Crash course on Nonlinear Control for Robotics

Daniele Pucci

VVV18
International Winter School on Humanoid Robot Programming
February 10th, 2018 - Santa Margherita Ligure, Italy

Nonlinear control

What does “control” mean?

Different meanings in different languages

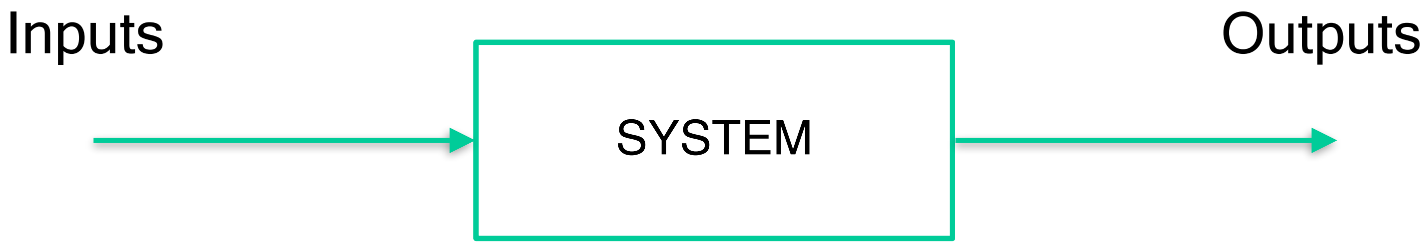
Italian: Daniele, puoi controllare che l’acqua della pasta bolle?

(Daniele, can you check that water for pasta boils?)

So, it may mean check, supervise, etc. depending on languages

What does “control” mean?

For us, it means “impose”



What kind of inputs do we have to apply to impose:

Output = Desired Output ?

Instantaneous and dynamical systems

If the Outputs depend instantaneously on the inputs

Outputs = Outputs(Inputs)

we say that the system is instantaneous

In this case, we may impose

Outputs(Inputs) = Desired Outputs

and find the associated inputs as

Inputs = inverse(Outputs)(Desired Outputs)

Instantaneous and dynamical systems

If the outputs depend on the system state

Outputs = Outputs(state)

and time derivative of the state depend upon the inputs

derivative(state) = function(state,inputs)

we say that the system is dynamical

What is the state of the system?

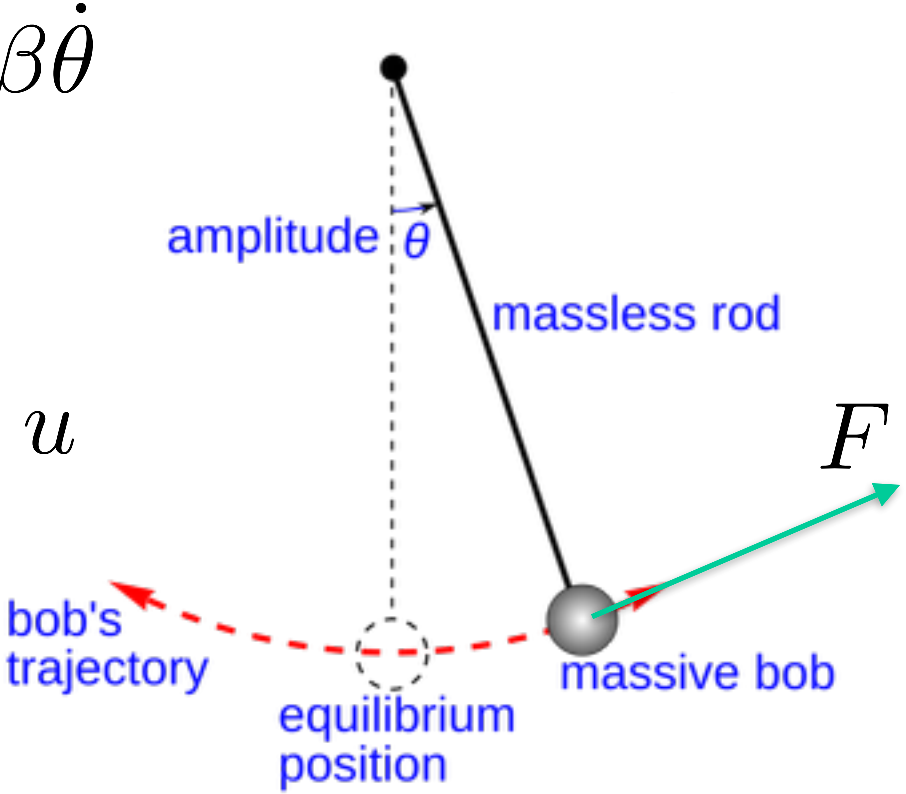
An example of dynamical system

Input = F Friction = $-lm\beta\dot{\theta}$

Output = θ

$$\ddot{\theta} = -\alpha \sin(\theta) - \beta\dot{\theta} + u$$

$$\alpha = \frac{g}{l} \qquad u = \frac{F}{lm}$$



An example of dynamical system

$$x_1 := \theta$$

$$x_2 := \dot{\theta}$$

$$x := \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\dot{x} = \begin{pmatrix} x_2 \\ -\sin(x_1) + u - x_2 \end{pmatrix}$$

Output $\quad y = Cx$

$$C := \begin{pmatrix} 1 & 0 \end{pmatrix}$$

$$\alpha = \beta = 1$$

The dynamical systems

Inputs

SYSTEM

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

$$y = h(x)$$

Outputs

State $x \in \mathbb{R}^n$

Inputs $u \in \mathbb{R}^p$

Outputs $y \in \mathbb{R}^m$

Dynamics $f(\cdot)$

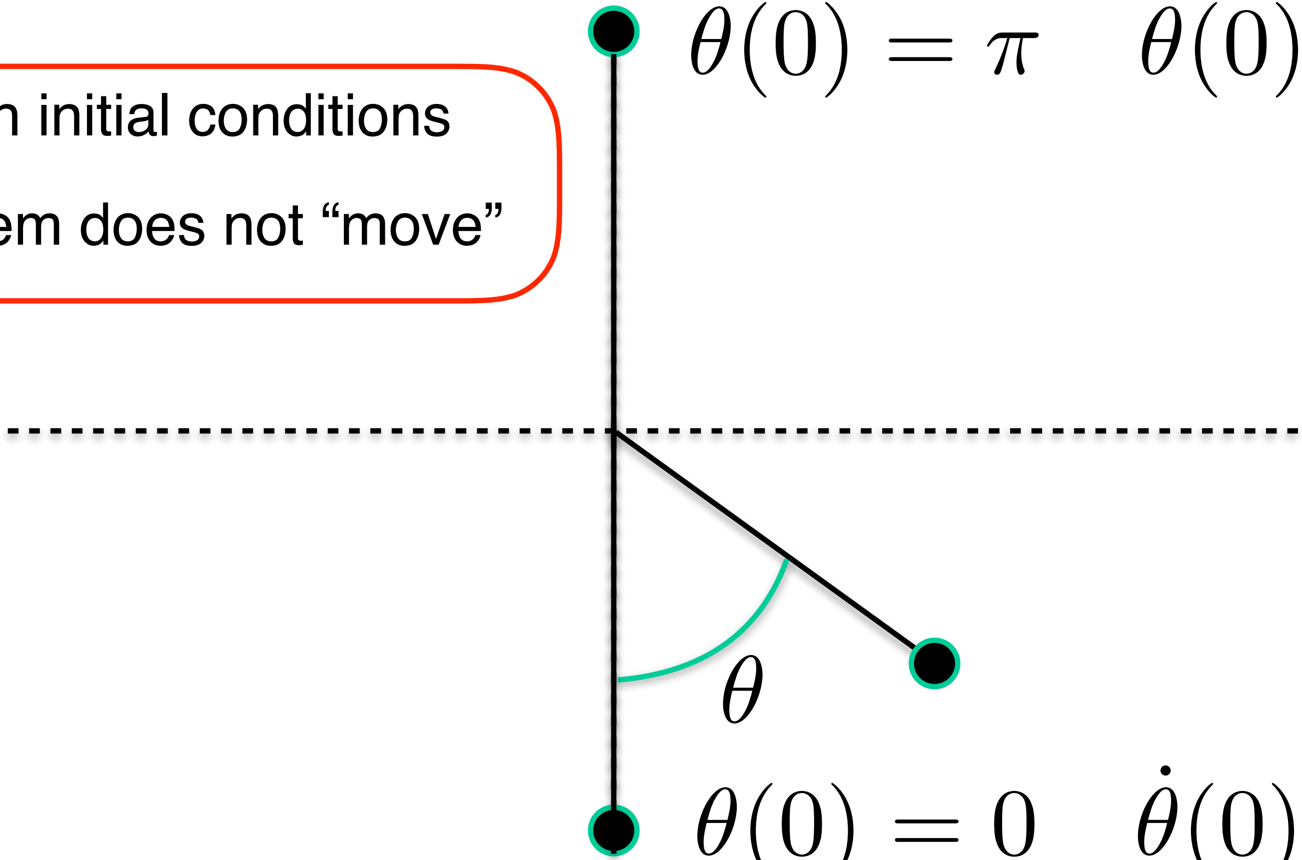
Output function $h(\cdot)$

The concept of equilibria: a case study

At both initial conditions

the system does not "move"

$$\theta(0) = \pi \quad \dot{\theta}(0) = 0$$



$$\theta(0) = 0 \quad \dot{\theta}(0) = 0$$

The concept of equilibria: a case study

Pendulum dynamics

At both initial conditions

the system does not "move"

$$\dot{x} = \begin{pmatrix} x_2 \\ -\sin(x_1) - x_2 \end{pmatrix} = 0$$

$$x_2 = \dot{\theta} = 0$$
$$\sin(x_1) = \sin(\theta) = 0$$

$$\dot{\theta} = 0$$
$$\theta = \{0, \pi\}$$

The concept of equilibria: the general case

Given

$$1) \quad \dot{x} = f(x)$$

Then a point

$$x_e \in \mathbb{R}^n$$

is an equilibrium for system 1) if and only if

$$f(x_e) = 0$$

Do equilibria have special properties?

The concept of stability of equilibria: a case study

The “distance” between

$$(\theta_e, \dot{\theta}_e) = 0$$ and

$$(\theta, \dot{\theta})(t)$$

should decrease

$$\theta(0) = \pi \quad \dot{\theta}(0) = 0$$

Stability of equilibrium point: if the system is initialised close to it, it will evolve close to it

$$\theta(0) = 0 \quad \dot{\theta}(0) = 0$$

The concept of stability of equilibria: a case study

$$x_e = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\dot{x} = \begin{pmatrix} x_2 \\ -\sin(x_1) - x_2 \end{pmatrix}$$

$$x(t) = \begin{pmatrix} \theta(t) \\ \dot{\theta}(t) \end{pmatrix}$$

Distance: $$V = |x - x_e|^2 = x_1^2 + x_2^2$$

Does this “distance” decrease close to x_e ?

$$\dot{V} = 2x_1x_2 + 2x_2\dot{x}_2 = \{\sin(x_1) \approx x_1\} = -2x_2^2 \leq 0$$

The concept of stability of equilibria: Lyapunov

Let $x_e \in \mathbb{R}^n$ be an equilibrium point for $\dot{x} = f(x)$, i.e. $f(x_e) = 0$

If there exists a differentiable function $V(x) : \mathbb{R}^n \to \mathbb{R}$ such that

1) $V(x) > 0, \quad V(x_e) = 0$

2) $\dot{V}(x) \leq 0$

Then, x_e is stable. Moreover if also

$$\dot{V}(x) < 0, \quad \dot{V}(0) = 0$$

Then, the system trajectories converge to the equilibrium, i.e. $x(t) \to x_e$

The concept of stability of equilibria: Lyapunov

Fact i)

The function $V(x)$ is not always the Euclidian distance

$$\dot{x} = \begin{pmatrix} x_2 \\ -\alpha \sin(x_1) - \beta x_2 \end{pmatrix}$$

$$V = \alpha x_1^2 + x_2^2 \quad \Rightarrow \quad \dot{V} = \{\sin(x_1) \approx x_1\} = -2\beta x_2^2 \leq 0$$

The concept of stability of equilibria: Lyapunov

Fact ii)

The function $V(x)$ may contain whatever nonlinear term

$$\dot{x} = \begin{pmatrix} x_2 \\ -\alpha \sin(x_1) - \beta x_2 \end{pmatrix}$$

$$V(x) = x_2^2 + 2\alpha(1 - \cos(x_1)) \Rightarrow$$

$$\dot{V}(x) = -2\beta x_2^2$$

No need of approximations

The concept of stability of equilibria: Lyapunov

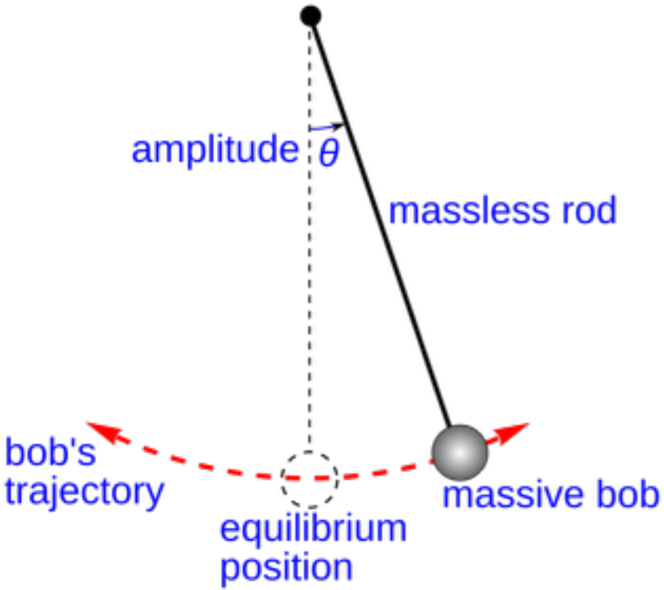
Fact iii)

$V(x)$ is often the "mechanical" energy of the system, scaled by some parameters

$$\ddot{\theta} = -\alpha \sin(\theta) - \beta\dot{\theta} + u$$

$$E = \frac{1}{2}mv^2 + mgh$$

$$V(x) = x_2^2 + 2\alpha(1 - \cos(x_1))$$



Take home messages

1) Nonlinear systems may have multiple equilibrium points

2) Lyapunov analysis can reveal if these equilibria are stable

3) Stability is a property of equilibrium points, not of the system

Applications to robotics

Robot Dynamics

$$f = ma \quad \text{for the robot?}$$

Robot Dynamics

$$\boxed{\frac{d}{dt}\frac{\partial}{\partial \dot{q}}\mathcal{L} - \frac{\partial}{\partial q}\mathcal{L} = \tau}$$

- $\mathcal{L} = T - U$: Lagrangian
- $q, \dot{q}, \ddot{q}$: joints' positions, velocities, and accelerations
- τ joint torques

$$\Downarrow$$

$$\boxed{M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) - J^\top F_{ext} = \tau}$$

$$\boxed{\begin{aligned} &M = M^\top > 0 \\ &\dot{M} - 2C = -(\dot{M} - 2C)^\top \end{aligned}}$$

- M, C, g: mass and Coriolis matrices, gravity torques
- F_{ext}, J: **vectorized** external forces and its Jacobian

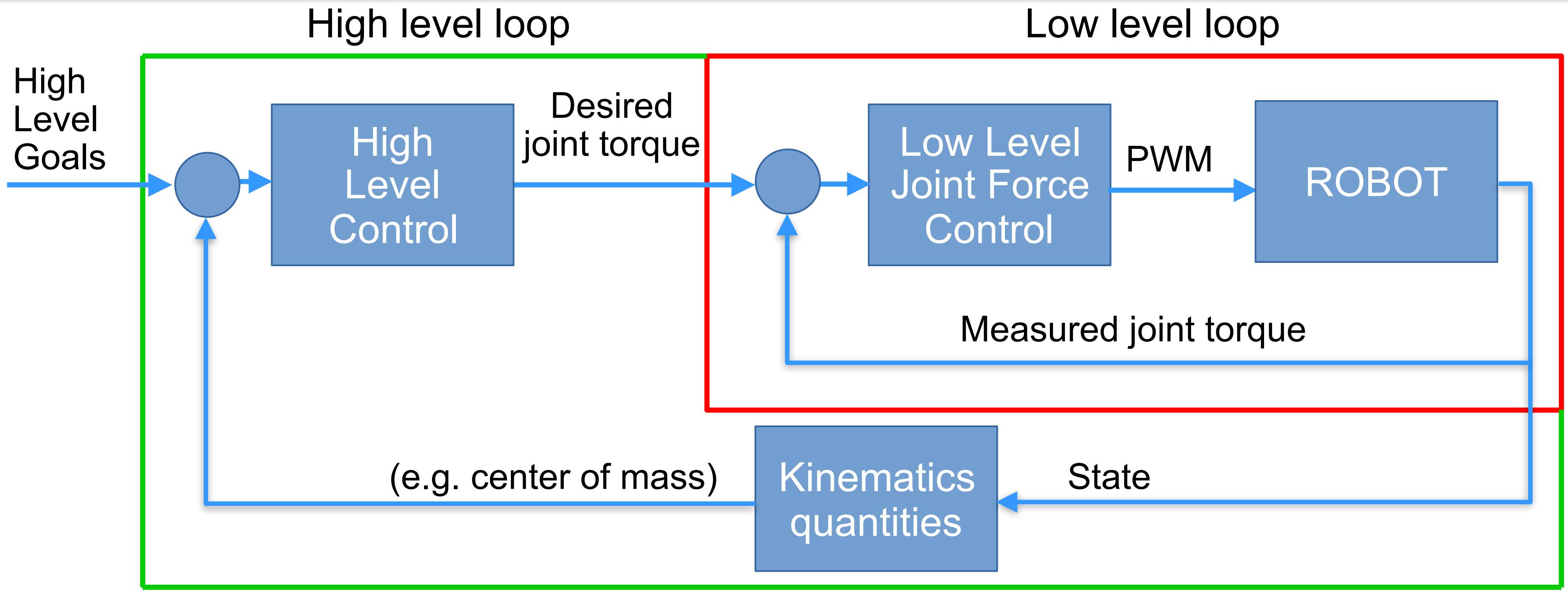
Robot Dynamics

$$\boxed{M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau}$$

Joint torques assumed τ as control input

We assume that τ can be chosen at will… in reality

Torque Control Architecture

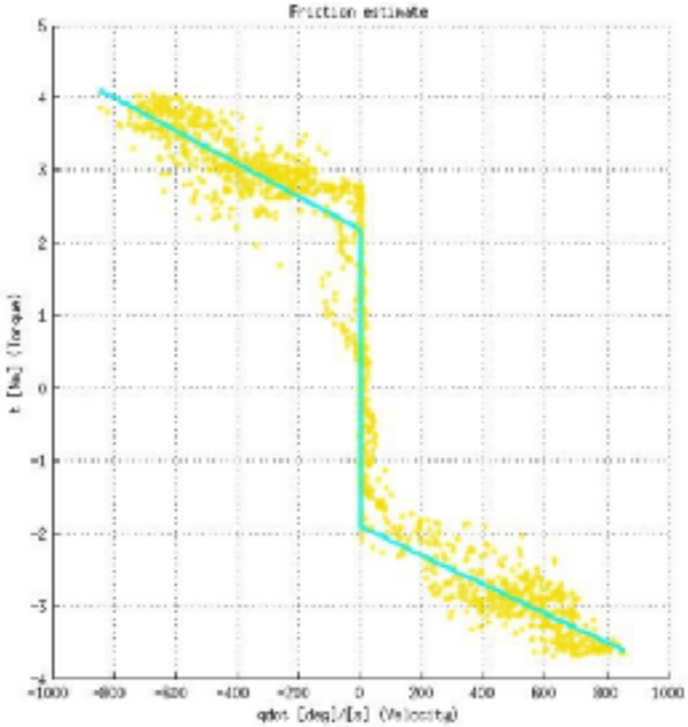


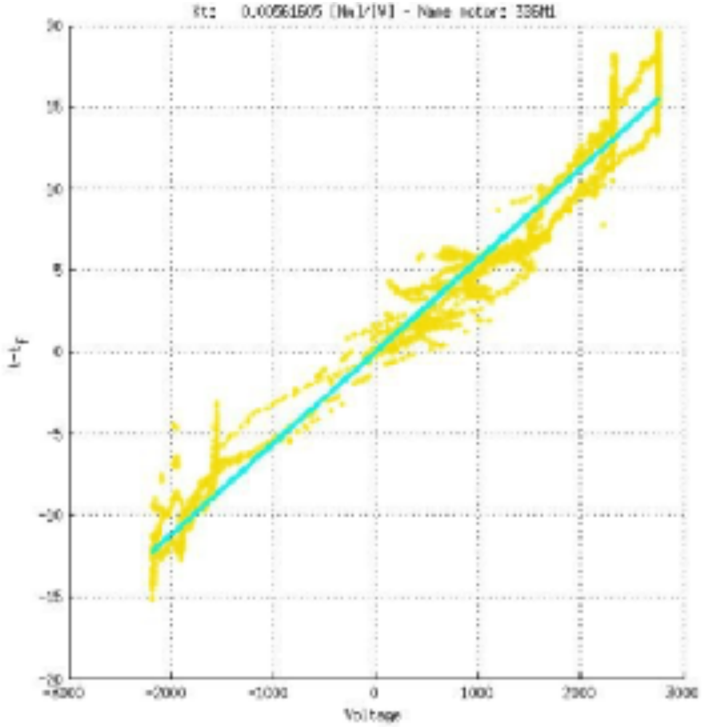
High level loop runs at 100 Hz

Low level loop runs at 1 kHz

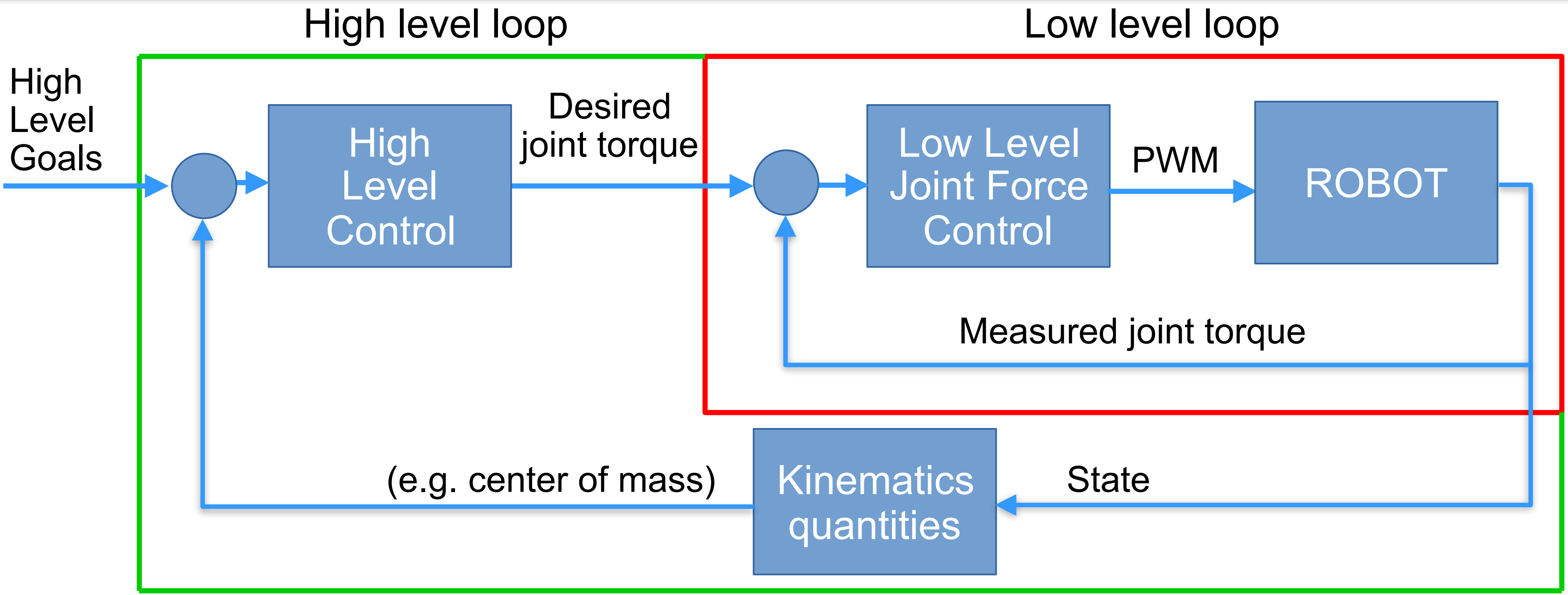
Torque Control Architecture

$$\tau = k_\tau PWM - k_v \dot{m} - k_c \mathrm{sign}(\dot{m}) \quad \Rightarrow \quad PWM = \bar{k}_t \tau + \bar{k}_v \dot{m} + \bar{k}_c \mathrm{sign}(\dot{m})$$





Torque Control Architecture



How can we choose the "desired" joint torques?

Control Objective

1) Stabilisation of a desired joint trajectory $q_d(t) \in \mathbb{R}^n$

2) Impose some joint compliance K_p

PD plus gravity compensation control

Joint position error: $q - q_d$

Joint velocity error: $\dot{q} - \dot{q}_d$

Joint torques ensuring stabilisation of joint trajectory:

$$\tau = M(q)\ddot{q}_d - K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d) + C(q, \dot{q})\dot{q}_d + g(q)$$

Joint stiffness: K_p

PD plus gravity compensation control

Fact i):

Stability and convergence of the control law can be proven by using

$$V = \frac{1}{2}(\dot{q} - \dot{q}_d)^\top M(q)(\dot{q} - \dot{q}_d) + \frac{1}{2}(q - q_d)^\top K_p(q - q_d)$$

$$\Rightarrow \quad \boxed{\begin{array}{l} M = M^\top > 0 \\ \dot{M} - 2C = -(\dot{M} - 2C)^\top \end{array}}$$

$$\dot{V} = -\frac{1}{2}(\dot{q} - \dot{q}_d)^\top K_d(\dot{q} - \dot{q}_d) \leq 0$$

PD plus gravity compensation control

Fact ii):

The control law to stabilise set points, i.e.

$$\dot{q}_d = \ddot{q}_d = 0$$

becomes

$$\tau = g(q) - K_p(q - q_d) - K_d\dot{q}$$

which is simple and does not need Coriolis and mass matrix

Computed torque control law

Joint position error: $q - q_d$

Joint velocity error: $\dot{q} - \dot{q}_d$

Joint torques ensuring stabilisation of joint trajectory:

$$\tau = M(q)\left[\ddot{q}_d - K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d)\right] + C(q, \dot{q})\dot{q} + g(q)$$

Joint stiffness: $M(q)K_p$

Computed torque control law

Fact i):

Stability and convergence of the control law can be proven by using

$$V = \frac{1}{2}(\dot{q} - \dot{q}_d)^\top(\dot{q} - \dot{q}_d) + \frac{1}{2}(q - q_d)^\top K_p(q - q_d)$$

$$\Rightarrow$$

$$\dot{V} = -\frac{1}{2}(\dot{q} - \dot{q}_d)^\top K_d(\dot{q} - \dot{q}_d) \leq 0$$

Computed torque control law

<u>Fact ii):</u>

Given the dynamics $$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

with

$$\tau = M(q)\left[\ddot{q}_d - K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d)\right] + C(q,\dot{q})\dot{q} + g(q)$$

gives

$$\ddot{q} = \ddot{q}_d - K_p(q - q_d) - K_d(\dot{q} - \dot{q}_d)$$

Decoupling

Computed torque control law

<u>Fact iii):</u>

The control law to stabilise set points, i.e.

$$\dot{q}_d = \ddot{q}_d = 0$$

becomes

$$\tau = C(q,\dot{q})\dot{q} + g(q) - M(q)K_p(q - q_d) - M(q)K_d\dot{q}$$

which needs Coriolis and mass matrix

Comparisons of control laws for set points

PD plus gravity compensation:

$$\tau = g(q) - K_p(q - q_d) - K_d\dot{q}$$

Prons
- needs only gravity
- ensures a constant stiffness
- robust

Cons
- does not ensure decoupling

Computed torque

$$\tau = C(q, \dot{q})\dot{q} + g(q) - M(q)K_p(q - q_d) - M(q)K_d\dot{q}$$

Prons
- ensures decoupling

Cons
- does not ensure constant stiffness
- Requires mass matrix and coriolis terms
- less robust