


國立臺灣大學機械工程學研究所碩士論文

指導教授：陸一平 博士



建立在分散式環境下之 3D CAD 幾何核心

研究生：許嘉豪 撰

中華民國八十九年六月

建立在分散式環境下之 3D CAD

幾何核心

許嘉豪¹ 陸一平²

國立台灣大學機械工程研究所

摘要

本論文主要是針對 CAD 軟體的幾何核心實作的方法，作整體性的探討。其中包括如何將幾何核心寫成軟體元件。

本論文概分為四個部分：一、曲線與曲面的探討與實作；二、交線與交面的探討的實作；三、曲面混成的探討；四、將幾何核心寫成軟體元件的方法。

¹ 研究生

² 指導教授

The Design of CAD Geometry Kernel Built Upon Distributed Environment

Chia-Hao Hsu¹ Yih-Ping Luh²

Department of Mechanical Engineering
Nation Taiwan University

Abstract

This thesis describes the method of how to realize the geometry kernel and it includes that how to make the geometry kernel to software components.

This thesis is composed of the four parts : 1.The search and realizing for curve and surface; 2.The search and realizing for intersection of curve and surface; 3.The search of surface blending; 4.The method of how to realize the geometry kernel in network.

¹Graduate Student

²Associate Professor

目錄

誌謝	錯誤! 尚未定義書籤。
中文摘要	II
英文摘要	III
目錄	IV
圖目錄	VII
第一章 導論	1
1-1 研究方向與動機.....	1
1-2 文獻回顧.....	2
1-3 章節摘要.....	3
第二章 幾何核心之構成要素	5
2-1 前言	5
2-2 處理功能.....	5
2-3 產生功能.....	7
第三章 曲線的理論簡介與實作	11
3-1 前言	11
3-2 參數曲線.....	11
3-3 NURBS 曲線與其實作	19
3-4 偏移曲線(Offset curves)的探討	41
第四章 曲面的理論簡介與實作	47

4-1 前言	47
4-2 參數曲面	47
4-3 NURBS 曲面	53
4-4 偏移曲面(Offset Surface).....	72
4-5 修剪曲面(Trimmed Surface).....	77
第五章 幾何交集之探討與實作	79
5-1 前言	79
5-2 線與線的交集計算	79
5-3 線與面的交集計算	82
5-4 面與面的交集計算	88
5-5 曲面交集的討論	99
第六章 曲面混成之探討	100
6-1 前言	100
6-2 求出偏移曲面的交線	101
6-3 建構與兩曲面相切的圓錐曲線	102
6-4 圓錐曲線的掃出	104
6-5 Corner blending	105
6-6 實例測試.....	112
6-7 討論.....	114
第七章 實作分散式幾何軟體元件	116
7-1 前言	116
7-2 分散式 CAD 之架構	117

7-3 幾何伺服器之架構	118
7-4 與其它模組之溝通	122
7-5 實作成果.....	123
第八章 回顧與展望	125

圖目錄

圖 2-1 Cones 的頂點.....	7
圖 2-2 平面上之曲線 Offset.....	8
圖 2-3 曲面之輪廓線示意圖.....	8
圖 2-4 曲面之 Offset.....	9
圖 3-1 Hermite curve 示意圖.....	12
圖 3-2 控制點數目為 3 的 Bézier 曲線.....	14
圖 3-3 控制點數目為 4 的 Bézier 曲線.....	14
圖 3-4 B-Spline 曲線示意圖.....	15
圖 3-5 以 NURBS 表示真圓.....	24
圖 3-6 插入 knots 前之 NURBS 曲線.....	25
圖 3-7 插入 knots 後之 NURBS 曲線.....	26
圖 3-8 degree = 2 的 Bézier 曲線.....	27
圖 3-9 以 NURBS 表示圓弧.....	28
圖 3-10 圓心角為 80° 之圓弧.....	29
圖 3-11 圓心角為 140° 之圓弧.....	30
圖 3-12 圓心角為 240° 之圓弧.....	31
圖 3-13 圓心角為 300° 之圓弧.....	32
圖 3-14 所求得之內插曲線之控制點.....	35

圖 3-15 求取近似曲線之流程圖	36
圖 3-16 近似曲線之一	40
圖 3-17 近似曲線之二	40
圖 3-18 近似曲線之三	41
圖 3-19 法向量非唯一	43
圖 3-20 產生自我相交	43
圖 3-21 迴圈或銳角	43
圖 3-22 某一線段消失	44
圖 3-23 需要移除線段	44
圖 3-24 需要增加曲線	44
圖 3-25 利用偏移控制多邊形來求得偏移曲線	45
圖 4-1 參數曲面示意圖	48
圖 4-2 Bézier 曲面示意圖	49
圖 4-3 三角形曲面之邊界資料	51
圖 4-4 建構一個 Gregory Patch	52
圖 4-5 Cylindrical Surface	58
圖 4-6 兩條階數與控制點數目不同的 NURBS 曲線	60
圖 4-7 圖所形成的 Ruled Surface	60
圖 4-8 角度為 90° 之 Revolved Surface	62
圖 4-9 Revolved Surface 所形成之環(torus)	62
圖 4-10 四條相接之曲線示意圖	63
圖 4-11 Coon 曲面形成示意圖	64

圖 4-12 四條相接之曲線.....	65
圖 4-13 Coon 曲面.....	66
圖 4-14 數條相鄰之曲線.....	67
圖 4-15 形成 <i>Skin Surface</i> 的第一步驟.....	68
圖 4-16 形成 <i>Skin Surface</i> 的第二步驟，將其 <i>knot vector</i> 及階數成相同.....	68
圖 4-17 形成 <i>Skin Surface</i> 的第三步驟，內插出控制點....	69
圖 4-18 <i>Skin Surface</i>	69
圖 4-19 路徑曲線上之 <i>local</i> 座標系	71
圖 4-20 對每一個 <i>local coordinate</i> 所形成的截面曲線	72
圖 4-21 <i>Offset Surface 1</i>	74
圖 4-22 <i>Offset Surface 2</i>	74
圖 4-23 <i>Trimmed domain using cubic NURBS curve</i>	78
圖 5-1 兩曲線之最短距離.....	79
圖 5-2 兩曲線交點相鄰極近.....	81
圖 5-3 兩曲線相切.....	81
圖 5-4 曲線與平面交點示意圖.....	82
圖 5-5 將曲面分割為三角形.....	85
圖 5-6 三角形向量與直線向量.....	85
圖 5-7 曲線與平面交於多個點.....	87
圖 5-8 直線與環狀曲面相切於兩點.....	87
圖 5-9 曲面交集之演算法.....	88

圖 5-10 封閉交線的起始點	90
圖 5-11 平面與曲面交線追蹤方向	91
圖 5-12 利用關鍵點之等 u 線求得封閉交線之起始點	93
圖 5-13 平面與曲面交出四條曲線	96
圖 5-14 平面與曲面交出兩條曲線	97
圖 5-15 平面與曲面交出一條封閉曲線	97
圖 5-16 兩圓柱面交出兩條曲線	98
圖 5-17 圓柱面與環面交出四條曲線	98
圖 6-1 “rolling-ball”求得 <i>blending surface</i> 示意圖	101
圖 6-2 偏移曲面交線	103
圖 6-3 求得 <i>blending surface</i> 截面之圓錐曲線	103
圖 6-4 <i>blending surface</i> 示意圖	104
圖 6-5 在角落附近之交集點的 <i>Domain</i> 值	106
圖 6-6 求得 <i>DIC-points</i> 流程圖	107
圖 6-7 <i>Corner blending</i> 之邊界曲線	109
圖 6-8 <i>Domains of n-sided Corner Blending</i>	111
圖 6-9 平面與傾斜之圓柱所構成之 <i>blending surface</i>	112
圖 6-10 兩圓柱所構成之 <i>blending surface 1</i>	112
圖 6-11 兩圓柱所構成之 <i>blending surface 2</i>	113
圖 6-12 圓環與曲面所構成之 <i>blending surface</i>	113
圖 6-13 使用 “setback” 技術進行 <i>corner blending</i>	115
圖 7-1 分散式 CAD 之架構圖	117

圖 7-2 幾何伺服器架構圖	119
圖 7-3 類別架構圖	120
圖 7-4 拓撲伺服器呼叫幾何伺服器之流程	123
圖 7-5 CAD 瀏覽器界面圖	124

第一章 導論

1-1 研究方向與動機

在幾年前，3D 的 CAD 軟體只能在中高階的電腦或工作站上才能有效的運作；直到最近，電腦的運算能力愈來愈強大，所以有愈來愈多的 CAD 軟體可在 PC 上運作，但仍然只限於零件或少量的組件，若稍微複雜的組件，如汽車、飛機等大型組件，在速度上的表現仍然不佳，所以筆者思考，能否以分散式的運算，來加快計算的速度，故筆者設計了一個幾何核心，來探討分散式的可能性。

再加上現在網際網路的發展一日千里，無遠弗界，Web 應用程式的發展也日漸蓬勃，所以 Web 也應該可以應用在 CAD 軟體上，透過瀏覽器即可以作電腦輔助設計，若是連線到速度較快的伺服器主機，可以增加 CAD 軟體的效能與速度，並可能進一步達到多人同時設計一個零件的目的。

基本上，一個 CAD 軟體大致上可分為幾何(Geometry)、拓樸(Topology)兩大核心，若是以特徵為基礎的 CAD(也就是所謂的 Feature base)，則必須加上特徵核心，本論文主要是在討論 CAD 軟體的幾何核心，包括若要實作一個幾何核心必須包含那些部分，並探討發展成分散式幾何核心的架構。

1-2 文獻回顧

在 CAD 的領域裡，幾何核心是最早開始發展的，在 1960 年中期開始迅速發展，S. A. Coons(1963)在麻省理工學院(MIT)，而 J. C. Ferguson 在波音(Boeing)開始專注在自由曲面的發展，同一時期，General Motor 發展了他們的 DAC-1 系統，而其它的公司，如 Douglas、Lockheed 和 McDonnell 等也開始發展他們的 CAD 系統。

在 CAD 系統中的幾何核心最早開始發展的地點在法國以及美國，在法國，P. de Casteljau 在 1959 年開始專注在曲線及曲面的設計，當時他為雪鐵龍汽車工作，之後，P. Bézier 在雷諾汽車發展了 Unisurf 系統；在 60 年代，J. Ferguson 在波音公司將三次雲線加入了設計的系統，且此時 S. Coons 也在麻省理工學院發展他自己的曲面；在 1974 年猶他州的一個研討會上，正式將這個領域稱為”電腦輔助幾何設計(Computer-Aided Geometric Design)”。

到了 70 年代，由於 W. Gordon 和 R. Riesenfeld 的研發，使得 B-spline 的曲線及曲面開始加入 CAD 的幾何核心中，而 B-spline 的理論基礎主要來自於 Mansfield、de Boor 和 Cox 等人，而之前 Bézier 所提出的方法也被視為 B-spline 的一種特殊情形，此時 B-spline 已經對整個 CAD 領域發生了很大的衝擊，以 B-spline 為基礎的其它理論也陸續產生，如三角形的曲面塊(triangular patches)和曲面分割(subdivision surfaces)等。

近幾年來，由於 NURBS 曲面是一個非常好用的曲面模型，所以以 NURBS 為基礎的 CAD/CAM 系統，也開始發展，到了 1983 年，NURBS 開始加入 IGES(initial graphics exchange specification)規格中。

1-3 章節摘要

本論本總共分為七個章節，各個章節的相關內容簡述如下：

第一章 導論

包括研究方向與動機，簡介本論文的主題與目標，另有文獻回顧的探討，與各章摘要。

第二章 幾何核心之構成要素

簡述若要實作一個幾何核心，所需實作的功能。

第三章 曲線的理論簡介與實作

此章節介紹曲線的種類與各種曲線的特性，並探討如何以程式實作。

第四章 曲面的理論簡介與實作

此章節介紹曲面的種類與各種曲面的特性，並探討如何以程式實作。

第五章 幾何交集計算之探討與實作

探討在幾何核心的實作上，所會用到的幾何交集的計算方法，包括線與線、線與面、面與面的交集等。

第六章 曲面混成之探討

探討曲面混成的方法。

第七章 實作分散式幾何軟體元件

吾人所實作出的幾何核心，是以動態函式庫(dll)的方式呈現，在本章節提出如何將其包裝成 DCOM(Distributed Component Object Model)的形式，以達到將來 CAD 軟體可分散式運算的目的。

第八章回顧與展望

針對 CAD 軟體的發展方向，提出筆者的看法。

第二章 幾何核心之構成要素

2-1 前言

一個 CAD 的幾何主要為點、線、面，針對於點，幾何核心的工作不多，主要是實作曲線和曲面的功能，而這些功能主要可分為處理（manipulation）功能，和產生（creation）功能，所謂處理功能，是指針對一個已經產生的曲線或曲面，對其作操作，例如求得曲線或曲面的座標，求得一個點在某曲面上的投影等；而產生功能則是指產生出一個新的曲線或曲面，如曲面交線，曲面偏移（offset）等。

2-2 處理功能

曲線處理功能

1. Evaluation：求得曲線上某參數的座標值或 n 次微分值。
2. Inverse point：給定某座標值，求得曲線上的參數。
3. Subdivision：將曲線分成好幾個曲線。
4. Extension：將曲線兩端延伸，通常是沿著兩端點之切線方向。
5. Length：求得曲線之總長度。

6. Reparameter：將曲線參數之 Domain 重新定義。
7. Detect selfintersect：偵測是否有自我相交的情形。
8. Constraint-Based Curve Manipulation：在某些限制條件下，如限制曲線上某一點的位置或法向量或切線向量等，對這個曲線作修改，詳細可參考文獻 Barry[3]。

曲面處理功能

1. Evaluation：求得曲面上某參數的座標值或 n 次微分
值，即 $S(u,v)$ 或 $\frac{\partial^{n+m} S(u,v)}{\partial u^n \partial v^m}$
2. Inverse point：給定某座標值，求得曲面上的 u,v 參數。
3. Subdivision：將曲面分成好幾個曲面。
4. Extension：將曲面延伸，通常是沿著切線方向。
5. Area：求得曲面之總面積。
6. Reparameter：將曲面參數之 Domain 重新定義。
7. Detect selfintersect：偵測是否有自我相交的情形。
8. IsoParameterCurve：等參數曲線，如等 u 線或等 v 線。
9. Detect degenerate：偵測曲面是否有退化的部分，如某一個點的法向量無法求出，如圖 2-1 之 Cone 的頂點

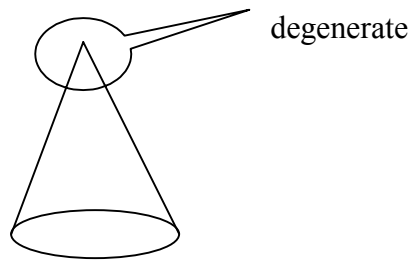


圖 2-1 Cones 的頂點

2-3 產生功能

曲線及曲面的產生方式有很多種，但其中應用最多的應該是 Interpolation 及 fitting，很多種產生方式都是以此作為基礎所產生出來的。

曲線產生功能

1. Interpolation：給定一些曲線的資料點，如座標及微分值，內插出符合資料點的曲線。
2. Curve fitting：類似 Interpolation，但並不是準確地通過資料點，而是以近似的方法求得曲線。
3. Offset：通常是對位於平面上的曲線，求得其在此平面上對每一點皆位移一固定距離的另一條曲線。

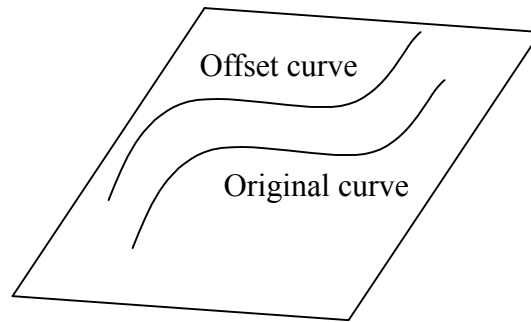


圖 2-2 平面上之曲線 Offset

4. Surface Surface Intersection：兩個曲面的交集所形成的曲線。
5. Silhouette curve：一個曲面由於視角不同所形成的輪廓線。

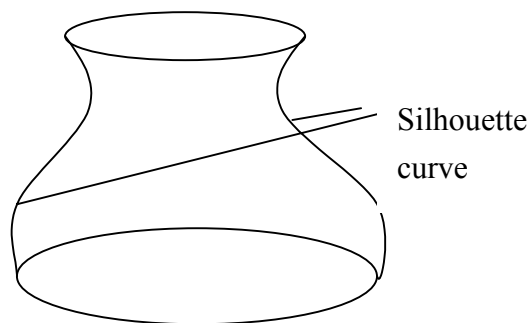


圖 2-3 曲面之輪廓線示意圖

曲面產生功能

1. Interpolation：給定一些曲面的資料點，如座標及微分值，內插出符合資料點的曲面。
2. Surface fitting：類似 Interpolation，但並不是準確地通過資料點，而是以近似的方法求得曲面。

3. Offset：可分為兩種 offset，一是以曲面的法向量為方向 offset，一是以某一點為放射中心的 offset。

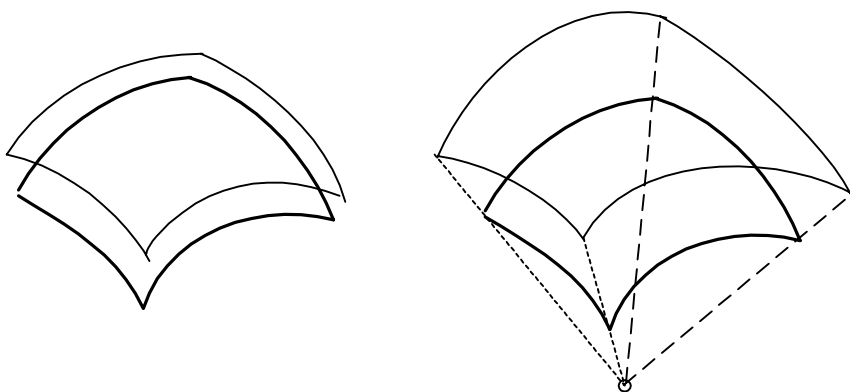


圖 2-4 曲面之 Offset

4. Rule Surface：給定兩條任意曲線，將兩條曲線之間等參數的點以直線連接所形成的曲面。
5. Revolution Surface：一條任意曲線繞一固定軸旋轉某角度，所形成的曲面。
6. Skin Surface：給定 N 條曲線，產生一個以此 N 條曲線為等參數曲線的曲面。
7. Swept Surface：一條任意曲線沿著一個路徑曲線所掃出的曲面。
8. n -Side patches：具有 n 個邊的曲線。
9. Curve net Surface：數條互相交錯的網狀曲線所 interpolation 出來的曲面。
10. Blending Surface：使兩曲面連接部分圓滑化的曲面。

11. Smoothing of free-form surfaces：將原本非平滑的曲面轉成平滑的曲面，可參考文獻 Ma and Peng[25]。

第三章曲線的理論簡介與實作

3-1 前言

一條曲線可被定義為一個點在只有一個自由度下的運動軌跡，在幾何核心中，表示曲線的方式可分為顯性方程式(explicit equation)、隱性方程式(implicit equation)以及參數方程式(parameter equation)。而其中應用最為廣泛也最為人所探討的是參數方程式，它又可分為 Hermite curve、Bézier curve、B-Spline curve 和 NURBS(Nonuniform Rational B-Spline) curve。

3-2 參數曲線

3-2-1 Hermite curve

首先介紹 Hermite curve，又稱 Ferguson curve，它是由一條曲線兩端點的邊界條件所定義而成，即端點的位置與切線向量所定義的。

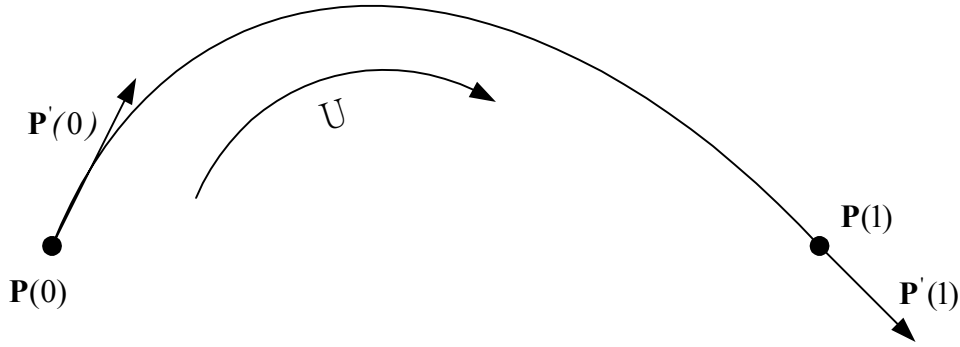


圖 3-1 Hermite curve 示意圖

若以參數方程式的形式可表示如下：

$$\mathbf{P}(u) = \sum_{i=0}^3 \mathbf{a}_i u^i = \mathbf{a}_3 u^3 + \mathbf{a}_2 u^2 + \mathbf{a}_1 u + \mathbf{a}_0 \quad (\text{Eq.3.1})$$

$\mathbf{P}(u)$ 代表在這條曲線上任何一點的位置向量，而 \mathbf{a} ， \mathbf{b} ， \mathbf{c} ， \mathbf{d} 則表示這條參數方程式的向量係數，它可由端點的邊界條件來求得。若我們定義參數 u 的上界為 0，下界為 1，則兩端點的位置向量可表示為 $\mathbf{P}(0)$ 與 $\mathbf{P}(1)$ ，切線向量可表示為 $\mathbf{P}'(0)$ 與 $\mathbf{P}'(1)$ ，如此一來，吾人可求得 \mathbf{a} ， \mathbf{b} ， \mathbf{c} ， \mathbf{d} 四個向量參數如下：

$$\begin{aligned} \mathbf{a}_3 &= 2\mathbf{P}(0) - 2\mathbf{P}(1) + \mathbf{P}'(0) + \mathbf{P}'(1) \\ \mathbf{a}_2 &= -3\mathbf{P}(0) + 3\mathbf{P}(1) - 2\mathbf{P}'(0) - \mathbf{P}'(1) \\ \mathbf{a}_1 &= \mathbf{P}'(0) \\ \mathbf{a}_0 &= \mathbf{P}(0) \end{aligned}$$

將上面四個參數代換到(Eq.3.1)之後，可得到以下形式的 Hermite 曲線方程式：

$$\begin{aligned} \mathbf{P}(u) &= (2u^3 - 3u^2 + 1)\mathbf{P}(0) + (-2u^3 + 3u^2)\mathbf{P}(1) \\ &\quad + (u^3 - 2u^2 + u)\mathbf{P}'(0) + (u^3 - u^2)\mathbf{P}'(1) \end{aligned} \quad (\text{Eq.3.2})$$

若以矩陣的形式表示，則如下所示：

$$\mathbf{P}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}(0) \\ \mathbf{P}(1) \\ \mathbf{P}'(0) \\ \mathbf{P}'(1) \end{bmatrix} \quad (\text{Eq.3.3})$$

3-2-2 Bézier 曲線

接著介紹 Bézier 曲線，Bézier 曲線的參數方程式表示如下：

$$\mathbf{P}(u) = \sum_{i=0}^n \mathbf{P}_i B_{i,n}(u) \quad u \in [0,1] \quad (\text{Eq.3.4})$$

其中 \mathbf{P}_i 代表 Bézier 曲線的特徵多邊形(characteristic polygon)的 $n+1$ 個頂點，這些頂點稱為控制點(control points)，而 $B_{i,n}(u)$ 稱為 Bernstein 多項式

$$B_{i,n}(u) = \binom{n}{i} u^i (1-u)^{n-i} \quad (\text{Eq.3.5})$$

由上式可看出 Bézier 曲線的參數方程式與 n 有關，即與 control points 的數目有關，當有三個控制點時， $n=2$ 時，

$$\mathbf{P}(u) = (1-u^2)\mathbf{P}_0 + 2u(1-u)\mathbf{P}_1 + u^2\mathbf{P}_2 \quad (\text{Eq.3.6})$$

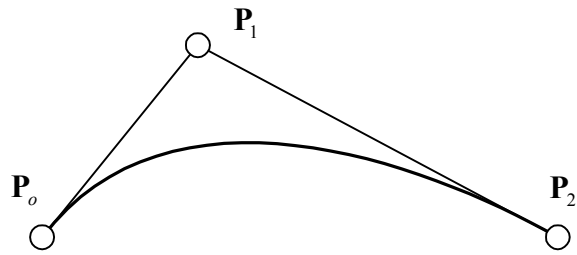


圖 3-2 控制點數目為 3 的 Bézier 曲線

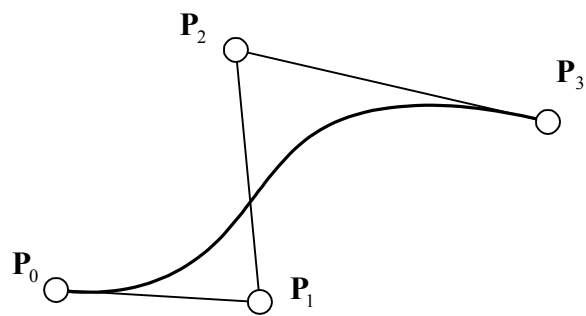


圖 3-3 控制點數目為 4 的 Bézier 曲線

當有四個控制點， $n=3$ 時，

$$\mathbf{P}(u) = (1-u)^3 \mathbf{P}_0 + 3u(1-u)^2 \mathbf{P}_1 + 3u^2(1-u) \mathbf{P}_2 + u^3 \mathbf{P}_3 \quad (\text{Eq.3.7})$$

3-2-3 B-Spline 曲線

接著我們介紹 B-Spline 曲線，它是由好幾個線段所組成，而每一個線段則是由少數幾個控制點所控制，如圖 3-4，Segment 1 是由 \mathbf{P}_0 ， \mathbf{P}_1 ， \mathbf{P}_2 三個控制點所控制，Segment 2 是由 \mathbf{P}_1 ， \mathbf{P}_2 ， \mathbf{P}_3 所控制，依此類推。

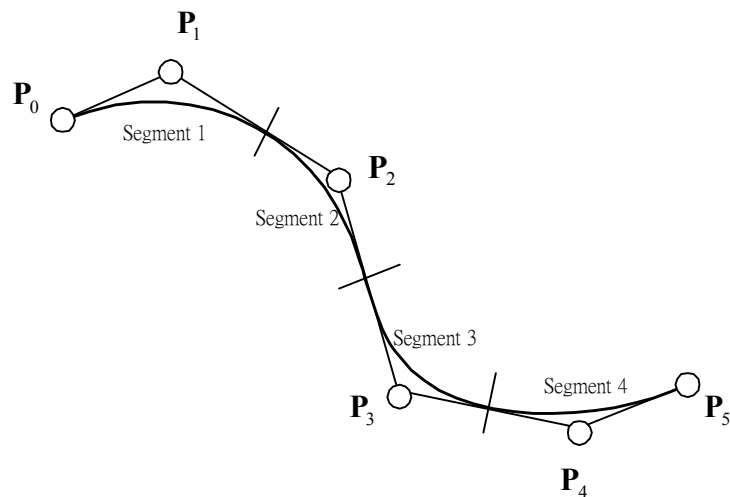


圖 3-4 B-Spline 曲線示意圖

由於 B-Spline 每一個線段只由少數幾個控制點所控制，所以若改變其中一個控制的位置，只會導致少數線段的改變，即局部的改變，這就是它與 Bézier 最大的不同點。

B-Spline 的參數方程式表示如下：

$$\mathbf{P}(u) = \sum_{i=0}^n \mathbf{P}_i N_{i,k}(u) \quad (\text{Eq.3.8})$$

其中 k 代表這條 B-Spline 曲線的階數，它與 Bezier 的參數方程式所不同的只有基底函數(basis function) $N_{i,k}(u)$ ，B-Spline 的基底函數定義如下：

$$\begin{aligned} N_{i,1}(u) &= 1 \quad \text{if} \quad t_i \leq u < t_{i+1} \\ &= 0 \quad \text{otherwise} \end{aligned} \quad (\text{Eq.3.9})$$

$$N_{i,k}(u) = \frac{(u - t_i)N_{i,k-1}(u)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - u)N_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}}$$

上式中的 t_i 稱做 knot value， t_i 的集合稱為 knot vector，這是 B-Spline 與 Bezier 曲線的另一個不同點，就是 B-Spline 的組成要素多了 knot vector。

舉例來說，若有一條 B-Spline 曲線，k=3，即表示三階，或次數(degree)=2，或稱為 quadratic B-Spline 曲線。它有六個控制點，knot vector 為 [0 0 0 1 2 3 4 4 4]

，則可依次求得下列各值：

$$\begin{aligned}
N_{0,1}(u) &= 1 \quad \text{if } 0 \leq u < 1 \\
&= 0 \quad \text{otherwise} \\
N_{1,1}(u) &= 1 \quad \text{if } 1 \leq u < 2 \\
&= 0 \quad \text{otherwise} \\
N_{2,1}(u) &= 1 \quad \text{if } 2 \leq u < 3 \\
&= 0 \quad \text{otherwise} \\
N_{3,1}(u) &= 1 \quad \text{if } 3 \leq u < 4 \\
&= 0 \quad \text{otherwise} \\
N_{4,1}(u) &= 1 \quad \text{if } 4 \leq u < 5 \\
&= 0 \quad \text{otherwise} \\
N_{5,1}(u) &= 1 \quad \text{if } 5 \leq u < 6 \\
&= 0 \quad \text{otherwise}
\end{aligned}$$

$$\begin{aligned}
N_{0,2}(u) &= 0 \\
N_{1,2}(u) &= (1-u)N_{2,1}(u) \\
N_{2,2}(u) &= uN_{2,1}(u) + (2-u)N_{3,1}(u) \\
N_{3,2}(u) &= (u-1)N_{3,1}(u) + (3-u)N_{4,1}(u) \\
N_{4,2}(u) &= (u-2)N_{4,1}(u) + (4-u)N_{5,1}(u) \\
N_{5,2}(u) &= (u-3)N_{5,1}(u) \\
N_{0,3}(u) &= (1-u)^2 N_{2,1}(u) \\
N_{1,3}(u) &= 0.5u(4-3u)N_{2,1}(u) + 0.5(2-u)^2 N_{3,1}(u) \\
N_{2,3}(u) &= 0.5u^2 N_{2,1}(u) + 0.5(-2u^2 + 6u - 3)N_{3,1}(u) + 0.5(3-u)^2 N_{4,1}(u) \\
N_{3,3}(u) &= 0.5(u-1)^2 N_{3,1}(u) + 0.5(-2u^2 + 10u - 11)N_{4,1}(u) + 0.5(4-u)^2 N_{5,1}(u) \\
N_{4,3}(u) &= 0.5(u-2)^2 N_{4,1}(u) + 0.5(-3u^2 + 20u - 32)N_{5,1}(u) \\
N_{5,3}(u) &= (u-3)^2 N_{5,1}(u)
\end{aligned}$$

故可得此條 B-Spline 曲線的四條線段的參數方程式分別如下：

$$\mathbf{r}_1(u) = (1-u)^2 \mathbf{P}_0 + 0.5u(4-3u)\mathbf{P}_1 + 0.5u^2\mathbf{P}_2$$

$$\mathbf{r}_2(u) = 0.5(2-u)^2 \mathbf{P}_1 + 0.5(-2u^2 + 6u - 3)\mathbf{P}_2 + 0.5(u-1)^2 \mathbf{P}_3$$

$$\mathbf{r}_3(u) = 0.5(3-u)^2 \mathbf{P}_2 + 0.5(-2u^2 + 10u - 11)\mathbf{P}_3 + 0.5(u-2)^2 \mathbf{P}_4$$

$$\mathbf{r}_4(u) = 0.5(4-u)^2 \mathbf{P}_3 + 0.5(-3u^2 + 20u - 32)\mathbf{P}_4 + (u-3)^2 \mathbf{P}_5$$

3-3 NURBS 曲線與其實作

目前來說，在 CAD 核心中最為廣泛使用的，莫過於 nonuniform rational B-Spline(NURBS)曲線，它的優點在於可正確地表示圓錐曲線，而且比 B-Spline 曲線有更好的 local control 的特性，它的參數方程式可表示如下：

$$\mathbf{P}(u) = \frac{\sum_{i=0}^n w_i \mathbf{P}_i N_{i,k}(u)}{\sum_{i=0}^n w_i N_{i,k}(u)} \quad (\text{Eq.3.10})$$

其中 w_i 代表每一個控制點的權重(weight)，它的基底函數 $N_{i,k}(u)$ 與 B-Spline 是相同的。所以若每一個控制點的 weight 都等於 1，則 NURBS 曲線與 B-Spline 是相等的，也就是說，NURBS 曲線較 B-Spline 曲線多了 weight 這個自由度來控制曲線。

在筆者的幾何核心中，NURBS 曲線扮演了重要的角色，而它需要探討的地方也很多，首先我們來看看，NURBS 曲線的組成要素有那些；若以 C 的結構來看，以下的結構可表示一條 NURBS 曲線：

```
////////////////////////////////////  
typedef struct curve  
{  
    int n;  
    int p;  
    double *knt;  
    double *Pw;  
} CURVE;  
////////////////////////////////////
```

n：控制點的數目-1

p：knots 的數目-1

knt：knot vector 所組成的陣列

Pw：control points 所組成的陣列，它是以 homogeneous coordinates 的方式呈現，即是用 $[w_x \ w_y \ w_z \ w]$ 來表示一個 control point 以及它的 weight。

而 NURBS 所要實作的內容依次有基底函數，曲線位置與其 n 次微分，曲線分解(subdivision)，內插(interpolation)曲線，近似(Approximation)曲線及求得點在線上的參數(inverse-point)等，另外還有插入 knots 和升階(degree elevate)等。

3-3-1 基底函數

有很多種方法來定義 B-Spline 曲線的基底函數，吾人採用的是 deBoor，Cox 和 Mansfield 等人所採用的 recurrence formula，它的定義如之前所提到，由公式可看出， $N_{i,p}(u)$ 是兩個(p-1)-degree 基底函數的組合，即 $N_{i,p-1}(u)$ 和 $N_{i+1,p-1}(u)$ 的線性組合，故若吾人想求得 pth-degree 的基底函數，會產生下面的表：

$$\begin{array}{ccccccc}
N_{i,0} & & & & & & \\
& N_{i,1} & & & & & \\
N_{i+1,0} & & N_{i,2} & & & & \\
& N_{i+1,1} & M & & N_{i,p} & & \\
N_{i+2,0} & M & & N_{i+p-2,2} & & & \\
M & & N_{i+p-1,1} & & & & \\
N_{i+p,0} & & & & & &
\end{array}$$

表中的第一列為步階函數(step function)，如之前所定義，所以筆者只需以迴圈依序將第一列之後的元素循序求出，即可得到單一基底函數。

基底函數的一次微分可經由證明求得如下之公式：

$$N'_{i,p} = \frac{p}{u_{i+p} - u_i} N_{i,p-1}(u) - \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \quad (\text{Eq.3.11})$$

若我們以 $N_{i,p}^{(k)}(u)$ 來表示 $N_{i,p}(u)$ 則我們可得下面公式：

$$N_{i,p}^{(k)}(u) = \frac{p}{u_{i+p} - u_i} N_{i,p-1}^{(k-1)}(u) - \frac{p}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}^{(k-1)}(u) \quad (\text{Eq.3.12})$$

詳細的證明方法可參閱 Piegl [22]。

3-3-2 曲線位置

在開始求曲線位置之前，必須先將(Eq.3.10)改寫成

$$\mathbf{Pw}(u) = \sum_{i=0}^n \mathbf{Pw}_i N_{i,k}(u) \quad (\text{Eq.3.13})$$

，其中 \mathbf{Pw}_i 為控制點的 homogenous 形式，如此一來便與 B-Spline 曲線的公式相同。而曲線的位置

$$\mathbf{P}(u) = \frac{\mathbf{Pw}(u)}{w(u)} \quad (\text{Eq.3.14})$$

NURBS 曲線的參數方程式是屬於片段(piecewise)的，所以要求得曲線上某一點的位置必須先知道它是那一個線段，亦即必須有一個 FindSpan 的函式，然後我們就可以得知與這個 Span 有關係的控制點，便可將上式簡化成：

$$\mathbf{Pw}(u) = \sum_{i=\text{span}-p}^{\text{span}} \mathbf{Pw}_i N_{i,k}(u) \quad (\text{Eq.3.15})$$

如此一來就可求位置向量的 homogenous 座標 $[wx \quad wy \quad wz \quad w]$ 。

3-3-3 曲線的 n 次微分

曲線的微分不如曲線的位置這麼單純，以下是 NURBS 曲線的 n 次微分的推導：

$$\mathbf{P}(u) = \frac{w(u)\mathbf{P}(u)}{w(u)} = \frac{\mathbf{A}(u)}{w(u)} \quad (\text{Eq.3.16})$$

其中 $\mathbf{A}(u)$ 為 $\mathbf{Pw}(u)$ 向量中前三個元素所組成的向量。

$$\begin{aligned}
 \mathbf{P}'(u) &= \frac{w(u)\mathbf{A}'(u) - w'(u)\mathbf{A}(u)}{w(u)^2} \\
 &= \frac{w(u)\mathbf{A}'(u) - w'(u)w(u)\mathbf{P}(u)}{w(u)^2} \\
 &= \frac{\mathbf{A}'(u) - w'(u)\mathbf{P}(u)}{w(u)}
 \end{aligned} \tag{Eq.3.17}$$

$\mathbf{A}'(u)$ 可經由下式求得：

$$\begin{aligned}
 \mathbf{Pw}'(u) &= \sum_{i=0}^{n-1} N_{i,p-1}(u)\mathbf{Q}_i \\
 \mathbf{Q}_i &= p \frac{\mathbf{Pw}_{i+1} - \mathbf{Pw}_i}{u_{i+p-1} - u_{i+1}}
 \end{aligned} \tag{Eq.3.18}$$

以上是求得一次微分的方法，而求得 n 次微分的方法同理可推導出如下：

$$\mathbf{P}^{(k)}(u) = \frac{\mathbf{A}^{(k)}(u) - \sum_{i=1}^k \binom{k}{i} w^{(i)}(u) \mathbf{P}^{(k-i)}(u)}{w(u)} \tag{Eq.3.19}$$

$\mathbf{A}^{(k)}(u)$ 可由下式求得：

$$\begin{aligned}
 \mathbf{Pw}^{(k)}(u) &= \sum_{i=0}^{n-k} N_{i,p-k}(u)\mathbf{Q}_i^{(k)} \\
 \mathbf{Q}_i^{(k)} &= \begin{cases} \mathbf{Q}_i & k=0 \\ \frac{p-k+1}{u_{i+p+1} - u_{i+k}} (\mathbf{Q}_{i+1}^{(k-1)} - \mathbf{Q}_i^{(k-1)}) & k>0 \end{cases}
 \end{aligned} \tag{Eq.3.20}$$

3-3-4 曲線分解(subdivision)

曲線分解就是將一條 NURBS 曲線依據所給定的參數值，將其分解成數條 NURBS 曲線，根據 You [9]，一條 NURBS 曲線若 multiple knots 的數目大於等於 degree，則這條曲線將會通過其中一個控制點，如下圖，吾人利用 NURBS 曲線來表示一個真圓，由於它有一些 multiple knots 所以會有控制點通過曲線上。

degree = 2

knot vector = [0 0 0 0.25 0.25 0.5 0.5 0.75 0.75 1 1 1]

number of control points = 8

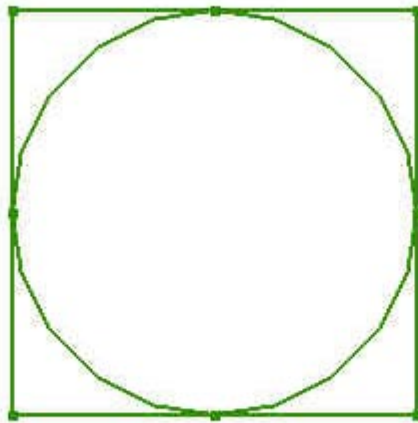


圖 3-5 以 NURBS 表示真圓

所以若我們要將曲線分解，則需要對曲線作插入 knots 的動

作，若需要在 $u = u_i$ 的地方將曲線分解，我們可以插入 knots p 次($p=\text{degree}$)。

下面舉一個例子：

插入 *knots* 前

degree = 3

knot vector = [0 0 0 0 0.6 1 1 1 1]

number of control points = 5

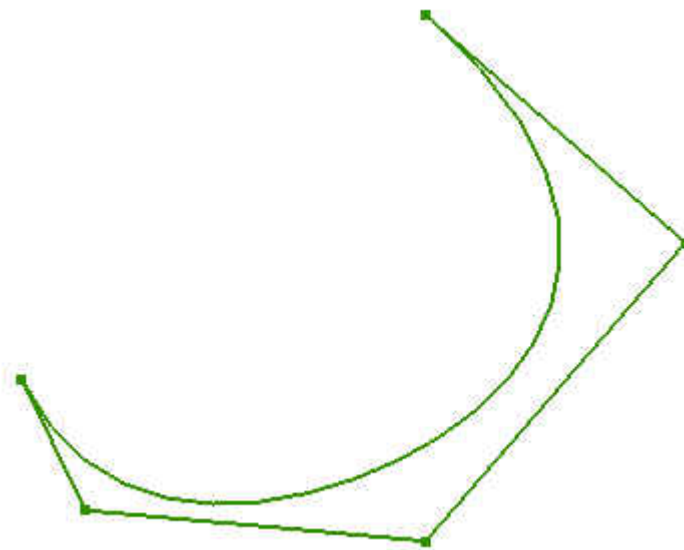


圖 3-6 插入 knots 前之 NURBS 曲線

插入 *knots* 0.3 3 次

degree = 3

knot vector = [0 0 0 0 0.3 0.3 0.3 0.6 1 1 1 1]

number of control points = 8

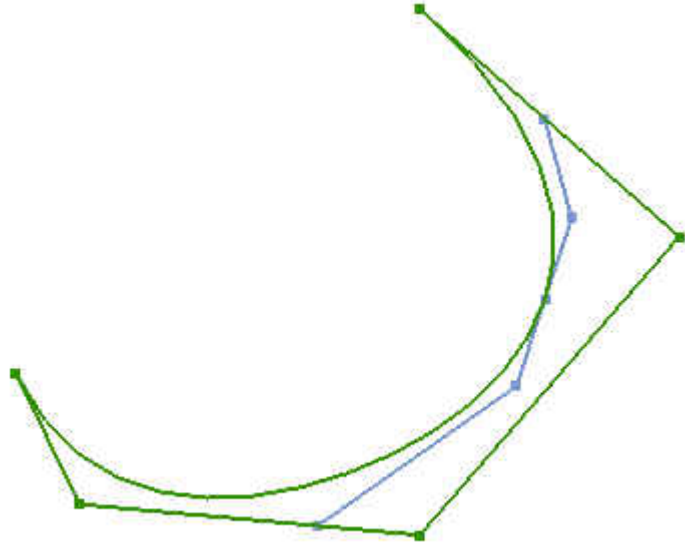


圖 3-7 插入 knots 後之 NURBS 曲線

如此一來，這條就被分成兩條，若需要前半部，可以取前半部的控制點加上前半部的 knots，可形成一條新的 NURBS 曲線。

3-3-5 以 NURBS 曲線表示圓

圓或圓弧是在 CAD 系統中，使用頻率僅次於直線的元素，由於吾人的幾何核心是以 NURBS 為基礎，所以常常要以 NURBS 曲線來表示圓或圓弧。

一個 degree=2 的 rational Bézier 的曲線可以表示如下：

$$C(u) = \frac{(1-u)^2 w_0 P_0 + 2u(1-u)w_1 P_1 + u^2 w_2 P_2}{(1-u)^2 w_0 + 2u(1-u)w_1 + u^2 w_2} \quad (\text{Eq.3.21})$$

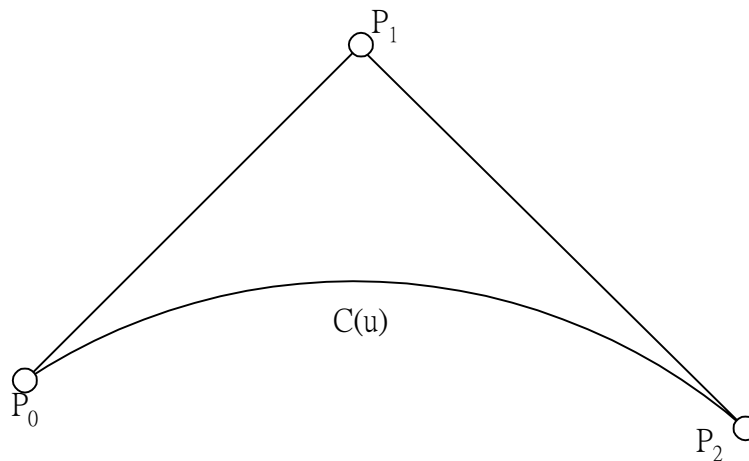


圖 3-8 degree = 2 的 Bézier 曲線

(Eq.3.21)可經由證明得知， $k = \frac{w_0 w_2}{w_1^2}$ ， k 為圓錐曲線形狀係數(conic shape factor)，

- if $k > 1$ ， $C(u)$ 為橢圓形。
- if $k = 1$ ， $C(u)$ 為拋物線。
- if $k < 1$ ， $C(u)$ 為雙曲線。

為求方便，我們固定 $w_0 = w_2 = 1$ ，調整 w_1 來決定曲線的形狀。

- $w_1^2 < 1 \Rightarrow$ 橢圓形。
- $w_1^2 = 1 \Rightarrow$ 拋物線。
- $w_1^2 > 1 \Rightarrow$ 雙曲線。

若要使 $C(u)$ 為圓的一部分，則 $\Delta P_0 P_1 P_2$ 必為一個等腰三角形，且 $w_1 = \cos \theta$ 。

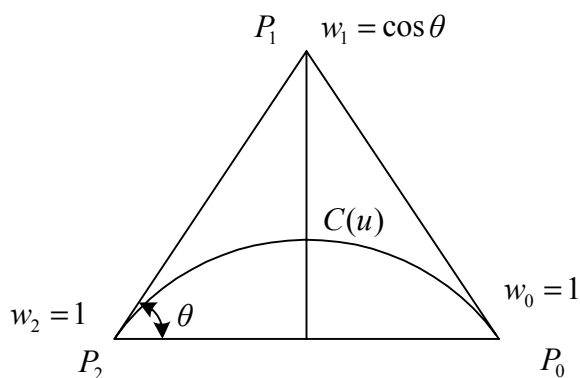


圖 3-9 以 NURBS 表示圓弧

若吾人的實作中，所要建構的圓弧根據其圓心大小，用不同數量的 rational Bézier 線段來表示，再將其轉換成 NURBS 的形式。

- $0 < \theta \leq 90$ ，arcs=1。
- $90 < \theta \leq 180.0$ ，arcs=2。
- $180 < \theta \leq 270$ ，arcs=3。
- $360 < \theta \leq 270$ ，arcs=4。

例如下面為不同圓心角的圓弧以 NURBS 來表示：

$\theta = 80^\circ$,

knot vector = $[0 \ 0 \ 0 \ 1 \ 1 \ 1]$

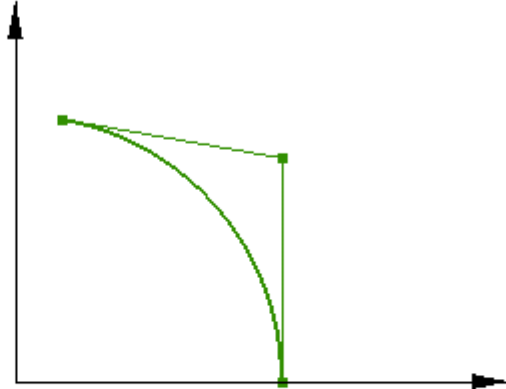


圖 3-10 圓心角為 80° 之圓弧

$\theta = 140$

knot vector = $[0 \ 0 \ 0 \ 1/2 \ 1/2 \ 1 \ 1 \ 1]$

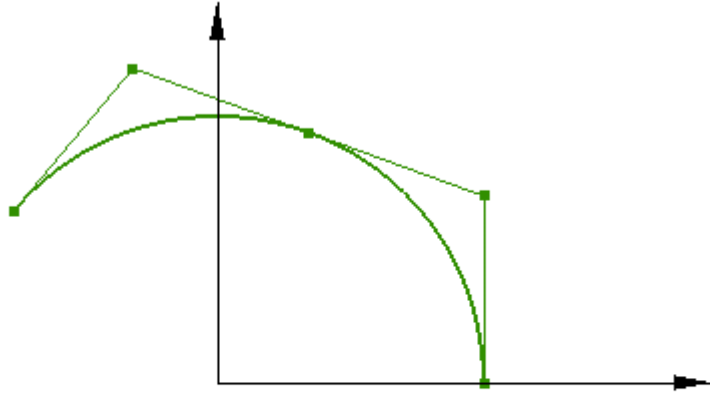


圖 3-11 圓心角為 140° 之圓弧

$\theta = 240$

knot vector = $[0 \ 0 \ 0 \ 1/3 \ 1/3 \ 2/3 \ 2/3 \ 1 \ 1 \ 1]$

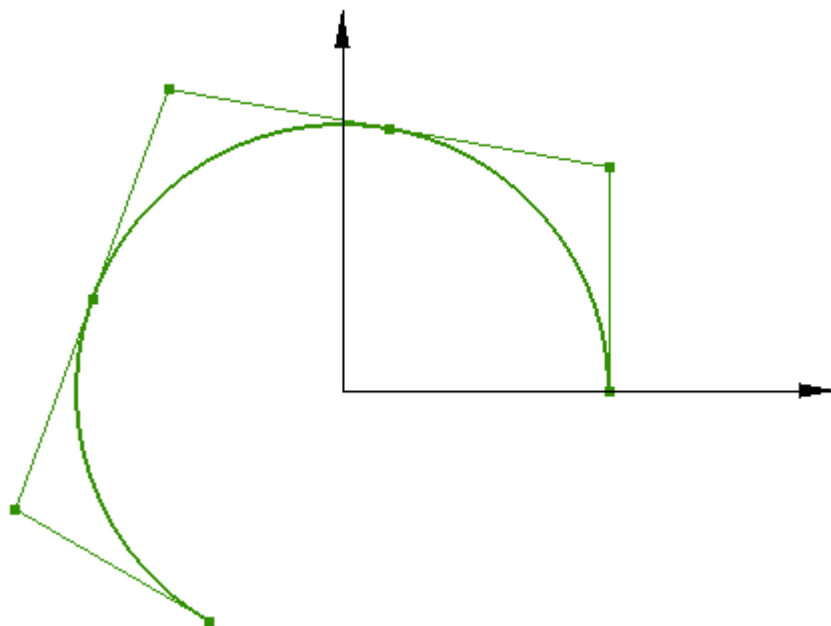


圖 3-12 圓心角為 240° 之圓弧

$\theta = 300$

knot vector =

$[0 \ 0 \ 0 \ 1/4 \ 1/4 \ 1/2 \ 1/2 \ 3/4 \ 3/4 \ 1 \ 1 \ 1]$

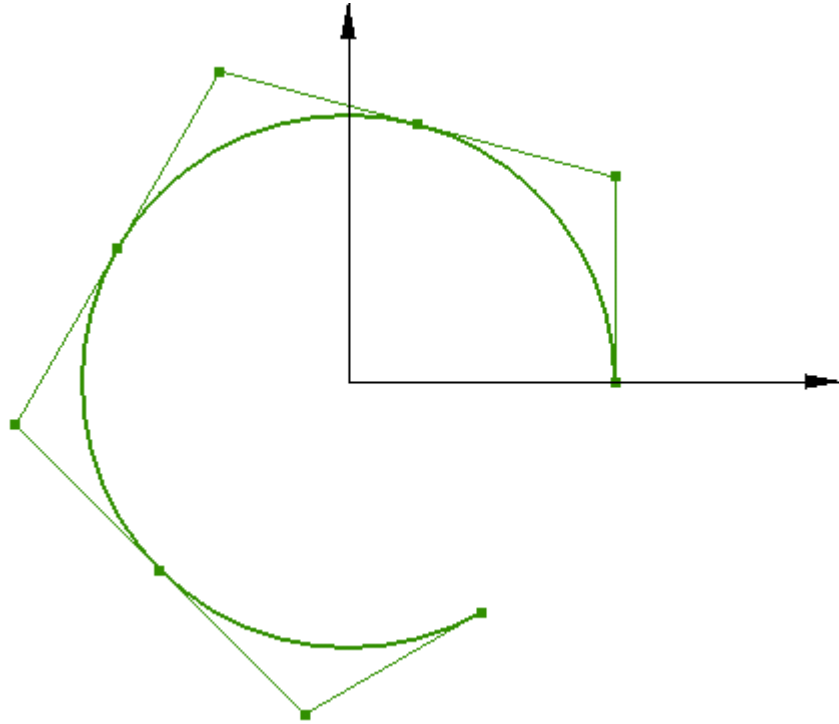


圖 3-13 圓心角為 300° 之圓弧

3-3-6 內插曲線(interpolation)

當 CAD 的使用者在繪圖時，常需要繪出一條曲線可通過指定的一些資料點，這個動作稱作 interpolate，由於組成一條 NURBS 曲線的要素很多，包括控制點，knot vector，weight 等，所以若要求出一條曲線通過某些固定點，就必須求出方才所提到的 NURBS 的組成要素，為了使問題單純化，我們假設 weight 都為 1，接下來的問題就只剩下控制點與 knot vector 了。

假設我們有 $n+1$ 的資料點 $\{Q_k\}$ ，如果我們對每一個資料點指定一個參數 \bar{u}_k ，且已知 knot vector $U = \{u_0, K, u_m\}$ ，則可

以解下面 $(n+1) \times (n+1)$ 的線性方程式來求得控制點 \mathbf{P}_i ：

$$\mathbf{Q}_k = \mathbf{P}(\bar{u}_k) = \sum_{i=0}^n N_{i,p}(\bar{u}_k) \mathbf{P}_i \quad (\text{Eq.3.22})$$

由此可知，我們必須先知道每一個資料點的參數 \bar{u}_k ，以及 knot vector。

以下有三種方法來求得 \bar{u}_k ：

1. 等距法

$$\begin{aligned} \bar{u}_0 &= 0 \\ \bar{u}_n &= 1 \\ \bar{u}_k &= \frac{k}{n} \quad k = 1, K, n-1 \end{aligned} \quad (\text{Eq.3.23})$$

這種方法所產生的曲線形狀較不理想，不建議使用。

2. 弦長法

$$\begin{aligned} d &= \sum_{k=1}^n |\mathbf{Q}_k - \mathbf{Q}_{k-1}| \\ \bar{u}_0 &= 0 \\ \bar{u}_n &= 1 \\ \bar{u}_k &= \bar{u}_{k-1} + \frac{|\mathbf{Q}_k - \mathbf{Q}_{k-1}|}{d} \quad k = 1, K, n-1 \end{aligned} \quad (\text{Eq.3.24})$$

這是最常用的方法，它可以得到較佳的資料點參數。

3. 向心法

$$\begin{aligned}
d &= \sum_{k=1}^n \sqrt{|\mathbf{Q}_k - \mathbf{Q}_{k-1}|} \\
\bar{u}_0 &= 0 \\
\bar{u}_n &= 1 \\
\bar{u}_k &= \bar{u}_{k-1} + \frac{\sqrt{|\mathbf{Q}_k - \mathbf{Q}_{k-1}|}}{d} \quad k=1, K, n-1
\end{aligned} \tag{Eq.3.25}$$

這是一個新的方法 Piegl [22], 當資料點相隔的距離差異過大時，這種方法所得的結果會比弦長法更好。

Knot vector 的求法大致來說有兩種：

1. 等距法

$$\begin{aligned}
u_0 = \Lambda = u_p &= 0 \\
u_{m-p} = \Lambda = u_m &= 1 \\
u_{j+p} &= \frac{j}{n-p+1} \quad j=1, K, n-p
\end{aligned} \tag{Eq.3.26}$$

2. 平均法

$$\begin{aligned}
u_0 = \Lambda = u_p &= 0 \\
u_{m-p} = \Lambda = u_m &= 1 \\
u_{j+p} &= \frac{1}{p} \sum_{i=j}^{j+p-1} u_i \quad j=1, K, n-p
\end{aligned} \tag{Eq.3.27}$$

等距法在使用上面的公式來求解時，可能會導致奇異 (singular) 的情形發生，若使用平均法，上面之聯定方程組可證明為正定 (positive)，故可用高斯消去法求解。

以下為一個例子：

Input：資料點 $(1,0)$ ， $(0.8,0.5)$ ， $(0.7,0.8)$ ， $(-0.2,0.5)$ ， $(-0.4,-0.2)$ ， $(-0.6,0)$

Degree = 3

Output：資料點參數 $\bar{u}_k = \{0, 0.1914, 0.3037, 0.6408, 0.8995, 1\}$

Kont vector $[0 \ 0 \ 0 \ 0 \ 0.3786 \ 0.4147 \ 1 \ 1 \ 1 \ 1]$

控制點

$(1, 0)$

$(0.7486, 0.2012)$

$(0.9410, 1.1921)$

$(-0.5584, 0.6211)$

$(-0.2120, -0.5539)$

$(-0.6, 0)$

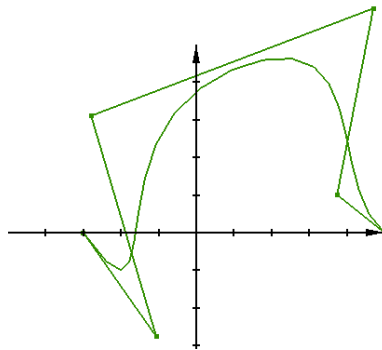


圖 3-14 所求得之內插曲線之控制點

3-3-7 近似曲線(Approximation)

所謂近似曲線是說，給定某些數量的資料點，我們想要得到一條曲線逼近這些點，但不需要完全正確地通過，近似曲線說起來要比內插曲線困難，因為對內插曲線來說，控制點的數量可由資料點的數目來求得，只剩下 degree、knot vector、和資料點參數未知；但近似曲線還多了二個未知數：控制點數量和誤差大小，所以吾人採用下面的流程來決定這二個額外的未知數：

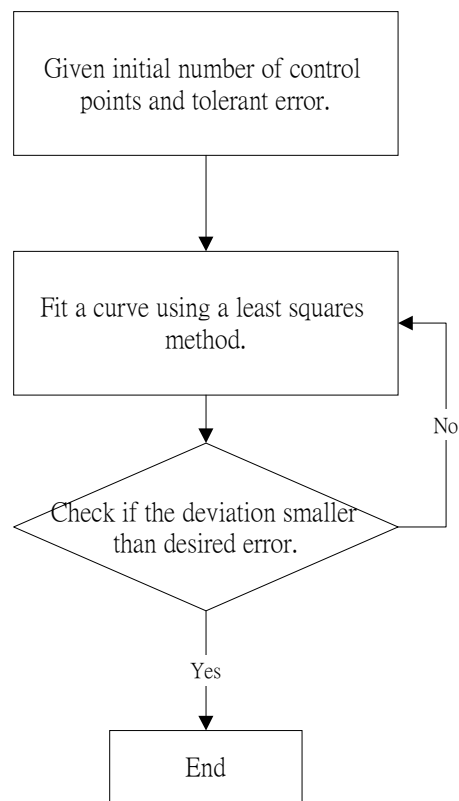


圖 3-15 求取近似曲線之流程圖

最小平方法

假設已經選定控制點的數目以及容許的誤差，首先可以使用在前面所介紹的方法來求解 knot vector 以及資料點參數，接著可以利用最小平方法求解出控制點的位置。

假設所需求解的曲線方程式次數(degree)為 p ，可得到如下的曲線方程式：

$$\mathbf{P}(u) = \sum N_{i,p}(u) \mathbf{P}_i \quad (\text{Eq.3.28})$$

吾人已知資料點 $\mathbf{Q}_0, \mathbf{Q}_1, \dots, \mathbf{Q}_m$ ，且 $\mathbf{Q}_0 = \mathbf{P}(0), \mathbf{Q}_m = \mathbf{P}(1)$ ，可以使用最小平方的概念，必須讓剩下的 $m-1$ 個資料點與真實曲線上的點的距離的平方和最小，也就是使下式為最小：

$$\sum_{k=1}^{m-1} \left| \mathbf{Q}_k - \mathbf{P}(\bar{u}_k) \right|^2 \quad (\text{Eq.3.29})$$

其中 \bar{u}_k 的計算方法同之前所提到計算資料點參數的方法。

令

$$\begin{aligned} f &= \sum_{k=1}^{m-1} \left| \mathbf{Q}_k - \mathbf{P}(\bar{u}_k) \right|^2 \\ &= \sum_{k=1}^{m-1} \left| \mathbf{R}_k - \sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k) \mathbf{P}_i \right|^2 \\ &= \sum_{k=1}^{m-1} \left(\mathbf{R}_k - \sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k) \mathbf{P}_i \right) \cdot \left(\mathbf{R}_k - \sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k) \mathbf{P}_i \right) \\ &= \sum_{k=1}^{m-1} \left[\mathbf{R}_k \cdot \mathbf{R}_k - 2 \sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k) (\mathbf{R}_k \cdot \mathbf{P}_i) + \left(\sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k) \mathbf{P}_i \right) \cdot \left(\sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k) \mathbf{P}_i \right) \right] \end{aligned} \quad (\text{Eq.3.30})$$

其中 $\mathbf{R}_k = \mathbf{Q}_k - N_{0,p}(\bar{u}_k)\mathbf{Q}_0 - N_{n,p}(\bar{u}_k)\mathbf{Q}_m \quad k=1, K, m-1$

吾人使用線性最小平方法 PiegI [22]，若要使 f 為最小，則必須使 f 對它每一個變數的偏微分為零，在上式中 f 有 $n-1$ 個變數 $\mathbf{P}_1, K, \mathbf{P}_{n-1}$ ，而 f 對變數的偏微分如下：

$$\begin{aligned} \frac{\partial f}{\partial P_l} &= \sum_{k=1}^{m-1} (-2N_{l,p}(\bar{u}_k)\mathbf{R}_k + 2N_{l,p}(\bar{u}_k) \sum_{i=1}^{n-1} N_{i,p}(\bar{u}_k)\mathbf{P}_i) = 0 \\ \Rightarrow \sum_{i=1}^{n-1} (\sum_{k=1}^{m-1} N_{l,p}(\bar{u}_k)N_{i,p}(\bar{u}_k))\mathbf{P}_i &= \sum_{k=1}^{m-1} N_{l,p}(\bar{u}_k)\mathbf{R}_k \end{aligned} \quad (\text{Eq.3.31})$$

上式可改寫成如下之聯立方程組：

$$(N^T N)\mathbf{P} = \mathbf{R} \quad (\text{Eq.3.32})$$

其中 N 為 $(m-1) \times (n-1)$ 的純量矩陣，

$$N = \begin{bmatrix} N_{1,p}(\bar{u}_1) & \Lambda & N_{n-1,p}(\bar{u}_1) \\ \mathbf{M} & \mathbf{O} & \mathbf{M} \\ N_{1,p}(\bar{u}_{m-1}) & \Lambda & N_{n-1,p}(\bar{u}_{m-1}) \end{bmatrix} \quad (\text{Eq.3.33})$$

\mathbf{R} 為 $n-1$ 個元素的向量，

$$\mathbf{R} = \begin{bmatrix} N_{1,p}(\bar{u}_1)\mathbf{R}_1 + \Lambda + N_{1,p}(\bar{u}_{m-1})\mathbf{R}_{m-1} \\ \mathbf{M} \\ N_{m-1,p}(\bar{u}_1)\mathbf{R}_1 + \Lambda + N_{n-1,p}(\bar{u}_{m-1})\mathbf{R}_{m-1} \end{bmatrix} \quad (\text{Eq.3.34})$$

\mathbf{P} 為 $n-1$ 個元素的向量

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_1 \\ \mathbf{M} \\ \mathbf{P}_{n-1} \end{bmatrix} \quad (\text{Eq.3.35})$$

由於 $\mathbf{P}_0, \mathbf{P}_n$ 必須等於資料點的起點與終點，即 $\mathbf{Q}_0, \mathbf{Q}_m$ ，所以若將上面之聯立方程組解出，就可求得代表此曲線的控制點 $\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_n$ 。

Knot vector 的求法可採用下面的公式：

$$\text{令 } d = \frac{m+1}{n-p+1} \quad (\text{Eq.3.36})$$

$$\begin{aligned} i &= \text{int}(jd) & \alpha &= jd - i \\ u_{p+j} &= (1-\alpha)\bar{u}_{i-1} + \alpha\bar{u}_i & j &= 1, \dots, n-p \end{aligned}$$

下面是實作出來的例子：

1. 資料點個數：20

控制點個數：6

容許誤差：不限制

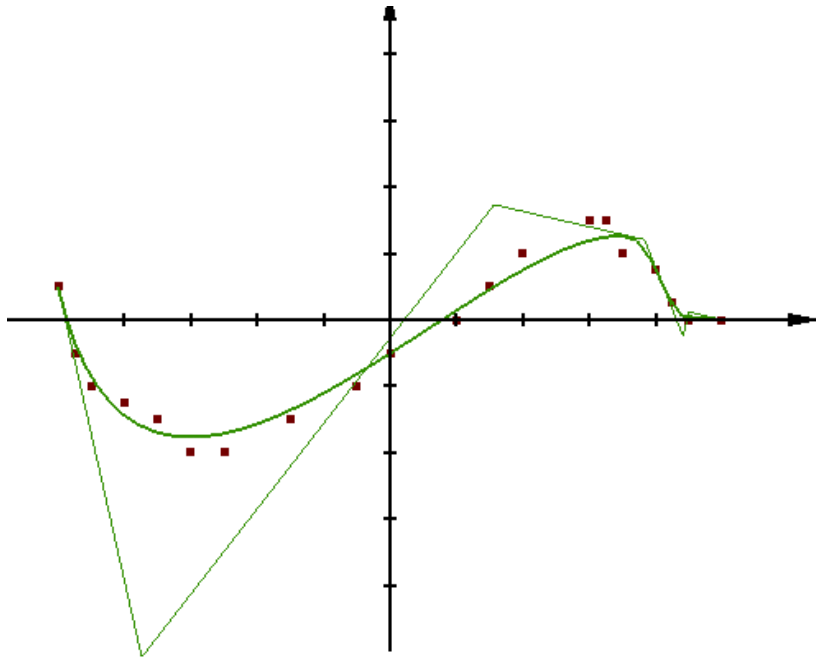


圖 3-16 近似曲線之一

2. 控制點個數初始值：6

容許誤差：0.01

Output -> 控制點個數：14

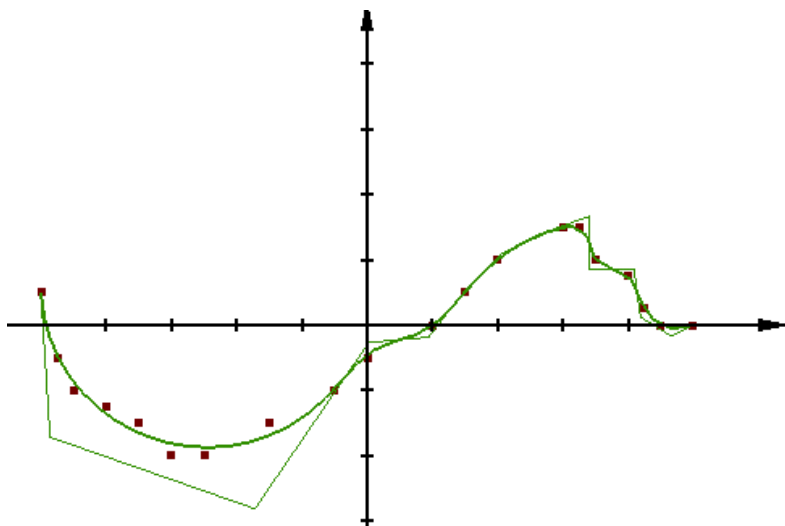


圖 3-17 近似曲線之二

3. 控制點個數初始值：6

容許誤差：0.0001

Output -> 控制點個數：20

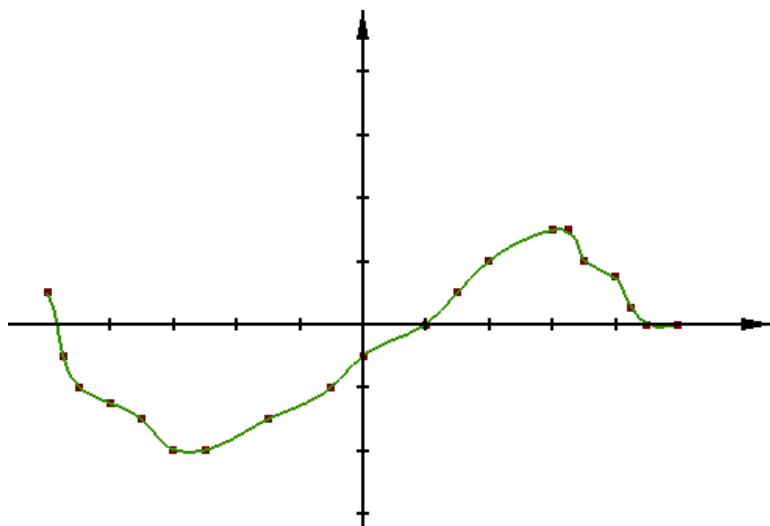


圖 3-18 近似曲線之三

3-4 偏移曲線(Offset curves)的探討

偏移曲線及曲面一直在工業界廣為應用，例如機械手臂的路徑設計、NC 工具機的刀具路徑的產生等，若針對曲線來說，我們都只探討在 2D 平面上的偏移，若以 $C(t) = (x(t), y(t))$ 來代表一條 2D 上的曲線，則其偏移曲線 $\tilde{C}(t)$ 可以被定義為如下：

$$\tilde{C}(t) = C(t) + d \cdot N(t) \quad (\text{Eq.3.37})$$

其中 d 代表偏移的距離， $N(t)$ 表示在曲線上的法向量。 $N(t)$ 可由下面的公式求出：

$$N(t) = \left(\frac{-\dot{y}(t)}{(\dot{x}^2 + \dot{y}^2)^{1/2}}, \frac{\dot{x}(t)}{(\dot{x}^2 + \dot{y}^2)^{1/2}} \right) \quad (\text{Eq.3.38})$$

其中 $\dot{x}(t)$ 和 $\dot{y}(t)$ 分別表示 $x(t)$ 和 $y(t)$ 的一次微分。

但是要產生偏移曲線並不是一件簡單的事，理由如下：

- 一條唯一的偏移曲線可能並不存在，因為在曲線上某一點的法向量可能不存在或有多個法向量，如圖。
- 通常來說，一條偏移曲線的形式與原來曲線的形式並不相同，例如一條 spline 曲線經過偏移後，可能不是一條 spline 曲線。
- 偏移曲線的形狀不僅僅與原來曲線的形狀有關，並且與偏移的距離與方向有關。
- 退化(degenerate)的情況會發生，以下舉幾個例子：
 1. 自我相交的情形，當曲面所佔據的空間相對於偏移距離來說太小時，會發生這種情形。
 2. 迴圈或尖端發生的情形，當曲面的曲率半徑小於偏移距離時，會發生這種情形。
 3. 其中一段曲線也許會消失。
 4. 一段新的曲線必須加入以維持曲線的完整性。
- 大部分的幾何核心都只提供少數幾種曲面，所以所求出的偏移曲線也必須是這幾種曲面之一。
- 既然大部分的偏移曲面部只能以近似的方式求出，所以偏移曲面的正確性、數值穩定度及方法的效率都必需加以探討。

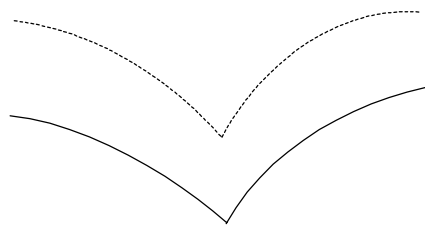


圖 3-19 法向量非唯一

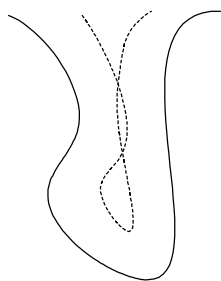


圖 3-20 產生自我相交

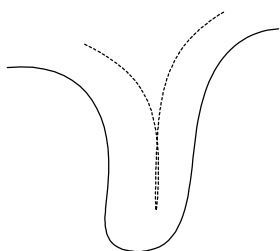


圖 3-21 迴圈或銳角

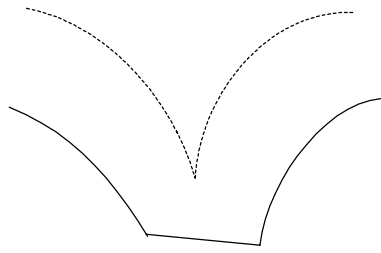


圖 3-22 某一線段消失

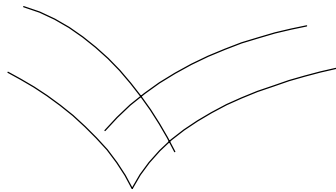


圖 3-23 需要移除線段

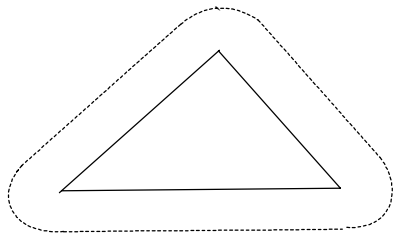


圖 3-24 需要增加曲線

在現階段，產生偏移曲線的方法皆使用近似的方法，這

些演算法都是以細分法(subdivision)為基礎，使用疊代的方式，來增加偏移曲線的準確性。

Tiller and Hanson[37]提出一種方法，使用 NURBS 曲線來近似偏移曲線。這個演算法先計算原來曲線的控制點多邊形，將這些直線偏移，新的控制點由剛才偏移的直線的交點求得，這種方法的準確性可以由細分法來控制，如果原來的曲線是直線或圓，可以在一次的疊代即得到所需的偏移曲線，若為其它種類的曲線，也可以在數次的疊代後得到理想的偏移曲線。

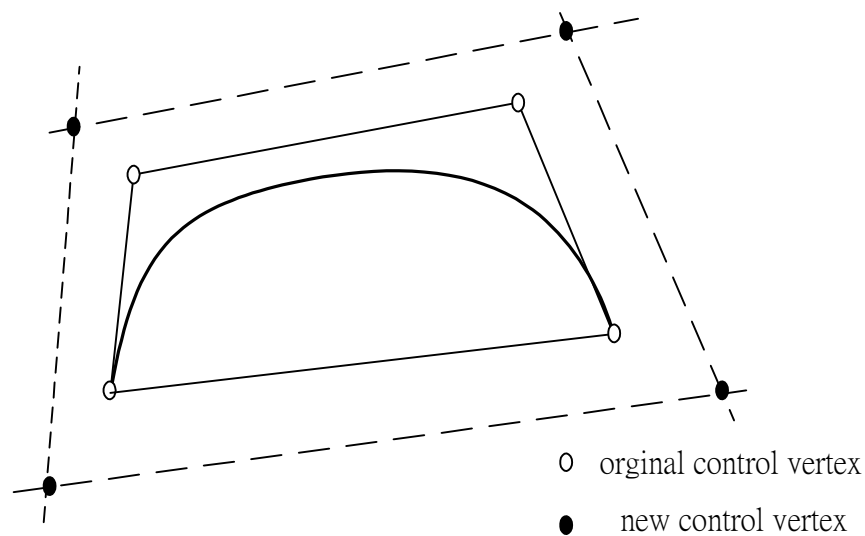


圖 3-25 利用偏移控制多邊形來求得偏移曲線

Coquillart[10]也是以 NURBS 來表示偏移曲線，她是將控制點偏移，偏移的法向量取距這個控制點最近的曲線上的點之法向量，而這個曲線上最近的點稱為控制節點(control node)，她還證明以這種方法來求得直線或圓的偏移是準確的。

而在產生偏移曲線時，偵測和處理奇異點也是一項必要的工作，Farouki 和 Neff[15][16]針對偏移曲線的拓撲和代數

性質提出了詳盡的研究，這些分析幫助我們了解有關複雜的奇異性的本質，並使我們有能力去處理它。

Farouki 提出如果給定一條曲線 $r(t)$ ，它的曲率為 $K(t)$ ，若其偏移距離為 d 的偏移曲線 $\tilde{r}(t)$ ，在 $t=t_c$ 時 $K(t_c)=1/d$ 且 $\tilde{K}(t_c)=0$ ，則在 t_c 這個點會發生尖端(cusp)。

自我相交的部分不只發生在尖端，也有可能發生在其它的情況，這也是我們必須加以偵測並且排除的，Hoschek[20]提出了一個演算法可以用來找出曲線自我相交的部分。

Tiller 和 Hanson's 也提供了一個簡單的方法來偵測自我相交的情形，他認為一個偏移曲面的方向改變時，也是它的控制多邊形方向改變的時候，所以偏移曲面的控制多邊形的自我相交約略可表示偏移曲面發生自我相交，

第四章 曲面的理論簡介與實作

4-1 前言

曲面在工業界上的應用十分廣泛，包括汽車外形設計，模具設計，乃至傢俱，日常用品等工業工程設計，都需要用到大量的曲面功能，所以曲面的 CAD 幾何核心中佔的份量相當大。曲面也如同曲線一般包括顯性曲面、隱性曲面以及參數曲面。

4-2 參數曲面

一個參數曲面可說是一個 patch，它是複雜曲面的基礎，它通常有二個參數 u 和 v ，如同以下的形式：

$$\begin{aligned}x &= x(u, v) \\ y &= y(u, v) \\ z &= z(u, v)\end{aligned}\tag{Eq.4.1}$$

在我的實作中，將參數限定在 0~1 之間，即 $u, v \in [0, 1]$ 。故這一個 patch 可能是如下圖所示：

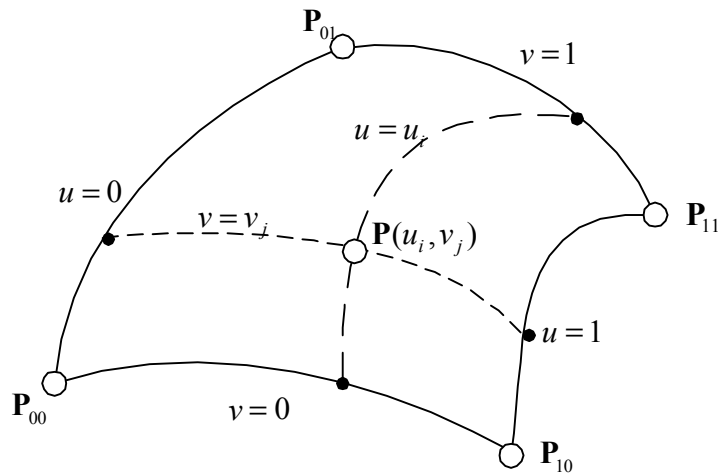


圖 4-1 參數曲面示意圖

參數曲面如同參數曲線一般，可分為 Hermite 曲面，Bézier 曲面，B-Spline 曲面及 NURBS 曲面。

4-2-1 Hermite surface

我們將 Hermite 曲線擴展成二維時，就成了 Hermite 曲面，又稱 bicubic Hermite 曲面，回憶上一章所介紹的 Hermite 曲線，其構成要素為端點的位置向量及切線向量。而 bicubic Hermite 曲面則需要這個 patch 四個角的位置向量及 u ， v 方向的切線向量外，還需要四個角的 twist vector，即 $\frac{\partial \mathbf{P}(u,v)}{\partial u \partial v}$ 等十六個條件。

它的參數式表示如下：

$$\mathbf{P}(u, v) = \sum_{i=0}^3 \sum_{j=0}^3 \mathbf{a}_{ij} u^i v^j \quad (\text{Eq.4.2})$$

其中 \mathbf{a}_{ij} 可由上述之構成 bicubic Hermite 曲面的十六個條件所求得。

4-2-2 Bézier surface

Bézier 曲面也是 Bézier 曲線的延伸，它的參數方程式可表示如下：

$$\mathbf{P}(u, v) = \sum_{i=0}^m \sum_{j=0}^n \mathbf{P}_{ij} B_{i,m}(u) B_{j,n}(v) \quad u, v \in [0,1] \quad (\text{Eq.4.3})$$

\mathbf{P}_{ij} 表示組成 Bézier 的 $(m+1) \times (n+1)$ 個控制點。 $B_{i,m}(u)$ 和 $B_{j,n}(v)$ 同樣為 Bernstein 多項式。所以一個 $m=3$ ， $n=3$ 的 bicubic Bézier patch 可能如下圖所示：

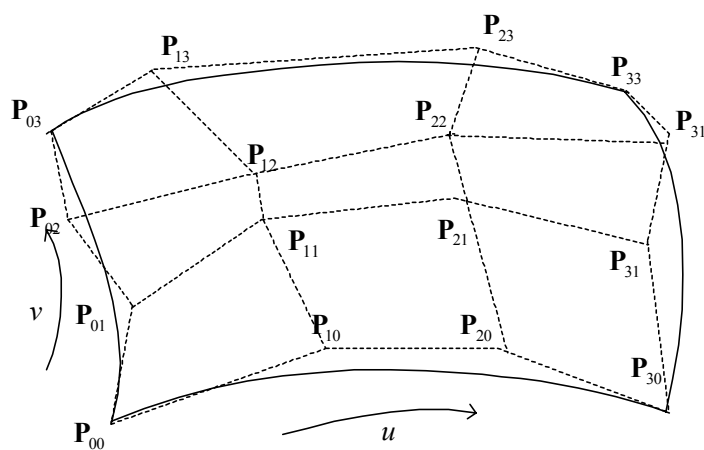


圖 4-2 Bézier 曲面示意圖

而 control points 所構成的網格狀多邊形稱為 Bézier 曲面的特徵多邊形(characteristic polyhedron)，它是 Bézier 曲面的重要特性之一，即曲面上任何一點都會在這個多邊形的內部。

4-2-3 B-Spline surface

B-Spline 曲面也是與 Bézier 曲面相類似，參數方程式表示如下：

$$\mathbf{P}(u, v) = \sum_{i=0}^m \sum_{j=0}^n \mathbf{P}_{ij} N_{i,k}(u) N_{j,l}(v) \quad u, v \in [0, 1] \quad (\text{Eq.4.4})$$

$N_{i,k}(u)$ 和 $N_{j,l}(v)$ 的定義與 B-Spline 曲線相同，k、l 分別代表 u 方向與 v 方向的次數(degree)。

4-2-4 Non-rectangular Gregory Patches

這一節探討三角形的曲面，它是由三角形的三個邊界曲線及邊界曲線正交方向的切線向量所組成，如圖所示：

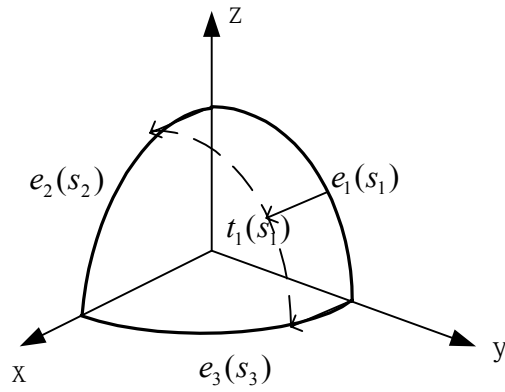


圖 4-3 三角形曲面之邊界資料

三角形曲面之邊界曲線的資料有下面兩個：

$e_i(s_i)$: i^{th} boundary curve with $0 \leq s_i \leq 1$ for $i = 1, 2, 3$

$t_i(s_i)$: cross-boundary tangent of $e_i(s_i)$

為了要從上述的邊界條件來建構一個平滑的曲面，我們要先定義三角形參數 domain，讓我們考慮一個單位高的正三角形，其頂點為 V1、V2、V3，讓 λ_i 表示這個正三角形中一點 v，到三個邊的垂直距離。

$$v = (\lambda_1, \lambda_2, \lambda_3)$$

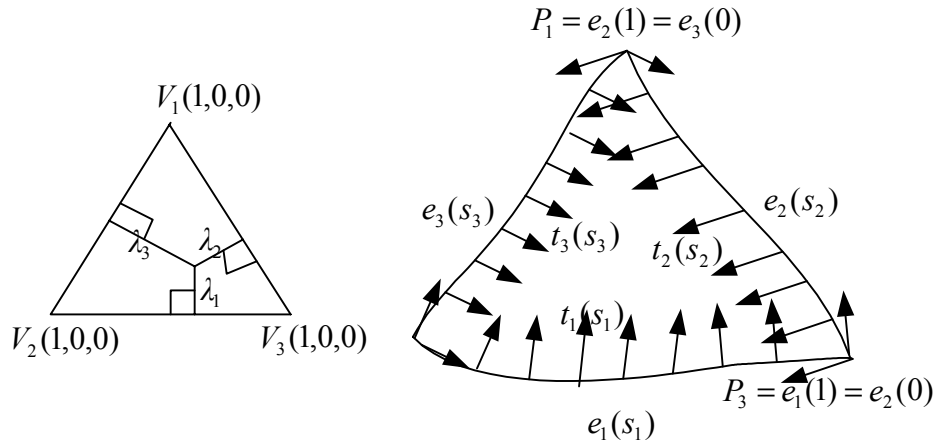


圖 4-4 建構一個 Gregory Patch

這些邊界曲線的參數 s_i 可以以 λ_i 來定義如下：

$$\begin{aligned} s_1 &= \lambda_3 / (\lambda_2 + \lambda_3) \\ s_2 &= \lambda_1 / (\lambda_3 + \lambda_1) \\ s_3 &= \lambda_2 / (\lambda_1 + \lambda_2) \end{aligned} \quad (\text{Eq.4.5})$$

接下來我們用邊界曲線的兩個資料 $e_i(s_i)$ 、 $t_i(s_i)$ 定義出線性泰勒內插值(linear Taylor interpolant) $r_i(s_i, \lambda_i)$ ：

$$r_i(s_i, \lambda_i) = e_i(s_i) + \lambda_i t_i(s_i) \quad \text{for } i = 1, 2, 3$$

where, $s_i \in [0, 1]$: parameter for boundary i

$\{\lambda_i\}$: barycentric coordinates

$e_i(s_i)$: boundary curve i

$t_i(s_i)$: cross-boundary tangent for boundary I

最後，在這三個邊界中間曲面的值可以用下面的公式建構出，這也是所謂的 triangular Gregory patch：

$$r(v) = \sum_{i=1}^3 \gamma_i(v) \{e_i(s_i) + \lambda_i t_i(s_i)\} \quad (\text{Eq.4.6})$$

where, $v = (\lambda_1, \lambda_2, \lambda_3)$: barycentric coordinates

s_i : curve parameter

$$\gamma_i(v) = \begin{cases} (1/\lambda_i)^2 / \sum_j (1/\lambda_j)^2 & \text{if } \lambda_i \neq 0 \\ 1 & \text{if } \lambda_i = 0 \end{cases}$$

以上是 triangular Gregory patch 產生的方法，它可以以同樣的方法產生出 n-side Gregory patch，詳細的部分可參考文獻 Gregory [17]。

4-3 NURBS 曲面

因 NURBS 曲面幾乎可以表示任何的參數曲面，所以吾人所實作之幾何核心在曲面方面，是以 NURBS 曲面為主，其參數方程式如下：

$$\mathbf{P}(u, v) = \frac{\sum_{i=0}^m \sum_{j=0}^n w_{ij} \mathbf{P}_{ij} N_{i,p}(u) N_{j,q}(v)}{\sum_{i=0}^m \sum_{j=0}^n w_{ij} N_{i,p}(u) N_{j,q}(v)} \quad (\text{Eq.4.7})$$

同樣地，NURBS 曲面的基底函數 $N_{i,p}(u)$ 和 $N_{j,q}(v)$ 是與 B-Spline 曲線的基底函數相同。

以 C 的結構來表示 NURBS 曲面如下面的形式：


```

/////////////////////////////////////////////////////////////////
typedef struct surface
{
    int n;
    int p;
    int m;
    int q;
    double *knu;
    double *knv;
    double *Pw;
} SURFACE;
/////////////////////////////////////////////////////////////////
n : u 方向控制點的數目-1

m : v 方向控制點的數目-1

p : u 方向 knots 的數目-1

q : v 方向 knots 的數目-1

knu : u 方向 knot vector 所組成的陣列

knv : v 方向 knot vector 所組成的陣列

Pw : 控制點的 homogenous coordinate 所組成的陣列

```

NURBS 曲面所要實作的內容與 NURBS 曲線相近，依次有曲面位置與其 n 次微分，等參數曲線(isoparametric curve)，內插(interpolation)曲面，近似(Approximation)曲面及求得點在線上的參數(inverse-point)等，另外還有一般曲面的形成方法，如 Cylindrical Surface、Ruled Surface、Revolved Surface、Coon Surface、Swept Surface 等。

NURBS 的曲面位置和其 n 次微分與 NURBS 曲線的求法相類似，只不過將其擴展成二維，在此不再贅述，可參考 Piegl [22]。

4-3-1 等參數曲線(isoparametric curve)

所謂等參數曲線，就是在曲面上，固定某一個參數(u 或 v)，所形成的一條曲線。

例如，若固定 $u = u_0$ ，則曲線的參數方程式可推導如下：

$$\begin{aligned} \mathbf{Pw}(v) &= \mathbf{Sw}(u_0, v) \\ &= \sum_{i=0}^n \sum_{j=0}^m N_{i,p}(u_0) N_{j,q}(v) \mathbf{Pw}_{i,j} \\ &= \sum_{j=0}^m N_{j,q}(v) \left(\sum_{i=0}^n N_{i,p}(u_0) \mathbf{Pw}_{i,j} \right) \\ &= \sum_{j=0}^m N_{i,p}(u_0) \mathbf{Q}_j^w(u_0) \end{aligned} \quad (\text{Eq.4.8})$$

由上面推導可知， $\mathbf{Q}_j^w(u_0)$ 為等 u 曲線的控制點，而等 u 曲線的 degree 與 knot vector 則與 來曲面 v 方向上的 degree 與 knot vector 相同，同理，若固定 $v = v_0$ ，可得等 v 曲線的參數方程式如下：

$$Pw(u) = \sum_{i=0}^n N_{i,p}(u) \mathbf{Q}_i^w(v_0) \quad (\text{Eq.4.9})$$

其中 $\mathbf{Q}_i^w(v_0) = \sum_{j=0}^m N_{j,q}(v_0) \mathbf{Pw}_{i,j}$ 。

4-3-2 內插(interpolation)曲面

內插曲面要求出一個曲面能通過所給定的一組資料點，這組資料點必須是 $(n+1) \times (m+1)$ 的陣列，假設這組資料點為 $\{Q_{k,l}\}$ ， $k=0,K,n$ 且 $l=0,K,m$ ，則它必符合下面方程式：

$$Q_{k,l} = P(\bar{u}_k, \bar{v}_l) = \sum_{i=0}^n \sum_{j=0}^m N_{i,p}(\bar{u}_k) N_{j,q}(\bar{v}_l) P_{i,j} \quad (\text{Eq.4.10})$$

若欲求解上面方程式來得到控制點，除了 degree 必須決定外，還有資料點參數 \bar{u}_k ， $k=0,K,n$ 和 \bar{v}_l ， $l=0,K,m$ 要計算出來，在此僅說明如何計算 \bar{u}_k ， \bar{v}_l 則是相類似。可以利用之前提到的弦長法或向心法，對每一個 l ，計算出參數 $\bar{u}_0, \bar{u}_K, \bar{u}_n$ ，再利用平均的方法，算出 $\bar{u}_0, \bar{u}_K, \bar{u}_n$ ，

$$\bar{u}_k = \frac{1}{m+1} \sum_{l=0}^m \bar{u}_k^l \quad k=0,K,n \quad (\text{Eq.4.11})$$

接下來就可以求解上式，但吾人並不打算一次就求出所有的控制點 $P_{i,j}$ ，因為這樣所需的時間較長，所以吾人採取作兩次內插曲線的方式，將所有的控制點求出。我們可將上式改寫成如下之形式：

$$\begin{aligned} Q_{k,l} &= \sum_{i=0}^n N_{i,p}(\bar{u}_k) \left(\sum_{j=0}^m N_{j,q}(\bar{v}_l) P_{i,j} \right) \\ &= \sum_{i=0}^n N_{i,p}(\bar{u}_k) R_{i,l} \end{aligned} \quad (\text{Eq.4.12})$$

$$\text{其中 } R_{i,l} = \sum_{j=0}^m N_{j,q}(\bar{v}_l) P_{i,j}$$

由上面可知，吾人可先利用(Eq.3.12)將 $R_{i,l}$ 求出，再利用

(Eq.4.10)將 $\mathbf{P}_{i,j}$ 求出。也就是說，吾人先對 $\mathbf{Q}_{0,l}, K, \mathbf{Q}_{n,l}$ 作 curve interpolation，可以得到 $\mathbf{R}_{0,l}, K, \mathbf{R}_{n,l}$ ， $l = 0, K, m$ ；之後再對 $\mathbf{R}_{i,0}, K, \mathbf{R}_{i,m}$ 作 curve interpolation，可以得到 $\mathbf{P}_{i,0}, K, \mathbf{P}_{i,m}$ ， $i = 0, K, n$ 。

4-3-3 圓柱曲面(Cylindrical Surface)

圓柱曲面就是有一條 NURBS 曲線 $\mathbf{C}(u) = \sum_{i=0}^n R_{i,p}(u) \mathbf{P}_i$ ，沿著一向量 \mathbf{W} 所掃出的曲面，這個曲面可表示成如下的形式：

$$\mathbf{S}(u, v) = \mathbf{C}(u) + v d\mathbf{W} \quad (\text{Eq.4.13})$$

但我們希望是以 NURBS 曲面的形式如下：

$$\mathbf{S}(u, v) = \sum_{i=0}^n \sum_{j=0}^1 R_{i,p;j,l}(u, v) \mathbf{P}_{i,j} \quad (\text{Eq.4.14})$$

此曲面在 u 方向的 degree 與 knot vector 與 $\mathbf{C}(u)$ 相同，在 v 方向上的 degree 為 2，knot vector $V = \{0 \ 0 \ 1 \ 1\}$ ，而控制點則為 $\mathbf{P}_{i,0} = \mathbf{P}_i$ ， $\mathbf{P}_{i,1} = \mathbf{P}_i + d\mathbf{W}$ ，權重則為 $w_{i,0} = w_{i,1} = w_i$ 。

以下為一個例子：

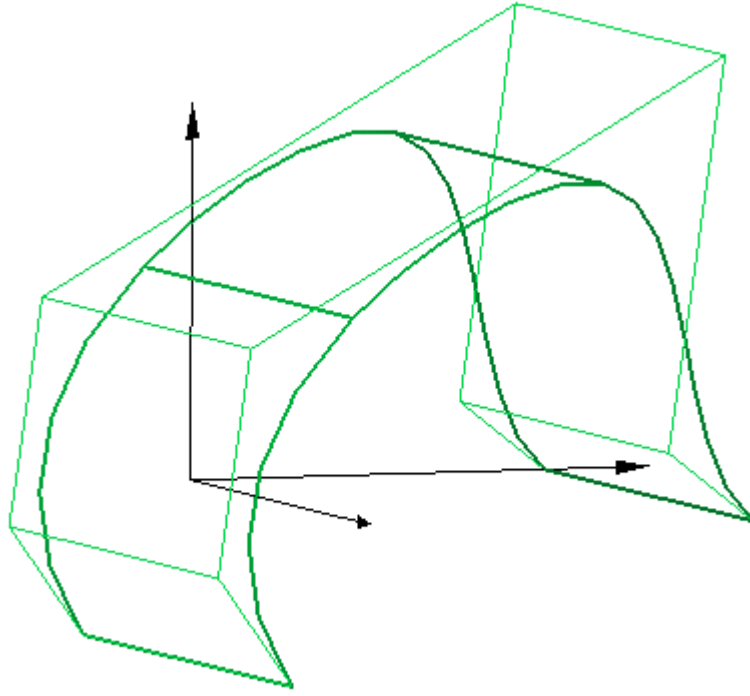


圖 4-5 Cylindrical Surface

4-3-4 Ruled Surface

Ruled surface 是指在空間中有兩條曲線，將這兩條曲線等參數的點以直線連接，所得到的曲面稱做 Ruled surface。

假設這兩條曲線的方程式如下：

$$\begin{aligned} \mathbf{C}_1(u) &= \sum_{i=0}^{n1} R_{i,p1}(u) \mathbf{P}_i^1 \\ \mathbf{C}_2(u) &= \sum_{i=0}^{n2} R_{i,p2}(u) \mathbf{P}_i^2 \end{aligned} \quad (\text{Eq.4.15})$$

若這兩條曲線的 knot vector 與 degree 相同，則其 Ruled surface 的曲面方程式可以表示如下：

$$S(u, v) = \sum_{i=0}^n \sum_{j=0}^1 R_{i,p;j,l}(u, v) \mathbf{P}_{i,j} \quad (\text{Eq.4.16})$$

其中 $p = p1 = p2$, $n = n1 = n2$, $\mathbf{P}_{i,0} = \mathbf{P}_i^1$, $\mathbf{P}_{i,1} = \mathbf{P}_i^2$ 。

若這兩條曲線的 knot vector 或 degree 不同，首先要使 degree 相同，degree 較小的必須升階，升階的方法已在前面提到。degree 相同後，就要使 knot vector 也相同，此時必須用到前面所提到的 Refine knots 或 knots insertion。

以下為一個例子：

Curve1->knot vector = [0 0 0 0 0.3868 0.6279 1 1 1 1]
degree = 3

Curve2->knot vector = [0 0 0 0.5 0.5 1 1 1]
degree = 2

Surface-> u-direction knot vector = [0 0 0 0 0.3868 0.5 0.5
0.5 0.6279 1 1 1 1]

v-direction knot vector = [0 0 1 1]

u-direction degree = 3

v-direction degree = 1

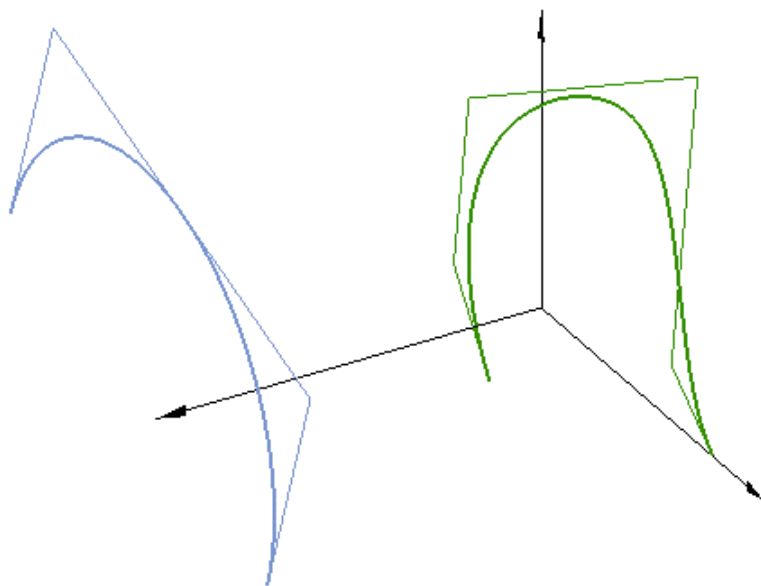


圖 4-6 兩條階數與控制點數目不同的 NURBS 曲線

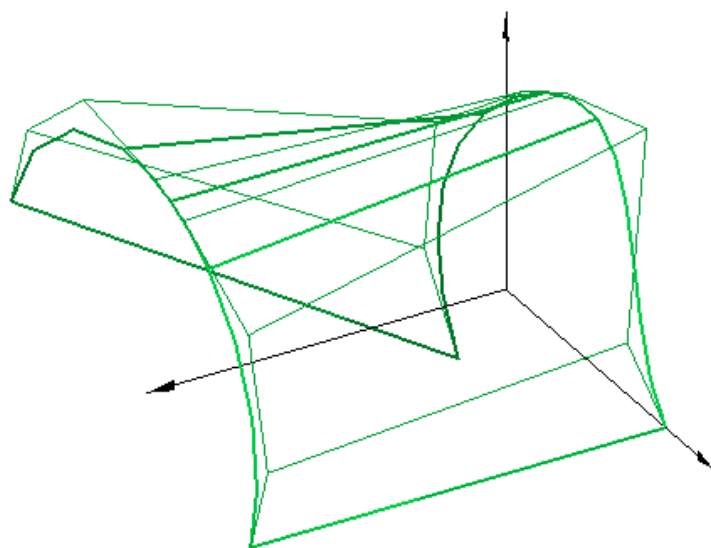


圖 4-7 圖 4-5 所形成的 Ruled Surface

4-3-5 Revolved Surface

給定一條曲線(based curve)，使其繞一固定軸旋轉，所掃出的曲面，我們稱之為 revolved surface。也就是說，在這條曲線上的每一點，都會以這個固定軸為圓心，劃出一條圓或圓的一部分的路徑。所以我們可以以這條 based curve 的 degree、knot vector 來當做 revolved surface 的 v 方向的 degree、knot vector，而 u 方向的 degree、knot vector 則使用圓的 degree 與 knot vector。至於控制點，則可以參考第二章之以 NURBS 表示圓的方法來求出。

例如若我們要掃出一個 1/4 的圓，則可以使用 u 方向的 knot vector = $\{0 \ 0 \ 0 \ 0.5 \ 0.5 \ 1 \ 1 \ 1\}$ ，weights = $\left\{1 \ \frac{\sqrt{2}}{2} \ 1\right\}$ ，曲面的方程式就可以表示如下：

$$S(u, v) = \sum_{i=0}^2 \sum_{j=0}^m R_{i,2;j,q}(u, v) \mathbf{P}_{i,j} \quad (\text{Eq.4.17})$$

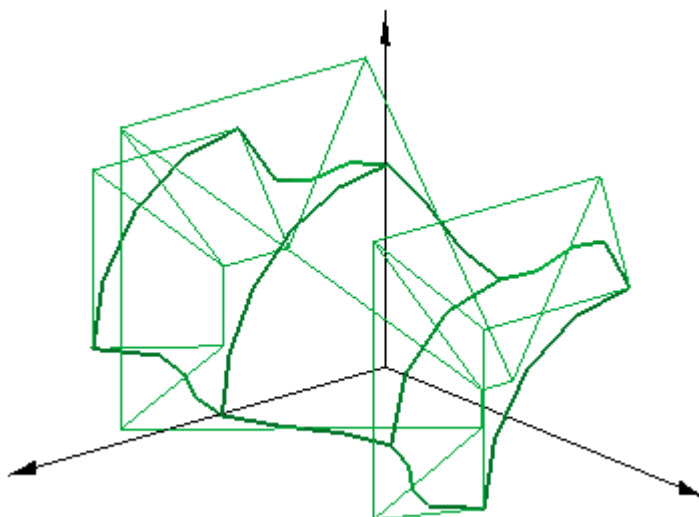


圖 4-8 角度為 90° 之 Revolved Surface

以下的例子為一個圓當做 based curve，對某一軸掃出 360° ，所形成的環。

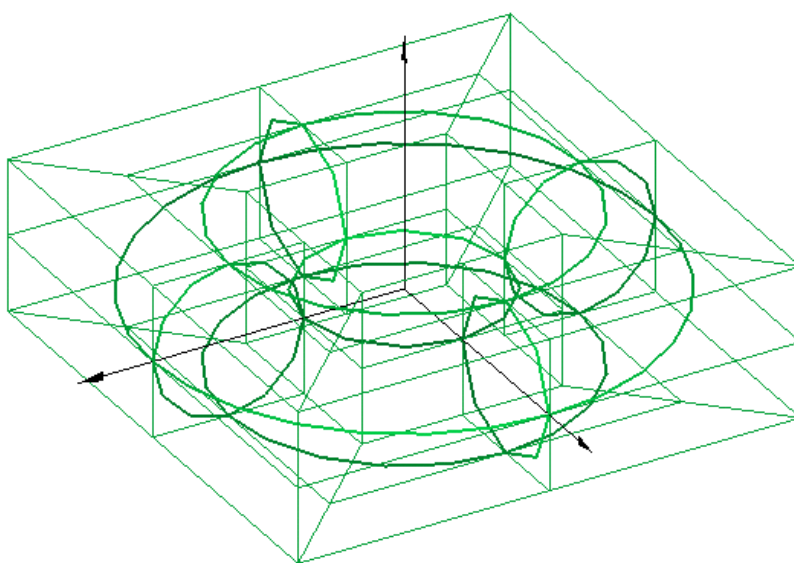


圖 4-9 Revolved Surface 所形成之環(torus)

4-3-6 Coon Surface

Coon surface 是由四條曲線所構成，分別為 coon surface 的四個邊界，如下圖：

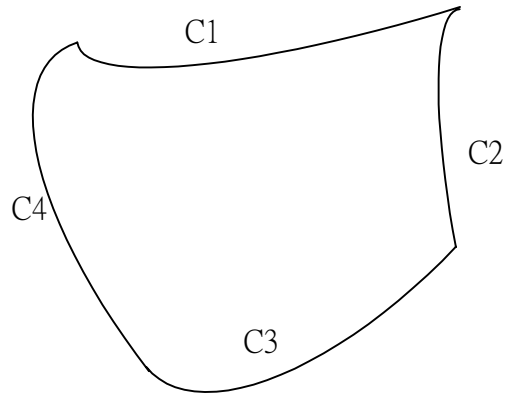


圖 4-10 四條相接之曲線示意圖

吾人採用的是線性內插法，所謂線性內插法就是在這四條曲線內部的點，是由下面的公式所構成：

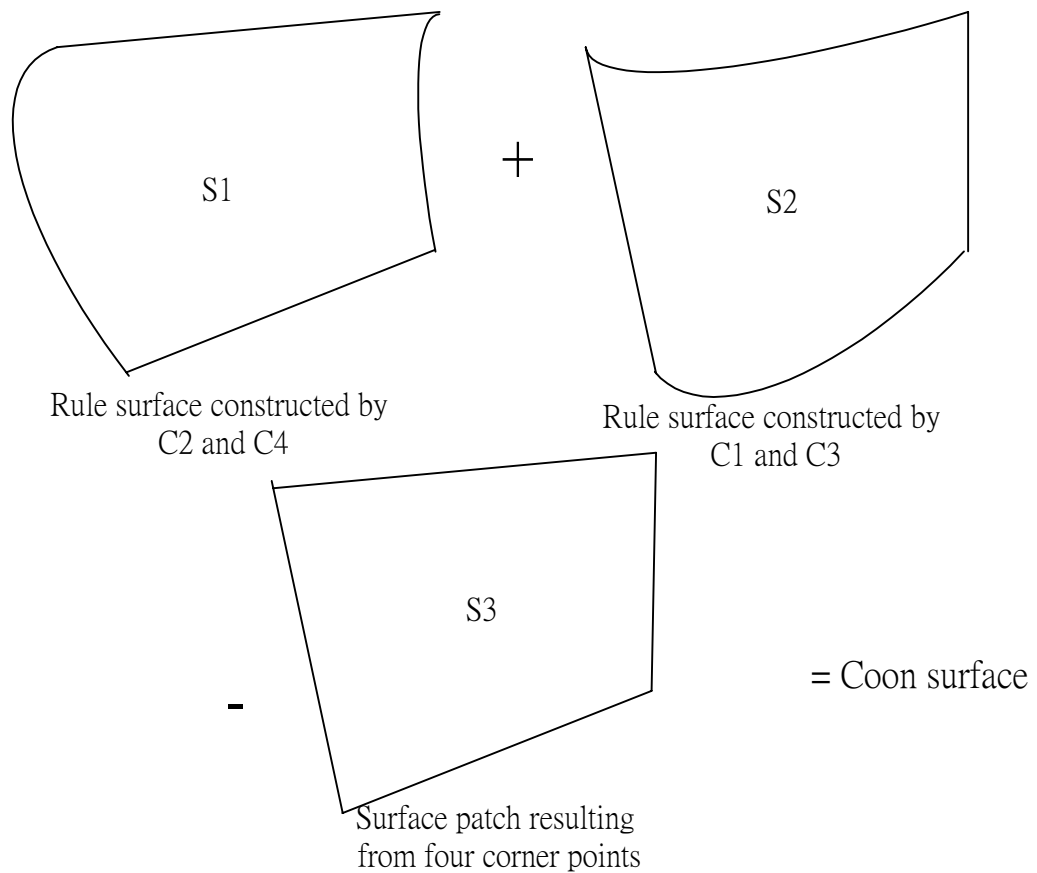


圖 4-11 Coon 曲面形成示意圖

吾人的 coon surface 求解方式就是如上圖，依序將 C_1 、 C_3 所組成的 rule surface 和 C_2 、 C_4 所組成的 rule surface 建構出來，再將由四個點所線性內插的 surface patch 建構出來，之後利用升階或插入 knots 的方式，使三個 surface patch 的 u 方向及 v 方向的 degree 及 knot 都相同，接著可由下面公式得知：

$$\begin{aligned}
S(u,v) &= \sum_{i=0}^n \sum_{j=0}^m R_{i,p;j,q}(u,v) \mathbf{P}_{i,j} \\
&= \sum_{i=0}^n \sum_{j=0}^m R_{i,p;j,q}(u,v) \mathbf{P}_{i,j}^1 + \sum_{i=0}^n \sum_{j=0}^m R_{i,p;j,q}(u,v) \mathbf{P}_{i,j}^2 - \sum_{i=0}^n \sum_{j=0}^m R_{i,p;j,q}(u,v) \mathbf{P}_{i,j}^3
\end{aligned}$$

(Eq.4.18)

所以

$$\mathbf{P}_{i,j} = \mathbf{P}_{i,j}^1 + \mathbf{P}_{i,j}^2 - \mathbf{P}_{i,j}^3 \quad (\text{Eq.4.19})$$

其中 $\mathbf{P}_{i,j}^1$ 表示 S1 的控制點， $\mathbf{P}_{i,j}^2$ 表示 S2 的控制點， $\mathbf{P}_{i,j}^3$ 表示 S3 的控制點。故可將 coon surface 的控制點求出。

以下為一個例子：

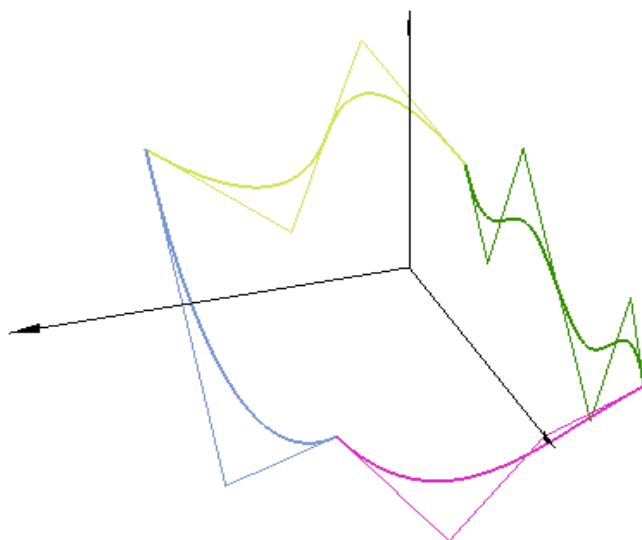


圖 4-12 四條相接之曲線

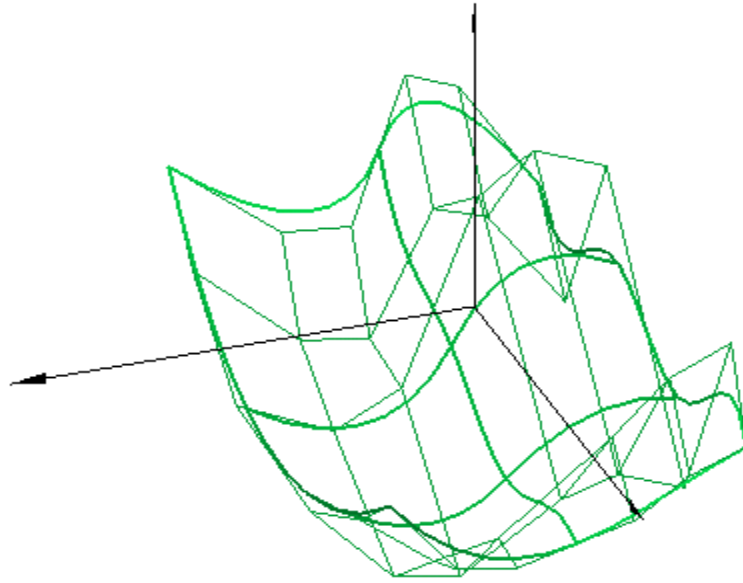


圖 4-13 Coon 曲面

4-3-7 Skin Surface

以數條已知的曲線來當做曲面的某幾條等參數曲線，所形成的曲面就稱為 skin surface，可以以下面的圖形來表示：

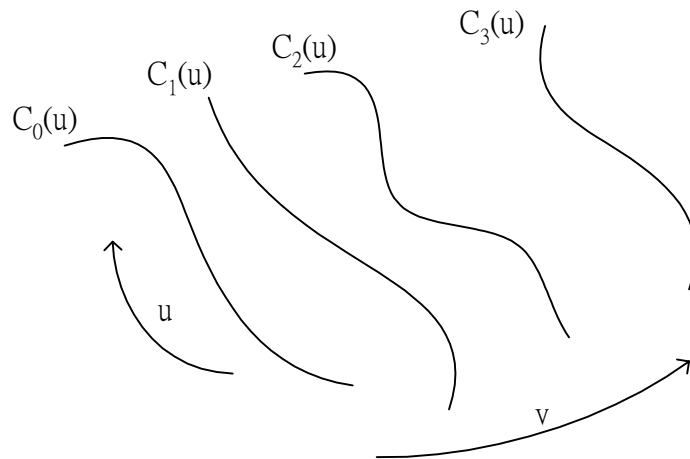


圖 4-14 數條相鄰之曲線

將上面的四條曲線中間內插出曲面的點，所構成的曲面就是 skin surface。

它與 rule surface 有些類似，只不過 rule surface 的已知條件只有兩條曲線，分別為 rule surface $v=0$ 及 $v=1$ 的等參數曲線，而 skin surface 的已知條件可能有 $K+1$ 條曲線，分別代表 skin surface $v=v_k, k=0, K, K$ 的等參數曲線，兩者的求解方法相類似，首先要使這 $K+1$ 條曲線 degree 相同，之後再使用 knot refinement 的方法使 $K+1$ 條曲線的 knot vector 都相同，接著決定這些曲線的參數 v_k ，使用與 surface interpolation 的方法，將曲線的控制點沿著 v 方向作 curve interpolation，就可以產生 skin surface 的控制點。

Skin surface 產生的步驟如下：

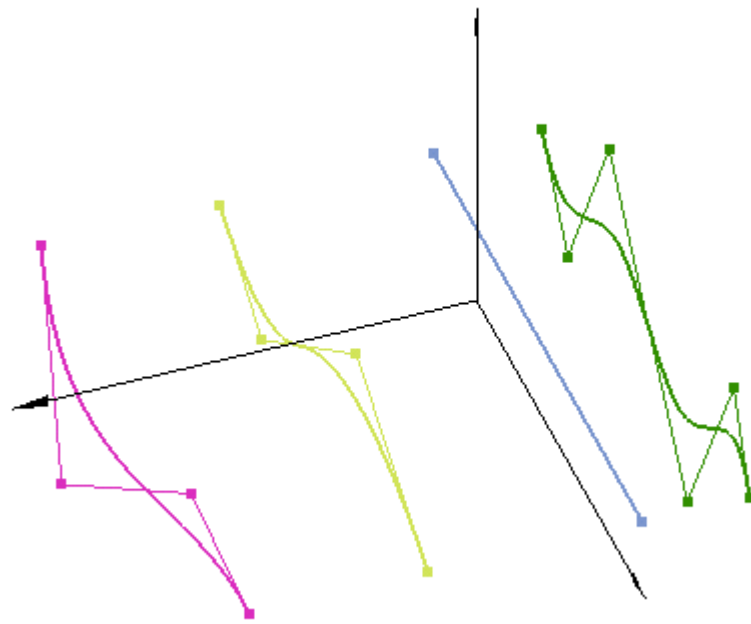


圖 4-15 形成 Skin Surface 的第一步驟

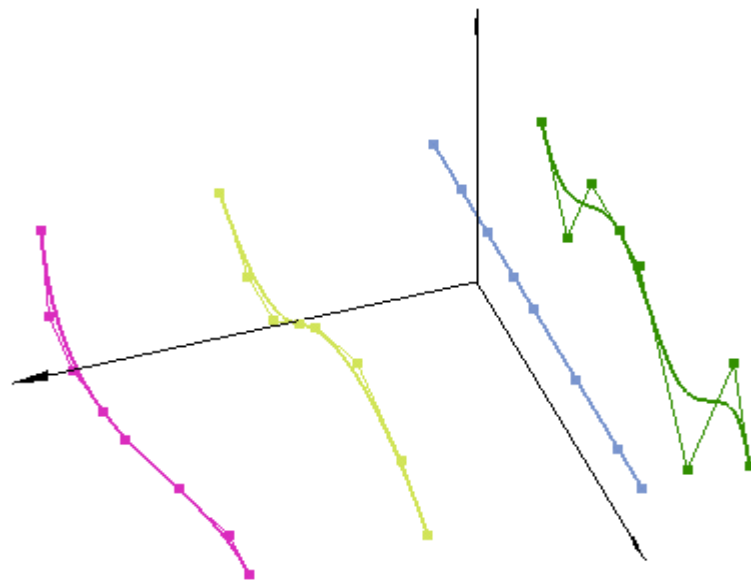


圖 4-16 形成 Skin Surface 的第二步驟，將其 knot vector 及階數成相同

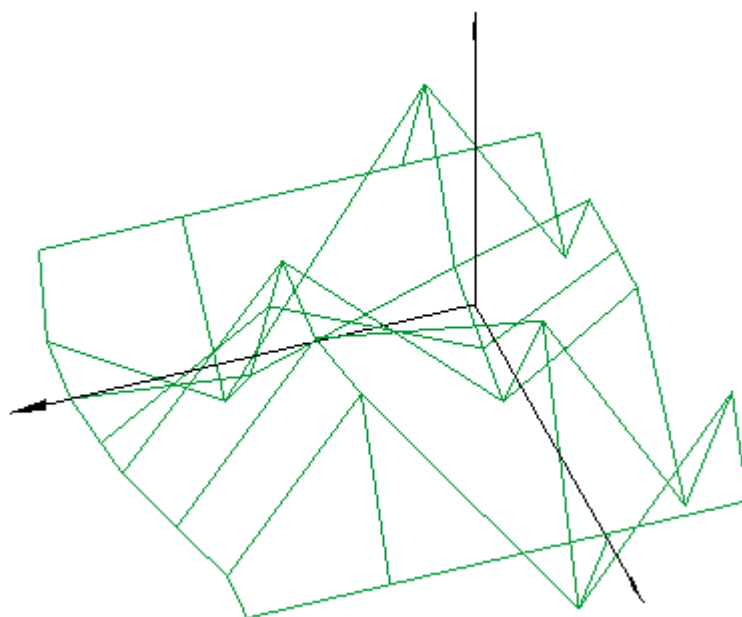


圖 4-17 形成 Skin Surface 的第三步驟，內插出控制點

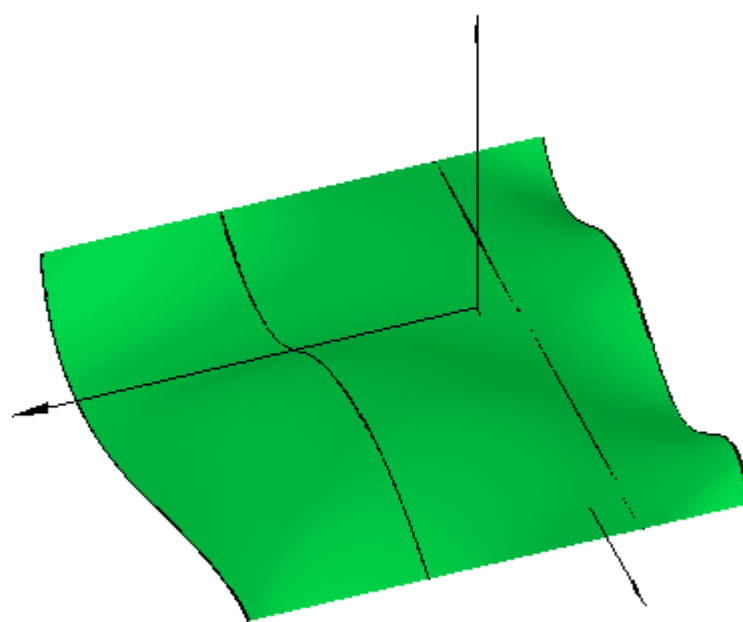


圖 4-18 Skin Surface

4-3-8 Swept Surface

所謂 Swept surface 是指一條截面曲線(section curve)沿著一條任意的路徑曲線(trjectory curve)，所掃出的一個曲面，它的數學表示式如下：

$$\mathbf{S}(u, v) = \mathbf{T}(v) + \mathbf{A}(v)\mathbf{C}(u) \quad (\text{Eq.4.20})$$

其中 $\mathbf{A}(v)$ 為一個 3×3 的矩陣，將 $\mathbf{C}(u)$ 作旋轉及縮放的轉換。

我們令 $\{\mathbf{O}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$ 來表示 global 座標系，且另外導入一個沿著路徑曲線 $\mathbf{T}(v)$ 的 local 座標系 $\{\mathbf{o}(v), \mathbf{x}(v), \mathbf{y}(v), \mathbf{z}(v)\}$ ，可以令

$$\begin{aligned} \mathbf{o}(v) &= \mathbf{T}(v) \\ \mathbf{x}(v) &= \frac{\mathbf{T}'(v)}{|\mathbf{T}'(v)|} \end{aligned} \quad (\text{Eq.4.21})$$

即 local 座標系的原點在路徑曲線 $\mathbf{T}(v)$ 上，而 x 軸為路徑曲線的切線方向，但還有一個方向(y 軸或 z 軸)尚未決定；所以我們使用 Frenet frame，假設 $\mathbf{T}(v)$ 為兩次可微分，則定義 $\mathbf{B}(v)$ 如下：

$$\mathbf{B}(v) = \frac{\mathbf{T}'(v) \times \mathbf{T}''(v)}{|\mathbf{T}'(v) \times \mathbf{T}''(v)|} \quad (\text{Eq.4.22})$$

且令

$$\mathbf{z}(v) = \frac{\mathbf{B}(v)}{|\mathbf{B}(v)|} \quad (\text{Eq.4.23})$$

(Eq.4.20) 中 $\mathbf{A}(v)$ 即為將 $\{\mathbf{O}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}\}$ 轉換為 $\{\mathbf{o}(v), \mathbf{x}(v), \mathbf{y}(v), \mathbf{z}(v)\}$ 的

轉換矩陣。

在 $T(v)$ 上面每一點的 local 座標系如下：

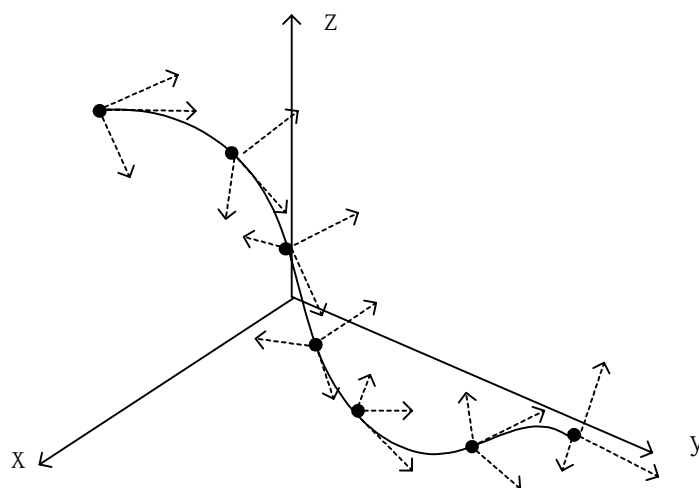


圖 4-19 路徑曲線上之 local 座標系

接下來有兩種方法可以求得我們所想要得到的 swept surface，第一種是利用(Eq.4.20)求得在 swept surface 上 $(n \times m)$ 個參考點，然後利用曲面內插或曲面逼近的方法，來得到通過這些參考點的曲面。

第二種方法，也是吾人所使用的方法，則是使用 skin surface 的方式，我們可以在路徑曲線 $T(v)$ 上取 n 個參考點，求得這 n 個參考點上相對應的 local 座標系後，將 $C(u)$ 做座標轉換後，可得到 n 條曲線，再將這 n 條曲線利用 skin surface 的方法求得一個曲面通過這 n 條曲線，即為我們所需要的 swept surface。

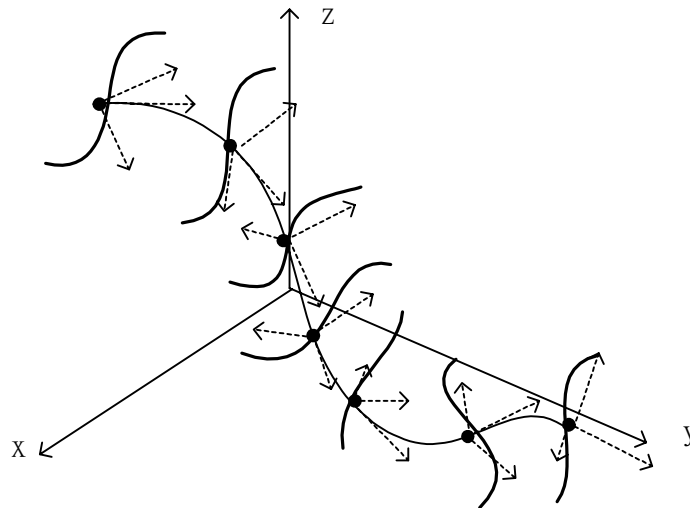


圖 4-20 對每一個 local coordinate 所形成的截面曲線

在求取 Swept surface 時也很容易遇到一些特殊的情形，例如退化(degenerate)和自我相交(selfintersection)的問題，所謂 degenerate 就是說，有一種情形是截面曲線與路徑曲線共平面，如此一來，所掃出的曲面便僅僅是一個平面；而自我相交的情形更是常發生，我們必須找出自我相交的部分，並對其做平滑化，詳細的說明可參考文獻 Martin and Stephenson[26]。

4-4 偏移曲面(Offset Surface)

偏移曲面在 CAD/CAM 應用系統中佔有重要的地位，除了薄殼特徵及在加工方面，刀具的運動軌跡之外，計算 blending surface 也與偏移曲面息息相關。但一直到現在，偏移曲面的求法一直沒有一個令人滿意的解決之道，主要在於，除了少數特殊曲面，如圓錐曲面，一個偏移曲面的數學表示法幾乎無法與 來曲面表示法一致，例如一個 3 階、控制點為 $(n+1) \times (m+1)$ 的 NURBS 曲面，它的偏移曲面就無法

完全正確地以 3 階、控制點為 $(n+1) \times (m+1)$ 的 NURBS 曲面表示。

4-4-1 一般偏移曲面的求法

偏移曲面的表示方法可以如下：

$$S^0(u, v) = S(u, v) + dN(u, v) \quad (\text{Eq.4.24})$$

其中 $S^0(u, v)$ 代表偏移曲面， $S(u, v)$ 為原來曲面， d 為偏移量， $N(u, v)$ 為原來曲面的法向量。

在本文中偏移曲面採用 Peigl, Tiller[23]所提出的方法，它是使用求得樣本點的方法，內插出偏移曲面，

$$S^0(u_i, v_i) = S(u_i, v_i) + dN(u_i, v_i) \quad i = 0, K, K+1 \text{ 為樣本點的個數。}$$

包括下面四個步驟：

1. 識別曲面，判斷其是否為特殊曲面。
2. 以曲面的二次導函數為基礎，求得偏移曲面的樣本點的座標及數目。
3. 使用內插曲面的方法建構出偏移曲面。
4. 在容許誤差範圍內，移除不必要的 knots。

詳細內容可參考文獻 Peigl [23]。

下面為兩個偏移曲面的例子：

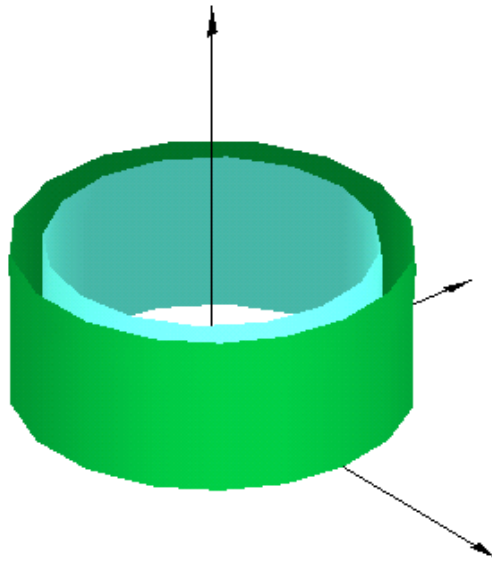


圖 4-21 Offset Surface 1

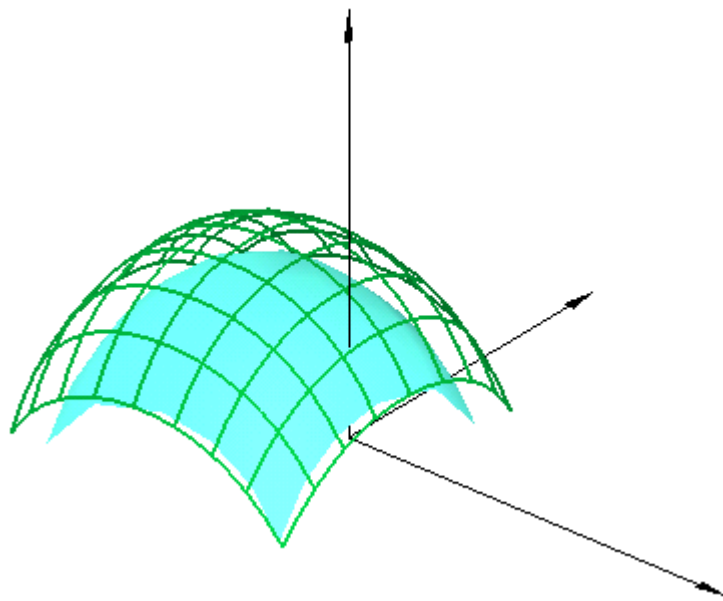


圖 4-22 Offset Surface 2

4-4-2 自我相交的探討

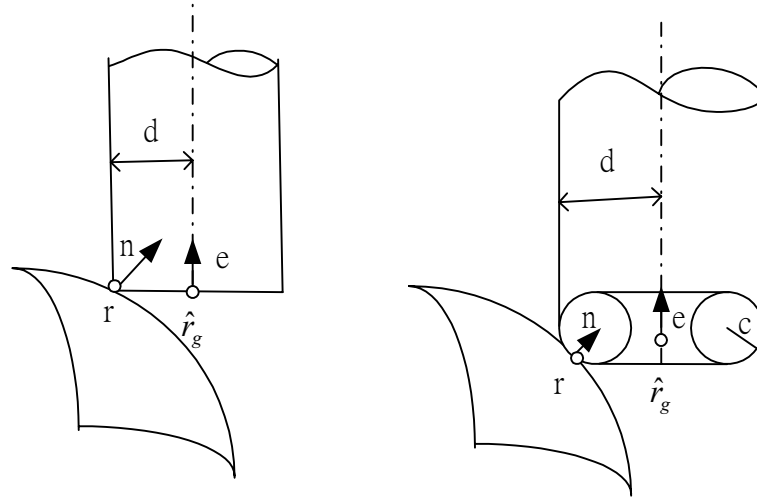
偏移曲面與在 3-4 節中所提到的偏移曲線相同，都有可能發生自我相交的可能性，都必須加以偵測並且排除，若一個偏移曲面發生自我相交，表示我們可以找到不相同的參數對 $(s, t) \neq (u, v)$ 使得

$$\mathbf{S}(s, t) + d \cdot \mathbf{N}(s, t) = \mathbf{S}(u, v) + d \cdot \mathbf{N}(u, v) \quad (\text{Eq.4.25})$$

Chen 和 Ravani[5]提出一個追蹤(marching)的法則來計算在偏移曲面上自我相交所形成的曲線，這個演算法會產生一條直線來近似自我相交所形成的曲線的一部分。

4-4-3 其它種類的曲面偏移

前面筆者所提及的曲面偏移，都是屬於直接以曲面的法向量做為偏移方向的曲面偏移，但在 3 軸的 NC 加工機中，很多並不是屬於這種所謂 ball-end cutter 所求出的曲面偏移，常常也有 cylindrical 及 fillet-end cutters 的曲面偏移。這種以在圓柱或圓角端刀具上的參考點移動的軌跡所形成的曲面，我們稱為一般偏移(general offsets)。這種偏移曲面第一次是由 Brechner[4]所提出。



上圖中左邊的刀具即為 cylindrical cutter，其軌跡所形成的偏移曲面的數學表示式可由下式來表示：

$$\hat{r}_g(u, v) = r(u, v) + d \frac{(\mathbf{e} \times \mathbf{n}(u, v)) \times \mathbf{e}}{|\mathbf{e} \times \mathbf{n}(u, v)|} \quad (\text{Eq.4.26})$$

其中 d : radius of the cutter

\mathbf{e} : unit vector along the tool axis

而上圖中右邊的刀具即為 toroidal cutter，其軌跡所形成的偏移曲面的數學表示式可由下式來表示：

$$\hat{r}_g(u, v) = r(u, v) + c\mathbf{n}(u, v) + (d - c) \frac{(\mathbf{e} \times \mathbf{n}(u, v)) \times \mathbf{e}}{|\mathbf{e} \times \mathbf{n}(u, v)|} \quad (\text{Eq.4.27})$$

其中 d : radius of the toroidal cutter

c : corner radius of the cutter

4-5 修剪曲面(Trimmed Surface)

Trimmed NURBS 曲面非常廣泛地應用在 CAD 軟體中，因為大部分複雜的物件都是由一連串修剪的動作組合而成，例如一個被 NURBS 曲面包圍的實體經過一些布林運算，就會產生一些 trimmed NURBS 曲面，然而在產生這些 trimmed 曲面後，接下來所要面對的問題，就是要如何將這些面顯示給使用者知道。最簡單的方法就是將這些 trimmed 曲面以三角形來近似，求出這些三角形的行為我們稱之為 "tessellate"，這個方法有下列幾點好處。

- 它與修剪曲面的複雜度較沒有關係，例如這個修剪曲面中間有較多的洞或沿著邊界剪裁，都不會影響到幾何計算的複雜度。
- 它可以產生單一的資料庫，例如這些三角形除了在顯示的時候使用外，還可以用在其它的計算，如刀具路徑或輪廓線。
- 我們操作三角形比直接操作 freeform 的曲面要來得容易且穩定。

但是它也有一些缺點：

- 為了適當地表示一個修剪曲面，若這個曲面的曲率較大，我們就必須使用較多數量的三角形。
- 在形成三角形時，若沒有處理適當，所產生的三角形可能會大小不一，且可能會導致數值上的問題。

Piegl[32]和 William[38]都對修剪曲線的 tessellation 的方法提出他們的演算法，Piegl 的 tessellation 方法的步驟如下，詳細可參考[32]：

- 步驟一：計算出這個曲面參數域(parameter domain)中最長邊的尺寸。

- 步驟二：求得參數域中剪裁曲線的近似多邊形。
- 步驟三：選擇在有效區域中的一點。
- 步驟四：將剪裁區域三角形化。
- 步驟五：將這個參數域中的三角形映射到曲面上，並建立的 3D 三角形的資料庫，使得我們可以做進一步的應用。

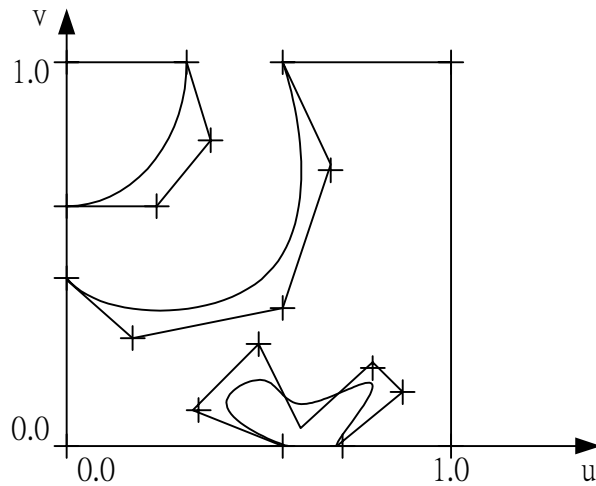


圖 4-23 Trimmed domain using cubic NURBS curve

第五章 幾何交集之探討與實作

5-1 前言

幾何的的交集在 CAD 系統中應用的相當頻繁，也是影響 CAD 系統速度及穩定性的重要因素，本章會分別介紹線與線，線與面及面與面交集的求法，而其中又以曲面與曲面的交集尤為重要，也最常被學術界所探討，而現在最為常用的解法，為 Barnhill[1][2]所提出的追蹤法。

5-2 線與線的交集計算

假設空間中有兩條曲線 $C_1(u)$ 及 $C_2(v)$ ，如下圖，則它們之間的交點可經由解下面的方程式求得。

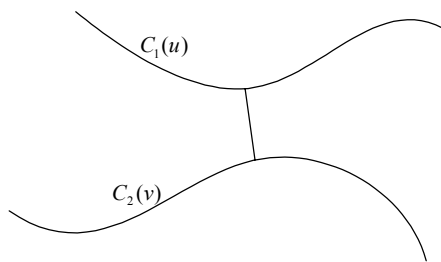


圖 5-1 兩曲線之最短距離

$$\begin{aligned}(C_1(u) - C_2(v)) \cdot C_1'(u) &= 0 \\ (C_1(u) - C_2(v)) \cdot C_2'(v) &= 0\end{aligned}\tag{Eq.5.1}$$

(Eq.5.1)式求得之解為 $C_1(u)$ 與 $C_2(v)$ 的交點，或兩條直線相距最近的點。所以求得解後，必須檢驗兩點之間的距離是否小於空間同點公差(SPPT)，若小於空間同點公差，則加入解串列。

但解上面的方程式可能有好幾個交點，一般的求解方法只能求出一個解，所以有幾種方法可以解決，如區間牛頓法陳源芳[41]，細分法劉志鴻[43]等。區間牛頓法的強韌性仍然有待討論，所以筆者採用細分法，且如果曲線為一個 NURBS 曲線，則可以利用它的 convex hull 的特性，使用細分法會比較快速。

5-2-1 程式求解步驟

以下為線與線求交點的程式求解步驟：

1. 輸入兩條參數曲線 $C_1(u)$ 及 $C_2(v)$ 。
2. 將 $C_1(u)$ 及 $C_2(v)$ 細分，利用上面公式，分別求出交點。
3. 檢查兩條曲線上的分別對應的點，是否符合空間同點公差的設定，若符合則加入求解串列。
4. 將求解串列中的點進行 merge 的動作，以去除重複的點。

5. 最後求解串列中的點即為真正的交點。

5-2-2 實例測試

1. 兩曲線交點相鄰極近：

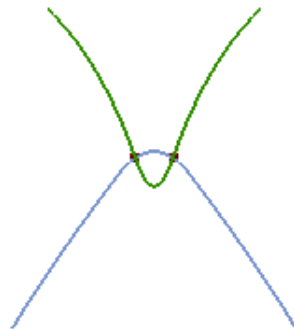


圖 5-2 兩曲線交點相鄰極近

2. 兩曲線相切：

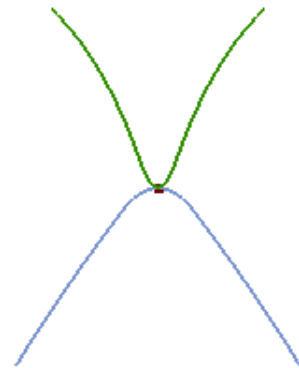


圖 5-3 兩曲線相切

5-3 線與面的交集計算

5-3-1 曲線與平面的交點

一個平面的方程式可以表示如下：

$$P \cdot N = d \quad \text{with } |N| = 1 \quad (\text{Eq.5.2})$$

其中

$P = (x, y, z)$: 座標值

$N = (a, b, c)$: 平面 π 的法向量

若我們要求解曲線 $C(u)$ 與平面 π 的交集的話，必須求解以下的方程式：

$$C(u) \cdot N - d = 0 \quad (\text{Eq.5.3})$$

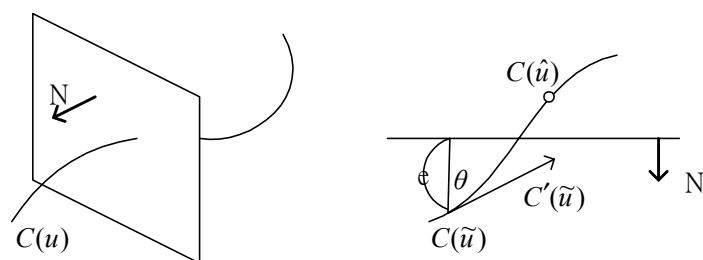


圖 5-4 曲線與平面交點示意圖

我們比較常用的解法是使用牛頓法疊代來求解，可先由曲線上的一個猜測的初始值 $C(\tilde{u})$ 開始，由於：

$$\begin{aligned} |C'(\tilde{u})| \cdot (\hat{u} - \tilde{u}) &\cong e / \cos \theta \\ \text{其中, } e &= (N \cdot C(\tilde{u}) - d) \\ \cos \theta &= (C'(\tilde{u}) \cdot N) / |C'(\tilde{u})| \end{aligned} \quad (\text{Eq.5.4})$$

所以可利用上式推導出下一個估計參數 \tilde{u} 如下：

$$\tilde{u} = \hat{u} - \{(N \cdot C(\hat{u}) - d) / (C'(\hat{u}) \cdot N)\} \quad (\text{Eq.5.5})$$

可由上式一直疊代到 $|C(\hat{u}) \cdot N - d| \leq \varepsilon$ 即可停止。

而找出初始值的方法筆者也是採用細分法，可將曲線 $C(u)$ 依參數等分，則可得參數序列 $\{u_i\}$ ，如果下面方程式成立，

$$[N \cdot C(u_i) - d] \cdot [N \cdot C(u_{i+1}) - d] < 0 \quad (\text{Eq.5.6})$$

，表式區間 $[u_i, u_{i+1}]$ 有解，則可令初始值 $\tilde{u} = (u_i + u_{i+1}) / 2$ 。

求解的步驟如下：

1. 輸入曲線及平面。
2. 利用(Eq.4.6)求得初始值 \tilde{u} 。
3. 由(Eq.4.5)式進行牛頓法疊代來求得下一個估計參數 \tilde{u} 。

4. 如果 $|C(\hat{u}) \cdot N - d| \leq \varepsilon$ 則傳回交點參數，否則回到步驟 3。

5-3-2 曲線與曲面的交點

若有一條曲線 $C(w)$ 與一個曲面 $S(u, v)$ 要求它們的交點，則需要求解下面的方程式：

$$\begin{aligned} \mathbf{f}(u, v, w) &= \mathbf{S}(u, v) - \mathbf{C}(w) = 0 \\ \Rightarrow \begin{cases} f_x(u, v, w) = 0 \\ f_y(u, v, w) = 0 \\ f_z(u, v, w) = 0 \end{cases} \end{aligned} \quad (\text{Eq.5.7})$$

但同樣地，上述方程式可能有很多解，所以筆者仍然採用細分法先求出初始值，然後以初始值代入(Eq.5.7)分別求出多個交點。

初始值的求法可參考文獻 Owen [30]，它將曲線以直線的近似線段來逼近，而曲面則以近似三角形平面網逼近，如圖 5-5：

則三角形平面的方程式可表示如下：

$$\begin{aligned} T(u', v') &= (1 - v')(a \cdot u' + b) + v' \cdot c \\ \text{其中 } 0 \leq u' \leq 1 \text{ 且 } 0 \leq v' \leq 1 \end{aligned} \quad (\text{Eq.5.8})$$

直線段方程式可表示如下：

$$\begin{aligned} L(w') &= dw' + e \\ \text{其中 } 0 \leq w' \leq 1 \end{aligned} \quad (\text{Eq.5.8})$$

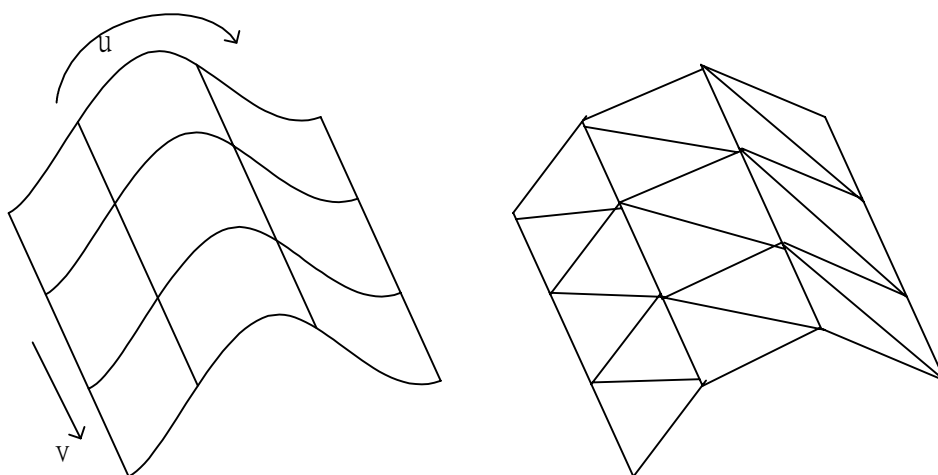


圖 5-5 將曲面分割為三角形

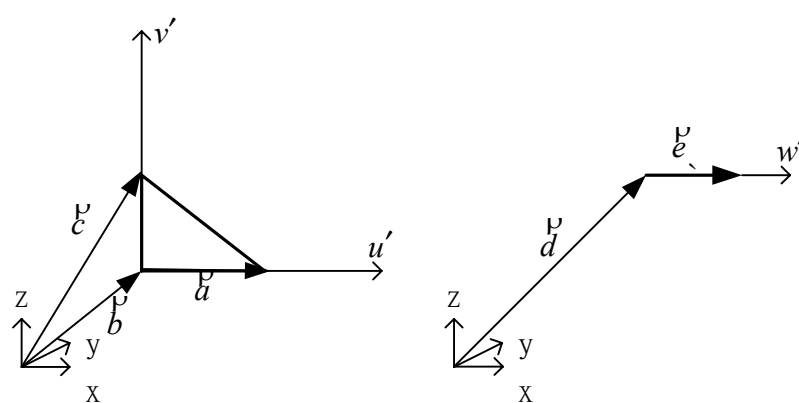


圖 5-6 三角形向量與直線向量

假設兩者相交，即 $T(u', v') = L(w')$ ，則可推導出如下之交點參數

$$\begin{aligned}
 w' &= \frac{[d \times (e - b)] \cdot b - [d \times (e - b)] \cdot e}{[d \times (e - b)] \cdot d} \\
 v' &= \frac{(d \times d) \cdot e - (d \times d) \cdot b}{(d \times d) \cdot (e - b)} \\
 u' &= \frac{[d \times (e - b)] \cdot e - [d \times (e - b)] \cdot b}{[d \times (e - b)] \cdot d(1 - v')}
 \end{aligned} \tag{Eq.5.8}$$

這些計算出來的參數值若轉換成曲線或曲面中真正的參數值，就可以當做求解(Eq.5.7)的初始值。

求解的步驟如下：

1. 輸入曲線及曲面。
2. 將曲線與曲面細分，利用(Eq.5.8)求出初始值。
3. 以步驟 2 求得之初始值求解(Eq.5.7)之非線性聯立方程組。
4. 對步驟 3 所得之解串列進行 merge 的動作，以去除重複的解。
5. 最後之解串列即為曲線與曲面的交點。

5-3-3 實例測試

1. 曲線與平面交於多個點：

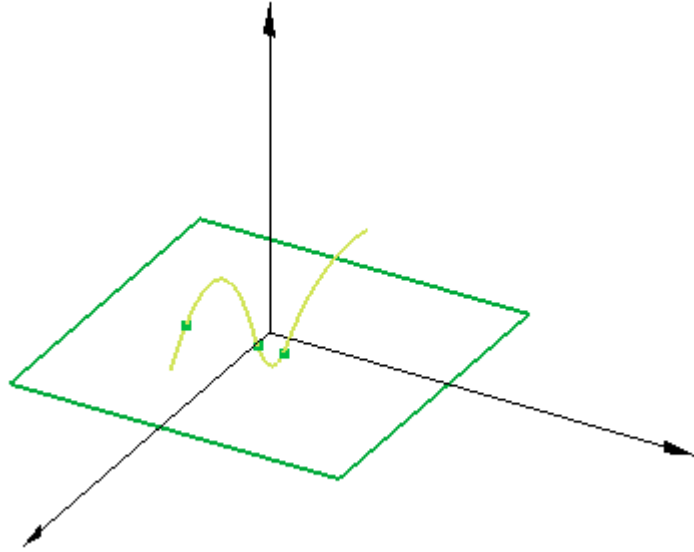


圖 5-7 曲線與平面交於多個點

2. 直線與環狀曲面相切於兩點：

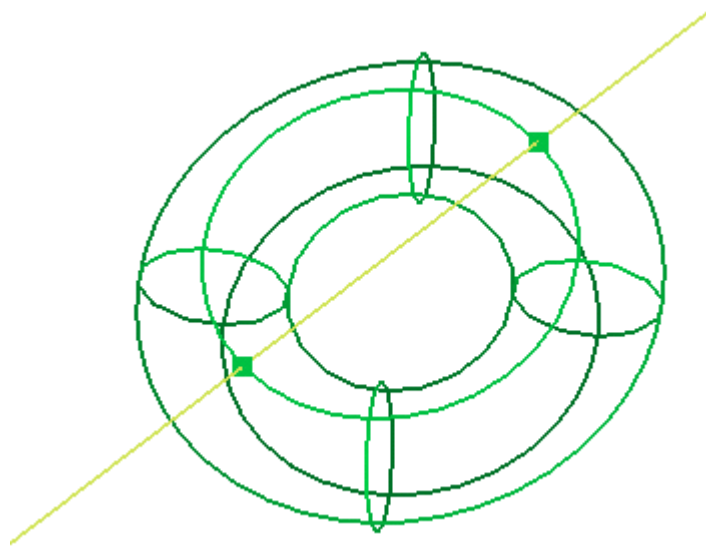


圖 5-8 直線與環狀曲面相切於兩點

5-4 面與面的交集計算

曲面與曲面的交集主要是要符合下面幾點的要求：

1. 必須是精確並且穩定的。
2. 必須預防曲面交集分支的遺漏。
3. 必須要有效率的。

為了達到 2.和 3.的目的，我們必須先使用曲面的控制多邊形來近似原來的曲面，來快速地找到我們所使用的追蹤法的起點，為了達到 1.的目的，我們使用牛頓法來求取追蹤時的下一個精確點。整個曲面交集的演算法可由下圖來表示。

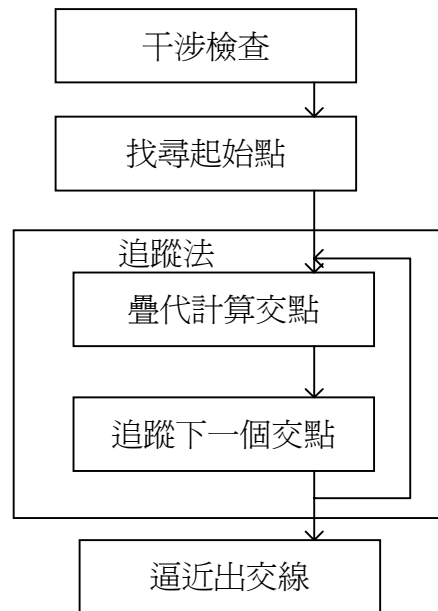


圖 5-9 曲面交集之演算法

其中干涉檢查通常是將曲面 subdivision 成好幾塊 Bézier 曲面塊，再利用 Bézier 曲面控制多邊形來檢驗兩曲面是否有交點，及交點的初始值大約在何處，詳細可參考 Barnhill[1]。

雖說以上的法則適用於任何曲面的交集，但筆者為求效率，還是將曲面交集分為曲面與平面的交集與曲面與曲面的交集兩種，以下將分別討論這種交集的解法。

5-4-1 曲面與平面的交集

曲面 $S(u, v)$ 與平面的交線可以用下面的方程式來表示：

$$\begin{aligned} 0 &= \mathbf{S}(u, v) \cdot \mathbf{P} - d \\ &= a \cdot x(u, v) + b \cdot y(u, v) + c \cdot z(u, v) - d \end{aligned} \quad (\text{Eq.5.9})$$

但我們無法正確地解出(Eq.4.9)，所以必須使用追蹤法，求出交線上的部分點集合，再將這些點串列 Fit 出近似的交線。

下面介紹求取交線的步驟：

1. 求得追蹤的起始點

追蹤的起始點可分為兩種，若交線不為一個封閉曲線，則在曲面的邊界上一定可以找到起始點，故可以對曲面的四條邊界分別對平面求取交點，所求之交點即為起始點；若交線為一條封閉曲線，則曲面的邊界並不會與平面有交點，如下圖，所以必須另外求得起始點。

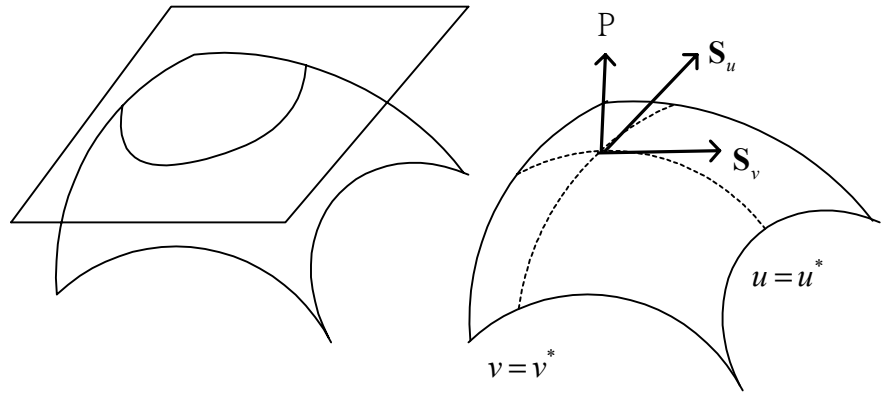


圖 5-10 封閉交線的起始點

求得封閉交線的起始點可參考文獻 Choi [7]，他提出這類的起始點即在找出曲面上某一點，它的法向量與平面的法向量一致，即求得曲面上一點使其符合下面之方程式：

$$\begin{aligned} \mathbf{P} \cdot \mathbf{S}_u &= 0 \\ \mathbf{P} \cdot \mathbf{S}_v &= 0 \end{aligned} \quad (\text{Eq.5.10})$$

其中 \mathbf{P} ：平面的法向量。

\mathbf{S}_u ：曲面在起始點對 u 方向的偏微分。

\mathbf{S}_v ：曲面在起始點對 V 方向的偏微分。

則可以利用(Eq.5.10)來求得封閉交線的起始點。

2. 決定追蹤方向

追蹤的方向一定是與曲面和平面都垂直的向量，即

$$\mathbf{C} = \pm(\mathbf{N} \times \mathbf{P}) / |\mathbf{N} \times \mathbf{P}| \quad (\text{Eq.5.11})$$

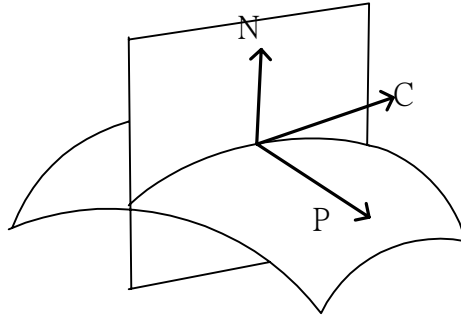


圖 5-11 平面與曲面交線追蹤方向

3. 決定追蹤增量

追蹤增量的求解方法可利用下面之線性方程式求得：

$$\begin{aligned} \Delta u(\mathbf{P} \cdot \mathbf{S}_u) + \Delta u(\mathbf{P} \cdot \mathbf{S}_v) &= 0 \\ \Delta u(\mathbf{C} \cdot \mathbf{S}_u) + \Delta u(\mathbf{C} \cdot \mathbf{S}_v) &= \delta \end{aligned} \quad (\text{Eq.5.12})$$

其中 δ 代表前進步幅，它是影響追蹤法效率及強韌性的重要因素，在下一節會說明它的求法。

4. 取得下一點曲面參數 $\hat{u} = u_0 + \Delta u$ 及 $\hat{v} = v_0 + \Delta v$ ，重覆步驟 2、步驟 3，直到達到曲面的邊界。

5-4-2 曲面與曲面的交集

曲面與曲面的交集求法大致上來說，與平面與曲面的交集求法相類似，它們的主要步驟都是：

1. 求取起始點(邊界點或關鍵點)。
2. 求取追蹤向量(包括追蹤步幅)。
3. 求得在曲面上真正的交點。
4. 重覆步驟 2,3，直到追蹤至曲面邊界。

而兩種求取交集最大的不同點在步驟 1 及步驟 3，平面與曲面求交集的方法已在上一節說明，關鍵點的求法可使用(Eq.5.10)，求得曲面上真正的交點可使用(Eq.5.12)，接下來筆者就要介紹曲面與曲面求解交集時，關鍵點的計算方法，及求得曲面上真正交點的方法，還有追蹤步幅的求法。

A. 關鍵點的求法：

關鍵點的求法由文獻郭錦誠[42]可得知，關鍵點必須符合下面的四個方程式：

$$\begin{aligned}
f_1 &= (\mathbf{P} - \mathbf{Q}) \cdot \mathbf{P}_u = 0 \\
f_2 &= (\mathbf{P} - \mathbf{Q}) \cdot \mathbf{P}_v = 0 \\
f_3 &= (\mathbf{P}_u \times \mathbf{P}_v) \cdot \mathbf{Q}_w = 0 \\
f_4 &= (\mathbf{P}_u \times \mathbf{P}_v) \cdot \mathbf{Q}_s = 0
\end{aligned}
\tag{Eq.5.13}$$

其中 \mathbf{P} 與 \mathbf{Q} 分別代表兩曲面 $S_1(u, v)$ 與 $S_2(w, s)$ 上的點，第一、二個方程式表示 $\overline{\mathbf{PQ}}$ 必須與曲面 $S_1(u, v)$ 的法向量平行，第三、四個方程式表示 $S_1(u, v)$ 在 \mathbf{P} 的法向量必須與 $S_2(w, s)$ 在 \mathbf{Q} 的法向量平行。可由上述四個方程式解出 u, v, w, s 四個未知數，接著可作一條 $S_1(u, v)$ 過 \mathbf{P} 點的等 u 曲線，若此等 u 曲線與曲面 $S_2(w, s)$ 有交點，則此交點可做為追蹤法的起始點。

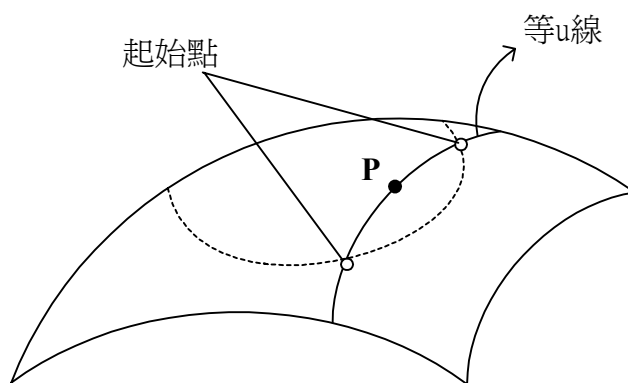


圖 5-12 利用關鍵點之等 u 線求得封閉交線之起始點

B.求得曲面上真正交點(refine point)

當在追蹤交點時，可由一個實際的交點，加上追蹤向量，則可得下一個近似的交點 \mathbf{A} ，我們必須把近似的交點收斂至真正的交線上，此時分為兩個步驟，第一個步驟是將近似交點分別投影在 \mathbf{S}_1 及 \mathbf{S}_2 上，投影的方法可參考 You [9]，所使用到之公式如下：

$$\begin{aligned}
 (i) \quad & \text{for } (u, v) = (u_i, v_i) \quad \mathbf{r}_i = \mathbf{S}_1(u_i, v_i) - \mathbf{A} \\
 & u_{i+1} = u_i + \frac{(\mathbf{S}_{1v} \times \mathbf{r})_i \cdot (\mathbf{S}_{1u} \times \mathbf{S}_{1v})_i}{(\mathbf{S}_{1u} \times \mathbf{S}_{1v})_i \cdot (\mathbf{S}_{1u} \times \mathbf{S}_{1v})_i} \\
 & v_{i+1} = v_i - \frac{(\mathbf{S}_{1u} \times \mathbf{r})_i \cdot (\mathbf{S}_{1u} \times \mathbf{S}_{1v})_i}{(\mathbf{S}_{1u} \times \mathbf{S}_{1v})_i \cdot (\mathbf{S}_{1u} \times \mathbf{S}_{1v})_i} \\
 & \text{until } |\mathbf{r}_{i+1} - \mathbf{r}_i| < SSPT
 \end{aligned} \tag{Eq.5.14}$$

$$\begin{aligned}
 (ii) \quad & \text{for } (w, s) = (w_i, s_i) \quad \mathbf{r}_i = \mathbf{S}_2(w_i, s_i) - \mathbf{A} \\
 & w_{i+1} = w_i + \frac{(\mathbf{S}_{2s} \times \mathbf{r})_i \cdot (\mathbf{S}_{2w} \times \mathbf{S}_{2s})_i}{(\mathbf{S}_{2w} \times \mathbf{S}_{2s})_i \cdot (\mathbf{S}_{2w} \times \mathbf{S}_{2s})_i} \\
 & s_{i+1} = s_i - \frac{(\mathbf{S}_{2w} \times \mathbf{r})_i \cdot (\mathbf{S}_{2w} \times \mathbf{S}_{2s})_i}{(\mathbf{S}_{2w} \times \mathbf{S}_{2s})_i \cdot (\mathbf{S}_{2w} \times \mathbf{S}_{2s})_i} \\
 & \text{until } |\mathbf{r}_{i+1} - \mathbf{r}_i| < SSPT
 \end{aligned} \tag{Eq.5.15}$$

第二個步驟是利用牛頓法收斂交點，我們已在前一個步驟求得四個初始的曲面參數 $[\hat{u}, \hat{v}, \hat{w}, \hat{s}]$ ，但它仍然不是真正的交點，但我們已知曲面與曲面的交點符合下面的方程式：

$$\mathbf{S}_1(u, v) - \mathbf{S}_2(w, s) = 0 \tag{Eq.5.16}$$

上式方程式有四個未知數 $[u, v, w, s]$ ，但我們只有三個方程式 (x, y, z 三個方向向量)，所以我們採用固定 u 或固定 v 這個未知數，使得 $u = \hat{u}$ ，或 $v = \hat{v}$ ，且以另外三個數 $\hat{v}, \hat{w}, \hat{s}$ 或 $\hat{u}, \hat{w}, \hat{s}$ 作為求解牛頓法的初始值，則可以解得另外三個未知數。

以下為收斂交點的公式：

$$\begin{aligned}
(i) \mathbf{r}_i &= \mathbf{S}_1(u_i, v_i) - \mathbf{S}_2(w_i, s_i) \\
\mathbf{D}_u &= [\mathbf{S}_{1v} \cdot (\mathbf{S}_{2w} \times \mathbf{S}_{2s})]_i \\
u_{i+1} &= \hat{u} \\
v_{i+1} &= v_i - \frac{\mathbf{S}_{2w} \cdot (\mathbf{S}_{2s} \times \mathbf{r})_i}{\mathbf{D}_u} \\
w_{i+1} &= w_i - \frac{\mathbf{S}_{2s} \cdot (\mathbf{S}_{1v} \times \mathbf{r})_i}{\mathbf{D}_u} \\
s_{i+1} &= s_i - \frac{\mathbf{S}_{1v} \cdot (\mathbf{S}_{2w} \times \mathbf{r})_i}{\mathbf{D}_u} \\
\text{until } |v_{i+1} - v_i| &< PSPT \\
|w_{i+1} - w_i| &< PSPT \\
|s_{i+1} - s_i| &< PSPT
\end{aligned} \tag{Eq.5.17}$$

$$\begin{aligned}
(i) \mathbf{r}_i &= \mathbf{S}_1(u_i, v_i) - \mathbf{S}_2(w_i, s_i) \\
\mathbf{D}_v &= [\mathbf{S}_{1u} \cdot (\mathbf{S}_{2w} \times \mathbf{S}_{2s})]_i \\
v_{i+1} &= \hat{v} \\
u_{i+1} &= u_i - \frac{\mathbf{S}_{2w} \cdot (\mathbf{S}_{2s} \times \mathbf{r})_i}{\mathbf{D}_v} \\
w_{i+1} &= w_i + \frac{\mathbf{S}_{2s} \cdot (\mathbf{S}_{1u} \times \mathbf{r})_i}{\mathbf{D}_v} \\
s_{i+1} &= s_i - \frac{\mathbf{S}_{1u} \cdot (\mathbf{S}_{2w} \times \mathbf{r})_i}{\mathbf{D}_v} \\
\text{until } |u_{i+1} - u_i| &< PSPT \\
|w_{i+1} - w_i| &< PSPT \\
|s_{i+1} - s_i| &< PSPT
\end{aligned} \tag{Eq.5.18}$$

C.求得追蹤步幅

追蹤步幅對追蹤法的效率影響甚巨，如果步幅太大，可能會導致無法收斂，且求得的點串列可能無法精確地表示交

線，若步幅太小，則會增加求解的時間。筆者使用高斯曲率來決定追蹤步幅的大小，若曲率愈大，則步幅必須較小，高斯曲率的求法可參考文獻 David [11]，如下所示：

$$K = \frac{\mathbf{A} \cdot \mathbf{C} - \mathbf{B}^2}{|\mathbf{S}_u \times \mathbf{S}_v|^4} \quad (\text{Eq.5.19})$$

其中

$$\mathbf{A} = [\mathbf{S}_u \times \mathbf{S}_v] \cdot \mathbf{S}_{uu}$$

$$\mathbf{B} = [\mathbf{S}_u \times \mathbf{S}_v] \cdot \mathbf{S}_{uv}$$

$$\mathbf{C} = [\mathbf{S}_u \times \mathbf{S}_v] \cdot \mathbf{S}_{vv}$$

筆者先求出一個參考步幅，再依據高斯曲率的大小，適當地調整步幅的大小，參考步幅即為曲面的平均長度。

5-4-3 實例測試

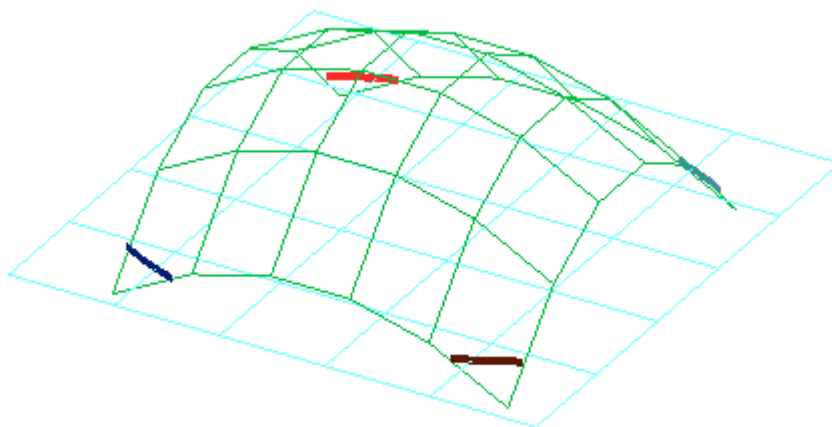


圖 5-13 平面與曲面交出四條曲線

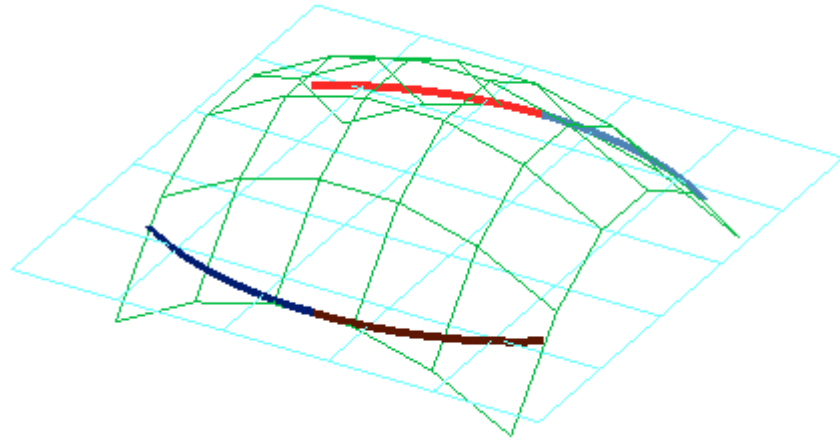


圖 5-14 平面與曲面交出兩條曲線

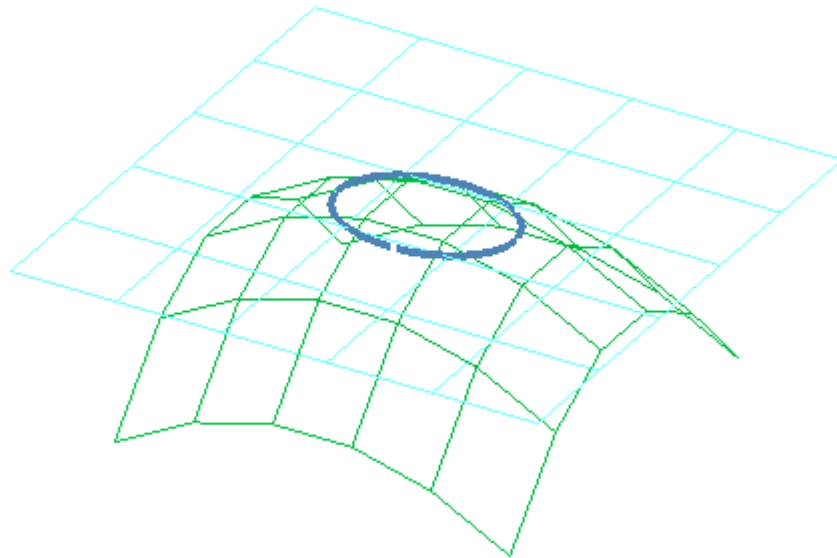


圖 5-15 平面與曲面交出一條封閉曲線

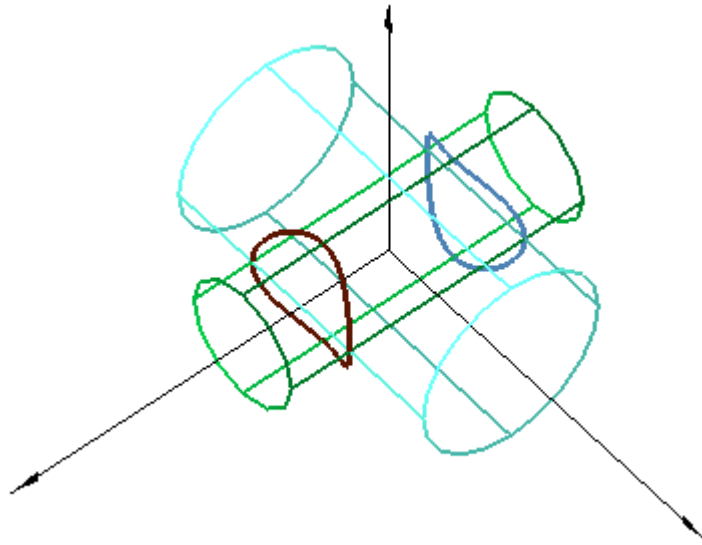


圖 5-16 兩圓柱面交出兩條曲線

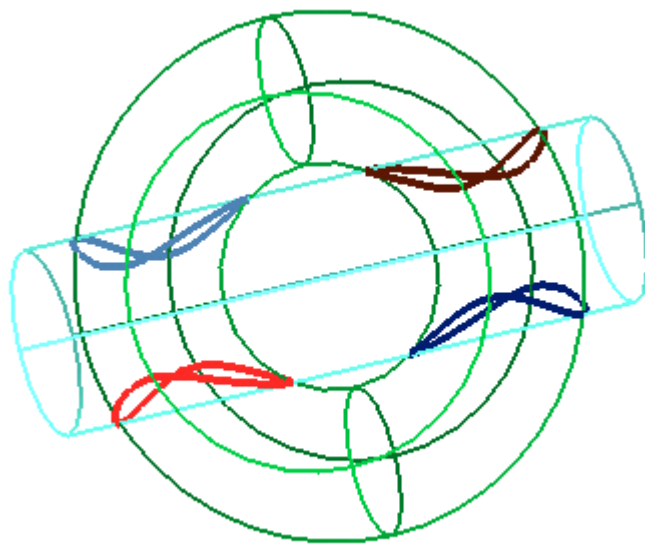


圖 5-17 圓柱面與環面交出四條曲線

5-5 曲面交集的討論

在幾何核心中，交集的計算佔了一個很大的份量，曲面交集的算法大抵可以分為解析法及數值法，解析法企圖利用求解方程式的根來求得精確的解，可惜這種方法有很多曲面不適用，因為有很多曲面可能無法以代數方程式來表示；而數值法，如細分法或追蹤法，則不須考慮曲面的代數方程式，即可求得逼近的解，但這種方法可能會由於分割的不夠細密，而遺失了小迴圈的解。在近幾年，有很多關於曲面交集這方面的研究，一般性的曲面交集問題常常使用網格法，Barnhill[2]、Barnhill and Kersey[1]、Sederberg[12]及Farouki[14]等人都有提出解決方案。而追蹤法則在最近被用在求取曲面交線的研究，追蹤法的準確性已可以靠適當地選取追蹤步幅來提升。而 Owen and Rockwood[30]則分析了奇異點的問題。

追蹤法所衍生的另一項課題，是起始點的求取與小迴圈的偵測。Muellenheim[29]提出了一種迭代的方法來計算起始點，而起始點也可以使用網格法及細分法來求得。Cheng[6]提出了一種不同的方法來偵測小迴圈，他使用兩個曲面之間有方向性的距離的梯度，來定義一個平面向量場，小迴圈的偵測則是找出在這個平面向量場中所有的臨界點(critical point)，因為這些臨界點可能就是在一個小迴圈的內部。

第六章 曲面混成之探討

6-1 前言

曲面混成即產生所謂的 blending surface，亦即在相鄰的兩個曲面之間，產生一個平滑連接兩曲面的曲面，這個動作被稱為”rounding”或”filleting”，它主要的作用在增加美觀程度，防止應力集中，加工容易等。

這一章介紹一種常用的方法來求得 blending surface，這是 Choi[7]所提出的”rolling-ball”的概念，它可以經由下面四個步驟來得到固定半徑的 blending surface。

1. 建構這兩個曲面的偏移曲面。
2. 找出這兩個偏移曲面的交線，以決定圓心軌跡。
3. 建構與兩曲面相切的圓錐曲線。
4. 利用步驟三的圓錐曲線掃出 blending surface。

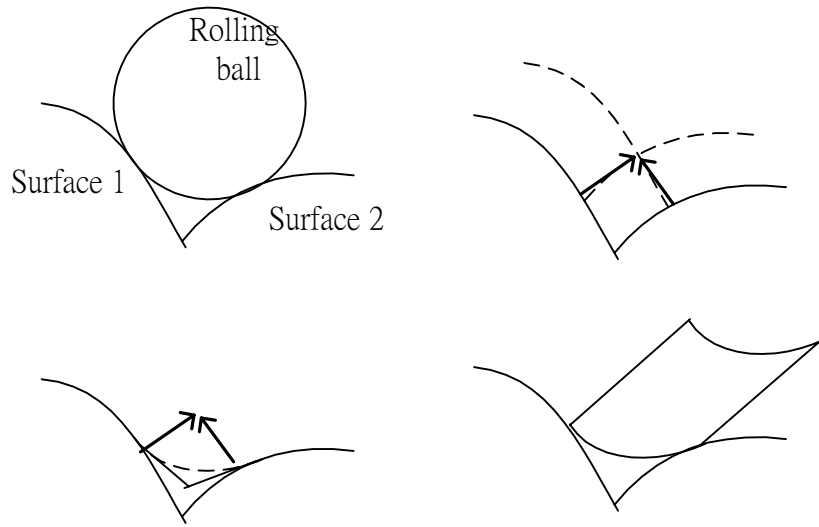


圖 6-1 “rolling-ball”求得 blending surface 示意圖

6-2 求出偏移曲面的交線

在此計算偏移曲面的交線有兩種方法，第一種方法是先求出如上一節所述之偏移曲面，在利用上一章求解曲線曲面交線的方法，計算出偏移曲面的交線，這一種方法比較費時，且求取偏移曲面時可能會有誤差產生，但求解交線的 Solver 可直接套用上一章所述面與面交集的解法。

第二種方法是直接以偏移曲面的數學表示式來計算交線，其表示方程式如下：

$$\begin{aligned} f^0(r,s) &= f(r,s) + dm(r,s) \quad \text{for } r,s \in [0,1] \\ g^0(t,u) &= g(t,u) + dn(t,u) \quad \text{for } t,u \in [0,1] \end{aligned} \quad (\text{Eq.6.1})$$

若欲求解上面兩個偏移曲面的交線，必須知道上面兩個偏移曲面的一次及二次導函數。而其導函數的公式如下，可參考Farouki[14]：

$$\begin{aligned}
 f_r^0 &\equiv \partial f^0(r, s) / \partial r \\
 &= \frac{\partial}{\partial r} (f(r, s) + dm(r, s)) \\
 &= f_r + d \left[\frac{E_r}{|E|} - \frac{E \cdot E_r}{|E|^3} E \right]
 \end{aligned} \tag{Eq.6.2}$$

其中

$$\begin{aligned}
 E &= f_r \times f_s \\
 E_r &= f_{rr} \times f_s + f_r \times f_{rs} \\
 f_r &= \partial f(r, s) / \partial r
 \end{aligned}$$

6-3 建構與兩曲面相切的圓錐曲線

當算出兩偏移曲面的交線後，必定會產生一個此交線上的點串列，這些點可以用下面的方程式來表示：

$$\begin{aligned}
 O &= f(r_j, s_j) + dm(r_j, s_j) \\
 &= g(t_j, u_j) + dn(t_j, u_j)
 \end{aligned} \tag{Eq.6.3}$$

而交線上的點串列在兩個曲面上的投影，即”rolling-ball”接觸兩個曲面的點為：

$$\begin{aligned}
 C_0 &= f(r_j, s_j) \\
 C_2 &= g(t_j, u_j)
 \end{aligned} \tag{Eq.6.4}$$

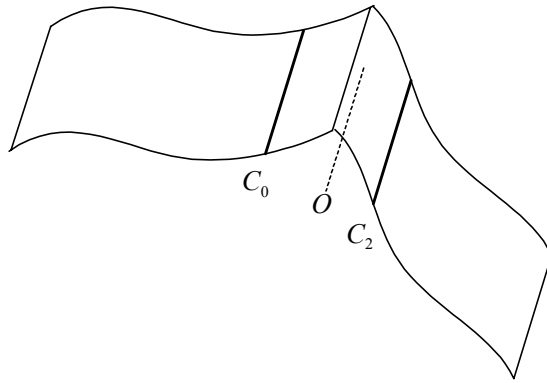


圖 6-2 偏移曲面交線

我們先建構一個截面通過 O, C_0, C_2 這三個點，在截面上可求得 C_1 如下圖：

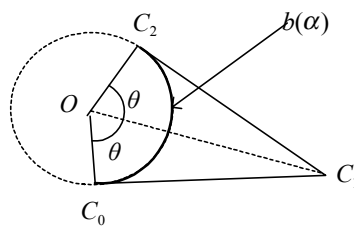


圖 6-3 求得 blending surface 截面之圓錐曲線

上圖中 C_1 與 C_0, C_2 的關係如下：

$$C_1 = \frac{1}{2}(C_0 + C_2) + \frac{d}{2} \frac{m \cdot n - 1}{m \cdot n + 1} (m + n) \quad (\text{Eq.6.5})$$

則 $b(\alpha)$ 可以 degree=2 的 NURBS 曲線表示如下：

$$b(\alpha) = \frac{w_0(1-\alpha)^2 C_0 + 2w_1(1-\alpha)\alpha C_1 + w_2\alpha^2 C_2}{w_0(1-\alpha)^2 + 2w_1(1-\alpha)\alpha + w_2\alpha^2} \quad (\text{Eq.6.6})$$

其中 $w_0 = w_2 = 1, w_1 = \cos \theta$ 。

6-4 圓錐曲線的掃出

我們針對每一個偏移曲面交線的點串列上的每一個點，分別計算出相對應的圓錐曲線，如下圖：

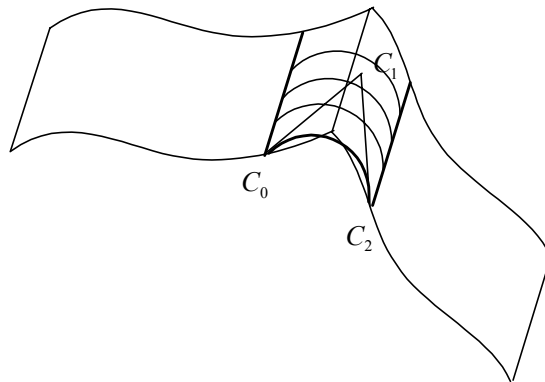


圖 6-4 blending surface 示意圖

接著可針對控制點 $([C0_i],[C1_i],[C2_i])$ ，其對應的 weight 為 $(1,\cos\theta_i,1)$ ，使用類似 skin surface 的方法，近似出 blending surface。

6-5 Corner blending

這一節主要是說明建構一個”Corner blending”的程序及方法，首先我先以三個曲面的 blending 作討論，最後會將其擴展為一般多個曲面混成的方法。

假設我們要做 $f(r,s)$ 、 $g(t,u)$ 及 $h(v,w)$ 這三個面的混成，而這三個曲面的偏移曲面可分別表示如下：

$$f^0(r,s) = f(r,s) + d \cdot m(r,s)$$

$$g^0(t,u) = g(t,u) + d \cdot n(t,u)$$

$$h^0(v,w) = h(v,w) + d \cdot o(v,w)$$

我們必須對這三個曲面分別求得其兩兩之間的 Edge blending，可以得到下面三組偏移曲面交線的解集合的 Domain 值：

$$\{(r_i, s_i), (t_i, u_i)\} : f(r,s) \& g(t,u) \text{ 之交點串列}$$

$$\{(t_j, u_j), (v_j, w_j)\} : g(t,u) \& h(v,w) \text{ 之交點串列}$$

$$\{(v_k, w_k), (r_k, s_k)\} : h(v,w) \& f(r,s) \text{ 之交點串列}$$

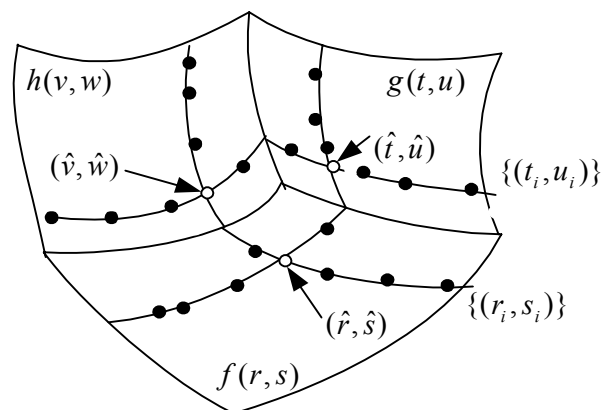


圖 6-5 在角落附近之交集點的 Domain 值

所以若要建構 corner blending，必須有下面三個步驟：

- 求出 DIC-points(Domain Intersection Corner points)。
- 建構 Corner blending 的邊界曲線。
- 建構 Corner blending patch。

6-5-1 求出 DIC-points

我們可以使用之前所提到的 edge blending 的方法，對每一對偏移曲面的交集求得求點串列，而這些交點串列在各自曲面上的 Domain 值可得到其交集 (\hat{r}, \hat{s}) 、 (\hat{t}, \hat{u}) 和 (\hat{v}, \hat{w}) ，這三個點我們稱為 DIC(domain intersection corner) points，可參考上圖之空心小圓圈。

我們可先假設一組 DIC-points (\hat{r}_0, \hat{s}_0) 、 (\hat{t}_0, \hat{u}_0) 和 (\hat{v}_0, \hat{w}_0) 來當做初始值，使用牛頓法來代出真正的 DIC-points。

由 Choi [7]可知，三個曲面之偏移曲面的交集可以下面之公式來近似求得：

$$G = \frac{a(\mathbf{n} \times \mathbf{o}) + b(\mathbf{o} \times \mathbf{m}) + c(\mathbf{m} \times \mathbf{n})}{\mathbf{m} \cdot (\mathbf{n} \times \mathbf{o})} \quad (\text{Eq.6.7})$$

where, $a = \mathbf{m} \cdot f^0(\hat{r}, \hat{s})$,

$b = \mathbf{n} \cdot g^0(\hat{t}, \hat{u})$,

$c = \mathbf{o} \cdot h^0(\hat{v}, \hat{w})$;

$\mathbf{m}, \mathbf{n}, \mathbf{o}$: unit normal vectors at $f(\hat{t}, \hat{s}), g(\hat{t}, \hat{u}), h(\hat{v}, \hat{w})$.

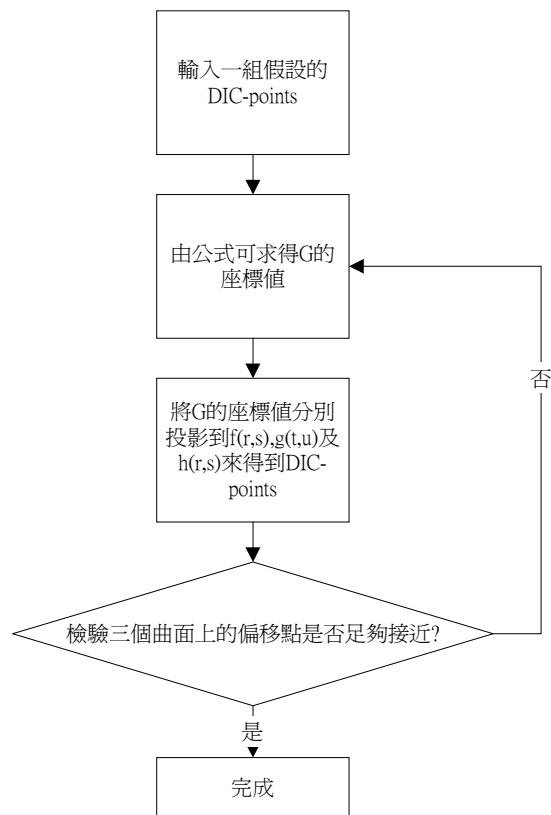


圖 6-6 求得 DIC-points 流程圖

6-5-2 建構 Corner blending 的邊界曲線

我們首先定義下面三個曲面：

$$\begin{aligned} b_1(\alpha, \beta) &: \text{edge blend between } f(r, s) \text{ and } g(t, u); \\ b_2(\alpha, \beta) &: \text{edge blend between } g(t, u) \text{ and } h(v, w); \\ b_3(\alpha, \beta) &: \text{edge blend between } h(v, w) \text{ and } f(r, s); \end{aligned}$$

由以上定義可知，若我們令 $\beta=0$ ，則上面三個曲面方程式就可變為 corner blending patch 的邊界曲線，即我們可以得到 corner blending patch 的邊界曲線如下：

$$e_i(\alpha) = b_i(\alpha, 0)$$

由 (Eq.5.7) 可得

$$\begin{aligned} e_1(\alpha) &\equiv b_1(\alpha, 0) \\ &= \frac{(1-\alpha)^2 C_0(0) + 2\varpi_1(1-\alpha)\alpha C_1(0) + \alpha^2 C_2(0)}{(1-\alpha)^2 + 2\varpi_1(1-\alpha)\alpha + \alpha^2} \quad (\text{Eq.6.8}) \end{aligned}$$

$$\begin{aligned} \text{where,} \quad C_0 &= f(\hat{r}, \hat{s}), \\ C_2 &= g(\hat{t}, \hat{u}) \\ C_1 &\text{ as given by (Eq.5.6)} \end{aligned}$$

而 cross-boundary tangent 可由下面公式求得

$$d_1(\alpha) = \frac{(1-\alpha)^2 D_0(0) + 2\varpi_1(1-\alpha)\alpha D_1(0) + \alpha^2 D_2(0)}{(1-\alpha)^2 + 2\varpi_1(1-\alpha)\alpha + \alpha^2} \quad (\text{Eq.6.9})$$

其中 $D_0(\beta) = \partial C_0(\beta) / \partial \beta = f_r \&+ f_s \&$

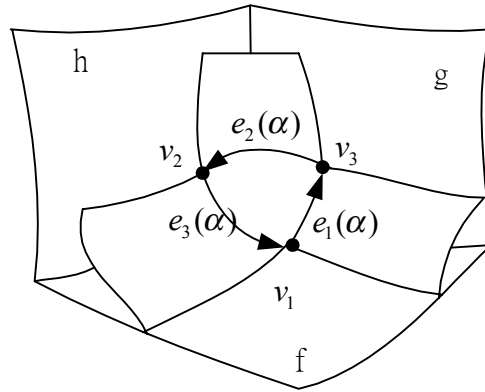


圖 6-7 Corner blending 之邊界曲線

6-5-3 建構 Corner blending patch

在第三章我們曾經提到如何建構一個 triangular Gregory patch，我們可以利用相同的方法建構出 corner blending patch，也就是說，我們可以下面的數學式來表示一個 corner blending patch：

$$c(W) = \sum_{i=1}^3 \gamma_i(W) \{e_i(\alpha_i) + \lambda_i \mu v_i d_i(\alpha_i)\} \quad (\text{Eq.6.10})$$

$$\begin{aligned}
\alpha_1 &= \lambda_3(\lambda_3 + \lambda_2) \\
\alpha_2 &= \lambda_1(\lambda_1 + \lambda_3) \\
\text{其中, } \alpha_3 &= \lambda_2(\lambda_2 + \lambda_1) \\
W &= (\lambda_1, \lambda_2, \lambda_3) \\
\gamma_i(W) &= \begin{cases} \frac{(1/\lambda_i)^2}{\sum_j (1/\lambda_j)^2}, & \text{if } \lambda_i \neq 0 \\ 1, & \text{if } \lambda_i = 0 \end{cases}
\end{aligned}$$

上式與 (Eq.3.6)有些許不同，它多引入了兩個參數 μ 和 v_i ， v_i 稱為 correction factors，它的作用是為了使每一個邊界曲線正交方向的切線向量，對曲面的影響力相同，它可以用下面的公式來求得一個合理的值：

$$v_i = |e_i(0.5) - v_i| / |d_i(0.5)| \quad ; \quad i = 1, 2, 3. \quad (\text{Eq.6.11})$$

上式中的 v_i 即為圖表 5-6-3 中所示之邊界曲線之頂點。

另外一個參數 μ (expansion factor)則是用來控制 corner blending 的豐滿度(fullness)，所謂 fullness 是指”rolling”ball 的中心到 corner blending 的中心 $c(1/3, 1/3, 1/3)$ 的距離，我們可用下面的公式求得：

$$d = \left| \left(\bar{e} + \frac{\mu}{3} \bar{d} \right) - \hat{f}^0 \right| \quad (\text{Eq.6.12})$$

$$\text{其中, } \bar{e} = (e_1(0.5) + e_2(0.5) + e_3(0.5)) / 3$$

$$\bar{d} = (v_1 d_1(0.5) + v_2 d_2(0.5) + v_3 d_3(0.5)) / 3$$

$$d \quad : \text{radius of the rolling ball}$$

$$\hat{f}^0 = f^0(\hat{r}, \hat{s}) \quad : \text{center of the rolling ball at the corner}$$

以上介紹了求取三個曲面相交的 corner blending 的方法，它可以很容易地擴展成 n-edges 的 corner blending，只

要將其 Domain 擴張成如下所示，詳細可參考文獻 Choi [7]：

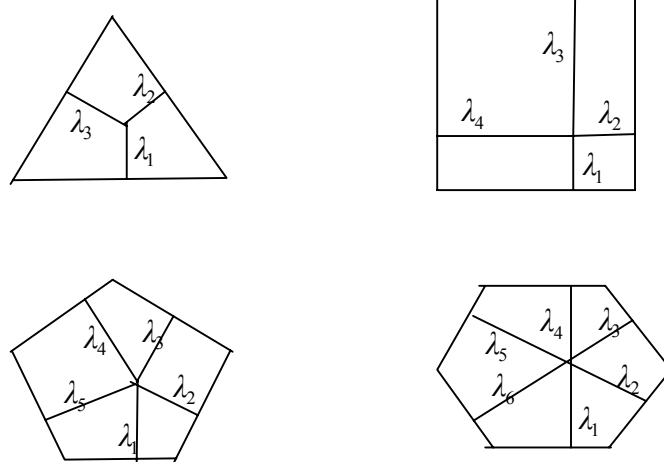


圖 6-8 Domains of n -sided Corner Blending

6-6 實例測試

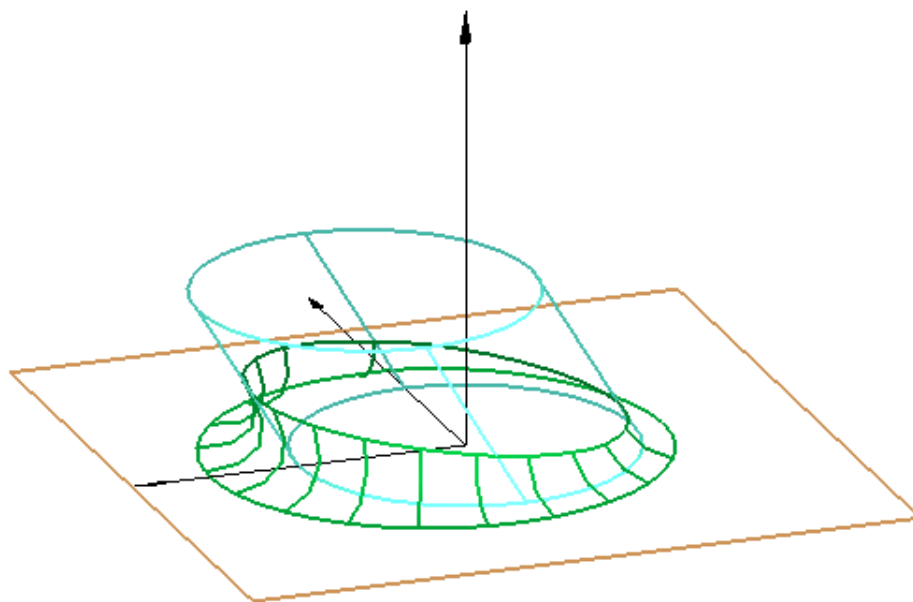


圖 6-9 平面與傾斜之圓柱所構成之 blending surface

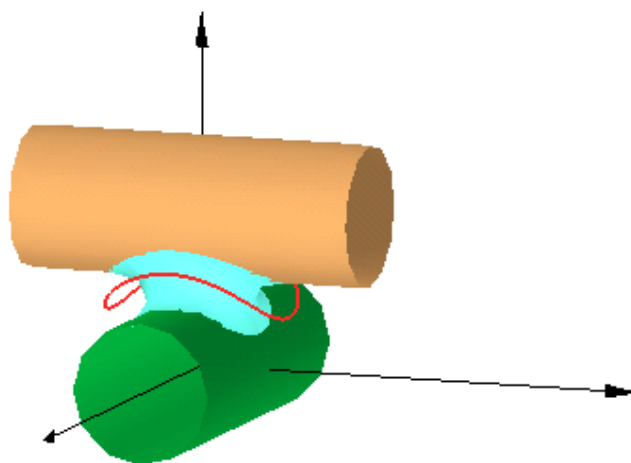


圖 6-10 兩圓柱所構成之 blending surface 1

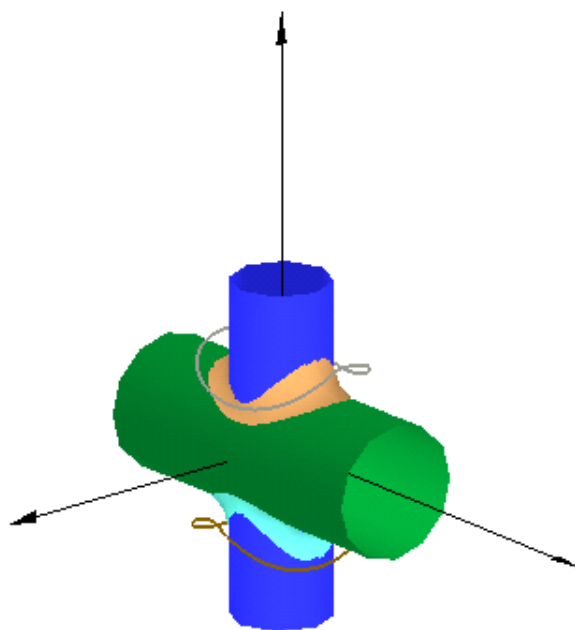


圖 6-11 兩圓柱所構成之 blending surface 2

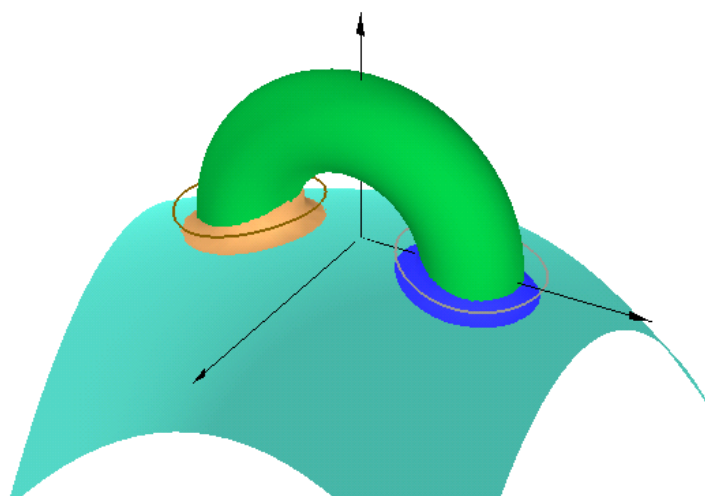


圖 6-12 圓環與曲面所構成之 blending surface

6-7 討論

本節所述為一般情況 blending 的探討，即 blending 的半徑為一個常數，且在做 corner blending 時並不會遇到奇異或特殊情形，但在實際的應用中，由於大部分的 blending 都是為了使尖銳的地方圓滑化，所以很有多邊(multiple edge)相遇在一個頂點的 blending 出現，且這些邊可能被使用者要求選定不同的半徑，很有可能以本章的方法並無法求得所期望的 blending 曲面。

一般的 rolling-ball 的方法，若是對 edge blending 來說，是非常有效率的，但是對 vertex(corner) blending 來說，卻沒有很完整的理論來解決這方面的問題，Higashi[18][19]則針對 corner blending 的問題提出他的解決方法，他對 corner blending 提出三個法則：

1. 當一個頂點被包圍的邊全是凸的邊或全是凹的邊時，我們必須先針對半徑較大的邊做 blending。
2. 當一個頂點同時被凸的邊及凹的邊包圍時，blending 的順序按照使用者點選的順序。
3. 當一個 fillet 的半徑比已經產生的 fillet 的半徑還大，且產生偏移曲面的方向與曲率向量的方向相同時，這個 fillet 就無法被產生。

針對傳統的 blending 的方法，大部分用在等半徑的邊，或相互垂直的邊，當多邊混成時，半徑不相等，或相交不為 90^0 ，或是凸或凹的邊時，常常會出現我們所無法預期的形狀出現，Tamas[35]提到可使用”setback”的技術來避免這種情形發生。

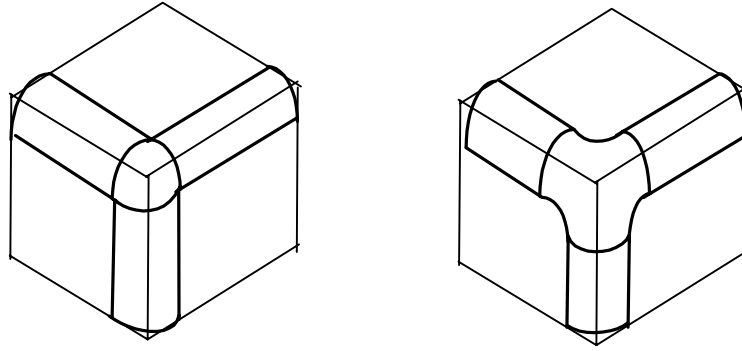


圖 6-13 使用”setback”技術進行 corner blending

例如當在做上圖的blending時，若其中兩個邊的blending半徑很小，而另一個邊的半徑很大時，如果使用傳統的3-side patch時，所產生出來的曲面是比較難處理的，但若使用”setback”技術，則可以產生較理想的曲面，

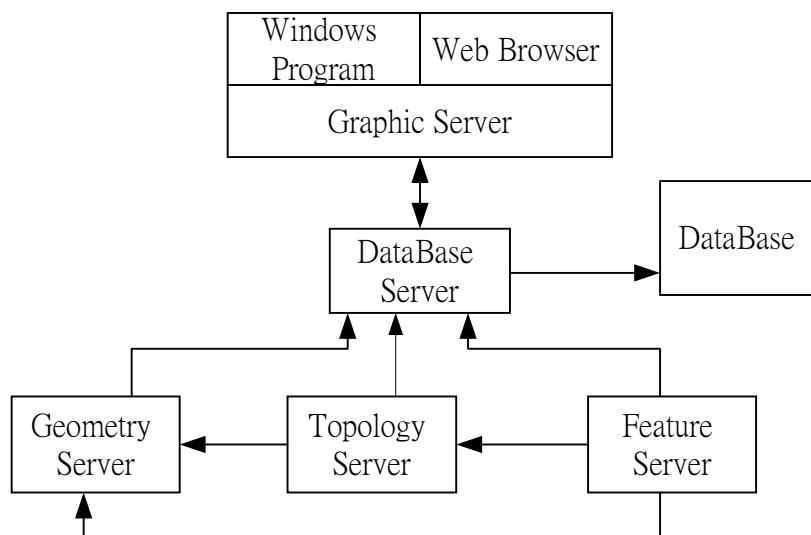
在文獻 Tamas[34]中，對最近幾年，各種求取 blending surface 的方法做一個大致上的整理，它將 blending 的方法分為參數曲面(parameter surfaces)和隱性曲面(implicit surfaces)兩大類，而它認為參數曲面具有較大的優勢，且在未來 blending surface 的發展，應在變化半徑的 blending，以及 blend surfaces 與 base surfaces 之間較高階數的連續性兩方面。

第七章 實作分散式幾何軟體元件

7-1 前言

由於幾何在 CAD 中所扮演的角色，大部分是屬於計算的工作，且是相當耗費 CPU 時間的工作，所以筆者希望將其實作成一個幾何伺服器，來提供 CAD 系統其它的元件來使用。在這裡筆者運用微軟的 DCOM(Distributed Component Object Model)的技術，將 CAD 中的幾何核心包裝成一個 COM 元件，以達到服務的功能，更可進一步的達到分散式的目的。

7-2 分散式 CAD 之架構



註:箭頭方向代表函式呼叫方向

圖 7-1 分散式 CAD 之架構圖

以下簡述各個元件所負責的工作及其功能：

Geometry Server

提供所有有關幾何的運算及功能，也是本論文所討論的重點，主要是開放來讓 Topology server, Feature server 呼叫，大致上的作用已在之前提及，在此不再贅述。

Topology Server

提供所有有關拓樸的運算及功能，如實體與實體之間的布林運算，主要是由 feature server 呼叫。

Feature Server

提供所有有關特徵的運算及功能，如伸出(extrude)，掃出(sweep)，導角(fillet)等，主要由 graphic server 呼叫。

Database Server

負責整合 geometry server，topology server 及 feature server，將此三個元件所產生的圖元(Entity)及資訊存入資料庫中，並負責與 graphic server 溝通。

Graphic server

主要為與使用者介面溝通，讀取 Database Server 需要顯示與使用者介面的圖元。

Database

為所有繪圖資訊所儲存的地方。

7-3 幾何伺服器之架構

筆者實作幾何伺服器主要可分為兩個部分，第一個為對內溝通的元件，包含大部分的幾何功能，並將其實作成一個動態函式庫(dll)，第二個為對外溝通的介面，歸納出在 CAD 系統中，幾何伺服器所需提供的功能，並將其實作成一個分散式物件模型(DCOM)。

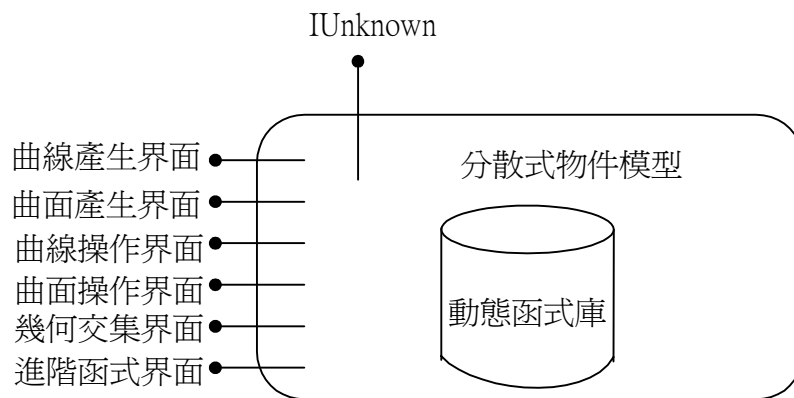


圖 7-2 幾何伺服器架構圖

7-3-1 幾何伺服器之動態函式庫部分

這一個部分是以 C++ 所寫成之類別函式庫(class library)，它的類別架構圖可以參考下圖：

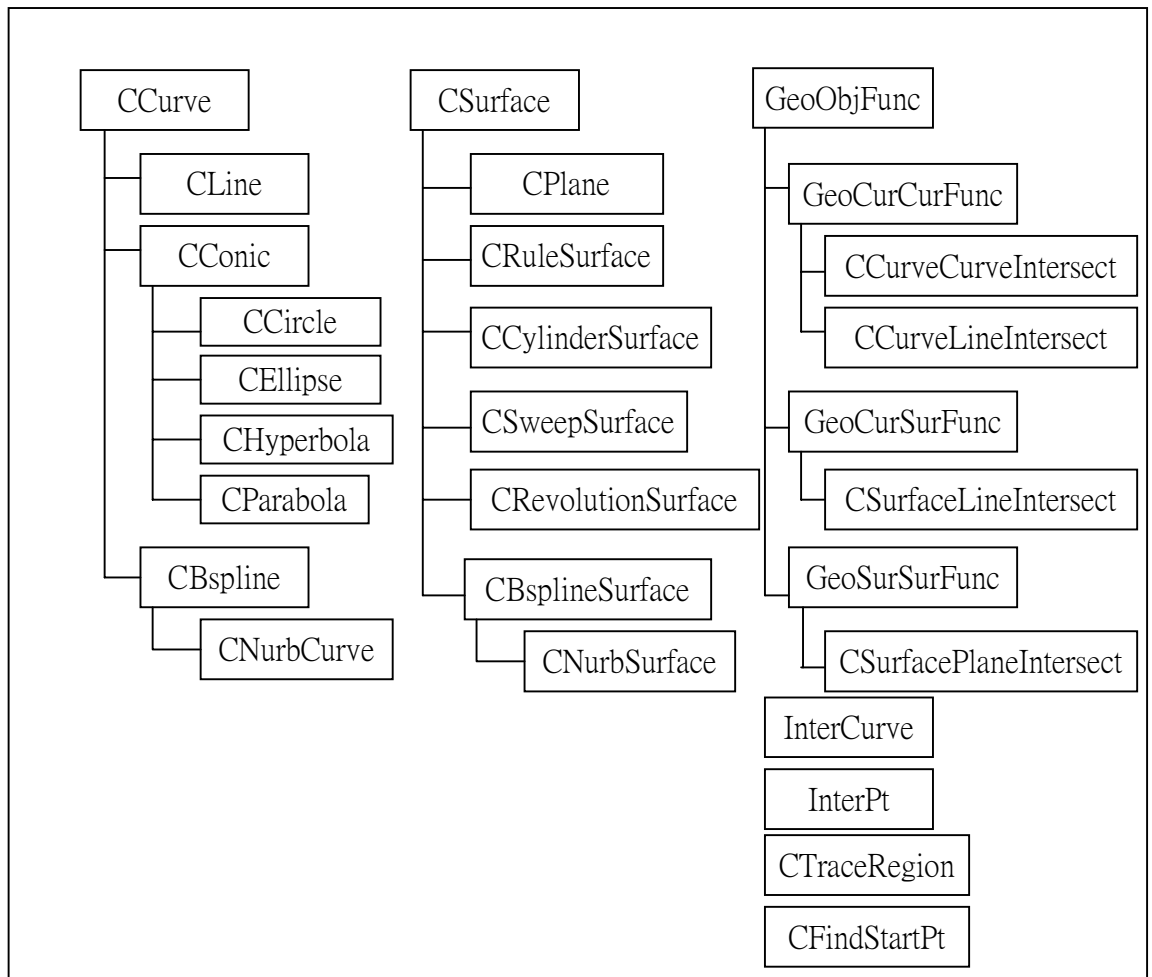


圖 7-3 類別架構圖

由圖表 7-3-2 可知，幾何核心大致上可以分為四個部分，

1. 曲線與其操作函式。
2. 曲面與其操作函式。
3. 幾何交集與其操作函式。
4. 其它進階函式(如 blending，healing 等)。

7-3-2 幾何伺服器之分散式物件模型部分

這一個部分是使用微軟的 DCOM 技術，使得筆者在上一節所實作之類別函式庫可以應用在分散式的環境上。COM 技術的主要精神在於界面與實作分離，也就是說，視我們想要提供那些服務，我們就必須提供那些界面，而這些界面是與外界溝通的標準，它不能隨意更動，否則便破壞了協定。

由上一節的歸納，可知我們必須提供以下之界面，以供拓樸伺服器及特徵伺服器使用：

1. 曲線產生界面。
2. 曲線操作界面。
3. 曲面產生界面。
4. 曲面操作界面。

5. 幾何交集界面。

6. 進階函式界面。

而上面所列之界面所實作之函式應大致分為兩種類型，第一種為直接將回傳資料存入資料庫；第二種則將資料傳回資料端，提供呼叫者利用後，再由呼叫者決定是否存入資料庫。

7-4 與其它模組之溝通

幾何核心與其它模組之溝通最主要的資料媒介，是在 7-2 節所提到的 database server。而 database server 主要是負責存取資料庫內的資料，以提供其它模組，如幾何伺服器、拓樸伺服器或特徵伺服器來使用。

舉個例子來說，若是拓樸伺服器需要知道某一個線 (ID=102)，在某一個參數($u=0.5$)下的座標值，則拓樸伺服器必須依照下面的步驟：

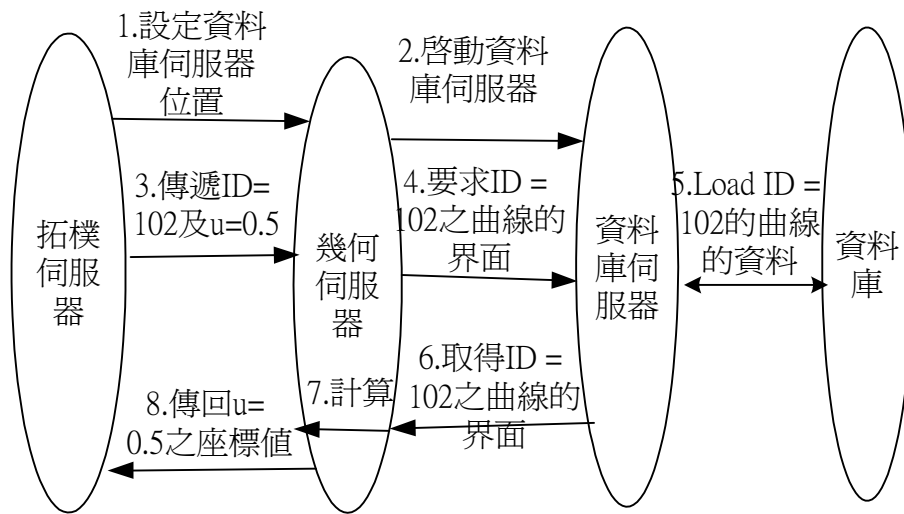


圖 7-4 拓樸伺服器呼叫幾何伺服器之流程

由圖 7- 可知，幾何伺服器本身並不儲存任何資料，需要用到資料時，再以 ID 來向資料庫伺服器索取資料，所以資料庫伺服器兩個最重要的函式就是儲存(Save)和載入(Load)，故幾何核心的資料儲存可參考陳昭偉[40]，他對資料庫伺服器有詳盡的探討。

7-5 實作成果

筆者將之前所提到之理論，實作出幾何伺服器，再配合拓樸伺服器，特徵伺服器[39]，繪圖伺服器[39]，資料庫伺服器及其它元件，可做出以瀏覽器為使用者界面之分散式 CAD 核心的軟體。如下圖

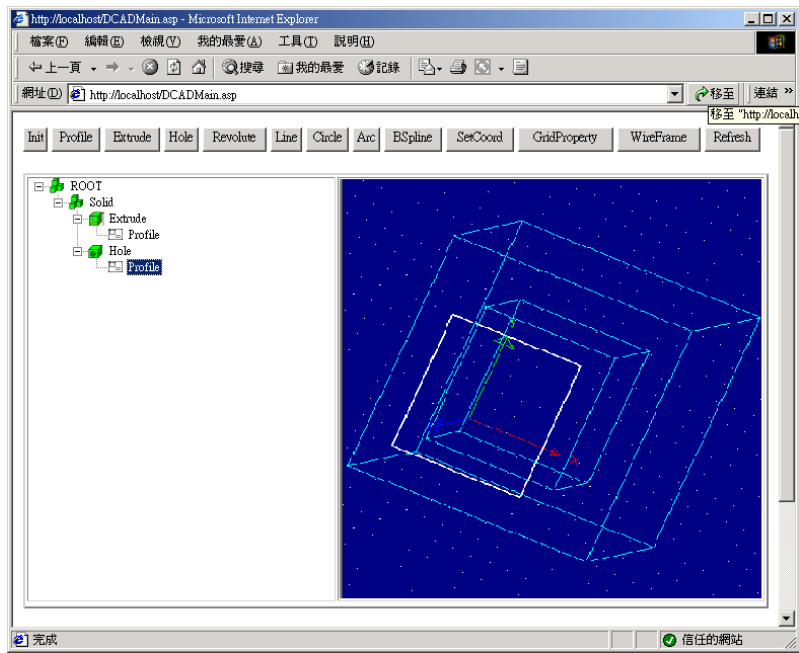


圖 7-5 CAD 瀏覽器界面圖

第八章 回顧與展望

本篇論文的研究重心可分為兩部分：(1)CAD 軟體幾何核心之設計。(2)分散式 CAD 核心之可能性探討與實作。CAD 軟體幾何核心之部分，所根據的理論架構已發展多年，筆者儘可能去研究現今使用最為廣泛之理論方法，作整體性的探討與整理，設計出幾何核心，其實作的部分多於理論。而分散式 CAD 核心則在目前鮮少人做這方面的研究，筆者以自己的方式實作，並以第一部分的核心為基礎。

筆者雖然在本文中進行研究並提出實例，但仍有一些方向仍然有很多的發展方向與空間，筆者以下將各個發展部分分類說明：

1. 將所求得的面與面的交線，可做曲線平滑化的步驟。
2. 在求解幾何交集方面，關鍵點的求解理論發展至今，仍然無法很有效率地將關鍵點正確地求出，故在這方面還有研究的空間。

3. 在求解幾何交集方面，如線與線或線與面的交集，可能發生其交集為一個線段，而非數個點，即線與線或線與面有部分重合時，筆者所發展的幾何核心無法得知正確的解。
4. 在做曲面混成時，尚未實作 corner blending 的部分。
5. 在做分散式 CAD 時，尚未考慮資料同步化的問題。

我相信，在不久後，電腦輔助設計軟體一定可以在瀏覽器上實現，並進而達到多人同時設計修改同一個模型。在此筆者只是對於這個目標作一個初步的研究，對於其中的細節，仍有許多理論尚待探討，不過對於未來 CAD 的發展提供了一個方向。

參考文獻

- [1]Barnhill, R.E., Kersey, S.N., “A marching method for parametric surface/surface intersection” , Computer Aided Geometric Design, Vol 7, p257-280, 1990.
- [2]Barnhill, R.E., G. Farin, M. Jordan, and Piper, B.R., “Surface/Surface Intersection” , Computer Aided Geometric Design, Vol 4, p3-16, 1987.
- [3]Barry Fowler, Richard Bartels, “Constraint-Based Curve Manipulation” ,IEEE Computer Graphics & Applications, September 1993.
- [4]Brechtner EL., “General tool offset curves and surfaces” , Geometry processing for design and manufacturing, SIAM, p101-21, 1992.
- [5]Chen YJ, Ravani B., “Offset surface generation and contouring in computer-aided design” , Journal of Mechanisms, Transmissions and Automation in Design: ASME Transaction, Vol 109, p133-42, 1987.

- [6]Cheng, K-P, “Using plane vector fields to obtain all the intersection curves of two general surfaces” in Strasser, W and Seidel, H(Eds.) Theory and Practices of Geometric Modeling Springer-Verlag, New York p187-204, 1989.

- [7] Choi, Byoung K., “Surface Modeling for CAD/CAM” , Amsterdam; New York: Elsevier, 1991.

- [8]Choi, Byoung K., W.S Yoo, C.S Lee, “Matrix representation for NURB curves and surfaces” , Computer-Aided Design, Vol 22, p235-240, 1990.

- [9]Chun-Fong You, “Computational Geometry Lecture Notes” ,Department of Mechanical Engineering NTU, 1995.

- [10]Coquillart, S. “Computing offsets of B-spline curves” , Computer Aided Design, Vol 19, p305-309, 1987.

- [11]David F. Rogers, J. Alan Adams, “Mathematical Elements for Computer Graphics” , Second Edition, McGRAW-HILL, 1990.

- [12] Farin G., Hagen H., Noltemeier H.(eds.) in cooperation with W. Knodel, “Geometric Modelling” , Wien ; New York : Springer-Verlag, 1993.

- [13]Farin G., “Trends in curve and surface design” ,
Computer Aided Design, Vol 21, p293-295, 1989.
- [14]Farouki, R. T., “The characterization of parametric
surface section” , Computer Vision, Graphics, and Image
Processing, Vol 33, p209-236, 1986.
- [15]Farouki, R. T. , Neff, C. A., “Analytic properties of plane
offset curves” ,Comput. Aided Geom. Des., 1990.
- [16]Farouki, R. T. , Neff, C. A., “Algebraic properties of
plane offset curves” , Comput. Aided Geom. Des., 1990.
- [17]Gregory, J. A., “N-sided surface patches”, In The
Mathematics of Surfaces, J. A. Gregory, Clarendon Press,
Oxford, 1986.
- [18]Higashi M., “High-quality solid-modelling system with
free-form surfaces” ,Computer Aided Design, Vol 25,
p173-183, 1993.
- [19]Higashi M., “Generation of a corner fillet around a vertex
by a rolling ball method” J. Jap. Soc. Precision eng. Vol 56,
No 12, p2219-2224, 1990.
- [20]Hoscheck, J., “Offset curves in the plane” , Comput.
Aided Des., Vol 17, p77-82, 1985.

- [21]Ian Braid, “Non-local blending of boundary models” ,Computer Aided Design, Vol 29, p89-100, 1997.

- [22]Les Piegl, Wayne Tiller, “The NURBS book” , New York :Springer, 1995.

- [23]Les Piegl, Wayne Tiller, “Computer offsets of NURBS curves and surfaces” , Computer-Aided Design,Vol 31, p147-156, 1999.

- [24] Les Piegl, Wayne Tiller, “Curve and surface constructions using rational B-splines” , Computer-Aided Design, Vol 19, p485-499, 1987.

- [25]Ma L., Peng Q., “Smoothing of free-form surfaces with Bézier patches” , Computer Aided Geometric Design, Vol 12, p231-249, 1995.

- [26]Martin R. R., Stephenson R. C., “Sweeping of three-dimensional objects” ,Computer Aided Design,vol 22, p223-233, 1990.

- [27]Mortenson, Michael E., “Geometric Modeling” , John Wiley & Sons, Inc. 1985.

- [28]Mortenson, Michael E., “Geometric Modeling” , Second Edition, John Wiley & Sons, Inc. 1995.

- [29]Mullenheim, G., “Convergence of a surface/surface intersection algorithm” ,Computer Aided Geometric Design, Vol 7, p415-425, 1990.

- [30]Owen, J. and Rockwood, A., “Intersection of general implicit surfaces” , In Geometric Modeling, Algorithms and Trends, ed. G. Farin. SIAM, Philadelphia, PA, 1987.

- [31]Pham B., “Offset curves and surfaces: a brief survey” , Computer Aided Design, Vol 24, p223-229, 1992.

- [32]Piegl L. A., Richard A. M., “Tessellating trimmed NURBS surfaces” , Computer Aided Design, Vol 27, p16-26, 1995.

- [33]Sederberg, T. W., “Planar piecewise algebraic curves” , Computer Aided Geometric Design, Vol 1, p241-255, 1984.

- [34]Tamas Varady, Ralph Martin, Janos Vida, “A survey of blending methods that use parametric surfaces” ,Computer Aided Design, Vol 26 ,p345-365, 1994.

- [35]Tamas Varady, Alyn Rockwood, “Geometric construction for setback vertex blending” ,Computer Aided Design, Vol 29, p413-425, 1997.

- [36]Takashi Maekawa, “An overview of offset curves and surfaces” , Computer Aided Design, Vol 31, p165-173, 1999.
- [37]Tiller, W. and Hanson, E. G. “Offsets of two-dimensional profiles” , IEEE Comput. Graph. & Applic. Vol 4, p36-46, 1984.
- [38]William L. L., “Tessellation of trimmed NURB surfaces” , Computer Aided Geometric Design, Vol 13, p163-177, 1996.
- [39]王大中, “建立在分散式環境下之 3D CAD 架構與特徵核心”, 國立台灣大學碩士論文, 中華民國 88 年六月。
- [40]陳昭偉, “建立在分散式環境下之 3D CAD 拓樸核心與動態資料庫”, 國立台灣大學碩士論文, 中華民國 88 年六月。
- [41]陳源芳, “非線性最佳化頻率曲線與曲面之交集探討”, 國立台灣大學碩士論文, 中華民國 87 年六月。
- [42]郭錦誠, “參數化曲線與曲面間交集的探討”, 國立台灣大學碩士論文, 中華民國 84 年六月。
- [43]劉志鴻, “B-spline 曲線/曲面交集問題之探討”, 國立台灣大學碩士論文, 中華民國 79 年六月。

[44]潘求新，“含自由曲面之實體圓角化的研究”，國立台灣大學碩士論文，中華民國 83 年六月。