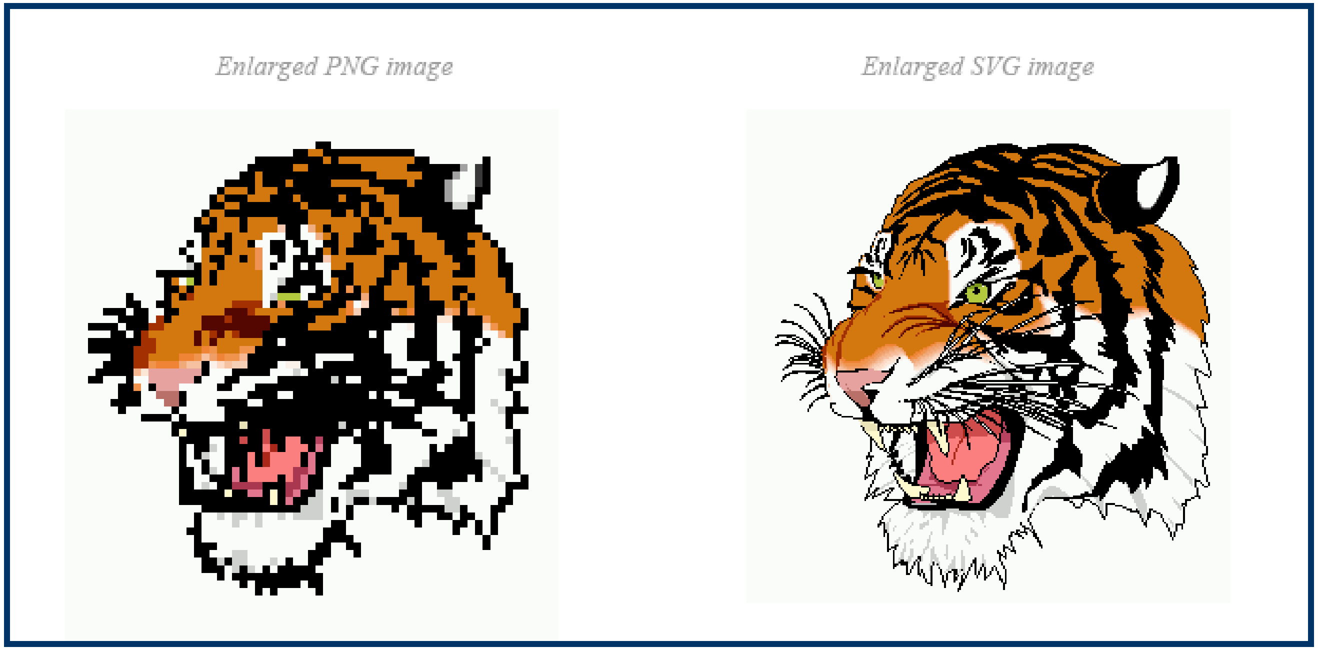
Curve di Bézier e Grafica Vettoriale



Funzioni Vettoriali o Curve in forma parametrica

Consideriamo un punto P che si muove nel piano $x\,y$ in un intervallo di tempo $a \le t \le b$. Le due coordinate di P saranno entrambe funzioni reali di t,

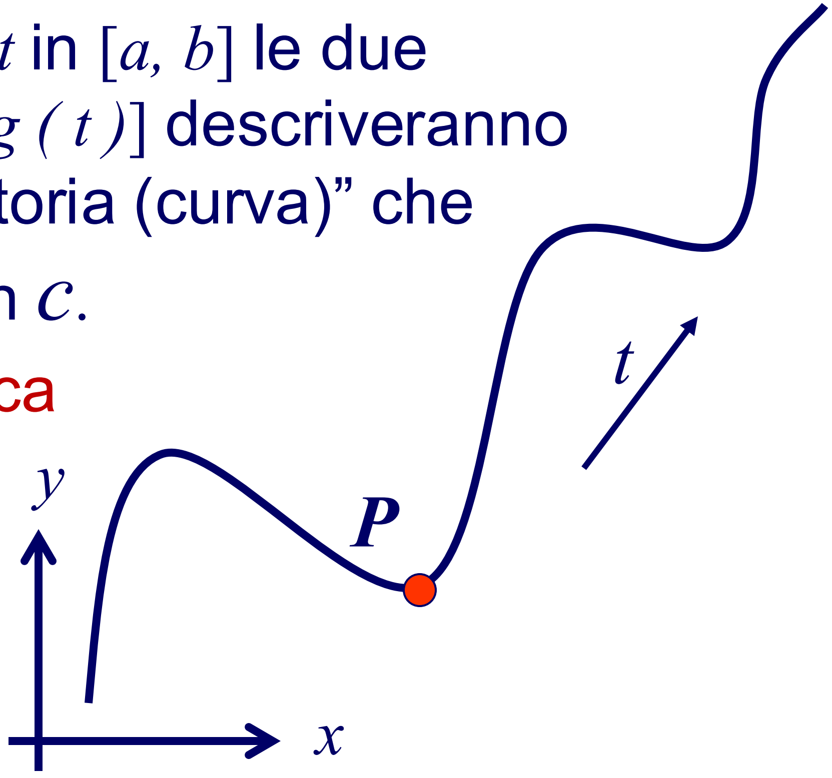
$$x = f\,(t)\;,\;\; y = g\,(t), \qquad t \in [a,b]\;.$$

Quindi al variare del tempo t in $[a, b]$ le due coordinate del punto $\boldsymbol{P} = [f\,(\,t\,)\,,\, g\,(\,t\,)]$ descriveranno nel piano Euclideo ”una traiettoria (curva)” che indicheremo con C.

Curva piana in forma parametrica

$$c(t) = [\,f(t),\;\; g(t)\,]$$

$$t \,\square\, [a,b]$$



Rappresentazione di Forme/Disegni 2D su un computer

Problema: rappresentare matematicamente forme 2D (disegni) su computer

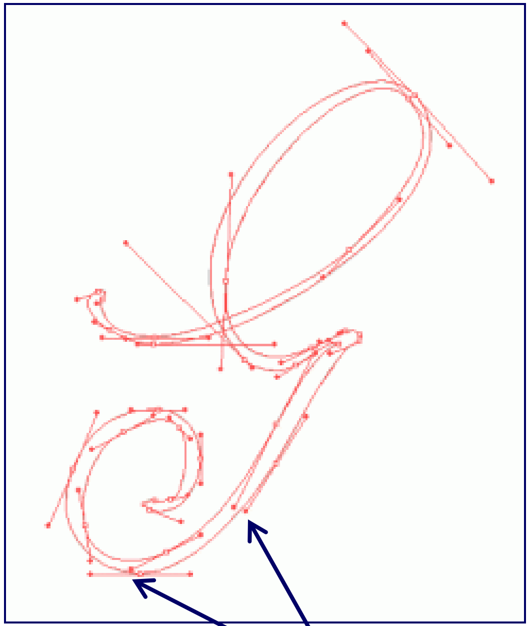
Soluzione: si usa una curva piana, in forma parametrica, polinomiale che ne definisce il contorno.

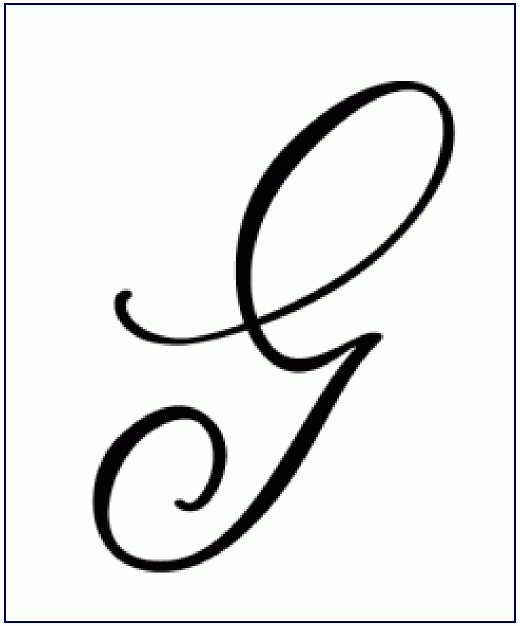
ma come?

1. Si specifichi una sequenza di punti $\boldsymbol{P}_i$, $i = 0, \ldots, n$, (detti *punti di controllo*) nel piano;

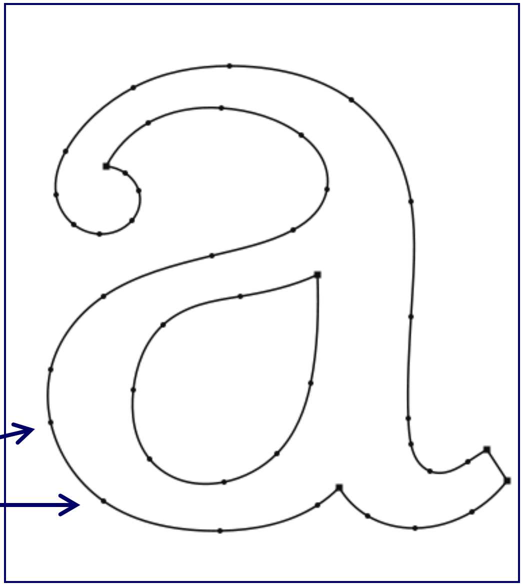
2. Si definisca una parametrizzazione cioè una funzione vettoriale (o curva piana, in forma parametrica, polinomiale, cioè $f(t)$ e $g(t)$ sono funzioni polinomiali), che abbia la forma dei punti $\boldsymbol{P}_i$, $i = 0, \ldots, n$ (*interpolazione/approssimazione* dei punti).

Esempi





Punti di Controllo



Interpolazione
dei Punti

Curva di Bézier

Una curva di Bézier di grado n, nel piano, è una curva, in forma parametrica, definita a partire da un insieme di punti $P_i = (x_i, y_i)$, $i=0,\ldots,n$ del piano detti punti di controllo e data da:

$$c(t) = \sum_{i=0}^{n} P_i B_{i,n}(t) \qquad t \in [0,1]$$

$$= \left[\sum_{i=0}^{n} x_i B_{i,n}(t) \; , \; \sum_{i=0}^{n} y_i B_{i,n}(t) \right]$$

dove ogni componente è una

funzione polinomiale nella base di Bernstein.

Si tratta di una curva in forma parametrica definita in termini del parametro t in [0,1] .

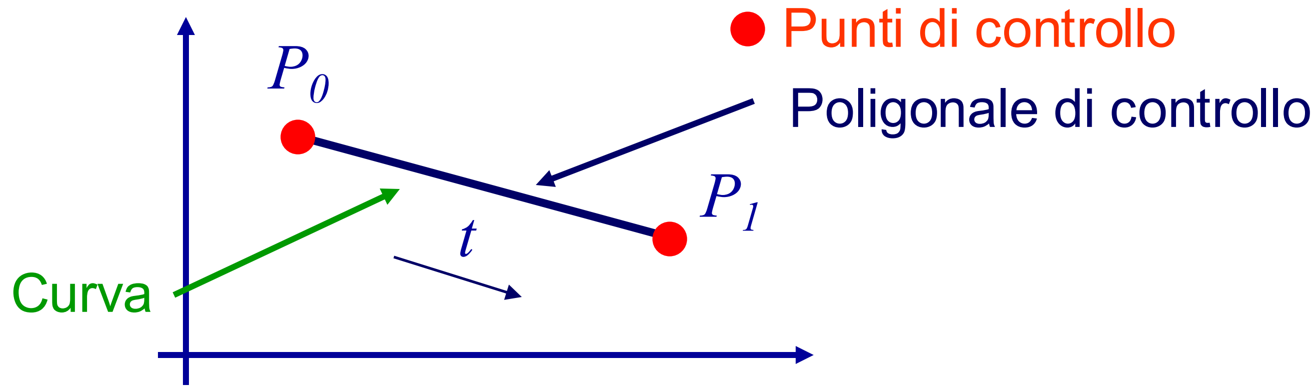
Al variare di t nell' intervallo [0,1], la $c(t)$ definisce i punti della curva.

Le Curve di Bézier

Esempi di curve di Bézier di grado n:

Caso $n=1$

$$c(t)=\sum_{i=0}^{1} P_i B_{i,1}(t) = P_0(1-t)+P_1 t \qquad t\in[0,1]$$



Se $P_0 = (x_0, y_0)$ e $P_1 = (x_1, y_1)$, in forma esplicita sarà:

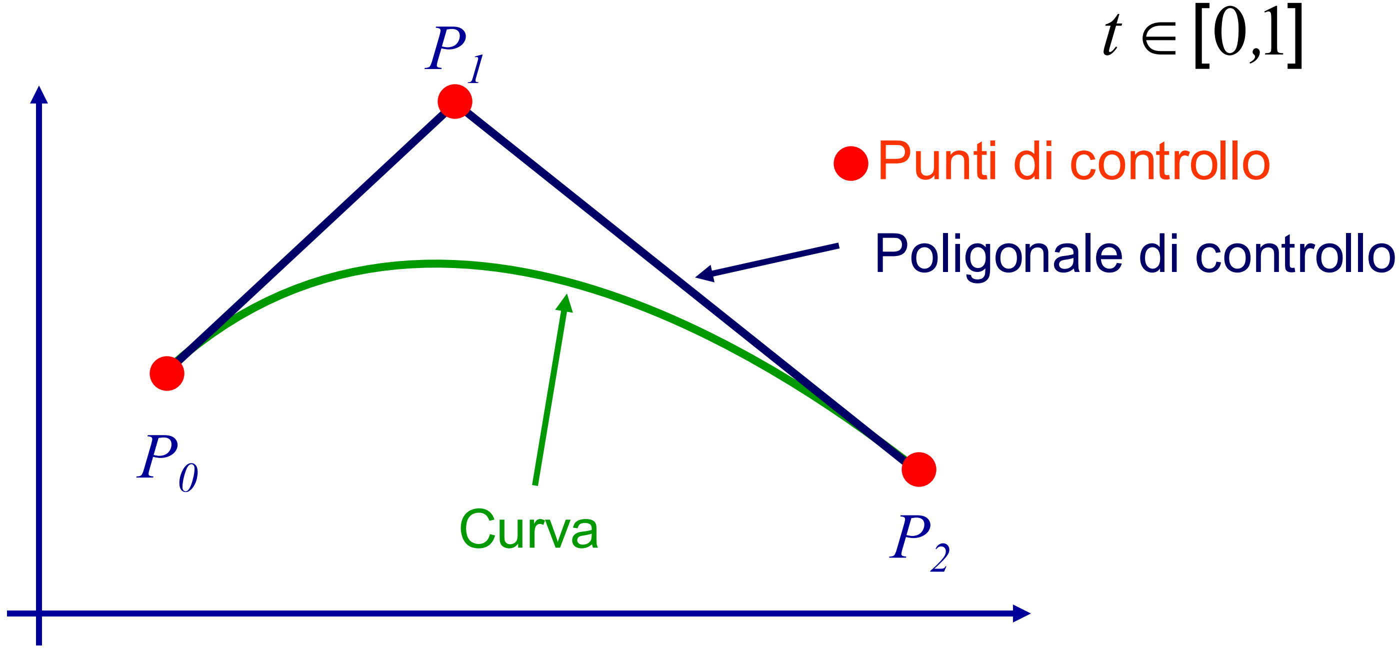
$$\begin{cases} x(t) = x_0(1-t) + x_1 t \\ y(t) = y_0(1-t) + y_1 t \qquad t \in [0, 1] \end{cases}$$

Le Curve di Bézier

Caso $n=2$

$$c(t) = \sum_{i=0}^{2} P_i B_{i,2}(t) = P_0(1-t)^2 + 2P_1(1\text{-}t)t + P_2 t^2$$

$$t \in [0,1]$$

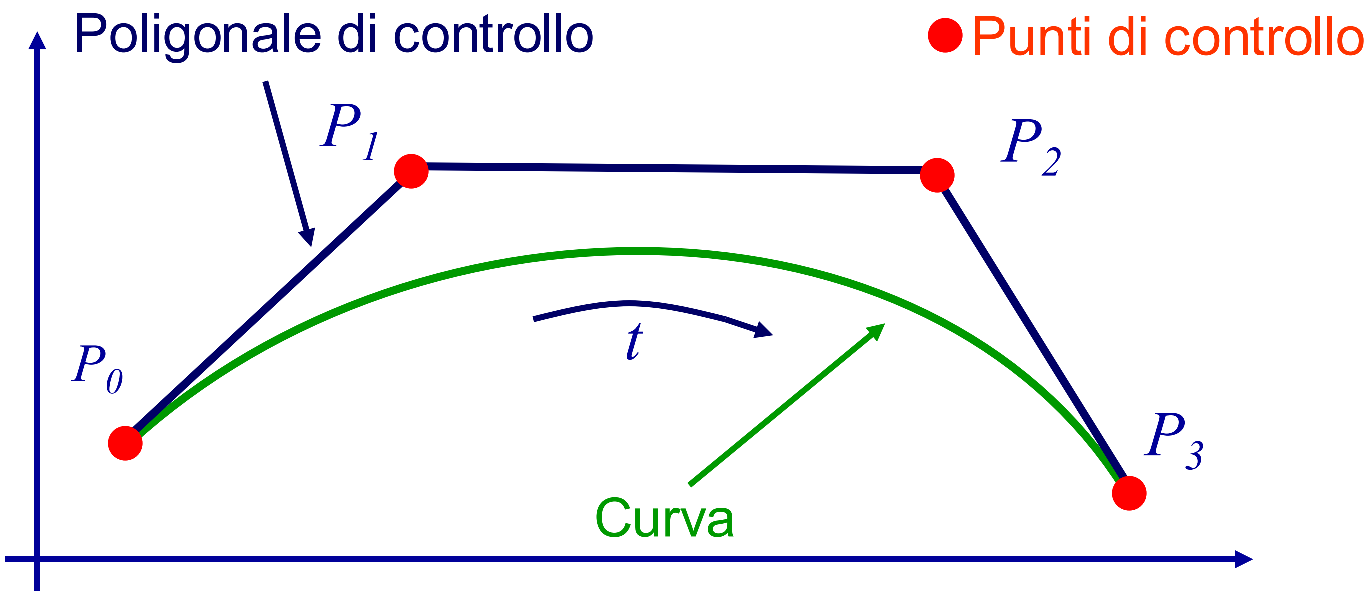


Le Curve di Bézier

Caso $n=3$

$$c(t)=\sum_{i=0}^{3} P_i B_{i,3}(t) = P_0(1-t)^3 + 3P_1(1\text{-}t)^2 t + 3P_2(1\text{-}t)t^2 + P_3 t^3$$

$$t \in [0,1]$$



Le Curve di Bézier e di de Casteljau

Il matematico francese de Casteljau, negli anni '60, diede una definizione di curva polinomiale 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 $k = 1,..,n$

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

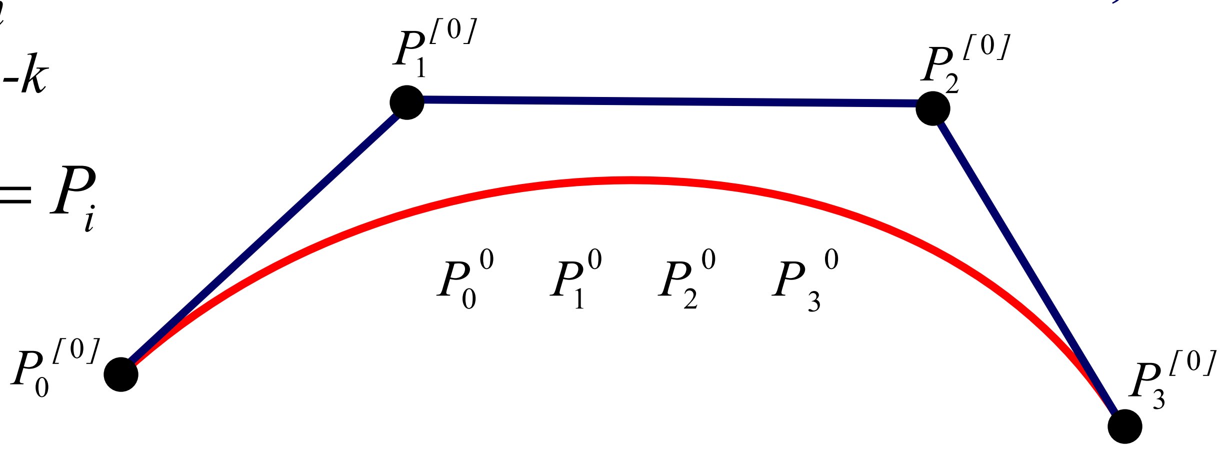
con $P_i^{[0]}(t) = P_i$

$i = 0,..,n$

e

$$c(t) := P_0^{[n]}(t)$$

Es. $n=3$, $k=0$



Le Curve di Bézier e di de Casteljau

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

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

Es. ***n=3***, ***k=1***

dove $\quad k = 1,..,n$

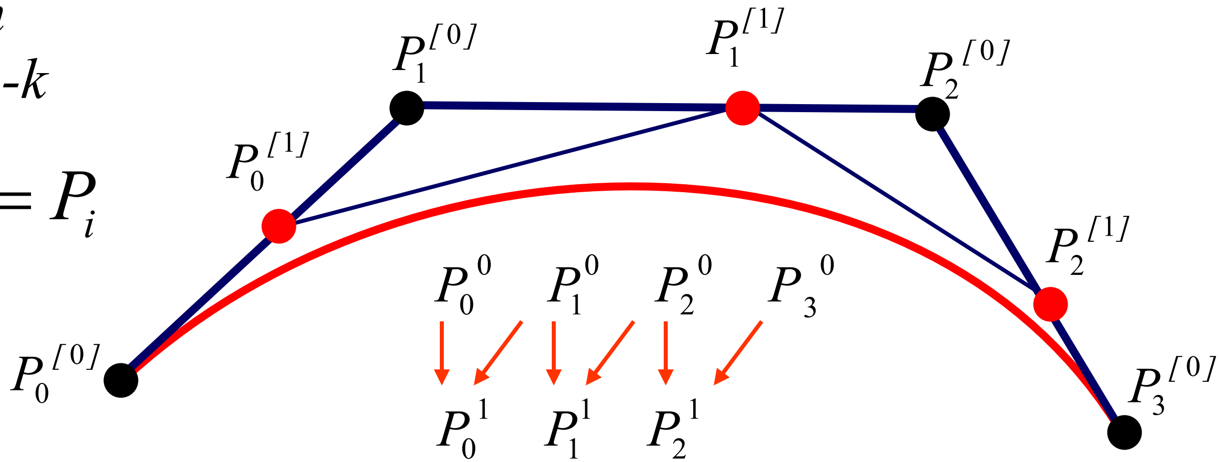
$\quad i = 0,..,n\text{-}k$

con $\quad P_i^{[0]}(t) = P_i$

$\quad i = 0,..,n$

e

$$c(t) := P_0^{[n]}(t)$$



Le Curve di Bézier e di de Casteljau

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

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

Es. ***n=3***, ***k=2***

dove $\quad k = 1,..,n$

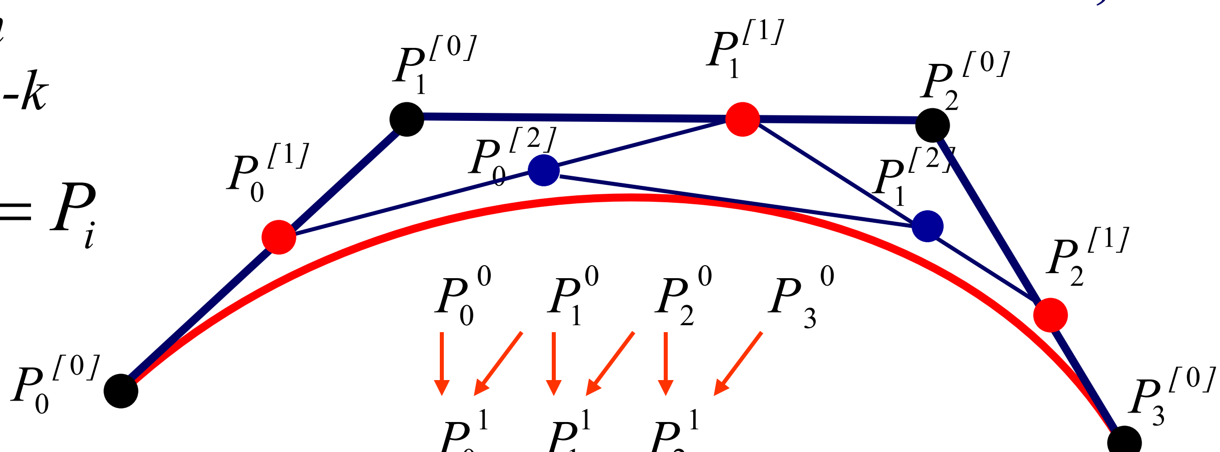
$\qquad\quad i = 0,..,n\text{-}k$

con $\quad P_i^{[0]}(t) = P_i$

$\qquad\quad i = 0,..,n$

e

$$c(t) := P_0^{[n]}(t)$$



$$\begin{array}{cccc} P_0^0 & P_1^0 & P_2^0 & P_3^0 \\ P_0^1 & P_1^1 & P_2^1 & \\ P_0^2 & P_1^2 & & \end{array}$$

Le Curve di Bézier e di de Casteljau

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

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

Es. $\boldsymbol{n=3}$, $\boldsymbol{k=3}$

dove $\quad k = 1,..,n$

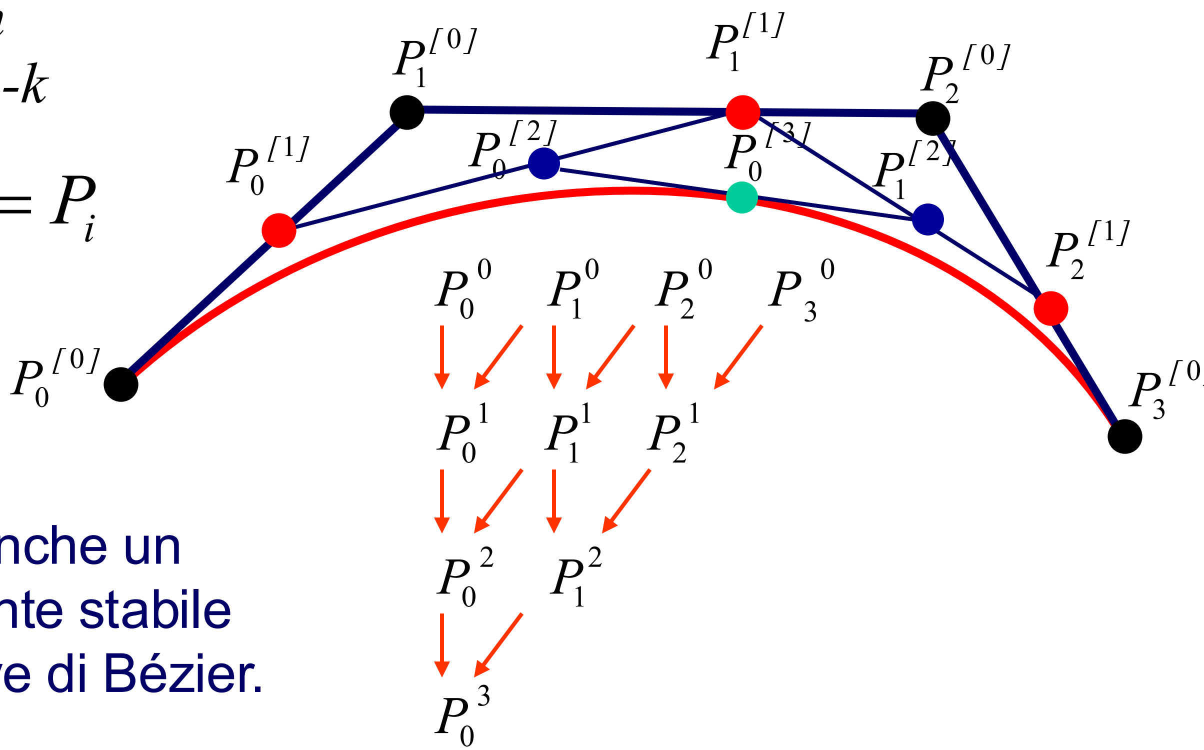
$\qquad\quad i = 0,..,n\text{-}k$

con $\quad P_i^{[0]}(t) = P_i$

$\qquad\quad i = 0,..,n$

e

$$c(t) := P_0^{[3]}(t)$$



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

Le Curve di Bézier e di de Casteljau

Proprietà:

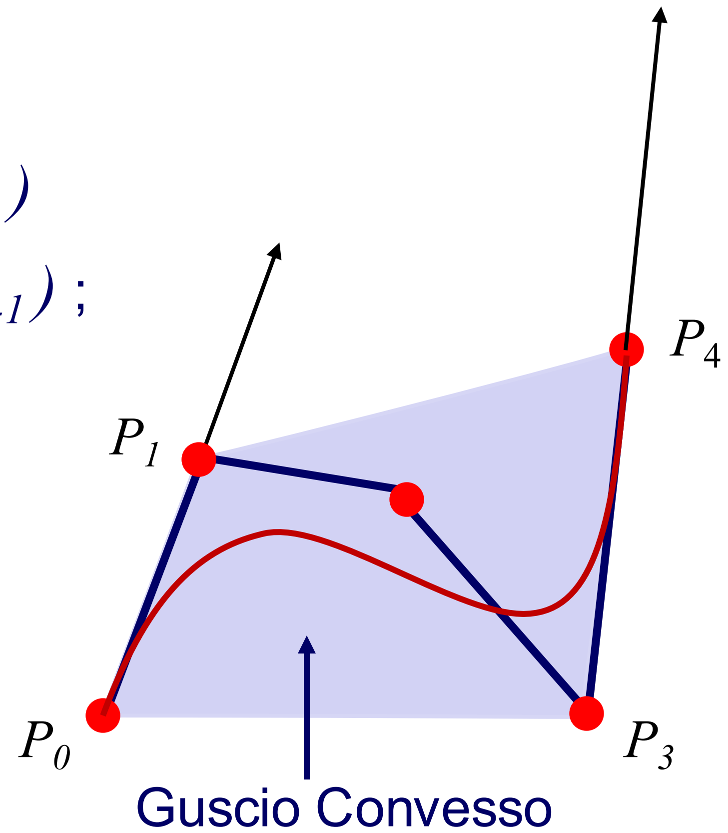
1. $c(t) \quad t \in [0,1]$ giace nel guscio convesso definito dai suoi punti di controllo;

2. $c(0) = P_0$

   $c(1) = P_n$;

3. $c'(0) = n(P_1 - P_0)$

   $c'(1) = n(P_n - P_{n-1})$;

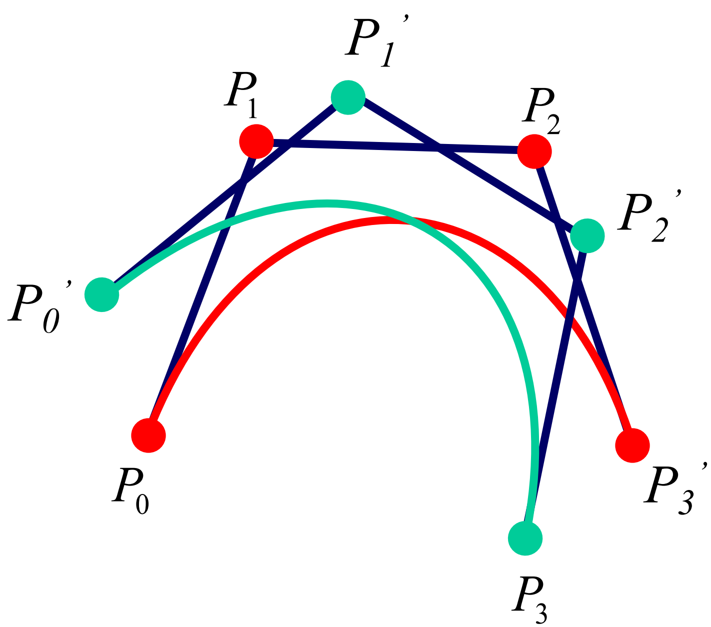


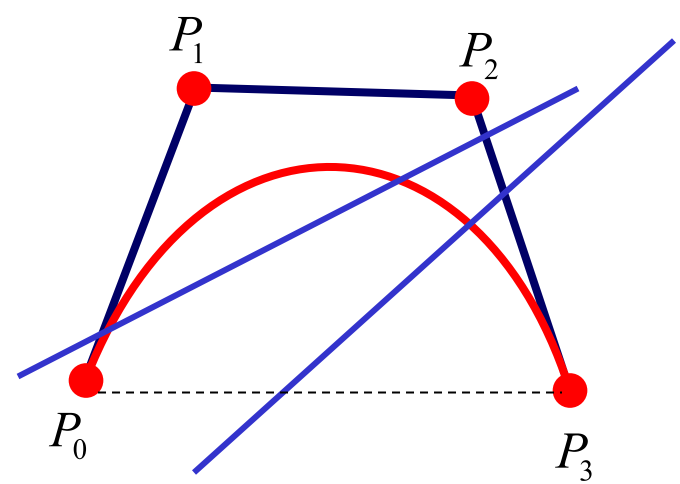
Le Curve di Bézier e di de Casteljau

Proprietà:

4. $c(t)$ è invariante per trasformazioni affini; in particolare per traslazione, scala, rotazione e deformazione lineare (shear);

5. $c(t)$ è approssimante in forma della poligonale di controllo.

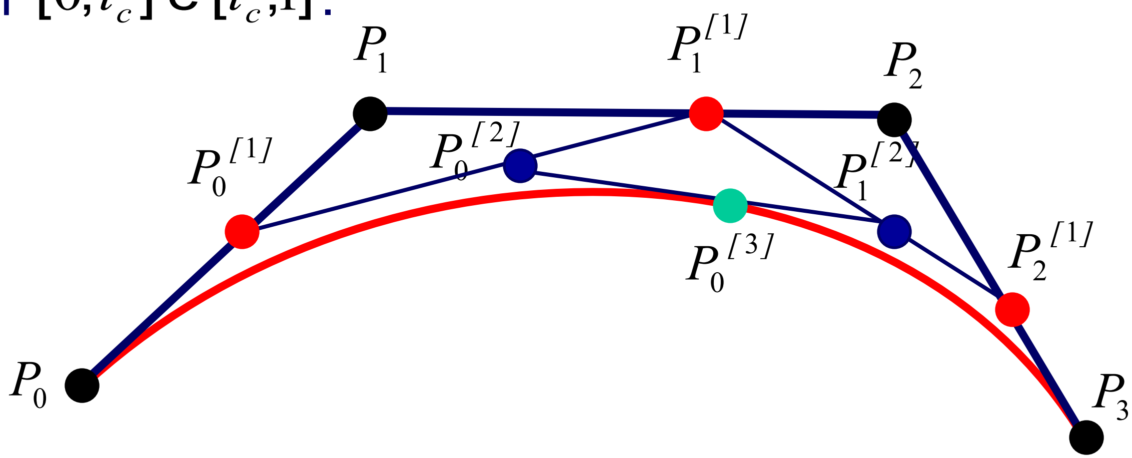




Le Curve di Bézier e la Suddivisione

La definizione o algoritmo di valutazione di de Casteljau di una curva di Bézier fornisce anche i punti di controllo delle curve di Bézier corrispondenti agli intervalli $[0, t_c]$ e $[t_c, 1]$.

Vediamolo nel caso $n=3$:



$$
\begin{array}{cccc}
P_0^0 & P_1^0 & P_2^0 & P_3^0 \\
P_0^1 & P_1^1 & P_2^1 & \\
P_0^2 & P_1^2 & & \\
P_0^3 & & &
\end{array}
$$

$(1\text{-}t) \downarrow \quad \swarrow \; t$

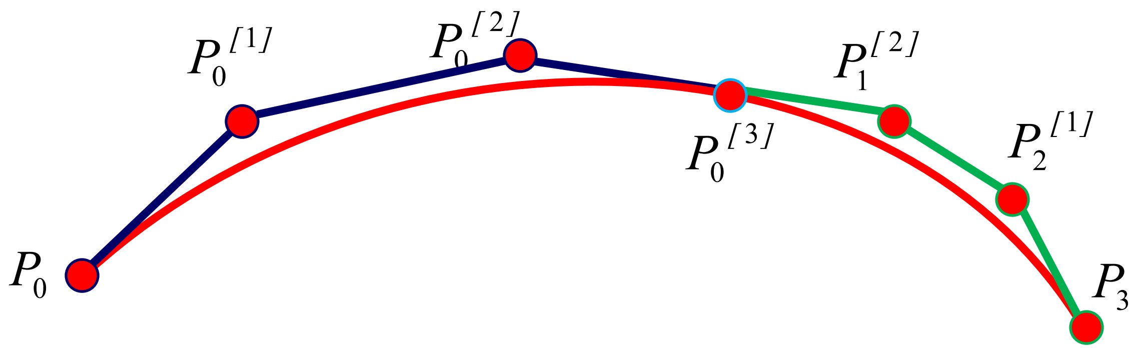
$$
c(t) = \sum_{i=0}^{3} P_i^0 B_{i,3}(t) = \begin{cases} \sum_{i=0}^{3} P_0^{\,i} B_{i,3}(t) & t \in [0, t_c] \\ \sum_{i=0}^{3} P_i^{3-i} B_{i,3}(t) & t \in [t_c, 1] \end{cases}
$$

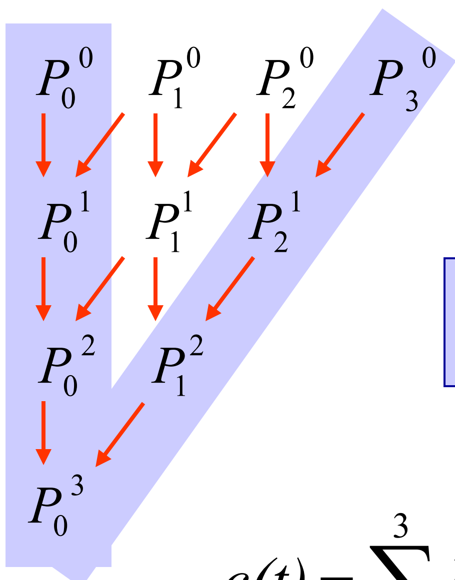
$t \in [0,1]$

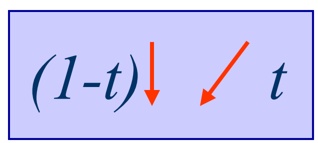
Le Curve di Bézier e la Suddivisione

La definizione o algoritmo di valutazione di de Casteljau di una curva di Bézier fornisce anche i punti di controllo delle curve di Bézier corrispondenti agli intervalli $[0,t_c]$ e $[t_c,1]$.

Vediamolo nel caso $n=3$:







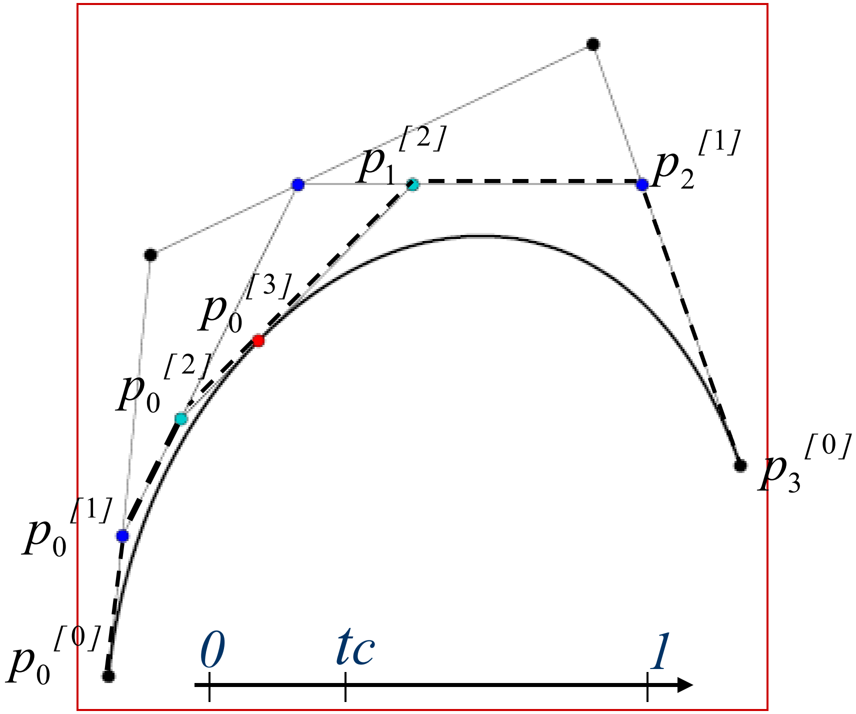
$$c(t) = \sum_{i=0}^{3} P_i^0 B_{i,3}(t) = \begin{cases} \displaystyle\sum_{i=0}^{3} P_0^i B_{i,3}(t) & t \in [0,t_c] \\ \displaystyle\sum_{i=0}^{3} P_i^{3-i} B_{i,3}(t) & t \in [t_c,1] \end{cases}$$

$t \in [0,1]$

Le Curve di Bézier e la Suddivisione: derivata prima e tangente

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

$t \in [0,1]$



La suddivisione permette di calcolare in modo semplice la derivata prima alla curva (o vettore tangente) in ogni punto con la regola vista di derivata agli estremi.

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

Attenzione: per suddivisione l'intervallo non è più unitario.

Curve Complesse

Una singola curva di Bézier può rappresentare solo una parte di una forma 2D complessa

Una soluzione potrebbe essere aumentare il grado

- questo aumenta le possibilità, ma al costo di più punti di controllo e polinomi di grado maggiore

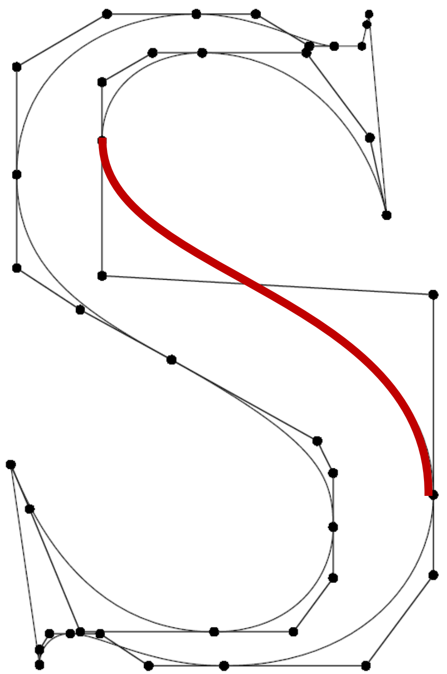
- il controllo è globale, cioè un punto di controllo influenza l’intera curva

Curve Complesse

In alternativa, la soluzione più comune è unire insieme più curve di Bézier di grado basso in una *piecewise curve*

(curva di Bézier a tratti)

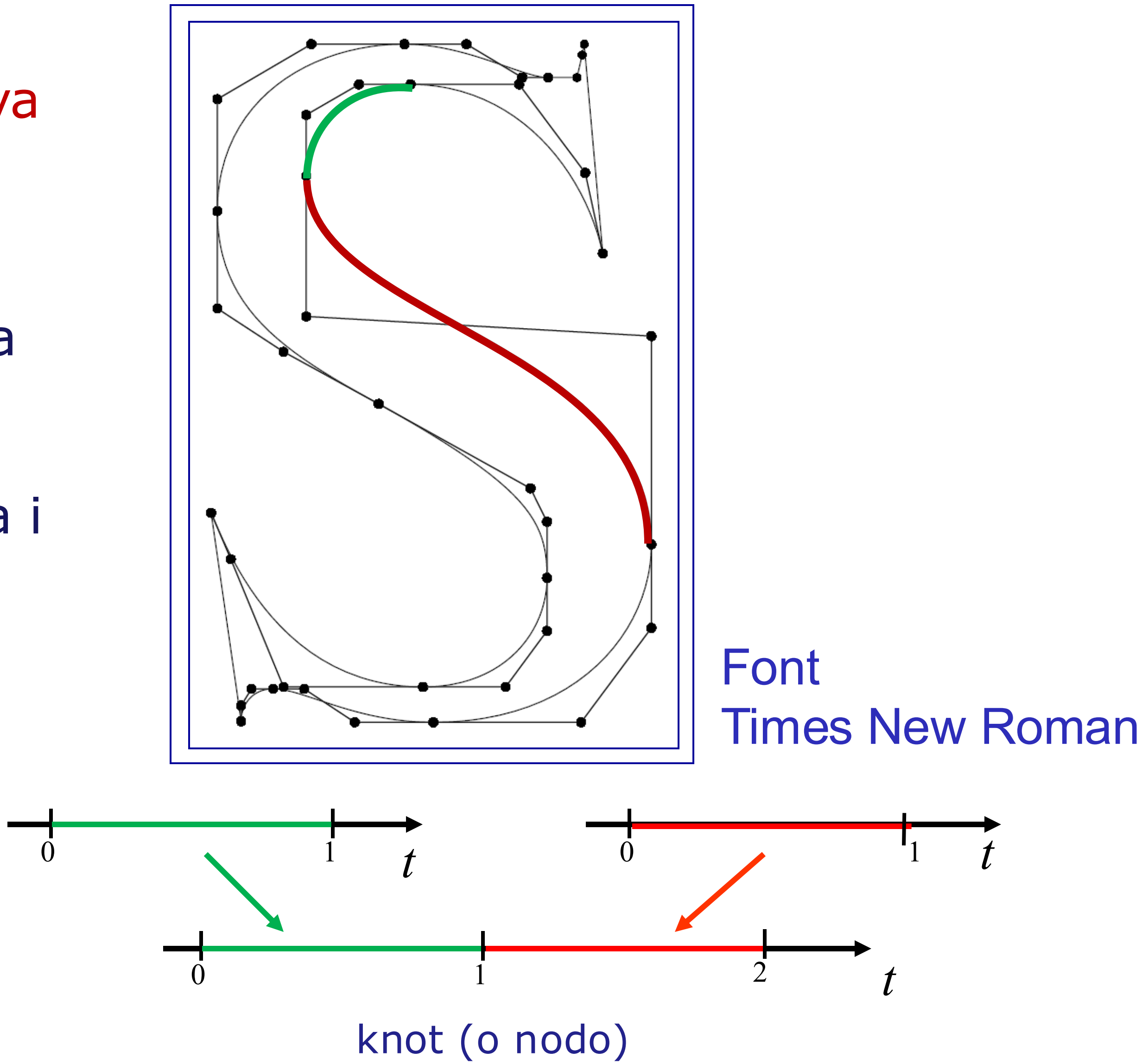
- una curva complessa in forma, può essere pensata in più tratti, ciascuno dei quali rappresentabile con una curva di Bézier di grado basso (per es. grado 3)
- *Controllo Locale*: ogni punto di controllo influenza solo una parte limitata della curva
- L'interazione e la modellazione sono più semplici



Curva di Bézier a tratti

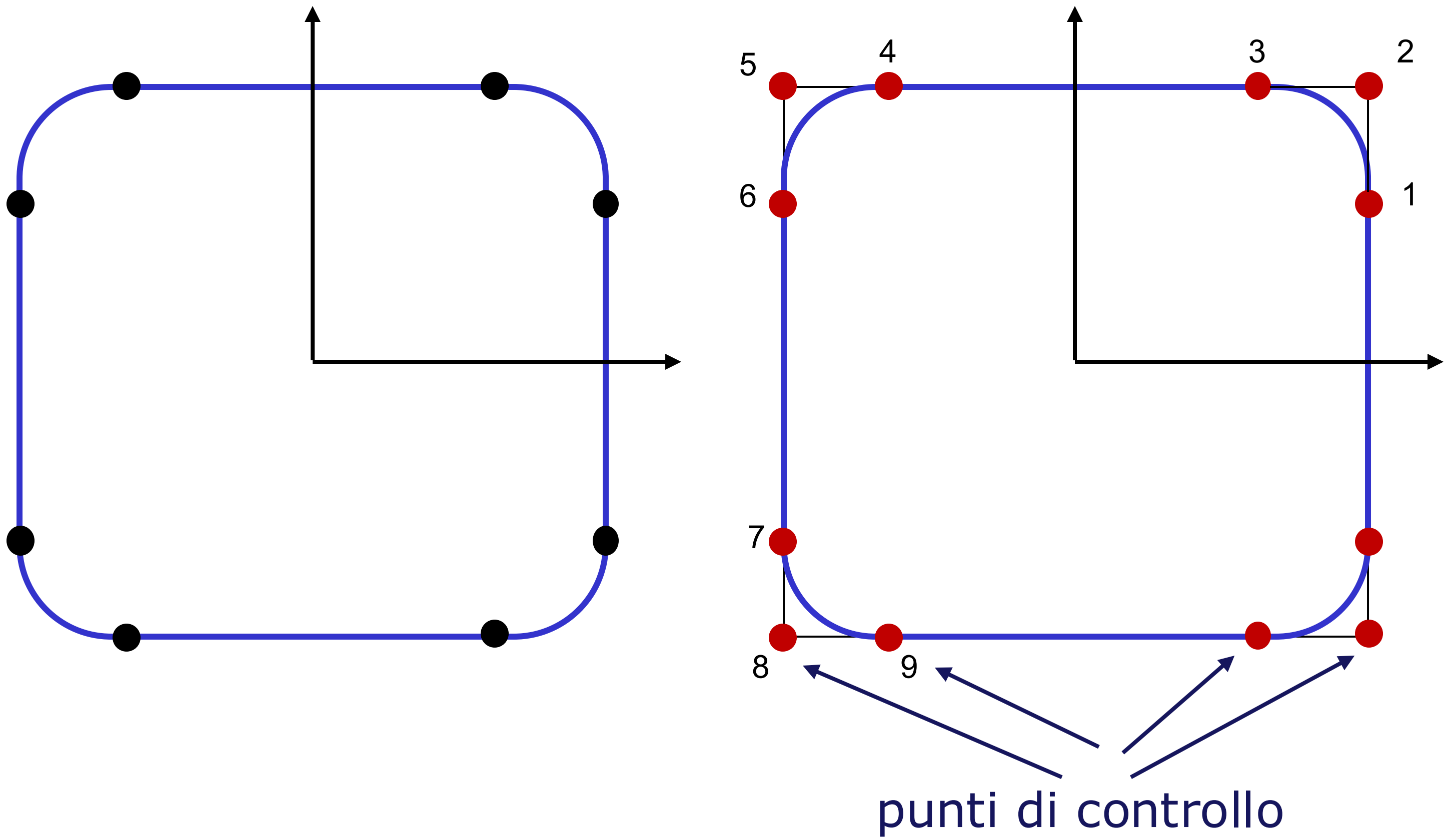
Ogni tratto è una curva di Bézier di stesso grado

La curva resta definita su un unico intervallo parametrico ottenuto mettendo in sequenza i singoli intervalli; i punti comuni o separatori dell'intervallo in sottointervalli sono detti knot (nodi).



Curve di Bézier a tratti multi-grado

Per il disegno di curve 2D lo standard de facto sono le curve di Bézier a tratti multi-grado; ogni tratto è una curva di Bézier di grado 1, 2 o 3.

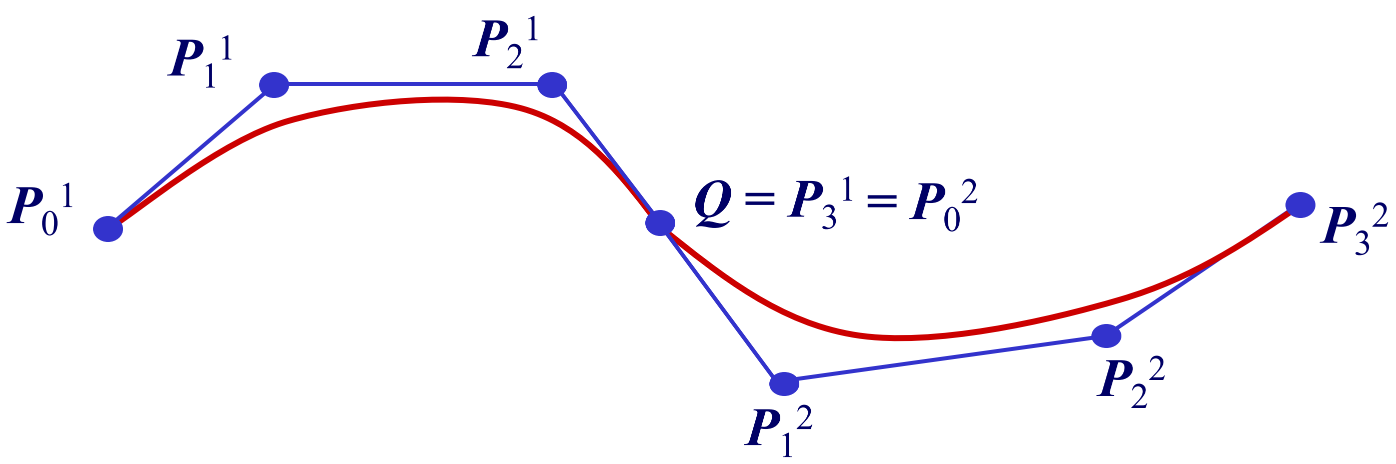


Continuità

- Quando due curve vengono unite, solitamente si vuole un ordine di continuità nei nodi (raccordo):
  - C^0, "C-zero", continuità posizionale, le curve condividono lo stesso punto dove si uniscono
  - C^1, "C-uno", continuità della derivata, le curve hanno lo stesso vettore tangente dove si uniscono
  - G^1, "G-uno", le curve hanno la stessa retta tangente e verso del vettore tangente, ma non il modulo
  - C^2, "C-due", continuità della derivata seconda, le curve hanno la stessa derivata seconda dove si uniscono
- Come facciamo ad assicurare che due curve di Bézier siano C^0, C^1, G^1, C^2, nei nodi?

Esempio: imporre la continuità

- Curve di Bézier cubiche:
  - Per definizione, le curve, interpolano i loro punti di controllo estremi, quindi si ha C^0 semplicemente uguagliando i punti di controllo estremi: $Q=P_3^1=P_0^2$
  - La continuità G^1 si ottiene ponendo $Q=P_3^1=P_0^2$ e facendo sì che P_2^1 , Q e P_1^2 siano allineati. Se vogliamo C^1 oltre ad essere allineati (G^1) devono essere anche equidistanti, cioé: $Q - P_2^1=P_1^2 - Q$



  - *La continuità* C^2 viene da ulteriori condizioni su P_2^2 e P_1^1

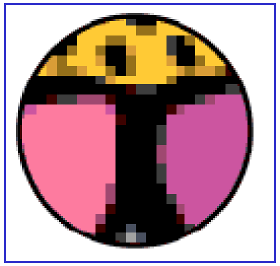
  Attenzione: ricordare di scalare le derivate se gli intervalli di definizione non sono di lunghezza unitaria.

Grafica 2D al Calcolatore

Ci sono due modi per definire un'immagine su un computer:

modalità Raster: cioè una matrice di valori interi di intensità, associata alla matrice dei pixel che costituiscono l'immagine;

modalità Vector: descrizione matematica delle curve che separano regioni di differente colore (outline);



Raster

Vector

Le immagini vettoriali sono scalabili, le immagini raster no.

Vector vs Raster

I principali vantaggi della grafica vector rispetto alla grafica raster sono la qualità ad ogni risoluzione, la maggior compressione dei dati e la più facile gestione delle eventuali modifiche.

Le immagini raster sono ideali per rappresentare foto (real life images), o per simulare il colore di materiali (textures);

Le immagini vettoriali sono migliori per tutti gli altri scopi.

Una dimostrazione pratica

PDF format, Adobe Systems 1993

Dimostrazione con Acrobat Reader

Vector vs Raster

La maggior parte dei dispositivi collegati ad un computer, come monitor, stampanti a matrici di punti e stampanti laser, sono dispositivi raster.

Ciò significa che tutti gli elementi prima di essere inviati a tali dispositivi (disegnati) devono essere trasformati in raster.

Il procedimento di conversione di un'immagine vector in una raster è detto rasterizing.

Con Immagini Digitali (cioè discrete) ci si riferisce sia a quelle raster che vector; l'unità elementare di un'immagine raster è il pixel (picture element).

Un esempio: le bitmap

Le bitmap sono immagini in bianco e nero tipicamente memorizzate e rappresentate in modalità raster.

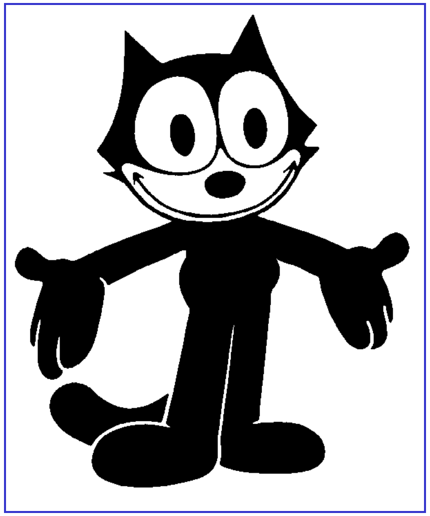
Per quanto detto sui vantaggi della grafica vector, spesso è utile convertire una bitmap in modalità vector.

Il procedimento di conversione di un'immagine raster in una vector è detto tracing (o vectorizing).

La Linea

Esempio: da bitmap a vector

Vettorizzazione con Curve di Bézier a tratti

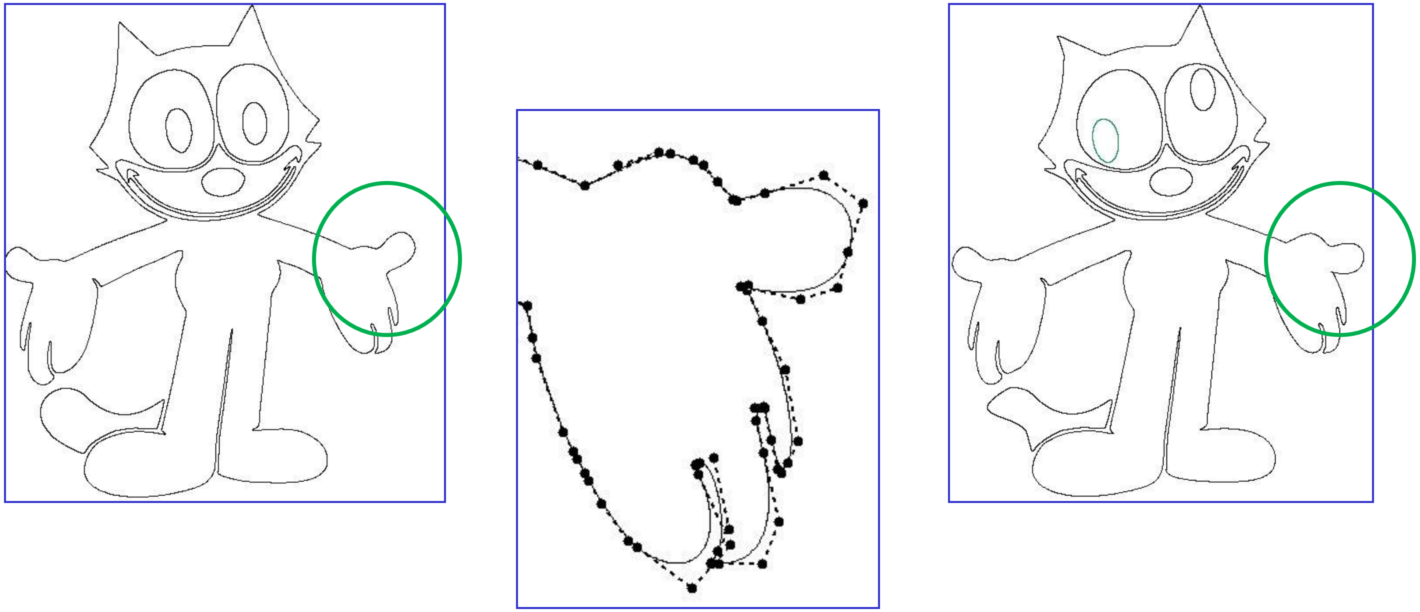


Bitmap: 672x777 pixel

Conversione vector

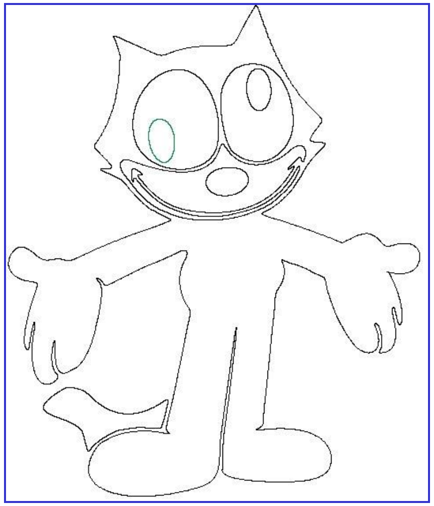
Modifica di Curve di Bézier a tratti

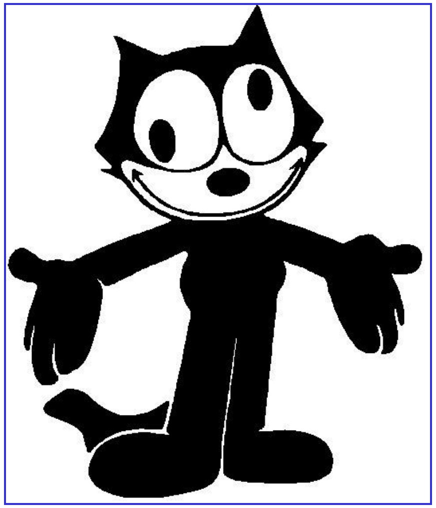
Modellazione mediante modifica dei Punti di Controllo



Esempio: da vector a bitmat:

Rasterizzazione di Curve di Bézier a tratti

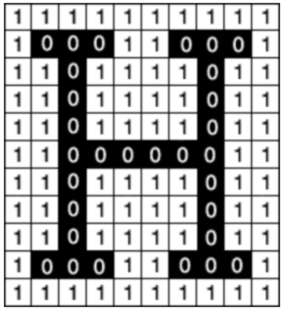




Bitmap: 672x777 pixel

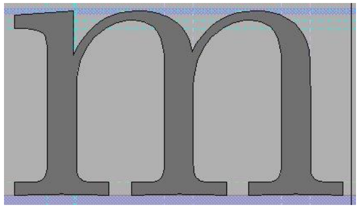
Un esempio: i font digitali

Un **font**, ovvero un tipo di carattere informatico, è una collezione indicizzata di glifi contenente informazioni su come associarvi un particolare codice, visualizzarli in differenti dimensioni e stamparli correttamente.



Ci sono due tipologie di font digitali:

Bitmap font: consiste in una matrice di pixel rappresentante l'immagine di un glifo;



Outline font: consiste in una descrizione delle curve che racchiudono lo spazio di un glifo;

Formati di font: type1 (Linux),

true type (Apple),

open type (Adobe e Microsoft).

Esempi di font

Cuningham Singleton à € di Perspectype Studio

Cuningham Singleton

Ramadhan Mubarak à € di Andrimada Creative

Ramadhan Mubarak

Richtive Script à € di Mans Greback

Richtive Script

Bellaty di SheillaType

Bellaty

Creepy Pumkin à € di Letter Art Studio

CREEPY PUMKIN

Esempi di font

Eagle di Woodcutter

Snake Mix di Woodcutter

Kitty Cats TFB di zanatlija

Lions di Woodcutter

Animal Tracks di Andrew D. Taylor

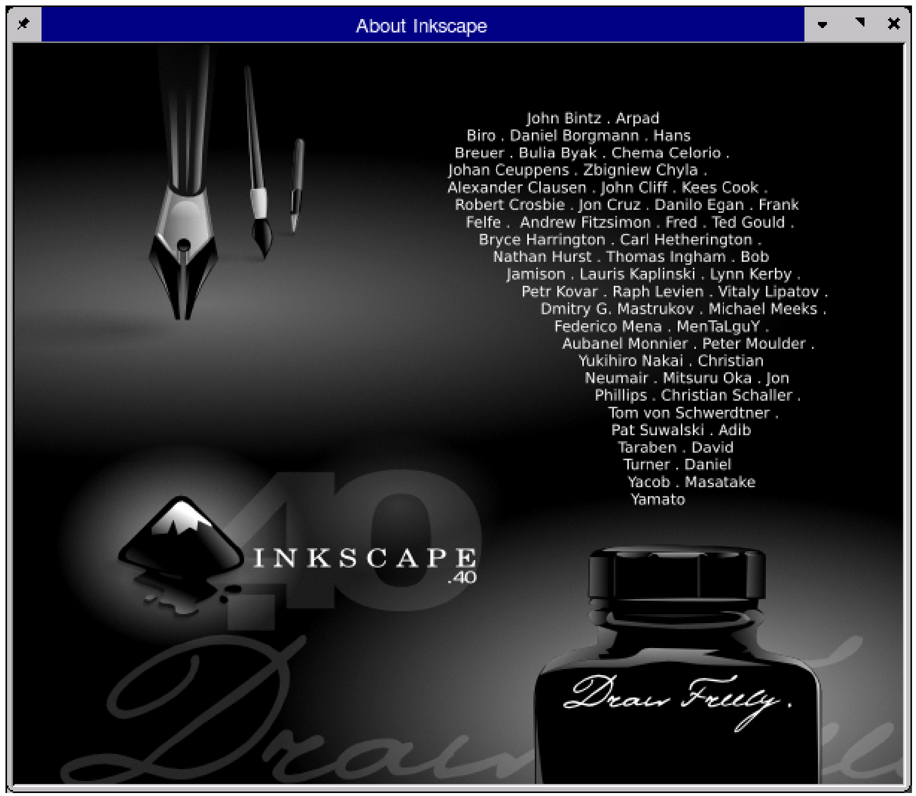
Deers di Woodcutter

Software e Formati comuni

LaTex e METAFONT
Postscript (ps, eps) GhostView
Adobe Acrobat Reader e Adobe Illustrator (pdf)
Adobe Flash (ex Macromedia Flash) grafica su web
PowerPoint, OpenOffice, LibreOffice
Inkscape, CorelDRAW
Gimp, Adobe Photoshop
Xfig (Linux)
Font Forge (Linux e Mac)
SVG (Scalable Vector Graphics)
Html5 (linguaggio di markup per il Web)
OpenGL, WebGL (librerie grafiche 3D)
Direct3D (libreria grafica 3D)
...

InkScape (Windows, Linux, Mac)

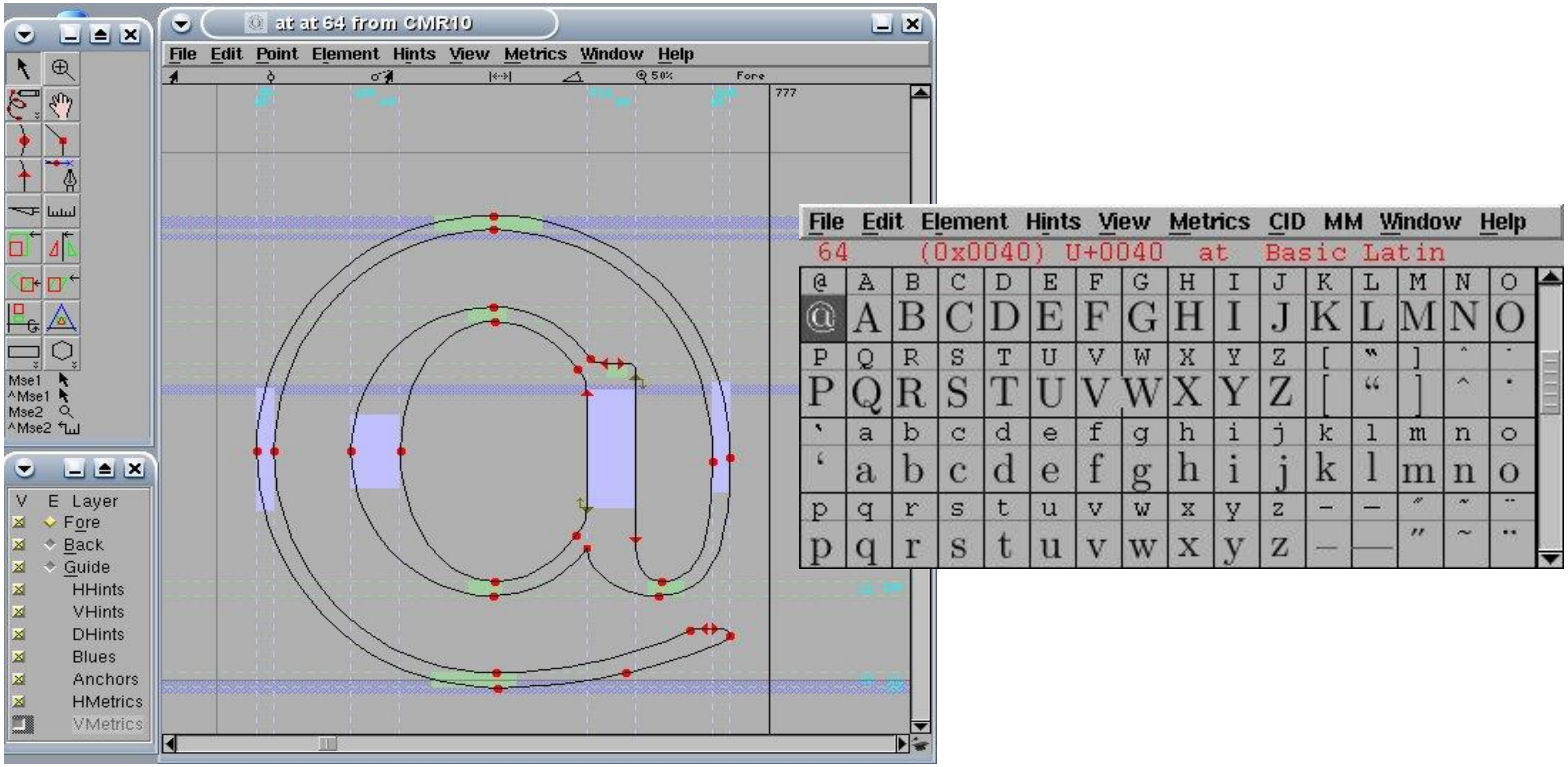
InkScape è un software libero OpenSource e licenza GPL per il disegno vettoriale basato sul formato SVG.



http://www.inkscape.org

Font Forge (Linux, Mac)

FontForge è un software libero OpenSource e licenza GPL che permette la creazione e modifica di font in molti formati standard; è un vero e proprio CAD 2D.



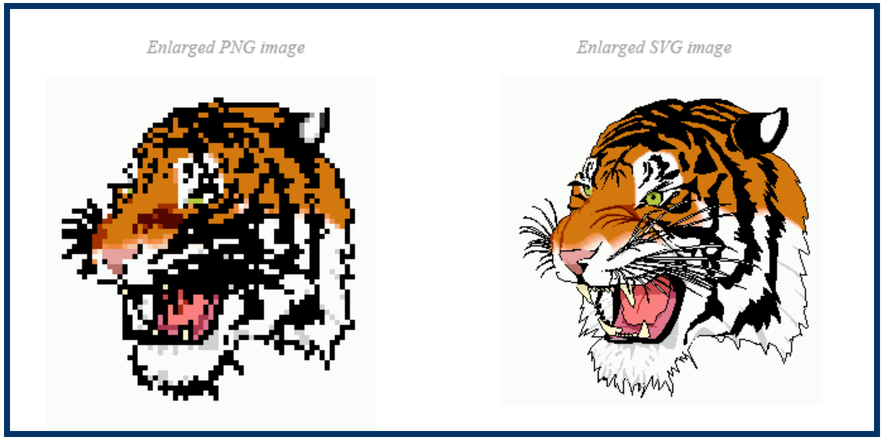
SVG (Scalable Vector Graphics)

SVG è un formato di grafica vettoriale, prodotto dal W3C (World Wide Web Consortium) consorzio noprofit che si occupa degli standard WEB.

Il sorgente di un file SVG è puro testo XML, ed è quindi componibile modularmente con qualsiasi altra applicazione XML.

Come linguaggio XML può essere processato con i tradizionali tool XML come parser, validatori, editor, e browser (SVG è supportato dalle attuali versioni di browser in modo nativo o mediante appositi plug-in).

SVG viene anche utilizzato per cellulari, smartphone, tablet, ecc.



SVG (Scalable Vector Graphics)

http://it.wikipedia.org/wiki/Scalable_Vector_Graphics