U}{\partial q}\right)^T = C(q)$

$\tau + K_E \delta q = -K_V \dot{q} \Rightarrow \tau = -K_V \dot{q} - K_E \delta q$

Suppose at time $\bar{t}$ we have $\begin{cases} \dot{q}(\bar{t}) = \emptyset \\ q(\bar{t}) = \bar{q} \neq q^* \end{cases}$

In closed loop we have

$$A(\bar{q})\ddot{q} + C(\bar{q}) = -K_E(\bar{q} - q^*)$$

① $\ddot{q}(t) \neq \emptyset \quad \dot{q} \neq \emptyset \quad \dot{V} < \emptyset$

② $\ddot{q}(\bar{t}) = \emptyset$ then $\underline{C(\bar{q}) = -K_E(\bar{q} - q^*)}$

$$C(q) = -K_E(\bar{q} - q^*)$$

$$(q - q^*) = -K_E^{-1} C(q)$$

Remember that

$$\|C(q)\| \leq C_\emptyset$$

$$\|\bar{q} - q^*\| = \|K_E^{-1} C(q)\| \leq \|K_E^{-1}\| \cdot \|C(\bar{q})\| \leq \|K_E^{-1}\| C_\emptyset =$$

distance

$$= \frac{1}{\min_i K_{E_i}} C_\emptyset$$

note that

$$\|M\|_2 = \max_i \sigma_i(M)$$

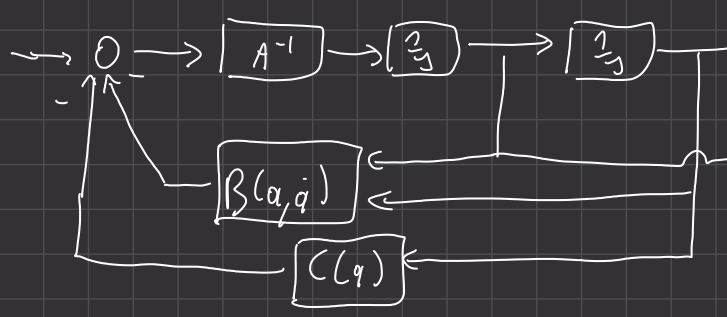
sing value

If square and invertible:

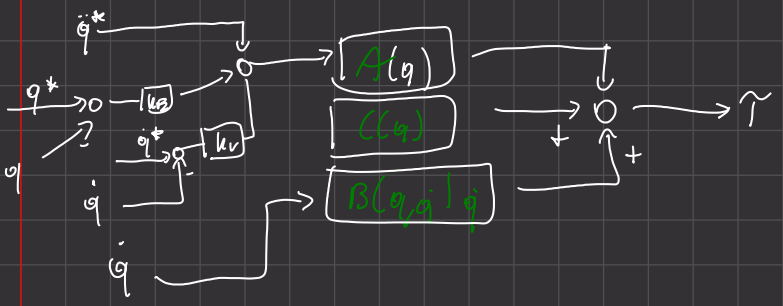
$$\|M^{-1}\|_2 = \frac{1}{\min_i \sigma_i(M)}$$

The simple linear can only reduce the error at the end to be small only if K_E is big.

NOTE $\quad \tau = A(q)u(t) + \underbrace{B(q,\dot{q})\dot{q} + C(q)}_{H(q,\dot{q})}$



this is $\int$



This is C_1

$$u(t) = \ddot{q}^* - K_V\,\delta\dot{q} - K_E\,\delta q$$

feed forward

$$u(t) = \ddot{q}^* - K_V(\dot{q} - \dot{q}^*) - K_E(q - q^*)$$

$$= \ddot{q}^* + K_V(\dot{q}^* - \dot{q}) + K_E(q^* - q)$$