Alma Mater Studiorum · Università di Bologna
**scuola di scienze**
**Corso di Laurea in Ingegneria e scienze informatiche**

# SUPERFICI DI SUDDIVISIONE: TEORIA, ALGORITMI ED APPLICAZIONI

**Relatore:**
**Dott.ssa**
**Damiana Lazzaro**

**Presentata da:**
**Michele Buzzoni**

**Seconda Sessione**
**Anno Accademico 2014/2015**

*Dedico questa tesi*
*alla mia famiglia*

# Introduzione

Le superfici di suddivisione sono un ottimo ed importante strumento utilizzato principalmente nell'ambito dell'animazione 3D poichè consentono di definire superfici di forma arbitraria. Questa tecnologia estende il concetto di B-spline e permette di avere un'estrema libertà dei vincoli topologici.
Per definire superfici di forma arbitraria esistono anche le *Non-Uniform Rational B-Splines* (NURBS) ma non lasciano abbastanza libertà per la costruzione di forme libere. Infatti, a differenza delle superfici di suddivisione, hanno bisogno di unire vari pezzi della superficie (*trimming*). La tecnologia NURBS quindi viene utilizzata prevalentemente negli ambienti CAD mentre nell'ambito della Computer Graphics si è diffuso ormai da più di 30 anni (anni 80), l'utilizzo delle superfici di suddivisione. Lo scopo di questa tesi è quello di riassumere, quindi, i concetti riguardo questa tecnologia, di analizzare alcuni degli schemi di suddivisione più utilizzati e parlare brevemente di come questi schemi ed algoritmi vengono utilizzati nella realtà per l'animazione 3D. La tesi sarà suddivisa in 4 capitoli: il capitolo 1 introduce il concetto di B-spline e le nozioni di base necessarie a comprendere gli schemi di suddivisione. Il capitolo 2 tratta l'algoritmo di Chaikin per la suddivisione di curve in preparazione al capitolo 3 in cui viene fatta una breve panoramica sugli schemi di suddivisione per poi trattare in dettaglio gli schemi di Doo-Sabin e Catmull-Clark. Infine il capitolo 4 tratta l'utilizzo delle superfici di suddivisione nell'animazione evidenziando i benefici che hanno portato in questo ramo della grafica computerizzata.

# Indice

# Elenco delle figure

# Elenco delle tabelle

# Capitolo 1

# Introduzione alle Curve B-spline e Nozioni di base sulla suddivisione

L'obiettivo di questo capitolo è introdurre il concetto di curve B-spline necessario a comprendere meglio gli schemi di suddivisione ed il loro funzionamento.

## 1.1 Curve B-spline

Nella progettazione 3D di forme libere, le spline sono un ottimo strumento in quanto facili da utilizzare e calcolare. Andiamo a trattare in particolare le curve B-spline che rivestono un ruolo fondamentale in questo ambito. Le curve B-spline sono curve parametriche approssimanti che consentono l'interpolazione di alcuni punti assegnati e, diversamente dalle curve di Bezier, permettono la modellazione locale di una forma. Ciò è possibile in quanto le curve B-spline utilizzano funzioni base di tipo polinomiale di grado predeterminato a supporto compatto che hanno influenza locale, in questo modo la posizione di un punto influenza solo una parte di curva.

### 1.1.1 Formulazione

la formulazione matematica delle curve B-spline è la seguente:

$$\sum_{i=0}^{n} p_i N_{i,k}(t)$$

con **k** ordine della curva: $2 \leq k \leq n+1$
$N_{i,k}(u)$ Funzione base. Polinomio di grado k – 1, a supporto compatto
$t \in [a,b]$
con $\mathbf{p}_i$ con i = 0,1,2 ... n
insieme di punti che formano il cosi detto *poligono di controllo* che fornisce una prima approssimazione della curva

## 1.2 Funzioni polinomiali definite a tratti

Sia [a,b] un intervallo chiuso e limitato dell'asse reale e sia $\Delta$ una partizione di [a,b] con:

$$\Delta = \{ a \equiv x_0 < x_1 < ... < x_k < x_{k+1} \equiv b \}$$

che genera k+1 sotto intervalli così definiti:
$I_i = [x_i,x_{i+1})$ i=0, ... ,k-1,
$I_k = [x_k,x_{k+1}]$.

Fissato un intero positivo $m$ con $m<k$ si definisce spline polinomiale di ordine $m$ (grado $m$ - 1), una funzione s(x) che in ciascun intervallo $I_i$ *con i=0, ... k* coincide con un polinomio $s_i(x)$ di ordine $m$, ed inoltre nei punti $x_i$ $i$=1,... $k$ detti **nodi semplici** della spline, soddisfa le seguenti condizioni di raccordo:

$$\frac{d^j s_{i-1}(x_i)}{dx^j} = \frac{d^j s_i(x_i)}{dx^j} \text{ per i} = 1, ...k \text{ e j} = 0, ..., m-2.$$

Le condizioni di raccordo ci garantiscono che $s(x) \in C^{m-2}_{[a,b]}$ ovvero è continua in [a,b] insieme alle sue derivate fino a quella di ordine *m* - 2.
Se i *k* nodi sono equamente distribuiti in [a,b] la spline è *uniforme*, in caso contrario è *non-uniforme*.
Al crescere del grado, le funzioni base diventano più smussate ed hanno un supporto maggiore.

## 1.3 Formule di Cox per la valutazione di una B-spline

Per calcolare le funzioni base, si utilizzano le formule di Cox, che consentono di calcolare $N_{i,m}(x)$ mediante combinazione di due funzioni base di ordine *m - 1*.
Più precisamente, definita

$$N_{i,1}(x) = \begin{cases} 1 & \text{se } x_i \leq x \leq x_i + 1 \\ 0 & \text{altrimenti} \end{cases}$$

per $h = 2, \ldots, m$ si ha

$$N_{i,h}(x) = \left( \frac{x - x_1}{x_i + h - 1} N_{i,h-1}(x) + \frac{x_{i+h} - x}{x_{i+h} - x_{i+1}} N_{i+1,h-1}(x) \right)$$

Le B-spline più comuni sono le quadratiche e le cubiche. Le funzioni base di B-spline uniformi quadratiche possono essere facilmente precalcolate e sono le stesse per ogni segmento (fatto salvo casi di translazione):

$$N_{i,2}(x) = \begin{cases} \frac{1}{2}x^2 \\ -x^2 + x + \frac{1}{2} \\ \frac{1}{2}(1-x)^2 \end{cases}$$

Le funzioni base di B-spline cubiche con vettore dei nodi uniforme sono, invece, le seguenti:

$$N_{i,2}(x) = \begin{cases} \frac{1}{6}x^3 \\ \frac{1}{6}(-3x^3 + 3x^2 + 3x + 1) \\ \frac{1}{2}(3x^3 - 6x^2 + 4) \\ \frac{1}{6}(1-t)^3 \end{cases}$$

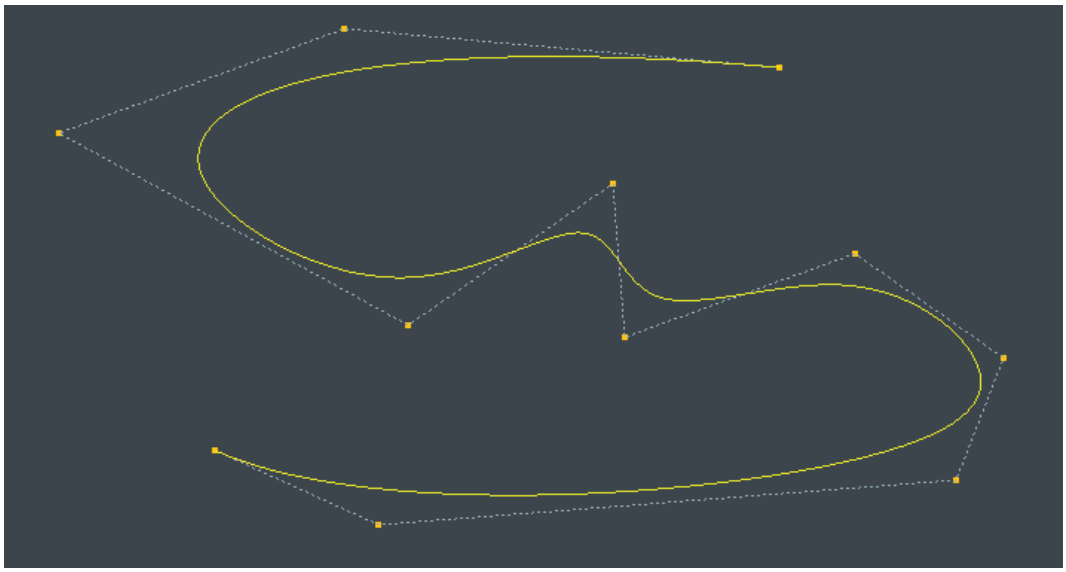


Figura 1.1: Esempio di curva spline polinomiale con il suo poligono di controllo.

## 1.4 Curve Spline polinomiali di ordine $m$ a nodi multipli

Le spline polinomiali a nodi semplici sono funzioni che presentano la massima regolarità per essere polinomiali a tratti. Infatti, se si aggiunge un altro ordine di raccordo per le derivate si otterrebbe semplicemente un polinomio di ordine $m$.
Questa massima regolarità di classe $C^{m-2}$ è a volte troppo elevata per permettere alla funzione spline di seguire bene l'andamento di dati che presentano un comportamento in parte molto regolare, in parte meno regolare. E' importante aggiungere gradi di libertà alla spline, riducendo all'occasione le condizioni di raccordo sulle derivate. Questo può essere fatto introducendo il concetto di **nodi multipli**.

### 1.4.1 Spline polinomiali a nodi multipli

Sia [a,b] un intervallo chiuso e limitato, sia $\Delta$ una partizione di [a,b] così definita:

$\Delta = \{ \, a \equiv x_0 < x_1 < ... < x_k < x_{k+1} \equiv b \, \}$

che induce una partizione di [a,b] in $k$+1 sotto intervalli:

$I_i = [x_i, x_{i+1})$ con $i$=0, ... ,$k$-1,
$I_k = [x_k, x_{k+1}]$.

Sia $m$ un intero positivo e sia $M = (m_1 , m_2 , ..., m_k )$ un vettore di interi positivi tali che $1 \leq m_i \leq m \; \forall i = 1 , ... \; k$. Si definisce funzione spline polinomiale di ordine m con nodi $x_1$ ,..., $x_k$ di molteplicità $m_1$ ,..., $m_k$, una funzione $s(x)$ che in ciascun sotto intervallo $I_i$ $i$=0,...,$k$ coincide con

un polinomio $s_i$ (x) di ordine $m$, e che nei nodi $x_i$ , $i$=1 ,..., $k$ soddisfi le condizioni di continuità:

$$\frac{d^j s_{i-1}(x_i)}{dx^j} = \frac{d^j s_i(x_i)}{dx^j} \text{ per } i = 1, ...k \text{ e } j = 0, ..., m - m_i - 1.$$

La differenza con le spline polinomiali a nodi semplici nasce dal fatto che è possibile variare il tipo di raccordo in un nodo $x_i$ a seconda del valore di $m_i$. In base a questo valore si può decidere sino a quale derivata effettuare il raccordo. Maggiore sarà la molteplicità, minori saranno le condizioni di raccordo sui nodi.

### 1.4.2 Esempi

Sia $m$=4.

Se, per un certo $i$, $1 \leq i \leq k$ , $m_i$ =1 il nodo $x_i$ è semplice

$$s_i^2(x_i) = s_{i-1}^2(x_i)$$
$$s_i^1(x_i) = s_{i-1}^1(x_i)$$
$$s_i(x_i) = s_{i-1}(x_i)$$

Se $m_i$ =2 il nodo $x_i$ è doppio

$$s_i^1(x_i) = s_{i-1}^1(x_i)$$
$$s_i(x_i) = s_{i-1}(x_i)$$

Se $m_i$ =3 il nodo $x_i$ è triplo

$$s_i(x_i) = s_{i-1}(x_i)$$

si osserva che man mano che aumenta la molteplicità dei nodi, si riducono le condizioni di raccordo nei nodi stessi.

# Capitolo 2

# Algoritmo di Chaikin per la generazione di curve B-spline

Per avvicinarci alle superfici di suddivisione dobbiamo passare prima per le curve di suddivisione. Uno degli algoritmi più diffusi è quello di Chakin che viene illustrato in questo capitolo in due diverse varianti.

## 2.1 Suddivisione di curve

A partire da un iniziale poligono di controllo, l'algoritmo di Chaikin è basato sul taglio degli angoli per la definizione di curve mediante raffinamenti ricorsivi. Al limite di tali raffinamenti l'algoritmo produce curve B-spline quadratiche uniformi.

## 2.2 L'algoritmo di Chaikin

L'idea generale è quella di partire da un insieme dato di punti di controllo $P_i^0$, iterare un procedimento che risulti un nuovo insieme di punti $P_i^1$, e ripetere il procedimento, producendo di volta in volta ulteriori insiemi di

punti $P_i^k$.

Le sequenze di punti sono di volta in volta le seguenti:

$$P_0^0,\ P_1^0\ ,P_2^0,\ \ldots\ ,\ P_{n0}^0$$
$$P_0^1,\ P_1^1\ ,P_2^1,\ \ldots\ ,\ P_{n1}^1$$
$$P_0^2,\ P_1^2\ ,P_2^2,\ \ldots\ ,\ P_{n2}^2$$
$$\ldots$$
$$P_0^k,\ P_1^k\ ,P_2^k,\ \ldots\ ,\ P_{nk}^k$$

Ogni punto $P_j^k$ viene calcolato come somma pesata dei punti $P_i^{k-1}$ :

$$P_j^k = \sum_{i=0}^{n_k-1} a_{ijk} P_i^{k-1}$$

dove gli $a_{ijk}$ sono coefficienti reali. Ogni iterazione produce un numero $n_k + 1$ di punti che è in generale differente da iterazione a iterazione. Se $n_k$ decresce in maniera stretta al crescere di k, allora il numero di punti diventa sempre più piccolo, fino a quando rimane un solo punto. È possibile anche che $n_k$ cresca all'aumentare di k, producendo di volta in volta più e più punti ad ogni iterazione. Dopo aver ottenuto un numero sufficiente di punti a soddisfare le nostre condizioni di precisione, si traccia la curva congiungendo nell'ordine i punti tramite segmenti. L'algoritmo di Chaikin può essere visto come caso particolare di questo modo di procedere.

Si parta dagli n+1 punti di controllo che denotiamo come $P_0^0\ ,P_1^0\ ,\ \ldots\ ,\ P_n^0$ e si applichi la seguente regola di raffinamento, come visibile in figura, tratta da [1].

$$P_{2j}^{k+1} = \frac{3}{4}P_j^k + \frac{1}{4}P_{j+1}^k$$
$$P_{2j+1}^{k+1} = \frac{1}{4}P_j^k + \frac{3}{4}P_{j+1}^k$$

La prima iterazione inizia con gli $n$+1 punti originali producendo i $2n$ punti. Ogni iterazione seguente aumenta il numero di punti portandoli più

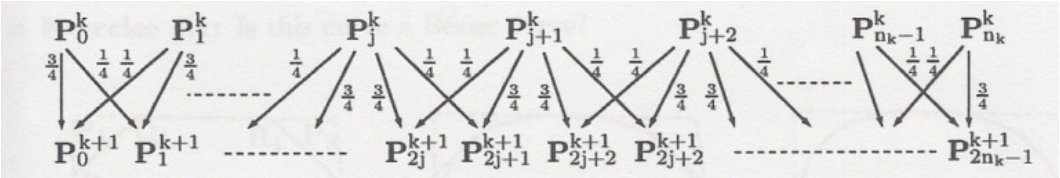


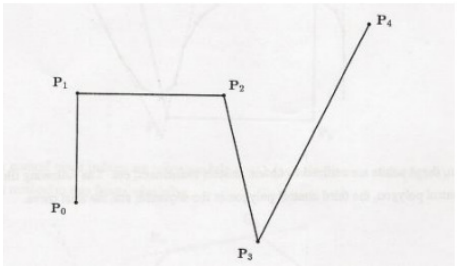

Figura 2.1: Regola di raffinamento dell'algoritmo di Chaikin

vicini alla posizione limite che consiste nella posizione teorica della curva. Dopo $k$ iterazioni, con $k$ che dipende dalla precisione richiesta, la curva viene rappresentata tracciando segmenti tra i punti prodotti nell'ultima iterazione. Una semplice interpretazione geometrica di questo fatto è la seguente: consideriamo per esempio un poligono di controllo composto da cinque punti, la regola di raffinamento sarà eseguita prendendo, per ogni $i$, il segmento tra $\mathbf{P}_i$ e $\mathbf{P}_{i+1}$, chiamando $\mathbf{Q}_i$ e $\mathbf{R}_i$ i punti che stanno rispettivamente a un quarto e a tre quarti di distanza da $\mathbf{P}_i$ sul segmento in questione (ciò in modo tale che, posto $\mathbf{M}_i$ il punto medio del segmento, i punti $\mathbf{P}_i$, $\mathbf{Q}_i$, $\mathbf{M}_i$, $\mathbf{R}_i$, $\mathbf{P}_{i+1}$ siano allineati, consecutivi e tali per cui ognuno disti dal precedente e dal successivo un quarto della lunghezza del segmento), e osservando che valgono le seguenti combinazioni lineari che sono anche baricentriche e convesse:

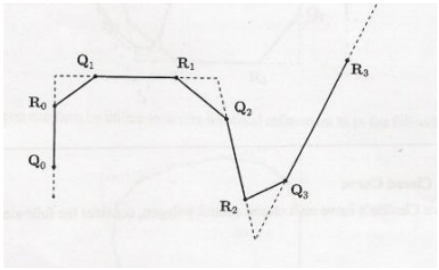
$$\mathbf{Q}_i = \frac{3}{4}\mathbf{P}_i + \frac{1}{4}\mathbf{P}_{i+1}$$
$$\mathbf{R}_i = \frac{1}{4}\mathbf{P}_i + \frac{3}{4}\mathbf{P}_{i+1}$$

In questo modo, partendo da *n+1* punti di controllo che definiscono un poligono di controllo composto da $n$ lati, arriviamo ad avere *2n* nuovi punti $\mathbf{Q}_i$ e $\mathbf{R}_i$. Questi possono ora essere collegati in modo da formare un nuovo poligono di controllo composto da *2n−1* lati. Ripetendo questo procedimento, i poligoni di controllo che via via possono essere formati si avvicinano sempre più alla posizione limite che consiste nella posizione della curva di Chaikin. Notare inoltre che gli $\mathbf{M}_i$ stanno tutti sulla curva; non solo, i punti medi di tutti i segmenti generati nei successivi passi dell'algoritmo di raffinamento

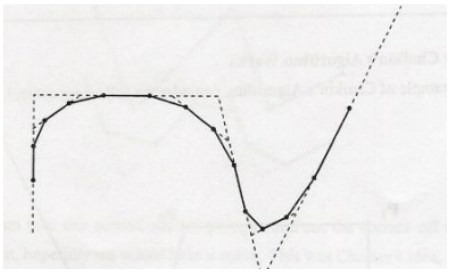
stanno sulla curva. Questa curva risulta essere una curva di Bezier, anche se definita da altri punti di controllo rispetto a quelli utilizzati nei passi dell'algoritmo, dal momento che essa non passa per il primo e per l'ultimo punto di controllo. L'algoritmo può essere facilmente adattato per la generazione di curve chiuse: l'unica modifica necessaria è quella di collegare il primo e l'ultimo punto e $\mathbf{P}_n$ e calcolarsi con essi $\mathbf{Q}_n$ e $\mathbf{R}_n$. Questo può essere fatto definendo $\mathbf{P}_{n+1}$ come duplicato di $\mathbf{P}_0$, analogamente a come si fa in generale quando si trattano curve chiuse.



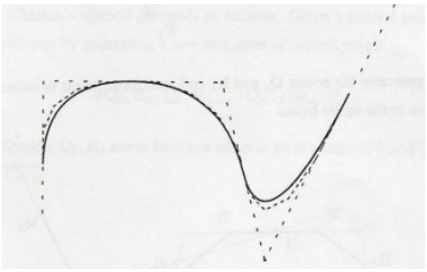

(a) Poligono di controllo aperto.




(b) Primo passo dell'algoritmo.



(c) Secondo passo dell'algoritmo.



(d) Curva limite.

Figura 2.2: Passi dell'algoritmo di suddivisione di Chaikin.

## 2.3 Algoritmo di suddivisione multipasso

Abbiamo appena visto che una curva di Chaikin è una curva di suddivisione, e che tale procedimento ha inizio a partire da una curva di controllo poligonale. Ad ogni passo del metodo di suddivisione, gli angoli della curva polinomiale vengono tagliati, e una nuova curva viene ottenuta e data in

input per il passo successivo; come limite del procedimento, si ottiene una curva di Chaikin. In questa sezione esamineremo un algoritmo di suddivisione multipasso per la generazione di tali curve, tale per cui, dato un intero positivo arbitrario $k$, sia possibile realizzare la curva poligonale risultante al $k$-esimo passo del procedimento di suddivisione direttamente a partire dal solo poligono di controllo iniziale. Questo procedimento rende più veloce la generazione di curve.

### 2.3.1 Come viene migliorato l'algoritmo

Di solito tutti i procedimenti di suddivisione considerano un poligono di controllo iniziale, denotato per esempio come $\mathbf{L}_0$ , e si occupano di raffinarlo passo dopo passo. Ad ogni passo, il procedimento viene applicato al poligono $\mathbf{L}_k$ una sola volta, restituendo un nuovo poligono $\mathbf{L}_{k+1}$ , che servirà da input per il passo successivo della suddivisione. In questo caso, $\mathbf{L}_k$ , con $k = 0, 1, 2, ...$, rappresenta il poligono di controllo risultante dopo il $k$-esimo passo del procedimento di suddivisione. Di conseguenza, secondo gli schemi di suddivisione classici, per ogni $k$, prima di poter calcolare $\mathbf{L}_{k+1}$ , è necessario già avere calcolato $\mathbf{L}_k$ . In questa parte di capitolo ricaveremo invece un metodo per calcolare $\mathbf{L}_k$ direttamente a partire dal solo $\mathbf{L}_0$ , secondo lo schema di suddivisione di Chaikin: in questo modo, non è necessario calcolare $\mathbf{L}_i$ per ogni $i = 1, 2, ..., k-1$ prima di arrivare a $\mathbf{L}_k$ , e a partire dallo stesso poligono di controllo $\mathbf{L}_0$ , la curva poligonale $\mathbf{L}_k$ che si ottiene con questo metodo è la stessa che si ottiene alla $k$-esima esecuzione del metodo tradizionale di Chaikin. In questo modo, è possibile risparmiare memoria in quanto non è necessario allocare i dati relativi ai precedenti passi della suddivisione, inoltre questo metodo risulta più veloce dell'algoritmo tradizionale di Chaikin.

### 2.3.2 L'algoritmo multipasso

Esaminiamo ora nel dettaglio in che cosa consiste questo algoritmo, proposto in [2], fornendo formule esplicite per il calcolo diretto di $\mathbf{L}_k$ direttamen-

te a partire da $\mathbf{L}_0$ . L'idea fondamentale di Chaikin è il taglio degli angoli: tutti gli angoli della curva poligonale vengono tagliati ricorsivamente, e man mano che il procedimento di suddivisione avanza, le curve poligonali ottenute di volta in volta diventano sempre più lisce; al limite, si ottiene una curva liscia, con condizioni di regolarità geometrica e parametrica determinate. Denotiamo adesso con

$$\mathbf{L}_0 = \{\mathbf{P}_{0,0}, \mathbf{P}_{1,0}, ..., \mathbf{P}_{n_0,0}\}$$

i punti di controllo iniziali, e analogamente per un $k$ qualunque:

$$\mathbf{L}_k = \{\mathbf{P}_{0,k}, \mathbf{P}_{1,k}, ..., \mathbf{P}_{n_k,k}\}$$

Così facendo, il primo intero a pedice indica il numero ordinale del punto (cominciando da 0), mentre il secondo il passo della suddivisione relativo al punto in questione. Tornando all'algoritmo, una volta posta questa notazione, e analizzati gli angoli della curva poligonale iniziale, troviamo che i punti $\mathbf{P}_{2^k(i-1)+j,k}$, dove $j = 1, 2, ..., 2^k$ , appartenenti alla curva poligonale $\mathbf{L}_k$ , dipendono soltanto da $\mathbf{P}_{i-1,0}$ , $\mathbf{P}_{i,0}$ , $\mathbf{P}_{i+1,0}$ . Pertanto, per induzione, si dimostra il seguente teorema che costituisce di fatto la procedura dell'algoritmo.

**Teorema 1.** *Tutti i punti in* $\mathbf{L}_k$ , *con* $k = 1, 2, ...,$ *possono essere calcolati direttamente a partire unicamente da* $\mathbf{L}_0$ *tramite le seguenti formule*:

$$\mathbf{P}_{0,k} = (2^{-1} + 2^{-(k+1)})\mathbf{P}_{0,0} + (2^{-1} - 2^{-(k+1)})\mathbf{P}_{1,0}$$

$$\mathbf{P}_{2^k(i-1)+j,k} = F(j,k)\mathbf{P}_{i-1,0} + G(j,k)\mathbf{P}_{i,0} + H(j,k)\mathbf{P}_{i+1,0}$$

$$\mathbf{P}_{2^k n_0 - 2^k + 1,k} = (2^{-1} - 2^{-(k+1)})P_{n_0-1,0} + (2^{-1} + 2^{-(k+1)})P_{n_0,0}$$

*dove* $i = 1, 2, ..., n_0 - 1$, $j = 1, 2, ..., 2^k$ , *e*:

$$F(j,k) = 2^{-1} - 2^{-(k+1)} - (j-1)(2^{-k} - 2^{-2k-1}j)$$

$$G(j,k) = 2^{-1} + 2^{-(k+1)} + (j-1)(2^{-k} - 2^{-2k}j)$$

$$H(j,k) = j(j-1)(2^{-2k-1})$$

*Dimostrazione.* Dimostriamo per induzione: le formule valgono per $k = 1$, infatti:

$$F(1,1) = \frac{1}{4};\ \ G(1,1) = \frac{3}{4};\ \ H(1,1) = 0$$

$$F(2,1) = 0;\ \ G(2,1) = \frac{3}{4};\ \ H(2,1) = \frac{1}{4}$$

da cui, applicandole alla terna di equazioni, precedentemente illustrate per i punti, arriviamo a:

$$\mathbf{P}_{2i,1} = \frac{3}{4}\mathbf{P}_{i,0} + \frac{1}{4}\mathbf{P}_{i+1,0}$$

$$\mathbf{P}_{2i+1,1} = \frac{1}{4}\mathbf{P}_{i,0} + \frac{3}{4}\mathbf{P}_{i+1,0}$$

per ogni $i = 0,\ 1,\ ...,\ n_0 - 1$. Si nota che queste due equazioni ricalcano fedelmente lo schema di Chaikin; pertanto, le formule valgono per $k = 1$. Supponiamo ora le formule valide per $k = m$ e dimostriamo che esse valgono anche per $k = m + 1$. Per provarlo, applichiamo l'algoritmo di Chaikin ai punti ottenuti al passo $m$-esimo della suddivisione:

$$\mathbf{P}_{0,m+1} = \frac{3}{4}\mathbf{P}_{0,m} + \frac{1}{4}\mathbf{P}_{1,m}$$

$$\mathbf{P}_{2^{(m+1)}(i-1)+j,m+1} = \frac{1}{4}\mathbf{P}_{2^m(i-1)+\frac{j-1}{2},m} + \frac{3}{4}\mathbf{P}_{2^m(i-1)+\frac{j+1}{2},m} \quad \text{per } j \text{ dispari}$$

$$\mathbf{P}_{2^{(m+1)}(i-1)+j,m+1} = \frac{3}{4}\mathbf{P}_{2^m(i-1)+\frac{j}{2},m} + \frac{1}{4}\mathbf{P}_{2^m(i-1)+\frac{j}{2},m} \quad \text{per } j \text{ pari}$$

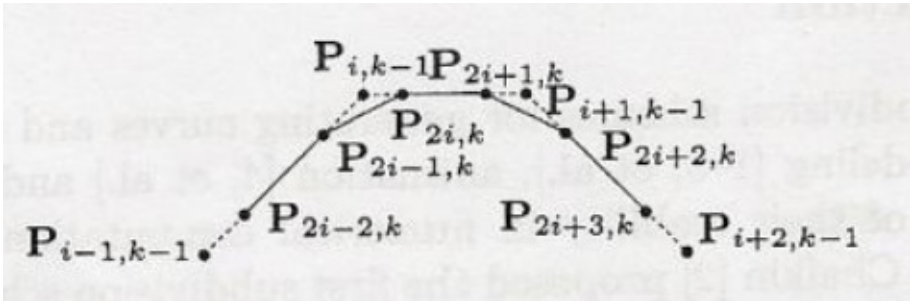
$$\mathbf{P}_{2^{(m+1)}n_0-2^{(m+1)}+1,m+1} = \frac{1}{4}\mathbf{P}_{2^m n_0-2^m,m} + \frac{3}{4}\mathbf{P}_{2^m n_0-2^m+1,m}$$

dove $i = 1, 2, ..., n_0-1$, e $j = 1, 2, ..., 2\ m+1$ . Si sostituiscano ora tutti i punti in $\mathbf{L}_m$ con le espressioni che contengono solo punti in $\mathbf{L}_0$ ; ciò è possibile poichè abbiamo supposto che le formule fossero valide per $k = m$, come ipotesi induttiva. Svolgendo i calcoli, si ottengono le formule che si otterrebbero sostituendo $k = m+1$, e ciò prova la validità di esse anche per il passo successivo. Tramite induzione si è dunque provato che tali espressioni valgono per ogni $k$.

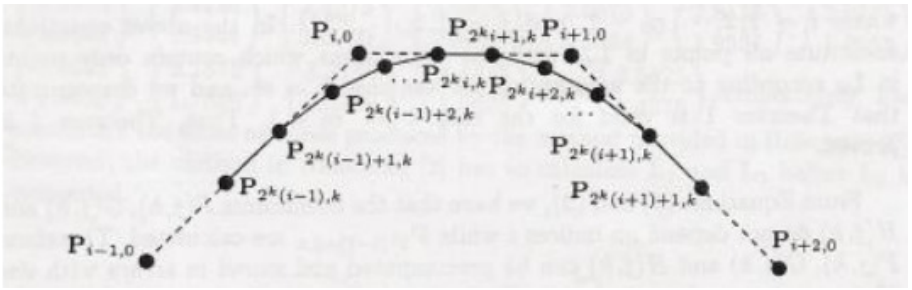
Poichè $F(j, k)$, $G(j, k)$, e $H(j, k)$ non dipendono dagli indici $i$, è possibile calcolarli preventivamente e allocarli in strutture dati di dimensioni $2^k$ in modo tale da rendere più veloci le operazioni. L'algoritmo può essere pertanto sintetizzato nei seguenti passi:

1. Si calcolino $F(j, k)$, $G(j, k)$, e $H(j, k)$, per ogni $j = 1, 2, ..., 2^k$ );
2. Si calcolino $\mathbf{P}_{0,k}$ e $\mathbf{P}_{2^k n_0 - 2^k + 1, k}$ ;
3. Si calcoli $\mathbf{P}_{2^k(i-1)+j,k}$ per ogni coppia $(i, j)$ da $(1, 1)$ a $(n_0 - 1, 2^k)$, dove si fa variare più internamente $j$ e più esternamente $i$.

Nelle figure in seguito è mostrato graficamente come si comportano i 2 algoritmi di Chaikin appena visti in questo capitolo.




(a) Passo di suddivisione semplice: $\mathbf{L}_k$ è calcolato a partire da $\mathbf{L}_{k-1}$. [2]




(b) Passo di suddivisione multiplo:$\mathbf{L}_k$ è calcolato direttamente a partire da $\mathbf{L}_0$. [2]

Figura 2.3: Algoritmi di Chaikin a confronto.

# Capitolo 3

# Superfici di suddivisione

L'algoritmo di Chaikin esposto nel capitolo precedente per la suddivisione di curve può essere usato come introduzione alle superfici di suddivisione. Si è visto infatti come da un poligono di controllo si è passati ad una curva, ottenendo quindi un effetto di *smoothing*. Lo stesso principio viene applicato alle superfici attraverso vari schemi di suddivisione che permettono quindi di colmare il gap tra rappresentazione *discreta* e *continua*. Nel seguito del capitolo viene proposta una classificazione ed analisi prestando particolare attenzione agli schemi di suddivisione di Doo-Sabin e Catmull-Clark che sono gli algoritmi maggiormente utilizzati in Computer graphics. Le superfici di suddivisione possono essere studiate sia nel caso uniforme che non uniforme. L'utilizzo di superfici di suddivisione non-uniformi implica l'utilizzo di B-spline non-uniformi in cui i nodi, al contrario della B-spline uniforme, possono essere posizionati arbitrariamente. Questo capitolo analizzerà solamente superfici di suddivisione uniformi per mantenere semplice la trattazione.

## 3.1 Panoramica

Prima di addentrarci nella trattazione di specifici schemi di suddivisione, è bene presentare una classificazione in base a diverse caratteristiche di tali schemi e chiarire la terminologia adottata.

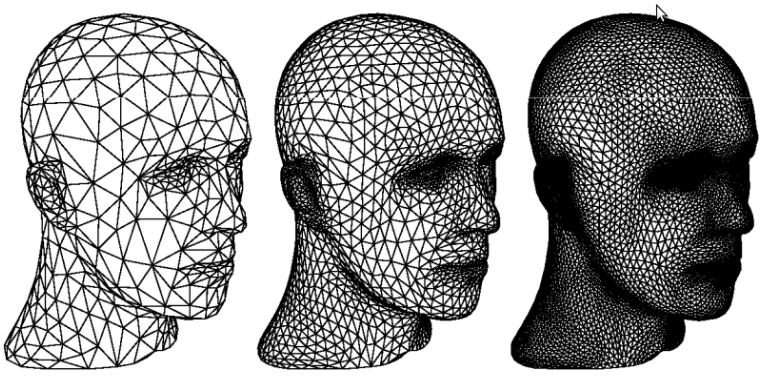


Figura 3.1: Esempio di effetto smoothing.

## 3.2 Caratteristiche degli schemi di suddivisione

Una prima fondamentale caratterizzazione degli schemi di suddivisione riguarda gli schemi *approssimanti* ed *interpolanti*. Se la superficie limite dello schema di suddivisione non attraversa i punti di controllo iniziali il metodo è detto approssimante. Al contrario, lo schema di suddivisione è interpolante se i vertici della mesh iniziale sono ancora vertici della superficie limite.
La gestione delle mesh può essere di due tipi in base alla costruzione delle sue facce: quelli definiti da tasselli *triangolari* e quelli definiti da tasselli *quadrilateri*. Una volta scelto il tipo di faccia, è necessario definire come sono legate le facce raffinate con quelle originali. Esistono due approcci principali che sono utilizzati per generare nuove facce raffinate: la divisione di faccia, e la divisione di vertice (figura 3.2). Gli schemi che utilizzano il primo metodo vengono detti *primali*, mentre gli schemi che utilizzano il secondo vengono detti *duali*. Negli schemi primali per creare le nuove facce della mesh si introduce un nuovo vertice per ogni lato e si uniscono tali vertici in modo da creare le nuove facce. Negli schemi duali ogni vertice viene splittato in modo da creare un nuovo vertice per ogni faccia adiacente a questo.

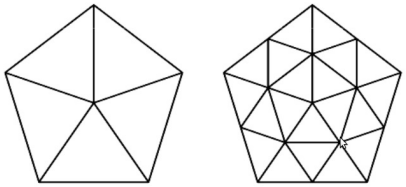
E possibile classificare gli schemi di suddivisione basandosi su 4 criteri:

**Tipo di mesh** triangolare o quadrangolare;

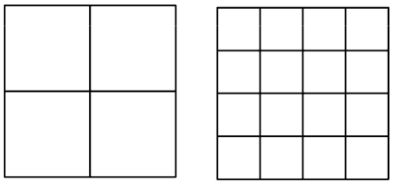
**Tipo di regola** primale (face-splitting) o duale (vertex-splitting);

**Schema Approssimante** o **interpolante**;

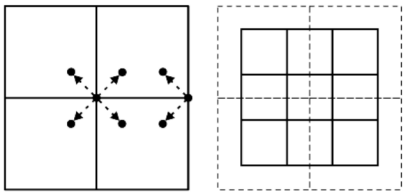
**Smoothness** proprietà di continuità della superficie limite ($C^0$,$C^1$,...).



(a) Schema Primale per mesh triangolare.



(b) Schema Primale per mesh quadrangolare.



(c) Schema Duale per mesh quadrangolare.

Figura 3.2: Esempi di schemi Primali e Duali.

Uno schema di suddivisione è definito da un insieme di regole *topologiche* e un insieme di regole *geometriche* per il raffinamento della mesh. Le regole topologiche riguardano, ad esempio, l'inserimento di nuovi vertici nei lati e nelle facce, l'aggiornamento di vertici già esistenti, la connessione dei nuovi vertici, l'eliminazione di alcuni vertici, lati e facce. Le regole geometriche, invece, sono utilizzate per calcolare le coordinate esatte dei nuovi vertici di controllo. Quando si progettano nuove regole geometriche bisogna tenere in considerazione alcuni fatti chiave, come ad esempio l'invarianza affine, la simmetria e il comportamento della superficie limite.
Per una mesh triangolare i vertici generati dalla suddivisione assumono valenza 6 (con il termine valenza riferita ad un vertice, si intende il numero

di lati che raggiunge il vertice stesso). I vertici di una mesh quadrangolare invece assumono valenza 4. Tali vertici sono detti vertici *regolari*. Una mesh si dice *semi-regolare* se, ad eccezione dei vertici iniziali, tutti i suoi vertici sono regolari. I vertici non regolari sono chiamati *straordinari*.
Negli schemi primali, i vertici della mesh grezza sono anche i vertici della mesh raffinata. Per ogni livello di suddivisione, tutti i nuovi vertici inseriti sono chiamati vertici *dispari*. Mentre i vertici ereditati dal livello precedente sono detti vertici *pari*. Negli schemi che utilizzano mesh quadrilatere, alcuni vertici sono inseriti in corrispondenza della divisione di lati, altri al posto di facce. Questi due tipi di vertici dispari sono chiamati rispettivamente vertici di bordo (*edge*) e di faccia (*face*), mentre i vertici pari sono detti punti di vertice (*vertex*).

| **Schema di suddivisione** | | **Tipo** | **Regola** | **Continuità** |
|---:|:---:|:---:|:---:|:---:|
| Butterfly | Interp. | Triang. | Primale | $C^1$ |
| Catmull-Clark | Approx. | Quadr. | Primale | $C^2$ |
| Doo-Sabin | Approx. | Quadr. | Duale | $C^1$ |
| Kobbelt | Interp. | Quadr. | Primale | $C^1$ |
| Kobbelt ($\sqrt{3}$) | Approx. | Triang. | Duale | $C^2$ |
| Loop | Approx. | Triang. | Primale | $C^2$ |

Tabella 3.1: Esempi di schemi di suddivisione con relative caratteristiche.

## 3.3 Superfici di suddivisione Interpolanti

Oltre agli schemi più usati che sono schemi approssimanti, che andremo ad esaminare nella prossima sezione, si è voluto parlare brevemente anche di altri schemi che però sono interpolanti per cercare di dare al lettore un quadro più completo. Trattiamo in maniera semplificata due importanti schemi interpolanti: schema Butterfly e schema Kobbelt.

### 3.3.1 Schema Butterfly

Questo schema viene utilizzato per tipi di mesh con facce triangolari. Esso trasforma ricorsivamente ogni faccia triangolare del poliedro di controllo in patch formate da quattro facce a loro volta triangolari che interpolano i vecchi punti di controllo. Così la nuova triangolazione conserva i punti di controllo della precedente triangolazione ed i nuovi vertici sono aggiunti sui bordi della vecchia triangolazione. La regola per inserire nuovi punti è una regola ad otto punti (*eight-point rule*) chiamata schema Butterfly poichè è basata sulla configurazione mostrata in figura 3.3 che ricorda appunto la forma di una farfalla. Aiutandoci con la figura 3.3 esaminiamo la regola per aggiungere il nuovo punto $\mathbf{E}_1$ corrispondente all'edge ($d_1$,$d_2$)

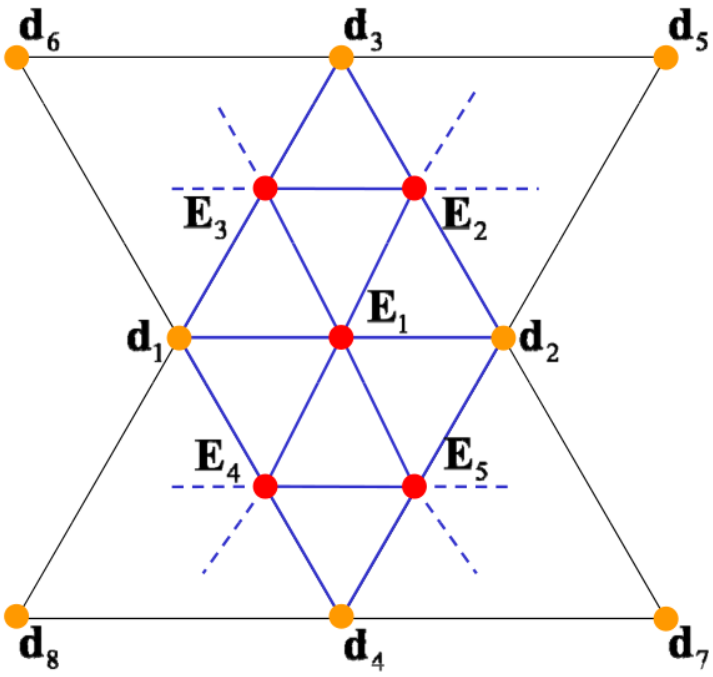



Figura 3.3: Schema Butterfly.

La formula è la seguente:

$$\mathbf{E}_1 = u(d_1 + d_2) + v(d_3 + d_4) - \omega(d_5 + d_6 + d_7 + d_8)$$

IL nuovo punto $\mathbf{E}_1$ viene connesso ai vecchi punti dell'edge $d_1$ e $d_2$ ed ai nuovi punti costruiti dagli edge ($d_1$,$d_3$) e ($d_2$,$d_3$). Una volta calcolati tutti i nuovi punti, il processo viene ripetuto in modo ricorsivo utilizzando i punti delle nuove triangolazioni. $u, v$, e $\omega$ sono parametri che devono essere scelti in modo che il limite del processo produca superfici con continuità $C^1$. Per ottenere questo tipo di continuità è necessario scegliere $u= \frac{1}{2}$, e $v = 2w$.

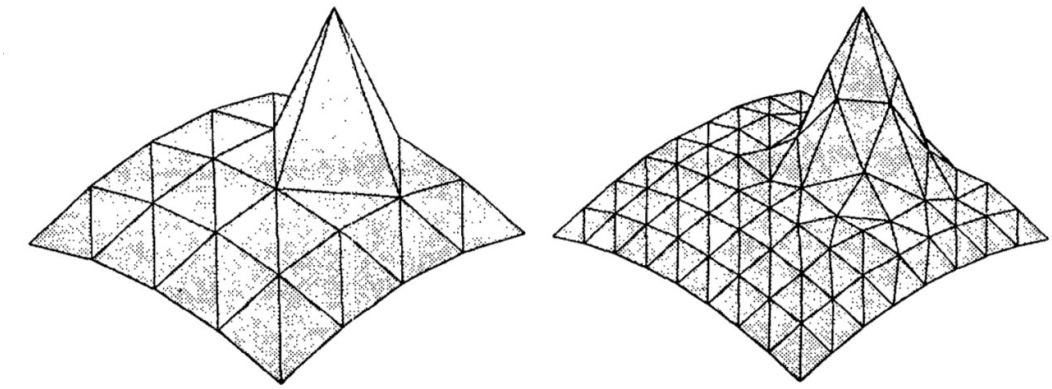
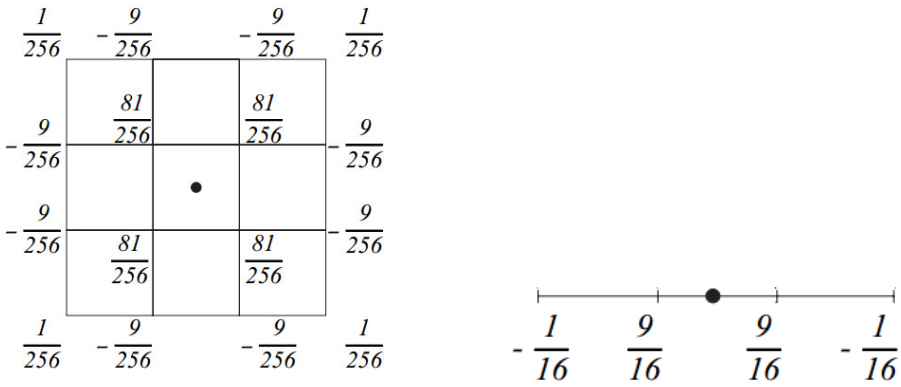


Figura 3.4: Un passo dello schema Butterfly con $\omega = \frac{1}{18}$.

### 3.3.2 Schema di Kobbelt

Questo schema di suddivisione venne descritto da Kobbelt in [9]. Per mesh regolari quadrilatere, esso si riduce al prodotto tensioriale dei quattro punti dello schema. La continuita $C^1$ è provata per tutte le valenze dei vertici. I tre tipi di punti vengono calcolati in questo modo: i punti di vertice vengono lasciati soli perchè sono i punti di controllo originali e non cambiano in questo schema interpolante. I punti di bordo vengono generati sui bordi, a metà tra i vecchi punti di bordo. I punti di faccia vengono calcolati invece attraverso una regola basata sui 4 punti di bordo della faccia stessa. Il discorso è diverso per vertici straordinari, ovvero con valenza diversa da quattro. In figura 3.5 vengono mostrate le maschere di suddivisione per il calcolo dei punti di bordo e di faccia. Per il calcolo dei vertici straordinari e per approfondire questo schema si rimanda a [9].




(a) Maschera per punti di faccia. (b) Maschera per punti di bordo.

Figura 3.5: Maschere per schema di Kobbelt.

## 3.4 Superfici di suddivisione Approssimanti

In questa sezione tratteremo degli schemi di suddivisione più importanti ed usati: Doo-Sabin e Catmull-clark entrambi appartenenti alla famiglia degli schemi approssimanti.

### 3.4.1 Schema di Doo-Sabin

Daniel Doo e Malcolm Sabin presero il paradigma di raffinamento sviluppato da George Chaikin e, adattando queste tecniche per le *superfici B-spline biquadratiche uniformi*, hanno sviluppato una nuova procedura di definizione della superficie basata sul raffinamento: a partire da una mesh di controllo rettangolare, la suddivisione di Doo-Sabin produce al limite superfici B-spline bi-quadratiche uniformi.
Date le maschere generate per suddivisione di patch B-spline biquadratiche uniformi, Doo e Sabin hanno osservato che i punti sono semplicemente la media di quattro punti particolari presi da un poligono:

**Il vertice** (*vertex point*) per il quale il nuovo punto viene definito;

**I due punti di bordo** (*edge points*), punti intermedi dei bordi che sono adiacenti a tale vertice nel poligono;

**Punto di faccia** (*face point*) media dei vertici del poligono;

Si può vedere quanto detto sopra nella figura seguente

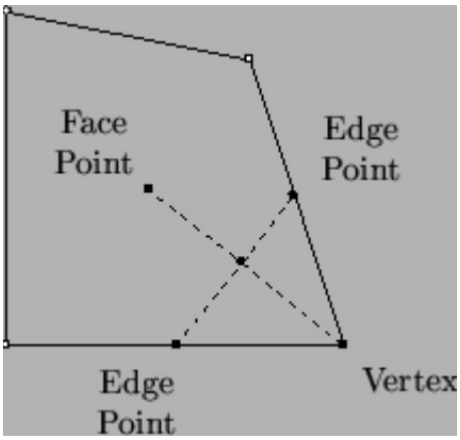



Figura 3.6: Definizione dei punti.

Il procedimento è molto semplice, basta generare questi nuovi punti su ogni faccia. Questi quattro nuovi punti vanno a formare una griglia rettangolare, la quale può essere utilizzata per ottenere altri nuovi punti, e così via iterando tale procedimento.

### Procedura per Mesh di topologia arbitraria

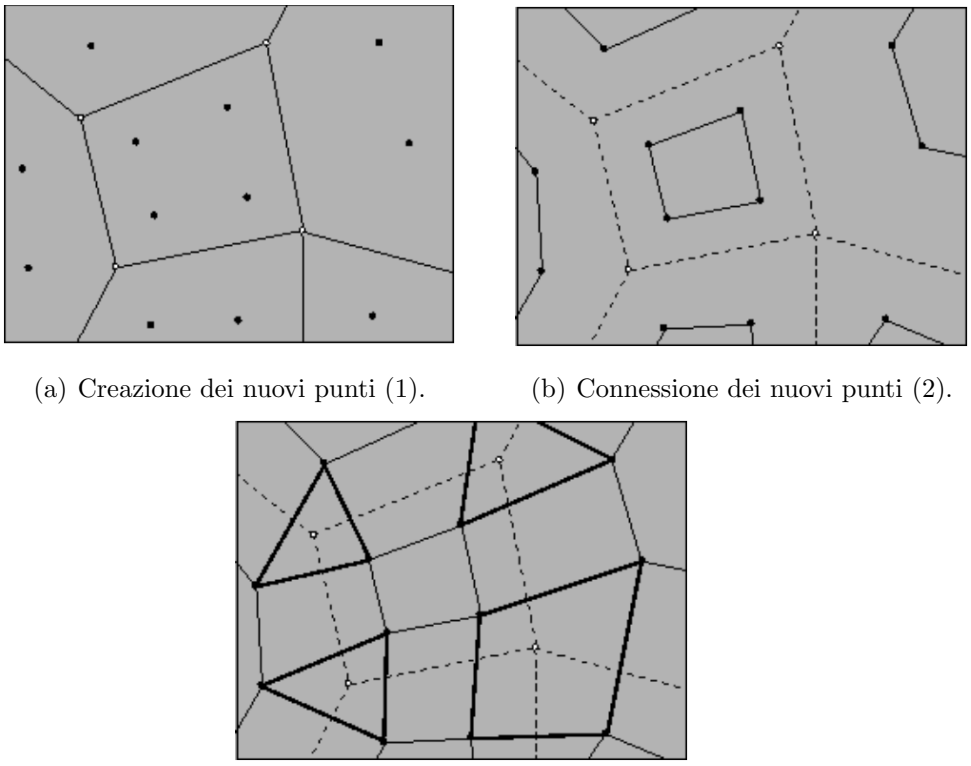
Questa descrizione dell'algoritmo può essere utilizzata per spiegare in modo semplice la procedura di raffinamento per superfici di topologia arbitraria. Le regole di raffinamento sono le seguenti:

1. Per ogni vertice $\mathbf{P}_i$ di ogni faccia della mesh, generare un nuovo punto $\mathbf{P}_i^1$ come media del vertice, dei due punti di bordo e del punto di faccia.

2. per ogni faccia, connettere tra loro i nuovi punti generati per ogni vertice della faccia.

3. per ogni vertice, connettere i nuovi punti generati per le facce che sono adiacenti al vertice in esame.

4. per ogni bordo, connettere i nuovi punti che sono stati generati per le facce che sono adiacenti al bordo in esame.



(a) Creazione dei nuovi punti (1). (b) Connessione dei nuovi punti (2).

(c) connessione di facce e bordi (3)(4)

Figura 3.7: Procedura per mesh di topologia arbitraria.

Per il caso di facce regolari (con 4 lati), un nuovo vertice può essere calcolato come

$$\mathbf{P}_0^{j+1} = \frac{9}{16}\mathbf{P}_0^j + \frac{3}{16}\mathbf{P}_1^j + \frac{1}{16}\mathbf{P}_2^j + \frac{3}{16}\mathbf{P}_3^j.$$

Mentre per il caso irregolare (facce con un numero di lati diverso da 4)

$$\mathbf{P}_0^{j+1} = \sum_{i=0}^{n-1} \alpha_i \mathbf{P}_i^j$$

dove $\alpha_i$ sono calcolati come

$$\alpha_i = \begin{cases} \frac{n+5}{4n} & \text{se } i = 0 \\ 3 + 2cos(2\pi i/n) & i = 1, 2, ..., n-1 \end{cases}$$

L'espressione per il caso regolare può essere ricavata dalla forma parametrica delle superfici B-spline bi-quadratiche uniformi.

Partiamo dall'equazione matriciale: consideriamo la superficie B-spline bi-quadratica uniforme $\mathbf{P}(u,v)$ definita da una matrice 3x3 di punti di controllo

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} \end{bmatrix}$$

e la seguente equazione (in forma matriciale)

$$\mathbf{P}(u,v) = \begin{bmatrix} 1 & u & u^2 \end{bmatrix} MPM^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

dove $M$ è la matrice 3x3

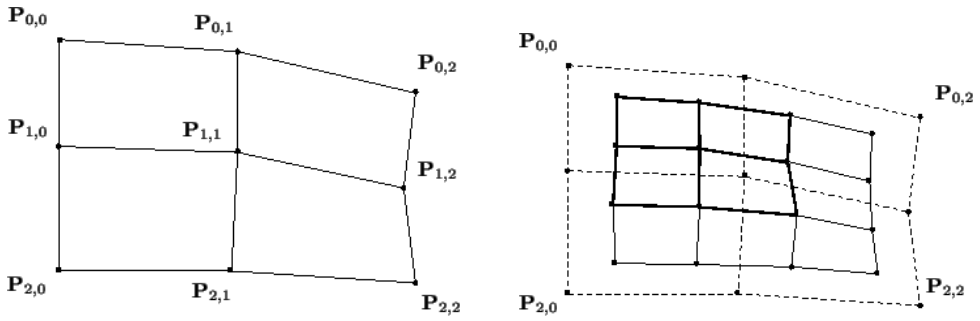
$$M = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

la matrice $M$ definisce la funzione base per la superficie B-spline bi-quadratica uniforme quando moltiplicata per

$$\begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

Prendiamo ora la patch rappresentata dalla matrice $\mathbf{P}$ ed andiamo a suddividerla in 4 sotto-patch. Ci concentriamo solo su una delle sotto-patch

(quella corrispondente a $0 \leq u, v \leq \frac{1}{2}$), mentre le altre seguiranno per simmetria. La figura seguente illustra i 16 punti ottenuti suddividendo in quattro sotto-patch. Va notato che i quattro punti di controllo "interni" sono utilizzati da ciascuna delle quattro componenti della suddivisione.




(a) Patch iniziale. (b) Patch dopo suddivisione.

Figura 3.8: Patch di una B-spline bi-quadratica.

per suddividere la superficie, consideriamo la riparametrizzazione della superficie $u^1 = \frac{u}{2}$ e $v^1 = \frac{v}{2}$ e definiamo questa nuova superficie $\mathbf{P}^1(u, v)$. Sostituiamo questo all'equazione e otteniamo

$$
\begin{aligned}
\mathbf{P}^1(u,v) &= \mathbf{P}(\frac{u}{2},\frac{v}{2}) \\
&= \begin{bmatrix} 1 & \frac{u}{2} & (\frac{u}{2})^2 \end{bmatrix} MPM^T \begin{bmatrix} 1 \\ \frac{v}{2} \\ (\frac{v}{2})^2 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} MPM^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 \end{bmatrix} MM^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} MPM^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}^T (M^{-1})^T M^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 \end{bmatrix} M \left( M^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} M \right) P \left( M^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix}^T (M^{-1})^T \right) M^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 \end{bmatrix} M \left( M^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} M \right) P \left( M^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} M \right)^T M^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 \end{bmatrix} MP^1M^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}
\end{aligned}
$$

dove $P^1 = SPS^T$ e

$$
S = M^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} M
$$

attraverso questo processo, abbiamo scritto la superficie $\mathbf{P}^1(u,v)$ come

$$\mathbf{P}^1(u,v) = \begin{bmatrix} 1 & u & u^2 \end{bmatrix} MSM^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

per alcune matrici di punti di controllo 3x3 $\boldsymbol{S}$. Questo implica che $P^1(u,v)$ è una patch B-spline bi-quadratica uniforme. La matrice $\boldsymbol{S}$ è tipicamente chiamata *splitting matrix* ed è data da

$$\begin{aligned} S &= \frac{1}{2}\begin{bmatrix} 2 & -1 & 0 \\ 2 & 1 & 0 \\ 2 & 3 & 4 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix} \\ &= \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix} \end{aligned}$$

quindi la mesh di punti di controllo $\mathbf{P}^1$ corrispondente alla path di suddivisione è legata alla mesh di punti di controllo originale dall'uguaglianza

$$\mathbf{P}^1 = SPS^T$$

Sostituendo alla formula appena citata abbiamo quindi

$$\begin{aligned} \mathbf{P}^1 &= \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} \end{bmatrix} \frac{1}{4}\begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix}^T \\ &= \frac{1}{16}\begin{bmatrix} 3\mathbf{P}_{0,0} + \mathbf{P}_{1,0} & 3\mathbf{P}_{0,1} + \mathbf{P}_{1,1} & 3\mathbf{P}_{0,2} + \mathbf{P}_{1,2} \\ \mathbf{P}_{0,0} + 3\mathbf{P}_{1,0} & \mathbf{P}_{0,1} + 3\mathbf{P}_{1,1} & \mathbf{P}_{0,2} + 3\mathbf{P}_{1,2} \\ 3\mathbf{P}_{1,0} + \mathbf{P}_{2,0} & 3\mathbf{P}_{1,1} + \mathbf{P}_{2,1} & 3\mathbf{P}_{1,2} + \mathbf{P}_{2,2} \end{bmatrix} \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 3 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \frac{1}{16}\begin{bmatrix} \mathbf{P}^1_{0,0} & \mathbf{P}^1_{0,1} & \mathbf{P}^1_{0,2} \\ \mathbf{P}^1_{1,0} & \mathbf{P}^1_{1,1} & \mathbf{P}^1_{1,2} \\ \mathbf{P}^1_{2,0} & \mathbf{P}^1_{2,1} & \mathbf{P}^1_{2,2} \end{bmatrix} \end{aligned}$$

dove i $\mathbf{P}^1_{i,j}$ possono essere scritti così

$$\begin{aligned}
\mathbf{P}^1_{0,0} &= \frac{1}{16}(3(3\mathbf{P}_{0,0} + \mathbf{P}_{1,0}) + (3\mathbf{P}_{0,1} + \mathbf{P}_{1,1}));\\
\mathbf{P}^1_{0,1} &= \frac{1}{16}((3\mathbf{P}_{0,0} + \mathbf{P}_{1,0}) + 3(3\mathbf{P}_{0,1} + \mathbf{P}_{1,1}));\\
\mathbf{P}^1_{0,2} &= \frac{1}{16}(3(3\mathbf{P}_{0,1} + \mathbf{P}_{1,1}) + (3\mathbf{P}_{0,2} + \mathbf{P}_{1,2}));\\
\mathbf{P}^1_{1,0} &= \frac{1}{16}(3(\mathbf{P}_{0,0} + 3\mathbf{P}_{1,0}) + (\mathbf{P}_{0,1} + 3\mathbf{P}_{1,1}));\\
\mathbf{P}^1_{1,1} &= \frac{1}{16}((\mathbf{P}_{0,0} + 3\mathbf{P}_{1,0}) + 3(\mathbf{P}_{0,1} + 3\mathbf{P}_{1,1}));\\
\mathbf{P}^1_{1,2} &= \frac{1}{16}(3(\mathbf{P}_{0,1} + 3\mathbf{P}_{1,1}) + (\mathbf{P}_{0,2} + 3\mathbf{P}_{1,2}));\\
\mathbf{P}^1_{2,0} &= \frac{1}{16}(3(3\mathbf{P}_{1,0} + \mathbf{P}_{2,0}) + (3\mathbf{P}_{1,1} + \mathbf{P}_{2,1}));\\
\mathbf{P}^1_{2,1} &= \frac{1}{16}((3\mathbf{P}_{1,0} + \mathbf{P}_{2,0}) + 3(3\mathbf{P}_{1,1} + \mathbf{P}_{2,1}));\\
\mathbf{P}^1_{2,2} &= \frac{1}{16}(3(3\mathbf{P}_{1,1} + \mathbf{P}_{2,1}) + (3\mathbf{P}_{1,2} + \mathbf{P}_{2,2})).
\end{aligned}$$

Ognuno di questi punti $\mathbf{P}_{i,j}$ utilizza i quattro punti su una certa faccia della maglia rettangolare, e calcola un nuovo punto pesando i quattro punti. Così, questo algoritmo può essere specificato utilizzando maschere di suddivisione, che definiscono i rapporti dei punti su una faccia per generare i nuovi punti.
Notiamo infine che la regola trovata per il calcolo di $\mathbf{P}^1_{2,2}$ è la stessa che utilizza l'algoritmo di Doo-Sabin per le facce regolari.

### 3.4.2 Schema di Catmull-Clark

Ed Catmull e Jim Clark, mentre erano ancora studenti presso l'università dello Utah, cercarono di estendere l'algoritmo di Doo e Sabin alle superfici B-spline bi-cubiche uniformi. Dal momento che il metodo di Doo-Sabin si basa sulla suddivisione delle patch B-spline bi-quadratiche uniformi, Catmull e Clark credevano che lo studio nel caso bi-cubico portasse ad un miglior schema di generazione di superfici di suddivisione.
Utilizzando la suddivisione delle superfici B-spline bi-cubiche uniformi, Cat-

mull e Clark, seguendo la metodologia di Doo e Sabin, osservano che le regole di suddivisione espresse per le Superfici B-spline cubiche non funzionano solo per maglie rettangolari arbitrarie, ma possono essere estese anche alle maglie di una topologia arbitraria. Questa estensione è stata compiuta generalizzando la definizione di un *face point*, modificando il metodo di calcolo dei *vertex point* (che si estende a quella del caso di B-spline uniformi), e descrivendo un metodo per riconnettere i punti in una mesh.

### Procedura di raffinamento

Data una mesh di punti di controllo con topologia arbitraria, possiamo generalizzare le specifiche dei face point, edge point e vertex point, dai calcoli delle superfici B-spline uniformi ottenendo la seguente procedura:

- per ogni faccia della mesh, generare i nuovi *face point* (che sono la media di tutti i punti originali che definiscono la faccia)
- generare i nuovi *edge point* (che sono calcolati come la media dei punti medi dell'*edge* di origine con i due nuovi *face point* delle facce adiacenti all'*edge*)
- calcolo dei nuovi *vertex point* (che sono calcolati come la media di $\mathbf{Q}$, $2\mathbf{R}$ e $\frac{(n-3)S}{n}$, dove $\mathbf{Q}$ è la media dei nuovi face point di tutte le facce adiacenti al *face point* originale, $\mathbf{R}$ è la media dei punti mediani di tutti gli *edge* originali incidenti al *vertex point* originale ed $\mathbf{S}$ è il *vertex point* originale.

La mesh viene riconessa dal seguente metodo

- ogni nuovo *face point* viene conesso ai nuovi *edge point* degli *edge* che definiscono la faccia originale.
- ogni nuovo *vertex point* viene connesso ai nuovi *edge point* di tutti gli *edge* originali incidenti nel *vertex point* originale.

**Esempi**

Come esempio di questo processo, consideriamo una mesh che consiste di quattro triangoli a formare un *pattern a diamante*, come in figura 3.9 *(a)*

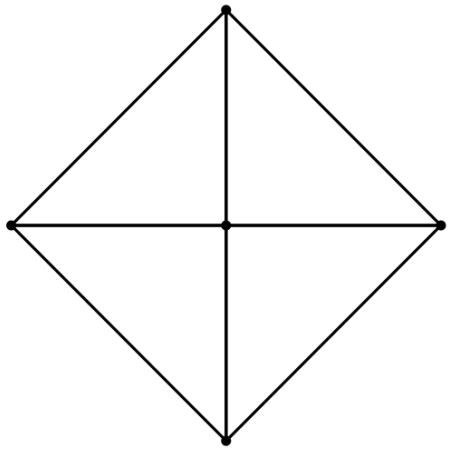
- Per prima cosa, costruiamo i face point, calcolati come la media dei punti che compongono ciascuna delle facce originali. Questi punti sono raffigurati in figura 3.9 *(b)* chiamati **F**

- Ora costruiamo gli *edge point*, calcolati come media dei quattro punti (i due punti originali che definiscono l'*edge*, e i due nuovi *face point* per le facce adiacenti all'*edge*). Nella figura 3.9 *(c)* sono riportati questi nuovi punti chiamati **E**

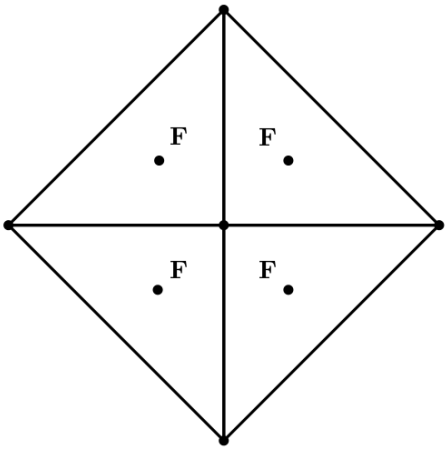
- Costruiamo ora il singolo vertex point. Questo punto, almeno nelle due dimensioni, coincide con il centro del diamante. Il punto **V** è mostrato in figura 3.9 *(d)*

- Infine, connettiamo gli edge ai punti che abbiamo generato: prima connettiamo i face point agli edge point corrispondenti ai bordi sulla faccia, e poi connettiamo i vertex point agli edge point (figura 3.9 *(e)*)

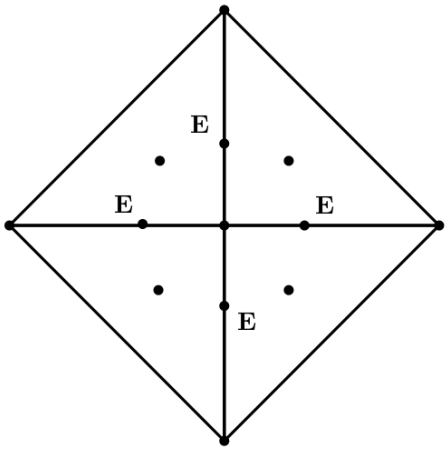
Notiamo che ciascuna faccia della mesh raffinata ha quattro bordi (edge), infatti, questo è vero in tutti casi, non importa quanti lati aveva la mesh di partenza.



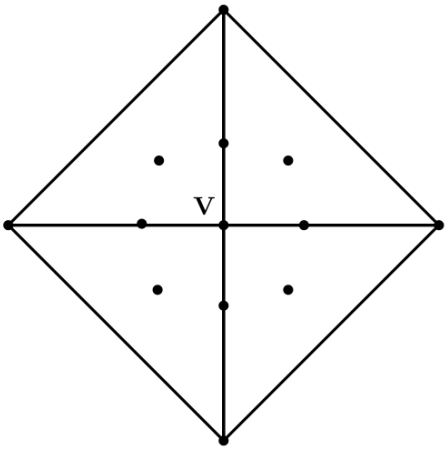
(a) mesh a diamante.



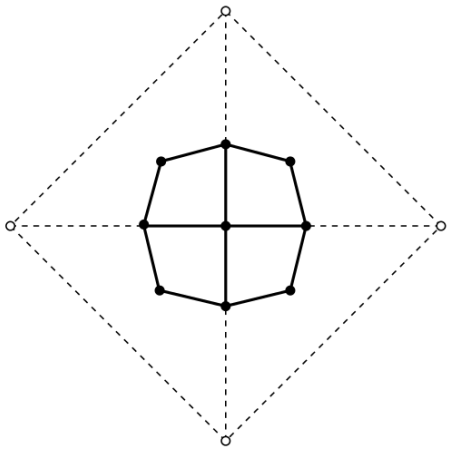

(b) costruzione dei nuovi face point.




(c) costruzione dei nuovi edge point.



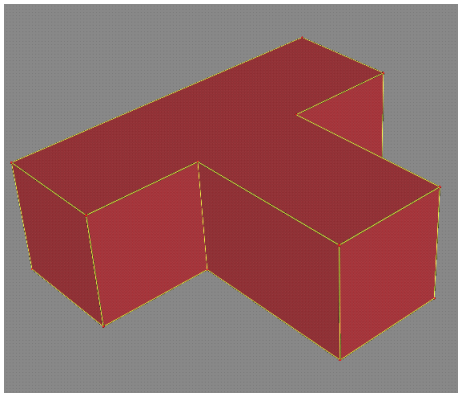

(d) costruzione del nuovo vertex point.



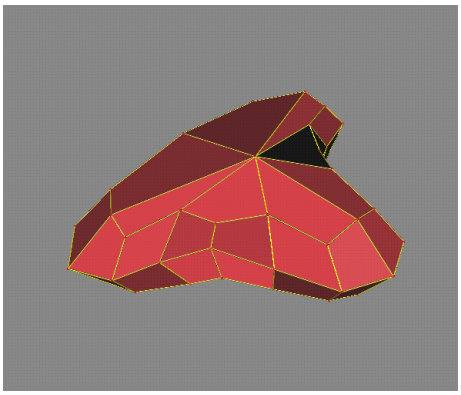
(e) risultato del processo.

Figura 3.9: Esempio di un processo di raffinamento.

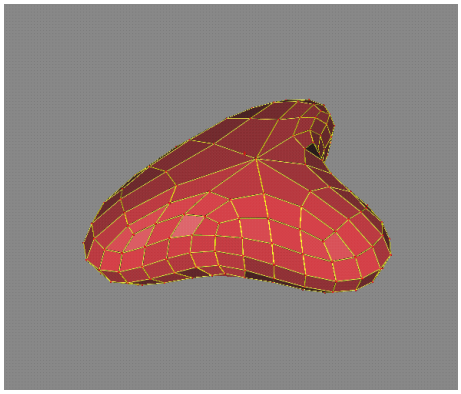
A seguire (figura 3.10) vengono mostrati tre passi del processo di raffinamento Catmull-Clark.
Dopo il primo passo di raffinamento, notiamo che la nuova serie di punti di controllo ha la proprietà che tutte le facce hanno quattro lati. È bene notare anche che i vertici corrispondenti ai punti di controllo originali mantengono la valenza (il numero di edge adiacenti al vertice). Al secondo passo, come al terzo, notiamo ancora che i vertici corrispondenti ai punti di controllo originali mantengono la loro valenza. Notare il vertice di valenza otto nella parte superiore del solido.
In qualsiasi porzione della superficie, una matrice 4x4 di punti di controllo di topologia rettangolare rappresenta una patch di superficie B-spline bi-cubica uniforme. Procedendo con la suddivisione, l'unico punto dove una topologia tale non potrà verificarsi è ai vertici che rappresentano i punti di controllo originali, la cui valenza non è 4 (siamo in presenta di vertici straordinari). Se tutti i vertici avvessero avuto valenza 4, allora saremmo stati in presenza di una superficie B-spline bi-cubica uniforme.
Catmull e Clark hanno dimostrato che le regole che hanno espresso per suddivisione cubica B-spline non funzionano solo per maglie rettangolari arbitrarie, ma possono anche essere estese a maglie di topologia arbitraria.



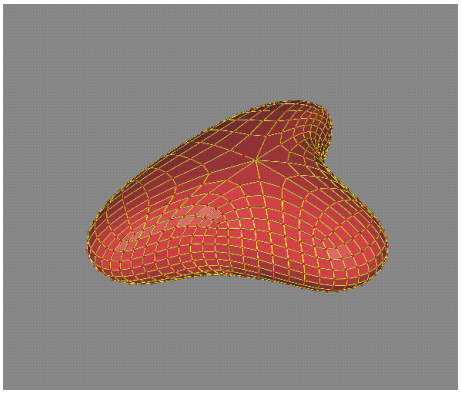
(a) mesh di partenza.



(b) Primo passo di raffinamento.



(c) Secondo passo di raffinamento.



(d) Terzo passo di raffinamento.

Figura 3.10: Esempio di tre passi di raffinamento Catmull-Clark di un solido.

Per spiegare, in termini matematici, come calcolare i nuovi punti attraverso l'algoritmo di Catmull-Clark ripartiamo dal ragionamento già fatto per le superfici Doo-Sabin (B-spline bi-quadratiche uniformi) ed applichiamolo al caso di superfici B-spline bi-cubiche.

Consideriamo la superficie B-spline Bi-cubica uniforme $\mathbf{P}(u, v)$ definita dalla matrice 4x4 di punti di controllo

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix}$$

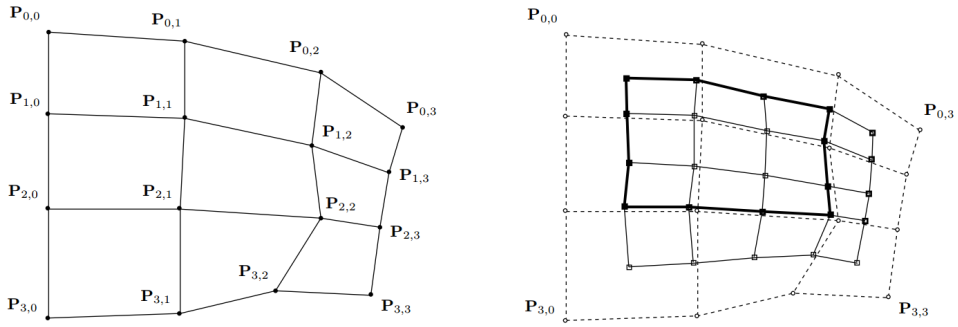
e la seguente equazione (in forma matriciale)

$$\mathbf{P}(u,v) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} MPM^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

dove $M$ è la matrice 4x4

$$M = \frac{1}{6} \begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

Come fatto anche per il caso di B-spline quadratiche, andiamo a prendere anche in questo caso la patch rappresentata dalla matrice $\mathbf{P}$ e suddividiamola in quattro sotto-patch. Ci concentriamo sempre solo su una delle sotto-pach ($0 \leq u, v \leq \frac{1}{2}$), mentre le altre seguiranno per simmetria.
La figura seguente illustra i 25 punti ottenuti suddividendo in quattro sotto-patch, in grassetto viene evidenziata la sotto-pach iniziale che andremo a considerare. Va notato anche in questo caso, che i nove punti interni sono utilizzati da ciascuna delle quattro componenti della suddivisione.




(a) Patch iniziale. (b) Patch dopo suddivisione.

Figura 3.11: Patch di una B-spline bi-cubica.

Generiamo questa sotto-pach riparametrizzando la superficie con le variabili $u^1$ e $v^1$ dove $u^1 = \frac{u}{2}$ e $v^1 = \frac{v}{2}$. Sostituendo queste all'equazione otteniamo

$$
\begin{aligned}
\mathbf{P}(u^1, v^1) &= \mathbf{P}(\frac{u}{2}, \frac{v}{2}) \\
&= \begin{bmatrix} 1 & \frac{u}{2} & (\frac{u}{2})^2 & (\frac{u}{2})^3 \end{bmatrix} MPM^T \begin{bmatrix} 1 \\ \frac{v}{2} \\ (\frac{v}{2})^2 \\ (\frac{v}{2})^3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} MPM^T \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix}^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} MM^{-1} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} MPM^T \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix}^T (M^{-1})^T M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} MSPS^T M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} MP^1 M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}
\end{aligned}
$$

dove $P^1 = SPS^T$ e

$$S = M^{-1} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} M$$

attraverso questo processo, abbiamo scritto la superficie $\mathbf{P}^1(u, v)$ come

$$\mathbf{P}^1(u, v) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} M P^1 M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

per alcune matrici di punti di controllo 4x4 $P^1$. Questo implica che $P^1(u, v)$ è una patch B-spline bi-cubica uniforme. La matrice $\boldsymbol{S}$ è tipicamente chiamata *splitting matrix*. Facendo i calcoli ed applicando la formula sopra citata otteniamo

$$S = \frac{1}{8} \begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \end{bmatrix}$$

Sostituendo quindi alla formula di $\mathbf{P}^1$ otteniamo

$$\mathbf{P}^1 = \frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} \frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \end{bmatrix}^T$$

$$= \frac{1}{8}\begin{bmatrix} 4\mathbf{P}_{0,0} + 4\mathbf{P}_{1,0} & 4\mathbf{P}_{0,1} + 4\mathbf{P}_{1,1} & 4\mathbf{P}_{0,2} + 4\mathbf{P}_{1,2} & 4\mathbf{P}_{0,3} + 4\mathbf{P}_{1,3} \\ \mathbf{P}_{0,0} + 6\mathbf{P}_{1,0} + 3\mathbf{P}_{2,0} & \mathbf{P}_{0,1} + 6\mathbf{P}_{1,1} + \mathbf{P}_{2,1} & \mathbf{P}_{0,2} + 6\mathbf{P}_{1,2} + \mathbf{P}_{2,2} & \mathbf{P}_{0,3} + 6\mathbf{P}_{1,3} + \mathbf{P}_{2,3} \\ 4\mathbf{P}_{1,0} + 4\mathbf{P}_{2,0} & 4\mathbf{P}_{1,1} + 4\mathbf{P}_{2,1} & 4\mathbf{P}_{1,2} + 4\mathbf{P}_{2,2} & 4\mathbf{P}_{1,3} + 4\mathbf{P}_{2,3} \\ \mathbf{P}_{1,0} + 6\mathbf{P}_{2,0} + 3\mathbf{P}_{3,0} & \mathbf{P}_{1,1} + 6\mathbf{P}_{2,1} + \mathbf{P}_{3,1} & \mathbf{P}_{1,2} + 6\mathbf{P}_{2,2} + \mathbf{P}_{3,2} & \mathbf{P}_{1,3} + 6\mathbf{P}_{2,3} + \mathbf{P}_{3,3} \end{bmatrix}$$

$$\frac{1}{8}\begin{bmatrix} 4 & 1 & 0 & 0 \\ 4 & 6 & 4 & 1 \\ 0 & 1 & 4 & 6 \\ 0 & 0 & 6 & 1 \end{bmatrix}$$

Eseguendo il calcolo matriciale otteniamo quindi i 16 nuovi punti.

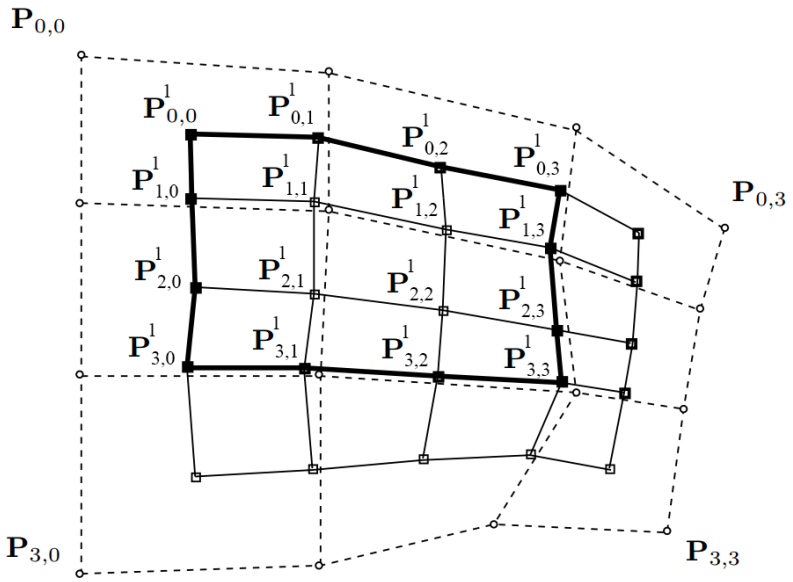



Figura 3.12: Patch di una B-spline bi-cubica dopo suddivisione.

$$\mathbf{P}^1_{0,0} = \frac{\mathbf{P}_{0,0} + \mathbf{P}_{1,0} + \mathbf{P}_{0,1} + \mathbf{P}_{1,1}}{4}$$
$$\mathbf{P}^1_{0,1} = \frac{\mathbf{P}_{0,0} + \mathbf{P}_{1,0} + 6(\mathbf{P}_{0,1} + \mathbf{P}_{1,1}) + \mathbf{P}_{0,2} + \mathbf{P}_{1,2}}{16}$$
$$\mathbf{P}^1_{0,2} = \frac{\mathbf{P}_{0,1} + \mathbf{P}_{1,1} + \mathbf{P}_{0,2} + \mathbf{P}_{1,2}}{4}$$
$$\mathbf{P}^1_{0,3} = \frac{\mathbf{P}_{0,1} + \mathbf{P}_{1,1} + 6(\mathbf{P}_{0,2} + \mathbf{P}_{1,2}) + \mathbf{P}_{0,3} + \mathbf{P}_{1,3}}{16}$$
$$\mathbf{P}^1_{1,0} = \frac{\mathbf{P}_{0,0} + \mathbf{P}_{0,1} + 6(\mathbf{P}_{1,0} + \mathbf{P}_{1,1}) + \mathbf{P}_{2,0} + \mathbf{P}_{2,1}}{16}$$
$$\mathbf{P}^1_{1,1} = \frac{\mathbf{P}_{0,0} + 6\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + 6(\mathbf{P}_{0,1} + 6\mathbf{P}_{1,1} + \mathbf{P}_{2,1}) + \mathbf{P}_{0,2} + 6\mathbf{P}_{1,2} + \mathbf{P}_{2,2}}{64}$$
$$\mathbf{P}^1_{1,2} = \frac{\mathbf{P}_{0,1} + \mathbf{P}_{0,2} + 6(\mathbf{P}_{1,1} + \mathbf{P}_{1,2}) + \mathbf{P}_{2,1} + \mathbf{P}_{2,2}}{16}$$
$$\mathbf{P}^1_{1,3} = \frac{\mathbf{P}_{0,1} + 6\mathbf{P}_{1,1} + \mathbf{P}_{2,1} + 6(\mathbf{P}_{0,2} + 6\mathbf{P}_{1,2} + \mathbf{P}_{2,2}) + \mathbf{P}_{0,3} + 6\mathbf{P}_{1,3} + \mathbf{P}_{2,3}}{64}$$
$$\mathbf{P}^1_{2,0} = \frac{\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + \mathbf{P}_{1,1} + \mathbf{P}_{2,1}}{4}$$
$$\mathbf{P}^1_{2,1} = \frac{\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + 6(\mathbf{P}_{1,1} + \mathbf{P}_{2,1}) + \mathbf{P}_{1,2} + \mathbf{P}_{2,2}}{16}$$
$$\mathbf{P}^1_{2,2} = \frac{\mathbf{P}_{1,1} + \mathbf{P}_{2,1} + \mathbf{P}_{1,2} + \mathbf{P}_{2,2}}{4}$$
$$\mathbf{P}^1_{2,3} = \frac{\mathbf{P}_{1,1} + \mathbf{P}_{2,1} + 6(\mathbf{P}_{1,2} + \mathbf{P}_{2,2}) + \mathbf{P}_{1,3} + \mathbf{P}_{2,3}}{16}$$
$$\mathbf{P}^1_{3,0} = \frac{\mathbf{P}_{1,0} + \mathbf{P}_{1,1} + 6(\mathbf{P}_{2,0} + \mathbf{P}_{2,1}) + \mathbf{P}_{3,0} + \mathbf{P}_{3,1}}{16}$$
$$\mathbf{P}^1_{3,1} = \frac{\mathbf{P}_{1,0} + 6\mathbf{P}_{2,0} + \mathbf{P}_{3,0} + 6(\mathbf{P}_{1,1} + 6\mathbf{P}_{2,1} + \mathbf{P}_{3,1}) + \mathbf{P}_{1,2} + 6\mathbf{P}_{2,2} + \mathbf{P}_{3,2}}{64}$$
$$\mathbf{P}^1_{3,2} = \frac{\mathbf{P}_{1,1} + \mathbf{P}_{1,2} + 6(\mathbf{P}_{2,1} + \mathbf{P}_{2,2}) + \mathbf{P}_{3,1} + \mathbf{P}_{3,2}}{16}$$
$$\mathbf{P}^1_{3,3} = \frac{\mathbf{P}_{1,1} + 6\mathbf{P}_{2,1} + \mathbf{P}_{3,1} + 6(\mathbf{P}_{1,2} + 6\mathbf{P}_{2,2} + \mathbf{P}_{3,2}) + \mathbf{P}_{1,3} + 6\mathbf{P}_{2,3} + \mathbf{P}_{3,3}}{64}$$

Facendo riferimento alla parte di patch evidenziata in figura 3.12, ognuno dei punti evidenziati può essere classificato in tre categorie: *punti di faccia* (face point), sono i punti della mesh raffinata che si trovano al centro dei rettangoli della mesh originale, *punti di lato o bordo* (edge point), sono i punti che giacciono sui lati che connettono due punti originali, e i *punti di*

*vertice* (vertex point), sono i punti che si trovano in prossimità dei vecchi punti di controllo.
I punti $\mathbf{P}^1_{0,0}$, $\mathbf{P}^1_{0,2}$, $\mathbf{P}^1_{2,0}$, $\mathbf{P}^1_{2,2}$ sono i punti di faccia. I punti di faccia vengono calcolati come la media dei vertici originali della faccia in questione:

$$\mathbf{P}^1_{0,0} = \frac{\mathbf{P}_{0,0} + \mathbf{P}_{1,0} + \mathbf{P}_{0,1} + \mathbf{P}_{1,1}}{4}$$

I punti di lato sono $\mathbf{P}^1_{0,1}$, $\mathbf{P}^1_{0,3}$, $\mathbf{P}^1_{1,0}$, $\mathbf{P}^1_{1,2}$, $\mathbf{P}^1_{2,1}$, $\mathbf{P}^1_{2,3}$, $\mathbf{P}^1_{3,0}$, $\mathbf{P}^1_{3,3}$. Un punto di lato (o bordo) è esprimibile come la media tra il punto centrale del lato originale e la media dei nuovi punti di faccia corrispondenti alle facce che condividono il lato. Ad esempio il punto $\mathbf{P}^1_{2,1}$ è calcolato come

$$\begin{aligned}
\mathbf{P}^1_{2,1} &= \frac{\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + 6(\mathbf{P}_{1,1} + \mathbf{P}_{2,1}) + \mathbf{P}_{1,2} + \mathbf{P}_{2,2}}{16} \\
&= \frac{\mathbf{P}_{1,0} + 6\mathbf{P}_{1,1} + \mathbf{P}_{1,2} + \mathbf{P}_{2,0} + 6\mathbf{P}_{2,1} + \mathbf{P}_{2,2}}{16} \\
&= \frac{\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + \mathbf{P}_{1,1} + \mathbf{P}_{2,1}}{16} + \frac{\mathbf{P}_{1,1} + \mathbf{P}_{1,2} + \mathbf{P}_{2,1} + \mathbf{P}_{2,2}}{16} + \frac{4\mathbf{P}_{1,1} + 4\mathbf{P}_{2,1}}{16} \\
&= \frac{\mathbf{P}^1_{2,0}}{4} + \frac{\mathbf{P}^1_{2,2}}{4} + \frac{4\mathbf{P}_{1,1} + 4\mathbf{P}_{2,1}}{16} \\
&= \frac{\frac{\mathbf{P}^1_{2,0} + \mathbf{P}^1_{2,2}}{2} + \frac{\mathbf{P}_{1,1} + \mathbf{P}_{2,1}}{2}}{2}
\end{aligned}$$

I punti rimanenti $\mathbf{P}^1_{1,1}$, $\mathbf{P}^1_{1,3}$, $\mathbf{P}^1_{3,1}$, $\mathbf{P}^1_{3,3}$ sono i punti di vertice. Questi punti sono un po più complessi da calcolare ma possiamo racchiudere il concetto nella formula

$$\mathbf{P}^1_{i,j} = \frac{Q + 2R + S}{4}$$

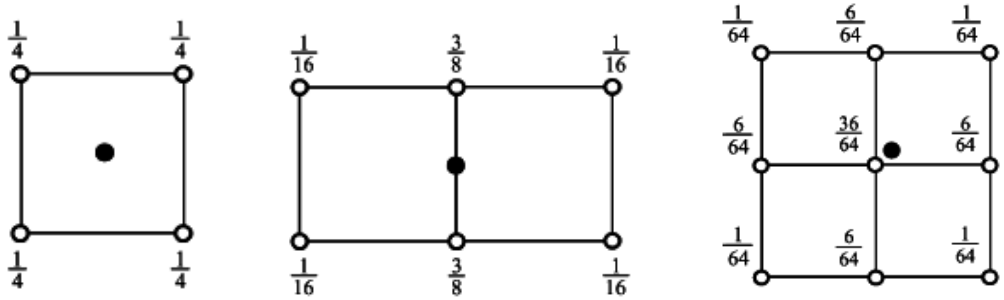
dove $Q$ è la media dei punti di faccia di tutte le facce adiacenti al vertice originale, $R$ è la media dei punti medi di tutti i lati adiacenti nel vecchio vertice ed $S$ è il punto di controllo stesso.

Ricaviamo questa espressione dal punto $\mathbf{P}^1_{1,1}$:

$$\begin{aligned}
\mathbf{P}^1_{1,1} &= \frac{\mathbf{P}_{0,0} + 6\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + 6(\mathbf{P}_{0,1} + 6\mathbf{P}_{1,1} + \mathbf{P}_{2,1}) + \mathbf{P}_{0,2} + 6\mathbf{P}_{1,2} + \mathbf{P}_{2,2}}{64} \\
&= \frac{\mathbf{P}_{0,0} + 6\mathbf{P}_{1,0} + \mathbf{P}_{2,0} + 6\mathbf{P}_{0,1} + 36\mathbf{P}_{1,1} + 6\mathbf{P}_{2,1} + \mathbf{P}_{0,2} + 6\mathbf{P}_{1,2} + \mathbf{P}_{2,2}}{64} \\
&= \frac{1}{4}\frac{\mathbf{P}^1_{0,0} + \mathbf{P}^1_{0,2} + \mathbf{P}^1_{2,0} + \mathbf{P}^1_{2,2}}{4} + \frac{1}{2}\frac{\frac{\mathbf{P}_{1,0}+\mathbf{P}_{2,2}}{2} + \frac{\mathbf{P}_{0,1}+\mathbf{P}_{2,2}}{2} + \frac{\mathbf{P}_{1,2}+\mathbf{P}_{2,2}}{2} + \frac{\mathbf{P}_{2,1}+\mathbf{P}_{2,2}}{2}}{4} + \frac{\mathbf{P}_{1,1}}{4} \\
&= \frac{Q}{4} + \frac{R}{2} + \frac{S}{4}
\end{aligned}$$

Ora tutti i sedici punti della suddivisione sono stati classificati in punti di faccia, di lato e di vertice ed il metodo geometrico per la suddivisione nel caso bi-cubico è concluso.

Nella figura 3.13 sono mostrate le maschere per la suddivisione Catmull-Clark. Si può far riferimento a queste per comprendere meglio il ragionamento proposto in questa ultima parte di capitolo.




(a) Maschera per un punto di faccia.

(b) Maschera per un punto di lato.

(c) Maschera per un punto di vertice.

Figura 3.13: Maschere per la suddivisione Catmull-Clark di mesh quadrilatere.

# Capitolo 4

# Superfici di suddivisione nell'animazione di personaggi

La creazione di personaggi accattivanti e credibili nella grafica computerizzata ha presentato una serie di sfide grafiche tra cui la modellazione, l'animazione ed il rendering di forme complesse come testa, mani ed indumenti. Inizialmente queste forme venivano modellate attraverso l'uso di superfici NURBS, nonostante le grandi restrizioni topologiche che queste superfici impongono. Al fine di superare queste restrizioni, agli inizi degli anni 80 vennero introdotte le superfici di suddivisione per la creazione di modelli e personaggi 3D.
Le Superfici di suddivisione della B-spline di Catmull-Clark furono sviluppate nel 1978 [6]. Furono utilizzate per la prima volta nell'ambito della Computer grafica 3D dalla Pixar nel film di animazione Geri's Game, del 1989.

In questo capitolo parleremo dell'utilizzo che viene fatto delle superfici di suddivisione nell'animazione facendo qualche riferimento su come la Pixar ha utilizzato questi strumenti per produrre lungometraggi di successo. Questo capitolo vuole essere conclusivo di quanto trattato in questa tesi, per questo motivo si è deciso di non soffermarsi troppo sugli aspetti matematici, ma vuole dare un'idea generale di come vengono affrontati vari problemi derivanti dall'uso delle superfici di suddivisione.

Figura 4.1: Immagine del corto animato "Geri's Game".

## 4.1 Difetti delle superfici NURBS

Il modo più comune per modellare superfici lisce complesse come quelle che si incontrano nell'animazione di un personaggio umano è quello di utilizzare un insieme di NURBS unite tra loro (trimming). Tuttavia queste superfici soffrono almeno di 2 problemi:

- il trimming è costoso e soggetto ad errori numerici.
- è difficile mantenere la regolarità, anche in fase di animazione.

Le superfici di suddivisione superano entrambi questi problemi: esse non richiedono fasi di trimming e la regolarità nelle forme è garantita automaticamente, anche in fase di animazione. L'uso di superfici di suddivisione ha posto nuove sfide nei processi produttivi. Nella modellazione, le superfici di suddivisione hanno permesso ai designer di liberarsi dalle preoccupazioni dei vincoli topologici tipici dell'utilizzo delle NURBS. Una singola superficie NURBS, come qualsiasi altra superficie parametrica, è limitata a rappresentare le superfici che sono topologicamente equivalenti ad un foglio, un cilindro o un torus. Questa è una limitazione fondamentale per qualsiasi superficie.

Una singola superficie di suddivisione, al contrario, può rappresentare superfici di topologia arbitraria. L'idea di base è di costruire una superficie da un poliedro qualsiasi, suddividendo ripetutamente ciascuna delle facce. Se la suddivisione viene eseguita in modo appropriato, il limite di questo processo di suddivisione sarà una superficie liscia. Una volta che i modelli sono stati costruiti con le superfici di suddivisione, i problemi di animazione sono generalmente più facili che con le superfici NURBS perché questi modelli sono senza cuciture (*seamless*), per cui la superficie rimane liscia (regolare) anche quando il modello è animato. Questi dettagli hanno dato la possibilità al settore dell'animazione 3D di fare notevoli passi in avanti, e verso la fine degli anni 80 hanno permesso alla Pixar di far uscire il loro primo cortometraggio "Geri's Game" in cui un simpatico vecchietto gioca a scacchi con se stesso. In particolare, le superfici di suddivisione sono state utilizzate per modellare la pelle della testa di Geri, le sue mani, i suoi vestiti, quali la giacca, i pantaloni, la camicia, la cravatta e le scarpe.

## 4.2 Supporto alla fisicità dei tessuti

La fisicità dei tessuti può essere interfacciata e relazionata con l'uso di modelli geometrici costruiti con superfici di suddivisione. Per le simulazioni fisiche, le proprietà base dei materiali sono generalmente specificate definendo un'*energia funzionale* che rappresenta l'attrazione o la resistenza di un determinato materiale alle possibili deformazioni. Tipicamente, l'energia è descritta come una superficie integrale, o come somma di termini discreti che sono funzioni della posizione della superficie di esempio o dei vertici di controllo. Il primo tipo di specifica tipicamente da luogo ad un approccio ad elementi finiti, mentre il secondo è associato a metodi con differenze finite. Approcci ad elementi finiti sono possibili con superfici di suddivisione. In generale comunque, superfici integrali ad elementi finiti devono essere stimate attraverso integrazione numerica e questo da luogo ad un insieme di casi particolari attorno ai punti straordinari. Per evitare queste casistiche,

la Pixar adottò un approccio a differenze finite, approssimando i tessuti con un sistema massa-molla [7], in cui tutta la massa è concentrata nei punti di controllo. Come abbiamo già detto, lontano dai punti straordinari, le facce generate dalla suddivisione Catmull-Clark diventano tutte facce quadrilatere. Questo fatto è ideale per rappresentare i tessuti che sono anche descritti a livello locale da una struttura reticolata. Costruendo le funzioni energetiche per la simulazione dei tessuti, facciamo corrispondere i bordi della mesh di suddivisione con la direzione della trama del tessuto simulato.

## 4.3 Gestione delle collisioni

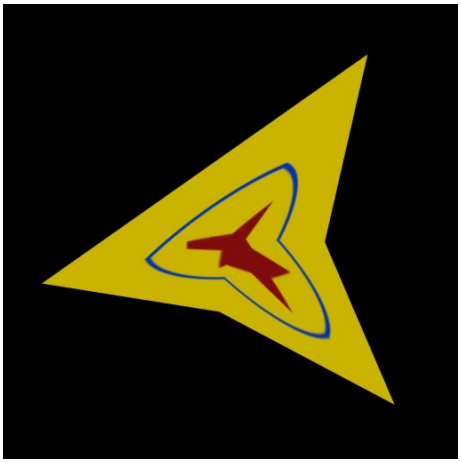
La gestione delle collisioni è una componente importante nella simulazione dinamica. L'approccio più semplice per determinare una collisione in una simulazione fisica, è quello di testare ogni elemento geometrico (punti, facce, lati) con ogni altro elemento geometrico con cui potrebbe generare collisione. Con $N$ elementi geometrici, il costo computazionale diverrebbe di $N^2$, che è proibitivo se gli elementi geometrici sono molto grandi. Per ottenere tempi pratici di esecuzione per grandi simulazioni, il numero di possibili collisioni deve essere abbattuto il più rapidamente possibile utilizzando qualche tipo di struttura dati spaziale (*spatial data structure*). Ci sono due categorie di strutture dati generalmente impiegate: Possiamo distribuire gli elementi in una struttura dati bidimensionale, oppure in una struttura dati tridimensionale. Usando una struttura dati bidimensionale si posso avere notevoli vantaggi se le superfici di connettività non cambiano. Avere superfici fisse significa non dover rigenerare la struttura ogni volta che una superficie si muove, allocare staticamente la struttura, e non dover mai ribilanciare l'albero di struttura.
Possiamo vedere la struttura dati come una struttura ad albero, e inserire gli elementi geometrici in questo modo: partiamo dai nodi foglia della gerarchia, ciascuno dei quali corrisponde ad una faccia della superficie di suddivisione, e costruiamo gli altri punti della gerarchia unendo le facce fino ad arrivare ad

un singolo punto in cui tutte le facce sono unite a formare l'intera superficie.

## 4.4 Rendering delle superfici di suddivisione

In questa sezione introduciamo il concetto di campi scalari variabili utilizzati su superfici di suddivisione, mostrando come questi vengano utilizzati per applicare texture parametriche alle superfici di suddivisione.
Le superfici NURBS sono strutturate in quattro metodi principali: mappatura parametrica della texture, colorazione 3D e texture solide. La colorazione 3D e le texture solide sono facili da applicare a qualsiasi tipo di primitiva, così queste tecniche sono già pronte per essere applicate alle superfici di suddivisione. È invece più difficile comprendere come il mapping parametrico possa essere applicato a queste superfici visto che, a differenza delle NURBS, esse non sono definite parametricamente. Il metodo attuale per texturizzare un modello poligonale è quello di assegnare delle coordinate della texture ad ogni vertice. Se le facce del poligono sono solo triangolari e quadrilatere, le coordinate della texture possono essere interpolate attraverso la faccia del poligono durante la scansione di conversione usando l'interpolazione lineare o bi-lineare. Le facce con più di 4 lati pongono una grande sfida. Un approccio può essere quello di dividere queste facce in un insieme di facce triangolari o quadrilatere. Una difficoltà con questo approccio è che le coordinate di texture non sono differenziabili su spigoli della mesh originale o su mesh pre-elaborate. Questa discontinuità può essere notata in figura 4.2 (a,b) specialmente quando il modello è animato.
Fortunatamente la situazione per le superfici di suddivisione è profondamente migliore rispetto ai modelli poligonali. Se le coordinate di texture assegnate ai vertici di controllo vengono suddivise usando le stesse regole di suddivisione che sono usate per le coordnate geometriche allora possiamo ottenere un texturing liscio (smooth). Quanto detto viene illustato in figura 4.2 (c,d), dove la superficie viene trattata come una superficie di Catmull-Clark (superficie limite).

(a) Texture mappata su un pentagono regolare composto da 5 triangoli.



(b) Il pentagono con i vertici spostati.

(c) Superficie di suddivisione dove la mesh di controllo è formata dagli stessi 5 triangoli di (a).

(d) La superficie di suddivione con i vertici posizionati come in (b).

Figura 4.2: Esempi di texture mapping.

# Conclusioni

L'uso delle superfici di suddivisione ha permesso ai modellatori di organizzare punti di controllo in un modo con cui è naturale poi acquisire le caratteristiche geometriche del modello, senza preoccuparsi di mantenere una struttura a griglia regolare richiesta dai modelli NURBS. Questa libertà ha due principali conseguenze. Innanzitutto, riduce drasticamente il tempo necessario per progettare e costruire un modello iniziale. In secondo luogo, e forse più importante, permette al modello iniziale di essere raffinato localmente. Le rifiniture locali non sono possibili con le superfici NURBS dato che un intera riga, punto di controllo, o entrambi devono essere aggiunti per preservare la struttura reticolata.
Il texturing delle superfici di suddivisione permette un maggior realismo nell'animazione di oggetti modellati tramite questa tecnica, un maggior controllo delle collisioni e della gestione dei tessuti. Al giorno d'oggi le superfici di suddivisione hanno un impiego molto rilevante nella grafica computerizzata. Questo importante strumento è stato potenziato nel corso degli anni ottenendo tempistiche computazionali sempre migliori.

# Bibliografia


[1] SALOMON David. *Curves and Surfaces for Computer Graphics*, Springer-Verlag, 2006.

[2] WU Ling, YONG Jun-Hai, ZHANG You-Wei, ZHANG Li. *Multi-step Subdivision Algorithm for Chaikin Curves*, CIS 2004, Springer-Verlag, 1232-1238, 2004.

[3] DOO, D. *A subdivision algorithm for smoothing down irregularly shaped polyhedrons. In Proced. Int'l Conf. Ineractive Techniques in Computer Aided Design*, IEEE Computer Soc ,157-165, 1978.

[4] DOO, D., AND SABIN, M. *Behaviour of recursive division surfaces near extraordinary points.*, Computer-Aided Design 10 (Sept. 1978), 356-360.

[5] CHAIKIN, G. *An algorithm for high speed curve generation.* Computer Graphics and Image Processing 3 (1974), 346-349.

[6] E. CATMULL and J. CLARK, *Recursively generated B-spline surfaces on arbitrary topological Meshes*, in Computer-Aided Design 10 (Sept. 1978).

[7] Andrew Witkin, David Baraff, and Michael Kass. *An intro- duction to physically based modeling.* SIGGRAPH Course Notes, Course No. 32, 1994.

[8] *Subdivision Surfaces in character animation* - De Rose, Kass, Truong - Siggraph 2008

[9] KOBBELT, L. *Interpolatory Subdivision on Open Quadrilateral Nets with Arbitrary Topology.* In Proceedings of Eurographics 96, Computer Graphics Forum, 409–420, 1996.