UNIVERSITÀ DEGLI STUDI
DI TORINO

FACOLTÀ DI SCIENZE M.F.N.

Corso di Laurea in Matematica
Tesi di Laurea Magistrale

T-spline: un passo oltre le NURBS

Relatore:
Prof. Catterina Dagnino

Candidato:
Fabio Roman

Sessione di laurea di Aprile 2012
Anno accademico 2010/2011

Indice

Introduzione

Gli strumenti matematici per la grafica computerizzata costituiscono uno dei campi di sviluppo più ferventi dell'analisi numerica. Dai risultati pionieristici di Bézier e de Casteljau molto é stato sviluppato: dalle curve e superfici polinomiali si é passati a quelle razionali introducendo la possibilità che punti di controllo distinti influissero in maniera diversa sulla forma dell'oggetto (attraverso l'attribuzione dei pesi), poi sono state concepite le B-spline per permettere un migliore controllo locale, in seguito anche queste sono state estese dal caso polinomiale a quello razionale, creando le `NURBS`, al momento uno dei capisaldi del `CAGD`. Ma la ricerca non si é affatto fermata ad esse: pur con tutti i vantaggi che queste offrono rispetto agli strumenti passati, sono state ideate delle loro evoluzioni, che permettessero di ottimizzare ulteriormente quanto già le `NURBS` permettevano di fare rispetto alle B-spline e a Bézier. In particolare, dopo una successione di passaggi intermedi, si é giunti alle **T-spline**, attualmente ritenute uno dei metodi più avanzati nel campo della computer graphics. Le T-spline sono l'oggetto principale di questa tesi. Nel capitolo 1 introdurremo dunque questo strumento matematico, illustrando come è stato sviluppato a partire dal concetto di B-spline e mostrando alcune costruzioni intermedie; argomento del capitolo 2 sarà una trattazione estensiva delle T-spline, spiegando di che cosa si tratta sia a livello intuitivo che a livello rigoroso. Nel capitolo 3 mostreremo alcune azioni relative alle superfici T-spline, quali l'inserimento di nuovi punti di controllo e la conversione in altri formati; nel capitolo 4 affronteremo un argomento prettamente teorico, studiando l'indipendenza lineare delle funzioni di miscelamento T-spline associate ad alcune particolari T-mesh, mentre nel capitolo 5 vedremo alcune procedure ed argomentazioni che riguardano disegno tecnico e `CAGD`. Oggetto del capitolo 6 saranno alcune applicazioni realizzate grazie al software *Rhinoceros*, mostrando come esso permetta di utilizzare le T-spline per gli usi pratici. Infine, nel capitolo 7 sarà presentato un metodo di watermarking applicato a superfici T-spline.
A complemento della tesi sono presentate due appendici, nelle quali vengono illustrate le potenzialità del software *Rhinoceros*, andando a vedere un elenco di funzionalità e comandi proposti dal programma.

Capitolo 1

Introduzione alle T-spline

Lo strumento matematico, di recente formalizzazione, affrontato da questa tesi, è costituito dalle superfici T-spline. Per prima cosa, andiamo a vedere in che cosa consistono questi nuovi oggetti: essi sono una generalizzazione del concetto di superfici `NURBS`, e la differenza consiste nel fatto che, mentre nel più noto caso delle B-spline razionali non uniformi, la topologia dei punti di controllo deve essere necessariamente rettangolare, nel caso di queste superfici viene ammessa la possibilità che la griglia degli stessi presenti dei buchi al proprio interno, o equivalentemente che una riga di punti di controllo possa essere fatta terminare senza che sia stato necessariamente raggiunto un numero di punti fissato a priori.
Questo può costituire un consistente vantaggio dal momento che si rende possibile ridurre, anche significativamente, il numero di punti di controllo necessari per rappresentare una superficie ad un certo livello di dettaglio: molto spesso infatti, in una `NURBS`, un numero di punti di controllo anche importante deve essere inserito soltanto per soddisfare la necessità di una topologia rettangolare, senza che alla loro presenza corrisponda un'effettiva utilità in termini di rappresentazione della superficie, ovvero essi non apportano informazioni geometriche significanti.
La presenza di un numero di punti di controllo superiore al necessario può portare problemi non soltanto di ordine computazionale, quali possono essere per esempio la necessità di avere a che fare con più dati a priori e la maggiore complessità nei calcoli con relativa diminuzione delle performance del calcolatore al momento di processare la mesh, ma anche questioni relative alla corretta visualizzazione dell'oggetto rappresentato. I punti in più possono infatti portare ad introdurre piccoli artefatti, che rendono la rappresentazione diversa, anche se leggermente, da quella desiderata, e disturbano la visione.
La trattazione seguente prende spunto da [1] e [2], due articoli tra i più autorevoli a trattare il tema.

1.1 Dalle B-spline alle T-spline

Il passaggio dal concetto di B-spline a quello di T-spline non è stato immediato. Il problema del controllo locale nell'operazione di raffinamento, ovvero la possibilità di inserire un singolo punto di controllo senza propagare la modifica ad un'intera riga o ad un'intera colonna, è stato affrontato per la prima volta in letteratura nel 1988, da Forsey e Bartels [3]. Essi proposero le cosiddette B-spline gerarchiche, sviluppate poi ulteriormente da Kraft nel 1998 [4], che ne diede una caratterizzazione naturale, costruendo uno spazio spline multilivello come spazio lineare di B-spline tensore prodotto su livelli di griglie differenti ed ordinate in maniera gerarchica, sul quale poi definire un meccanismo di selezione per le B-spline, che garantisse l'indipendenza lineare in modo tale da formare una base.
Altri studi, come [5] e [6], si occuparono di ottenere lo stesso risultato proponendo variazioni sul tema, che tentassero di volta in volta di evitare le complicazioni che si sarebbero venute a formare imponendo questo tipo di gerarchia.
Il vantaggio della rappresentazione T-spline è quello di non necessitare in alcuna misura di una gerarchia: tutto il raffinamento locale viene svolto su di un'unica griglia di controllo, pertanto il livello gerarchico è unico, con tutti i punti di controllo che hanno la stessa influenza sulla forma della superficie.
Consideriamo d'ora in avanti, per semplicità, curve e superfici cubiche, anche se il discorso si può facilmente estendere a strutture di ogni grado, in particolar modo se esso è dispari. La maggior parte del lavoro di ricerca viene compiuto su cubiche, e anche in questa trattazione, se non diversamente specificato nel contesto, ci si limiterà a lavorare su oggetti di grado 3.

1.2 Le B-spline gerarchiche

Una superficie B-spline gerarchica, strumento che come detto si colloca storicamente tra le `NURBS` e le T-spline, consiste di una successione di livelli, ognuno dei quali possiede una collezione di patch B-spline. Il livello 0 ha la risoluzione più bassa, descrivendo la forma basilare della superficie, e i suoi *figli* (detti *sovrastrati*, dall'inglese *overlay*) descrivono alcuni dettagli della superficie, corrispondendo ad aree raffinate della superficie *genitore* ([17]).
Un sovrastrato al livello k viene generato scegliendo una patch sul suo genitore al livello $k-1$, eseguendo un passo di raffinamento, e manipolando i punti di controllo della patch raffinata; in questo modo, un sovrastrato al livello k risulta avere due volte la risoluzione del suo genitore al livello $k-1$.
Allo stesso livello, i sovrastrati devono essere disgiunti a coppie; se ce ne sono due ad intersezione non vuota, essi vengono fusi in un'unico sovrastrato.
Per garantire la continuità C^2 tra un sovrastrato e il suo genitore, la manipolazione di punti di controllo deve essere ristretta a quelli centrali, lasciando

invariati quelli periferici. Per una superficie B-spline bicubica, un sovrastrato ha una mesh di dimensione non inferiore a 7×7, e i punti di controllo sui tre anelli esterni devono rimanere invariati.
Nello schema delle B-spline gerarchiche, una maniera conveniente per rappresentare un sovrastrato è di scomporlo additivamente in una parte derivata dal genitore (*reference*) e in una parte consistente nello spostamento dei punti di controllo (*offset*). Più precisamente, ogni punto di controllo $P^{(k)}$ di un sovrastrato al livello k può essere rappresentato nella forma $P^{(k)} = R^{(k)} + O^{(k)}$, dove $R^{(k)}$ discende dalla superficie genitore al livello $k-1$ tramite un raffinamento midpoint, mentre $O^{(k)}$ è il vettore di spostamento, che identifica ogni cambiamento nei punti di controllo al livello k; per quelli che restano immutati, naturalmente, $O^{(k)} = 0$.
L'offset è ciò che attua la differenza di forma della superficie tra un livello di rappresentazione e il successivo: ogni cambiamento ad un livello più basso si propaga di conseguenza a tutti i livelli più alti, e il variare $O^{(k)}$ produce un cambiamento locale. In termini computazionali, il vantaggio dell'offset è che rende necessario memorizzare soltanto gli spostamenti non nulli; nel caso di un sovrastrato di dimensione 7×7, ce n'è soltanto uno.
Questa è la formulazione tradizionale, tuttavia il concetto di B-spline gerarchiche si può generalizzare. Due aspetti che permettono di andare oltre tale formulazione, considerando anche casi un po' diversi, riguardano la possibilità di non effettuare necessariamente un raffinamento midpoint per determinare $R^{(k)}$, e quello di utilizzare le superfici B-spline razionali senza limitarsi soltanto a quelle polinomiali. Come risultato della prima generalizzazione, un nodo in una sequenza di nodi di un sovrastrato può essere una combinazione convessa arbitraria di due nodi adiacenti nella sequenza di nodi della superficie genitore; per via della seconda generalizzazione, si utilizzano le coordinate omogenee. Si mantiene il fatto che i punti di controllo sui tre anelli esterni debbano rimanere invariati.

1.3 Intervalli tra nodi nelle T-spline

Un intervallo tra nodi è un intero non negativo assegnato ad ogni lato di una griglia di controllo di una B-spline, al fine di avere informazioni sui nodi stessi.
Il concetto si introduce in maniera efficace trattando il caso bidimensionale delle curve, affermando che ogni intervallo tra nodi è la differenza tra due nodi consecutivi nel vettore dei nodi; se la curva non è periodica, si assegnano due ulteriori intervalli, uno prima del primo punto ad indicare la differenza tra i valori dei primi due nodi, e uno dopo l'ultimo punto, pari alla differenza tra i valori degli ultimi due nodi. La figura 1.1 mostra un tale esempio.
Si osserva che per tutti i lati del poligono di controllo, tranne il primo e l'ultimo, l'intervallo tra nodi relativo ad ogni lato, corrisponde alla lunghezza

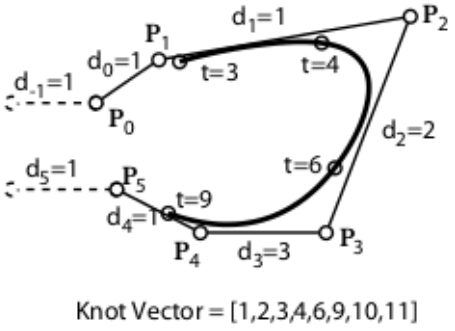


Figura 1.1: Curva B-spline cubica con intervalli tra nodi.

del segmento di curva in cui il lato mappa, anche se in termini parametrici, ovvero è la differenza tra il valore del parametro nel punto finale del segmento e il valore del punto iniziale.
Questa osservazione ci può essere utile a comprendere come, aggiungendo una costante a tutti i nodi di un dato vettore di nodi, dal momento che gli intervalli tra i nodi non cambiano, la curva mantiene la stessa forma e le stesse caratteristiche; pertanto, siamo liberi di scegliere l'origine, ovvero di porre lo zero del valore dei nodi, di volta in volta dove ciò possa risultare più comodo.
Il passaggio al caso tridimensionale delle superfici è poi immediato, consistendo esso nell'utilizzo di due parametri per i nodi anziché uno, e nel calcolo degli intervalli tra nodi come le variazioni del parametro che cambia, dal momento che, considerata la griglia, immaginando di spostarsi su di un lato di essa, da un punto ad un altro direttamente collegato, si trova a variare soltanto uno dei due parametri.
Il caso delle T-spline presenta la complicazione della non necessaria rettangolarietà della griglia. In questo caso, gli intervalli tra nodi ci permettono di imporre un sistema di coordinate nodali *locali* sulla superficie, definendo una griglia più fitta regolare (cioè rettangolare), come mostrato in figura 1.2.
Assegniamo per esempio al punto $\mathbf{P}_{00}$ le coordinate nodali locali (d_0, e_0). Sulla griglia regolare più fitta che abbiamo ottenuto (potremmo chiamarla *rettangolarizzazione*) possiamo definire due vettori locali di nodi come:

$$\{d\} = \{0, \overline{d}_0, \overline{d}_1, \ldots\} \qquad ; \qquad \{e\} = \{0, \overline{e}_0, \overline{e}_1, \ldots\}$$

dove:

$$\overline{d}_i = \sum_{j=0}^{i} d_j \quad ; \quad \overline{e}_i = \sum_{j=0}^{i} e_j$$

e definiamo anche, potendoci essere utile come convenzione, $\overline{d}_{-1} = \overline{e}_{-1} = 0$.
In questo modo possiamo etichettare ogni punto di controllo in un modo, detto *etichetta polare*, che renda espliciti i valori dei nodi vicini al quale fa

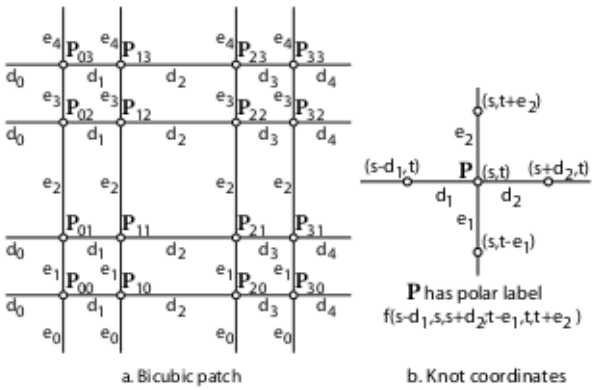


Figura 1.2: Regione di una mesh di controllo `NURBS` etichettata con intervalli tra nodi.

riferimento lo stesso, rispetto al sistema di coordinate locali imposto.
Per il generico punto $\mathbf{P}_{ij}$, l'etichetta polare si trova ad essere:

$$f(\overline{d}_{i-1}, \overline{d}_i, \overline{d}_{i+1}; \overline{e}_{i-1}, \overline{e}_i, \overline{e}_{i+1})$$

e la superficie definita da questa griglia rettangolarizzata si può scrivere come:

$$\mathbf{P}(s,t) = \sum_i \sum_j \mathbf{P}_{ij} N_i^3(s) N_j^3(t)$$

dove le $N_i^3(s)$ sono le funzioni di base B-spline cubiche su $\{d\}$ e le $N_j^3(t)$ sono le funzioni di base B-spline cubiche su $\{e\}$ (3 è grado, non ordine).

1.4 Un passo intermedio tra B-spline e T-spline: le PB-spline

Introduciamo ora delle curve e superfici che si possono ritenere un passo oltre le B-spline, ma ancora un passo indietro in confronto alle T-spline: le PB-spline, dove PB sta per *point based*. In termini di "via di mezzo", rispetto all'approccio gerarchico visto nel § 1.2, si evita tutta la complicazione della struttura multilivello, ma allo stesso tempo si organizzano i punti di controllo in una maniera meno intuitiva, senza un vero schema topologico per essi, rendendo il tutto gestibile con meno agevolezza.
L'equazione di una PB-spline può essere scritta come:

$$\mathbf{P}(s,t) = \frac{\sum_{i=1}^n \mathbf{P}_i w_i B_i(s,t)}{\sum_{i=1}^n w_i B_i(s,t)} = \sum_{i=1}^n \mathbf{P}_i R_i(s,t) \qquad (s,t) \in \mathbf{D}$$

dove i $\mathbf{P}_i$ sono i punti di controllo, i w_i sono pesi, e le $B_i(s,t)$ sono funzioni di base definite come:

$$B_i(s,t) = N_{i0}^3(s) N_{i0}^3(t)$$

dove $N_{i0}^3(s)$ è la funzione B-spline di base cubica associata al vettore dei nodi:

$$\mathbf{s}_i = [s_{i0}, s_{i1}, s_{i2}, s_{i3}, s_{i4}]$$

mentre $N_{i0}^3(t)$ è la funzione B-spline di base cubica associata al vettore dei nodi:

$$\mathbf{t}_i = [t_{i0}, t_{i1}, t_{i2}, t_{i3}, t_{i4}]$$

come illustrato in figura 1.3; in alternativa, si può scrivere come combinazione lineare dei punti di controllo con coefficienti funzioni di miscelamento $R_i(s,t)$, definite da

$$R_i(s,t) = \frac{w_i B_i(s,t)}{\sum_{j=1}^{n} w_j B_j(s,t)}$$

Dunque, per specificare una PB-spline, è necessario e sufficiente fornire una sequenza di punti di controllo e una coppia di vettori di nodi per ognuno di essi.

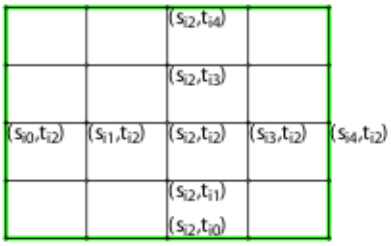


Figura 1.3: Linee nodali per la funzione base $B_i(s,t)$.

Ad ogni punto di controllo $\mathbf{P}_i$ si può associare un *dominio d'influenza* $\mathbf{D}_i$, corrispondente al prodotto degli intervalli dei nodi $(s_{i0}, s_{i4}) \times (t_{i0}, t_{i4})$. Tali domini possono anche in teoria essere non allineati con gli assi, sebbene si lavori di solito su domini allineati con gli assi, al fine di evitare complicazioni. Il concetto di dominio di influenza può essere utile per discutere alcune proprietà. Prima fra tutte, quella dell'inviluppo convesso, che è valida anche per le PB-spline, e in questo caso si formalizza nel seguente modo: denotando $C(s,t) = \{\mathbf{P}_i | (s,t) \in \mathbf{D}_i\}$, allora $\mathbf{P}(s,t)$ sta nell'inviluppo convesso di $C(s,t)$ nel senso classico del termine.
Questa proprietà ci permette di compiere delle considerazioni sul dominio $\mathbf{D}$, che è quello sul quale l'intera PB-spline è definita. Le uniche restrizioni su di esso sono date dal fatto che dovrà verificarsi $\sum_{i=1}^{n} B_i(s,t) > 0$ per ogni $(s,t) \in \mathbf{D}$, e che $\mathbf{D}$ dovrà essere connesso. Questo implica che $\mathbf{D} \subset \{\mathbf{D}_1 \cup \mathbf{D}_2 \cup \cdots \cup \mathbf{D}_n\}$, ma non richiede che lo stesso debba essere per forza rettangolare.
Inoltre, proprio per essa, se $(s,t) \in \mathbf{D}$ sta soltanto in un dominio di influenza $\mathbf{D}_i$, allora $\mathbf{P}(s,t) = P_i$; se $(s,t) \in \mathbf{D}$ sta in due domini di influenza $\mathbf{D}_i, \mathbf{D}_j$ e solo in quelli, allora $\mathbf{P}(s,t)$ sta sul segmento che collega P_i con P_j, e così via.

Pertanto, sarebbe opportuno che ogni punto in $\mathbf{D}$ stia almeno in tre domini di influenza distinti; in generale, tuttavia, non c'è una scelta "migliore" per $\mathbf{D}$, assegnati i $\mathbf{D}_i$.

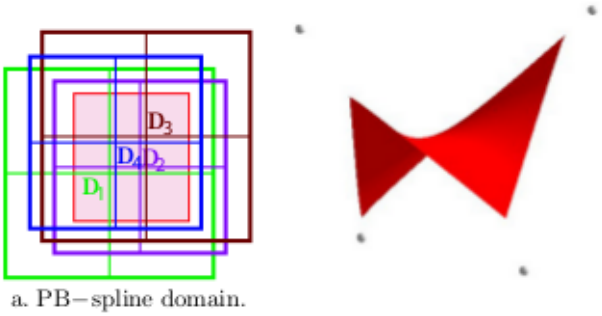


Figura 1.4: Una PB-spline cubica con quattro punti di controllo.

La figura 1.4 mostra uno spazio di parametri in cui sono rappresentati i $\mathbf{D}_i$ per una PB-spline composta da quattro funzioni di miscelamento; la superficie risultante è mostrata sulla destra. Le etichette $\mathbf{D}_i$ sono posizionate al centro dei rispettivi domini, e una possibile scelta per $\mathbf{D}$ è evidenziata in rosso.
Un'osservazione importante per concludere il paragrafo, e per caratterizzare anche a livello intuitivo queste curve, sta nel fatto che non esiste alcuna mesh: i vettori dei nodi per una funzione di base sono completamente indipendenti dai vettori dei nodi di ogni altra funzione di base. Questo permette più libertà rispetto alla topologia rettangolare che viene invece richiesta nei casi B-spline e `NURBS`, ma allo stesso tempo rende meno agevole il controllo delle forme ottenute; serve trovare un compromesso, e uno buono è proprio costituito dalle T-spline che tratteremo ora.

Capitolo 2

Le T-spline nel dettaglio

2.1 Le T-spline come particolari PB-spline

Una T-spline può essere vista come una PB-spline per la quale è stato imposto un ordine sui punti di controllo, tramite una griglia di controllo chiamata *T-mesh*. Una T-mesh serve a due scopi: per prima cosa, fornisce la possibilità di maneggiare la superficie in una maniera più intuitiva, rispetto a quanto si possa fare con l'insieme dei punti di controllo di una PB-spline, che come abbiamo detto è completamente arbitrario; inoltre, i vettori dei nodi $\mathbf{s}_i$ e $\mathbf{t}_i$ per ogni funzione di base si ricavano direttamente da essa, senza la necessità di doverli definire esplicitamente e/o di dover trovare un metodo sensato per assegnarli ([1]).
La figura 2.1 mostra la preimmagine di una T-mesh nello spazio dei parametri (s,t). Gli s_i denotano la coordinata s, mentre i t_i denotano la coordinata t; inoltre, i d_i e gli e_i denotano gli intervalli tra nodi, con i lati più esterni a contenere le condizioni agli estremi. Di conseguenza, per esempio, $s_4 = s_3 + d_3$, e $t_5 = t_4 + e_4$. Ad ogni vertice sono assegnate coordinate nodali: per esempio, $\mathbf{P}_1$ ha coordinate (s_3, t_2), mentre $\mathbf{P}_2$ ha coordinate $(s_5 - d_8, t_3)$.

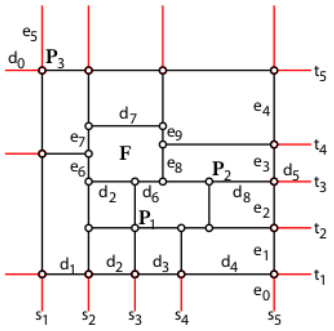


Figura 2.1: Preimmagine di una T-mesh

Una T-mesh è fondamentalmente una griglia rettangolare che permette le

cosiddette *T-giunzioni*. Se, considerata la preimmagine di una T-mesh, chiamiamo *s-lati* i segmenti per i quali s è costante (quelli verticali nell'immagine), e *t-lati* i segmenti per i quali t è costante (quelli orizzontali nell'immagine), possiamo identificare le T-giunzioni come i vertici condivisi da un s-lato e due t-lati, oppure da due s-lati e un t-lato. Ogni lato di una T-mesh è etichettato da un intervallo tra nodi; l'insieme di tali intervalli deve rispettare le seguenti regole:

- la somma degli intervalli tra nodi su lati opposti di ogni faccia deve essere uguale; per esempio, per la faccia denotata con **F** nella figura 2.1, dovrà valere $d_2 + d_6 = d_7$ e $e_6 + e_7 = e_8 + e_9$.
- se una T-giunzione su di un lato di una faccia può essere connessa ad una T-giunzione su di un lato opposto della faccia, dividendo di conseguenza la faccia in due facce, senza violare la regola precedente, questo lato deve venire incluso.

Ad ogni $\mathbf{P}_i$ corrisponde una funzione base $B_i(s,t)$ definita in termini di vettori di nodi $\mathbf{s}_i = [s_{i0}, s_{i1}, s_{i2}, s_{i3}, s_{i4}]$ e $\mathbf{t}_i = [t_{i0}, t_{i1}, t_{i2}, t_{i3}, t_{i4}]$.
Le coordinate nodali di $\mathbf{P}_i$ sono (s_{i2}, t_{i2}); i nodi s_{i3} e s_{i4} si trovano considerando una linea nello spazio dei parametri $\mathbf{R}(\alpha) = (s_{i2} + \alpha, t_{i2})$. Allora s_{i3} e s_{i4} sono le coordinate s dei primi due s-lati intersecati dalla linea, escludendo l'iniziale (s_{i2}, t_{i2}). Gli altri nodi in $\mathbf{s}$ e in $\mathbf{t}$ si trovano in maniera simile.
Nell'esempio in figura, per $\mathbf{P}_1$, $\mathbf{s}_i = [s_1, s_2, s_3, s_4, s_5 - d_8]$ e $\mathbf{t}_i = [t_1 - e_0, t_1, t_2, t_3, t_4 + e_9]$. Allo stesso modo, per $\mathbf{P}_2$, $\mathbf{s}_i = [s_3, s_3 + d_6, s_5 - d_8, s_5, s_5 + d_5]$ e $\mathbf{t}_i = [t_1, t_2, t_3, t_4, t_5]$. $\mathbf{P}_3$ è un punto di controllo al margine, e in questo caso la scelta di s_{30} e di t_{34} è irrilevante, possiamo dunque prendere $\mathbf{s}_i = [s_1 - d_0, s_1 - d_0, s_1, s_2, s_2 + d_7]$ e $\mathbf{t}_i = [t_1, t_5 - e_4 + e_9 - e_7, t_5, t_5 + e_5, t_5 + e_5]$.
Una volta determinati i vettori dei nodi per ogni funzione base, l'equazione di una T-spline è quella di una PB-spline con quella scelta per i vettori.

$$\mathbf{P}(s,t) = \frac{\sum_{i=1}^{n} \mathbf{P}_i w_i B_i(s,t)}{\sum_{i=1}^{n} w_i B_i(s,t)} = \sum_{i=1}^{n} \mathbf{P}_i R_i(s,t) \qquad (s,t) \in \mathbf{D}$$

La motivazione della seconda regola è mostrata in figura 2.2. In quel caso, l'intervallo tra nodi pari a zero significa che sarebbe possibile avere un lato a collegare **P** con **A**, ma sarebbe possibile anche averne uno che collega **P** con **B**, risultando a seconda del caso in un vettore dei nodi **T** per **P** differente nei due casi. Tale regola serve a risolvere questa ambiguità, obbligando a decidere quale scelta intraprendere nella definizione stessa della T-mesh.
Osserviamo infine che, se la T-mesh è una griglia regolare, allora la T-spline si riduce ad una tradizionale superficie B-spline tensore prodotto.

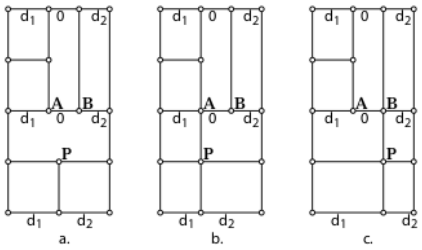


Figura 2.2: Possibile ambiguità.

2.2 Spazi T-spline

Definiamo uno spazio T-spline come l'insieme di tutte le T-spline che possiedono la stessa T-mesh in termini di topologia, intervalli tra nodi, e sistema di coordinate dei nodi.
Uno spazio T-spline può essere rappresentato con un singolo diagramma di una preimmagine di una T-mesh (figura 2.1); naturalmente si può parlare di preimmagine di uno spazio T-spline, possedendo tutte le T-spline di uno spazio la stessa preimmagine.
Uno spazio T-spline S_1 si dice *sottospazio* di S_2 (denotando $S_1 \subset S_2$) se è possibile passare da una T-spline in S_1 ad una T-spline in S_2 tramite un procedimento di raffinamento locale, che vedremo in seguito.
Se T_1 è una T-spline, allora $T_1 \in S_1$ significa che T_1 ha una griglia di controllo la cui topologia e i cui intervalli tra nodi sono specificati da S_1.
La figura 2.3 mostra una successione incapsulata di spazi T-spline, ovvero tale per cui $S_1 \subset S_2 \subset S_3 \subset \ldots \subset S_n$.

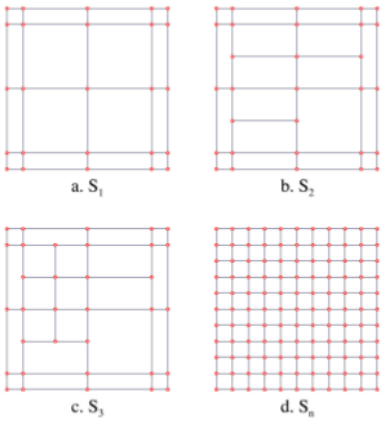


Figura 2.3: Successione incapsulata di spazi T-spline.

2.3 Una definizione rigorosa di T-mesh e T-spline

Vediamo ora una definizione un po' più astratta, ma in un certo senso più rigorosa, di T-mesh e T-spline ([18], [19]). Siano $\mathcal{I}_i = \{1, 2, \ldots, n\}$ e $\mathcal{I}_j = \{1, 2, \ldots, m\}$ due insiemi di *indici*, sia $\mathfrak{U} \subseteq \mathcal{I}_i \times \mathcal{I}_j$ un insieme di *ancore*, e sia $\mathfrak{E}$ un insieme di *lati* chiusi orizzontali o verticali tra ancore, ovvero:

$$\mathfrak{E} \subseteq \{[(i_1, j_1), (i_2, j_2)] \text{ in maniera che } (i_1, j_1) \neq (i_2, j_2) \in \mathfrak{U}, \text{ con } i_1 = i_2 \text{ o } j_1 = j_2\}$$

dove $[(i_1, j_1), (i_2, j_2)]$ rappresenta un lato chiuso (o un segmento di linea) con (i_1, j_1) e (i_2, j_2) come estremi.
Siano ora $\Xi_s = \{s_{-1}, \ldots, s_{n+2}\}$ e $\Xi_t = \{t_{-1}, \ldots, t_{m+2}\}$ vettori di nodi aperti sull'intervallo $[0, 1]$, ovvero tali per cui:

- $\Xi_s : 0 = s_{-1} = s_0 = s_1 = s_2 < s_3 \leq s_4 \leq \ldots \leq s_{n-3} \leq s_{n-2} < s_{n-1} = s_n = s_{n+1} = s_{n+2} = 1$;
- $\Xi_t : 0 = t_{-1} = t_0 = t_1 = t_2 < t_3 \leq t_4 \leq \ldots \leq t_{m-3} \leq t_{m-2} < t_{m-1} = t_m = t_{m+1} = t_{m+2} = 1$;
- i nodi interni possiedono molteplicità non superiore a 3.

Una ***T-mesh*** $\mathfrak{M} = \mathfrak{M}(\mathcal{I}_i, \mathcal{I}_j, \mathfrak{U}, \mathfrak{E}, \Xi_s, \Xi_t)$ è definita dagli oggetti precedenti qualora siano verificate le seguenti due condizioni:

1. $\forall e_1, e_2 \in \mathfrak{E}, e_1 \cap e_2 \in \mathfrak{U}$, ovvero esiste un'ancora su ogni intersezione tra lati, e i lati non si sovrappongono.
2. $\forall a \in \mathfrak{U}, \exists e \in \mathfrak{E} | a \in e$, ovvero non esistono ancore isolate.

Si può analizzare come la prima condizione porti ad una definizione delle T-mesh leggermente estesa rispetto a quella tradizionale, dove sono ammesse solo T-giunzioni, L-giunzioni, o giunzioni tra quattro lati, ma che di fatto non inerisce la nostra trattazione.
Una rappresentazione grafica della T-mesh $\mathfrak{M}$ può essere ottenuta disegnando ancore e lati nel cosiddetto *spazio degli indici*, o nello *spazio indici/parametri* o ancora nello *spazio dei parametri*, come mostrato in figura 2.4.
Nello spazio degli indici, i rappresenta l'indice orizzontale, e j quello verticale. Nello spazio dei parametri, le variabili orizzontali e verticali sono rispettivamente s e t. Una volta che una T-mesh $\mathfrak{M}$ è data nello spazio degli indici, la corrispondente T-mesh nello spazio dei parametri si ottiene mappando gli indici sui corrispondenti vettori dei nodi, ovvero eseguendo $i \mapsto s_i, j \mapsto t_j$. Fissato ora $\overline{j} \in \mathcal{I}_j$, definiamo:

$$\mathcal{I}_i^{\overline{j}} := \{i : (i, \overline{j}) \in \mathfrak{U} \text{ o esiste un lato verticale } e \in \mathfrak{E} \text{ tale che } (i, \overline{j}) \in e\}$$

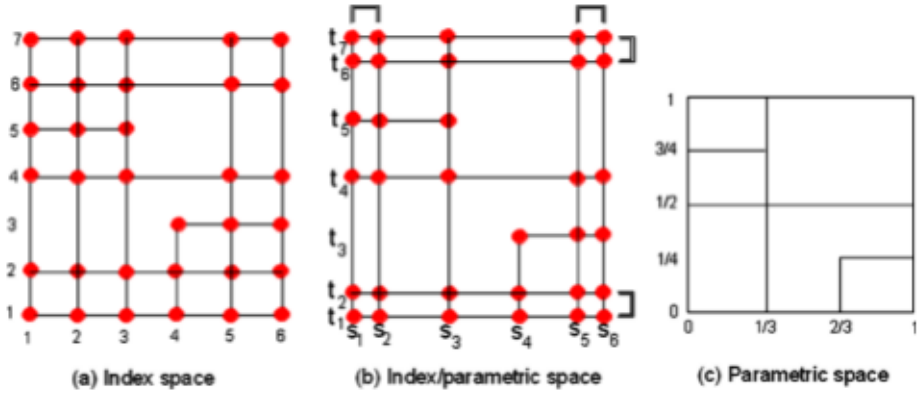


Figura 2.4: Spazi di indici e parametri.

e $\Xi_s^{\bar{j}}$ come il vettore degli s-nodi corrispondenti agli indici in $\mathcal{I}_i^{\bar{j}}$. In maniera simile, per $\bar{i} \in \mathcal{I}_i$ fissato, definiamo:

$$\mathcal{I}_j^{\bar{i}} := \{j : (\bar{i}, j) \in \mathfrak{U} \text{ o esiste un lato orizzontale } \in \mathfrak{E} \text{ tale che } (\bar{i}, j) \in e\}$$

e $\Xi_t^{\bar{i}}$ come il vettore degli t-nodi corrispondenti agli indici in $\mathcal{I}_j^{\bar{i}}$.
Per esempio, consideriamo la T-mesh in figura 2.4. In questo caso, $n = 6, m = 7, \Xi_s = \{s_{-1}, s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$ e
$\Xi_t = \{t_{-1}, t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9\}$ sono i vettori dei nodi globali della T-mesh $\mathfrak{M}$ in figura 2.4. Abbiamo le seguenti relazioni tra indici e nodi:

$$\begin{aligned} \mathcal{I}_i &= \{1, 2, \ldots, 6\} \to \{s_1, s_2, \ldots, s_6\} = \{0, 0, 1/3, 2/3, 1, 1\}, \\ \mathcal{I}_j &= \{1, 2, \ldots, 7\} \to \{s_1, s_2, \ldots, s_7\} = \{0, 0, 1/4, 1/2, 3/4, 1, 1\} \end{aligned}$$

e inoltre:

$$\begin{aligned} \mathcal{I}_i^3 &= \{1, 2, 3, 4, 5, 6\}, \quad \Xi_s^3 = \{0, 0, 1/3, 2/3, 1, 1\} \\ \mathcal{I}_i^5 &= \{1, 2, 3, 5, 6\}, \quad \Xi_s^5 = \{0, 0, 1/3, 1, 1\} \\ \mathcal{I}_j^4 &= \{1, 2, 3, 4, 6, 7\}, \quad \Xi_t^4 = \{0, 0, 1/4, 1/2, 1, 1\} \end{aligned}$$

Abbiamo anche un'ancora come ogni vertice della T-mesh nello spazio degli indici; ogni ancora è indicata da un circoletto in figura 2.4a. Benché non strettamente necessario, piazziamo sempre due linee di ancore al bordo dello spazio degli indici, che corrispondono a nodi di molteplicità 4.
Da una parte, la mesh nello spazio dei parametri è proprio ciò in cui viene mappato l'insieme degli indici, ma dall'altra essa non contiene tutte le informazioni della corrispondente T-mesh nello spazio degli indici, in quanto alcuni nodi potrebbero essere ripetuti, e il solo spazio parametrico non permette di distinguere tra casi in cui l'insieme dei nodi è lo stesso, e cambia soltanto la molteplicità di ognuno di essi.

Per ovviare a tale ambiguità, introduciamo la rappresentazione indici/parametri (figura 2.4b.), nella quale sono presenti tutte le informazioni. Le linee verticali e orizzontali che sono tracciate tra di loro più vicine delle altre, e contrassegnate da parentesi parallele, corrispondono a nodi ripetuti.
Sia $\mathfrak{M}$ una T-mesh. Definiamo su $\mathfrak{M}$ un insieme di funzioni B-spline tensore prodotto, chiamate *funzioni di miscelamento T-spline*, sullo spazio dei parametri $[0,1]^2$. Associamo una funzione di miscelamento T-spline ad ogni ancora di $\mathfrak{M}$. Dato $(\bar{i},\bar{j}) \in \mathfrak{U}$, la corrispondente funzione di miscelamento viene costruita come prodotto di una spline nella direzione s, $s \mapsto N_s^{(\bar{i},\bar{j})}(s)$, e di una nella direzione t, $t \mapsto N_t^{(\bar{i},\bar{j})}(t)$. $N_s^{(\bar{i},\bar{j})}$ ha $s_{\bar{i}}$ come nodo centrale ed è costruita sul vettore dei nodi $\Xi_s^{\bar{j}}$, mentre $N_t^{(\bar{i},\bar{j})}$ ha $t_{\bar{j}}$ come nodo centrale ed è costruita sul vettore dei nodi $\Xi_t^{\bar{i}}$. La B-spline associata a $(\bar{i},\bar{j}) \in \mathfrak{U}$ è:

$$(s,t) \mapsto B^{(\bar{i},\bar{j})}(s,t) = N_s^{(\bar{i},\bar{j})}(s) N_t^{(\bar{i},\bar{j})}(t)$$

Per esempio, costruiamo la funzione di miscelamento associata all'ancora $(3,4)$ della T-mesh in figura 2.4a. Dai vettori dei nodi:

$$\begin{aligned} \Xi_s^4 &= \{s_1, s_2, s_3, s_5, s_6\} = \{0,0,1/3,1,1\} \\ \Xi_t^3 &= \{t_1, t_2, t_4, t_5, t_6\} = \{0,0,1/2,3/4,1,1\} \end{aligned}$$

possiamo estrarre i vettori di nodi locali:

$$\{s_1, s_2, s_3, s_5, s_6\} = \{0,0,1/3,1,1\} \text{ e } \{t_1, t_2, t_4, t_5, t_6\} = \{0,0,1/2,3/4,1\}$$

e costruire la T-spline:

$$B^{(3,4)}(s,t) := N[0,0,1/3,1,1](s) N[0,0,1/2,3/4,1](t) := N_s^{(3,4)}(s) N_t^{(3,4)}(t)$$

Ogni ancora è in corrispondenza biunivoca con la sua corrispondente funzione di miscelamento T-spline bicubica. Con la data T-mesh $\mathfrak{M}$, le funzioni di miscelamento T-spline bicubiche associate generano uno spazio di funzioni, denotato come:

$$S(\mathfrak{M}) := \text{span}\{B^{(i,j)}(s,t) : B^{(i,j)}(s,t) = N_s^{(i,j)}(s) N_t^{(i,j)}(t), (i,j) \in \mathfrak{U}\}$$

dove $\mathfrak{U}$ è l'insieme delle ancore sulla T-mesh $\mathfrak{M}$.
Dal momento che i nodi interni hanno molteplicità non superiore a 3, varrà $S(\mathfrak{M}) \subset C^0(\hat{\Omega})$, dove $\hat{\Omega}$ è il dominio parametrico.
Se le funzioni di miscelamento $B^{(i,j)}(s,t)$, con $(i,j) \in \mathfrak{U}$, sono linearmente indipendenti, allora segue immediatamente che $\dim(S(\mathfrak{M})) = \#\mathfrak{U}$, dove $\#$ denota la cardinalità. Infatti, esse per costruzione sono un insieme di generatori, e nel caso in cui siano anche linearmente indipendenti, formano una base dello spazio di funzioni, la cui dimensione corrisponde al numero di funzioni che costituiscono questa base, ovviamente uguale al numero di

ancore, essendo funzioni ed ancore in biezione. Tuttavia questo è in generale falso, e anzi la determinazione dell'indipendenza lineare o meno di $S(\mathfrak{M})$ per T-mesh generiche è un problema ancora aperto, risolto soltanto per alcune tipologie di T-mesh, come vedremo nel capitolo 4.
Una volta definite le funzioni di miscelamento, possiamo definire le superfici T-spline nel seguente modo: sia $\mathfrak{M}$ una T-mesh. Una superficie T-spline in $\mathbb{R}^3$ è l'immagine $\tilde{\Omega}$ di una mappa $\tilde{\mathbf{F}} : \hat{\Omega} \to \tilde{\Omega}$, dove:

$$\tilde{\mathbf{F}} = \sum_{(i,j)\in\mathfrak{U}} \tilde{\mathbf{C}}^{(i,j)} B^{(i,j)}$$

I coefficienti $\tilde{\mathbf{C}}^{(i,j)} \in \mathbb{R}^3$ sono i *punti di controllo* per la superficie $\tilde{\Omega}$.
Siano ora (x_1, x_2, x_3) le coordinate dei punti in $\mathbb{R}^3$. Supponiamo che tutti i $\tilde{\mathbf{C}}^{(i,j)}$ stiano nel semispazio $x_3 \geq C$ per una data costante $C > 0$.
Consideriamo ora la trasformazione proiettiva sul piano $x_3 = 1$ definita da $(x_1, x_2, x_3) \mapsto (x_1/x_3, x_2/x_3, 1)$. Questa trasformazione proiettiva mappa la superficie $\tilde{\Omega}$ nella superficie piana Ω, identificata dal corrispondente dominio bidimensionale $\Omega \subset \mathbb{R}^2$, il dominio fisico di interesse. Allora Ω è parametrizzata dalla mappa $\mathbf{F} : \hat{\Omega} \to \Omega$ data da:

$$\mathbf{F} = \frac{\sum_{(i,j)\in\mathfrak{U}} w^{(i,j)} \mathbf{C}^{(i,j)} B^{(i,j)}}{\sum_{(\hat{i},\hat{j})\in\mathfrak{U}} w^{(\hat{i},\hat{j})} B^{(\hat{i},\hat{j})}}$$

dove $\mathbf{C}^{(i,j)} \in \mathbb{R}^2$ e $w^{(i,j)}$ sono i punti di controllo e i pesi della geometria `NURBS`, e sono associati ai punti di controllo $\tilde{\mathbf{C}}^{(i,j)} \in \mathbb{R}^3$, chiamati *punti di controllo proiettivi* della geometria `NURBS`, dalla relazione:

$$(w^{(i,j)} \mathbf{C}^{(i,j)}, w^{(i,j)}) = \tilde{\mathbf{C}}^{(i,j)}$$

Capitolo 3

Azioni sulle superfici T-spline

Sulle superfici T-spline possono essere compiute molteplici azioni. Andiamo a vederne alcune, relative alla possibilità di raffinarle inserendo nuovi punti di controllo, e alla conversione di esse in e da formati noti sviluppati precedentemente alle T-spline, in particolare B-spline e B-spline gerarchiche.

3.1 Inserimento di punti di controllo: un primo algoritmo

Per prima cosa, consideriamo il problema di inserire un nuovo punto di controllo all'interno di una già esistente T-mesh.
Se l'unico obiettivo dell'inserimento di punti è di permettere un controllo maggiore, è possibile semplicemente aggiungere nuovi punti, lasciando le coordinate degli altri invariate. Questa operazione permette di raggiungere lo scopo di aumentare la possibilità di controllo della superficie incrementandone la versatilità, ma allo stesso tempo porta ad una modifica della forma della T-spline, perlomeno alla parte di essa che si troverà ad essere influenzata dalla presenza dei nuovi punti di controllo.
Per evitare di incorrere in questo inconveniente, può essere più opportuno inserire i punti di controllo all'interno della T-mesh in maniera tale che la forma della T-spline non risulti modificata, attraverso una procedura nota come *inserzione locale di un nodo*, che permette di aggiungere un singolo punto di controllo all'interno della mesh, e che viene applicata tante volte quanti sono i nuovi punti di controllo.
Consideriamo l'esempio mostrato in figura 3.1, dove il punto di controllo $\mathbf{P'}_3$ è inserito nel lato $\mathbf{P}_2\mathbf{P}_4$ utilizzando gli intervalli tra nodi mostrati. Vale un'ulteriore regola:

- $\mathbf{P'}_3$ può essere inserito soltanto se $\mathbf{t}_1 = \mathbf{t}_2 = \mathbf{t}_4 = \mathbf{t}_5$; si veda la figura 3.1. Ricordiamo che $\mathbf{t}_i$ è il t-vettore dei nodi per le funzioni base B_i. Se il punto di controllo vuole essere inserito su di un lato verticale, allora a dover essere uguali sono i quattro vettori dei nodi s_i più vicini.

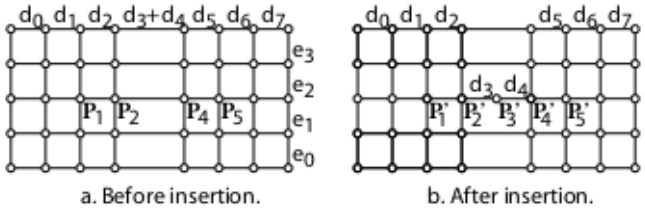


Figura 3.1: Inserzione locale di un nodo in una T-mesh.

L'inserzione locale di un nodo viene svolta inserendo il nodo in tutte le funzioni base il cui vettore dei nodi si trova ad essere alterato dalla presenza del nuovo punto di controllo. Nell'esempio di figura 3.1, esse sono B_1, B_2, B_4, B_5, ovvero le funzioni di base corrispondenti ai punti di controllo $\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_4, \mathbf{P}_5$. Utilizzando l'algebra delle forme polari, e tenendo conto dell'ultima regola introdotta, è immediato mostrare che:

$$\begin{aligned}
\mathbf{P}'_1 &= \mathbf{P}_1 \\
\mathbf{P}'_2 &= \frac{d_4\mathbf{P}_1 + (d_1+d_2+d_3)\mathbf{P}_2}{d_1+d_2+d_3+d_4} \\
\mathbf{P}'_3 &= \frac{(d_4+d_5)\mathbf{P}_2 + (d_2+d_3)\mathbf{P}_4}{d_2+d_3+d_4+d_5} \\
\mathbf{P}'_4 &= \frac{(d_4+d_5+d_6)\mathbf{P}_4 + d_3\mathbf{P}_5}{d_3+d_4+d_5+d_6} \\
\mathbf{P}'_5 &= \mathbf{P}_5
\end{aligned}$$

Per quest'ultima regola, inoltre, non è sempre possibile inserire immediatamente un punto di controllo su di un lato qualsiasi. Per esempio, nella figura 3.2.a, $\mathbf{A}$ non può essere inserito, poiché $\mathbf{t}_2$ non è uguale a $\mathbf{t}_1, \mathbf{t}_4, \mathbf{t}_5$, possedendo un nodo (e conseguentemente un punto di controllo) in più. Un modo per inserirlo è quello di rendere prima gli altri vettori di nodi uguali ad esso, aggiungendo anche in essi un punto di controllo in più, come mostrato in figura 3.2.b.

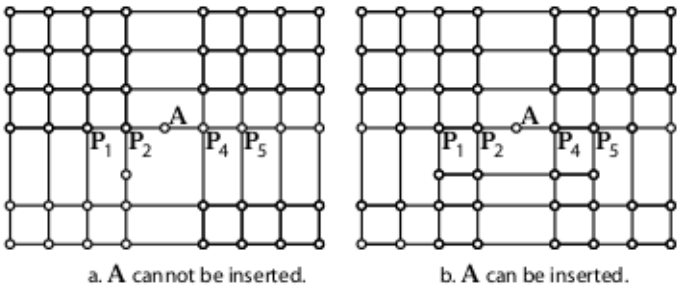


Figura 3.2: Operazione preliminare per poter inserire un nodo.

L'inserzione locale di un nodo può essere compiuta soltanto su di un lato

esistente. Per inserire un punto di controllo nel mezzo di una faccia, è necessario creare prima un lato attraverso essa.
Una delle utilità dell'operazione dell'inserimento locale è quella di rendere alcune caratteristiche particolari, come per esempio rendere appuntita una superficie. La figura 3.3 mostra una siffatta costruzione. Inserendo alcune righe adiacenti di punti di controllo con intervallo tra nodi pari a zero, e dal momento che due intervalli tra nodi adiacenti pari entrambi a zero introducono un nodo localmente triplo, la superficie diventa localmente continua soltanto di classe C^0 in corrispondenza di tale nodo (ricordiamo che stiamo lavorando con superfici cubiche), e pertanto permette la realizzazione di un effetto "punta", che può essere controllato lavorando sulla posizione dei nuovi punti di controllo.

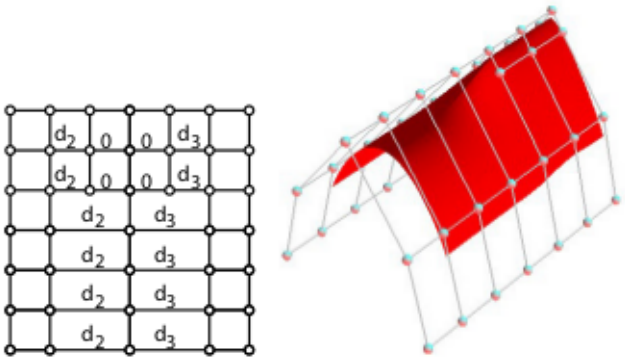


Figura 3.3: Una superficie resa appuntita tramite l'inserimento locale.

3.1.1 T-spline standard e non standard

Possiamo suddividere le T-spline in due categorie: quelle standard e quelle non standard. Definiamo una T-spline standard se:

$$\forall i \; w_i = 1 \Rightarrow \sum_{i=1}^{n} w_i B_i(s,t) \equiv \sum_{i=1}^{n} B_i(s,t) \equiv 1$$

L'insieme delle T-spline standard gode delle seguenti proprietà, utili anche in relazione ad un parallelo con le più semplici B-spline:

- una superficie B-spline tensore prodotto può essere vista come un caso particolare di T-spline standard, appartenendo a tale insieme;
- applicando l'inserzione dei nodi ad una T-spline standard, si ottiene ancora una T-spline standard;
- una T-spline formata fondendo due superfici B-spline, con una tecnica che discuteremo in seguito (§ 5.1), risulta essere una T-spline standard.

Una T-spline standard può essere scomposta in patch polinomali bicubiche. Una T-spline non standard può pure essere scomposta in patch bicubiche, ma che saranno razionali e non polinomiali.

3.1.2 Estrazione delle patch di Bézier

Può essere utile rappresentare le patch che compongono una T-spline in forma di Bézier, per esempio nel caso in cui si voglia applicare alla superficie un algoritmo di poligonizzazione della stessa (*tesselation*), come per esempio quello descritto in [7], che, tenendo conto della presenza delle T-giunzioni, possa permettere di ottenere una forma poligonizzata del nostro oggetto, che può servire come rappresentazione alternativa.
I domini delle patch di Bézier che comprendono una T-spline standard possono essere determinati estendendo tutte le T-giunzioni di due "unità", come mostrato in figura 3.4. I rettangoli in figura 3.4.b sono domini di Bézier. La ragione di ciò si capisce considerando i vettori dei nodi per le funzioni base di ogni punto di controllo.

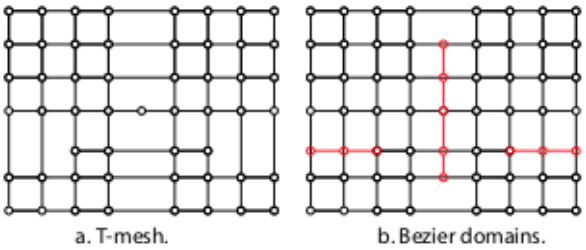
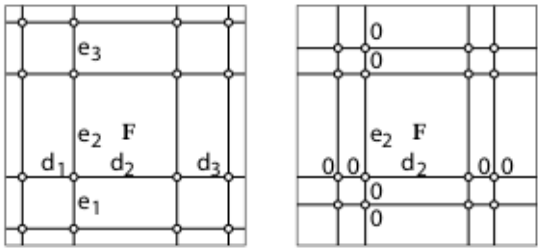


Figura 3.4: Domini di Bézier nella preimmagine di una T-mesh

I punti di controllo di Bézier possono essere ottenuti effettuando un'inserzione di nodi locale ripetuta, ricordando che una superficie B-spline può essere espressa in forma di Bézier utilizzando nodi multipli, e che un intervallo tra nodi pari a zero implica un nodo doppio. Per la configurazione di intervalli tra nodi in figura 3.5.b, la griglia 4×4 di punti di controllo che circondano F sono i punti di controllo di Bézier di tale patch. Pertanto, i punti di controllo di Bézier per la faccia F in figura 3.5.a possono essere determinati effettuando un procedimento di inserzione locale di nodi.

3.2 Inserimento di punti di controllo: un secondo algoritmo

Presentiamo ora un secondo algoritmo per il raffinamento locale di T-spline. Si affrontano di seguito: il problema del raffinamento delle funzioni di miscelamento; trasformazioni lineari negli spazi B-spline; l'algoritmo di raffinamento locale vero e proprio ([2]).



a. Face **F**. b. Bézier control points.

Figura 3.5: Determinazione dei punti di conrollo di Bézier utilizzando l'inserzione locale di nodi.

3.2.1 Raffinamento delle funzioni di miscelamento

Se $\mathbf{s} = [s_0, s_1, s_2, s_3, s_4]$ è un vettore di nodi, e $\tilde{\mathbf{s}}$ è un vettore di nodi con m nodi, in maniera tale che $\mathbf{s}$ si trovi ad essere una sottosuccessione di $\tilde{\mathbf{s}}$, allora $N[s_0, s_1, s_2, s_3, s_4](s)$ può essere scritta come combinazione lineare delle $m-4$ funzioni base B-spline definite sulle sottostringhe di lunghezza 5 in $\tilde{\mathbf{s}}$. Vediamo ora le equazioni di raffinamento delle funzioni di base per il caso $m = 6$. Le equazioni per $m > 6$ possono essere ricavate tramite applicazione ripetuta di queste.
Se $\mathbf{s} = [s_0, s_1, s_2, s_3, s_4], N(s) = N[s_0, s_1, s_2, s_3, s_4](s)$, e $\tilde{\mathbf{s}} = [s_0, k, s_1, s_2, s_3, s_4]$, allora:

$$N(s) = c_0 N[s_0, k, s_1, s_2, s_3](s) + d_0 N[k, s_1, s_2, s_3, s_4](s)$$

dove $c_0 = \frac{k-s_0}{s_3-s_0}$ e $d_0 = 1$. Se $\tilde{\mathbf{s}} = [s_0, s_1, k, s_2, s_3, s_4]$,

$$N(s) = c_1 N[s_0, s_1, k, s_2, s_3](s) + d_1 N[s_1, k, s_2, s_3, s_4](s)$$

dove $c_1 = \frac{k-s_0}{s_3-s_0}$ e $d_1 = \frac{s_4-k}{s_4-s_1}$. Se $\tilde{\mathbf{s}} = [s_0, s_1, s_2, k, s_3, s_4]$,

$$N(s) = c_2 N[s_0, s_1, s_2, k, s_3](s) + d_2 N[s_1, s_2, k, s_3, s_4](s)$$

dove $c_2 = \frac{k-s_0}{s_3-s_0}$ e $d_2 = \frac{s_4-k}{s_4-s_1}$. Se $\tilde{\mathbf{s}} = [s_0, s_1, s_2, s_3, k, s_4]$,

$$N(s) = c_3 N[s_0, s_1, s_2, s_3, k](s) + d_3 N[s_1, s_2, s_3, k, s_4](s)$$

dove $c_3 = 1$ e $d_3 = \frac{s_4-k}{s_4-s_1}$ Se $k \leq s_0$ o $k \geq s_4$, $N(s)$ non cambia.
Su di una funzione T-spline $B(s,t)$ può essere effettuata l'inserzione di un nodo sia in s che in t, dividendola in due funzioni di miscelamento la cui somma corrisponde a quella iniziale. Inserzioni ulteriori portano ad un numero maggiore di funzioni di miscelamento, sempre a somma pari a quella di partenza. La figura 3.6.a mostra il vettore dei nodi per una funzione di miscelamento T-spline B_1, mentre la figura 3.6.b mostra un raffinamento

dei vettori dei nodi di figura 3.6.a. Applicando opportuamente le equazioni scritte in precedenza, possiamo ottenere:

$$B_1(s,t) = c_1^1\tilde{B}_1(s,t) + c_1^2\tilde{B}_2(s,t) + c_1^3\tilde{B}_3(s,t) + c_1^4\tilde{B}_4(s,t)$$

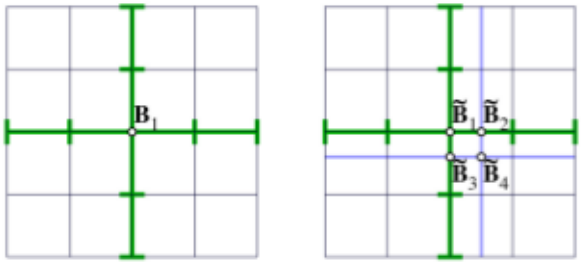


Figura 3.6: Raffinamento di $B_1(s,t)$.

3.2.2 Trasformazioni lineari negli spazi T-spline

Abbiamo già definito gli spazi T-spline in § 2.2. Ciò che andiamo a fare ora è di caratterizzare alcune trasformazioni lineari in essi.
Data una T-spline $\mathbf{P}(s,t) \in S_1$, denotiamo con $\mathbf{P}$ il vettore colonna dei punti di controllo per $\mathbf{P}(s,t)$; consideriamo anche una seconda T-spline $\tilde{\mathbf{P}}(s,t) \in S_2$, tale per cui $\mathbf{P}(s,t) \equiv \tilde{\mathbf{P}}(s,t)$, e denotiamo con $\tilde{\mathbf{P}}$ il vettore colonna dei punti di controllo per $\tilde{\mathbf{P}}(s,t)$. Allora esiste una trasformazione lineare che mappa $\mathbf{P}$ in $\tilde{\mathbf{P}}$, e che possiamo denotare come:

$$M_{1,2}\mathbf{P} = \tilde{\mathbf{P}}$$

La matrice $M_{1,2}$ si ottiene nel modo seguente: innanzittutto $\mathbf{P}(s,t)$ e $\tilde{\mathbf{P}}(s,t)$ si scrivono come:

$$\mathbf{P}(s,t) = \sum_{i=1}^{n} \mathbf{P}_i B_i(s,t) \qquad\qquad \tilde{\mathbf{P}}(s,t) = \sum_{j=1}^{\tilde{n}} \tilde{\mathbf{P}}_j \tilde{B}_j(s,t)$$

Dal momento che $S_1 \subset S_2$, ogni $B_i(s,t)$ può essere scritta come combinazione lineare delle $\tilde{B}_j(s,t)$:

$$B_i(s,t) = \sum_{j=1}^{\tilde{n}} c_i^j \tilde{B}_j(s,t)$$

La richiesta che $\mathbf{P}(s,t) \equiv \tilde{\mathbf{P}}(s,t)$ è soddisfatta se:

$$\tilde{\mathbf{P}}_j = \sum_{i=1}^{n} c_i^j \mathbf{P}_i$$

Pertanto, l'elemento di riga j e colonna i di $M_{1,2}$ è c_i^j. In questo modo, è possibile trovare matrici di trasformazione generiche $M_{i,j}$ che mappano ogni T-spline di S_i in un'equivalente T-spline di S_j, nell'ipotesi che $S_i \subset S_j$.
La definizione di un sottospazio T-spline $S_i \subset S_j$ può richiedere un'ulteriore osservazione. Se $S_i \subset S_j$, allora i punti di controllo della preimmagine di S_i sono anche punti di controllo della preimmagine di S_j. Il viceversa non è sempre vero: potrebbe infatti accadere che non sia possibile passare da un insieme all'altro, dal momento che non è detto che si possa semplicemente aggiungere un punto di controllo ad una T-mesh, senza doverne aggiungere anche altri (cfr. § 3.1). L'algoritmo che vedremo ora ci permetterà, tra le altre cose, di calcolare sovraspazi validi di un dato spazio T-spline.

3.2.3 Algoritmo di raffinamento locale

Abbiamo già visto nel § 3.1 che il raffinamento locale di una T-spline consiste nell'inserimento di uno o più punti di controllo all'interno di una T-mesh senza cambiare la forma della superficie T-spline. Questa procedura viene anche chiamata inserzione locale di nodi, dal momento che l'aggiunta di punti di controllo in una T-mesh deve essere accompagnata dall'inserimento di nodi nelle funzioni di miscelamento che interrvengono.
L'algoritmo di raffinamento che vedremo ora si compone di due fasi: una fase topologica e una geometrica. Quella topologica identifica quali punti di controllo (eventualmente nessuno) devono essere inseriti oltre a quelli richiesti; dopo che tutti i punti di controllo nuovi richiesti sono stati identificati, le coordinate cartesiane e i pesi per la T-mesh raffinata sono calcolati utilizzando le trasformazioni lineari viste nel § 3.2.2.
Rispetto all'algoritmo visto in § 3.1, si ha la garanzia di funzionamento in ogni situazione, e la richiesta, di solito, di un numero significativamente minore di punti di controllo da inserire oltre a quelli desiderati.
Soffermiamoci ora sulla fase topologica dell'algoritmo.
Per comprendere al meglio quanto segue è importante ricordare come, in una T-spline, le funzioni di miscelamento e la T-mesh sono strettamente legate: ad ogni punto di controllo corrisponde una funzione di miscelamento, e i vettori dei nodi di ogni funzione di miscelamento sono definiti dalla regola vista nel § 2.1, che riportiamo nuovamente:

> *I vettori dei nodi* $\mathbf{s}_i$ *e* $\mathbf{t}_i$ *per la funzione di miscelamento associata a* $\mathbf{P}_i$ *sono determinati come segue.* (s_{i2}, t_{i2}) *sono le coordinate nodali di* P_i*. Consideriamo una linea nello spazio dei parametri:* $\mathbf{R}(\alpha) = (s_{i2} + \alpha, t_{i2})$*. Allora* s_{i3} *e* s_{i4} *sono le coordinate s dei primi due s-lati intersecati dalla linea, escludendo* (s_{i2}, t_{i2})*; ricordiamo che come s-lato intendiamo un segmento verticale in cui s è costante. Gli altri nodi in* $\mathbf{s}_i$ *e* $\mathbf{t}_i$ *sono determinati in maniera simile.*

Nella nostra discussione, scorporeremo temporaneamente le funzioni di miscelamento dalla T-mesh. Questo significa che durante lo svolgimento dell'algoritmo, permetteremo temporaneamente l'esistenza di funzioni di miscelamento che violano la regola cui sopra, e punti di controllo ai quali non è associata alcuna funzione di miscelamento.
La nostra discussione distingue tre possibili violazioni in cui si può incorrere durante lo svolgimento dell'algoritmo di raffinamento:

- **1** Una funzione di miscelamento è priva di un nodo richiesto dalla regola per la T-mesh corrente.
- **2** Una funzione di miscelamento possiede un nodo non compatibile con la regola per la T-mesh corrente.
- **3** Un punto di controllo non ha associata ad esso alcuna funzione di miscelamento.

Se non si verifica alcuna violazione, allora la T-spline è valida. Se si verificano delle violazioni, l'algoritmo le risolve una ad una, in modo tale da finire con il trovare sempre un sovraspazio valido.
La fase topologica del nostro algoritmo di raffinamento locale consiste dunque dei seguenti passi:

1. si inseriscono tutti i punti di controllo desiderati all'interno della T-mesh;
2. se una funzione di miscelamento viola il punto 1, allora si inserisce un opportuno nodo;
3. se una funzione di miscelamento viola il punto 2, allora si inserisce un appropriato punto di controllo all'interno della T-mesh;
4. si ripetono gli ultimi due passi fino a portarsi ad una situazione in cui non sono più presenti violazioni; questa si raggiunge con certezza in quanto la risoluzione di tutti i casi di violazione dei punti 1 e 2 implica la risoluzione di tutti i casi di violazione del punto 3.

Mostriamo ora un esempio. La figura 3.7.a mostra una T-mesh iniziale all'interno della quale vogliamo inserire un punto di controllo, $\mathbf{P}_2$. Poiché essa nella sua situazione iniziale è valida, non sono presenti violazioni. Ma se ci limitiamo ad inserire quel punto, come mostrato in figura 3.7.b, senza cambiare alcuna delle funzioni di miscelamento, arriviamo ad una situazione in cui si presentano diverse violazioni. Poiché $\mathbf{P}_2$ ha coordinate nodali (s_3, t_2), quattro funzioni di miscelamento si trovano a violare il punto 1: quelle centrate in $(s_1, t_2), (s_2, t_2), (s_4, t_2)$, e (s_5, t_2). Per eliminare queste violazioni, dobbiamo inserire un nodo in s_3 per ognuna di queste funzioni di miscelamento. La funzione di miscelamento centrata in (s_2, t_2) è

$N[s_0, s_1, s_2, s_4, s_5](s)N[t_0, t_1, t_2, t_3, t_4](t)$. L'inserimento di un nodo $s = s_3$ nel vettore s dei nodi di questa funzione di miscelamento ne comporta la scomposizione nelle due funzioni:

- $c_2N[s_0, s_1, s_2, s_3, s_4](s)N[t_0, t_1, t_2, t_3, t_4](t)$ (figura 3.7.c)
- $d_2N[s_1, s_2, s_3, s_4, s_5](s)N[t_0, t_1, t_2, t_3, t_4](t)$ (figura 3.7.d)

La funzione di miscelamento $c_2N[s_0, s_1, s_2, s_3, s_4](s)N[t_0, t_1, t_2, t_3, t_4](t)$ soddisfa la regola, così come i raffinamenti delle funzioni di miscelamento centrate in $(s_1, t_2), (s_4, t_2), (s_5, t_2)$. Tuttavia, il vettore t dei nodi della funzione di miscelamento $d_2N[s_1, s_2, s_3, s_4, s_5](s)N[t_0, t_1, t_2, t_3, t_4](t)$ viola il punto 2, poiché il vettore t della funzione di miscelamento si trova ad essere $[t_0, t_1, t_2, t_3, t_4]$, ma la regola non vuole un nodo in t_3. Questo problema non può essere risolto raffinando questa funzione di miscelamento, rendendo necessaria l'aggiunta di un nuovo punto di controllo dentro la T-mesh.
Il punto di controllo richiesto è P_3 in figura 3.7.e; l'inserimento di questo punto risolve il caso di violazione del punto 2, ma comporta un nuovo caso di violazione del punto 1, in quanto, come mostrato in figura 3.7.f, la funzione di miscelamento centrata in (s_2, t_3) si trova ad avere un vettore s che non include s_3, come richiesto dalla regola. Risulta pertanto necessario inserire anche tale nodo in tale vettore, per non avere più violazioni.
Questo algoritmo termina con certezza in un numero finito di passi, in quanto gli unici raffinamenti di funzioni di miscelamento e le uniche inserzioni di punti di controllo devono coinvolgere valori dei nodi che già esistono nella T-mesh, o che vengono aggiunte con il primo passo dell'algoritmo. Nel caso peggiore, ci si trova ad avere tutte le righe parziali di punti di controllo estese in maniera da coprire l'intera superficie; in pratica, tuttavia, capita solitamente che i punti in più da aggiungere siano pochi, se non nessuno.

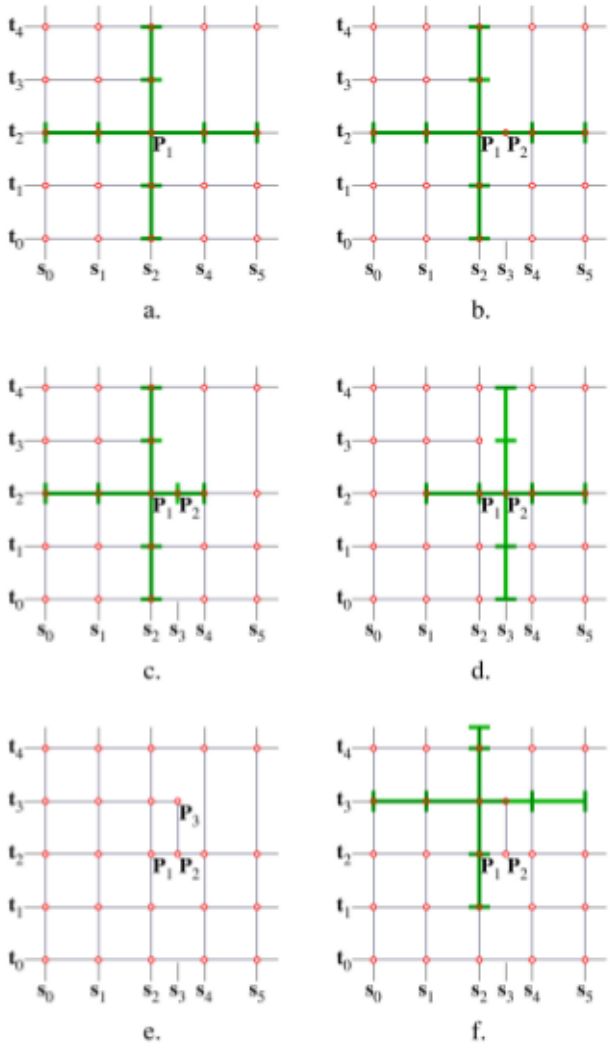


Figura 3.7: Esempio di raffinamento locale.

3.3 Conversione di una T-spline in una superficie B-spline

L'algoritmo di raffinamento visto nel § 3.2 permette, tra le altre cose, di convertire con facilità una T-spline di S_1 in una superficie B-spline di S_n equivalente: per fare ciò, è sufficiente utilizzare come matrice di trasformazione (§ 3.2.2) $M_{1,n}$.
Abbiamo già definito nel § 3.1.1 che cosa si intende per T-spline standard; nel contesto delle matrici di trasformazione, si può vedere che una condizione necessaria e sufficiente affinché una T-spline sia standard è che la somma di ogni riga di $M_{1,n}$ sia pari ad 1.
L'algoritmo di inserzione può portare ad un risultato interessante: può venirsi a creare una T-spline tale per cui non siano tutti i w_i pari ad 1, ma che verifica $\sum_{i=1}^{n} w_i B_i(s,t) \equiv 1$. Chiamiamo queste T-spline come *semi-standard*; la figura 3.8 ne mostra un paio di esempi, con i relativi intervalli tra nodi.

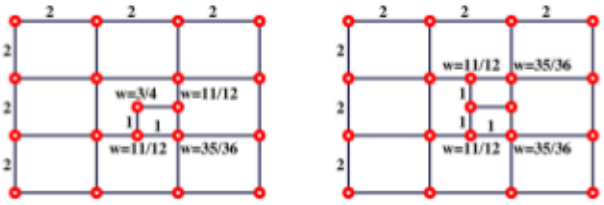


Figura 3.8: T-spline semi-standard.

Per verificare che queste T-spline sono effettivamente semi-standard, è sufficiente convertirle in superfici B-spline, e riscontrare che i pesi dei punti di controllo sono tutti unitari.
La nozione di spazio T-spline permette di giungere a definizioni più precise di T-spline semi-standard e non standard, potendole fornire come:

- Uno spazio T-spline S è *semi-standard* se, pur non essendo $w_i = 1 \; \forall i$, esiste almeno un elemento di S tale per cui $\sum_{i=1}^{n} w_i B_i(s,t) \equiv 1$.
- Uno spazio T-spline è *non standard* se non esiste alcun elemento in esso tale per cui $\sum_{i=1}^{n} w_i B_i(s,t) \equiv 1$.

Queste definizioni sono migliori in quanto permettono di verificare se una T-spline razionale, ovvero tale per cui i pesi non sono tutti uguali ad 1, è standard, semi-standard, o non standard; la distinzione viene effettuata in base al tipo di spazio T-spline cui appartiene.

3.4 Conversione di una B-spline gerarchica in una T-spline

Abbiamo visto nel § 1.2 cosa sono le B-spline gerarchiche e quale sia il ruolo che ricoprono nello *stato dell'arte*, in termini di fasi di sviluppo dagli strumenti classici B-spline e `NURBS` a giungere agli strumenti recenti quali le T-spline. Vediamo ora come sia possibile passare da questo formato al più recente e viceversa, andando cioè a studiare i termini di convertibilità tra il formato B-spline gerarchiche e quello T-spline.
Data una superficie B-spline razionale gerarchica, è possibile appiattire la struttura gerarchica creando una superficie B-spline razionale classica (non gerarchica); essa si troverà a contenere un certo numero di punti di controllo superflui che non avranno altro scopo oltre a quello di soddisfare requisiti topologici della formulazione tensore prodotto delle superfici B-spline.
La trasformazione di una superficie B-spline gerarchica in una superficie T-spline, consiste nel combinare tutti i sovrastrati in una superficie a strato singolo, in maniera tale da mantenere la forma della superficie risultante identica a quella dell'originale, ma con soltanto un numero ridotto di punti superflui in più.
L'approccio seguito per realizzare una tale trasformazione usa un metodo ricorsivo per generare la superficie T-spline; si comincia con la superficie B-spline di livello 0, e si aggiungono tutti i sovrastrati di primo livello per formare una superficie T-spline, in seguito si aggiungono tutti i sovrastrati di secondo livello all'interno della T-spline formata in maniera tale da crearne una più elaborata, e il procedimento viene ripetuto fino a quando non ci sono più sovrastrati da aggiungere. Sostanzialmente si tratta dunque di definire un algoritmo che permetta di aggiungere un sovrastrato all'interno di una superficie T-spline.
Supponiamo di avere una superficie T-spline $P(s,t)$ e un sovrastrato $V(s,t)$. La superficie T-spline avrà equazione omogenea:

$$P(s,t) = \sum_{i=1}^{n} P_i B_i(s,t)$$

e assumiamo che sia il risultato dell'inserimento di tutti i sovrastrati di livello inferiore a $V(s,t)$ e di un certo numero di sovrastrati dello stesso livello di $V(s,t)$ sulla superficie B-spline di livello 0. Il sovrastrato $V(s,t)$ è una superficie B-spline razionale bicubica, di equazione in forma omogenea:

$$V(s,t) = \sum_{i=l-1}^{r+1} \sum_{j=b-1}^{t+1} V_{ij} N_i(s) N_j(t)$$

dove i V_{ij} sono i punti di controllo (in coordinate omogenee), $N_i(s)$ e $N_j(t)$ sono funzioni base B-spline cubiche definite sulle successioni di nodi

$\{s_{l-2}, s_{l-1}, s_l, \ldots, s_r, s_{r+1}, s_{r+2}\}$ e $\{t_{b-2}, t_{b-1}, t_b, \ldots, t_t, t_{t+1}, t_{t+2}\}$. L'intervallo dei parametri del sovrastrato $[s_l, s_r] \times [t_b, t_t]$. Dalla formulazione della B-spline gerarchica, risulta che s_l, s_r, t_b, t_t sono anche nodi della superficie genitore di $V(s,t)$, e pertanto corrispondono a determinati lati nella T-mesh di $P(s,t)$.
Considerando la scomposizione additiva in reference e offset, ci è necessario memorizzare soltanto l'offset, dal momento che ogni punto V_{ij} del sovrastrato è la somma del punto reference R_{ij} e del vettore di offset O_{ij}, con i punti reference che sono tutti già determinati dalla superficie genitore; essi formano in particolare una superficie:

$$R(s,t) = \sum_{i=l-1}^{r+1} \sum_{j=b-1}^{t+1} R_{ij} N_i(s) N_j(t)$$

che è una rirappresentazione esatta della superficie genitore, o della superficie T-spline $P(s,t)$ nel dominio $[s_l, s_r] \times [t_b, t_t]$. Se utilizziamo tutti i vettori di offset per definire a sua volta una superficie:

$$O(s,t) = \sum_{i=l-1}^{r+1} \sum_{j=b-1}^{t+1} O_{ij} N_i(s) N_j(t)$$

allora $V(s,t) = R(s,t) + O(s,t)$. Osservando inoltre che se $i < l+2, i > r-2, j < b+2, j > t-2$, allora $O_{ij} = 0$, il sovrastrato $V(s,t)$ può essere scritto come:

$$V(s,t) = \sum_{i=1}^{n} P_i B_i(s,t) + \sum_{i=l+2}^{r-2} \sum_{j=b+2}^{t-2} O_{ij} N_i(s) N_j(t)$$

Se consideriamo O_{ij} come un punto di controllo associato ai nodi (s_i, t_j), allora questa equazione fornisce una descrizione del sovrastrato. Tuttavia, senza ulteriori accorgimenti, non possiamo asserire che si tratta di una T-spline, in quanto non abbiamo l'assicurazione che quanto compiuto faccia mantenere la struttura di T-mesh necessaria affinché si possa trattare effettivamente di una T-spline; di per sé, possiamo soltanto affermare di avere dato una descrizione, ma in termini di una più debole PB-spline, non in termini di T-spline.
Per avere la garanzia che questa PB-spline sia effettivamente una T-spline, dobbiamo controllare se è necessario aggiungere nodi nelle funzioni di base o nuovi punti di controllo nella mesh, ed eventualmente eseguire tali operazioni. Alla luce di ciò, un algoritmo per inserire un sovrastrato all'interno di una T-spline può essere posto nella seguente forma:

1. Si inseriscono i punti $O_{ij} (i = l+2, \ldots, r-2; j = b+2, \ldots, t-2)$ con le loro rispettive funzioni base all'interno della T-mesh, posizionandoli in base ai loro nodi associati.

2. Se una funzione base possiede un nodo che non è previsto dalla T-mesh, si aggiunga un punto appropriato all'interno della stessa, in modo tale da fare sì che il nodo si trovi ad essere previsto dalla T-mesh modificata.

3. Se una funzione base è priva di un nodo richiesto dalla T-mesh, si esegua l'inserzione necessaria all'interno di tale funzione di base, in maniera tale che essa si trovi a possedere il nodo richiesto.

4. Si ripetano i passaggi 2. e 3. fino a quando non ci sono più operazioni di questi tipi da compiere.

5. Se un punto possiede soltanto un lato ad esso incidente, si prolunghi il lato fino a raggiungere il primo punto di intersezione tra lato e T-mesh nello spazio della preimmagine.

6. Si inserisca il punto di intersezione e si torni al passo 2.

I passi 5. e 6. vanno effettuati per evitare la presenza di lati penzolanti.

La figura 3.9.a mostra una superficie B-spline descritta da una mesh di controllo 6×6 (in linea tratteggiata) e un sovrastrato descritto da una mesh 7×7 raffinata (in linea continua). Per prima cosa inseriamo l'unico punto di offset diverso da zero della mesh 7×7, ovvero il punto centrale, all'interno della mesh B-spline 6×6, generando la mesh di figura 3.9.b. Tuttavia, in questo modo le funzioni di base corrispondenti a questo punto di offset diverso da zero possiedono nodi che non sono indicati nella mesh, rendendo necessario l'inserimento di quattro punti di controllo intorno ad esso (figura 3.9.c). Ora, ogni funzione di base si associa propriamente ad un punto di controllo nella mesh, ma sono presenti quattro lati penzolanti, per la cui sistemazione è necessario compiere i passi 5. e 6, che portano all'inserimento di altri 5 punti e alla creazione della T-mesh finale, mostrata in figura 3.9.d. Si osserva che tale T-mesh è piuttosto compatta.

3.5 Conversione di una T-spline in una B-spline gerarchica

In confronto alla trasformazione da superficie B-spline gerarchica a superficie T-spline, la trasformazione inversa che converte una superficie T-spline in una superficie B-spline gerarchica (razionale), è meno intuitiva, in quanto si richiede di creare una gerarchia da una T-mesh, che come tale, non la ammette (essendo di fatto a singolo strato).
L'obiettivo di base è quello di generare una successione di superfici B-spline, in maniera tale che esse siano organizzate in maniera gerarchica, e che topologicamente ogni B-spline sia il raffinamento di un'altra B-spline della successione, fatta naturalmente eccezione per una e una sola di esse, che

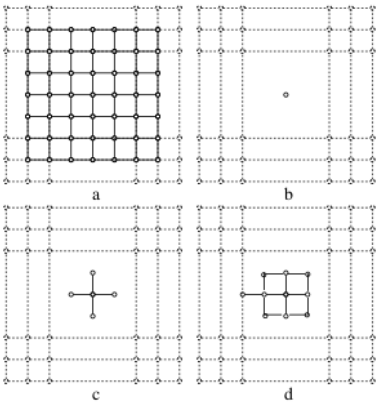


Figura 3.9: Esempio di conversione da B-spline gerarchica a T-spline.

prende il ruolo di superficie base.
Il processo per questo obiettivo coinvolge due aspetti: un aspetto topologico e uno geometrico. Il primo identifica la topologia della superficie B-spline di base e i sovrastrati ad ongi livello; il secondo calcola le coordinate dei punti di controllo per tutte le superfici. Un algoritmo si può riassumere fondamentalmente nei seguenti passi:

0. Si preprocessa la T-mesh data.
1. Si pone $k = 0$, $MeshCorrente =$ la T-mesh preprocessata.
2. Si estrae una superficie B-spline al livello k dalla $MeshCorrente$.
3. Si determinano le superfici di offset per il livello $lev = k + 1$.
4. Per la mesh di ogni superficie di offset al livello lev, si eseguono i seguenti passi:
   - si processa la mesh;
   - se la mesh processata non è una mesh B-spline, allora si pone $MeshCorrente =$ la mesh processata;
   - si pone $k = lev$;
   - si ritorna al passo 2.

Il passo di preprocessing 0. consiste nell'utilizzare l'inserzione di nodi locale T-spline in maniera tale da fare in modo che i tre anelli esterni della T-mesh non contengano alcun punto di T-giunzione. La figura 3.10 mostra, a sinistra, una data T-mesh, e a destra la stessa mesh dopo aver eseguito su di essa il passo di preprocessing.

Il passo 1. è una mera inizializzazione e su di esso non ci soffermeremo; i passi successivi invece sono notevolmente più articolati e rendono opportuna una spiegazione nel dettaglio.

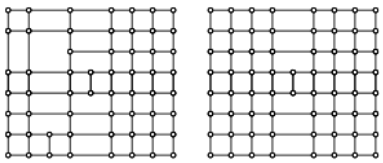


Figura 3.10: Preprocessing di una data T-mesh.

3.5.1 Estrazione di B-spline

Mentre una T-mesh permette T-giunzioni, ovvero le linee della T-mesh non necessitano di attraversare l'intera griglia di controllo, la mesh di una B-spline tensore prodotto deve essere topologicamente una griglia rettangolare. Pertanto, per estrarre una mesh B-spline da una T-mesh, si richiede di eliminare tutte le righe e le colonne parziali della T-mesh. Ciò che rimane dopo la rimozione di righe e colonne parziali è una mesh $m \times n$, che può essere utilizzata come specificazione topologica della nostra mesh di controllo B-spline.
Una volta che la topologia della mesh B-spline è specificata, ne resta da determinare la geometria. Possono esistere diverse soluzioni per la geometria delle B-spline, ne presentiamo una. Le formule per i punti di controllo B-spline sono derivate basandosi sulla forma polare delle B-spline; requisito è quello di rendere il vettore di offset nei punti dei tre anelli più esterni dei livelli più alti pari a zero.
Per ogni punto di controllo P nella mesh B-spline estratta (vedi figura 3.11a), ne calcoliamo la sua etichetta polare PL_P, denotandola come $PL_P = (s_{-1}, s_0, s_1) \times (t_{-1}, t_0, t_1)$. Il punto P corrisponde ad un punto di controllo, chiamiamolo Q nella T-mesh. Calcoliamo anche l'etichetta polare di Q, PL_Q, nella T-mesh. Se $PL_P = PL_Q$, allora copiamo semplicemente le coordinate di Q su quelle di P. Altrimenti, creiamo una T-mesh temporanea duplicando la T-mesh esistente e utilizzando l'algoritmo di inserzione locale di nodi T-spline per inserire punti nella T-mesh temporanea, formando una griglia 5×5 centrata intorno a Q. Questa griglia 5×5 è parte della mesh di controllo più fine della superficie B-spline.
Denotiamo i punti come $Q_{ij}, (i, j = -2, \ldots, 2)$, con Q_{00} corrispondente a Q; si faccia riferimento alla figura 3.11b) per le etichette. Allora P può essere calcolato come:

$$P = \sum_{i=-1}^{1} \sum_{j=-1}^{1} c_i d_j Q_{ij}$$

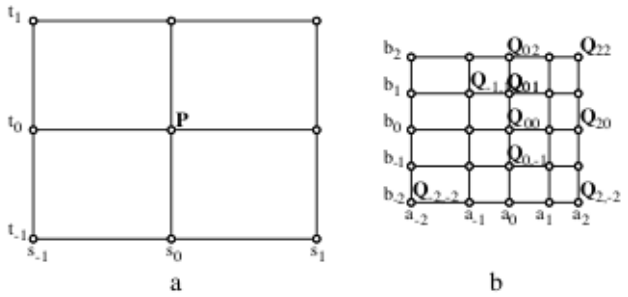


Figura 3.11: Calcolo di un punto di controllo B-spline P a partire da Q_{ij}.

dove:

$$\begin{aligned} c_{-1} &= \frac{(a_1 - s_{-1})(a_1 - s_1)}{(a_1 - a_{-2})(a_1 - a_{-1})} \\ c_1 &= \frac{(a_{-1} - s_{-1})(a_{-1} - s_1)}{(a_2 - a_{-1})(a_1 - a_{-1})} \\ c_0 &= 1 - c_{-1} - c_1 \\ d_{-1} &= \frac{(b_1 - t_{-1})(b_1 - t_1)}{(b_1 - b_{-2})(b_1 - b_{-1})} \\ d_1 &= \frac{(b_{-1} - t_{-1})(b_{-1} - t_1)}{(b_2 - b_{-1})(b_1 - b_{-1})} \\ d_0 &= 1 - d_{-1} - d_1 \end{aligned}$$

3.5.2 Determinazione degli offset di più alto livello

Una volta estratta una mesh B-spline da una T-mesh al livello k, la mesh B-spline e la T-mesh non definiscono necessariamente la stessa superficie. In questo caso, è necessario costruire delle superfici di offset di più alto livello per compensare la differenza.
Per prima cosa, identifichiamo la regione per queste superfici di offset di più alto livello. Per ogni punto P nella preimmagine della T-mesh al livello k nello spazio dei parametri (s, t), definiamo quattro quantità $s_{min}, t_{min}, s_{max}, t_{max}$ in maniera tale che s_{min} sia la coordinata s di un nuovo punto che è ottenuto spostando il punto P a sinistra di due posizioni, s_{max} sia la coordinata s di un nuovo punto che è ottenuto spostando il punto P a destra di due posizioni, t_{min} sia la coordinata t di un nuovo punto che è ottenuto spostando il punto P in basso di due posizioni, t_{max} sia la coordinata t di un nuovo punto che è ottenuto spostando il punto P in alto di due posizioni.
Per ogni riga o colonna parziale nella T-mesh, definiamo il suo *box-2*, denotandolo $O(s_l, t_b, s_r, t_u)$, come il rettangolo identificato dal suo angolo in basso a sinistra (s_l, t_b) e dal suo angolo in alto a destra (s_r, t_u), dove s_l e t_d sono i minimi di s_{min} e t_{min} di tutti i vertici che stanno sulla riga o colonna

parziale, mentre s_r e t_u sono i massimi di s_{max} e t_{max} di tutti i vertici che stanno sulla riga o colonna parziale.
Una volta calcolati tutti i box-2, andiamo a vedere se si sovrappongono. Qualora ciò accada, ovvero esistano due box-2 non disgiunti, li uniamo, in maniera tale da formare un più grande box a delimitarli, che consideriamo anche come box-2; ripetendo la cosa per ogni coppia, arriveremo ad un punto in cui ogni box-2 sta per una regione di un sovrastrato, senza più sovrapposizioni.
Una volta che la regione di una superficie di offest nello strato successivo è stata identificata, ne restano da determinare topologia e geometria. Estendiamo il box-2 aggiungendo un ulteriore anello; considerando che la preimmagine corrisponde biunivocamente alla T-mesh, prendiamo la porzione di T-mesh al livello k che corrisponde al box-2 esteso, formando con essa una T-mesh temporanea di livello $k+1$ per la superficie all'interno della regione identificata. Tramite la costruzione del box-2, può essere verificato che la T-mesh di livello $k+1$ possiede la caratteristica che i tre anelli più esterni della mesh non contengono T-giunzioni.
La superficie definita dalla T-mesh al livello k nella regione è descritta da:

$$P^{(k)}(s,t) = \sum_{i=1}^{n} P_i^{(k)} B_i(s,t)$$

dove i $P^{(k)}$ sono i punti di controllo nella T-mesh al livello k o nella T-mesh temporanea al livello $k+1$. Al fine di ottenere la superficie di offet, restringiamo la superficie B-spline estratta (dalla T-mesh al livello k) all'interno della regione, scrivendo la sua equazione come:

$$R^{(k)}(s,t) = \sum_{i=l-1}^{r+1} \sum_{j=b-1}^{t+1} R_{ij}^{k} N_i(s) N_j(t)$$

Allora la superficie di offset è $P^{(k)}(s,t) - R^{(k)}(s,t)$. Ciò fornisce il seguente algoritmo per determinare la topologia e la geometria della superficie di offset:

1. Si inseriscono i punti $-R_{ij}^{(k)} (i = l-1, \ldots, r+1; j = b-1, \ldots, t+1)$ con le loro rispettive funzioni di base all'interno della T-mesh temporanea, posizionandoli in base ai nodi associati di $R^{(k)_{ij}}$.

2. Se una funzione base è priva di un nodo richiesto dalla T-mesh, si esegua l'inserzione necessaria all'interno di tale funzione di base, in maniera tale che essa si trovi a possedere il nodo richiesto.

3. Se una funzione base possiede un nodo che non è previsto dalla T-mesh, si aggiunga un punto appropriato all'interno della stessa, in modo tale da fare sì che il nodo si trovi ad essere previsto dalla T-mesh modificata.

4. Si ripetano i passaggi 2. e 3. fino a quando non ci sono più operazioni di questi tipi da compiere.

5. La mesh in output è la T-mesh richiesta al livello $k+1$ che definisce la superficie di offset.

3.5.3 Processing di uno strato

Una volta ottenuta la T-mesh al livello $k+1$ per la superficie di offset, si verifica se essa consiste in una mesh B-spline. In caso affermativo, essa può diventare una componente finale (nodo foglia) della rappresentazione B-spline gerarchica, senza richiedere ulteriore processing; in caso negativo, è necessario lavorarci sopra ancora un po'. Possiamo estendere tutte le righe e le colonne parziali in maniera tale da creare una B-spline, oppure estrarre ulteriormente strati per generare livelli più alti.
Per bilanciare opportunamente la profondità della gerarchia con la dimensione di ogni offset, può essere opportuno definire una quantità, per ogni riga o colonna parziale nella T-mesh al livello $k+1$, chiamata *deficitarietà*, che misura quanti nodi nella B-spline sottostante deve attraversare questa riga o colonna parziale per raggiungere il bordo.
Se la deficitarietà è grande, allora la riga o la colonna richiede un percorso lungo per raggiungere il bordo, il che può significare che porta con sé dettagli rilevanti e pertanto è opportuno inserirla nel sovrastrato dello strato corrente. Per questa ragione, calcoliamo la media di tutte le deficitarietà di tutte le righe e colonne parziali. Se una riga o colonna parziale possiede deficitarietà minore di quella media aumentata di 3, allora la estendiamo al massimo, realizzando una riga o colonna completa.
Se dopo l'estensione non sono presenti T-giunzioni, allora si forma una B-spline che può essere trattata come componente finale collegata con la superficie genitore; se invece ne sono ancora presenti, la mesh va passata per un'ulteriore scomposizione al livello successivo.

3.5.4 Esempio

Illustriamo l'algoritmo topologicamente con un esempio.
La figura 3.12a mostra una T-mesh che ha quattro T-giunzioni; vogliamo convertire la T-spline in una B-spline gerarchica. Per prima cosa eliminiamo le righe e le colonne parziali, ottenendo una mesh B-spline (figura 3.12b); in seugito, dalla T-mesh, costruiamo due box-2, uno per la riga parziale e uno per la colonna parziale (figura 3.12c). I due box-2 si sovrappongono, e devono pertanto essere uniti in uno singolo, come mostrato in figura 3.12d; questo box identifica la regione del sovrastrato di livello successivo. L'espansione della regione delimitata dal box di un'ulteriore anello fornisce una T-mesh temporanea. L'inserimento del *negativo* della B-spline estratta all'interno di questa T-mesh temporanea porta ad avere una T-mesh per l'offset, che

topologicamente coincide con la mesh temporanea mostrata in figura 3.12e. Infine, estendiamo sia la riga parziale che la colonna parziale fino al bordo, creando una mesh 11×10 in figura 3.12f, che definisce l'offset del sovrastrato. La B-spline gerarchica che si ottiene alla fine consiste della superficie B-spline di livello 0 di figura 3.12b e dell'offset di livello 1, in figura 3.12f.

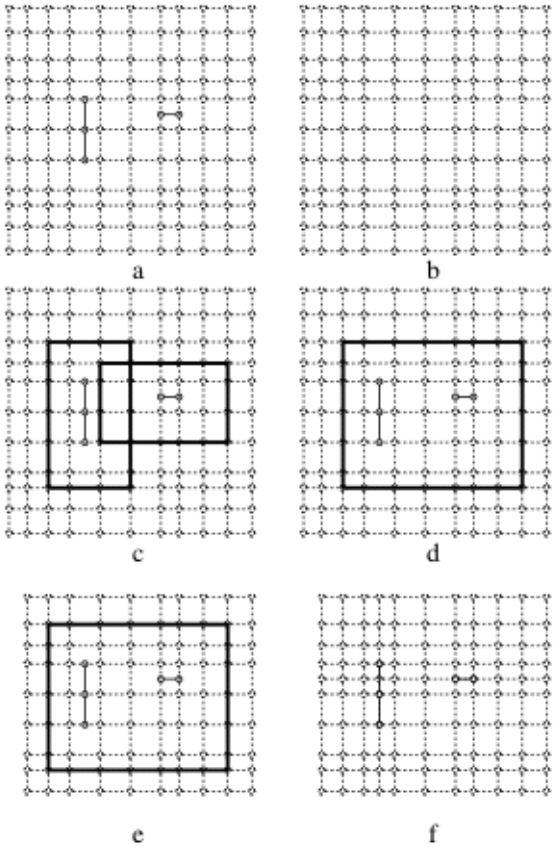


Figura 3.12: Conversione di una T-spline in una B-spline gerarchica.

Capitolo 4

Indipendenza lineare delle funzioni di miscelamento T-spline

Vediamo ora le T-spline da un punto di vista più vicino all'algebra lineare; nella fattispecie, andremo ad investigare l'indipendenza lineare o meno delle funzioni di miscelamento. Il possesso di tale proprietà o meno dipende naturalmente dalla T-mesh, ed esistono esempi di mesh per le quali le funzioni di miscelamento sono linearmente indipendenti, così come ce ne sono per cui si trovano ad essere linearmente dipendenti. Non esiste al momento una caratterizzazione generale; tuttavia, considerando che di particolare utilità possono essere le T-mesh tali che le funzioni di miscelamento sono linearmente indipendenti, sono stati compiuti studi che permettessero di affermare l'indipendenza lineare di esse per T-mesh costruite a partire da mesh per le quali essa sia già nota.
Ricordiamo infatti, come visto nel § 2.3, che se le funzioni di miscelamento $B^{(i,j)}(s,t)$, con $(i,j) \in \mathfrak{U}$, sono linearmente indipendenti, allora $\dim(S(\mathfrak{M})) = \#\mathfrak{U}$; infatti, esse sono in numero pari al numero di ancore, e generano ad ogni modo $S(\mathfrak{M})$ per costruzione, dunque in questo caso ne formano una base.
In questo contesto, basandoci sull'algoritmo di raffinamento locale (§ 3.1, 3.2), analizzeremo l'indipendenza lineare delle funzioni di miscelamento T-spline bicubiche corrispondenti a particolari T-mesh. Quanto segue è tratto da [18]; uno sviluppo molto recente sull'argomento è oggetto di [24].

4.1 Premesse

Utilizzando la notazione di § 2.3, e con un lieve abuso di notazione, possiamo dire che:

$$\mathfrak{M} \text{ è una T-mesh linearmente indipendente}$$
$$\Leftrightarrow B^{(i,j)}, (i,j) \in \mathfrak{U} \text{ sono linearmente indipendenti}$$

e abbiamo detto che ciò implica la dimensione di S pari a $\#\mathfrak{U}$.
Abbiamo già classificato (§ 3.1.1, 3.3) gli spazi T-spline in standard, semi-

standard, e non standard. Ricapitolando brevemente, uno spazio T-spline è:

- standard quando $\sum_{(i,j)\in\mathfrak{U}} B^{(i,j)}(s,t) = 1$;
- semistandard quando $1 \in S$, ma non vale la somma precedente;
- non standard quando $1 \notin S$.

Nell'ultimo caso, è possibile considerare lo spazio delle T-spline razionali, come:

$$S_R := \text{span}\left\{ \frac{B^{(i,j)}(s,t)}{\sum_{(k,l)\in\mathfrak{U}} w^{(k,l)} B^{(k,l)}(s,t)}, (i,j) \in \mathfrak{U} \right\}$$

dove $w^{(k,l)}$ sono pesi strettamente positivi. Osserviamo che:

$$B^{(i,j)}(s,t), (i,j) \in \mathfrak{U} \text{ sono linearmente indipendenti } \Leftrightarrow$$
$$\Leftrightarrow \frac{B^{(i,j)}(s,t)}{\sum_{(k,l)\in\mathfrak{U}} w^{(k,l)} B^{(k,l)}(s,t)}, (i,j) \in \mathfrak{U} \text{ sono linearmente indipendenti}$$

per cui, possiamo considerare soltanto lo spazio S, senza generalizzare allo spazio S_R.
Possiamo associare a $\bar{j} \in \mathcal{I}_j$ un operatore lineare $\mathcal{L}_t^{\bar{j},r}$ che misura il salto della derivata r-esima rispetto a t nel punto $t = t_j$:

$$\mathcal{L}_t^{\bar{j},r}(f)(s) := \lim_{t\to t_{\bar{j}}^+} \frac{\partial^r}{\partial t^r} f(s,t) - \lim_{t\to t_{\bar{j}}^-} \frac{\partial^r}{\partial t^r} f(s,t)$$

In maniera simile, per ogni $\bar{i} \in \mathcal{I}_i$, possiamo definire un operatore lineare $\mathcal{L}_s^{\bar{i},r}$ che misura il salto della derivata r-esima rispetto a s nel punto $s = s_i$.
Si nota che per una funzione di miscelamento T-spline bicubica $B^{(i,j)}(s,t) = N_s^{(i,j)}(s) N_t^{(i,j)}(t), (i,j) \in \mathfrak{U}$ sulla data T-mesh $\mathfrak{M}$:

$$\begin{aligned} \mathcal{L}_t^{\bar{j},r}(B^{(i,j)})(s) &= N_s^{(i,j)}(s) \mathcal{L}_t^{\bar{j},r}(N_t^{(i,j)}(t)) \\ \mathcal{L}_s^{\bar{i},r}(B^{(i,j)})(t) &= \mathcal{L}_s^{\bar{i},r}(N_s^{(i,j)}(s)) N_t^{(i,j)}(t) \end{aligned}$$

4.2 Indipendenza lineare delle T-spline

Volendo investigarne l'indipendenza lineare o meno, ci si chiede innanzitutto se, per caso, non sia che *tutte* le T-mesh si trovino ad essere linearmente indipendenti. Come abbiamo già affermato nel § 2.3, la risposta è negativa, e ora lo mostriamo con un controesempio, considerando la T-mesh in figura 4.1.

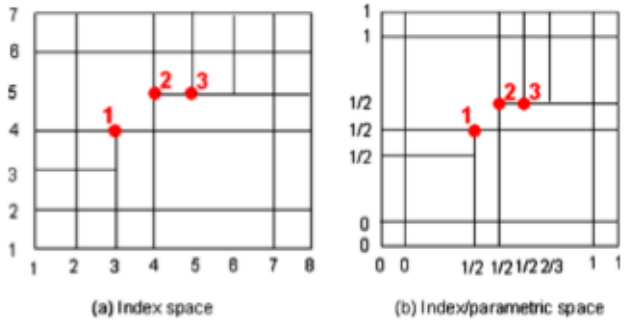


Figura 4.1: Un controesempio di T-spline linearmente dipendenti.

Le T-spline associate alle ancore $(3,4),(4,5),(5,5)$ mostrate nello spazio degli indici (figura 4.1a.), che corrispondono all'immagine ripetuta $(s_3,t_4) = (s_4,t_5) = (s_5,t_5) = (1/2,1/2)$ nello spazio indici/parametri (figura 4.1b.), sono ottenute come segue:

$$\begin{aligned}\Theta_1(s,t) &= N[0,0,1/2,1/2,1](s)N[0,1/2,1/2,1,1](t)\\ \Theta_2(s,t) &= N[0,0,1/2,1/2,2/3](s)N[0,1/2,1/2,1,1](t)\\ \Theta_3(s,t) &= N[0,1/2,1/2,2/3,1](s)N[0,1/2,1/2,1,1](t)\end{aligned}$$

Per la regola dell'inserzione di nodi, $\Theta_1 = \Theta_2 + \frac{1}{3}\Theta_3$, e le funzioni di miscelamento T-spline bicubiche associate alla T-mesh di figura 4.1a. sono linearmente dipendenti. Questo significa che le T-spline non formano sempre un insieme di funzioni linearmente indipendenti, o equivalentemente che non tutte le T-mesh sono linearmente indipendenti.
Analizzeremo ora l'indipendenza lineare di funzioni di miscelamento T-spline bicubiche associate a T-mesh di interesse pratico e ottenute tramite applicazione dell'algoritmo di raffinamento locale.
Introduciamo un po' di notazione: sia $\mathfrak{M}_0$ una T-mesh iniziale, $\mathfrak{M}_1$ la T-mesh ottenuta da $\mathfrak{M}_0$ tramite l'aggiunta di una o più ancore, $\mathfrak{M}_0,\mathfrak{M}_1,\ldots,\mathfrak{M}_l$ una successione di T-mesh tali per cui $\mathfrak{M}_l$ sia ottenuta da $\mathfrak{M}_{l-1}$ attraverso l'aggiunta di una o più ancore.
Senza perdita di generalità, e per avere una notazione adeguata, supponiamo che tutte le T-mesh abbiano in comune gli insiemi di indici $\mathcal{I}_i$ e $\mathcal{I}_j$, e i vettori dei nodi Ξ_s e Ξ_t. Ogni T-mesh $\mathfrak{M}_k$ è caratterizzata dal suo insieme di ancore $\mathfrak{U}_k$ e di lati $\mathfrak{E}_k$. Tutte le quantità cui prima siano indicizzate anche dall'indice di T-mesh k, al fine di specificare che sono riferite alla T-mesh $\mathfrak{M}_k$; per esempio, fissato un $\overline{j} \in \mathcal{I}_i$, definiamo $\mathcal{I}^{\overline{j}}_{i,k}$ e $\Xi^{\overline{j}}_{s,k}$ come prima, e inoltre la successione degli spazi generati dalle T-spline bicubiche relative alle T-mesh

$\mathfrak{M}_0, \mathfrak{M}_1, \ldots, \mathfrak{M}_l$ è:

$$\begin{aligned} S_0 &:= \text{span}\{B_0^{(i,j)}(s,t) : B_0^{(i,j)}(s,t) = N_{s,0}^{(i,j)}(s)N_{t,0}^{(i,j)}(t), (i,j) \in \mathfrak{U}_0\} \\ S_1 &:= \text{span}\{B_1^{(i,j)}(s,t) : B_1^{(i,j)}(s,t) = N_{s,1}^{(i,j)}(s)N_{t,1}^{(i,j)}(t), (i,j) \in \mathfrak{U}_1\} \\ &\ldots \\ S_l &:= \text{span}\{B_l^{(i,j)}(s,t) : B_l^{(i,j)}(s,t) = N_{s,l}^{(i,j)}(s)N_{t,l}^{(i,j)}(t), (i,j) \in \mathfrak{U}_l\} \end{aligned}$$

Avendo assunto che la T-mesh $\mathfrak{M}_{k+1}$ è ottenuta da $\mathfrak{M}_k$ attraverso l'aggiunta di ancore, ovvero $\mathfrak{U}_0 \subset \mathfrak{U}_1 \subset \ldots \subset \mathfrak{U}_l$, e osservando che ciò non implica in automatico $S_0 \subset S_1 \subset \ldots \subset S_l$, diciamo che $\mathfrak{M}_{k+1}$ è un *raffinamento* di $\mathfrak{M}_k$ se vale $S_k \subset S_{k+1}$. L'algoritmo di raffinamento che abbiamo già visto genera effettivamente nuove T-mesh che sono dei raffinamenti delle precedenti secondo questa definizione.
Vale il seguente risultato:

Proposizione 4.2.1. *Sia $\mathfrak{M}_0$ una T-mesh linearmente indipendente, e $\mathfrak{M}_1$ un suo raffinamento. Se $\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$ è un insieme linearmente indipendente, e la somma di S_0 con span$\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$ è diretta, allora $\mathfrak{M}_1$ è linearmente indipendente e in particolare vale:*

$$S_0 \oplus span\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\} = S_1$$

Dimostrazione. Poiché $\mathfrak{M}_1$ è un raffinamento di $\mathfrak{M}_0$, entrambi gli insiemi $\{B_0^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_0\}$ e $\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$ sono contenuti in S_1, e generano pertanto un sottoinsieme di S_1. Proviamo ora che tale sottoinsieme non può altro che essere S_1 stesso calcolandone la dimensione; vale infatti:

$$\dim(S_0 \oplus \text{span}\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}) = \#\mathfrak{U}_0 + \#(\mathfrak{U}_1 \backslash \mathfrak{U}_0) = \#\mathfrak{U}_1$$

dunque $S_0 \oplus \text{span}\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$ ed S_1 coincidono e la tesi è dimostrata. □

A questo punto possediamo una strategia per provare l'indipendenza lineare delle funzioni di miscelamento T-spline: data una T-mesh linearmente indipendente $\mathfrak{M}_0$, e un suo raffinamento $\mathfrak{M}_1$, dobbiamo mostrare che:

(i) Le funzioni aggiunte, ovvero $\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$, sono linearmente indipendenti;

(ii) La somma tra S_0 e tale insieme è diretta.

I seguenti lemmi ci possono essere molto utili in tal senso.

Lemma 4.2.2. *Consideriamo una T-mesh linearmente indipendente $\mathfrak{M}_0$, e una T-mesh $\mathfrak{M}_1$ ottenuta aggiungendo ancore a $\mathfrak{M}_0$ con la proprietà che esiste un unico $\bar{j} \in \mathcal{I}_j$ tale per cui $\mathfrak{U}_1 \backslash \mathfrak{U}_0 = \{(i,\bar{j}), i \in \mathcal{I}_i\} \cap \mathfrak{U}_1$, e tale per cui $\forall (i,j) \in \mathfrak{U}_0, t_j \neq t_{\bar{j}}$. Allora le funzioni $\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$ sono linearmente indipendenti, e la seguente somma è diretta:*

$$S_0 \oplus span\{B_1^{(i,\bar{j})}(s,t) : (i,\bar{j}) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0\}$$

La prima delle due condizioni date afferma che aggiungiamo ancore su di una nuova linea orizzontale nello spazio degli indici, mentre la seconda che questa linea viene mandata in una nuova linea nello spazio dei parametri. Prima di passare alla dimostrazione, osserviamo che $\mathfrak{M}_1$ non è necessariamente un raffinamento di $\mathfrak{M}_0$, come illustrato in figura 4.2, in quanto l'algoritmo di raffinamento locale potrebbe richiedere inserzioni in più; inoltre, è opportuno ricordare che non ci sono restrizioni nella posizione di nuove ancore nella direzione orizzontale.

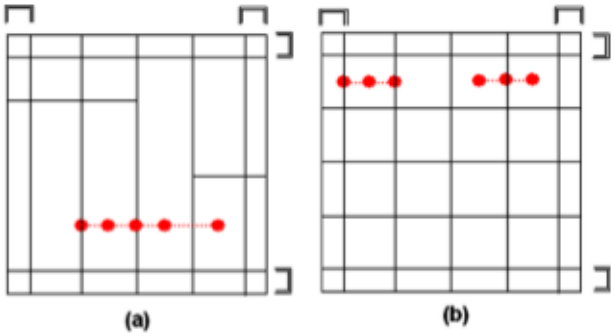


Figura 4.2: Le linee puntate denotano nuove linee sulla T-mesh $\mathfrak{M}_0$ nello spazio indici/parametri. Nuove ancore sono identificate con cerchi, e le parentesi parallele segnalano linee ripetute.

Possiamo ora dimostrare il lemma.

Dimostrazione. Osserviamo per iniziare che la tesi è equivalente alla seguente implicazione: se:

$$\sum_{(i,j)\in\mathfrak{U}_0} \alpha^{(i,j)} B_0^{(i,j)}(s,t) + \sum_{(i,\bar{j})\in\mathfrak{U}_1\backslash\mathfrak{U}_0} \beta^{(i,\bar{j})} B_1^{(i,\bar{j})}(s,t) = 0$$

allora:

$$\begin{cases} \alpha^{(i,j)} = 0, \ \forall (i,j) \in \mathfrak{U}_0 \\ \beta^{(i,\bar{j})} = 0, \ \forall (i,\bar{j}) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0 \end{cases}$$

Supponiamo l'ipotesi e applichiamo l'operatore di salto $\mathcal{L}_t^{\bar{j},3}$ al primo membro. Osserviamo che $\mathcal{L}_t^{\bar{j},3}$ mappa tutte le $B_0^{(i,j)}(s,t), (i,j) \in \mathfrak{U}_0$, nella funzione nulla, dal momento che esse sono C^∞ in $t = t_{\bar{j}}$. Questo porta a ridurre

l'equazione in:

$$\sum_{(i,\bar{j})\in\mathfrak{U}_1\backslash\mathfrak{U}_0} \beta^{(i,\bar{j})}\mathcal{L}_t^{\bar{j},3}(B_1^{(i,\bar{j})}(s,t)) = \sum_{(i,\bar{j})\in\mathfrak{U}_1\backslash\mathfrak{U}_0} \beta^{(i,\bar{j})}\mathcal{L}_t^{\bar{j},3}(N_{t,1}^{(i,\bar{j})}(t))N_{s,1}^{(i,\bar{j})}(s) = 0$$

Qui, $\mathcal{L}_t^{\bar{j},3}(N_{t,1}^{(i,\bar{j})}(t))$ sono coefficienti tutti non nulli, dal momento che $N_{t,1}^{(i,\bar{j})}(t), (i,\bar{j}) \in \mathfrak{U}_1\backslash\mathfrak{U}_0$ sono soltanto C^2 in $t = t_{\bar{j}}$; sappiamo anche che le spline univariate $N_{s,1}^{(i,\bar{j})}(s)$, associate al vettore dei nodi $\Xi_{s,1}^{\bar{j}}$, sono linearmente indipendenti. Risulta pertanto che $\beta^{(i,\bar{j})} = 0$ per ogni $(i,\bar{j}) \in \mathfrak{U}_1\backslash\mathfrak{U}_0$; la sostituzione nell'ipotesi e l'indipendenza lineare delle $B_0^{(i,j)}(s,t)$ fornisce infine l'altra delle due conseguenze. □

Scambiando le linee verticali con quelle orizzontali nell'enunciato del lemma precedente, si ottiene un risultato equivalente: in quel caso, l'aggiunta di nuove ancore deve avvenire con la proprietà che esiste un unico $\bar{i} \in \mathcal{I}_i$ tale per cui $\mathfrak{U}_1\backslash\mathfrak{U}_0 = \{(\bar{i},j), j \in \mathcal{I}_j\} \cap \mathfrak{U}_1$, e tale che $\forall (i,j) \in \mathfrak{U}_0, s_i \neq s_{\bar{i}}$. La prima delle due condizioni afferma che aggiungiamo ancore su di una nuova linea verticale nello spazio degli indici, e la seconda che la linea è mappata in una nuova linea nello spazio dei parametri.
Estendiamo ora il lemma al caso in cui nuove ancore sono inserite su di una linea verticale la cui immagine è già presente nello spazio dei parametri, ovvero, trattiamo il caso di ripetizione di nodi nella direzione verticale. Il controesempio di figura 4.1 ci dice che questa operazione non è sempre possibile, tuttavia si può procedere se vengono imposte alcune restrizioni sulla posizione delle nuove ancore.

Lemma 4.2.3. *Consideriamo una T-mesh linearmente indipendente $\mathfrak{M}_0$, e una T-mesh $\mathfrak{M}_1$ ottenuta aggiungendo ancore a $\mathfrak{M}_0$ con la proprietà che esista un unico $\bar{j} \in \mathcal{I}_j$ tale per cui $\mathfrak{U}_1\backslash\mathfrak{U}_0 = \{(i,\bar{j}), i \in \mathcal{I}_i\} \cap \mathfrak{U}_1$. Sia inoltre $r_{\bar{j},0} = \max_i \#\{j : (i,j) \in \mathfrak{U}_0, t_j = t_{\bar{j}}\}$. Se è vero che:*

$$\forall i \in \{i : (i,\bar{j}) \in \mathfrak{U}_1\backslash\mathfrak{U}_0\}, \quad \#\{j : (i,j) \in \mathfrak{U}_1 : t_j = t_{\bar{j}}\} = r_{\bar{j},0} + 1$$

allora le funzioni $\{B_1^{(i,j)}(s,t) : (i,j) \in \mathfrak{U}_1\backslash\mathfrak{U}_0\}$ sono linearmente indipendenti, e la seguente somma è diretta:

$$S_0 \oplus span\{B_1^{(i,\bar{j})}(s,t) : (i,\bar{j}) \in \mathfrak{U}_1\backslash\mathfrak{U}_0\}$$

Osserviamo che $r_{\bar{j},0}$ è il massimo numero di ripetizioni nella direzione verticale che può verificarsi in $t = t_{\bar{j}}$. L'ultima ipotesi effettuata si traduce nella possibilità di aggiungere ancore soltanto in posizioni in cui si verifica il massimo numero di ripetizioni dei nodi verticali, come mostrato in figura 4.3. Osserviamo che il lemma precedente è un caso particolare di questo, in cui $r_{\bar{j},0} = 0$.

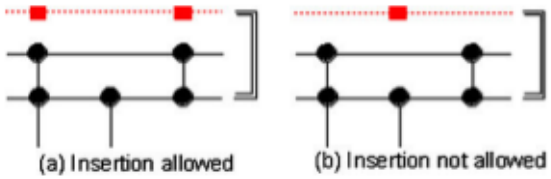


Figura 4.3: Le nuove ancore sono rappresentate da rettangoli; le linee puntate denotano linee nuove sulla T-mesh nello spazio indici/parametri; parentesi parallele segnalano linee ripetute. Le figure (a) e (b) rappresentano parte dello spazio indici/parametri, e mostrano quando l'inserzione di nuove ancore sia ammessa o meno.

Dimostrazione. L'operatore lineare $\mathcal{L}_t^{\bar{j},(3-r_{\bar{j},0})}$ mappa una funzione in S_0 nella funzione nulla, dal momento che $B_0^{(i,j)}(s,t),(i,j) \in \mathfrak{U}_0$ è $C^{3-r_{\bar{j},0}}$ in $t = t_{\bar{j}}$. Le spline univariate $N_{s,1}^{(i,\bar{j})}(s)$ associate al vettore dei nodi $\Xi_{s,1}^{\bar{j}}$ sono chiaramente linearmente indipendenti.
Per ogni $(i,\bar{j}) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0, B_1^{(i,\bar{j})(s,t)}$ sono mappate tramite l'operatore $\mathcal{L}_t^{\bar{j},(3-r_{\bar{j},0})}$ nelle spline univariate:

$$\mathcal{L}_t^{\bar{j},(3-r_{\bar{j},0})} B_1^{(i,\bar{j})}(s,t) = N_{s,1}^{(i,\bar{j})}(s) \mathcal{L}_t^{\bar{j},(3-r_{\bar{j},0})}(N_{t,1}^{(i,\bar{j})}(t))$$

dove i $\mathcal{L}_t^{\bar{j},(3-r_{\bar{j},0})}(N_{t,1}^{(i,\bar{j})}(t))$ sono fattori non nulli dal momento che $B_1^{(i,\bar{j})}(s,t),(i,\bar{j}) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0$ non sono $C^{3-r_{\bar{j},0}}$, ma soltanto $C^{2-r_{\bar{j},0}}$ in $t = t_{\bar{j}}$. Di conseguenza, le funzioni $\{\mathcal{L}_t^{\bar{j},(3-r_{\bar{j},0})}(B_1^{(i,\bar{j})})\}_{(i,\bar{j}) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0}$ sono linearmente indipendenti, e la dimostrazione può proseguire ricalcando la falsariga del lemma precedente. □

Naturalmente, è possibile scambiare le righe orizzontali con quelle verticali, ottenendo un risultato equivalente per l'inserimento di ancore su di una linea verticale $i = \bar{i}$; l'applicazione dei risultati fin qui ottenuti su di una successione di T-mesh raffinate porta ad un risultato chiave.

Teorema 4.2.4. *Siano date T-mesh $\mathfrak{M}_0, \ldots, \mathfrak{M}_l$. Assumiamo che $\mathfrak{M}_0$ sia linearmente indipendente, e $\mathfrak{M}_k, k = 1, \ldots, l$, siano ottenute come raffinamenti sequenziali secondo il procedimento di inserimento di ancore descritto prima. Allora $\mathfrak{M}_l$ è linearmente indipendente.*

Dimostrazione. Segue dai lemmi precedenti che $\{B_0^{(i,j)}(s,t)\}_{(i,j) \in \mathfrak{U}_0}$ e $\{B_1^{(i,j)}(s,t)\}_{(i,j) \in \mathfrak{U}_1 \backslash \mathfrak{U}_0}$ sono linearmente indipendenti (LI). Per la proposizione 4.2.1, la T-mesh $\mathfrak{M}_1$ è LI. Possiamo ora procedere con un'induzione finita: ripetendo il procedimento per $k < l$, otteniamo che, supposta $\mathfrak{M}_k$ LI, $\{B_k^{(i,j)}(s,t)\}_{(i,j) \in \mathfrak{U}_k}$ e $\{B_{k+1}^{(i,j)}(s,t)\}_{(i,j) \in \mathfrak{U}_{k+1} \backslash \mathfrak{U}_k}$ sono LI, per cui anche $\mathfrak{M}_{k+1}$ è LI; prendendo $k = l - 1$, il massimo valore ammissibile, segue la tesi. □

4.3 Esempi di T-spline linearmente indipendenti

Mostriamo ora come il teorema precedente possa essere utilizzato per provare l'indipendenza lineare in diversi casi interessanti. Dal momento che l'assunzione principale del teorema è che la successione di mesh $\mathfrak{M}_k, k = 1, \ldots, l$ sia ottenuta applicando l'algoritmo di raffinamento locale, in tutti gli esempi una nuova T-mesh $\mathfrak{M}_{k+1}$ è sempre un raffinamento della data T-mesh $\mathfrak{M}_k$, e dunque S_k è un sottospazio di S_{k+1}. Particolare interesse ricoprono le ancore (nodi) aggiuntive, che possono essere presenti in quanto l'algoritmo di raffinamento locale può agire su di una nuova ancora (nodo). Il risultato precedente non può essere applicato a tutti i tipi di ancore aggiuntive, per cui, quando una nuova ancora è inserita in una data T-mesh $\mathfrak{M}_k$, ci limitiamo *al caso in cui le sue ancore aggiuntive stanno su di un nuovo lato (o su di una nuova linea) su* $\mathfrak{M}_k$*, che le collega alla nuova ancora.* Ancore aggiuntive possono creare L-giunzioni nelle griglie T-spline (figure 4.4, 4.6b, 4.7). Qui, l'aggiunta di una nuova linea in una T-mesh $\mathfrak{M}_k$ è una successione di inserzioni di ancore, nella quale nuove ancore e le loro conseguenti ancore aggiuntive (*se presenti*) stanno *solo sulla nuova linea in* $\mathfrak{M}_k$.
Per mettere in evidenza le operazioni di raffinamento che andremo a compiere, prenderemo sempre come T-mesh iniziale $\mathfrak{M}_0$ una B-mesh, disegnandola nello spazio dei parametri. Metteremo in evidenza l'inserimento di (immagini di) nuove ancore, eventualmente ripetute; in tal caso, le rappresenteremo vicine. Le diverse forme utilizzate (figura 4.4b) servono a distinguerle per gruppo: per esempio, se un'ancora triangolare (o rettangolare) è inserita in figura 4.4a, le tre ancore a forma di stella sono quelle aggiuntive richieste dall'algoritmo di raffinamento locale. Inoltre, semplifichiamo anche la notazione grafica secondo la figura 4.5: la ripetizione di nodi su nuove linee aggiunte è spesso incorporata in un singolo passo, ovvero si disegna quanto in figura 4.5b invece di quanto in figura 4.5a; nel caso in cui non ci possa essere ambiguità, accettiamo la rappresentazione ridotta di figura 4.5c.
Vediamo ora esempi di T-mesh linearmente indipendenti di interesse pratico. In ognuno di questi esempi, viene fornita una spiegazione dettagliata.

4.3.1 T-mesh semplici

Qualora non vi possa essere ambiguità, omettiamo le parentesi parallele ad indicare ripetizione di nodi, nello spazio indici/parametri delle figure in questione. I primi due esempi sono illustrati in figura 4.6. Incominciamo con una T-mesh uniforme $\mathfrak{M}_0$ presentata in figure 4.6(a-1) e 4.6(b-1). Come primo passo dei raffinamenti, aggiungiamo ancore rettangolari verticalmente alla T-mesh $\mathfrak{M}_0$ due (o tre) volte. Queste aggiunte portano ad avere una T-mesh $\mathfrak{M}_1$; il lemma 4.2.3 ci garantisce che essa è linearmente indipendente. Nel raffinamento successivo, inseriamo delle ancore circolari (o una sola ancora circolare e le sue ancore aggiuntive indicate da stelle) alla T-mesh $\mathfrak{M}_1$, che

fornisce una T-mesh $\mathfrak{M}_2$ mostrata in figure 4.6(a-5) e 4.6(b-5). Notiamo che, per prima cosa, i lati verticali in figura 4.6(b-4) possono effettivamente essere aggiunti, per il teorema 4.2.4; inoltre ad ogni passo applichiamo l'algoritmo di raffinamento locale, per cui, sempre per il teorema 4.2.4, la T-mesh $\mathfrak{M}_2$ è linearmente indipendente.
Negli esempi seguenti, le T-mesh di partenza $\mathfrak{M}_0$ mostrate nella colonna più a sinistra di figura 4.7 sono differenti dalle T-mesh uniformi. L'esempio di figura 4.6a ci assicura che queste T-mesh $\mathfrak{M}_0$ sono linearmente indipendenti. Aggiungiamo ancore circolari e le loro conseguenti ancore aggiuntive, portandoci ad avere una T-mesh $\mathfrak{M}_1$. Osserviamo anche che i lati verticali in figura 4.7(a-3) e 4.7(b-3) possono essere aggiunti per il teorema 4.2.4; il nostro risultato asserisce che la T-mesh $\mathfrak{M}_1$ è linearmente indipendente.

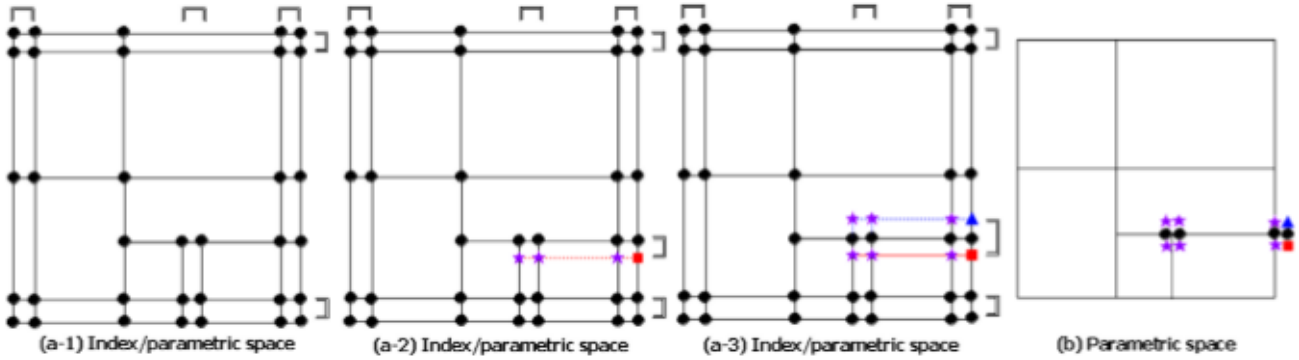


Figura 4.4: Nuove ancore sono rappresentate da rettangoli/triangoli/stelle. Le linee puntate denotano linee nuove aggiunte sulla T-mesh nello spazio indici/parametri. Le parentesi parallele in (a) segnalano linee ripetute.

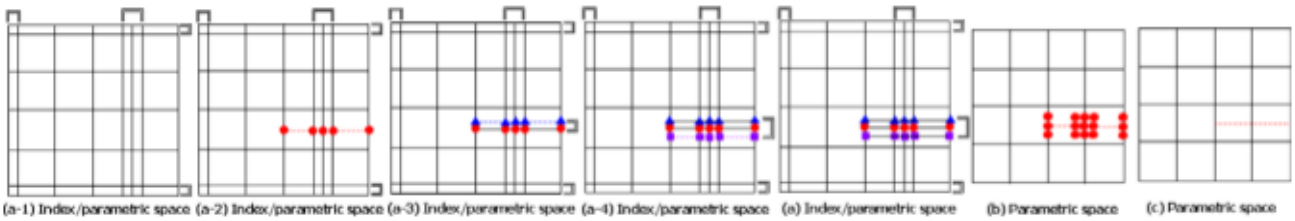


Figura 4.5: Le linee puntate denotano nuove linee aggiunte. (a) Nuove ancore sono rappresentate da cerchi/rettangoli/triangoli sulla T-mesh nello spazio indici/parametri. La stessa T-mesh nello spazio dei parametri, rappresentata in due modi differenti. (b) Se i passi di raffinamento locali eseguiti in (a) sono combinati, nuove ancore possono essere rappresentate da cerchi. (c) La linea nuova aggiunta è C^0 nella direzione t. Le parentesi parallele indicano linee ripetute.

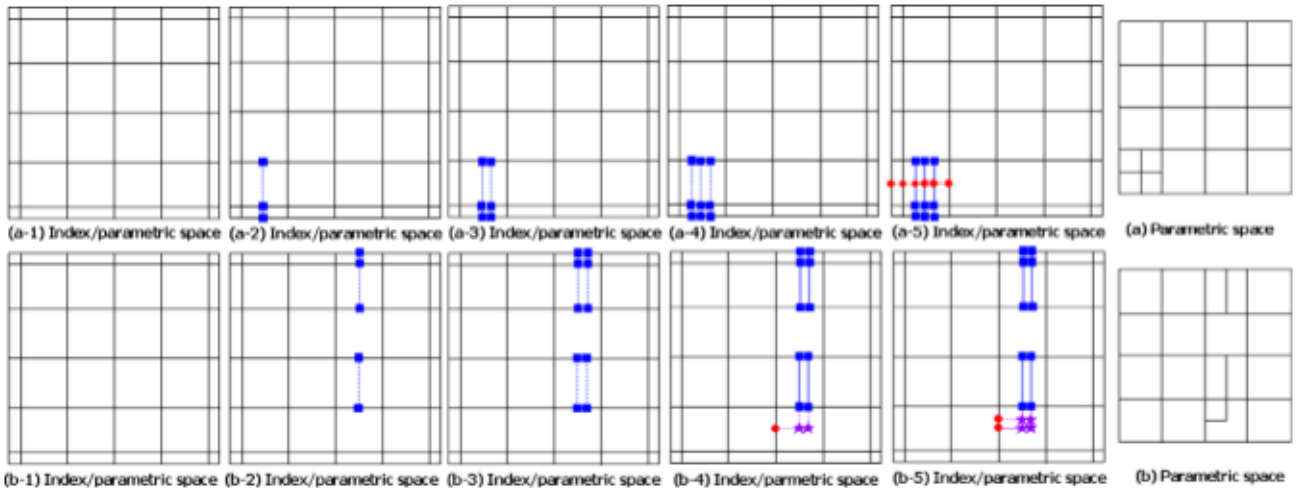


Figura 4.6: Successioni di T-mesh. Le linee nuove aggiunte sono indicate con linee puntate, e rettangoli, cerchi o stelle contrassegnano le ancore nuove; in particolare, i rettangoli blu indicano le ancore aggiunte su preesistenti linee orizzontali, i cerchi rossi quelle aggiunte su verticali, le stelle viola, in (b), le ancore aggiuntive che risultano necessarie dopo gli inserimenti precedenti. L'ordine di continuità delle T-spline corrispondenti, lungo la nuova linea nello spazio dei parametri, è dato da C^{3-r}, dove r è il numero di ancore ripetute nella direzione s o t.

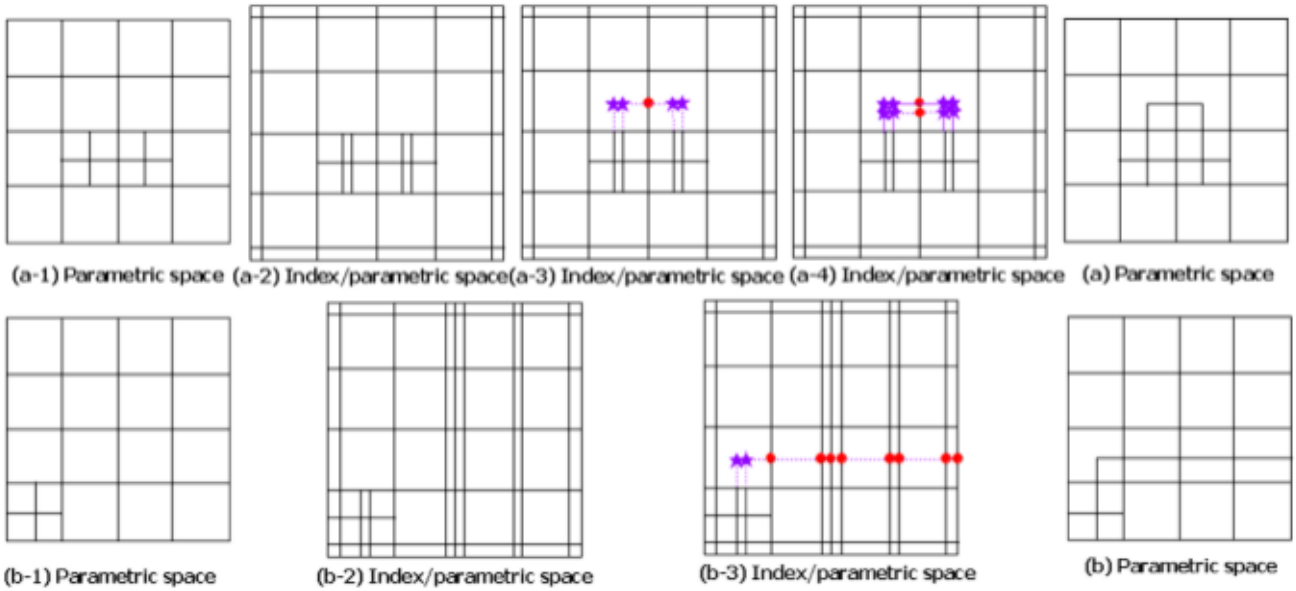


Figura 4.7: Successioni di T-mesh. Le linee nuove aggiunte sono indicate con linee puntate. Le nuove ancore sono contrassegnate da cerchi/stelle. In particolare, le ancore a forma di stella sono quelle aggiuntive.

4.3.2 T-mesh gerarchiche

Ci interessiamo ora dell'indipendenza lineare delle T-mesh gerarchiche.
Una T-mesh gerarchica è il risultato di una T-suddivisione gerarchica, ovvero una suddivisione ottenuta progressivamente, dividendo ad ogni passo in due una e una sola cella, attraverso una linea verticale o orizzontale; più formalmente, possiamo definirla in maniera induttiva come o il rettangolo iniziale (base dell'induzione), o una suddivisione ottenuta da una suddivisione già nota come T-suddivisione gerarchica dividendo in due una e una sola cella (passo induttivo). La figura 4.8 mostra un siffatto esempio.

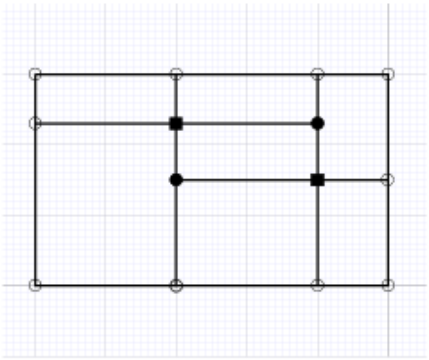


Figura 4.8: T-suddivisione gerarchica.

Non tutte le T-mesh si possono ottenere come risultato di un procedimento di questo tipo: un controesempio è mostrato in figura 4.9.

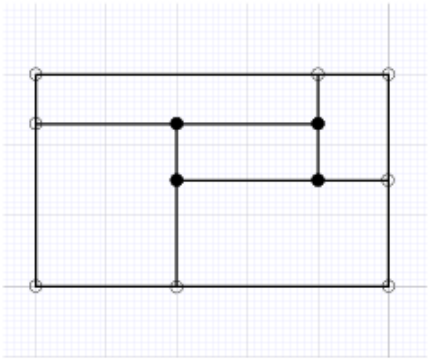


Figura 4.9: T-suddivisione non gerarchica.

Incominciamo dunque con T-mesh uniformi $\mathfrak{M}_0$. Con una T-mesh meno raffinata, l'algoritmo di raffinamento locale genera ad ogni passo T-mesh

gerarchiche più fini, come illustrato in figura 4.10. È facile osservare che le nuove linee aggiunte, contrassegnate da linee puntate in figura 4.10, si inseriscono nel contesto del teorema 4.2.4, e sono pertanto permesse dall'algoritmo di raffinamento locale; inoltre, il teorema stesso implica che le corrispondenti T-mesh gerarchiche $\mathfrak{M}_k$ siano linearmente indipendenti. Per comprendere meglio l'indipendenza lineare delle T-mesh gerarchiche, tutte le nuove linee in figura 4.10 non sono ripetute. Vogliamo porre l'enfasi sul fatto che sotto le assunzioni del secondo lemma le nuove linee inserite possono essere ripetute.

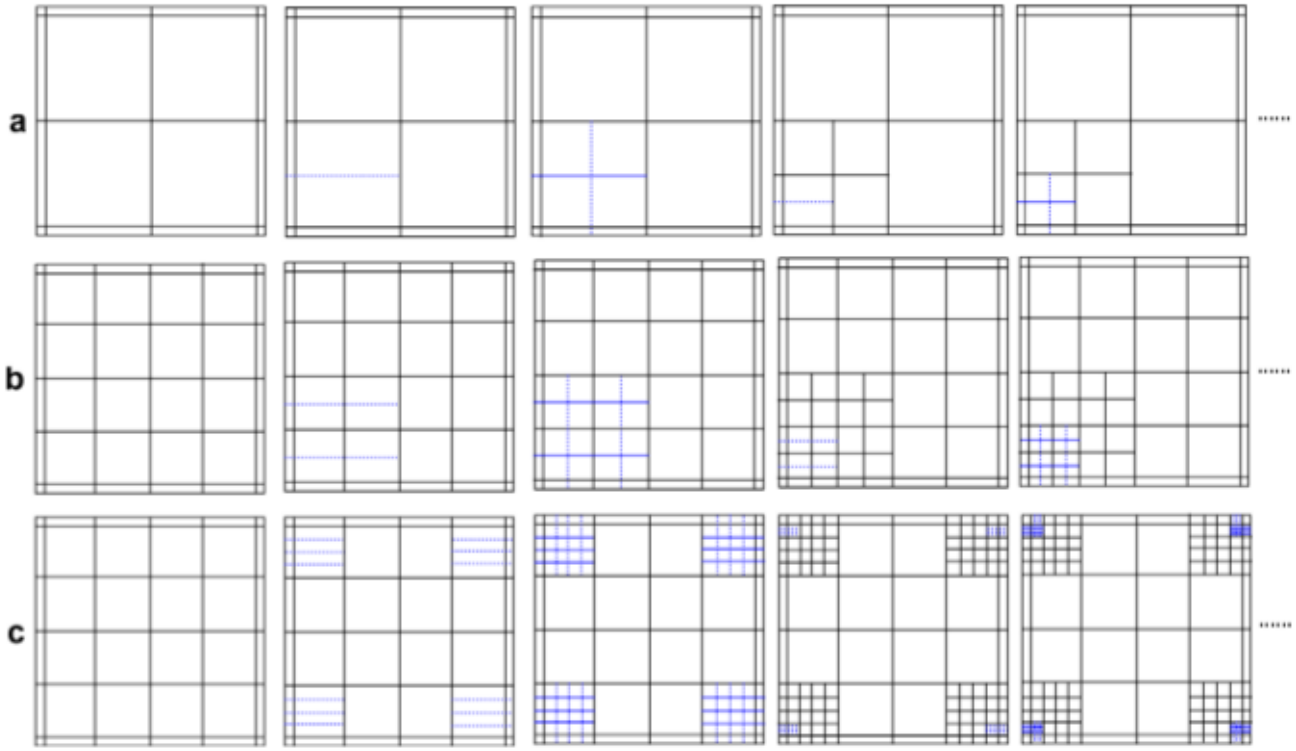


Figura 4.10: Successioni di T-mesh gerarchiche nello spazio indici/parametri. Le linee puntate contrassegnano linee nuove aggiunte.

4.3.3 T-mesh non conformi e T-mesh geometricamente raffinate su di uno strato limite

Prestiamo ora attenzione al caso delle T-mesh non conformi e a quelle associate ad un raffinamento geometrico verso uno strato limite.
Una T-mesh non conforme è una mesh in cui il raffinamento in una direzione viene compiuto in modo tale da lasciare una discrepanza in corrispondenza di una linea nell'altra direzione. Per esempio, una B-mesh è conforme, in

quanto tutte le varie parti sono esattamente collegate tra loro in corrispondenza dei nodi, e non ad esempio a metà di un lato; la T-mesh di figura 4.11a, invece, non lo è, e costituisce un esempio notevole di T-mesh non conforme. Una T-mesh associata ad un raffinamento geometrico verso uno strato limite è invece una mesh nella quale si raffina progressivamente nei pressi di un bordo, concentrando cioè l'aumento di dettaglio, corrispondente all'infittimento della griglia, verso il perimetro esterno della T-mesh (figura 4.11b).
Il primo esempio riguarda una T-mesh non conforme, presentata in figura 4.11a. Incominciando dalla discretizzazione B-spline $\mathfrak{M}_0$, mostrata in figura 4.11(a-1), sette linee puntate sono aggiunte alla T-mesh $\mathfrak{M}_0$. Queste aggiunte forniscono una nuova T-mesh $\mathfrak{M}_7$, che è un raffinamento di $\mathfrak{M}_0$. Alla fine, la T-mesh non conforme è linearmente indipendente.
Nel secondo esempio, una T-mesh uniforme $\mathfrak{M}_0$ continua ad essere raffinata su di uno strato limite come mostrato in figura 4.11b. Ad ogni passo, sono aggiunte linee puntate. Segue dal teorema 4.2.4 che le T-mesh geometricamente raffinate risultanti sullo strato limite sono linearmente indipendenti.

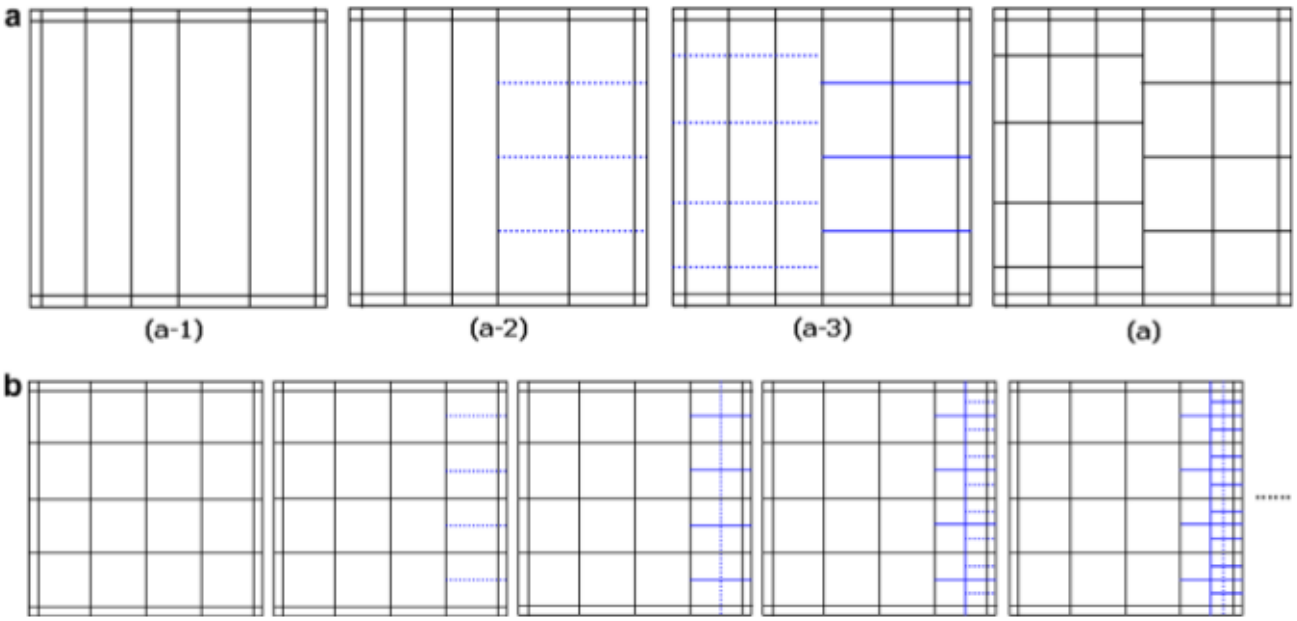


Figura 4.11: (a) Una successione di T-mesh non conformi. (b) Una successione di T-mesh associate ad un raffinamento geometrico verso uno strato limite. Le linee puntate denotano linee nuove aggiunte sulle T-mesh nello spazio indici/parametri.

4.3.4 T-mesh ancora più particolari

Un ultimo caso di T-mesh che trattiamo, considerato particolarmente peculiare, è quello presentato in figura 4.12. In [25], tale caso viene descritto come *worst case scenario*, ad indicare che si tratta di una situazione piuttosto atipica, ma che proprio per questo merita di essere trattata a parte.
Supponiamo di ottenere una T-mesh $\mathfrak{M}_0$ come tensore prodotto di due vettori dei nodi globali:

$$\begin{aligned} \Xi_s &= \{0,0,0,0,1/2,1/2,1/2,1,1,1,1\} \\ \Xi_t &= \{0,0,0,0,1/2,1/2,1/2,1,1,1,1\} \end{aligned}$$

come mostrato nella parte più in alto a sinistra di figura 4.12a.
Effettuiamo ora un raffinamento suddividendo le patch contenenti la diagonale da $(0,0)$ a $(1,1)$ in quattro parti e aggiungendo ulteriori ancore (vedi figura 4.12a).
Al fine di applicare i risultati a questo caso, la T-mesh $\mathfrak{M}_{k+1}$ deve essere costruita dalla T-mesh meno raffinata $\mathfrak{M}_k$, tramite l'algoritmo di raffinamento locale, in maniera tale per cui una nuova linea sia aggiunta in $\mathfrak{M}_k$ dove la nuova linea dritta collega a nuove ancore e alle loro conseguenti ancore aggiuntive.
Per esempio, in figura 4.12b, l'inserimento della nuova ancora nel quadrante superiore destro, dovuto alla richiesta suddivisione in quattro parti dello stesso, comporta l'inserimento di ancore aggiuntive al centro dei quadranti superiore sinistro ed inferiore destro; le nuove linee che devono essere aggiunte, secondo quanto detto sopra, sono pertanto quelle indicate in blu nella penultima immagine della riga.
Le figure 4.12b e 4.12c mostrano che la T-mesh relativa ad ogni livello può essere costruita dalla T-mesh meno raffinata tramite inserzione di nuove linee puntate che si inseriscano nel contesto del teorema 4.2.4; dunque, ad ogni livello, le T-mesh risultanti sono linearmente indipendenti.

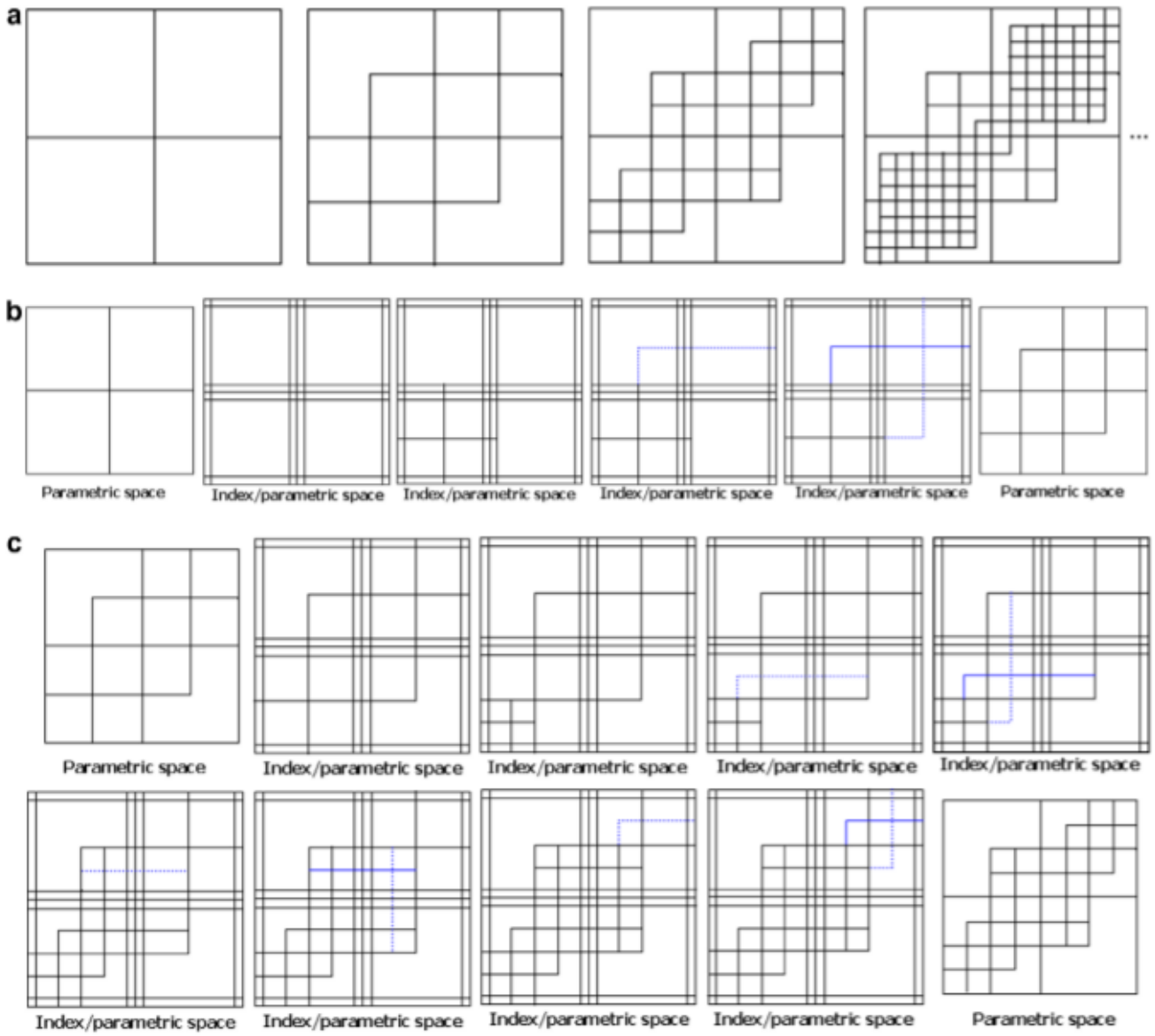


Figura 4.12: (a) Una successione di T-mesh nello spazio dei parametri. (b) $\mathfrak{M}_1$ è un raffinamento di $\mathfrak{M}_0$. (c) $\mathfrak{M}_2$ è un raffinamento di $\mathfrak{M}_1$. Le linee nuove aggiunte sono contrassegnate da linee puntate.

Capitolo 5

Le T-spline per il disegno tecnico e per il CAGD

Parlando delle T-spline in generale nel capitolo 1, ne abbiamo fin da subito disquisito in termini di utilità per il disegno tecnico e per la grafica computerizzata: benchè non l'unico, si tratta dello scopo per il quale le T-spline sono state concepite, e pertanto quello per il quale sono stati affrontati i maggiori sviluppi dell'argomento. Vediamo ora dunque alcune procedure ed argomentazioni relative alle T-spline ed inerenti disegno tecnico e CAGD.

5.1 Fusione di due B-spline in una T-spline

Ci occupiamo di come fondere due B-spline dotate di vettori dei nodi differenti in una singola T-spline ([1]). Spesso, nella modellizzazione geometrica, parti diverse di un oggetto vengono modellate in maniera indipendente l'una dall'altra, tramite diverse superfici B-spline che possiedono vettori dei nodi differenti. La figura 5.1 mostra un problema che può verificarsi: le due griglie di controllo più a sinistra sono definite su vettori di nodi differenti. Prima di unirle in una singola B-spline, è necessario effettuare operazioni di inserimento dei nodi, dal momento che i vettori dei nodi devono trovarsi ad essere gli stessi; ma questo può portare ad un aumento anche considerevole del numero di righe di punti di controllo, che può diventare ancora meno gestibile nel momento in cui le superfici B-spline che si vogliono unire sono più di due.

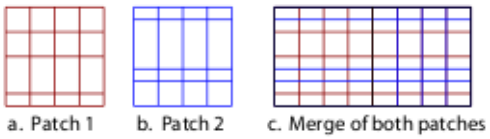


Figura 5.1: L'unione di due B-spline.

Uno dei motivi per cui è stata ideata la creazione delle T-spline è proprio la risoluzione di questa questione; soluzioni precedenti, quali l'utilizzo di superfici `NURSS`, un altro concetto evoluto rispetto alle B-spline (si vedano [8] e [9]), si erano mostrate non ottimali, portando all'ideazione di questo nuovo strumento (si veda in tal senso anche la figura 5.2).

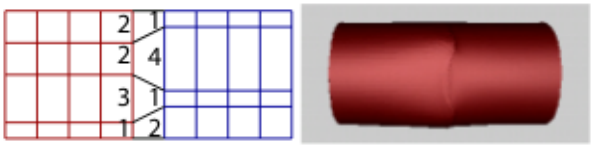


Figura 5.2: L'unione di due B-spline usando le `NURSS` cubiche.

La figura 5.3 mostra la fusione di due B-spline in una T-spline.

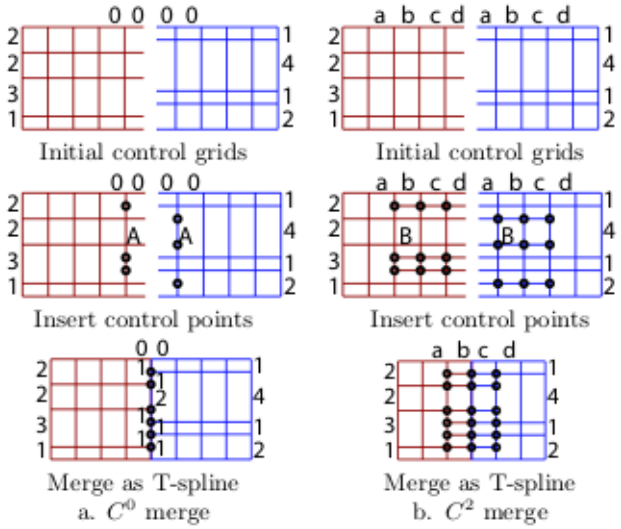


Figura 5.3: L'unione di due B-spline usando le T-spline.

Per un'unione di classe C^n, dove n può variare da -1, caso in cui non viene garantita nemmeno la continuità, a 2, il caso della continuità massima possibile, compatibilmente con il fatto che si tratti di superfici cubiche, $n+1$ colonne di punti di controllo su di una patch vengono fatte corrispondere ad altrettante $n+1$ colonne di punti di controllo sull'altra patch.
La figura 5.3.a mostra un'unione C^0, in cui innanzitutto ogni B-spline dovrà avere nodi tripli, ovvero intervalli tra nodi nulli doppi, vicino agli estremi condivisi (gli zeri che compaiono nell'immagine), ed inoltre una colonna di punti di controllo (una poiché $n = 0$ e $n+1 = 1$) si troverà ad essere condivisa dopo l'unione. Se gli intervalli tra nodi per le due T-spline sono differenti sulla colonna in comune, si rende necessario l'inserimento di punti di controllo per renderli coincidenti.

Tuttavia, a differenza del caso visto prima dell'unione senza fare uso delle T-spline, in cui nuovi punti di controllo andavano aggiunti in tutta la superficie, in questo contesto è sufficiente inserirne, al limite, sulle due colonne (una per patch) che si troveranno ad essere una colonna sola dopo l'unione, riducendo notevolmente il numero di punti nuovi necessari.
Un inconveniente minore che può verificarsi nel corso di questo procedimento, è che i punti di controllo che devo essere uniti, si trovino a possedere coordinate cartesiane lievemente differenti; per esempio, che il punto A sulla patch a sinistra differisca leggermente dal punto A sulla patch a destra. In tal caso, la soluzione consigliata è quella di prendere la media aritmetica delle due posizioni per determinare quella che assumerà il punto dopo l'unione.
La figura 5.3.b mostra invece un'unione C^2. L'idea di base è la stessa del caso C^0, con la differenza che in questo caso a dover corrispondere saranno quattro intervalli tra nodi anziché due, ma non ci sarà alcun vincolo sul fatto che essi debbano assumere il valore zero, e che le colonne di punti di controllo a dover essere unite (con l'eventuale aggiunta di altri punti di controllo per farle corrispondere) saranno tre e non una.
Le figure seguenti mostrano rispettivamente:

- il risultato di una fusione C^0 e di una fusione C^1, viste come immagini e non nel dominio dei parametri (figura 5.4);
- una mano, in cui parti distinte sono modellate in maniera indipendente (figura 5.5);
- l'applicazione del procedimento di fusione trattato al modello della mano, in maniera tale da eliminare una discrepanza (figura 5.6).

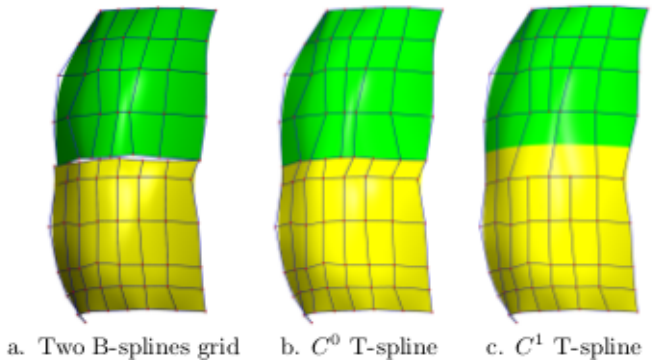


Figura 5.4: L'unione di due B-spline vista come immagine.

Vediamo ora la fusione di due B-spline in una T-spline sotto un punto di vista più formale ([19]). Utilizzando la notazione del § 2.3, e denotando come *B-mesh* le mesh tensore prodotto associate alle B-spline bivariate ed

Figura 5.5: Una mano, composta da diverse superfici B-spline.

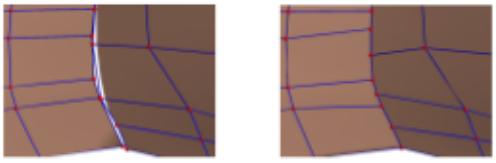


Figura 5.6: Una discrepanza tra due superfici B-spline, eliminata con una T-spline.

alle superfici B-spline (in maniera analoga all'uso del termine *T-mesh*), siano $\mathfrak{M}^l(\mathcal{I}_i^l, \mathcal{I}_j^l, \mathfrak{U}^l, \mathfrak{E}^l, \Xi_s^l, \Xi_t^l)$ e $\mathfrak{M}^r(\mathcal{I}_i^r, \mathcal{I}_j^r, \mathfrak{U}^r, \mathfrak{E}^r, \Xi_s^r, \Xi_t^r)$ le due B-mesh a sinistra e a destra, che possiamo naturalmente riportare con la notazione delle T-mesh, essendo una B-mesh identificabile come un caso particolare di T-mesh in cui la griglia è completa (figure 5.7.a e 5.7.b); ad esse, saranno associati dei rispettivi punti di controllo e pesi.
Consideriamo ora una fusione C^0. Sia $\Xi_t = \{t_{-1}, \ldots, t_{m+2}\}$ il vettore dei nodi ottenuto fondendo i due vettori di nodi Ξ_t^l e Ξ_t^r; per prima cosa, dobbiamo raffinare le parti delle mesh che devono essere fuse. Queste parti sono associate alla colonna più a destra di ancore in $\mathfrak{M}^l$ e alla colonna più a sinistra di ancore in $\mathfrak{M}^r$; allora, in $\mathfrak{M}^l$ tutte le ancore $(1, t_j)$ nello spazio parametrico, per i j compresi tra 1 e m per i quali non siano già presenti, vengono inserite (figura 5.7.c), e allo stesso modo in $\mathfrak{M}^r$ si aggiungono tutte le ancore $(0, t_j)$ nello spazio parametrico, per i j compresi tra 1 e m per i quali non siano già presenti (figura 5.7.d).
Denotiamo con $\mathfrak{M}_+^l$ e $\mathfrak{M}_+^r$ le due T-mesh raffinate, che a questo punto non saranno più B-mesh; i nuovi punti di controllo e i nuovi pesi vengono ottenuti con l'algoritmo di inserzione di nodi, per cui le parametrizzazioni della geometria $\mathbf{F}^l$ e $\mathbf{F}^r$ restano invariate.
Ora, queste due T-mesh vengono riscalate di un fattore 1/2 rispetto alla direzione orizzontale, cambiando il dominio di $\mathfrak{M}_+^l$ da $[0,1]^2$ a $[0,1/2] \times [0,1]$, e quello di $\mathfrak{M}_+^r$ da $[0,1]^2$ a $[1/2,1] \times [0,1]$; a questo punto, vengono fuse sullo spazio parametrico $[0,1]^2$, sovrapponendo le due parti raffinate (figura 5.7.e). Ciò dà luogo ad una nuova T-mesh $\mathfrak{M}(\mathcal{I}_i, \mathcal{I}_j, \mathfrak{U}, \mathfrak{E}, \Xi_s, \Xi_t)$, il cui vettore dei

nodi Ξ_s è dato da:

$$\begin{aligned}\Xi_s &= \{s_{-1}, \ldots, s_{n^l} + s_{n^r} + 1\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{s^l_{n^l-1}}{2}, \frac{1}{2}, \frac{1+s^r_2}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\}\end{aligned}$$

con tre nodi coincidenti in corrispondenza della linea di giunzione, in termini parametrici $s = 1/2$. Le funzioni di miscelamento in $S(\mathfrak{M})$ sono pertanto continue in $s = 1/2$, e i punti di controllo proiettivi vengono mediati in corrispondenza della giunzione, e lasciati invariati altrove; la geometria risultante possiede continuità C^0 sulla giunzione.
Il caso di una fusione C^1 si tratta in maniera simile, ma in questo caso devono essere raffinate due colonne di ancore in ogni B-mesh, e poi sovrapposte nel procedimento di fusione (figure 5.8.c e 5.8.d); la T-mesh risultato del processo si trova in questo caso ad avere il vettore dei nodi orizzontale corrispondente a:

$$\begin{aligned}\Xi_s &= \{s_{-1}, \ldots, s_{n^l} + s_{n^r}\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{s^l_{n^l-1}}{2}, \frac{1+s^r_2}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{1}{2}, \frac{1}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\}\end{aligned}$$

con soltanto due nodi coincidenti associati alla linea di giunzione $s = 1/2$ (figura 5.8.e); le funzioni in $S(\mathfrak{M})$ sono differenziabili con continuità in $s = 1/2$, e ancora una volta i punti di controllo proiettivi sono mediati in corrispondenza della giunzione. La geometria risultante ha continuità C^1 sulla giunzione.
Sulla stessa falsariga si tratta la situazione di una fusione C^2: sarà necessario in tal caso raffinare tre colonne di ancore in ogni B-mesh, e la T-mesh risultato del processo si troverà ad avere il vettore dei nodi orizzontale con soltanto un nodo associato alla linea di giunzione $s = 1/2$, corrispondente a:

$$\begin{aligned}\Xi_s &= \{s_{-1}, \ldots, s_{n^l} + s_{n^r} - 1\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{s^l_{n^l-2}}{2}, \frac{1+s^r_2}{2}, \frac{1+s^r_3}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{s^l_{n^l-2}}{2}, \frac{s^l_{n^l-1}}{2}, \frac{1+s^r_3}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\} \\ &= \left\{\frac{s^l_{-1}}{2}, \ldots, \frac{s^l_{n^l-2}}{2}, \frac{1}{2}, \frac{1+s^r_3}{2}, \ldots, \frac{1+s^r_{n^r+2}}{2}\right\}\end{aligned}$$

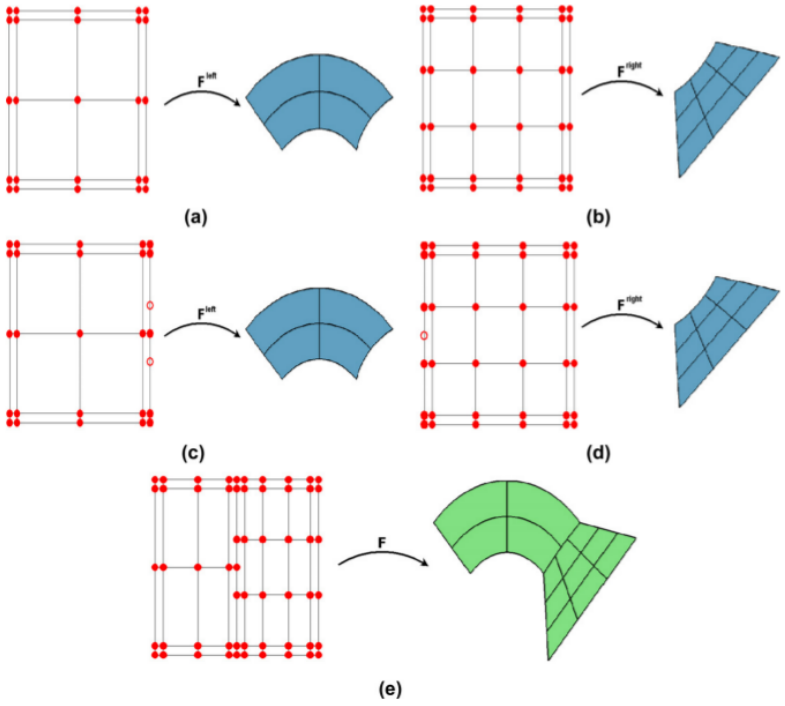


Figura 5.7: Una fusione C^0.

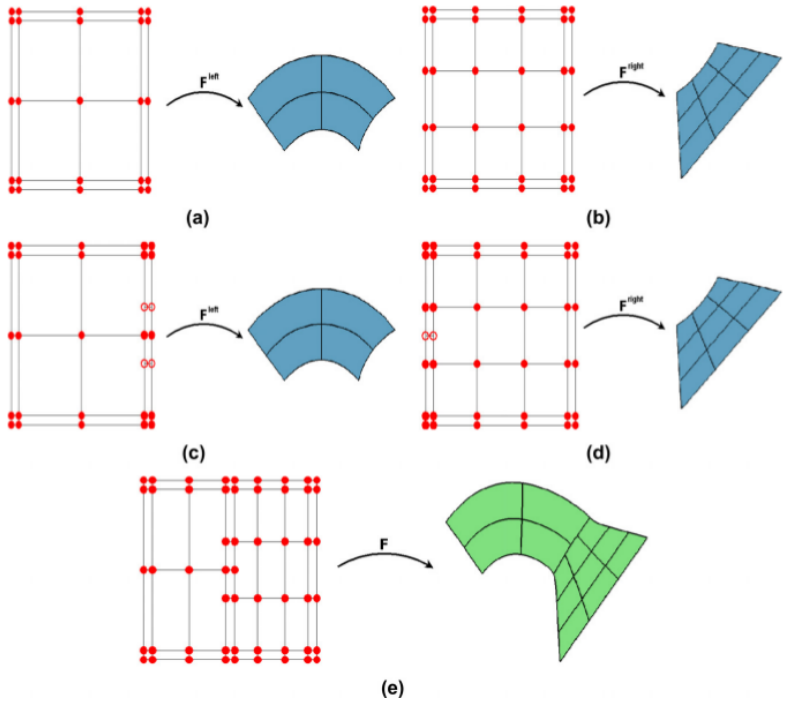


Figura 5.8: Una fusione C^1.

5.2 Semplificazione di una T-spline

Discutiamo ora la possibilità di semplificare una T-spline; con questa terminologia, intendiamo un processo attraverso il quale rimuovere i punti di controllo superflui ([2]). Tale procedimento acquisisce particolare rilevanza nel momento in cui vengono fatte entrare in gioco potenza computazionale dei calcolatori ed eventualmente necessità di rappresentazione in tempo reale: in tale contesto l'ottimizzazione delle strutture si trova ad essere molto utile se non necessaria, e se la rimozione di oggetti superflui è in grado di alleggerire il carico del sistema a fronte di una sufficientemente contenuta difficoltà di operazione, può essere opportuno effettuare semplificazioni del tipo che andiamo a vedere.
Una prima idea può essere quella di effettuare un procedimento inverso rispetto al raffinamento locale visto nel § 3.2.3; benché non sia possibile invertire l'algoritmo in senso stretto, se ne può derivare uno che fondamentalmente faccia il contrario del raffinamento. Tuttavia, una simile procedura, pur permettendo di rimuovere un singolo punto di controllo in maniera relativamente agevole, va incontro a difficoltà nel momento in cui si rende necessario rimuoverne un consistente numero.
Si riscontra dunque la necessità di applicare qualcosa di differente, e l'idea è quella di applicare delle tecniche multirisolutive opportunamente adattate.
Consideriamo una successione incapsulata di spazi T-spline: $S_0 \subset S_1 \subset \ldots \subset S_n$, e sia $T_n \in S_n$ la superficie (T-spline o `NURBS`) che vogliamo semplificare. Denotiamo con $T_i \in S_i$ la migliore approssimazione ai minimi quadrati di T_n, con $i < n$ ([10]); scegliamo poi S_0 come una griglia di punti di controllo di dimensione 4×4 in modo tale che i domini di T_0 e T_n siano lo stesso.
Denotando infine con $\mathbf{P}^i$ il vettore di punti di controllo per T_i, abbiamo che l'errore di approssimazione è pari a $D_{i,n} = (M_{i,n}\mathbf{P}^i - \mathbf{P}^n) \in S^n$, dove $M_{i,n}$ è la matrice di trasformazione introdotta nel § 3.2.2.
Il raffinamento da S_i a S_{i+1} viene svolto dividendo in due alcune facce, dette *facce incriminate*. Esse (in S_i) sono le facce i cui domini $[s_{min}, s_{max}] \times [t_{min}, t_{max}]$ contengono le coordinate nodali di un punto di controllo in $D_{i,n}$ la cui *lunghezza* eccede l'errore di approssimazione; tale definizione richiede di specificare che cosa si intende con lunghezza di un punto di controllo, e questa quantità viene definita come la norma 2 del punto stesso, visto come elemento di $\mathbf{P}^4$ (la radice quadrata della somma dei quadrati delle quattro componenti).
Nel caso di T-spline standard e B-spline polinomiali, il quarto elemento di ogni punto di controllo in $D_{i,n}$ è sempre pari a 0, semplificando un po' le cose; nel caso più generale (razionale), questa misura per la lunghezza funziona piuttosto bene se l'errore di approssimazione è piccolo.

La figura 5.9 mostra questa procedura. I punti di controllo D_1, D_2, D_3 in figura 5.9.a sono punti di controllo in $D_{i,n}$ la cui lunghezza è maggiore della

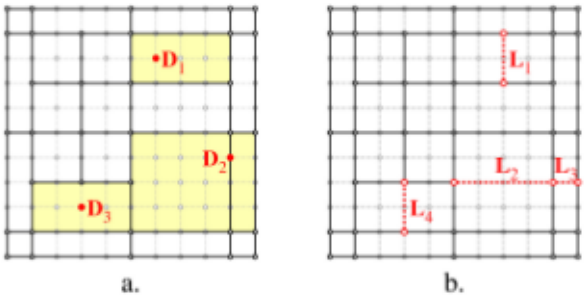


Figura 5.9: Identificazione e divisione di facce offensive in S_i.

soglia (l'errore di approssimazione). Le quattro facce colorate di giallo nella figura 5.9.a contengono uno di questi tre punti, e vengono segnate come da dividere.
Ci sono vari modi per dividere le facce, un caposaldo è però quello di dividerle, in generale, *all'incirca* a metà, nella direzione per la quale la faccia ha il maggior numero di linee nodali. Più nel dettaglio, se una faccia ha m linee nodali interne nella direzione s, e $n < m$ linee nodali interne nella direzione t, allora la divisione va effettuata nella direzione s, e attraverso la linea nodale numero $\frac{m+1}{2}$.
Nell'esempio sopra, la faccia che contiene D_3 in figura 5.9.a viene divisa attraverso la linea L_4 in figura 5.9.b, e la faccia che contiene D_1 viene divisa tramite la linea L_1; D_2 sta sulla linea di delimitazione tra due facce, entrambe da dividere. La faccia contenente L_2 possiede tre linee nodali interne nella direzione s e tre nella direzione t, per cui si sarebbe potuta scegliere indifferentemente anche l'altra direzione per dividerla.
Parlando di divisione di una faccia, intediamo l'effettuazione di un raffinamento locale in cui i due estremi del segmento che si trova a dividere la faccia vengono inseriti all'interno della T-mesh utilizzando la fase topologica dell'algoritmo di inserimento locale visto in § 3.2.3. L'algoritmo di raffinamento locale può necessitare l'inserzione di alcuni punti di controllo in più all'interno della T-mesh, al fine di soddisfare la regola; il ricorso ad esso è necessario al fine di avere la garanzia di formare una sequenza incapsulata di spazi T-spline.
Questo metodo ravvisa alcune similarità con un metodo basato sulla scomposizione wavelet delle B-spline [11], con il vantaggio però di poter dividere soltanto le facce dove l'errore è troppo grande e quello di non necessitare il calcolo esplicito della scomposizione. La figura 5.10 mostra un confronto tra l'applicazione dell'algoritmo di semplificazione T-spline che abbiamo appena visto (a destra), e la costruzione di una T-spline a partire dalla scomposizione wavelet B-spline (a sinistra). Osserviamo che il metodo diretto richiede soltanto 1109 punti di controllo, mentre quello precedente ne necessita di 1926, il 74% in più.

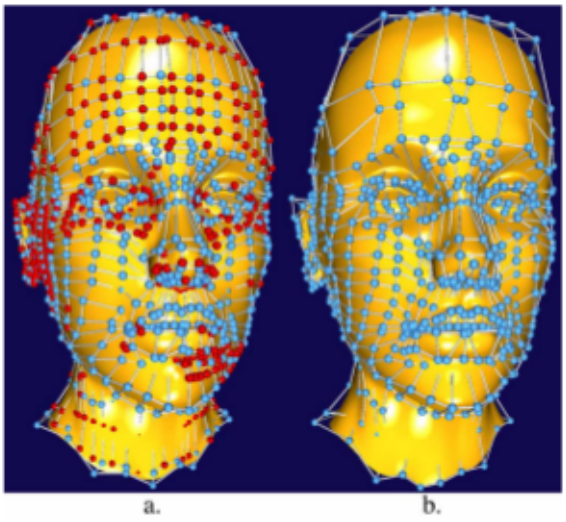


Figura 5.10: La semplificazione T-spline diretta (b.) a confronto con la scomposizione wavelet B-spline (a).

5.3 Confronto tra T-spline e NURBS

Introducendo le T-spline nel capitolo 1, una delle prime cose che abbiamo detto è stata che esse permettono di ridurre, anche notevolmente, il numero di punti di controllo necessari per rappresentare una superficie ad un certo livello di dettaglio. Andiamo dunque a vedere alcuni esempi pratici in tal senso.
Le figure 5.11, 5.12 e 5.13 mostrano tre modelli (cortesia di Zygote Media Group, Inc.) resi sia come NURBS, che come T-spline, applicando l'algoritmo di semplificazione (§ 5.2) sulle NURBS. Osserviamo che in tutti e tre i casi la resa attraverso le T-spline richiede un numero di punti di controllo notevolmente inferiore, circa la metà.

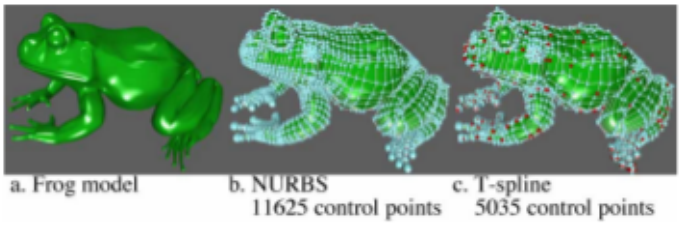


Figura 5.11: Modello di una rana.

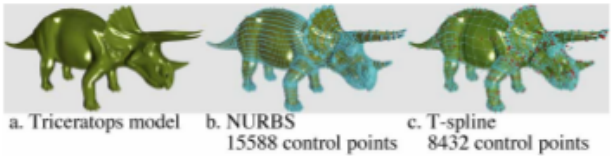


Figura 5.12: Modello di un triceratopo.

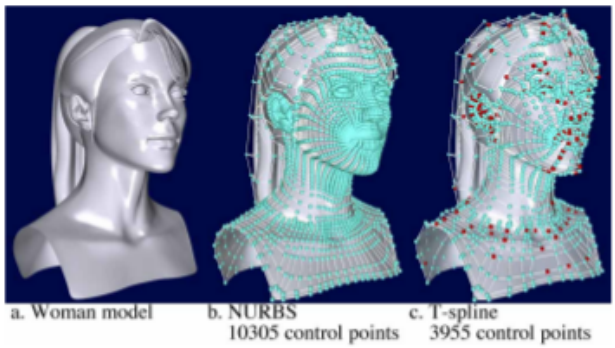


Figura 5.13: Modello di una donna.

La figura 5.14 mostra invece un'applicazione leggermente differente: invece di considerare una conversione da un modello `NURBS` ad uno T-spline, si sviluppa direttamente un modello come T-spline, in modo tale da minimizzare ad ogni passo della creazione il numero di punti di controllo superflui, per poi convertirlo alla fine come `NURBS`, in modo tale da averlo disponibile in entrambi i formati. Anche in questo caso, il vantaggio della rappresentazione attraverso il nuovo paradigma appare evidente.

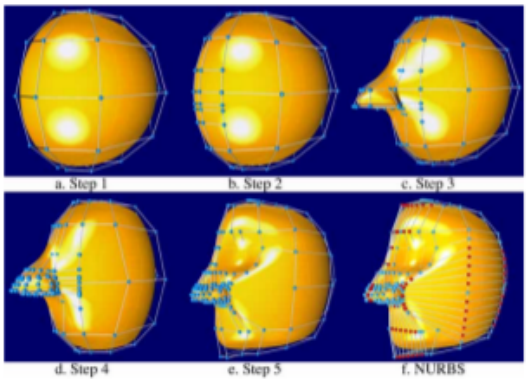


Figura 5.14: Modello di una testa: passi iniziali di creazione utilizzando le T-spline. (f) mostra il risultato della conversione di (e) in `NURBS`.

5.4 Oltre le T-spline: le superfici T-NURCC

Pur costituendo un ambito di ricerca e di lavoro recente, le T-spline sono a loro volta state già generalizzate in una classe di superfici più ampia, nota come superfici T-NURCC o più brevemente T-NURCCs, dove l'acronimo sta per *T-Non-Uniform Rational Catmull-Clark Surfaces*. Si tratta di una generalizzazione che prende spunto, finendo con il comprenderle anche in sé, dalle superfici di Catmull-Clark, una classe di superfici che si ottengono come limite di un procedimento di suddivisione; in particolare, rendendo più generali in questo modo le cose, un programma di modellazione in grado di gestire le T-NURCCs, permetterà sia un approccio di tipo spline, sia uno basato su di un raffinamento iterativo del modello.
Trattiamo ora una loro caratterizzazione in relazione a quanto abbiamo visto finora per le T-spline.
Innanzitutto, il discorso sugli intervalli tra nodi sviluppato nel § 1.3 vale anche per esse.
Ora, in § 5.1 abbiamo menzionato le NURSS [8], limitandoci però a dire che si tratta di un concetto che supera quello di B-spline; affrontiamo ora con qualche dettaglio in più l'argomento.
Le NURSS, acronimo per *Non-Uniform Recursive Subdivision Surfaces*, sono delle superfici che generalizzano il concetto di superfici B-spline tensore prodotto non uniformi.
Mentre nel caso delle superfici B-spline vengono richieste alcune limitazioni, quali il fatto che tutte le facce devono possedere quattro lati, e che gli intervalli tra nodi su lati opposti di ogni faccia devono risultare uguali, nel caso nelle NURSS questi limiti vengono superati. Possono essere inoltre presenti alcuni vertici in cui non si incontrano esattamente quattro facce, che vengono detti *punti straordinari*.
Le NURSS costituiscono anche una generalizzazione delle superfici di Catmull-Clark: se tutti gli intervalli tra nodi sono uguali, una NURSS degenera in una superficie di Catmull-Clark.
Nelle NURCC, invece, si richiede che i lati opposti di ogni faccia composta da quattro lati (solo di esse, nulla si chiede sulle altre) abbiano gli stessi intervalli tra nodi, lasciando tutto il resto identico al caso delle NURSS.
Apparentemente si tratterebbe di un passo indietro, in quanto, mentre tra le B-spline e le NURSS abbiamo tolto delle condizioni, tra le NURSS e le NURCC ne aggiungiamo una. Invece, ciò permette di definire per le NURCC regole di raffinamento (suddivisione) stazionarie che permettono anche un raffinamento di tipo locale, a differenza del caso NURSS che non permette alcuna delle due cose. [8]
Le T-NURCC sono NURCC che ammettono la presenza di T-giunzioni nella loro griglia di controllo, similmente a quanto le T-spline sono B-spline e NURBS che le permettono.

5.4.1 Raffinamento locale e T-giunzioni

Le regole di raffinamento per le NURCC sono identiche a quelle per le NURSS, con la richiesta, come abbiamo visto, che i lati opposti di ogni faccia composta da quattro lati abbiano gli stessi intervalli tra nodi.
Vediamo ora, concentrandoci sulle T-NURCC, come le T-giunzioni possano essere utilizzate per effettuare un raffinamento locale vicino ad un punto straordinario.
Definiamo innanzittutto il concetto di *valenza* di un vertice: questa quantità altro non è che il numero di facce che lo condividono. Si tratta di un concetto semplice, ma che si presenterà con una certa frequenza d'ora in avanti, e tramite il quale possiamo subito dare una definizione alternativa di punto straordinario: un punto è tale se è vertice di valenza diversa da quattro (potendo possederla di un valore superiore o inferiore ad esso).
Per semplificare la nostra discussione, inoltre, richiediamo che tutti i punti straordinari siano separati da almeno quattro facce, e che tutte le facce possiedano quattro lati; questi requisiti possono essere eventualmente raggiunti effettuando un certo numero, solitamente piccolo, di passi di raffinamento globale.
Da questo punto in avanti, tutto il raffinamento può essere effettuato a livello locale. Per esempio, ogni sottogriglia regolare e di larghezza opportuna di una griglia di controllo di una NURCC può essere sottoposta ad inserzione locale di nodi, e anche il raffinamento vicino ad un punto straordinario può essere svolto a livello locale, ovvero, in termini qualitativi, restando nelle "vicinanze" del punto stesso.
Per spiegare ciò, descriviamo prima un metodo per effettuare l'inserzione locale di nodi nelle vicinanze di un singolo vertice di valenza quattro in una T-spline. Con riferimento alla figura 5.15, iniziamo con la griglia di controllo di colore nero. Ora, è possibile utilizzare la procedura di inserimento di punti di controllo per inserire tutti i punti rossi nella riga 1, procedendo poi di seguito con la riga 2, la colonna 3, e la colonna 4. Poi i punti verdi possono essere inseriti, seguiti dai blu.
Ciò che si ottiene è un raffinamento locale nelle immediate vicinanze di un punto di controllo nero, e notiamo che questo schema di raffinamento può dividere le facce secondo un rapporto arbitrario ρ. Per un punto di valenza 4, cambiare ρ non cambia la superficie limite, dal momento che ciò che viene fatto si riduce all'inserimento di nodi in una B-spline, ma quando questo schema viene applicato al caso dei punti straordinari, ρ si trova ad essere un parametro di forma, la cui modifica porta ad effetti come quelli della figura 5.16, in cui, assieme alla griglia di controllo iniziale in alto a sinistra, vengono rappresentate tre superfici limite, rispettivamente per $\rho = 0.1$ (in alto a destra), $\rho = 0.5$ (in basso a sinistra), $\rho = 0.9$ (in basso a destra).
Consideriamo ora il problema del raffinamento locale per T-NURCC nei pressi di un punto straordinario isolato, facendo riferimento alla figura 5.17.

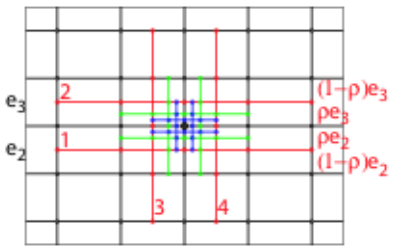


Figura 5.15: Raffinamento locale vicino ad un punto di controllo di valenza 4.

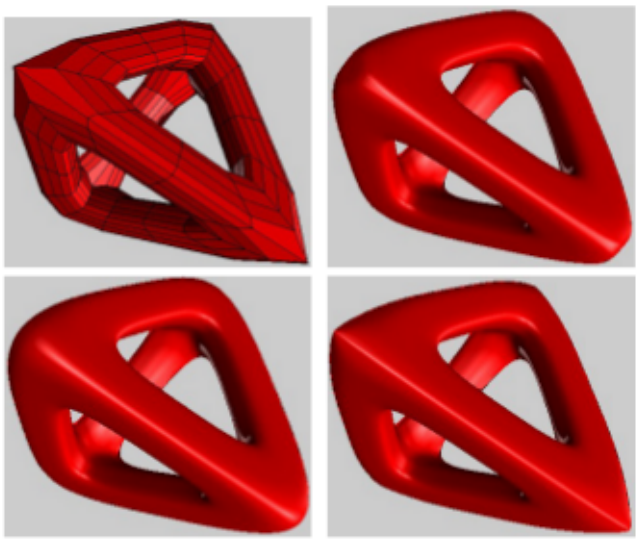


Figura 5.16: Valori distinti di ρ portano a superfici diverse.

L'intervallo tra nodi d_1 viene diviso negli intervalli tra nodi ρd_1 e $(1-\rho)d_1$, e lo stesso viene svolto per ogni altro intervallo tra nodi adiacente al punto straordinario. Se $\rho = \frac{1}{2}$ e se tutti i gli intervalli tra nodi iniziali sono uguali, la superficie limite che si ottiene utilizzando questo raffinamento locale è equivalente ad una superficie di Catmull-Clark.
Le lettere minuscole si riferiscono agli intervalli tra nodi, mentre quelle maiuscole ai punti. I vertici A, B, C, D, Q, R, S, T sono i punti di controllo iniziali, prima del raffinamento. Dopo il raffinamento, questi vertici vengono rimpiazzati da nuovi vertici denotati con apici: A', B', C', D', Q', R'. Si ottengono le seguenti equazioni, a partire da risultati noti sulle `NURSS`/`NURCC` ([8]) e

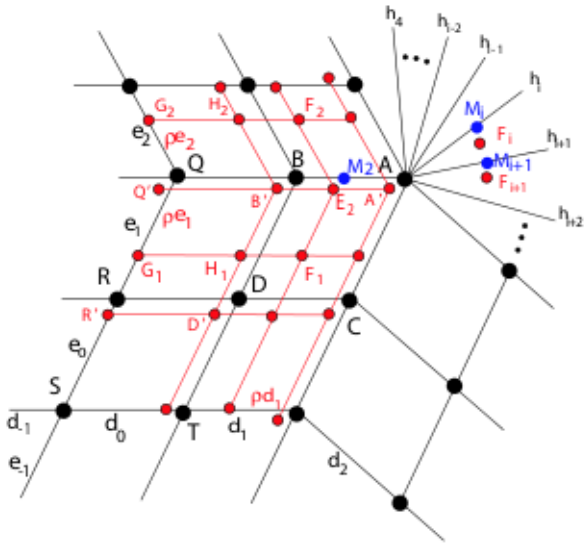


Figura 5.17: Raffinamento locale vicino ad un punto di controllo di valenza n.

applicando ripetutamente le formule viste nel § 3.1:

$$\begin{aligned} F_1 &= \frac{[e_0+(1-\rho)e_1][(d_0+(1-\rho)d_1)A+(\rho d_1+d_2)B]}{(d_0+d_1+d_2)(e_0+e_1+e_2)} \\ &+ \frac{[\rho e_1+e_2][(d_0+(1-\rho)d_1)C+(\rho d_1+d_2)D]}{(d_0+d_1+d_2)(e_0+e_1+e_2)} \end{aligned}$$

Ci sono tre tipi di punti sui lati: E, H e G.

$$E_2 = \rho M_2 + (1-\rho)\frac{e_2F_1+e_1F_2}{e_1+e_2}$$

dove:

$$M_2 = \frac{2\rho d_1+d_2+h_4}{2(d_0+d_1)+d_2+h_4}B + \frac{2d_0+2(1-\rho)d_1}{2(d_0+d_1)+d_2+h_4}A$$

$$\begin{aligned} H_1 &= \frac{\rho d_1[(\rho e_1+e_2)R+(e_0+(1-\rho)e_1)Q]}{(d_{-1}+d_0+d_1)(e_0+e_1+e_2)} \\ &+ \frac{[d_{-1}+d_0+(1-\rho)d_1][(\rho e_1+e_2)D+(e_0+(1-\rho)e_1)B]}{(d_{-1}+d_0+d_1)(e_0+e_1+e_2)} \end{aligned}$$

$$G_1 = \frac{\rho e_1+e_2}{e_0+e_1+e_2}R + \frac{e_0+(1-\rho)e_1}{e_0+e_1+e_2}Q$$

Ci sono cinque differenti tipi di punti di vertice: quelli che rimpiazzano A, B, Q, D, R, che denoteremo allo stesso modo ma con un apice. Sarà:

$$A' = \rho^2 A + 2\rho(1-\rho)\frac{\sum_{i=0}^{n-1} m_i M_i}{\sum_{i=0}^{n-1} m_i} + (1-\rho)^2\frac{\sum_{i=0}^{n-1} f_i F_i}{\sum_{i=0}^{n-1} f_i}$$

dove n è la valenza, $m_i = (h_{i-1} + h_{i+1})(h_{i-2} + h_{i+2})/2$, $f_i = h_{i-1}h_{i+2}$.

$$
\begin{aligned}
B' &= (1-\rho)\frac{e_1 H_2 + e_2 H_1}{e_1 + e_2} + \rho\left[\frac{\rho d_1}{d_{-1} + d_0 + d_1} Q + \frac{d_{-1} + d_0 + (1-\rho)d_1}{d_{-1} + d_0 + d_1} B\right] \\
Q' &= \rho Q + (1-\rho)\frac{e_1 G_2 + e_2 G_1}{e_1 + e_2} \\
R' &= \frac{\rho e_1}{e_{-1} + e_0 + e_1} S + \frac{e_{-1} + e_0 + (1-\rho)e_1}{e_{-1} + e_0 + e_1} R \\
D' &= \frac{\rho d_1 \rho e_1 S + [d_{-1} + d_0 + (1-\rho)d_1]\rho e_1 T}{(d_{-1} + d_0 + d_1)(e_{-1} + e_0 + e_1)} \\
&+ \frac{[\rho d_1 R + (d_{-1} + d_0 + (1-\rho)d_1)D][e_{-1} + e_0 + (1-\rho)e_1]}{(d_{-1} + d_0 + d_1)(e_{-1} + e_0 + e_1)}
\end{aligned}
$$

Il raffinamento locale in un punto straordinario è illustrato in figura 5.18, dove viene mostrata una `T-NURCC` su cui sono stati effettuati quattro passi di raffinamento locale. I punti gialli mettono in evidena quattro T-giunzioni. Notare che questa mesh raffinata localmente possiede soltanto 2496 facce; utilizzando il raffinamento globale delle superfici di Catmull-Clark, sarebbero state necessarie 393216 facce per giungere alla stessa precisione.

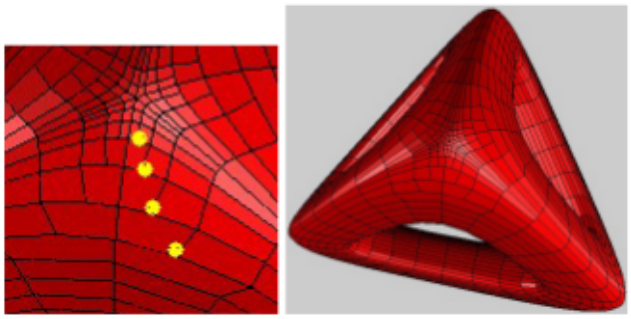


Figura 5.18: Una mesh di Catmull-Clark raffinata localmente.

In questa discussione abbiamo assunto che i vertici straordinari fossero separati da almeno quattro facce; ciò può essere sempre raggiunto, come già detto, compiendo un piccolo numero preliminare di passi di raffinamento globale. Sarebbe possibile anche ottenere un procedimento di raffinamento locale che non richiede i passi iniziali di raffinamento globale, ma ciò comporta il dover considerare casi particolari in più.
Eccetto che nei punti straordinari, le `NURCC` sono C^2, tranne nel caso di presenza di intervalli tra nodi nulli, che portano ad un abbassamento della continuità. Nei punti straordinari dove tutti i lati possiedono lo stesso valore come intervallo tra nodi, un'analisi degli autovalori per il caso di valenza pari a 3 mostra che la superficie è G^1 per ogni valore accettabile di $\rho(0 < \rho < 1)$. Un risultato simile può essere ottenuto per il caso di valenza pari a 5; per

valenze maggiori, la trattazione analitica può risultare più complicata, ma un campionamento di valori specifici per ρ non ha rivelato alcun caso in cui la continuità geometrica non fosse verificata. Anche se si fa venire meno l'ipotesi che tutti i lati possiedano lo stesso valore come intervallo tra nodi, le prove empiriche suggeriscono che la continuità G^1 continui a valere.

Capitolo 6

T-spline con Rhino

6.1 Introduzione su Rhino

Abbiamo già avuto ampiamente occasione di dire che uno degli scopi principali per il quale la teoria inerente i metodi numerici per la grafica è stata sviluppata, e viene tuttora ampliata, è quello dell'applicazione a, e del relativo utilizzo di, software per computer. Già nella prima metà degli anni '70, quando Chaikin introdusse il suo algoritmo di suddivisione, caposaldo nella costruzione di B-spline come limite di una procedura di raffinamento, l'impostazione del procedimento era di stampo informatico, senza trascurare da una parte il formalismo matematico, ma dall'altro ponendo in enfasi l'aspetto computazionale di una tale procedura.
Con la diffusione capillare dell'elettronica avvenuta in particolar modo dalla seconda metà degli anni '80 in poi, l'utilizzo di questo tipo di programmi è diventato usuale anche al di fuori degli ambienti specializzati in cui era prevalentemente diffuso in precedenza. Software di progettazione vengono utilizzati correntemente in ambito ingegneristico, architettonico, edilizio, ed il loro uso si estende anche ad aspetti lavorativi in cui ne viene svolto un uso indiretto, come il mondo del cinema e quello dei videogiochi.
Nonostante la notevole diffusione, uno dei limiti all'uso di questi software che resta, è costituito dall'elevato costo che le licenze di utilizzo usualmente possiedono; per alcuni di questi, si può arrivare anche a cifre dell'ordine della decina di migliaia di dollari/euro. Tuttavia, negli ultimi anni si stanno diffondendo interessanti alternative freeware, alcune delle quali funzionanti su sistemi operativi basati sul kernel di Linux, che ambiscono a permettere a chiunque di cimentarsi con l'uso di questo tipo di programmi, senza essere costretto a spendere cifre che, se già per un'azienda possono essere ritenute significative, per un privato possono scoraggiare del tutto dall'utilizzo.
In alternativa, alcuni produttori dei software commerciali rendono liberamente disponibili delle versioni di prova dei loro prodotti: utilizzabili per un periodo di tempo limitato, generalmente di 30 giorni, o in alternativa con disponibili soltanto le funzioni di base, permettono di usufruire sotto

tali condizioni del programma senza costi per l'utente. Tale pratica ha lo scopo di far conoscere al possibile cliente le potenzialità del prodotto, nella possibilità, per l'azienda che lo produce, che la controparte effettivamente finisca per acquistare il software; allo stesso tempo, ci si può fare un'idea di ciò che permette di fare il programma e delle sue varie funzionalità, senza che necessariamente si preveda di comprarlo. Questo permette in particolare un utilizzo didattico, potendo visionare ed usufruire di diverse tra le funzioni (se non tutte, per un periodo di tempo di solito sufficiente a farsi un'idea generale) anche senza fare riferimento al prodotto completo.
Considereremo nella fattispecie il software *Rhinoceros*, abbreviato comunemente in *Rhino*. Si tratta di un software per il `CAGD` della *Robert McNeel & Associates*, la cui prima versione è stata rilasciata nel 1993, mentre la versione considerata attuale è la 4.0, per la precisione il suo rilascio SR9, che porta la data del 24 marzo 2011. Esiste anche un rilascio stabile della 5.0, che però non ha ancora avuto una diffusione capillare.
La guida all'uso della versione 4.0 ([16]) descrive ampiamente le potenzialità di questo software, prodotto commerciale a licenza proprietaria. In particolare ci sono di interesse gli oggetti geometrici, strettamente legati a quanto affrontato nella teoria del corso; d'altro canto, il sottotitolo stesso del nome del programma è *`NURBS` modeling for Windows*, anche se per la precisione è in sviluppo pure una versione per Mac, il che aprirebbe tra l'altro la possibilità ad un rilascio anche per Linux, essendo entrambi sistemi operativi basati sull'architettura UNIX.
Nativamente, Rhino si limita a gestire curve e superfici `NURBS`, non contemplando il caso più avanzato delle T-spline. D'altro canto queste ultime, come abbiamo più volte avuto occasione di ribadire nel corso della nostra trattazione, sono uno strumento molto recente, sviluppato di fatto negli ultimi 10 anni o poco più, ancora più recente di un software come Rhinoceros, che abbiamo visto vedere la luce nella sua prima incarnazione all'inizio degli anni '90. Tuttavia, è disponibile un add-on che permette di lavorare anche con le superfici T-spline.

6.1.1 Superfici tagliate

Se si vogliono aggirare rapidamente le limitazioni della topologia rettangolare `NURBS`, è possibile tagliare una superficie, con degli opportuni strumenti proposti dal programma (vedi Appendice A). Sappiamo che le T-spline superano tale limitazione, ma essendo una teoria sviluppata molto di recente, non sono supportate (o lo sono solo parzialmente) da altri software, e pertanto non possono essere utilizzate in tutte le loro potenzialità in talune situazioni, specie se si necessita di convertire un file realizzato con Rhino in un altro formato (come vedremo nel § 6.3); questa soluzione alternativa, nativa di Rhino, può pertanto risultare utile.
Una superficie tagliata è composta di due parti: una superficie sottostante

che definisce la forma della geometria, e le curve di taglio che individuano i confini di taglio e definiscono la parte di superficie visibile. La parte rimanente, pur continuando ad essere presente nell'ambiente di lavoro, viene rimossa dalla vista. È possibile anche creare direttamente delle superfici tagliate, con altri appositi comandi.
Compiendo le operazioni precedenti, tuttavia la superficie risultante non è più necessariamente una NURBS, anzi in generale non lo è affatto, e ciò comporta una serie di limitazioni in quanto può essere possibile svolgere su tale superficie successivamente al taglio.

6.1.2 Curvatura

Alcune verifiche utili da compiere, attraverso strumenti del programma, sono quelle sul valore della curvatura in un punto di una curva, la visualizzazione della circonferenza di curvatura (proprietà, queste due, che ineriscono le continuità C^2 e G^2), e la verifica della continuità delle curve nei punti di break.
A tale scopo, è presente il comando **GraficoCurvatura**, che permette di visualizzare il grafico di curvatura di curve e superfici. Le linee del grafico rappresentano la direzione perpendicolare alla curva in quel dato punto, e la lunghezza della linea evidenzia il valore (in modulo) della curvatura.

6.2 Esempi applicativi dell'uso di T-spline in Rhino

Vediamo ora come utilizzare l'add-on per le T-spline in Rhino, incominciando a vedere come questo si presenta all'interno del contesto di un software NURBS-based.
Tutte le superfici T-spline sono totalmente compatibili con le NURBS, e sono indipendenti dalla risoluzione, il che significa che, a seconda delle impostazioni di tassellazione, è possibile ingrandire vicino come si desidera un oggetto e non vedere alcuna sfaccettatura, proprio come una superficie NURBS.
La compatibilità NURBS di T-spline permette ad un designer di mescolare varie tecniche di modellazione. è possibile creare una superficie T-spline con i suoi strumenti dedicati e quindi utilizzare qualsiasi strumento di Rhino su di essa, come booleane, raccordi, deformazioni o analisi. Ogni superficie T-spline può essere trattata come una polisuperficie di Rhino quando si eseguono i suoi comandi.
Le T-spline possono essere utilizzate per creare un intero modello, ma il software permette anche di utilizzarle per aggiungere componenti organici per modelli già esistenti di Rhino, per esempio realizzati tramite il paradigma NURBS.
L'add-on, sfruttando la struttura delle T-spline, fornisce agli utenti un modo facile e rapido di esplorare forme attraverso un sistema intuitivo, e può aiutare i progettisti ad accorciare i tempi di sviluppo tra concept e modelli

3D.
La figura 6.1 mostra la schermata iniziale di Rhino in cui sono state aperte le tre toolbox relative alle T-spline. Osserviamo che, per contraddistinguere le nuove icone da quelle standard del programma, ed in particolare da quelle relative alle superfici, la colorazione dominante in esse è un fucsia, in contrasto con il blu presente per la maggiore nelle icone delle superfici.
È importante specificare che questo add-on non fa distinzione tra T-spline e `T-NURCC`: per semplicità di linguaggio è stato deciso di chiamarlo semplicemente "T-splines" senza scendere nei particolari, ma in realtà utilizza indifferentemente le superfici T-spline semplici (e propriamente dette) come le loro evoluzioni `T-NURCC`; parlando formalmente, si potrebbe anche semplicemente dire che usi le superfici `T-NURCC`, dal momento che come abbiamo visto le superfici T-spline sono un loro sottoinsieme.
Questa caratterizzazione emerge in particolar modo nel momento in cui si converte una superficie `T-NURCC`, che non sia una T-spline semplice, in `NURBS`: dal momento che esse non permettono di rappresentare la topologia arbitraria di una `T-NURCC`, il programma divide la mesh in regioni che non contengono punti straordinari al loro interno, andando a formare rettangoli nei quali possono essere presenti punti straordinari al più agli angoli. Questi rettangoli vengono poi convertiti in `NURBS` singolarmente, facendo sì che il risultato della conversione sia una polisuperficie, piuttosto che una superficie singola. D'altro canto non sarebbe possibile convertire una superficie `T-NURCC` in un'unica superficie `NURBS`.
Per questo motivo, se non diversamente specificato, ogni volta che si parlerà di "T-spline" d'ora in avanti in questo capitolo, si intenderanno in generale T-spline semplici oppure `T-NURCC`.
Stabilito questo, possiamo dunque mostrare alcuni esempi applicativi relativi all'utilizzo dell'add-on, sempre con un particolare occhio di riguardo all'aspetto matematico. Un elenco dettagliato dei relativi comandi con rispettiva descrizione è argomento dell'appendice B.

Nel primo esempio proviamo a rendere appuntita una superficie liscia, ovvero a modificarla in maniera tale che si trovino ad essere presenti zone in cui la continuità sia soltanto C^0 (§ 3.1 e comando **Piega**).
Iniziamo realizzando una superficie liscia; generiamo dunque una **Sfera-Quad**, osservando che si tratta di una `T-NURCC` e non di una T-spline semplice, contenendo (almeno) un punto di valenza 3, come ben visibile dalla visuale prospettica delle figure che seguono.

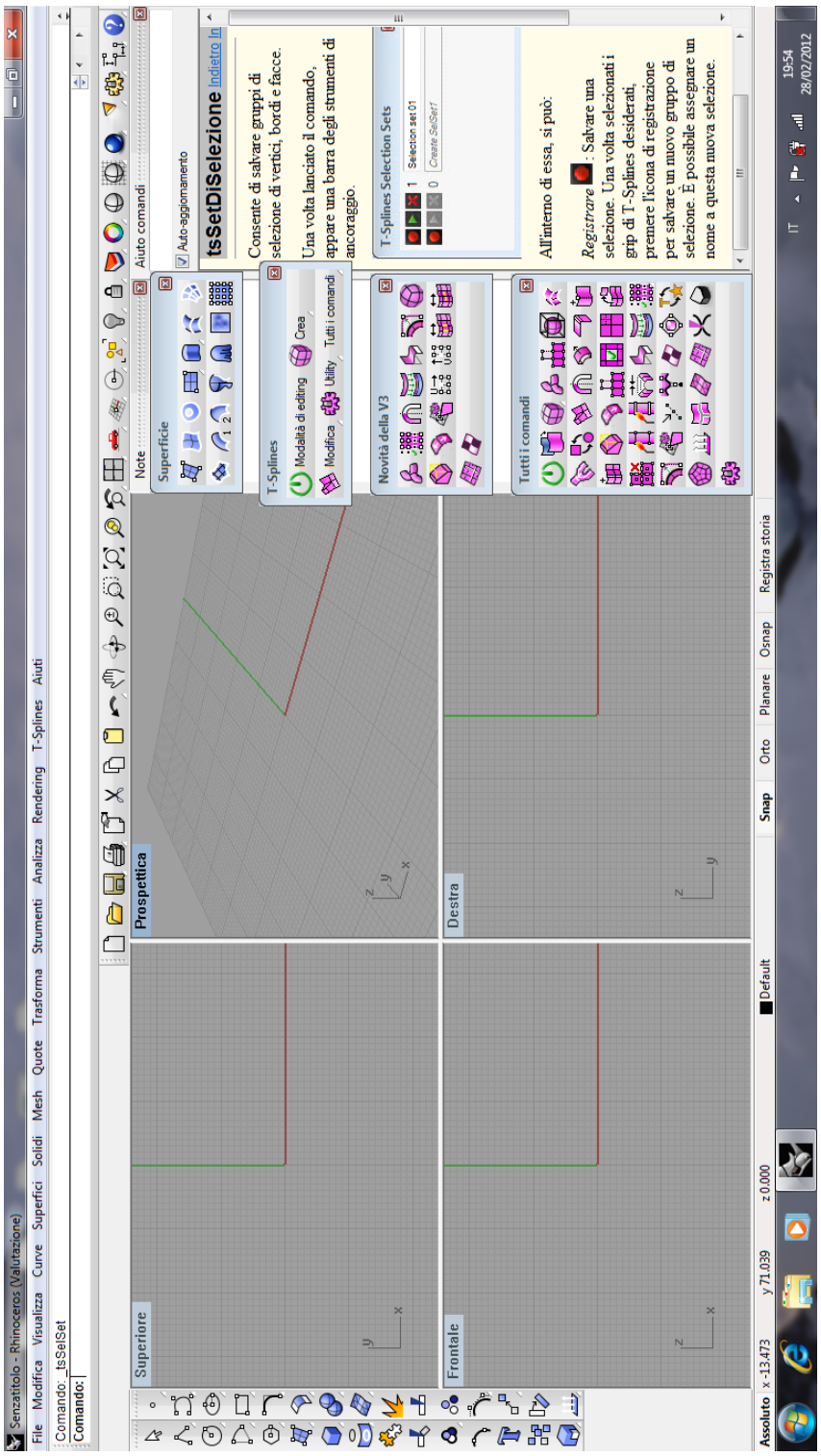


Figura 6.1: Le toolbox relative alle T-spline hanno le icone contraddistinte dalla colorazione fucsia, a differenza di quelle relative alle superfici tradizionali dominate dal blu.

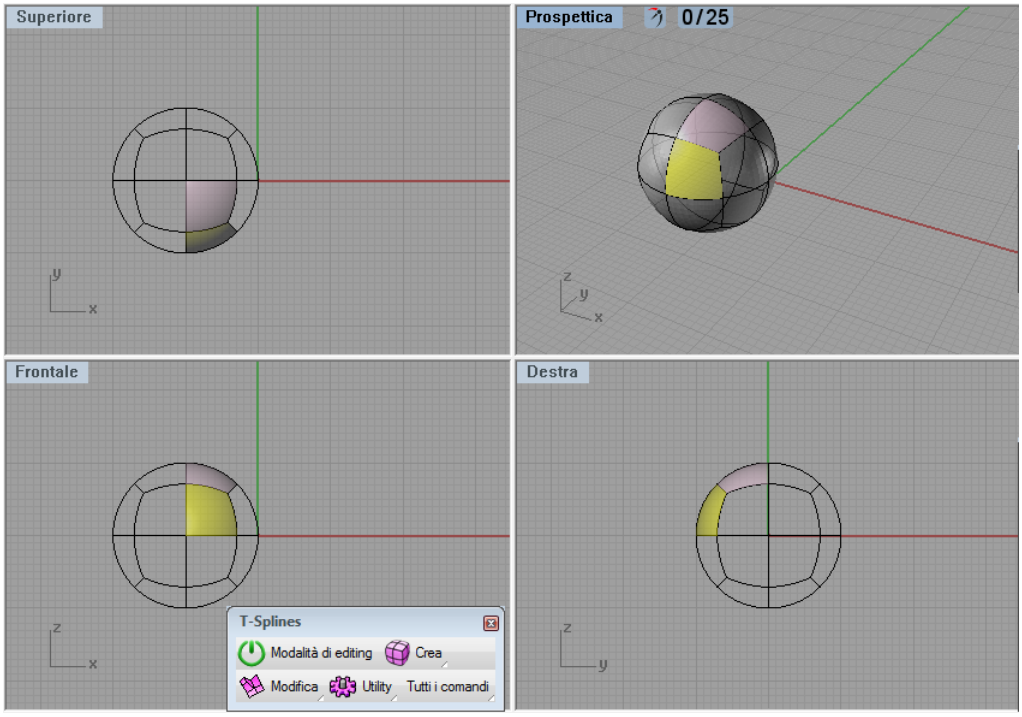


Figura 6.2: **SferaQuad**, con un paio di facce evidenziate.

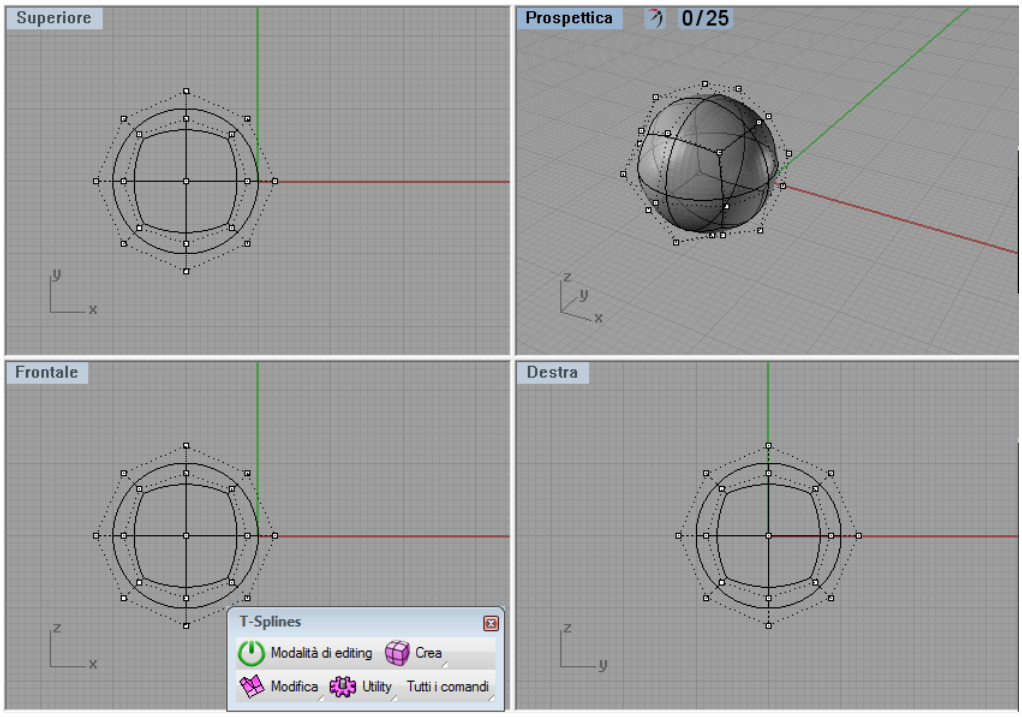


Figura 6.3: **SferaQuad**, con i punti di controllo in evidenza.

Selezioniamo ora un bordo, sul quale fare in modo che in sua corrispondenza la continuità si trovi ad essere C^0. A questo bordo applichiamo il comando **Piega**, ottenendo una modifica della forma della superficie.

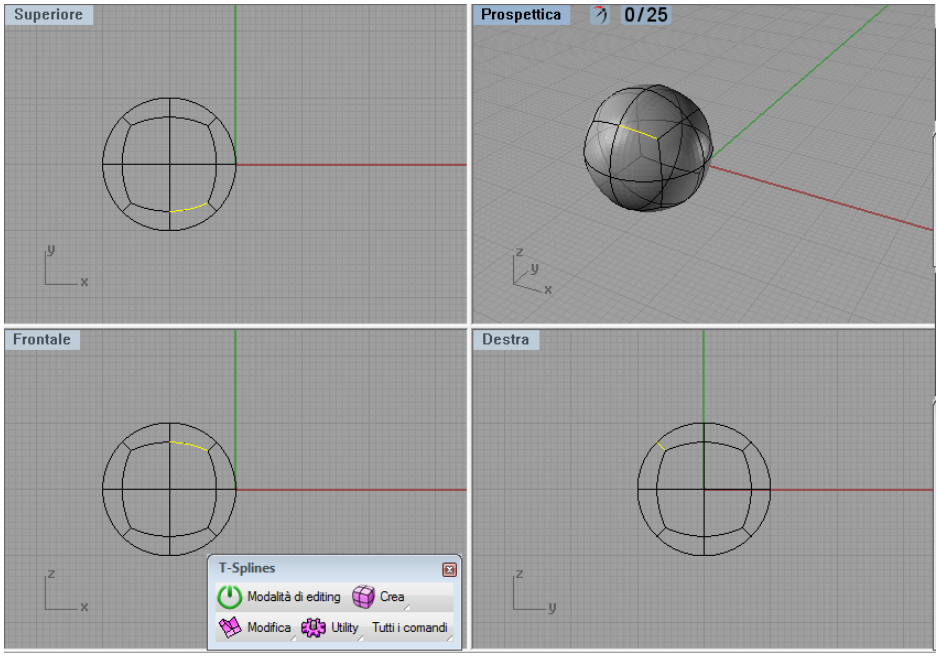
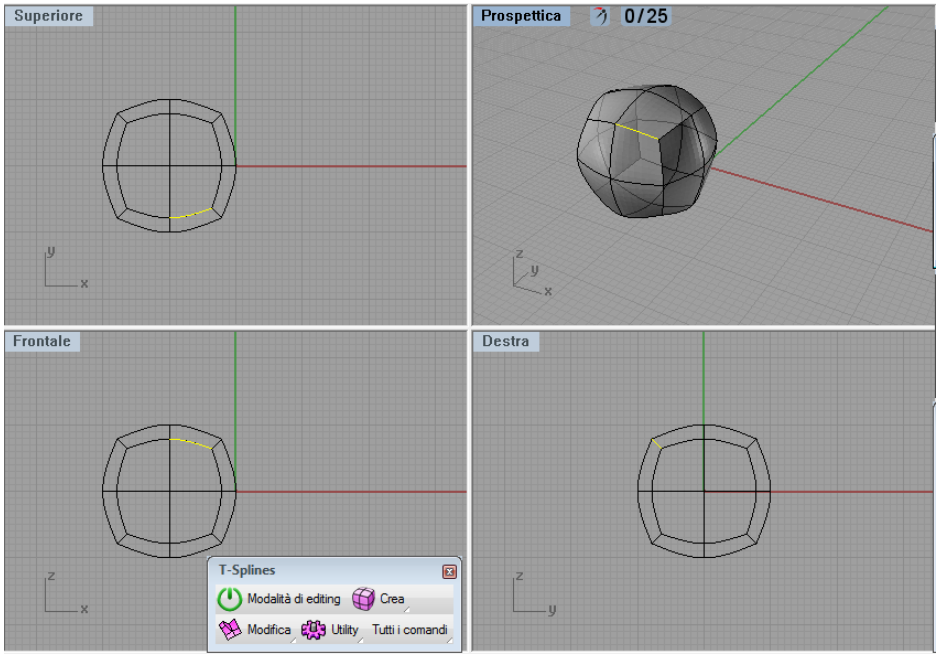


Figura 6.4: Un bordo della **SferaQuad** selezionato (sopra), e il corrispondente nella superficie trasformata (sotto).



Osserviamo che la perdita di continuità è locale: su altri bordi la superficie continua a restare almeno C^1 o comunque G^1, come visibile in figura 6.5, in cui la superficie è stata opportunamente ruotata rispetto alla figura 6.4.

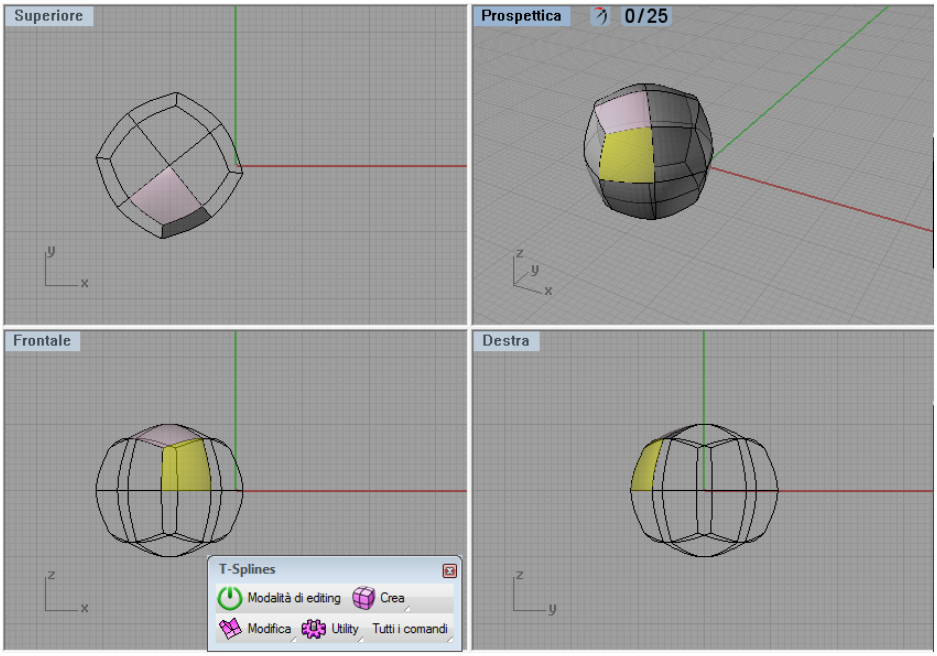
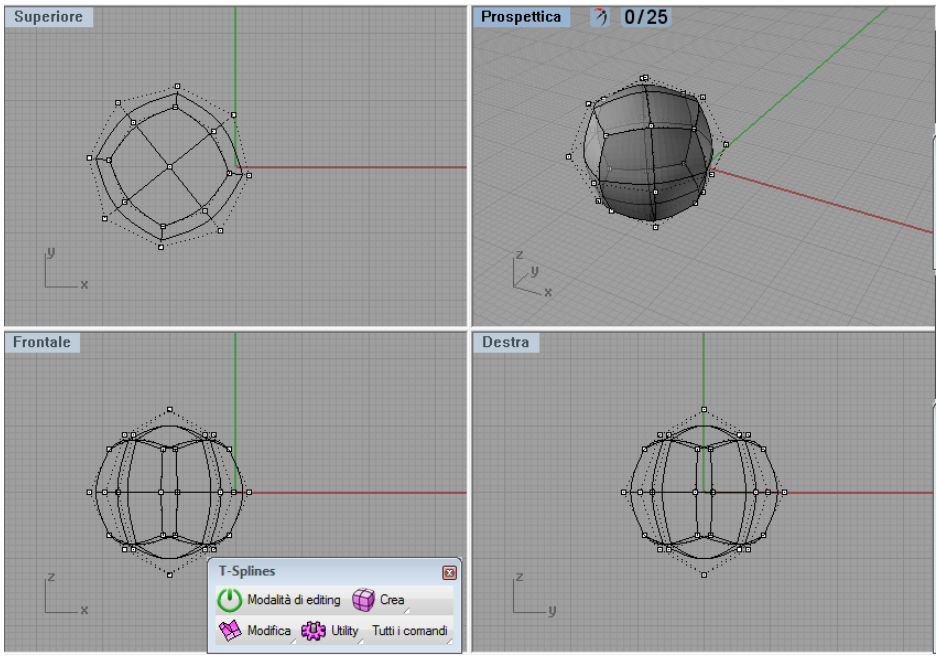
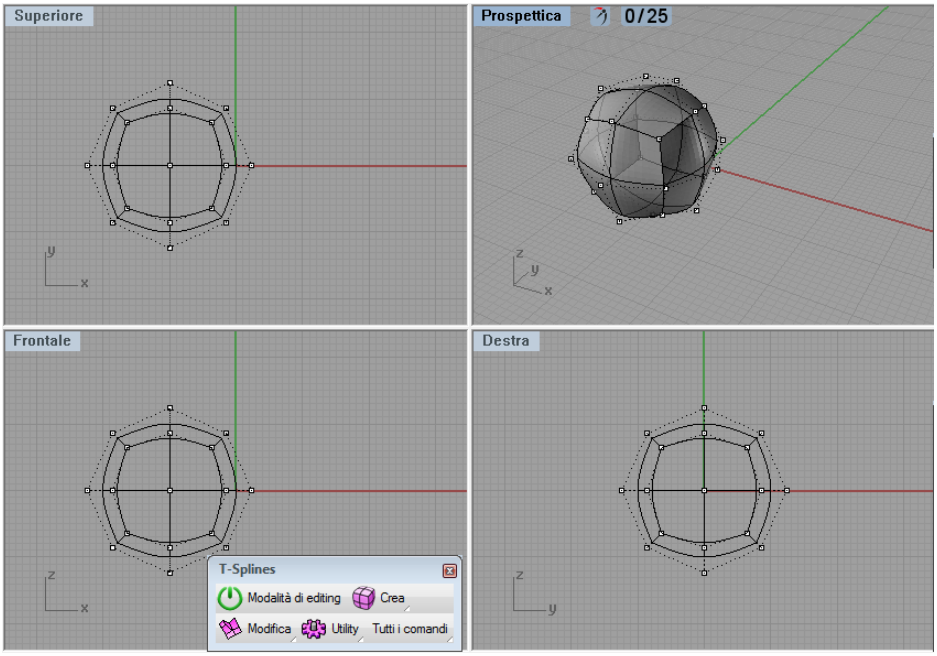


Figura 6.5: La perdita di continuità è locale.





La figura 6.6 mostra un altro esempio dell'applicazione di questa procedura. In questo caso è ben visibile il "rigonfiamento equatoriale" della superficie, in cui corrispondenza, almeno per la parte che sta di fronte, viene meno la continuità di tangenza, lasciando soltanto quella di posizione. Sugli "emisferi" invece viene mantenuta la continuità iniziale.

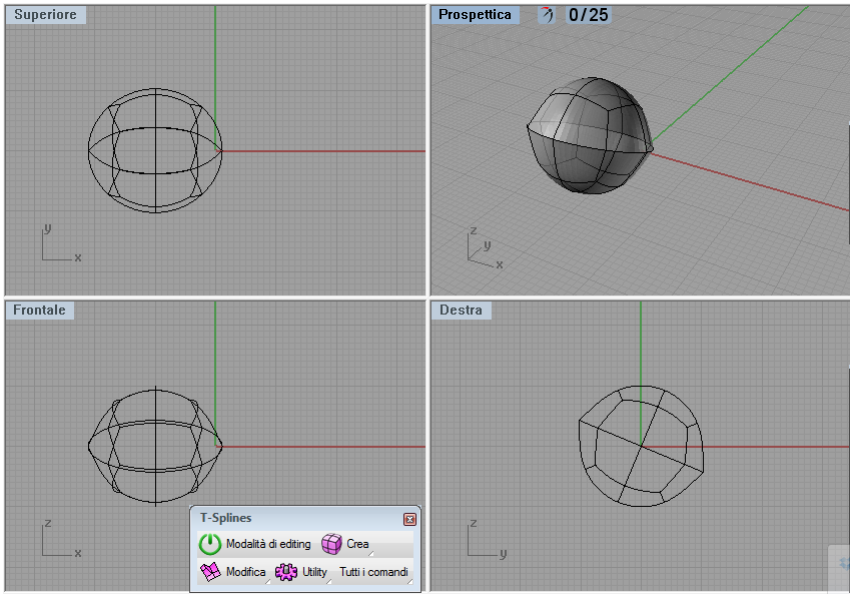


Figura 6.6: Un altro esempio.

Nell'esempio successivo effettueremo la modifica del peso di un punto di controllo. Considereremo ancora una sferaquad, prenderemo un suo punto di controllo, e cambieremo il valore del suo peso, fissato di default a 1, una volta aumentandolo a 5 (figure 6.7 e 6.8), una volta diminuendolo a 0.2 (figure 6.9 e 6.10). Vedremo in quale modo ciò influisce sulla forma della superficie.

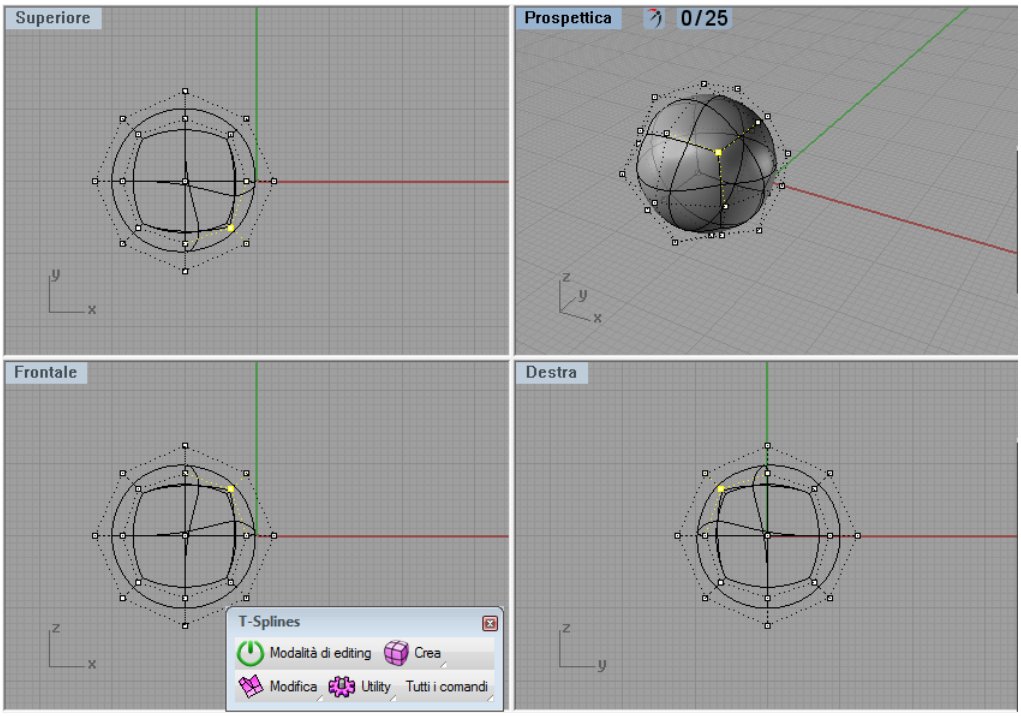


Figura 6.7: Il peso del punto di controllo selezionato è stato aumentato a 5. Si osserva, sia nella visuale in prospettiva che nelle proiezioni ortogonali, come le isocurve, rispetto alle figure 6.2 e 6.3 che mostrano una sferaquad non modificata, si avvicinino al punto, mentre le patch adiacenti lo stesso si rimpiccioliscano. Ciò è coerente con un "addensamento" della superficie nei pressi del punto, che implica anche il suo passaggio più vicino ad esso, in accordo con quanto accade compiendo l'analoga operazione nel caso delle `NURBS` o delle curve razionali in generale.

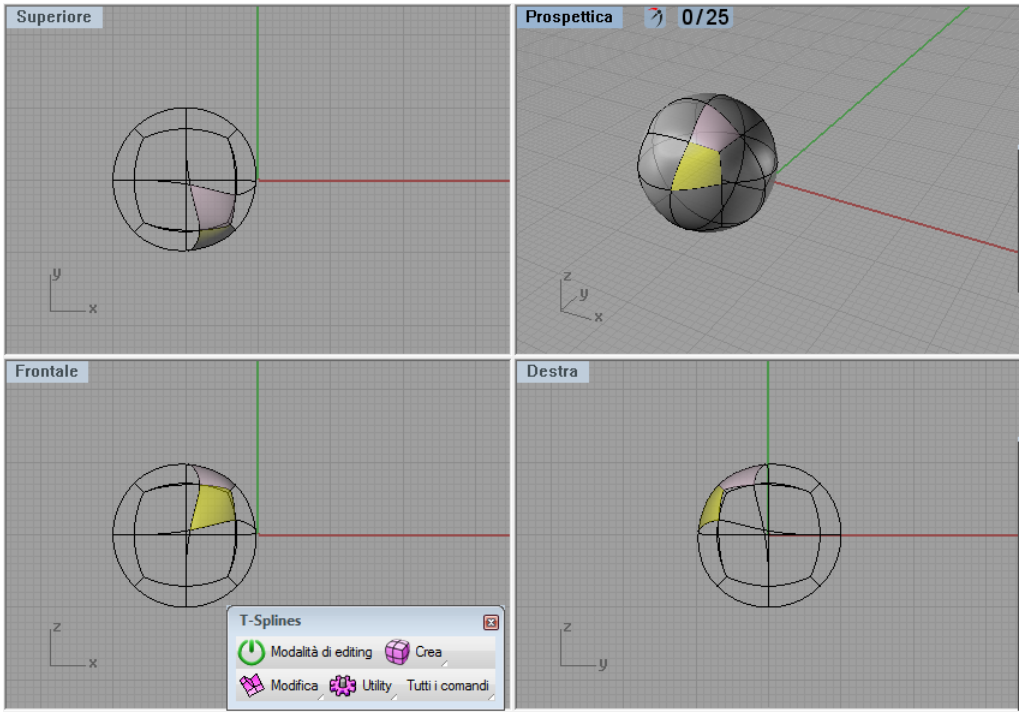


Figura 6.8: La situazione precedente, senza visualizzare i punti di controllo ed evidenziando un paio di facce; viene resa ulteriormente l'idea della modifica subita dalla superficie in seguito alla modifica del valore del peso del punto.

Nella figura 6.9, il peso del punto di controllo selezionato è stato diminuito a 0.2. Al contrario di prima, le isocurve si allontanano dal punto, mentre le patch adiacenti lo stesso si ingrandiscono. Ciò è coerente con una "rarefazione" della superficie nei pressi del punto, che implica anche il suo passaggio più lontano ad esso, in accordo con quanto accade compiendo l'analoga operazione nel caso delle `NURBS` o delle curve razionali in generale.

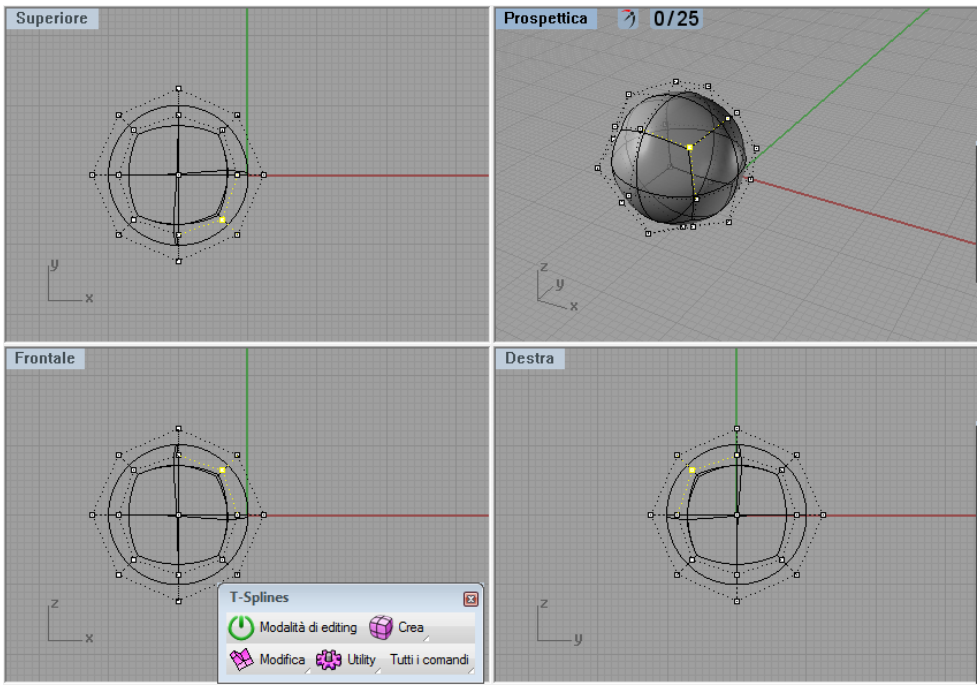


Figura 6.9: Il peso del punto di controllo selezionato è stato diminuito a 0.2.

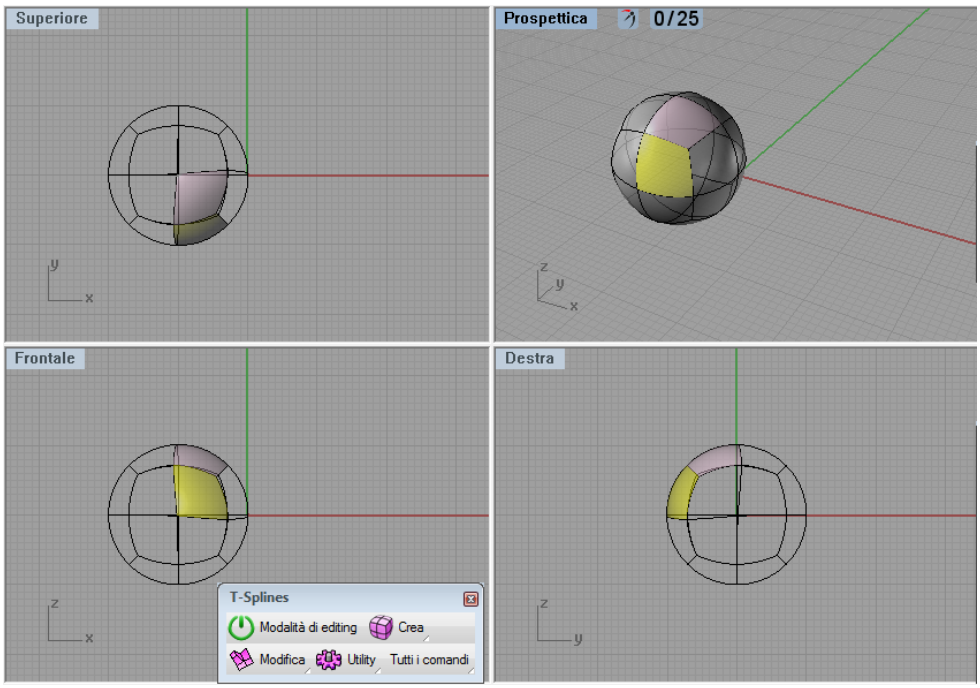


Figura 6.10: La situazione precedente, senza visualizzare i punti di controllo ed evidenziando un paio di facce.

In questo esempio proveremo invece una conversione da T-spline a NURBS di una sferaquad, osservando come la trasformazione nel formato nativo del programma renda la struttura dell'oggetto più complicata rispetto a quella risultante come T-spline. Rhino non è in grado di visualizzare esplicitamente i punti di controllo della sferaquad sotto forma di NURBS, in quanto la stessa contiene dei punti straordinari, e la sua conversione in NURBS può avvenire soltanto sotto forma di polisuperficie, con il software che in generale non rende possibile mostrare i punti di controllo di queste; tuttavia, la presenza in figura 6.11 di un numero di isocurve visualizzate ben maggiore rispetto alla figura 6.3, indica come di punti di controllo ne siano stati aggiunti, ed in quantità consistente.

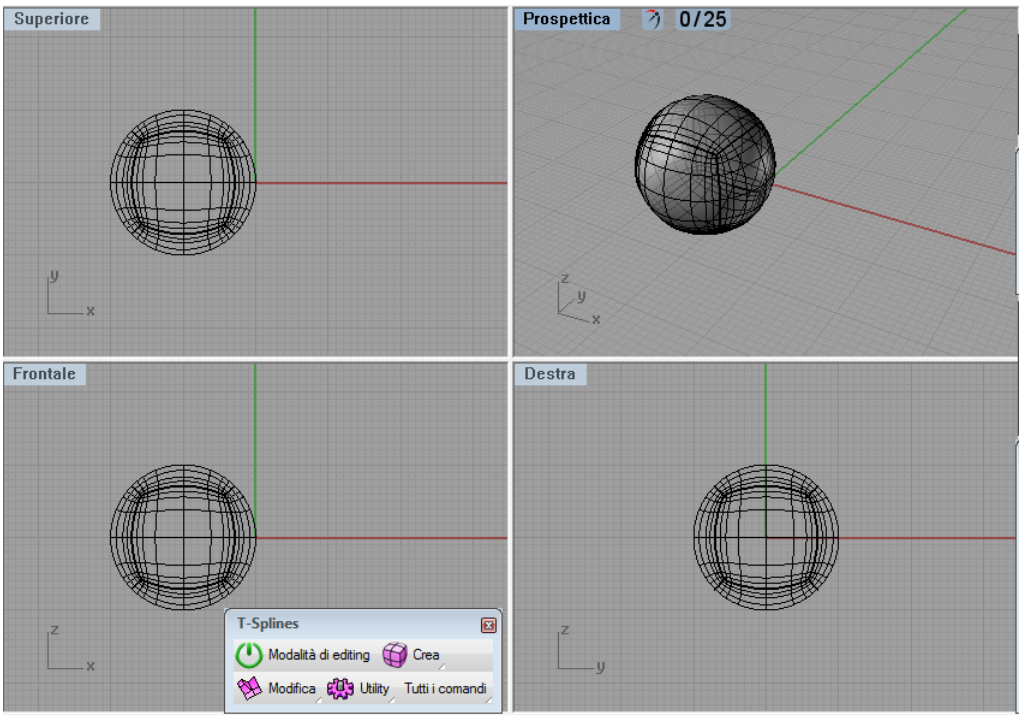


Figura 6.11: Conversione in NURBS.

Infine, come ultimo esempio, vedremo l'aggiunta di un punto di controllo.
La figura 6.12 mostra un inserimento semplice: in accordo con la teoria, Rhino si limita ad aggiungere il singolo punto evidenziato in giallo, senza inserirne o spostarne di altri, ma la superficie si modificha, perlomeno localmente.
La figura 6.13 mostra invece un inserimento esatto: il punto di controllo inserito è ancora evidenziato in giallo, ma vediamo che di punti di controllo ne vengono inseriti anche altri, in questo caso in gran numero, dovendosi mantenere le proprietà della sfera, quali per esempio chiusura e simmetrie.

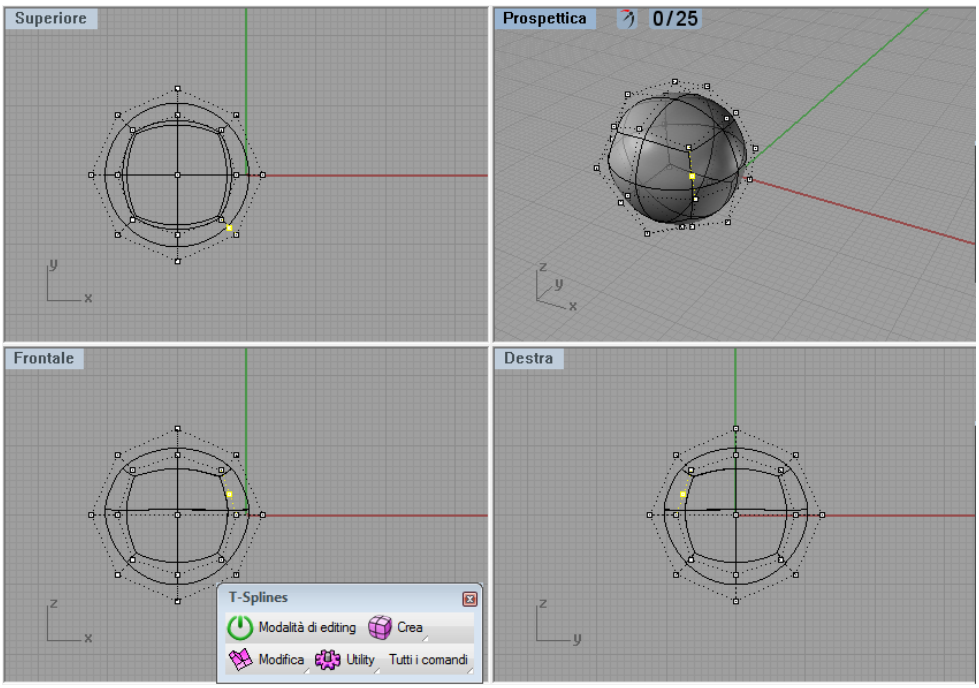


Figura 6.12: Inserimento semplice di un punto.

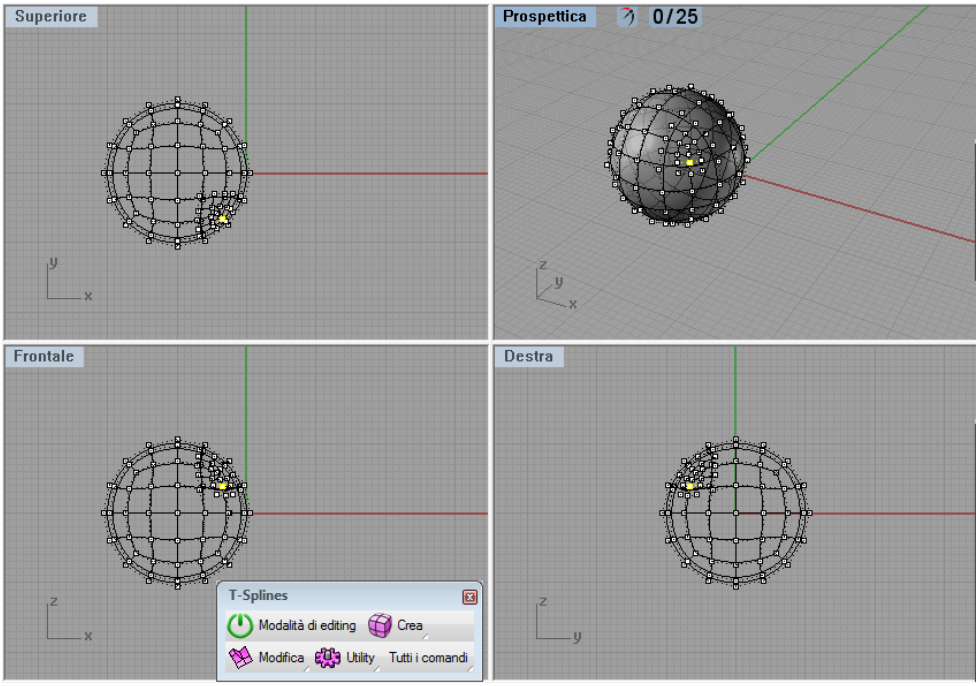


Figura 6.13: Inserimento esatto di un punto.

Quanto appena ottenuto può essere ripetuto per una superficie T-spline semplice, priva di punti straordinari: gli esempi precedenti sono stati compiuti tutti con una sfera per semplicità di visualizzazione, in quanto trattandosi di una forma elementare le deformazioni conseguenti alle operazioni compiute su di essa sono immediatamente visibili ad occhio, ma il discorso è generale. Nel caso di assenza di punti con valenza diversa da quattro, è possibile convertire la superficie T-spline in un'unica superficie `NURBS`; a tale scopo, consideriamo una curva `NURBS`, dalla quale, tramite il comando **Forma tubolare**, generiamo una superficie basata su di essa. Vedremo appunto il risultato di questa conversione, che in figura 6.11 era stata possibile soltanto sotto forma di polisuperficie composta da un certo numero di superfici distinte (sei, per la precisione).

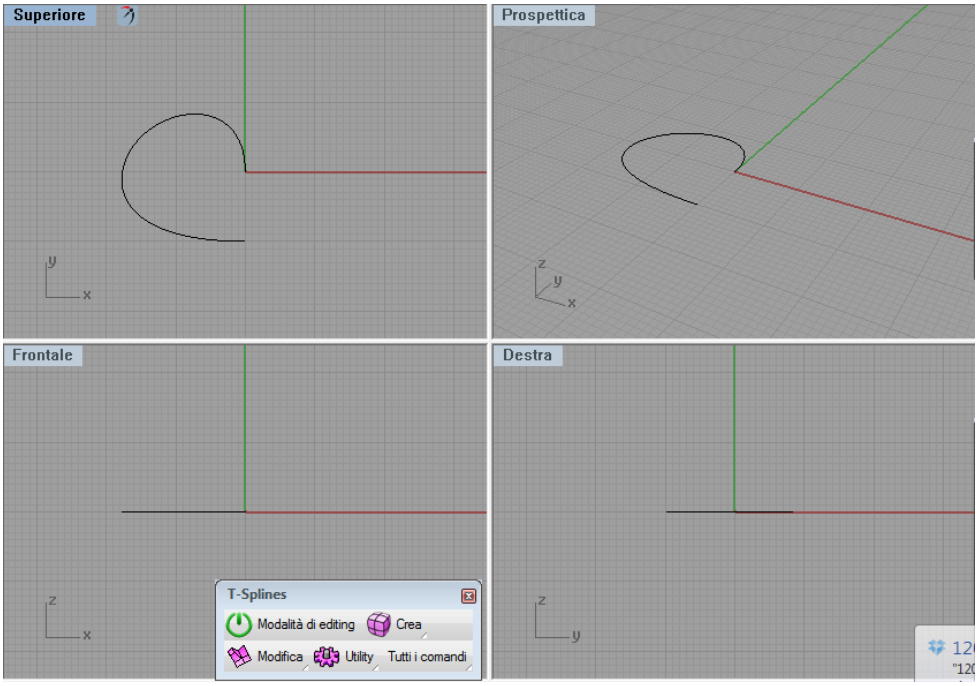


Figura 6.14: Curva `NURBS`.

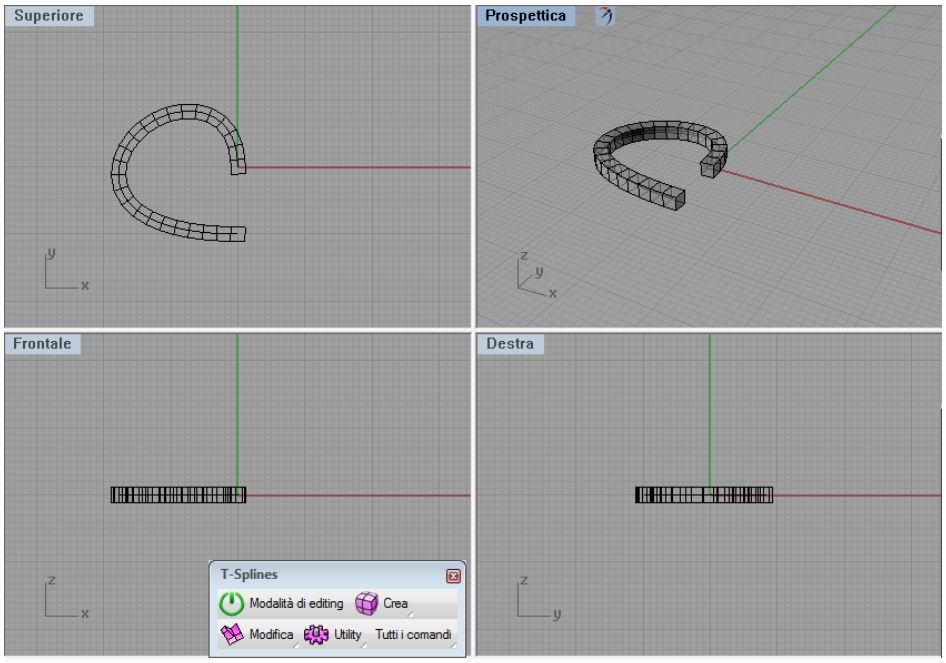


Figura 6.15: Superficie "tubolare" generata a partire dalla curva di figura 6.14, con la curva di partenza mantenuta nell'ambiente di lavoro.

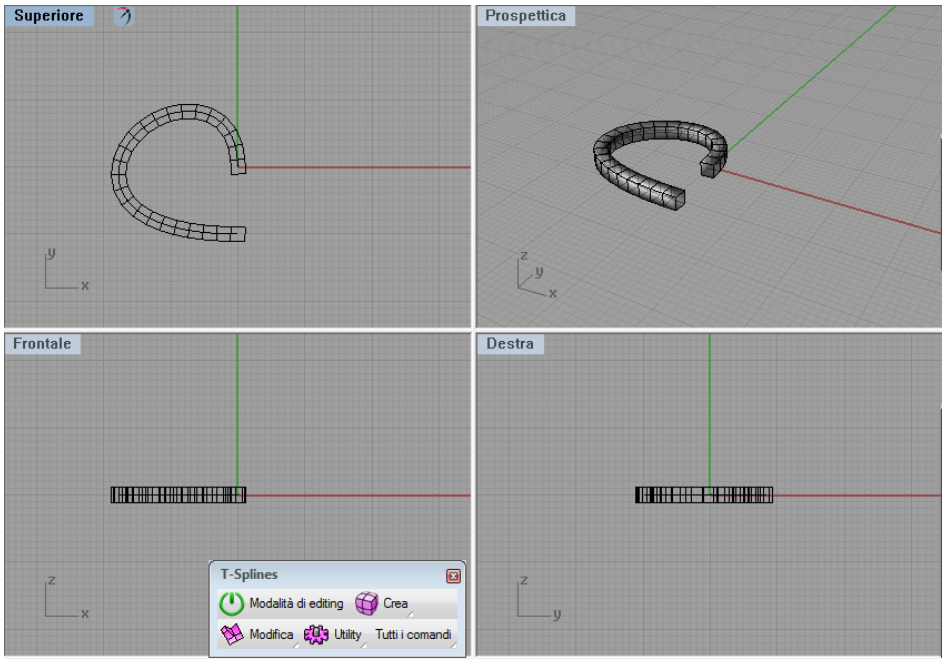


Figura 6.16: Conversione in superficie `NURBS` della superficie di figura 6.15. Si osservi la differenza di visualizzazione nella vista prospettica (una sorta di "addensamento visivo").

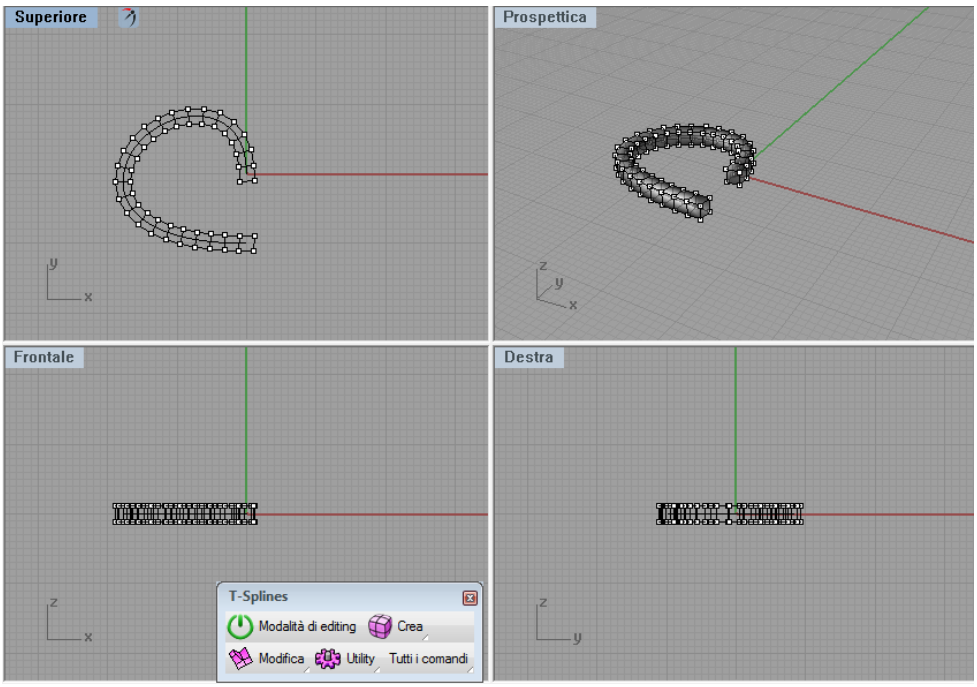


Figura 6.17: Visualizzazione dei punti di controllo della superficie di figura 6.16. La superficie è singola, altrimenti Rhino non permetterebbe di renderli visibili (come accaduto nel caso della sfera).

Se vogliamo rendere visibili gli intervalli tra nodi, ci viene utile un comando concepito per il debug ([20]):

```
_tsMacroUtil _LabelKnots
```

Le figure successive mostrano il risultato per una sferaquad, disegnata questa volta di dimensioni maggiori rispetto agli esempi precedenti per evitare l'addensamento delle etichette. In particolare: la figura 6.18 mostra semplicemente gli intervalli (che vediamo essere, in questo caso, tutti unitari); la figura 6.19 li mostra dopo l'inserimento semplice del punto selezionato (evidenziato in giallo), e vediamo che il lato intersecato viene diviso in due, ad entrambi i quali viene assegnato un intervallo di 1/2; le due immagini della figura 6.20 li mostrano dopo l'inserimento esatto dello stesso punto, e notiamo che propagandosi la modifica su tutta la superficie, tutti gli intervalli vengono almeno dimezzati, alcuni dei quali trovandosi ad essere ridotti ulteriormente (in corrispondenza di un maggior incremento locale del numero dei punti di controllo).

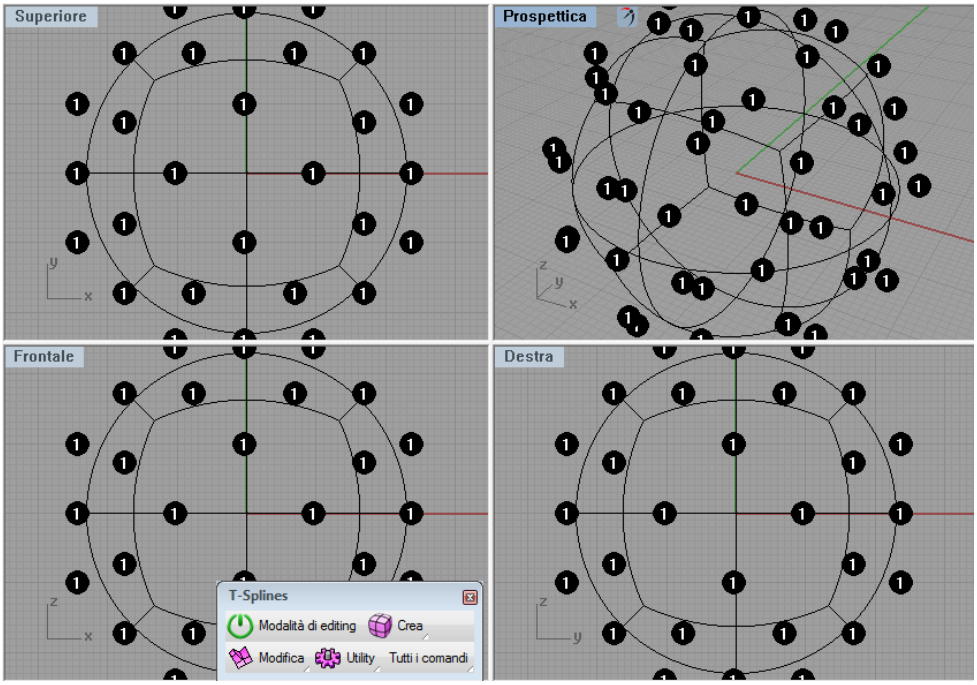


Figura 6.18: Intervalli tra nodi in una SferaQuad.

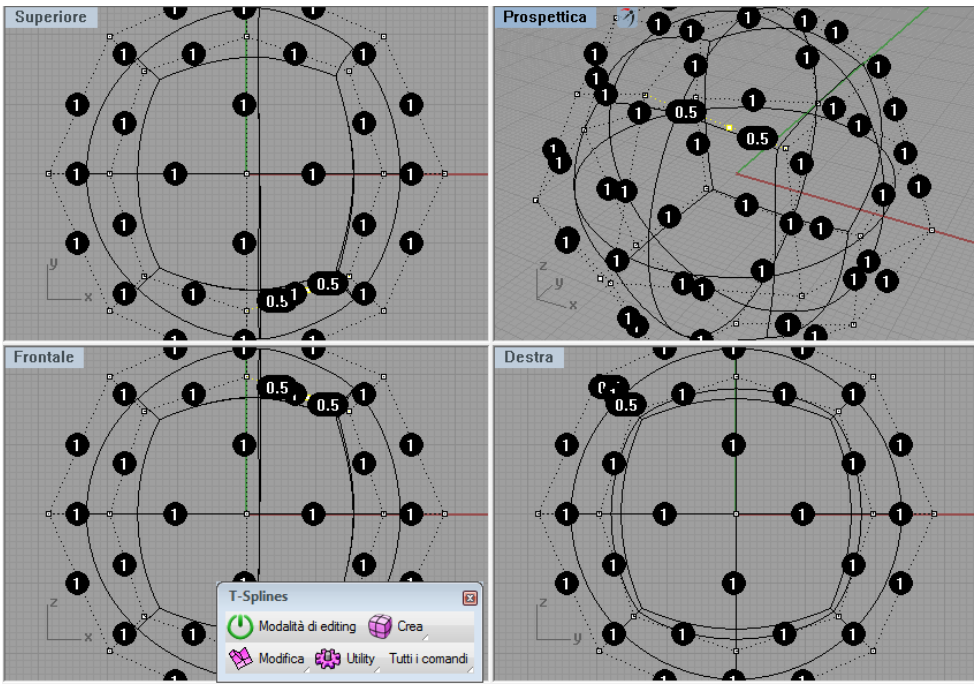


Figura 6.19: Intervalli tra nodi in una SferaQuad, con inserimento semplice di un punto.

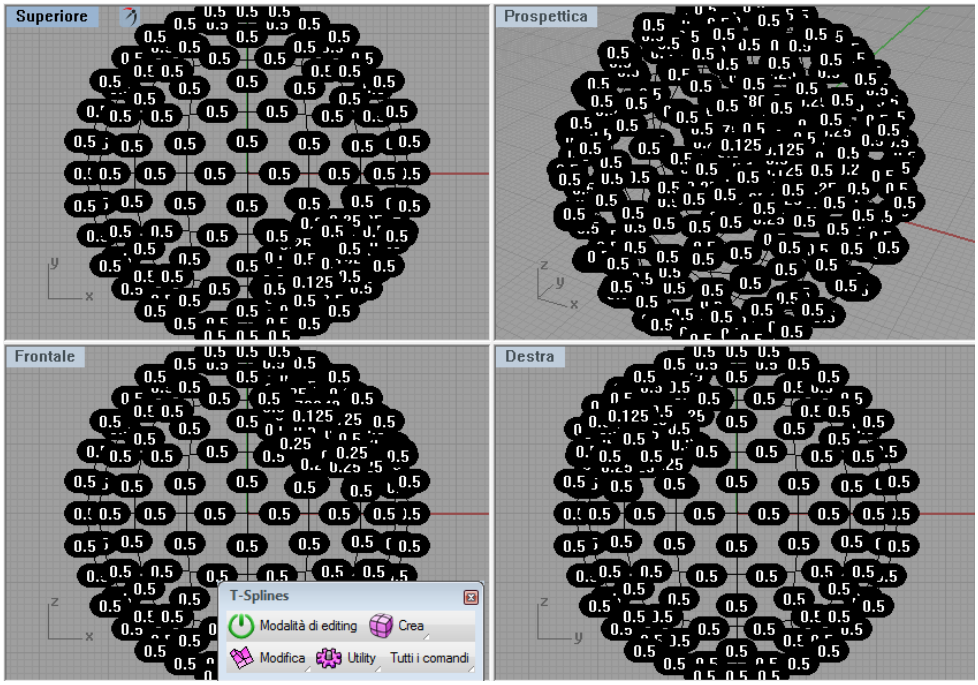
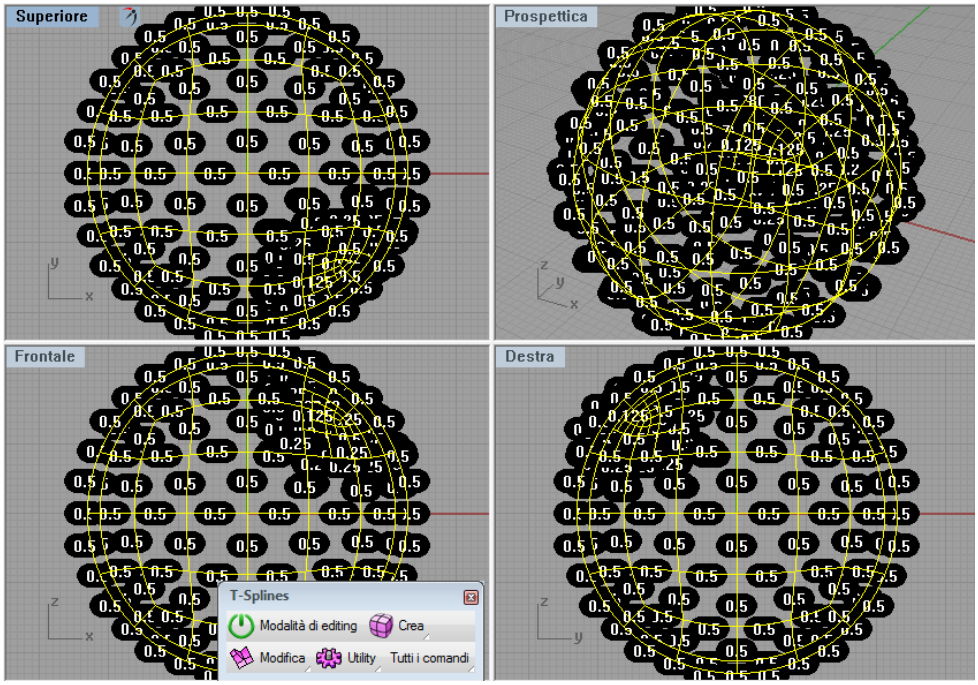


Figura 6.20: Intervalli tra nodi in una SferaQuad, con inserimento esatto.



6.3 Le proprietà delle entità geometriche

Benchè faccia uso nativamente di NURBS, e grazie all'add-on anche di T-spline, purtroppo Rhino, ma il discorso si può estendere anche alla maggior parte dei software di CAD, non fornisce la possibilità di leggere le proprietà delle entità geometriche come informazione esplicita. I motivi di questa limitazione sono principalmente due: la maggior parte degli utilizzatori di questi programmi sono disegnatori, quali possono essere ingegneri, architetti, o progettisti in genere, e ad essi in generale poco importa della matematica che sta alle spalle di ciò che si può realizzare, essendo loro interessati ad usare i software prettamente sotto l'aspetto pratico; inoltre, più dettagli vengono svelati su come vengono utilizzati gli strumenti teorici, e più si rende facile per un eventuale concorrente creare un software che possa andare a rivaleggiare con quelli già presenti, nuocendo potenzialmente al business delle case produttrici di questi programmi.
Tuttavia, è possibile reperire informazioni sulla struttura matematica degli oggetti indirettamente. Rhino infatti mette a disposizione la possibilità di salvare gli oggetti creati con esso in formati differenti da quello nativo, in maniera tale da permettere la compatibilità delle creazioni realizzate tramite esso con altri software. Diversi di questi formati sono *interpretabili*, ovvero la loro specifica è accessibile. In pratica questo implica che è possibile aprire i risultati della conversione anche con un semplice editor di testo e, conoscendo il modo in cui i dati vengono riportati, estrapolarli e ottenere le informazioni desiderate. ([23])
Vediamo nella pratica cosa significa questo. Consideriamo la SferaQuad utilizzata negli esempi precedenti, rappresentazione T-spline di una sfera, in questo caso centrata in $(-10, 0, 0)$ e di raggio 10. Esportandola nel formato STEP, generando un file .stp, otteniamo un output strutturato nel seguente modo. C'è un preambolo, contenente informazioni non correlate alla geometria:

```
ISO-10303-21;
HEADER;
/* Generated by software containing ST-Developer
 * from STEP Tools, Inc. (www.steptools.com)
 */
/* OPTION: using custom schema-name function */

FILE_DESCRIPTION(
/* description */ (''),
/* implementation_level */ '2;1');

FILE_NAME(
/* name */ 'rhino_sferaquad',
```

```
/* time_stamp */ '2012-03-18T23:09:46+01:00',
/* author */ (''),
/* organization */ (''),
/* preprocessor_version */ 'ST-DEVELOPER v10',
/* originating_system */ '',
/* authorisation */ '');

FILE_SCHEMA (('CONFIG_CONTROL_DESIGN'));
ENDSEC;

DATA;
```

Seguono poi tutte le informazioni riferite alla geometria, raggruppate in campi. Nella prima parte, da #10 a #75, vengono definite facce, lati, e vertici, con riferimento alle B-spline definite nei campi successivi, superfici (#76-#81) e curve (#82-#93). Negli argomenti, i numeri preceduti dal carattere '#' sono i richiami ad altri campi, ad esempio la stringa #180 costituisce un richiamo al campo 180. Riportiamo queste righe; tutti i campi omessi non sono stati riportati in quanto simili tra di loro, differenziandosi soltanto per i riferimenti a elementi diversi, e pertanto è stato trascritto solamente il primo di ogni tipo: per esempio, i campi dal #15 al #19 sono ancora del tipo `=ADVANCED_FACE(_,_,_,_=;)`, e cambiano soltanto i richiami interni.

```
#10=SHAPE_REPRESENTATION_RELATIONSHIP('','',#180,#12);
#11=MANIFOLD_SOLID_BREP('brep_1',#13);
#12=ADVANCED_BREP_SHAPE_REPRESENTATION($,(#11,#182),#179);
#13=CLOSED_SHELL($,(#14,#15,#16,#17,#18,#19));
#14=ADVANCED_FACE($,(#20),#76,.T.);
#20=FACE_OUTER_BOUND($,#26,.T.);
#26=EDGE_LOOP($,(#32,#33,#34,#35));
#32=ORIENTED_EDGE($,*,*,#56,.T.);
#56=EDGE_CURVE($,#68,#69,#82,.T.);
#68=VERTEX_POINT($,#1838);
```

Nel campo #76 viene definita una superficie. Vediamo come vengono strutturate le informazioni:

```
#76=(
BOUNDED_SURFACE()
B_SPLINE_SURFACE(3,3,((#488,#489,#490,#491,#492,#493,#494,#495,#496,#497,
#498,#499,#500,#501,#502),(#503,#504,#505,#506,#507,#508,#509,#510,#511,
#512,#513,#514,#515,#516,#517),(#518,#519,#520,#521,#522,#523,#524,#525
#526,#527,#528,#529,#530,#531,#532),(#533,#534,#535,#536,#537,#538,#539,
#540,#541,#542,#543,#544,#545,#546,#547),(#548,#549,#550,#551,#552,#553,
#554,#555,#556,#557,#558,#559,#560,#561,#562),(#563,#564,#565,#566,#567,
#568,#569,#570,#571,#572,#573,#574,#575,#576,#577),(#578,#579,#580,#581,
#582,#583,#584,#585,#586,#587,#588,#589,#590,#591,#592),(#593,#594,#595,
```

```
#596,#597,#598,#599,#600,#601,#602,#603,#604,#605,#606,#607),(#608,#609,
#610,#611,#612,#613,#614,#615,#616,#617,#618,#619,#620,#621,#622),(#623,
#624,#625,#626,#627,#628,#629,#630,#631,#632,#633,#634,#635,#636,#637),(#638,
#639,#640,#641,#642,#643,#644,#645,#646,#647,#648,#649,#650,#651,#652),(#653,
#654,#655,#656,#657,#658,#659,#660,#661,#662,#663,#664,#665,#666,#667),(#668,
#669,#670,#671,#672,#673,#674,#675,#676,#677,#678,#679,#680,#681,#682),(#683,
#684,#685,#686,#687,#688,#689,#690,#691,#692,#693,#694,#695,#696,#697),(#698,
#699,#700,#701,#702,#703,#704,#705,#706,#707,#708,#709,#710,#711,#712)),
.UNSPECIFIED.,.F.,.F.,.F.)
B_SPLINE_SURFACE_WITH_KNOTS((4,2,2,1,1,1,2,2,4),(4,2,2,1,1,1,2,2,4),(0.,
0.125,0.25,0.5,1.,1.5,1.75,1.875,2.),(0.,0.125,0.25,0.5,1.,1.5,1.75,1.875,
2.),.UNSPECIFIED.)
GEOMETRIC_REPRESENTATION_ITEM()
RATIONAL_B_SPLINE_SURFACE(((0.999999999999997,0.999999999999998,1.00437705575538,
1.00841741491422,1.00808071831766,0.999999999999996,1.,0.999999999999999,
1.,0.999999999999999,1.00808071831764,1.00841741491421,1.00437705575539,
0.999999999999997,0.999999999999997),(0.999999999999997,0.998540981414867,
1.00356264396289,1.00862409621542,1.00815637154096,1.,1.,0.999999999999999,
1.,1.,1.00800506509431,1.00851001947661,1.00354539529799,0.998540981414868,
0.999999999999997),(1.00437705575539,1.00354539529798,1.0048159344842,
1.00766503350888,1.00684022216753,1.,1.,0.999999999999999,1.,1.,1.00654234156365,
1.00749005056741,1.0048159344842,1.00356264396289,1.00437705575539),
(1.00841741491421,1.00851001947661,1.00749005056741,1.0095151017094,1.00918496219804,
1.,1.,1.,1.,1.,1.0090326466879,1.0095151017094,1.00766503350888,1.00862409621542,
1.00841741491421)(1.00808071831764,1.00800506509431,1.00654234156365,1.0090326466879,
1.00874717262042,1.,1.,0.999999999999999,1.,1.,1.00874717262042,1.00918496219804,
1.00684022216754,1.00815637154096,1.00808071831764,(0.999999999999997,
0.999999999999998,1.,1.,1.,1.,1.,1.,1.,1.,0.999999999999999,1.,1.,1.,
0.999999999999997),(1.,1.,1.,1.,1.,1.,1.,0.999999999999993,0.999999999999999,
0.999999999999999,1.,1.,1.,1.,1.),(0.999999999999999,0.999999999999997,
0.999999999999998,1.,1.,0.999999999999996,0.999999999999989,1.00000000000005,
1.,1.,1.,0.999999999999998,1.,0.999999999999997,1.),(1.,1.,1.,1.,1.,1.,
0.999999999999999,1.,1.,1.,1.,1.,1.,1.,1.),(0.999999999999999,1.,1.,1.,1.,1.,
0.999999999999998,1.,1.,1.,1.,1.,1.,1.,1.),(1.00808071831764,1.00815637154096,
1.00684022216754,1.00918496219804, 1.00874717262042,1.,1.,0.999999999999997,1.,
1.,1.00874717262042,1.0090326466879,1.00654234156365,1.00800506509431,
1.00808071831765),(1.00841741491421,1.00862409621542,1.00766503350888,
1.0095151017094,1.0090326466879,1.,1.,0.999999999999998,1.,1.,1.00918496219804,
1.0095151017094,1.00749005056741,1.0085100194766,1.00841741491422),
(1.00437705575539,1.00356264396289,1.0048159344842,1.00749005056741,
1.00654234156365,1.,1.,1.,1.,1.,1.00684022216754,1.00766503350888,1.0048159344842,
1.003545395298,1.00437705575539),(0.999999999999997,0.998540981414865,
1.00354539529798,1.00851001947662,1.00800506509431,0.999999999999998,1.,
0.999999999999996,1.,1.,1.00815637154096,1.00862409621542,1.00356264396289,
0.998540981414865,0.999999999999998),(0.999999999999997,0.999999999999998,
1.00437705575538,1.00841741491422,1.00808071831766,0.999999999999996,1.,
0.999999999999999,1.,0.999999999999999,1.00808071831764,1.00841741491421,
1.00437705575539,0.999999999999997,0.999999999999997)))
REPRESENTATION_ITEM($)
```

```
SURFACE()
);
```

Possiamo dividere il codice nelle seguenti parti:

- `BOUNDED_SURFACE()` ci dice che la superficie è limitata;
- `B_SPLINE_SURFACE(3,3,((...),(...),...,(...)),.UNSPECIFIED.,.F.,.F.,.F.)` ci dice che si tratta di una superficie B-spline di grado 3 in entrambe le direzioni, specificando poi i punti di controllo (una griglia 15×15, raggruppati a quindicine per parentesi). Questo ci mostra che, purtroppo, l'esportazione in questo formato non è in grado di mantenere la struttura T-spline dell'oggetto, necessitando di convertirlo in B-spline prima di poter dare in output queste informazioni; nel caso della SferaQuad, abbiamo già visto in figura 6.11 che la conversione avviene trasformando la T-spline in una polisuperficie `NURBS`, che vedremo ora essere composta da sei distinte superfici B-spline, (in generale) razionali e non uniformi.
- `B_SPLINE_SURFACE_WITH_KNOTS((4,2,2,1,1,1,2,2,4),(4,2,2,1,1,1,2,2,4),`
  `(0.,0.125,0.25,0.5,1.,1.5,1.75,1.875,2.),`
  `(0.,0.125,0.25,0.5,1.,1.5,1.75,1.875,2.),.UNSPECIFIED.)`
  ci dà le informazioni sui vettori dei nodi, elencandoci di seguito: molteplicità dei nodi corrispondenti nella direzione u, molteplicità dei nodi corrispondenti nella direzione v, vettore nella direzione u, vettore nella direzione v. I vettori dei nodi (in questo caso coincidenti) sono cioè esprimibili, nella forma tradizionale, come:
  `(0.,0.,0.,0.,0.125,0.125,0.25,0.25,0.5,1.,`
  `1.5,1.75,1.75,1.875,1.875,2.,2.,2.,2.)`
  Si noti la molteplicità 4 dei nodi estremi: ciò è coerente con la richiesta che si fa nelle `NURBS` di prendere i nodi estremi con molteplicità pari al grado (in questo caso 3) più uno, in maniera tale che tutte le funzioni di base B-spline risultino definite.
- `RATIONAL_B_SPLINE_SURFACE(((...),(...),...,(...)))` ci dice che la superficie B-spline è effettivamente una superficie razionale, e ci specifica i pesi per ogni punto di controllo, con riferimento alla griglia impostata definendo i punti.

Anche i campi #77-#81 definiscono superfici, nella stessa maniera in cui è stata definita la #76: sono le sei superfici che formano la polisuperficie `NURBS` risultato della trasformazione da T-spline.
Il campo #82 definisce invece una curva, per la precisione una isocurva:

```
#82=(
BOUNDED_CURVE()
B_SPLINE_CURVE(3,(#188,#189,#190,#191,#192,#193,#194,#195,#196,#197,#198,
```

```
#199,#200,#201,#202,#203,#204,#205,#206,#207,#208,#209,#210,#211,#212),
.UNSPECIFIED.,.F.,.F.)
B_SPLINE_CURVE_WITH_KNOTS((4,3,3,3,3,3,3,3,4),(0.,0.125,0.25,0.5,1.,1.5,
1.75,1.875,2.),.UNSPECIFIED.)
CURVE()
GEOMETRIC_REPRESENTATION_ITEM()
RATIONAL_B_SPLINE_CURVE((1.,0.999999999999999,1.00437705575539,1.0063972353348,
1.00841741491421,1.00808071831764,1.00538714554509,0.999999999999997,
0.999999999999998,0.999999999999999,1.,1.,1.,0.999999999999999,1.,1.,1.,
0.999999999999999,1.00538714554509,1.00808071831764,1.00841741491421,
1.0063972353348,1.00437705575539,0.999999999999998,1.))
REPRESENTATION_ITEM($)
);
```

Da quello che abbiamo appena visto sulle superfici, l'interpretazione di questi dati segue di conseguenza, con la differenza che ci sono liste di punti di controllo e di pesi (e non griglie) essendo la topologia monodimensionale e non bidimensionale, e che c'è soltanto un vettore dei nodi (con rispettive molteplicità) invece che due.
I campi #83-#93 definiscono altre isocurve, mentre quelli dal #94 al #186 sono tutti campi tecnici relativi al software. Infine, quelli da #187 a #1846 sono tutti punti, richiamati in qualche modo dalle strutture precedenti; quasi tutti sono punti di controllo delle isocurve o delle superfici. Un esempio (in questo caso l'origine di uno dei semiassi) è il seguente:

```
#187=CARTESIAN\_POINT(\$,(0.,0.,0.));
```

Il file termina con due dichiarazioni di chiusura:

```
ENDSEC;
END-ISO-10303-21;
```

Quanto compiuto per il formato `.stp` può essere ripetuto per altri formati. Vediamone rapidamente alcuni.
Il salvataggio in formato `WAMIT` (`.gdf`) genera un file composto dal seguente header:

```
Esportazione file Rhino->WAMIT (IGDEF=1)
1 9.80665  ULEN GRAV
0  0  ISX  ISY
6  1  NPATCH, IGDEF
```

in cui tra le altre cose si nota la presenza del valore di accelerazione di gravità sulla superficie terrestre, a suggerire che il formato possa essere utilizzato per applicazioni che coinvolgano problemi di meccanica. Dopo questo preambolo seguono due coppie di quantità e poi l'elenco dei nodi; a differenza di prima, vediamo che i vettori dei nodi sono riportati in formato tradizionale, piuttosto che separando molteplicità e singoli valori dei nodi come nel caso dell'esportazione in `STEP`.

```
12 12
4 4
 0 0 0
 0 0.125 0.125
 0.25 0.25 0.5
 1 1.5 1.75
 1.75 1.875 1.875
 2 2 2
 2
 0 0 0
 0 0.125 0.125
 0.25 0.25 0.5
 1 1.5 1.75
 1.75 1.875 1.875
 2 2 2
 2
```

A questo elenco ne segue uno di punti di controllo, che inizia con:

```
 -15.7735026918963 5.77350269189626 -5.77350269189626
 -15.8386966086517 5.64298001828837 -5.83883144874875
 -15.9044248683163 5.50108067971025 -5.90478152698678
 -16.0406842736365 5.18195853404949 -6.04122340731491
 -16.111833313967 5.00338509775588 -6.11233500405774
 [...]
```

Il tutto viene ripetuto per sei volte, quante sono le superfici che compongono la polisuperficie `NURBS`, risultato della trasformazione da T-spline. Non sono però presenti informazioni sui pesi; probabilmente il formato non supporta le B-spline razionali, limitandosi a gestire quelle polinomiali, e tali informazioni vengono perse durante la conversione.

Il formato `IGES` (`.igs`), a parte un header e un brevissimo footer non collegati alla geometria, presenta del codice di questo tipo:

```
128,14,14,3,3,0,0,0,0,0,0.0D0,0.0D0,0.0D0,0.0D0,0.125D0,0.125D0, 0000025P     13
0.25D0,0.25D0,0.5D0,1.0D0,1.5D0,1.75D0,1.75D0,1.875D0,1.875D0,   0000025P     14
2.0D0,2.0D0,2.0D0,2.0D0,0.0D0,0.0D0,0.0D0,0.0D0,0.125D0,0.125D0, 0000025P     15
0.25D0,0.25D0,0.5D0,1.0D0,1.5D0,1.75D0,1.75D0,1.875D0,1.875D0,   0000025P     16
2.0D0,2.0D0,2.0D0,2.0D0,0.9999999999999973D0,                    0000025P     17
0.9999999999999973D0,1.004377055755385D0,1.008417414914209D0,    0000025P     18
1.008080718317643D0,0.9999999999999973D0,1.0D0,                  0000025P     19
0.9999999999999999D0,1.0D0,0.9999999999999994D0,                 0000025P     20
[...]
1.008080718317654D0,1.008417414914219D0,1.004377055755386D0,     0000025P     88
0.9999999999999985D0,0.9999999999999973D0,-15.77350269189626D0,  0000025P     89
-5.773502691896256D0,-5.773502691896256D0,-15.83883144874876D0,  0000025P     90
[...]
```

```
-5.83869660865165D0,-4.226497308103742D0,5.77350269189626D0,     0000025P    314
-5.773502691896257D0,0.0D0,2.0D0,0.0D0,2.0D0;                    0000025P    315
128,14,14,3,3,0,0,0,0,0,0.0D0,0.0D0,0.0D0,0.0D0,0.125D0,0.125D0, 0000027P    316
[...]

Legenda:
"3,3" gradi, "0.0D0,0.0D0,0.0D0,0.0D0,0.125D0,0.125D0,0.25D0,0.25D0,0.5D0,1.0D0,
1.5D0,1.75D0,1.75D0,1.875D0,1.875D0,2.0D0,2.0D0,2.0D0,2.0D0" vettore nodi u,
"0.0D0,0.0D0,0.0D0,0.0D0,0.125D0,0.125D0,0.25D0,0.25D0,0.5D0,1.0D0,1.5D0,1.75D0,
1.75D0,1.875D0,1.875D0, 2.0D0,2.0D0,2.0D0,2.0D0" vettore nodi v,
da riga 17 a riga 89, i numeri "vicini a 1" sono pesi,
da riga 89 a riga 315, tutto il resto sono punti di controllo.
la cosa viene ripetuta per sei volte, quante le polisuperfici NURBS.
```

Nel file `RenderMan` (`.rib`) le sei superfici vengono rappresentate con questa sintassi. Si ricordi che l'ordine è il grado aumentato di uno:

```
AttributeBegin
TextureCoordinates [ 0 1 1 1 0 0 1 0 ]
NuPatch 15 4 [ 0 0 0 0 0.0625 0.0625 0.125 0.125 0.25 0.5 0.75 0.875 0.875 0.9375
0.9375 1 1 1 1 ] 0 1 15 4 [ 0 0 0 0 0.0625 0.0625 0.125 0.125 0.25 0.5 0.75 0.875
0.875 0.9375 0.9375 1 1 1 1 ] 0 1 "Pw" [ -15.7735 -5.7735 -5.7735 1 -15.8388
-5.64298 -5.8387 1 -15.9744 -5.52516 -5.93027 1.00438 -16.1762 -5.22558 -6.09153
1.00842 -16.2425 -5.04382 -6.16122 1.00808 -16.3375 -4.40262 -6.33747 1
[...]
AttributeEnd

Sintassi:
NuPatch {num_nodi-ordine u} {ordine u} [ {elenco nodi u} ] nodo_min_u nodo_max_u
        {num_nodi-ordine v} {ordine v} [ {elenco nodi v} ] nodo_min_v nodo_max_v
"Pw" [ {elenco punti di controllo: x y z w} ]
```

Rispetto ai formati precedenti, gli elementi dei vettori dei nodi risultano divisi per due.

Il file `ACIS` (`.sat`) viene scritto, per ciò che concerne le superfici, in questo modo (ne riportiamo sempre una delle sei):

```
-5 spline-surface $-1 forward { exactsur nurbs 3 3 both open open none none 9 9
0 3 0.125 2 0.25 2 0.5 1 1 1
1.5 1 1.75 2 1.875 2 2 3
0 3 0.125 2 0.25 2 0.5 1 1 1
1.5 1 1.75 2 1.875 2 2 3
-15.773502691896264 -5.7735026918962564 -5.7735026918962564 0.99999999999999734
-15.838831448748758 -5.6429800182883687 -5.8386966086516479 0.99999999999999734
-15.904781526986783 -5.5010806797102436 -5.9044248683163323 1.0043770557553853
-16.041223407314913 -5.1819585340494871 -6.0406842736364563 1.0084174149142091
-16.112335004057744 -5.0033850977558787 -6.1118333139669891 1.0080807183176432
[...] (continua con tutti gli altri punti di controllo)

Sintassi:
{ exactsur nurbs grado_u grado_v <altreinfo>
```

```
nodo_min_u molt_nodo_min_u ...
... nodo_max_u molt_nodo_max_u
nodo_min_v molt_nodo_min_v ...
... nodo_max_v molt_nodo_max_v
punti_x punti_y punti_z punti_w (pesi)
<altreinfo> }
```

mentre ritornano le isocurve che abbiamo già visto nella conversione `STEP` (che ricordiamo essere in numero di dodici), in questo caso rappresentate da una scrittura del tipo:

```
-71 intcurve-curve $-1 forward { exactcur nurbs 3 open 9
0 3 0.125 3 0.25 3 0.5 3 1 3
1.5 3 1.75 3 1.875 3 2 3
-15.773502691896264 -5.7735026918962564 -5.7735026918962564 1
-15.838831448748758 -5.6429800182883687 -5.8386966086516479 0.99999999999999911
-15.904781526986783 -5.5010806797102445 -5.9044248683163323 1.0043770557553862
[...] (continua con tutti gli altri punti di controllo)

Sintassi:
{ exactsur nurbs grado <altreinfo>
nodo_min molt_nodo_min ...
... nodo_max molt_nodo_max
punti_x punti_y punti_z punti_w (pesi)
<altreinfo> }
```

Osserviamo che, per qualche motivo interno alle specifiche, i nodi estremi vengono rappresentati con molteplicità 3 invece di 4.

Infine, il formato `Parasolid` (`.x_t`) memorizza i dati in una forma meno intuitiva, ma che presenta ancora gli elementi costituenti di una `NURBS` quali vettori dei nodi con corrispondenti molteplicità e punti di controllo con relativi pesi, come mostra il seguente frammento di codice (peraltro i vettori dei nodi si ripetono esattamente sei volte nel documento, a confermare quanto ottenuto in precedenza):

```
4 2 2 1 1 1 2 2 4
127 9 313 4 2 2 1 1 1 2 2 4 128 9 314 0 0.125 0.25 0.5 1 1.5 1.75 1.875 2 128 9
315 0 0.125 0.25 0.5 1 1.5 1.75 1.875 2 45 900 316 -0.15773502691896221 -0.0577
35026918962415 -0.057735026918962415 0.99999999999999734 -0.15642980018288349 -0
.05838696608651639 -0.058388314487487436 0.99999999999999845 -0.1556892977411405
7 -0.059302688651684331 -0.059306270849537657 1.0043770557553846 -0.153097513782
41085 -0.060915312195334595 -0.060920748913237753 1.0084174149142204 -0.15124623
226542175 -0.061612213173816176 -0.061617270614886747 1.0080807183176557
```

Osserviamo che i valori dei punti di controllo sono divisi per cento rispetto a quelli ottenuti in tutti i casi precedenti. Questo è avvenuto per una ragione tecnica del formato: i file `Parasolid` fanno sempre uso dei metri ([21]), e se le unità di Rhino fanno riferimento ad unità diverse dai metri, le geometrie esportate verranno ridimensionate del relativo fattore; in questo caso, su Rhino si è lavorato in centimetri, per cui è avvenuta in automatico la conversione, che però non va ad inficiare in alcun modo l'aspetto matematico

della cosa.

Abbiamo visto tutti questi esempi per mostrare come vari software e i loro rispettivi formati facciano uso delle NURBS per rappresentare curve, superfici, e solidi, e come questa peculiarità possa essere sfruttata per ricavare informazioni su punti di controllo, nodi, e pesi, che altrimenti rimarrebbero nascoste. Per quanto riguarda le T-spline in sè, trattandosi di un formato molto recente, non è presente tutto questo supporto; tuttavia, è possibile ottenere informazioni grazie al formato `.tsm`, supportato dagli stessi creatori dell'add-on.
In questo formato, la SferaQuad viene descritta, riportando il testo su due colonne per motivi tipografici, come:

```
#TS0200

degree 3
cap-type 1
star-smoothness 0
units 0.01 centimeters
f 33 0
f 1 0
f 2 0
f 4 0
f 6 0
f 8 0
f 46 0
f 13 0
f 17 0
f 20 0
f 26 0
f 30 0
f 65 0
f 34 0
f 37 0
f 42 0
f 76 0
f 51 0
f 57 0
f 62 0
f 69 0
f 73 0
f 80 0
f 85 0
e 1 1
e 0 1
e 2 1
e 4 1
e 3 1
e 6 1
e 8 1
e 7 1
e 12 1
e 10 1
e 13 1
e 14 1
e 17 1
e 20 1
e 18 1
e 23 1
e 24 1
e 26 1
e 28 1
e 30 1
e 34 1
e 32 1
e 35 1
e 37 1
e 39 1
e 42 1
e 40 1
e 47 1
e 44 1
e 51 1
e 48 1
e 54 1
```

```
e 55 1
e 57 1
e 59 1
e 62 1
e 64 1
e 66 1
e 67 1
e 70 1
e 71 1
e 74 1
e 77 1
e 78 1
e 81 1
e 82 1
e 88 1
e 92 1
v 0 NORTH
v 24 NORTH
v 32 NORTH
v 55 NORTH
v 64 NORTH
v 88 NORTH
v 3 NORTH
v 28 NORTH
v 39 NORTH
v 59 NORTH
v 67 NORTH
v 92 NORTH
v 7 NORTH
v 31 NORTH
v 40 NORTH
v 63 NORTH
v 71 NORTH
v 95 NORTH
v 10 NORTH
v 74 NORTH
v 14 NORTH
v 78 NORTH
v 18 NORTH
v 82 NORTH
v 44 NORTH
v 48 NORTH
l 33 12 11 0 0 1 0
l 5 35 33 0 1 0 0
l 11 3 5 0 2 2 0
l 2 16 15 6 2 4 0
l 9 39 36 6 3 3 0
l 36 1 2 6 1 2 0
l 15 7 9 6 4 5 0
l 6 21 19 12 4 7 0
l 19 40 41 12 5 6 0
l 41 4 6 12 3 5 0
l 46 23 22 18 6 9 0
l 16 2 0 18 2 1 0
l 0 45 46 18 0 8 0
l 22 14 16 18 7 10 0
l 13 27 25 20 7 11 0
l 21 6 3 20 4 4 0
l 3 11 13 20 2 10 0
l 25 18 21 20 8 12 0
l 17 31 29 22 8 14 0
l 50 8 7 22 5 7 0
l 29 48 50 22 9 13 0
l 7 15 17 22 4 12 0
l 27 13 10 1 7 9 0
l 10 52 53 1 6 15 0
l 53 26 27 1 10 16 0
l 31 17 14 7 8 11 0
l 24 58 56 7 10 17 0
l 14 22 24 7 7 16 0
l 56 30 31 7 11 18 0
l 61 20 18 13 9 14 0
l 28 63 61 13 11 19 0
l 18 25 28 13 8 18 0
l 65 47 45 2 12 21 0
l 45 0 1 2 0 0 0
l 38 66 65 2 13 20 0
l 1 36 38 2 1 22 0
l 35 5 4 8 1 3 0
l 43 70 68 8 14 23 0
l 68 34 35 8 13 22 0
l 4 41 43 8 3 24 0
l 8 50 49 14 5 26 0
l 39 9 8 14 3 6 0
l 49 71 72 14 15 25 0
l 72 37 39 14 14 24 0
l 76 54 52 24 16 28 0
l 12 33 32 24 0 21 0
```

```
l 52 10 12 24 6 8 0              l 42 84 83 16 15 40 0
l 32 75 76 24 12 27 0            l 70 43 42 16 14 25 0
l 20 61 60 25 9 30 0             l 83 67 70 16 21 39 0
l 84 42 40 25 15 26 0            l 80 88 86 19 22 41 0
l 40 19 20 25 5 13 0             l 47 65 64 19 12 36 0
l 60 82 84 25 17 29 0            l 86 44 47 19 16 27 0
l 23 46 44 3 6 28 0              l 64 79 80 19 20 42 0
l 58 24 23 3 10 15 0             l 85 92 89 21 23 43 0
l 44 86 87 3 16 31 0             l 77 69 67 21 20 38 0
l 87 57 58 3 18 32 0             l 89 74 77 21 22 42 0
l 63 28 26 9 11 17 0             l 67 83 85 21 21 44 0
l 55 91 90 9 18 33 0             l 51 94 93 23 17 45 0
l 26 53 55 9 10 32 0             l 81 73 71 23 21 40 0
l 90 62 63 9 19 34 0             l 71 49 51 23 15 29 0
l 94 51 48 15 17 30 0            l 93 78 81 23 23 44 0
l 48 29 30 15 9 19 0             l 54 76 74 5 16 41 0
l 59 95 94 15 19 35 0            l 91 55 54 5 18 31 0
l 30 56 59 15 11 34 0            l 74 89 91 5 22 46 0
l 69 77 75 4 20 36 0             l 88 80 78 11 22 43 0
l 75 32 34 4 12 20 0             l 95 59 57 11 19 33 0
l 34 68 69 4 13 37 0             l 57 87 88 11 18 46 0
l 73 81 79 10 21 38 0            l 78 93 95 11 23 47 0
l 66 38 37 10 13 23 0            l 92 85 82 17 23 45 0
l 79 64 66 10 20 37 0            l 82 60 62 17 17 35 0
l 37 72 73 10 14 39 0            l 62 90 92 17 19 47 0

0g -16.387104591064279 -6.3871045910642534 -6.3871045910642534 1
0g -16.387104591064283 6.3871045910642561 -6.3871045910642579 1
0g -9.9999999999999982 -8.5606220778362241 -8.5606220778362392 1
0g -9.9999999999999982 8.5606220778362445 -8.5606220778362241 1
0g -3.6128954089357381 -6.3871045910642552 -6.3871045910642552 1
0g -3.6128954089357421 6.3871045910642597 -6.3871045910642561 1
0g -18.560622077836232 -8.5606220778362392 8.711492370599165e-16 1
0g -18.560622077836211 8.5606220778362427 2.5597519116382605e-15 1
0g -9.9999999999999947 -12.342601774397698 -4.9717288727805346e-15 1
0g -10 12.342601774397689 -1.1303955299844823e-15 1
0g -1.4393779221637626 -8.5606220778362427 6.4421615277745881e-15 1
0g -1.4393779221637559 8.5606220778362392 1.1198940275863462e-15 1
0g -16.387104591064258 -6.3871045910642597 6.3871045910642534 1
0g -16.387104591064272 6.3871045910642517 6.3871045910642517 1
0g -10.000000000000004 -8.5606220778362392 8.5606220778362445 1
0g -9.9999999999999964 8.5606220778362427 8.5606220778362241 1
0g -3.6128954089357386 -6.3871045910642525 6.3871045910642463 1
0g -3.6128954089357448 6.3871045910642543 6.3871045910642543 1
```

```
0g -18.560622077836207 2.779624592443058e-15 -8.5606220778362374 1
0g -1.4393779221637619 1.8407472438685706e-15 -8.560622077836241 1
0g -22.342601774397721 -4.5628199234508944e-16 -3.5492546095771175e-17 1
0g 2.3426017743976959 1.0660896636389826e-15 -3.4693045427008706e-15 1
0g -18.560622077836232 2.5586159515418268e-15 8.5606220778362463 1
0g -1.4393779221637621 1.4498131355830276e-15 8.5606220778362445 1
0g -10.000000000000007 -6.5859842684524661e-15 -12.342601774397698 1
0g -10.000000000000002 -3.8220970419929065e-15 12.342601774397693 1

tol 1.0000000000000001e-05
ver 1044
```

Vediamo come interpretare quanto riportato, anche in base a quanto dichiarato in [22]:

- `#TS0200` indica il formato.
- `degree 3` indica il grado.
- `units 0.01 centimeters` indica l'unità di misura internamente utilizzata da Rhino.
- `cap-type` e `star-smoothness` controllano come viene approssimata la superficie intorno ai punti straordinari durante la conversione in `NURBS`. La superficie limite di una `T-NURCC` in un intorno di un punto straordinario è definita tramite raffinamento ripetuto; questo significa che si rende necessario approssimare la superficie limite nei pressi di un punto straordinario affinchè la stessa sia compatibile con lo schema `NURBS`. Più precisamente: `cap-type` stabilisce la continuità delle patch generate nella regione di approssimazione, e può valere 0, a garantire soltanto la continuità di posizione (C^0), o, come in questo caso, 1, a garantire una continuità geometrica di tangenza (G^1); `star-smoothness` stabilisce quante suddivisioni locali sono da effettuarsi prima del procedimento di approssimazione, in questo caso 0, in quanto lo stesso è in grado di ottenere il risultato desiderato da solo. Processi di approssimazione meno precisi, implementati in versioni precedenti dell'add-on, richiedevano invece l'esecuzione di alcune suddivisioni, solitamente in numero di 2.

Gli oggetti che seguono dello stesso tipo sono numerati progressivamente a partire da 0, cioè alla prima faccia a comparire nell'ordine viene associato il numero 0, alla seconda il numero 1, e così via.

- `f <link> <flags>` indica che si tratta di una faccia, ne indica il semilato di origine (link), ovvero un angolo (una giunzione regolare, non una T-giunzione o una L-giunzione) che appartenga alla faccia, e che ne

funga da origine (anche parametrica, nel caso in cui la faccia possieda uno spazio parametrico consistente, cosa che solitamente accade), e ne indica le *flags*, ovvero le eventuali informazioni aggiuntive codificate in un numero intero; in questo caso esse sono tutte pari a 0, ad indicare che non ci sono informazioni particolari legate ad esse.

- `e <link> <intervallo>` indica che si tratta di un lato, ne indica un semilato (link), quello indicato per orientare il lato, e specifica poi l'intervallo tra nodi associato, ovvero la lunghezza del lato in termini parametrici; i lati vengono considerati avere una direzione, con inizio sul vertice posseduto da tale semilato, e fine sul vertice alla fine di tale semilato.

- `v <link> <direzione>` rappresenta un vertice topologico sulla superficie, indicandone il semilato che deve avere origine nel vertice, e la direzione in cui il semilato punta nel sistema di coordinate locali relativo al vertice; il sistema di coordinate locali è lo stesso utilizzato per le funzioni di miscelamento e per i punti di controllo. La direzione può essere nord, sud, est, ovest, o invalida, quest'ultima nel caso in cui non abbia una topologia locale consistente.

- `l <prev> <succ> <opp> <vertice> <faccia> <lato> <flags>` rappresenta un link (semilato). I campi indicano di seguito: `prev` il link precedente in senso antiorario rispetto alla sua faccia genitore, `succ` il link successivo in senso antiorario rispetto alla sua faccia genitore, `opp` il link successivo sul lato corrente, `vertice` il vertice associato, `faccia` la faccia associata, `lato` il lato associato, `flags` è congruo modulo 4 a 0 se si tratta di un angolo, 1 se è una T-giunzione, 2 se è una L-giunzione. Al momento quell'intero viene utilizzato soltanto per codificare questa informazione (e quindi possono essere effettivamente assunti soltanto i valori 0, 1, e 2), ma in futuro potrà essere utilizzato per inserire altre info.

- `0g <x> <y> <z> <w>` specifica i grip (punti di controllo). Le prime tre coordinate sono da considerarsi pesate, per cui le effettive posizioni cartesiane sono date da $(x/w, y/w, z/w)$.

- `tol 1.0000000000000001e-05` indica il valore di tolleranza per calcoli eseguiti con le T-spline, sia per gli intervalli tra nodi che per la geometria.

- `ver 1044` specifica la versione della topologia; il valore viene incrementato ogni volta che vengono compiute modifiche all'add-on.

Come mostrato, viene presentata una serie di informazioni che non esplicita totalmente in termini matematici l'oggetto, ma che può comunque essere d'aiuto nella comprensione del modello.

Capitolo 7

T-spline e watermarking

7.1 Introduzione al watermarking

È noto anche ai non addetti ai lavori che uno dei principali problemi della produzione e della distribuzione di contenuti multimediali, a fronte della possibilità di comunicazione pressoché immediata con tutto il mondo, è quella del rispetto e della gestione dei diritti d'autore.
In tutto il mondo milioni di persone utilizzano programmi di condivisione peer-to-peer, semplici applicativi che permettono di scambiare file da computer a computer passando attraverso dei server dedicati. Questi software forniscono la possibilità di distribuire molto rapidamente una propria creazione, ma allo stesso tempo permettono di far circolare senza un adeguato controllo materiale protetto da copyright, per il quale l'autore o gli autori avrebbero diritto ad avere riconosciuto in una determinata forma il proprio lavoro.
Se dunque anche un semplice utente finale può essere in grado di aggirare un certo tipo di restrizioni, si può immaginare quanto sarebbe possibile fare in campo professionale, senza l'utilizzo di opportune precauzioni che permettano di stabilire univocamente, nel caso di disputa, il detentore dei diritti di un dato contenuto. Si rendono pertanto necessari sistemi di identificazione da inserire direttamente all'interno dei file, e realizzati in maniera tale da essere il più difficile possibile da alterare o da rimuovere, da parte di soggetti non autorizzati a farlo.
Il termine *watermarking* si riferisce proprio all'inclusione di informazioni all'interno di un file multimediale o di altro genere, che può essere successivamente rilevato o estratto per trarre informazioni sulla sua origine e provenienza, ma (perlomeno elementarmente) non modificato o rimosso da chi non ne ha diritto. Per mezzo del watermark il documento è ancora accessibile, ma contrassegnato in modo permanente.
Tali indicazioni, dette *watermark*, possono essere evidenti per l'utente del file (per esempio nel caso di una indicazione di copyright applicata in sovraimpressione su una immagine digitale) o latenti (nascoste all'interno del file).

Quanto segue ora è tratto da [12].
La tecnica del watermarking digitale può venire utilizzata con diversi scopi, alcuni dei quali sono:

- rendere manifesto a tutti gli utenti chi sia il legittimo proprietario del documento, e questo è il caso in cui si opta per lasciare il marchio visibile, considerando che naturalmente un marchio visibile può essere accompagnato da un altro mark non visibile;
- dimostrare l'originalità di un documento non contraffatto;
- evitare la distribuzione di copie non autorizzate;
- marcare alcune caratteristiche specifiche del documento;
- segnare il percorso di vendita del documento, utilizzando un marchio differente per ciascun acquirente.

7.1.1 Prime proprietà

I watermark possono essere utilizzati per diversi scopi e, quindi, devono soddisfare a ben determinate esigenze. Esistono però delle caratteristiche comuni a tutti i watermark:

- il legittimo proprietario o un'autorità indipendente di controllo devono poter facilmente estrarre le informazioni del watermark;
- il recupero del watermark deve provare in maniera non ambigua l'identità del proprietario;
- deve essere possibile sovrapporre più watermark sul documento, senza che quelli precedenti siano distrutti;
- il watermark deve essere inserito all'interno del segnale da proteggere per maggiore sicurezza e portabilità.

Per soddisfare tutte queste richieste i watermark devono essere:

- Invisibili dal punto di vista grafico: inserire un segnale di watermark comporta necessariamente un seppur piccolo degrado dell'immagine; questo degrado deve essere il più lieve possibile in modo da non alterare la percezione del documento. Il grado d'alterazione deve essere deciso dal proprietario del documento, il quale può scegliere tra forti alterazioni, che danno una garanzia di robustezza a eventuali attacchi, e deboli alterazioni, che non degradano il prodotto.
- Codificati a chiave: ogni segnale di watermark è associato a una particolare sequenza di bit detta chiave (key). La chiave serve sia per

produrre il segnale di watermark che per riconoscerlo all'interno di un documento. La chiave è privata e caratterizza univocamente il legittimo proprietario del documento. Solo chi è in possesso della chiave (il proprietario o un ente autorizzato) è in grado di dimostrare la presenza del watermark nel prodotto digitale. Il numero di chiavi possibili deve essere sufficientemente grande da poter escludere probabilisticamente almeno fino ad un certo grado la possibilità di risoluzione del marchio tramite l'uso di attacchi a tentativi o di forza bruta.

- Efficienti statisticamente: un documento firmato con un segnale di watermark deve essere facilmente riconoscibile se si conosce la giusta chiave. La probabilità di falso positivo, ovvero che la chiave nella fase di riconoscimento venga rifiutata pur essendo corretta, deve essere sufficientemente bassa.

- Invisibili dal punto di vista binario, a livello statistico: possedere un gran numero di documenti digitali, tutti firmati con la stessa chiave, non deve rendere riconoscibile (e quindi eliminabile) la firma. Diversi prodotti firmati con la stessa chiave devono generare segnali di watermark differenti. Dobbiamo essere sicuri che il riconoscimento della chiave all'interno dell'immagine da parte di terzi sia impossibile.

- Multipli: deve essere possibile inserire un elevato numero di segnali di watermark all'interno dello stesso documento; ognuno di questi segnali può essere riconosciuto mediante la corrispondente chiave.

- Robusti: sui documenti digitali possono venir fatte numerose operazioni per migliorare la loro qualità o per comprimere la loro dimensione; per esempio, si pensi a tutti i tipi di trasformazioni geometriche che vengono compiute correntemente sulle immagini. I segnali di watermark devono essere tali da non venire eliminati da questo tipo di operazioni, né da operazioni mirate ad alterare o cancellare il marchio stesso.

- Invertibili da chi ne ha diritto: il legittimo proprietario del documento deve poter rimuovere il watermark. In realtà questa proprietà non si può ottenere esattamente se devono essere garantite la robustezza e la resistenza alle aggressioni. Si deve pertanto tendere al migliore rapporto tra facilità di rimozione da parte di chi è autorizzato a farlo, e difficoltà a compiere la stessa operazione da parte di chi non lo è.

In alcuni casi per invertibilità si intende la possibilità di generare un falso watermark e un falso documento originale che sia uguale a quello vero. In questo modo dall'inserimento del falso watermark si ottiene un documento che è perfettamente uguale (invertibilità) o solo percettibilmente uguale

(quasi invertibilità) a quello reale. Come detto, se si vuole considerare il watermarking come prova inconfutabile per l'applicazione del copyright, allora i marchi devono essere non invertibili o quasi non invertibili.

7.1.2 Nozioni elementari

Consideriamo ora il caso delle immagini, riportando alcune nozioni sul marking, che possono poi essere estese per altre tipologie di contenuti multimediali, ma che per semplicità e per l'ambito della trattazione ci limiteremo discutere circa i file che rappresentano immagini.
Un simile schema può essere composto di due algoritmi:

- uno di codifica, che:
  - prende in input l'immagine originale;
  - restituisce in output:
    - la corrispondente immagine opportunamente marcata;
    - il marchio vero e proprio.
- uno di decodifica, che:
  - prende in input:
    - l'immagine marcata;
    - l'immagine originale, nel caso in cui sia necessaria per verificare la presenza del marchio (dipende dall'implementazione).
  - restituisce in output il marchio associato.

L'aggiunta di un watermark ad un'immagine può essere vista come l'inserimento di una componente di *rumore*, ovvero di "disturbo", nell'immagine stessa.
Poniamo ora la seguente terminologia:

- sia V l'immagine originale;
- sia W il watermark da inserire, che potrebbe dipendere da V; possiamo indicarlo come un insieme di elementi, $\{w_1, w_2, \ldots, w_n\}$.
- sia V_w l'immagine marchiata.

In virtù di quanto affermato precedentemente, possiamo anche introdurre due funzioni, una di codifica E e una di decodifica D, tali per cui:

$$\begin{aligned} E(V, W) &= V_w \\ D(V, V_w) &= W' \end{aligned}$$

Abbiamo anche detto che, a seconda dell'implementazione, può non essere necessaria l'immagine originaria per verificare la presenza del marchio; in tal

caso si parla di watermark *cieco*. Qualora si presentasse tale situazione, la seconda uguaglianza non dipenderebbe pertanto da V e si potrebbe scrivere più semplicemente come:

$$D(V_w) \quad = \quad W'$$

Si noti che abbiamo utilizzato, per l'output della decodifica, la notazione W' e non W. Questo è motivato dal fatto che i due marchi non sono necessariamente identici, in quanto l'immagine può essere stata modificata tra la fase di codifica e di decodifica. Non potendo pertanto stabilire in maniera esatta la coincidenza, si rende necessaria l'introduzione di una funzione di comparazione C_δ che permetta di stabilire se i due watermark corrispondono, dove δ è un valore di soglia opportunamente stabilito. Se c è il risultato di $C_\delta(W, W')$, i watermark possono essere considerati corrispondenti soltanto se $c > \delta$.
Se denotiamo con $F = \{f_1, f_2, \ldots, f_n\}$ alcune particolari caratteristiche dell'immagine, possiamo identificare la codifica di un watermark come l'applicazione di un operatore di inserimento $\oplus$ tra le f_i e le w_i, e la decodifica come l'applicazione di un operatore di estrazione $\ominus$; nella fattispecie:

$$\begin{aligned} f'_i &= f_i \oplus w_i \qquad i = 1, \ldots, n \\ w'_i &= f'_i \ominus f_i \qquad i = 1, \ldots, n \end{aligned}$$

dove:

- $F' = \{f'_1, f'_2, \ldots, f'_n\}$ sono le caratteristiche corrispondenti dell'immagine con watermark;
- $W' = \{w'_1, w'_2, \ldots, w'_n\}$ è il watermark estratto.

7.1.3 Tecniche generali di inserimento

Un watermark può essere inserito all'interno di un'immagine con diverse tecniche. Vediamone tre classi, nelle loro principali caratteristiche.

Le tecniche basate sul dominio spaziale modificano direttamente i valori dei pixel dell'immagine, in base al codice che deve essere incluso. Ci sono diverse possibilità per attuare questa modifica:

- Si aggiungono o sottraggono ai valori dei pixel dei numeri pseudocasuali, che dipendono da una chiave generatrice. Per poter rilevare il watermark si deve possedere la chiave, garantendo così che solo gli utenti autorizzati possano estrarlo.
  Il metodo non è in grado di resistere ad attacchi di tipo collusivo, ovvero se un gruppo di utenti si coalizza per attaccare il sistema di

marchiatura, per esempio combinando le proprie copie contenenti ciascuna un marchio differente, allo scopo di crearne una nuova di elevata qualità e priva di marchio.
La resistenza può essere attuata tramite tecniche di *pre-warping*, ovvero modificando in maniera unica e percettivamente irrilevante l'immagine prima della marchiatura, e garantendo una forte degradazione della qualità dell'immagine generata da un attacco a collusione.

- Si modificano solo i valori di alcuni pixel scelti pseudocasualmente, a cui si somma o si sottrae un valore in base alla loro posizione. Il metodo non è particolarmente robusto, specialmente nei confronti dei tentativi di contraffazione. Infatti applicando un secondo watermark più globale, può diventare difficile riconoscere quello più particolare, che anzi può essere fatto sembrare parte del secondo marchio.

- L'immagine viene suddivisa in blocchi, solitamente di forma quadrata; alcuni di essi sono scelti in modo pseudocasuale e la modifica avviene aggiungendo o sottraendo un determinato numero ai valori di alcuni pixel contenuti nei blocchi selezionati. Il metodo non sopporta la rotazione e il cropping, dal momento che l'operazione non è invariante per tali tipi di trasformazioni.

- Si lascia cadere il bit meno significativo di pixel opportunamente scelti. Il metodo ha grossi problemi nel caso in cui l'immagine venga traslata o ridimensionata, per la stessa ragione di prima.

- Se l'immagine è a colori, si inserisce il watermark in una certa banda del colore, in modo che non sia visibile normalmente, ma che renda il documento inutilizzabile se lo si vuole stampare. Il metodo ha gli stessi difetti di quello precedente.

Una seconda classe è data dalle tecniche basate sulle trasformate: si utilizzano la trasformata discreta del coseno, la trasformata wavelet o la trasformata di Fourier e il watermark viene incorporato nei coefficienti della trasformazione.
Tali tecniche operano seguendo questo schema: l'immagine viene convertita in una o più matrici in cui sono riportati i valori numerici dei singoli pixel, per esempio le loro componenti R, G, B; a queste matrici sono applicate una delle trasformazioni invertibili sopra citate. Alcuni dei coefficienti della trasformata vengono modificati, ottenendo l'inserimento del watermark. Infine, applicando la trasformata inversa, si ricompongono delle matrici, e quindi un'immagine, simile a quella di partenza, a cui, però è stato applicato il marchio.

Si possono infine applicare delle tecniche ibride, ovvero tecniche che pur sfruttando le trasformate matematiche non mancano di adattabilità spaziali.

Queste tecniche sono molto efficaci nell'adattare il livello di inserimento del watermark al contenuto locale dell'immagine. D'altra parte questa proprietà provoca una sensibilità alla traslazione delle immagini e allo scaling.

7.2 Watermarking di superfici T-spline

Vediamo ora un paio di tecniche di watermarking per le superfici T-spline. Piuttosto che agire sugli elementi di un'immagine generica, quali per esempio i valori cromatici dei pixel citati poco fa, che sono stati mostrati in generale come applicazione a strumenti semplici, i metodi seguenti vanno a modificare, sempre impercettibilmente e con la possibilità di invertire l'operazione, gli elementi caratterizzanti di queste superfici, quali vettori dei nodi e punti di controllo.
Esistono anche tecniche più articolate, per esempio basate sulla trasformazione wavelet non uniforme (si veda [15]), sulle quali però non ci soffermeremo, presupponendo esse strumenti teorici che vanno al di là degli scopi della nostra trattazione.

7.2.1 Algoritmo per la steganografia

Con il termine *steganografia* si intendono le tecniche che permettono di inserire informazioni segrete all'interno di dati che apparentemente non appaiono poter nascondere alcunché. I metodi steganografici si basano generalmente sull'assunzione che l'esistenza di queste informazioni segrete non sia nota ad altri all'infuori del mittente e del destinatario, e vengono principalmente utilizzati in comunicazioni dirette tra due parti. Per questo motivo di solito non viene richiesta la robustezza dei metodi usati, e pertanto l'informazione nascosta può non essere più recuperata se i dati hanno subito troppe manipolazioni. Quanto segue ricalca [14].
Per inserire le informazioni, imponiamo innanzitutto un sistema di coordinate nodali, specificando l'origine nella preimmagine (figura 7.1).

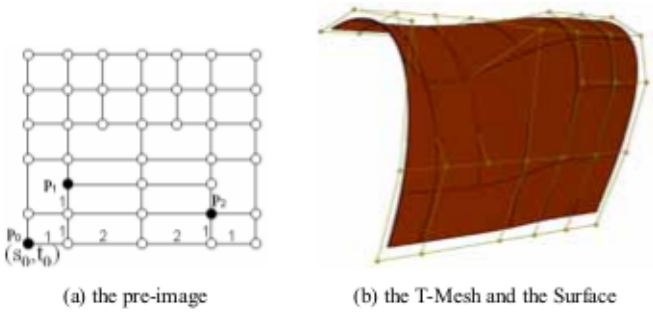


Figura 7.1: Coordinate nodali e origine nella preimmagine.

Siano ora $(s_j, t_j), j = 1, \ldots, n$ le coordinate nodali per i punti di controllo P_j, e siano $P(s_j, t_j)$ i punti sulla superficie T-spline associati a (s_j, t_j). L'idea di questo algoritmo è quello di inserire vettori di dati $W_k = [W_k^x, W_k^y, W_k^z]$ nei punti $P(s_{j_k}, t_{j_k})$, dove $k = 1, \ldots, r, r < n$; questi dati possono essere numeri in virgola mobile, per esempio trasformati da binari, e portano alla generazione di nuovi punti T_k, definiti come $T_k = P(s_{j_k}, t_{j_k}) + W_k, k = 1, \ldots, r$, attraverso i quali si richiede il passaggio della superficie modificata.
Le superfici che soddisfano tale condizione sono tuttavia molteplici, e dipendono da come vengono modificati i punti di controllo, non essendoci, in generale, una sola disposizione di essi che verifichi il passaggio per tali punti compatibilmente con l'impostazione generale. Ne scegliamo dunque una, e la nostra scelta ricade su quella che minimizza il cambiamento di forma rispetto alla superficie originaria; per determinarla, consideriamo una perturbazione $\epsilon_i = [\epsilon_j^x, \epsilon_j^y, \epsilon_j^z], i = 1, \ldots, n$, per i punti di controllo, in modo tale che la superficie modificata:

$$\tilde{P}(s,t) = \sum_{i=1}^{n} (\mathbf{P}_i + \epsilon_i) R_i(s,t) = P(s,t) + \sum_{i=1}^{n} \epsilon_i R_i(s,t)$$

passi attraverso i punti T_k, ovvero soddisfi le equazioni:

$$\begin{aligned} T_k &= \tilde{P}(s_{j_k}, t_{j_k}) = \sum_{i=1}^{n} (\mathbf{P}_i + \epsilon_i) R_i(s_{j_k}, t_{j_k}) \\ &= P(s_{j_k}, t_{j_k}) + \sum_{i=1}^{n} \epsilon_i R_i(s_{j_k}, t_{j_k}), \qquad k = 1, \ldots, r \end{aligned}$$

Per determinare $\epsilon_i, i = 1, \ldots, n$, impostiamo un problema di ottimizzazione, cercando di trovare i valori che:

- soddisfano $T_k - P(s_{j_k}, T_{j_k}) - \sum_{i=1}^{n} \epsilon_i R_i(s_{j_k}, T_{j_k}) = 0, k = 1, \ldots, r$;
- minimizzano $\sum_{i=1}^{n} ||\epsilon_i||^2$.

Per risolverlo, definiamo la funzione di Lagrange:

$$L = \sum_{i=1}^{n} ||\epsilon_i||^2 + \sum_{k=1}^{r} \lambda_k (T_k - P(s_{j_k}, t_{j_k}) - \sum_{i=1}^{n} \epsilon_i R_i(s_{j_k}, t_{j_k}))$$

dove $\lambda_k = [\lambda_k^x, \lambda_k^y, \lambda_k^z], k = 1, \ldots, r$ sono i moltiplicatori di Lagrange e $|| \cdot ||$ è la norma euclidea. Sia:

$$\frac{\partial}{\partial \lambda_k^x} = \frac{\partial}{\partial \lambda_k^y} = \frac{\partial}{\partial \lambda_k^z} = 0, \qquad \frac{\partial}{\partial \epsilon_i^x} = \frac{\partial}{\partial \epsilon_i^y} = \frac{\partial}{\partial \epsilon_i^z} = 0$$

per $k = 1, \ldots, r, i = 1, \ldots, n$, ne segue il sistema lineare:

$$\begin{cases} 2\epsilon_i + \sum_{k=1}^{r} \lambda_k R_i(s_{j,k}, t_{j,k}) = 0, & i = 1, \ldots, n \\ T_k - P(s_{j,k}, t_{j,k}) - \sum_{i=1}^{n} \epsilon_i R_i(s_{j,k}, t_{j,k}) = 0, & k = 1, \ldots, r \end{cases}$$

la cui risoluzione porta ad ottenere gli ϵ_i desiderati. Ponendo infine:

$$\tilde{\mathbf{P}}_i = \mathbf{P}_i + \epsilon_i, \quad i = 1, \ldots, n$$

otteniamo l'equazione della superficie T-spline con i dati inseriti:

$$\tilde{P}(s,t) = \sum_{i=1}^{n} \tilde{\mathbf{P}}_i R_i(s,t) = \sum_{i=1}^{n} (\mathbf{P}_i + \epsilon_i) R_i(s,t)$$

Si può provare che il cambiamento di forma tra la superficie originale e quella con i dati è limitata da:

$$||\tilde{P}(s,t) - P(s,t)|| \leq \max_{i=1}^{n} ||\epsilon_i||$$

Dal momento che ogni punto sulla T-spline ha tre coordinate, ed in ogni coordinata può essere inserito un numero a virgola mobile, il massimo quantitativo di dati che possono essere inseriti nella superficie T-spline è pari a $3n$, dove n è il numero di punti di controllo.
Se vogliamo inserire una sequenza di numeri binari, possiamo trasformarli in numeri in virgola mobile, per esempio associando al valore binario 1 il valore reale α, e al valore binario 0 il valore reale $-\alpha$, dove α è un numero in virgola mobile positivo, con il quale controllare la *potenza* dell'informazione.
Una maniera per controllarla è quella di definire una tolleranza e imporre che la quantità $m = \max_{i=1}^{n} ||\epsilon_i||$ sia inferiore ad essa, dimezzando di volta in volta il parametro α fino a quando tale condizione non sia verificata, ovvero:

1. si pone $\alpha := 2 \cdot tol$, dove tol è la tolleranza data;
2. si applica il watermark alla superficie e si calcola $m = \max_{i=1}^{n} ||\epsilon_i||$;
3. **while** $(m > tol)$

   $\alpha = \alpha/2$;

   si applica il watermark alla superficie e si ricalcola m;

   end while
4. **return** α

Per l'estrazione, si richiede di avere a disposizione il modello originale: l'algoritmo è dunque non cieco. L'origine del sistema di coordinate dei nodi (s_0, t_0) e le coordinate (s_{j_k}, t_{j_k})) possiedono il ruolo di chiavi per l'estrazione; è sufficiente calcolare gli $\epsilon_i, i = 1, \ldots, n$, ricordando che essi sono pari a $\tilde{\mathbf{P}}_i - \mathbf{P}_i$, e che i dati inseriti si riassumono in $W_k = \sum_{i=1}^{n} \epsilon_i R_i(s_{j_k}, t_{j_k}), k = 1, \ldots, r$. Naturalmente, dai valori a virgola mobile che si ricavano, si può poi risalire agli eventuali numeri binari da cui si era partiti nel caso di conversione.
Per essere più precisi, non è sufficiente nemmeno la superficie originale di per sé al fine di estrarre i dati: è necessario infatti conoscere la sequenza di parametri (s_{j_k}, t_{j_k}), che una parte in possesso del disegno originale ma non del modello completo potrebbe non possedere.
Avevamo detto che in questo tipo di metodi non viene solitamente richiesta la robustezza dei metodi usati, e che l'informazione nascosta può non essere più recuperata se i dati hanno subito troppe manipolazioni. È ciò che accade in questo caso: dal momento che ogni W_k è associato con diversi ϵ_i e $R_i(s_{j_k}, t_{j_k})$, se la superficie è modificata anche leggermente, può non essere facile estrarre i dati presenti con perfezione.
Tuttavia, è possibile con qualche piccolo accorgimento rendere il metodo invariante almeno per traslazione, rotazione, e scaling uniforme, come mostrato in [13]. Scegliamo tre punti non allineati a, b, c dal modello originale, e consideriamo i corrispondenti a', b', c' del modello trasformato: per far coincidere il modello originale con quello trasformato, effettuiamo le seguenti operazioni:

1. calcoliamo la distanza tra due punti, per esempio a e b, del modello originale, e facciamo lo stesso per i due punti corrispondenti,nel caso a' e b', del modello marchiato; calcoliamo il rapporto $|a'b'|/|ab|$ e scaliamo il modello originale di tale fattore;

2. trasliamo il modello originale in maniera tale che un punto venga fatto coincidere con il corrispondente, per esempio a si vada a trovare al posto di a';

3. ruotiamo il vettore ab in maniera tale da farlo coincidere con $a'b'$, e poi ruotiamo ac per farlo coincidere con $a'c'$ mantenendo inalterato ab.

Si veda anche la figura 7.2.

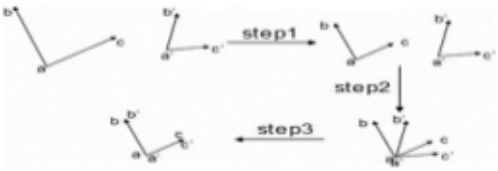


Figura 7.2: Operazioni per far coincidere modello originale e trasformato.

Un esempio dell'applicazione di questa procedura è dato in figura 7.3. La figura 7.3.a mostra una superficie T-spline costituita da 41 punti di controllo, le cui coordinate x, y, z sono tutti numeri in virgola mobile compresi nell'intervallo $[0.1, 1.0]$. Per inserire 123 quantità binarie (bit, valori che possono soltanto essere 0 o 1), generiamo innanzitutto i vettori $W_j = [W_j^x, W_j^y, W_j^z], j = 1, \ldots, 41$, dove W_j^x, W_j^y, W_j^z sono numeri in virgola mobile, posti pari a 0.001 se la quantità binaria da associare è 1, pari a -0.001 se la quantità binaria da associare è 0. Inseriamo W_j all'interno dei punti sulla superficie $P(s_j, t_j), j = 1, \ldots, 41$; la figura 7.3.b mostra la superficie con i dati inseriti.

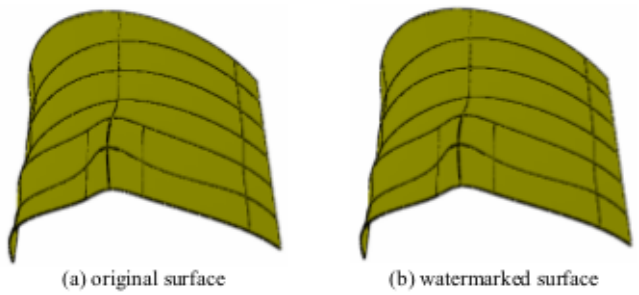


Figura 7.3: Applicazione della procedura di steganografia.

Per estrarre i dati inseriti, calcoliamo innanzitutto i W_j, e ricaviamo i rispettivi numeri binari valutando i segni dei numeri in virgola mobile.
Restano da testare gli errori, ovvero le differenze tra la superficie marchiata e quella originale. Definiamo innanzitutto l'errore assoluto e relativo come:

$$absError = max_{i=1}^{n} ||\epsilon_i|| \quad relError = \frac{\sum_{i=1}^{n} ||\epsilon_i||}{\sum_{i=1}^{n} ||\mathbf{P}_i||}$$

Posto questo, impostiamo 100 sequenze di numeri binari casuali, facendo eseguire il programma per 100 volte. La figura 7.4 mostra gli errori assoluti e quelli relativi riscontratisi di volta in volta; come visibile, essi sono generalmente piccoli, rendendo l'impercettibilità del marchio buona.

7.2.2 Algoritmo per il watermarking robusto che preserva la forma

A differenza della steganografia, il watermarking vero e proprio richiede solitamente la robustezza nei confronti di attacchi che possono essere volti al renderlo inefficace; inoltre, in alcune applicazioni `CAD`, specialmente quelle di stampo industriale, la forma delle superfici deve essere mantenuta a seguito dell'inserimento dei dati, e anche una piccola differenza può risultare non

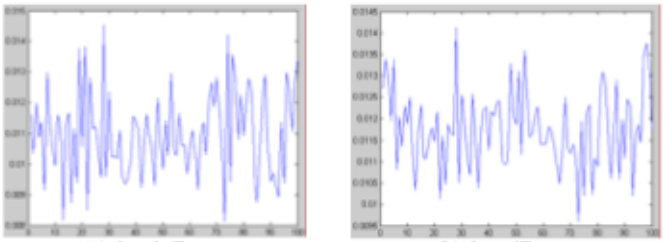


Figura 7.4: Errori assoluti ed errori relativi.

tollerabile.
L'algoritmo di cui discuteremo ora, anch'esso descritto in [14], è basato sull'inserzione locale di nodi nelle superfici T-spline,

7.2.2.1 Inserimento del watermark

Per inserire il watermark, scegliamo innanzitutto l'origine del sistema di coordinate nodali (s_0, t_0) come prima. Ora, scegliamo una curva isoparametrica, o parte di essa; supponiamo di sceglierne una tale per cui sia t ad essere costante, considerano che il caso di s costante si tratta in maniera del tutto simmetrica.
Scegliamo due punti (s_1, t_1) e (s_2, t_2) sulla curva isoparametrica (per quanto detto $t_1 = t_2$) a stabilire i due estremi della sezione che considereremo, come in figura 7.5.

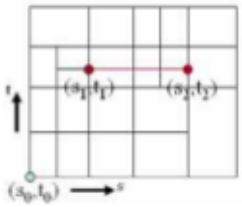


Figura 7.5: Scelta di una sezione di curva isoparametrica a t costante.

Supponiamo che i dati del watermark consistano in 48 bit; raggruppiamoli a quattro a quattro considerando 12 cifre esadecimali $h_1, h_2, \ldots, h_{12}$. Inseriamo il marchio, come detto, utilizzando l'inserzione locale di nodi, con le posizioni di inserzione $\hat{s}_i, i = 1, \ldots, 12$ definite da:

$$\hat{s}_i = s_1 + \frac{(s_2 - s_1)}{12}\left(\frac{h_i + 7.5}{30k_1} + i - 1\right), \quad i = 1, \ldots, 12$$

dove k_1 è la chiave, che va scelta in maniera tale che sia $s_1 < \hat{s}_i < s_2$. L'idea è quella di suddividere l'intervallo $[s_1, s_2]$ in 12 parti, implementando il marchio nella fascia compresa tra 1/4 e 3/4 di ogni parte.

7.2.2.2 Estrazione del watermark

In questo caso l'estrazione del marchio non richiede il modello originale: il procedimento è cieco. A fungere da chiavi sono sufficienti le quantità $k_1, (s_1, t_1), (s_2, t_2), (s_0, t_0)$. Vediamo il procedimento:

1. Scegliamo l'origine del sistema di coordinate dei nodi (s_0, t_0) esattamente come nel procedimento di inserimento del marchio;
2. Troviamo la curva isoparametrica per cui t è costante e pari a t_1;
3. Per ogni h_i, proviamo i valori da 0 a 15 (F) e verifichiamo se esiste il corrispondente $\hat{s}_i$, ricordando che, affinché l'esito sia positivo, per ogni i deve risultare uno e un solo valore di h_i;
4. Se $\hat{s}_i, i = 1, \ldots, 12$ esiste (ed è unico), allora il marchio si ottiene considerando la successione degli h_i.

7.2.2.3 Esempi

Un esempio dell'applicazione di questo procedimento è dato in figura 7.6. La figura 7.6.a mostra la superficie T-spline originale con la relativa T-mesh, in cui vengono inseriti come watermark 48 numeri binari all'interno della terza curva isoparametrica rispetto a t a partire dal basso; la superficie marchiata viene mostrata in figura 7.6.b
La T-mesh marchiata differisce da quella originale per quanto riguarda le parti che coinvolgono la terza curva isoparametrica rispetto a t, ma la forma della superficie non cambia a seguito dell'inserimento del marchio.

Si può provare che questo algoritmo è resistente nei confronti di diversi

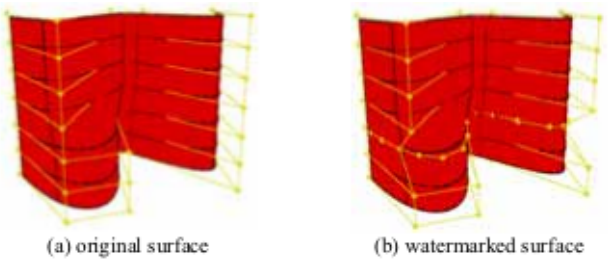


Figura 7.6: Applicazione della procedura di steganografia.

attacchi che non cambiano la forma, quali: scaling degli intervalli tra nodi, raffinamento locale, inserzione di punti di controllo, riparametrizzazione lineare, traslazione dei punti di controllo, rotazione dei punti di controllo. Ciò si verifica in quanto l'algoritmo inserisce i dati in posizioni definite come *relative* al dominio topologico, non assolute.
La resistenza si ha anche verso attacchi che la forma la modificano, quali

spostamento dei punti di controllo che non sia un'isometria e modifica dei valori dei pesi. Questo grazie al fatto che, benché cambi la superficie, non viene modificato il dominio topologico.

Appendice A

Guida a Rhino

A.1 Le geometrie di Rhino

Con *geometrie* si intendono i tipi di struttura geometrica supportati dal software. Strutture di base sono i punti (dimensione 0), le curve `NURBS` (dimensione 1), e le superfici `NURBS` (dimensione 2), mentre strutture più avanzate possono essere quelle che nel linguaggio del programma vengono chiamate *policurve* e *polisuperfici*, oppure i solidi e le mesh poligonali. Vediamo ora nel dettaglio alcune di queste geometrie.

A.1.1 Punti

I punti, o più formalmente gli *oggetti punto*, definiscono un singolo punto nello spazio tridimensionale. Si tratta, come intuitivamente comprensibile, degli oggetti più semplici che Rhino possa definire. Disponibili ovunque nello spazio, vengono utilizzati come semplice riferimento il più delle volte.

A.1.2 Curve

Tralasciando per un attimo il formalismo matematico e/o informatico, e parlando con un linguaggio più intuitivo, la guida paragona una curva ad un pezzo di filo metallico. Con questa similitudine si intende che una curva può essere un qualsiasi elemento lineare, sia esso retto oppure tortuoso, aperto oppure chiuso.
Nella terminologia del software, vengono chiamate *policurve*, termine che abbiamo introdotto già in precedenza, le curve costituite da più segmenti di curva uniti alle estremità. Si tratta di quelle che vengono chiamate anche *curve composte*.
Tra gli strumenti disponibili per disegnare delle curve, si possono citare quelli che permettono di tracciare direttamente linee rette, *polilinee* (analogamente alle policurve, si tratta di segmenti connessi tra di loro), archi, cerchi, poligoni, ellissi, eliche, spirali, in generale i più importanti e utilizzati tipi di

curve aperte e chiuse, piane e non piane.
Tuttavia, può essere di maggiore interesse, in riferimento all'aspetto matematico, la possibilità di realizzare delle curve a partire da un insieme ordinato di punti di controllo, o imponendo il passaggio (interpolazione) per alcuni punti, che comporta il relativo calcolo dei rispettivi punti di controllo.

A.1.3 Superfici (non tagliate)

Sempre parlando informalmente, una superficie si può immaginare come un foglio rettangolare di gomma, deformabile in determinate maniere.
Una superficie `NURBS` può consistere in una forma semplice, come un piano (prendendo tutti i punti di controllo sullo stesso piano, per la proprietà dell'inviluppo convesso la superficie deve coincidere con parte di tale piano), un cilindro, ma anche in una superficie di grande complessità, una superficie *freeform* (a forma libera) come si usa dire.
Tutti i comandi di creazione di superfici di Rhino permettono di realizzare una superficie sotto forma di `NURBS`, nel senso che, indipendentemente dalla complessità, per il programma la superficie costruita viene trattata come tale; è possibile renderle direttamente, oppure a partire da curve già esistenti nell'ambiente di lavoro.
Come è stato visto a lezione, i punti di controllo delle superfici `NURBS` sono a topologia rettangolare, per cui è ragionevole aspettarsi che la struttura di una superficie di questo tipo sia tipicamente rettangolare. Per esempio, una superficie semplice e chiusa, come un cilindro, può essere vista come un foglio rettangolare avvolto in maniera tale che i due bordi opposti coincidono.

A.1.4 Superfici tagliate

Se si vogliono aggirare rapidamente le limitazioni di una topologia rettangolare, è possibile tagliare la superficie, con degli opportuni strumenti proposti dal programma. Una superficie tagliata è composta di due parti: una superficie sottostante che definisce la forma della geometria, e le curve di taglio che individuano i confini di taglio e definiscono la parte di superficie visibile. La parte rimanente, pur continuando ad essere presente nell'ambiente di lavoro, viene rimossa dalla vista. è possibile anche creare direttamente delle superfici tagliate, con altri appositi comandi.
Compiendo le operazioni precedenti, tuttavia la superficie risultante non è più necessariamente una `NURBS`, anzi in generale non lo è affatto, e ciò comporta una serie di limitazioni in quanto può essere possibile svolgere su tale superficie successivamente al taglio.
Per questa ragione, è spesso importante conoscere se la superficie su cui si sta lavorando sia tagliata oppure meno; a tal proposito, il comando **Proprietà** permette di evidenziare lo stato della superficie sotto questo aspetto.
Allo stesso modo, per evitare potenziali perdite di informazioni, ciascuna

superficie tagliata conserva le informazioni sulla relativa superficie di origine, ed è possibile rimuovere i contorni di una curva di taglio per rendere la superficie non tagliata, con il comando **AnnullaTaglio**.
Tali curve definiscono i confini della superficie sottostante, in genere di dimensioni maggiori rispetto ad esse, e non visibile in quanto non viene rappresentato tutto ciò che si trova all'esterno della curva di taglio, pur essendo come detto presente nell'ambiente di lavoro.
Particolare è il caso in cui una curva di taglio attraversa diagonalmente una superficie. In tal caso non avrà alcuna relazione con la struttura dei punti di controllo della superficie stessa, e ciò è visibile selezionando una superficie tagliata attivandone i punti di controllo. Vengono resi visibili i punti dell'intera superficie sottostante.
Altro caso particolare di interesse è quello di una superficie creata da una curva piana e chiusa, che può essere anch'essa una superficie tagliata.

A.1.5 Superfici e curve isoparametriche

Rhino permette di impostare una modalità di visualizzazione, la **Wireframe**, che fa apparire le curve isoparametriche. Tali curve, che nella teoria si ottengono fissando uno dei due parametri della superficie e facendo variare l'altro, nella pratica aiutano a visualizzare la forma della stessa. Pur non definendo la superficie, a differenza, per esempio, dei poligoni nelle mesh poligonali, rappresentano un utile aiuto visuale per rappresentare la superficie sullo schermo. Selezionando una superficie, un certo numero di curve isoparametriche viene evidenziato all'interno dell'ambiente di lavoro. Un caso particolare delle curve isoparametriche è costituito dalle curve di bordo, che rappresentano i confini della superficie; sono quelle che si ottengono fissando uno dei due parametri con il valore minimo o massimo del rispettivo intervallo di variazione. Spesso tali curve vengono utilizzate come input per altri comandi.

A.1.6 Polisuperfici e solidi

Analogamente alle policurve, in Rhino vengono chiamate *polisuperfici* gli oggetti costituiti da due o più superfici unite tra di loro. In una polisuperficie i punti di controllo non sono visualizzabili, ma è possibile con gli opportuni strumenti esplicitarne la suddivisione nelle varie superfici di cui è composta, editarle separatamente, e quindi riunirle.
Una superficie o una polisuperficie che racchiude un volume di spazio definisce un *solido*. Un tale oggetto viene generato ogni volta che uno degli oggetti precedentemente nominati viene chiuso completamente.
Rhino è in grado di creare sia solidi a singola superficie che solidi basati su polisuperfici, anche se i punti di controllo sono direttamente visualizzabili solo nel primo caso, nel secondo caso bisogna passare attraverso la scompo-

sizione.
Esempi di solidi a singola superficie sono dati da solidi di rotazione quali sfere, toroidi ed ellissoidi; solidi polisuperfici sono invece parallelepipedi, coni, tronchi di cono, e cilindri. Si noti che, benché alcuni dei solidi del secondo elenco siano più comuni di un solido come il toro, dal punto di vista del software tali solidi sono più impegnativi da realizzare.

A.1.7 Mesh poligonali

Molti programmi di modellazione utilizzano, piuttosto che le `NURBS`, le mesh poligonali, per scopi quali rendering, animazione delle geometrie, stereolitografia, visualizzazione e analisi di elementi finiti. A tale proposito è stato introdotto il comando **Mesh** per convertire le geometrie `NURBS` in mesh poligonali, e poterle esportare verso altri software. Inoltre, tale comando è in grado di generare oggetti mesh.
Benché invece la conversione di un modello mesh in un modello `NURBS` non sia altrettanto semplice o scontata, essendo le modalità interne di definizione della geometria profondamente diverse, Rhino dispone di diversi comandi in grado di assistere l'utente nella costruzione di geometrie `NURBS` a partire da modelli mesh. Sono ad esempio disponibili alcuni comandi che consentono l'estrazione dei vertici di una mesh o il disegno di curve su superfici di tale tipo.

A.2 Editing di curve e superfici

Come abbiamo introdotto precedentemente, per l'utilizzo in ambito professionale può essere necessario compiere operazioni un po' differenti da quelle previste tradizionalmente dai modelli `NURBS`, quali possono essere lo spostamento di un punto di controllo o la variazione di un peso; esse possono consistere per esempio in tagli o divisioni, e in generale l'oggetto che si ottiene non è più, almeno nella forma, ma talvolta anche nella sostanza, di tipo `NURBS`. A tal proposito Rhino mette a disposizione una serie di comandi, che permettono per esempio di connettere o di suddividere curve e superfici.

A.2.1 Unisci, Esplodi, Tronca, Suddividi

Il comando **Unisci** permette di connettere curve o superfici, componendo con esse un unico oggetto. è lo strumento con cui si possono creare policurve e polisuperfici a partire dai rispettivi oggetti semplici.
Il comando **Esplodi** compie l'operazione inversa, vale a dire, data una policurva o una polisuperficie, rimuove la connessione. Come detto in precedenza, in una polisuperficie, non vengono visualizzati i punti di controllo, che invece sono rappresentati nelle superfici; tale comando è quindi utile qualora si voglia modificare ciascuna singola superficie in termini dei rispettivi punti

di controllo.
I comandi **Tronca** e **Suddividi** permettono invece di effettuare un taglio sull'oggetto, con la differenza che, mentre il primo comando, elimina le parti selezionate, il secondo le conserva. In particolare, tale comando permette di suddividere una superficie con una curva, un'altra superficie, una polisuperficie, o anche le sue stesse curve isoparametriche.
Ricordiamo inoltre il già citato comando **AnnullaTaglio**, che rimuove la curva di taglio di una superficie, ed offre un'opzione per mantenere tale curva, al fine di poterla utilizzare di nuovo in caso di necessità.

A.2.2 Editing dei punti di controllo

Come ampiamente visto nella trattazione teorica, il sistema più immediato per modificare la forma di una curva o di una superficie `NURBS` consiste nel modificarne i suoi punti di controllo, in termini di posizione quanto in termini di peso. Rhino rende disponibili svariati strumenti per modellare curve tramite questo sistema. Sono presenti comandi come **Ricostruisci**, **Normalizza**, **SmussaOggetto**, che permettono di ridistribuire i punti di controllo in una maniera guidata. In alternativa è possibile controllare manualmente la posizione di singoli punti o gruppi di punti di controllo, direttamente tramite lo spostamento e il trascinamento degli stessi, o indirettamente tramite comandi quali **EditorManiglie** e **MuoviUVN**.
Per agire direttamente su tali punti, si utilizza il comando **PuntiOn**, richiamabile immediatamente con **F10**, per attivarli. Al termine delle modifiche, l'analogo comando **PuntiOff**, richiamabile con **Esc**, permette di disattivarli. Si ricorda che i punti di controllo delle polisuperfici non possono essere attivati per l'editing. Il motivo per cui è stata introdotta questa limitazione nel programma è che, la modifica non oculata dei punti di controllo di una polisuperficie, potrebbe separare i bordi delle superfici che la compongono, e creare in essa delle fessure.
Spostando i punti di controllo di una curva, essa viene ridisegnata in modo smussato. Come ben noto dalla teoria, la curva si sposta verso le nuove posizioni assunte dai punti, senza passare per essi; questa proprietà di base delle curve `NURBS` garantisce una deformazione morbida. Una volta attivati i punti di controllo, è possibile utilizzare i comandi di trasformazione di Rhino per modificarli; è possibile parimenti ricostruire una superficie aggiungendo e ridistribuendo i suoi punti di controllo.
Per aumentare il controllo sulla forma di una curva, è possibile aggiungere ulteriori punti di controllo, come la trattazione teorica ci insegna; modificando quelli già presenti, è possibile rimuovere le discontinuità che eventualmente ci sono, creando curve uniformi con la possibilità di aggiungere e rimuovere dettagli. Infine, il tasto **Canc** permette di cancellare uno o più punti di controllo di una curva, modificandone di conseguenza la forma.

A.3 Trasformazioni

Una componente matematica essenziale per il CAGD è costituita dalle trasformazioni spaziali. Esse includono le operazioni di spostamento, copia, copia speculare, disposizione in serie, rotazione, scaling, shear, e altre ancora.

A.3.1 Comandi

Il comando **Sposta** viene utilizzato per spostare un oggetto di una certa distanza, o per sistemarlo con maggior precisione, tramite l'uso degli snap. Il modo più semplice di spostare un oggetto è il classico *drag and drop*: si seleziona l'oggetto con un clic e lo si trascina. La pressione del tasto **Alt** insieme alle **frecce** di spostamento permette di effettuare piccoli spostamenti con passo costante, per posizionamenti di precisione.
Il comando **Copia** permette di creare delle copie di un oggetto. Alcuni dei comandi che vedremo in seguito possiedono un'opzione Copia, che permette all'utente di effettuare delle copie degli oggetti a cui viene applicato il rispettivo comando, piuttosto che avere il comando stesso applicato sull'oggetto di origine. Anche la copia può essere effettuata tramite drag and drop, tenendo premuto **Alt** mentre si trascinano gli oggetti.
Il comando **Ruota** permette di ruotare un oggetto in relazione al piano di costruzione.
I comandi realtivi al ridimensionamento di un oggetto forniscono all'utente il controllo sulla direzione della scala. è possibile scalare gli oggetti in modo uniforme, o con un fattore di scala diverso per ogni direzione.
Il comando **CopiaSpeculare** crea una copia dell'oggetto, ma il cui orientamento è invertito rispetto ad un asse di riflessione, confrontato con l'oggetto di origine. Di default il comando crea una copia, ma è possibile, tramite un'apposita opzione, riflettere l'oggetto di origine senza generarne una copia.
I comandi di orientamento combinano le operazioni di movimento, copia, scala e rotazione, permettendo di definire la posizione e la dimensione degli oggetti con un unico comando. A tal proposito, si ricordi dalla teoria che, le operazioni di movimento, scala, e rotazione, sono definite da una matrice (anche la traslazione se si lavora in coordinate proiettive), e che una composizione qualsiasi di tali operazioni può essere espressa come prodotto matriciale, quindi a sua volta con una matrice.
Infine, si segnala il comando **Serie**, che permette di distribuire copie di un oggetto spaziandole in un modo uniforme su righe e colonne.

A.4 Analisi di curve e superfici

Rhino dispone di avanzati strumenti di analisi e valutazione, per fornire informazioni dettagliate dal punto di vista matematico sugli oggetti che

considera.

A.4.1 Misura di distanza, angolo e raggio

È possibile chiedere al programma informazioni riguardo posizioni, distanze, angoli, e raggi di curvatura, per esempio. In particolare:

- **Distanza** visualizza la distanza tra due punti.
- **Angolo** visualizza l'angolo tra due linee.
- **Raggio** visualizza il raggio di una curva in uno qualsiasi dei suoi punti.
- **Lunghezza** visualizza la lunghezza di una curva.
- **ValutaPt** visualizza le coordinate di un punto qualsiasi.

A.4.2 Direzione di curve e superfici

Curve e superfici sono caratterizzate da direzioni: supponendo di lavorare nella situazione standard in cui i parametri variano da 0 a 1, nel caso delle curve la direzione va dall'estremo in cui il parametro della curva assume il valore 0, che viene preso come punto iniziale, fino al punto in cui tale parametro assume il valore 1, in corrispondenza del punto finale.
Nel caso delle superfici, ogni punto identifica due direzioni, quella in corrispondenza della quale uno dei due parametri (diciamo u) aumenta e l'altro (diciamo v) rimane costante, e quella in cui accade il viceversa.
Rhino permette di visualizzare queste direzioni con il comando **Dir**.
Nel caso di una curva, vengono mostrate le frecce di direzione, dirette dall'inizio verso la fine della curva. In particolare, se non ci sono state modifiche, la direzione indicata è quella della curva originaria.
Nel caso delle superfici, vengono visualizzate le due direzioni u, v, e la normale alla superficie. Naturalmente, tale direzione è immediatamente riconoscibile in quanto le frecce che la rappresentano sono perpendicolari alla superficie, mentre le due direzioni coordinate sono rappresentate da frecce adagiate sulla superficie. Le normali di superfici chiuse vengono di solito dirette verso l'esterno. Il comando in questione permette anche di modificare tali direzioni, importante se si intende applicare una texture ad una superficie.

A.4.3 Curvatura

Altre verifiche che possono risultare utili sono quelle sul valore della curvatura in un punto di una curva, la visualizzazione della circonferenza di curvatura, e la verifica della continuità delle curve nei punti di break.
A tale scopo, è presente il comando **GraficoCurvatura**, che permette di

visualizzare il grafico di curvatura di curve e superfici. Le linee del grafico rappresentano la direzione perpendicolare alla curva in quel dato punto, e la lunghezza della linea evidenzia il valore (in modulo) della curvatura.

A.4.4 Analisi visiva di superfici

Determinare le irregolarità delle superfici valutandone analiticamente curvatura, tangenza, e proprietà di questo tipo, può essere meno intituivo che farlo per le curve. Per venire incontro a tale evenienza, Rhino introduce dei comandi di analisi visiva per le superfici, che utilizzano la stima di una superficie `NURBS` e le tecniche di rendering per visualizzare il livello di smusso di una superficie, attraverso una mappa di riflessione e la proiezione di colori e strisce, che consentono di valutarne curvatura ed eventuali discontinuità.
Il comando **AnalisiCurvatura** permette di analizzare la curvatura di una superficie, utilizzando una mappa in falsi colori. Viene permesso di analizzare la curvatura gaussiana, quella media, quella minima e quella massima.
Il comando **MappaAmbiente** permette di visualizzare una bitmap sull'oggetto in modo che la scena sembri riflessa da una superficie metallica lucida. Gli strumenti disponibili consentono di individuare e correggere eventuali difetti riscontrati sulla superficie. In particolare, la mappa di ambiente con anelli fluorescenti simula delle sorgenti di luce che illuminano una superficie metallica riflettente.
Il comando **Zebra** visualizza una superficie con delle strisce riflesse. In questo modo, vengono messi in evidenza gli eventuali difetti, e la condizione di curvatura e tangenza tra superfici adiacenti.
Il comando **AnalisiAngoloDiSformo** visualizza in falsi colori l'angolo di sformo relativo al piano di costruzione attivo al momento dell'esecuzione del comando; la direzione di proiezione per questo comando è l'asse z del piano di costruzione.

A.4.5 Valutazione dei bordi

Molti problemi relativi ad una geometria, come il fallimento di un'unione, possono essere causati dai bordi interrotti delle superfici, oppure dallo spostamento dei bordi con operazioni di editing dei punti. Un *bordo* è un oggetto separato che fa parte della rappresentazione del contorno di una superficie. Per questa evenienza, può essere utile far intervenire il comando **MostraBordi**, che evidenzia tutti i bordi di una superficie.
Ciò può riscontrarsi di particolare utilità nel caso dei solidi. Nonostante una polisuperficie possa apparire chiusa, può infatti accadere che non lo sia affatto. Soltanto una verifica delle sue proprietà, compiuta con l'omonimo comando, può confermarlo o negarlo. Alcune operazione e requisiti di esportazione di un modello richiedono che le polisuperfici siano chiuse, per cui questa informazione può essere particolarmente importante. La qualità

di un modello costituito da polisuperfici chiuse è naturalmente migliore di quella di un modello in cui siano presenti delle imperfezioni o piccole fessure dovute alla presenza di bordi non uniti.
Per tutti questi motivi, Rhino rende disponibili alcuni strumenti per l'individuazione di possibili bordi aperti. Una superficie viene definita *aperta* quando non risulta unita ad alcuna altra superficie, e analogamente si parla di *polisuperficie aperta* quando si considera una polisuperficie che presenta dei bordi aperti. L'utilizzo del comando che permette di mostrare i bordi che abbiamo nominato prima permette di visualizzare i bordi non uniti, insieme all'altro che permette di esaminare le proprietà e i dettagli di un oggetto.
Rhino dispone anche di strumenti per suddividere un bordo (**SuddividiBordo**), per unire bordi coincidenti ad un'estremità (**UnisciBordi**), o per unire superfici aventi bordi aperti (**UnisciBordiAperti**). In base alle tolleranze interne, è possibile ricostruire i bordi (**RicostruisciBordi**).

A.4.6 Diagnostica

Esistono infine alcuni strumenti di diagnostica che permettono di fornire informazioni sulla struttura dei dati di un oggetto e di selezionare gli oggetti che presentano delle imperfezioni. Comandi quali **Rapporto**, **Verifica**, **SelOggettiImperfetti**, e **FileAudit3dm**, permettono di diagnosticare i problemi eventualmente presenti in una superficie in fase di modellazione, mostrandosi di grande utilità.

A.5 Alcuni esempi di utilizzo

Vediamo ora alcuni esempi vari di realizzazioni elementari che possono essere svolte con il software.

A.5.1 La schermata iniziale

Aprendo il software, la schermata si presenta come visibile in figura A.1.

Ciò che balza subito all'occhio sono le visualizzazioni predefinite dell'ambiente di lavoro. Rhino presenta sullo schermo quattro finestre, tre di esse corrispondenti alle rispettive proiezioni ortogonali, quella chiamata **Superiore**, che visualizza la proiezione sul piano $z = 0$, la **Frontale**, ovvero la proiezione sul piano $y = 0$, la **Destra**, che sarebbe quella sul piano $x = 0$. La quarta finestra, **Prospettica**, effettua una visualizzazione tridimensionale con l'uso della prospettiva. Le schermate di visualizzazione sono modificabili, è possibile per esempio ruotarle per far vedere lo spazio di lavoro sotto angoli di azimut ed elevazione arbitrari, ma quelle predefinite sono le più utili per quanto riguarda gli esempi che svolgeremo.

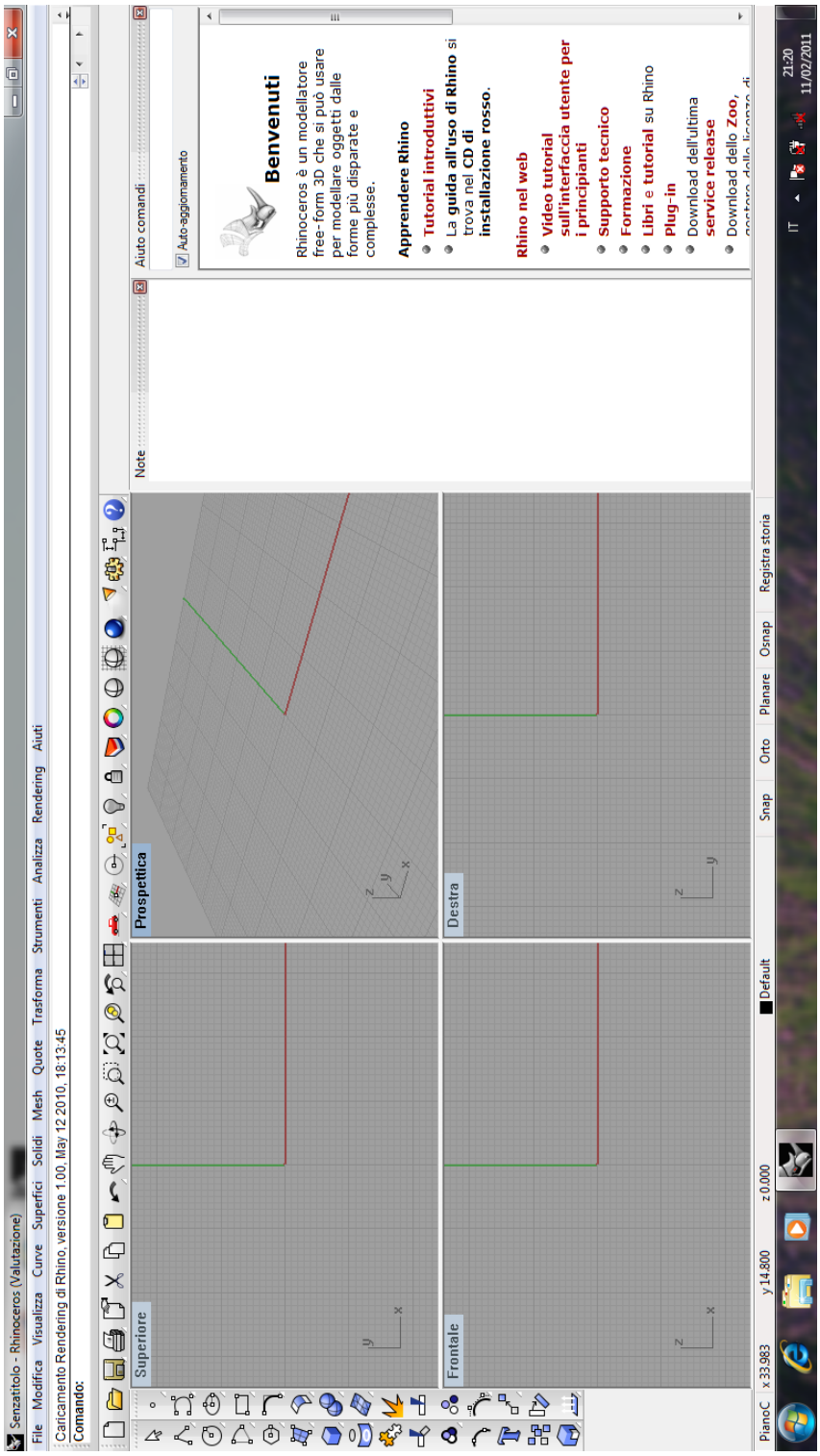


Figura A.1: Schermata iniziale

A.5.2 Tracciatura di una curva B-spline piana

Vediamo ora come realizzare una curva B-spline avente punti di controllo scelti e di grado scelto. Selezioniamo, come visibile in figura A.2, con il tasto sinistro del mouse il pulsante della toolbar che permette di realizzare una curva per punti di controllo; premendolo con il tasto destro, si sarebbe realizzata una generica curva per punti, con la possibilità eventualmente di tracciare proprio una curva per punti di controllo, ma avendo già definito i punti e modificando il valore della relativa opzione dopo. In questo modo, invece, la tracciatura della curva è immediata all'atto della definizione dei punti.

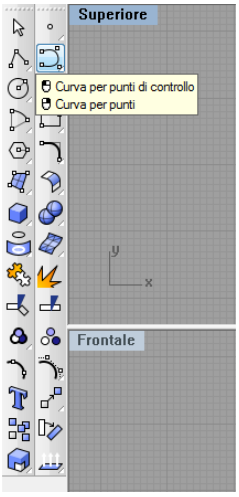


Figura A.2: Selezione

Operata la selezione, vediamo in figura A.3, che ci viene chiesto dove far iniziare la curva. Possiamo però decidere prima di variarne il grado, il cui valore predefinito è 3; cliccando su Grado, il prompt dei comandi rende digitabile il nuovo grado della curva (al massimo 11), come visibile in figura A.4.

Figura A.3: Grado predefinito

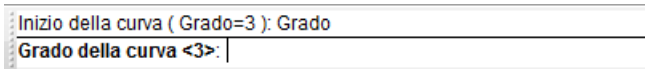


Figura A.4: Grado cambiato

Va da sé che, se il numero dei punti di controllo non è sufficiente per generare una curva in un determinato grado, viene plottata la curva che ha tali punti di controllo e il massimo grado possibile, ovvero la corrispondente curva di Bezier. Ad ogni modo, per quanto concerne i nostri scopi, una curva di grado 3 va bene.
Selezioniamo pertanto i punti di controllo, lavorando sul piano xy, e tracciamo la nostra curva piana (figura A.5).

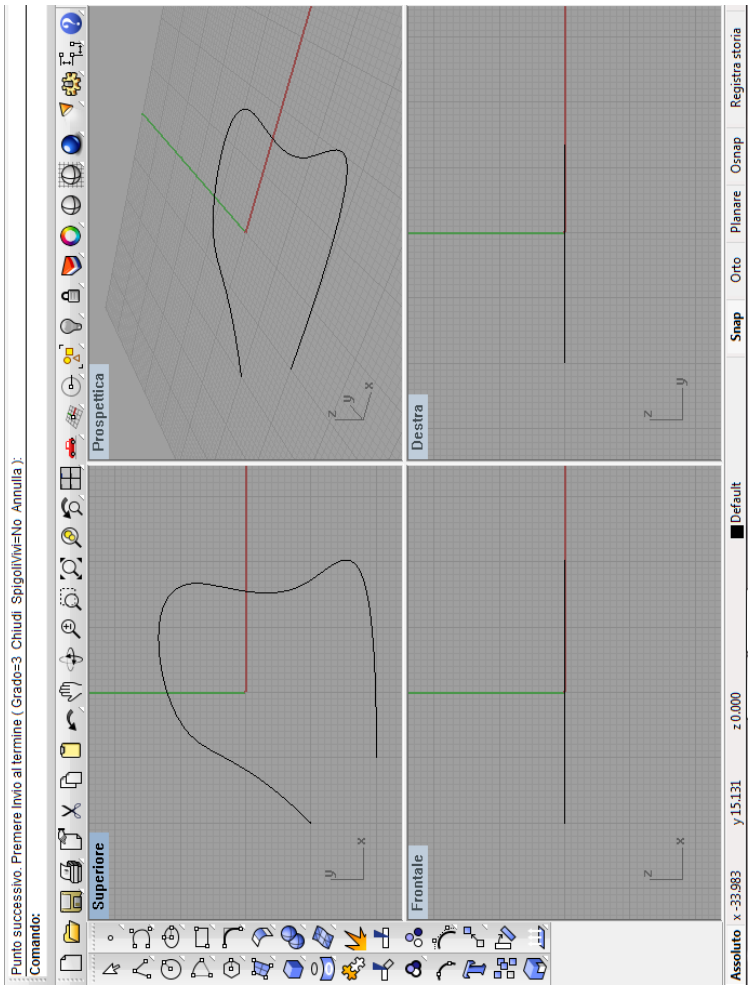


Figura A.5: Curva tracciata

Notare che le due proiezioni Frontale e Destra si riducono ad un insieme di segmenti collineari, dal momento che $z = 0$ per ogni punto della curva. Facciamo ora visualizzare i relativi punti di controllo, la cui visualizzazione di default viene meno al momento il quale la curva viene completata. Per farlo, andiamo sull'icona evidenziata in figura A.6 e selezioniamo la curva, ottenendo il risultato visibile nella stessa immagine.

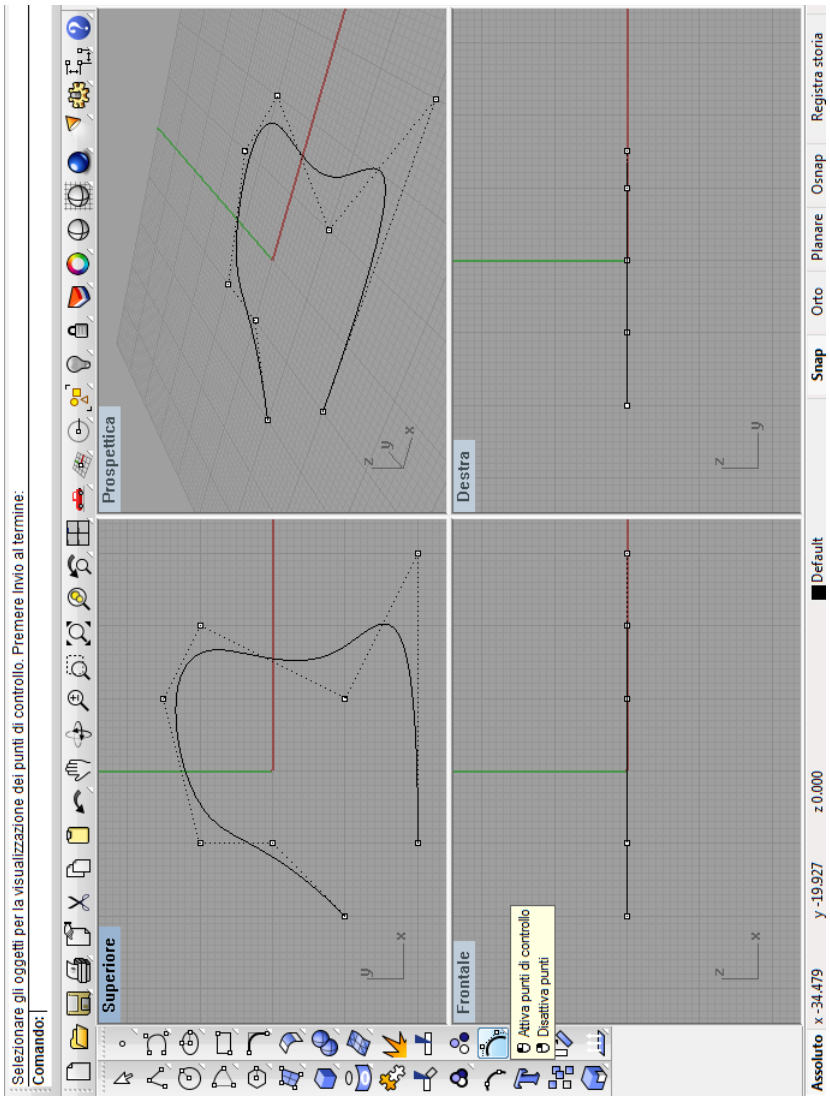


Figura A.6: Punti di controllo in evidenza

Per modificare la posizione di un punto di controllo, è sufficiente selezionarlo e trascinarlo; ciò che segue da una tale operazione è illustrato in figura A.7. A livello teorico, la scelta del vettore dei nodi, caratterizzante della curva B-spline che risulta, viene effettuata in automatico.

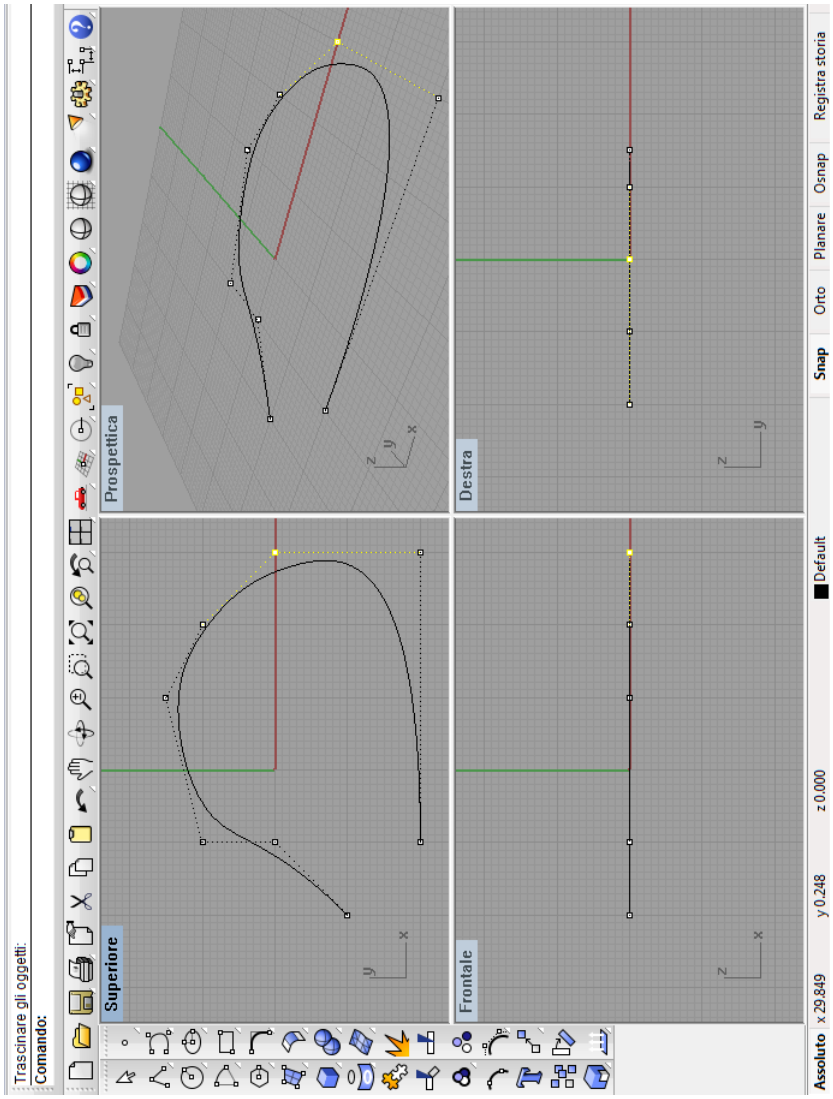


Figura A.7: Spostamento di un punto di controllo

A.5.3 Tracciatura di una curva B-spline nello spazio

Il metodo è analogo alla tracciatura di una curva nel piano, ma può essere utile, se non si vogliono modificare le visualizzazioni, dare i punti direttamente dalla riga di comando, fornendo nell'ordine le tre coordinate x, y e z separate da una virgola come delimitatore, come visibile in figura A.8.

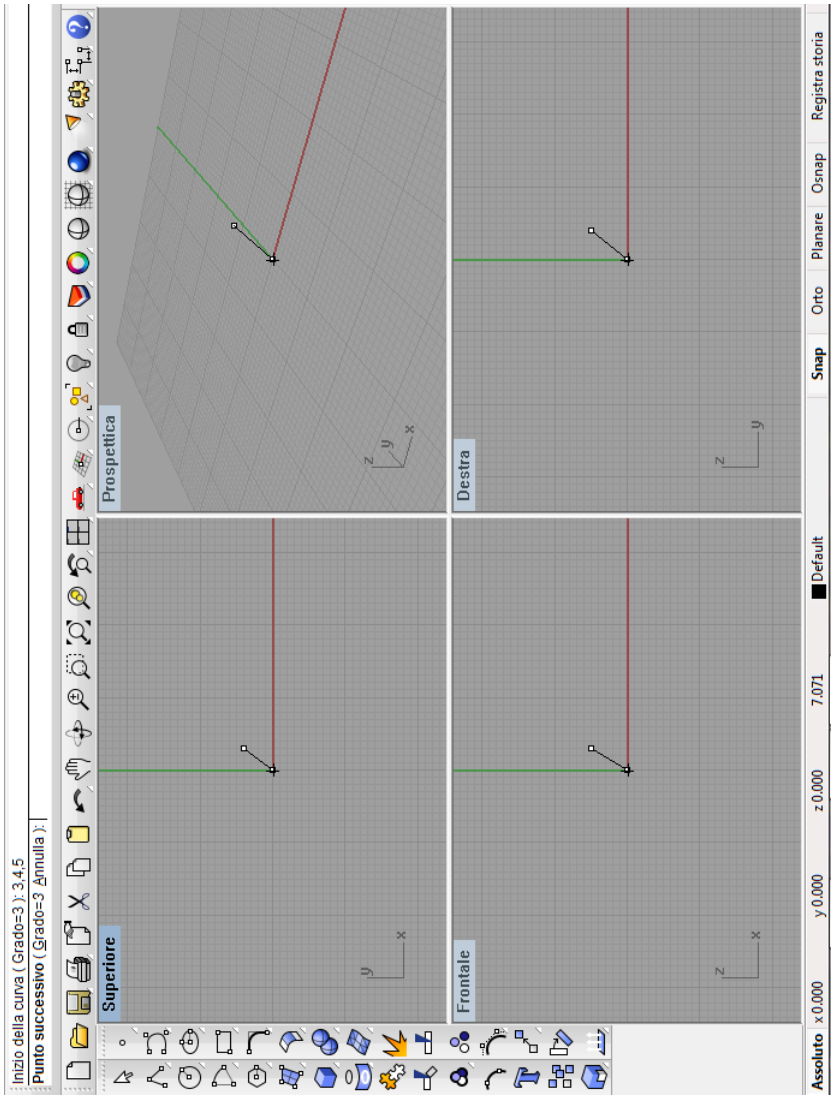


Figura A.8: Punto nello spazio 3D assegnato da riga di comando

A.5.4 Costruzione di una superficie B-spline

Per realizzare una superficie B-spline, ci possono tornare utili due comandi, digitabili direttamente dalla riga di comando, oppure richiamabili attraverso un pulsante, che tuttavia non si trova nelle due toolbar principali visualizzate di default, bensì nella toolbar Superficie, che si può inserire cliccando col tasto destro nello spazio delle toolbar sotto i pulsanti e spuntando il relativo flag. Il primo dei due comandi è disponibile anche dal menu Superficie della barra superiore.
Si tratta di **SrfGrigliaPt** (o **SrfPtGrid**), e **SrfGrigliaPtControllo** (o **SrfControlPtGrid**). Il primo comando permette di creare una superficie che interpola una griglia di punti, mentre il secondo crea una superficie che prende i punti dati come punti di controllo. I punti possono essere forniti per via grafica, o ancora dalla riga di comando, secondo la stessa sintassi del caso precedente.
La figura A.9 mostra il risultato dell'esecuzione dei due comandi per lo stesso insieme di punti dati. Vengono tracciati con una linea più spessa i bordi della superficie, e con una linea più sottile le isocurve, in questo caso una per ognuna delle due direzioni coordinate.
Si vede che le due figure sono distinte, dal momento che i punti di controllo, se si eccettuano quelli ai quattro angoli, non sono punti di interpolazione, tranne naturalmente nel caso banale di superfici di grado 1, che coincidono con il proprio poliedro di controllo.
Come nel caso delle curve, con lo stesso pulsante, previo selezionamento della superficie di cui si desidera farlo, è possibile modificare i punti di controllo, e la superficie cambierà di conseguenza.

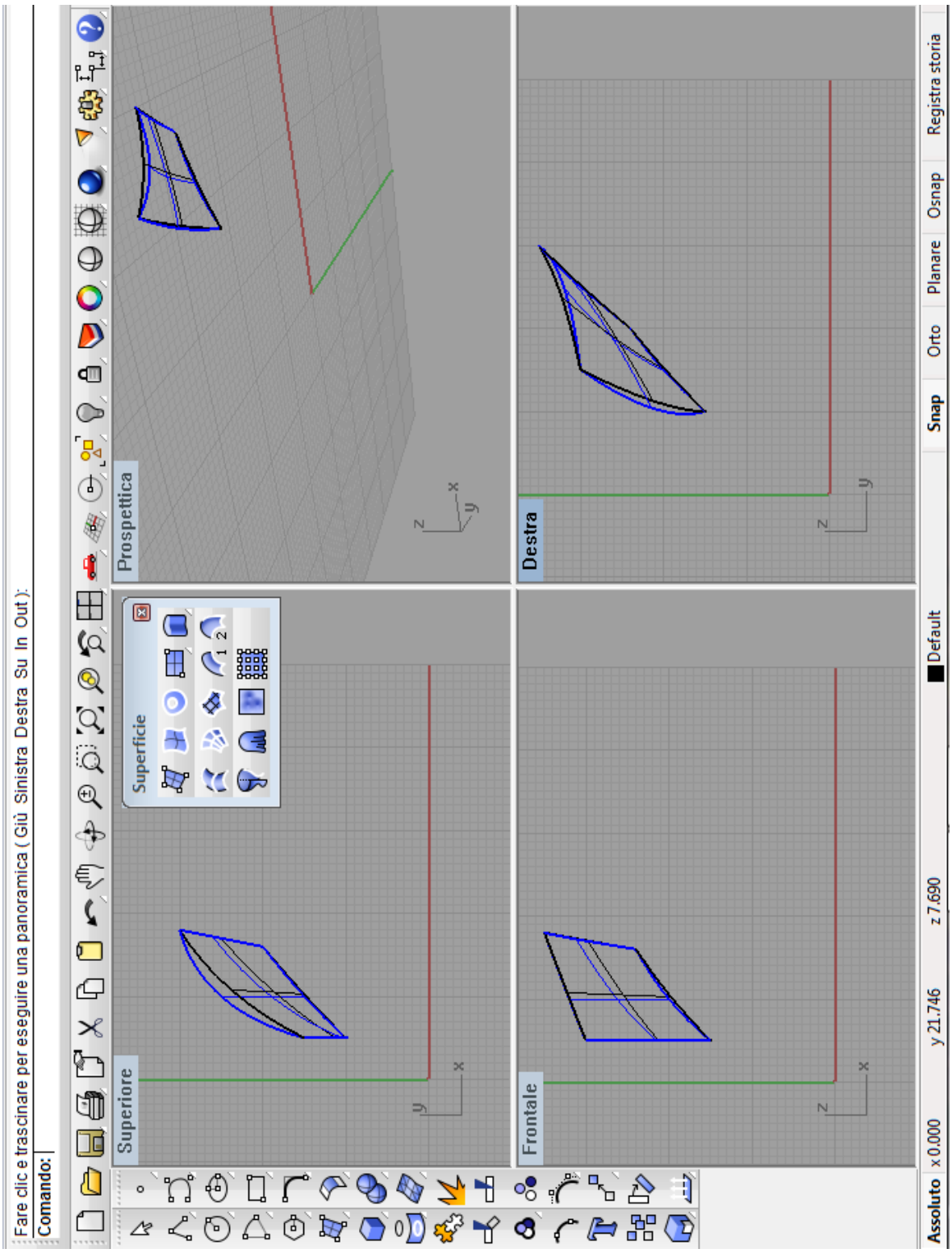


Figura A.9: Superficie interpolante e non

Appendice B

I comandi dell'add-on T-spline

Presentiamo ora un elenco di tutti i comandi messi a disposizione dall'add-on. La *Posizione* indica dove si trova, di default, l'icona nel rispettivo toolbox, nell'ipotesi che esso sia ridimensionato per stare nella colonna delle note, come in figura (7 icone per riga). La notazione è (*Riga*, *Colonna*).
Il *Comando sinistro* indica l'azione che permette di eseguire l'icona se cliccata con il tasto sinistro del mouse.
Il *Comando destro* indica l'azione che permette di eseguire l'icona se cliccata con il tasto destro del mouse.

Novità della V3

Icona	Posizione	Comando sinistro	Comando destro
	$(1,1)$	tsFormaTubolare	-
	$(1,2)$	Set di selezione	-
	$(1,3)$	Simmetria on	Simmetria off
	$(1,4)$	Riporta punti di controllo	-
	$(1,5)$	Combina superfici	-
	$(1,6)$	Interpola superficie su punti di controllo	-
	$(1,7)$	SferaQuad	-
	$(2,1)$	Smussa bordi	-
	$(2,2)$	Scorrimento bordi	-
	$(2,3)$	Impostazioni snap Ritopo	-

	$(2,4)$	ts Successivo U	ts Prec U
	$(2,5)$	ts Successivo V	ts Prec V
	$(2,6)$	Incrementa anello	Ritira anello
	$(2,7)$	Incrementa ciclo V	Ritira ciclo
	$(3,1)$	Estrai bordi	-
	$(3,2)$	Modifica UV	Punti interattivi della mesh texture attivati

Tutti i comandi

Icona	Posizione	Comando sinistro	Comando destro
	$(1,1)$	Attiva modalità di editing	Disattiva modalità di editing
	$(1,2)$	Converti `NURBS` o mesh in T-spline	Converti in `NURBS`, polisuperficie o mesh di Rhino
	$(1,3)$	Cubo	-
	$(1,4)$	Forma tubolare	-
	$(1,5)$	Aggiungi faccia	-
	$(1,6)$	T-spline da linee	Estrai poligono di controllo
	$(1,7)$	Loft della superficie T-spline	-
	$(2,1)$	Adatta T-spline a curve	-
	$(2,2)$	Commuta morbido	-
	$(2,3)$	Estrudi facce, bordi o curve	-
	$(2,4)$	Simmetria on	Simmetria off
	$(2,5)$	Assegna spessore	-
	$(2,6)$	Piega	Rimuovi piega
	$(2,7)$	Inserisci punto (semplice)	Inserisci punto (esatto)

	(3, 1)	Inserisci bordo (semplice)	Inserisci bordo (esatto)
	(3, 2)	Smussa bordi	-
	(3, 3)	Scorrimento bordi	-
	(3, 4)	Aggiungi faccia	-
	(3, 5)	Riempi foro	-
	(3, 6)	Suddividi faccia (esatto)	Suddividi faccia (semplice)
	(3, 7)	Facce identiche	-
	(4, 1)	Cancella	-
	(4, 2)	Salda punti	-
	(4, 3)	Fondi bordi	Annulla saldatura bordi
	(4, 4)	Collega	-
	(4, 5)	Combina superfici	-
	(4, 6)	Riporta punti di controllo	-
	(4, 7)	Set di selezione	-
	(5, 1)	Interpola superficie su punti di controllo	-
	(5, 2)	Impostazioni snap Ritopo	-
	(5, 3)	Spiana punti	-
	(5, 4)	Assegna peso ai punti T-spline	-
	(5, 5)	Modifica UV	Punti interattivi della mesh texture attivati
	(5, 6)	Rendi uniforme	-
	(5, 7)	Modifica layout	-
	(6, 1)	Standardizza	-

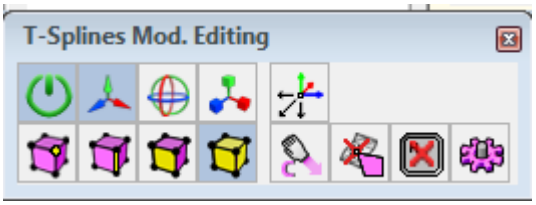
	(6, 2)	Inverti normali	-
	(6, 3)	Definisci regioni per la conversione in `NURBS`	Seleziona bordi per GraficoCurvatura
	(6, 4)	Converti T-spline in mesh	-
	(6, 5)	Estrai bordi	-
	(6, 6)	Suddividi curve	-
	(6, 7)	Estrudi linee	-
	(7, 1)	Opzioni	Informazioni su T-Splines

B.1 I comandi nel dettaglio

Vediamo adesso le funzionalità dei vari comandi introdotti dall'add-on, con particolare interesse all'aspetto matematico presente alle spalle. Considerata la natura dell'add-on, se non diversamente specificato, ogni volta che si parlerà di "T-spline" si intenderanno in generale T-spline semplici oppure `T-NURCC`.

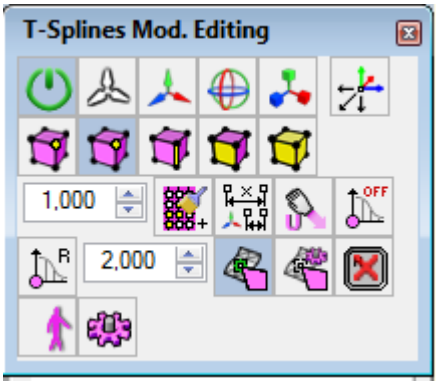
 Attiva modalità di editing

Questo comando apre una toolbar posta di default nella parte sinistra nello schermo, ma che può essere disancorata e resa una toolbox posizionabile ovunque, come è stato fatto qui principalmente per questioni tipografiche (evitando un'immagine molto stretta e molto lunga).



Secondo questa disposizione dei pulsanti, la riga superiore di questa toolbox, a parte il primo pulsante che permette di disattivare la bar (o la box), contiene le cosiddette funzionalità di *manipolazione*: nella fattispecie, il secondo permette di traslare, il terzo di ruotare, il quarto di scalare. L'ultimo pulsante, leggermente defilato dagli altri, permette di modificare il punto di perno, permettendo scaling e rotazioni da un altro punto e non dal centro.
I quattro pulsanti di sinistra della riga inferiore permettono di modificare il

tipo di selezione, tra gli oggetti di dimensione 0 (i vertici), quelli di dimensione 1 (i bordi), quelli di dimensione 2 (le facce), e quelli di dimensione 3 (gli oggetti solidi).
I quattro pulsanti di destra ineriscono funzionalità tecniche del software, piuttosto che aspetti geometrici.
Dalle opzioni (pulsante più in basso a destra), è possibile aggiungere altri pulsanti rispetto a quelli predefiniti, che permettono, tra le altre cose, di manipolare gli oggetti in maniera più *morbida*, ovvero tale per cui lo spostamento di un vertice eserciti un effetto più graduale sulla superficie adiacente, anche se Rhino non ci dà indicazioni su cosa voglia dire ciò di preciso a livello matematico.



Converti NURBS o mesh in T-spline

Questo comando permette di convertire una NURBS o una mesh in una T-spline. Soffermiamoci sulla funzionalità relativa alla conversione NURBS. Abbiamo visto nel § 6.1.1 che Rhino permette di tagliare le superfici; tali tagli non sono però supportati dalle superfici T-spline, che perciò perdono le informazioni relative ad essi in caso di conversione.
Inoltre, Rhino può lavorare soltanto con superfici T-spline di grado 3. Abbiamo detto fin da subito che tutta la teoria sulle T-spline può essere estesa al caso di superfici di grado qualsiasi, in particolar modo di grado dispari (nel caso pari ci sarebbe qualche complicazione in più dovuta alla questione di un verso *privilegiato* per il vettore dei nodi), ma abbiamo anche visto che di fatto tutti gli studi compiuti e tutte le costruzioni realizzate sull'argomento utilizzano le cubiche, e anche questo add-on si limita ad usare oggetti di terzo grado.
La conversione da NURBS a T-spline può quindi essere effettuata direttamente se la prima ha grado pari a 3. Se la NURBS ha grado almeno 4, si può ancora effettuare una conversione esatta, anche se questa comporta la loro ricostruzione al grado 3; se la NURBS ha grado 1 o 2, una ricostruzione al grado 3 può comportare una leggera modifica della stessa, tuttavia è possibile aumentarne semplicemente il grado garantendo il mantenimento esatto

della forma, ma ciò comporta la presenza di punti di controllo sovrapposti (in quanto in particolare si chiede di mantenere l'ordine di continuità, che invece aumenterebbe salendo di grado, e il modo di contrastare questo incremento è proprio l'utilizzo di punti di controllo multipli), che renderanno più difficile manipolare la superficie T-spline in maniera morbida.

 **Converti in NURBS, polisuperficie o mesh di Rhino**

La funzionalità attivata dal tasto destro del mouse permette fondamentalmente di compiere l'operazione inversa: a partire da un modello T-spline, generare una superficie NURBS equivalente, ovvero convertita precisamente, senza deformare in alcun modo la superficie ed in particolare mantenendo gli ordini di continuità.
Come abbiamo già detto, ciò è strettamente possibile se e soltanto se il modello T-spline non contiene punti straordinari, ovvero se si tratta di una T-spline propriamente detta; in caso contrario, nel quale siamo in presenza di una T-NURCC che non è una T-spline semplice, la conversione potrà essere effettuata comunque, ma restituirà una polisuperficie e non più una singola superficie.

 **Cubo**

Crea una primitiva T-spline che può essere un cubo o un parallelepipedo, in funzione del numero di punti di controllo nelle tre direzioni coordinate del poliedro. Queste primitive sono la base di un metodo di costruzione che prevede il loro utilizzo come punto di partenza di un box modeling: partire da esse e avvicinarsi progressivamente in modo approssimativo alla forma voluta dell'oggetto finale. Si noti tuttavia che in questo modo non viene utilizzata e di fatto non viene nemmeno resa visibile tutta la matematica che vi sta dietro, la cui conoscenza da parte di un utilizzatore professionista del programma come un disegnatore non è di solito richiesta né presente.

 **Forma tubolare**

Genera una superficie tubolare a partire da una rete di curve o linee, creando una giuntura morbida in corrispondenza di ciascuna intersezione tra le curve.

 **Aggiungi faccia**

Crea una faccia T-spline a partire da vertici specificati. Di default, la faccia può essere soltanto aggiunta su di un oggetto T-spline esistente (per esempio una primitiva cubo o parallelepipedo) e non creata a sé, nello spirito del

metodo di costruzione cui sopra, ma è possibile disattivare questa limitazione e crearne di nuove in ogni luogo dello spazio di lavoro.

 **T-spline da linee**

Crea una superficie T-spline a partire da una rete di linee; particolarmente utile per generare primitive custom più prossime alla forma desiderata rispetto a quelle predefinite, permettendo di raggiungere la forma voluta prima.

 **Estrai poligono di controllo**

Col tasto destro, si compie l'operazione inversa: da un oggetto T-spline viene estratto il poligono di controllo, che può essere modificato al fine di creare un nuovo oggetto T-spline con il comando precedente.

 **Loft della superficie T-spline**

Crea una superficie T-spline che può contenere una quantità variabile di dettagli, sfruttando la non necessarietà della topologia rettangolare: i punti di controllo vengono mantenuti lontani dalle zone in cui non sono necessari, diminuendo il loro numero totale a parità di dettaglio e facilitando l'editing. Il loft chiede in input delle curve, in base alle quali verrà generata la superficie, con i punti determinanti da queste. Viene offerta la possibilità di far passare la superficie esattamente per le curve, ma anche solo di approssimarla, se questo comporta l'utilizzo di un numero minore di punti di controllo. In generale imponendo che ciascuna fila di punti di controllo possieda lo stesso numero di punti delle curve in input, rispettivamente, il passaggio della superficie attraverso le curve non viene garantito, ma si può ottenere una buona approssimazione; per imporre il passaggio, bisogna aumentarne il numero. Una soluzione intermedia che contempli un compromesso tra numero di punti di controllo e precisione nell'approssimazione è proposta dal software a fianco delle altre due.

 **Adatta T-spline a curve**

Adatta una superficie T-spline ad una rete di curve non rettangolare, utilizzando le intersezioni come punti di controllo. È possibile migliorare l'adattamento aumentando il numero di punti di controllo, inserendone altri sulle curve; inoltre viene permesso di parametrizzare secondo la lunghezza della corda, particolarmente utile quando si hanno segmenti di curva lunghi sul lato opposto di una faccia composta da segmenti di curva brevi.

Commuta morbido

Fa passare la superficie T-spline dalla modalità *morbida* (una T-spline vera e propria) alla modalità *sfaccettata* (come mesh), permettendo di lavorare alternativamente nei due modi.

Estrudi facce, bordi o curve

Nel linguaggio industriale, l'*estrusione* è un processo di produzione di deformazione plastica che consente di produrre pezzi a sezione costante. In questo contesto, si intende come estrusione l'aggiunta di dettagli particolari. In particolare, l'estrusione dei bordi consente l'aggiunta di geometrie sui contorni del modello, l'estrusione di una curva la creazione di superfici T-spline a partire da un singolo tracciato, l'estrusione di facce l'aggiunta di una colonna di nuove facce intorno alla faccia originale e collegandola alla superficie.

Simmetria on

Applica una simmetria assiale o radiale ad una superficie T-spline, permettendo di rilevare una simmetria esistente su un modello completo oppure applicare una nuova simmetria a un modello parziale per completarlo.

Simmetria off

Il comando indotto dal tasto destro del mouse ha l'effetto contrario di rimuovere le costrizioni di simmetria esistenti, con la possibilità di disattivarle soltanto su una parte della superficie.

Assegna spessore

Prende una superficie assegnandole uno spessore duplicandola, collegando poi le due superfici risultanti lungo i bordi.

Piega

Inserisce delle pieghe nette sui bordi T-spline, ovvero fa in modo che in loro corrispondenza la continuità si trovi ad essere C^0. Si veda anche § 3.1.

Rimuovi piega

Col tasto destro, permette di rimuovere le pieghe precedentemente inserite.

Inserisci punto (semplice)

Consente l'inserimento di punti nei bordi, nonché l'estensione dei punti di controllo esistenti. Inserimento *semplice* significa che non viene modificata la posizione di alcun altro punto, ma ciò in generale comporta un cambiamento, anche se plausibilmente locale e limitato, della forma della superficie (cfr. § 3.1).

Inserisci punto (esatto)

Anche questo comando permette l'inserimento di punti nei bordi, e l'estensione dei punti di controllo esistenti, ma in maniera *esatta*, ovvero con la garanzia che la forma della superficie non subisca alcuna modifica, ma con la possibilità di dover aggiungere un numero maggiore di punti di controllo rispetto a quelli desiderati, o di doverne spostare alcuni di già esistenti (si veda ancora § 3.1). Inoltre, la superficie viene standardizzata (§ 3.1.1), dal momento che l'inserimento di punti mantenendo intatta la forma della superficie può essere compiuto soltanto su una superficie standard.

Inserisci bordo

Permette di inserire nuovi bordi; anche in questo caso, l'inserimento può essere semplice oppure esatto, e valgono tutte le considerazioni effettuate a proposito dell'inserimento di punti.

Smussa bordi

Sostituisce i bordi selezionati con un *canale* di facce, ovvero crea delle nuove facce per rendere la superficie più smussata in corrispondenza della selezione.

Scorrimento bordi

Fa scorrere i bordi selezionati lungo i bordi adiacenti; consiste di fatto in uno spostamento di punti di controllo, che viene compiuto da Rhino mentre guida l'utente con un'interfaccia intuitiva.

Aggiungi faccia

Crea una faccia T-spline a partire da vertici specificati. Di default, la faccia può essere soltanto aggiunta su di un oggetto T-spline esistente (per esempio una primitiva cubo o parallelepipedo) e non creata a sé, nello spirito del metodo di costruzione, ma è possibile disattivare questa limitazione e crearne di nuove in ogni luogo dello spazio di lavoro.

Riempi foro

Chiude i fori presenti in una superficie T-spline, lasciando il modello editabile come singola superficie. A differenza delle `NURBS` che richiedono nel caso superfici multiple o polisuperfici, con le T-spline è possibile realizzare un oggetto contenente fori attraverso la costruzione di una singola superficie, eventualmente utilizzando punti straordinari.

Suddividi faccia

Suddivide una faccia in quattro facce; è il modo più semplice offerto da Rhino per aggiungere dettagli ad un modello T-spline. Anche in questo caso la suddivisione può essere svolta in maniera semplice o esatta, con le considerazioni valide nel caso di inserimento di punti o di bordi.

Facce identiche

Crea una copia di facce T-spline, copiando le posizioni dei punti di controllo. Si noti che questo implica che, nel caso di copia di una selezione aperta, i bordi di contorno della selezione potranno trovarsi ad avere una curvatura diversa rispetto alla superficie originale, in quanto le condizioni ai margini possono essere assenti o differenti.

Cancella

Elimina gli oggetti selezionati.

Salda punti

Consente la saldatura di punti all'interno di una singola T-spline, oppure combinando due superfici T-spline in una.

Fondi bordi; annulla saldatura

Permette la fusione di bordi all'interno di una singola T-spline, oppure la combinazione di due T-spline in una; la saldatura può essere annullata con il comando inverso richiamabile dal tasto destro.

Collega

Realizza un collegamento tra due superfici T-spline, oppure due parti della stessa superficie, inserendo delle facce intermedie di unione. Il comando è applicabile alle facce oppure ai bordi di contorno; selezionando due gruppi di facce, viene generato un collegamento tra i contorni delle regioni selezionate, con conseguente rimozione delle facce.

Combina superfici

Posiziona il bordo di contorno di una superficie T-spline rispetto ad una superficie `NURBS` o curva di Rhino, con continuità di posizione (G^0), tangenza (G^1), o curvatura (G^2). Si tratta di un'applicazione della procedura descritta nel § 5.1, anche se si permette di utilizzare già una T-spline come elemento da combinare (nel § 5.1 abbiamo visto la fusione di due B-spline), e ci si limita a richiedere la più debole continuità geometrica rispetto ad una più forte continuità parametrica.

Riporta punti di controllo

Prende tutti i vertici di una T-spline e li riporta sui punti più vicini della superficie di destinazione. Il comando può essere utilizzato per riportare tutti i punti della superficie T-spline, oppure tutti i vertici del suo poligono di controllo.

Set di selezione

Consente di salvare gruppi di selezione di vertici, bordi e facce; si tratta di un comando puramente tecnico, che serve per facilitare il lavoro all'utilizzatore del software.

Interpola superfice su punti di controllo

Sposta i punti di controllo della T-spline in modo tale che la nuova superficie passi per le posizioni precedenti dei suoi punti di controllo. Particolarmente

utile per imporre il passaggio di una superficie per alcuni punti, senza utilizzare procedure complicate, ma solamente costruendola dando i punti di controllo come punti di passaggio, e poi imponendone l'interpolazione.

 **Impostazioni snap Ritopo**

Permette di modificare alcune impostazioni relative al programma; anche questo comando è puramente tecnico.

 **Spiana punti**

Spiana un insieme di almeno quattro punti di controllo non complanari su di un singolo piano, che si troverà ad essere passante per il gruppo dei punti. Il piano viene individuato in maniera tale che sia passante per il baricentro dei punti e individuato da due direzioni determinate dalle posizioni relative dei punti. Sostanzialmente si tratta di spostare i punti di controllo affinché sia soddisfatta tale modifica.

 **Assegna peso ai punti T-spline**

Assegna un peso ai punti di controllo di una T-spline. Si tratta di uno dei comandi più legati all'aspetto matematico delle superfici T-spline, andando esso ad inerire direttamente la definizione stessa di tale strumento.

 **Modifica UV**

Consente di modificare il layout UV delle texture applicate alle T-spline ed alle mesh di rendering T-spline. Molto utile per i disegnatori che vogliono dare una rappresentazione più completa all'oggetto su cui stanno lavorando, ma poco legato all'aspetto matematico dello strumento.

 **Punti interattivi della mesh texture attivati**

Si tratta di un complemento al comando precedente, che serve ad attivare punti interattivi su mesh texture.

 **Rendi uniforme**

Rende uniformi gli intervalli tra nodi (§ 1.3) della superficie T-spline. Ciò risulta particolarmente utile se, dopo l'aggiunta di nuovi punti di controllo, la superficie si trova a presentare delle escrescenze di troppo laddove i punti sono raggruppati in maniera più densa.

Modifica layout

Consente di risolvere gli errori che impediscono al modello di venire visualizzato come una T-spline morbida. Questo comando consente anche di ottimizzare la smussatura e l'andamento delle isocurve del modello cambiando le T-giunzioni in punti straordinari e viceversa. È sempre possibile trasformare una T-giunzione in un punto straordinario, ma il viceversa è invece soltanto realizzabile se sono presenti le seguenti due condizioni:

- la valenza del vertice è minore di 4;
- la faccia risultante verso cui punterà la T-giunzione è un quadrilatero.

In alcune situazioni, si richiede il passaggio da una T-giunzione ad un punto straordinario per poter visualizzare la superficie in modalità morbida. A tale scopo, questo comando mette a disposizione un'opzione, *AutoRipara*, che trasforma automaticamente tutte le T-giunzioni nei punti straordinari necessari per la visualizzazione della superficie morbida.
Una situazione sgradevole che può presentarsi è quella delle cosiddette *stelle erronee*. Un tale oggetto è un punto che si trova ad essere allo stesso tempo T-giunzione e punto straordinario; il termine prende ragione dal fatto che i punti straordinari vengono anche chiamati *punti stella*. Esse si possono generare in vari modi, per esempio quando si estrude una faccia con una T-giunzione sul vertice, operazione che lo fa diventare anche un punto straordinario.
Le stelle erronee di solito danno luogo a delle superfici di bassa qualità, e questo ne rende preferibile l'assenza. Si possono risolvere estendendo la T-giunzione, ciò può essere effettuato aggiungendo manualmente un punto, standardizzando, o autoriparando con l'opzione di cui abbiamo precedentemente parlato.

Standardizza

Visualizza i punti aggiunti in caso di standardizzazione, di default nascosti. Si noti che la standardizzazione viene sempre eseguita, perlomeno quando la T-spline si trova in modalità morbida, al fine di mantenere la compatibilità con le superfici `NURBS`.

Inverti normali

Inverte la direzione delle normali di una T-spline.

Definisci regioni per la conversione T-spline in NURBS

Consente all'utente di definire le regioni in cui la superficie T-spline verrà suddivisa in patch NURBS. Di default, quando le superfici T-spline vengono convertite in NURBS, i bordi fuoriuscenti dagli eventuali punti straordinari vengono estesi per formare il contorno delle patch NURBS, e tutti i punti T all'interno di queste zone vengono estesi per andare a formare delle NURBS rettangolari. Questo comando offre all'utente l'opportunità di ridefinire in che modo la T-spline viene suddivisa in NURBS.

Seleziona bordi per GraficoCurvatura

Consente all'utente di attivare determinati bordi T-spline mentre è attivato il grafico di curvatura (vedi § 6.1.2).

Converti T-spline in mesh

Crea una mesh poligonale a partire dalla T-spline specificata in input, permettendo di convertire rapidamente una superficie T-spline nella nota controparte tipologica.

Estrai bordi

Estrae i bordi selezionati dalla superficie T-spline, similmente a quanto sia possibile estrarre ordinariamente le isocurve dalle NURBS.

Suddividi curve

Consente di suddividere una rete di curve che si intersecano, in vari segmenti di curva.

Estrudi linee

Estrude un poligono di controllo bidimensionale in un poligono di controllo tridimensionale, che può essere utilizzato per creare una superficie T-spline.

Opzioni

Apre una finestra dalla quale visionare e modificare le opzioni globali.

 **Informazioni su T-Splines**

Apre una finestra di informazioni varie sull'add-on.

 **SferaQuad**

Crea una *SferaQuad*, una forma sferica con una topologia cubica; utile come alternativa al cubo o al parallelepipedo, in termini di primitiva.

 **Successivo/Precedente** U

Seleziona l'elemento successivo (tasto sinistro del mouse) o precedente (tasto destro del mouse) nella direzione parametrica u, sia esso un vertice, un bordo, o una faccia.

 **Successivo/Precedente** V

Seleziona l'elemento successivo (tasto sinistro del mouse) o precedente (tasto destro del mouse) nella direzione parametrica v, sia esso un vertice, un bordo, o una faccia.

 **Incrementa/ritira anello**

 **Incrementa/ritira ciclo**

Permettono di aumentare o diminuire la selezione coerentemente con la topologia locale degli oggetti che si stanno selezionando.

Bibliografia e sitografia

[1] Sederberg T. W., Zheng J., Bakenov A., Nasri A., *T-splines and T-NURCCs*. ACM Transactions on Graphics 22, 3 (July), pp. 477-484, 2003.

[2] Sederberg T. W., Cardon D.L., Finnigan G.T., North N.S., Zheng J., Lyche T., *T-spline Simplification and Local Refinement*. ACM Transactions on Graphics 23(3), pp. 276-283, 2004.

[3] Forsey D., Bartels R.H., 1988, *Hierarchical B-spline refinement*. Computer Graphics 22, 4 (agosto 1988), 205-212.

[4] Kraft R., 1998, *Adaptive and linearly independent multilevel B-splines*. Surface Fitting and Multiresolution Methods (Mehaute A.L., Rabut C., Schumaker L.L.), Eds. Vanderbilt University Press, Nashville, 209-218.

[5] Grinspun E., Krysl P., Schroder P., 2002, *CHARMS: A simple framework for adaptive simulation*. ACM Transactions on Graphics 21, 3 (luglio 2002), 281-290.

[6] Weller F., Hagen H, 1995, *Tensor product spline spaces with knot segments*. Mathematical Methods for Curves and Surfaces (Daehlen M., Lyche T., Schumaker L.L.), Eds. Vanderbilt University Press, Nashville, 563-572.

[7] Rockwood A.P., Heaton K., Davis T., 1989, *Real-time rendering of trimmed surfaces*. Proceedings of SIGGRAPH 89, 107-116.

[8] Sederberg T.W., Zheng J., Sewell D., Sabin M., 1998, *Non-uniform recursive subdivision surfaces*. Proceedings of SIGGRAPH 98 (luglio), 387-394, `ISBN 0-89791-999-8`.

[9] Bakenov A., 2001, *T-Splines: Tensor Product B-spline Surfaces with T-Junctions*. Master's thesis, Brigham Young University.

[10] Lyche T., 1993, *Knot removal for spline curves and surfaces*. Approximation Theory VII, Academic Press (Cheney E.W., Chui C.K., Schumaker L.L.), Eds., 207-227.

[11] Daehlen M., Lyche T., 1992, *Decomposition of splines*. Mathematical Methdos in Computer Aided Geometric Design II (Lyche T., Schumaker L.L.), Academic Press, New York, Eds., 135-160.

[12] `http://it.wikipedia.org/wiki/Watermarking`

[13] Bin W., Ri-Jing P., et al. *A Fragile Watermarking Algorithm for T-Spline Surfaces*. Information, Computing and Telecommunication, 2009. YC-ICT '09. IEEE Youth Conference on, pp. 546-549, 2009.

[14] Bin W., Ri-Jing P., et al. *Watermarking T-Spline Surfaces*. Communication Technology, 2008. ICCT 2008. 11th IEEE International Conference on, pp. 773-776, 2008.

[15] Bin W., Ri-Jing P., et. al. *Robust Watermarking for T-spline Surfaces Based on Nonuniform B-spline Wavelets Transformation*. Information Computing and Telecommunications (YC-ICT), 2010 IEEE Youth Conference on, pp. 335-338, 2010.

[16] Rhinoceros: `NURBS` modeling for Windows Versione 4.0: guida all'uso.

[17] Wang Y., Zheng J., Seah H.S., *Conversion between T-splines and hierarchical B-splines*. Proceedings of the Eighth IASTED International Conference, Computer Graphics and Imaging, 2005.

[18] Buffa A., Cho D., Sangalli G., *Linear independence of the T-spline blending functions associated with some particular T-meshes*. Comput. Methods Appl. Mech. Engrg. 199, pp. 1437–1445, 2010.

[19] Beirao da Veiga L., Buffa A., Cho D., Sangalli G., *IsoGeometric analysis using T-splines on two-patch geometries*. Comput. Methods Appl. Mech. Engrg. 200, pp. 1787-1803, 2011.

[20] `http://www.tsplines.com/forum/viewtopic.php?t=34477`

[21] `http://www.it.rhino3d.com/4/help/FileIO/file_formats.htm#x_t`

[22] `http://www.tsplines.com/forum/viewtopic.php?t=91`

[23] `http://www.cad3d.it/forum1/showthread.php?t=31002`

[24] Li X., Zheng J., Sederberg T.W., Hughes T.J.R, Scott M.A., *On linear independence of T-spline blending functions*. Computer Aided Geometric Design, Volume 29 Issue 1, January, 2012.

[25] Dörfel M., Jüttler B., Simeon B., *Adaptive isogeometric analysis by local h-refinement with T-splines*. Comput. Methods Appl. Mech. Engrg. 199 (5-8), pp. 264-275, 2010.