



Lezione 3

L'arte dei numeri (o meglio: i numeri dell'arte)

L'**arte** e la **matematica** sono due discipline molto più vicine di quanto potrebbe sembrare. Del resto tutti abbiamo avuto modo di osservare questo collegamento in numerose e famose opere d'arte!

La prima di tutte è l'*Uomo vitruviano*, di Leonardo da Vinci.

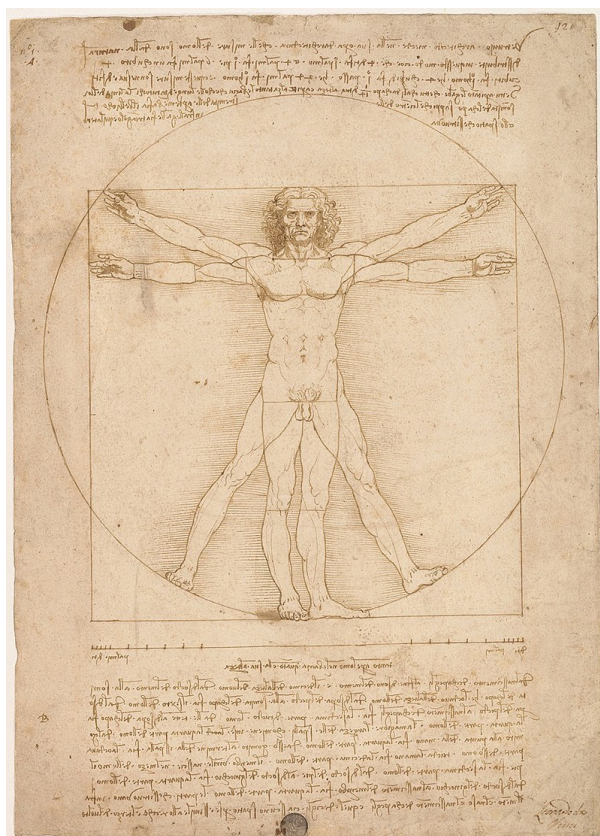


Figura 9: Uomo vitruviano, Leonardo da Vinci.
1490 circa, penna e inchiostro su carta.

Questo disegno è legato alla matematica non solo per la presenza di un cerchio e di un quadrato, ma in quanto l'uomo è disegnato secondo le *proporzioni ideali* del corpo umano, che corrispondono al **rapporto aureo** (il rapporto tra due grandezze a e b , con $a > b$, tali per cui vale $\frac{a+b}{a} = \frac{a}{b}$). Questa costante, così come il pi greco, è conosciuta fin dall'antichità e si può trovare in numerose realtà: a partire dalla successione di Fibonacci, fino all'arte e alla musica.

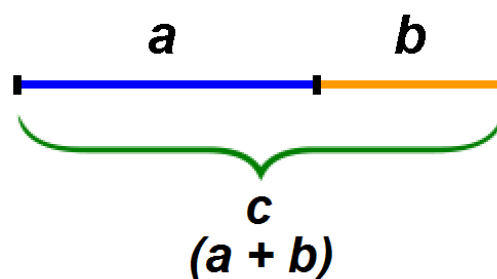


Figura 10: Rapporto aureo rappresentato su un segmento.

Credits: [Custeped](#)

Credits: [wikipedia](#)

Approfondimento - Leonardo da Vinci e la matematica:

<https://www.leonardo-da-vinci.net/it/matematica/>

Poi abbiamo i **frattali** (particolari oggetti geometrici che si ripetono su diverse scale di grandezza), la **prospettiva**, la **simmetria**...



Figura 11: Costruzione prospettica del quadro Città ideale (1480-1490), autore sconosciuto.

Credits: [anonimo](#)



Figura 12: Quadro di Jackson Pollock (Cathedral, 1947), che mostra caratteristiche simili ai frattali.

Credits: [Detlef Schobert](#), [CC BY-ND 2.0](#)



Figura 13: La spirale aurea del Nautilus

Credits: [OneEagle](#)

Approfondimento:

[https://www.youtube.com/watch?](https://www.youtube.com/watch?v=BMJ_B9_ab74)

[v=BMJ_B9_ab74](https://www.youtube.com/watch?v=BMJ_B9_ab74)

Approfondimento:

http://archivio.torinoscienza.it/recensioni/i_frattali_di_pollock_20398.html

Proprio parlando di frattali ci avviciniamo al prossimo interessante argomento del calcolo numerico. Infatti, i frattali sono stati importanti per lo sviluppo del **computer graphics**! Grazie ai frattali è possibile generare realistici paesaggi, e modellare l'aspetto delle venature del marmo:

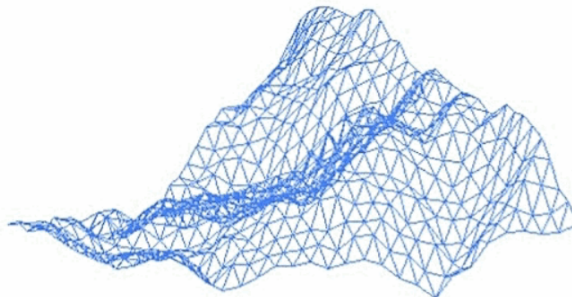



Figura 14: Creazione di una montagna tramite l'utilizzo di triangoli frattali.

Credits: [António Miguel de Campos](#)

La modellazione e la rappresentazione di curve e superfici sono il collegamento tra **computer grafica** e **calcolo numerico**!

Ma prima torniamo a carta e penna 

Esercizio 07

Prendete un foglio a quadretti e disegnate un quadrato di 10 x 10 cm.

Riuscite a disegnare una curva utilizzando solamente delle linee rette?

Esercizio 07 +

Se siete riusciti in questa impresa grafica, provate a scrivere l'algoritmo per replicare la costruzione della curva.

Esiste un semplice modo per risolvere l'esercizio, e con pochi passi, un foglio, una matita ed un righello sarete in grado di disegnare una curva (che ovviamente sarà una *approssimazione* di una curva) utilizzando solamente linee rette:

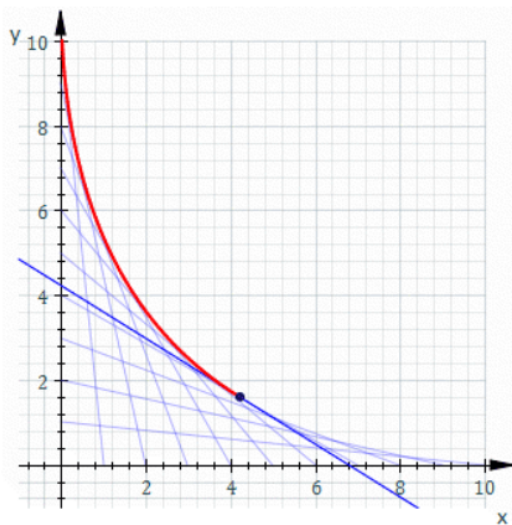


Figura 15: Animazione della creazione di un involuppo.

Credits: [Sam Derbyshire](#)

- prendete due lati del quadrato (ad esempio il sinistra e quello in basso), poi segnate e numerate i centimetri partendo dall'angolo in comune (come se fosse un piano cartesiano);
- partendo dalla posizione (1, 0) tracciate una retta che arriva in posizione (0, 9);
- partendo dalla posizione (2, 0) tracciate una retta che arriva in posizione (0, 8);
- ... continuate così fino ad arrivare all'ultima retta, che dal punto (9, 0) arriva in (0,1).



Avete creato un **involuppo**!

Cos'è un inviluppo?

Un inviluppo è una **curva piana** creata a partire da una collezione (*famiglia*) di curve. Più precisamente, un **inviluppo di una data famiglia di curve** è una **curva tangente** ad ogni elemento della famiglia stessa.

La curva che abbiamo creato (o meglio, *approssimato*) con carta, matita e righello è un inviluppo di una **famiglia di rette**, o, più precisamente, una **curva di Bézier**!



Una **curva di Bézier** è una **curva planare parametrica**, che può essere costruita a partire da un insieme di punti detti **punti di controllo**, tramite una **interpolazione lineare**. Collegati i punti di controllo in modo ordinato da linee rette, si ottiene il **poligono di controllo**, che approssima la forma della curva di Bézier.

▼ Chi è Bézier?

Approfondimento: La storia delle curve di Bézier

Le curve di Bézier prendono il nome dal loro inventore: Pierre Bézier (1910 - 1999), ingegnere e matematico francese. Negli anni Settanta, lavorando alla Renault, si pose il problema di come poter rappresentare arbitrarie curve al computer, per poter fruttare applicazioni CAD per la produzione della carrozzeria delle automobili.

Qualche anno prima, in un'altra azienda, la Citroën, il fisico e matematico francese Paul de Casteljau si era posto lo stesso problema.

In modo indipendente e a qualche anno di distanza, i due matematici trovarono il modo per definire questo particolare insieme di curve, che ora sono famose con il nome di Pierre Bézier, il primo ad averle pubblicizzate. L'algoritmo che ne descrive i calcoli di costruzione, invece, prende il nome di Paul de Casteljau, il primo ad aver sviluppato un *metodo numerico* per calcolare le curve, e ad averlo applicato per la modellazione di parti di automobili.

Guardiamo meglio la definizione di curva di Bézier. Possiamo notare due importanti caratteristiche:

- Le curve di Bézier sono **curve parametriche**, ovvero dipendono da un parametro (che chiameremo t);
- Possiamo calcolare e costruire queste curve utilizzando l'**interpolazione lineare**, che altro non è che un metodo del calcolo numerico!

Interpolazione lineare

Interpolare significa ottenere un nuovo punto sul piano cartesiano a partire da un insieme di altri punti, ipotizzando che tutti appartengano ad una particolare funzione.

L'**interpolazione lineare** approssima un valore compreso tra due punti, che si considera appartenenti ad una funzione lineare. Una semplice forma di interpolazione lineare, che è anche l'ingrediente necessario alla valutazione delle curve di Bézier, è la **media ponderata**.

Dati due punti A e B , un punto tra di essi può essere rappresentato in base al valore che assume il parametro t .

Questa interpolazione lineare è un metodo che tutti abbiamo usato, ed usiamo nella vita di tutti i giorni.

Ad esempio: arrivati alla stazione (A), partiamo in treno verso la meta (B) della nostra prossima vacanza, quando,

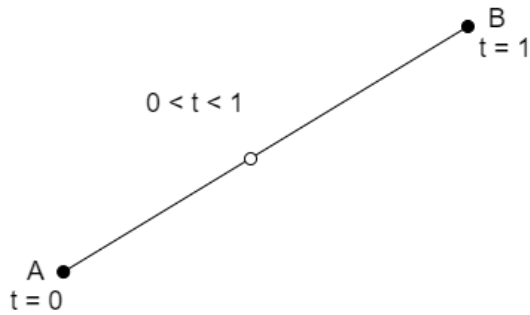


Figura 16: Segmento parametrico.

impazienti di arrivare, chiediamo al nostro compagno di viaggio quanto manca all'arrivo. "Siamo a due terzi del viaggio!", ci risponde.

Ecco che il nostro amico ha applicato una interpolazione lineare tra la stazione di partenza del nostro viaggio, e quella di arrivo. Ci ha infatti detto che il *punto* in cui ci troviamo dista da *A* di un valore $t = 2/3$ rispetto al totale, e che quindi dista da *B* di un valore $t = 1/3$.

Esercizio 08

Rileggendo bene la definizione e l'esempio di interpolazione lineare, guardate come abbiamo fatto a disegnare un involucro, e individuate in che modo viene applicata a questo caso. Cercate di formalizzarlo.

Per ora abbiamo visto esempi di curve di Bézier costruite a partire da tre punti di controllo, che viene chiamata **curva di Bézier quadratica** (oppure curva di Bézier di secondo grado).

▼ Qual'è la curva di Bézier costruita su due soli punti?

È semplicemente il segmento formato dai due punti di controllo.

E utilizzando tre punti di controllo? Otteniamo una **curva di Bézier cubica**. Ecco un esempio:

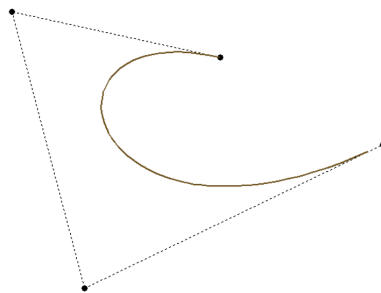


Figura 17: Curva di Bézier cubica (dove la poligonale di controllo è tratteggiata).

Ma possiamo **generalizzare** e aggiungere un numero arbitrario di punti!

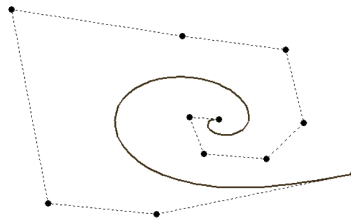


Figura 18: Curva di Bézier con undici punti di controllo (dove la poligonale di controllo è tratteggiata).



Ogni curva di Bézier con n punti di controllo e $n - 1$ segmenti nella poligonale di controllo ha grado $= n - 1$.

Costruiamo una curva di Bézier: **algoritmo di de Casteljau**

Abbiamo visto che le curve di Bézier quadratiche vengono costruite tramite tre applicazioni dell'interpolazione lineare: si interpolano gli estremi dei due segmenti della poligonale di controllo ottenendo due nuovi punti, che vengono usati per l'ultima interpolazione che dà come risultato un punto della curva di Bézier.

La formula dell'**interpolazione lineare** (*Lerp*) tra due punti A e B , e parametro t è:

$$Lerp(t, a, b) = (1-t) \cdot a + t \cdot b$$



Figura 19: Costruzione di una curva di Bézier quadratica.

Credits: Phil Tregonin (1),(2).

Questo stesso procedimento viene applicato anche con curve di (qualsiasi) grado maggiore. Consideriamo il caso di grado 3.

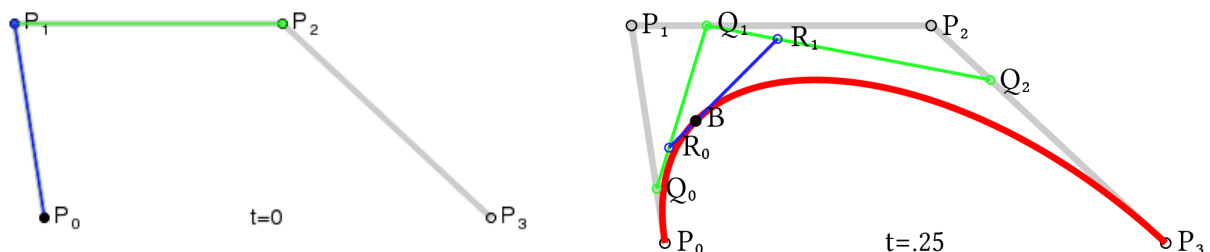


Figura 20: Costruzione di una curva di Bézier cubica.

Credits: Phil Tregonin₍₁₎, ₍₂₎.

La costruzione della curva cubica, illustrata anche dalla Figura 20, è la seguente:

- $Q_0 = \text{Lerp}(t, P_0, P_1)$
- $Q_1 = \text{Lerp}(t, P_1, P_2)$
- $Q_2 = \text{Lerp}(t, P_2, P_3)$
 - $R_0 = \text{Lerp}(t, Q_0, Q_1)$
 - $R_1 = \text{Lerp}(t, Q_1, Q_2)$
 - $B = \text{Lerp}(t, R_0, R_1)$

Il punto B è un punto della curva di Bézier! Basta ripetere queste sei interpolazioni lineari per un insieme scelto di valori di $t \in [0, 1]$ e si costruisce la curva di Bézier con l'**algoritmo di de Casteljau**!

Generalizzando ancora di più, la descrizione del procedimento per un **numero arbitrario di punti di controllo** è questo:

Dati n punti di controllo, per ogni segmento della poligonale di controllo ($n - 1$ segmenti) si interpolano gli estremi e si ottengono $n - 2$ nuovi punti. Questi si collegano in modo ordinato da rette ($n - 3$ segmenti), i cui estremi vengono nuovamente interpolati. Si **ripete** il procedimento fino a quando non si rimane con **due punti ed un solo segmento**. Variando il valore di t , l'interpolazione sull'ultimo segmento rimasto corrisponde alla curva di Bézier.

Approfondimento: per una spiegazione (anche visiva) delle curve di Bézier guardate questo video fino al minuto 4:21 <https://www.youtube.com/watch?v=aVwxzDHniEw>

Esercizio 09

Prendete carta, penna e righello e provate a disegnare una curva di Bézier cubica!

Utilizzate quattro vertici e tre lati di un quadrato come punti e poligonale di controllo e applicate l'algoritmo di de Casteljau.

▼ Serve un suggerimento?

Guardate l'animazione della costruzione della curva cubica presentata sopra e osservate bene come si muovono i punti. Quante curve riuscite a tracciare dal loro movimento?

L'**algoritmo di de Casteljau** permette di **valutare la curva di Bézier** in un insieme di **punti dipendenti dal parametro t** . Se dobbiamo rappresentare graficamente la curva per visualizzarla sullo schermo di un computer, ogni coppia ordinata di punti può essere collegata da una linea retta.

! Quindi, maggiori sono i valori di t per cui viene valutata la curva di Bézier, più accurata sarà l'approssimazione della stessa (in modo simile a quanto accade con il numero degli intervalli nel metodo dei rettangoli).

Alcune osservazioni prima di passare all'implementazione:

- Osservando la ripetizione delle interpolazioni dell'algoritmo di de Casteljau, si evidenzia uno **schema triangolare**, come nella sottostante. La base dello schema a piramide sono i punti di controllo P_0, P_1, \dots che servono a calcolare i punti successivi: Q_0 dipende da P_0 e P_1 del livello precedente, e lo stesso vale per gli altri punti. Dopo aver calcolato il punto Q_0 , il punto P_0 non serve più!

Procedendo con i livelli, i punti necessari diminuiscono, fino ad arrivare alla fine dello schema, dove troviamo l'unico valore di interesse alla fine dell'esecuzione dell'algoritmo di de Casteljau: il punto B , che è un punto della curva.

Lo schema è generalizzabile per curve di Bézier di qualsiasi grado, e ci aiuta a capire come poter implementare l'algoritmo!

$$\begin{array}{lll}
 P_0 & Q_0 = \text{Lerp}(t, P_0, P_1) & R_0 = \text{Lerp}(t, Q_0, Q_1) \quad B = \text{Lerp}(t, R_0, R_1) \\
 P_1: & Q_1 = \text{Lerp}(t, P_1, P_2) & R_1 = \text{Lerp}(t, Q_1, Q_2) \\
 P_2 & Q_2 = \text{Lerp}(t, P_2, P_3) & \\
 P_3 & &
 \end{array}$$

Figura 21: Schema triangolare delle interpolazioni ripetute dell'algoritmo di de Casteljau, nel caso di una curva di Bézier cubica.

Esercizio 10

In questo esercizio dovrai implementare l'algoritmo di de Casteljau per la costruzione di curve di Bézier di ordine qualsiasi.

Il file `BezierCurves.py` contiene il codice di una applicazione grafica che permette di disegnare e visualizzare le curve di Bézier. Il comportamento è il seguente.

- L'applicazione presenta due modalità: la modalità "modellazione" (quella iniziale) permette di creare e modificare la curva di Bézier; la modalità "animazione" mostra come la curva viene costruita (in questa modalità non è possibile la modifica);

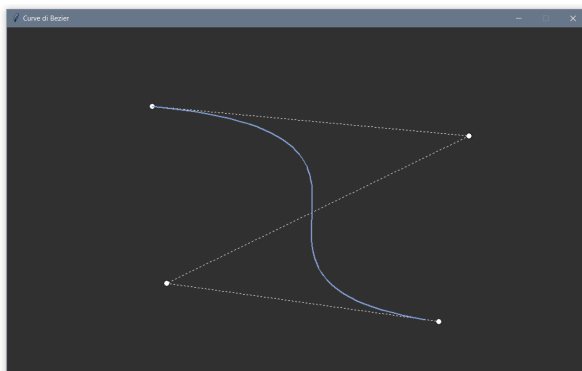


Figura 22: Screenshot dell'applicazione in modalità "modellazione".

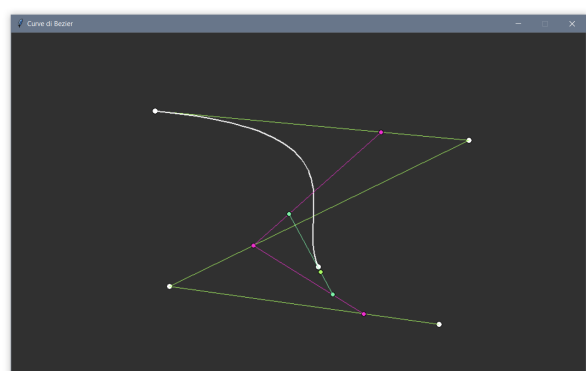


Figura 23: Screenshot dell'applicazione in modalità "animazione".

- Con un singolo click del tasto sinistro sulla finestra si aggiungono nuovi punti di controllo;
- Con un singolo click del tasto destro sopra ad un punto esistente si rimuove lo stesso;
- Cliccando e trascinando con il tasto sinistro un punto esistente è possibile modificarne la posizione;
- Il tasto `c` pulisce la finestra cancellando tutti i punti di controllo;
- Il tasto `a` permette di passare dalla modalità “modellazione” alla modalità “animazione”. Il passaggio non interferisce con la curva attuale.
- Il tasto `q` fa terminare l’esecuzione dell’applicazione.

Nota: l’applicazione è realizzata tramite `tkinter`, libreria grafica di Python, che dovrete avere tutti, in quanto facente parte della libreria standard di Python.

Ma l’applicazione non disegna ancora le curve! Perché manca una cosa fondamentale: l’**implementazione dell’algoritmo di de Casteljau**! La lista `self.nodes` della classe `Curva_Bezier` contiene l’elenco dei punti di controllo, espressi come tuple (x, y) . Implementa il metodo `deCasteljau` che prende in input il parametro `t`, e utilizza i punti di controllo per calcolare il punto della curva di Bézier per il parametro `t` dato, restituendolo sempre in una tupla (x, y) . Tutto il resto del codice è completo, e anche le chiamate al metodo `deCasteljau` sono presenti.

Esercizio 10+

Osserva la chiamata al metodo `deCasteljau` all’interno del metodo `render`. Prova a modificare il numero delle chiamate a `deCasteljau`, variando opportunamente il valore di `t`. Osserva cosa succede se vengono valutati pochi valori di `t`.

Esercizio 10++

Sperimenta e gioca con l’applicazione!

Osserva il comportamento delle curve di Bézier all’aumentare del numero di punti di controlli. Cosa succede?



Le curve di Bézier hanno innumerevoli applicazioni!

Una limitazione delle curve di Bézier è che all’aggiunta di punti di controllo la curva si allontana progressivamente dalla poligonale, e diventa più difficile da controllare. **Spezzando una lunga curva in più curve di Bézier** che condividono gli estremi, è però possibile **limitare la crescita del grado della curva** e anche creare degli angoli.

- Le curve di Bézier, infatti, sono utilizzate per la creazione di **Typeface**;

Approfondimenti:

1. Un esempio interattivo della lettera “D”:
<https://www.geogebra.org/m/csrHNbAY>
2. Una breve descrizione delle curve di Bézier e della loro applicazione alle font:

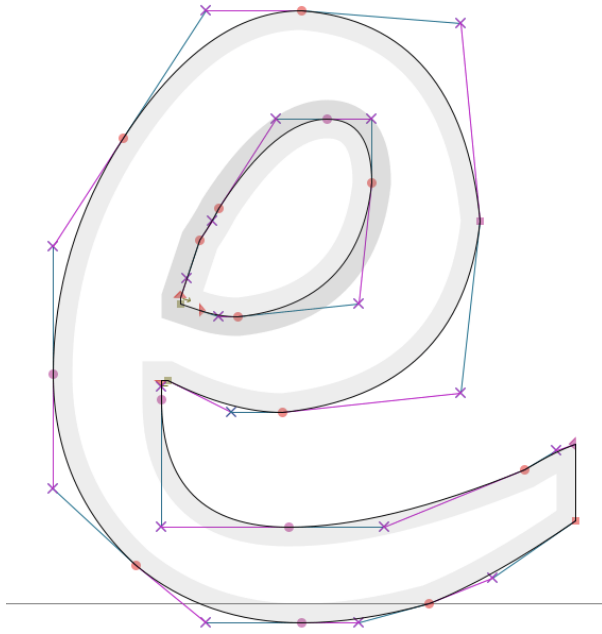


Figura 24: esempio di typeface della lettera “e”.

Credits: [jdhao](#), CC BY-NC-ND 4.0

- In robotica vengono utilizzate per la definizione di **traiettorie**;
- In medicina e nelle scienze biologiche si utilizzano come metodo di approssimazione (ad esempio: approssimare i biomarker della malattia di Alzheimer.[5])
- Nel campo dell'architettura sono usate per la progettazione di costruzioni;
- Nel campo della produzione industriale sono un fondamentale strumento di **modellazione**;
- Una estensione delle curve di Bézier nello **spazio tridimensionale** genera le **superfici di Bézier**, utilizzate ampiamente nella modellazione grafica.

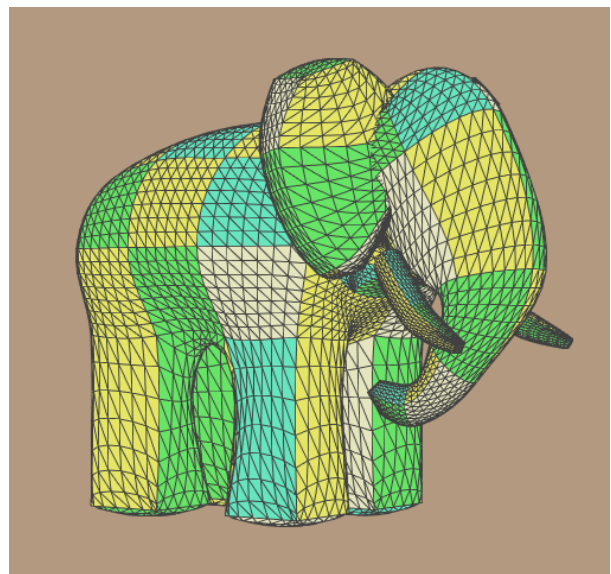


Figura 25: Modello formato da un insieme di patch (“toppe”) di Bézier.

Credits: [Philip Rideout](#)

- Le curve e le superfici di Bézier sono fondamentali nella realizzazione di **animazioni** ed **effetti speciali**, così come moltissimi altri **metodi del calcolo numerico**, e sono quindi presenti in qualsiasi software per la computer grafica 3D (ad esempio: Blender).

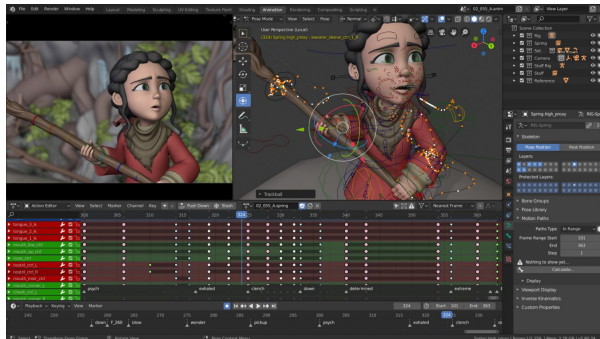


Figura 26: Schermata di animazione di *Blender*, in cui sono visibili le curve che definiscono il movimento dell'animazione.

Credits: Blender

Approfondimenti:

- Semplice tool interattivo per l'applicazione delle curve di Bézier all'animazione:
<https://www.geogebra.org/m/bx3qCjA9>
- Due video sulle curve nell'animazione:
<https://www.youtube.com/watch?v=zaNUFmhD5PM>
<https://www.youtube.com/watch?v=G7sxV2G-OEo>
- La matematica alla Pixar:
<https://www.youtube.com/watch?v=mX0NB9IyYpU&t=10s>
- I *computer* e la matematica alla Pixar:
<https://www.youtube.com/watch?v=2NzTAaYgk4Q>

Concludendo:

Il **calcolo numerico** permette di risolvere problemi matematici che non hanno soluzioni analitiche, o approssimare quelli che sono troppo complicati. E questi problemi matematici si trovano in ogni disciplina scientifica: dalla fisica alla biologia e la medicina.

Ma il calcolo numerico, come abbiamo visto, è fondamentale in innumerevoli situazioni, anche pratiche e quotidiane. Offre strumenti per realizzare e studiare varie tipologie di curve e superfici, applicabili in svariati contesti. Ha rivoluzionato l'animazione e il rendering, e ha reso possibile realizzare accurate simulazioni fisiche, e ha formato le fondamenta per il *machine learning*.

Ecco perché i metodi del calcolo numerico sono così importanti.

Concludiamo (*veramente*) la lezione 3:

Avete dei dubbi su quello che abbiamo visto in questa lezione? Quale argomento avete capito meglio? E quale peggio? **Scrivetelo qui:** http://scrumbler.ca/lezione03_calcolo_numerico0822

Immagine di testata: Webb Space Telescope (NASA, ESA, CSA, STScI)