UNIVERSITÀ DEGLI STUDI DI ROMA "SAPIENZA"

Appunti

GRAFICA COMPUTAZIONALE

Docente:
Biancamaria Della Vecchia

Studenti:
Francesco Gabbuti
Diego Lucaccini
Piervincenzo Russo
Marco Vallecoccia

Anno Accademico 2008-2009

Indice

Introduzione

Negli ultimi anni la grafica computazionale ha avuto molta diffusione in ambito informatico. Essa si occupa essenzialmente di visualizzare oggetti mediante il computer:

–> **Grafica per applicazioni economiche (business graphics)**.

- Presentazione dei risultati
- Nessun algoritmo complicato
- Semplice applicazione della computer graphics

–> **Animazione.** Sfruttando la velocità dei processori è possibile codificare programmi per animare disegni con risoluzione elevata, così da dare la sensazione del movimento continuo. I campi di applicazione:

- videogames
- films
- analisi cinematica. Ad esempio simulare gli spostamenti del corpo di un pilota per migliorarne il posizionamento e ridurre i traumi.

–> **Progettazione elettronica (programmazione dei circuiti integrati e schede**.

- architettonica (abitabilità, studio mobilio, ecc..)
- industriale (progetto tecnico e studio estetico.
- meccanica (CAD). Rende automatico il controllo delle macchine utensili (CAM – Computer Aided Manufacturing)

Le caratteristiche del CAD rendono possibile l'espressione della forma di un oggetto come insieme di numeri, aumentando la facilità di operazione:

1. rappresentazione numerica (applicabilità a varie forme geometriche)
2. operare semplici calcoli
3. indipendente dal sistema di coordinate

4. uso destinato anche a non matematici

Gli oggetti in questione vengono spesso rappresentati attraverso forme **parametriche** che godono delle seguenti caratteristiche:

- esplicita. Dati dei punti è possibie costruire l'oggetto.
- usa la forma vettoriale, combinazione lineare a coefficienti vettoriali di funzioni scalari
- indipendente dal sistema di riferimento
- permette di calcolare separatamente le componenti e di utilizzare le stesse routine.

Capitolo 1

Curve di Bezier

Nel campo matematico della analisi numerica una curva di Bézier è un'importante curva parametrica polinomiale usata nella computer graphics.

1.1 Storia

Le curve di Bézier furono largamente pubblicizzate nel 1962 dal francese Pierre Bézier, un ingegnere meccanico francese che lavorava per la fabbrica Renault, che le usò per disegnare le carrozzerie delle automobili. I problemi con cui Bezier aveva a che fare riguardavano la gestione delle macchine a controllo numerico che tranciavano i pezzi di lamiera con cui fare le carrozzerie delle auto. Le curve furono realizzate nel 1959 da Paul de Casteljau usando l'algoritmo di De Casteljau. Bézier stesso, molto onestamente, ammise che alle stesse curve era giunto anche il suo collega della Citroën De Casteljau, ma i vincoli di segretezza imposti da questa fabbrica hanno fatto sì che tali curve restassero legate al suo nome. Bézier stabilì un modo di realizzare le curve che partiva da due punti e una linea vettoriale appunto, un sistema innovativo che permette ancora oggi agli operatori grafici di realizzare disegni curvilinei bellissimi e precisi.

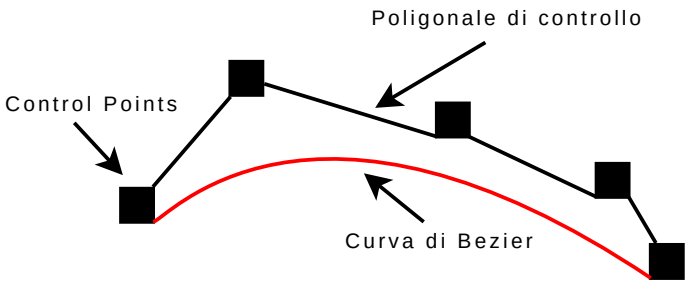
1.2 Definizione

Una curva di Bezier è costituita da:

1. Punti di Controllo [**Control Points**]
2. Poligono di controllo, ovvero la poligonale che unisce i control points

La curva di Bezier viene descritta attraverso la formula:

$$B_n(t) = \sum_{k=0}^{n} \mathbf{p}_k \cdot b_{n,k}(t)$$



con $\mathbf{p}_k \; \varepsilon \; \mathcal{R}^d$, $d \geq 2$
$b_{n,k}(t) = \binom{n}{k} t^k (1-t)^{n-k}$ Funzione di blending $\qquad$ t $\in [0,1]$

Osservando la formula che descrive una curva di Bezièr, notiamo che il punto t sulla curva corrisponde alla media "pesata" di tutti i punti di controllo, dove i pesi sono i coefficienti $B_{n,k}(t)$.

1.3 Proprietà di una curva di Bezier

1. Ben definita. La curva dipende solo da P_i e non dalla scelta dell'origine delle coordinate. Quindi occorre verificare che le funzioni di blending, in somma, diano 1:

$$\sum_{k=0}^{n} b_{n,k}(t) = 1$$

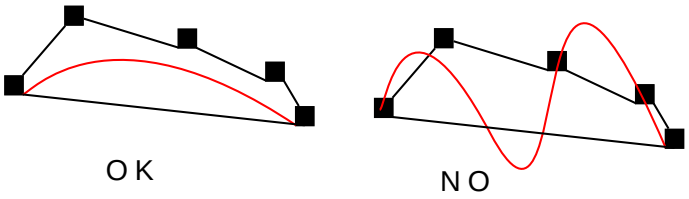
Ovvero:
$\sum_{k=0}^{n} \binom{n}{k} t^k (1-t)^{n-k} = 1$
Essendo:

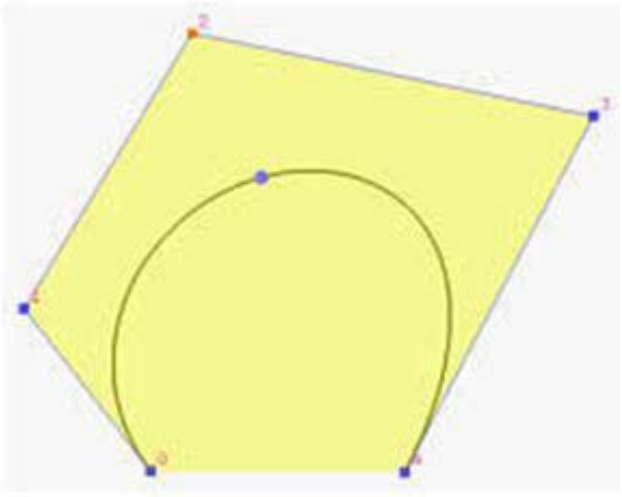
$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$

se sostiuiamo a= t e b= 1-t otteniamo:

$$(t+1-t) = \sum_{k=0}^{n} \binom{n}{k} t^k (1-t)^{n-k}$$

2. La curva giace nel dominio convesso[**convex hull**]. Il dominio convesso è il più piccolo insieme convesso che incorpora tutti i punti.





Si verifica così:

$$b_{n,k}(t) \geq 0$$

Ovvero:

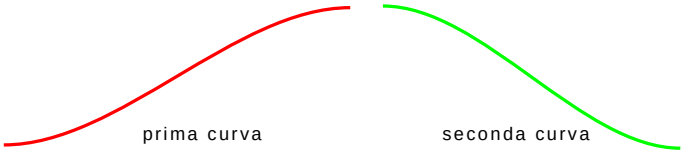
$$b_{n,k}(t) = \binom{k}{t} t^k (1-t)^{n-k} \geq 0 \qquad t \in [0,1]$$

3. Regolarità della curva (smooth).

   - la curva deve essere dolce e regolare
   - le curve di Bezier possono essere facilmente derivate

4. Interpolazione agli estremi. Le curve di Bezier passano sempre attraverso il primo e l'ultimo punto di controllo. Questo è utile per raccordare bracci di curve diverse.

   Quindi: $b_{n,k}(0) = \begin{cases} 0, & k \neq 0 \\ 1, & k = 0 \end{cases}$ $b_{n,k}(1) = \begin{cases} 0, & k \neq n \\ 1, & k = n \end{cases}$



5. Simmetria: invertendo l'ordine dei punti del poligono di controllo, non cambia la forma della curva.

$$b_{n,k}(t) = b_{n,n-k}(1-t)$$

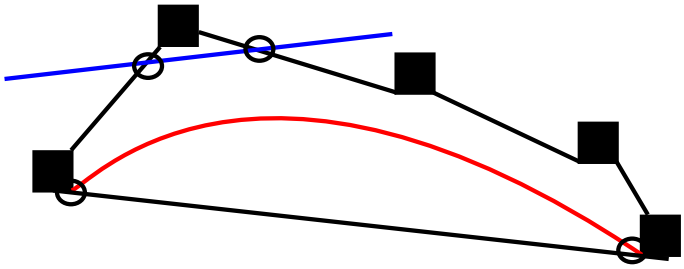
6. Riproduce punti e rette: se i punti del poligono di controllo, nella loro totalità, collassano nel primo punto, anche la curva collasserà nel primo punto.

$$\sum b_{n,k}(t) = 1$$

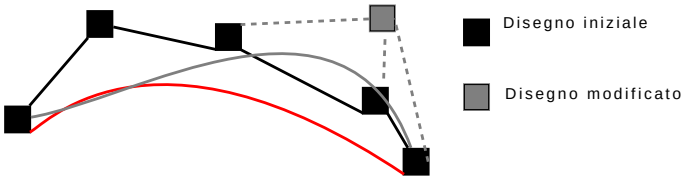
Se invece i punti si alineeranno avvicinandosi ad una retta, anche la curva seguirà l'andamento.

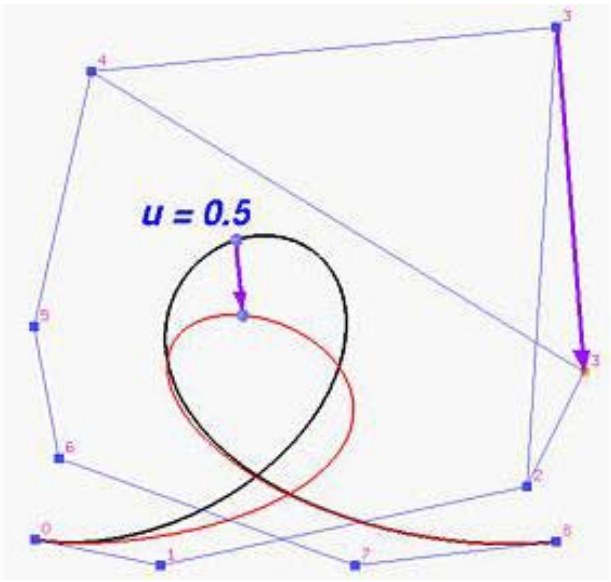
$$\sum b_{n,k}(t)k = nt$$

7. Riduzione delle oscillazioni[**Variation diminishing**]. Il numero di intersezioni di una qualsiasi retta con la curva è sempre minore o uguale al numero di intersezioni della stessa retta con il poligono di controllo.



8. Controllo locale [**Proprietà negativa**]. Spostando un unico punto di controllo, tutta la curva verrà modificata.





9. Invarianza per traformazioni affini. Se occorre una trasformazione (ovvero una rotazione, traslazione, ecc..), si opera sui punti di controllo per ottenere la curva trasformata.

10. Dipendenza grado-punti di controllo[**Proprietà negativa**]. La curva ha grado dipendente dal numero di punti di controllo. Il polinomio aumenta di grado con l'aumento dei punti di controllo. Il grado di una curva di Beziér definita da $n+1$ punti di controllo è n. Ad esempio se il poligono di controllo è costituito da 100 punti, la corrispondente curva di Bezièr ha grado 99. Lavorare con un polinomio di grado elevato è molto negativo per calcoli matematici in quanto crea molta instabilità.

11. La derivata di una curva di Bezier è ancora una curva di Bezier di grado $n-1$

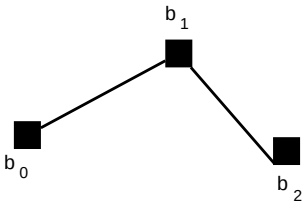
Quindi le curve di Bezier sono limitate e inutilizzabili (a meno di accorgimenti particolari), per le proprietà **8** e **10**.

1.4 Algoritmo di De Casteljau

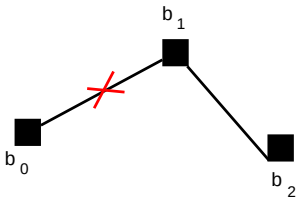
L'algoritmo di De Casteljau viene utilizzato per ricavare le curve di Bezier, lavorando su combinazioni lineari. Alcuni esempi pratici:

1.4.1 Esempio 3 punti

$$B_2(t) = \sum_{k=0}^{2} \binom{2}{k} t^k (1-t)^{2-k} b_k$$

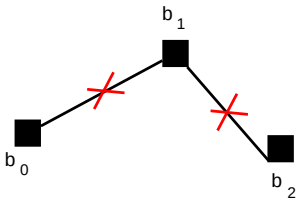


Supponiamo di scegliere $t = \frac{1}{2}$. Si procede così: Calcoliamo il punto medio del segmento $\overline{b_0 b_1}$:



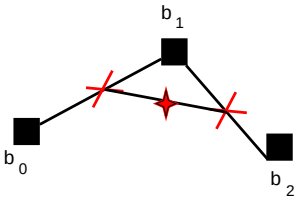
$$b_0^1 = (1-t)b_0 + tb_1$$

Stessa cosa per $\overline{b_1 b_2}$:



$$b_1^1 = (1-t)b_1 + tb_2$$

Unendo b_0^1 e b_1^1 e trovando il suo punto medio:



$$b_0^2 = (1-t)b_1^1 + tb_1^1$$

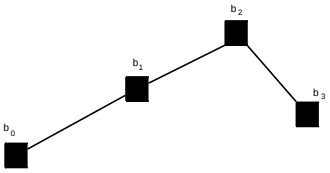
il punto b_0^2 è la curva di Bezier per $t = \frac{1}{2}$.
La verifica si ottiene verificando:

$$b_0^2 = (1-t)[/1-t)b_0 + tb_1] + t[(1-t)b_1 + tb_2]$$

$$=$$

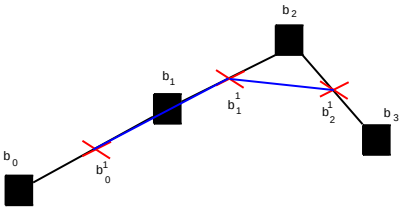
$$(1-t)^2 b_0 + 2t(1-t)b_0 + t^2 b_2 = B_2(\frac{1}{2})$$

1.4.2 Esempio 4 punti



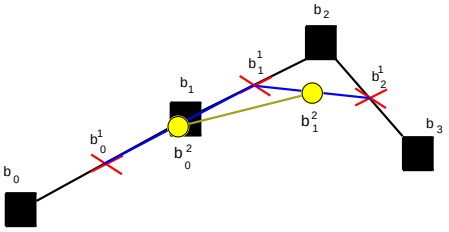
$$B_3(t) = \sum_{k=0}^{3} \binom{3}{k} t^k (1-t)^{3-k} b_k$$

Scegliamo $t = \frac{1}{2}$.
Si procede selezionando i punti medi di ogni segmento:



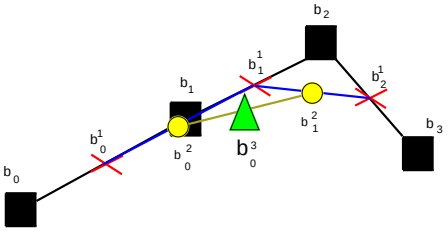
$$b_0^1 = (1-t)b_0 + tb_1$$
$$b_1^1 = (1-t)b_1 + tb_2$$
$$b_2^1 = (1-t)b_2 + tb_3$$

Si opera, nuovamente, la ricerca dei punti medi nei segmenti $\overline{b_0^1 b_1^1}$ e $\overline{b_1^1 b_2^1}$:



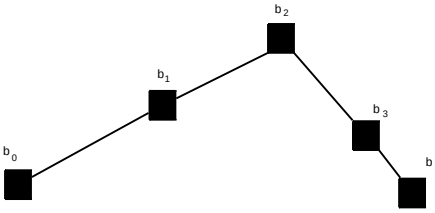
$$b_0^2 = (1-t)b_0^1 + tb_1^1$$
$$b_1^2 = (1-t)b_1^1 + tb_2^1$$

Calcolando il punto medio del segmento $\overline{b_0^2 b_1^2}$, otterremo la curva di Bezier desiderata:



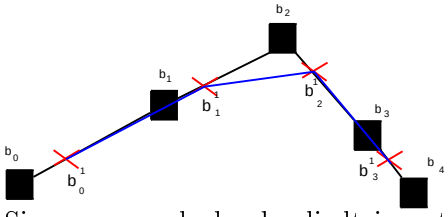
$$b_0^3 = (1-t)b_0^2 + tb_1^2$$

1.4.3 Esempio 5 punti



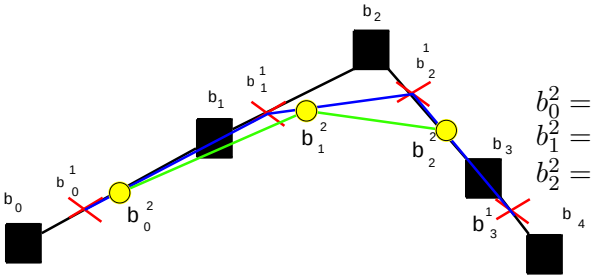
$B_4(t) = \sum_{k=0}^{4} \binom{4}{k} t^k (1-t)^{4-k} b_k$

Scegliamo $t = \frac{1}{3}$.



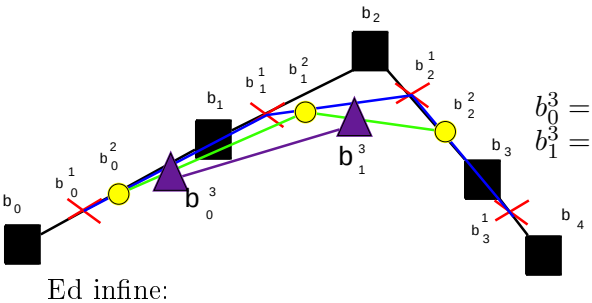
$$b_0^1 = (1-t)b_0 + tb_1$$
$$b_1^1 = (1-t)b_1 + tb_2$$
$$b_2^1 = (1-t)b_2 + tb_3$$
$$b_3^1 = (1-t)b_3 + tb_4$$

Si prosegue calcolando gli altri punti, sul poligono di controllo considerando $t = \frac{1}{3}$:



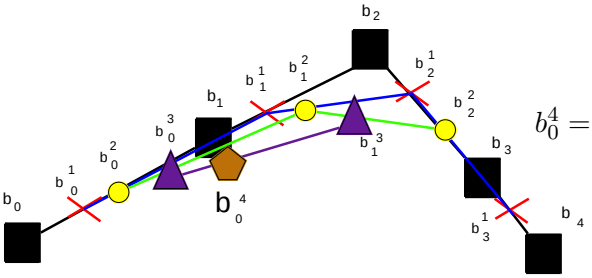
$$b_0^2 = (1-t)b_0^1 + tb_1^1$$
$$b_1^2 = (1-t)b_1^1 + tb_2^1$$
$$b_2^2 = (1-t)b_2^1 + tb_3^1$$

Il prossimo passo:



$$b_0^3 = (1-t)b_0^2 + tb_1^2$$
$$b_1^3 = (1-t)b_1^2 + tb_2^2$$

Ed infine:



$$b_0^4 = (1-t)b_0^3 + tb_1^3$$

1.5 L'algoritmo di De Casteljau: definizione

Dati b_i^0 punti di controllo di partenza, si costruiscono i punti successivi in tal maniera:

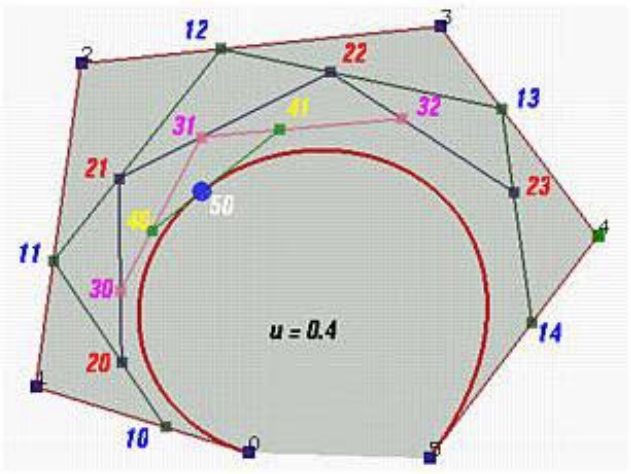
$$b_i^r = (1-t)b_i^{r-1} + tb_{i+1}^{r-1}$$

con $r = 1, \ldots, n$ e $i = 0, \ldots, n-r$. La curva di Bezier potrà essere calcolata in b_0^r ovvero:

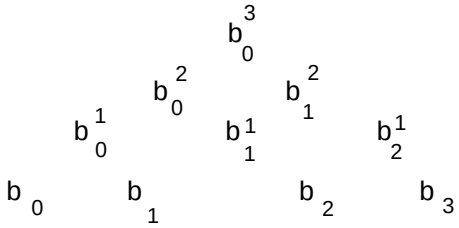
$$b_0^r = b^r(t)$$

1.5.1 Considerazioni

L'algoritmo di De Casteljau è stabile e ricorsivo.
La stabilità è dimostrata considerando i coefficienti $1-t$ e t entrambi ≥ 0. La ricorsività considerando che l'algoritmo si riapplica a se stesso.



Grazie alla **Piramide di De Casteljau** è possibile ordinare le generazioni:

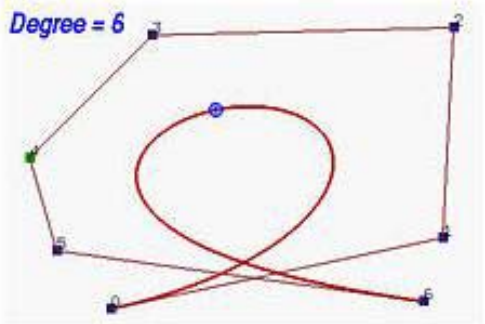


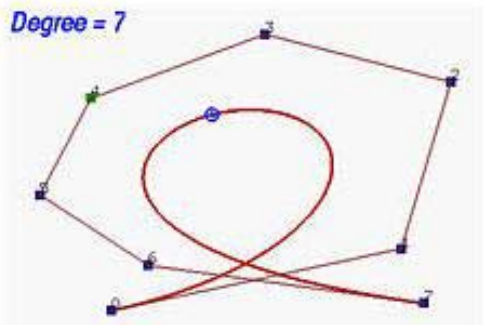
1.6 Algoritmo degree elevation

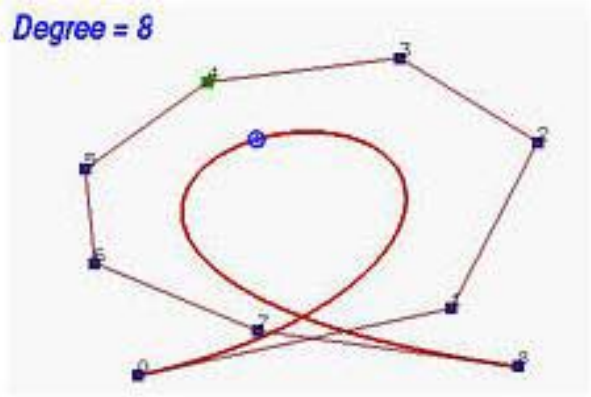
L'algoritmo degree elevation (**elevazione di grado**) permette di aumentare la flessibilità della curva aggiungendo dei punti di controllo. In particolare si parte da un poligono con n punti e si arriva ad un poligono che ne ha $n+1$. Quindi:

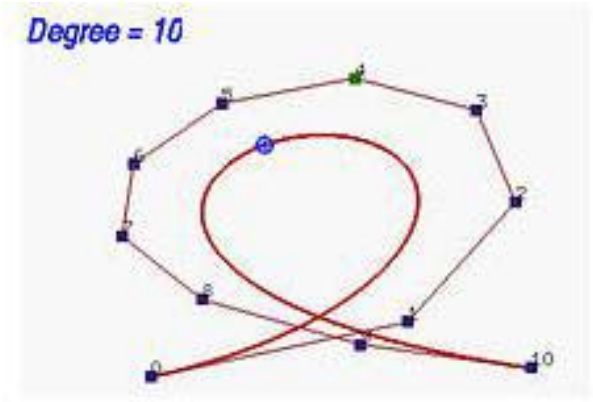
$$b_j^1 = \frac{j}{n+1} \cdot b_{j-1} + \left(\frac{j}{n+1}\right) b_j$$

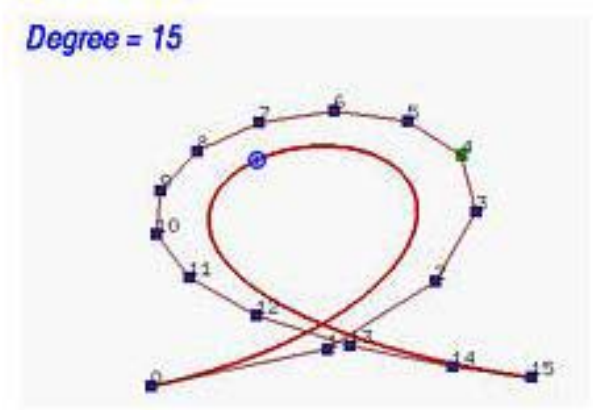
con $j = 1, \cdots, n$.
Chiaramente b_0 e b_{n+1} sono gli estremi del poligono. L'algoritmo può essere utilizzato iterativamente.

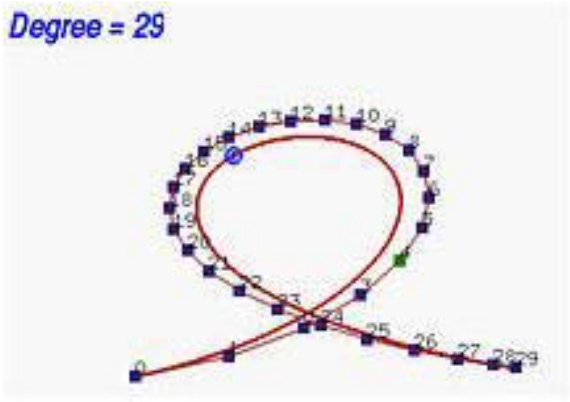

Degree = 6


Degree = 7


Degree = 8







1.6.1 Considerazioni

Aumentando la procedura all'infinito, la spezzata (ovvero il poligono di controllo) si avvicinerà sempre più alla curva di Bezier. L'algoritmo è stabile e ricorsivo, ma molto lento; esso non viene usato in grafica.

1.7 Algoritmo di suddivisione

L'algoritmo di suddivisione è utile quando si vuole modificare parte della curva di Bezier. In particolare, occorre determinare nuovi punti di controllo $r = 0, \ldots, n$, definiti nel nuovo intervallo, tali che la nuova curva di Bezier $C^n(s)$ sia coincidente con la vecchia nell'intervallo di riferimento. I punti di controllo vengono definiti considerando il lato sinistro del triangolo di De Casteljau. La suddivisione della curva ha diverse applicazioni. Per esempio è utile per calcolare l'intersezione di due curve di Bézier, per il rendering delle curve di Bézier e per rendere più semplice il disegno della curva. Supponiamo di aver disegnato una curva ma di non essere soddisfatti del risultato. Possiamo allora voler suddividere la curva in due parti in un punto appropriato, una soddisfacente e l'altra no. Possiamo quindi dimenticarci quella soddisfacente e concentrarci su quella non soddisfacente. Possiamo applicare la suddivisione quante volte vogliamo. Tuttavia va tenuto presente che se vogliamo che i segmenti di curva suddivisi si uniscano in modo dolce dobbiamo mantenere il punto di unione ed i due punti adiacenti collineari.

1.8 Derivata di una curva di Bezier

Sia $C(t) = \sum_{i=0}^{n} b_{n,i}(t)P_i$, la sua derivata, per un punto fissato $c'(t)$:

$$\frac{d}{dt}c(t) = c'(t) = \sum_{i=0}^{n-1} b_{n-1,i}(t) \cdot [n(P_{i+1} - P_i)]$$

Analizzando la parte destra dell'equazione, si può notare che la curva risultante è la stessa, ma con un grado più basso:

$$\sum_{i=0}^{n-1} b_{n-1,\,i}(t)Q_i$$

Gli estremi della curva:

$$C'(0) = n(P_1 - P_0) \qquad\qquad C'(1) = n(P_n - P_{n-1})$$

Dal punto di vista fisico, la derivata è un **odografo**, ovvero il vettore normale alla curva in un punto particolare.

1.9 Continuità

Spesso per descrivere forme utili occorre raccordare più curve. Questa operazione è possibile impostando un grado di continuità comune.

Ci sono due tipi di continuità: **parametrica** e **geometrica**. In generale due curve, continue parametricamente, saranno anche continue geometricamente, ma il viceversa non è verificato. La continuità parametrica viene indicata con C^i dove i indica il grado della continuità parametrica. In particolare C^0 indica che le due curve hanno semplicemente un punto in comune, C^1 indica che le curve hanno sia un punto in comune sia la loro derivata prima è uguale in direzione e modulo.

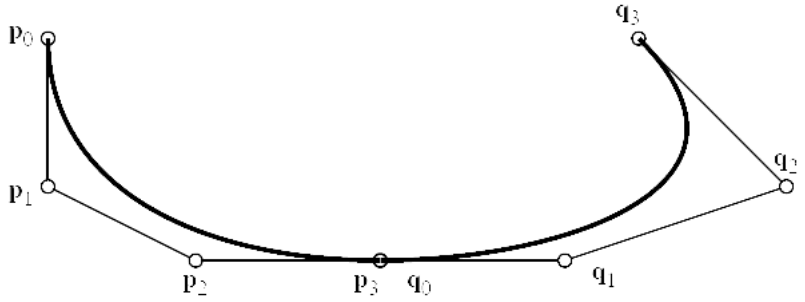


Figura 1.1: Curve di Bezier in C^1

La continuità geometrica, espressa con G^1 ha dei requisiti meno restrittivi; i moduli delle derivate prime possono essere anche diversi ma le loro direzioni devono essere uguali, risulta essere quindi un buon compresso tra C^0 e C^1.

1.10 Curve di Bezier Razionali

Alcune curve che sembrano semplici, come il cerchio, non possono essere descritte da una curva di Bézier, quindi abbiamo bisogno di maggiori gradi di libertà. La curva di Bézier razionale aggiunge pesi che possono essere aggiustati. Il numeratore è una curva di Bézier pesata e il denominatore è una somma pesata di funzioni di blending.
Dati $n+1$ punti di controllo P_i, la curva di Bézier razionale è data da:

$$B(t) = \frac{\sum_{i=0}^{n} b_{i,n}(t) P_i w_i}{\sum_{i=0}^{n} b_{i,n}(t) w_i}$$

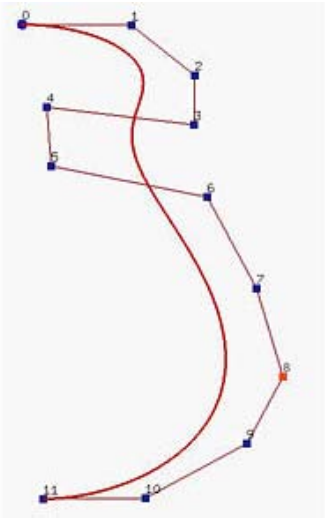
o semplicemente:

$$B(t) = \frac{\sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i w_i}{\sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} w_i}$$

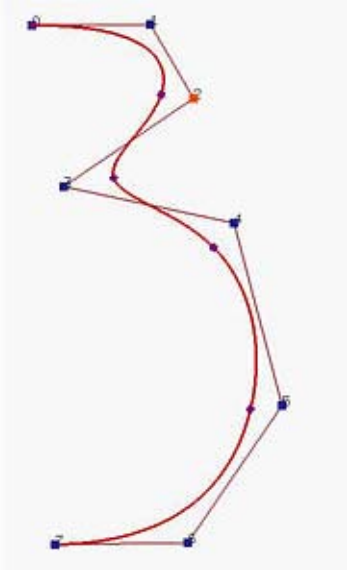
Capitolo 2

Curve B-Splines

Nel campo della CAGD , una curva può avere una forma così complessa da non poter essere rappresentata da una curva di Beziér. Supponiamo di voler disegnare il profilo di un vaso.

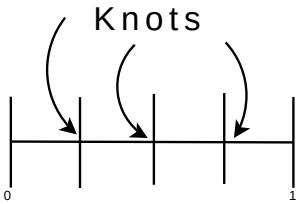


In figura possiamo osservare come sia difficile per una curva di beziér modellare il "collo" del vaso in corrispondenza del segmento P_4P_5. Aumentare il grado della curva aggiunge flessibilità ma di conseguenza aumenta in modo significativo lo sforzo di trasformazione e di manipolazione. Inoltre lavorare con una curva di Bezièr di grado elevato può causare errore nel calcolo numerico. In questi casi si ricorre alle curve B-Spline che sono generalizzazioni delle curve di Bézier.

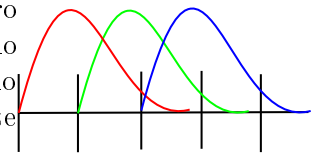


In figura osserviamo una curva B-Spline di grado 3 definita da 8 punti di controllo. Infatti, ci sono cinque segmenti di una curva di Bézier di grado 3 uniti per formare la curva B-Spline definita dai punti di controllo. Questi punti suddividono la curva B-Spline in segmenti di una curva di Bézier. Possiamo muovere dei punti di controllo per modificare la forma della curva proprio come facciamo con le curve di Bézier. Possiamo anche modificare la suddivisione della curva. Quindi, le curve B-Spline hanno un grado di libertà più alto per disegnare una curva.

La premessa utile per introdurre le curve B-Spline viene riassunta in due punti:

1. 

Il dominio viene suddiviso dai **knots** (nodi).

2. Le funzioni base non sono diverse da zero sull'intero intervallo. Questo implica che le funzioni assumono valori diversi da zero solo per una parte dell'intervallo di riferimento e sono, di conseguenza, completamente 'locali".


Il vettore dei nodi (denotato con U) è composto da $m+1$ numeri non decrescenti $u_0 \leq u_1 \leq \ldots \leq u_n$. L'intervallo $[u_i, u_{i+1})$ corrisponde all'intervallo tra due knots e si chiama **knot span**.

Scegliendo alcuni u_i tali che $u_i = u_{i+1} = u_{i+k-1}$, essi appariranno collassati in un unico nodo con il conseguente collasso degli intervalli tra nodi. In questo caso il nodo u_i è un **nodo multiplo** e viene soddisfatta la **molteplicità k** ovvero k nodi di fatto coincidono.

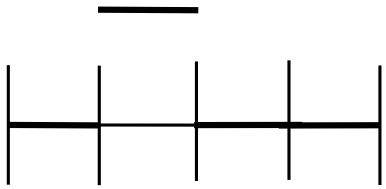
Se invece scegliamo $u_i \neq u_{i+1}$ ovvero tutti i nodi diversi tra loro avremo un **vettore semplice**. Se scegliamo i u_i tutti equidistanti avremo un **vettore uniforme**. Infine se $u_0 = 0$ e $u_m = 1$ si opererà con un **vettore standard**.

Le funzioni base $N_{i,p}(u)$vengono definite considerando un parametro p che viene fissato.

Generalmente il valore di p è 2 oppure 3. Questo rende possibile il disegno di curve regolari.

Considerando $p = 0$ (il che indica che il polinomio associato avrà grado 0 e quindi la funzione sarà costante e parallela all's X):

$$N_{i,0} = \begin{cases} 1, & u_i \leq u < u_{i+1} \\ 0, & altrimenti \end{cases}$$

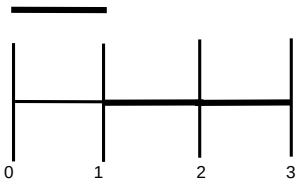


Il risultato, come si evince dalla figura, rappresenta funzioni a gradino. Generalizzando, si introduce la relazione di ricorrenza di Cox-DeBoor:

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} \cdot N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} \cdot N_{i+1,p-1}(u)$$

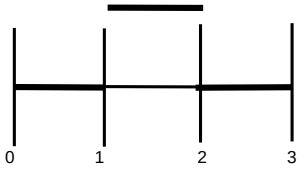
2.0.1 Esempio B-Spline

L'esempio, in forma integrale, può essere reperito on-line sul sito internet del corso. Consideriamo l'intervallo $[0, 3]$, $p = 0$.
I knots saranno $u_0 = 0$, $u_1 = 1$, $u_2 = 2$, $u_3 = 3$ e $u_0 = 0$, $u_1 = 1$ $u_2 = 2$
Calcoliamo la curva per $i = 0$,$p = 0$ in $[0, 1)$:



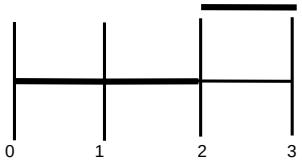
$$N_{0,0} = \begin{cases} 1, & \text{tra 0 e 1} \\ 0, & fuori \end{cases}$$

Con $i = 1$ e $p = 0$, considerando $[1, 2)$:



$$N_{1,0} = \begin{cases} 1, & \text{tra 1 e 2} \\ 0, & fuori \end{cases}$$

E infine $i = 2$, $p = 0$ in $[2, 3]$:



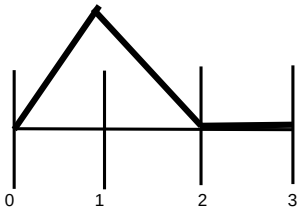
$$N_{2,0} = \begin{cases} 1, & \text{tra 2 e 3} \\ 0, & fuori \end{cases}$$

I grafici risultanti rappresentano delle funzioni a gradino, assolutamente inadeguate per modellare delle curve non essendoci continuità.

Il prossimo passo è incrementare di una unità la p:

$$N_{0,1}(u) = \frac{u - u_0}{u_1 - u_0} \cdot N_{0,0}(u) + \frac{u_2 - u}{u_2 - u_1} \cdot N_{1,0}(u) =$$
$$u \cdot N_{0,0}(u) + (2 - u) \cdot N_{1,0}(u)$$

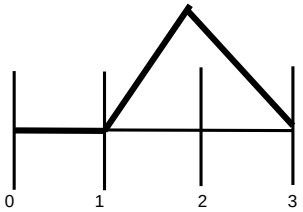
Quindi per $u \; \varepsilon \; [0, 1)$:
$u \cdot 1 + (2 - u) \cdot 0 = u$;
per $u \; \varepsilon \; [1, 2)$:
$u \cdot 0 + (2 - u) \cdot 1 = 2 - u$.
Per l'intervallo $[2, 3]$ si ha 0.



Allo stesso modo calcoliamo $N_{1,1}$:

$$N_{1,1}(u) = \frac{u - u_1}{u_2 - u_1} \cdot N_{1,0}(u) + \frac{u_3 - u}{u_3 - u_2} \cdot N_{2,1}(u) =$$
$$u - 1 \cdot N_{1,0}(u) + 3 - u \cdot N_{2,0}(u)$$

Quindi per $u \; \varepsilon \; [1, 2)$:
$u - 1 \cdot 1 + (3 - u) \cdot 0 = u - 1$;
per $u \; \varepsilon \; [2, 3)$:
$u - 1 \cdot 0 + (3 - u) \cdot 1 = 3 - u$.
Per l'intervallo $[1, 2]$ si ha 0.



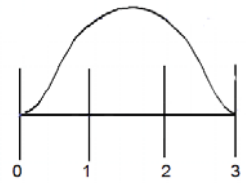
Nuovamente, questa rappresentazione evidenzia la difficoltà di modellare una qualsiasi curva. Inoltre le derivate delle due rette sono diverse.

Si effettua un nuovo incremento di p. Per $p = 2$:

$$N_{0,2}(u) = \frac{u - u_0}{u_2 - u_0} \cdot N_{0,1}(u) + \frac{u_3 - u}{u_3 - u_1} =$$

$$0,5u \cdot N_{0,1}(u) + 0,5(3 - u)N_{1,1}(u).$$

Quindi per $u \varepsilon [0, 1)$:
$0,5 \cdot u^2$ per $u \varepsilon [1, 2)$:
$0,5 \cdot (3 - u)^2$
Per l'intervallo $[2, 3]$:
$0,5 \cdot (-3 + 6u - 2u^2)$
Calcolando anche $N_{1,2}$ e $N_{2,2}$
(è possibile visionare l'esempio integrale
sulle dispense pubblicate sul sito internet
del corso)si ottiene la curva mostrata in figura.



2.1 Definizione di curve B-Spline

Costruiamo la curva:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u)P_i$$

La forma di una curva B-Spline è molto simile a quella di una curva di Bezièr (somma, funzioni base, punti). A differenza di una curva di Bezièr, una curva B-Spline coinvolge più informazioni: un insieme di $n+1$ punti di controllo, un vettore dei knot di $m + 1$ knots e un grado p.

2.2 Proprietà delle B-Spline

Sebbene restano valide le proprietà già elencate e discusse per le curve di Bèzier, vengono indicate alcune proprietà particolari che dimostrano la superiorità funzionale delle curve B-Spline.

1. $N_{i,p}$ polinomio di grado p in u. Questo implica che per ogni intervallo viene definito un polinomio. In totale si otterrà un polinomio a tratti raccordabile.

2. $N_{i,p}(u) \geq 0$. La curva non si discosta dai punti del poligono di controllo.

3. Supporto locale (fuori dall'intervallo $[u_i, u_{i+p+1})$ le $N_{i,p}$ valgono 0. Quindi è possibile modellare bene una curva poichè non tutte le $N_{i,p}(u)$ giocano un ruolo.

4. Partizione unità. Si ha che $\sum N_{i,p}(u) = 1$. Quindi la curva non dipende dalla scelta dell'origine.

5. Se il numero di knots è $m+1$, il grado delle funzioni base è p, ed il numero delle funzioni base di grado p è $n+1$, allora $m = n+p+1$. (questo concetto verrà approfondito in seguito)

6. $N_{i,p}(u)\ \varepsilon\ c^{p-k}$. La smoothness dipende dalla k ovvero dalla molteplicità. All'aumentare di k si perde regolarità. Di conseguenza all'aumentare del grado p cresce la continuità, ovvero la regolarità.

7. In ogni nodo interno di molteplicità k, in numero di funzioni base non nullo è al più $p-k+1$

2.2.1 Osservazioni

> Una curva B-Spline non passa necessariamente per i punti di controllo estremi del poligono di controllo. La strategia da usare per imporre il passaggio è assegnare molteplicità p ai knots che coincidono con gli estremi del poligono di controllo. Ponendo che u_0 e u_m abbiano molteplicità p, la curva passa per gli estremi.

> Le curve di Bèzier sono un caso particolare delle curve B-Spline. . Il grande vantaggio rispetto alle curve di Bezièr è quello del controllo locale e dell'indipendenza del grado del polinomio dal numero dei punti del poligono di controllo. Il grado infatti viene fissato dal designer e corrisponde al parametro p.

> Proprietà **Strong convex hull**: la curva è contenuta nel dominio convesso. Data la curva B-Spline $C(u)$ con $u\varepsilon[u_i, u_{i+1})$, la curva giace nel dominio convesso definito dai punti $P_{i-p} \dots P_i$

> Proprietà della derivata:

$$\frac{dC(u)}{du} = C'(u) = \sum_{i=0}^{n-1} N_{i+1,p-1}(u) Q_i$$

$$Q_i = \frac{p \cdot (P_{i+1} - P_i)}{u_{i+p+1} - u_{i+1}}$$

Ovvero siamo in presenza di una curva B-Spline, ma di un grado inferiore.

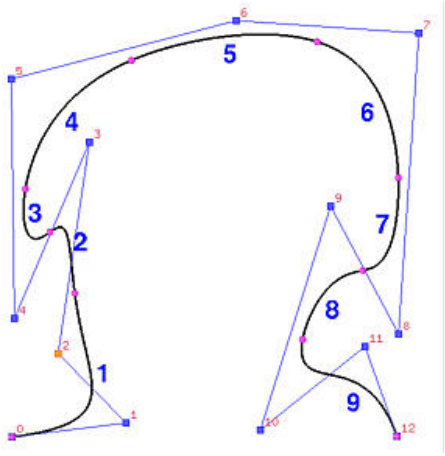
2.3 Curve di Bèzier vs. Curve B-Spline

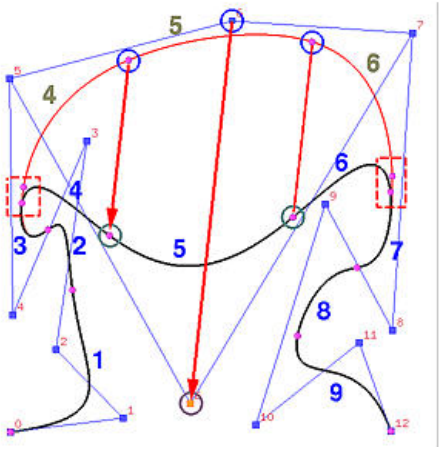
Analizziamo qui di seguito quali sono i punti che accomunano le curve di Bèzier e le B-Spline e quali invece le differenze:

1. La curva è contenuta all'interno dell'involucro convesso del poligono di controllo?

- Bèzier: SI
- B-Spline: SI

2. Come si lega il grado della curva rispetto al poligono di controllo?
   - Bèzier: Grado = Numero di vertici -1
   - B-Spline: Indipendente dal poligono di controllo
3. La curva tocca il poligono di controllo?
   - Bèzier: Si, nel primo e nell'ultimo punto
   - B-Spline: No, ma può essere fatto, collassando tanti nodi quanto il grado della curva. Operando questa routine, difatti si può farla passare per qualsiasi punto del poligono.
4. La forma della curva e del poligono sono in qualche modo correlate?
   - Bèzier: No, solo in modo molto approssimato
   - B-Spline: Si e li si può addirittura far coincidere, con particolari accorgimenti
5. Si può avere una trasformazione affine sulla curva?
   - Bèzier: Si, operando sui vertici del poligono di controllo
   - B-Spline: Si, operando sui vertici del poligono di controllo
6. Si può avere il controllo locale sulla curva?
   - Bèzier: NO
   - B-Spline: SI
7. Si possono disegnare cerchi ed ellissi?
   - Bèzier: NO
   - B-Spline: NO





In figura osseriviamo il controllo locale di una curva B-Spline.

2.4 Il vantaggio di utilizzare curve B-Spline

Le curve B-Spline richiedono più informazioni (cioè, il grado della curva e un vettore dei knot) e una teoria più complessa rispetto alle curve di Bézier. Ma, i maggiori vantaggi compensano questo inconveniente. Primo, una curva B-Spline può essere una curva di Bézier. Secondo, le curve B-Spline soddisfano tutte le proprietà importanti che hanno le curve di Bézier. Terzo, le curve B-Spline forniscono più flessibilità di controllo rispetto a ciò che possono fare le curve di Bézier. Per esempio, il grado di una curva B-Spline è separato dal numero di punti di controlli. Più precisamente, possiamo utilizzare curve di grado più basso e mantenere ancora un gran numero di punti di controllo. Possiamo cambiare la posizione di un punto di controllo senza cambiare globalmente la forma dell'intera curva (proprietà di modifica locale). Poichè le curve B-Spline soddisfano la proprietà strong convex hull, esse hanno un maggiore controllo della forma. Inoltre, ci sono altre tecniche per disegnare ed editare la forma di una curva come cambiare knots.

Comunque, va ricordato che le curve B-spline (di conseguenza anche le curve di Beziér) sono ancora curve polinomiali e le curve polinomiali non possono rappresentare molte utili curve semplici come cerchi e le elissi. Quindi è richiesta, una generalizzazione delle B-spline, NURBS. Discuteremo le NURBS più avanti.

2.5 Algoritmo Knot insertion

Obiettivo: inserire un nuovo nodo nel vettore dei knots, senza cambiare la forma della curva. Considerando la proprietà $m = n + p + 1$ appare chiaro che all'aumentare di m segue un aumento di n in quanto il valore di p viene fissato. All'aumentare dei nodi all'interno del vettore dei knot segue quindi

l'aumento del numero dei punti del poligono di controllo. Dalla proprietà della **strong convex hull** la curva $C(t)$, $t\varepsilon[u_k, u_{k+1}, \ldots)$ giace nel dominio convesso definito da $P_k, P_{k-1} \ldots, P_{k-p}$ e le funzioni base degli altri p_i sono $= 0$ (ovvero sono nulle). Perciò l'inserimento viene fatto considerando unicamente i punti $P_k, \ldots, P_{k-p}$.

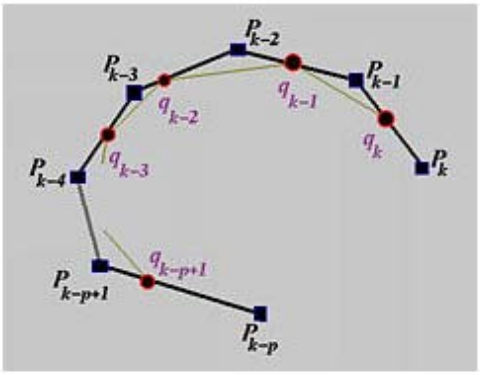
Occorre trovare p nuovi punti Q_k tali che il vecchio poligono tra P_{k-p} e P_k sia sostituito dai $P_{k-p}Q_{k-p+1} \ldots Q_k P_k$:

$$Q_i = (1 - a_i) \cdot P_{i-1} + a_i P_i \qquad a_i = \frac{t - u_i}{u_{i+p} - u_i}$$

$$k - p + 1 \leq i \leq k$$

2.5.1 Algoritmo di De Boor

E' una generalizzazione dell'algoritmo di De Casteljau. Un modo veloce e numericamente stabile per trovare un punto su una curva B-Spline dato un u ($u\ \varepsilon\ [0, 1)$). Ricordiamo che incrementando la molteplicità di un knot interno decresce il numero di funzioni base non nulle che attravesano questo knot. Infatti se la molteplicità di questo knot è k, ci sono al più $p - k + 1$ funzioni base non nulle che attraversano questo knot. Questo implica che in un knot di molteplicità p, ci sarà solo una funzione base (essendo $p - p + 1 = 1$) non nulla il cui valore in corrispondenza di questo knot è 1 per la proprietà di partizione delll'unità.
Se $u = u_i$, poichè $N_{i,p}(u) \neq 0$ in $[u_i, u_{i+1})$, il punto sulla curva $C(u)$ viene influenzato da esattamente un punto di controllo P_i.
Quindi $C(u) = N_{i,p}(u)P_i = P_i$.
Se un knot viene inserito ripetutamente in modo che la sua molteplicità sia p, l'ultimo nuovo punto di controllo generato, è il punto della curva che corrisponde ad u. Dopo aver inserito u più volte (portando la sua molteplicità a p), la curva deve passare per questo nuovo punto, esso è il punto che sulla curva corrisponde ad u.
Questa ossservazione ci fornisce una **tecnica per trovare** $C(u)$. Inseriamo u finchè la sua molteplicità diventa p e l'ultimo punto è $C(u)$.
L'algoritmo di inserimento dei knots può essere facilmente modificato per soddisfare il nostro scopo. Dobbiamo inserire u solo il numero di volte necessario a far diventar u, un knot di molteplicità p.



Algoritmo di De Boor: dettaglio

L'algoritmo riceve in input u e produce l'output $C(u)$, $u\ \varepsilon[u_k, u_{k+1})$ e $u = u_k$. Definiamo un contatore s che indica la molteplicità del knot u:

- Se $s = 0$, sia $h = p$ (cioè inseriamo u, p volte)
- Altrimenti se $u = u_k$ ha molteplicità s, sia $h = p - s$ (cioè inseriamo u, $p - s$ volte)

I nuovi punti di controllo vengono così definiti:

$$P_{k-s,0}P_{k-s-1,0}\dots P_{k-p,0}$$

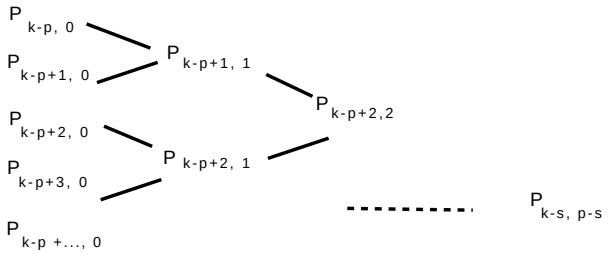
dove lo 0 indica lo step iniziale;

$$P_{i,r} = (1 - a_{i,r}) \cdot P_{i-1,r-1} + a_{i,r} \cdot P_{i,r-1}$$

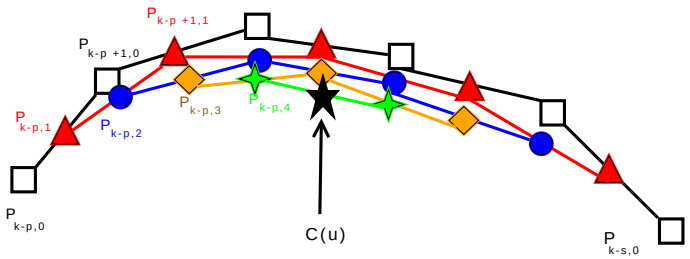
con $k - p + r \leq i \leq k - s$ e:

$$a_{i,r} = \frac{u - u_i}{u_{i+p-r+1} - u_i}$$

Il punto di controllo $P_{k-s,p-s}$ rappresenta la curva B-Spline C(u).



Graficamente:



Considerazioni

Il trovare il punto di controllo $P_{i,r}$ dipende rispettivamente dall'indice i, dal contatore r indica il numero di iterazioni per portare u ad avere molteplicità p e dal grado p, a differenza dell'algoritmo di Casteljau nel quale la dipendenza era solo sull'indice i. L'algoritmo di de Boor è stabile e ricorsivo. La stabilità permette di non amplificare l'errore di elaborazione, dato che si lavora con combinazione lineare positiva di somma 1; la ricorsività implica la riapplicazione a se stesso.

2.6 Il problema della scelta dei knots

Al fine di modellare un oggetto garantendo una certa estetica, è necessario scegliere adeguatamente i knots. In generale si vuole ottenere:

- minima curvatura
- bel grafico
- rispetto di alcune proprietà definite (es. aereodinamicità)

Scegliendo knots equidistanti ovvero:

$$u_{j+p} = \frac{j}{n-p+1}$$

$$j = 1 \dots n-p$$

i risultati sono molto scadenti. Infatti è possibile che la modellazione si discosti dall'oggetto desiderato.
Calcoliamo u_{j+p} in maniera diversa (average):

$$u_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} t_i$$

con $j = 1, \dots, n-p$. A questo punto il problema è come scegliere t_i.

- $t_i = \frac{i}{n}$. Non va bene, ottengo una distribuzione uniforme.

- **Chord Length**: setto $t_0 = 0$ e $t_n = 1$, i restanti $t_k = \frac{1}{L}\left(\sum_{i=1}^{k} |P_i - P_{i-1}|\right)$ con $L = \sum |P_i - P_{i-1}|$ con $L = \sum |P_i - P_{i-1}|$. Quindi se una curva si avvicina al poligono di controllo, la lunghezza del segmento della curva tra due punti di controllo vicini è prossima alla lunghezza della **corda** di questi due punti e la lunghezza della curva è prossima alla lunghezza totale del poligono di controllo.
- **Centripetal**: $t_k = \frac{1}{L}\left(\sum_{i=1}^{k} |P_i - P_{i-1}|^{\alpha}\right)$:
  1. se $\alpha = 1$, abbiamo Chord Length
  2. se $\alpha = \frac{1}{2}$, centipetal originale.

2.7 Interpolazione globale

Dati $n+1$ punti $d_0, d_1, \ldots, d_n$ occorre costruire una curva B-Spline di grado p fissato tale che passi attraverso i punti $d_0, d_1, \ldots, d_n$ appena indicati. I valori dei parametri $t_0, \ldots, t_n$ sono calcolati utilizzando i metodi noti. Da questi parametri si calcola il vettore dei knots di $m+1$ punti. Quindi l'incognita del sistema è costituita dagli $n+1$ punti di controllo $p_0, p_1 \ldots, p_n$ con

$$p(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i$$

Quindi:

$$d_k = p(t_k) = \sum_{i=0}^{n} N_{i,p}(t_k) P_i$$

con $k = 0, \ldots, n$. La matrice delle funzioni base calcolata dei punti parametrizzati:

$$N = \begin{bmatrix} N_{0,p}(t_0) \ldots N_{n,p}(t_0) \\ N_{0,p}(t_n) \ldots N_{n,p}(t_n) \end{bmatrix}$$

$$D = \begin{bmatrix} d_{0,1} & d_{0,2} & \ldots & d_{0,s} \\ \vdots & & & \\ d_{n,1} & d_{n,2} & \ldots & d_{n,s} \end{bmatrix}$$

I d_k sono trattati come vettori nello spazio s-dimensionale. Mentre:

$$P = \begin{bmatrix} P_{0,1} & P_{0,2} & \ldots & P_{0,s} \\ \vdots & & & \\ P_{n,1} & P_{n,2} & \ldots & P_{n,s} \end{bmatrix}$$

rappresenta il vettore "incognita". Tipicamente lo spazio s è di dimensione 2 o 3. Per ogni i, quindi, occorre risolvere il sistema $D = N \cdot P$ colonna per colonna:

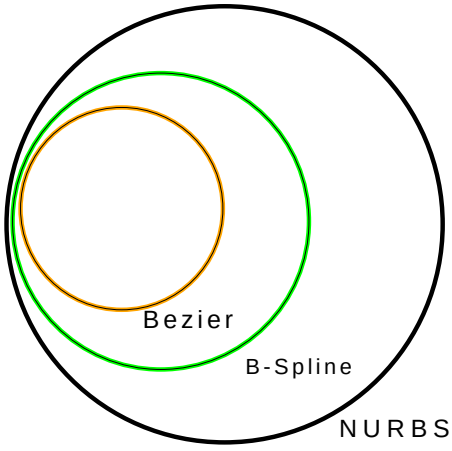
sia d^i i-esima colonna di D e p^i la i-esima colonna di P

$$d^i = N \cdot p^i$$

Capitolo 3

Curve Nurbs

Le curve NURBS modellano cerchi ed elissi. L'acronimo NURBS indica Non Uniform Rational B-Spline. In particolare il termine Rational indica che le funzioni sono a tratti del tipo polinomio su polinomio (anche di grado diverso). Inoltre esse rappresentano un caso generale di B-Spline. Quindi in un certo senso:



3.1 Storia

La paternità delle NURBS è dibattuta. Vi sono almeno due candidati che competono per questo titolo.

- Ken Versprille.Egli si autopromuove come inventore asserendo di aver ideato le NURBS quando anni fa lavorava per ComputerVison, al tempo una delle maggiori aziende CAD del mondo. Il 5 aprile 2005 Ken ha ricevuto dalla CAD Society il CAD Society Lifetime Award per questa suo contributo alla comunità CAD.

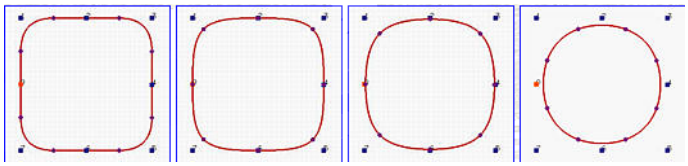
- Les A. Piegl e Wayne Tiller. La comunità scientifica riconosce unanimemente in Les A. Piegl e Wayne Tiller i padri delle NURBS. A testimonianza del loro contributo fondamentale a questa tecnologia, vi è il poderoso volume The NURBS Book di ben 645 pagine, che costituisce l'indisputato testo di riferimento per chiunque si occupi di NURBS. Les A. Piegl è professore di computer science e ingegneria presso l'University of South Florida mentre Wayne Tiller è attualmente occupato in Geomware, un'azienda creata per sviluppare e commercializzare la libreria NURBS chiamata Nlib, descritta nel volume.

3.2 Perchè usare le curve NURBS?

Come già visto in precedenza le curve B-Spline non hanno espressività tale da rendere possibile la modellazione di ellissi e cerchi, sebbene siano molto pressibili e ricche di proprietà interessanti. I cerchi e le ellissi possono essere rappresentate solamente con funzioni razionali. Occorre quindi estendere ulteriormente le B-Spline.

3.2.1 Perchè, le curve B-Spline non riescono a rappresentare un cerchio?

Ebbene consideriamo un piccolo esempio grafico:



Le quattro immagini rapprentano delle curve B-Spline di grado diverso con 8 punti di controllo. Da sinistra verso destra i gradi delle curve sono rispettivamente 2, 3, 5, 10. Appare chiaro che il grado 2 della prima curca (prima figura a sinistra) è fin troppo basso per rappresentare un cerchio (ma piuttosto un quadrato con i bordi arrotondati); con il grado 3 si inizia ad apprezzare qualche miglioramento, ma siamo ancora lontani dal nostro obiettivo, ovvero disegnare un cerchio. Così anche il grado 5 è insufficiente e solo il grado 10 si avvicina al risultato sperato. Ma non abbiamo ancora disegnato un cerchio e inoltre un grado tanto alto è inaccettabile!

3.3 Modellare le curve NURBS

Le curve NURBS vengono modellate usando:

- $u_0 \dots u_n$: **vettore dei nodi**,

- $N_{i,p}(u)$: **funzione base**,
- p: grado fissato,
- $w_i \geq 0$: **pesi** che rappresentano l'importanza del nodo associato. In teoria il peso w_i associato al punto potrebbe essere anche negativo, in realtà l'associare un valore $w_i \leq 0$ è una pratica non utilizzata.

Ai punti di controllo viene, quindi, attribuito un peso che identifica l'apporto dato alla modellazione della curva. Con w alto, la curva tenderà ad avvicinarsi al punto di controllo ad esso associato mentre con $w = 0$ la curva non sarà influenzata da quel punto di controllo. Chiaramente con w negativo la curva si allontanerà dal punto di controllo indicato. Occorre ricordare che il setting negativo del peso è una forma poco utilizzata.

3.4 Definizione di curva NURBS

Una curva NURBS viene indicata come:

$$C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i P_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i}$$

Nel caso in cui $w_i = 1$, $C(u) =$ B-Spline. Infatti:

$$C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i P_i}{1}$$

3.4.1 Risultati immediati

Dalla definizione di curva NURBS seguono due risultati molto importanti:

1. Se tutti i pesi sono uguali ad 1, la curva NURBS si riduce ad una B-Spline. Questo è banale poichè i control points in forma omogenea sono identici al loro equivalente Cartesiano.
2. Le curve NURBS sono razionali. Anche questo è facilmente osservabile: le funzioni base, moltiplicate per il vettore dei control points P_i sono una forma polinomiale di grado p; allo stesso modo il denominatore è la somma di tutti i coefficienti ed è ancora in forma polinomiale di grado p. Quindi il coefficiente del vettore control point è il quoziente di due forme polinomiali di grado p e inevitabilmente $C(u)$ è razionale.

3.5 Coordinate omogenee

La geometria Euclidea classica non ha il concentto di ∞; si opera in $\mathcal{R}^2$ o $\mathcal{R}^3$ utilizzando rispettivamente (x, y) e (x, y, z).

L'idea è quella di estendere il concetto classico:
in $\mathcal{R}$, consideriamo la coppia (a, w) punto che indica $\frac{a}{w}$ con $a\ \varepsilon\mathcal{R}$ e $w \neq 0$.
Se $w = 0$, quindi $(a, 0)$ abbiamo ∞. Siamo in presenza di un punto all'infinito.
Questo estende concettualmente la trattazione dello spazio di riferimento $\mathcal{R}$.
Nel caso di $\mathcal{R}^2$, abbiamo come punto di partenza:

$$Ax + By + C \leq 0$$

La coppia (x, y) diventa $\left(\frac{x}{w}\right), \left(\frac{y}{w}\right)$ ovvero:

$$A\left(\frac{x}{w}\right) + B\left(\frac{y}{w}\right) + C \leq 0,$$

$$Ax + By + Cw = 0$$

Ad esempio prendiamo la tripla $(3, 4, 5)$ dove $x = 3$, $y = 4$ e $w = 5$; ebbene la trasformazione in coordinate omogenee ha come risultato la coppia $\left(\frac{3}{5}\right), \left(\frac{4}{5}\right)$

3.5.1 Trasformazioni Euclidee

- Non cambiano la lunghezza o angoli.

  1. Traslazioni. Sono le trasformazioni più semplici, date le coordinate omogenee:

     $$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

     l'obiettivo è traslare nel piano aggiungendo un vettore con coordinate $[h, k]$. Questo vuol dire che a x devo aggiungere h, a y devo aggiungere k e la terza coordinata rimane invariata. Questa trasformazione può essere anche espressa in forma matriciale:

     $$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

     E' facile osservare che $x' = x + h$ e $y' = y + k$.
     La trasformazione inversa:

     $$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -h \\ 0 & 1 & -k \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

  2. Rotazioni

$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ ruotato di un angolo α rispetto all'origine:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

3. Traslazione più rotazione.
   Chiaramente le operazioni espresse in forma matriciale sono più compatte e possono includere più istruzioni. Ad esempio è possibile esprimere le operazioni di traslazione e rotazione in un'unica forma matriciale:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & h \\ \sin\alpha & \cos\alpha & k \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Trasformazioni affini. Cambiano la forma di un oggetto.

  1. Scaling. Si ingrandice o rimpicciolisce un oggetto; di fatto scala le coordinate di un fattore p,q,r che se < 1 rimpicciolisce l'oggetto, invece se > 1 ingrandisce l'oggetto. Quindi:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} p & 0 & 0 & 0 \\ 0 & q & 0 & 0 \\ 0 & 0 & r & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

     I coefficienti si trovano tutti sulla diagonale principale.

  2. Shear. Spinge l'oggetto in una direzione parallela ad un asse.
     Direzione x:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \alpha & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

     Direzione y:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ \alpha & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Trasformazioni proiettive. Sono le trasformazioni più generali, cambiano la forma dell'oggetto.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

L'ultima riga non è necessariamente composta da soli 0 e 1.
Esempio:

$$\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & 0 \\ -1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

Questo tipo di trasformazione ha la stessa capacità operativa delle altre trsformazioni con l'aggiunta del **cambiamento forma** (es. un ellissi diventa un cerchio e v.v.).

3.6 Calcolo di una curva NURBS in un punto fissato u

3.6.1 Algoritmo

Si parte da una B-Spline:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i$$

E' possibile esprimere P_i mediante le sue **coordinate euclidee**:

$$P_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$$

A questo punto è possibile riscrivere P_i come un vettore colonna costituito da quattro componenti:

$$P_i = \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

A questo punto consideriamo un peso w_i; poichè sappiamo che moltiplicando le coordinate omogenee per un punto $\neq 0$, non avremo uno spostamento della curva, operiamo la suddetta operazione:

$$P_i = \begin{bmatrix} w_i \cdot x_i \\ w_i \cdot y_i \\ w_i \cdot z_i \\ w_i \end{bmatrix}$$

Quindi, sostituendo il tutto alla formula iniziale:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) \begin{bmatrix} w_i \cdot x_i \\ w_i \cdot y_i \\ w_i \cdot z_i \\ w_i \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot x_i \\ \sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot y_i \\ \sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot z_i \\ \sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \end{bmatrix}$$

A questo punto ci troviamo in presenza di una curva B-Spline 4-dimensionale. Per ottenere la curva NURBS ricercata, occorre dividere ogni elemento per l'ultimo:

$$C(u) = \begin{bmatrix} \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot x_i}{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i} \\ \\ \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot y_i}{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i} \\ \\ \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot z_i}{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i} \\ \\ \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i}{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i} = 1 \end{bmatrix} = \frac{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i \cdot P_i}{\sum_{i=0}^{n} N_{i,p}(u) \cdot w_i}$$

Quindischematicamente :

1. Trasformare la curva NURBS in una B-Spline 4-dimensionale
2. Applicare l'algoritmo di De Boor
3. Trasformare la curva B-Spline 4-dimensionale in una curva NURBS, dividendo ogni componente per il l'ultimo componente

3.7 Interpretazione geometrica

Quello che emerge dalla trattazione delle curve NURBS è che esse sono una semplice trasposizione delle curve B-Spline. Infatti, consideriamo un vettore control point formato da quattro componenti:

$$P_i^w = (w_i x_i, w_i y_i, w_i z_i, w_i)$$

Possiamo considerare questo punto P_i^w come un punto nello spazio 4-dimensionale e conseguentemente, $C(u)$ diventa una curva B-Spline nello spazio a quattro

dimensioni:

$$C^w(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i^w = \sum_{i=0}^{n} N_{i,p}(u) \begin{bmatrix} w_i x_i \\ w_i y_i \\ w_i z_i \\ w_i \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{n} N_{i,p}(u)(w_i x_i) \\ \sum_{i=0}^{n} N_{i,p}(u)(w_i y_i) \\ \sum_{i=0}^{n} N_{i,p}(u)(w_i z_i) \\ \sum_{i=0}^{n} N_{i,p}(u) w_i \end{bmatrix}$$

Come già affermato in precedenza, dividere le prime tre coordinate per la quarta è equivalente a proiettare un punto 4-dimensionale nel piano $w = 1$. Dal momento che la curva è convertibile in una NURBS semplicemente dividendo le prime tre coordinate per la quarta, si conclude che una curva NURBS nello spazio 3-dimensionale è effettivamente la proiezione di una curva B-Spline in uno spazio 4-dimensionale.

3.8 Proprietà importanti

Dato un insieme di $n+1$ control points $P_0, P_1, \ldots, P_n$ ad ognuno dei quali è associato un peso non negativo w_i (per esempio P_i ha peso $w_i \geq 0$), e un vettore di knot $U = u_0, u_1, \ldots, u_m$ costituito da $m+1$ knots, la curva NURBS di grado p come segue:

$$C = \sum_{i=0}^{n} R_{i,p}(u) P_i$$

e $R_{i,p}$ è definita come segue:

$$R_{i,p}(u) = \frac{N_{i,p}(u) w_i}{\sum_{j=0}^{n} N_{j,p}(u) w_i}$$

Si è scelto di utilizzare una notazione differente poichè si vuole riscrivere la definizione di una curva NURBS in una forma molto vicina ad una curva B-Spline. Nella notazione $R_{i,p}(u)$, rappresenta le funzioni base della curva NURBS. A questo punto è chiaro che le curve NURBS sono una generalizzazione delle B-Spline, per questo motivo hanno tutte le proprietà delle B-Spline. Le proprietà che seguono enfatizzano questo concetto:

1. $R_{i,p}(u)$ è una funzione razionale di grado p in u.
2. Non negatività, per ogni i e p, $R_{i,p}(u)$ è non negativa.
3. Supporto locale. $R_{i,p}$ assume valori definiti solo nell'intervallo $[u_i, u_{i+p+1}$, mentre all'esterno dell'intervallo il valore è zero. Questo implica un maggiore controllo della curva poichè non tutte le funzioni base hanno un ruolo fondamentale.
4. Partizione dell'unità. La somma di tutte le funzioni base nell'intervallo $[u_i, u_i + 1)$ è 1.

5. Se il numero dei knots è $m+1$, il grado delle funzioni base p, e il loro numero è $n+1$, allora $m = n+p+1$.

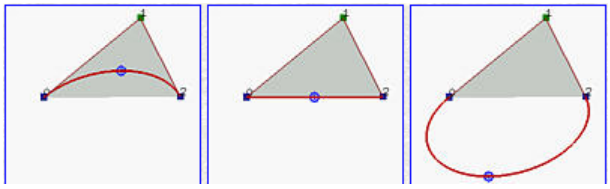
6. La funzione base $R_{i,p}(u)$ è una cuva composta da funzioni razionali di grado p con punti di giunzione ai knots nell'intervallo $[u_i, u_{i+p+1})$.

7. Al knot di molteplicità k, la funzione base $R_{i,p}$ è continua C^{p-k}. Questo implica che aumentando la molteplicità, decresce il livello di continuità e aumentando il grado aumenta la continuità.

8. Se $w_i = c$ per tutte le i, dove c è una costante diversa da zero, $R_{i,p}(u) = N_{i,p}(u)$.

A questo punto diviene necessario indicare quali sono le caratteristiche peculiari delle curve NURBS, ovvero:

- Le funzioni base sono razionali.
- Supporto locale.
- Partizione unità.
- Variation diminishing. Se la curva è contenuta in un piano, significa che nessuna linea interseca una curva NURBS più volte di quanto essa intersechi il poligono di controllo.
- Strong convex hull. La curva è contenuta nel dominio convesso generato dai suoi punti di controllo. In aggiunta, se u è contenuto all'interno dell'intervallo $[u_i, u_{i+1})$, allora $C(u)$ è nel dominio convesso dei control points $P_{i-p}, P_{i-p+1}, \ldots, P_i$. Con questo appare chiaro, che tutti i pesi devono essere non negativi. Se qualcuno di essi lo fosse, la strong convex hull o comunque la convex hull, non sarebbe presente. Prendiamo come punto di riferimento la figura che segue:



Nella prima figura (quella a sinistra), viene rappresentata una curva di grado 2, con $n = 2$ e $m = 5$. (Da notare inoltre che i primi e gli ultimi tre knots sono schiacciati). Il peso associato ai due control points alle estremità del poligono di controllo è 1 e il peso associato al knot intermedio 0.5. La curva giace nel dominio convesso.

Nella figura al centro, il peso associato al knot mediano è 0. Da momento che questo punto di controllo non influenza sulla curva, il

risultato è determinato dalle estremità. La curva giace ancora all'interno del dominio convesso.

Se il peso viene cambiato in -0.5, la curva non è più contenuta nel dominio convesso e quindi la proprietà non è rispettata.

- Invarianza per trasformazioni proiettive: la trasformazione va fatta sui punti di controllo e non sulla definizione di C(u).

3.9 Curve di Bèzier vs. Curve B-Spline vs. Curve Nurbs

Analizziamo qui di seguito quali sono i punti che accomunano le curve di Bèzier e le B-Spline e quali invece le differenze:

1. La curva è contenuta all'interno dell'involucro convesso del poligono di controllo?
   - Bèzier: SI
   - B-Spline: SI
   - NURBS: SI (a patto che il peso associato ai punti di controllo sia $w_i \geq 0$)
2. Come si lega il grado della curva rispetto al poligono di controllo?
   - Bèzier: Grado = Numero di vertici -1
   - B-Spline: Indipendente dal poligono di controllo
   - NURBS: Indipendente dal poligono di controllo
3. La curva tocca il poligono di controllo?
   - Bèzier: Si, nel primo e nell'ultimo punto
   - B-Spline: No, ma può essere fatto, collassando tanti nodi quanto il grado della curva. Operando questa routine, difatti si può farla passare per qualsiasi punto del poligono.
   - NURBS: No, ma può essere fatto, collassando tanti nodi quanto il grado della curva. Operando questa routine, difatti si può farla passare per qualsiasi punto del poligono
4. La forma della curva e del poligono sono in qualche modo correlate?
   - Bèzier: No, solo in modo molto approssimato
   - B-Spline: Si e li si può addirittura far coincidere, con particolari accorgimenti

- NURBS: Si e li si può addirittura far coincidere, con particolari accorgimenti

5. Si può avere una trasformazione affine sulla curva?
   - Bèzier: Si, operando sui vertici del poligono di controllo
   - B-Spline: Si, operando sui vertici del poligono di controllo
   - NURBS: Si, operando sui vertici del poligono di controllo, inoltre è invariante anche per trasformazioni proiettive.
6. Si può avere il controllo locale sulla curva?
   - Bèzier: NO
   - B-Spline: SI
   - NURBS: SI
7. Si possono disegnare cerchi ed ellissi?
   - Bèzier: NO
   - B-Spline: NO
   - NURBS: SI

Capitolo 4

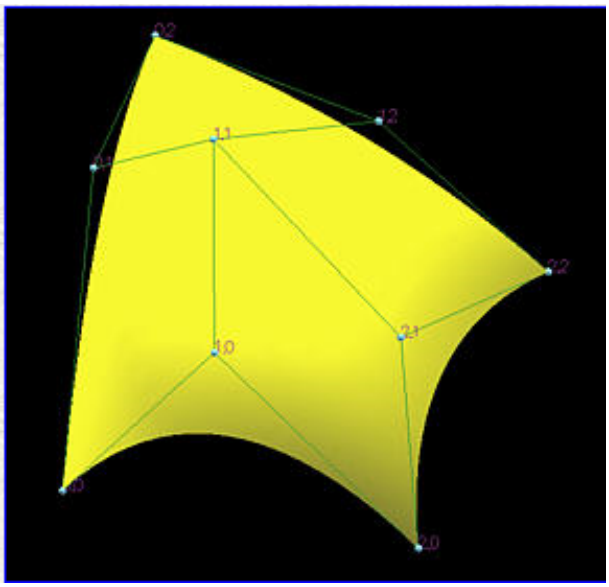
Superfici

4.1 Superfici di Bezier

Le superfici di Bezier vengono definite considerando un insieme di punti p_{ij} per $i = 0, \ldots, m$ e $j = 0, \ldots, n$. Questi punti sono distribuiti su di un reticolo che prende il nome di **control net**. La superficie di Bezier viene quindi così definita:

$$P(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_{m,i}(u) B_{n,j}(v) P_{ij}$$

Dal momento che il grado di $B_{m,i}(u)$ è m e il grado di $B_{n,j}(v)$ è n, possiamo affermare che il grado di una superficie di Bezier è (m,n). La figura che segue mostra una superficie di Bézier definita da 3 righe e 3 colonne ovvero 9 control points. La superficie ha grado (2,2):



4.1.1 Proprietà

1. $B_{m,i}B_{n,j} \geq 0$. Ovvero le funzioni base sono tutte positive: questa proprietà garantisce che la superficie sarà contenuta nel dominio convesso.

2. $\sum\sum B_{m,i}B_{n,j} = 1$. Quindi la superficie dipende unicamente dai control points e non dalla scelta delle coordinate.

3. $p(u,v)$ passa per $p(0,0)$, $p(m,0)$, $p(m,n)$, $p(0,n)$. Questo ci suggerisce che la superficie è facilmente raccordabile.

4. convex hull.

4.1.2 Algoritmo

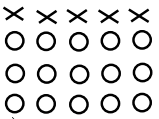
$$P(u,v) = \sum_{i=0}^{m} B_{m,i}(u) \left[\sum_{j=0}^{n} B_{n,j}(v) p_{ij} \right]$$

Invece di calcolare la superficie effettuando le operazioni in cascata, porremo:

$$q_i(v) = \sum_j B_{n,j}(v) p_{ij}$$

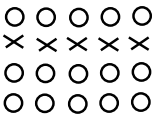
A questo punto, utilizzando l'algoritmo di De Casteljau, calcoliamo q_i per ogni i:
per $i = 0$:

$$q_0(v) = \sum_j B_{n,j}(v) p_{0,j}$$



Che rappresenta la prima riga (contraddisinta dalle x) presa in esame nella control net;
per i=1:

$$q_1(v) = \sum_j B_{n,j}(v) p_{1,j}$$



Si ottiene, quindi:

$$P(u,v) = \sum B_{m,i}(u) q_i(v)$$

E' nuovamente possibile utilizzare l'algoritmo di De Casteljau, poichè si ottiene una curva definita sui nuovi punti.

4.1.3 Curve isoparametriche

L'idea è che fissato un parametro (u), variando v ho una particolare curva sulla superficie. In questo modo è possibile conoscere i bordi della superficie ed eventualmente raccordare più superfici (tecnicamente si possono facilmente

raccordare diverse **patch** (pezze) nell'ottica di creare una figura maggiormente complessa. Un esempio calzante è il pallone da calcio costituito da una serie di rombi che in toto assumono forma sferica).

4.1.4 Curve isoparametriche speciali

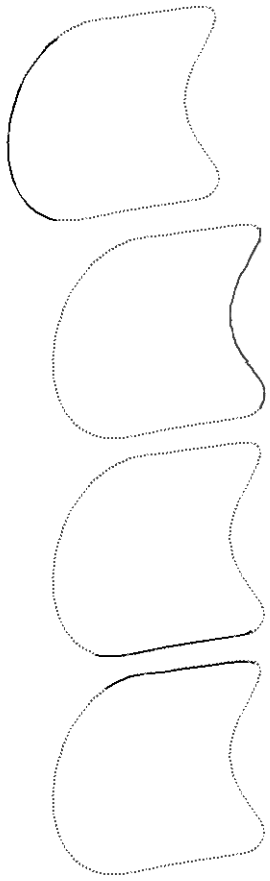
Quindi al variare dei due parametri u e v è possibile ottenere delle curve di bordo:

$$P(0,v) = \sum_j B_{n,j}(v) p_{0,j}$$

$$P(1,v) = \sum_j B_{n,j}(v) p_{m,j}$$

$$P(u,0) = \sum_j B_{n,j}(v) p_{i,0}$$

$$P(u,1) = \sum_j B_{n,j}(v) p_{i,m}$$



4.1.5 Rappresentazione tensoriale

Un oggetto bidimensionale può essere rappresentato attraverso il prodotto di due oggetti unidimensionali. In particolare la superficie di Bezier bidimensionale viene espressa attraverso due curve unidimensionali:

$$p(u,v) = [B_{m,0}(u) \dots B_{m,m}(u)] \begin{bmatrix} p_{0,0} & p_{o,n} \\ p_{m,o} & p_{m,n} \end{bmatrix} \begin{bmatrix} B_{n,0}(v) \\ B_{n,n}(v) \end{bmatrix}$$

Chiaramente, il vettore $[B_{m,0}(u) \dots B_{m,m}(u)]$ è composto delle funzioni di base su u; il vettore $\begin{bmatrix} p_{0,0} & p_{o,n} \\ p_{m,o} & p_{m,n} \end{bmatrix}$ è costituito dalla control net; il vettore trasposto $\begin{bmatrix} B_{n,0}(v) \\ B_{n,n}(v) \end{bmatrix}$ è ottenuto dal vettore delle funzioni base su v

4.2 Superfici B-Spline

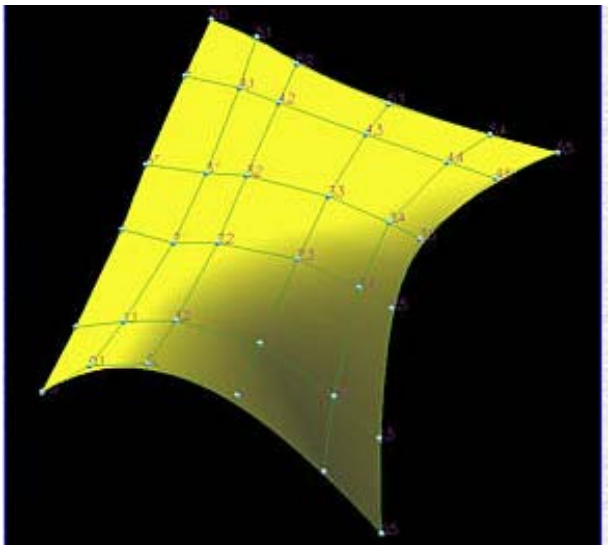
Anche in questo caso abbiamo un insieme di punti $P_{i,j}$ con $i = 0, \dots, m$ e $j = 0, \dots, n$. In aggiunta si considera $u = u_0, \dots, u_h$ e $v = v_0, \dots, v_k$. La relazione che lega i knots ai punti della superficie è espressa da:

$$h = m + p + 1 \qquad\qquad k = n + q + 1$$

Quindi definiamo una superficie B-Spline:

$$P(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} N_{i,p}(u) N_{j,q}(v) P_{ij}$$

La figura che segue, mostra una superficie B-Spline costituita da 36 control points disposti su 6 righe e 6 colonne:

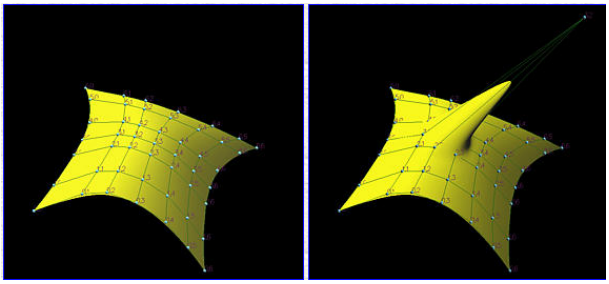


4.2.1 Proprietà

- Non negatività: $N_{i,p}(u)N_{j,q}(v)$ è non negativo per tutti gli p, q, i, j e u, v nell'intervallo tra $0, 1$.
- Partizione dell'unità: la somma di tutte le $N_{i,p}(u)N_{j,q}(v)$ è 1 per ogni u e v nell'intervallo tra $0, 1$.

- Strong Convex Hull: se (u, v) è in $[u_i, u_{i+1}) \cdot [v_i, v_{j+1})$, allora $p_{u,v}$ giace nel dominio convesso definito dai control points $p_{h,k}$ dove $i - p \leq h \leq i$ e $j - q \leq h \leq j$. Questa proprietà deriva direttamente dalla proprietà dello strong convex hall per le curve B-Spline.
  Per la direzione u, se u è in $[u_i, u_{i+1})$, allora ci sono al più $p + 1$ funzioni base diverse da zero, su questo intervallo dei knot, etichettate $N_{j,q}(v), N_{j-1,q}(v), \ldots, N_{j-q,q}(v)$. Così solo i control points dalla colonna $j - q$ alla colonna j hanno funzioni base diverse da zero nella direzione v. Combinando questi due fatti insieme, solamente i control points nell'intervallo dalla riga $i - p$ alla riga i e le colonne da $j - q$ a q hanno funzioni base diverse da zero. Dal momento che queste funzioni sono maggiori di zero e la loro somma è 1, $p_{u,v}$ giace nel dominio convesso definito da questi control points. Chiaramente, la superficie definita dal rettangolo $[u_i, u_{i+1} \cdot [v_i, v_{j+1})$ giace completamente all'interno nella stesso dominio convesso.

- Schema di modifica locale: $N_{i,p}(u)N_{j,q}(v) = 0$, se (u, v) è fuori dal rettangolo $[u_i, u_{i+1}) \cdot [v_i, v_{j+1})$. Dalla proprietà del controllo locale, sappiamo che nella direzione u, $N_{i,p}(u)$ è diversa da zero in $[u_i, u_{i+1})$ ed è zero altrove. Se il control point $p_{3,2}$ è spostato in una nuova locazione, solo l'area di vicinato subisce una deformazione e altrove la forma rimane invariata:



- Invarianza per trasformazioni affini: questo implica che per effettuare una trasformazione su una superficie B-Spline, è possibile applicare la trasformazione a tutti i punti di controllo, il risultato non cambia se appliamo la trasformazione all'equazione della superficie.

- Se $m = p, n = q$ e $U = \{0, 0, \ldots, 1, 1, \ldots, 1\}$, allora la superficie B-Spline diventa una superficie di Bézier.

4.2.2 Algoritmo

Come nel caso delle superfici di Bezier è possibile scomporre le due sommatorie, ponendo:

$$q_1 = \sum_{j=0}^{n} N_{j,q}(u)p_i, j$$

In effetti q_1 rappresenta proprio una curva B-Spline, facilmente calcolabile appliando l'algoritmo di de Boor. Quindi sostituendo otterremo:

$$P_{u,v} = \sum_{i=0}^{m} N_{i,p}(u) q_i(u)$$

E anche in questo caso otteniamo una curva B-Spline facilmente calcolabile.

4.3 Considerazioni

Il grande vantaggio delle supefici B-Spline rispetto le Bezier viene riassunto dalla proprietà di controllo locale. Infatti la variazione di un punto del poligono di controllo di una B-Spline comporta solo una variazione locale della superficie.

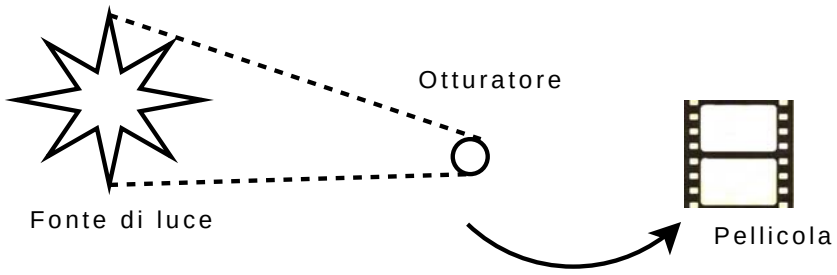
Capitolo 5

Codifica Digitale

5.1 La macchina fotografica

Si analizzeranno le due tipologie di macchine fotografiche: **analogica** e **digitale**.

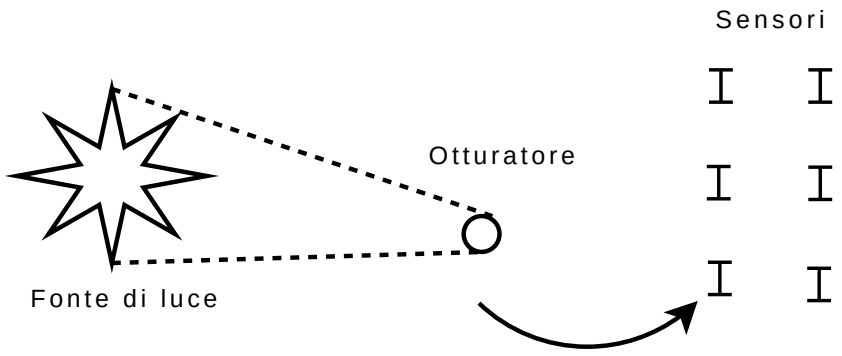
5.1.1 La macchina fotografica analogica



Il segnale, considerato continuo, passa attraverso l'otturatore e impressiona la pellicola. Le operazioni sulla immagine sono facilmente eseguibili.

5.1.2 La macchina fotografica digitale

I sensori CMOS, capaci di ditinguere i tre colori RGB, renderanno possibile l'associazione di un colore (espresso per ogni elemento della terna) rappresentante l'intensità di luce che ha colpito il sensore presente in quella specifica area. La rappresentazione della immagine segue una matrice **discreta** di punti. Consideriamo un immagine 1920 x 1080; ebbene essa occupa circa 2 milioni di pixel. L'obiettivo è la corretta gestione della matrice di punti.



Esempio - Trasferimento immagine

Supponiamo di avere una immagine a 6Mpixel e una connessione internet a 50Kbaud. Quanto tempo occorrerà per inviare scaricare l'immagine dal web?

- L'occupazione è data da: 6Mpixel ·3 (componenti) = 18Mb
- Il data transfer è di 6000 Byte/sec.
- Il tempo necessario è calcolato: $18.000.000 : 6.000 = 3.000$ sec. ovvero 50 minuti

Avendo una connessione a banda larga, capace di trasmettere a 750Kbit/sec. occorrerebbero solo 22 secondi.

5.2 Algoritmi di compressione

Esistono vari metodi per comprimere e gestire al meglio la matrice di punti rappresentante l'immagine; essi sono più o meno validi e performanti:

Metodo accetta

Esso è il metodo più semplice da implementare e permette di comprimere l'immagine con un rapporto di compressione molto alto. Purtroppo questo metodo elimina molta informazione poichè opera eliminando le righe e le colonne pari (oppure dispari) dalla matrice di riferimento. Chiaramente molti pixel importanti vengono totalmente eliminati. Il metodo equivale semplicemente, a degradare la risoluzione di una macchina fotografica, il che è, ovviamente, improponibile: occorre trovare un metodo alternativo.

Rappresentazione parametrica

Sfruttando delle approssimazioni è possibile rappresentare alcune immagini sottoforma di funzioni matematiche. Ad esempio: supponiamo di avere un'im-

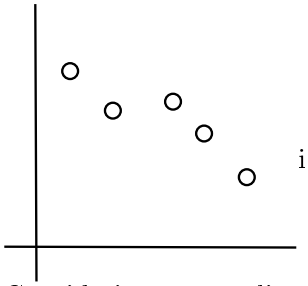
magine intrinsicamente semplice come un fascio di luce obliquo. Ebbene una possibile rappresentazione potrebbe essere fatta sfruttando l'equazione parametrica di una retta:

$$s = ax + b$$

L'immagine dipenderà solo da due parametri, tuttavia in questo modo, possono essere rappresentati solo segnali molto semplici e quindi non realistici.Occorre passare da una rappresentazione nello spazio fisico ad una rappresentazione funzionale.

Rappresentazione funzionale

Soffermiamoci quindi su una singola riga della matrice. Indichiamo su un grafico i vari punti della riga in questione:



Dobbiamo trovare una funzione che passi attraverso i punti del grafico.

Consideriamo un polinomio di grado n comunque non maggiore del numero di pixel -1:

$$f(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_{n-1}x^{n-1}$$

Ma i polinomi non funzionano, il grado troppo alto causa forme molto oscillanti.

5.2.1 Combinazione lineare di un certo insieme di funzioni

Invece di utilizzare il polinomio sopraindicato, utilizzeremo alcune funzioni note addizionate tra loro e moltiplicate per alcuni parametri (rappresentazione lineare):

$$f(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + \ldots$$

Essendo note le $f(x)$, ovvero l'intensità associata al pixel, essendo già fissate le φ (funzioni particolari note [ad esempio funzioni coseno]), ed essendo nota la posizione x, l'obiettivo è calcolare i coefficienti. Ad esempio nel primo pixel l'intensità si calcola:

$$f_1 = a_0\varphi_0(x_1) + a_1\varphi_1(x_1) + a_2\varphi_2(x_1) + \ldots + a_{n-1}\varphi_{n-1}(x_1)$$

Nel secondo:

$$f_2 = a_0\varphi_0(x_2) + a_1\varphi_1(x_2) + a_2\varphi_2(x_2) + \ldots + a_{n-1}\varphi_{n-1}(x_2)$$

E così via:

$$f_n = a_0\varphi_0(x_n) + a_1\varphi_1(x_n) + a_2\varphi_2(x_n) + \ldots + a_{n-1}\varphi_{n-1}(x_n)$$

Si ottiene un sistema di equazioni lineari in n incognite che risolto fornisce la soluzione (sotto alcune ipotesi sempre verificate nelle applicazioni; in particolare i nodi $x_1 \ldots x_n$ devono essere distinti).

5.2.2 Scelta della funzione φ

La φ viene scelta considerando la velocità dell'algoritmo e accuratezza della rappresentazione. Operando una scelta sbagliata si rischia di non avere una rappresentazione corretta e di rendere il processo di calcolo molto complesso (dell'ordine di $O(n!)$).

5.3 Trasformata veloce di Fourier

L'utilizzo della trasformata di Fourier riduce la complessità a $O(N \cdot \log_2 N)$. Le macchine fotografiche digitali implementano l'algoritmo di compressione attraverso un circuito integrato, ovvero ottengono il risultato voluto senza bisogno di una architettura di calcolo; questo comporta la possibilità di miniaturizzare le componenti e di rendere il calcolo **real-time**.

5.3.1 Caso trigonometrico

L'idea è quella di associare alle φ le funzioni trigonometriche sin e cos, ottenendo:

$$T(x) = a_0 + a_1\cos(2\pi x) + a_2\cos(4\pi x) + a_3\cos(6\pi x) + \ldots + b_1\sin(2\pi x) +$$

$$b_2\sin(4\pi x) + b_3\sin(6\pi x) + \ldots$$

Calcolando $T(0)$ e $T(1)$:

$$T(0) = a_0 + a_1 + a_2 + a_3 + \ldots + 0$$

$$T(1) = a_0 + a_1 + a_2 + a_3 + \ldots + 0$$

abbiamo lo stesso valore.

Ricorda:
$\sin 2\pi = 0$
$\cos 2\pi = 1$

Calcolando $T(x+1)$:

$$T(x+1) = a_0 + a_1\cos(2\pi x) + a_2\cos(4\pi x) = T(x)$$

Il periodo del polinomio è 1, ciò significa che l'immagine si ripete sia a destra che a sinistra dell'intervallo di riferimento $[0, 1]$:

5.4 Teorema di Parceval

Il teorema di Parceval rende possibile la compressione di un segnale. Per ogni componente del segnale si calcola la sua energia come l'integrale della funzione elevata al quadrato ovvero:

$$\int f^2(x)dx = \sum a_k^2 + b_k^2$$

5.5 Sintesi e Analisi di Fourier

La forumula che esprime la sintesi di Fourier è la seguente:

$$T_N(x_k) = \sum_{j=0}^{N} (a_j \cos 2j\pi x_k + b_j \sin 2j\pi x_k)$$

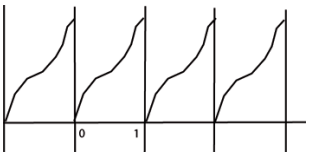
Questo è un polinomio compatto di grado N, vero per ogni pixel che, noti a_j e b_j ci permette di esprimere la sintesi di Fourier. Il nostro obiettivo è quindi calcolare a_j e b_j.
I pixel sono tutti equidistanti e considerato l'intervallo $[0, 1]$:
$x_k = \frac{k}{2N}$ $\qquad 0 \leq k \leq 2N.$
Inoltre il valore del polinomio nel punto finale è uguale al valore nel punto iniziale per la periodicità della funzione. Calcoliamo quindi quante sono queste equazioni:

$$T_n(x_0) = f_0 = \sum_{j=0}^{N} \left(a_j \cos \frac{2j\pi}{2N} \cdot 0 + b_j \sin \frac{2j\pi}{2N} \cdot 0 \right)$$

$$f_1 = \sum_{j=0}^{N} \left(a_j \cos \frac{2j\pi}{2N} \cdot 1 + b_j \sin \frac{2j\pi}{2N} \cdot 1 \right)$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$

$$f_{2N-1} = \sum_{j=0}^{N} \left(a_j \cos \frac{2j\pi(2N-1)}{2N} + b_j \sin \frac{2j\pi(2N-1)}{2N} \right)$$



Quindi il numero di equazione è 2N; il numero di a_j è $N+1$ così come il numero di b_j è $N+1$.

5.5.1 Eliminare alcune incognite

Il numero di incognite risultante è $2N+2$, numero che rende incompatibile il sistema. Ma analizzando f_0 e f_N:

$$f_0 = b_0 \cdot \frac{\cos 2j\pi x}{2N} = b_0 \sin 0 = 0$$

$$f_N = b_N \cdot \frac{\sin 2N\pi k}{2N} = b_N \sin \pi k = 0$$

Il numero di incognite si riduce quindi a $2N$.

5.5.2 Risoluzione del sistema

Una importante proprietà del sistema trigonometrico è la sua ortagonalità:

$$\int_0^1 \sin 2k\pi x \ \ \sin 2e\pi x \ \ dx =$$

$$= \int_0^1 \cos 2k\pi x \ \ \cos 2e\pi x \ \ dx = \begin{cases} 0 & k \neq e \\ \\ \neq 0 & k = e \end{cases}$$

$$\int_0^1 \sin 2k\pi x \ \ \cos 2e\pi x \ \ dx = 0$$

Tale proprietà permette di ottenere la risoluzione del sistema di equazioni (e quindi i coefficienti a_j e b_j) analiticamente, ovvero senza costosi metodi numerici:

$$\begin{cases} a_j = \frac{1}{2N} \sum_{k=0}^{2N-1} f_k \cdot \cos \frac{2jk\pi}{2N} \\ \\ b_j = \frac{1}{2N} \sum_{k=0}^{2N-1} f_k \cdot \sin \frac{2jk\pi}{2N} \end{cases}$$

Il numero di operazioni che occorre per calcolare a_j e b_j è dato dalla somma di:

- $2N$ addizioni
- $2N$ moltiplicazioni
- 1

per una sola componente. In totale si hanno $8N^2 + 2N$ operazioni.
Lavorando con N molto alti, il $2N$ può essere eliminato, abbiamo così un ordine di grandezza pari a $O(N^2)$.

5.5.3 Riepilogo numeri complessi

Un numero complesso è rappresentato da una parte reale e da una parte immaginaria:

$$(a + ib)$$

Proprietà importanti

i è l'unità immaginaria:
$i = \sqrt{-1}$ $\qquad$ $i^2 = -1$

La somma di due numeri complessi viene così definita:

$$(a + ib) + (c + id) = a + ib + c + id = (a + c) + i(b + d)$$

La moltiplicazione:

$$a + ib) \cdot (c + id) = ac + iad + ibc + i^2bd =$$

$$= ac + iad + ibc - bd =$$

$$= (ac - bd) + i(ad + bc)$$

Esponenziale:

$e^{ix} = \cos x + i \sin x$ $\qquad$ $e^{-ix} = \cos(-x) + i \sin(-x) = \cos x - i \sin x$

Formule di Eulero:

$e^{ix} + e^{-ix} = 2 \cos x$ $\qquad$ $\cos x = \frac{e^{ix} + e^{-ix}}{2}$

$e^{ix} - e^{-ix} = 2i \sin x$ $\qquad$ $\sin x = \frac{e^{ix} + e^{-ix}}{2i}$

Infine:
$\frac{1}{i} = -i$ $\qquad$ $\frac{1}{i} \cdot \frac{i}{i} = \frac{i}{-1} = -i$

5.5.4 Aumentare la compattezza della $T_N(x)$

A questo punto, partendo dalla $T_N(x)$:

$$T_n(x) = \sum_{j=0}^{N} (a_j \cos 2j\pi x + b_j \sin 2j\pi x)$$

sostituendo le formule di Eulero, otteniamo:

$$T_n(x) = \sum_{j=0}^{N} \left(a_j \frac{e^{i2j\pi x} + e^{i2j\pi x}}{2} + b_j \frac{e^{i2j\pi x} - e^{-i2j\pi x}}{2} \right)$$

Metto in evidenza e^{i} e e^{-i}, ottenendo

$$T_n(x) = \sum_{i=0}^{N} e^{i2j\pi x}\left(\frac{a_j - ib_j}{2}\right) + e^{-i2j\pi x}\left(\frac{a_j + b_j}{2}\right) =$$

$$= \sum_{j=-N}^{N} c_j \cdot e^{i2j\pi x} \qquad (1)$$

L'equazione (1) è detta **sintesi di Fourier**. Occorre notare che la sommatoria va da N (positivo) a $-N$ (negativo).
A questo punto:

$$c_j = \begin{cases} \frac{a_j - ib_j}{2} & 0 < j < N \\ \frac{a_j + ib_j}{2} & -N < j < 0 \\ \frac{a_N}{2} & j = N \quad o \quad j = -N \\ a_0 & j = 0 \end{cases}$$

L'analisi di Fourier è data da:

$$C_j = \frac{1}{N}\sum_{k=0}^{N-1} f_k e^{\frac{-i2j\pi k}{N}}$$

Dove C_j rappresenta il valore dei coefficienti per ogni valore di RGB. L'utilizzo delle tecniche citate può essere riassunto secondo lo schema:

$$f_k \Rightarrow \frac{Sintesi\ \ di\ \ Fourier}{Anti\ \ Trasformata} \Rightarrow C_j \Rightarrow \frac{Analisi\ \ di\ \ Fourier}{Trasformata} \Rightarrow f_k$$

L'obiettivo rimane sempre calcolare C_j, ma è possibile lavorare su N punti alla volta anzichè 2N:

$$C_j = \frac{1}{N}\sum_{k=0}^{\frac{N}{2}-1} f_k\, e^{\frac{-i2j\pi k}{N}} + f_{k+\frac{N}{2}}\, e^{\frac{-i2j(k+\frac{N}{2})\pi}{N}}$$

Poniamo $w = e^{\frac{-i2\pi}{N}}$:

$$C_j = \frac{1}{N}\sum_{k=0}^{\frac{N}{2}-1} f_k w^{jk} + f_{k+\frac{N}{2}} w^{jk} \cdot w^{j\frac{N}{2}}$$

Considerando che il termine $w^{j\frac{N}{2}}$ può essere scritto come:

$$w^{j\frac{N}{2}} = (e^{\frac{-i2\pi}{N}})^{j\frac{N}{2}} = e^{\frac{-i2\pi j\frac{N}{2}}{N}} =$$

$$e^{\frac{-i2\pi j}{2}} = e^{-i\pi j}$$

Ricordiamo che $e^{ix} = \cos x + i \sin x$, allora:

$$e^{-i\pi j} = \cos(-\pi j) + (i)\sin(-\pi j) = \cos(\pi j) - i\sin(\pi j) = \cos(\pi j).$$

$$cos(\pi j) = \begin{cases} 1 \qquad per \quad j \quad pari \\ -1 \qquad per \quad j \quad dispari \end{cases}$$

Quindi possiamo riscrivere la nostra C_j come:

$$C_j = \frac{1}{N} \sum_{k=0}^{\frac{N}{2}-1} f_k w^{jk} + f_{k+\frac{N}{2}} w^{jk}(-1)^j$$

A questo punto abbiamo due componenti diverse a seconda che j sia pari o dispari:

$$\begin{cases} C_{2m_1} = \frac{1}{N} \sum_{k=0}^{\frac{N}{2}-1} (f_k + f_{k+\frac{N}{2}}) w^{2m_1 k} \\ \\ C_{2m_1+1} = \frac{1}{N} \sum_{k=0}^{\frac{N}{2}-1} (f_k - f_{k+\frac{N}{2}}) w^{(2m_1+1)k} \end{cases}$$

Considerando ora che:

$$\frac{1}{N} = \frac{1}{2} \cdot \frac{1}{\frac{N}{2}} = \frac{1}{2} \cdot \frac{2}{N} = \frac{1}{2}$$

Ponendo $w^2 = w_1$ e $f_k + f_{k+\frac{N}{2}} = f_k^{(1)}$, otteniamo:

$$\begin{cases} C_{2m_1} = \frac{1}{2} \sum_{k=0}^{\frac{N}{2}-1} f_k^{(1)} \cdot w_1^{m_1 k} \\ \\ C_{2m_1+1} = \frac{1}{2} \sum_{k=0}^{\frac{N}{2}-1} f_{k+\frac{N}{2}}^{(1)} \cdot w_1^{m_1 k} \end{cases}$$

Praticamente:

$$\left.\begin{matrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_7 \end{matrix}\right\} \quad \left.\begin{matrix} f_0^{(1)} = f_0 + f_4 \\ f_1^{(1)} = f_1 + f_5 \\ f_2^{(1)} = f_2 + f_6 \\ f_3^{(1)} = f_3 + f_7 \end{matrix}\right\} \frac{N}{2} termini \quad \left.\begin{matrix} f_4^{(1)} = f_0 - f_4 \\ f_5^{(1)} = (f_1 - f_5)w^1 \\ f_6^{(1)} = (f_2 - f_6)w^2 \\ f_7^{(1)} = (f_3 - f_7)w^3 \end{matrix}\right\} \frac{N}{2} termini$$

Vantaggi di questa rappresentazione

La formula di partenza aveva una complessità asintotica pari ad $O(N^2)$ poichè le operazioni necessarie erano proprio N^2. Utilizzando questa rappresentazione si ha:

1. Coefficienti pari (C_{2m_1}):

   - $f_k^{(1)} = 1 \cdot \frac{N}{2}$
   - $w \cdot f_k = \frac{N}{2}$
   - somma totale $= \frac{N}{2}$

   per un totale di $\frac{N}{2} + \frac{N}{2} + \frac{N}{2} = \frac{3}{2}N$ operazioni per determinare un singolo coefficiente

2. Coefficienti dispari($C_{2_m 1+1}$):

   - $f_k^{(1)}$ (somme e prodotto) $= 2 \cdot \frac{N}{2}$
   - $w \cdot f_k = \frac{N}{2}$
   - somma totale $= \frac{N}{2}$

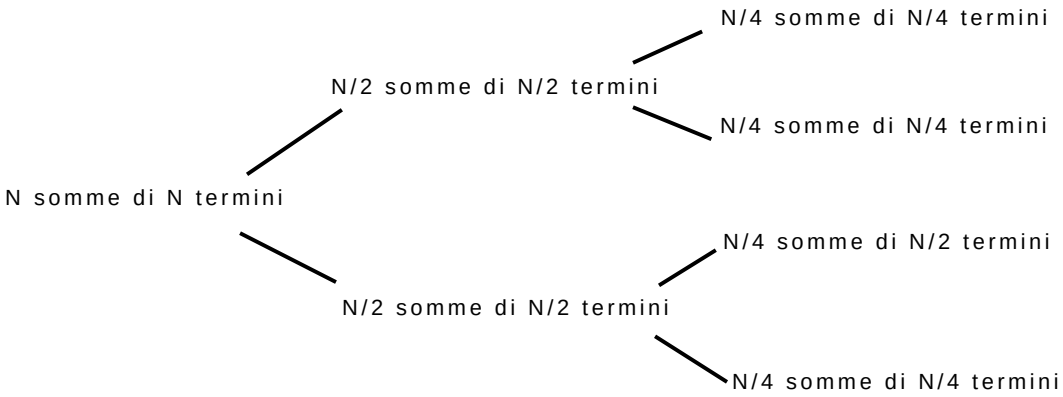
   per un totale di $2 \cdot \frac{N}{2} + \frac{N}{2} + \frac{N}{2} = 2N$ operazioni per ogni singolo coefficiente

Per tutti i coefficienti:

$$\left.\begin{matrix} \frac{3}{2}N \cdot \frac{N}{2} = \frac{3}{4}N^2 \\ 2N \cdot \frac{N}{2} = N^2 \end{matrix}\right\} \frac{7}{4}N^2$$

Quindi in sostanza si è guadagnato poco. Tuttavia analizzando bene la formula di partenza, ogni singolo coefficiente è somma di N termini (gli f_k) moltiplicati per gli esponenziali; nella formula di arrivo il singolo coefficiente è somma di $\frac{N}{2}$ numeri (gli $f_k^{(1)}$) moltiplicati per esponenziali. Questo ci suggerisce che le due formule sono compatibili nella forma (salvo il fatto che la prima formula è su N termini e la seconda su $\frac{N}{2}$). Quindi è possibile ulteriormente suddividere le due formule (le nuove saranno su $\frac{N}{4}$ termini).

Graficamente:



L'idea è quella di iterare questo processo fino a che la somma è composta da un unico termine che è il coefficiente di Fourier.

5.5.5 Esempio pratico

In questo esempio supponiamo di avere:

$$\begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \end{pmatrix} \Rightarrow \begin{pmatrix} f_0 + f_4 = f_0^{(1)} \\ f_1 + f_5 = f_1^{(1)} \\ f_2 + f_6 = f_2^{(1)} \\ f_3 + f_7 = f_3^{(1)} \\ \cdots\cdots\cdots \\ f_0 - f_4 = f_4^{(1)} \\ f_1 - f_5 \cdot w = f_5^{(1)} \\ f_2 - f_6 \cdot w^2 = f_6^{(1)} \\ f_3 - f_7 \cdot w^3 = f_7^{(1)} \end{pmatrix} \Rightarrow \begin{pmatrix} f_0^{(1)} + f_2^{(1)} = f_0^{(2)} \\ f_1^{(1)} + f_3^{(1)} = f_1^{(2)} \\ \cdots\cdots\cdots \\ f_0^{(1)} - f_2^{(1)} = f_2^{(2)} \\ f_1^{(1)} - f_3 \cdot w_1 = f_3^{(2)} \\ \cdots\cdots\cdots \\ f_4^{(1)} + f_6^{(1)} = f_4^{(2)} \\ f_5^{(1)} + f_7^{(1)} = f_5^{(2)} \\ \cdots\cdots\cdots \\ f_4^{(1)} - f_6^{(1)} = f_6^{(2)} \\ f_5^{(1)} - f_7^{(1)} \cdot w_1 = f_7^{(2)} \end{pmatrix} \Rightarrow \begin{pmatrix} f_0^{(2)} + f_1^{(2)} = c \\ \cdots\cdots\cdots \\ f_0^{(2)} - f_1^{(2)} = c \\ \cdots\cdots\cdots \\ f_2^{(2)} + f_3^{(2)} = c \\ \cdots\cdots\cdots \\ f_2^{(2)} - f_3^{(2)} = c \\ \cdots\cdots\cdots \\ f_4^{(2)} + f_5^{(2)} = c \\ \cdots\cdots\cdots \\ f_4^{(2)} - f_5^{(2)} = c \\ \cdots\cdots\cdots \\ f_6^{(2)} + f_7^{(2)} = c \\ \cdots\cdots\cdots \\ f_6^{(2)} - f_7^{(2)} = c \end{pmatrix}$$

Utilizzando questo metodo si riesce facilmente ad ottenere una complessità asintotica pari ad $O(n \cdot \log N)$.

5.5.6 Considerazioni

Nella formulazione di questo metodo di sviluppo abbiamo implicitamente trascurato i $\frac{1}{2}$, quindi occorre riscalarli facendo $\frac{c}{\frac{1}{2}}$. Inoltre abbiamo anche perso l'ordine dei coefficienti. Per ovviare al problema ricostuiamo l'ordine facendo una riflessione a 180° dei bit. Ad esempio:

$$\begin{pmatrix} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{pmatrix} \Rightarrow \begin{pmatrix} 000 \\ 100 \\ 010 \\ 110 \\ 001 \\ 101 \\ 011 \\ 111 \end{pmatrix}$$

Quindi l'ordine è:

$$\begin{pmatrix} C_0 \\ C_4 \\ C_2 \\ C_6 \\ C_1 \ C_5 \\ C_3 \\ C_7 \end{pmatrix}$$

5.6 Trasformata inversa di Fourier

Per effettuare la trasformata inversa di Fourier, è possibile adottare due strategie:

- Algoritmo ad-hoc
- Complesso coniugato della trasformata diretta

Analizziamo la prima strategia, ovvero la progettazione di un algoritmo ad-hoc. Ebbene:

$$IFT(x) = \sum_j (a_j + ib_j) \cdot (\cos + i \sin) = \sum_j a_j \cos + ia_j \sin + ib_j \cos - b_j \sin$$

La seconda strategia può essere descritta mediate pochi passi:

1. Si utilizza la stessa parte reale, ma si cambia di segno alla parte immaginaria
2. Applichiamo la trasformata di Fourier normalmente

3. Riapplichiamo il complesso coniugato (ovvero passo 1)

Ovvero:

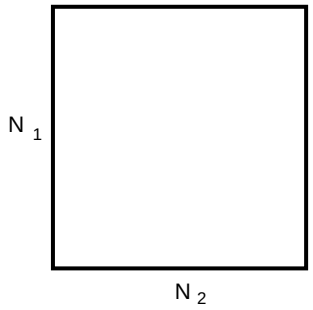
$$FT(\overline{x}) = \sum_j (a_j - ib_j) \cdot (\cos - i \sin) = \sum_j a_j \cos - ia_j \sin - ib_j \cos - b_j \sin$$

$$\overline{FT}(\overline{x}) = \sum_j (a_j + ib_j) \cdot (\cos + i \sin) = \sum_j a_j \cos + ia_j \sin + ib_j \cos - b_j \sin$$

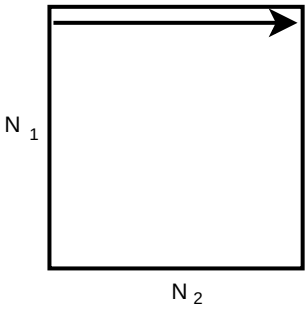
Le due strategie proposte sono perfettamente identiche!

5.7 Trasformata Bi-dimensionale

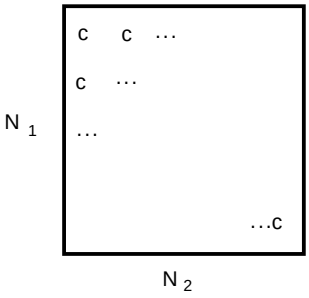
Consideriamo la matrice rappresentante la nostra immagine:



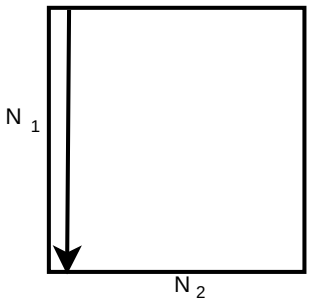
Applichiamo, come di consueto, la trasformata di Fourier per la prima riga:



Procediamo per tutte le righe ottenendo una nuova matrice, costituita dai coefficienti calcolati:



A questo punto prendiamo in esame la prima colonna e applichiamo ad essa la trasformata di Fourier:



Iteriamo la trasformata per tutte le colonne della matrice coefficienti. L'ordine di esecuzione (righe, colonne oppure colonne, righe) non influisce sul risultato finale.

5.8 Trasformata inversa di Fourier in 2-dimensioni

Nel caso della 1-dimensione si ha una dipendeza univoca, ovvero:

$$f(x) = a_0 \cdot \varphi_0(x) + a_1 \cdot \varphi_0(x) + \ldots + a_{n-1}\varphi_{n-1}(x)$$

mentre nelle due dimensioni si considerano separatamente gli apporti dati dalle due componenti, ovvero:

- $\varphi_0^x(x), \varphi_1^x(x), \ldots, \varphi_{n-1}^x(x)$. Asse di riferimento x.
- $\varphi_0^y(y), \varphi_1^y(y), \ldots, \varphi_{n-1}^y(y)$. Asse di riferimento y

La relazione funzionale deve, quindi, considerare sia la componente x che la componente y. Si applica il prodotto cartesiano delle due basi:

$$\varphi_0^x \cdot \varphi_0^y + \varphi_0^x \cdot \varphi_1^y \ldots$$

$$\varphi_1^x \cdot \varphi_0^y + \varphi_1^x \cdot \varphi_1^y \ldots$$

Quindi:

$$f(x,y) = a_{0,0}\varphi_0^x(x) \cdot \varphi_0^y(y) + a_{0,1}\varphi_0^x(x) \cdot \varphi_1^y(y) + a_{0,2}\varphi_0^x(x) \cdot \varphi_2^y(y) +$$

$$+ \ldots + a_{0,N-1}\varphi_0^x(x) \cdot \varphi_{N-1}^y(y) + a_{1,0}\varphi_1^x(x) \cdot \varphi_0^y(y) + a_{1,1}\varphi_1^x(x) \cdot \varphi_1^y(y) +$$

$$+ \ldots + a_{1,N-1}\varphi_1^x(x) \cdot \varphi_{N-1}^y(y) + \ldots\ldots + \ldots\ldots + \ldots\ldots + \ldots\ldots + \ldots\ldots +$$

$$+ a_{N-1,0}\varphi_{N-1}^x(x) \cdot \varphi_0^y(y) + a_{N-1,1}\varphi_{N-1}^x(x) \cdot \varphi_1^y(y) + \ldots +$$

$$a_{N-1,N-1}\varphi_{N-1}^x(x) \cdot \varphi_{N-1}^y(y).$$

Questa rappresentazione è ancora lineare e anche separabile, è possibile individuare i singoli prodotti delle componenti quindi è possibile separare il contributo da x dal contributo di y.

5.9 FFT con potenze diverse da 2

L'algoritmo FFT richiede che i dati siano potenza di 2 per poter operare. Ma se questo requisito non è soddisfatto?
E' possibile ricorrere alla tecnica dello **zero padding**, ovvero si raggiungere il multiplo di 2 più vicino inserendo degli 0. Ad esempio supponiamo di avere 50 dati, per raggiungere il multiplo di 2 più vicino occorre inserire 14 0, ottenendo così un totale di 64.

In letteratura, comunque, esistono altri algoritmi che si basano su potenze diverse dal 2. Ad esempio considerando potenze di 3 si potrà, sostanzialmente, dividere lo spazio di azione in 3 parti, sfruttando idealmente lo stesso principio visto in precedenza. Questi algoritmi sono però poco utuli per il real-time e, generalmente, si implementano via software. Un esempio particolarmente significativo è dato dall'algoritmo di compressione JPEG che opera considerando blocchi di 8x8 pixel. In particolare l'algoritmo utilizza solo termini coseno.

Capitolo 6

Simulazione

6.1 Simulazione

Lo sviluppo e la ricerca in campo scientifico-tecnologico di questi ultimi anni hanno portato l'uomo a considerare l'opportunità di progetti ambiziosi e di portata infinitamente grande dal punto di vista delle difficoltà tecnice e di progettazione. Esempi intuitivi e più comuni sono la costruzione di grandi edifici come stadi e grattacieli, missioni esplorative spaziali o ancora sistemi di comunicazione su larga scala come la rete internet.

Figura 6.1: Simulazione del ponte dello stretto di Messina

Ma esistono situazioni più sottili o progetti più sofisticati come la nuova chirurgia basata sul laser, le ricerche di massa su campioni di microparticelle, gli studi sugli ecosistemi a livello micro e macroscopico, oppure l'ortopedia con le nuove protesi. In ultima analisi esistono progetti specifici i cui benefici fanno parte della nostra vita quotidiana quali la disposizione ottima degli airbag

relativamente all'anatomia delle auto, la definizione della posizione perfetta delle uscite di sicurezza in un locale pubblico o gli studi di viabilità in relazione alla posizione e ad i periodi dei semafori.

Tutti questi lavori hanno in comune un unico denominatore: la necessità di effettuare un "primo" tentativo che deve essere il più vicino possibile alla soluzione finale ottima. In altre parole pensate ad un chirurgo che, operando con il laser sulla retina dei suoi pazienti, prima di determinare l'esatta intensità del raggio di luce debba effettuare una serie di prove con il rischio di rendere cieche alcune persone. Altro esempio può essere quello di un cinema che organizza le sue uscite di sicurezza senza badare al fatto che i percorsi previsti non sono attraversabili dai diversamente abili... Non è pensabile che il cinema si doti di un sistema di evaquazione adattabile solo dopo che è andato a fuoco un paio di volte e magari coinvolgendo nell'incendio persone in sedia a rotelle. La risposta a questa necessità è stata fornita dall'informatica sottoforma di quelle che vengono definite simulazioni.

6.2 Cosa vuol dire simulazione?

Un'applicazione di simulazione ha come obiettivo quello di far risultare all'utente l'esperienza simulata quanto più simile all'esperienza reale. Simulare è quindi un processo che coinvolge l'utente o l'analista in prima persona come supervisore di un sistema che si sviluppa e modifica nel tempo; il sistema ha delle caratteristiche che ricordano e riproducono il più fedelmente possibile quelle di una situazione di vita che l'utente potrebbe dover affrontare o che ha già affrontato. I principali vantaggi dell'utilizzo della simulazione sono quelli di:

- valutare: attraverso la simulazione l'utente/analista è in grado di trarre le sue considerazioni e conclusioni riguardo il progetto simulato.
- prevedere: gli effetti del progetto specifico possono essere riscontrati e valutati prima che il progetto stesso venga intrapreso. Questa è forse la caratteristica più importante della simulazione, molto spesso infatti accade che i risultati della simulazioni siano diversi dalle aspettative indotte dall'intuizione: si è sperimentato per esempi attraverso la simulazione che il posizionamento di un pilastro a breve distanza dall'uscita di sicurezza di un cinema sia vantaggioso per la viabilità in caso di incendi, fattore che ad un primo esame sembrerebbe invece dannoso.
- "far risparmiare": i costi, il superamento di difficoltà tecnice, il rischio di errore o la sperimentazione graduale di un lavoro possono essere completamente risparmiati attraverso una simulazione adeguata: una compagnia telefonica americana ha risparmiato svariati milioni di dollari simulando le conseguenze di un nuovo piano tariffario

sperimentale per cellulari: la compagnia notò che attraverso questa nuova tariffa il numero di nuovi clienti stimati non sarebbe stato sufficiente a coprire il capitale perso per gli sconti ai clienti della nuova tariffa.

Per comprendere a fondo in cosa consiste la simulazione vediamo ora le applicazioni più riuscite della stessa:

- Sicurezza: si considerino per esempio i cosiddetti crash test, prima erano dei d ispendiosissimi scontri reali tra macchine vere contro muri appositamente progettati, adesso sono una semplice immagine che si muove su un computer. Questa immagine simula perfettamente, cioè riproduce con un ottimo livello di fedeltà, quello che succederebbe se facessimo schiantare una macchina contro un muro reale.

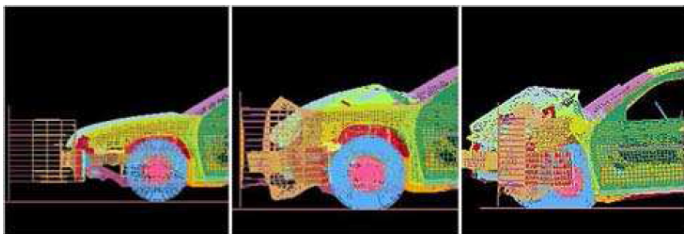


Figura 6.2: Simulazione Crash-Test

- Efficienza: possiamo pensare agli studi di aerodinamica. Immaginare di dover produrre dieci diversi tipi di alettoni per macchine da corsa, testarli uno per uno e poi sceglierne solo quello che risulta più adatto degli altri è evidentemente una procedura dispendiosa e con un altissimo livello di spreco. Ad oggi gli alettoni, le chiglie delle navi, i profili delle auto e delle ali degli areoplani, gli impennaggi delle frecce da gara e molti altri elementi degli attrezzi sportivi sono tutti prima progettati tramite la simulazione e poi realizzati fisicamente.

- Apprendimento: software simulatori di volo sono diventati ormai parte integrante nello studio di un giovane apprendista pilota, le meccaniche e i procedimenti di decollo, atteraggio e manovre vengono perfettamente assimilate attraverso l'uso del simulatore.

- Intrattenimento ludico: settore sicuramente più giovane è alla ricerca di simulazioni il più realistico possibile a scopo di intrattenimento dell'utente, l'esempio più eclatante è quello di "Second Life", un simulatore di vita reale disponibile on-line. La maggior parte dei videogiochi sono ottimi esempi di questa applicazione.

Figura 6.3: Second Life

- Mercato/economia: una determinata azienda potrebbe testare la riuscita del suo nuovo prodotto sul mercato attraverso una simulazione computerizzata. Abbiamo visto in precedenza l'esempio della compagnia telefonica americana.
- Situazioni particolari: le camere per la simulazione di condizioni a gravità 0 hanno fatto straparlare i nostri programmi televisivi per la divulgazione scientifica ed il film Apollo 13 ha egregiamente pubblicizzato l'esistenza di stanza per la simulazione dell'attività all'interno di shuttle e basi orbitanti. Andando avanti possiamo pensare all'esplosione che le biotecnologie hanno avuto in questi anni elaborando applicazioni in grado di testare i dettagli di una protesi interna a partire dalla semplice immagine di una radiografia.

In conclusione la simulazione è un procedimento che ha applicazioni nei più svariati campi e nelle più diverse discipline e che si è fatta strumento fondamentale nel progresso scientifico-tecnologico della nostrà società. Oggi possiamo immaginare di simulare praticamente qualuque cosa, i più fantasiosi racconti di fantascienza parlano di robot antropomorfi in grado di simulare le emozioni umane; ovviamente la contraddizione giace nella definizione stessa di simulazione: un processo che riproduce con un certo grado di fedeltà un sistema o una situazione della vita reale. Questo è dovuto al fatto che per effettuare una simulazione vengono definite e codificate una serie di regole che insieme prendono il nome di modello.

Figura 6.4: Camera di simulazione a gravità 0

6.3 Cosa vuol dire modello?

Un modello è un insieme di regole applicate ad un dominio tale che, dato uno stato iniziale, è possibile prevedere, dopo un certo numero di transazioni temporali, quali saranno le caratteristiche del nuovo stato del dominio. In poche parole, se il nostro sistema è costituito da un grattacielo con un determinata resistenza R ed un vento che soffia sempre dalla stessa direzione con una determinata forza V e le regole imposte sono che ogni secondo la forza del vento aumenta di 1 ed il grattacielo crolla se la forza de vento è superiore alla sua resistenza, dati R1 e V1 posso con certezza assoluta dire che il grattacielo della simulazione crollerà dopo V1-R1 secondi. Se il modello per la simulazione che ho creato è sufficientemente veritiero e quindi rappresenta abbastanza fedelmente la situazione di un vero grattacielo sottoposto all'azione di un vero vento che aumenta costantemente di potenza, c'è un'ottima probabilità che il vero grattacielo crolli se sottoposto per V1-R1 secondi al vero vento. Quello appena presentato è evidentemente un esempio banale e poco veritiero, possiamo però riprendere alcuni degli esempi presentati in precedenza e cercare di abbozzare un modello per alcuni di questi. Banalmente possiamo generare un applicazione che ci permetta di definire un numero di posti a sedere P, un numero di servizzi sanitari S, ed un coefficiente di bisogno dell'utilizzo dei servizi B. Possiamo immaginare che il sistema invierà una segnalazione di errore ogni volta che cercheremo di aggiungere un posto a sedere che faccia in modo che il rapporto tra i gabinetti e i possibili utenti della struttura sia inferiore a B, in questo modo simuliamo la necessità di un certo numero di servizi per ogni

Figura 6.5: Torri rotanti - esempio di architettura dinamica

persona che probabilmente entrerà nel nostro edificio. Aggiungendo un livello di complessità alla nostra precedente applicazione immaginiamo di stabilire un coefficiente di difficoltà D per ogni zona di edificio percorribile che possiamo definire all'interno del nostro edificio. Definiamo poi dei coefficienti di abilità A assegnati ad ogni possibile utente della struttura. Il programma dovrà segnalare errore ogni volta che tentiamo di costruire un edificio che permetta l'entrata con un coefficiente di difficoltà D minore di quello dell'uscita o dei percorsi per raggiungere i sanitari. In questo modo saremo sicuri che l'edificio che andremo a costruire garantirà a chi è in grado di entrarci la possibilità di accedere ai servizi e alle uscite. Ancora, ipotizziamo di avere due strade che vengono imboccate da un certo numero di macchine che dipende dal momento della giornata e che queste strade formino un incrocio regolato da un semaforo. I periodi del semaforo dovranno essere definiti dal rapporto tra il numero di macchine in attesa e quelle che occupano l'incrocio tenendo conto del fatto che questo rapporto varia a seconda dell'ora corrente. Questi sono semplici esempi di quello che un modello può rappresentare. Al giorno d'oggi esistono modelli complessi utilizzati in economia, statistica, demografia ed un'infinità di altri campi di ricerca.

6.4 Complessità e accuratezza: la relazione tra simulazione e modello

Nel paragrafo precedente si è parlato di modelli complessi e di accuratezza di un modello analizzando più esempi, si è quindi introdotto a livello intuitivo

il concetto di complessità di un modello e il concetto di accuratezza di un modello. Entrambi i concetti costituiscono il punto di contatto tra una simulazione ed il modello sottostante ad essa: in particolare la complessità di un modello e quindi di una simulazione è rapprasentata dal numero di elementi che la simulazione prende in considerazione, dalla grandezza del dominio in cui questi elementi sussistono ed infine dal numero di possibili diverse interazioni tra elementi distinti. Generalmente un modello tende a non essere mai troppo complicato e a tenere conto solamente degli aspetti principali e caratterisitici dell'oggetto da simulare, il progetto viene astratto e semplificato il più possibile in modo da ottenere un modello veritiero che risulti comunque facile da maneggiare: è infatti impensabile cimentarsi con complicati sistemi di equazioni differenziali quando la maggior parte delle variabili che entrano in gioco sono poco rilevanti ai fini della valutazione della simulazione stessa. I vantaggi dietro alla simulazione sono proprio quelli di studiare il fenomeno simulato attraverso un modello semplice e gestibile; esagerare nei dettagli e negli elementi da prendere in considerazione diminuirebbe i benefici che traiamo dal concetto stesso di simulazione. L'accuratezza di una simulazione è quindi indicativa della frequenza di eventi che agiscono sul nostro sistema reale che non sono presi in considerazione nella nostra simulazione. Prendendo l'esempio del nostro grattacielo sottoposto alla forza del vento possiamo definirlo un sistema semplice perché prende in considerazione due dati di cui uno solo variabile in N e una sola relazione banale (maggioranza) tra i due elementi. Per quanto riguarda l'accuratezza di questo sistema è un sistema quantomai poco accurato in quanto dato un generico grattacielo è quantomai improbabile che esista un vento che soffia con potenza costantemente in aumento sempre dalla stessa direzione contro di esso, e se anche questo fosse vero ci sarebbero sicuramente un gran numero di altri elementi che entrerebbero in gioco, quali ad esempio l'effetto che un vento in grado di spezzare un grattacielo avrebbe sull'intorno del sistema che stiamo prendendo in considerazone. Il nostro sistema sarebbe già molto più accurato se la potenza del vento non fosse in costante aumento ma semplicemente variabile in maniera casuale nel tempo; il sistema risulterebbe ancora più accurato, nonché più semplice se la forza del vento avesse un valore massimo che può raggiungere. Aumenteremmo l'accuratezza del sistema ma anche la sua complessità inserendo due fattori di oscillazione, uno che rappresenta lo stato di movimento del grattacielo e uno lo stato di variazione della potenza del vento. A questo punto avremmo però bisogno di aumentare il numero di regole che determinano le relazioni tra i vari coefficienti del sistema e quindi il sistema si complicherebbe aumentando così in accuratezza. E' quindi intuitivo e facilmente riscontrabile come i concetti di complessità e accuratezza siano in realtà molto legati tra di loro: in generale è vero che una simulazione è tanto più accurata quanto è più complessa e quindi è complesso il suo modello, anche se in realtà non esiste una regola definita sulla relazione tra questi due fattori. Il livello di dettaglio e accuratezza vengono decisi a priori dall'analista e sono ancora un oggetto di studio e di ricerca di grande importanza.

6.5 Limiti dei modelli per la simulazione

La simulazione, come ogni attività svolta sfruttando le capacità di un calcolatore elettronico, richiede un certo quantitativo di risorse logiche e fisiche. I limiti di questa disciplina sono costituiti esattamente da questa condizione, unitamente alla durata del processo di codifica delle regole del modello. Supponiamo per un istante di voler simulare il sistema innegabilmente più complicato che esiste in natura: l'essere umano. Riduciamo drasticamente le situazioni di vita che questa nostra simulazione vuole rappresentare ed immaginiamo che il nostro automa abbia semplicemente il compito di decidere quale vestito indossare per una indefinita giornata. Dovremmo scrivere delle regole per far si che l'automa scelga dei capi di abbiagliamento:

- della sua taglia
- adatti al lavoro che deve svolgere
- utili rispetto agli elementi con cui dovrà interagire
- che si adattino al gusto personale simulato dal nostro automa

In pratica dobbiamo tenere in considerazione se il vestito è troppo grande, se è troppo piccolo, se deve essere elegante, se deve essere comodo, se deve potersi sporcare, se deve potersi bagnare, se deve essere a norma di legge, se ha una forma piacevole ecc.. Immaginate adesso di dover combinare tutti i possbili accoppiamenti tra biancheria intima, calzini, magliette, pantaloni, camice, giacche, felpe, cappelli, cinte, cravatte, orologi, scarpe, borse e quant'altro possa esistere nel guardaroba di un essere umano. Si tratta di un lavoro enorme che può essere ottimizzato tramite un altro lavoro enorme e si consideri che quanto presentato sopra è esclusivamente il sistema di relazione tra un essere umano e un numero ristretto di oggetti. Quando poi si va a tentare di stabilire una serie di regole per l'interazione tra due esseri umani allora il compito è semplicemente impossibile, anche perché gli esseri umani sono in grado a loro volta di simulare, quindi si entrerebbe in un discorso che ha le caratteristiche di una ricorsione senza una fine. Ci troviamo di fronte a dei colli di bottiglia di tipo tecnico nei mezzi a disposizione per effettuare la nostra simulazione: anche ammettendo di avere un computer con una memoria capace di processare dei file contenenti tutti i possibili comportamenti umani, ci vorrebbe almeno qualche decina di anni per raccogliere e codificare tutti i dati necessari ad una simulazione pressocchè veritiera. Il rapporto tra **complessità** ed **accuratezza** costituisce quindi il **limite stesso della simulazione** e quindi dei modelli su cui questa si basa.

Figura 6.6: Una lince (predatore) e un coniglio (preda)

6.6 Esempi di modello

6.6.1 Il modello preda predatore

La potenza della scienza dei modelli è costituita dalla generalità che un modello dà ad un problema: riprendendo l'esempio banale del grattacielo e del vento, il modello prevede, una forza, una resistenza e un punto di rottura determinato dalla relazione di maggioranza dei due parametri. Questo stesso modello può essere adottato anche su sistemi diversi da quello di un grattacielo esposto al vento: immaginiamo un'altalena ed il peso del bambino che sale sopra, la molla di una bilancia ed il peso massimo che questa può sostenere, una giostra e la forza centrifuga a cui sono sottoposi i suoi seggiolini, e via dicendo. Restringiamo adesso il nostro campo di studi ad uno specifico modello che è stato oggetto di studi e al quale sono state applicate nel corso degli anni una notevole quantità di variazioni per ottimizzarlo alla situazione contingente. Il modello in questione prende il nome di **Modello Preda Predatore** (predator-prey) ed ha come obiettivo quello di simulare l'andamento demografico di due popolazioni immaginando che una tragga nutrimento dall'altra. Indipendentemente dal fatto che si tratti di carote e conigli, lupi o agnelli, vampiri ed umani, nobiltà e plebe, dirigenti di azienda e operai o qualunque altro caso assimilabile a questi, l'obbiettivo per il modello è quello di tenere traccia del cambiamento dei valori di alcuni fattori allo scorrere del tempo. Abbiamo pertando già definito alcuni aspetti del nostro modello che sono: una popolazione di prede, una popolazione di predatori, e lo scorrere del tempo. Vediamo ora come è stato affrontato il problema da due matematici che hanno vissuto a cavallo tra il 19° e 20° secolo.

6.6.2 Il modello Lotka - Volterra

Alfred James Lotka e Vito Volterra sono due scienziati che nel corso della loro vita si sono dedicati a diversi aspetti della matematica, della fisica, della

chimica e della statistica. Pur non collaborando tra loro svilupparono nel 1924 Lotka, e nel 1926 Volterra lo stesso modello per la soluzione della dinamica preda-predatore, per questo si è scelto di attribuire l'origine del modello ad entrambi gli autori. La loro idea fu che le caratteristiche che assicurano la

Figura 6.7: A sinistra Alfred J. Lotka e a destra Vito Volterra

sopravvivenza alle prede e ai predatori sono incompatibili ed è l'interazione dovuta alla lotta per la sopravvivenza a influenzare la dinamica delle popolazioni delle prede e dei predatori. Entriamo ora in dettaglio: la soluzione proposta è costituita da un sistema di equazioni differenziali non lineari del primo ordine:

$$\begin{cases} \frac{dx(t)}{dt} = n_x x(t) - m_x y(t)x(t) \\ \\ \frac{dy(t)}{dt} = n_y x(t)y(t) - m_y y(t) \end{cases}$$

Abbiamo qundi due funzioni che rappresentano il parametro demografico delle due popolazioni al variare del tempo: chiameremo $x(t)$ la popolazione delle prede e $y(t)$ quella dei predatori. Ogni preda ha la facoltà di riprodursi fino ad una volta per ogni unità di tempo, pertanto indicheremo con n_x il tasso di natalità della popolazione delle prede. Ogni predatore ha la facoltà di nutrirsi massimo di una preda per unità di tempo, indichiamo quindi con m_x il tasso di nutrimento che la popolazione dei predatori impone alla popolazione delle prede ossia il tasso di mortalità della popolazione delle prede. Ogni predatore ha la capacità di riprodursi solo se le condizioni circostanti sono favorevoli, cioè se ci sono abbastanza prede per sfamare tutta la popolazione dei predatori, chiamiamo quindi n_y il tasso di natalità dei predatori. Si assume che un predatore, non avendo nemici naturali all'interno del sistema, raggiunga senza difficoltà il termine del suo ciclo di vita e muoia quindi di vecchiaia, chiamiamo m_y il tasso di mortalità della popolazione dei predatori. $\frac{dy(t)}{dt}$ (da ora in poi utilizzeremo la notazione $x'(t)$) rappresenta quindi il tasso di crescita della

popolazione delle prede mentre $y'(x)$, quello dei predatori. Riepilogando:

- $x(t)$ = popolazione delle prede al tempo t
- $y(t)$ = popolazione dei predatori al tempo t
- n_x = tasso di natalità delle prede
- n_y = tasso di natalità dei predatori
- m_x = tasso di mortalità delle prede
- m_y = tasso di mortalità dei predatori

Cerchiamo ora di approfondire i nostri studi sull'argomento analizzando il sistema con i tradizionali metodi matematici: Immaginiamo alcune situazioni limite per cercare di capire cosa succederebbe ad esempio se come parametri iniziali una delle due popolazioni fosse di 0 individui, ecco due casi interessanti:

Primo caso limite

Ipotizziamo per esempio un sistema in cui non esiste la specie "predatori", pertanto y(t) = 0 per ogni t appartenente a R. Le nostre equazioni risulterebbero:

$$\begin{cases} \frac{dx(t)}{dt} = n_x x(t) - m_x 0 x(t) \\ \frac{dy(t)}{dt} = n_y x(t) 0 - m_y 0 \end{cases}$$

Pertanto avremmo:

- $x'(t) = n_x x(t)$
- $y'(t) = 0$

da questi due risultati deduciamo che la popolazione di prede, in assenza di predatori crescerà linearmente all'infinito secondo il suo tasso di natalità, situazione che almeno a livello intuitivo rispecchierebbe il vero: la preda non avendo motivo di morire se non quello dell'età ha libertà di riprodursi.

Secondo caso limite

Nel caso opposto, cioè con una popolazione di prede inesistente e quindi x(t) = 0 per ogni t appartenente ad R, otterremmo:

$$\begin{cases} \frac{dx(t)}{dt} = n_x 0 - m_x y(t) 0 \\ \frac{dy(t)}{dt} = n_y 0 y(t) 0 - m_y y(t) \end{cases}$$

Pertanto risulta:

- $x'(t) = 0$
- $y'(x) = -m_x y(t)$

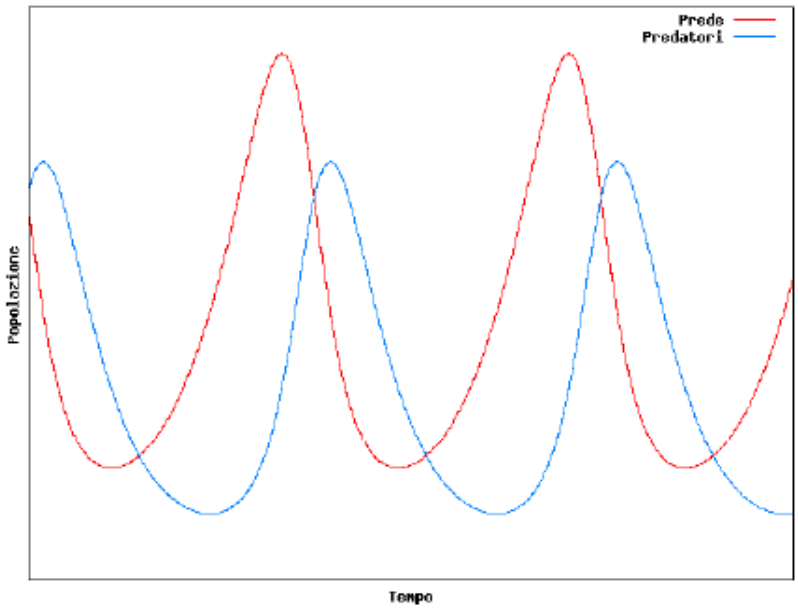
Da cui si deduce facilmente che la popolazione dei predatori si estinguerà a sua volta in maniera lineare secondo il suo fattore di mortalità: il predatore trovandosi a meno della sua principale se non unica fonte di cibo non riesce a sopravvivere. Un altro caso di interesse rilevante è quello dei punti stazionari del sistema, cioè quei punti nei quali entrambe le variazioni demografiche risultano perfettamente uguali a 0:

$$\begin{cases} 0 = n_x x(t) - m_x y(t) x(t) \\ \\ 0 = n_y x(t) y(t) 0 - m_y y(t) \end{cases} \quad -> \quad \begin{cases} 0 = x(t)(n_x - m_x y(t)) \\ \\ 0 = y(t)(n_y x(t) - m_y) \end{cases}$$

Osservando la forma in cui abbiamo riscritto il sistema è facile intuire che esistono due punti stazionari:

$$\begin{cases} x(t) = 0 \\ \\ y(t) = 0 \end{cases} \quad -> \quad \begin{cases} x(t) = \frac{m_y}{n_y} \\ \\ y(t) = \frac{n_x}{m_x} \end{cases}$$

Da questo possiamo concludere che le uniche due situazioni in cui il sistema resta ivariato nel tempo sono quando otteniamo la completa estinzione oppure quando il numero della nascita delle prede è perfettamente coincidente al numero di prede mangiate dai predatori. Si trascura di dimostrare che il primo di questi due punti $(0, 0)$ non è un punto dall'equilibrio stabile, mentre il secondo è un punto stabile. Da queste affermazioni trarremo soltanto alcune conclusioni senza spendere troppo tempo nella dimostrazione matematica. Un punto stazionario dall'equilibrio instabile è un punto verso cui i valori negli altri punti difficilmente saranno attratti, questo ci garantisce che la coesistenza delle due specie, data una situazione di partenza ragionevolmente distante dall'estremo 0, difficilmente porterà alla reciproca estinzione. Al contrario la stabilità dell'equilibrio del secondo punto stazionario ci garantisce cha il sistema oscillerà periodicamente intorno a questo valore. Scrivere le formule trigonometriche di questa oscillazione non è cosa semplice a livello matematico. Limitiamoci a confermare le nostre conclusioni osservando il grafico dell'andamento del coefficiente demografico delle due specie.



Dal grafico si nota come paradossalmente la situazione stazionaria sia una situazione di tipo oscillatorio: il numero delle prede e dei predatori varierà sempre, mantenendo però il sistema stabile e permetterà alle due specie di coesistere nello stesso ambiente per un periodo di tempo indefinito.

Limiti del modello

I limiti del modello elaborato da Lotka e Volterra in realtà è intrinseco alla definizione stessa di modello che abbiamo cercato di dare nei paragrafi predenti: ovvero la semplificazione del sistema ignorandone alcuni fattori. In questo caso le equazioni non tengono conto che:

1. La specie predatori non può consumare un infinita quantità di prede
2. Vengono ignorate le condizioni ambientali, fattore invece fondamentale

Capitolo 7

Automi cellulari

7.1 Introduzione

Abbiamo analizzato a tutti gli effetti il significato di simulazione e affrontato il concetto di modello. Quello che ora vogliamo fare è prendere in considerazione un tipo particolare di modello e studiarne le caratteristiche per avere una concezione e una coscienza più profonda del significato e sopratutto del ruolo che i modelli hanno nello studio delle scienze. La seguente presentazione ha lo scopo di introdurre quelli che vengono definiti gli automi cellulari, un tipo di modello piuttosto recente le cui sorprendenti potenzialità continuano ad essere studiate e ad essere oggetto di ricerca. Ci soffermeremo sopratutto sugli utilizzi che i suddetti automi cellulari hanno per rendere l'idea della vasta gamma di applicazioni nelle più diverse discipline che un modello in genere può avere.

7.2 Definizione

Un automa cellulare (cellular automaton) è un modello discreto che consiste in uno spazio cartesiano di una o più dimensioni (solitamente una griglia) di elementi detti celle. Ogni cella può trovarsi ad ogni istante di tempo t, in un uno ed un unico di diversi stati. Lo stato di una cella varia secondo una o più regole di transizione uguali per tutte le celle: ogni cella cioè viene trattata allo stesso modo. Le regole di cambiamento di stato dipendono dalle altre celle "vicine".

7.2.1 Definizione formale

Un automa cellulare è una quadrupla (D, S, r, R) dove:

- D è la dimensione dell'automa in Z^D. Esso può essere:
  - a 1 dimensione (automa vettore)
  - a 2 dimensioni (griglia, triangolo esagono ecc..)

  - a 3 dimensioni (cubo ecc..)
  - a n dimensioni, con n intero finito

- S è l'insieme degli stati che una cella può assumere per istanza di tempo.
- r è il "raggio di vicinato", ovvero il raggio entro la quale ogni cella si "accorge" dello stato dei proprio vicini: con r = 1 in una griglia una cella sarà a conoscenza dello stato delle 8 ad essa adiacente.
- R è l'insieme delle regole che determinano il passaggio da uno stato all'altro (le regole sono le stesse per ogni cella)

7.2.2 Parole chiave

- generazione: ogni volta che le regole vengono applicate e si passa dall'istante di tempo t_i all'istante t_{i+1} viene creata una nuova generazione.
- configurazione: l'insieme degli stati delle celle all'istante t_i

7.3 Le origini del modello

Nei primi anni del 1940 Stanislaw Ulam lavorava presso il Los Alamos National Laboratory sulla crescita della struttura dei cristalli servendosi di una semplice griglia come modello. Contemporaneamente Von Neumann (che allora era suo collega a Los Alamos) stava lavorando su un problema assai più arduo: i "sistemi autoreplicanti": la sua idea originale era quella di un "robot" in grado di poter ricreare un altro robot uguale a se stesso. La collaborazione tra i due nacque quando Ulam si accorse che il collega avrebbe potuto utilizzare i modelli a griglia su cui stava lavorando come modello matematico astratto per i suoi sistemi autoreplicanti. Nacque così il primo sistema integrato con automi cellulari : un autoreplicatore e costruttore basato su automi dal ristretto raggio (solo le celle adiacenti erano considerati i vicini) con 29 stati per cella. Questo sistema è passato alla storia come "universal constructor". Nel il 1969 il tedesco Konrad Zuse pubblico "Calculating Sistem", libro nel quale per la prima volta venne introdotta una visione della fisica discreta come un gigantesco automa deterministico computazionale che lavorava a stati discreti. Questo fu il primo trattato di quella che oggi viene chiamata fisica digitale. Negli anni 70 ritroviamo gli automi cellulari nell rivista Scientific American in un articolo di Martin Gardner: Game of Life. Inventato da John Conway il "gioco" non era altro che un automa cellulare su griglia a 2 stati: vedermo in seguito in dettaglio in cosa consisteva ed il successo che ebbe. Nel 1983 Stephen Wolfram pubblicò il primo di una serie di articoli che investigavano le proprietà nascoste di una nuova classe di automi cellulari ancora sconosciuti

Figura 7.1: A destra John Von Neumann e a sinistra Stanislaw Ulam

che lui chiamò ementary cellular atuomata. L'inaspettata complessità che si celava dietro alle semplici regole di questi modelli portò Wolfram a pensare che la complessità della natura potessere essere dovuta a meccanismi simili. Lasciata l'accademia Wolfram si dedicò completamente allo studio degli automi celluari che culminò nella pubblicazione di un nuovo libro: A new kind of science, un trattato di 1280 pagine in cui vengono discusse tutte le scoperte e riultati ottenuti e in cui vengono mostrati i collegamenti che gli automi hanno con le altre discipline. Il libro fu rivisto, riadattato e aggiornato nel 2005 da Rudy Rucker con il nome di "The Lifebox".

7.4 Dettagli e caratteristiche

L'automa cellulare sicuramente più gettonato è quello a 2 dimensioni. Il modo più efficace per rappresentarlo una griglia bidimensionale su un immaginario foglio infinito con un set di regole da definire. I vicini di una cella sono gli 8 "quadrati" che la toccano: per ogni cella e i suoi corrispondenti vicini abbiamo cosi 2 x 2^8 configurazioni possibili: per ognuna di queste possibili configurazioni le nostre regole ci indicheranno come sarà lo stato della cella di mezzo all'istante di tempo successivo. Nella realtà però, (come per i famosi "nastri" infiniti della macchina di Turing) accade che non abbiamo a disposizione un foglio di dimensioni infinite, quindi la nostra griglia sarà finita. Nasce allora il problema di come gestire le celle ai bordi della griglia dato che queste ultime non possiedono 8 vicini come le altre. Una possibile soluzione è quella di lasciare che gli stati delle celle ai bordi siano sempre gli stessi della configurazione di partenza, ma così facendo si andrebbe anche a influenzare l'andamento delle celle ad esse adiacenti. Si potrebbe invece definire queste ultime come celle speciali alle quali attribuire regole diverse, ma questo andrebbe a violare la definizione di automa cellulare stesso che prevede che tutte le cellule siano equivalenti (e

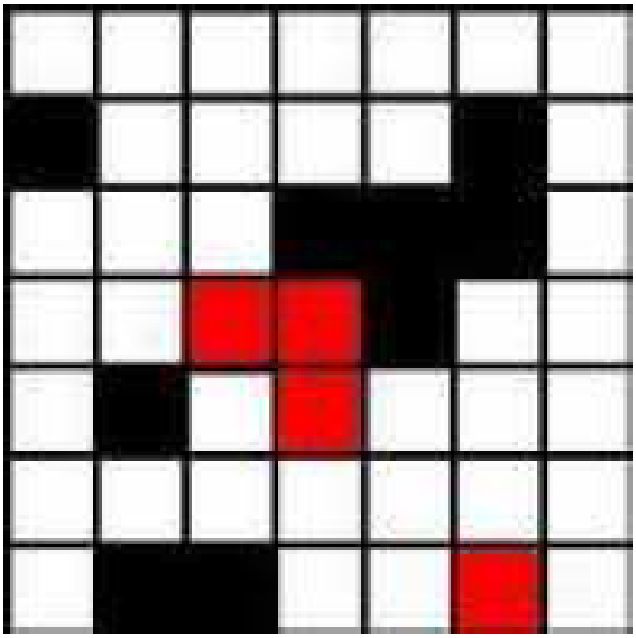


Figura 7.2: Automa cellulare a 2 dimensioni e 3 stati

quindi anche soggette alle stesse regole di transizione). Il metodo più usato è quello del "toro", ovvero considerare la griglia come se fosse mappata su una superficie toroidale (a ciambella) in modo che si finisca alla corrispondente posizione nella griglia in basso se si proviene dall'alto, alla corrispondente posizione nella griglia a destra se si proviene da sinistra. Con questo "trucco" abbiamo risolto il problema delle celle isolate ai bordi costruendo una griglia di dimenzioni artificialmente infinite.

7.4.1 Possibilità di un automa cellulare

- ammettere una configurazione stabile: ovvero quella configurazione che resta uguale a se stessa all'istante di tempo successivo (e quindi si stabilizzerà così per sempre)
- essere periodico: si dice periodico di ciclo n quell'automa che dopo n configurazioni consequenziali ritorna ad una configurazione particolare C_i e così via, la configurazione stabile è considerata come ciclo di lunghezza 1.
- essere reversibile: se per ogni possibile configurazione all'istante t_i ne esiste una ed una sola precedente. Immaginando un automa cellulare come una funzione che mappa una configurazione in un'altra, la reversibilità può essere intesa come la garanzia che la funzione sia

biiettiva. Questa caratteristica può essere riscontrata attraverso un algoritmo conosciuto negli automi vettoriali, ma è stato dimostrato che non esiste algoritmo in grado di decidere con sicurezza se, dato un insieme di regole l'automa corrispondente sia reversibile. In generale la reversibilità è indecidibile per sistemi a 2 o più dimensioni. In alcuni casi di non reversibilità esistono delle particolari configurazioni per le quali non esistono configurazioni precedenti: esse vengono chiamate "Garden of Eden Patterns", in altre parole, non esistono configurazioni che si evolveranno nel GoE pattern.

- essere probabilistico: se per ogni cella la regola ad ogni istanza di tempo si applica con probabilità fissata p ci troviamo di fronte a delle regole non deterministiche.
- essere continuo: viene definito coninuto l'automa per il quale vengono utilizzate funzioni continue invece che regole deterministiche per la transizione da uno stato all'altro. Gli stati di ogni cella possono essere per esempio un sottointervallo dei numeri reali (S in $[0, 1]$).
- cambiare le sue regole a seconda del tempo o dello spazio: per esempio alcune zone della griglia potrebbero essere soggette a regole diverse o queste ultime potrebbero cambiare dopo un intervallo di tempo prestabilito o anche scelto a caso con probabilità p

7.5 Utilizzi

Il passo in avanti più importante nell'utilizzo degli automi cellulari dopo la loro creazione da parte di Von Neumann e Ulman è stato quello degli studi di Wolfram, che nel corso degli anni 80 si dedicò allo studio delle loro proprietà e con la pubblicazione del libro "A new kind of science" impressionò la comunità della scienza con la dimostrazione dell'utilità e potenzialità dei suddetti modelli. Gli automi cellulari forniscono una rappresentazione immediata (e, in un certo senso, semplice) di fenomeni in cui l'evoluzione globale dipende da leggi locali: la bellezza che si cela dietro ad alcuni sistemi complessi presenti in natura è che le azioni di semplici componenti con comunicazione e informazioni locali diano contributo e coordinazione al processo di evoluzione globale del fenomeno.

7.5.1 Automi e algoritmi decentralizzati

Recentemente è nato un grande interesse nel campo della ricerca nello studiare sistemi decentralizzati come i network di sensori o più sofisticati sitemi microscopici che lavorino su informazioni locali. M. Mitchell, professore in CS alla Portland State University sta studiando attraverso gli atuomi cellulari algoritmi genetici decentralizzati. Quali sono le caratteristiche affinchè un algoritmo

sia decentralizzato? Se prendiamo, per esempio, il caso di un grafo, un algoritmo decentralizzato è quell'algoritmo che cerca il cammino tra un vertice sorgente ed uno destinatario che gode di queste proprietà:

- ogni vertice "conosce" la sorgente e il destinatario
- ad ogni passo ogni vertice conosce la lista dei vertici che sono stati passati
- ogni vertice conosce solamente i propri vicini

E' osservando l'utlima proprietà che saltano in mente gli automi cellulari: la località. Mitchell e i suoi collboratori applicano cosi' algoritmi decentralizzati per evolvere la struttura di un automa cellulare: il suo gruppo di lavoro ha utilizzato un singolo array bidimensionale in cui ogni aveva solo 2 stati e 2 vicini (quello destro e quello sinistro). L'array può essere pensato come un circolo in cui l'ultima cella e la prima sono vicine. L'evoluzione dell'array è stata tracciata attraverso la variazione di 0 e di 1 (i 2 stati) dopo ogni iterazione: la ricerca si è interessata di conoscere le regioni con più o meno densità di 0 e 1. La motivazione di questo approccio nasce dalla necessità di capire i sistemi naturali e di poter costruire sistemi decentralizzati computazionali. Il gruppo di Mitchell ha mostrato 3 livelli di processazione dell'informazione durante l'iterazione dell'automa:

- il primo livello è quello della trasmissione e "salvataggio" delle informazioni dovuto dalle celle
- il secondo è comprendere la geometria delle soubroutines in regioni di media scala come per esempio la grandezza di regioni ad alta o bassa densità di 0 o 1
- L'ultimo livello è la computazione globale del sistema vista "dall'alto"

7.5.2 Automi e crittografia

L'idea del loro utilizzo in crittografia nasce dal fatto che, date certe regole ed una configurazione, è immediato passare da una configurazione all'altra ma non lo è affatto il contrario. E' inoltre molto difficile data una successione di configurazioni trovare le regole che passino da una alla successiva. Questo è un campo di ricerca recentissimo e le cui potenzialità sono ancora sconosciute.

7.5.3 Automi e sistemi biotici naturali

Alcuni oggetti presenti in natura utilizzano automi cellulari nel loro funzionamento. Schemi come quelli di alcune conchiglie di mare, come quelli del Conus e della Cymbiola sono generati da naturali automi. I pigmenti di ogni regione interagiscono con l'attivazione/inibizione dei pigmenti delle regioni limitrofe

obbedendo a regole matematiche precise. Per esempio, i "disegni" della specie Conus textile si comportano come un automa cellulare a due dimensioni che segue le "Rules 30".

Figura 7.3: Conus textile

7.5.4 Automi e chimica

Negli anni 50 il chimico Zhabotinsky scoprì cuna particolare miscela facendo reagire bromato di potassio, solfato di cerio (IV) e acido citrico in soluzione di acido solforico diluito. La reazione dava origine a una spettacolare sequenza geometrica con cerchi concentrici e spirali che si propagavano sulla superficie della miscela chiamata reazione oscillatoria che poteva perdurare per ore. Nel 1998 il professore della Scientific American A. K. Dewdney propose un automa cellulare la cui evoluzione assomigliava veramente alla reazione di Belousov-Zhabotiksky. Anche il comportamento dei gas perfetti, o il movimento di filamenti di DNA all'interno di una soluzione sono altri esempi di come questi automi entrano in gioco in sistemi di tipo chimico.

7.5.5 Automi e processori nei computer

Alcuni processori odierni (basati sull'architettura di Von Neumann) utilizzano parti integrate che funzionano come automi cellulaulari in cui le celle comunicano con i vicini attraverso cariche elettriche, vibrazioni o cariche magnetiche.

Capitolo 8

Modelli basati su agenti

8.1 Introduzione

Nei modelli basati su agente (Agent-Based Model = ABM), un sistema è modellato come una collezione di entità autonome capaci di prendere delle decisioni, queste entità prendono il nome di agenti. Ogni agente valuta la propria situazione e prende delle decisioni in base allo schema di regole che caratterizza il sistema in cui l'agente è inserito. Ogni agente può così eseguire diverse funzioni appropriate all'elemento della vita reale che esso rappresenta, come per esempio produrre, consumare o vendere. Le ripetute interazioni tra gli agenti di un sistema sono l'oggetto principale della ricerca sviluppata tramite la simulazione basata su agente, questa ricerca fa affidamento alla potenza dei calcolatori per analizzare le dinamiche che non sono alla portata dei modelli matematici. Il più semplice fra tutti gli ABM è quello che al suo interno prevede la presenza di agenti tutti dello stesso tipo e delle relazioni che intercorrono tra questi. Anche in un semplice modello basato su agente possiamo notare interessanti sequenze di comportamenti che rivelano informazioni interessanti riguardo le dinamiche reali del sistema che stiamo simulando. Inoltre gli agenti possono essere in grado di evolversi, ossia di mostrare comportamenti imprevedibili. I modelli basati su agente più sofisticati spesso includono al loro interno reti neurali, algoritmi evolutivi o altre tecniche di apprendimento per permettere una simulazione più realistica delle dinamiche di adattamento. Più che uno strumento tecnico i modelli basati su agente sono una "forma mentis" che consiste nel descrivere un sistema dalla prospettiva dei suoi elementi costituenti. Un certo numero di ricercatori afferma che i classici modelli ad equazioni differenziali siano un'altrenativa ai modelli basati su agente, ma questo non è esatto, così come non è esatto che un insieme di equazioni differenziali che rappresentino ognuna il comportamento di un singolo agente sono un modello basato su agente. Un sinonimo di modello basato su agente può essere modello microscopico, ed una valida alternativa ad esso può essere un modello macroscopico. Poiché ad oggi le simulazini basate su agente hanno acquisito

una certa popolarità, è bene sapere quali sono le loro proprietà ed i loro punti di forza e quali sono le situazioni in cui è il caso di applicarle. Più avanti in queste pagine risponderemo a questa esigenza prima classificando e discutendo dei vantaggi che portano in modelli basati su agente nella simulazione, e poi analizzando diversi esempi. Ma ora vediamo di soffermarci sull'aspetto più importante e di darne una definizione precisa: l'agente, il vero e proprio cuore del modello.

8.2 Cos'è un agente?

Sebbene non esista un consenso definitivo sulle definizione esatta del termine agente sono più le caratteristiche su cui si è d'accordo che quelle sui cui si è in disaccordo. Alcuni modellatori considerano qualsiasi tipo di componente indipendente (software, modello, individuo) essere un agente, altri inistono che il "comportamento" di un agente debba essere adattivo per essere considerato un agente: l'etichetta agente dovrebbe essere quindi riservata a componenti che in qualche modo imparino dalle loro esperienze e modifichino il loro comportamento in base ad esse. Nel 1997 Casti affermò che un agente dovrebbe contenere entrambe le regole di basso livello che quelle di alto livello per modificare il proprio comportamento ("rules that change the rules"). Da un punto di vista più pratico consideriamo però tutti gli agenti aventi certe caratteristiche:

- un agente è identificabile, un individuo discreto con un set di caratteristiche e regole che ne governino il comportamento e che gode della capacità di fare decisioni.
- un agente è localizzato, esso vive in un sistema in cui interagisce con altri agenti. Gli agenti hanno protocolli di comunicazione e la capacità di rispondere a stimolazioni esterne.
- un agente è "Goal-directed", ovvero ha obiettivi da raggiungere rispettando le regole del suo comportamento
- un agente è autonomo, deve essere in grado di sopravvivere indipendentemente dal contesto e dall'interazione con altri agenti, almeno in specifiche situazioni.
- un agente è flessibile, e può avere la capacità di imparare dalle esperienze modificando le proprie regole di comportamento

Diffirentemente da quanto abbiamo visto per gli automi cellulari, che sono un caso specifico dei sistemi basati su agente, gli agenti di questi ultimi sono diversi, eterogenei e dinamici negli attributi e nelle regole di comportamento.

8.3 ABM vs modelli differenziali

Come abbiamo visto spesso il concetto di modello basato su agente non viene ben compreso e si pensa che sia possibile sostiure gli ABM con i modelli differenziali. Vediamo perchè questo non è possibile:

- i modelli differenziali tengono in maniera intrinseca nella loro forma (l'equazione) il comportamento generale del sistema nella sua globalità, mentre gli ABM utilizzano la prospettiva dell'agente, osservando il globale dall'evoluzione del locale. Il modello differenziale ignora il singolo ed è interessato solo al risultato finale.
- se il comportamento e le caratteristiche dei singoli sono sufficientemente diversi tra loro un'equazione differenziali diventerebbe troppo complessa e si dovrebbe servire di un numero di variabili troppo grande per essere utile da studiare, questo non succede con un modello basato su agente.
- un ABM è di tipo bottom-up, il modello differenziale è di tipo top-down

8.4 Perchè utilizzare i modelli basati su agente?

Una delle ragioni che ha reso i modelli basati su agente così popolari è che sono di facile implementazione: in pratica, una volta che se ne è sentito parlare, si può programmare un modello basato su agente. Il fatto che la tecnica sia facile da utilizzare non vuol dire però che i concetti siano facili da padroneggiare: nonostante i modelli basati su agente siano tecnicamente semplici risultano essere concettualmente profondi. Questa combinazione inusuale porta spesso ad un uso improprio della simulazione basata su agente

8.4.1 Benefici dei modelli basati su agente

I principali vantaggi possono essere catalogati in tre gruppi:

1. acquisizione di informazioni riguardo fenomeni emergenti
2. naturale inclinazione alla descrizione di un sistema
3. flessibilità

È ragionevolmente chiaro, comunque, che la capacità dei modelli basati su agente di interagire con fenomeni imprevisti è quello che determina poi tutti gli altri vantaggi.

1. acquisizione di informazioni riguardo fenomeni emergenti

i fenomeni emergenti sono tutti quei comportamenti che si manifestano a causa delle interazioni tra gli agenti. Per definizione non possiamo ridurre questi fenomeni ad una parte del sistema, in quanto risulta chiaro che il tutto è maggiore della somma delle sue parti proprio a causa delle interazioni che scaturiscono tra le parti stesse. Un fenomeno emergente potrebbe avere proprietà decuplicate rispetto alle proprietà di una singola parte. Per esempio un ingorgo stradale, che è il risultato del comportamento e delle interazioni di diversi autisti, può muoversi nella direzione opposta a quella in cui si muovono le macchina che lo causano. Queste caratteristiche dei fenomeni emergenti li rendono difficili da comprendere e quasi impossibili da predire, anzi spesso portano a conclusioni intuitivamente inaccettabili. Molti esempi di questo tipo saranno descritti in seguito. I modelli basati su agente sono per loro natura l'approccio canonico allo studio dei fenomeni emergenti: nella simulazione basata su agente si modellano e simulano i comportamenti delle unità costituenti del sistema (gli agenti) e le loro interazioni, in modo tale da poter analizzare i fenomeni dalla loro nascita fino al loro termine mentre la simulazione procede. Il comportamento collettivo di un gruppo è un fenomeno emergente. Sarà utile quindi sfruttare i modelli basati su agente lì dove c'è una buona possibilità che si presentino fenomeni emergenti, come ad esempio:

- quando il comportamento individuale non è lineare e può essere caratterizzato da soglie, regole if-then, o accoppiamenti non lineari. Descrivere le discontinuità nei comportamenti individuali tramite un'equazione differenziale è difficile.

- Quando i comportamenti individuali mostrano processi legati alla memoria, alla dipendenza dai percorsi, all'isteresi, ai comportamenti non markoviani, o a correlazioni temporali incluso l'apprendimento e l'adattamento l'uso degli ABM è di moltovantaggioso.

- Quando le interazioni tra agenti sono eterogenee e possono generare effetti a catena. Si può presumere di garantire una mescolanza omogenea e globale tra le interazioni grazie a delle equazioni di flusso aggregate, ma la topologia della rete di interazioni può condurre a significative deviazioni dai comportamenti di aggregazione previsti.

- Quando la media non è significativa. Le equazioni differenziali aggregate tendono ad addolcire le fluttuazioni, i modelli ad agente no; questo è importante perché sotto certe condizioni le fluttuazioni possono essere amplificate: il sistema è lineare e stabile ma non per perturbazioni oltre una certa larghezza

Poiché la simulazione basata su agente genera fenomeni emergenti che possono essere analizzati dal principio al termine, è di crescente interesse stabilire cosa costituisce una spiegazione di questo tipo di fenomeni.

1. naturale inclinazione alla descrizione di un sistema

In molti casi gli ABM sono la più naturale forma per descrivere e simulare un sistema composto da entità dotate di comportamenti. Ogniqualvolta si cerca di descrivere una situazione di traffico, di flusso in un mercato o come funzionano organizzazioni sociali gli ABM sono decisamente vicini alla realtà. Per esempio viene più naturale servirsi di questi modelli per descivere come delle persone si muovono per fare shopping in un supermercato piuttosto che con un equazione che governi le dinamiche della densità delle stesse. La differenza fra il processo di buisness e le sue attività è un altro naturale esempio per capire quanto sono più naturali a volte gli ABM. Un processo di buisness è un astrazione, talvolta utile, la quale spesso è difficile da gestire e prevedere per le persone all'interno dell'organizzazione. Il modello basato du agente guarda all'organizzazzione dal punto di vista non del buisness ma da quello delle sue attività, che sono poi quelle che le persone nella realtà svolgono.

1. flessibilità

La flessibilità risiede nel fatto che è facile e veloce cambiare in qualunque momento il numero, il set di attributi/caratterstiche e le regole di comportamento degli agenti.

8.5 ABM, alcune applicazioni

Vediamo adesso alcuni esempi pratici di come sono stati utilizzati questi modelli.

8.5.1 Evacuazioni

Fughe precipitose di massa indotte dal panico spesso conducono a fatalità per persone che vengono buttate a terra e travolte dalla folla. Questi fenomeni capitano spesso in situazioni come incendi in strutture o risse in luoghi chiusi e talvolta apparentemente accadono senza una ragione ovvia. Esempi recenti includono il panico nello Zimbabwe e al Roskilde Rock concert in Danimarca. La frequenza di queste sventure sembra aumentare di continuo con la crescita della densità di popolazione combinata con le caratteristiche della società moderna in cui eventi a cui partecipano migliaia di persone come concerti pop, eventi sportivi o dimostrazioni sono sempre più frequenti. Le persone colpite dal panico sono ossessionate da impulsi incontrollati indidualisti indipendenti da culturà o società. La riduzione di attenzione in questi casi causa spesso che le uscite di sicurezza o quelle meno visibili vengano del tutto ignorate. In aggiunta a questo avviene una sorta di contagio sociale che si trasforma in una sorta di patologia di massa che si dilaga attraverso le azioni delle persone. Questi comportamenti spesso portano a fatalità che devono essere assolutamente evitate. In termini di agenti, il panico collettivo è un fenomeno emergente risultato

dall'interazione delle persone con lo stesso obiettivo a breve termine di uscire dal posto in pericolo nel quale si trovano. Per esempio, consideriamo il caso di un evacuazione da un edificio chiuso in caso di incendio e assumiamo che esista solo un uscita. Come si potrebbe aumentare il flusso di persone che esce dall'edificio? Uno si potrebbe anche chiedere: quale sarebbe l'effetto di piazzare una colonna proprio di fronte all'uscita, a un metro dalla porta? Intuitivamente uno potrebbe pensare che la colonna porti ad una diminuzione del flusso di uscita, ma, attraverso gli ABM e ad esperimenti reali si è verificato che la colonna aumenta invece che diminuire il flusso e diminuisce invece il numero di incidenti come persone travolte e calpestate. Eppure evidentemente la colonna gioca un ruolo di regolazione dividendo il flusso in due "sottoflussi" ordinati e costringengo le persone a prestare un'attenzione maggiore per evitare di sbattere contro la colonna che risulterà utile ai fini di controllare il panico.

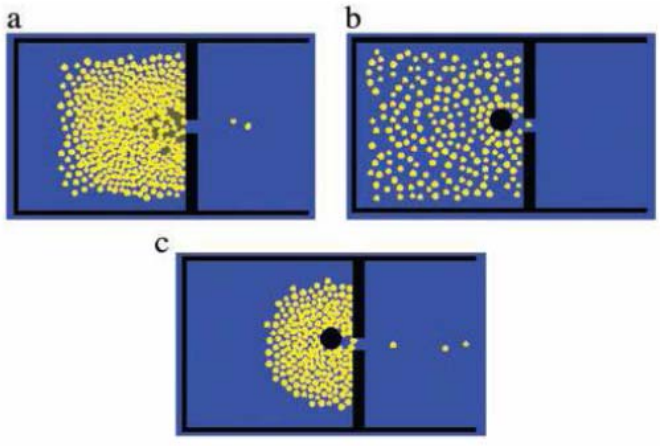
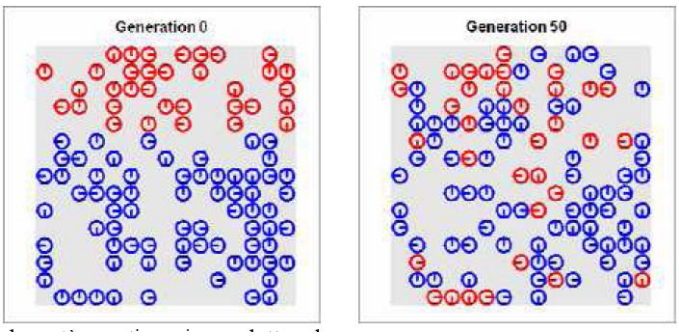


Figura 8.1: Simulazione di evacuazione senza colonna (a) e con colonna (b,c)

8.5.2 Interazioni sociali

In questo esempio ci sono due tipi di agenti: gli agenti blu e quelli rossi. Inizialmente ci sono un numero di agenti blu doppio rispetto a quello di agenti rossi. Ogni parte cerca di convincere l'altra che la sua posizione è migliore ma c'è una clausola: ogni agente rosso ha una capacità di convincere un agente blu doppia rispetto a quella blu. Un agente ha bisogno solo di essere circondato da metà agenti rossi per adottare la posizione di un agente rosso. La simulazione vedrà dominante una parte o si verificherà una sorta di compromesso?



8.5.3 Controllo dei flussi

Un'ovvia applicazione degli ABM nel controllo dei flussi è la gestione del traffico. Uno dei più ambiziosi progetti in questa area è stato intrapreso per molti anni al Los Alamos National Laboratory. Un team della sezione Technology and Safety Assessement ha rilasciato una software di simulazione del traffico metropolitano che possa essere usato dalle agenzie per la pianificazione dei servizi metropolitani. Il TRansportation ANalysis SIMulation System (TRANSIMS) è dotato di una popolazione virtuale che percorre percorsi giornalieri (va a lavoro, per negozi, a scuola ecc..) e crea una regione metropolitana con una completa rappresentazione degli individui della regione, le loro attività e tutte le infrastrutture. Questo "mondo" virtuale imita i comportamenti usuali dei viaggiatori di quelle specifiche regioni. Il software permette determinati tipi di controlli come:

- quando e dove rendere fuori servizio una particolare corsa
- come distribuire le corse attraverso lo spazio disponibile
- qual'è la tolleranza nel tempo d'attesa dei viaggiatori
- se e quando estendere le ore di servizio

Nella simulazione gli agenti rappresentano un realistico e modificabile mix di servizi (attrazioni, negozi ecc..) ed esigenze (viaggiatori con differenti preferenze). Inserendo risorse e dati presi dalla realtà come i servizi, i tempi di attesa, il numero di persone, il numero di veicolo e via dicendo il modello genera informazioni sul flusso del traffico metropolitano. L'utente può pianificare ed osservare un numero infinito di scenari per studiare le dinamiche del luogo e i suoi effettivi eventuali accorgimenti. Gli ABM si rivelano un'altra volta uno strumento molto potente perchè mappare le preferenze e comportamenti degli agenti da una parte e le caratteristiche dell'ambiente dall'altra sarebbe troppo complesso con un modello matematico.

Figura 8.2: Servizi metropolitani dell'Oregon

8.5.4 Organizzazioni

Un'area promettente dell'applicazione degli ABM è quella della simulazione organizzativa. E' certamente possibile modellare il comportamento emergente di un'organizzazione o di una parte di essa in un determinato contesto. Un organizzazione umana è spesso soggetta a rischi operazionali. Consideriamo le istituzioni finanziarie. I rischi comprendono forme di sistemi informativi inadeguati, problemi nei controlli interni, frodi, perdite di capitale ecc.. Sebbene molte banche si siano dotate spesso di sofisticati sistemi per affrontare rischi di mercato questo non è sempre stato sufficiente e non ha impedito che vere e proprie catastrofi avvenissero. Il problema è che molte banca mancano di una "traccia storica" delle loro operazioni fallimentari e delle cause delle loro perdite. Diversamente dai rischi di mercato, i rischi organizzativi e operazionali sono interni all'organizzazione e una connessione matematica o statistica definita tra i fattori di rischio e la frequenza di operazioni dannose non esiste. C'è un pressione in continua crescita da parte delle istituzioni finanziare di quantificare i rischi operazionali in modo da convinvere entrambi gli investitori e le entità regolatrici. Più precisamente un'istituzione finanziaria deve essere in grado di quantificare i rischi e tenerli sotto controllo e ottimizzare il capitale economico. Date le caratteristiche dei rischi operazionali una simulazione bottom-up sembra l'appoccio ideale. Ernst e Young hanno applicator gli ABM per misurare i rischi manageriali alla società Societe´ Generale Asset Management (SGAM). Hanno cosi iniziato a modellare processi di buisness servendosi delle sezioni di lavoro come agenti identificandone le caratteristiche

nelle attività, nelle interazioni fra esse e nella percentuale di operazioni pericolose affidate alle stesse. In altre parole gli ABM risultano perfetti non solo per stimare i rischi operazionali nelle istituzioni finanziarie ma per modellare i rischi di un'organizzazione in generale.

Capitolo 9

Game of life

L'esempio più conosciuto di automa cellulare è quello di "Game of Life", conosciuto più semplicemente con il nome di Life, ideato dal matematico britannico John H. Conway nel 1970.

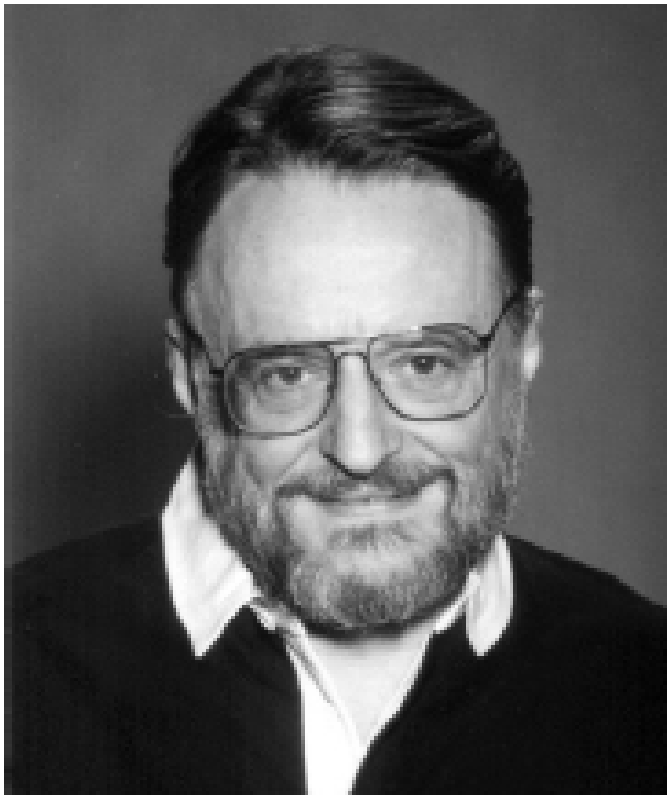


Figura 9.1: John Horton Conway

Il gioco è uno "zero-player game", intendendo con questo che l'evoluzione

del gioco è determinata dal suo stato iniziale e non richiede interazioni con un utente. L'unica interazione che si ha con Life è quella di creare una configurazione iniziale ed osservarne l'evoluzione. Il gioco non è altro che un automa bidimensionale sviluppato su griglia ortogonale, nella quale ogni cella può avere uno di due possibili stati: vita o morte. Dichiariando viva la cella colorata di nero e morta la cella non colorata, ognuna di esse interagisce con i propri 8 vicini seguendo queste regole:

- Una cella viva muore se all'istante t ha meno di 2 vicini o più di 3 vicini vivi.
- Una cella viva con esattamente 2 o 3 vicini vivi sopravvive alla prossima generazione.
- Una cella "nasce" (passa da morta a viva) se è circondata da esattamente 3 celle vive.

Gli stati delle celle vengono controllati contemporaneamente per tutte le celle ad ogni istanza di tempo. L'evoluzione del gioco dipende quindi unicamente dallo stato iniziale dell'automa.

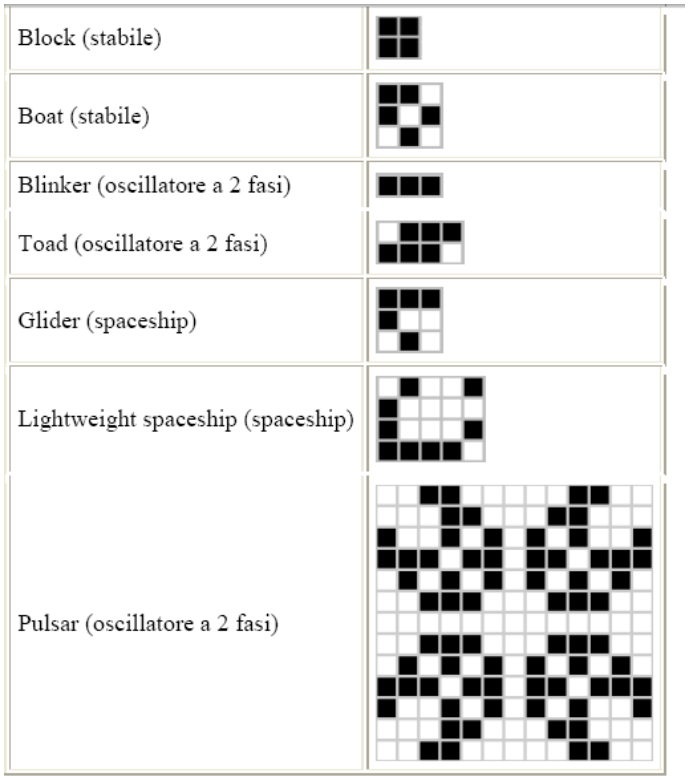
9.1 Le origini del gioco

Game of Life nacque da una semplificazione da parte del matematico Conway del modello che Von Neumann aveva utilizzato negli anni 1940 per il suo costruttore universale. Il successo del gioco avvenne con la sua pubblicazione nella rivista di divulgazione scientifica American Scientific nell'ottobre 1970 nella sezione Giochi Matematici di Martin Gardner. Come scrisse lo stesso Gardner: "Il gioco ideato da Conway divenne istantaneamente famoso, ed aprì inoltre un nuovo campo nella ricerca della matematica: il campo degli automi cellulari... " Anche molto tempo dopo la sua pubblicazione il gioco suscitava grandi attenzioni a causa delle sorprendenti vie nelle quali le configurazioni si evolvevano. Il gioco fece quindi da pioniere nella la scoperta dell'importanza degli automi cellulari. E' da sottolineare che quando il gioco fu pubblicato l'evoluzione dell'automa era tracciata a mano su carta (rendendo il lavoro estenuante) ma la fortuna di Conway fu proprio l'introduzione nel mercato di una nuova gerazione di minicomputer economici che permise a chiunque di farlo girare per ore sulle proprie macchine.

9.2 Esempi di configurazione

Esistono molti tipi di configurazioni possibili in Life, che includono configurazioni stabili ("still lives"), periodiche/oscillatorie, e configurazioni che si auto-traslano (Spaceships) attraverso lo spazio della griglia restando invariate. Vediamone alcuni esempi:

Il numero di configurazioni stabili di n celle con n = 1,2,3..... sono 0, 0, 0,



2, 1, 5, 4, 9, 10, 25, 46, 240, 619, 1353... I più diffusi sistemi oscillatori sono di periodo 2, ma si sono trovati casi di oscillatori di periodo 4, 8, 14, 15, 30 e altri più rari. La configurazione "Diehard" si evolve in maniera casuale fino a

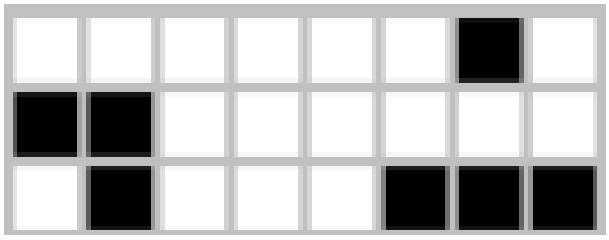


Figura 9.2: Diehard

svanire del tutto dopo ben 130 generazioni. "Acorn" impiega 5206 generazioni per stabilizzarsi in un sistema di 25 Spaseship e altrettanti oscillatori. Questi e altri sono esempi di configurazioni e proprietà intrinseche del gioco ancora

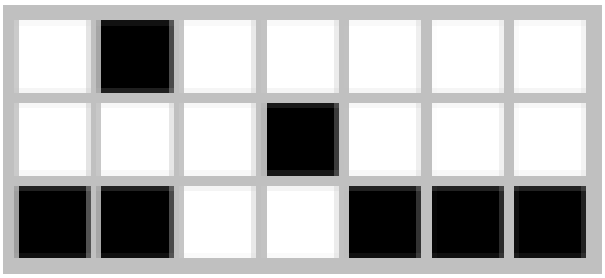


Figura 9.3: Acorn

oggetto di ricerca. Conway originariamente congetturò che nessuna configurazione iniziale potesse crescere indefinitivamente all'infinito: in altre parole nessun pattern in input con un numero finito di celle vive potesse crescere al disopra di un limite superiore finito. Egli offrì un premio di 50 dollari a chi riuscisse a dimostrare che questa congettura fosse falsa entro la fine del 1970. Un modo per dimostrare la falsità della sua congettura era quello di trovare patterns che "aggiungessero" continuamente altri pattern alla griglia: un "gun" (fucile) vorrebbe essere quella configurazione che "spara" ripetutamente oggetti che si muovono per la griglia (spaceship) mentre un "puffer train" (treno a vapore) la configurazione che lascia dietro di se "fumo" o tracce di detriti persistenti. La sfida fu vinta nel novembre dello stesso anno da un gruppo di lavoro dell MIT condotto da Bill Gosper: il Gosper Gun produceva la sua prima struttura auto-traslante spaceship dopo 15 iterazioni e un'altra nuova ogni 30. Il Gosper Gun è tutt'oggi la configurazione conosciuta di tipo "gun" più piccola. I pattern che emergono da regole semplici sono considerate una forma di bellezza. Piccoli sub-pattern isolati senza alcuna simmetria iniziale hanno una naturale tendenza intrinseca a diventare simmetrici, una volta che ciò accade la simmetria tende a diventare sempre più ricca e non verrà mai persa se non per il contatto con altri sub-pattern vicini. Che sia per puro caso fortuito o per intuito, il gioco che nacque dalla mente di Conway si rivelò un vero e proprio successo ed è forse grazie ad esso se si sono introdotti con successo modelli cellulari come strumento per la simulazione.