UNIVERSITÀ DEGLI STUDI DI PADOVA

—

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE

—

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA MECCANICA

TITOLO TESI

RELATORE: CH.MO PROF. ING. MATTEO MASSARO
CORRELATORE: CH.MO PROF. ING. GIULIO ROSATI
CORRELATORE: CH.MO PROF. ING. ALBERTO DORIA

CONTRORELATORE: CH.MO ING. MATTEO BOTTIN

LAUREANDO: AUTORE
MATRICOLA: 000000

ANNO ACCADEMICO 2022-2023

ai miei genitori...

“ Ora ti dirò tutta la mia idea intorno alla Casa dello Specchio, bla bla ”

Lewis Carroll - Alice attraverso lo specchio, 1871

Indice

Sommario

Un display aptico è un dispositivo meccanico in grado di rappresentare all'operatore umano un'impedenza meccanica variabile: è perciò adatto ad operazioni complesse come la simulazione, o la telemanipolazione, soprattutto nei casi in cui il solo feedback visivo non sia sufficiente per una corretta esecuzione dei compiti, come ad esempio il contesto medico-chirurgico.

Presso i laboratori del DIMEG è stata costruita un display aptico a cinque gradi di libertà, denominata PIROGA5, costituita da un end-effector a forma di penna sorretto da sei fili, collegati ad altrettanti motori.

per far vedere un esempio di nota

In questo lavoro di tesi sono stati sviluppati diversi programmivolti all'utilizzo di PIROGA5 in un ambiente virtuale; quest'ultimo viene generato tramite funzioni NURBS, il modo più generale di rappresentare forme complesse nello spazio.

Introduzione

La tesi che viene descritta nelle pagine che seguono si inserisce nel contesto del progetto **RIME** (Robot in Medical Environment), che ha come obiettivo finale la costruzione e l'effettuazione di una serie di test su un prototipo per interventi di chirurgia spinale, in telechirurgia.

In particolare, il prototipo costruito nei laboratori del Dipartimento di Innovazione Meccanica e Gestionale (DIMEG) di Padova consiste nella parte *master* di un sistema aptico (il cui *slave* è in lavorazione a Vicenza), costituito da un end-effector collegato da dei fili a un sistema di motori ed encoder, denominato Piroga5.

La tesi è consistita nello sviluppo di alcuni programmi che permettono di utilizzare il master all'interno di un ambiente virtuale, ricevendone forze consistenti con spostamenti e collisioni con oggetti virtuali.

Il capitolo 1 è introduttivo al lavoro svolto e consiste in tre parti, abbastanza distinte per argomenti trattati: nella prima si illustrano alcune tematiche relative alla realtà virtuale, con dei riferimenti alla storia del pensiero umano, nella seconda parte è analizzato il ruolo di un'interfaccia e viene chiarito cosa siano le interfacce aptiche, mentre nella terza viene descritto il loro utilizzo nella medicina moderna, in particolare nei sistemi **CIS** (Computer Integrated Surgery).

Poichè i programmi sviluppati non sono a sè stanti, ma vengono inseriti in un insieme di moduli software e hardware di differenti tipi e funzioni, si illustra nel capitolo 2 il funzionamento complessivo dell'intero sistema, per permettere al lettore interessato di contestualizzare meglio gli argomenti trattati in questa

tesi. Vengono descritte, in particolare, modalità e tipologie di connessione dei vari programmi.

Come appare dal titolo, l'ambiente virtuale viene creato ricorrendo a superfici NURBS, che sono il modo più generale di descrizione di forme nello spazio. Per capire come funzionano e in cosa codeste NURBS si differenzino da altri metodi analoghi, si trattano tutti i principali tipi di rappresentazione parametrica di curve e superfici, in ordine di complessità crescente (che corrisponde all'ordine storico): curve di Bézier, curve B-Spline, curve e superfici NURBS. Oltre ad una descrizione matematica abbastanza completa, viene spiegato come le superfici siano state effettivamente implementate e gestite all'interno dei programmi sviluppati, con riferimento ai formati dei file usati. Infine si parla del programma ShowNURBS, sviluppato in ambiente MatLAB, finalizzato, come dice il nome, alla visualizzazione e gestione veloce di superfici NURBS.

Capitolo 1

Nurbs

àèìòù

In questo capitolo verranno descritti alcuni dei metodi più utilizzati per definire curve e superfici nello spazio, fino ad arrivare, in particolare, alle superfici NURBS. Sarà poi illustrato come quest'ultime, all'interno dei programmi sviluppati per questo lavoro di tesi, vengano generate, gestite, ed esportate da un programma all'altro.

1.1 Curve e superfici parametriche

La fonte di questo argomento è [?], mentre ulteriori approfondimenti si trovano in [?, ?, ?, ?, ?, ?, ?, ?, ?].

1.1.1 Forma implicita e forme parametriche

I due metodi più comuni per descrivere curve e superfici nello spazio sono la forma implicita e le forme parametriche.
Nel primo caso si usano delle equazioni a 1, 2 o 3 incognite il cui insieme di soluzioni fornisce le coordinate della figura da rappresentare. Ad esempio, nel piano XY, il cerchio di raggio unitario e centrato nell'origine degli assi è specificato dall'equazione $f(x, y) = x^2 + y^2 - 1 = 0$.
Invece, nelle forme parametriche, tutte le coordinate di ogni punto della curva sono esplicitamente definite da una funzione, legata ad un parametro indipendente;

in generale, quindi, una curva in uno spazio a 3 dimensioni è espressa da:

$$\mathbf{C}(u) = (x(u), y(u), z(u)) \qquad a \leq u \leq b \tag{1.1}$$

Tornando all'esempio del cerchio di raggio unitario, esso può essere descritto dalle seguenti funzioni parametriche

$$\begin{cases} x(u) = \cos(u) \\ y(u) = \sin(u) \end{cases} \qquad 0 \leq u \leq 2\pi \tag{1.2}$$

oppure da queste:

$$\begin{cases} x(u) = \frac{1-t^2}{1+t^2} \\ y(u) = \frac{2t}{1+t^2} \\ y(u) = \frac{2t}{1+t^2} \\ y(u) = \frac{2t}{1+t^2} \end{cases} \qquad 0 \leq t \leq 1$$

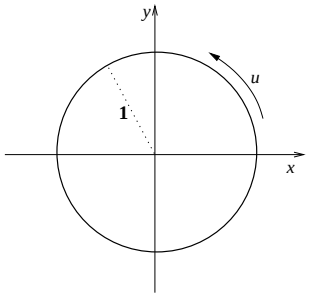

Figura 1.1: Il cerchio di raggio unitario, disegnato ricorrendo alla forma parametrica della funzione 1.2

Le rappresentazioni parametriche possono venir interpretate come il moto di una particella: la variabile indipendente u, che rappresenta il tempo, definisce il verso principale di attraversamento.

Come si è appena visto, esistono più rappresentazioni parametriche possibili. Tra la forma implicita e quella parametrica è difficile dire quale sia la migliore, dato che ognuna ha i suoi vantaggi e svantaggi, a seconda delle applicazioni. In generale, tuttavia, la forma esplicita è più facile da usare: ad esempio, per definire curve in uno spazio a tre dimensioni, la forma implicita richiede di calcolare l'intersezione di due superfici, mentre nel caso parametrico è banale passare a spazi a n-dimensioni, aggiungendo n coordinate. Nel corso di questo lavoro ci si concentrerà esclusivamente sulle rappresentazioni parametriche.

1.1.2 Curve di Bézier

Queste curve sono delle forme parametriche polinomiali: una curva di Bézier di grado n è così definita:

$$\mathbf{C}(u) = \sum_{i=0}^{n} B_{i,n}(u)\mathbf{P}_i \qquad 0 \leq u \leq 1 \tag{1.3}$$

dove le funzioni base (*blending functions*) $\{B_{i,n}(u)\}$ sono i cosiddetti polinomi di Bernstein di grado n, dati da

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}u^i(1-u)^{n-i} \tag{1.4}$$

mentre i coefficienti geometrici $\{\mathbf{P}_i\}$ sono detti *punti di controllo*.

A differenza delle curve costituite da generali funzioni polinomiali, le curve di Bézier hanno, nei parametri che le definiscono, dei significati geometrici. Infatti il poligono formato dai punti di controllo $\{\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_n\}$ approssima la forma della curva abbastanza bene (si può dimostrare che la curva è contenuta interamente nel poligono convesso formato dai punti di controllo).
Per quanto riguarda poi le funzioni base, è utile specificare subito alcune delle loro proprietà, che verranno usate anche nei successivi tipi di rappresentazione:

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

2. Partizione dell'unità: $\sum_{i=0}^{n} B_{i,n}(u) = 1$ per ogni $0 \leq u \leq 1$;

3. Valori agli estremi: $B_{0,n}(0) = B_{n,n}(1) = 1$;

4. $B_{i,n}(u)$ ha solo un massimo, posto a $u = i/n$, nell'intervallo [0,1];

5. Definizione ricorsiva: $B_{i,n}(u) = (1-u)B_{i,n-1}(u) + uB_{i-1,n-1}(u)$; si pone allora: $B_{i,n}(u) = 0$ se $i < 0$ o $i > n$;

In sostanza le funzioni base assegnano dei pesi positivi (propr. 1), la cui somma dà l'elemento unitario (propr. 2), ad ogni punto di controllo. La proprietà 5 è importante perché suggerisce un algoritmo ricorsivo per calcolare, dato un valore di u, i valori delle funzioni base, e permettere così la rappresentazione della curva.

Curve di Bézier razionali

Nonostante le curve di Bézier offrano tanti vantaggi, tra cui la facilità di utilizzo nel disegno di forme semplici, tuttavia ci sono alcune figure che non possono essere rappresentate perfettamente in forma polinomiale, come cerchi, ellissi, iperboli, cilindri coni, sfere, etc. In geometria queste figure sono dette *coniche* e vengono descritte con *funzioni polinomiali razionali*, date cioè dal rapporto di due polinomi:

$$[x(u), y(u), z(u)] = \left(\frac{X(u)}{W(u)}, \frac{Y(u)}{W(u)}, \frac{Z(u)}{W(u)}\right)$$

con $X(u), Y(u), Z(u), W(u)$ polinomi (si noti come ogni funzione parametrica abbia il medesimo denominatore $W(u)$).
Vengono quindi definite le curve razionali di Bézier di n-esimo grado come:

$$\mathbf{C}(u) = \frac{\sum_{i=0}^{n} B_{i,n}(u) w_i \mathbf{P}_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i} \qquad 0 \leq u \leq 1 \tag{1.5}$$

I punti di controllo e le funzioni base sono le stesse viste in (1.3), mentre i w_i sono dei valori scalari, detti *pesi*, e sono sempre positivi. In conseguenza di ciò, si può anche scrivere:

$$\mathbf{C}(u) = \sum_{i=0}^{n} R_{i,n}(u) \mathbf{P}_i \qquad 0 \leq u \leq 1 \tag{1.6}$$

dove

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

Le $R_{i,n}(u)$ sono dette *funzioni base razionali* della curva e mantengono le proprietà delle funzioni base viste prima. Inoltre, dato che se $w_i = 1$ per ogni i, si ha $R_{i,n}(u) = B_{i,n}(u)$, abbiamo che le funzioni $B_{i,n}(u)$ sono un caso particolare delle $R_{i,n}(u)$, e, più in generale, le curve polinomiali di Bézier sono un caso particolare delle curve razionali di Bézier.

1.1.3 Curve B-Spline

Uno dei maggiori svantaggi delle curve di Bézier appena trattate, è dato dal fatto che il grado della funzione cresce con il numero dei punti di controllo (a causa delle

proprietà dei coefficienti $B_{i,n}(u)$, come si vede dall'espressione 1.4): in pratica, per avere una curva che passi attraverso n punti, è necessario un polinomio di grado $(n-1)$. Ciò comporta inefficienza negli algoritmi e risultati numericamente instabili. La soluzione a questo è rappresentata da somme di polinomi o somme di polinomi razionali.

L'intervallo di variabilità del parametro $u \in [0,1]$ viene suddiviso in sottointervalli tramite dei punti di spezzamento (*breakpoints*), tali da avere:

$$u_0 = 0 < u_1 < u_2 < \ldots < u_n = 1$$

Ognuno di questi sottointervalli diventa il dominio del polinomio corrispondente, che definisce quindi un segmento dell'intera curva. Tutti i segmenti sono costruiti in maniera tale da unirsi nei punti di spezzamento con un certo livello di continuità. I vari segmenti che formano $\mathbf{C}$(u) vengono indicati con un pedice $C_i(u)$. Tutte le forme polinomiali viste precedentemente possono essere usate per definire $C_i(u)$; ad esempio, prendendo curve di Bézier di grado 3, e suddividendo in quattro intervalli il dominio di u, ottenenendo $U = \{u_0, u_1, u_2, u_3\}$, ricaviamo una curva con ben 12 punti di controllo (quattro per ogni intervallo). Se però imponiamo la continuità della stessa, dobbiamo porre $\mathbf{P}_1^3 = \mathbf{P}_2^0$ e allo stesso modo $\mathbf{P}_2^3 = \mathbf{P}_3^0$. Imponendo la continuità delle derivate prime aumentano le condizioni sui punti di controllo. Vanno perciò cercate delle funzioni base opportune, che dipendano anche dal livello di continuità richiesto, e cosa analoga anche per l'insieme di punti di spezzamento.

Dopo varie considerazioni si arriva alle seguenti definizioni; preso il vettore $U = \{u_0, \ldots, u_m\}$, contenente una sequenza non decrescente di numeri reali, essi vengono chiamati *nodi* e U *vettore dei nodi*. La i-esima funzione base B-spline di grado p (e ordine $p+1$), espressa come $N_{i,p}(u)$, è definita nel seguente modo ricorsivo:

$$\begin{aligned} N_{i,0}(u) &= \begin{cases} 1 & \text{se } u_i \leq u \leq u_{i+1} \\ 0 & \text{altrimenti} \end{cases} \\ N_{i,p}(u) &= \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \end{aligned} \tag{1.7}$$

Con queste funzioni base possiamo finalmente arrivare alla definizione della

curva B-spline di grado p:

$$\mathbf{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P}_i \quad a \leq u \leq b \tag{1.8}$$

dove $\{\mathbf{P}_i\}$ sono i punti di controllo, e $\{N_{i,p}(u)\}$ sono le funzioni base B-spline di grado p, viste in 1.7, definite sul vettore dei nodi ($m+1$ nodi):

$$U = \{\underbrace{a, \ldots, a}_{p+1}, u_{p+1}, \ldots, u_{m-p-1}, \underbrace{b, \ldots, b}_{p+1}\}$$

Si ha che il grado p, il numero di punti di controllo, $n+1$ e il numero dei nodi $m+1$ sono così legati:

$$m = n + p + 1 \tag{1.9}$$

Poiché costituite da funzioni polinomiali, è relativamente facile calcolare le derivate, di qualsiasi ordine

$$\mathbf{C}^{(k)}(u) = \sum_{i=0}^{n} N_{i,p}^{(k)}(u)\mathbf{P}_i \quad a \leq u \leq b \tag{1.10}$$

1.1.4 Curve NURBS

Una curva NURBS[1], di grado p, è definita nel modo seguente

$$\mathbf{C}(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u)w_i\mathbf{P}_i}{\sum_{i=0}^{n} N_{i,p}(u)w_i} \quad a \leq u \leq b \tag{1.11}$$

dove $\{\mathbf{P}_i\}$ sono i punti di controllo, che formano il poligono di controllo, w_i sono i pesi, e $\{N_{i,p}(u)\}$ sono le funzioni base B-spline di grado p, definite sul vettore dei nodi ($m+1$ nodi):

$$U = \{\underbrace{a, \ldots, a}_{p+1}, u_{p+1}, \ldots, u_{m-p-1}, \underbrace{b, \ldots, b}_{p+1}\}$$

Ponendo

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

[1] **N**on-**U**niform **R**ational **B**-**S**pline

possiamo riscrivere l'equazione (1.11) come

$$\mathbf{C}(u) = \sum_{i=0}^{n} R_{i,p}(u)\mathbf{P}_i \quad a \leq u \leq b \tag{1.12}$$

Le $\{R_{i,p}(u)\}$ sono le *funzioni base razionali* ed hanno proprietà derivate da quelle viste nei casi precedenti:

1. Non negatività: $R_{i,p}(u) \geq 0$ per ogni i, p e $u \in [a, b]$;
2. Partizione dell'unità: $\sum_{i=0}^{n} R_{i,p}(u) = 1$ per ogni $u \in [a, b]$;
3. Valori agli estremi: $R_{0,p}(0) = R_{n,p}(1) = 1$;
4. Supporto locale limitato: $R_{i,p}(u) = 0$ per $u \notin [u_i, u_{i+p+1})$: ciò implica che tra due nodi qualsiasi, al più $p + 1$ funzioni base razionali $R_{i,p}(u)$ siano diverse da zero;
5. In un nodo, $R_{i,p}(u)$ è derivabile $p-k$ volte, dove k è la moltiplicità del nodo stesso (tra un nodo e l'altro esistono tutte le derivate);
6. Se $w_i = 1\ \forall i$, si ha $N_{i,p}(u) = R_{i,p}(u)$.

Dalla proprietà 6, abbiamo che le funzioni $R_{i,n}(u)$ sono un caso particolare delle $N_{i,n}(u)$, e più in generale, le curve B-spline sono un caso particolare delle curve NURBS. Inoltre, una curva NURBS con nessun nodo interno degenera in una curva razionale di Bézier. Dalla proprietà 3, abbiamo invece che $\mathbf{C}$(a)=$\mathbf{P}_0$ e $\mathbf{C}$(b)=$\mathbf{P}_n$; nella figura si può vedere come la curva si avvicini o allontani dagli altri punti di controllo a seconda della disposizione dei nodi.

1.1.5 Superfici e volumi NURBS

è relativamente facile generalizzare a più parametri quanto visto finora, per ottenere superfici e volumi Nurbs. Abbiamo visto che una curva $\mathbf{C}$(u) è una funzione vettoriale basata sul parametro u, ed è quindi una mappa $\Re \to \Re^3$. Una superficie è una funzione vettoriale di due parametri, u, v, e quindi è una trasformazione $\Re^2 \to \Re^3$, da cui $\mathbf{S}(u, v)) = (x(u, v), y(u, v), z(u, v))$. Il metodo più semplice e più usato per ottenere delle funzioni bivarianti è usare il prodotto tensiorale, ossia

costruire le funzioni base bivarianti come prodotto di funzioni base monovarianti. In questo modo, ad esempio, una superficie di Bézier (cfr. 1.3, pag. 3) diventa:

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,p}(u)B_{j,q}(v)\mathbf{P}_{i,j}$$

Utilizzando le coordinate uniformi (Appendice, pag. 23) possiamo descrivere una *superficie razionale di Bézier* come:

$$\mathbf{S}^w(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,p}(u)B_{j,q}(v)\mathbf{P}^w_{i,j}$$

Allo stesso modo si può ottenere una superficie B-Spline, e Nurbs: più precisamente, una superficie Nurbs di grado p nel vettore u e di grado q nel vettore v è data da

$$\mathbf{S}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_i} \qquad \begin{matrix} a \le u \le b \\ c \le v \le d \end{matrix} \tag{1.13}$$

In questo caso abbiamo dunque due vettori dei nodi

$$U = \{\underbrace{a,\ldots,a}_{p+1}, u_{p+1},\ldots,u_{r-p-1}, \underbrace{b,\ldots,b}_{p+1}\}$$

$$V = \{\underbrace{c,\ldots,c}_{q+1}, u_{q+1},\ldots,u_{s-q-1}, \underbrace{d,\ldots,d}_{q+1}\}$$

con $r = n+p+1$ e $s = m+q+1$, e l'insieme $\{\mathbf{P}_{i,j}\}$ forma una rete *bidirezionale* di punti di controllo. Anche in questo caso la formula può essere riscritta in un modo alternativo: ponendo

$$R_{i,j}(u,v) = \frac{N_{i,p}(u)N_{j,q}(v)w_{i,j}}{\sum_{k=0}^{n}\sum_{l=0}^{m} N_{k,p}(u)N_{l,q}(v)w_{k,l}}$$

arriviamo alla forma, analoga a (1.12):

$$\mathbf{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} R_{i,j}(u,v)\mathbf{P}_{i,j} \qquad \begin{matrix} a \le u \le b \\ c \le v \le d \end{matrix} \tag{1.14}$$

è conveniente rappresentare una superficie Nurbs utilizzando le coordinate omogenee, ottenendo:

$$\mathbf{S}^w(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\mathbf{P}^w_{i,j}$$

Si può notare come $\mathbf{S}^w(u,v)$ sia un prodotto tensioriale, definito in uno spazio a quattro dimensioni, mentre $\mathbf{S}(u,v)$ non lo sia, non essendo $R_{i,j}(u,v)$ prodotto di funzioni base monovarianti. Tuttavia, al di là di queste differenze di notazione, le proprietà di $R_{i,j}(u,v)$ sono praticamente le stesse delle funzioni $N_{i,j}(u,v)$:

1. Non negatività: $R_{i,p}(u,v) \geq 0 \ \ \forall\, i,p,\ u \in [a,b],\ v \in [c,d]$;
2. Partizione dell'unità: $\sum_{i=0}^{n}\sum_{j=0}^{m} R_{i,j}(u,v) = 1 \ \ \forall\, u \in [a,b],\ v \in [c,d]$;
3. Estremi: $R_{0,0}(a,c) = R_{n,0}(b,c) = R_{0,m}(a,d) = R_{n,m}(b,d) = 1$;
4. Supporto locale limitato: $R_{i,p}(u,v) = 0$ per $u \notin [u_i, u_{i+p+1})$ o $v \notin [v_j, v_{j+q+1})$
5. Se $w_{i,j} = 1\ \forall i,j$, si ha $R_{i,j}(u,v) = N_{i,p}(u)N_{j,q}(v)\ \forall i,j$;

Dalla proprietà 3, abbiamo $\mathbf{S}(a,c) = \mathbf{P}_{0,0}$, $\mathbf{S}(b,c) = \mathbf{P}_{n,0}$, $\mathbf{S}(a,d) = \mathbf{P}_{0,m}$, $\mathbf{S}(b,d) = \mathbf{P}_{n,m}$, mentre dalla 5 abbiamo che le superfici non razionali B-Spline e le superfici razionali di Bézier sono casi particolari delle superfici NURBS.

Quanto detto si può generalizzare al caso di NURBS triparametriche, che potremmo definire *volumi NURBS*, ossia funzioni $\Re^3 \to \Re^3$. Estendendo dunque a tre fattori il prodotto tensoriale, otteniamo la formula seguente, valida per un volume Nurbs di grado p nel parametro u, di grado q nel parametro v e di grado r nel parametro w:

$$\mathbf{V}(u,v,w) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m}\sum_{k=0}^{l} N_{i,p}(u)N_{j,q}(v)N_{k,r}(w)w_{i,j,k}\mathbf{P}_{i,j,k}}{\sum_{i=0}^{n}\sum_{j=0}^{m}\sum_{k=0}^{l} N_{i,p}(u)N_{j,q}(v)N_{k,r}(w)w_{i,j,k}} \tag{1.15}$$

dove $a \leq u \leq b,\ c \leq v \leq d,\ e \leq w \leq f$; ci sono tre vettori dei nodi U, V e W, e l'insieme $\{\mathbf{P}_{i,j,k}\}$ forma una rete *tridirezionale* di punti di controllo. Anche la formula appena vista può essere semplificata nella notazione, come è stato fatto per le curve e le superfici, e ne eredita le proprietà.

1.2 Gestione delle NURBS

Nel corso di questo lavoro di tesi sono state usate Nurbs di terzo grado, perché i miglioramenti nelle rappresentazioni derivanti da un aumento ulteriore di grado

non giustificano il forte incremento della complessità e della lentezza delle procedure di calcolo. Per le stesse ragioni, i pesi sono stati posti pari a 1.
Si sono usate per lo più NURBS biparametriche, mentre quelle triparametriche, le cui definizioni proprietà sono derivabili direttamente da (1.12) e (1.14) sono state usate nelle MetaNurbs, come verrà spiegato in seguito.
Per specificare le coordinate dei punti di controllo si è usato il millimetro come unità di misura, tenendo presente che le dimensioni approssimative dell'area di lavoro sono:

Asse	Minimo	Massimo
x	-160 mm	160 mm
y	-160 mm	160 mm
z	-240 mm	240 mm

Vedremo ora come vengono effettivamente trattate queste particolari entità geometriche.

1.2.1 La classe Nurbs

All'interno dei vari programmi sviluppati, le Nurbs vengono gestite come oggetti, istanze della classe `Nurbs`. Tale classe contiene tutte le variabili e le funzioni collegate (che diventano *funzioni membro*), e permette di gestire superfici con un numero arbitrario di assi e fino a tre parametri (u,v,w). Ciò ha consentito, rispetto al codice sviluppato precedentemente, una più chiara e moderna organizzazione dei programmi, e inoltre una riduzione delle chiamate a puntatore: essendo infatti ogni funzione definita sull'oggetto di cui essa è membro, l'ambito di validità (lo *scope*) delle variabili è già indicato implicitamente.

In Appendice è riportata la dichiarazione della classe, contenuta nell'header `nurbs_context.h`: la definizione delle routine non è tutta nel file corrispondente `nurbs_context.cpp` ma suddivisa in più file (che importano nell'instestazione la stessa dichiarazione), ognuno con un insieme omogeneo di routine (nella dichiarazione delle funzioni della classe, sono indicati, con un commento, i nomi dei file in cui ci sono le corrispondenti definizioni).

Poiché ogni oggetto Nurbs alloca della memoria dinamica, è necessario prestare attenzione quando tale oggetto viene creato o distrutto (ad esempio quando

viene passato come parametro tra una funzione e l'altra). Poiché non si è fatto l'overload degli operatori, nei programmi si ricorre ad un oggetto particolare, una Nurbs "minima", con identificativo ID pari a 0. Ogni volta che le Nurbs vengono vengono cancellate, o anche in fase di inizializzazione, esse vengono ridotte a questo oggetto particolare.

Tutti i file che concorrono a definire l'oggetto Nurbs (le sue variabili, le sue funzioni membro), insieme al file `misc_functions.cpp` (con il relativo *header*), che contiene alcune funzioni e strutture di interesse generale, sono messi nella directory `\nurbs`, che viene condivisa dai programmi sviluppati, come fosse una libreria comune; ciò per favorire la compatibilità tra i software e uno sviluppo coerente e omogeneo degli stessi.

1.2.2 I formati dei file: nurbs, BSF e IGES

Il formato di file **nurbs**, usato da Atlantis, è quello che meglio permette la memorizzazione di una superficie, andando a scrivere su disco non solo la rete di punti di controllo e i vettori dei nodi, ma anche i coefficienti delle funzioni base, evitando di doverle ricalcolare ogni volta.

Oltre a questo formato si è usato anche quello **BSF**, che invece memorizza soltanto le informazioni base, ossia il grado dei polinomi, i vettori dei nodi e la rete di punti di controllo. Tale formato è stato definito da D.F.Rogers in [**?**], e rappresenta un buon compromesso tra quantità di informazioni da memorizzare e facilità di esportazione con alti programmi, pur non permettendo di andare oltre alle Nurbs biparametriche e con tre assi.

Il formato **IGES** (*Initial Graphics Exchange Specification*), è molto usato nell'industria e nei programmi di modellizzazione: con esso è possibile definire delle forme come curve Nurbs, e memorizzarle in un modo particolare, che sostanzialemente ricalca quelli appena visti, ossia i vettori dei nodi, seguiti dall'elenco dei punti di controllo, con altri importanti dettagli come il grado dei polinomi, etc.

1.2.3 Le NURBS con rigidezza variabile

Come verrà illustrato nel capitolo dedicato all'ambiente virtuale, il programma di generazione delle forze trattava le superfici come corpi a rigidezza costante, ossia restituiva un valore direttamente proporzionale (tramite una costante) allo spessore di penetrazione. Per superare questo limite, sono state sviluppate le superfici a rigidezza variabile. Esse sono delle NURBS normali, ma con assi in più, corrispondenti ai coefficienti di diverso grado. In questo modo i punti di controllo determinano non solo le coordinate geometriche dei punti circostanti (tramite la formula 1.12), ma anche le proprietà di rigidezza. Infatti quando la superficie viene rappresentata (ad esempio in Crizia, o nella creazione dei file densiometrici), o si effettuano dei calcoli sulle sue proprietà geometriche, vengono presi in considerazione gli assi 1, 2 e 3, corrispondenti a *x,y,z*. Non appena bisogna far riferimento alle proprietà di rigidezza, si calcolano i valori sull'asse 4 per avere il coefficiente di primo grado, sull'asse 5 per il coefficiente di secondo grado, e così via.

Per poter gestire le Nurbs con rigidezza variabile è stato necessario ricorrere ad un nuovo formato che permettesse l'archiviazione di superfici con un numero arbitrario di assi: è stato dunque definito il formato **VSN** (*Variable Stiffness Nurbs*). Esso è un'evoluzione del BSF, in modo da mantenere la stessa facilità di comprensione e modifica manuale, e permette di memorizzare Nurbs con un numero arbitrario di assi e con uno, due o tre parametri.

Nella tabella 1.1 è specificato il formato dei file VSN; dopo la riga d'intestazione c'è una serie di variabili che definiscono il numero di assi, di parametri, etc. Ci sono poi i vettori dei nodi, fino a 3, ed infine l'elenco dei punti di controllo: i vettori dei nodi sono costituiti da un numero di elementi pari a $numero\ di\ vertici + grado + 1$ ($grado + 1 = ordine$), mentre la matrice dei punti di controllo contiene $numero\ vertici\ U \times numero\ vertici\ V \times numero\ vertici\ W$ righe, e in ogni riga (legata ad un punto di controllo) ci sono le coordinate *x, y, z,* e i coefficienti di rigidezza, seguiti dalla coordinata uniforme, che corrisponderebbe al peso (che, come detto prima, vale sempre 1): ci sono quindi $axis + 1$ colonne.

Tipo Parametro	**Tipo Parametro**
`[Surface Identification String]`	50 caratteri max
`[degree of basis],[ID]`	grado dei polinomi ID
`[param],[axis]`	n°parametri, n°assi (peso escluso)
`[number of vertices U],...`	n°vertici di U, V e W
`[knot vector U]`	il vettore dei nodi U
`[knot vector V]`	il vettore dei nodi V
`[knot vector W]`	il vettore dei nodi W
`[CONTROL POINTS MATRIX]`	matrice dei punti di controllo

Tabella 1.1: Formato dei file VSN

1.3 Il programma ShowNurbs

Il programma ShowNurbs funziona in ambiente MatLAB e permette di visualizzare, in modo veloce e preciso, Nurbs memorizzate in formato BSF. Per il suo sviluppo si è ricorsi a due toolbox reperiti su Internet: il primo (di Bastiaan N. Veelo, 1996, ver 1.1) permette di rappresentare B-Spline leggendole da file .bsf, mentre il secondo (di D.M. Spink, 2000, ver 1.0) visualizza Nurbs, dopo aver introdotto manualmente da tastiera i valori di input (grado, vettori dei nodi, punti di controllo). L'uso del formato BSF del primo toolbox, rivelatosi abbastanza semplice e funzionale da utilizzare, ha dato spunto per la sua gestione anche all'interno dei programmi sviluppati.

Quello che si è dunque fatto è stato utilizzare il primo toolbox per la lettura dei valori di interesse dai file .bsf e poi i metodi matematici del secondo per il corretto calcolo delle superfici. Utilizzando il programma GUIDE (*Graphics User Interface Design Enviroment*) di MatLAB, è stata creata un'interfaccia grafica a finestre di dialogo, con pulsanti, barra dei menù e figura incorporata. Con essa si possono dunque caricare facilmente i file, e visualizzarli, premendo il pulsante Redraw!. A seconda della modalità scelta, è possibile mostrare a video i punti di controllo, il relativo reticolo o una ricostruzione della superficie (o anche tutto insieme, per capire come i punti di controllo vadano a *piegare* la curva). Facendo clic sulla figura e spostando il mouse è possibile ruotarla nelle tre dimensioni e

la presenza costante del reticolo consente di fare confronti diretti con le misure fisiche delle superfici in esame. Il disegno della superficie vera e propria può essere più o meno preciso a seconda del livello di risoluzione, modificabile tramite uno slider.

In un secondo tempo, sono state aggiunte delle funzionalità per modificare le superfici e salvarle: è possibile dunque cambiare la descrizione (che appare come titolo, sopra alla figura) ed effettuare traslazioni e ridimensionamenti in qualsiasi asse, attraverso finestre di dialogo richiamabili dal menù Edit.

Capitolo 2

Nurbs

In questo capitolo verranno descritti alcuni dei metodi più utilizzati per definire curve e superfici nello spazio, fino ad arrivare, in particolare, alle superfici NURBS. Sarà poi illustrato come quest'ultime, all'interno dei programmi sviluppati per questo lavoro di tesi, vengano generate, gestite, ed esportate da un programma all'altro.

2.1 Curve e superfici parametriche

La fonte di questo argomento è [**?**], mentre ulteriori approfondimenti si trovano in [**?**, **?**, **?**, **?**, **?**, **?**, **?**].

2.1.1 Forma implicita e forme parametriche

I due metodi più comuni per descrivere curve e superfici nello spazio sono la forma implicita e le forme parametriche.
Nel primo caso si usano delle equazioni a 1, 2 o 3 incognite il cui insieme di soluzioni fornisce le coordinate della figura da rappresentare. Ad esempio, nel piano XY, il cerchio di raggio unitario e centrato nell'origine degli assi è specificato dall'equazione $f(x, y) = x^2 + y^2 - 1 = 0$.
Invece, nelle forme parametriche, tutte le coordinate di ogni punto della curva sono esplicitamente definite da una funzione, legata ad un parametro indipendente;

in generale, quindi, una curva in uno spazio a 3 dimensioni è espressa da:

$$\mathbf{C}(u) = (x(u), y(u), z(u)) \qquad a \leq u \leq b \tag{2.1}$$

Tornando all'esempio del cerchio di raggio unitario, esso può essere descritto dalle seguenti funzioni parametriche

$$\begin{cases} x(u) = \cos(u) \\ y(u) = \sin(u) \end{cases} \qquad 0 \leq u \leq 2\pi \tag{2.2}$$

oppure da queste:

$$\begin{cases} x(u) = \frac{1-t^2}{1+t^2} \\ y(u) = \frac{2t}{1+t^2} \\ y(u) = \frac{2t}{1+t^2} \\ y(u) = \frac{2t}{1+t^2} \end{cases} \qquad 0 \leq t \leq 1$$

Come si è appena visto, esistono più rappresentazioni parametriche possibili. Tra la forma implicita e quella parametrica è difficile dire quale sia la migliore, dato che ognuna ha i suoi vantaggi e svantaggi, a seconda delle applicazioni. In generale, tuttavia, la forma esplicita è più facile da usare: ad esempio, per definire curve in uno spazio a tre dimensioni, la forma implicita richiede di calcolare l'intersezione di due superfici, mentre nel caso parametrico è banale passare a spazi a n-dimensioni, aggiungendo n coordinate. Nel corso di questo lavoro ci si concentrerà esclusivamente sulle rappresentazioni parametriche.

Conclusioni

In questo lavoro di tesi sono stati sviluppati diversi programmi, sia in ambiente MatLAB che in VisualC++, allo scopo di testare l'interfaccia aptica Piroga5, utilizzandola nella simulazione di un ambiente virtuale.

Tale ambiente virtuale è stato creato con l'utilizzo di superfici NURBS, gestite da routine apposite, sviluppate in lavori precedenti, il cui codice è stato riscritto utilizzando la struttura a classi, e trattando quindi ogni entità NURBS come un oggetto, con le sue variabili ed i suoi metodi.

È stato *sviluppato* un programma, che gestisce l'ambiente, restituendo i valori delle forze di reazione conseguenti agli urti con gli oggetti virtuali, e grazie a dei metodi innovativi nel calcolo del punto di minima distanza tra la punta dell'end-effector e la superficie dell'oggetto virtuale, la velocità di elaborazione è stata incrementata molto, consentendo di superare il KHz nelle frequenze di lavoro. Ciò ha permesso di creare ambienti non più limitati ad una singola superficie: con due oggetti la frequenza di lavoro si aggira sui 1100Hz. Inoltre sono state implementate anche superfici con caratteristiche di rigidezza variabile da punto a punto e in base anche allo spessore di penetrazione.

Uno dei possibili sviluppi di questo lavoro è sicuramente rappresentato dalle MetaNurbs, ultimando i metodi e le tecniche già iniziati e purtroppo non completati per mancanza di tempo. Quando esse saranno completamente operative, permetteranno di raggiungere frequenze di lavoro elevatissime consentendo così di gestire molte più superfici contemporaneamente. A tal proposito andranno necessariamente migliorate le routine per l'importazione di NURBS da file IGES.

Appendice A

Glossario

In questa sezione sono presenti il glossario dei termini usati e un elenco di chiavi di ricerca (*keyword*), con relativa traduzione italiana, utili per il reperimento degli argomenti sui motori di ricerca internazionali.

A.1 Glossario di riferimento

- **ABDUZIONE** - *abduction*
  Movimento di allontanamento del braccio dal tronco.
- **ADDUZIONE** - *adduction*
  Movimento di avvicinamento del braccio dal tronco.
- **APPARATO CAPSULO-LEGAMENTOSO**
  Sono strutture deputate a mantenere unite le estremità di due ossa contigue, e allo stesso tempo permettere il movimento di una rispetto all'altra (articolarità).
- **AFASIA**
  Perdita della capacità di comunicare oralmente, per segno o per iscritto, oppure incapacità di comprendere tali forme di comunicazione, perdita della capacità di usare il linguaggio.
- **AGRAFIA**
  Incapacità di esprimere per iscritto il proprio pensiero.

- ...

A.2 Chiavi di ricerca

Inglese	**Italiano**
post stroke rehabilitation	riabilitazione post ictus
upper limb	arto superiore
wire-driven robot	robot a cavi
degrees of freedom	gradi di libertà
robot-aided rehabilitation	riabilitazione assistita da robot
subacute phase	fase acuta
chronic-phase	fase cronica

Appendice B

Le coordinate omogenee

L'uso delle coordinate omogenee permette di rappresentare in modo alternativo funzioni razionali, quali quelle viste per le curve di Bézier razionali (eq. 1.5) e le Nurbs (eq. 1.11). L'idea è di rappresentare una curva razionale in uno spazio a n dimensioni, come una curva polinomiale (non razionale) in uno spazio a $n+1$ dimensioni. Un punto in uno spazio 3D, scritto come $\mathbf{P} = (x, y, z)$, diventa $\mathbf{P}^w = (wx, wy, wz, w)$, con $w \neq 0$. La trasformazione che permette di passare dalla coordinate omogenee a quelle "normali" è:

$$\mathbf{P} = H\{\mathbf{P}^w\} = H\{(X, Y, Z, W)\} = H\Big\{\Big(\frac{X}{W}, \frac{Y}{W}, \frac{Z}{W}\Big)\Big\}$$

Ora, dato un insieme di punti di controllo $\{\mathbf{P}_i\}$ e di pesi w_i è possibile costruire dei punti di controllo pesati $\mathbf{P}_i^w = (w_i x, w_i y, w_i z, w_i)$, e definire curve di Bézier non-razionali in uno spazio a quattro dimensioni

$$\mathbf{C}^w(u) = \sum_{i=0}^{n} B_{i,p}(u)\mathbf{P}_i^w$$

Poi, applicando la trasformazione vista sopra, si arriva alla forma razionale vista in eq. 1.5; come esempio vediamo la prima coordinata

$$x(u) = \frac{X(u)}{W(u)} = \frac{\sum_{i=0}^{n} B_{i,p}(u) w_i x_i}{\sum_{i=0}^{n} B_{i,n}(u) w_i}$$

In questo modo possiamo definire anche superfici razionali come quella già vista a pag.8 e che qui viene riportata

$$\mathbf{S}^w(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,p}(u)B_{j,q}(v)\mathbf{P}^w_{i,j}$$

da cui si ottiene, portandola allo spazio a 3 dimensioni:

$$\mathbf{S}(u,v) = H\{\mathbf{S}^w(u,v)\} = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,p}(u)B_{j,q}(v)w_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} B_{i,p}(u)B_{j,q}(v)w_{i,j}}$$

Ringraziamenti

Qua vanno messi i ringraziamenti (facoltativi)