DIPARTIMENTO D
INGEGNERIA
AEROSPAZIALE

UNIVERSITÀ DEGLI STUDI DI PISA

FACOLTÀ DI INGEGNERIA

DIPARTIMENTO DI
INGEGNERIA
AEROSPAZIALE

Dipartimento di Ingegneria Aerospaziale

Generazione di configurazioni aerodinamiche mediante NURBS (Non-Uniform Rational B-Splines)

Relatori:

Prof. Aldo Frediani

Prof. Tullio Franzoni

Dott. Marco Franciosi

Candidati:

Andrea Rimondi

Anno Accademico 2002/2003

Indice

Indice 3
Indice delle figure 5
Sommario 8
Introduzione 9
Capitolo 1 11
1. Introduzione 11
1.1. La rappresentazione implicita e la rappresentazione parametrica 11
1.2. Le forme parametriche 13
1.2.1. Polinomi 13
1.2.2. Le curve di Bézier 14
1.2.3. Le curve Rational-Bézier 17
1.3. Le B-Spline 20
1.3.1. Premessa 20
1.3.2. Le funzioni di base di una curva B-Spline 21
1.3.3. Derivate di una funzione di base B-Spline 22
1.3.4. Curve B-Spline 23
1.3.5. Derivata di una B-Spline 25
1.3.6. Superfici B-Spline 25
1.3.7. Derivata di una superficie B-Spline 28
1.4. Le NURBS 28
1.4.1. Derivata di una NURBS 30
1.4.2. Superfici NURBS 31
1.5. Potenzialità delle NURBS 32
Capitolo 2 37
2. Interpolazione tramite NURBS 37
2.1. Premessa 37
2.2. Parametrizzazione di una curva 37
2.3. Inserimento di un nodo 40
2.3.1. Inserimento di un nodo in una curva 40
2.3.2. Inserimento di un nodo in una superficie 42
2.4. L'interpolazione globale 43
2.4.1. Interpolazione di curve 43

2.4.2. Interpolazione globale con derivate agli estremi assegnate 49
2.4.3. Interpolazione globale di superfici 51
2.5. L'interpolazione locale 53
2.5.1. Interpolazione di curve 53
2.5.2. Curve interpolanti con derivate agli estremi assegnate 59
2.5.3. Interpolazione di superfici 61
2.5.4. Superfici interpolanti con derivate agli estremi assegnate 66
2.5.5. Interpolazione con superfici lineari-cubiche 69
Capitolo 3 72
3. Intersezione di superfici 72
3.1. Premessa 72
3.2. Point inversion 74
3.2.1. Point inversion : curva 74
3.2.2. Point inversion : superficie 75
3.3. Intersezione di superfici : subdivision method 77
3.3.1. Suddivisione di curve 81
3.3.2. Suddivisione di superfici 82
3.3.3. Calcolo dei parallelepipedi contenenti le superfici 82
3.3.4. Test di planarità della superficie 83
3.3.5. Calcolo dell'intersezione di due piani 83
3.3.6. Ordinamento dei punti dell'intersezione 83
3.3.7. Esempi di intersezioni 84
3.4. Intersezione di superfici : lattice evaluation 87
4. Conclusioni 90
5. Appendice 91
Derivata di una funzione di base B-Spline 91
Bibliografia 94
Ringraziamenti 95

Indice delle figure

Figura 1-1 Polinomi di Bernstein per n=3 16
Figura 1-2 Esempi di curve di Bézier: in a) si vede come si possa ottenere una curva chiusa semplicemente facendo coincidere il primo e l'ultimo punto, in b) è mostrato un cappio; in tutti i casi la curva alle estremità è tangente al poligono di controllo 17
Figura 1-3 Interpretazione geometrica delle coordinate omogenee (a) arco di circonferenza, b) curva generica) 20
Figura 1-4 Funzioni base B-Spline di secondo grado definite sul vettore dei nodi U=[0,0,0,1,2,3,4,4,5,5,5] 22
Figura 1-5 Rappresentazione delle funzioni base di una superficie B-Spline 26
Figura 1-6 NURBS creata mediante 7 punti di controllo 33
Figura 1-7 Superficie NURBS con funzioni di grado diverso nelle due direzioni 34
Figura 1-8 Arco di circonferenza rappresentato mediante NURBS 35
Figura 1-9 Elica 3D rappresentata mediante NURBS 35
Figura 1-10 Derivate di una curva NURBS 35
Figura 1-11 Derivata di una superficie NURBS 35
Figura 1-12 Esempio di superficie costruita partendo da due curve NURBS con interpolazione lineare 36
Figura 1-13 Sfera e toroide ottenuti per rivoluzione di semplici forme bidimensionali 36
Figura 2-1 Riparametrizzazione di una curva 38
Figura 2-2 Esempio di interpolazione globale 46
Figura 2-3 Funzioni di base $N_{i,3}$ definite sul vettore di nodi $U = \left\{0,0,0,0,\frac{28}{51},1,1,1,1\right\}$ 47
Figura 2-4 Esempio di interpolazione globale 48
Figura 2-5 interpolazione globale libera 49
Figura 2-6 modifica della posizione del secondo punto di controllo per ottenere derivata iniziale orizzontale. La curva non interpola più il secondo punto 50
Figura 2-7 modifica della posizione del secondo punto di controllo dopo avere aggiunto un nodo per ottenere derivata iniziale orizzontale. La curva interpola ancora il secondo punto. 50
Figura 2-8 Interpolazione globale con derivate agli estremi assegnate. 51
Figura 2-9 Esempio di interpolazione globale con punti allineati 51

Figura 2-10 Procedura per la creazione di una superficie interpolante B-Spline mediante interpolazione globale. 53
Figura 2-11 Punti di controllo di un segmento di Bézier 54
Figura 2-12 Confronto fra l'interpolazione globale e l'interpolazione locale in presenza di punti allineati 57
Figura 2-13 Esempio di interpolazione di curva chiusa, in questo caso, un'approssinazione dei una circonferenza mediante B-Spline cubica 57
Figura 2-14 Confronto fra un'approssimazione della circonferenza mediante interpolazione di 4 punti mediante B-Spline cubica e effettiva circonferenza. 58
Figura 2-15 Andamento della curvatura in funzione del parametro *u* lungo la curva. 58
Figura 2-16 Esempio di interpolazione di un profilo supercritico 59
Figura 2-17 Esempio di interpolazione di un profilo sottile 59
Figura 2-18 a) Interpolazione locale senza specificare derivate al contorno; b) c) d) interpolazione locale con derivate al contorno assegnate, vari casi. 60
Figura 2-19 60
Figura 2-20 Raccordo di due curve NURBS con continuità G^1 mediante imposizione di derivate agli estremi 61
Figura 2-21 Procedura per la creazione di una superficie interpolante B-Spline mediante interpolazione locale. 65
Figura 2-22 Esempio di interpolazione mediante B-Spline in caso di curva chiusa in una direzione. 65
Figura 2-23 Esempio di interpolazione mediante B-Spline in caso di curva chiusa in entrambe le direzione. 65
Figura 2-24 Esempio di interpolazione mediante B-Spline in caso di punti complanari. 66
Figura 2-25 Interpolazione mediante B-Spline senza specificare alcuna derivata iniziale 67
Figura 2-26 Interpolazione mediante B-Spline con derivate parziali assegnate lungo il bordo.. 68
Figura 2-27 Smooth stitch: superfici originali, nessuna derivata specificata. 68
Figura 2-28 Smooth stitch: la superficie a destra è stata modificate per avere la stesa derivata della superficie a sinistra, rimasta originale. 68
Figura 2-29 Smooth stitch: dettaglio. 69
Figura 2-30 Interpolazione bicubica, forma in pianta ottenuta. 69
Figura 2-31 Interpolazione bicubica: ala con diedro. 70
Figura 2-32 Interpolazione lineare-cubica: forma in pianta 70

Figura 2-33 Interpolazione lineare-cubica: ala con diedro. 71
Figura 2-34 Interpolazione bicubica: ala con diedro freccia. 71
Figura 3-1 Algoritmo di Point inversion, risultati. 75
Figura 3-2 Algoritmo di Point inversion, risultati. 77
Figura 3-3 procedura di suddivisione di una curva 81
Figura 3-4 suddivisione di una superficie, punti di controllo originali (giallo) e punti di controllo delle superfici ottenute (blu e rosso) 82
Figura 3-5 Intersezione cilindro-cilindro: risultati nello spazio e nel piano dei parametri. 85
Figura 3-6 Intersezione toro-toro: risultati nello spazio e nel piano dei parametri. 85
Figura 3-7 Intersezione cilindro -toro: risultati nello spazio e nel piano dei parametri. 86
Figura 3-8 Intersezione Ala - Fusoliera: risultati nello spazio. 87
Figura 3-9 Esempio di superfici intersecate mediante il metodo *lattice evaluation*:superfici da intersecare 88
Figura 3-10 Esempio di superfici intersecate mediante il metodo *lattice evaluation*: risultato dell'intersezione 89

Sommario

Questo lavoro si propone di esaminare varie forme di rappresentazione matematica di curve e superfici al fine di individuare prima, e di utilizzare poi, la più opportuna rappresentazione che consenta di descrivere geometrie complesse quali ali e fusoliere e corpi aerodinamici in genere.

Nel primo capitolo vengono illustrate diverse rappresentazioni, dalle più semplici quali la rappresentazione implicita e la rappresentazione parametrica, a quelle via via più complesse quali curve di Bézier e B-Spline, fino ad arrivare alla rappresentazione mediante NURBS (Non-Uniform Rational B-Spline). Di ognuna di queste rappresentazioni vengono anche illustrate le proprietà fondamentali e i principali limiti.

Nel secondo capitolo si affronta il primo problema fondamentale della modellazione di curve e superfici: l'interpolazione. Vengono qui presentati alcuni algoritmi di interpolazione illustrandone pro e contro mediante esempi applicati a casi pratici. All'interno di questo capitolo vengono anche descritti alcuni algoritmi fondamentali per la manipolazione delle curve NURBS.

Nel terzo capitolo si affronta il problema dell'intersezione di curve e di superfici. Viene presentata una breve panoramica dei principali metodi descritti in letteratura e viene sviluppato un algoritmo di tipo "divide and conquer" per la soluzione pratica del problema dell'intersezione. Questo algoritmo viene poi utilizzato per scrivere due procedure per il calcolo dell'intersezione. All'interno del capitolo vengono presentati alcuni algoritmi fondamentali per la modifica delle superfici NURBS.

Introduzione

Le moderne tecniche di progettazione volte alla riduzione dei tempi e all'aumento della qualità dei prodotti, si rivolgono sempre più di frequente a rappresentazione tridimensionale su calcolatori. Nell'ambito della progettazione aeronautica, l'introduzione di codici di calcolo fluidodinamico estremamente evoluti, unitamente all'aumento della potenza di calcolo degli stessi calcolatori, ha permesso di rendere disponibile a costi accettabili la simulazione aerodinamica di intere configurazioni di aeromobili. Il problema del tempo di calcolo si è quindi spostato sul tempo di realizzazione del modello 3D. Recentemente presso l'Università di Pisa è stato sviluppato un codice di calcolo specificatamente volto alla realizzazione di mesh di superficie di aeromobili, rivelandosi poi un ottimo codice capace di realizzare mesh particolarmente complesse anche di configurazioni diverse e molto complesse. Questo lavoro svolto ha dimostrato la possibilità di realizzare un pacchetto software in grado di modellare una superficie complessa, come quella di un aeroplano, in modo parametrico, rapido e modulare. Questo lavoro ha però messo in luce alcune lacune del codice, prima fra tutte l'interpolazione mediante spline cubiche, essendo queste curve fortemente limitate dall'uso di un sistema di riferimento fisso, e quindi incapaci di generare linee verticali rispetto all'asse di riferimento rispetto alle quali sono calcolate. La soluzione a questo problema pare essere l'utilizzo di curve di interpolazione più versatili delle spline cubiche, capaci di rappresentare mediante poche informazioni una curva complessa come un cappio o un cerchio. Queste curve vengono chiamate NURBS (acronimo di Non-Uniform-Rational-B-Spline) e sono già state introdotte con successo nel codice sviluppato presso il Dipartimento di Ingegneria Aerospaziale dell'Università di Pisa per modellare il raccordo ala-fusoliera. Lo scopo di questa tesi è quello di estendere l'uso di queste curve all'intero codice, sostituendo le spline cubiche con le NURBS. Il codice è stato scritto interamente in MATLAB. L'introduzione di queste curve, unitamente a una migliore conoscenza delle potenzialità del codice esistente ha condotto a focalizzare meglio gli obiettivi del progetto. È stata quindi messa da parte l'idea di avere un codice che generasse la mesh direttamente importabile in FLUENT, essendo questo un compito svolto meglio da generatori di mesh dedicati come GAMBIT o TGRID, preferendo sfruttare la capacità delle NURBS di rappresentare superfici complesse per definire la geometria esterna del corpo. Quindi l'obiettivo si è spostato a questo punto a scrivere un codice che sia in grado di descrivere in un formato importabile dai vari generatori di griglia partendo da dati propri del campo aeronautico (freccia, svergolamento ecc…). è stato deciso quindi di utilizzare una

rappresentazione dei dati in uscita in formato IGES, essendo questo un formato in grado di trasportare tra i diversi codici CAD le informazioni di superfici e curve in formato NURBS.

Il lavoro di tesi ha consentito di risolvere alcuni problemi che il codice MSD mostrava, primo fra tutti quello di avere una rappresentazione efficiente delle superfici aerodinamiche. Inoltre è stato possibile, mediante gli strumenti sviluppati durante questo lavoro, ottenere forme più complesse eliminando alcune limitazioni che impedivano il calcolo di intersezioni multiple in fusoliera.

Mediante gli strumenti sviluppati in questa tesi, è stato possibile aggiornare il codice MSD per rappresentare l'intera configurazione dell'aeromobile mediante NURBS, rendendo così possibile un risparmio di tempo di calcolo considerevole e ottenere una rappresentazione accurata con una mole di dati notevolmente inferiore. Questa ultima si ripercuote anche sul salvataggio e l'esportazione; nonostante non sia ancora stato scritto un filtro si esportazione IGES, le stime sulla dimensione del file esportato sono molto positive, generando per la fusoliera un file di circa 110Kb invece che diversi Mb. Il formato di esportazione risulta inoltre compatibile con i più diffusi software CAD e generatori di griglia.

Gli sviluppi futuri del codice includono oltre alla messa a punto di un filtro IGES, anche un miglioramento delle procedure di intersezione, per renderle più robuste e capaci di intersecare superfici generiche in tempi ridotti. Inoltre è necessario sviluppare un'interfaccia grafica più "user-friendly", unitamente a un manuale operativo per consentire a un utente di comprendere il funzionamento del codice e poterlo utilizzare indipendentemente.

Capitolo 1

1. Introduzione

Questo lavoro si propone di utilizzare le curve di interpolazione NURBS per la generazione della configurazione completa di un velivolo.

Nel seguito di questo capitolo verranno esposte le basi matematiche sulle quali si fonda la rappresentazione spaziale dei corpi, a partire dalle strutture più semplici fino alle NURBS, la cui trattazione sarà oggetto del successivo capitolo.

La più semplice forma di interpolazione è l'interpolazione tramite spline cubiche in forma esplicita, costituita da un polinomio cubico a tratti con continuità C^1 interna e con derivate assegnate agli estremi. Questa forma è presente nel codice Matlab, in cui viene richiamata mediante il comando spline. I problemi principali della rappresentazione in forma esplicita sono, come noto, l'impossibilità di descrivere tratti di curva verticali e la presenza di forti errori numerici per tratti di curve con derivate prime molto elevate (tratti quasi verticali). Per ovviare a questo inconveniente si possono prendere in considerazione 2 alternative, ovvero la rappresentazione implicita e la rappresentazione parametrica, descritte nel paragrafo successivo.

1.1. La rappresentazione implicita e la rappresentazione parametrica

Per rappresentazione implicita di una curva si intende una descrizione del tipo $f(x,y)=0$. Per ogni data curva esiste una unica rappresentazione in forma implicita a meno di una costante moltiplicativa.

Nella rappresentazione parametrica, invece, la curva è rappresentata da una funzione vettoriale del tipo

$$C(u)=[x(u),y(u)] \text{ dove } a \le u \le b \qquad \textbf{(1-1)}$$

L'intervallo *[a,b]* è arbitrario, e per questo di solito viene normalizzato a *[0,1]*.

Le due forme ora descritte hanno pregi e difetti. In generale valgono le seguenti proprietà:

1. Aggiungendo una coordinata *z(u)*, la forma parametrica $C(u)=[x(u),y(u),z(u)]$ descrive una curva nello spazio, mentre la forma implicita è adatta a descrivere solo curve su un piano (*xy, xz, yz)*

2. Risulta piuttosto complicato descrivere curve limitate o pezzi di superfici mediante descrizione implicita, mentre risulta immediato in forma parametrica con la semplice limitazione dell'intervallo di variazione del parametro *u*. D'altra parte, però, descrivere una curva nono limitata, come ad esempio una retta è agevole in forma implicita (*f(x,y)=ax+by+c*) mentre è più difficile da implementare in forma parametrica.
3. Le curve parametriche hanno una direzione naturale di percorrenza, da *C(a)* a *C(b)*, che le curve implicite non hanno, rendendo facile la generazione di punti lungo una curva o superficie.
4. La forma parametrica è più naturale per quanto concerne la rappresentazione di forme su un computer. I coefficienti di molte forme parametriche (Bézier e B-Spline, ad esempio) hanno un notevole significato geometrico. Questo si traduce in un modo intuitivo di progettare la curva e in algoritmi stabili e affidabili.
5. La complessità di molte operazioni geometriche dipende dalla rappresentazione scelta. Ad esempio, calcolare un punto su una superficie è facile in forma parametrica e più complesso in forma implicita, mentre determinare se un punto giace o no su una curva è immediato in forma implicita mentre è più complesso in forma parametrica.
6. Nelle rappresentazioni in forma parametrica a volte si deve convivere con anomalie proprie della rappresentazione, indipendenti dalla geometria; ad esempio in una sfera rappresentata in forma parametrica come:

   $S(u,v) = [X(u,v), Y(u,v), Z(u,v)]$

   dove

   $$\begin{cases} X(u,v) = \sin(u)\cos(v) \\ Y(u,v) = \sin(u)\sin(v) \\ Z(u,v) = \cos(u) \end{cases}$$

   $0 \le u \le \pi$

   $0 \le v \le 2\pi$

   se calcoliamo la normale alla superficie così definita come

   $$N = \frac{S_u \times S_v}{|S_u \times S_v|}$$

   dove × rappresenta l'operazione di prodotto vettoriale

   si nota subito come questa non possa essere calcolata per ogni punto *S(u,v)* con $0 \le v \le 2\pi$ e $u=0$ o $u=\pi$. L'impossibilità di calcolare la normale uscente dai poli

di una sfera è dovuta alla parametrizzazione scelta, essendo la normale una proprietà geometrica definita per ogni punto della sfera.

I problemi di interesse in questo lavoro riguardano la generazione di configurazioni dimensioni finite, utilizzando il computer e in cui è importane che i coefficienti che definiscono le curve abbiano un significato geometrico Per tali motivi nel seguito si farà riferimento solo a rappresentazioni parametriche delle curve e superfici.

1.2. Le forme parametriche

Tra le diverse rappresentazioni parametriche disponibili, vengono ora esaminate quelle che rispondono al meglio alle esigenze in esame, quali ad esempio la possibilità di implementazione sul calcolatore in maniera efficiente, la possibilità di rappresentare le forme di interesse e la possibilità di modificare a posteriori parti della curva lasciando invariato il resto della curva stessa. In particolare, il campo viene ristretto alla classe di rappresentazioni che :

1. riescono a descrivere con accuratezza numerica tutte le curve di interesse quali rette, curve continue e spezzate e superfici aerodinamiche;
2. siano facilmente e efficientemente manipolabili all'interno di un computer e in particolare:
    a. il calcolo di punti e derivate sia agevole,
    b. il calcolo sia abbastanza insensibile a errori di arrotondamento e di troncamento,
    c. le funzioni richiedano poca memoria per essere salvate;
3. siano semplici e matematicamente ben conosciute.

1.2.1. Polinomi

Una classe semplice di funzioni che soddisfa i criteri 2 e 3 sono i polinomi che tuttavia non soddisfano in requisito 1, in quanto esistono molte ed importanti funzioni che non possono essere descritte mediante polinomi, come ad esempio le coniche. In questi casi le curve possono essere ottenute in forma approssimata.

Una curva polinomiale può essere espressa come somma di monomi o come curve di Bézier. Le due rappresentazioni sono equivalenti da un punto di vista matematico generale, ma, dal punto di vista pratico, le curve di Bézier sono molto più adatte a rappresentazioni mediante l'uso del calcolatore, come verrà mostrato in seguito.

Una somma di potenze di grado *n* è data da:

$$C(u) = [x(u), y(u), z(u)] = \sum_{i=0}^{n} A_i u^i \quad \text{dove } 0 \le u \le 1 \qquad \textbf{(1-2)}$$

e dove A è la matrice dei coefficienti, e u è il parametro (variabile indipendente). La (1-2) può essere scritta per componenti come segue:

$$\begin{aligned} x(u) &= \sum_{i=0}^{n} \alpha_i u^i = \alpha_0 u^0 + \alpha_1 u^1 + \alpha_2 u^2 + \ldots + \alpha_{n-1} u^{n-1} + \alpha_n u^n \\ y(u) &= \sum_{i=0}^{n} \beta_i u^i = \beta_0 u^0 + \beta_1 u^1 + \beta_2 u^2 + \ldots + \beta_{n-1} u^{n-1} + \beta_n u^n \\ z(u) &= \sum_{i=0}^{n} \gamma_i u^i = \gamma_0 u^0 + \gamma_1 u^1 + \gamma_2 u^2 + \ldots + \gamma_{n-1} u^{n-1} + \gamma_n u^n \end{aligned} \qquad \textbf{(1-3)}$$

che a sue volta si presta a una rappresentazione più compatta in forma matriciale:

$$C(u) = [A] \cdot \begin{bmatrix} u^0 \\ u^1 \\ \ldots \\ u^n \end{bmatrix} = [A_i]^T [u^i] \qquad \textbf{(1-4)}$$

Le funzioni u^i (= *1, u, u*^{2}*, u*^{3}*, ...*) sono chiamate base.

1.2.2. Le curve di Bézier

Un'altra rappresentazione parametrica di una curva è la rappresentazione di Bézier, che è una particolare somma di potenze.

La rappresentazione di Bézier di grado n di una curva data è del tipo:

$$C(u) = \sum_{i=0}^{n} B_{i,n}(u) P_i \qquad \text{dove } 0 \le u \le 1$$

ovvero, in forma scalare

$$\begin{aligned} x(u) &= \sum_{i=0}^{n} B_{i,n}(u) x_i \\ y(u) &= \sum_{i=0}^{n} B_{i,n}(u) y_i \\ z(u) &= \sum_{i=0}^{n} B_{i,n}(u) z_i \end{aligned} \qquad \textbf{(1-5)}$$

dove P_i sono n+1 punti di controllo e le $B_{i,n}(u)$ sono le "funzioni base" costituite dai classici polinomi di Bernstein:

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \qquad \textbf{(1-6)}$$

L'unione dei punti di controllo costituisce il "poligono di controllo"; i punti di tale poligono sono scelti in funzione della curva da rappresentare e sono in numero pari alle

funzioni di Bernstein (*n+1*). Queste funzioni posseggono alcune importanti proprietà che elenchiamo di seguito:

i. $B_{i,n}(u) \geq 0$ per ogni i,n e $0 \leq u \leq 1$

ii. $\sum_{i=0}^{n} B_{i,n}(u) = 1$ ogni u in $0 \leq u \leq 1$

iii. $B_{0,n}(0) = B_{n,n}(1) = 1$

iv. $B_{i,n}(u)$ ha uno e un solo massimo su [0,1], cioè in $u = i/n$

v. per ogni n, i polinomi $B_{i,n}(u)$ sono simmetrici rispetto a $u = ½$

vi. $B_{i,n}(u) = (1-u)B_{i,n-1}(u) + uB_{i-1,n-1}(u)$; definiamo $B_{i,n}(u) = 0$ se $i<0$ o $i>n$

inoltre, derivando la (1-5) si ottiene:

$$C'(u) = \sum_{i=0}^{n} B'_{i,n}(u) P_i$$

essendo:

$$B'_{i,n}(u) = \frac{dB_{i,n}(u)}{du} = n\left(B_{i-1,n-1}(u) - B_{i,n-1}(u)\right)$$

con:

$$B_{-1,n-1}(u) \equiv B_{n,n-1}(u) \equiv 0 \qquad \textbf{(1-7)}$$

quindi:

$$C'(u) = \sum_{i=0}^{n} n\left(B_{i-1,n-1}(u) - B_{i,n-1}(u)\right)P_i = n\sum_{i=0}^{n-1} B_{i,n-1}(u)\left(P_{i+1} - P_i\right)$$

ovvero:

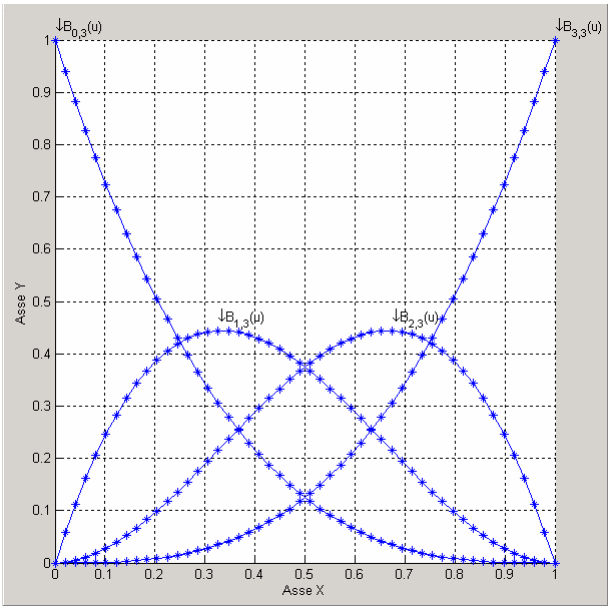
$$C'(0) = n(P_1 - P_0)$$
$$C'(1) = n(P_n - P_{n-1})$$

ne deriva che la curva *C(u)* è tangente al poligono di controllo nei punti iniziale e finale.

Le curve di Bézier hanno altre interessanti caratteristiche; per esempio è possibile, mantenendo inalterata la curva iniziale, aumentarne il grado, spezzarla in due curve indipendenti, unire due curve con continuità prescritta, calcolarne agevolmente la derivata in qualunque punto, aggiungere punti intermedi.

La rappresentazione di Bézier richiede l'infittimento dei punti di controllo in zone con elevata curvatura, con il conseguente innalzamento del grado della curva stessa; inoltre non è in grado di rappresentare le coniche e, in particolare la circonferenza in forma esatta. Atal fine vengono definite le curve Rational-Bézier.

Le figure sottostanti riportano alcuni esempi di curve di polinomi di Bernstein.



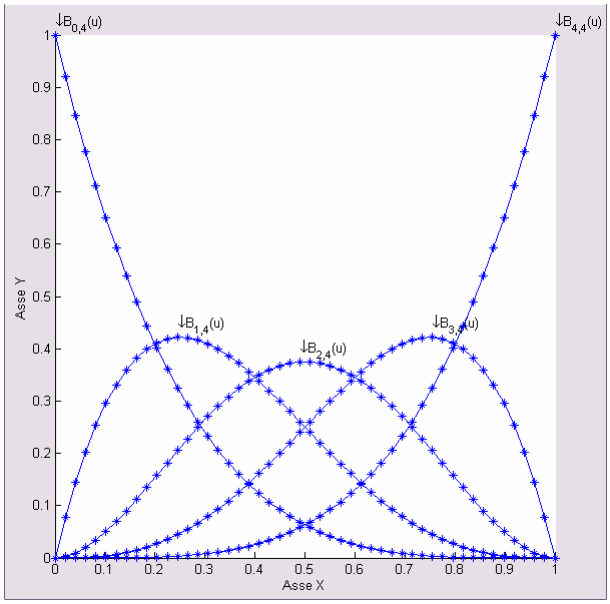
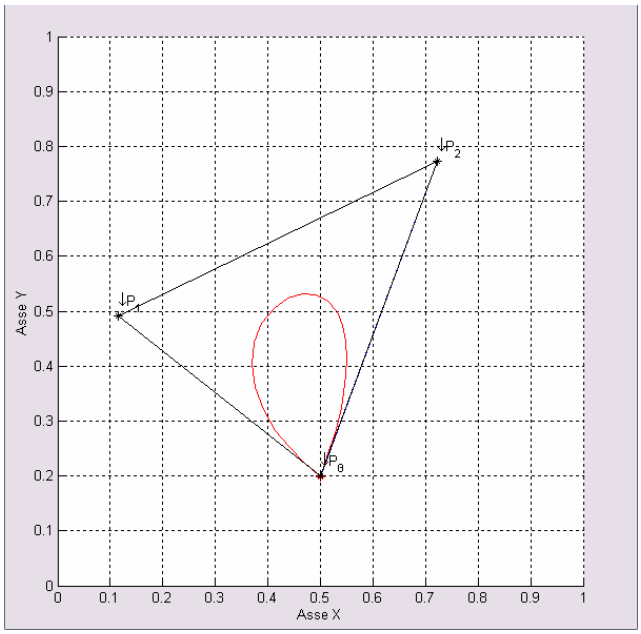
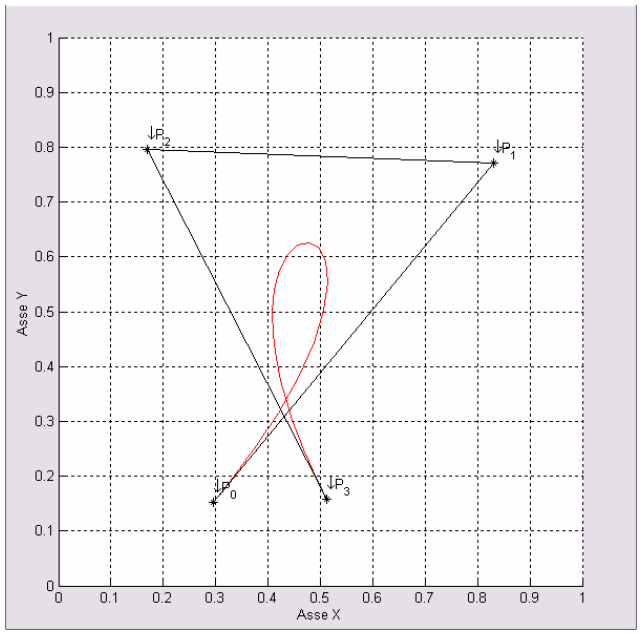


Figura 1-1 Polinomi di Bernstein per n=3 Polinomi di Bernstein per n=4

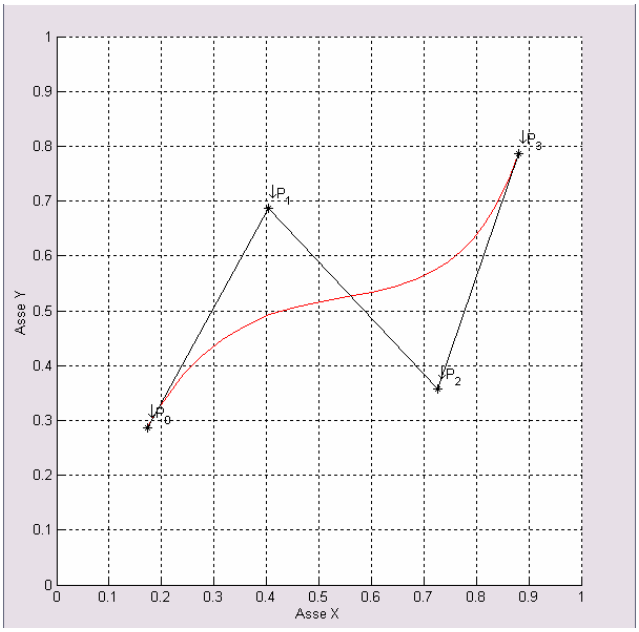
Di seguito sono riportate alcune curve di Bézier di grado 3 (n=3).



a)



b)



c)

Figura 1-2 Esempi di curve di Bézier: in a) si vede come si possa ottenere una curva chiusa semplicemente facendo coincidere il primo e l'ultimo punto, in b) è mostrato un cappio; in tutti i casi la curva alle estremità è tangente al poligono di controllo

1.2.3. Le curve Rational-Bézier

Le coniche possono essere rappresentate in forma esatta mediante funzioni polinomiali fratte.

Dalla matematica classica è noto che ogni conica può essere rappresentata come rapporto di polinomi del tipo:

$$x(u)=\frac{X(u)}{W(u)} \quad \text{e} \quad y(u)=\frac{Y(u)}{W(u)} \qquad \textbf{(1-8)}$$

dove *X(u), Y(u), W(u)* sono polinomi.

Ad esempio in Tabella 1-1 sono illustrate alcune forme razionali di comuni coniche.

$x(u)=\frac{1-u^2}{1+u^2}$ e $y(u)=\frac{2u}{1+u^2}$	cerchio unitario centrato nell'origine
$x(u)=\frac{1-u^2}{1+u^2}$ e $y(u)=\frac{4u}{1+u^2}$	ellisse centrato nell'origine con semiasse maggiore y pari a 2 e semiasse minore x pari a 1
$x(u)=\frac{-1+2u}{1+2u-2u^2}$ e $y(u)=\frac{4u(1-u)}{1+2u-2u^2}$	Iperbole con centro in $(0,{}^4/_3)$

$x(u)=u$ e $y(u)=u^2$	Parabola con vertice nell'origine e y asse di simmetria

Tabella 1-1 varie rappresentazioni razionali di coniche

Tali luoghi posso essere rappresentate da una curva di Bézier definita in questo modo:

$$\mathrm{C}(\mathrm{u})=\frac{\sum_{i=0}^{n} B_{i,n}(u)w_i P_i}{\sum_{j=0}^{n} B_{j,n}(u)w_j} \quad 0\le u\le 1 \qquad \textbf{(1-9)}$$

in cui i P_i e i $B_{i,n}(u)$ sono quelli definiti in precedenza e i w_i sono scalari chiamati "pesi", che supporremo positivi (per motivi di stabilità numerica degli algoritmi). In questa definizione, quindi, il denominatore comune è dato da $\mathrm{W}(\mathrm{u})=\sum_{j=0}^{n} B_{j,n}(u)w_j$ il quale, per le ipotesi fatte sui pesi, è sempre positivo per $0\le u\le 1$.

La (1-9), in analogia alla (1-5), può essere scritta nel modo seguente:

$$C(u)=\sum_{i=0}^{n} R_{i,n}(u)P_i \quad , \quad 0\le u\le 1 \qquad \textbf{(1-10)}$$

dove:

$$R_{i,n}(u)=\frac{B_{i,n}(u)w_i}{\sum_{j=0}^{n} B_{j,n}(u)w_j} \qquad \textbf{(1-11)}$$

Le proprietà di queste funzioni sono le stesse dei polinomi di Bernstein da cui derivano. Inoltre dalla (1-11) possiamo ricavare la seguente proprietà:

vii. se $w_i = 1$ per ogni i, allora $R_{i,n}(u) = B_{i,n}(u)$ per ogni i; quindi le curve di Bézier sono sottocasi delle curve Rational-Bézier

queste proprietà matematiche hanno importanti conseguenze geometriche[1]:

i. guscio convesso: le curve sono contenute nel guscio convesso definito dai punti di controllo P_i

ii. invarianza alle trasformazioni: per ruotare, traslare o scalare una curva di questo tipo è sufficiente applicare la trasformazione ai suoi punti di controllo

iii. proprietà di diminuzione della variazione: nessuna retta interseca la curva più volte di quante interseca il poligono di controllo.

[1] Per le dimostrazioni vedere [14]

iv. Interpolazione agli estremi: $C(0) = P_0$ e $C(1) = P_n$

v. La derivata *k*-esima in u = 0 (u = 1) dipende dai primi (ultimi) *k*+1 punti di controllo e pesi; in particolare C'(0) e C'(1) sono paralleli a $P_1 - P_0$ e $P_n - P_{n-1}$ rispettivamente

Le curve di cui alle espressioni (1-10) e (1-11) possono essere ottenute attraverso una procedura che presenta potenziali ventaggi a livello di implementazione. L'idea è di usare coordinate omogenee per rappresentare una curva razionale in *n*-dimensioni come una curva polinomiale in *n+1*-dimensioni. Preso un punto in uno spazio tridimensionale euclideo P = (*x,y,z*), scriviamo tale punto come P^w = (*wx,wy,wz,w*) = (*X,Y,Z,W*) in uno spazio 4-dimensionale, con $w \neq 0$. Ovviamente, possiamo ottenere P dividendo ogni coordinata di P^w per *w* e scartando l'ultima coordinata; tale operazione ha il significato geometrico di mappare la curva 4-dimensionale sull'iperpiano W=1 (esempi sono illustrati nella Figura 1-3 in cui sono tracciate 2 curve con $P_0 = (0,1,1)$, $P_1 = (1,1,1)$, $P_2 = (1,0,2)$, interpretando la prima volta come curva 3D e la seconda volta come curva 2D in coordinate omogenee dove la terza dimensione rappresenta i pesi). Chiamiamo H questa operazione di mappatura prospettica con centro nell'origine e la scriviamo come segue:

$$P = H\left\{P^w\right\} = H\left\{\left(X,Y,Z,W\right)\right\} = \begin{cases} \left(\frac{X}{W},\frac{Y}{W},\frac{Z}{W}\right) \Leftrightarrow W \neq 0 \\ versore\left(X,Y,Z\right) \Leftrightarrow W = 0 \end{cases} \qquad \textbf{(1-12)}$$

Quindi definire $C^w(u) = \sum_{i=0}^{n} B_{i,n}(u) P_i^w$ poi applicare la mappatura *H* conduce alle stesse espressioni in (1-10) e (1-11).

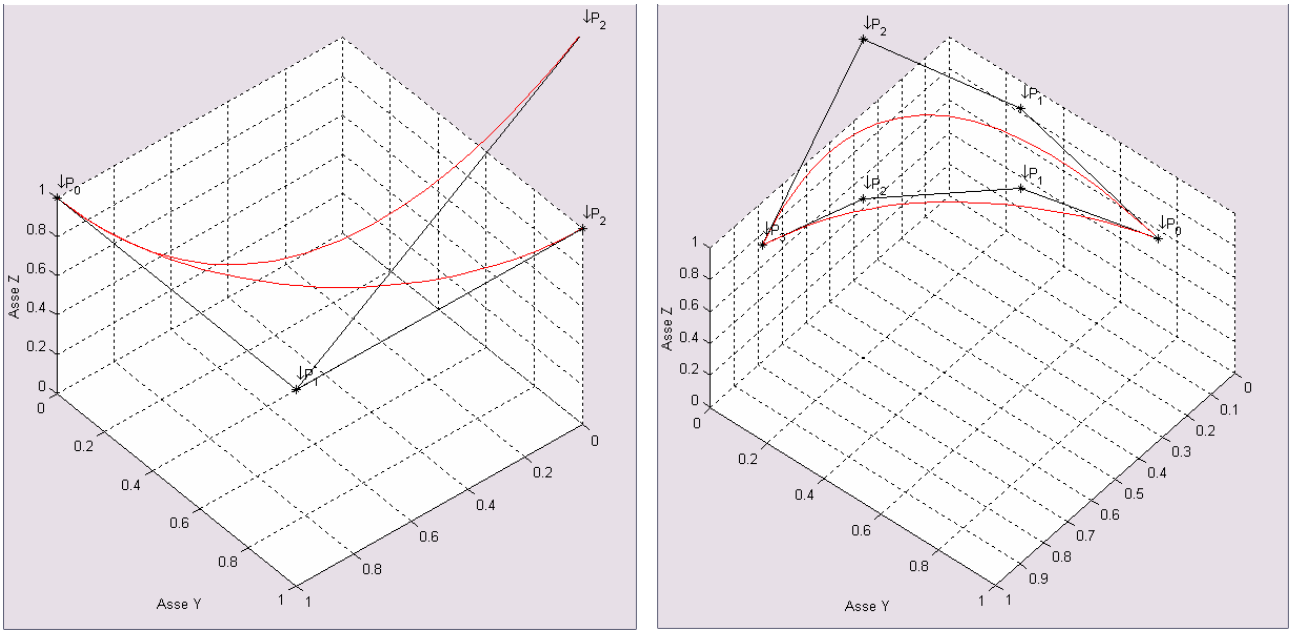


Figura 1-3 Interpretazione geometrica delle coordinate omogenee (a) arco di circonferenza, b) curva generica)

1.3. Le B-Spline

1.3.1. Premessa

Le curve costituite da un unico polinomio sono spesso poco adeguate per descrivere forme geometriche complesse; le principali controindicazioni sono:

- Per soddisfare un grande numero di vincoli serve un grado molto elevato, dato che è necessario un polinomio di grado *n-1* per interpolare *n* punti. Curve di grado elevato sono però difficili da manipolare e numericamente instabili a causa della proliferazione degli errori di troncamento.
- Descrivere una curva con un unico segmento di Bézier non è pratico poiché, anche se è possibile modificare la forma della curva spostando i punti di controllo, lo spostamento interessa comunque tutta la curva.

La soluzione per ovviare a questi inconvenienti è di utilizzare curve (o superfici) polinomiali a tratti, cioè una rappresentazione della forma

$$C(u) = \sum_{i=0}^{n} f_i(u) P_i \qquad \textbf{(1-13)}$$

in cui P_i sono i punti di controllo, e f_i sono funzioni polinomiali non nulle solo su una porzione limitata dell'itervallo di variazione del parametro u e nulle in tutto il resto

dell'intervallo. Questa proprietà viene chiamata "*local support",* e ha come conseguenza che una modifica a un punto di controllo modifica la curva in un intorno del punto soltanto e non l'intera curva.

1.3.2. Le funzioni di base di una curva B-Spline

Sia $U = \{u_0, \ldots, u_m\}$ una sequenza di numeri reali detti nodi, con $u_i \leq u_{i+1}$, con $i = 0, \ldots, m-1$. U è detto "vettore dei nodi". Definiamo la "i-esima funzione base di grado p" (ordine $p+1$), indicata con $N_{i,p}(u)$, come segue:

$$N_{i,0}(u) = \begin{cases} 1 & u_i \leq u < u_{i+1} \\ 0 & altrove \end{cases}$$
$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \qquad \textbf{(1-14)}$$

La (1-14) è nota come formula ricorsiva di Cox-deBoor.

Osservazioni:

- $N_{i,0}(u)$ è una funzione costante a tratti uguale a zero ovunque tranne che nell'intervallo $u \in [u_i, u_{i+1})$
- Per ogni $p>0$, $N_{i,p}(u)$ è una combinazione lineare di due funzioni base di grado $p-1$
- Per calcolare un set di funzioni di base è necessario conoscere il grado p e il vettore dei nodi U
- Se nel calcolo delle funzioni base si determina un quoziente del tipo 0/0, questo viene definito uguale a zero
- I nodi non devono essere necessariamente distinti; esiste così la possibilità di avere sotto-intervalli di lunghezza nulla
- La definizione delle funzioni di base è ottenuta per ricorsione
- Le funzioni di base definite su un vettore di nodi

  $$U = \left\{ \underbrace{0,\ldots,0}_{p+1}, \underbrace{1,\ldots,1}_{p+1} \right\}$$

  generano dei polinomi di Bernstein di grado p e sono pertanto le stesse funzioni di base di una curva di Bézier; in questo senso, quindi, le curve B-Spline sono generalizzazioni delle curve di Bézier.

Di seguito riportiamo alcune proprietà fondamentali delle funzioni base delle B-Spline:

i. *Local support*: $N_{i,p}(u) = 0$ se u è al di fuori dell'intervallo $[u_i, u_{i+p+1})$

ii. In ogni sotto-intervallo $[u_j, u_{j+1})$ al massimo $p+1$ delle $N_{i,p}(u)$ possono essere diverse da zero, nominalmente $N_{j-p,p}(u), \ldots, N_{j,p}(u)$

iii. $N_{i,p}(u) \geq 0$ per ogni i, p e u (non-negatività)

iv. Per ogni sotto-intervallo $[u_i, u_{i+1})$, la somma di tutte le funzioni non nulle in quell'intervallo è uguale a 1 per qualunque u (*partition of unity*)

v. Tutte le derivate di $N_{i,p}(u)$ esistono all'interno di un sotto-intervallo (dove la funzione è un polinomio) e agli estremi del sotto-intervallo la funzione è $p-k$ volte derivabile, dove k è la molteplicità del nodo. Quindi aumentare la molteplicità di un nodo riduce il grado di continuità delle curva in quel punto.

vi. A parte il caso $p=0$, $N_{i,p}(u)$ ha esattamente un massimo.

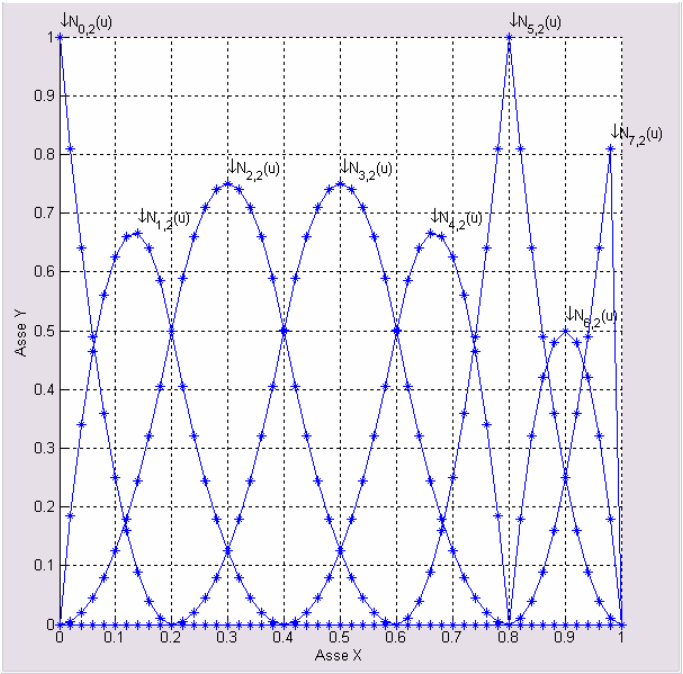


Figura 1-4 Funzioni base B-Spline di secondo grado definite sul vettore dei nodi U=[0,0,0,1,2,3,4,4,5,5,5]

1.3.3. Derivate di una funzione di base B-Spline

La derivata di una funzione di base è data da[2]:

$$N'_{i,p}(u) = \frac{p}{u_{i+p} - u_i} N_{i,p-1}(u) - \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \qquad \textbf{(1-15)}$$

[2] Per la dimostrazione vedere Appendice

Questa formula si presta bene alla ricorsione, anzi, definisce la derivata di una funzione base come combinazione lineare di due funzioni base di grado *p-1*. Continuando a derivare si può ottenere la formula generale:

$$N_{i,p}^{(k)}(u) = p\left(\frac{N_{i,p-1}^{(k-1)}(u)}{u_{i+p} - u_i} - \frac{N_{i+1,p-1}^{(k-1)}(u)}{u_{i+p+1} - u_{i+1}}\right) \qquad \textbf{(1-16)}$$

Alternativamente esiste un'altra formula per calcolare la derivata *k*-esima in termini delle funzioni $N_{i,p-k}(u), \ldots, N_{i+k,p-k}(u)$ [14] :

$$N_{i,p}^{(k)}(u) = \frac{p!}{(p-k)!}\sum_{j=0}^{k} a_{k,j} N_{i+j,p-k}(u) \qquad \textbf{(1-17)}$$

con

$$\begin{aligned} a_{0,0} &= 1 \\ a_{k,0} &= \frac{a_{k-1,0}}{u_{i+p-k+1} - u_i} \\ a_{k,j} &= \frac{a_{k-1,j} - a_{k-1,j-1}}{u_{i+p+j-k+1} - u_{i+j}} \\ a_{k,k} &= \frac{-a_{k-1,k-1}}{u_{i+p+1} - u_{i+k}} \end{aligned} \qquad \textbf{(1-18)}$$

1.3.4. Curve B-Spline

Una curva B-Spline di grado *p* è definita come segue:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \quad \text{con} \quad a \le u \le b \qquad \textbf{(1-19)}$$

Dove $\{P_i\}$ sono gli *n+1* punti di controllo e $N_{i,p}(u)$ sono le funzioni base B-Spline definite in (1-14) su un vettore di nodi non-periodico e non-uniforme $U = \left\{\underbrace{a,\ldots,a}_{p+1}, u_{p+1},\ldots,u_{m-p-1}, \underbrace{b,\ldots,b}_{p+1}\right\}$ in cui si dicono "nodi" gli m+1 punti di *U*; si può assumere che il vettore sia definito su [0,1], quindi *a=0, b=1*.

La procedura di calcolo di un punto sulla curva ad un dato *u* consiste di 3 passi:

1. trovare a quale sotto-intervallo appartiene *u*
2. calcolare le funzioni di base diverse da zero
3. moltiplicare il valore delle funzioni base non nulle per i corrispondenti punti di controllo

Le proprietà di queste curve sono elencate di seguito, molte delle quali sono conseguenza diretta delle corrispondenti proprietà delle funzioni di base.

i. Se $n = p$ e $U = \{0,...,0,1,...,1\}$, allora $C(u)$ è una curva di Bézier.

ii. $C(u)$ è una curva polinomiale a pezzi perché lo sono le sue funzioni di base; il grado p, il numero di punti di controllo $n+1$, e il numero di nodi $m+1$ sono legati dalla seguente relazione: $\boldsymbol{m = n + p + 1}$[3]

iii. Interpolazione agli estremi: $C(0) = P_0$ e $C(1) = P_n$

iv. Invarianza alle trasformazioni affini: applicare una trasformazione affine ai punti di controllo equivale ad applicarla a tutti i punti della curva. Sia **r** un punto in ε^3 (spazio euclideo tridimensionale). Una trasformazione affine indicata con Φ, trasforma ε^3 in ε^3 e ha questa forma: $\Phi(\mathbf{r}) = A\mathbf{r} + \mathbf{v}$, dove A è una matrice 3 x 3 e **v** è un vettore. Le trasformazioni affini includono traslazioni, rotazioni, scalature ecc… questa proprietà deriva direttamente dalla proprietà iv nel paragrafo 1.3.2 (*partition of unity*)

v. Proprietà del guscio convesso forte: le curve sono contenute nel guscio convesso definito dai punti di controllo P_i

vi. Modifica locale: muovendo il punto di controllo P_i cambia la curva solo nell'intervallo $[u_i , u_{i+p+1})$

vii. Il poligono di controllo rappresenta una approssimazione lineare della curva. Questa approssimazione migliora aggiungendo nodi o elevando il grado

viii. Muovendosi lungo una curva da u=0 a u=1 le $N_{i,p}(u)$ si comportano come interruttori all'attraversamento dei nodi, "spegnendo" un punto di controllo e "accendendone" un altro

ix. proprietà di diminuzione della variazione: ogni retta interseca la curva al più nel numero di volte in cui interseca il poligono di controllo

x. la continuità e la derivabilità segue dalle caratteristiche delle funzioni di base, quindi tutte le derivate di $C(u)$ esistono all'interno di un sotto-intervallo (dove la funzione è un polinomio) e agli estremi del sotto-intervallo (in corrispondenza dei nodi) la funzione è $p-k$ volte derivabile, dove k è la molteplicità del nodo. Quindi aumentare la molteplicità di un nodo riduce il grado di continuità delle curva in quel punto.

[3] Per le dimostrazioni vedere [14]

xi. È possibile e a volte utile usare punti di controllo multipli. Questo ha interessanti effetti sulla continuità geometrica e algebrica della curva come, per esempio, ottenere una cuspide. Tale proprietà viene utilizzata per la suddivisione di curve e superfici.

1.3.5. Derivata di una B-Spline

La derivata di una B-Spline si calcola mediante le derivate delle sue funzioni di base (§1.3.3), per cui:

$$C^{(k)}(u) = \sum_{i=0}^{n} N_{i,p}^{(k)}(u) P_i \qquad \textbf{(1-20)}$$

si osserva che calcolare la derivata di una B-Spline in un punto è agevole quanto calcolarne il valore; in particolare, mediante l'uso di queste curve, è possibile ottenere la funzione *C'(u)* in forma esplicita, e si verifica che la derivata prima di una B-Spline di grado *p* è una B-Spline di grado *p-1*, e precisamente:

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

in cui :

$$Q_i = p \frac{P_{i+1} - P_i}{u_{i+p+1} - u_{i+1}} \qquad \textbf{(1-21)}$$

e il vettore dei nodi si ottiene eliminando il primo e l'ultimo nodo della serie:

$$U = \left\{ \underbrace{0,\ldots,0}_{p}, u_{p+1},\ldots,u_{m-p-1}, \underbrace{1,\ldots,1}_{p} \right\}$$

1.3.6. Superfici B-Spline

Una superficie B-Spline si ottiene prendendo un reticolo bi-direzionale di punti di controllo, due vettori di nodi, e il prodotto di due curve B-Spline, ottenendo:

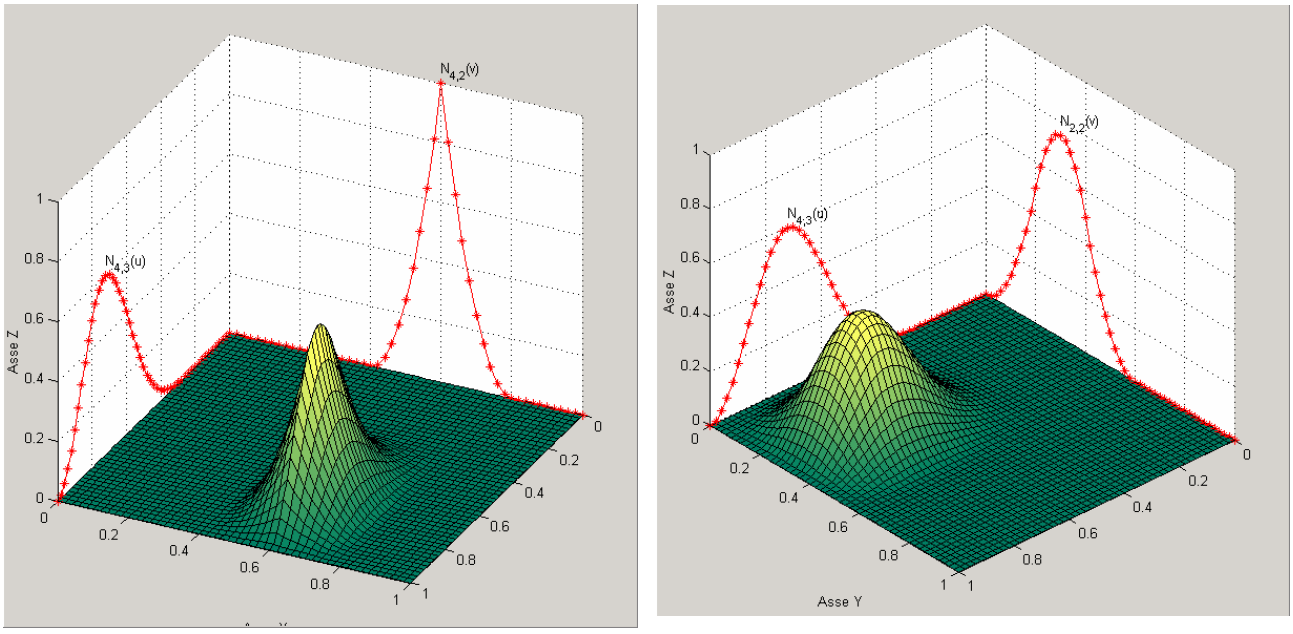
$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,q}(u) P_{i,j} \quad \text{con}$$

$$U = \left\{ \underbrace{0,\ldots,0}_{p+1}, u_{p+1},\ldots,u_{r-p-1}, \underbrace{1,\ldots,1}_{p+1} \right\} \qquad \textbf{(1-22)}$$

$$V = \left\{ \underbrace{0,\ldots,0}_{q+1}, u_{q+1},\ldots,u_{s-q-1}, \underbrace{1,\ldots,1}_{q+1} \right\}$$

Nella Figura 1-5 sono rappresentate due funzioni base di una superficie B-Spline, prodotto di due funzioni base B-Spline, definite come segue:

$N_{i,3}(u)$ (asse x) sono le funzioni base di grado 3 definite su vettore di nodi $U = \left\{0,0,0,0,\frac{1}{4},\frac{1}{2},\frac{3}{4},1,1,1,1\right\}$ e $N_{j,2}(u)$ (asse y) sono le funzioni base di grado 2 definite su vettore di nodi $V = \left\{0,0,0,\frac{1}{5},\frac{2}{5},\frac{3}{5},\frac{3}{5},\frac{4}{5},1,1,1\right\}$



a) b)

Figura 1-5 Rappresentazione delle funzioni base di una superficie B-Spline

Le operazioni necessarie per valutare un punto della curva sono:

- Trovare il sotto-intervallo in cui è contenuto u, $[u_i, u_{i+1})$
- Calcolare le funzioni non nulle $N_{i-p,p}(u), \ldots, N_{i,p}(u)$
- Trovare il sotto-intervallo in cui è contenuto v, $[v_j, v_{j+1})$
- Calcolare le funzioni non nulle $N_{j-q,q}(v), \ldots, N_{i,q}(v)$
- Moltiplicare le funzioni di base calcolate per i rispettivi punti di controllo

L'ultimo step assume la forma:

$$S(u,v) = \left[N_{k,p}(u)\right]^T \cdot \left[P_{k,l}\right] \cdot \left[N_{l,q}(v)\right] \text{ con } i-p \le k \le i,\ j-q \le l \le j \qquad \textbf{(1-23)}$$

Le proprietà di queste superfici discendono dalle proprietà delle analoghe curve B-Spline dalle quali sono costruite:

i. *Non-negatività*: $N_{i,p}(u)\, N_{j,q}(v) \ge 0$ qualunque i,j,p,q,u,v

ii. $\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u) N_{j,q}(v) = 1$ per ogni $(u,v) \in [0,1]\times[0,1]$ (*partition of unity*)

iii. Se $n = p$ e $m = q$, $U = \{0,...,0,1,...,1\}$, e $V = \{0,...,0,1,...,1\}$, allora $N_{i,p}(u)\ N_{j,q}(v) = B_{i,n}(u)\ B_{j,m}(u)$ qualunque i , j; la superficie B-Spline degenera in una superficie di Bézier, prodotto di polinomi di Bernstein.

iv. *Local support*: $N_{i,p}(u)\ N_{j,q}(v) = 0$ se (u,v) è al di fuori dell'intervallo $[u_i\ ,\ u_{i+p+1}) \times [v_j\ ,\ v_{j+q+1})$ (esempio in Figura 1-5)

v. In ogni sotto-intervallo $[u_i\ ,\ u_{i+1}) \times [v_j\ ,\ v_{j+1})$ al massimo $(p+1)(q+1)$ funzioni base possono essere diverse da zero, e precisamente le $N_{i,p}(u)\ N_{j,q}(v)$ con $i - p \le i \le i$ e $j - q \le j \le j$

vi. Se $p > 0$ e $q > 0$, allora $N_{i,p}(u)\ N_{j,q}(v)$ ha esattamente un massimo (vedi Figura 1-5).

vii. Tutte le derivate parziali di $N_{i,p}(u)\ N_{j,q}(v)$ esistono all'interno di un sotto-intervallo rettangolare essendo qui la funzione un prodotto di polinomi , mentre agli estremi del sotto-intervallo rettangolare la funzione è p-k volte derivabile in direzione u e (q-k) in direzione v, dove k è la molteplicità del nodo. Nella Figura 1-5 si vede come nel caso a) la derivata parziale di $N_{4,3}(u)\ N_{4,2}(v)$ in direzione v è discontinua lungo la linea $v = {}^3/_5$ dove $N_{4,2}(v)$ ha un punto angoloso. La derivata parziale di $N_{4,3}(u)\ N_{4,2}(v)$ in direzione u è continua ovunque, perché $N_{4,3}(u)$ è di classe C^2.

Le superfici B-Spline hanno le seguenti proprietà, derivabili dalle analoghe proprietà delle curve:

i. Se $n = p$ e $m = q$, $U = \{0,...,0,1,...,1\}$, e $V = \{0,...,0,1,...,1\}$, allora $S(u,v)$ è una superficie di Bézier.

ii. La superficie interpola gli angoli della griglia: $S(0,0) = P_{0,0}$, $S(1,0) = P_{n,0}$, $S(0,1) = P_{0,m}$, $S(1,1) = P_{n,m}$

iii. Una trasformazione affine della curva si ottiene applicandola ai suoi punti di controllo.

iv. Proprietà del guscio convesso forte: la superficie è contenuta nel guscio convesso definito dai punti di controllo $P_{i,j}$

v. La superficie definita dai punti di controllo definisce una approssimazione delle superficie B-Spline. Come nel caso delle curve, minore è il grado e migliore è l'approssimazione.

vi. Modifica locale: muovendo il punto di controllo $P_{i,j}$ cambia la curva solo nell'intervallo $[u_i\ ,\ u_{i+p+1}) \times [v_j\ ,\ v_{j+q+1})$

vii. La derivabilità di S(*u,v*) discende direttamente dalle derivabilità delle funzioni base, in particolare, S(*u,v*) è *p-k* volte derivabile in direzione *u* e (q-*k*) in direzione *v* in corrispondenza del nodo *u* o *v* di molteplicità *k*.

1.3.7. Derivata di una superficie B-Spline

Siano (*u,v*) fissati, di solito si è interessati al valore di tutte le derivate parziali di S(*u,v*) fino all'ordine *d*, cioè:

$$\frac{\partial^{k+l}}{\partial^k u \partial^l v} S(u,v) \qquad \text{con} \qquad 1 \le k + l \le d \qquad \textbf{(1-24)}$$

Come per le curve, otteniamo la derivata mediante le derivate selle funzioni di base, in particolare avremo:

$$\frac{\partial^{k+l}}{\partial^k u \partial^l v} S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}^{(k)} N_{j,q}^{(l)} P_{i,j} \qquad \textbf{(1-25)}$$

Purtroppo, nonostante la notevole generalizzazione delle B-Spline, queste restano funzioni polinomiali e, in quanto tali, non possono riprodurre esattamente molte forme, tra cui alcune semplici come la circonferenza.

1.4. Le NURBS

Per ovviare ai problemi esposti nel precedente paragrafo, principalmente quello di non poter rappresentare curve semplici come il cerchio, sono state introdotte le NURBS, generalizzazione ulteriore delle B-Spline.

Combinando le nozioni del paragrafo 1.2.3 e 1.3.4 otteniamo una B-Spline razionale, ovvero una Non Uniform Rational B-Spline (il non uniform si riferisce al vettore dei nodi)

Le NURBS sono curve razionali definite da punti di controllo e dai relativi pesi, e rappresentate in forma parametrica formalmente simili alle precedenti. Una NURBS di grado p è così definita:

$$C(u) = \sum_{i=0}^{n} R_{i,p}(u) P_i \qquad \textbf{(1-26)}$$

dove P_i sono n+1 punti di controllo, w_i i pesi che supponiamo non negativi e $u \in [0,1]$ e le funzioni base $R_{i,p}$ sono le funzioni base della NURBS definite come rapporti di funzioni base B-Spline di grado p su $U = \left\{ \underbrace{0,...,0}_{p+1}, u_{p+1},...,u_{m-p-1}, \underbrace{1,...,1}_{p+1} \right\}$ ovvero:

$$R_{i,p}(u) = \frac{N_{i,p}(u)w_i}{\sum_{j=0}^{n} N_{j,p}(u)w_j} \qquad \textbf{(1-27)}$$

l'unione dei punti di controllo fornisce il poligono di controllo.

Dall'equazione (1-27) e dalle proprietà delle funzioni base B-Spline viste al paragrafo 1.3.2 si derivano le proprietà delle NURBS:

i. *Local support*: $R_{i,p}(u) = 0$ se u è al di fuori dell'intervallo $[u_i , u_{i+p+1})$. Inoltre, per ogni dato intervallo $[u_i , u_{i+1})$ al massimo $p+1$ funzioni base sono diverse da zero e precisamente $R_{i-p,p}(u),\ldots, R_{i,p}(u)$.

ii. $R_{i,p}(u) \geq 0$ per ogni i, p e $u \in [0,1]$(non-negatività)

iii. $R_{0,p}(u) = R_{n,p}(u) = 1$

iv. Per ogni sotto-intervallo $[u_i , u_{i+1})$, la somma di tutte le funzioni non nulle in quell'intervallo è uguale a 1 per qualunque u (*partition of unity*)

v. Tutte le derivate di $R_{i,p}(u)$ esistono all'interno di un sotto-intervallo (dove la funzione è un polinomio) e agli estremi del sotto-intervallo la funzione è p-k volte derivabile, dove k è la molteplicità del nodo. Quindi aumentare la molteplicità di un nodo riduce il grado di continuità delle curva in quel punto.

vi. A parte il caso $p = 0$, $R_{i,p}(u)$ ha esattamente un massimo nell'intervallo $[0,1]$.

vii. Se tutti i pesi w_i sono uguali, allora $Ri_{,p}(u) = N_{i,p}(u)$; cioè le B-Spline sono casi particolari delle NURBS, in cui tutti i pesi hanno lo stesso valore, cioè $w_i = a$ con $a \neq 0$

Dalle proprietà elencate sopra si derivano le seguenti importanti caratteristiche geometriche:

viii. Dalla proprietà iii segue che $C(0) = P_0$ e $C(1) = P_n$

ix. Invarianza alle trasformazioni affini e alle trasformazioni prospettiche (importante per applicazioni CAD)

x. Proprietà del guscio convesso forte: le curve sono contenute nel guscio convesso definito dai punti di controllo P_i

xi. $C(u)$ è infinitamente derivabile all'interno dei sotto-intervalli, e p-k volte in corrispondenza di un nodo di molteplicità k.

xii. proprietà di diminuzione della variazione: ogni retta interseca la curva al più nel numero di volte con cui interseca il poligono di controllo

xiii. Una NURBS senza nodi interni è una curva di tipo rational-Bézier, visto che le $R_{i,p}(u) = B_{i,n}(u)$. Come casi speciali delle NURBS ci sono anche le B-Spline e le curve di Bézier non-razionali

xiv. Approssimazione locale: se il punto P_i o il peso w_i vengono modificati questo modificherà la curva solo nell'intervallo $[u_i, u_{i+p+1})$

La proprietà xiv è molto importante per la manipolazione delle curve, in quanto consente di apportare modifiche locali alla curva modificando la posizione del punto di controllo o del suo peso; in particolare, modificare il peso ha il seguente effetto:

- aumentando il valore del peso w_i la curva viene attratta da P_i
- diminuendo il valore peso w_i la curva viene allontanata da P_i
- al variare del valore del peso, ogni punto della curva si muove lungo una retta uscente da P_i
- se il peso diventa negativo, la tendenza resta, ma si perde la proprietà xii (diminuzione della variazione)

Come nel caso delle curve rational-Bézier l'uso di coordinate omogenee consente di manipolare le NURBS in modo efficiente.

Per un dato insieme di punti di controllo P_i e relativi pesi w_i costruiamo i corrispondenti punti di controllo pesati $P_i^w = (wx,wy,wz,w) = (X,Y,Z,W)$ e definiamo una curva B-Spline non-razionale (polinomio a tratti) nello spazio 4-dimensionale come $C^w(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i^w$.

Applicando la mappatura prospettica come definita al paragrafo 1.2.3 otteniamo:

$$C(u) = H\{C^w(u)\} = H\left\{\sum_{i=0}^{n} N_{i,p}(u) P_i^w\right\} = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i P_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i} = \sum_{i=0}^{n} R_{i,p}(u) P_i \qquad \textbf{(1-28)}$$

1.4.1. Derivata di una NURBS

Le derivate di una curva razionale sono complesse in quanto coinvolgono il denominatore con potenze alte. Nel paragrafo 1.3.3 abbiamo visto come derivare una curva non-razionale; quelle formule ovviamente si applicano anche a $C^w(u)$ in quanto curva non razionale. In questo paragrafo vedremo come sia possibile correlare le derivate di $C^w(u)$ con le derivate di $C(u)$

Sia:

$$C(u) = \frac{w(u)C(u)}{w(u)} = \frac{A(u)}{w(u)}$$

dove A(u) è la funzione vettoriale le cui tre coordinate sono le prime tre coordinate di $C^w(u)$. Derivando otteniamo:

$$C'(u) = \frac{w(u)A'(u) - w'(u)A(u)}{w(u)^2} = \frac{w(u)A'(u) - w'(u)w(u)C(u)}{w(u)^2} = \frac{A'(u) - w'(u)C(u)}{w(u)} \quad \textbf{(1-29)}$$

Siccome $A^w(u)$ e $w(u)$ rappresentano le coordinate di $C^w(u)$, possiamo ottenere le derivate applicando la (1-21). Per ottenere le derivate di ordine superiore si deriva A(u) usando la regola di Leibnitz:

$$A^{(k)}(u) = \left(w(u)C(u)\right)^{(k)} = \sum_{i=0}^{k} \binom{k}{i} w^{(i)}(u) C^{(k-i)}(u)$$

$$= w(u)C^{(k)}(u) + \sum_{i=1}^{k} \binom{k}{i} w^{(i)}(u) C^{(k-i)}(u)$$

da cui si ottiene

$$C^{(k)}(u) = \frac{A^{(k)}(u) - \sum_{i=1}^{k} \binom{k}{i} w^{(i)}(u) C^{(k-i)}(u)}{w(u)} \quad \textbf{(1-30)}$$

1.4.2. Superfici NURBS

Una superficie NURBS di grado p in direzione u e q in direzione v è definita in (1-31). Anche in questo caso, ovviamente devono essere soddisfatte le relazioni fra grado, numero di nodi e numero di punti di controllo, in entrambe le direzioni, cioè:

$r = n + p + 1$ e $s = m + q + 1$

$$S(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u) \cdot N_{j,q}(v) \cdot w_{i,j}(u) P_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u) \cdot N_{j,q}(v) \cdot w_{i,j}(u)} \quad \textbf{(1-31)}$$

con vettori dei nodi:

$$U = \left\{ \underbrace{0,\ldots,0}_{p+1}, u_{p+1}, \ldots, u_{r-p-1}, \underbrace{1,\ldots,1}_{p+1} \right\}$$

$$V = \left\{ \underbrace{0,\ldots,0}_{q+1}, u_{q+1}, \ldots, u_{s-q-1}, \underbrace{1,\ldots,1}_{q+1} \right\} \quad \textbf{(1-32)}$$

Definendo la funzione di base polinomiale razionale:

$$R_{i,j}(u,v) = \frac{N_{i,p}(u) \cdot N_{j,q}(v) \cdot w_{i,j}(u)}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u) \cdot N_{j,q}(v) \cdot w_{i,j}(u)} \qquad \textbf{(1-33)}$$

la superficie (1-31) può essere scritta come:

$$S(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} R_{i,j}(u,v) P_{i,j} \qquad \textbf{(1-34)}$$

Le proprietà delle NURBS sono sostanzialmente le stesse delle B-Spline, le riassumiamo brevemente qui:

- $R_{i,p}$ è una funzione razionale di grado p tale che $R_{i,p} \geq 0$ qualunque u, e $R_{i,p} \neq 0$ solo in $[u_i, u_{i+p+1})$
- La curva NURBS è composizione di curve razionali ognuna di grado p
- Cambiare la posizione di un punto di controllo p_i modifica la curva $p(u)$ solo in $[u_i, u_{i+p+1})$
- Deve essere soddisfatta la relazione $\boldsymbol{m = n + p + 1}$
- Le curve B-Spline e Bézier sono casi particolari delle NURBS, infatti una B-Spline è una NURBS con pesi tutti uguali, mentre una curva di Bézier è una NURBS con $n = p$.
- Mediante queste curve è possibile descrivere esattamente curve coniche
- Aumentando il valore del peso di un punto di controllo avvicina la curva al punto stesso.

Questa ulteriore generalizzazione da un lato ha portato ad avere a disposizione uno strumento matematico in grado di generare curve molto complesse con relativa semplicità, dall'altro ha complicato la gestione della curva, l'inserzione di nodi ecc…

Nel capitolo successivo, verrà ricercato un algoritmo efficiente per approssimare una curva, di cui siano noti alcuni punti, mediante una NURBS; ciò si ottiene calcolando i punti di controllo e i pesi che la definiscono.

1.5. Potenzialità delle NURBS

Come detto le curve NURBS hanno la capacità di rappresentare curve anche complesse mediante un limitato numero di punti di controllo. Ad esempio, in Figura 1-6 si vede che la sola definizione di una spezzata con 6 segmenti (7 punti di controllo) genera la curva NURBS (in nero), generata con i seguenti dati:

Punti di controllo (n + 1 = 7)			
	X	Y	Z
P_0	0.50000	3.00000	0
P_1	1.50000	5.50000	0
P_2	4.50000	5.50000	0
P_3	3.00000	1.50000	0
P_4	7.50000	1.50000	0
P_5	6.00000	4.00000	0
P_6	8.50000	4.50000	0

Nodi (m + 1 = 10)	
u_0	0
u_1	0
u_2	0
u_3	0.2500
u_4	0.5000
u_5	0.7500
u_6	0.7500
u_7	1.0000
u_8	1.0000
u_9	1.0000

La curva NURBS in figura, allo stesso modo delle curve B-Splines, presenta un punto angoloso in corrispondenza di un nodo ripetuto ($u_5 = u_6 = 0.75$), in esso risulta $C^{2-2} = C^0$, infatti p = m-n-1 = 9-6-1 = 2. La curva in figura, malgrado il punto angoloso intermedio, è una unica curva. Per forzare la curva a passare per uno qualsiasi dei punti di controllo, è sufficiente che la molteplicità del nodo corrispondente sia uguale al grado della NURBS. In questi punti, inoltre, la NURBS è tangente alle linee di controllo.

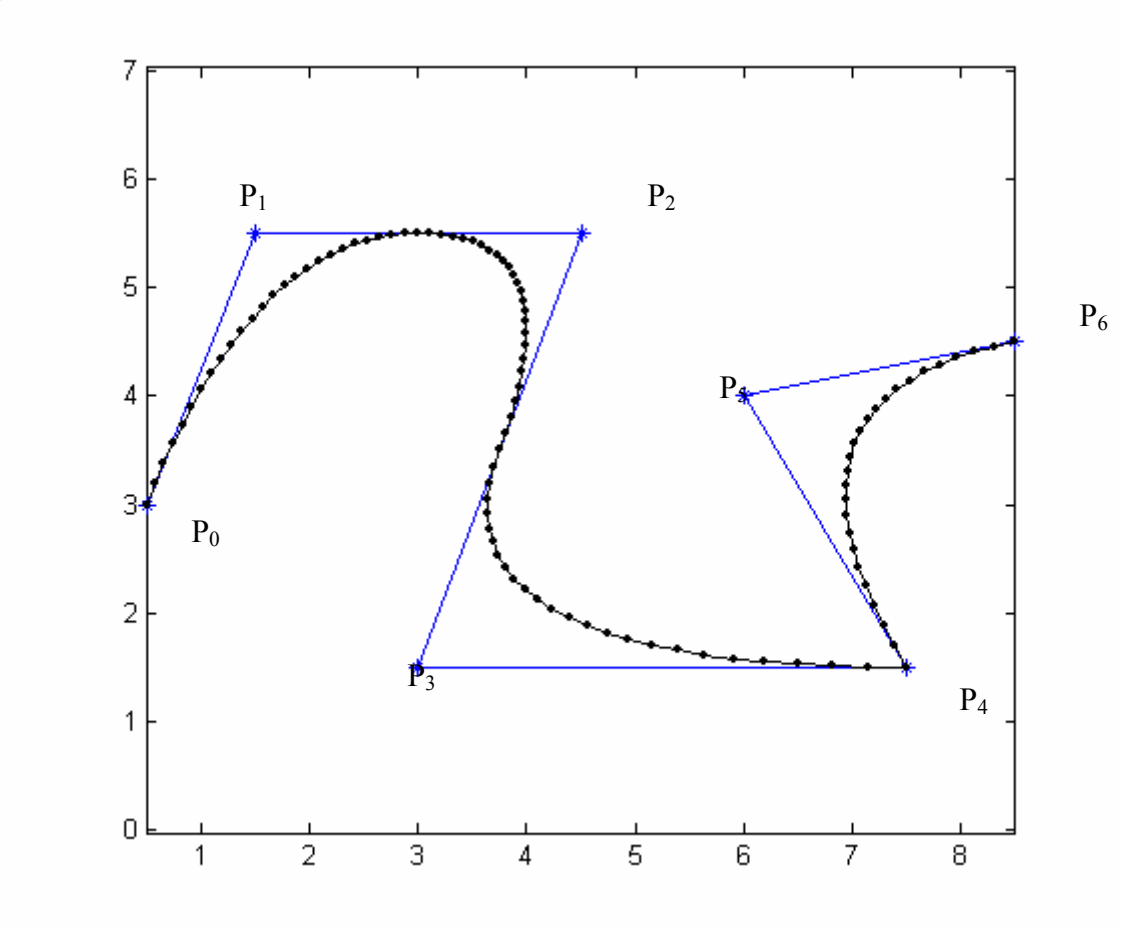


Figura 1-6 NURBS creata mediante 7 punti di controllo

Estendendo il concetto a due dimensioni si ottengono delle superfici; in questo caso servono oltre ai punti di controllo, due vettori di nodi, che possono anche essere di dimensioni differenti, rendendo così possibile avere gradi diversi nelle due direzioni.

Ad esempio, in Figura 1-7, è rappresentata la superficie NURBS creata con una griglia di punti 5x5 ($n = 4$, $m = 4$)e due vettori di nodi. Nella direzione x il vettore di nodi è composto da 8 elementi, per cui il grado della superficie in questa direzione $p = m-n-1 = 7-4-1 = 2$; nella direzione y il vettore dei nodi è composto da 7 elementi, per cui $p = m-n-1 = 6-4-1 = 1$. Il risultato è che la superficie è di secondo grado in direzione x mentre solo di primo, quindi lineare a tratti, in direzione y.

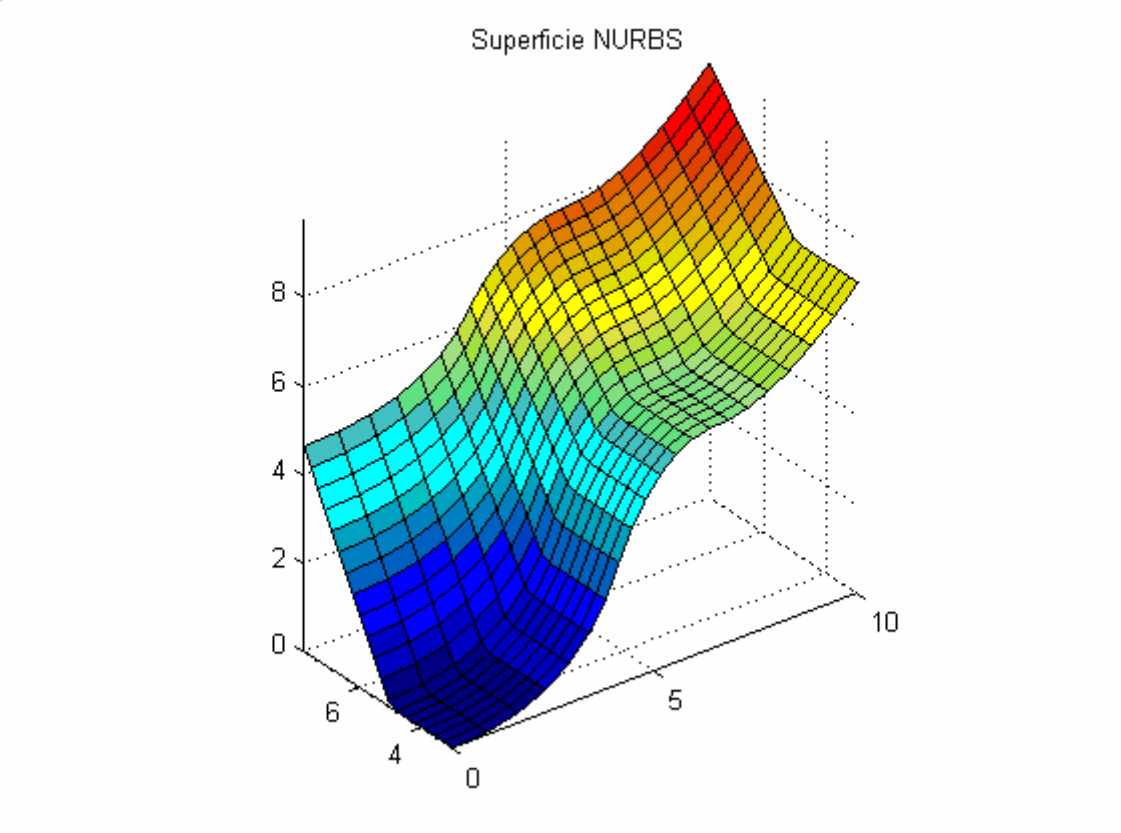


Figura 1-7 Superficie NURBS con funzioni di grado diverso nelle due direzioni

Come già detto, con queste curve è possibile rappresentare esattamente anche i cerchi, come mostrato in Figura 1-8, mentre in Figura 1-9 è rappresentata un elica 3D di facile definizione.

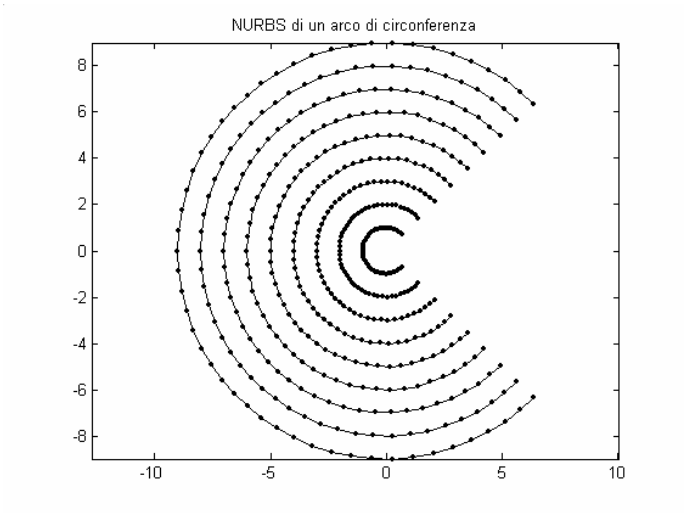


Figura 1-8 Arco di circonferenza rappresentato mediante NURBS

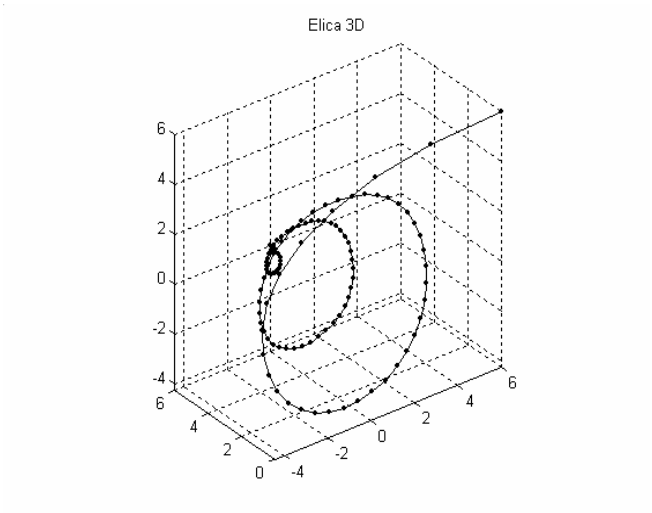


Figura 1-9 Elica 3D rappresentata mediante NURBS

Esistono poi algoritmi che calcolano in maniera efficiente la derivata (vedi paragrafo 1.4.1) in un punto qualunque di una curva o superficie NURBS come schematicamente mostrato in Figura 1-10 e Figura 1-11

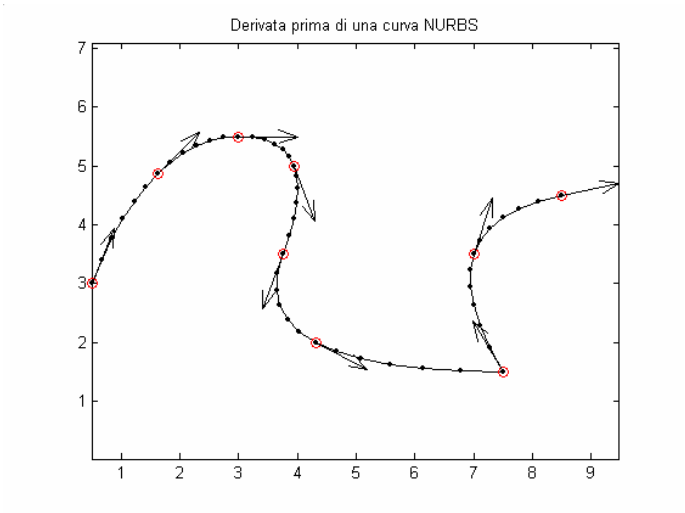


Figura 1-10 Derivate di una curva NURBS

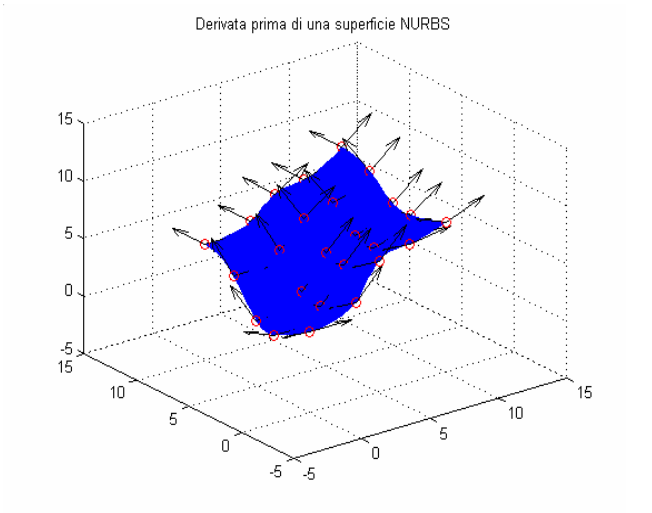


Figura 1-11 Derivata di una superficie NURBS

Nelle figure seguenti sono rappresentate alcune forme che sono rappresentabili molto semplicemente mediante NURBS. La superficie di Figura 1-12 è costruita interpolando la curva di Figura 1-6 con un arco di circonferenza, mentre la Figura 1-13 dimostra una forma complessa ottenuta per rivoluzione di un arco di circonferenza e due circonferenze.

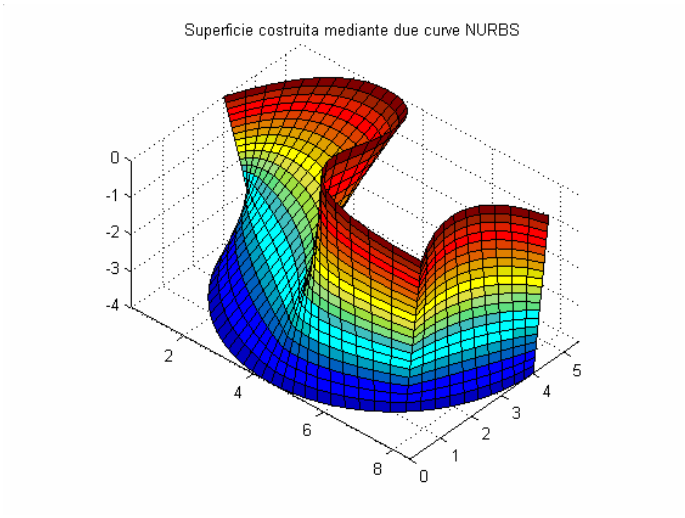


Figura 1-12 Esempio di superficie costruita partendo da due curve NURBS con interpolazione lineare

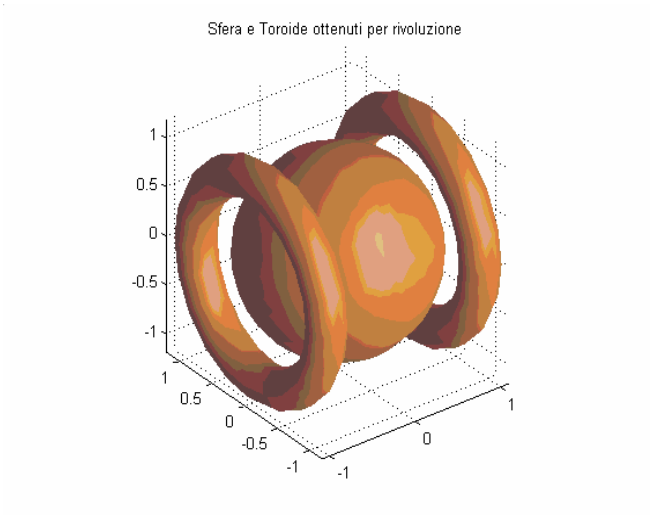


Figura 1-13 Sfera e toroide ottenuti per rivoluzione di semplici forme bidimensionali

Capitolo 2

2. Interpolazione tramite NURBS

2.1. Premessa

Nel capitolo precedente sono state analizzate alcune rappresentazioni matematiche di curve e superfici nello spazio. Di tutte le rappresentazioni analizzate, le B-Spline e le NURBS, hanno il maggiore interesse pratico per il presente lavoro; talora verrà fatto uso delle curve di Bézier per comodità di calcolo.

Questo capitolo riguarda il problema dell'interpolazione. Per quanto riguarda le curve di interesse, questo problema può essere formulato come segue:

"dato un insieme di punti nello spazio, calcolare i punti di controllo, i nodi e i pesi che generano la curva che interpola i punti dati con livello di continuità prescritto e prescritte derivate agli estremi".

Questo problema è risolubile mediante diversi algoritmi insieme ad ipotesi iniziali. La prima ipotesi è il grado della curva che si vuole ottenere, legato al tipo di continuità richiesto. Di solito, per applicazioni CAD, è sufficiente che la curva abbia continuità C^1 (curva "smooth"). Per ottenere questa condizione durante l'assemblaggio dei vari tratti di curva è necessario potere imporre, oltre alla condizione di interpolazione dei punti dati, anche l'uguaglianza delle derivate nei punti di unione (di solito, per comodità, si fanno coincidere con i punti da interpolare); il minimo grado che consente di soddisfare questa condizione (ed è quello normalmente scelto per rappresentare curve e superfici nei più diffusi software CAD) è $p = 3$.

2.2. Parametrizzazione di una curva

Prima di procedere è interessante fare una distinzione fra continuità "matematica" e continuità "geometrica". La continuità C^1 non è strettamente necessaria per avere una curva o superficie di tipo "smooth", ma l'uguaglianza della direzione delle derivate nel punto di unione è sufficiente per non avere cuspidi nella curva, anche se il valore assoluto delle derivate è diverso. Dunque è sufficiente che le due curve abbiano direzione tangente comune, indipendentemente dal modulo della derivata. Questa condizione è meno restrittiva della precedente e viene di solito indicata con G^1 (ed è detta continuità "geometrica"). Questa nuova definizione di continuità semplifica il problema del calcolo della curva interpolante.

Infatti, come noto, le derivate di una curva parametrica in genere (ed in particolare le derivate di una NURBS) vengono calcolate rispetto al parametro u ed è possibile, mantenendo inalterata la curva, cambiare il vettore dei nodi in modo da ottenere una diversa spaziatura dei nodi stessi e derivate diverse alle estremità. Sia $C(u) = [x(u) , y(u) , z(u)]$ una arbitraria curva parametrica definita su $u \in [a,b]$ e sia $u = f(s)$ una funzione a valori reali definita su $s \in [c,d]$ e che soddisfi le seguenti condizioni:

- $f'(s) > 0$ per ogni $s \in [c,d]$ (f è strettamente crescente in $s \in [c,d]$)
- $a = f(c)$ e $b = f(d)$ (f trasforma l'intervallo $[a,b]$ in $[c,d]$)

La composizione di $C(u)$ con $f(s)$ data da:

$$C(s) = C(f(s)) = [x(f(s)) , y(f(s)) , z(f(s))]$$

è detta "riparametrizzazione" di $C(u)$ (vedi Figura 2-1).

Le derivate prima e seconda della curva sono le seguenti:

$$C'(s) = C'(u)f'(s) \qquad C''(s) = C'(u)f''(s) + C''(u)\,(f'(s))^2$$

Come si nota, la direzione della derivata prima è rimasta invariata, mentre per le derivate di ordine superiore cambia sia la direzione che il valore assoluto.

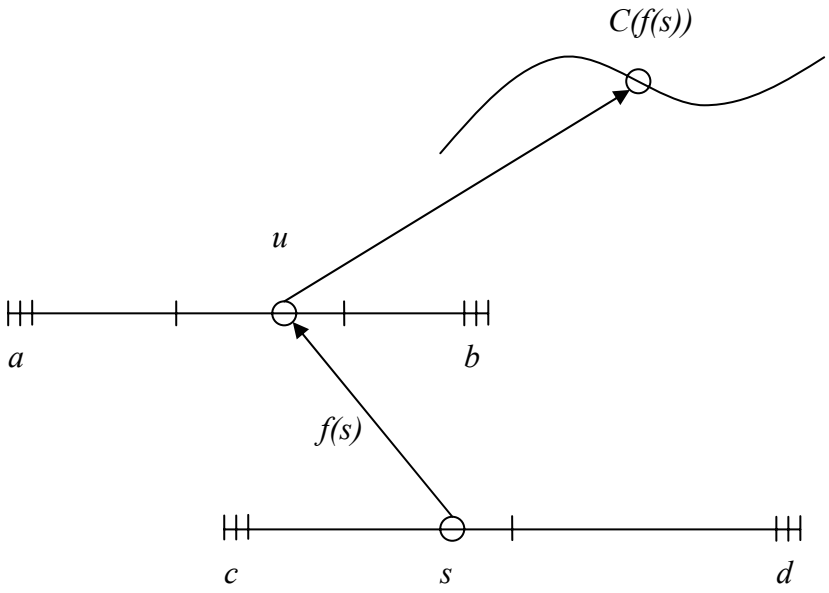


Figura 2-1 Riparametrizzazione di una curva

Esempio:

Sia $C(u) = [x(u) , y(u)] = [u , -2u^2 + 2u]$

definita sull'intervallo $u = [0,1]$. Supponiamo di voler riparametrizzare la curva con un nuovo parametro $s \in [c,d]$ in modo che a u = $^1/_2$ corrisponda s = $^3/_5$, cioè:

$$C\left(u=\frac{1}{2}\right)=C\left(s=\frac{3}{5}\right)=\left(\frac{1}{2},\frac{1}{2}\right)$$

quindi u deve soddisfare 3 condizioni:

$$0 = f(0) \qquad {}^1/_2 = f({}^3/_5) \qquad 1 = f(1)$$

scegliamo per f(s) un polinomio quadratico:

$$u = f(s) = as^2 + bs + c$$

dalla prima condizione si ottiene $c = 0$, le altre due formano il seguente sistema:

$$\begin{cases} \frac{1}{2} = \frac{9}{25}a + \frac{3}{5}b \\ 1 = a + b \end{cases}$$

risolvendo otteniamo:

$$\begin{cases} a = \frac{5}{12} \\ b = \frac{7}{12} \end{cases}$$

per cui:

$$u = f(s) = \frac{5}{12}s^2 + \frac{7}{12}s$$

da cui, sostituendo, otteniamo la curva riparametrizzata:

$$C(s) = \left[\frac{5}{12}s^2 + \frac{7}{12}s, -\frac{25}{72}s^4 - \frac{35}{36}s^3 + \frac{11}{72}s^2 + \frac{7}{6}s\right]$$

come è facile verificare,

$$C(s=0) = C(u=0) = [0,0]$$
$$C\left(s=\frac{3}{5}\right) = C\left(u=\frac{1}{2}\right) = \left[\frac{1}{2},\frac{1}{2}\right]$$
$$C(s=1) = C(u=1) = [1,0]$$

Inoltre si vede come sia cambiato il modulo della derivata agli estremi mentre non è cambiato l'argomento, infatti:

$$C'(u=0) = [1,2] = \left(\sqrt{5}\angle 90° - \arctan\left(\frac{1}{2}\right)\right)$$

$$C'(s=0) = \left[\frac{7}{12},\frac{7}{6}\right] = \left(\frac{7\sqrt{5}}{12}\angle 90° - \arctan\left(\frac{1}{2}\right)\right)$$

mentre, per quanto riguarda la derivata seconda, si ottiene un valore diverso sia in valore assoluto, sia in argomento. Infatti:

$$C''(u=0)=[0,-4]=(4\angle -90°)$$

$$C''(s=0)=\left[\frac{5}{6},\frac{11}{36}\right]=\left(\frac{\sqrt{1021}}{36}\angle \arctan\left(\frac{11}{30}\right)\right)$$

Quanto visto finora può essere utilizzato, oltre che per imporre una derivata opportuna agli estremi, anche per ridistribuire in maniera più uniforme i nodi lungo la curva; ciò consente un migliore controllo sulla curva stessa in fase di editing e la possibilità di ridurre il grado o il numero di punti di controllo.

2.3. Inserimento di un nodo

2.3.1. Inserimento di un nodo in una curva

Il problema dell'inserimento di un nodo in una curva può essere enunciato come segue:

"data una curva NURBS, inserire il nodo $\overline{u}$ in modo da non modificare la curva stessa"

Sia $C(u)=\sum_{i=0}^{n} N_{i,p}(u)P_i$ una curva definita su $U=\{u_0,...,u_m\}$. Sia $\overline{u}\in[u_k,u_{k+1})$ e sia $\overline{U}=\{\overline{u}_0=u_0,...,\overline{u}_k=u_k,\overline{u}_{k+1}=\overline{u},\overline{u}_{k+2}=u_{k+1},...,\overline{u}_{m+1}=u_m\}$ il nuovo vettore dei nodi dopo l'inserimento di $\overline{u}$. Se Ψ_U e $\Psi_{\overline{U}}$ sono gli spazi vettoriali di tutte le curve definite in U e $\overline{U}$, risulta $\Psi_U\subset\Psi_{\overline{U}}$ e $\dim(\Psi_{\overline{U}})=\dim(\Psi_U)+1$; per questo motivo $C(u)$ ha una rappresentazione su $\overline{U}$ della forma:

$$C(u)=\sum_{i=0}^{n+1}\overline{N}_{i,p}(u)Q_i \qquad \textbf{(2-1)}$$

Dove le $\overline{N}_{i,p}(u)$ sono le funzioni di base di grado *p* definite sul vettore di nodi $\overline{U}$. L'algoritmo di inserimento di un nodo calcola i punti $\{Q_i\}$; è importante notare che l'inserimento di un nodo comporta solo il cambio di spazio vettoriale, la curva non è cambiata né geometricamente, né parametricamente.

I punti $\{Q_i\}$ si ottengono dalla risoluzione del sistema lineare seguente

$$\sum_{i=0}^{n} N_{i,p}(u)P_i=\sum_{i=0}^{n+1}\overline{N}_{i,p}(u)Q_i \qquad \textbf{(2-2)}$$

Sostituendo in *u (n+2)* valori si ottiene un sistema lineare di *(n+2)* equazioni nelle *(n+2)* incognite $\{Q_i\}$ risolto il quale si ottengono i nuovi punti di controllo. I valori scelti per *u* devono soddisfare la condizione che il sistema sia non singolare.

Tuttavia questo procedimento non è molto efficiente, e ne esiste uno più diretto. Per $\bar{u} \in [u_k, u_{k+1})$, la proprietà ii del paragrafo 1.3.2 implica che:

$$\sum_{i=k-p}^{k} N_{i,p}(u) P_i = \sum_{i=k-p}^{k+1} \bar{N}_{i,p}(u) Q_i \qquad \textbf{(2-3)}$$

per ogni $\bar{u} \in [u_k, u_{k+1})$.

Inoltre:

$$\begin{aligned} N_{i,p}(u) &= \bar{N}_{i,p}(u) \\ N_{i,p}(u) &= \bar{N}_{i+1,p}(u) \end{aligned} \quad \text{per} \quad \begin{aligned} i &= 0,\ldots,k-p-1 \\ i &= k+1,\ldots,n \end{aligned} \qquad \textbf{(2-4)}$$

Le equazioni (2-4) e (2-3) insieme alla indipendenza lineare delle funzioni di base assicurano che:

$$\begin{aligned} P_i &= Q_i \\ P_i &= Q_{i+1} \end{aligned} \quad \text{per} \quad \begin{aligned} i &= 0,\ldots,k-p-1 \\ i &= k+1,\ldots,n \end{aligned} \qquad \textbf{(2-5)}$$

Ora consideriamo $N_{i,p}(u)$ per $i = k-p,\ldots,k$. Possono essere espresse in termini di $\bar{N}_{i,p}(u)$ per $i = k-p,\ldots,k+1$:

$$N_{i,p}(u) = \frac{\bar{u} - \bar{u}_i}{\bar{u}_{i+p+1} - \bar{u}_i} \bar{N}_{i,p}(u) + \frac{\bar{u}_{i+p+2} - \bar{u}}{\bar{u}_{i+p+2} - \bar{u}_{i+1}} \bar{N}_{i+1,p}(u) \qquad \textbf{(2-6)}$$

Questa formula si dimostra in maniera non banale per induzione su *p*[4]. Sostituendo l'equazione (2-6) in (2-3) si ottiene:

$$\begin{aligned} &\left(\frac{\bar{u} - \bar{u}_{k-p}}{\bar{u}_{k+1} - \bar{u}_{k-p}} \bar{N}_{k-p,p}(u) + \frac{\bar{u}_{k+2} - \bar{u}}{\bar{u}_{k+2} - \bar{u}_{k-p+1}} \bar{N}_{k-p+1,p}(u) \right) P_{k-p} \\ &+ \left(\frac{\bar{u} - \bar{u}_{k-p+1}}{\bar{u}_{k+2} - \bar{u}_{k-p+1}} \bar{N}_{k-p+1,p}(u) + \frac{\bar{u}_{k+3} - \bar{u}}{\bar{u}_{k+3} - \bar{u}_{k-p+2}} \bar{N}_{k-p+2,p}(u) \right) P_{k-p+1} \\ &\vdots \\ &+ \left(\frac{\bar{u} - \bar{u}_k}{\bar{u}_{k+p+1} - \bar{u}_k} \bar{N}_{k,p}(u) + \frac{\bar{u}_{k+p+2} - \bar{u}}{\bar{u}_{k+p+2} - \bar{u}_{k+1}} \bar{N}_{k+1,p}(u) \right) P_k \\ &= \bar{N}_{k-p.p}(u) \cdot Q_{k-p,p}(u) + \ldots + \bar{N}_{k+1,p}(u) \cdot Q_{k+1,p}(u) \end{aligned}$$

eguagliando i coefficienti e usando il vettore dei nodi U invece che $\bar{U}$, otteniamo:

[4] Le dimostrazioni mediante il metodo delle differenze divise possono essere trovate in [3]

$$
\begin{aligned}
0 &= \overline{N}_{k-p,p}(u)\left(Q_{k-p} - P_{k-p}\right) \\
&+ \overline{N}_{k-p+1,p}(u)\left(Q_{k-p+1} - \frac{\overline{u} - u_{k-p+1}}{u_{k+1} - u_{k-p+1}} P_{k-p+1} - \frac{u_{k+1} - \overline{u}}{u_{k+1} - u_{k-p+1}} P_{k-p} \right) \\
&\vdots \\
&+ \overline{N}_{k,p}(u)\left(Q_k - \frac{\overline{u} - u_k}{u_{k+p} - u_k} P_k - \frac{u_{k+p} - \overline{u}}{u_{k+p} - u_k} P_{k-1} \right) + \overline{N}_{k+1,p}(u)\left(Q_{k+1} - P_k\right)
\end{aligned}
\qquad \textbf{(2-7)}
$$

per $i = k - p + 1, \ldots, k$ definiamo

$$\alpha_i = \frac{\overline{u} - u_i}{u_{i+p} - u_i} \qquad \textbf{(2-8)}$$

e notiamo che

$$1 - \alpha_i = \frac{u_{i+p} - \overline{u}}{u_{i+p} - u_i} \qquad \textbf{(2-9)}$$

Usando ancora l'indipendenza lineare delle funzioni di base e sostituendo le equazioni (2-8) e (2-9) nell'equazone (2-7) si ottiene:

$$
\begin{aligned}
&Q_{k-p} = P_{k-p} \\
&Q_i = \alpha_i P_i + (1 - \alpha_i) P_{i-1} \qquad\qquad k - p + 1 \le i \le k \\
&Q_{k+1} = P_k
\end{aligned}
\qquad \textbf{(2-10)}
$$

Finalmente, combinando le equazioni (2-10) con le (2-5) otteniamo la formula per calcolare i nuovi punti di controllo:

$$Q_i = \alpha_i P_i + (1 - \alpha_i) P_{i-1}$$

dove

$$\alpha_i = \begin{cases} 1 & i \le k - p \\ \dfrac{\overline{u} - u_i}{u_{i+p} - u_i} & k - p + 1 \le i \le k \\ 0 & i \ge k + 1 \end{cases} \qquad \textbf{(2-11)}$$

L'equazione (2-11) ci assicura che solo p nuovi punti di controllo devono essere calcolati.

2.3.2. Inserimento di un nodo in una superficie

L'inserimento di un nodo in una superficie viene effettuato applicando le formule del paragrafo precedente alle colonne e alle righe della matrice dei punti di controllo. In particolare, detta $P_{i,j}$ la matrice dei punti di controllo, con $0 \le i \le n$, $0 \le j \le m$, il nodo $\overline{u}$ si inserisce nella superficie effettuando $m + 1$ inserimenti, uno per ogni colonna di $P_{i,j}$. Analogamente viene inserito un nodo $\overline{v}$.

2.4. L'interpolazione globale

2.4.1. Interpolazione di curve

Una delle tecniche di interpolazione prese in considerazione da questo lavoro è detta "interpolazione globale". Un primo problema che dobbiamo affrontare è quello della scelta dei pesi. In effetti, tranne in particolari casi in cui sia necessario rappresentare esattamente curve della famiglia delle coniche, è sufficiente imporre tutti i pesi uguali a 1 ($w_i = 1 \ \forall \ i$). Questa assunzione ha come risultato la costruzione di curve interpolanti di tipo B-Spline, perfettamente compatibili con i nostri scopi. Inoltre abbiamo un altro vantaggio che deriva da questa scelta: il sistema che andremo a descrivere tra poco risulterà lineare, permettendoci di risolverlo agevolmente, con notevole efficienza di calcolo e stabilità numerica, mediante uno dei tanti algoritmi a disposizione in letteratura.

Sia $\{Q_k\}$, $k=0,\ldots,n$, sia un insieme di punti da interpolare con una B-Spline di grado p. Se assegniamo un valore del parametro, $\bar{u}_k$, ad ogni Q_k e scegliamo un appropriato vettore dei nodi $U = \{u_0, \ldots, u_m\}$, possiamo impostare un sistema di equazioni lineari di dimensione $(n + 1) \times (n + 1)$ del tipo:

$$Q_k = C(\bar{u}_k) = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k) P_i \qquad \textbf{(2-12)}$$

in cui i punti di controllo P_i sono le $n + 1$ incognite. Sia r il numero di coordinate di Q_k in cui, nel caso di curve $r = 2$ (ma il metodo vale anche per $r = 3$ (superficie) e per $r = 4$ (coordinate omogenee)). L'equazione (2-12) ha una sola matrice di coefficienti, $N_{i,p}(u_k)$, indipendentemente dal numero delle coordinate r (un esempio verrà illustrato in seguito).

La scelta dei valori $\bar{u}_k$ e del vettore U influisce sia sulla forma della curva che sulla sua parametrizzazione. Per il resto del paragrafo supporremo che il parametro u sia compreso nell'intervallo [0,1] (è possibile sia estendere le conclusioni a qualunque intervallo, sia riparametrizzare un qualunque intervallo per trasformarlo in [0,1] senza variare la parametrizzazione). Esistono principalmente 3 metodi per scegliere il vettore dei nodi U, e sono:

1. Equispaziati:

$$\bar{u}_0 = 0$$
$$\bar{u}_n = 1$$
$$\bar{u}_k = \frac{k}{n} \qquad \textbf{(2-13)}$$
$$per$$
$$k = 1,...,n-1$$

Questo metodo è sconsigliato, perché in caso di dati non uniformemente spaziati può produrre curve contenenti cappi e in generale forme non corrispondenti alla reale curva di cui i punti dati sono approssimazione.

2. Spaziatura in base alla lunghezza della spezzata:

   Sia *d* la lunghezza totale della spazzata definita dai punti, cioè:

$$d = \sum_{k=1}^{n} |Q_k - Q_{k-1}| \qquad \textbf{(2-14)}$$

   allora sarà:

$$\bar{u}_0 = 0$$
$$\bar{u}_n = 1$$
$$\bar{u}_k = u_{k-1} + \frac{|Q_k - Q_{k-1}|}{d} \qquad \textbf{(2-15)}$$
$$per$$
$$k = 1,...,n-1$$

   questo è il metodo più diffuso, in quanto è generalmente adeguato a rappresentare curve differenti ed è poco sensibile alla spaziatura dei punti da interpolare. Inoltre fornisce una buona parametrizzazione, nel senso che approssima una parametrizzazione uniforme.

3. Metodo "centripeto"

   Sia:

$$d = \sum_{k=1}^{n} \sqrt{|Q_k - Q_{k-1}|} \qquad \textbf{(2-16)}$$

   allora sarà:

$$\bar{u}_0 = 0$$
$$\bar{u}_n = 1$$
$$\bar{u}_k = u_{k-1} + \frac{\sqrt{|Q_k - Q_{k-1}|}}{d} \qquad \textbf{(2-17)}$$
$$per$$
$$k = 1,...,n-1$$

questo è un metodo nuovo che fornisce migliori risultati del precedente nel caso di curve ad elevata curvatura.

Definiti i valori del parametro $\bar{u}_k$, il vettore dei nodi *U* potrebbe essere definito in modo da risultare uniformemente spaziato, cioè:

$$\begin{aligned} & u_0 = ... = u_p = 0 \\ & u_{m-p} = ... = u_m = 1 \\ & u_{j+p} = \frac{j}{n-p+1} \\ & per \\ & j = 1,...,n-p \end{aligned} \qquad \textbf{(2-18)}$$

Questo metodo, però, è sconsigliato in quanto, se utilizzato insieme all'equazione (2-15) può portare il sistema (2-12) ad essere singolare. Il vettore dei nodi, quindi, viene di solito scelto con un tecnica di *averaging* come segue:

$$\begin{aligned} & u_0 = ... = u_p = 0 \\ & u_{m-p} = ... = u_m = 1 \\ & u_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} \bar{u}_i \\ & per \\ & j = 1,...,n-p \end{aligned} \qquad \textbf{(2-19)}$$

Questa tecnica porta a un vettore di nodi che rispecchia la distribuzione del parametro $\bar{u}_k$; inoltre, utilizzando l'equazione (2-19) in combinazione con la (2-15) o la (2-17), si ottiene un sistema definito positivo, limitato con semi-larghezza di banda pari al massimo a *p*, cioè $N_{i,p}\left(\bar{u}_k\right) = 0$ per ogni $|i-k| \geq p$, che può essere risolto mediante eliminazione gaussiana senza pivoting.

In seguito è riportato un esempio in un caso semplice, bidimensionale (*r = 2*), e con soli 5 punti.

Sia $\{Q_k\} = \{(0,0),(3,4),(-1,4),(-4,0),(-4,-3)\}$, e supponiamo di voler interpolare questi punti mediante una B-Spline di grado *p = 3*. Faremo uso delle equazioni (2-15) e (2-19) per calcolare i parametri $\bar{u}_k$ e i nodi u_j, per poi scrivere e risolvere il sistema lineare definito dall'equazione (2-12). Le lunghezze dei segmenti della spezzata sono:

$$|Q_1 - Q_0| = 5 \quad |Q_2 - Q_1| = 4 \quad |Q_3 - Q_2| = 5 \quad |Q_4 - Q_3| = 3$$

e la lunghezza totale della spezzate è *d* = 17 (Figura 2-2). Di conseguenza avremo:

$$\bar{u}_0 = 0 \qquad \bar{u}_1 = \frac{5}{17} \qquad \bar{u}_2 = \frac{9}{17} \qquad \bar{u}_3 = \frac{14}{17}$$

$$\bar{u}_4 = 1;$$

utilizzando l'equazione (2-19), otteniamo:

$$u_4 = \frac{1}{3}\left(\frac{5}{17} + \frac{9}{17} + \frac{14}{17}\right) = \frac{28}{51}$$

quindi

$$U = \left\{0,0,0,0,\frac{28}{51},1,1,1,1\right\}$$

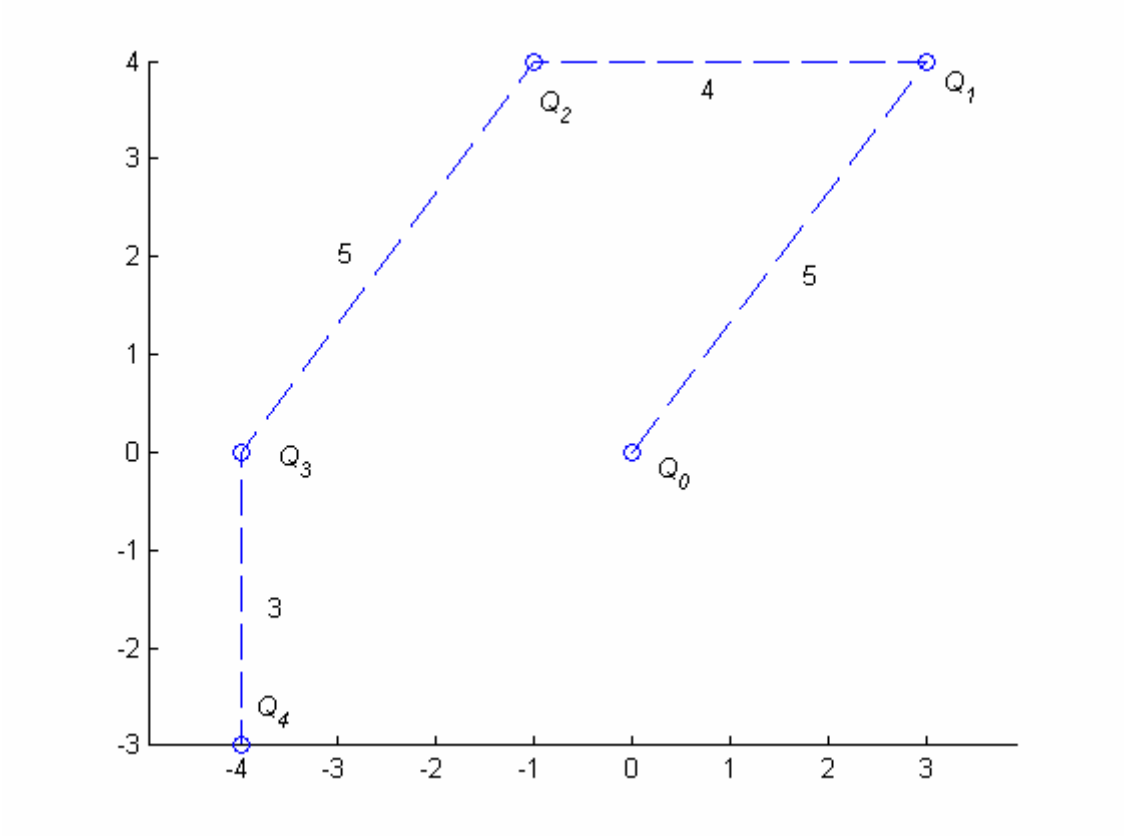


Figura 2-2 Esempio di interpolazione globale

Le funzioni di base necessarie per la valutazione dei punti di controllo sono le $N_{i,3}$ e sono illustrate nella Figura 2-3.

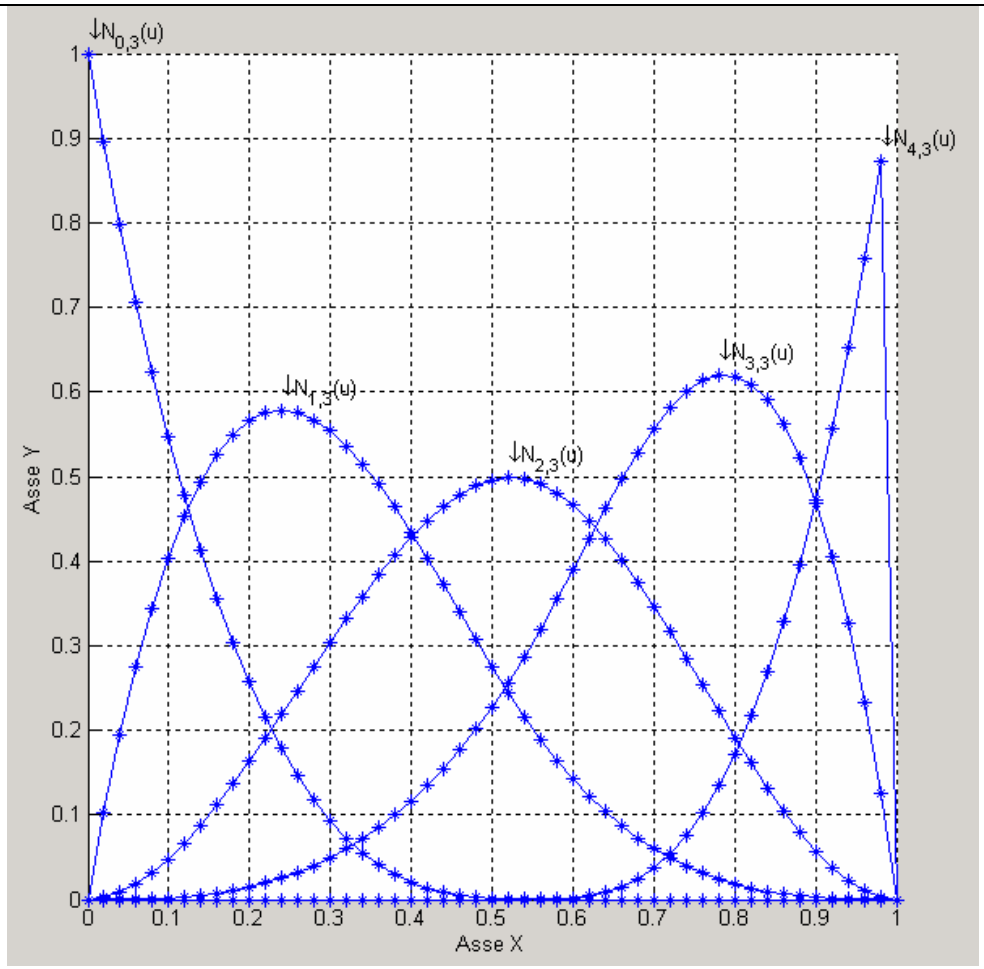


Figura 2-3 Funzioni di base $N_{i,3}$ definite sul vettore di nodi $U = \left\{0,0,0,0,\frac{28}{51},1,1,1,1\right\}$.

Il sistema risulta allora:

$$A \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = \begin{bmatrix} Q_0 \\ Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix}$$

dove la matrice A del sistema è definita dall'equazione (2-12) e risulta:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ N_{0,3}\left(\frac{5}{17}\right) & N_{1,3}\left(\frac{5}{17}\right) & N_{2,3}\left(\frac{5}{17}\right) & N_{3,3}\left(\frac{5}{17}\right) & 0 \\ N_{0,3}\left(\frac{9}{17}\right) & N_{1,3}\left(\frac{9}{17}\right) & N_{2,3}\left(\frac{9}{17}\right) & N_{3,3}\left(\frac{9}{17}\right) & 0 \\ 0 & N_{1,3}\left(\frac{14}{17}\right) & N_{2,3}\left(\frac{14}{17}\right) & N_{3,3}\left(\frac{14}{17}\right) & N_{4,3}\left(\frac{14}{17}\right) \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

cioè:

$$\begin{vmatrix} 1.0000 & 0 & 0 & 0 & 0 \\ 0.1001 & 0.5580 & 0.2956 & 0.0463 & 0 \\ 0.0000 & 0.2310 & 0.4987 & 0.2703 & 0 \\ 0 & 0.0122 & 0.1558 & 0.6065 & 0.2255 \\ 0 & 0 & 0 & 0 & 1.0000 \end{vmatrix}$$

Questa matrice è la stessa per i due sistemi (uno per ogni coordinata), cioè:

$$A \cdot \begin{bmatrix} P^x{}_0 \\ P^x{}_1 \\ P^x{}_2 \\ P^x{}_3 \\ P^x{}_4 \end{bmatrix} = \begin{bmatrix} Q^x{}_0 \\ Q^x{}_1 \\ Q^x{}_2 \\ Q^x{}_3 \\ Q^x{}_4 \end{bmatrix} \quad \text{e} \quad A \cdot \begin{bmatrix} P^y{}_0 \\ P^y{}_1 \\ P^y{}_2 \\ P^y{}_3 \\ P^y{}_4 \end{bmatrix} = \begin{bmatrix} Q^y{}_0 \\ Q^y{}_1 \\ Q^y{}_2 \\ Q^y{}_3 \\ Q^y{}_4 \end{bmatrix}$$

il risultato è riportato qui sotto e in Figura 2-4.

$\{P\}$ ={(0 , 0) , (7.3170 , 3.6868) , (-2.9581 , 6.6783) , (-4.4950 , -0.6737) , (-4.0000 , -3.0000}.

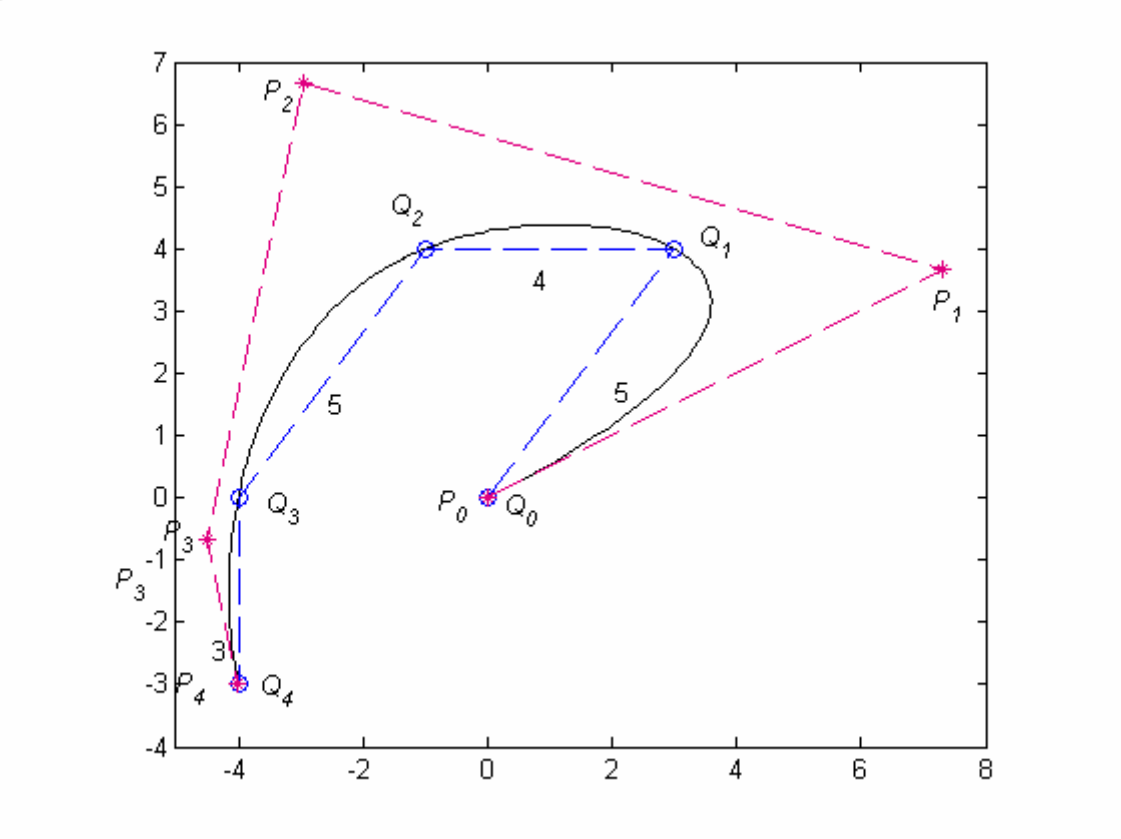


Figura 2-4 Esempio di interpolazione globale

Con questo tipo di interpolazione è possibile interpolare una serie qualunque di dati in maniera esatta. Una caratteristica di questo tipo di approccio è che di solito non viene mantenuta la col linearità di punti (Si osservi la Figura 2-9). Questo accade anche nel caso di

interpolazioni di superfici, rendendo difficile interpolare dati complanari in presenza di gradini.

Esistono in letteratura alcuni algoritmi basati sul precedente metodo che accettano come ingresso anche la derivata agli estremi, consentendo così di creare una curva interpolante con le derivate iniziale e finale assegnate. Non le tratteremo esplicitamente in questo lavoro in quanto non estensivamente utilizzate nel nostro lavoro; noteremo solo come sia possibile tramite l'uso delle proprietà della NURBS modificare una curva esistente per poter assegnare le derivate agli estremi.

2.4.2. Interpolazione globale con derivate agli estremi assegnate

Partendo da una curva NURBS ottenuta con l'algoritmo precedentemente mostrato, è possibile manipolare la curva stessa per ottenere derivate iniziali assegnate. Data la curva NURBS che interpola globalmente i dati in Figura 2-5, sappiamo dalle proprietà delle curve B-Spline, la curva definita dalla formulazione (1-19) interpola gli estremi e la derivata iniziale è proporzionale al vettore Q_1-Q_0 mentre la derivata finale è proporzionale al vettore Q_{n-1}-Q_n. Questa proprietà suggerisce che, modificando la posizione del secondo e del penultimo punto di controllo, è possibile forzare la curva ad avere le prefissate derivate agli estremi.

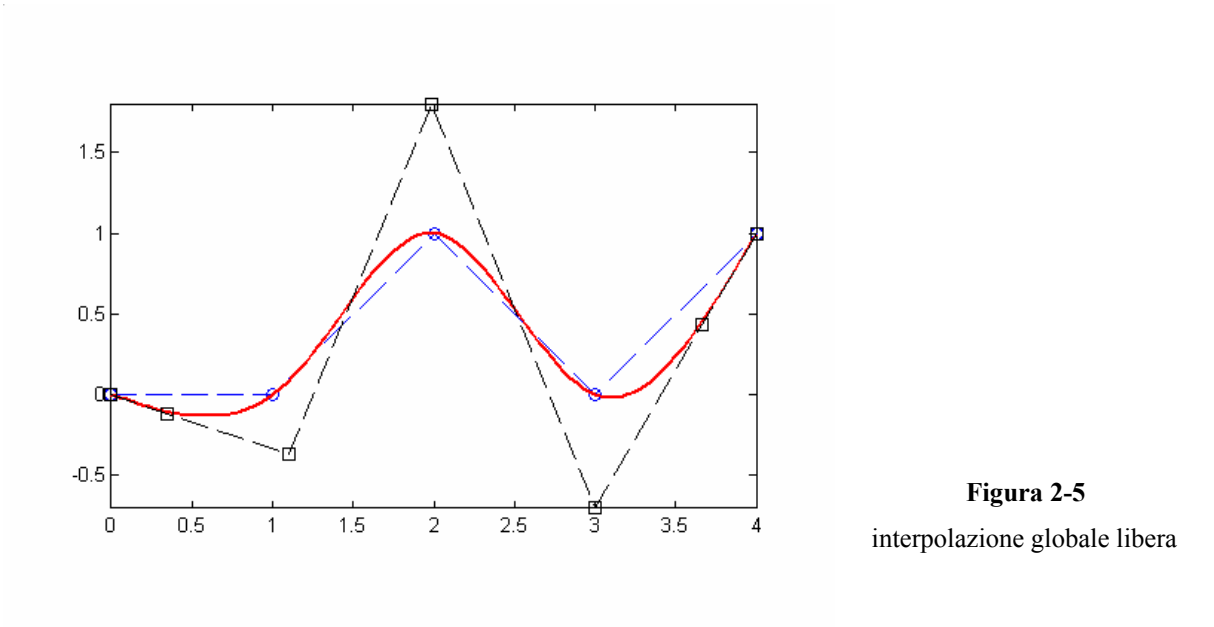


Figura 2-5
interpolazione globale libera

Tuttavia non è detto che modificando la posizione di un punto di controllo, la curva continui ad interpolare i dati (un esempio è illustrato in Figura 2-6). Per risolvere questo problema, basta ricordare la proprietà vi del paragrafo 1.3.4: la modifica del punto P_1 modifica la curva solo nell'intervallo $[u_1,u_5]$ per cui, aggiungendo un nodo (vedi paragrafo 2.3) e il corrispondente punto di controllo, si ottiene che il secondo punto di interpolazione è

al di fuori dell'intervallo di influenza del secondo punto di controllo (è il suo estremo), per cui non è influenzato dalla modifica effettuata (Figura 2-7).

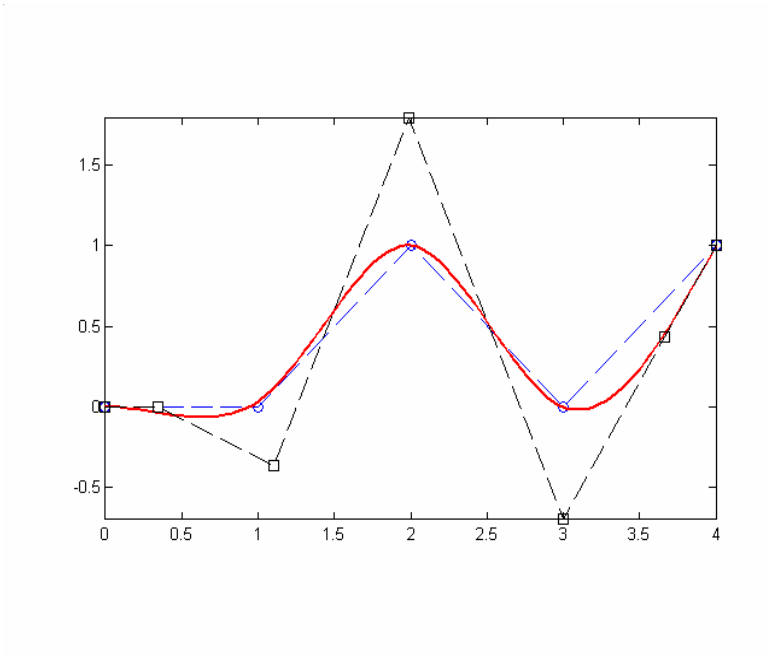


Figura 2-6 modifica della posizione del secondo punto di controllo per ottenere derivata iniziale orizzontale. La curva non interpola più il secondo punto

Nella Figura 2-8 si vede la curva originale con derivate agli estremi imposte sia sul punto iniziale che sul punto finale.

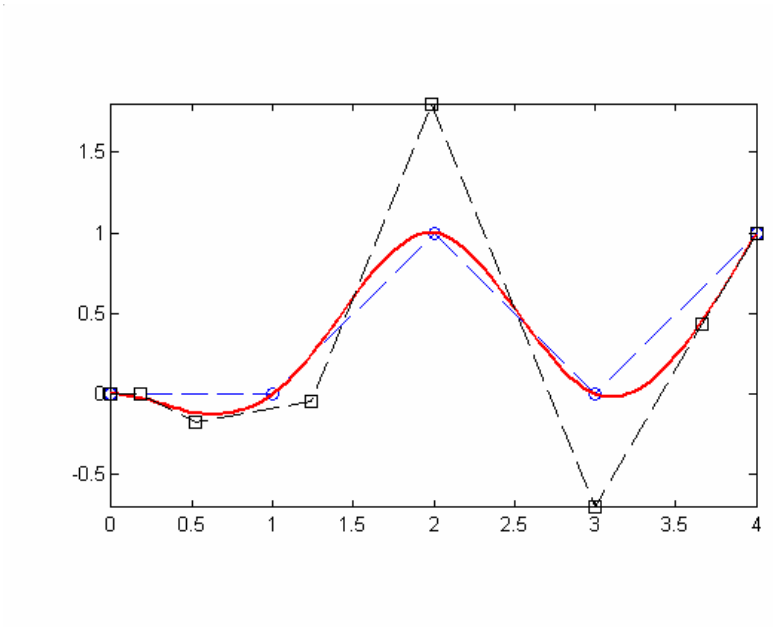


Figura 2-7 modifica della posizione del secondo punto di controllo dopo avere aggiunto un nodo per ottenere derivata iniziale orizzontale. La curva interpola ancora il secondo punto.

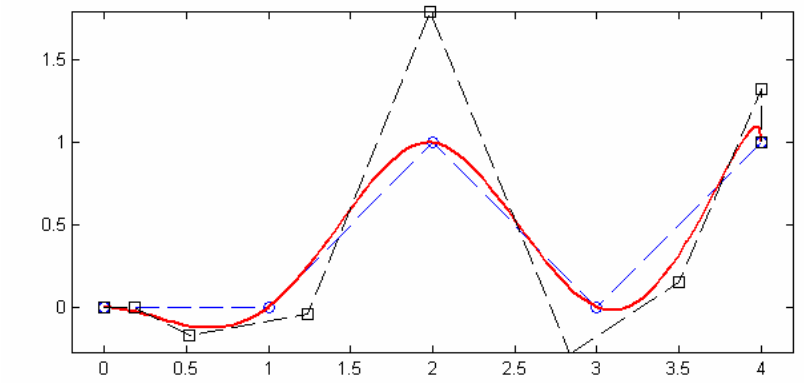


Figura 2-8
Interpolazione globale con derivate agli estremi assegnate.

Nel prossimo paragrafo ci occuperemo invece dell'interpolazione di griglie di punti mediante superfici B-Spline.

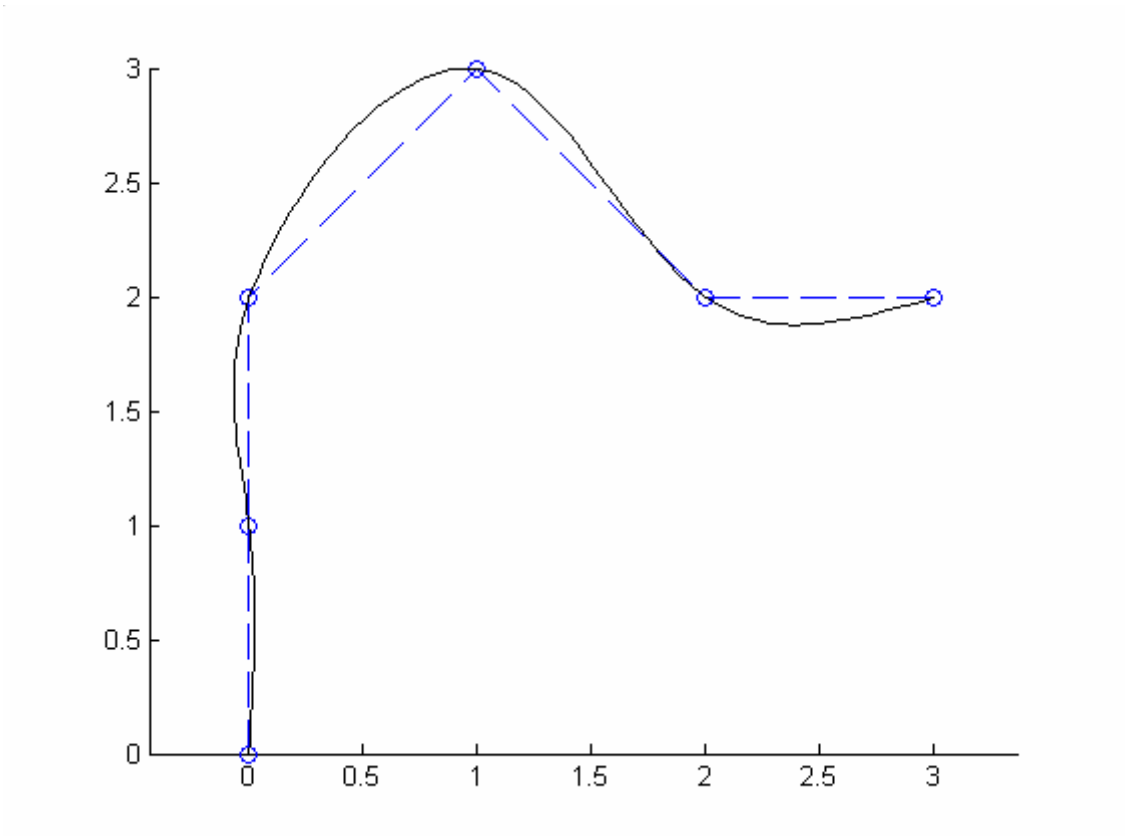


Figura 2-9 Esempio di interpolazione globale con punti allineati

2.4.3. Interpolazione globale di superfici

In questo paragrafo vedremo come creare una superficie B-Spline che interpoli esattamente una griglia di $(n + 1) \times (m + 1)$ punti. Siano dunque $\{Q_{k,l}\}$, $k=0,\ldots,n$, $l=0,\ldots,m$ i

punti da interpolare, e supponiamo di voler costruire una superficie B-Spline di grado (*p,q*), cioè:

$$Q_{k,l} = S(\bar{u}_k, \bar{v}_l) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(\bar{u}_k) N_{j,q}(\bar{v}_l) P_{i,j} \qquad \textbf{(2-20)}$$

Anche in questo caso la priorità è calcolare valori ragionevoli per i parametri ($\bar{u}_k$, $\bar{v}_l$) e per i vettori dei nodi *U* e *V*. Esiste un metodo abbastanza semplice che fornisce buoni risultati, e che è stato scelto nel nostro lavoro. Usando le equazioni (2-15) e (2-17) si calcolano i parametri $\bar{u}_0{}^l$, … , $\bar{u}_n{}^l$ per tutti gli *l*, poi viene fatta una media fra tutti gli $\bar{u}_k{}^l$, *l* = 0, … ,*n* per ottenere gli $\bar{u}_k$, cioè:

$$\bar{u}_k = \frac{1}{m+1}\sum_{l=0}^{m} \bar{u}_k{}^l \qquad \textbf{(2-21)}$$

La procedura viene ripetuta analogamente per ottenere i parametri $\bar{v}_l$. Esistono diversi modi di implementare questa procedura, alcuni dei quali si distinguono per la rapidità, altri per l'efficiente utilizzo della memoria; la scelta ricadrà sugli algoritmi più veloci, visto che su un moderno calcolatore il requisito di memoria è ampiamente soddisfatto.

Una volta ottenuti i parametri $\bar{u}_k$ e $\bar{v}_l$, i corrispondenti vettori dei nodi *U* e *V* verranno costruiti per mezzo dell'equazione (2-19).

Adesso si considera il calcolo dei punti di controllo delle superfici. L'equazione (2-20) rappresenta un sistema di (*n* + 1) × (*m* + 1) equazioni lineari nelle incognite $P_{i,j}$, tuttavia esiste un metodo più efficiente dal punto di vista della procedura numerica, che sfrutta la proprietà della superficie *S(u,v)*; infatti la (2-20) può essere riscritta nel modo seguente:

$$Q_{k,l} = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k)\left(\sum_{j=0}^{m} N_{j,q}(\bar{v}_l) P_{i,j}\right) = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k) R_{i,l} \qquad \textbf{(2-22)}$$

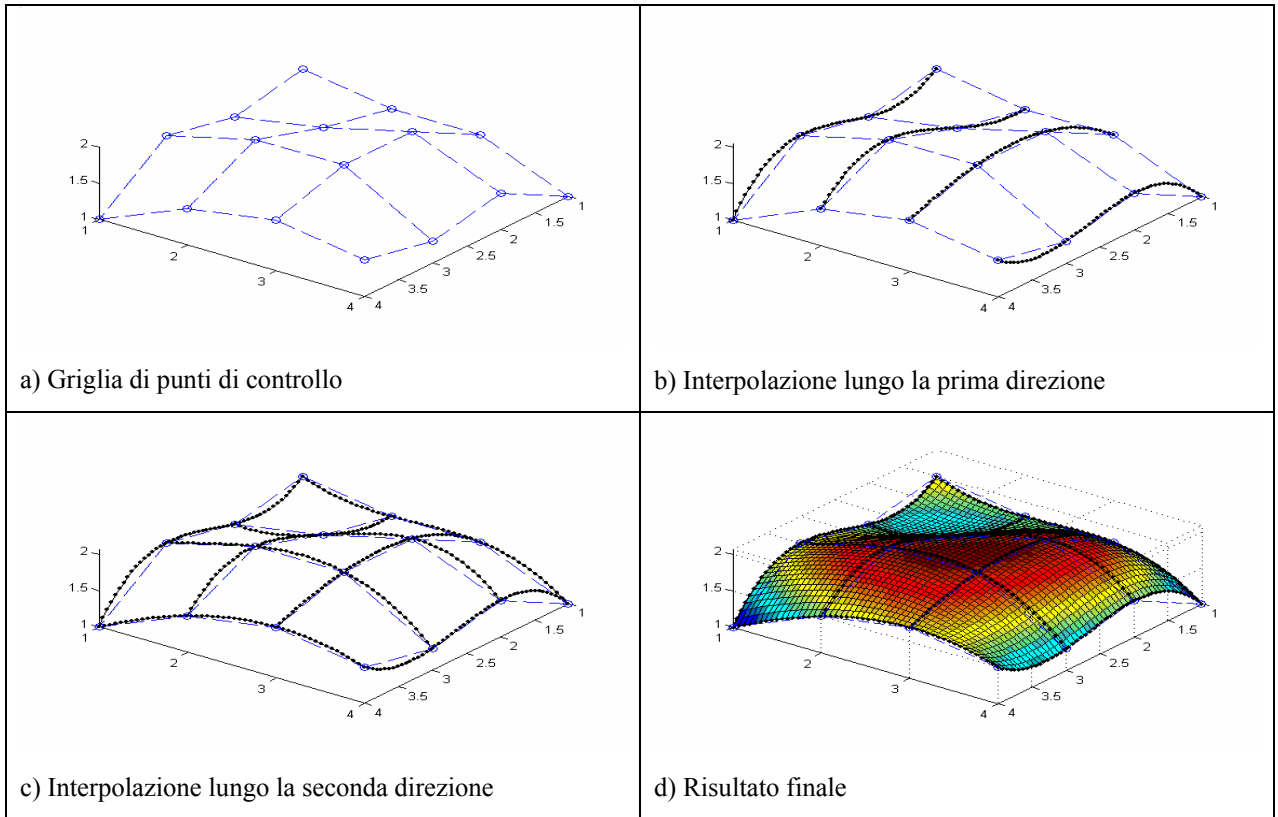
dove:

$$R_{i,l} = \sum_{j=0}^{m} N_{j,q}(\bar{v}_l) P_{i,j} \qquad \textbf{(2-23)}$$

L'equazione (2-22) può essere interpretata come la generazione dei punti di controllo primain una direzione, rappresentata dalla (2-22) (e geometricamente in Figura 2-10b), e poi nella seconda relativamente ai punti di controllo nella precedente (equazione (2-23)) ottenedo i pinti di controllo $P_{i,j}$. L'algoritmo per calcolare i punti di controllo è il seguente (Figura 2-10):

1. Utilizzando *U* e $\bar{u}_k$, si eseguono *m* + 1 interpolazioni di curve attraverso i punti $Q_{0,l}$, ... , $Q_{n,l}$ (per *l = 0 , ... , m*); in questo modo si calcolano gli $R_{i,l}$

2. Utilizzando V e $\bar{v}_l$, si eseguono $n + 1$ interpolazioni di curve attraverso i punti $R_{i,0}$, ... , $R_{i,m}$ (per $i = 0$, ... , n); in questo modo si calcolano i $P_{i,j}$



a) Griglia di punti di controllo

b) Interpolazione lungo la prima direzione

c) Interpolazione lungo la seconda direzione

d) Risultato finale

Figura 2-10 Procedura per la creazione di una superficie interpolante B-Spline mediante interpolazione globale.

Ovviamente, essendo l'algoritmo appena descritto simmetrico, si ottiene lo stesso risultato sia interpolando prima in direzione u poi in direzione v sia facendo l'opposto.

2.5. L'interpolazione locale

2.5.1. Interpolazione di curve

Sia $\{Q_k\}$, $k=0,...,n$, un insieme di punti da interpolare. La notazione "interpolazione locale" consiste in un metodo di costruzione della curva interpolante che costruisce n segmenti di curva polinomiali o razionali, $C_i(u)$, $i = 0$, ... , $n-1$, con Q_i e Q_{i+1} estremi di $C_i(u)$ e che i segmenti siano uniti con un assegnato livello di continuità. Per quanto riguarda le NURBS, questo procedimento risulta nella costruzione di $n-1$ segmenti di Bézier che rispondano alle caratteristiche date, e alla successiva scelta di un appropriato vettore di nodi. Sia $\bar{u}_i$ il parametro iniziale di $C_i(u)$ (e finale di $C_{i-1}(u)$) in modo che $C_i(u)$ e $C_{i-1}(u)$ si incontrano in $\bar{u}_i$; in questo lavoro verranno prese in considerazione solo le condizioni di

continuità C^1 e G^1 (vedi paragrafo 2.2), e curve di grado 3, per le stesse ragioni viste nel paragrafo precedente.

Per ottenere i segmenti di Bézier di grado 3 è necessario calcolare i due punti intermedi, ognuno dei quali giace sulla tangente alla curva passante per l'estremo più vicino (Figura 2-11). Per questo motivo, per costruire i vari segmenti di Bézier, è necessario conoscere oltre ai punto da interpolare, anche le derivate in quei punti. In alcuni casi queste derivate possono essere note, in altri casi andranno stimate direttamente dall'algoritmo di interpolazione, sulla base dei punti vicini.

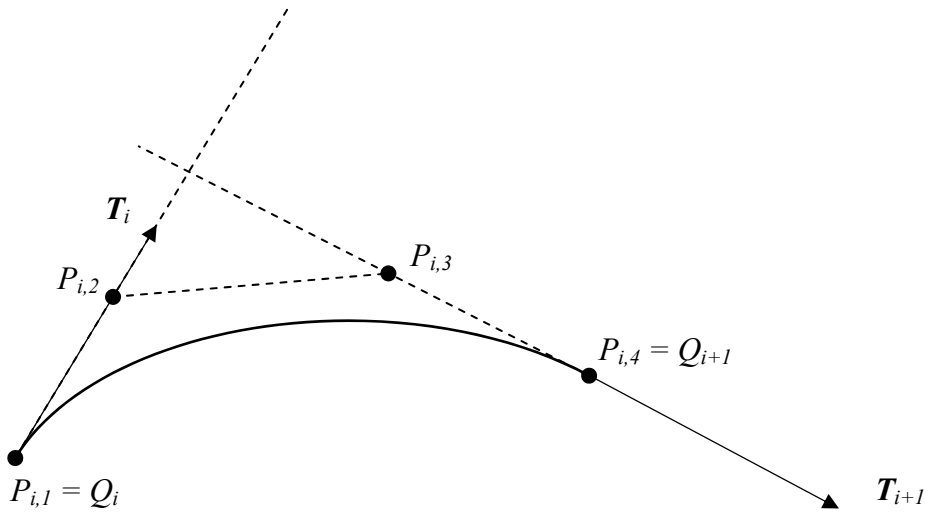


Figura 2-11 Punti di controllo di un segmento di Bézier

Per la stima delle derivate, esistono vari metodi, tra cui il seguente, (il più interessante per i nostri scopi):

Sia:

$q_k = Q_k - Q_{k-1}$ il vettore congiungente due punti successivi, allora possiamo stimare la tangente nei punti Q_k come:

$$T_k = \frac{V_k}{|V_k|} \qquad \textbf{(2-24)}$$

dove:

$$V_k = (1-\alpha_k)q_k + \alpha_k q_{k+1}$$

La tangente così ottenuta dipende dal parametro α, il quale può essere calcolato in questo modo (detto 2metodo dei cinque punti):

$$\alpha_k = \frac{\left|q_{k-1} \times q_k\right|}{\left|q_{k-1} \times q_k\right| + \left|q_{k+1} \times q_{k+2}\right|} \qquad \textbf{(2-25)}$$

con

$k = 0,...,n$

Questo metodo ha il vantaggio di tracciare un segmento di Bézier rettilineo in corrispondenza di tre punti allineati. Il denominatore dell'equazione (2-25) si annulla se Q_{k-2}, Q_{k-1}, Q_k sono allineati e Q_k, Q_{k+1}, Q_{k+2} sono allineati. Questo accade se in Q_k la curva forma uno spigolo, oppure nel caso in cui ci sia un segmento rettilineo fra Q_{k-2} e Q_{k+2}. In questi casi il parametro α_k può essere definito in due modi:

- $\alpha_k = 1$, che implica $V_k = q_{k+1}$; questo produce uno spigolo in Q_k.
- $\alpha_k = ½$, che implica $V_k = ½(q_k + q_{k+1})$; questa scelta produce un arrotondamento dello spigolo in Q_k.

La stima delle tangenti agli estremi va trattata a parte in quanto non esistono punti da uno dei due lati. La formula in (2-25) necessita infatti della definizione di q_{-1}, q_0, q_{n+2}, q_{n+1}; La stima di questi vettori è stata effettuata nel modo seguente:

$$q_0 = 2\,q_1 - q_2 \qquad q_{-1} = 2\,q_0 - q_1$$
$$q_{n+1} = 2\,q_n - q_{n-1} \qquad q_{n+2} = 2\,q_{n+1} - q_n \qquad \textbf{(2-26)}$$

In questo modo è possibile ottenere una stima delle tangenti anche nei punti estremi della curva; nel caso in cui si abbia una curva chiusa e si voglia chiuderla con continuità almeno G^1, allora il metodo non è più adeguato e deve essere modificato come segue:

$$q_0 = q_n - q_{n-1} \qquad q_{-1} = q_{n-1} - q_{n-2}$$
$$q_{n+1} = q_1 - q_0 \qquad q_{n+2} = q_2 - q_1$$

questa modifica permette di eguagliare le tangenti nel punto di incontro degli estremi della curva, e di ottenere una curva di tipo "smooth" (G^1 *continua*).

Prima di descrivere la procedura per ottenere i punti di controllo, è interessante osservare come, per una curva di Bézier di grado 3, sia possibile imporre che la derivata sia la medesima agli estremi e nel punto di mezzo, cioè:

$$C(u), u \in [0,1]$$
$$\alpha = \left|C'(0)\right| = \left|C'\left(\frac{1}{2}\right)\right| = \left|C'(1)\right| \qquad \textbf{(2-27)}$$

Dalle (2-27) e (1-7) si ottiene:

$$P_1 = P_0 + \frac{1}{3}\alpha T_0$$
$$P_2 = P_3 - \frac{1}{3}\alpha T_3 \qquad \textbf{(2-28)}$$

Applicando la (1-7) anche al punto intermedio si ricava:

$$C'\left(\frac{1}{2}\right) = 6\left(P_1^2 - P_0^3\right) \qquad \textbf{(2-29)}$$

Dove P_1^2 e P_0^3 si ricavano facendo uso dell'algoritmo di DeCasteljau:

$$P_1^2 - P_0^3 = \frac{1}{8}\left(P_3 + P_2 - P_1 - P_0\right) \qquad \textbf{(2-30)}$$

Eguagliando quindi il valore assoluto della derivata ad α e sostituendo si ottiene:

$$16\alpha^2 = \alpha^2\left|T_0 + T_3\right|^2 - 12\alpha\left(P_3 - P_0\right)\cdot\left(T_0 + T_3\right) + 36\left|P_3 - P_0\right|^2 \qquad \textbf{(2-31)}$$

L'equazione (2-31) ha una sola soluzione positiva; sostituendo questa soluzione in (2-28) si ottengono i due punti di controllo intermedi cercati. Chiamiamo i punti di controllo di un segmento *k-esimo* : $P_{k,0} = Q_k$, $P_{k,1}$, $P_{k,2}$, $P_{k,3} = Q_{k+1}$; il procedimento per calcolare il vettore dei nodi è il seguente:

1. calcolare α mediante la (2-31) e calcolare i punti intermedi $P_{k,1}$ e $P_{k,2}$ mediante le (2-28)
2. $\bar{u}_{k+1} = \bar{u}_k + 3\left|P_{k,1} - P_{k,0}\right|$

L'algoritmo qui descritto crea *n* segmenti di Bézier ognuno dei quali ha derivata unitaria agli estremi, per cui la curva definita dai punti di controllo:

Q_0 , $P_{0,1}$, $P_{0,2}$, $P_{1,1}$, $P_{1,2}$, ..., $P_{n-2,1}$, $P_{n-1,1}$, $P_{n-1,2}$, Q_n

e dai nodi:

$$U = \left\{0,0,0,0,\frac{\bar{u}_1}{\bar{u}_n},\frac{\bar{u}_1}{\bar{u}_n},\frac{\bar{u}_2}{\bar{u}_n},\frac{\bar{u}_2}{\bar{u}_n},\ldots,\frac{\bar{u}_{n-1}}{\bar{u}_n},\frac{\bar{u}_{n-1}}{\bar{u}_n},1,1,1,1\right\}$$

è una curva B-Spline che interpola i punti assegnati con continuità C^1.

Nelle figure sottostanti sono rappresentate alcune curve ottenute con questo metodo.

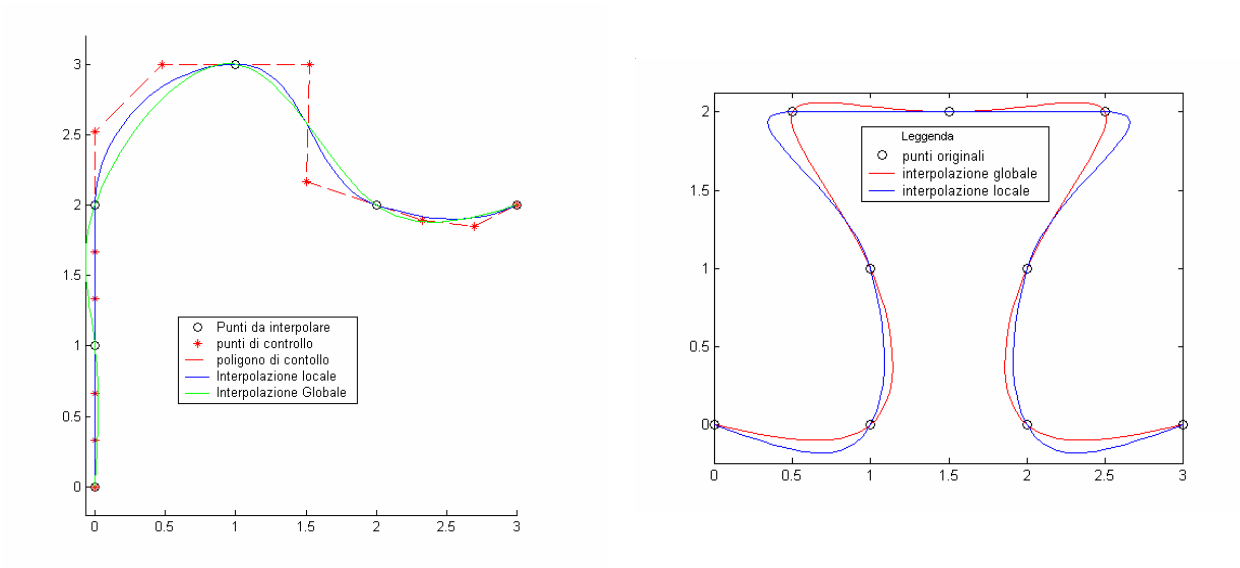


Figura 2-12 Confronto fra l'interpolazione globale e l'interpolazione locale in presenza di punti allineati

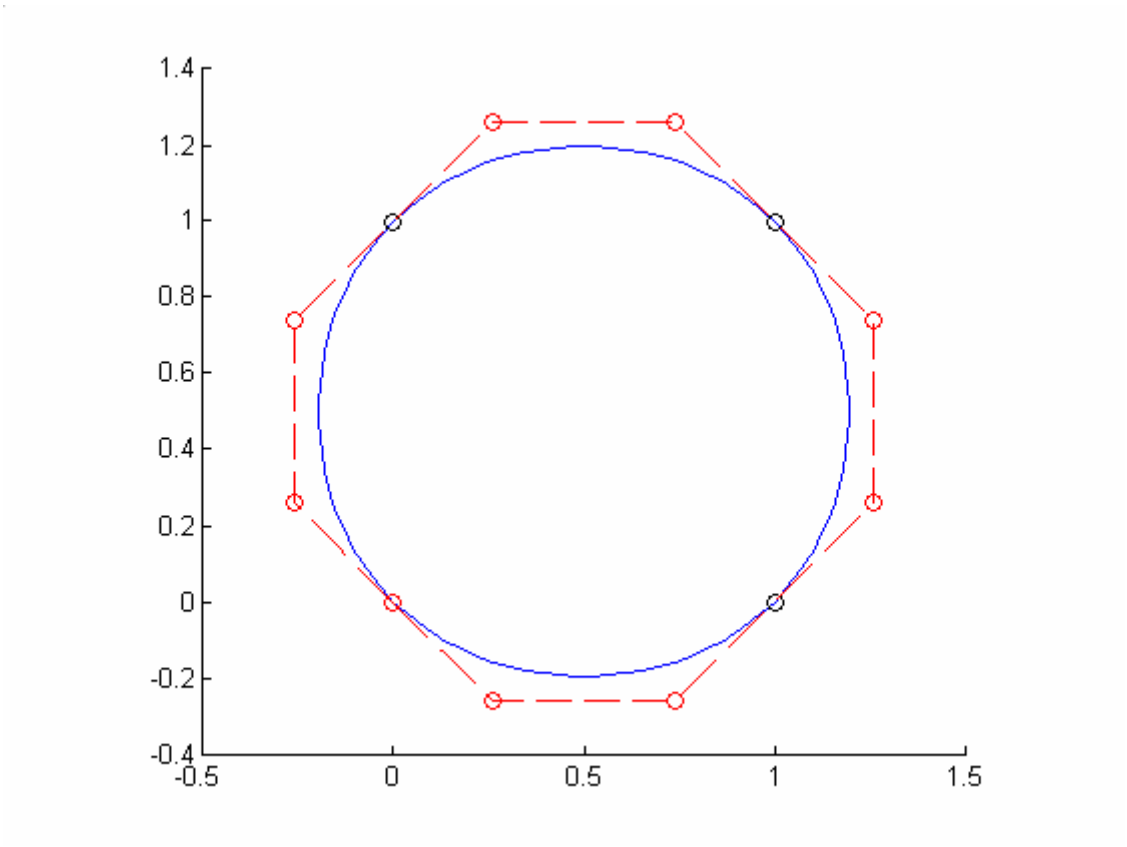


Figura 2-13 Esempio di interpolazione di curva chiusa, in questo caso, un'approssinazione dei una circonferenza mediante B-Spline cubica

Osservazione: per capire quanto questa approssimazione del cerchio sia vicina al cerchio che interpola esattamente i 4 punti dati (circonferenza con centro in [0.5 , 0.5] e raggio $r = \sqrt{2}$ possiamo fare 2 controlli: il primo è di vedere la distanza fra il cerchio

approssimato e quello reale, la seconda è di calcolare la curvatura κ del cerchio approssimato. Il primo controllo è rappresentato in forma grafica nella Figura 2-14, mentre per il secondo si è fatto uso della seguente formula:

$$\kappa(u) = \frac{|C'(u) \times C''(u)|}{|C'(u)|^3} \qquad \textbf{(2-32)}$$

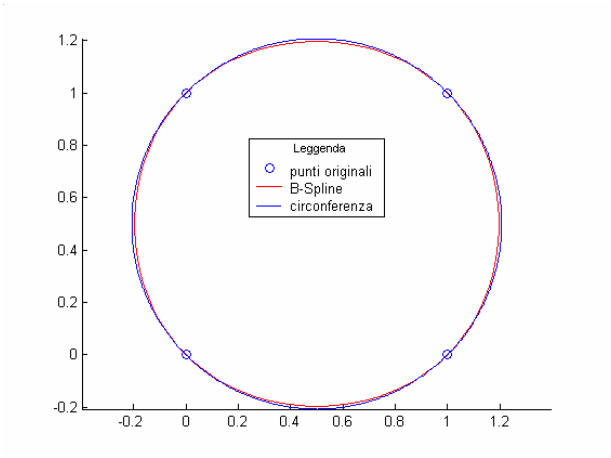


Figura 2-14 Confronto fra un'approssimazione della circonferenza mediante interpolazione di 4 punti mediante B-Spline cubica e effettiva circonferenza.

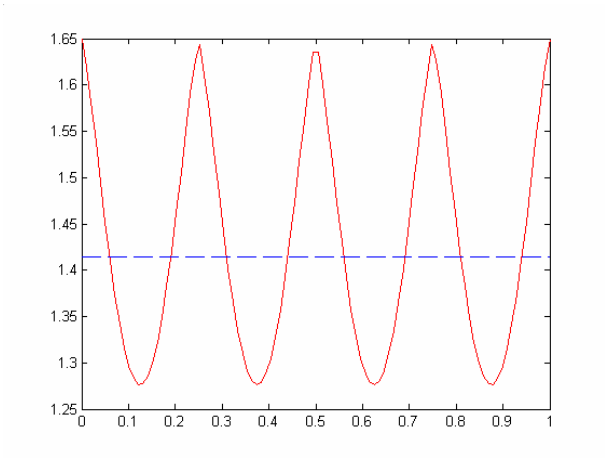


Figura 2-15 Andamento della curvatura in funzione del parametro *u* lungo la curva.

Nella Figura 2-16 e Figura 2-17 sono rappresentati due profili ottenuti interpolando mediante la procedura esposta in questo paragrafo i profili stessi definiti per punti. Si nota come sia possibile con questa procedura ottenere ottime approssimazioni dei dati da interpolare anche in presenza di punti angolosi o di forte curvatura.

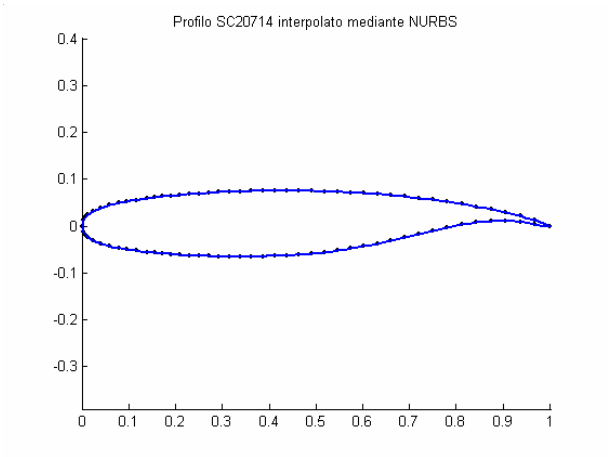


Figura 2-16 Esempio di interpolazione di un profilo supercritico

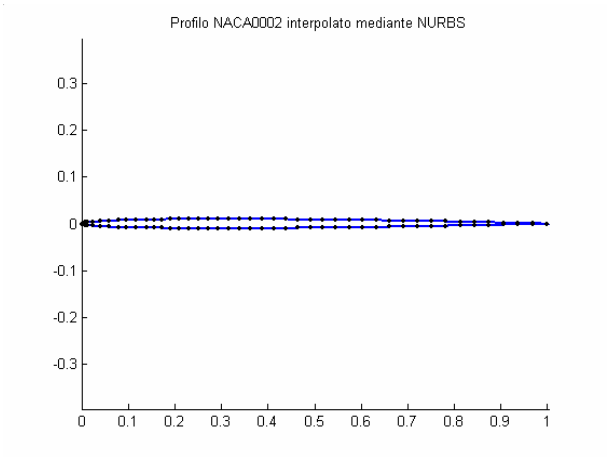
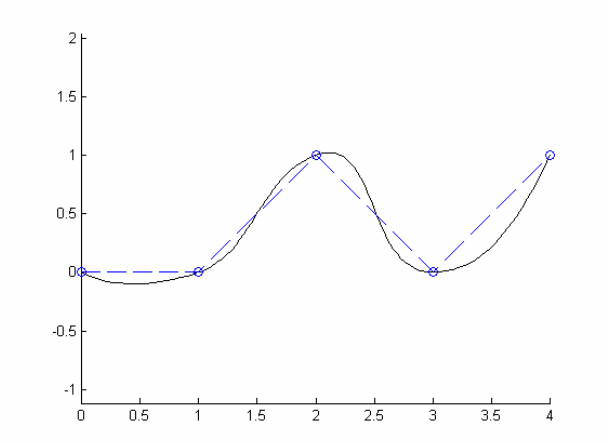


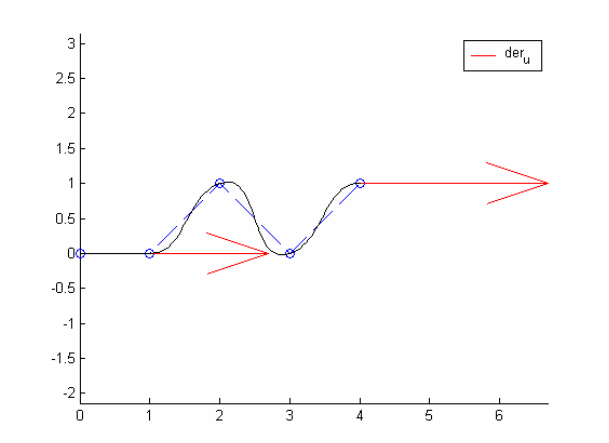
Figura 2-17 Esempio di interpolazione di un profilo sottile

2.5.2. Curve interpolanti con derivate agli estremi assegnate

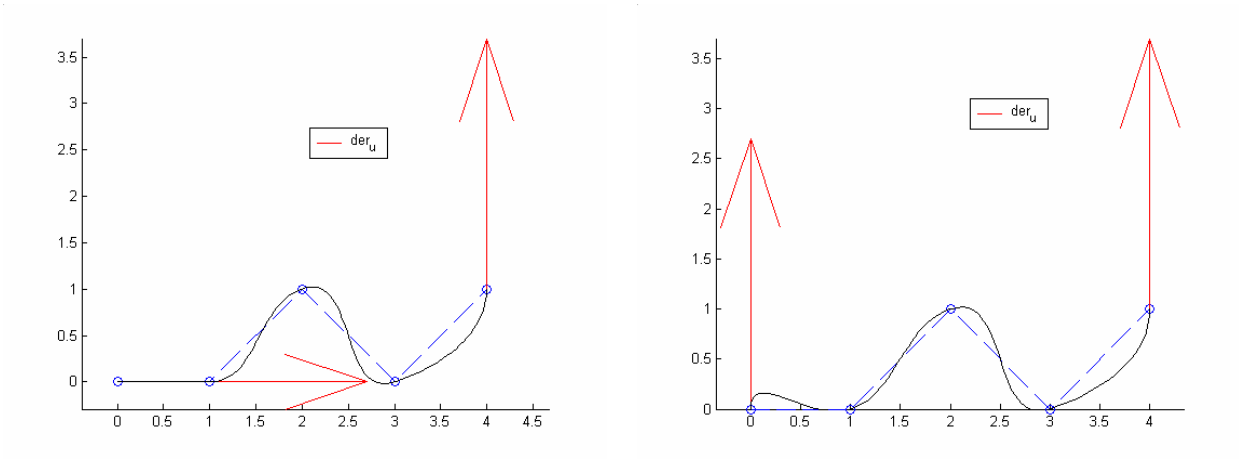
Modificando leggermente lo schema precedente è possibile imporre le derivate ai due estremi. Questa modifica si rivela fondamentale per poter garantire un continuità di classe G^1 tra due curve adiacenti create interpolando due set indipendenti di punti (vedi figure). Cominciamo notando che se sono presenti solo due punti da interpolare, le formule di stima (2-26) non sono applicabili, in quanto necessitano di almeno 3 punti. In questo caso l'algoritmo di interpolazione salta il calcolo delle (2-26) e sostituisce le derivate agli estremi con un versore parallelo alla direzione Q_1-Q_2, realizzando così una interpolazione lineare in quella direzione. Da questa modifica è chiaro come sia possibile modificare l'algoritmo per tenere conto di derivate introdotte dall'utente: è sufficiente sostituire il blocco di stima delle derivate al contorno con le derivate specificate in ingresso. Il risultato di questa modifica è illustrato nelle figure seguenti:



a)



b)



c) d)

Figura 2-18 a) Interpolazione locale senza specificare derivate al contorno; b) c) d) interpolazione locale con derivate al contorno assegnate, vari casi.

Come applicazione pratica di questo algoritmo, illustriamo di seguito come sia possibile unire con continuità G^1 due curve. La prima curva è la stessa di Figura 2-18 a), mentre la seconda è quella illustrata in Figura 2-19. Per unire le due curve con la continuità specificata, è necessario calcolare la derivata finale della prima curva e la derivata finale della seconda curva. La prima ci serve per imporre la continuità in $x = 4$ la seconda per garantire che la derivata finale della curva 2 rimanga inalterata. Una volta calcolate le derivate mediante le metodologie illustrate nel paragrafo 1.3.5, possiamo procedere alla creazione della curva 2 modificata imponendo le derivate appena calcolate. Il risultato è illustrato nella Figura 2-20.

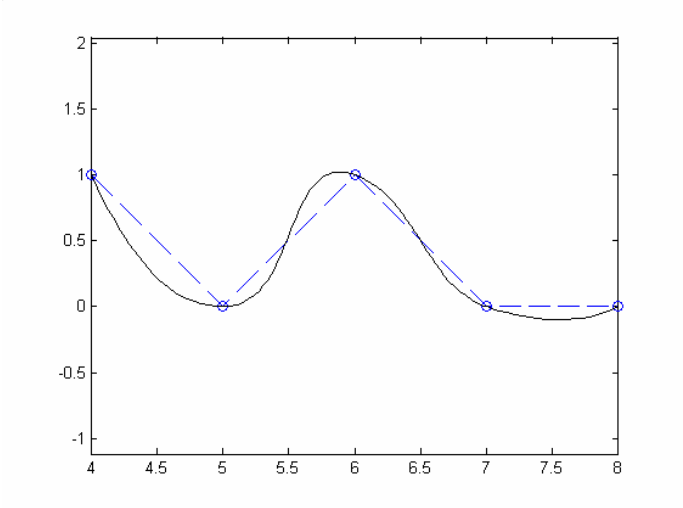


Figura 2-19

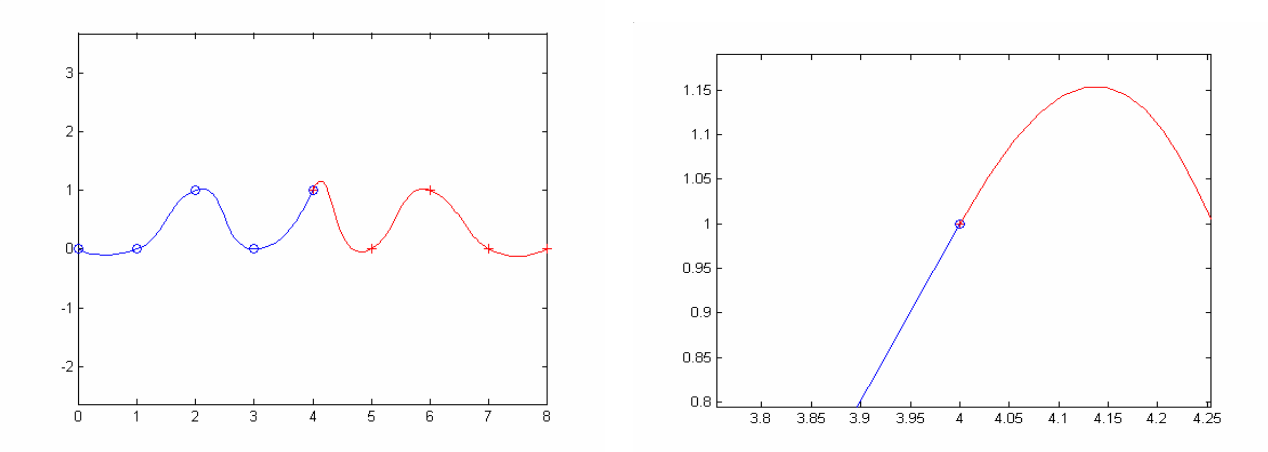


Figura 2-20 Raccordo di due curve NURBS con continuità G^1 mediante imposizione di derivate agli estremi

2.5.3. Interpolazione di superfici

Quanto visto nel paragrafo precedente può essere esteso al caso di interpolazione locale di superfici. Questo algoritmo ha una considerevole importanza per il nostro lavoro, come verrà sottolineato in seguito. Il risultato sarà, quindi, una superficie B-Spline bicubica e di classe $C^{1,1}$, cioè di classe C^1 in entrambe le direzioni. Sia $\{Q_{k,l}\}$, $k = 0,\ldots,n$ e $l = 0,\ldots,m$ una griglia di $(n + 1) \times (m + 1)$ punti e siano $\{(\bar{u}_k, \bar{v}_l)\}$ i corrispondenti valori dei parametri u e v calcolati mediante le equazioni (2-15) e (2-17) e (2-21), come descritto nel paragrafo 0. il procedimento seguente produce la superficie

$$S(\bar{u}_k, \bar{v}_l) = \sum_{i=0}^{2n+1} \sum_{j=0}^{2m+1} N_{i,3}(\bar{u}_k) N_{j,3}(\bar{v}_l) P_{i,j} \qquad \textbf{(2-33)}$$

costruendo nm superfici di Bézier bicubiche $\{B_{k,l}(u,v)\}, k = 0,...,n-1, l = 0,...,m-1$ nelle quali, $Q_{k,l}$, $Q_{k+1,l}$, $Q_{k,l+1}$, $Q_{k+1,l+1}$ sono gli spigoli, e che saranno unite con continuità C^1. Eccetto per i bordi della superficie, tutte le righe e le colonne di punti di controllo contenenti i punti originali $Q_{k,l}$,verranno rimosse, lasciando per la B-Spline finale $(2n + 2)(2m + 2)$ punti di controllo. I vettori dei nodi saranno, analogamente a quanto visto nel precedente paragrafo 0:

$$U = \{0,0,0,0,\bar{u}_1,\bar{u}_1,\bar{u}_2,\bar{u}_2 \ldots,\bar{u}_{n-1},\bar{u}_{n-1},1,1,1,1\}$$

e **(2-34)**

$$V = \{0,0,0,0,\bar{v}_1,\bar{v}_1,\bar{v}_2,\bar{v}_2 \ldots,\bar{v}_{n-1},\bar{v}_{n-1},1,1,1,1\}$$

Una superficie di Bézier bicubica ha 16 punti di controllo, 4 interni e 12 sul bordo. I 12 punti sul bordo vengono trovati all'inizio dell'algoritmo per tutte le superfici di Bézier interpolando prima le (m +1) righe poi le (n + 1) colonne usando il metodo descritto al

paragrafo precedente, con l'unica differenza che questa volta i parametri $\{(\bar{u}_k, \bar{v}_l)\}$ li abbiamo già calcolati per altra via. Questo significa che in questo caso possiamo solo imporre che la derivata direzionale delle superfici di Bézier sia 1 solo sui bordi; anche se questa condizione ci garantisce comunque la continuità $C^{1,1}$, non potendo imporre anche la derivata nel punto intermedio pari a 1 potrebbe portare ad una parametrizzazione peggiore che nel caso precedente. I parametri $\{(\bar{u}_k, \bar{v}_l)\}$ sono stati calcolati prima in quanto, per una superficie B-Spline, la parametrizzazione lungo tutte le righe deve essere unica, come pure quella lungo le colonne; in altre parole, ci sono solo 2 vettori di nodi, ognuno dei quali è valido per tutte sequenze di punti nella sua direzione (vedi Figura 2-21).

La procedura, quindi, prevede come primo passo l'interpolazione lungo le due direzioni separatamente, nel modo seguente:

per ogni $l = l_0$ si costruisce la B-Spline cubica che interpola i punti $Q_{0,l_0},...,Q_{n,l_0}$; sia r_{l_0} la lunghezza totale della spezzata lungo la l_0 *–esima* riga, si calcoli la tangente T_{k,l_0} ad ogni punto Q_{k,l_0} mediante le equazioni (2-24) (2-25); allora i punti di controllo del segmento di curva di Bézier saranno:

$$\begin{aligned} P_{1,0}^{k,l_0} &= Q_{k,l_0} + aT_{k,l_0}^u \\ P_{2,0}^{k,l_0} &= Q_{k+1,l_0} - aT_{k+1,l_0}^u \end{aligned} \qquad \textbf{(2-35)}$$

dove:

$$a = \frac{r_{l_0}\left(\bar{u}_{k+1} - \bar{u}_k\right)}{3} = \frac{r_{l_0}\Delta\bar{u}_{k+1}}{3}$$

La curva che risulta da questi calcoli è una curva C^1 continua con derivate di valore assoluto pari a r_{l_0} in corrispondenza di tutti punti di controllo Q_{k,l_0}. Lo stesso procedimento è applicato a tutte le (n + 1) colonne.

Rimangono adesso da calcolare solo i 4 punti interni di ciascuna superficie di Bézier; per fare questo abbiamo bisogno di stimare tutte le derivate parziali miste, $D_{k,l}^{uv}$ in corrispondenza di ogni $Q_{k,l}$. Per la stima delle $D_{k,l}^{uv}$ utilizziamo il metodo a tre punti di Bessel, cioè:

siano r_l e s_k le lunghezze totali delle spezzate lungo la *l-esima* riga e *k-esima* colonna; allora:

$$\begin{aligned} D_{k,l}^u &= r_l T_{k,l}^u \\ D_{k,l}^v &= s_k T_{k,l}^v \end{aligned} \qquad \textbf{(2-36)}$$

sia:

$$d^{vu}_{k,l} = (1-\alpha_k)\frac{D^{v}_{k,l} - D^{v}_{k-1,l}}{\Delta\bar{u}_k} + \alpha_k \frac{D^{v}_{k+1,l} - D^{v}_{k,l}}{\Delta\bar{u}_{k+1}}$$

$$d^{uv}_{k,l} = (1-\beta_l)\frac{D^{u}_{k,l} - D^{u}_{k,l-1}}{\Delta\bar{v}_l} + \beta_l \frac{D^{u}_{k,l+1} - D^{u}_{k,l}}{\Delta\bar{v}_{l+1}}$$

dove:

(2-37)

$$\alpha_k = \frac{\Delta\bar{u}_k}{\Delta\bar{u}_k + \Delta\bar{u}_{k+1}}$$

$$\beta_l = \frac{\Delta\bar{v}_l}{\Delta\bar{v}_l + \Delta\bar{v}_{l+1}}$$

con $\Delta\bar{u}_k = \bar{u}_k - \bar{u}_{k-1}$ e $\Delta\bar{v}_l = \bar{v}_l - \bar{v}_{l-1}$

Per quanto riguarda i punti estremi, servono formule di stima; nel caso di derivate parziali miste, queste assumono la forma seguente:

prima e ultima riga:

$$d^{uv}_{k,0} = 2\frac{D^{u}_{k,1} - D^{u}_{k,0}}{\Delta\bar{v}_1} - d^{uv}_{k,1}$$

$$d^{uv}_{k,m} = 2\frac{D^{u}_{k,m} - D^{u}_{k,m-1}}{\Delta\bar{v}_m} - d^{uv}_{k,m-1} \qquad \text{con} \quad k = 1, \ldots, n\text{-}1$$

(2-38)

prima e ultima colonna:

$$d^{vu}_{0,l} = 2\frac{D^{v}_{1,l} - D^{v}_{0,l}}{\Delta\bar{u}_1} - d^{vu}_{1,l}$$

$$d^{vu}_{n,l} = 2\frac{D^{v}_{n,l} - D^{v}_{n-1,l}}{\Delta\bar{u}_n} - d^{vu}_{n-1,l} \qquad \text{con} \quad j = 1, \ldots, m\text{-}1$$

Infine, per i 4 spigoli valgono:

$$d^{uv}_{0,0} = 2\frac{D^{u}_{0,1} - D^{u}_{0,0}}{\Delta\bar{v}_1} - d^{uv}_{0,1}$$

$$d^{uv}_{n,m} = 2\frac{D^{u}_{n,m} - D^{u}_{n,m-1}}{\Delta\bar{v}_m} - d^{uv}_{n-1,m-1}$$

$$d^{vu}_{0,0} = 2\frac{D^{v}_{1,0} - D^{v}_{0,0}}{\Delta\bar{u}_1} - d^{vu}_{1,0}$$

$$d^{vu}_{n,m} = 2\frac{D^{v}_{n,m} - D^{v}_{n-1,m}}{\Delta\bar{u}_n} - d^{vu}_{n-1,m-1}$$

Infine possiamo calcolare le derivate parziali miste:

$$D_{k,l}^{uv} = \frac{\alpha_k d_{k,l}^{uv} + \beta_l d_{k,l}^{vu}}{\alpha_k + \beta_l} \qquad \textbf{(2-39)}$$

A questo punto è possibile calcolare i 4 punti di controllo intermedi di superficie di Bézier:

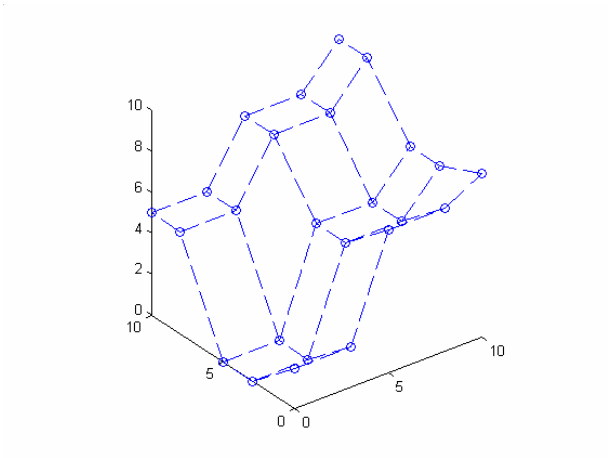
$$\begin{aligned} P_{1,1}^{k,l} &= \gamma D_{k,l}^{uv} + P_{0,1}^{k,l} + P_{1,0}^{k,l} - P_{0,0}^{k,l} \\ P_{2,1}^{k,l} &= -\gamma D_{k+1,l}^{uv} + P_{3,1}^{k,l} - P_{3,0}^{k,l} + P_{2,0}^{k,l} \\ P_{1,2}^{k,l} &= -\gamma D_{k,l+1}^{uv} + P_{1,3}^{k,l} - P_{0,3}^{k,l} + P_{0,2}^{k,l} \\ P_{2,2}^{k,l} &= \gamma D_{k+1,l+1}^{uv} + P_{2,3}^{k,l} + P_{3,2}^{k,l} - P_{3,3}^{k,l} \end{aligned} \qquad \textbf{(2-40)}$$

dove:

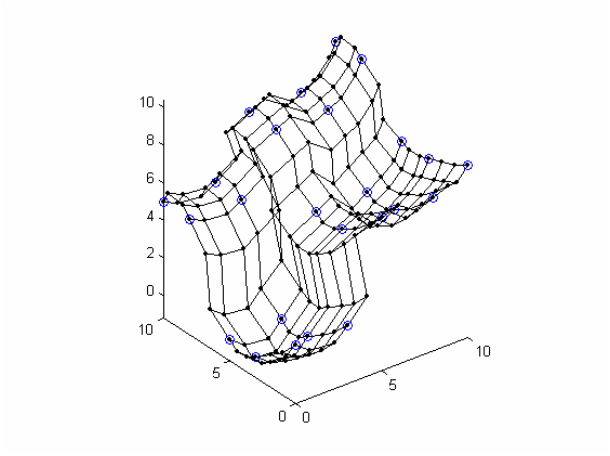
$$\gamma = \frac{\Delta\bar{u}_{k+1}\Delta\bar{v}_{l+1}}{9}$$

La superficie B-Spline cercata è definita dai punti di controllo calcolati in precedenza, sopprimendo tutte le righe e le colonne contenenti i punti da interpolare, ad eccezione delle due righe e delle 2 colonne estreme (Figura 2-21c).

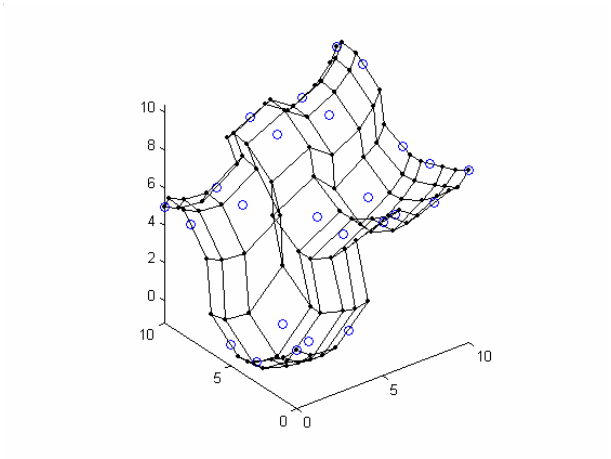
Nella figure sottostante sono evidenziati i vari passaggi:



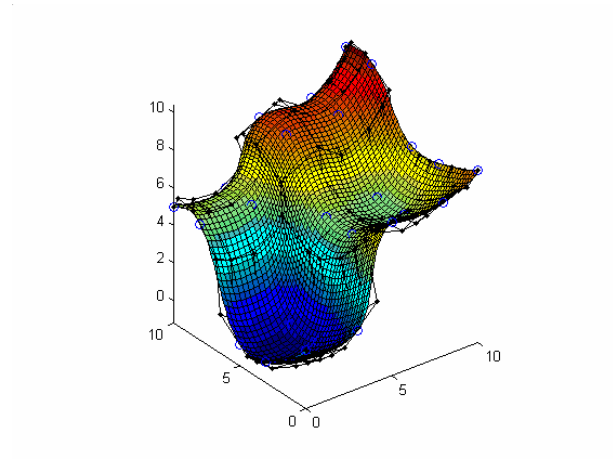
a) Griglia di punti di controllo



b) Calcolo dei punti di controllo nelle due direzioni separatamente



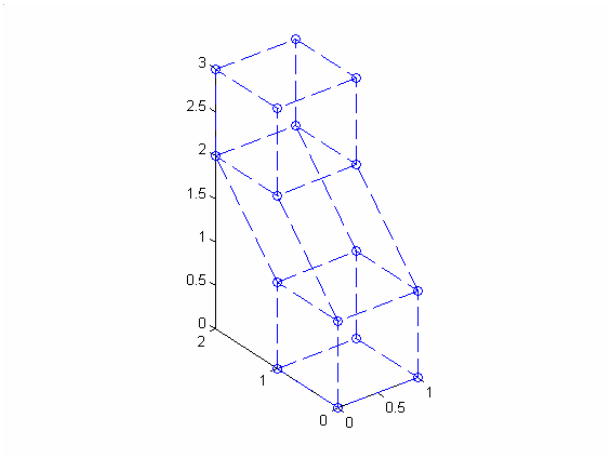
c) rimozione dei punti di controllo superflui



d) Risultato finale

Figura 2-21 Procedura per la creazione di una superficie interpolante B-Spline mediante interpolazione locale.

Mediante questa procedura sono facilmente ottenibili forme chiuse in una o in tutte e due le direzioni (*u,v*) come mostrato in Figura 2-23.



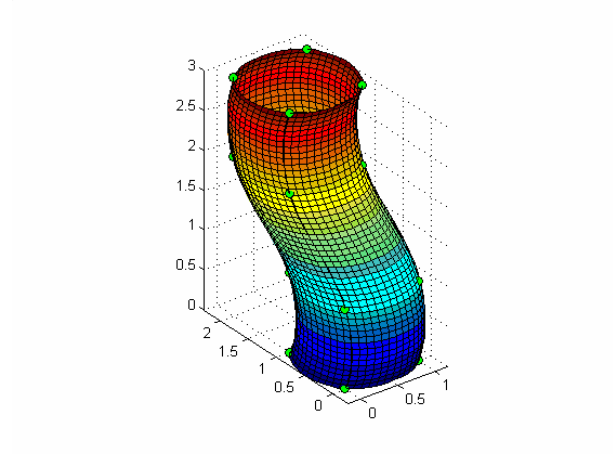
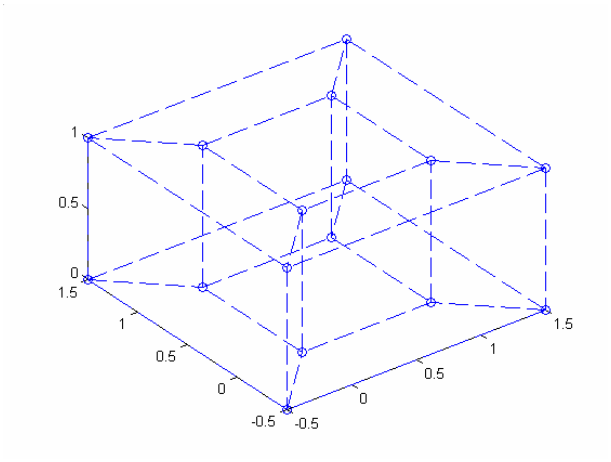


Figura 2-22 Esempio di interpolazione mediante B-Spline in caso di curva chiusa in una direzione.



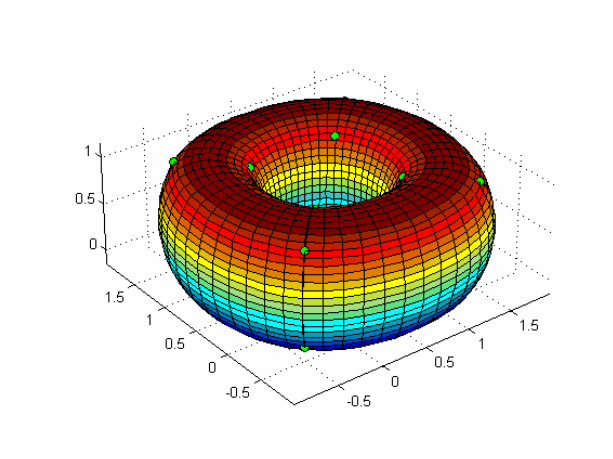
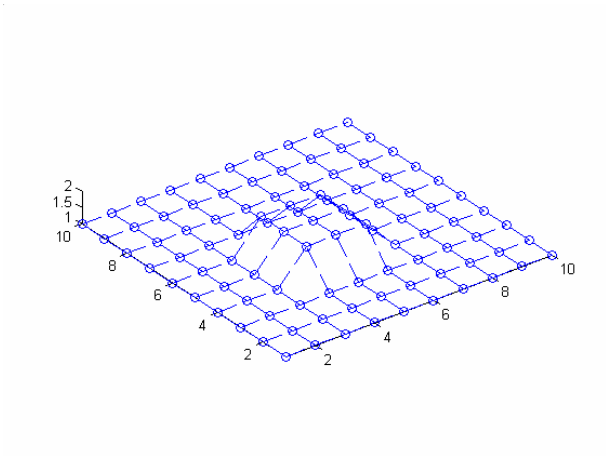
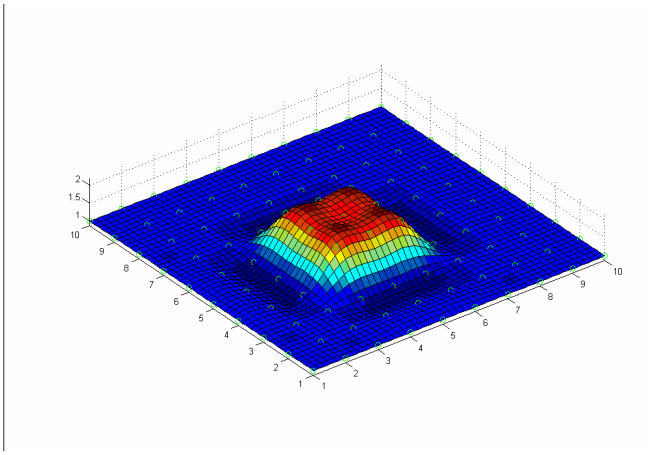


Figura 2-23 Esempio di interpolazione mediante B-Spline in caso di curva chiusa in entrambe le direzione.

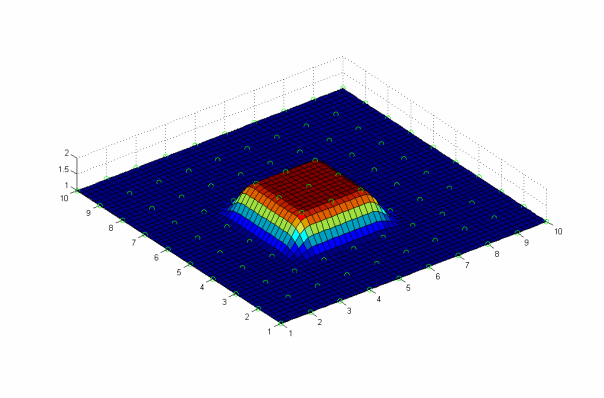
Una interessante conseguenza dell'algoritmo appena descritto, riguarda l'interpolazione di punti complanari. Nella Figura 2-24 ad esempio è mostrato come i dati in a) generino superfici molto differenti in caso vengano interpolati mediante un metodo globale b) o tramite un algoritmo locale, quale quello esposto in questo paragrafo.



a) griglia di punti complanari



b) interpolazione mediante metodo globale dei punti in a)



b) interpolazione mediante metodo locale dei punti in a)

Figura 2-24 Esempio di interpolazione mediante B-Spline in caso di punti complanari.

2.5.4. Superfici interpolanti con derivate agli estremi assegnate

Modificando leggermente lo schema precedente è possibile imporre le derivate parziali nelle due direzioni su tutti i lati. Questa modifica si è resa necessaria per poter garantire un continuità di classe G^1 tra due superfici adiacenti create interpolando due set indipendenti di punti (ad esempio, per unire il tip dell'ala a una paratia verticale o un winglet). Cominciamo notando che se in una delle due direzioni sono presenti solo due punti da interpolare, le formule di stima (2-38) non sono applicabili, in quanto necessitano di almeno 3 punti. In questo caso la funzione che esegue l'interpolazione sostituisce le derivate parziali nella direzione in questione con un versore parallelo alla direzione Q_1-Q_2, realizzando così una interpolazione lineare in quella direzione. Da questa modifica è chiaro come sia possibile modificare l'algoritmo per tenere conto di derivate introdotte dall'utente: è sufficiente

sostituire il blocco di stima delle derivate al contorno con le derivate specificate in ingresso. Il risultato di questa modifica è illustrato nelle figure seguenti.

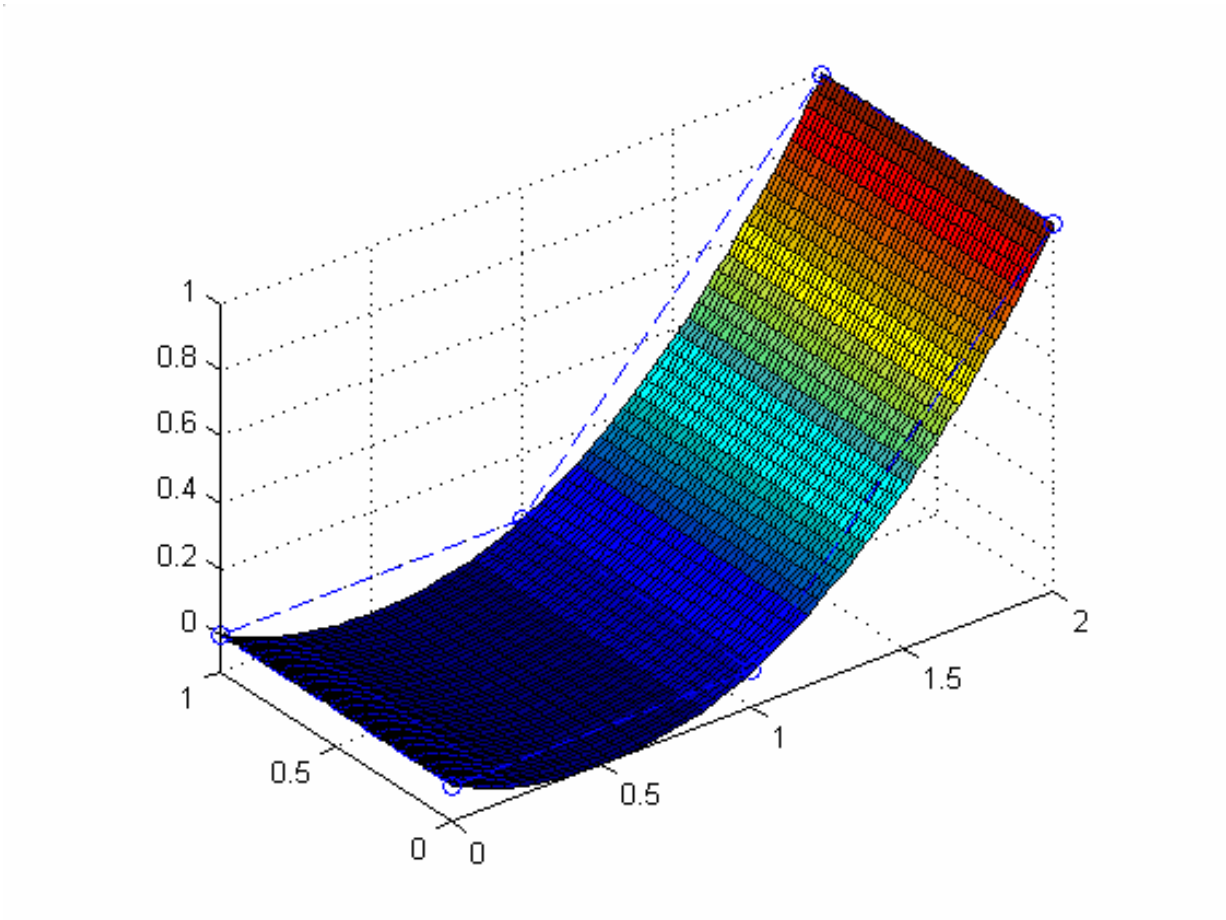


Figura 2-25 Interpolazione mediante B-Spline senza specificare alcuna derivata iniziale

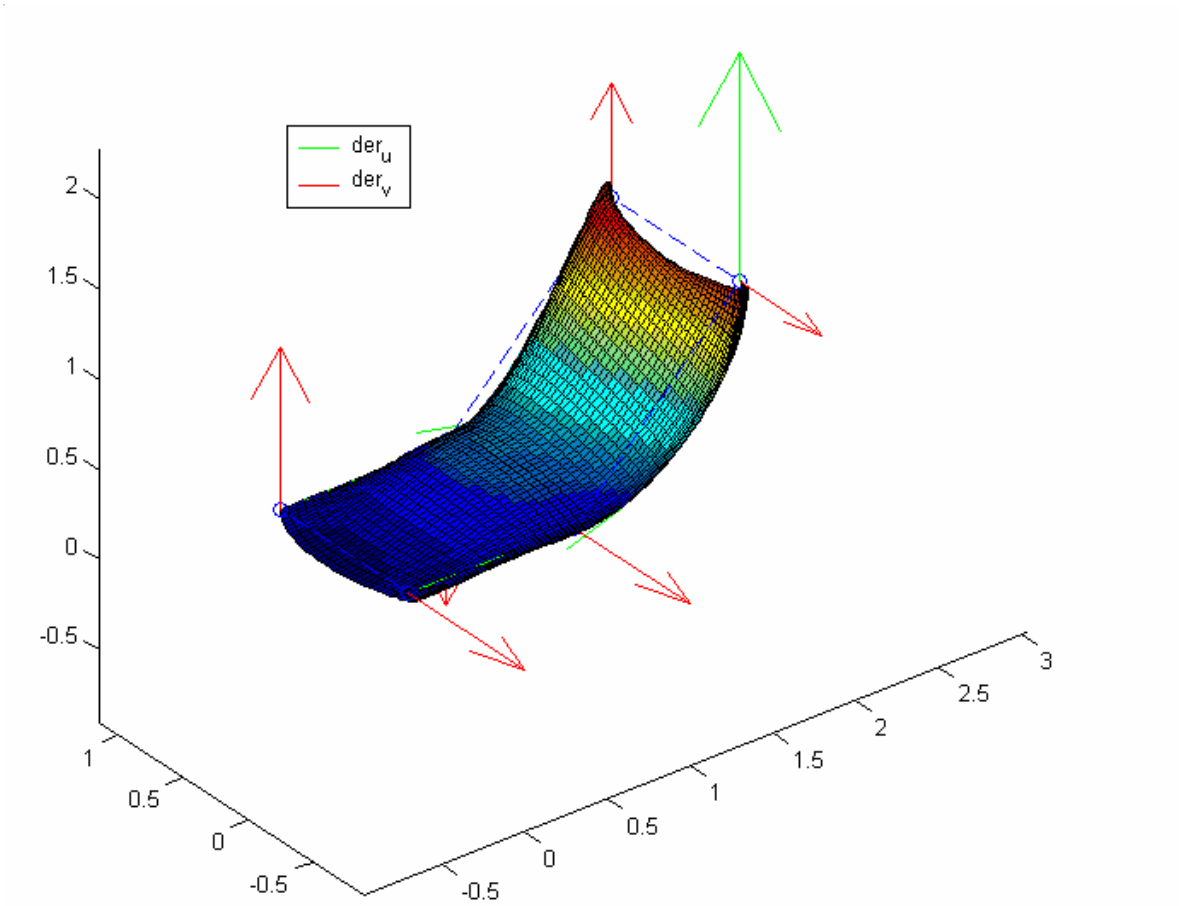


Figura 2-26 Interpolazione mediante B-Spline con derivate parziali assegnate lungo il bordo..

In modo analogo a quanto visto per le curve possiamo utilizzare questa funzione per unire due superfici diverse in modo che abbiano la stessa tangente all'interfaccia. Per fare questo, abbiamo calcolato le derivate parziali nelle due direzioni della superficie sinistra, e

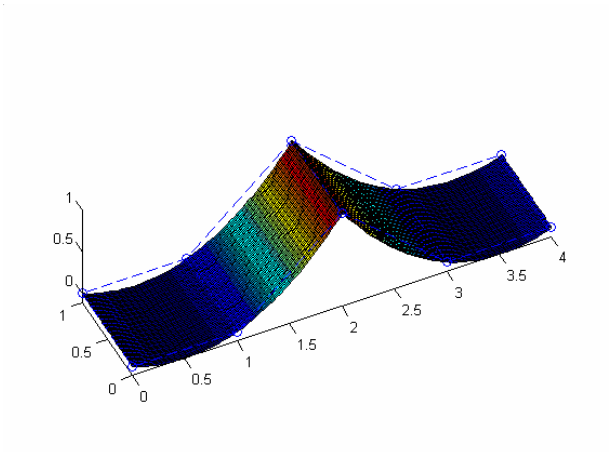


Figura 2-27 Smooth stitch: superfici originali, nessuna derivata specificata.

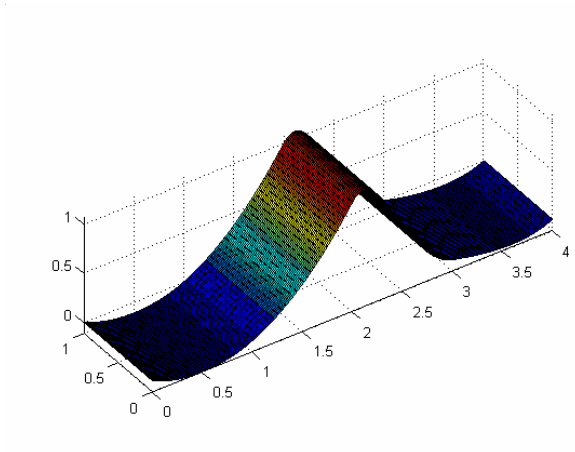


Figura 2-28 Smooth stitch: la superficie a destra è stata modificate per avere la stesa derivata della superficie a sinistra, rimasta originale.

imposte per la superficie destra. Inoltre, abbiamo anche imposto che le derivate finali della superficie sinistra restassero invariate. Il risultato è illustrato nelle figure successive.

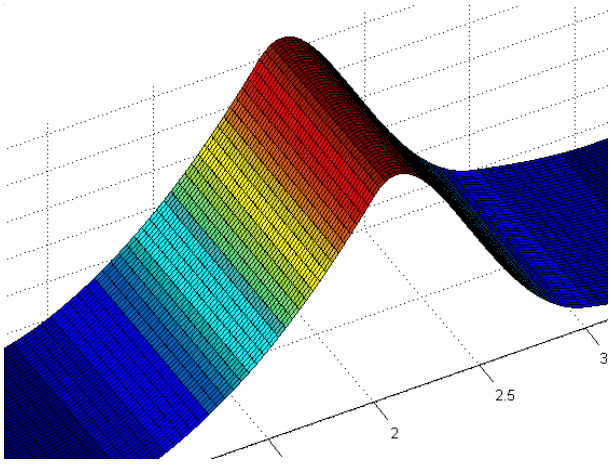


Figura 2-29 Smooth stitch: dettaglio.

2.5.5. Interpolazione con superfici lineari-cubiche

Il lavoro svolto finora si è incentrato sulla creazione di superfici interpolanti di grado 3 in entrambe le direzioni. In alcuni casi si è reso necessario sviluppare una procedura di interpolazione che fosse in grado di interpolare una matrice di punti con polinomi di grado differente nelle due direzioni. Il caso più evidente è la creazione della superficie alare. Utilizzando un polinomio cubico sia in direzione della corda sia in direzione dell'apertura, si ottiene nel caso di tre profili con rastremazione una forma in pianta come quella mostrata in Figura 2-30.

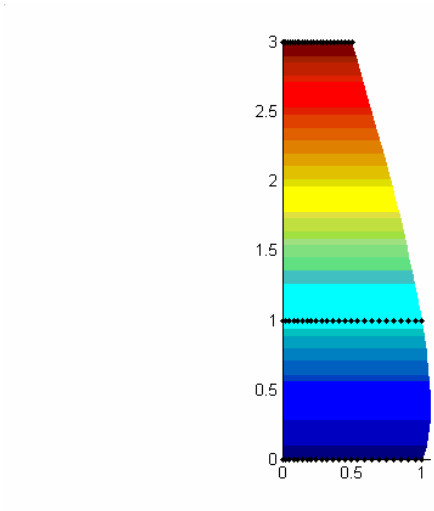


Figura 2-30 Interpolazione bicubica, forma in pianta ottenuta.

Nel caso in cui il profilo all'estremità sia posizionato in modo da dare un diedro all'ala, il risultato dell'interpolazione bicubica è anche peggiore come mostrato nella Figura 2-31. per ovviare a questo problema è stata scritta una funzione di interpolazione apposita, la quale in ingresso, oltre ai punti da interpolare, richiede di specificare in quela delle due

direzioni l'interpolazione verrà effettuata mediante polinomi di grado 1, ovvero in quale direzione interpolare i punti linearmente. Per fare questo si è modificato la funzione di interpolazione locale sopprimendo i punti di controllo intermedi calcolati, mantenendo quindi solo quelli originali nella direzione in cui si vuole l'interpolazione lineare, e riscrivendo il vettore dei nodi in quella direzione come:

$$U = \left\{0,0,\frac{1}{N},\frac{1}{N},\frac{2}{N},\frac{2}{N},...,\frac{N-1}{N},\frac{N-1}{N},1,1\right\}$$ dove N è il numero di punti da interpolare nella direzione scelta.

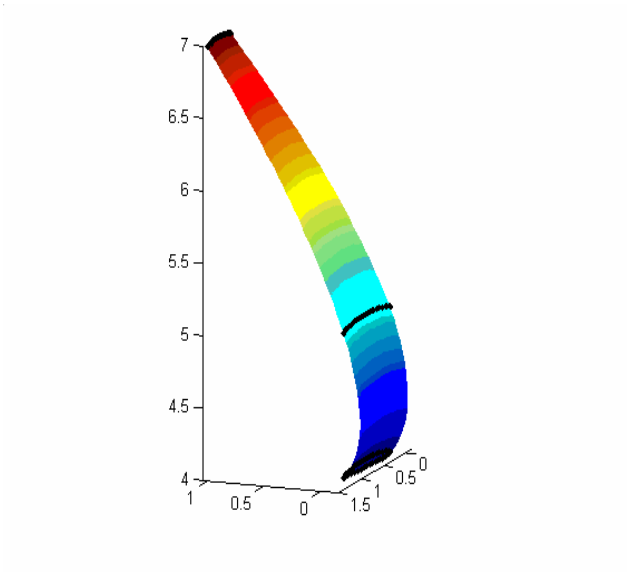


Figura 2-31 Interpolazione bicubica: ala con diedro.

Il risultato di questo algoritmo è mostrato nelle figure seguenti, dove viene ovviamente scelta come direzione di interpolazione lineare la direzione dell'apertura.

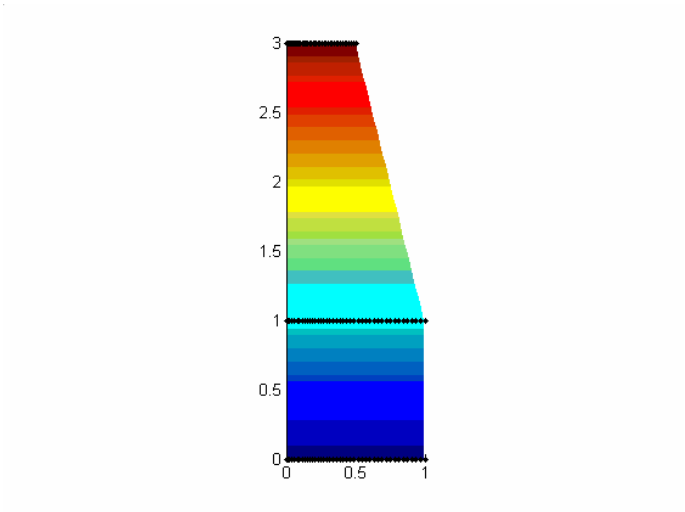


Figura 2-32 Interpolazione lineare-cubica: forma in pianta

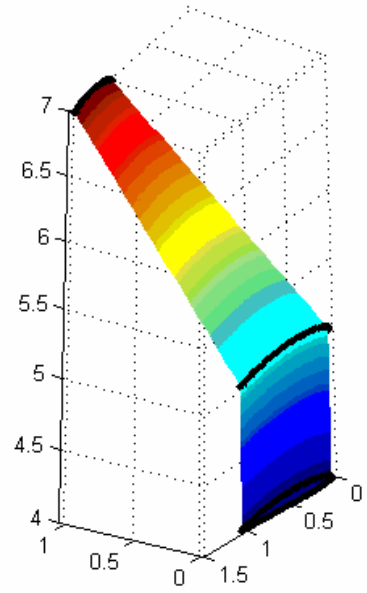


Figura 2-33 Interpolazione lineare-cubica: ala con diedro.

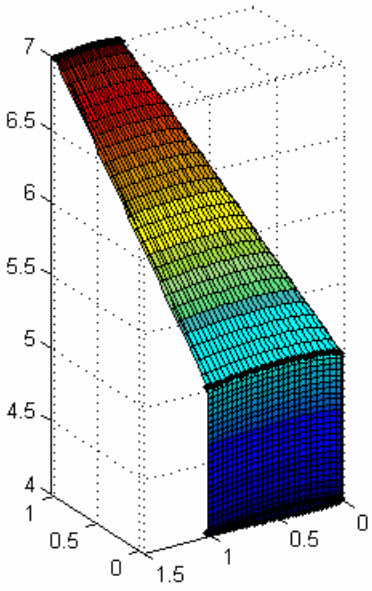


Figura 2-34 Interpolazione bicubica: ala con diedro freccia.

Capitolo 3

3. Intersezione di superfici

3.1. Premessa

Il problema dell'intersezione di superfici parametriche (tra cui le più importanti sono appunto le NURBS) è un problema comune in ogni applicazione di computer grafica e progettazione assistita da calcolatore (CAD). Questo problema, che è un'area di ricerca attiva da più di 30 anni, è noto con l'acronimo di SSI (Surface to Surface Intersection) ed è stato elencato come uno dei problemi fondamentali nella integrazione di modellazione geometrica e solida [Hoffmannn 1989; Requicha e Rossignac 1992]. Le caratteristiche fondamentali di un algoritmo di SSI sono la stabilità numerica, la possibilità di lavorare con superfici generiche, la capacità di funzionare senza l'intervento dell'operatore, di produrre curve di intersezione accurate, di riconoscere punti singolari (punti in cui l'intersezione si auto-interseca), intersezioni aperte e chiuse (loop) anche di piccole dimensioni.

Durante questi anni si sono delineati 4 principali metodi per ricavare l'intersezione; questi sono detti:

- Subdivision methods
- Lattice evaluation
- Analytic methods
- Marching methods (tracing methods)

Prima di proseguire, vediamo in cosa consistono i metodi precedentemente menzionati:

Subdivision methods: L'idea di base di questi metodi è di decomporre il dominio di ricerca in domini più piccoli ricorsivamente, riconducendosi a risolvere un problema più semplice. La decomposizione continua fino a che il livello di semplicità desiderato è raggiunto, per poi fermarsi e risolvere il problema direttamene. L'ultimo passo è di unire tutte le intersezioni trovate per ottenere la soluzione finale. Questi algoritmi convergono tutti al limite, ma per ottenere risultati accurati servono tolleranze basse e conseguenti tempi di calcolo molto lunghi. Nel caso in cui la suddivisione sia arrestata troppo presto, si potrebbero perdere piccole intersezioni.

Lattice evaluation: Questi metodi decompongono il problema in un problema di complessità minore, come il calcolo di intersezioni curva-superficie. L'ultimo passo è quello di collegare tutti i punti trovati in un'unica curva. Il problema maggiore è quello della determinazione del passo che garantisca una soluzione robusta. Inoltre, questi metodi, spesso non sono in grado di rilevare piccole intersezioni.

Analytic Methods: I metodi analitici sono basati sulla rappresentazione esplicita delle curve di intersezione, ma possono essere applicate per curve di grado non elevato. Molti metodi sono stati sviluppati sfruttando le proprietà geometriche delle superfici, ma l'estensione di questi metodi a curve di grado più elevato non è di solito semplice.

Marching methods: Questi sono i metodi più utilizzati, principalmente perché facili da implementare e abbastanza generici. La principale caratteristica di questi metodi è la generalità, che consente di applicarli a casi di superfici definite in modi diversi. L'idea alla base di questi metodi è quella di rappresentare la curva di intersezione analiticamente, determinare un punto di partenza per ogni curva e, utilizzando le caratteristiche locali delle superfici, tracciare l'intersezione. La curva di intersezione si può definire algebricamente come il luogo dei punti con distanza zero da entrambe le superfici. L'intersezione di due superfici consiste di curve chiuse e curve aperte. Per determinare il punto di partenza delle curve aperte è sufficiente intersecare i bordi di una superficie con l'altra, mentre il problema della determinazione dei punti di partenza delle curve chiuse è più complesso.

Purtroppo, nessuno dei precedenti metodi è in grado di soddisfare contemporaneamente i tre requisiti di accuratezza, robustezza e efficienza [Hoffmannn 1989; Pratt 1986; Requicha e Rossignac 1992]; allora è necessario fare ricorso a metodi misti, detti "ibridi", i quali fanno uso contemporaneamente di 2 o più dei metodi sopraccitati.

Per questi motivi, sembra che molti software CAD commerciali utilizzino ancora metodi di decomposizione poliedrica per suddividere il dominio per poi applicare metodi analitici o più frequentemente metodi di *tracing* per calcolare le intersezioni; questi, però, sono ancora poco precisi e danno luogo a proliferazione degli errori, ma sono robusti.

3.2. Point inversion

Il primo algoritmo presentato risolve un problema collaterale che può risultare utile nella risoluzione del problema dell'intersezione. Il problema può essere enunciato come segue:

"Dato un punto su una superficie parametrica, determinare il valore dei parametri corrispondenti"

Questo problema, noto come *Point Inversion*, può essere risolto risolvendo un problema più generale:

"Dato un punto qualsiasi e una superficie parametrica, determinare il valore dei parametri corrispondenti alla sua proiezione"

Ovviamente, se il punto giace sulla superficie, questo problema è lo stesso di prima, mentre se il punto non giace esattamente sulla superficie, questo algoritmo trova la sua proiezione, consentendo così di tollerare eventuali errori numerici.

3.2.1. Point inversion : curva

L'algoritmo presentato utilizza il metodo di Newton per minimizzare la distanza tra il punto P e la curva. Il metodo converge rapidamente, ma necessita di un buon valore iniziale; per ottenerlo si è deciso di valutare preventivamente la distanza di P da una griglia di n punti valutati ad intervalli regolari sulla curva. Il punto sulla curva che ha la minima distanza da P è il candidato valore iniziale delle iterazioni.

Assumendo adesso un valore iniziale u_0, scriviamo il prodotto scalare:

$$f(u) = C'(u) \cdot (C(u) - P) \qquad \textbf{(3-1)}$$

La distanza di P dalla curva è minima quando $f(u) = 0$, sia che P sia sulla curva $(C(u) - P) = 0$, sia che P sia perpendicolare alla curva, $(C(u) - P)\, C'(u) = 0$. L'iterazione di Newton standard ha la forma:

$$u_{i+1} = u_i - \frac{f(u_i)}{f'(u_i)} = u_i - \frac{C'(u_i) \cdot (C(u_i) - P)}{C''(u_i) \cdot (C(u_i) - P) + |C'(u_i)|^2} \qquad \textbf{(3-2)}$$

Due tolleranze possono essere specificate, una per la zero-distanza euclideo (ε_1), una per il zero-coseno (ε_2).

I criteri di convergenza sono allora dati da:

- punto stazionario: $|(u_{i+1} - u_i) C'(u_i)| \le \varepsilon_1$

- coincidenza: $(C(u_i) - P) \le \varepsilon_1$

- perpendicolarità: $\frac{|C'(u_i)\cdot(C(u_i)-P)|}{|C'(u_i)|\cdot|(C(u_i)-P)|} \le \varepsilon_2$

inoltre, a ogni passo è necessario controllare che il parametro non sia uscito dal range ($u_{i+1} \in [a,b]$):

- se la curva è aperta:
  - se ($u_{i+1} < a$) $u_{i+1} = a$
  - se ($u_{i+1} > b$) $u_{i+1} = b$
- se la curva è chiusa:
  - se ($u_{i+1} < a$) $u_{i+1} = b - (a - u_{i+1})$
  - se ($u_{i+1} > b$) $u_{i+1} = a + (u_{i+1} - b)$

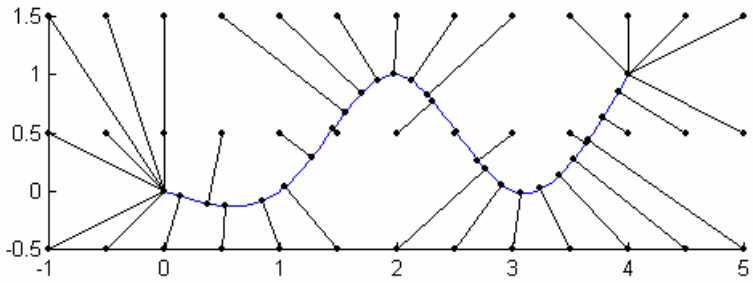


Figura 3-1 Algoritmo di Point inversion, risultati.

3.2.2. Point inversion : superficie

Analogamente al paragrafo precedente, possiamo definire un algoritmo che calcola la proiezione di un punto su una superficie.

Assumendo adesso un valore iniziale (u_0, v_0), scriviamo il prodotto scalare:

$$r(u,v) = (S(u,v) - P)$$
$$f(u,v) = r(u,v)\cdot S_u(u,v) = 0 \qquad \textbf{(3-3)}$$
$$g(u,v) = r(u,v)\cdot S_v(u,v) = 0$$

In questo caso, l'iterazione di Newton necessita della risoluzione di un sistema 2 x 2:

$$\delta_i = \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} u_{i+1} - u_i \\ v_{i+1} - v_i \end{bmatrix}$$

$$J_i = \begin{bmatrix} f_u & f_v \\ g_u & g_v \end{bmatrix} = \begin{bmatrix} |S_u|^2 + r \cdot S_{uu} & S_u \cdot S_v + r \cdot S_{uv} \\ S_u \cdot S_v + r \cdot S_{vu} & |S_v|^2 + r \cdot S_{vv} \end{bmatrix}$$

$$\kappa_i = -\begin{bmatrix} f(u_i, v_i) \\ g(u_i, v_i) \end{bmatrix} \qquad \textbf{(3-4)}$$

$$J_i \delta_i = \kappa_i$$

$$u_{i+1} = u_i + \Delta u$$

$$v_{i+1} = v_i + \Delta v$$

I criteri di convergenza sono allora dati da:

- punto stazionario: $|(u_{i+1} - u_i) S_u(u_i, v_i) + (v_{i+1} - v_i) S_v(u_i, v_i)| \le \varepsilon_1$

- coincidenza: $(S(u_i, v_i) - P) \le \varepsilon_1$

- perpendicolarità: $\dfrac{|S_u(u_i, v_i) \cdot (S(u_i, v_i) - P)|}{|S_u(u_i, v_i)| \cdot |(S(u_i, v_i) - P)|} \le \varepsilon_2$ e $\dfrac{|S_v(u_i, v_i) \cdot (S(u_i, v_i) - P)|}{|S_v(u_i, v_i)| \cdot |(S(u_i, v_i) - P)|} \le \varepsilon_2$

inoltre a ogni passo è necessario controllare che i parametri non siano usciti dal range ($u_{i+1} \in [a,b]$, $v_{i+1} \in [c,d]$):

- se la curva è aperta in direzione *u*:
  - se ($u_{i+1} < a$) $u_{i+1} = a$
  - se ($u_{i+1} > b$) $u_{i+1} = b$
- se la curva è aperta in direzione *v*:
  - se ($v_{i+1} < c$) $v_{i+1} = c$
  - se ($v_{i+1} > d$) $v_{i+1} = d$
- se la curva è chiusa in direzione *u*:
  - se ($u_{i+1} < a$) $u_{i+1} = b - (a - u_{i+1})$
  - se ($u_{i+1} > b$) $u_{i+1} = a + (u_{i+1} - b)$
- se la curva è chiusa in direzione *v*:
  - se ($v_{i+1} < c$) $v_{i+1} = d - (c - v_{i+1})$
  - se ($v_{i+1} > d$) $v_{i+1} = c + (v_{i+1} - d)$

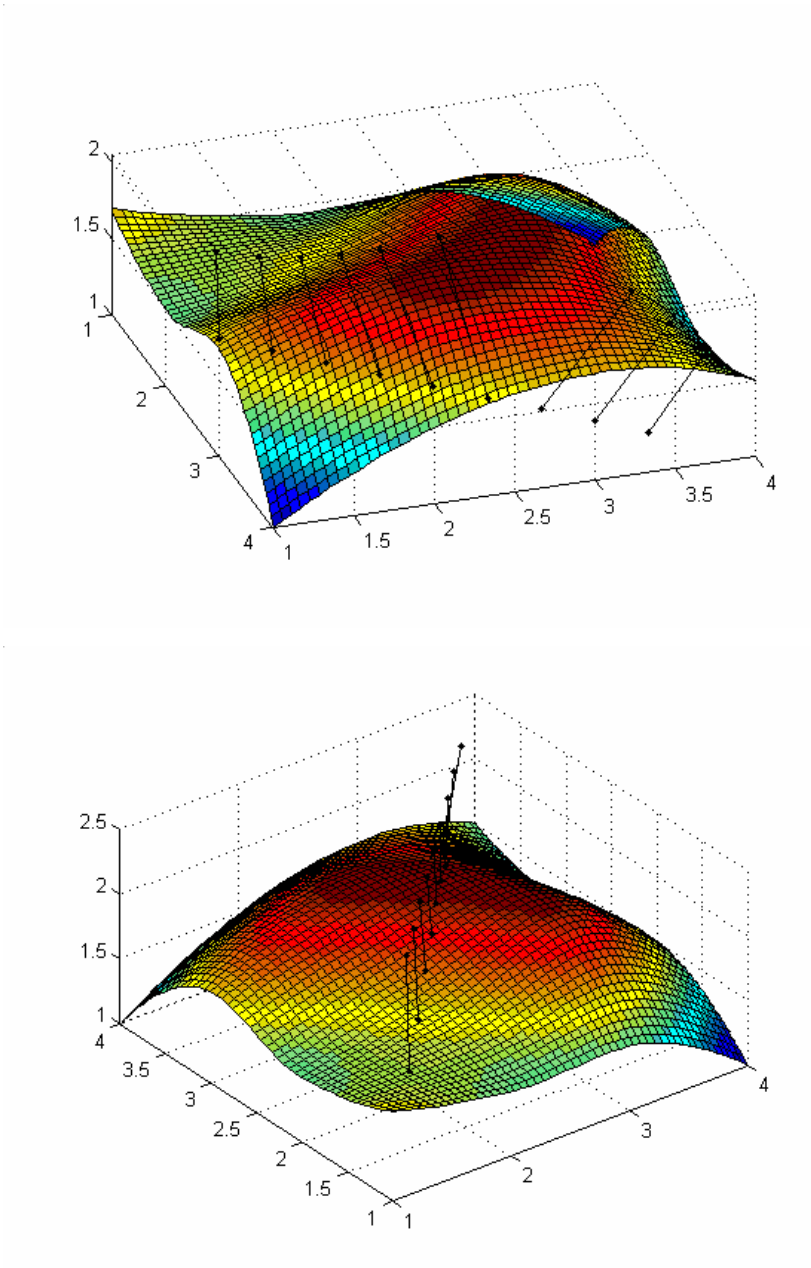
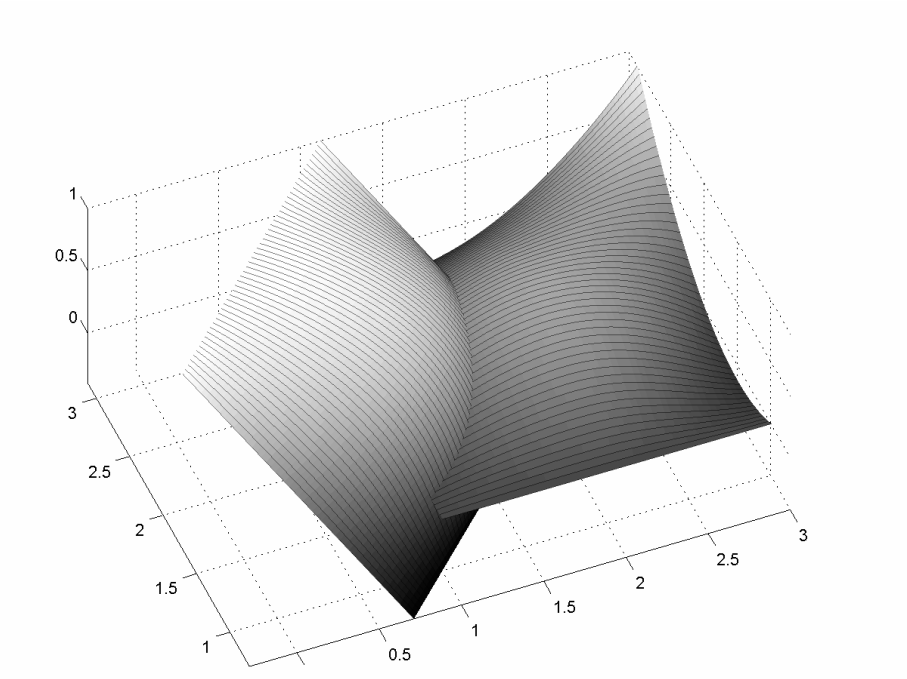


Figura 3-2 Algoritmo di Point inversion, risultati.

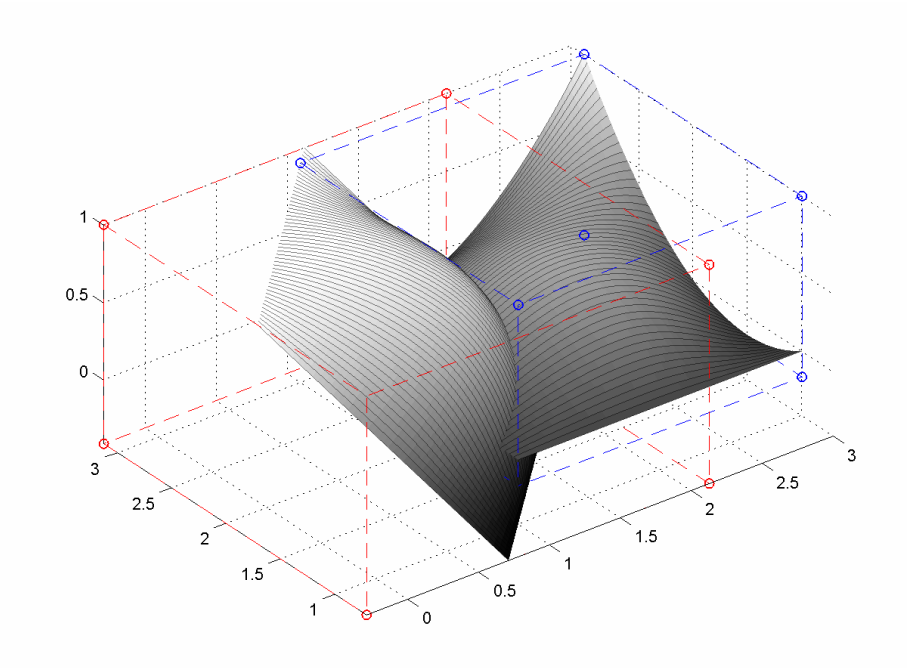
3.3. Intersezione di superfici : subdivision method

In questo paragrafo presentiamo l'algoritmo di intersezione basato sul criterio "divide and conquer" della categoria dei metodi di suddivisione. Questo algoritmo, sebbene non sia di tipo ottimale per i motivi visti nella premessa, viene implementato perché servirà poi da base per i metodi più veloci di tipo "marching methods". L'idea è quella di costruire un algoritmo efficiente per decomporre il dominio delle due superfici da calcolare in tanti sottodomini minori, tali da rendere possibile ed efficiente l'uso di metodi diretti di tipo analitico. L'algoritmo avrà anche il compito di selezionare in maniera efficiente le coppie di sottodomini che sicuramente contengono punti di intersezione tra le superfici. Il presente algoritmo, una volta ottenuta una suddivisione soddisfacente, calcolerà i punti di intersezione

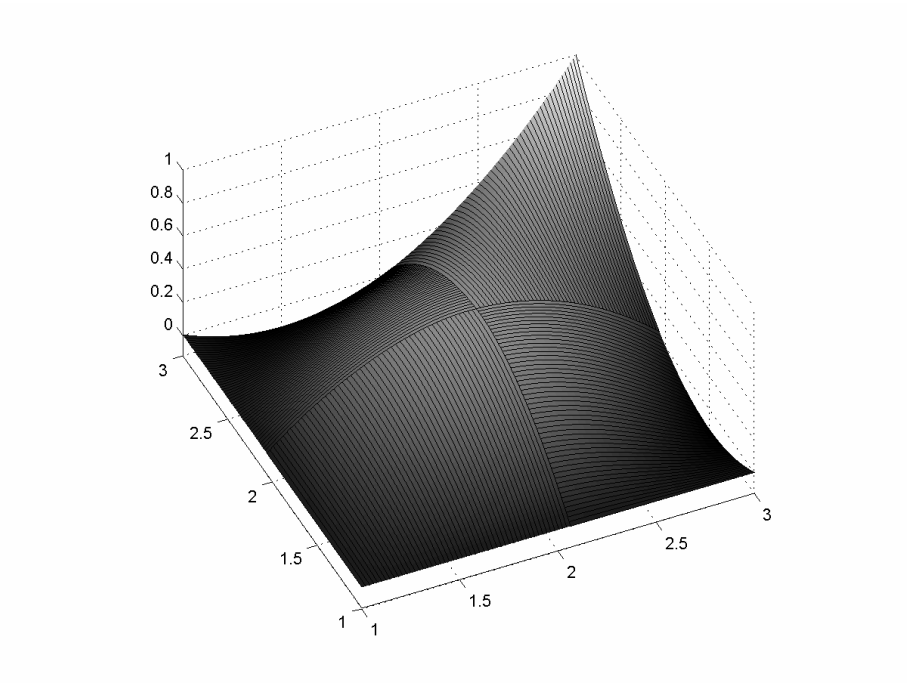
approssimando le superfici con tasselli e calcolando l'intersezione fra piani. Questo stesso algoritmo verrà poi modificato per ottenere un intersecatore per calcolare l'intersezione curva-superficie. L'algoritmo sviluppato può essere sintetizzato nei seguenti passi:



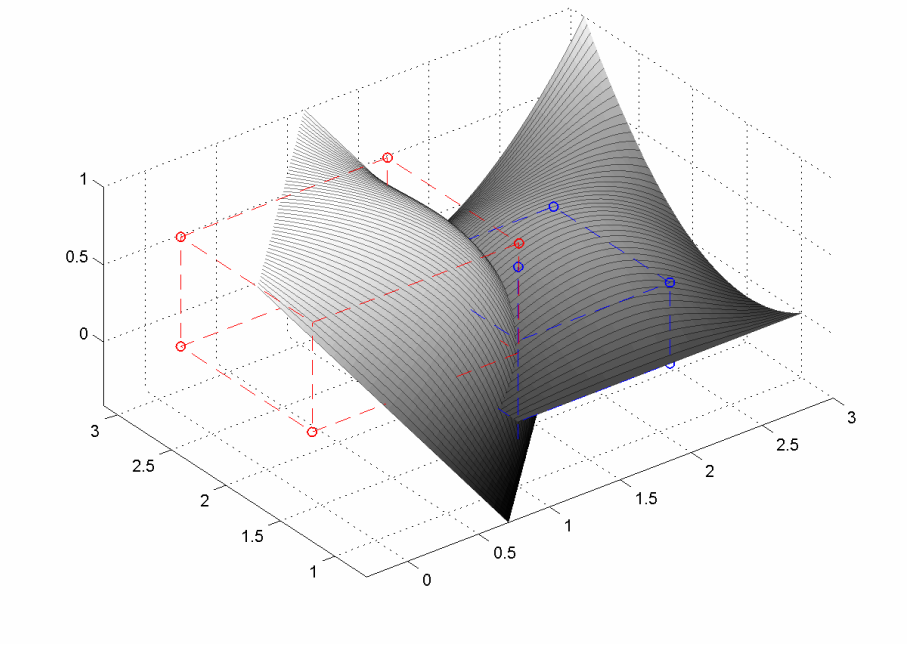
1. inizio algoritmo, in ingresso vengono fornite le due superfici da intersecare e opzionalmente una tolleranza entro cui deve essere calcolata la loro intersezione



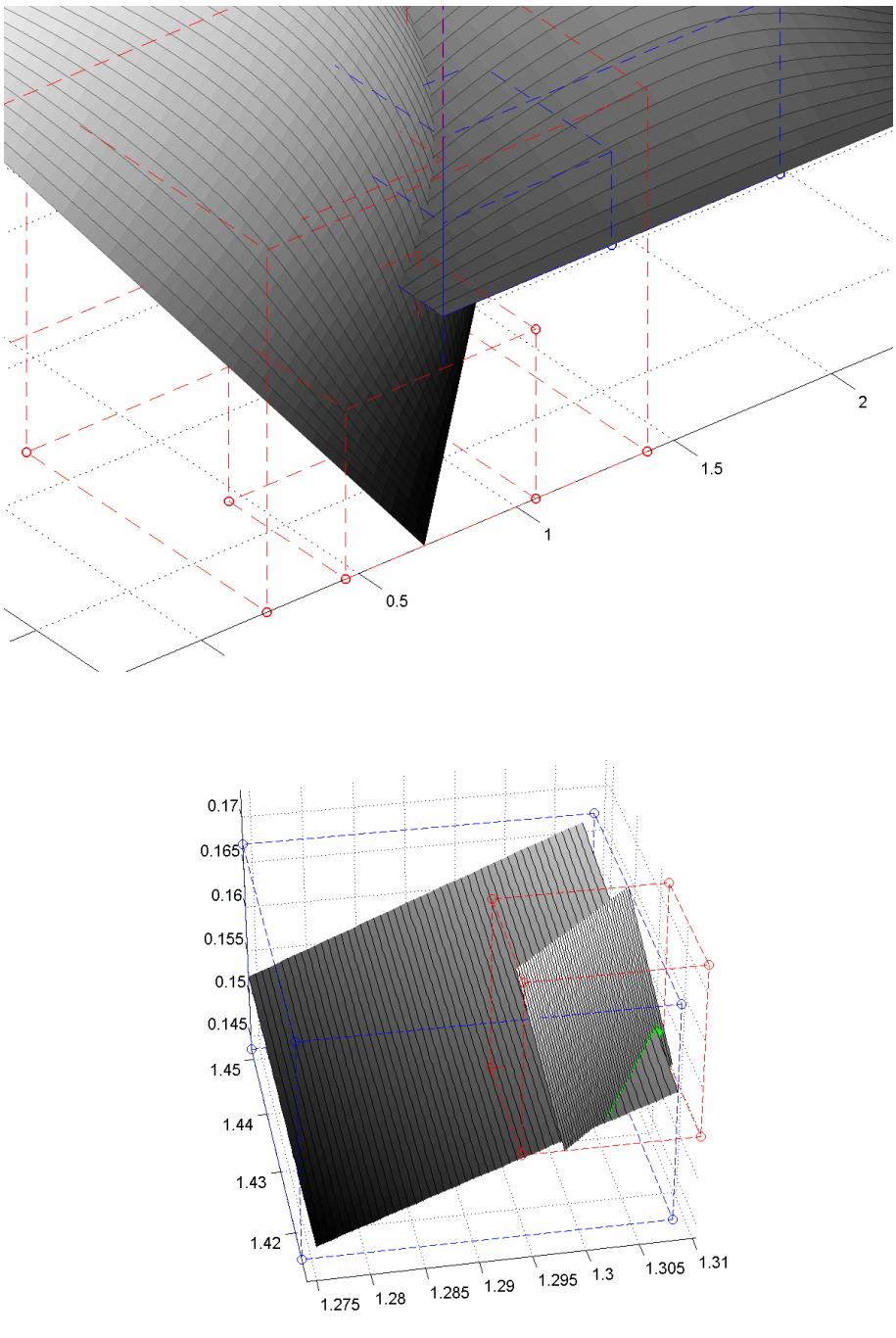
2. calcolo dei parallelepipedi che contengono le superfici.
3. test di intersezione dei parallelepipedi mediante proiezione sui tre assi coordinati



4. se i parallelepipedi si intersecano le superfici vengono suddivise in 4 e il procedimento ricomincia



5. il procedimento di suddivisione continua fino a che le porzioni di superficie ottenute non sono approssimabili con piani nella tolleranza specificata.



6. una volta ottenute due superfici quasi-piane, vengono intersecati i piani che le approssimano ottenendo così due punti dell'intersezione.

Una volta ottenuti i punti dell'intersezione è necessario ordinarli (perché in genere questo metodo produce punti in ordine casuale sparsi) e, se necessario, ottenere per ogni curva le intersezioni nello spazio dei parametri *u,*v mediante una procedure di *point inversion*. Durante la procedura di intersezione, l'algoritmo fa uso di alcune procedure specifiche per eseguire operazioni sulle superfici. Una di queste è la procedura che data una superficie NURBS la suddivide in 4 superfici distinte.

3.3.1. Suddivisione di curve

Si presenta l'algoritmo di suddivisione di curve, essendo questo direttamente estendibile alle superfici. Dalle proprietà delle NURBS, è noto che, in corrispondenza di un nodo di molteplicità esattamente uguale al grado della curva, la curva stessa interpola il punto di controllo corrispondente. Questa proprietà è utilizzata per suddividere la curva secondo la seguente procedura:

1- data la curva completa, e il nodo in corrispondenza del quale la si vuole dividere, si aggiunge quel nodo (paragrafo 2.3.1) fino a che la sua molteplicità è uguale al grado della superficie.

2- Le 2 curve si ottengono dividendo il nuovo vettore dei nodi e la corrispondente matrice dei punti di controllo in corrispondenza del nodo aggiunto.

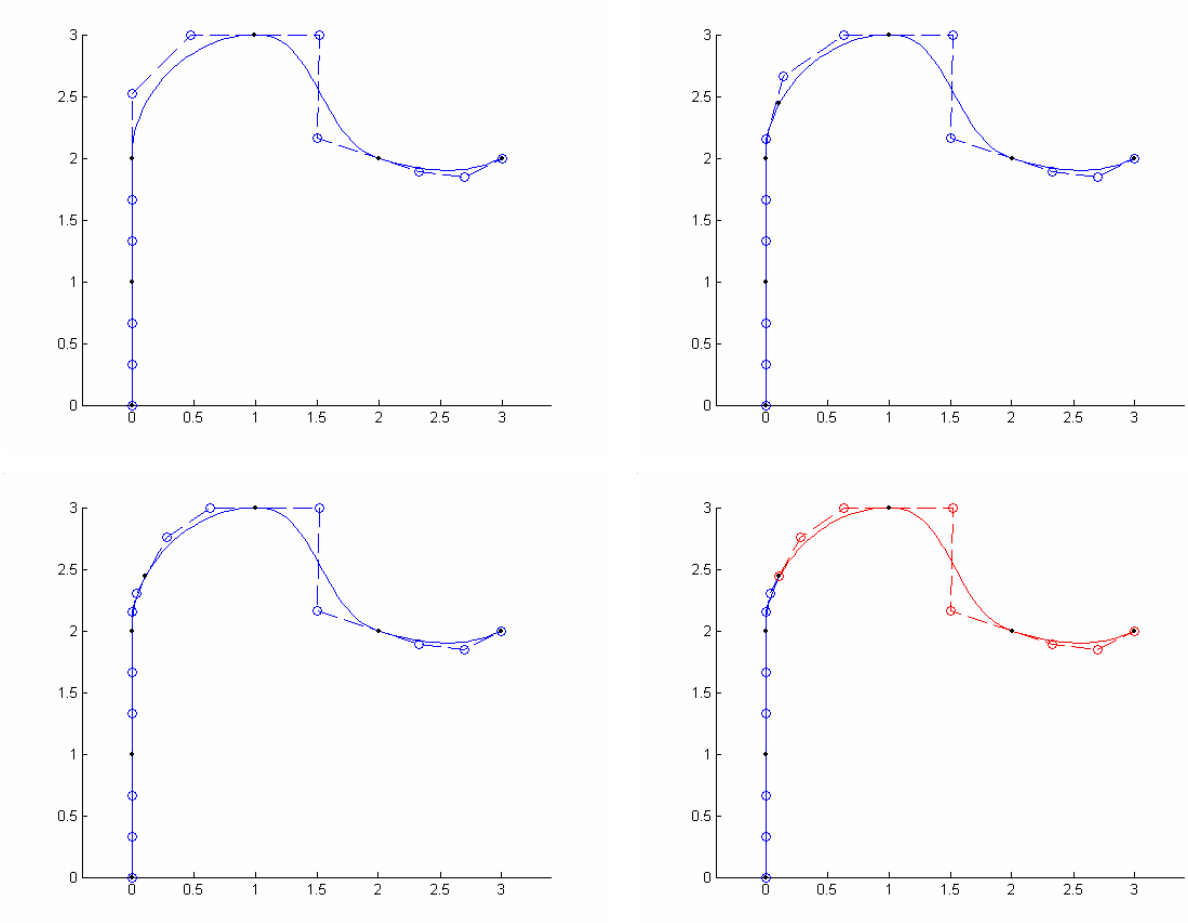


Figura 3-3 procedura di suddivisione di una curva

Siano P_i i punti di controllo della curva originale, e $U=\{a,\ldots,a,\ldots,b,\ldots,b\}$ il vettori dei nodi corrispondente; inserendo u_0 p-volte ottengo p nuovi punti di controllo. Il nuovo vettore

dei nodi delle due curve diventa allora $U_1=\{a,...,a,..., u_0, ..., u_0\}$ e $U_2=\{u_0,... u_0,...,b,...,b\}$, e i punti di controllo sono quelli ottenuti dall'inserzione dei nodi, suddivisi in corrispondenza del nodo inserito, che diventa quindi l'ultimo della prima curva e il primo della seconda curva.

3.3.2. Suddivisione di superfici

Per quanto riguarda la suddivisione delle superfici la tecnica è analoga, solo che vanno aggiunti nodi in tutti e due i vettori U e V e alla fine si ottengono 4 superfici come si vede in

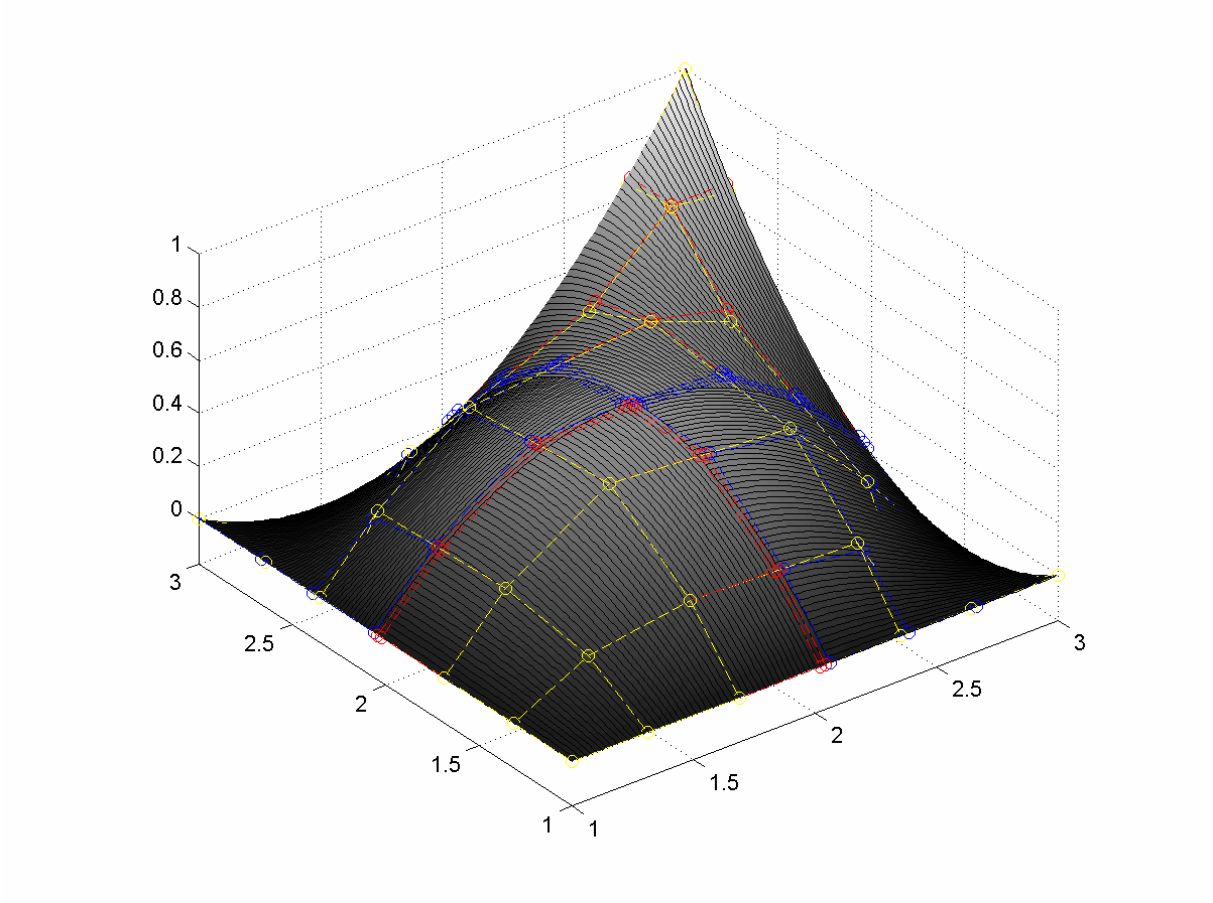


Figura 3-4 suddivisione di una superficie, punti di controllo originali (giallo) e punti di controllo delle superfici ottenute (blu e rosso)

3.3.3. Calcolo dei parallelepipedi contenenti le superfici

Dalla proprietà v del paragrafo 1.3.4 (guscio convesso forte), possiamo affermare che ogni curva o superficie è contenuta nel guscio convesso formato dai suoi punti di controllo. L'unica eccezione a queste proprietà è quella di una curva NURBS definita con pesi negativi. Per questa ragione, come visto in precedenza, ci asterremo dall'usare curve con pesi nulli o negativi, essendo possibile evitarlo mediante l'introduzione di più punti di controllo.

3.3.4. Test di planarità della superficie

Facendo sempre riferimento alla proprietà del guscio convesso possiamo affermare che se tutti i punti di controllo di una curva o superficie giacciono fra due piani paralleli distanti meno di una prestabilita tolleranza, allora anche la curva o superficie giace al loro interno. Mediante questa osservazione possiamo testare la planarità di una superficie senza valutarne il valore in alcun punto (procedimento computazionalmente dispendioso) ma limitandoci a fare dei controlli sui punti di controllo.

3.3.5. Calcolo dell'intersezione di due piani

L'intersezione fra due piani viene calcolata calcolando l'intersezione linea-piano fra i lati di un piano e l'altro piano. Se l'intersezione calcolata giace all'interno del piano allora viene considerata una intersezione valida e scritta nel vettore di uscita. Questo algoritmo, nella sua semplicità, è di fondamentale importanza per la velocità e la precisione dell'intero procedimento, in quanto è questo l'algoritmo che calcola effettivamente i punti dell'intersezione, e viene eseguito migliaia di volte per ogni intersezione. Questo intersecatore piano-piano può essere sostituiti da un intersecatore triangolo-triangolo, anche se al momento non abbiamo trovato nessun intersecatore che risulti all'atto pratico considerevolmente più veloce.

3.3.6. Ordinamento dei punti dell'intersezione

In linea di principio abbiamo 3 spazi entro cui ordinare i punti: lo spazio tridimensionale dell'intersezione reale e i due spazi bidimensionali delle tracce dell'intersezione sul piano dei parametri u e v. La scelta dello spazio usato per riordinare sequenzialmente i punti trovati è ricaduta sullo spazio tridimensionale, in quanto è quello a cui appartengono i punti calcolati dal nostro intersecatore. L'algoritmo è piuttosto semplice e non molto robusto; molto margine di miglioramento non è ancora stato sfruttato, e alcune idee per futuri sviluppi verranno suggerite a fine paragrafo. L'ordinamento avviene supponendo che, dato un qualsiasi punto, il suo successivo nell'ordinamento sia quello che si trova alla minima distanza euclidea. Questa supposizione, anche se ragionevole per molte intersezioni, non è sempre vera, e pertanto è da usarsi con cautela nei casi più delicati. L'algoritmo costruisce il vettore dei punti ordinati partendo dal primo punto nel vettore iniziale, e togliendo da questo di volta in volta il punto più vicino all'ultimo punto scritto nel vettore ordinato. In questo modo, ad ogni ciclo, il vettore di ingresso diminuisce di una unità e il vettore ordinato aumenta fino a che ogni punto nel vettore in ingresso è stato ordinato. Una

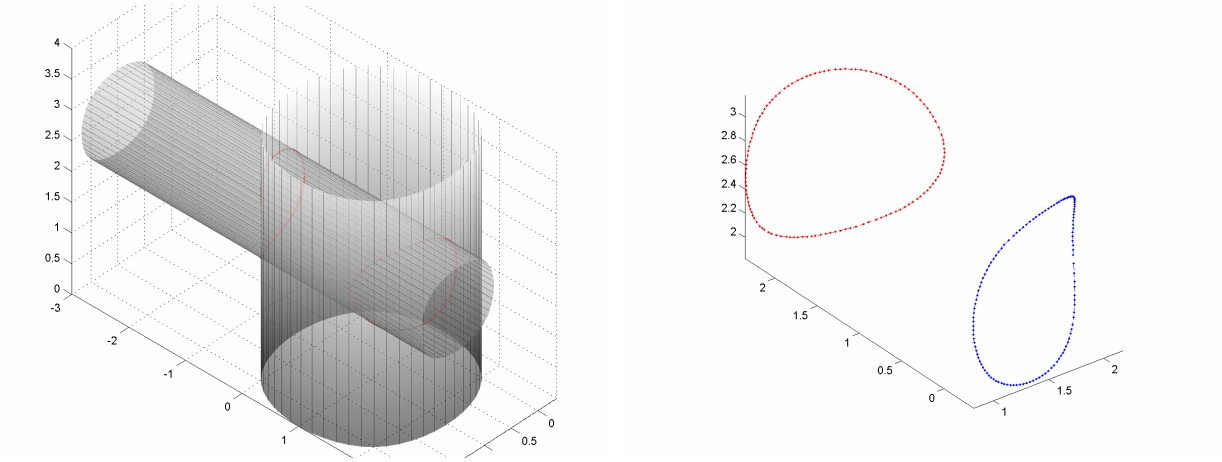
volta finita questa procedura è necessario scansionare il vettore ottenuto in cerca di eventuali branche di intersezione diverse. Infatti, in molti casi, l'intersezione di due superfici da origine a due o più curve di intersezione separate, che è necessario distinguere per poter fornire all'interpolatore descritto nel paragrafo 2.5.1, l'esatta sequenza di punti da interpolare. Per fare questo abbiamo supposto che la distanza media dei punti successivi appartenenti ad una curva sia molto minore della distanza fra due punti successivi appartenenti a due curve di intersezione diverse. Questa ipotesi è stata implementata suddividendo il vettore ordinato in diverse branche ogni volta che la distanza fra due punti successivi risultasse maggiore di 10 volte la distanza media. Questa tecnica, sebbene elementare, si è rivelata molto accurata e abbastanza soddisfacente per i nostri scopi. In definitiva, l'algoritmo sviluppato, nonostante molto rozzo e poco robusto si è rivelato sufficientemente accurato per i nostri scopi, avendo noi a che fare con intersezioni abbastanza semplici con una o al più due branche.

Per poter discriminare meglio due punti successivi è possibile utilizzare le informazioni dei due spazi che non sono stati presi in considerazione finora: gli spazi bidimensionali dei parametri. Infatti, da una analisi incrociata della distanza di due punti successivi nei tre spazi è possibile discriminare quale sia l'effettivo punto che vada accodato nel vettore ordinato.

3.3.7. Esempi di intersezioni

In questo paragrafo riportiamo alcune intersezioni calcolate con il metodo esposto in questo capitolo:

Cilindro-Cilindro



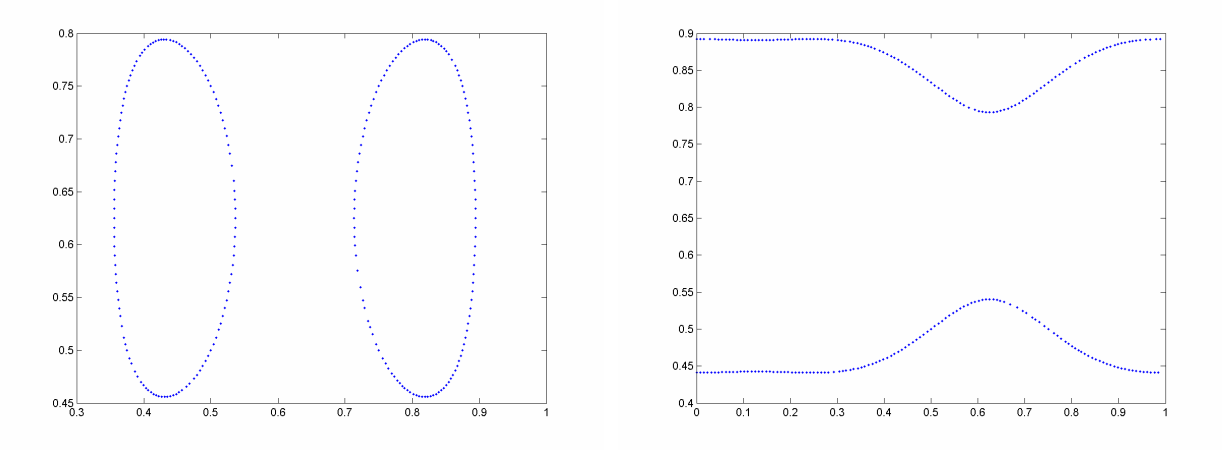


Figura 3-5 Intersezione cilindro-cilindro: risultati nello spazio e nel piano dei parametri.

Toro-Toro

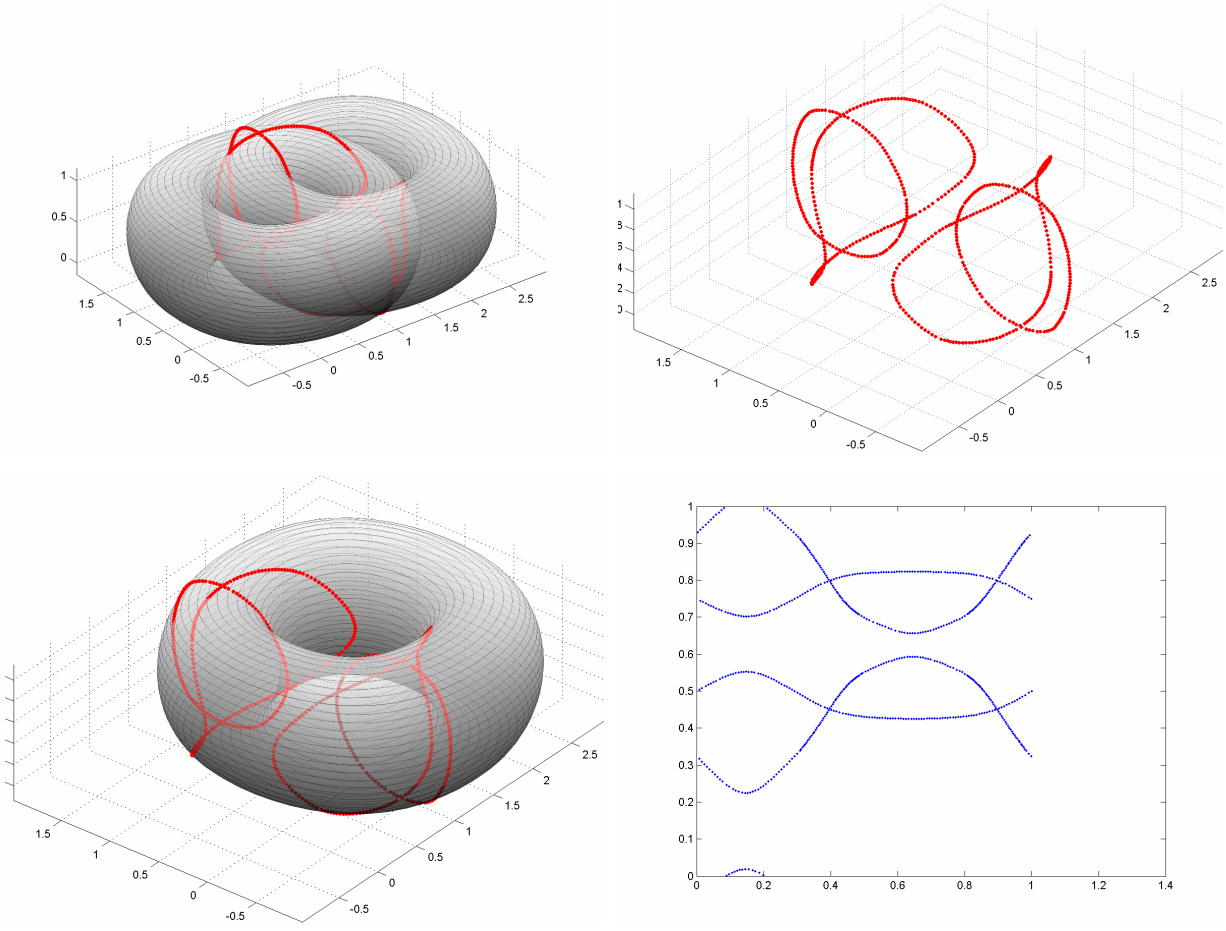


Figura 3-6 Intersezione toro-toro: risultati nello spazio e nel piano dei parametri.

Cilindro-Toro

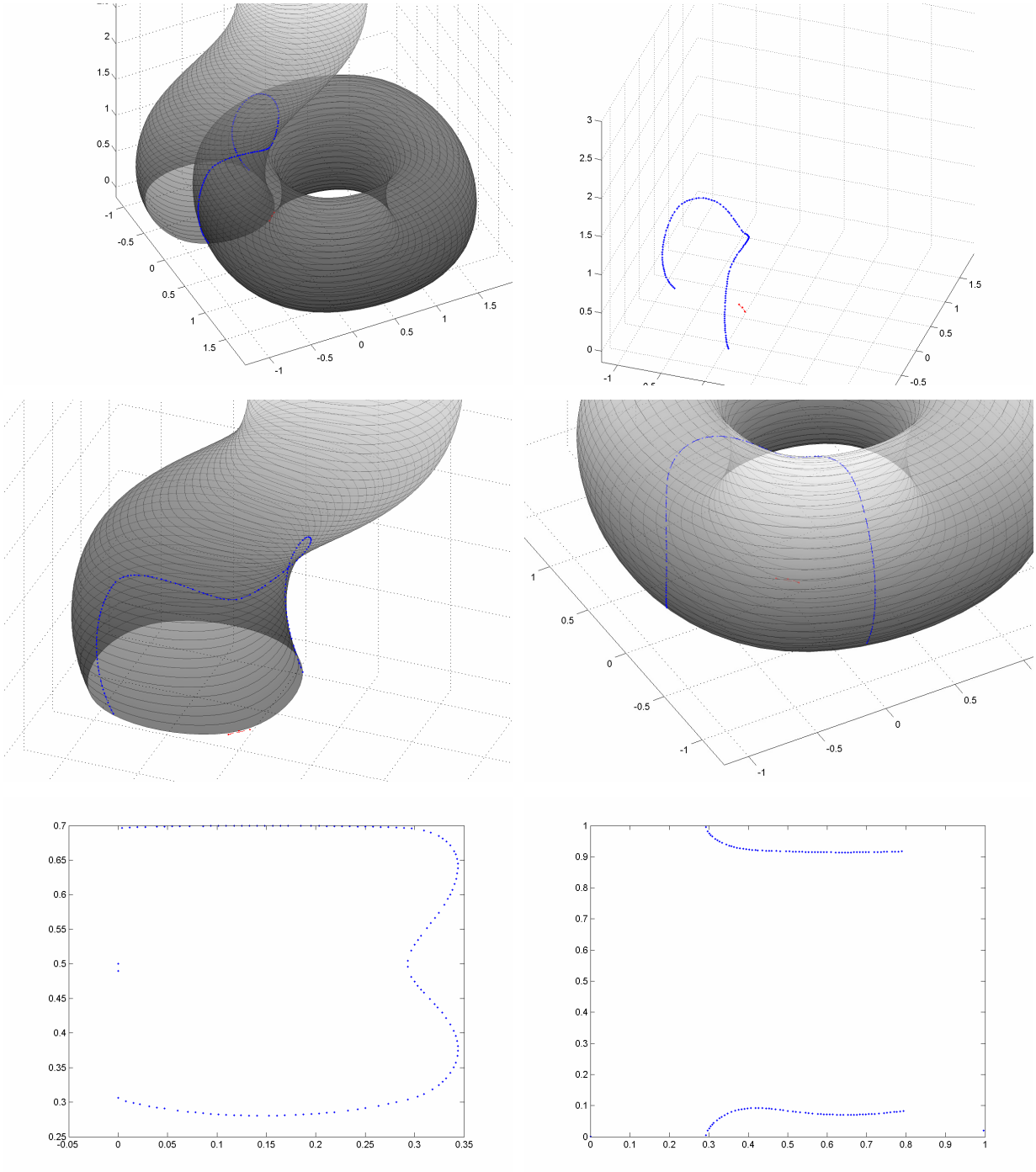


Figura 3-7 Intersezione cilindro -toro: risultati nello spazio e nel piano dei parametri.

Ala - Fusoliera

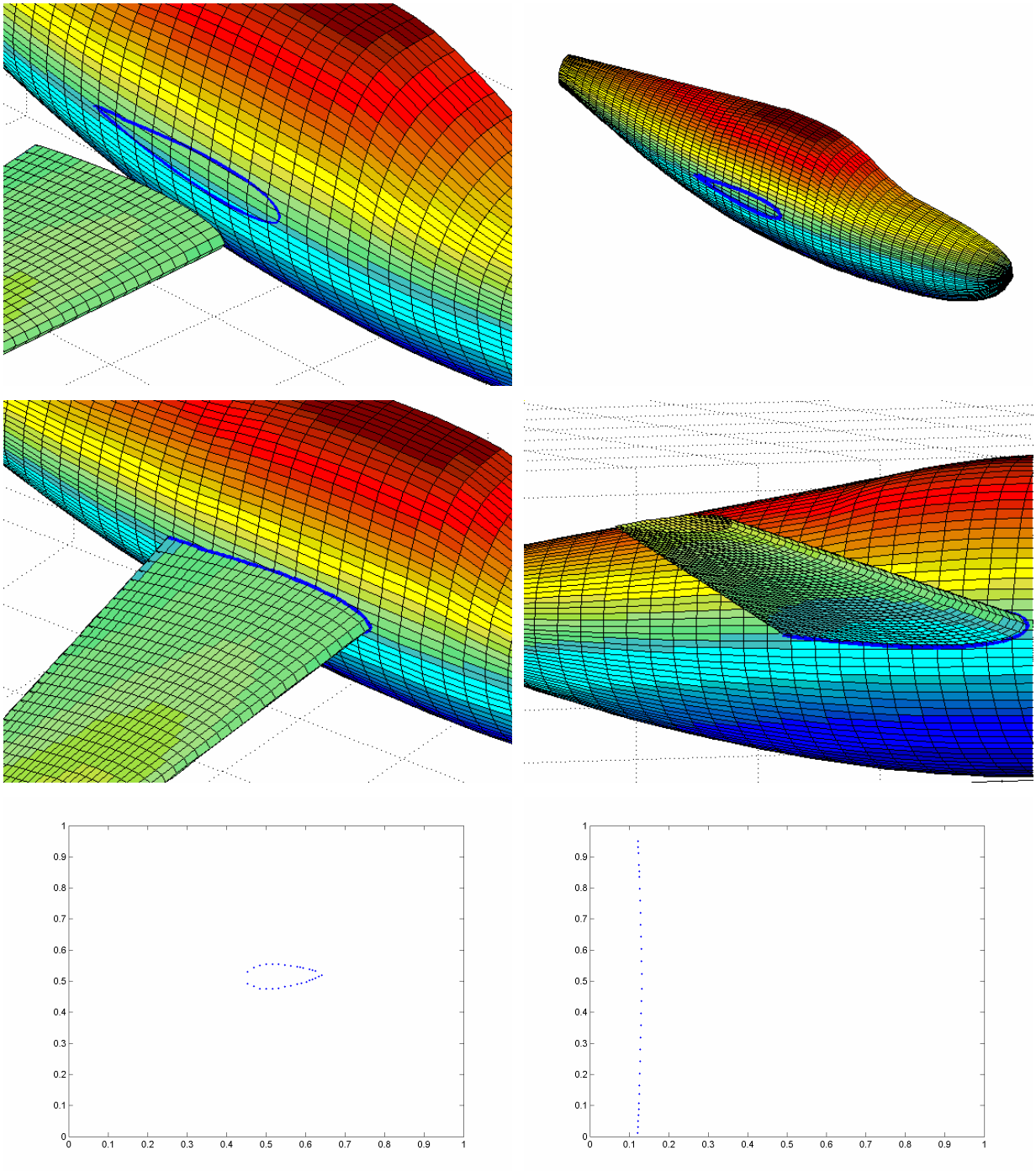


Figura 3-8 Intersezione Ala - Fusoliera: risultati nello spazio.

3.4. Intersezione di superfici : lattice evaluation

In questo paragrafo prenderemo in considerazione un algoritmo di intersezione di tipo *lattice evaluation*, a volte utile per il calcolo rapido di intersezioni poco complesse, come l'intersezione ala-fusoliera. La base dell'algoritmo è un intersecatore linea-superficie

direttamente derivato dall'algoritmo di intersezione esposto al paragrafo precedente. Questo algoritmo viene utilizzato per intersecare due superfici di cui una è identificata da due sezioni interpolate linearmente (baia di un ala fra due profili) (vedi Figura 3-9). L'algoritmo per prima cosa valuta una delle due superfici in un numero di punti predefinito, dalla cui scelta dipende l'accuratezza della curva trovata, (nel caso in figura, la superficie suddivisa in linee è l'ala, schematizzata mediante segmenti di retta fra punti omologhi dei due profili) poi richiama per ogni segmento l'intersecatore linea-superficie per calcolare le intersezioni. Una volta finito, il risultato è un vettore di punti di intersezione già ordinati lungo il profilo, pronti per essere interpolati mediante una delle procedure esposte al capitolo 2. nel caso fosse desiderabile, è possibile proiettare i punti trovati sulla superficie della fusoliera trovando i corrispondenti valori dei parametri *u* e *v* mediante l'algoritmo descritto al paragrafo 3.2.2.

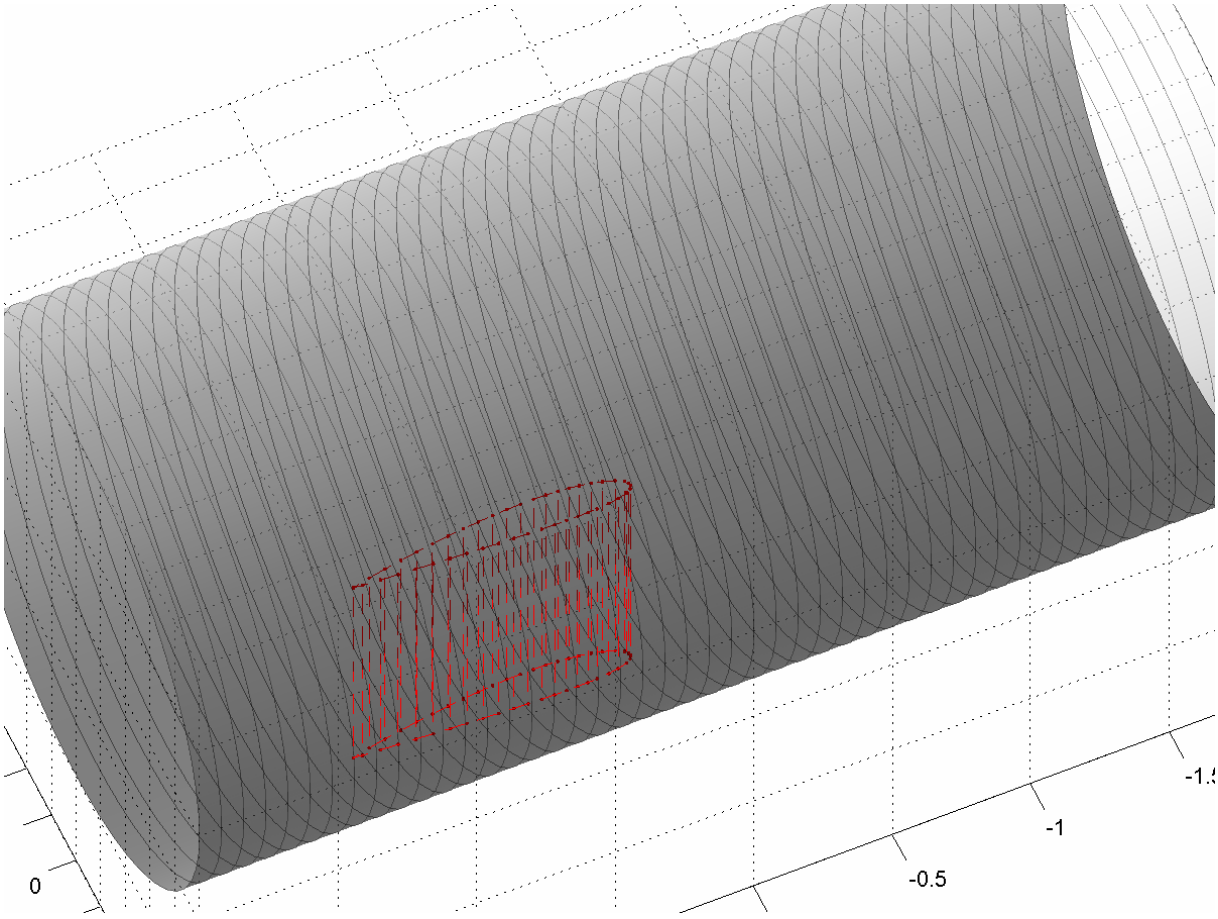


Figura 3-9 Esempio di superfici intersecate mediante il metodo *lattice evaluation*:superfici da intersecare

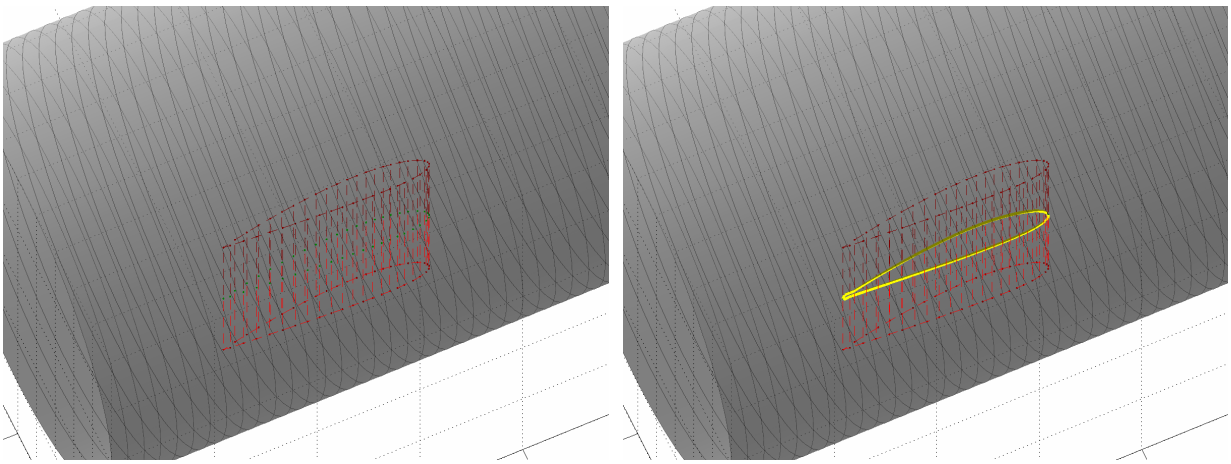


Figura 3-10 Esempio di superfici intersecate mediante il metodo *lattice evaluation*: risultato dell'intersezione

4. Conclusioni

Il lavoro di tesi ha consentito di risolvere alcuni problemi che il codice MSD mostrava, primo fra tutti quello di avere una rappresentazione efficiente delle superfici aerodinamiche. Inoltre è stato possibile, mediante gli strumenti sviluppati durante questo lavoro, ottenere forme più complesse eliminando alcune limitazioni che impedivano il calcolo di intersezioni multiple in fusoliera.

Mediante gli strumenti sviluppati in questa tesi, è stato possibile aggiornare il codice MSD per rappresentare l'intera configurazione dell'aeromobile mediante NURBS, rendendo così possibile un risparmio di tempo di calcolo considerevole e ottenere una rappresentazione accurata con una mole di dati notevolmente inferiore. Questa ultima si ripercuote anche sul salvataggio e l'esportazione; nonostante non sia ancora stato scritto un filtro si esportazione IGES, le stime sulla dimensione del file esportato sono molto positive, generando per la fusoliera un file di circa 110Kb invece che diversi Mb. Il formato di esportazione risulta inoltre compatibile con i più diffusi software CAD e generatori di griglia.

Gli sviluppi futuri del codice includono oltre alla messa a punto di un filtro IGES, anche un miglioramento delle procedure di intersezione, per renderle più robuste e capaci di intersecare superfici generiche in tempi ridotti. Inoltre è necessario sviluppare un'interfaccia grafica più "user-friendly", unitamente a un manuale operativo per consentire a un utente di comprendere il funzionamento del codice e poterlo utilizzare indipendentemente.

5. Appendice

Derivata di una funzione di base B-Spline

La derivata di una funzione di base B-Spline è data da:

$$N'_{i,p}(u) = \frac{p}{u_{i+p} - u_i} N_{i,p-1}(u) - \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \qquad \textbf{(5-1)}$$

Dimostriamo questa formula per induzione su p. Per $p = 1$, $N_{i,p-1}$ e $N_{i+1,p-1}$ sono uguali a 0 o 1, quindi $N'_{i,p}$ è:

$$\frac{1}{u_{i+1} - u_i} \qquad \text{oppure} \qquad -\frac{1}{u_{i+2} - u_{i+1}}$$

Assumendo che la formula (5-1) sia valida per $p-1$, $p>1$, dimostriamo per indizione che è valida per ogni p. derivando il prodotto $(fg)' = f'g + fg'$ ottengo:

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

Si ottiene quindi:

$$N'_{i,p} = \frac{1}{u_{i+p} - u_i} N_{i,p-1} + \frac{u - u_i}{u_{i+p} - u_i} N'_{i,p-1} - \frac{1}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1} + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N'_{i+1,p-1} \qquad \textbf{(5-2)}$$

Sostituendo l'equazione (5-1) nell'equazione (5-2) per $N'_{i,p-1}$ e $N'_{i+1,p-1}$ conduce alla seguente espressione:

$$
\begin{aligned}
N'_{i,p} &= \frac{1}{u_{i+p}-u_i}N_{i,p-1} - \frac{1}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1} \\
&+ \frac{u-u_i}{u_{i+p}-u_i}\left(\frac{p-1}{u_{i+p-1}-u_i}N_{i,p-2} - \frac{p-1}{u_{i+p}-u_{i+1}}N_{i+1,p-2}\right) \\
&+ \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}\left(\frac{p-1}{u_{i+p}-u_{i+1}}N_{i+1,p-2} - \frac{p-1}{u_{i+p+1}-u_{i+2}}N_{i+2,p-2}\right) \\
&= \frac{1}{u_{i+p}-u_i}N_{i,p-1} - \frac{1}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1} \\
&+ \frac{p-1}{u_{i+p}-u_i}\frac{u-u_i}{u_{i+p-1}-u_i}N_{i,p-2} \\
&+ \frac{p-1}{u_{i+p}-u_{i+1}}\left(\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} - \frac{u-u_i}{u_{i+p}-u_i}\right)N_{i+1,p-2} \\
&- \frac{p-1}{u_{i+p+1}-u_{i+1}}\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+2}}N_{i+2,p-2}
\end{aligned}
\tag{5-3}
$$

Osservando che:

$$
\begin{aligned}
&\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} - \frac{u-u_i}{u_{i+p}-u_i} = -1 + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} + 1 - \frac{u-u_i}{u_{i+p}-u_i} \\
&= \frac{u_{i+p+1}-u_{i+1}}{u_{i+p+1}-u_{i+1}} + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} + \frac{u_{i+p}-u_i}{u_{i+p}-u_i} - \frac{u-u_i}{u_{i+p}-u_i} \\
&= \frac{u_{i+p}-u}{u_{i+p}-u_i} - \frac{u-u_{i+1}}{u_{i+p+1}-u_{i+1}}
\end{aligned}
\tag{5-4}
$$

Si ottiene allora:

$$
\begin{aligned}
N'_{i,p} &= \frac{1}{u_{i+p}-u_i}N_{i,p-1} - \frac{1}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1} \\
&+ \frac{p-1}{u_{i+p}-u_i}\left(\frac{u-u_i}{u_{i+p-1}-u_i}N_{i,p-2} + \frac{u_{i+p}-u}{u_{i+p}-u_{i+1}}N_{i+1,p-2}\right) \\
&- \frac{p-1}{u_{i+p+1}-u_{i+1}}\left(\frac{u-u_{i+1}}{u_{i+p}-u_{i+1}}N_{i+1,p-2} + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+2}}N_{i+2,p-2}\right)
\end{aligned}
\tag{5-5}
$$

Usando la formula ricorsiva di Cox-deBoor (equazione (1-14)) le espressioni nelle parentesi possono essere sostituite da $N_{i,p-1}$ e $N_{i+1,p-1}$ rispettivamente:

$$
\begin{aligned}
N'_{i,p} &= \frac{1}{u_{i+p} - u_i} N_{i,p-1} - \frac{1}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1} \\
&+ \frac{p-1}{u_{i+p} - u_i} N_{i,p-1} - \frac{p-1}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1} \\
&= \frac{p}{u_{i+p} - u_i} N_{i,p-1} - \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}
\end{aligned}
\qquad \textbf{(5-6)}
$$

che completa la dimostrazione.

Bibliografia

[1] Alefeld, G., Herzberger, J., *Introduction to interval computations*, Academic Press Inc. 1983

[2] Butterfield, K.R., The computation of all derivatives of a B-Spline basis, *Jour. Inst. Math. Applic.*, Vol.17, pp. 15-25,1976

[3] De Boor, C., *A Practical Guide to Splines*, New York, Springler-Verlag 1978

[4] De Boor, C., *The splines Toolbox* Matlab user's guide Ver. 3, The Mathworks

[5] Huber, E., *Intersecting general parametric surfaces using bounding volumes*,

[6] Jean, B., Hamann, B., *And efficient surface-surface intersection algorithm using adaptive surface triangulation and space partitioning trees*, Math. Engng. Ind., Vol. 7, No. 1, pp. 25-40, VSP 1998

[7] Krishnan, S., Manocha, D., *An Efficient Surface Intersection Algorithm Based on Lower-Dimensional Formulation*, ACM Transactions on Graphics, Vol. 16, No. 1, January 1997, Pages 74–106.

[8] Kyu-Yeul Lee, Doo-Yeoun Cho, Tae-Wan Kim, *A Tracing algorithm for Surface/Surface Intersections on Surface boundaries*.

[9] Manocha, D., Demmel, J., *Algorithms for intersecting parametric and algebraic curves*. Technical Report UCB/CSD 92/698, Computer Science Division, University of California at Berkeley, 1992.

[10] Manocha, D., Demmel, J., *Algorithms for Intersecting Parametric and Algebraic Curves II: Multiple Intersections*. Technical Report UCB/CSD, Computer Science Division, University of California at Berkeley, 1992.

[11] Manocha, D., Canny, J.,*A new approach for surface intersection*, International Journal of Computational Geometry & Applications Vol. 1, No. 4 (1991) 491-516

[12] Manocha, D., Krishnan, S., *Algebraic Pruning: A Fast Technique for Curve and Surface Intersection*, Department of Computer Science, University of North Carolina, Chapel Hill, NC 27599-3175

[13] Papaleo, L., *Interactive curve and design*, Department of computer science, University of Genova.

[14] Piegl, L., Tiller W., *The NURBS Book* 2nd edition, Springer, 1997

Ringraziamenti

Desideriamo ringraziare tutti coloro che hanno contribuito alla realizzazione di questa tesi. In particolare il Professor Aldo Frediani, per la sua disponibilità durante tutto lo svolgimento del lavoro, l'Ing. Guido Saporito e l'Ing. Matteo Gasperini per avermi coinvolto in questo bellissimo progetto e avermi introdotto all'utilizzo del codice MSD. Desidero inoltre ringraziare il professor Tullio Franzoni e il Dott. Marco Franciosi del dipartimento di Matematica Applicata "Ulisse Dini" dell'Università di Pisa insieme agli altri docenti e dottorandi che hanno mostrato il loro interesse per questo lavoro.

Ringrazio anche il Professor Giulio Casciola del Dipartimento di Matematica dell'Università di Bologna per i fondamentali consigli forniti per procedere nella realizzazione degli algoritmi di intersezione.

Infine ringrazio i miei genitori, mio fratello, i parenti e tutti gli amici che mi hanno sostenuto e sopportato per tutto questo tempo.

Un ringraziamento speciale va a Alberto Tarroni, Giorgio Gasparroni e Vincenzo Ferracci, compagni storici di appartamento in compagnia dei quali ho affrontato progetti, esami, momenti difficili, e grazie ai quali ho trascorso e superato serenamente questi anni di università.