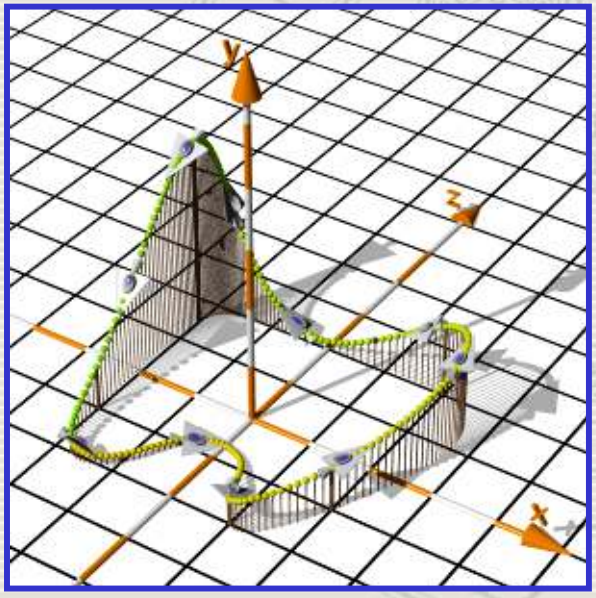
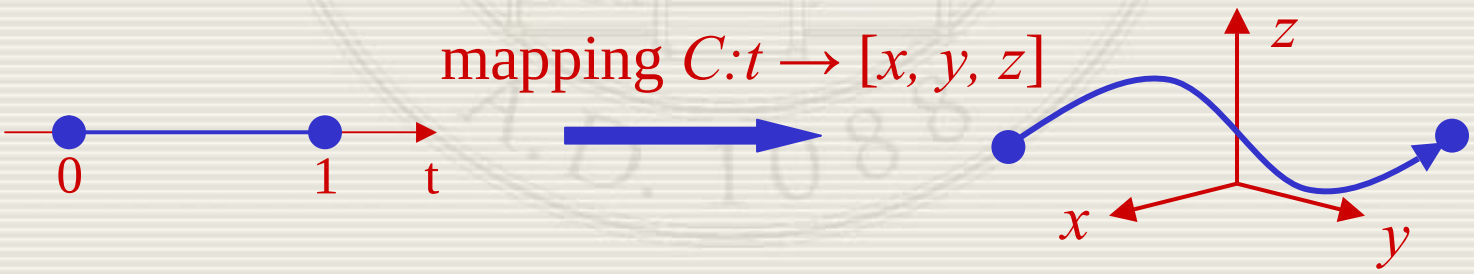
Curve nella Computer Graphics



Curve: che cosa sono?

- Definiamo uno spazio parametrico
  - 1D (per curve)
- Definiamo un mapping fra lo spazio dei parametri e lo spazio 2D o 3D
  - una funzione che prende valori parametrici (scalari) e restituisce punti 2D/3D
- Il risultato è una curva in forma parametrica (funzione vettoriale)

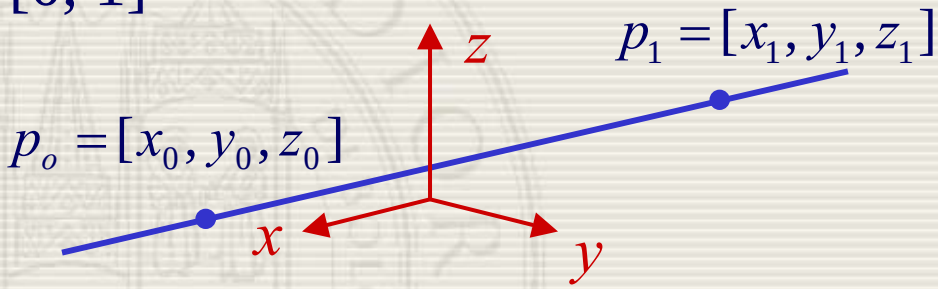


Curve in forma parametrica

- Abbiamo già visto una retta 2D in forma parametrica; e in 3D?

$$p(t) = (1-t)\, p_0 + t\, p_1 \qquad t \in [0, 1]$$

$$\begin{cases} x = x_0 t + (1-t) x_1 \\ y = y_0 t + (1-t) y_1 \\ z = z_0 t + (1-t) z_1 \end{cases}$$



- Si noti che x, y e z sono determinati ciascuno da una espressione che coinvolge:
  - il parametro t
  - le coordinate dei due punti assegnati p_0 e p_1

Un segmento retto è un esempio di curva 3D in forma parametrica

Curve 3D in forma parametrica

Curva in forma parametrica:

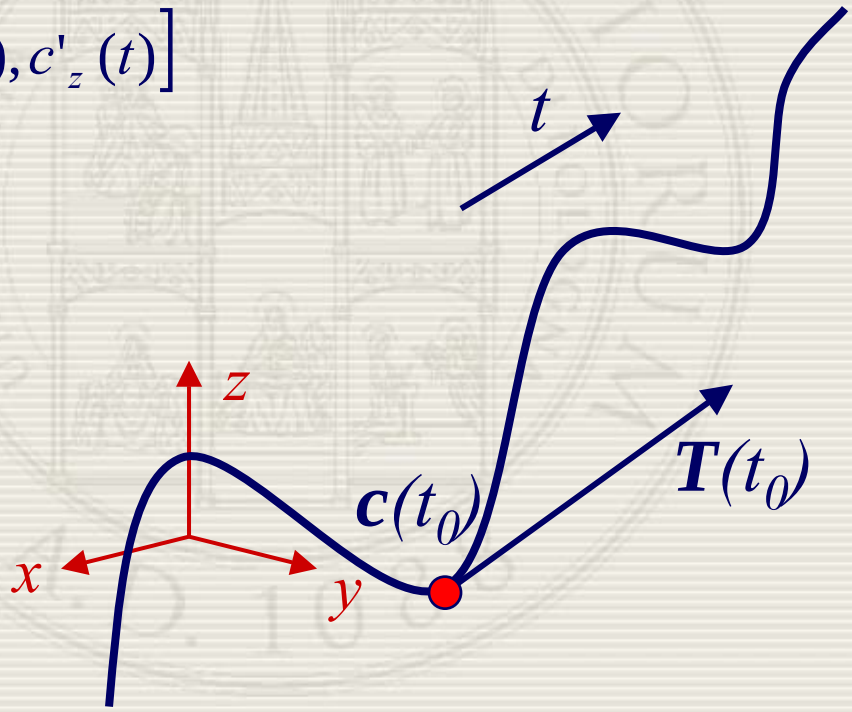
$$c(t) = \left[c_x(t), c_y(t), c_z(t)\right] \qquad t \in [0,1]$$

Vettore Tangente alla curva:

$$c'(t) = \left[c'_x(t), c'_y(t), c'_z(t)\right]$$

Versore Tangente:

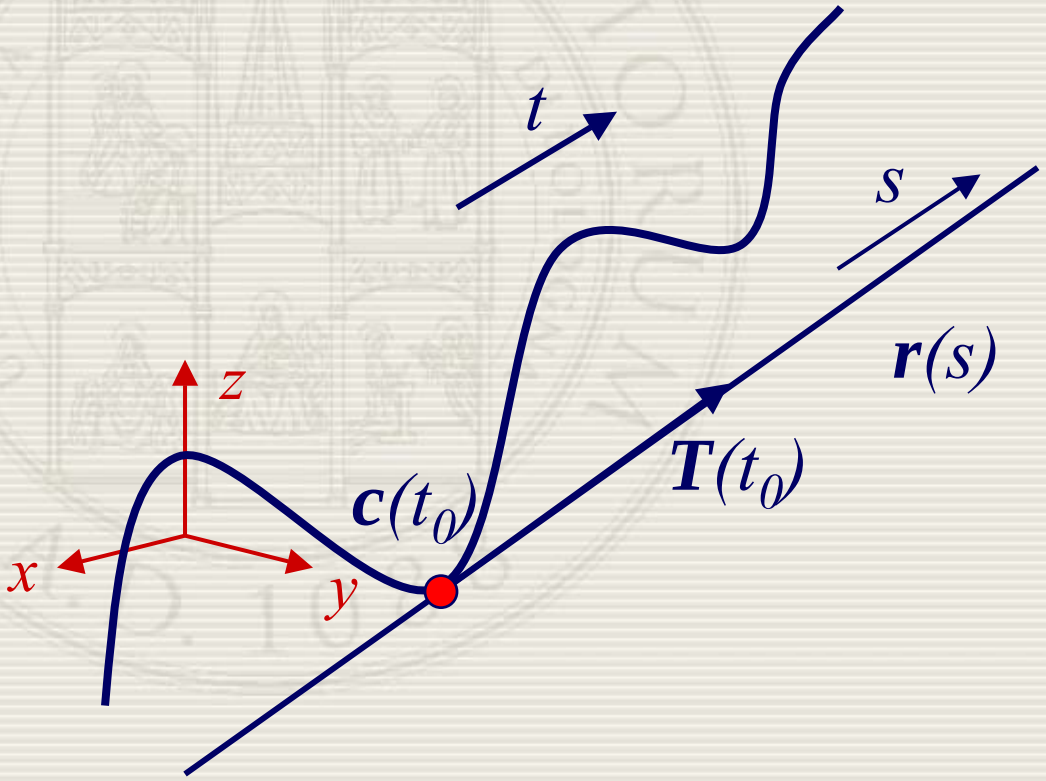
$$T(t) = \frac{c'(t)}{\left\|c'(t)\right\|}$$



Retta Tangente alla curva

Retta tangente $r(s)$ alla curva $c(t)$ in t_0 :

$$r(s) = c(t_0) + sT(t_0) \text{ con } s \in R$$



Esempio 2D e 3D

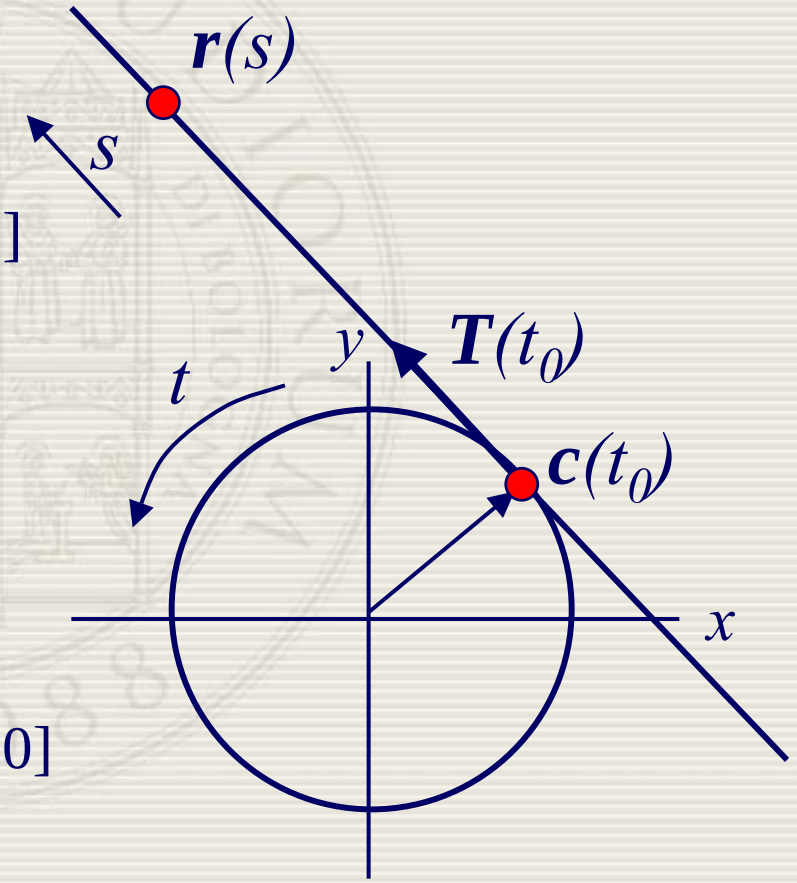
$$c(t) = [\cos t, \sin t]$$

$$c'(t) = [-\sin t, \cos t]$$

$$T(t) = [-\sin t, \cos t]$$

$$r(s) = c(t_0) + sT(t_0)$$

$$= [\cos t_0 - s\sin t_0, \sin t_0 + s\cos t_0]$$



$$c(t) = [\cos t, \sin t, 0]$$

$$c'(t) = [-\sin t, \cos t, 0]$$

$$T(t) = [-\sin t, \cos t, 0]$$

$$r(s) = c(t_0) + sT(t_0)$$

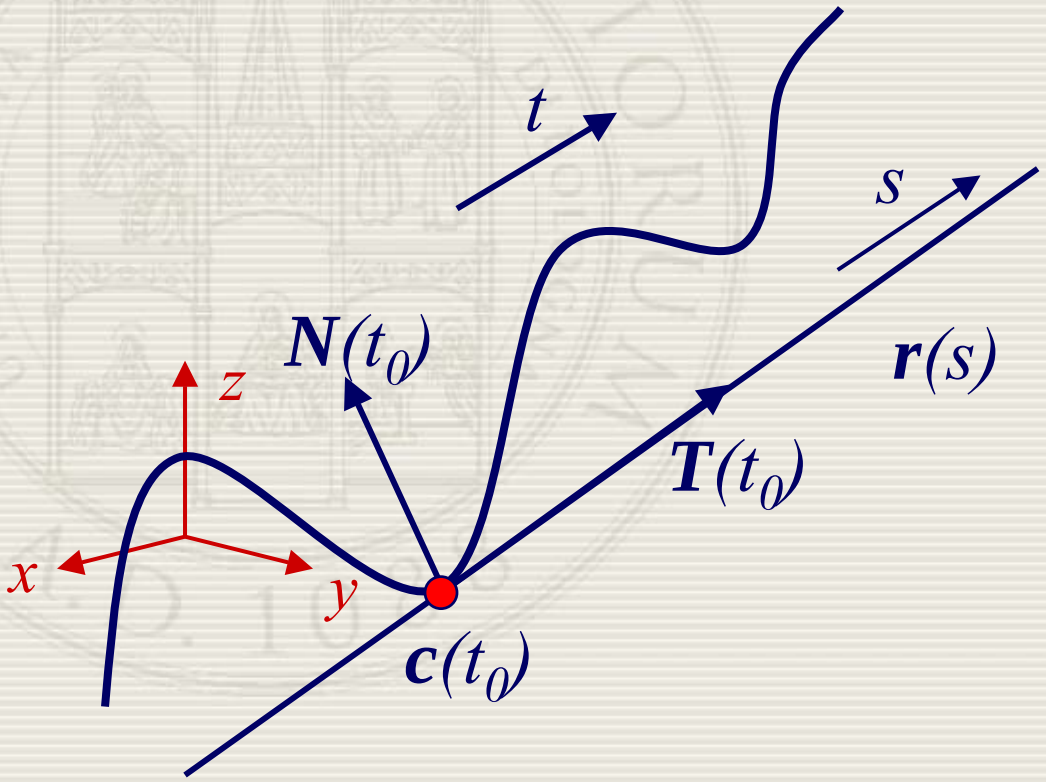
$$= [\cos t_0 - s\sin t_0, \sin t_0 + s\cos t_0, 0]$$

Normale alla curva

Vettore normale alla curva $c(t)$ in t_0 :

Sia $c''(t) = \left[c''_x(t), c''_y(t), c''_z(t)\right]$

$$N(t) = \frac{T'(t)}{\left\|T'(t)\right\|}$$



Frenet Frame

- Tangente unitaria

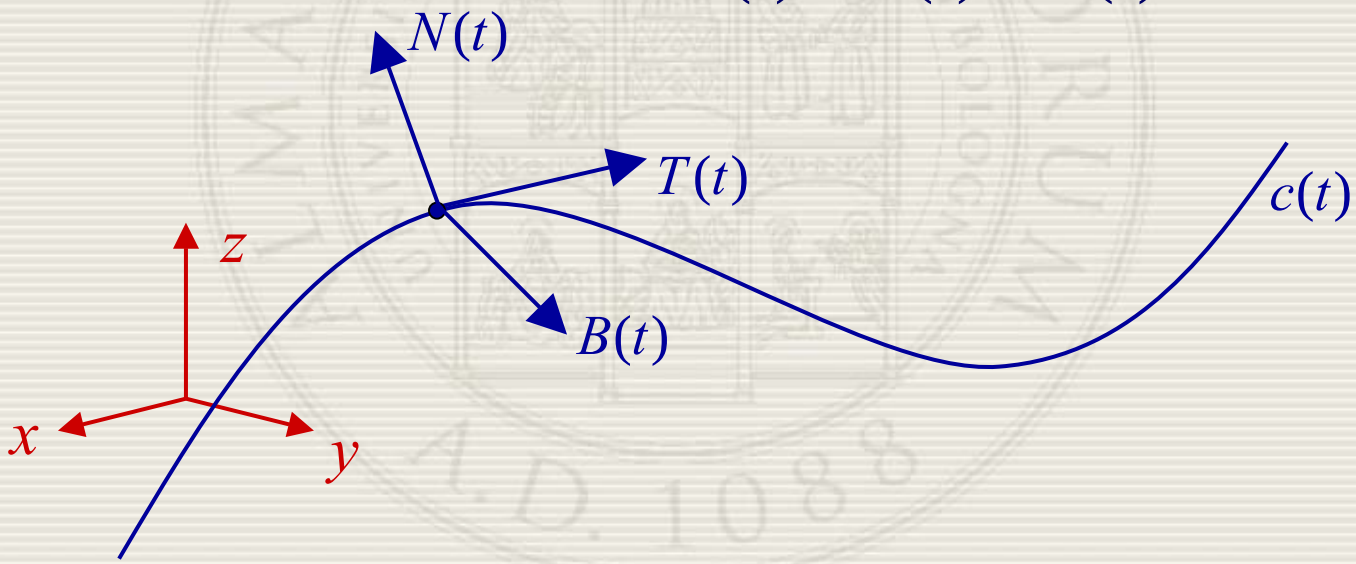
$$T(t) = \frac{c'(t)}{\|c'(t)\|}$$

- Normale unitaria

$$N(t) = \frac{T'(t)}{\|T'(t)\|}$$

- Binormale

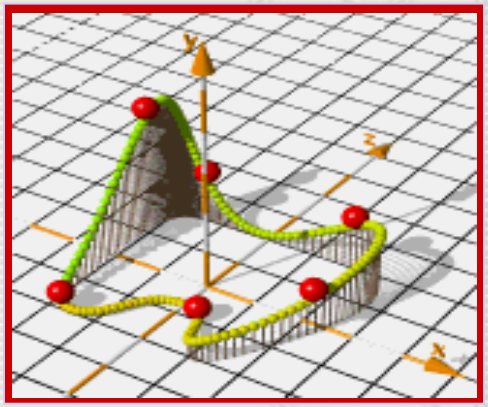
$$B(t) = T(t) \times N(t)$$

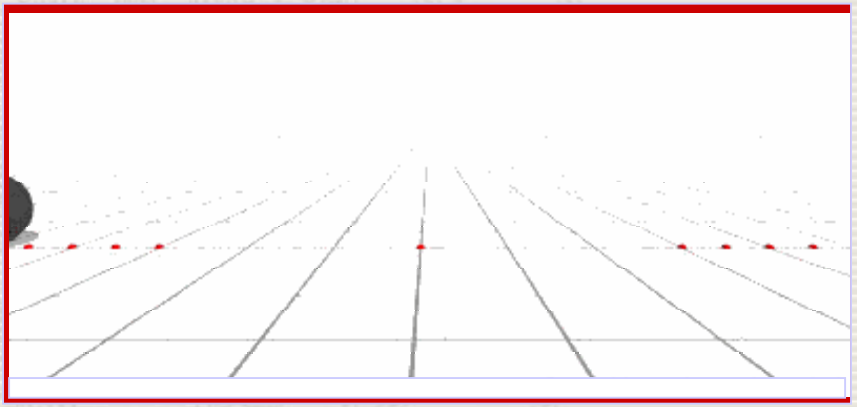


Fornisce un sistema di riferimento (frame) ortogonale in ogni punto della curva

Utilità del Frenet Frame

- Movimento camera
- Movimento di un oggetto lungo un percorso
- Movimento con una certa regola





Problemi: il Frenet frame diviene instabile nei punti di flesso o non definito quando

$$T'(t) = 0$$

Rappresentazione di forme: Curve 2D/3D

Problema: progettare una forma 2D o 3D matematicamente (modello matematico, modellazione geometrica)

Soluzione: si usa una funzione vettoriale a componenti polinomiali (spazio delle funzioni polinomiali); ma come?

1. Si specifichi una sequenza di punti $\mathbf{p}_i$, i = 1,..., N, (detti *control-point*);
2. Si definisca una parametrizzazione 1D;
3. Si definisca un mapping polinomiale (curva in forma parametrica), smooth/fair che *interpoli* o *approssimi* i control-point.

Curve di Bézier

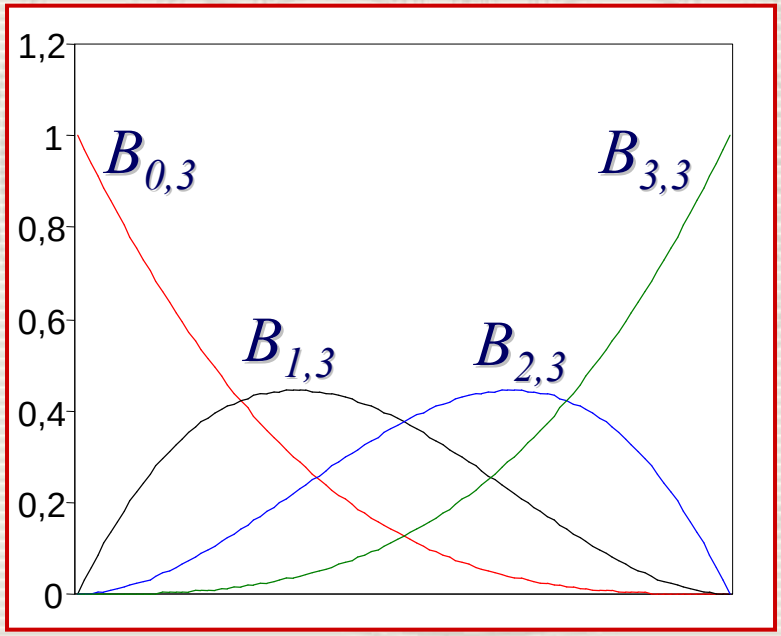
- Per lavorare nello spazio polinomiale dobbiamo scegliere una base di rappresentazione
- Differenti scelte di funzioni base permettono di rappresentare una stessa curva in modi differenti
  - La scelta di una base di funzioni è importante sia per questioni numeriche e computazionali, ma soprattutto perché i punti di controllo (i coefficienti) siano informativi sulla forma della curva
- Per questi motivi la scelta cade sulla base polinomiale nota come base di Bernstein;

Dati $n+1$ punti di controllo (CP) $\mathbf{p}_i$ (o poligonale di controllo), la curva è definita come:

$$C(t)=\sum_{i=0}^{n}\mathbf{p}_i B_{i,n}(t) \quad t\in[0,1] \qquad \text{dove} \quad B_{i,n}(t)=\binom{n}{i}t^i(1-t)^{n-i}$$

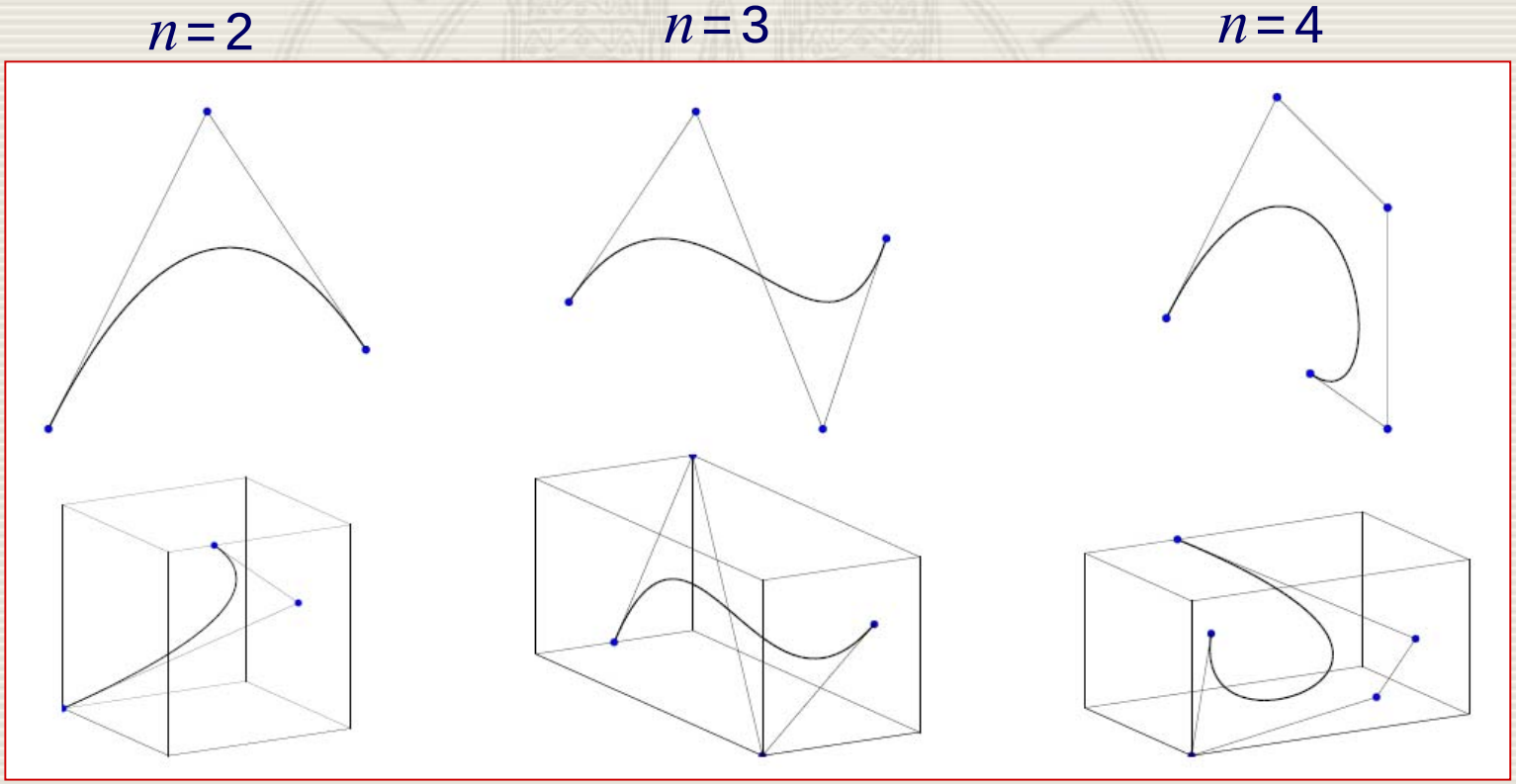
Curve di Bézier

- Le funzioni $B_{i,n}$ sono i *polinomi base di Bernstein* di grado n e sono dette *blending functions*; sono non negative e la loro somma vale 1.



Curve di Bézier

$$C(t) = \sum_{i=0}^{n} \mathbf{p}_i B_{i,n}(t) \quad t \in [0,1]$$

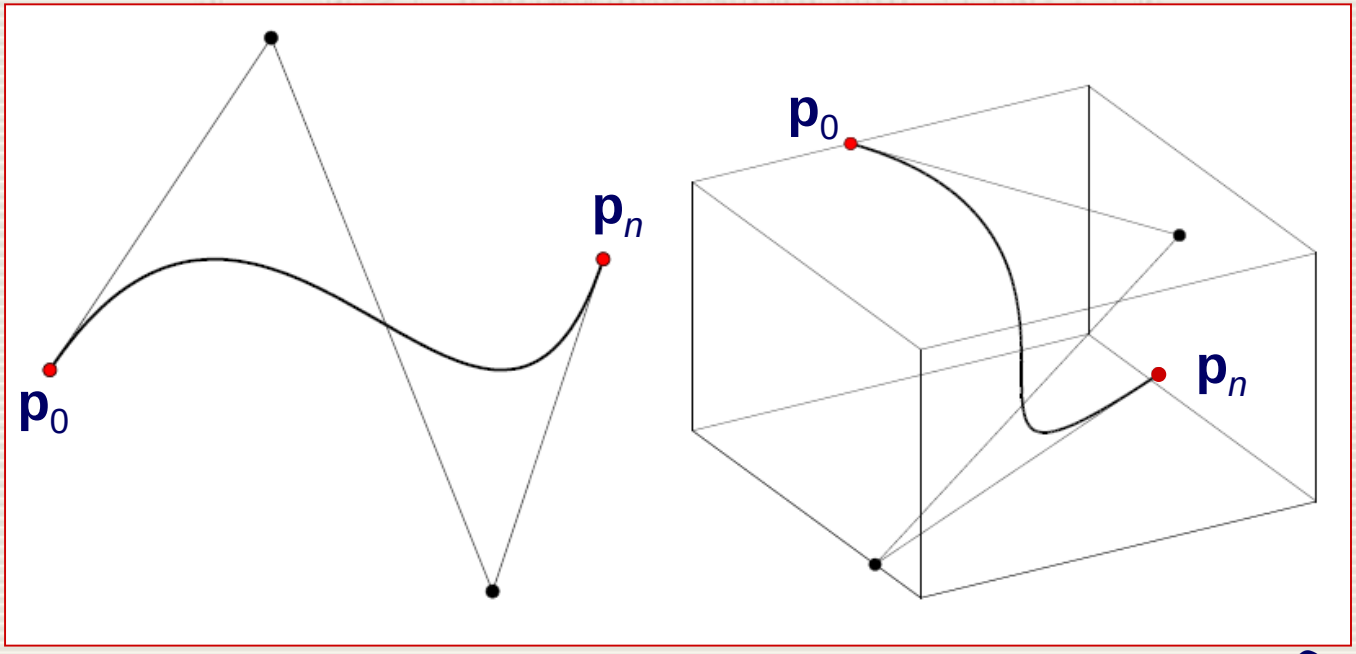


Proprietà delle Curve di Bézier

- La curva inizia dal primo punto di controllo e finisce nell'ultimo;
- La tangente alla curva nel primo punto ha la stessa direzione del primo segmento della poligonale di controllo;
- La tangente nell'ultimo punto ha la stessa direzione dell'ultimo segmento della poligonale di controllo;

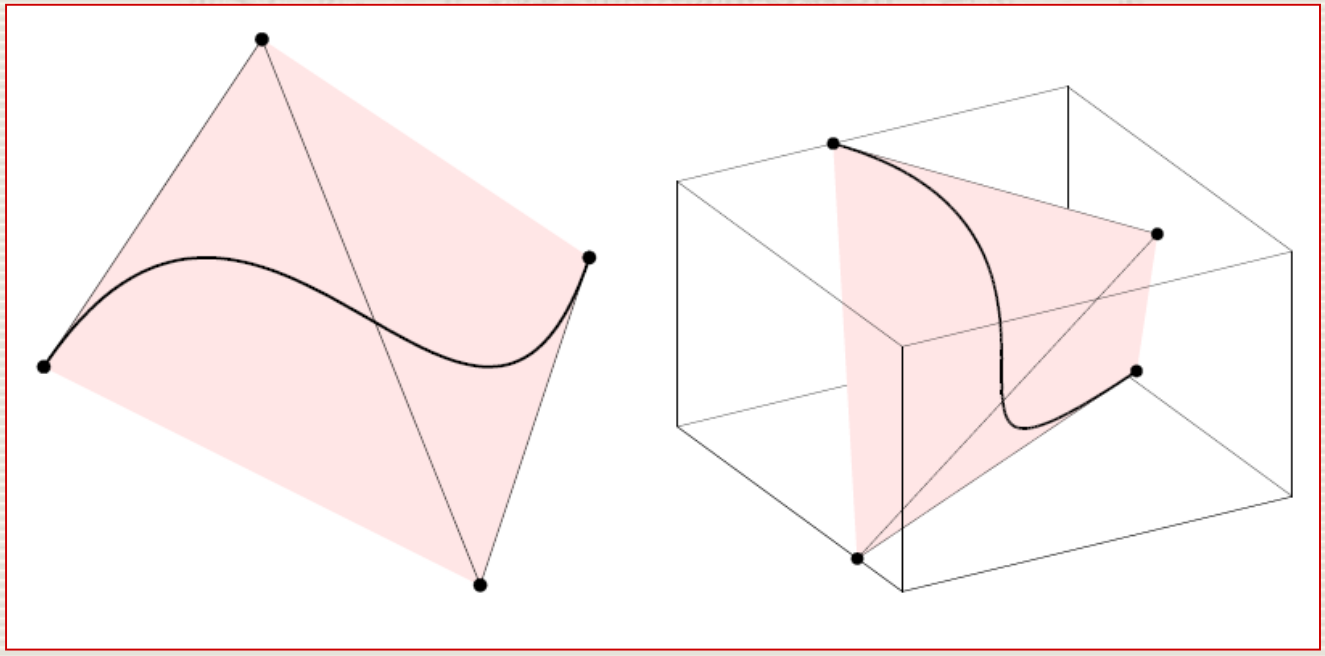
$$C(0)=\mathbf{p}_0 \qquad C(1)=\mathbf{p}_n$$

$$C'(0)=n(\mathbf{p}_1-\mathbf{p}_0) \qquad C'(1)=n(\mathbf{p}_n-\mathbf{p}_{n-1})$$



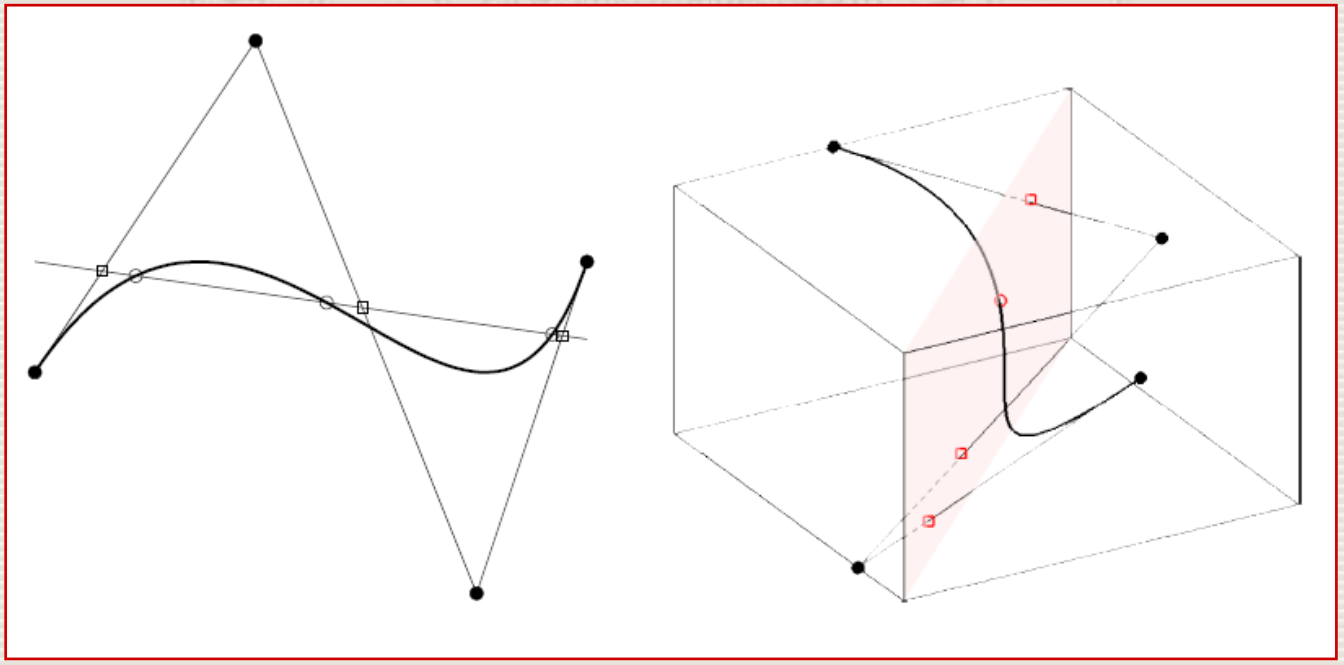
Proprietà delle Curve di Bézier

- La curva giace internamente al guscio convesso definito dai punti di controllo;



Proprietà delle Curve di Bézier

- $C(t)$ è approssimante in forma della poligonale di controllo;
- $C(t)$ è invariante per trasformazioni affini; in particolare per traslazione, scala, rotazione e deformazione lineare (shear);



Le Curve di Bézier e de Casteljau

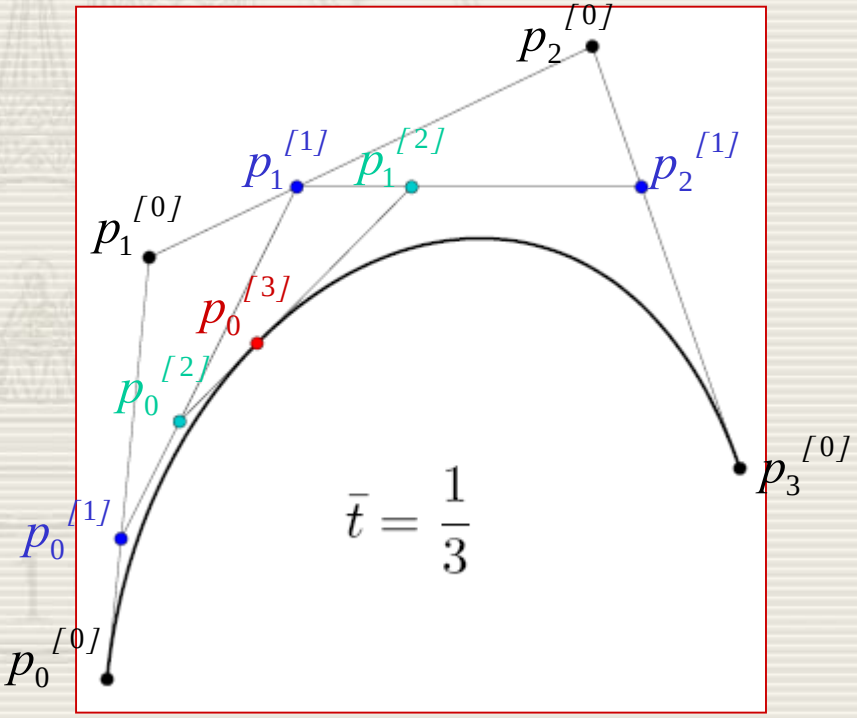
Il matematico francese de Casteljau, negli anni '60, diede una definizione di curva di Bézier basata su "corner cutting" successivi:

$$p_i^{[k]}(t) = (1-t)p_i^{[k-1]}(t) + tp_{i+1}^{[k-1]}(t) \qquad t \in [0,1]$$

dove $k = 1,..,n$
$i = 0,..,n\text{-}k$

con $p_i^{[0]}(t) = p_i$
$i = 0,..,n$

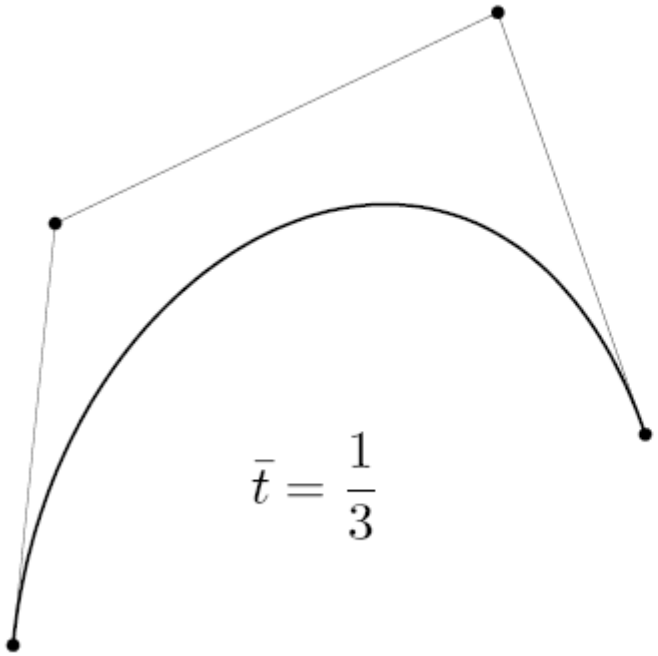
Questa definizione è un algoritmo numericamente stabile per il calcolo delle curve di Bézier.



Es. ***n=3, k=3***

Le Curve di Bézier e de Casteljau

$$p_i^{[k]}(t) = (1-t)p_i^{[k-1]}(t) + tp_{i+1}^{[k-1]}(t) \qquad t \in [0,1]$$



$$\bar{t} = \frac{1}{3}$$

$$p_0 = p_0^{[0]}$$

$$p_1 = p_1^{[0]}$$

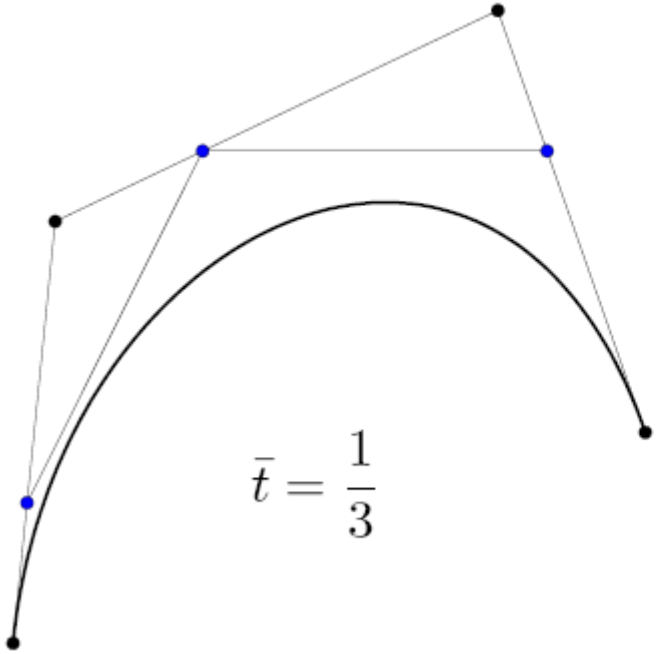
$$p_2 = p_2^{[0]}$$

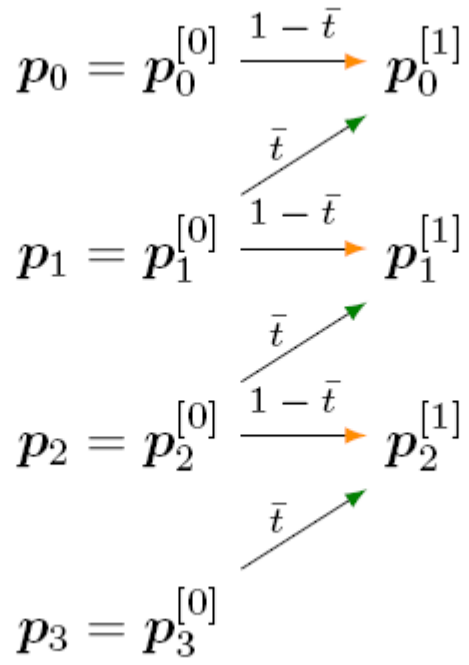
$$p_3 = p_3^{[0]}$$

$$p_i^{[0]}(\bar{t}) = p_i \qquad i = 0,..,3$$

Le Curve di Bézier e de Casteljau

$$p_i^{[k]}(t) = (1-t)p_i^{[k-1]}(t) + tp_{i+1}^{[k-1]}(t) \qquad t \in [0,1]$$

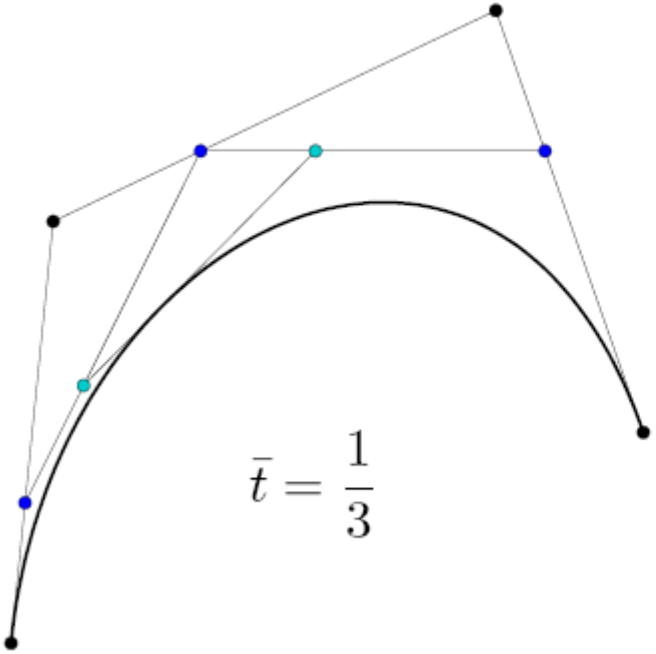


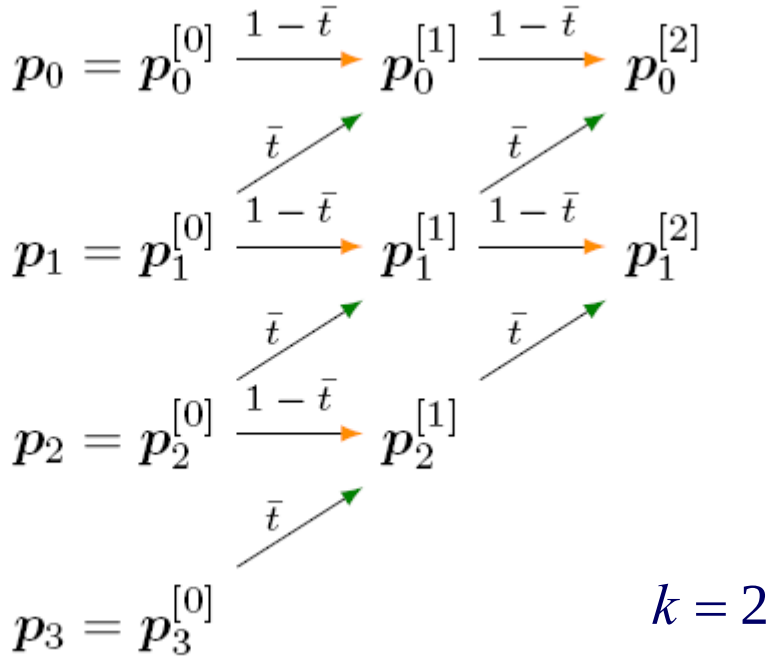


$k = 1$

Le Curve di Bézier e de Casteljau

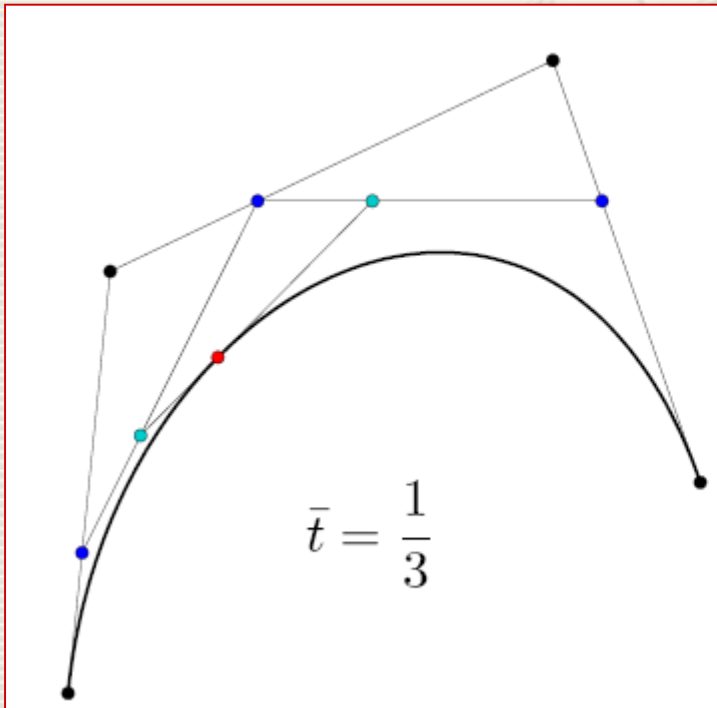
$$p_i^{[k]}(t) = (1-t)p_i^{[k-1]}(t) + tp_{i+1}^{[k-1]}(t) \qquad t \in [0,1]$$

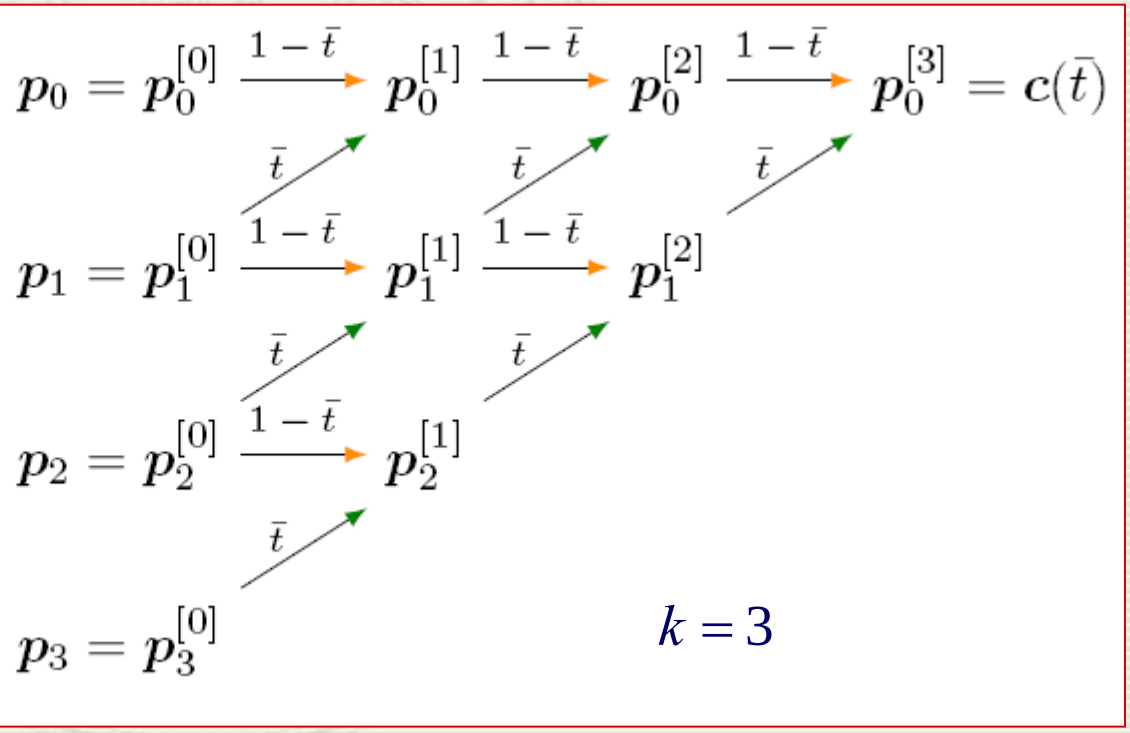




Le Curve di Bézier e de Casteljau

$$p_i^{[k]}(t) = (1-t)p_i^{[k-1]}(t) + tp_{i+1}^{[k-1]}(t) \qquad t \in [0,1]$$





Rendering Curve di Bézier

- Si valuti la curva in un numero fissato di valori parametrici e si disegni la polyline definita da questi punti della curva
- Vantaggi:
  - molto semplice
- Svantaggi:
  - Costoso per valutare la curva in molti punti
  - Non è facile determinare in quanti valori valutare e dove valutare lungo la curva
  - Non è facile renderlo adattivo; in particolare è difficile misurare la distanza della polyline dalla curva.

Codice di Esempio

DEMO:

SDL2_prg1516/SDL2prg2_gl/bezie3d

Esempio

Vogliamo progettare un moto rettilineo di un corpo da una posizione Q_0 ad una Q_1, con velocità iniziale V_0 e finale V_1, in un tempo di 5 secondi, nella forma di Bezier. Sarà:

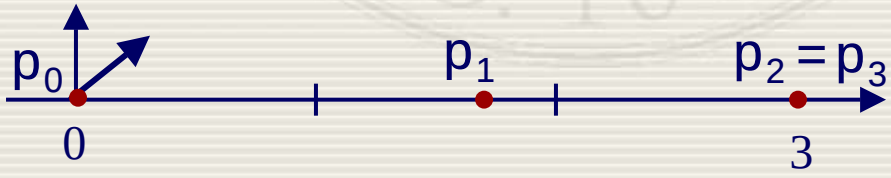
$$C(t)=\sum_{i=0}^{3}\mathbf{p}_i B_{i,3}(t) \qquad t\in[0,5]$$

dove $Q_0=C(0)=\mathbf{p}_0$ $\quad Q_1=C(5)=\mathbf{p}_3$

$V_0=C'(0)=3/5(\mathbf{p}_1-\mathbf{p}_0)$ $\quad V_1=C'(5)=3/5(\mathbf{p}_3-\mathbf{p}_2)$

Sia $Q_0=(0,0,0)$, $Q_1=(3,0,0)$, $V_0=(1,0,0)$, $V_1=(0,0,0)$, allora

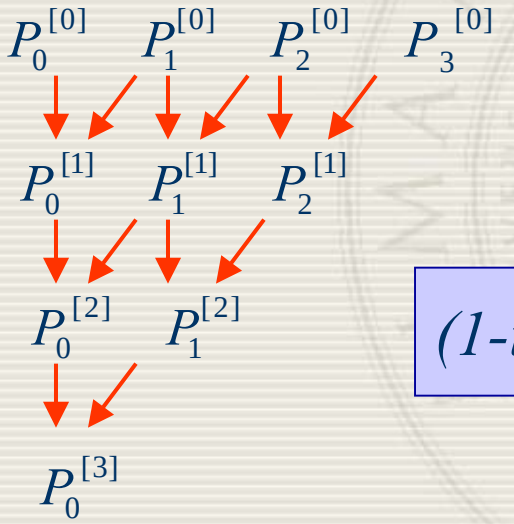
$p_0=(0,0,0)$, $p_1=p_0+5/3V_0=(5/3,0,0)$
$p_3=(3,0,0)$, $p_2=p_3-5/3V_1=(3,0,0)$



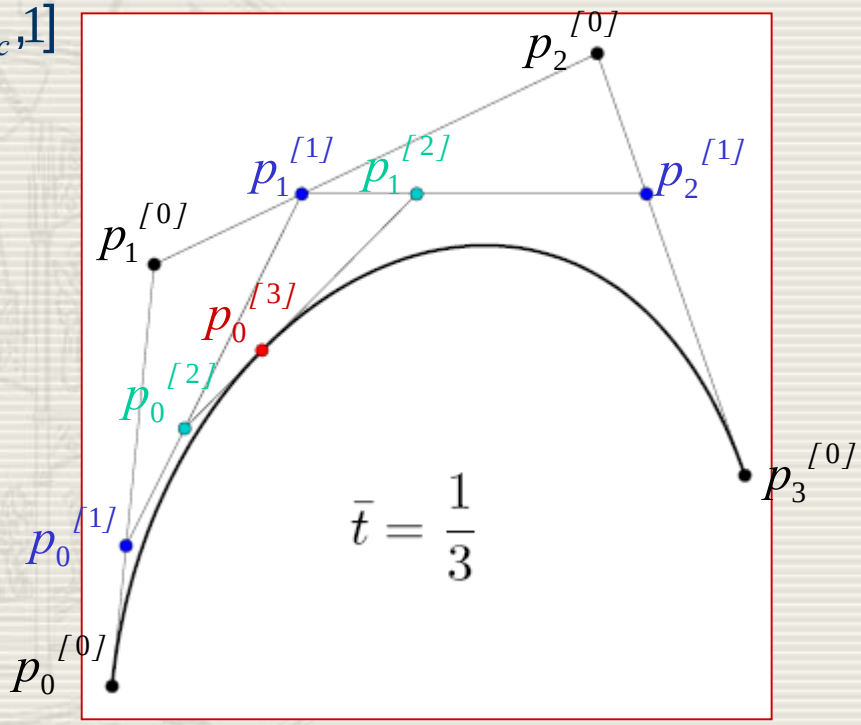
Le Curve di Bézier e la Suddivisione

La definizione o algoritmo di valutazione di de Casteljau di una curva di Bézier in corrispondenza di un punto tc, fornisce oltre al valore della curva anche i punti di controllo delle curve di Bézier corrispondenti agli intervalli $[0,t_c]$ e $[t_c,1]$

Vediamo nel caso $n=3$:



$(1\text{-}tc)$ ↓ ↙ tc



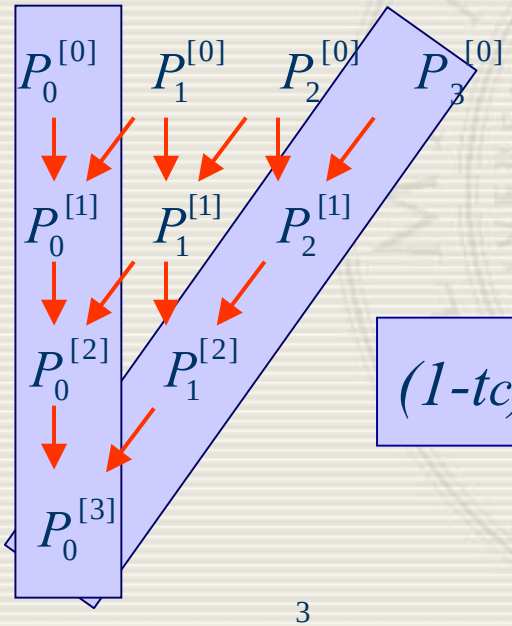
$$C(t) = \sum_{i=0}^{3} p_i^{[0]} B_{i,3}(t) = \begin{cases} \sum_{i=0}^{3} p_0^{[i]} B_{i,3}(t) & t \in [0,t_c] \\ \sum_{i=0}^{3} p_i^{[3-i]} B_{i,3}(t) & t \in [t_c,1] \end{cases}$$

$t \in [0,1]$

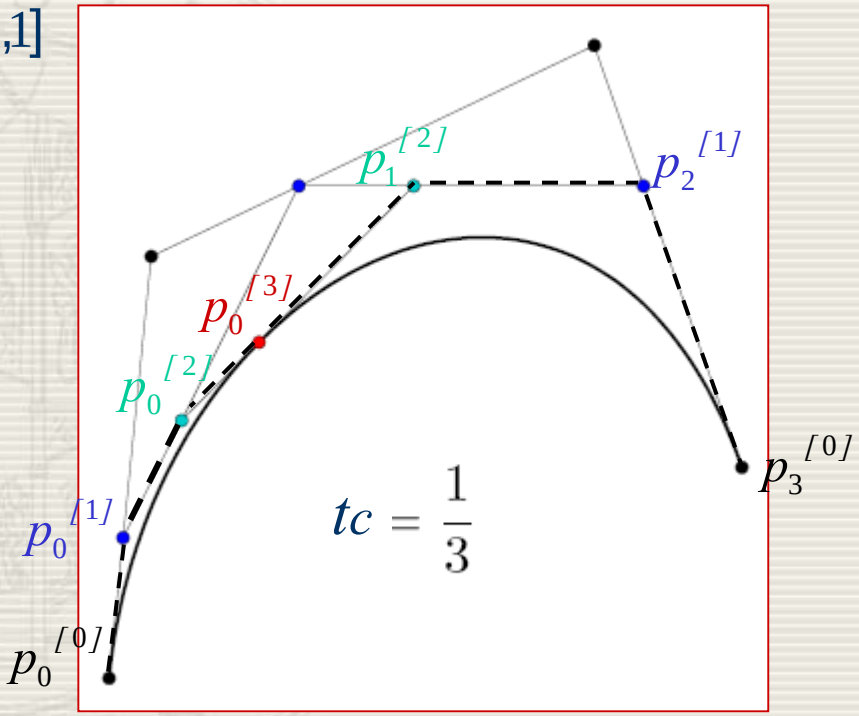
Le Curve di Bézier e la Suddivisione

La definizione o algoritmo di valutazione di de Casteljau di una curva di Bézier in corrispondenza di un punto tc, fornisce oltre al valore della curva anche i punti di controllo delle curve di Bézier corrispondenti agli intervalli $[0,t_c]$ e $[t_c,1]$

Vediamo nel caso $n=3$:



$(1\text{-}tc)$ ↓ ↙ tc



$$C(t) = \sum_{i=0}^{3} p_i^{[0]} B_{i,3}(t) = \begin{cases} \sum_{i=0}^{3} p_0^{[i]} B_{i,3}(t) & t \in [0,t_c] \\ \sum_{i=0}^{3} p_i^{[3-i]} B_{i,3}(t) & t \in [t_c,1] \end{cases}$$

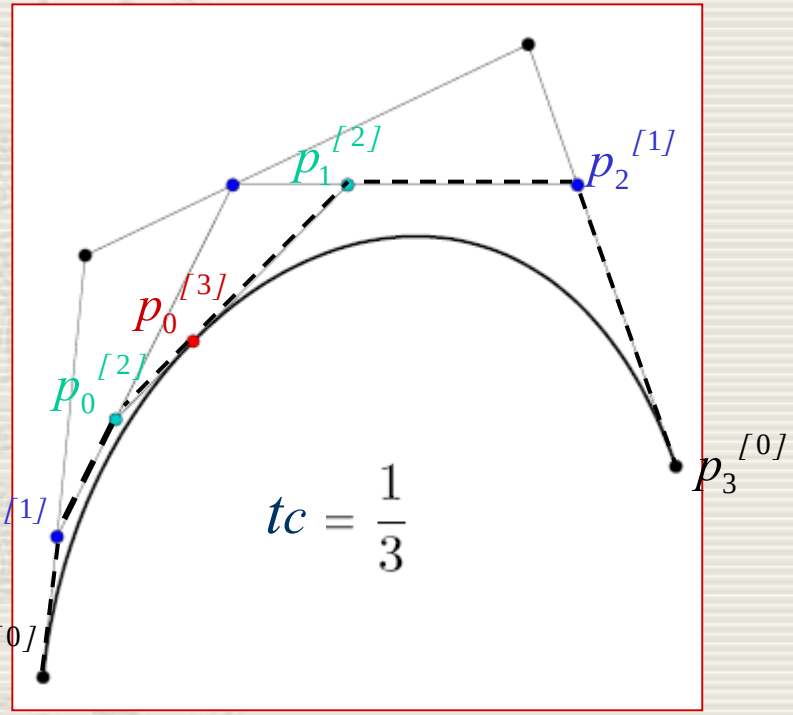
$t \in [0,1]$

Le Curve di Bézier e la Suddivisione

La definizione o algoritmo di valutazione di de Casteljau di una curva di Bézier in corrispondenza di un punto tc, fornisce oltre al valore della curva anche i punti di controllo delle curve di Bézier corrispondenti agli intervalli $[0,t_c]$ e $[t_c,1]$

$$C(t)=\sum_{i=0}^{3} p_i^{[0]} B_{i,3}(t) = \begin{cases} \sum_{i=0}^{3} p_0^{[i]} B_{i,3}(t) & t\in[0,t_c] \\ \sum_{i=0}^{3} p_i^{[3-i]} B_{i,3}(t) & t\in[t_c,1] \end{cases}$$

$t \in [0,1]$



La suddivisione permette di calcolare in modo semplice la tangente alla curva in ogni punto

$$C'(t_c) = n(p_0^{[n]} - p_0^{[n-1]}) = n(p_1^{[n-1]} - p_0^{[n]})$$

$t \in [0,1]$

Rendering Curve di Bézier

- Ricordiamo che una curva di Bézier giace interamente nel guscio convesso dei suoi punti di controllo
- Se i punti di controllo sono quasi allineati, allora il guscio convesso è quasi un segmento che approssima bene la curva
- Ancora, una curva di Bézier può essere divisa in due curve di Bézier più piccole che rappresentano esattamente la curva originale

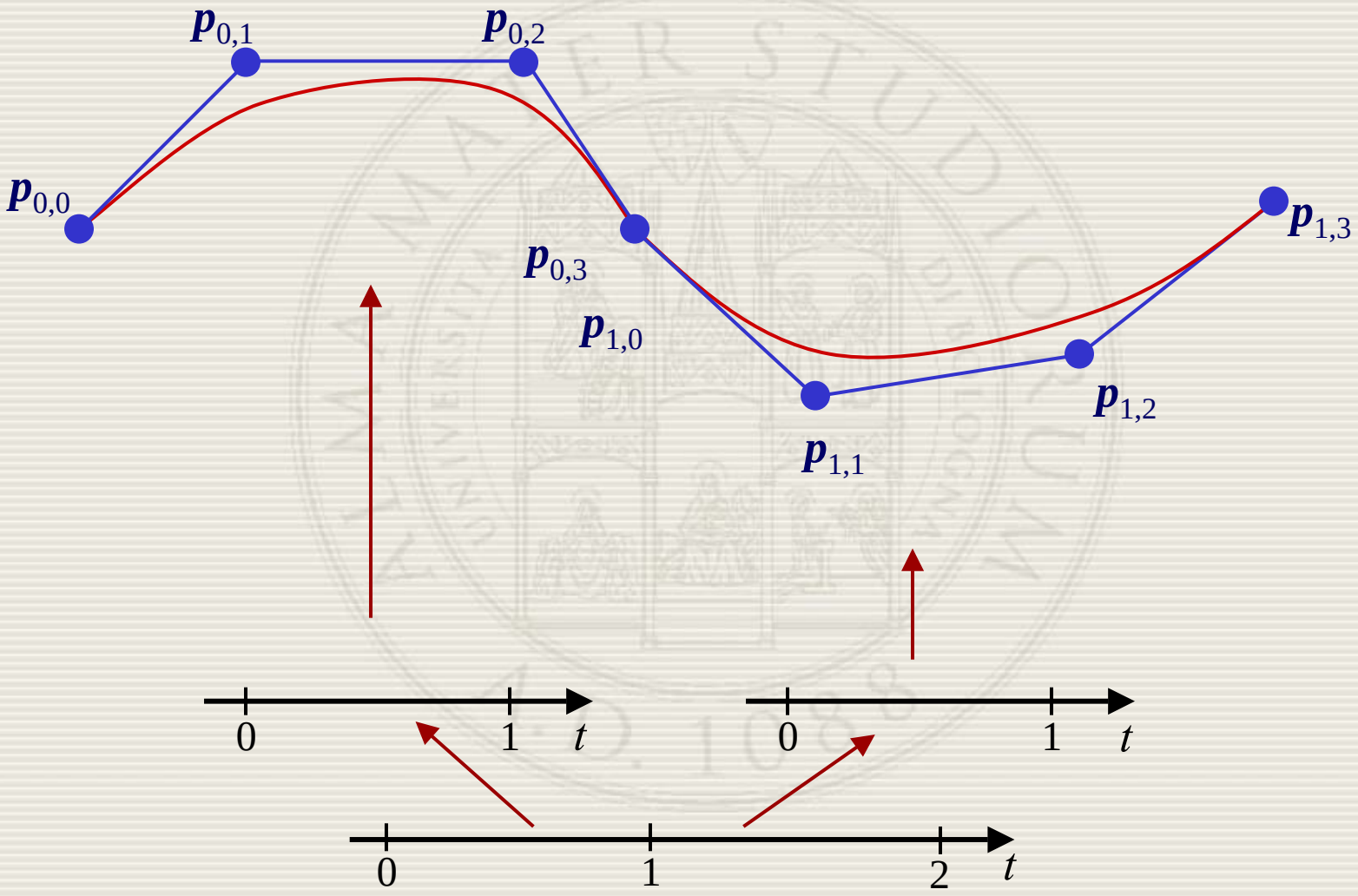
- Questo suggerisce il seguente algoritmo di disegno:
  - dividi ricorsivamente la curva in due sotto-curve (algoritmo di suddivisione)
  - ferma il processo quando i punti di controllo di ogni sotto-curva sono quasi allineati
  - disegna il segmento di estremi il primo ed ultimo punto di controllo

Curve Complesse

- Una singola curva di Bézier può rappresentare solo una limitata gamma di forme
- Una soluzione potrebbe essere aumentare il grado
  - questo aumenta le possibilità, ma al costo di più punti di controllo e polinomi di grado maggiore
  - il controllo è globale; un punto di controllo influenza l'intera curva

- In alternativa, la soluzione più comune è unire insieme più curve di Bézier di grado basso in una *piecewise curve (curva a tratti)*
  - una curva complessa in forma, può essere pensata in più tratti, ciascuno dei quali rappresentabile con una curva di Bézier di grado basso (per es. cubica)
  - *Controllo Locale*: ogni punto di controllo influenza solo una parte limitata della curva
  - L'interazione e la modellazione sono più semplici

Curva di Bézier a tratti

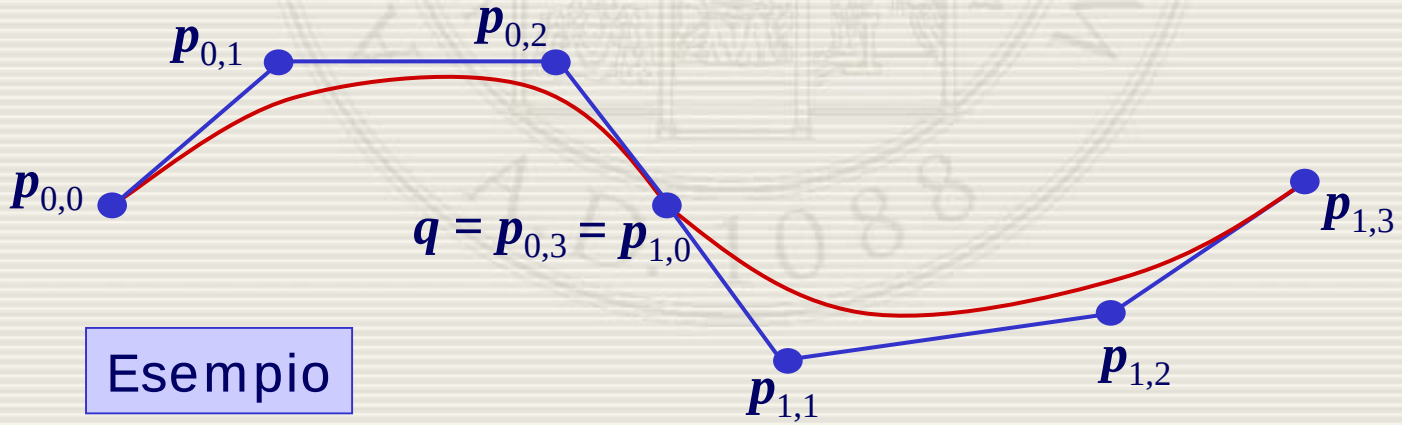


Continuità

- Quando due curve vengono unite, solitamente si vuole un ordine di continuità negli estremi:
  - C^0, “C-zero”, continuità point-wise, le curve condividono lo stesso punto dove si uniscono
  - C^1, “C-one”, continuità della derivata, le curve hanno la stessa derivata parametrica dove si uniscono
  - C^2, “C-two”, continuità della derivata seconda, le curve hanno la stessa derivata seconda dove si uniscono
  - C^k possibile continuità più alta
- Come facciamo ad assicurare che due curve di Bézier siano C^0, C^1, C^2 dove si uniscono?

Imporre la Continuità: esempio

- Curve di Bézier cubiche:
  - Per definizione interpolano i loro punt di controllo estremi, quindi si ha C^0 semplicemente uguagliando i punti di controllo estremi: $\boldsymbol{q} = \boldsymbol{p}_{0,3} = \boldsymbol{p}_{1,0}$
  - La continuità C^1 si ottiene ponendo $\boldsymbol{q} = \boldsymbol{p}_{0,3} = \boldsymbol{p}_{1,0}$, e facendo sì che $\boldsymbol{p}_{0,2}$, $\boldsymbol{q}$ e $\boldsymbol{p}_{1,1}$ siano allineati, e precisamente $\boldsymbol{q} - \boldsymbol{p}_{0,2} = \boldsymbol{p}_{1,1} - \boldsymbol{q}$
  - *La continuità* C^2 viene da ulteriori condizioni su $\boldsymbol{p}_{0,1}$ e $\boldsymbol{p}_{1,2}$



Esempio

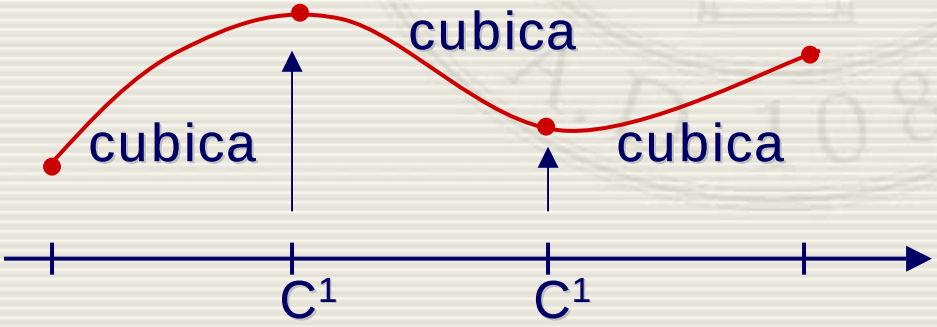
Problemi con Curve di Bézier

- Una curva complessa richiede molti segmenti di curve di Bézier
- Mantenere la continuità richiede vincoli sulla posizione dei punti di controllo
  - L'utente non può muovere arbitrariamente i punti di controllo e automaticamente mantenere la continuità
  - I vincoli devono essere mantenuti esplicitamente
  - Non risulta intuitivo gestire punti di controllo che risultano vincolati

Curva di Bézier a tratti

Si noti che nel caso C^0, la curva di Bézier a tratti dell' esempio resta definita da 7 CP (1 in meno);
nel caso C^1, la curva resta definita da 6 CP (2 in meno), e così via.
Allora se si vuole costruire una curva a tratti con una assegnata continuità è semplice determinare la dimensione dello spazio relativo o in altre parole quante infinità di curve si avranno.

Esempio: se si vuole progettare una curva a 3 tratti cubici, con continuità C^1 in ogni punto di raccordo, avremo 3x4-4, cioè 8 gradi di libertà od anche uno spazio di dimensione 8.



n=grado polinomi
k=ordine di continuità
con k<n

Dai Polinomi alle Spline

Definito il numero di tratti (N+1), il grado n dei polinomi e l'ordine di continuità k_i $i=1,..,N$ (con $k_i < n$) in ogni punto di raccordo, allora resta definito uno spazio funzionale S, che chiameremo spazio spline, e la sua dimensione è:

$$\dim(S) = (n+1)(N+1) - \sum_{i=1}^{N} (k_i + 1)$$

E' possibile determinare una base, tipo Bernstein, per questo spazio polinomiale a tratti?

Sì e si chiama base delle funzioni B-spline.

Le curve spline sono funzioni vettoriali le cui componenti sono funzioni scalari di uno spazio spline

Curve Spline

- Le curve spline sono, più semplicemente, una rappresentazione matematica compatta di curve polinomiali di grado n a tratti definite su una sequenza di intervalli parametrici, detta partizione nodale

Definiamo una partizione nodale di [a,b] con una sequenza di punti x_i detti nodi, tali che:

$$x_0 = a < x_1 < x_2 < \ldots < x_N < x_{N+1} = b$$

- Ci sono molti tipi di curve spline: possono essere differenti per grado (lineare, quadratica, cubica, …) e partizione nodale (uniforme o non-uniforme)
- In genere l'ordine di continuità k_i è definibile arbitrariamente da nodo a nodo ($k_i<n$); spesso ci si limita a spline con massimo ordine di continuità ossia $k_i=n$-1 per ogni $i=1,\ldots,N$, allora
  - spline lineari C^0, quadratiche C^1, cubiche C^2, ecc.
  - tale spazio avrà dimensione $N+n+1$

Curve Spline

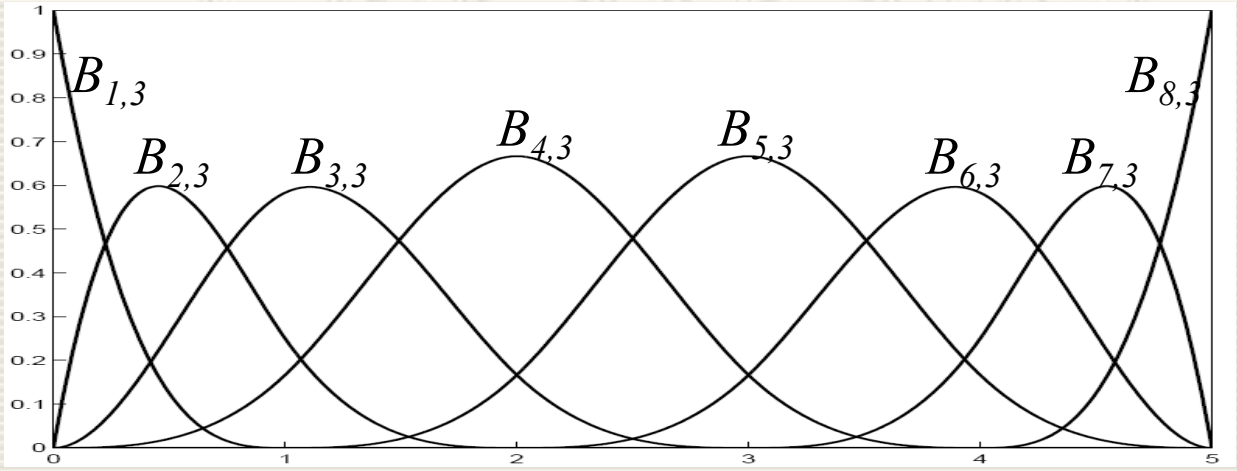
- L'espressione matematica rassomiglia a quella di una curva di Bézier, ma con funzioni B-spline definite sulla partizione estesa di nodi

$$\{t_i\}_{i=1,\ldots,N+2(n+1)}$$

tale che:

$t_1 = \ldots = t_{n+1} = x_0$, $t_{n+1+i} = x_i$, $i = 1,\ldots,N$, $t_{N+n+2} = \ldots = t_{N+2(n+1)} = x_{N+1}$

le funzioni B-spline $B_{i,n}(t)$ sono a supporto compatto, cioè sono nulle fuori dell'intervallo $[t_i, t_{i+n+1}]$ con n il grado polinomiale



$n = 3$

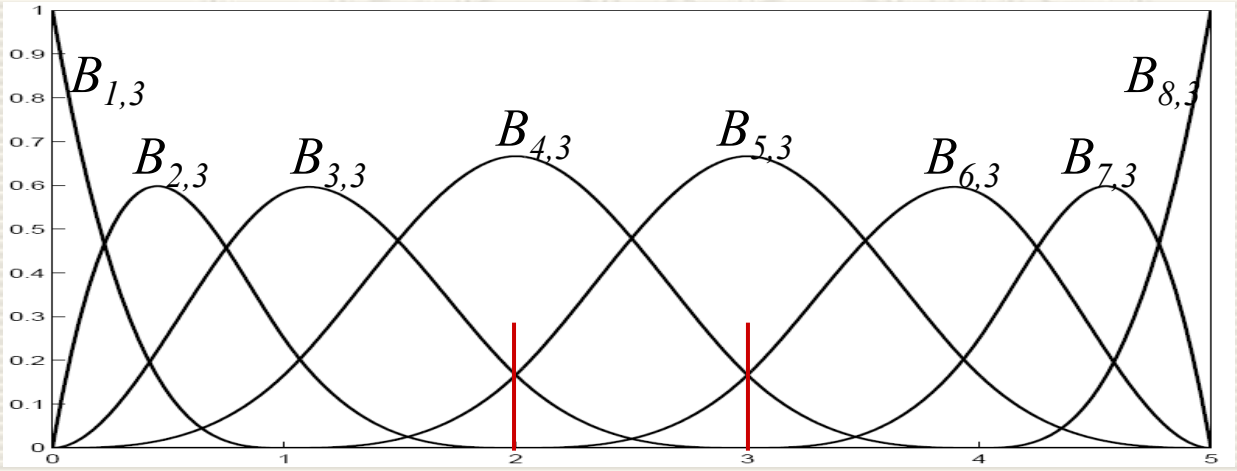
$t_1 = \ldots = t_4 = 0$ $t_5 = 1$ $t_6 = 2$ $t_7 = 3$ $t_8 = 4$ $t_9 = \ldots = t_{12} = 5$

Curve Spline

Sono ancora chiamate *blending functions*, e descrivono come miscelare (blend) i punti di controllo per dar luogo alla curva

$$C(t) = \sum_{i=1}^{N+n+1} p_i B_{i,n}(t)$$

Se ci restringiamo ad un intervallo nodale, per esempio [2,3], per il fatto che sono a supporto compatto, le uniche B-spline non nulle saranno in numero di $n+1$ (4 nell'esempio) e per l'esattezza le $B_{i,3}(t)$ per i=3,4,5,6;



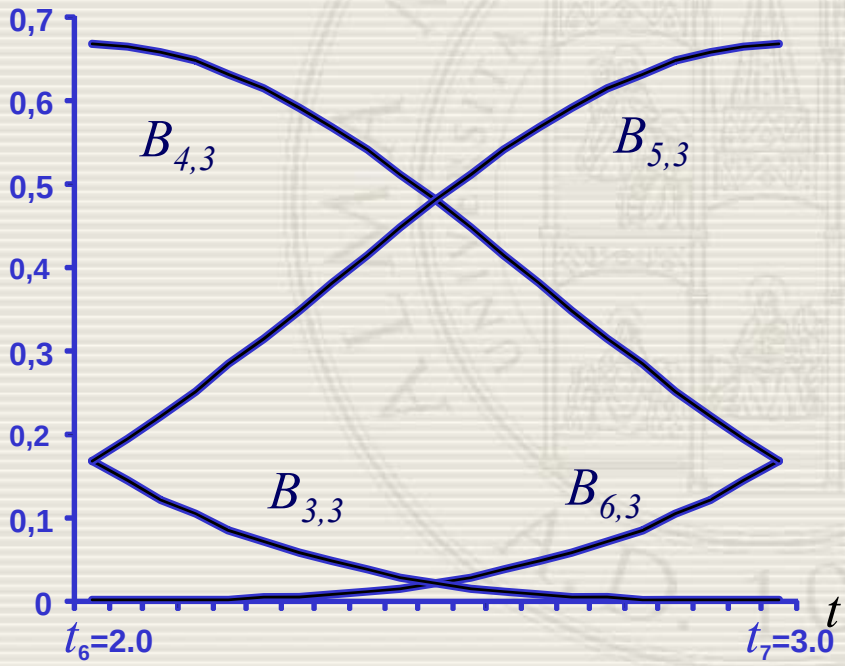
$n=3$

$t_1 = \ldots = t_4 = 0$ $t_5 = 1$ $t_6 = 2$ $t_7 = 3$ $t_8 = 4$ $t_9 = \ldots = t_{12} = 5$

Valutazione di Curve Spline

L'osservazione precedente permette di valutare in modo efficiente una curva spline.

Assegnato il parametro t, si determina l'intervallo nodale in cui è contenuto, sia $[t_k, t_{k+1}]$, quindi:



$$C(t) = \sum_{i=k-n}^{k} p_i B_{i,n}(t)$$

$$= \sum_{i=3}^{6} p_i B_{i,n}(t)$$

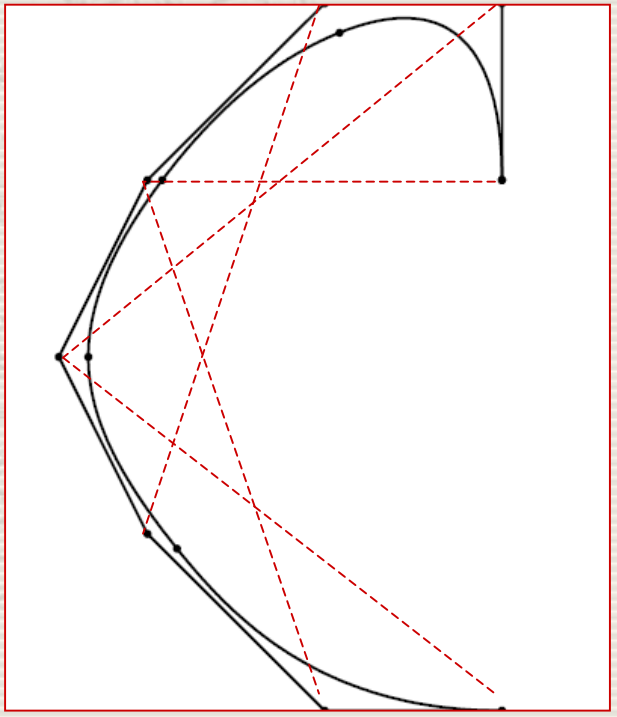
Cioè la valutazione di un punto della curva spline costa come la valutazione di una curva di Bézier a partire da $n+1$ punti di controllo.

Si dice che le curve spline sono a controllo locale

Curve Spline

- La curva spline giace all'interno del guscio convesso? interpola gli estremi?

- Le funzioni B-spline hanno somma 1 e sono non negative;
  - La curva è quindi sempre interna al guscio convesso dei suoi punti di controllo
  - La curva ha la proprietà del guscio convesso locale, cioè ogni tratto è interno al guscio convesso dato dagli $n+1$ punti di controllo che lo definiscono.
- La curva interpola i suoi estremi
- La curva è approssimante in forma della poligonale
- La curva è invariante per trasforma-zioni affini



Curva Spline Non Uniforme: riassumiamo

- Curva spline:

$$C(t) = \sum_{i=1}^{N+n+1} p_i B_{i,n}(t)$$

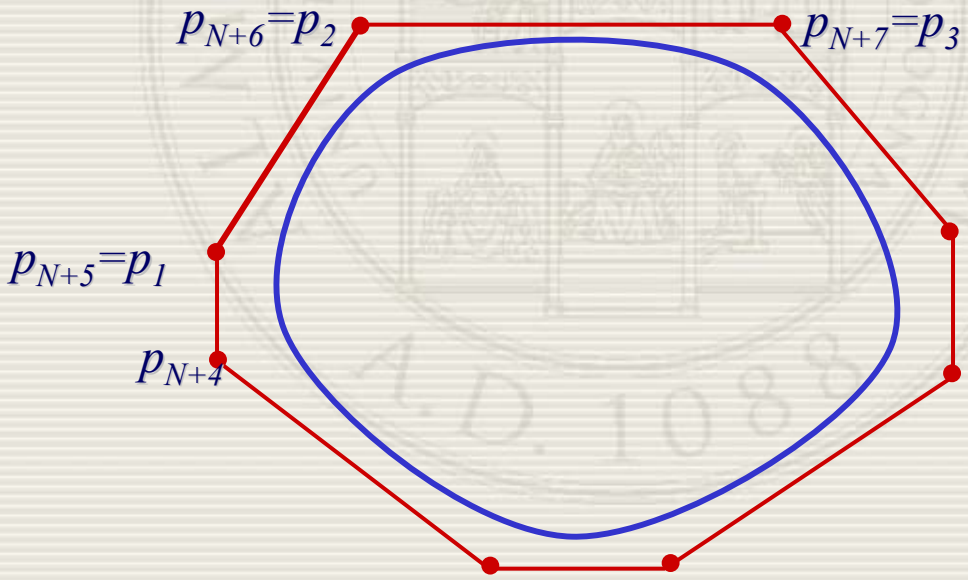
  - $N+n+1$ è il numero totale di punti di controllo
  - $\boldsymbol{n}$ è il grado della curva
  - $\boldsymbol{B_{i,n}}$ sono le B-spline non uniformi (*blending functions*) di grado $\boldsymbol{n}$
  - p_i sono i punti di controllo
  - Ciascuna $\boldsymbol{B_{i,n}}$ è non nulla solo in un certo intervallo $[t_i, t_{i+n+1}]$, detto supporto, così che la curva ha controllo locale

Curva Chiusa

- Per creare una curva spline chiusa, si replichino i primi punti di controllo alla fine della sequenza:

  $p_1, \ldots, p_{N+n+1}, p_1, p_2, p_3$ *(esempio n=3)*

  se ne devono replicare tanti quanto il grado della curva;
- Inoltre si deve definire una partizione nodale estesa periodica.



Algoritmo di Valutazione di de Boor (curve spline)

Siano dati i punti di controllo $p_i \; i = 1,..,N+n+1$ della curva $C(t)$ (n il grado polinomiale) e la partizione estesa di nodi

$$\{t_i\}_{i=1,...,N+2(n+1)} \qquad [a,b] = [t_{n+1}, t_{N+1}]$$

Si vuole valutare la curva $C(t)$ per $t \in [t_l, t_{l+1}] \quad l \in \{n+1,...,N\}$

$$\mathbf{p}_i^{[r]}(t) = (1-\alpha_i^{[r]})\mathbf{p}_{i-1}^{[r-1]}(t) + \alpha_i^{[r]}\mathbf{p}_i^{[r-1]}(t)$$

$$r = 1,..,n$$

$$i = l-n+r,..,l$$

$$\alpha_i^{[r]} = \frac{t - t_i}{t_{i+n+1-r} - t_i}$$

$$\mathbf{p}_i^{[0]}(t) = \mathbf{p}_i \qquad i = 1,..,N+n+1$$

$\mathbf{p}_1^{[0]} \ldots \; \mathbf{p}_{l-n}^{[0]} \; \mathbf{p}_{l-n+1}^{[0]} \; \ldots \; \mathbf{p}_l^{[0]} \; \ldots \mathbf{p}_{N+n+1}^{[0]}$

$\mathbf{p}_{l-n+1}^{[1]} \qquad \mathbf{p}_l^{[1]}$

$\mathbf{p}_{l-n+2}^{[2]} \qquad \mathbf{p}_l^{[2]}$

$\mathbf{p}_l^{[n]} = C(t)$

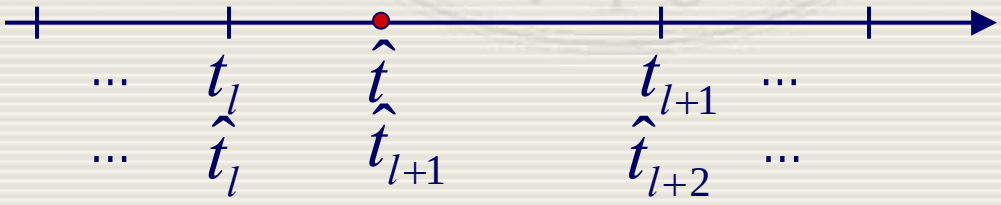
Knot-insertion

Data una curva spline C(t) in uno spazio spline *S* è possibile rappresentarla esattamente in uno spazio spline $\hat{S}$ ottenuto da *S* per inserzione di un nodo $\hat{t} \in [t_l, t_{l+1})$

Se $\{t_i\}_{i=1,\ldots,N+2(n+1)}$ è la partizione estesa in S, sia

$\{\hat{t}_i\}_{i=1,\ldots,N+2(n+1)}$ la partizione estesa in $\hat{S}$ dove

$$\hat{t}_i = \begin{cases} t_i & i \le l \\ \hat{t} & i = l+1 \\ t_{i-1} & i > l+2 \end{cases}$$



Knot-insertion: Algoritmo di Bohm

Sia $$C(t) = \sum_{i=1}^{N+n+1} p_i B_{i,n}(t)$$

la curva spline in S, allora per knot-insertion sarà

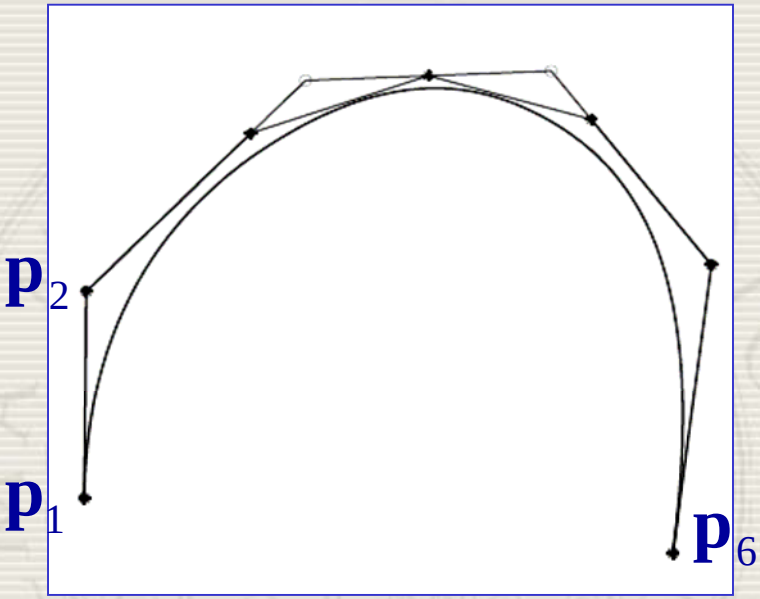
$$C(t) = \sum_{i=1}^{N+n+2} \hat{p}_i B_{i,n}(t)$$

dove

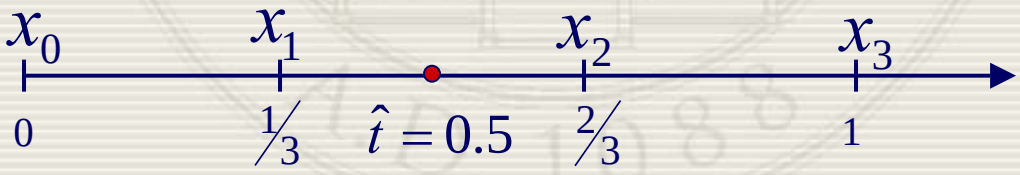
$$\hat{p}_i = \begin{cases} p_i & i \le l-n \\ (1-\lambda_i)p_{i-1} + \lambda_i p_i & l-n+1 \le i \le l+1 \\ p_{i-1} & i > l+1 \end{cases}$$

con $\lambda_i = \dfrac{\hat{t} - t_i}{t_{i+n+1} - t_i}$ $\qquad \hat{t} \in [t_l, t_{l+1})$

Knot-insertion : Algoritmo di Bohm



$$n = 3, N = 2, N + n + 1 = 6$$



$$t_1 = \ldots = t_4 \qquad t_5 \qquad t_6 \qquad t_7 = \ldots = t_{10}$$

Knot-insertion

L'algoritmo di knot-inserion non modifica la curva e può essere utile per:

- Editare la curva: si aumentano i punti di controllo con i quali si può cambiare la forma della curva;
- Convertire la curva spline in una sequenza di curve di Bézier (knot-insertion multiplo);
- Controllare la continuità della curva;
- Disegnare la curva per raffinamento.

Knot-insertion : raffinamento

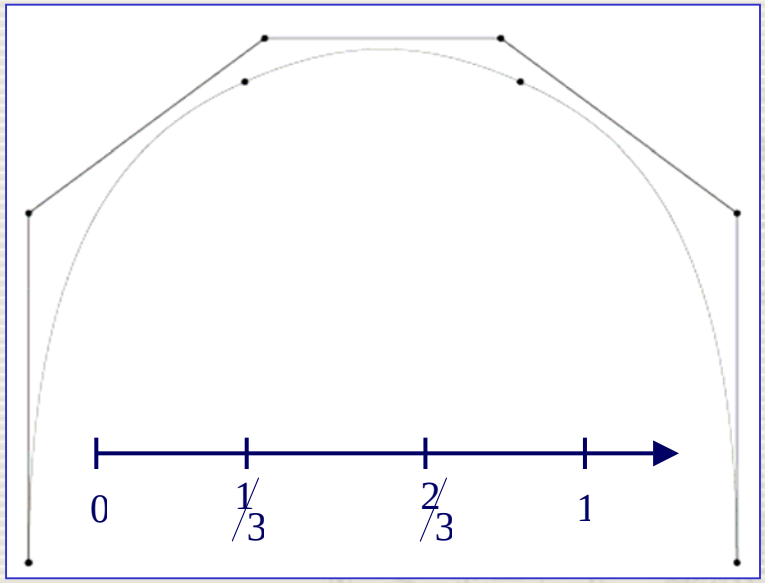
Con raffinamento si intende il processo di inserire un nodo in ogni intervallo nodale e più precisamente in corrispondenza del suo punto medio.

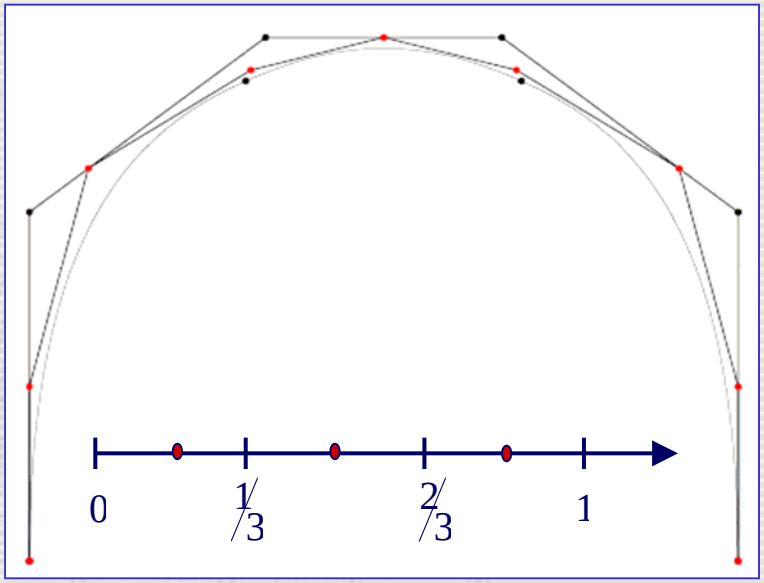
Così facendo la poligonale di controllo della curva viene modificata in una poligonale con più punti di controllo e più prossima alla curva (il knot-insertion è un corner-cutting algorithm);

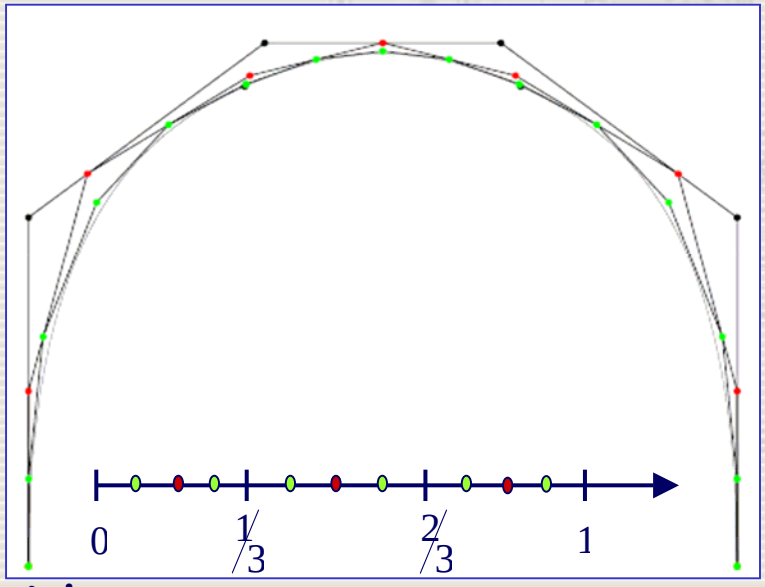
Ripetendo il procedimento di raffinamento più volte si ottiene una successione di poligonali di controllo che converge alla curva stessa (convergenza alla curva);

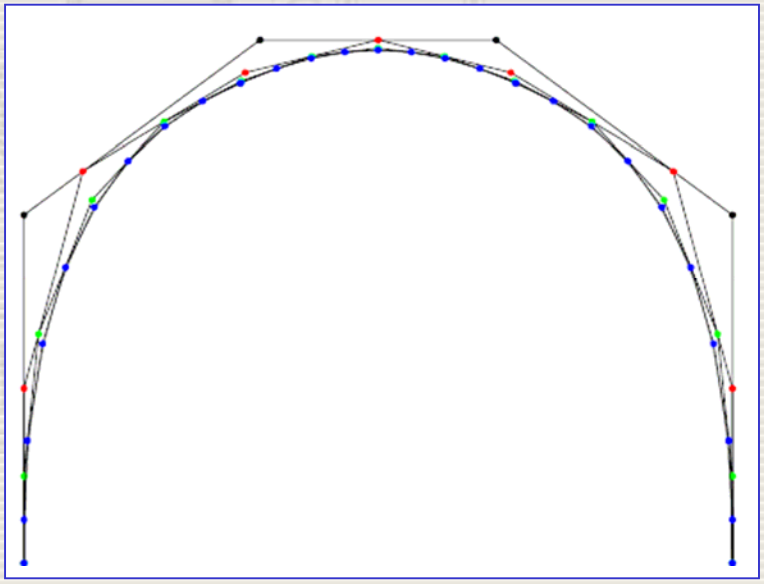
Dopo un certo numero finito di passi, la poligonale è così prossima alla curva che può essere disegnata in sua vece.

Knot-insertion : raffinamento









Curve NURBS (Razionali)

- Una curva NURBS 3D può essere vista come la proiezione di una curva spline 4D nello spazio 3D
  - Esattamente come la proiezione in uno spazio affine:

$$[x(t), y(t), z(t), w(t)] \rightarrow \left[\frac{x(t)}{w(t)}, \frac{y(t)}{w(t)}, \frac{z(t)}{w(t)} \right]$$

  - $x(t), y(t), z(t)$ e $w(t)$ sono funzioni spline non-uniformi
- Vantaggi:
  - Invarianti per proiezione prospettica, così che possono essere valutate nello spazio del piano di proiezione
  - Possono rappresentare esattamente sezioni coniche: parabola, ellisse (circonferenza), iperbole
    - Le curve spline (polinomiali) possono solo approssimare le coniche