*Corso di Robotica 1*

# Pianificazione di traiettorie

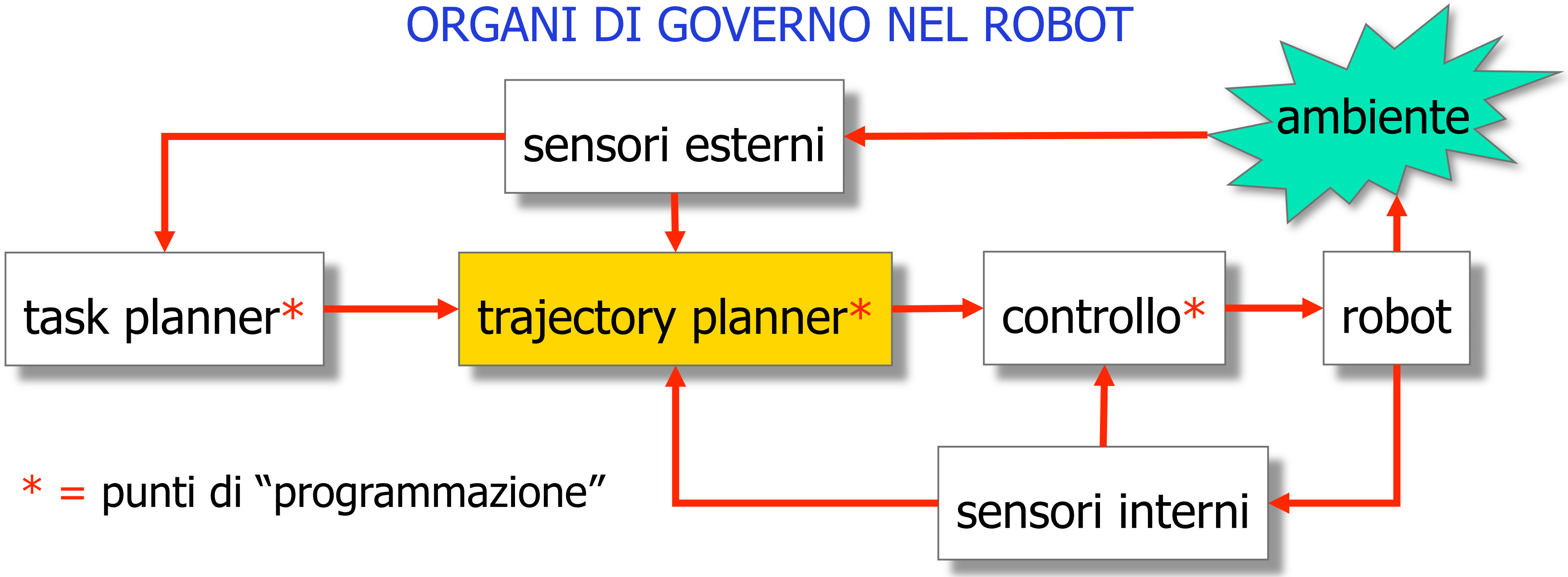
Prof. Alessandro De Luca

Dipartimento di Informatica e Sistemistica Antonio Ruberti

Sapienza Università di Roma

# Pianificazione delle traiettorie

## ORGANI DI GOVERNO NEL ROBOT




descrizione dell'azione
come sequenza di
configurazioni con
eventuali scambi
di forze di contatto

PIANIFICATORE DI TRAIETTORIE

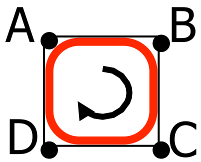
riferimenti
(continui o discreti)
per il controllore

# Schema standard di pianificazione
## nei robot industriali

1. specifica nodi cartesiani (posizione+orientamento) mediante teach-box
2. specifica da programma della velocità (media) tra i punti in % rispetto a valore massimo (diverso tra spazio cartesiano e giunti)
3. interpolazione lineare in spazio giunti tra punti campionati sulla traiettoria

esempi di features aggiuntive

a) overfly




b) sensor-driven STOP

c) circonferenza per 3 punti

limiti essenziali

- programmazione semi-manuale (tipica dei linguaggi di "prima generazione")
- scarsa visualizzazione

➡ è richiesta una formalizzazione matematica delle traiettorie

# Dal compito alla traiettoria

TRAIETTORIA
- di movimento $p_d(t)$
- di interazione $F_d(t)$

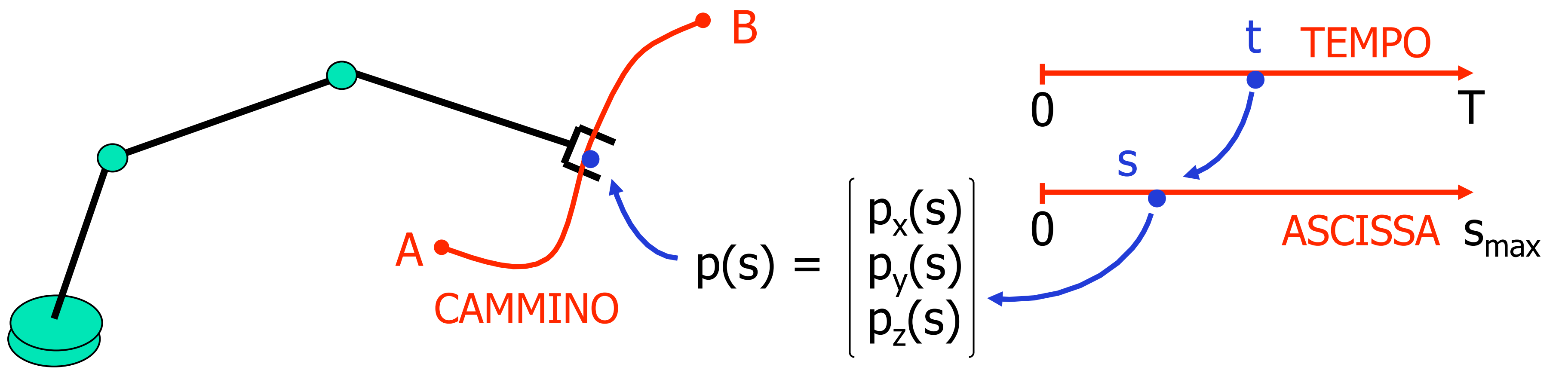
||

CAMMINO GEOMETRICO parametrizzato da s: $p = p(s)$
(ad esempio, s è l'ascissa curvilinea)

+

LEGGE ORARIA descrive l'andamento di $s = s(t)$

$\Rightarrow p(s(t))$




esempio: TASK planner fornisce A, B
TRAJECTORY planner genera p(t)

# Sequenza operativa di pianificazione

TASK planning

1

- sequenza nodi in spazio cartesiano
    - interpolazione in spazio cartesiano
- cammino geometrico cartesiano (posiz + orient): $p = p(s)$
    - campionamento e inversione cinematica

2

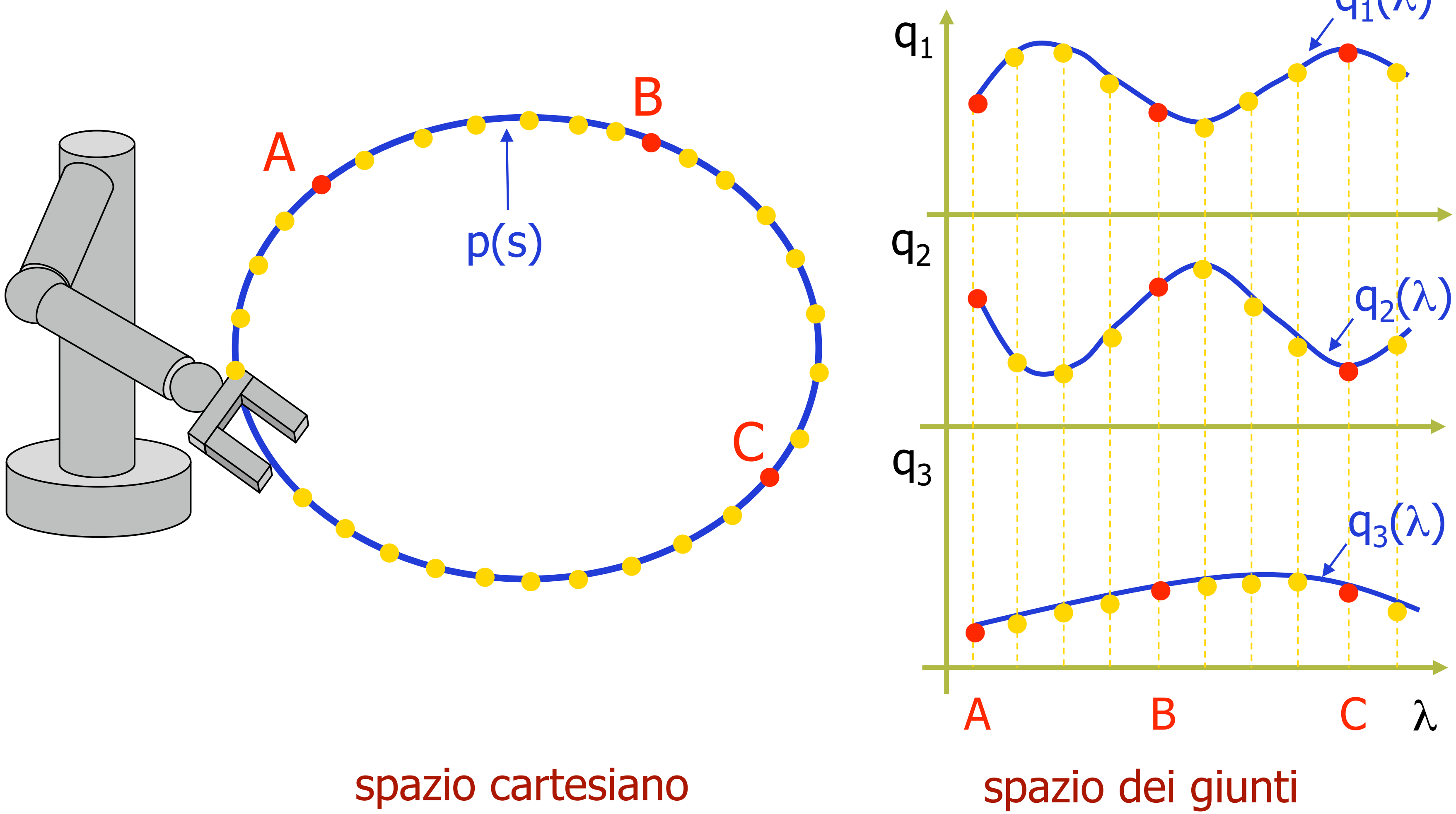
- sequenza nodi in spazio giunti
    - interpolazione in spazio giunti
- cammino geometrico in spazio giunti $q = q(\lambda)$

inversione analitica

da tenere in conto durante il processo di pianificazione

- obstacle avoidance
- carico computazionale on-line/off-line
- sequenza 2 "più fitta" della 1

# Esempio


A
B
C
p(s)
$q_1$
$q_2$
$q_3$
$q_1(\lambda)$
$q_2(\lambda)$
$q_3(\lambda)$
A
B
C
$\lambda$
spazio cartesiano
spazio dei giunti

# Pianificazione cartesiana vs. nei giunti

- pianificazione nello spazio cartesiano
  - permette una visualizzazione più diretta del cammino
  - obstacle avoidance, assenza di "wandering"
- pianificazione nello spazio dei giunti
  - non richiede l'inversione cinematica in linea
- inversione cinematica
  - a volte occorrono anche $\dot{q}$ e $\ddot{q}$ (o derivate superiori)
    - le specifiche cartesiane non dipendono solo dal cammino, ma anche dalla legge oraria su di esso
  - per robot ridondanti scelta tra le $\infty^{n-m}$ soluzioni inverse, in base a criteri di ottimalità o compiti ausiliari aggiuntivi
  - la pre-pianificazione fuori linea non è sempre possibile
    - ad es., quando c'è interazione con l'ambiente o compiti dipendenti da informazioni sensoriali

# Cammino e legge oraria

- dopo la scelta del cammino, per completare una traiettoria occorre una legge di moto

$$p = p(s) \qquad s = s(t) \qquad \text{(spazio cartesiano)}$$
$$q = q(\lambda) \qquad \lambda = \lambda(t) \qquad \text{(spazio dei giunti)}$$

  - se $s(t) = t$, si ha una parametrizzazione naturale con il tempo
- la legge di moto
  - è scelta in base alle specifiche del compito (fermarsi in un punto, procedere a velocità costante, ecc.)
  - è scelta in base a criteri di ottimo (tempo minimo, minima energia, ecc.)
  - i vincoli sono imposti dagli attuatori (max coppia, max velocità) o dal compito (max carico inerziale/accelerazione su payload)

N.B. sui cammini parametrizzati si ha una decomposizione spazio-temporale, ad es. nel cartesiano

$$\dot{p}(t) = \frac{dp}{ds}\dot{s} \qquad \ddot{p}(t) = \frac{dp}{ds}\ddot{s} + \frac{d^2p}{ds^2}\dot{s}^2$$

# Classificazione traiettorie

- spazio
  - cartesiano, di giunto
- compito
  - punto-punto, continuo, concatenato
- cammino
  - rettilineo, polinomiale, esponenziale, …
- legge oraria
  - bang-bang in accelerazione, trapezoidale in velocità, …
- moto coordinato o indipendente
  - i moti di tutti i giunti (o di tutte le componenti cartesiane) iniziano e finiscono negli stessi istanti (t=0 e t=T) = unica legge oraria

  o
  - i moti sono pianificati in modo indipendente (a seconda delle richieste di spostamento e dei vincoli di attuazione del robot)

# Caratteristiche notevoli

- efficienza di calcolo e memorizzazione
  - ad es., solo i coefficienti di una funzione polinomiale
- predicibilità (vs. "wandering") e accuratezza (vs. "overshoot" sulla posizione finale)
- flessibilità (per concatenazione, overfly, ...)
- continuità

  (almeno $C^1$, ma anche fino al jerk = $\frac{da}{dt}$ )

# Pianificazione nello spazio dei giunti

- $q = q(t)$ oppure $q = q(\lambda)$, $\lambda = \lambda(t)$
- si lavora per componenti $q_i$ del vettore q
- la definizione della traiettoria avviene in modo implicito risolvendo un problema di condizioni al contorno con un'opportuna classe di funzioni
- classi tipiche: polinomi (cubici, quintici,…), cosinusoidi, clotoidi, …
- condizioni imposte
  - passaggio per punti
  - velocità iniziali, finali, di passaggio
  - accelerazioni iniziali, finali
  - continuità fino alla derivata del k-mo ordine ($\mathbf{C}^k$)

molte considerazioni e/o metodi sono applicabili direttamente anche alla pianificazione cartesiana!

# Polinomio cubico (point-to-point)

$\boxed{q(0) = q_{in}}$ $\boxed{q(T) = q_{fin}}$ $\boxed{\dot{q}(0) = v_{in}}$ $\boxed{\dot{q}(T) = v_{fin}}$ ← 4 condizioni

$$q(\tau) = q_{in} + \Delta q\,[\underbrace{a\,\tau^3 + b\,\tau^2 + c\,\tau + d}_{\text{polinomio } q_N(\tau) \text{ ``doubly normalized''}}]$$

$\Delta q = q_{fin} - q_{in}$

$\tau = t/T,\ \tau \in [0, 1]$

4 coefficienti ⟶ polinomio $q_N(\tau)$ "doubly normalized"

$q_N(0) = 0 \Leftrightarrow d = 0$ $\qquad$ $q_N(1) = 1 \Leftrightarrow a + b + c = 1$

$q_N'(0) = dq_N/d\tau|_{\tau=0} = c = v_{in}T/\Delta q$ $\qquad$ $q_N'(1) = dq_N/d\tau|_{\tau=1} = 3a + 2b + c = v_{fin}T/\Delta q$

CASO PARTICOLARE $v_{in} = v_{fin} = 0$

$q_N'(0) = 0 \Leftrightarrow c = 0$

$$\left.\begin{array}{l} q_N(1) = 1 \Leftrightarrow a + b = 1 \\ q_N'(1) = 0 \Leftrightarrow 3a + 2b = 0 \end{array}\right\} \Leftrightarrow \begin{array}{l} a = -2 \\ b = 3 \end{array}$$

# Polinomio quintico

$q(\tau) = a\tau^5 + b\tau^4 + c\tau^3 + d\tau^2 + e\tau + f$ 6 coefficienti

$\tau = t/T, \tau \in [0, 1]$

permette di soddisfare 6 condizioni, ad es. (nel tempo scalato $\tau$)

$\boxed{q(0) = q_0}$ $\boxed{q(1) = q_1}$ $\boxed{q'(0) = v_0T}$ $\boxed{q'(1) = v_1T}$ $\boxed{q''(0) = a_0T^2}$ $\boxed{q''(1) = a_1T^2}$

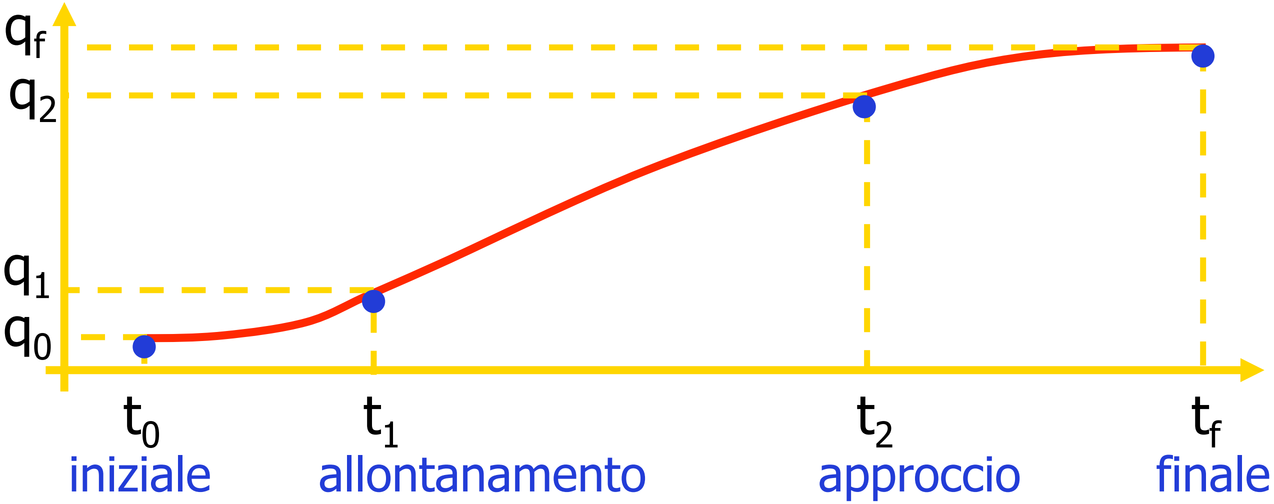
$$q(\tau) = (1 - \tau)^3[q_0 + (3q_0+v_0T)\tau + (a_0T^2+6v_0T+12q_0)\tau^2/2]$$
$$+\tau^3 [q_1 + (3q_1 -v_1T)(1-\tau) + (a_1T^2 -6v_1T+12q_1) (1 - \tau)^2/2]$$

CASO PARTICOLARE $v_0 = v_1 = a_0 = a_1 = 0$

$q(\tau) = q_0 + \Delta q\,[6\tau^5 - 15\tau^4 + 10\tau^3]$ $\Delta q = q_1 - q_0$

# Polinomio 4-3-4

tre fasi (Lift off, Travel, Set down) nelle operazioni di pick and place




$q_L(t)$ = pol. 4° ordine
$q_T(t)$ = pol. 3° ordine
$q_S(t)$ = pol. 4° ordine

14 coefficienti

condizioni al contorno

$q(t_0) = q_0 \quad q(t_1^-) = q(t_1^+) = q_1 \quad q(t_2^-) = q(t_2^+) = q_2 \quad q(t_f) = q_f$ } 6 passaggio

$\dot{q}(t_0) = \dot{q}(t_f) = 0 \qquad \ddot{q}(t_0) = \ddot{q}(t_f) = 0$ } 4 vel/acc iniziale/finale

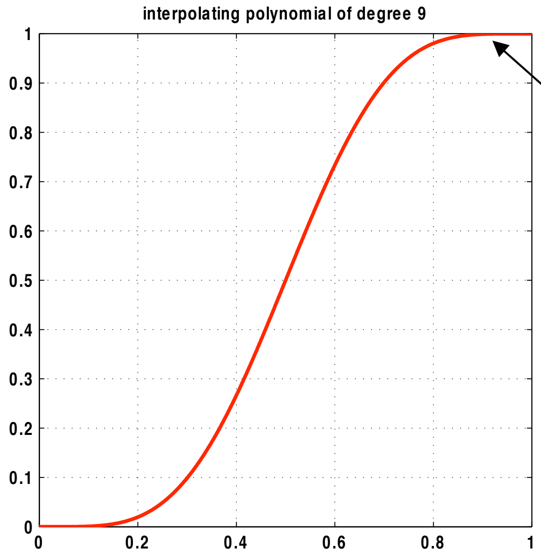
$\dot{q}(t_i^-) = \dot{q}(t_i^+) \qquad \ddot{q}(t_i^-) = \ddot{q}(t_i^+) \quad i = 1,2$ } 4 continuità

# Polinomi di grado elevato

- si comportano in modo soddisfacente se si devono soddisfare condizioni al contorno simmetriche e nulle sulle derivate di ordine superiore
  - sono quindi polinomi di grado dispari
  - in tal caso, i coefficienti del polinomio (doppiamente normalizzato) sono sempre interi, a segni alternati, a somma totale unitaria, nulli per i termini fino alla potenza (grado-1)/2
- in tutte le altre situazioni (ad es., per interpolare un numero N grande di punti) non sono consigliabili
  - polinomi di grado N hanno N-1 massimi o minimi
  - oscillazioni al di fuori dei punti di interpolazione (wandering)

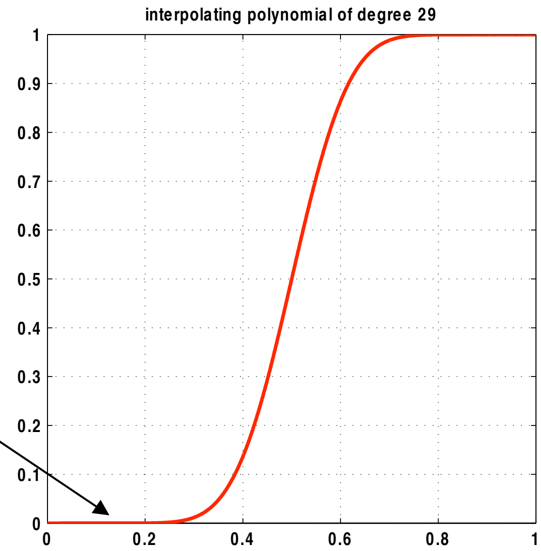
# Esempi

grado
9




4 derivate
nulle

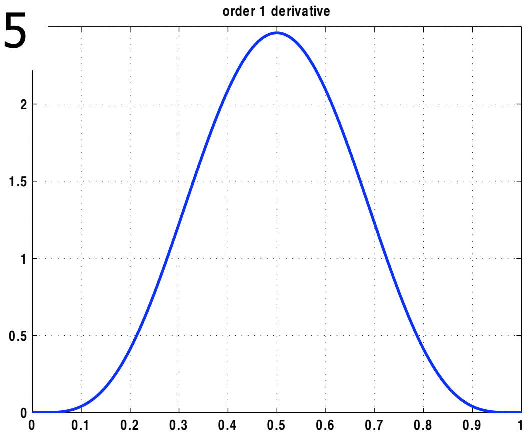
14 derivate
nulle!



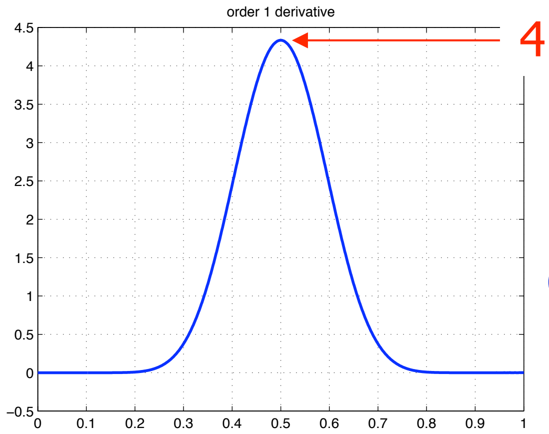

grado
29

assenza di
overshoot

2.5




velocità
normalizzate




4.5

picco
centrale
di velocità

# Interpolazione con splines

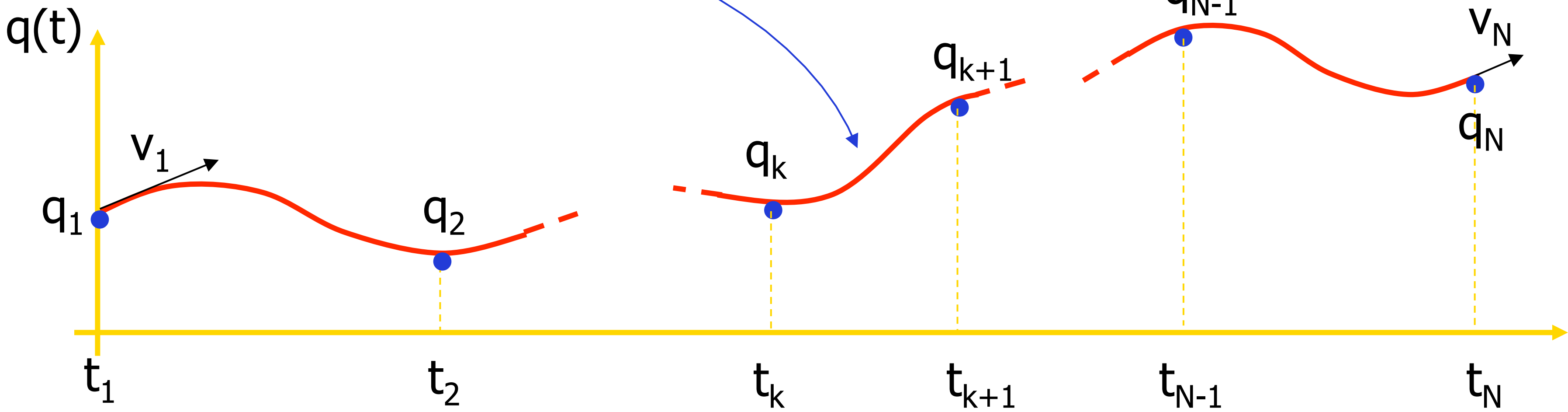
- problema

  interpolare N nodi con continuità fino all'accelerazione
- soluzione

  spline: N-1 polinomi cubici concatenati in modo da passare per gli N nodi ed essere continui in velocità e accelerazione negli N-2 nodi interni
- 4(N-1) coefficienti
- 4(N-1) - 2 condizioni, ovvero
  - 2(N-1) di passaggio per ogni cubica nei due nodi ai suoi estremi
  - N-2 di continuità in velocità (nodi interni)
  - N-2 di continuità in accelerazione (nodi interni)
- rimangono 2 parametri liberi
  - ad esempio per fissare velocità iniziale $v_1$ e finale $v_N$
- trattazione nel tempo (variabile t), ma analoga nello spazio ($\lambda$)

# Costruzione di una spline

$$q = \theta(t) = \{\theta_K(t),\ t \in [t_k,\ t_k + h_k]\}$$




se il percorso è ciclico ($q_1 = q_N$)
si può porre $v_1 = v_N$, $a_1 = a_N$

intervalli $h_k$ temporali

$$\boxed{\theta_K(\tau) = a_{k0} + a_{k1}\,\tau + a_{k2}\,\tau^2 + a_{k3}\,\tau^3} \qquad \tau \in [0,\ h_k],\ \tau = t - t_k$$

continuità
in velocità e accelerazione $\Rightarrow$

$$\dot{\theta}_K(h_k) = \dot{\theta}_{K+1}(0) \qquad k = 1, \ldots, N\text{-}2$$
$$\ddot{\theta}_K(h_k) = \ddot{\theta}_{K+1}(0)$$

# Algoritmo efficiente

1. supposte note le velocità $v_k$ ai nodi interni, la spline è univocamente determinata

$$\theta_K(0) = q_K = a_{K0}$$
$$\dot{\theta}_K(0) = v_K = a_{K1}$$

$$\begin{pmatrix} h_K^2 & h_K^3 \\ 2h_K & 3h_K^2 \end{pmatrix} \begin{pmatrix} a_{K2} \\ a_{K3} \end{pmatrix} = \begin{pmatrix} q_{K+1} - q_K - v_K h_K \\ v_{K+1} - v_K \end{pmatrix} \quad (1)$$

2. imponendo la continuità in accelerazione si ottiene

$$\ddot{\theta}_K(h_k) = 2\, a_{K2} + 6\, a_{K3}\, h_K = \ddot{\theta}_{K+1}(0) = 2\, a_{K+1,2}$$

3. esprimendo i coefficienti $a_{k2}$, $a_{k3}$, $a_{k+1,2}$ in funzione delle velocità (vedi passo 1.) si ottiene un sistema lineare sempre (facilmente) risolubile, nella forma

$$\begin{pmatrix} A(h) \end{pmatrix} \begin{pmatrix} v_2 \\ v_3 \\ \vdots \\ v_{N-1} \end{pmatrix} = \begin{pmatrix} b(h,q,v_1,v_N) \end{pmatrix}$$

matrice tridiagonale sempre invertibile

incognite

da risostituire in (1)

termini noti

# Struttura di A(h)

$$\begin{pmatrix} 2(h_1+h_2) & h_1 & & & & \\ h_3 & 2(h_2+h_3) & h_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & h_{N-2} & 2(h_{N-3}+h_{N-2}) & h_{N-3} \\ & & & h_{N-1} & 2(h_{N-2}+h_{N-1}) \end{pmatrix}$$

matrice dominante diagonale (per $h_k > 0$)

[la stessa per ogni giunto]

# Struttura di $b(h,q,v_1,v_N)$

$$
\begin{pmatrix}
\dfrac{3}{h_1 h_2}\left[h_1^2(q_3 - q_2) + h_2^2(q_2 - q_1)\right] - h_2 v_1 \\
\dfrac{3}{h_2 h_3}\left[h_2^2(q_4 - q_3) + h_3^2(q_3 - q_2)\right] \\
\vdots \\
\dfrac{3}{h_{N-3} h_{N-2}}\left[h_{N-3}^2(q_{N-1} - q_{N-2}) + h_{N-2}^2(q_{N-2} - q_{N-3})\right] \\
\dfrac{3}{h_{N-2} h_{N-1}}\left[h_{N-2}^2(q_N - q_{N-1}) + h_{N-1}^2(q_{N-1} - q_{N-2})\right] - h_{N-2} v_N
\end{pmatrix}
$$

# Proprietà delle splines

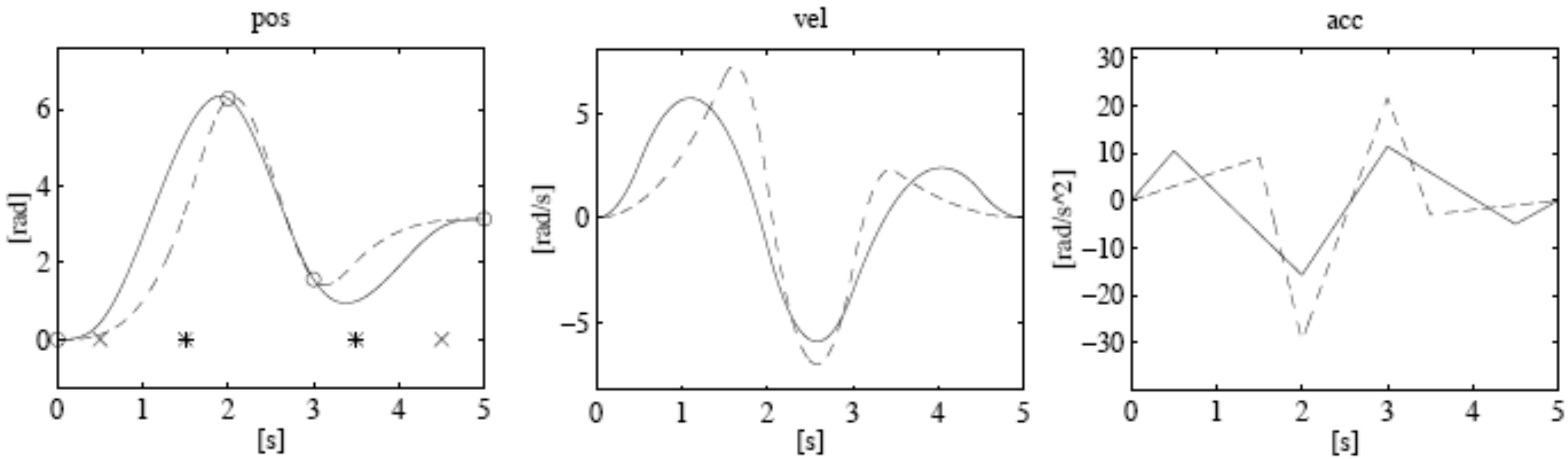
- le splines sono le funzioni a minima curvatura tra tutte le interpolanti con derivata seconda continua
- la spline è univocamente determinata dai dati $q_1,\ldots,q_N$, $h_1,\ldots,h_N$, $v_1$, $v_N$
- il tempo totale di percorrenza è $T = \Sigma h_k = t_N - t_1$
- gli intervalli $h_k$ si possono scegliere in modo da minimizzare T (funzione obiettivo lineare) sotto i vincoli (nonlineari) di massima velocità e massima accelerazione
- per compiti ciclici ($q_1=q_N$), è preferibile imporre la continuità in velocità e accelerazione in $t_1=t_N$ per "quadrare" il sistema
  - infatti, scegliere solo $v_1=v_N$ non garantisce continuità in accelerazione
  - in tal modo invece il primo=ultimo nodo saranno trattati come tutti gli altri nodi interni
- se sono specificate anche le accelerazioni iniziali e finali, si può modificare opportunamente lo schema

# Accelerazioni iniziali e finali assegnate

- servono due parametri aggiuntivi per imporre accelerazione iniziale $\alpha_1$ e finale $\alpha_N$
- si inseriscono due "nodi fittizi" nel primo e nell'ultimo intervallo, portando così il numero di cubiche usate da N-1 a N+1
- nei nodi fittizi occorre imporre solo condizioni di continuità in posizione, velocità e accelerazione
  $\Rightarrow$ rimangono due parametri liberi (uno nella cubica iniziale e uno in quella finale) per soddisfare le condizioni al contorno sulle accelerazioni
- la spline risultante cambierà forma in funzione del posizionamento (nel tempo) dei due nodi fittizi aggiunti

# Esempio numerico

- N=4 nodi (3 polinomi cubici)
  - valori di giunto $q_1 = 0$, $q_2 = 2\pi$, $q_3 = \pi/2$, $q_4 = \pi$
  - in $t_1 = 0$, $t_2 = 2$, $t_3 = 3$, $t_4 = 5$ (quindi con $h_1 = 2$, $h_2 = 1$, $h_3 = 2$)
  - velocità al contorno: $v_1 = v_4 = 0$
- 2 nodi fittizi per imporre le accelerazioni iniziali/finali (5 polinomi cubici)
  - accelerazioni al contorno: $\alpha_1 = \alpha_4 = 0$
  - due scelte: in $t_1' = 0.5$ e $t_4' = 4.5$ ($\times$), oppure in $t_1'' = 1.5$ e $t_4'' = 3.5$ ($*$)




——— = scelta′ --------- = scelta″