UNIVERSIDAD NACIONAL DE INGENIERÍA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE MATEMÁTICA

Sistema para la Construcción de Curvas y Superficies NURBS

por

Jaime Osorio Ubaldo

Tesis para Optar
el Título Profesional de
LICENCIADO en MATEMÁTICA

Prof: William Carlos Echegaray Castillo
Asesor

UNI, diciembre del 2009

CIP - CATALOGO DE PUBLICACIÓN

Osorio Ubaldo, Jaime

Sistema para la Construcción de Curvas y Superficies NURBS / Jaime Osorio Ubaldo. – EPM - FC - UNI, 2009.

80 p.: il.

Tésis (Licenciatura)—Universidad Nacional de Ingeniería, Facultad de Ciencias, Escuela Profesional de Matemática, Lima, 2009. Asesor: William Carlos Echegaray Castillo

A mis padres, hermanas y esposa

Agradezco al Profesor William Carlos Echegaray Castillo por su orientación y también deseo agradecer a todas aquellas personas que de una u otra forma me ayudaron a terminar este trabajo.

RESUMEN

En el campo del Diseño Geométrico Asistido por Computador se requiere manipular curvas y superficies suaves para el diseño del contorno de un objeto como las alas de un avión, la carrocería de un automóvil, el rostro de una persona, etc.

Esta tesis en su primer capítulo hace un estudio sobre las curvas y superficies de Bezier que fueron desarrolladas paralelamente por Pierre Bezier y Paul D'Casteljau para el diseño de automóviles. En el segundo capítulo veremos las curvas y superficies B-Spline y las NURBS (Non Uniform Rational B-Spline) que son las generalizaciones de las anteriores. Finalmente en el tercer capítulo implementaremos un sistema en software libre que nos permita construir estas curvas y superficies utilizando para ello el lenguaje de programación Java y el gestor de base de datos MySql.

Indice general

Indice de figuras

1 CONCEPTOS PRELIMINARES

La curva polinomial de Bezier se desarrolló en la década de los 60, para el trazado de dibujos técnicos, diseño aeronáutico y de automóviles. Su denominación es en honor a Pierre Bezier, quien ideó un método de descripción matemática de las curvas que se comenzó a utilizar con éxito en los programas de CAD (Diseño Asistido por Computadora).

1.1. Polinomio de Bernstein

Definición 1.1. Sean los puntos $p_k \in R^n, k = 0, \cdots, n$ y $\sum_{k=0}^{n} \beta_k = 1$ entonces

$$\sum_{k=0}^{n} p_k \beta_k$$

es denominado combinación afín de los p_k.

Definición 1.2. Sean los puntos $p_k \in R^n, k = 0, \cdots, n$ y $\sum_{k=0}^{n} \beta_k = 1$ y $\beta_k \geq 0$ entonces

$$\sum_{k=0}^{n} p_k \beta_k$$

es una combinación convexa de los p_k. Los puntos $p_0, \cdots, p_n$ definen por

$$co\{p_0, \cdots, p_n\} = \{\sum_{k=0}^{n} p_k \beta_k, \beta_k \geq 0, \sum_{k=0}^{n} \beta_k = 1\}$$

la cápsula convexa de los p_k.

Definición 1.3. Una aplicación $\phi : R^n \longrightarrow R^n$ donde $\phi(x) = Ax + v$, $A \in R^{nxn}$, $v \in R^n$ es denominado transformación afín.

Ejemplo 1.1. Si $A = I, v \neq \theta$ es una traslación. □

Definición 1.4. Los puntos $p_0, p_1, \cdots, p_n \in R^n$ son llamados afín independientes, si los n vectores

$$p_1 - p_0, \cdots, p_n - p_0$$

son linealmente independientes.

Teorema 1.1. Sea $\phi : R^n \longrightarrow R^n$ una transformación afín.

1. Combinaciones afines quedan invariantes bajo transformaciones afines, es decir
$$\phi(\sum_{k=0}^{n} p_k \beta_k) = \sum_{k=0}^{n} \beta_k \phi(p_k)$$

2. ϕ es determinado únicamente por los puntos $p_0, p_1, \cdots, p_n$ afín independientes y sus imágenes $\phi(p_0), \phi(p_1), \cdots, \phi(p_n)$

3. Sean $a, b, c \in R^n$ colineales, se define
$$ratio(a, b, c) = \frac{\|b - a\|_2}{\|c - b\|_2}$$
entonces se cumple que $ratio(a, b, c) = ratio(\phi(a), \phi(b), \phi(c))$

Demostración:

1. Sea $\sum_{k=0}^{n} \beta_k p_k \in R^n$ con $\sum_{k=0}^{n} \beta_k = 1$

$$\begin{aligned}
\phi(\sum_{k=0}^{n} \beta_k p_k) &= A(\sum_{k=0}^{n} \beta_k p_k) + v \\
&= \sum_{k=0}^{n} \beta_k A p_k + v \\
&= \sum_{k=0}^{n} \beta_k A p_k + v \sum_{k=0}^{n} \beta_k \\
&= \sum_{k=0}^{n} \beta_k (A p_k + v) \\
&= \sum_{k=0}^{n} \beta_k \phi(p_k)
\end{aligned}$$

2. Sea $p_0, p_1, \cdots, p_n$ afín independientes.

$$\begin{aligned}
\phi(p_0) &= A p_0 + v \\
\phi(p_k) &= A p_k + v; k = 1, \cdots, n
\end{aligned}$$

Restando obtenemos que $\phi(p_k) - \phi(p_0) = A(p_k - p_0)$, con lo que se observa que A es único, además $\phi(0) = v = \sum_{k=0}^{n} \beta_k p_k$ es único.

3. Podemos establecer que

$$\begin{aligned}
b - a &= \lambda(c - b) \\
\|b - a\|_2 &= \|\lambda(c - b)\|_2 = |\lambda| \|c - b\|_2 \\
ratio(a, b, c) &= \frac{\|b - a\|_2}{\|c - b\|_2} = |\lambda|
\end{aligned}$$

Veamos

$$\begin{aligned}
\|\phi(b) - \phi(a)\|_2 &= \|Ab - Aa\|_2 \\
&= \|A(b-a)\|_2 \\
&= \|A\lambda(c-b)\|_2 \\
&= |\lambda|\|A(c-b)\|_2 \\
&= |\lambda|\|Ac - Ab\|_2 \\
&= |\lambda|\|\phi(c) - \phi(b)\|_2
\end{aligned}$$

Con lo que

$$ratio(\phi(a), \phi(b), \phi(c)) = \frac{\|\phi(b) - \phi(a)\|_2}{\|\phi(c) - \phi(b)\|_2} = |\lambda|$$

■

Definición 1.5. Para $n \in N$ se definen los polinomios de Bernstein de grado n como

$$B_k^n(t) = \begin{pmatrix} n \\ k \end{pmatrix} t^k(1-t)^{n-k}, t \in [0,1], k = 0, 1, \cdots, n$$

donde

$$\begin{pmatrix} n \\ k \end{pmatrix} = \begin{cases} \frac{n!}{k!(n-k)!}, & k \in \{0, 1, \cdots, n\} \\ 0, & en\ otro\ caso \end{cases}$$

Propiedad 1.1. Sea $n \in N$ y el polinomio de Bernstein $B_k^n(t)$ entonces resulta que estos polinomios cumplen las siguientes propiedades:

1. Son simétricos

$$B_k^n(t) = \frac{n!}{k!(n-k)!} t^k(1-t)^{n-k} = B_{n-k}^n(1-t)$$

2. Las únicas raíces son 0 y 1

$B_k^n(0) = 0; B_k^n(1) = 0$ para $k \in \{1, \cdots, n-1\}$ pero $B_0^n(0) = 1 = B_n^n(1)$

3. Forman una partición de la unidad

$$\begin{aligned}\sum_{k=0}^{n} B_k^n(t) &= \sum_{k=0}^{n} \binom{n}{k} t^k (1-t)^{n-k} \\ &= (t + (1-t))^n \\ &= 1\end{aligned}$$

4. Son positivos en $]0, 1[$

$$B_k^n(t) > 0, \forall t \in]0, 1[$$

5. Satisfacen la relación de recurrencia

$$B_k^n(t) = (1-t)B_k^{n-1}(t) + tB_{k-1}^{n-1}(t), k \in \{1, 2, \cdots, n\}$$

considerando que $B_0^0(t) = 1$ y $B_k^n(t) = 0$ para $k \notin \{0, \cdots, n\}$ ya que

$$\begin{aligned}B_k^n(t) &= \binom{n}{k} t^k (1-t)^{n-k} \\ &= \left[\binom{n-1}{k} + \binom{n-1}{k-1}\right] t^k (1-t)^{n-k} \\ &= \binom{n-1}{k} t^k (1-t)^{n-k} + \binom{n-1}{k-1} t^k (1-t)^{n-k} \\ &= (1-t)B_k^{n-1} + tB_{k-1}^{n-1}\end{aligned}$$

6. Elevación de grado

Un polinomio de Bernstein de grado $n-1$ puede escribirse como una combinación lineal de polinomios de Bernstein de grado n.

$$\begin{aligned} tB_k^n(t) &= \binom{n}{k} t^{k+1}(1-t)^{n-k} \\ &= \binom{n}{k} t^{k+1}(1-t)^{(n+1)-(k+1)} \\ &= \frac{\binom{n}{k}}{\binom{n+1}{k+1}} B_{k+1}^{n+1}(t) \\ &= \frac{k+1}{n+1} B_{k+1}^{n+1}(t) \end{aligned}$$

Además

$$\begin{aligned} (1-t)B_k^n(t) &= \binom{n}{k} t^k(1-t)^{n+1-k} \\ &= \frac{\binom{n}{k}}{\binom{n+1}{k}} B_k^{n+1}(t) \\ &= \frac{n-k+1}{n+1} B_k^{n+1}(t) \end{aligned}$$

Finalmente

$$\frac{1}{\binom{n}{k}} B_k^n(t) + \frac{1}{\binom{n}{k+1}} B_{k+1}^n(t) = t^k(1-t)^{n-k} + t^{k+1}(1-t)^{n-(k+1)}$$

$$
\begin{aligned}
&= t^k(1-t)^{n-k-1}((1-t)+t) \\
&= t^k(1-t)^{n-k-1} \\
&= \frac{1}{\binom{n-1}{k}} B_k^{n-1}(t)
\end{aligned}
$$

Por lo tanto

$$
\begin{aligned}
B_k^{n-1}(t) &= \binom{n-1}{k}\left[\frac{1}{\binom{n}{k}} B_k^n(t) + \frac{1}{\binom{n}{k+1}} B_{k+1}^n(t)\right] \\
&= \left(\frac{n-k}{n}\right) B_k^n(t) + \left(\frac{k+1}{n}\right) B_{k+1}^n(t)
\end{aligned}
$$

7. Conversión de la base de Bernstein a la base de potencias

   El conjunto $\{1, t, t^2, \cdots, t^n\}$ forman una base para el espacio de polinomios de grado $\leq n$, y los polinomios de Bernstein pueden escribirse en términos de esta base de potencias

$$
\begin{aligned}
B_k^n(t) &= \binom{n}{k} t^k(1-t)^{n-k} \\
&= \binom{n}{k} t^k \sum_{i=0}^{n-k} (-1)^i \binom{n-k}{i} t^i \\
&= \sum_{i=0}^{n-k} (-1)^i \binom{n}{k} \binom{n-k}{i} t^{i+k} \\
&= \sum_{i=k}^{n} (-1)^{i-k} \binom{n}{k} \binom{n-k}{i-k} t^i \\
&= \sum_{i=k}^{n} (-1)^{i-k} \binom{n}{i} \binom{i}{k} t^i
\end{aligned}
$$

8. Conversión de la base de potencias a la base de Bernstein

   Mostremos ahora como un elemento de la base de potencias puede escribirse como una combinación de los polinomios de Bernstein.

$$\begin{aligned}
\binom{n}{i} t^i &= \binom{n}{i} t^i (1-t+t)^{n-i} \\
&= \binom{n}{i} t^i \sum_{k=0}^{n-i} \binom{n-i}{n-i-k} t^k (1-t)^{n-i-k} \\
&= \sum_{k=0}^{n-i} \binom{n}{i} \binom{n-i}{n-i-k} t^{k+i} (1-t)^{n-i-k} \\
&= \sum_{k=0}^{n-i} \binom{i+k}{i} \binom{n}{i+k} t^{k+i} (1-t)^{n-i-k} \\
&= \sum_{k=0}^{n-i} \binom{i+k}{i} B_{k+i}^n(t)
\end{aligned}$$

9. Derivadas

   La derivada de un polinomio de Bernstein de grado n son polinomios de grado $n-1$

$$\begin{aligned}
\frac{d}{dt} B_k^n(t) &= \frac{d}{dt} \binom{n}{k} t^k (1-t)^{n-k} \\
&= k \binom{n}{k} t^{k-1} (1-t)^{n-k} - (n-k) \binom{n}{k} t^k (1-t)^{n-k-1} \\
&= \frac{n(n-1)!}{(k-1)!(n-k)!} t^{k-1} (1-t)^{n-k} - \frac{n(n-1)!}{k!(n-k-1)!} t^k (1-t)^{n-k-1} \\
&= n \left[\frac{(n-1)!}{(k-1)!(n-k)!} t^{k-1} (1-t)^{n-k} - \frac{(n-1)!}{k!(n-k-1)!} t^k (1-t)^{n-k-1} \right] \\
&= n \left[B_{k-1}^{n-1}(t) - B_k^{n-1}(t) \right]
\end{aligned}$$

10. Los polinomios de Bernstein de grado n son una base para el espacio de los polinomios de grado menor o igual que n.

    Por 8) los polinomios de grado menor o igual a n pueden escribirse como una combinación de polinomios de Bernstein.

    Probemos que son linealmente independientes, supongamos que

$$\sum_{k=0}^{n} b_k \binom{n}{k} t^k (1-t)^{n-k} = 0, t \in]0,1[$$

    dividiéndolo por $\binom{n}{k} (1-t)^n$ y haciendo $s = (\frac{t}{1-t})$ obtenemos $\sum_{k=0}^{n} b_k s^k = 0$ como $s^k > 0$ se tiene que $b_0 = b_1 = b_2 = \cdots = b_n = 0$.

Ejemplo 1.2.

Para n=1:

$$\begin{aligned} B_0^1(t) &= 1-t \\ B_1^1(t) &= t \end{aligned}$$

Para n=2:

$$\begin{aligned} B_0^2(t) &= (1-t)^2 \\ B_1^2(t) &= 2t(1-t) \\ B_2^2(t) &= t^2 \end{aligned}$$

Para n=3:

$$\begin{aligned} B_0^3(t) &= (1-t)^3 \\ B_1^3(t) &= 3t(1-t)^2 \\ B_2^3(t) &= 3t^2(1-t) \\ B_3^3(t) &= t^3 \end{aligned}$$

□

1.2. Curva de Bezier

Definición 1.6. Se denomina curva de Bezier al polinomio

$$c(t) = \sum_{k=0}^{n} p_k B_k^n(t), t \in [0,1]$$

Donde p_k son denominados los puntos de control de la curva de Bezier y $B_k^n(t)$ son los polinomios de Bernstein.

Propiedad 1.2. Sea $n \in N$ y $c(t)$ la curva de Bezier entonces se cumplen las siguientes propiedades:

1. Simetría
   Como $B_k^n(t) = B_{n-k}^n(1-t)$ entonces

$$\sum_{k=0}^{n} p_k B_k^n(t) = \sum_{k=0}^{n} p_{n-k} B_k^n(1-t)$$

2. Los extremos del segmento de curva son

$$c(0) = p_0; c(1) = p_n$$

3. $c(t)$ es una combinación afín de sus puntos de control, esto debido a que los polinomios de Bernstein suman uno, según la definición (1.2).

4. $c(t)$ es afin invariante es decir, dada una aplicación afín ϕ, la curva imagen $\phi(c)$ tiene a los $\phi(p_k), k = 0, ..., n$ como puntos de control sobre $t \in [0, 1]$, según el teorema (1.1).

5. $c(t)$ es una combinación convexa de sus puntos de control, ya que los polinomios de Bernstein son no negativos en $[0, 1]$. Por lo tanto el segmento de curva $c(t), t \in [0, 1]$ yace en la cápsula convexa de sus puntos de control.

6. Control pseudolocal

   Si dos curvas de Bezier $c(t)$ y $\bar{c}(t)$ tienen los mismos puntos de control excepto uno digamos el k-ésimo, entonces

$$c(t) - \bar{c}(t) = B_k^n(t)(p_k - \bar{p}_k)$$

   dado que $B_k^n(t)$ alcanza un máximo y esto ocurre para $t = \frac{k}{n}$ la máxima diferencia entre ambas curvas sucede en la zona correspondiente a un entorno del valor del parámetro $t = \frac{k}{n}$, decreciendo la diferencia hacia 0 al alejarse las curvas de esta zona y coincidir finalmenete para $t = 0$ y $t = 1$. Por lo tanto, el movimiento o modificación de un punto de control aunque afecta a la forma de toda la curva, es más notable en las cercanías del punto movido, con lo cual podemos concluir que las curvas de Bezier no presentan suficiente control local .

7. Derivada

De la definición de $B_k^n(t)$ y la propiedad (1.1) obtenemos

$$c'(t) = n\sum_{k=0}^{n-1}(p_{k+1} - p_k)B_k^{n-1}(t)$$

es tangente, en sus segmentos extremos, al polígono formado por los puntos p_0, p_1, p_{n-1}, p_n para

$$\begin{aligned} c'(0) &= n(p_1 - p_0) \\ c'(1) &= n(p_n - p_{n-1}) \end{aligned}$$

Ahora llamando $\Delta p_j = p_{j+1} - p_j$ con lo que podemos decir que $c'(t)$ es también una curva de Bezier de grado $n-1$ con puntos de control $n\Delta p_k$, además

$$c''(t) = n(n-1)\sum_{k=0}^{n-2}\Delta(\Delta p_k)B_k^{n-2}(t)$$

donde $\Delta(\Delta p_k) = \Delta p_{k+1} - \Delta p_k = (p_{k+2} - p_{k+1}) - (p_{k+1} - p_k)$
y se representa por $\Delta^2 p_k$.
En general

$$c^r(t) = n(n-1)\cdots(n-r+1)\sum_{k=0}^{n-r}\Delta^r p_k B_k^{n-r}(t)$$

donde $\Delta^r p_j = \Delta^{r-1}p_{j+1} - \Delta^{r-1}p_j$

Las curvas de Bezier fueron desarrolladas casi paralelamanete por Paul De Casteljau en 1959 y Pierre Bezier en 1962, veamos en que consistió el algoritmo de Casteljau.

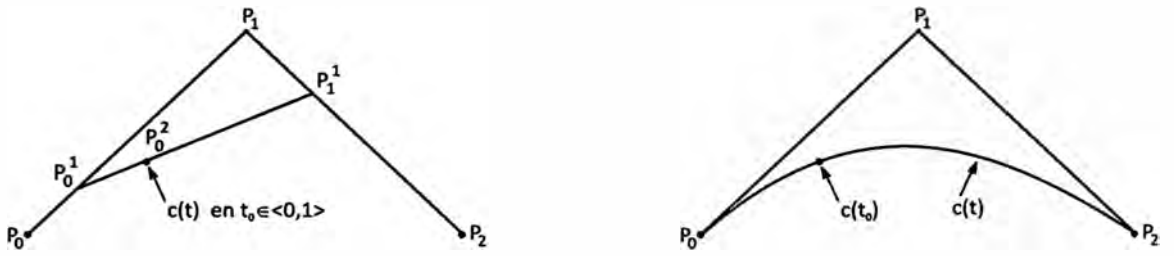

Figura 1.1: Algoritmo de Casteljau con tres puntos de control

Algoritmo 1.1. (De Casteljau) Dada la curva de Bezier $c(t) = \sum_{k=0}^{n} p_k B_k^n(t)$ usando las siguientes relaciones de recurrencia para los polinomios de Bernstein

$$\begin{aligned} p_k^0 &= p_k \\ p_k^{i+1} &= (1-t)p_k^i + tp_{k+1}^i; t \in [0,1]; k = 0, \cdots, n-i-1; i = 0, \cdots, n-1 \end{aligned}$$

es decir

$$\begin{aligned} p_k^1 &= (1-t)p_k^0 + tp_{k+1}^0; \ \ k = 0, \cdots, n-1 \\ p_k^2 &= (1-t)p_k^1 + tp_{k+1}^1; \ \ k = 0, \cdots, n-2 \\ &\cdots \\ p_k^n &= (1-t)p_k^{n-1} + tp_{k+1}^{n-1}; \ \ k = 0 \end{aligned}$$

y agrupando repetidamente se obtiene

p_0^0

$p_1^0 \ p_0^1$

$p_2^0 \ p_1^1 \ p_0^2$

$\cdots$

$p_n^0 \ p_{n-1}^1 \ p_{n-2}^2 \ \cdots p_0^n$

con lo cual

$$c(t) = \sum_{k=0}^{n} p_k^0 B_k^n(t) = \sum_{k=0}^{n-1} p_k^1 B_k^{n-1}(t) = \sum_{k=0}^{0} p_k^n B_k^0(t) = p_0^n$$

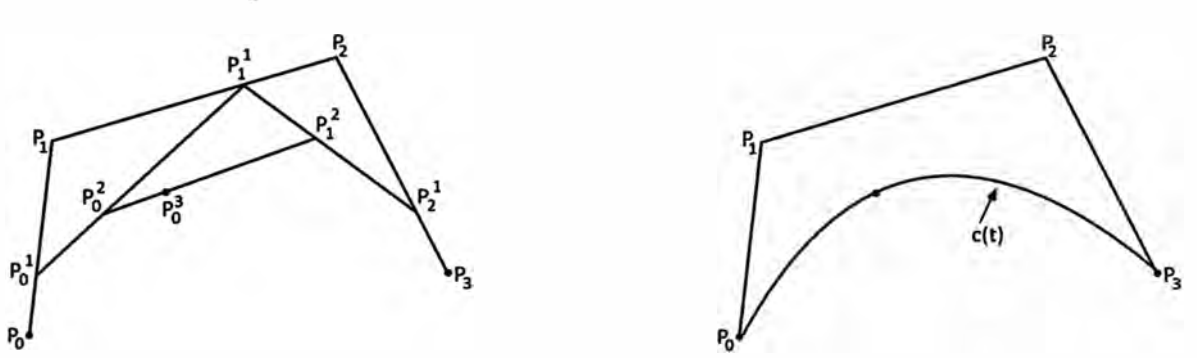

Figura 1.2: Algoritmo de Casteljau con cuatro puntos de control

Ejemplo 1.3. Si tenemos los puntos p_0, p_1, p_2 y elegimos un $t \in]0,1[$ entonces

$$p_0^1 = (1-t)p_0 + tp_1$$

$$p_1^1 = (1-t)p_1 + tp_2$$

Luego

$$p_0^2 = (1-t)p_0^1 + tp_1^1$$

Reemplazando tendremos

$$p_0^2 = (1-t)^2 p_0 + 2t(1-t)p_1 + t^2 p_2 = c(t)$$

si repetimos este proceso para todo $t \in]0,1[$ se genera la Curva de Bezier.

1.3. Superficie de Bezier

Definición 1.7. Una superficie paramétrica es aquella función continua S que aplica D en R^3.

$$S : D \longrightarrow R^3$$

$$(u,v) \rightarrow (x(u,v), y(u,v), z(u,v))$$

donde D es un conjunto abierto conexo en R^2.

Definición 1.8. En una superficie paramétrica $S(u,v)$, si consideramos un valor de $u = \bar{u}$, obtendremos una curva contenida en la superficie.

$$c(v) = S(\bar{u}, v)$$

que se denomina curva isoparamétrica.

Definición 1.9. Una superficie de Bezier es definido por

$$S(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} p_{ij} B_i^n(u) B_j^m(v)$$

a los coeficientes $p_{ij} \in R^3$ se les denomina puntos de control, que forman una malla de control, $B_i^n(u)$ y $B_j^m(v)$ son los polinomios de Bernstein definidos en $[0,1]$.

Ejemplo 1.4. Si $n = m = 1$ se obtendría la superficie bilineal

$$\begin{aligned}
S(u,v) &= \sum_{i=0}^{1}\sum_{j=0}^{1} p_{ij} B_i^1(u) B_j^1(v) \\
S(u,v) &= p_{00} B_0^1(u) B_0^1(v) + p_{01} B_0^1(u) B_1^1(v) + p_{10} B_1^1(u) B_0^1(v) + \\
&\quad p_{11} B_1^1(u) B_1^1(v) \\
S(u,v) &= p_{00}(1-u)(1-v) + p_{01}(1-u)v + p_{10}u(1-v) + p_{11}uv
\end{aligned}$$

□

Propiedad 1.3. Las superficies de Bezier cumplen lo siguiente:

1. Interpolación de los vértices de la malla de control.

$$S(0,0) = p_{00}$$

$$S(0,1) = p_{0m}$$

$$S(1,0) = p_{n0}$$

$$S(1,1) = p_{nm}$$

2. Derivadas Parciales

Derivando

$$S(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} p_{ij} B_i^n(u) B_j^m(v)$$

con respecto a u:

$$\begin{aligned}
\frac{\delta S(u,v)}{\delta u} &= \sum_{i=0}^{n}\sum_{j=0}^{m} p_{ij}\left(\frac{dB_i^n(u)}{du}\right) B_j^m(v) \\
\frac{\delta S(u,v)}{\delta u} &= n\sum_{i=0}^{n}\sum_{j=0}^{m} (p_{i+1,j} - p_{ij}) B_i^{n-1}(u) B_j^m(v)
\end{aligned}$$

con respecto a v

$$\frac{\delta S(u,v)}{\delta v} = m\sum_{i=0}^{n}\sum_{j=0}^{m-1} (p_{i,j+1} - p_{ij}) B_i^n(u) B_j^{m-1}(v)$$

Veamos que ocurre en las curvas frontera o bordes, por ejemplo para la curva isoparamétrica $v = 0$

$$\frac{\delta S(u,0)}{\delta u} = n\sum_{i=0}^{n}\sum_{j=0}^{m} (p_{i+1,0} - p_{i0}) B_i^{n-1}(u)$$

es tangente, en sus segmentos extremos, al polígono formado por los puntos $p_{00}, p_{10}, p_{n0}, p_{n-1,0}$

$$\begin{aligned}
\frac{\delta S(0,0)}{\delta u} &= n(p_{10} - p_{00}) \\
\frac{\delta S(1,0)}{\delta u} &= n(p_{n0} - p_{n-1,0})
\end{aligned}$$

3. Se encuentra en la cápsula convexa

$$\begin{aligned}\sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u)B_j^m(v) &= \sum_{i=0}^{n}(\sum_{j=0}^{m} B_j^m(v))B_i^n(u) \\ &= \sum_{i=0}^{n} B_i^n(u) \\ &= 1\end{aligned}$$

además $B_i^n(u)B_j^m(v) \geq 0$, entonces cualquier punto de la superficie

$$S(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} p_{ij}B_i^n(u)B_j^m(v)$$

es una combinación convexa de los puntos de control p_{ij}.

4. Invarianza Afín

   Debido a que $S(u,v)$ es una combinación convexa, también lo es una combinación afín entonces queda invariante bajo transformaciones afines, es decir para efectuar una transformación afín (rotación, traslación) basta con transformar los puntos de control y construir apartir de estos la nueva superficie.

5. Curvas isoparamétricas

$$c(u) = S(u,\bar{v}) = \sum_{i=0}^{n}\left[\sum_{j=0}^{m} p_{ij}B_j^m(\bar{v})\right] B_i^n(u)$$

   es una curva de Bezier de grado n con puntos de control

$$q_i = \sum_{j=0}^{m} p_{ij}B_j^m(\bar{v}), i = 0, \cdots, n$$

6. Control pseudolocal

   La modificación de un punto de control afecta a la forma de toda la superficie, tiene mayor influencia en las proximidades del punto movido.

Lo mismo que ocurre en la Curva de Bezier, para ejercer un verdadero control en su forma sería necesario utilizar grados elevados de los polinomios de Bernstein.

Si por ejemplo $n = m = 3$, $S(u, v)$ contiene 16 términos y este número crece rapidamente haciendo inviable el cálculo de puntos sobre la superficie, aparte de los conocidos problemas de inestabilidad numérica que ocurren con los polinomios de grado alto.

Esto se resuelve si utilizamos superficies B-splines que además incorpora condiciones de continuidad en cada punto de unión mediante la introducción de nodos múltiples.

2 NURBS

La curva de Bezier presenta algunas deficiencias en cuanto al control local, estos resultados se mejorarían cuando Cox y De Boor empezaron a trabajar para los laboratorios de investigación de la General Motors y encontraron su generalización en la Curva B-Spline.
Otra dificultad era que la representación polinomial es incapaz de modelar circunferencias y las secciones cónicas como la parábola, elipse e hipérbola, así como las superficies generadas por estas, este problema conduce a considerar curvas y superficies racionales de Bezier y las NURBS(Non Uniform Rational B-Spline) que están definidas por la división de polinomios.

2.1. Polinomio Spline

Definición 2.1. Sea $a = y_0 < y_1 < \cdots < y_k < y_{k+1} = b$ y $\Delta = \{y_i\}_0^{k+1}$ una partición del intervalo $[a, b]$ en $k+1$ intervalos, $I_i = [y_i, y_{i+1}[$ tal que $i = 0, \cdots, k-1$ y $I_k = [y_k, y_{k+1}]$, dado un entero positivo m y sea P_m el espacio de los polinomios de orden m, llamaremos a

$$PP_m(\Delta) = \{f \in P_m / p_0, p_1, \cdots, p_k \in P_m; f(x) = p_i(x); x \in I_i; i = 0, \cdots, k\}$$

espacio de polinomio por partes de orden m (grado $m-1$) con nodos $y_1, \cdots, y_k$.

Definición 2.2. Sea $\{u_i\}$ un conjunto de funciones definidas en un intervalo I, y sea $t_1, t_2, \cdots, t_m$ puntos en I tal que $t_1 < t_2 < \cdots < t_m$, se define

$$D\begin{pmatrix} t_1, \cdots, t_m \\ u_1, \cdots, u_m \end{pmatrix} = \begin{Vmatrix} u_1(t_1)u_2(t_1)\cdots u_m(t_1) \\ u_1(t_2)u_2(t_2)\cdots u_m(t_2) \\ \cdots \\ u_1(t_m)u_2(t_m)\cdots u_m(t_m) \end{Vmatrix}$$

Definición 2.3. Dados los puntos $a = t_1 \leq t_2 \leq \cdots \leq t_{r+1} = b$ y la función f definida en $[a, b]$ se define la diferencia dividida de orden r para los puntos $t_1, \cdots, t_{r+1}$ por

$$[t_1, \cdots, t_{r+1}]f = \frac{D\begin{pmatrix} t_1, t_2, \cdots, t_r, t_{r+1} \\ 1, x, \cdots, x^{r-1}, f \end{pmatrix}}{D\begin{pmatrix} t_1, t_2, \cdots, t_r, t_{r+1} \\ 1, x, \cdots, x^{r-1}, x^r \end{pmatrix}}$$

Definición 2.4. Sea [a,b] un intervalo cerrado finito, y sea $\Delta = \{y_i\}_1^k$ con $a = y_0 < y_1 < \cdots < y_k < y_{k+1} = b$ una partición en $k+1$ intervalos $I_i = [y_i, y_{i+1}[; i = 0, 1, \cdots, k-1; I_k = [y_k, y_{k+1}]$ y m un entero positivo con $M = (m_1, \cdots, m_k)$ un vector de enteros con $1 \leq m_i \leq m$, $i = 1, 2, \cdots, k$.

$S(P_m, M, \Delta) = \{S : S_0, S_1, \cdots, S_k \in P_m / S(x) = S_i(x)$ para $x \in I_i, i = 0, 1, \cdots, k$ y $D^j S_{i-1}(y_i) = D^j S_i(y_i)$ para $j = 0, 1, \cdots, m-1-m_i, i = 1, 2, \cdots, k\}$

es el espacio de polinomios splines de orden m cuyos nodos $y_1, y_2, \cdots, y_k$ son de multiplicidad $m_1, m_2, \cdots, m_k$ respectivamente. $M = (m_1, m_2, \cdots, m_k)$ es el vector multiplicidad que nos permite controlar la suavidad de los splines en los nodos.

Para $m_i = m$ los polinomios S_{i-1} y S_i son discontinuos en x_i.

Ejemplo 2.1. Si $M = (m, m, \cdots, m)$ entonces

$$S(P_m, M, \Delta) = PP_m(\Delta)$$

Donde $PP_m(\Delta)$ es el espacio de polinomios por partes. □

Ejemplo 2.2. Si $M = (1, 1, \cdots, 1)$ entonces

$$S(P_m, M, \Delta) = S_m(\Delta)$$

Donde $S_m(\Delta)$ es el espacio de splines de orden m con nodos simples, es decir el $S_m(\Delta)$ es un subespacio de $C^{m-2}[a,b]$. □

Definición 2.5. Sea $\cdots \leq y_{-1} \leq y_0 \leq y_1 \leq y_2 \leq \cdots$ una secuencia de números reales, dado un entero i y $m > 0$ definimos

$$B_i^m(x) = \begin{cases} (-1)^m[y_i, \cdots, y_{i+m}](x-y)_+^{m-1}, y_i < x < y_{i+m} \\ 0, \ \textit{en otro caso} \end{cases}$$

Para todo real x, llamaremos B-Spline de orden m asociado con los nodos $y_i, \cdots, y_{i+m}$.

Ejemplo 2.3. Si $m = 1$ entonces

$$B_i^1(x) = \begin{cases} \frac{1}{y_{i+1}-y_i}, y_i < x < y_{i+1} \\ 0, \ \textit{en otro caso} \end{cases}$$

□

Teorema 2.1. Sea $m \geq 2, y_i < y_{i+m}, \forall x \in R$

$$B_i^m(x) = \frac{(x-y_i)B_i^{m-1}(x) + (y_{i+m}-x)B_{i+1}^{m-1}(x)}{y_{i+m}-y_i}$$

Demostración

Observe que $(x-y)_+^{m-1} = (x-y)(x-y)_+^{m-2}$

Aplicando la regla de Leibnitz's para diferencias divididas de un producto

$$\begin{aligned} B_i^m(x) &= (-1)^m[y_i, \cdots, y_{i+m}](x-y)_+^{m-1} \\ &= (-1)^m[y_i](x-y)[y_i, \cdots, y_{i+m}](x-y)_+^{m-2} \\ &\quad +(-1)^m[y_i, y_{i+1}](x-y)[y_{i+1}, \cdots, y_{i+m}](x-y)_+^{m-2} \end{aligned}$$

Además

$$(-1)^m[y_i,\cdots,y_{i+m}](x-y)_+^{m-2} = \frac{(-1)^{m-1}}{(y_{i+m}-y_i)}\{[y_i,\cdots,y_{i+m-1}](x-y)_+^{m-2} - [y_{i+1},\cdots,y_{i+m}](x-y)_+^{m-2}\}$$

Reemplazando en $B_i^m(x)$

$$\begin{aligned}
B_i^m(x) &= (x-y_i)\frac{(-1)^{m-1}}{y_{i+m}-y_i}\left\{[y_i,\cdots,y_{i+m-1}](x-y)_+^{m-2} - [y_{i+1},\cdots,y_{i+m}](x-y)_+^{m-2}\right\} \\
&\quad +(-1)^m[y_i,y_{i+1}](x-y)[y_{i+1},\cdots,y_{i+m}](x-y)_+^{m-2} \\
B_i^m(x) &= \frac{x-y_i}{y_{i+m}-y_i}B_i^{m-1}(x) - \left\{[y_i,y_{i+1}](x-y) + \frac{x-y_i}{y_{i+m}-y_i}\right\}(-1)^{m-1} \\
&\quad [y_{i+1},\cdots,y_{i+m}](x-y)_+^{m-2} \\
B_i^m(x) &= \frac{x-y_i}{y_{i+m}-y_i}B_i^{m-1}(x) - \left\{\frac{x-y_{i+1}-x+y_i}{y_{i+1}-y_i} + \frac{x-y_i}{y_{i+m}-y_i}\right\}B_{i+1}^{m-1}(x) \\
B_i^m(x) &= \frac{(x-y_i)B_i^{m-1}(x) + (y_{i+m}-x)B_{i+1}^{m-1}(x)}{y_{i+m}-y_i}
\end{aligned}$$

■

Teorema 2.2. Sea $y_i < y_{i+m}$ y D_+ el operador derivada por la derecha, entonces

$$D_+B_i^m(x) = \frac{(m-1)[B_i^{m-1}(x) - B_{i+1}^{m-1}(x)]}{y_{i+m}-y_i}$$

Demostración:

Si y_i o y_{i+m} tienen multiplicidad m, la demostración es trivial, sino

$$\begin{aligned}
D_+B_i^m(x) &= (-1)^m[y_i,\cdots,y_{i+m}]D_+(x-y)_+^{m-1} \\
&= (-1)^m[y_{i+1},\cdots,y_{i+m}](m-1)(x-y)_+^{m-2} \\
&= \frac{(-1)^m(m-1)\left\{[y_{i+1},\cdots,y_{i+m}](x-y)_+^{m-2} - [y_i,\cdots,y_{i+m-1}](x-y)_+^{m-2}\right\}}{y_{i+m}-y_i} \\
&= \frac{(m-1)\left[B_i^{m-1}(x) - B_{i+1}^{m-1}(x)\right]}{y_{i+m}-y_i}
\end{aligned}$$

■

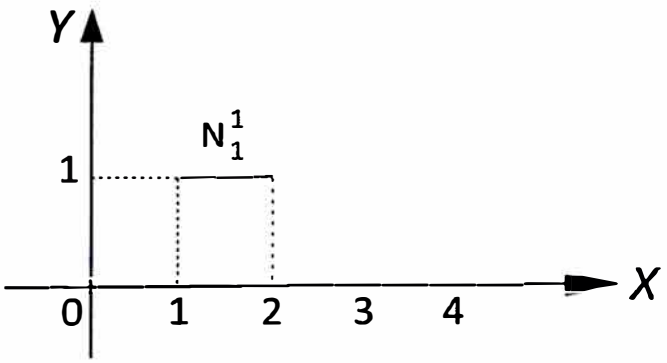

Figura 2.1: B-Spline de Oden 1

Definición 2.6. Sea

$$N_i^m(x) = (y_{i+m} - y_i)B_i^m(x)$$

donde B_i^m es el B-Spline según la definición (2.5), llamaremos a $N_i^m(x)$ el B-Spline normalizado asociado con los nodos $y_i, y_{i+1}, \cdots, y_{i+m}$.

Ejemplo 2.4. Para $m = 1$ el B-Spline normalizado asociado con $y_i < y_{i+1}$ es

$$N_i^1(x) = \begin{cases} 1, y_i \leq x < y_{i+1} \\ 0, \ en\ otro\ caso \end{cases}$$

□

Teorema 2.3. Sea $m \geq 2$, $y_i < y_{i+m}, \forall x \in R$

$$N_i^m(x) = \left(\frac{x - y_i}{y_{i+m-1} - y_i}\right) N_i^{m-1}(x) + \left(\frac{y_{i+m} - x}{y_{i+m} - y_{i+1}}\right) N_{i+1}^{m-1}(x)$$

Demostración: Según la definición (2.6)

$$N_i^m(x) = (y_{i+m} - y_i)B_i^m(x)$$

Por teorema (2.1)

$$\begin{aligned} N_i^m(x) &= (x - y_i)B_i^{m-1}(x) + (y_{i+m} - x)B_{i+1}^{m-1}(x) \\ N_i^m(x) &= \left(\frac{x - y_i}{y_{i+m-1} - y_i}\right) N_i^{m-1}(x) + \left(\frac{y_{i+m} - x}{y_{i+m} - y_{i+1}}\right) N_{i+1}^{m-1}(x) \end{aligned}$$

Propiedad 2.1. Los B-Splines normalizados cumplen lo siguiente:

1. Soporte Local

$$N_i^m(x) = 0, \forall x \notin [y_i, y_{i+m}[$$

Probemos por inducción, para $m = 1$, se cumple por definición $N_i^1(x) = 0, x \notin [y_i, y_{i+1}[$ Supongamos ahora que se cumple para $m = k$ tendremos

$$\begin{aligned} N_i^k(x) &= 0, \forall x \notin [y_i, y_{i+k}[\\ N_{i+1}^k(x) &= 0, \forall x \notin [y_{i+1}, y_{i+k+1}[\end{aligned}$$

Por teorema (2.3)

$$N_i^{k+1}(x) = \left(\frac{x - y_i}{y_{i+k} - y_i}\right) N_i^k(x) + \left(\frac{y_{i+k+1} - x}{y_{i+k+1} - y_{i+1}}\right) N_{i+1}^k(x)$$

para $x \notin [y_i, y_{i+k+1}[$ se obtiene que $N_i^{k+1}(x) = 0$.

2. No negatividad

$$N_i^m(x) \geq 0, \forall x \in [y_i, y_{i+m}[$$

Probemos por inducción,para $m = 1$, se cumple por definición $N_i^1(x) = 1, \forall x \in [y_i, y_{i+1}[$.
Supongamos ahora que se cumple para $m = k$

$$\begin{aligned} N_i^k(x) &\geq 0, \forall x \in [y_i, y_{i+k}[\\ N_{i+1}^k(x) &\geq 0, \forall x \in [y_{i+1}, y_{i+k+1}[\end{aligned}$$

además para $x \in [y_i, y_{i+k+1}[$

$$\begin{aligned} \frac{x - y_i}{y_{i+k} - y_i} &\geq 0 \\ \frac{y_{i+k+1} - x}{y_{i+k+1} - y_{i+1}} &\geq 0 \end{aligned}$$

por tanto por el teorema (2.3)

$$N_i^{k+1}(x) \geq 0, \forall x \in [y_i, y_{i+k+1}[$$

Teorema 2.4. Los B-Splines normalizados forman una partición de la unidad, es decir

$$\sum_{i=j+1-m}^{j} N_i^m(x) = 1, \forall x \in [y_j, y_{j+1}[$$

Demostración:

Demostremos utilizando inducción matemática, para $m = 1$ se cumple, por el ejemplo(2.4). Ahora asumiremos que se cumple para B-Splines de orden $m-1$, entonces por recursión tendremos que

$$\begin{aligned}
\sum_{i=j+1-m}^{j} N_i^m(x) &= \sum_{i=j+1-m}^{j} \left[(x - y_i)B_i^{m-1}(x) + (y_{i+m} - x)B_{i+1}^{m-1}(x)\right] \\
&= \sum_{i=j+2-m}^{j} \left[(x - y_i + y_{i+m-1} - x)B_i^{m-1}(x)\right] \\
&= \sum_{i=j+2-m}^{j} N_i^{m-1}(x) = 1
\end{aligned}$$

■

2.2. Cálculo de los B-Splines Normalizados

Asumiendo que $\{y_0, y_1, \cdots, y_n\}$ es el conjunto de nodos con $y_i = i$ es decir $\{0, 1, \cdots, n\}$.

1. B-Splines Normalizados Constantes

   Para $m = 1$ tendremos

$$N_i^1(x) = \begin{cases} 1, x \in [i, i+1[\\ 0, \ en\ otro\ caso \end{cases}$$

2. B-Splines Normalizados Lineales

   Para $m = 2$ tendremos que

$$\begin{aligned} N_0^2(x) &= \frac{x - y_0}{y_1 - y_0} N_0^1(x) + \frac{y_2 - x}{y_2 - y_1} N_1^1(x) \\ &= xN_0^1(x) + (2 - x)N_1^1(x) \\ &= \begin{cases} x, 0 \leq x < 1 \\ 2 - x, 1 \leq x < 2 \\ 0, \ en\ otro\ caso \end{cases} \end{aligned}$$

$$\begin{aligned} N_1^2(x) &= \frac{x - y_1}{y_2 - y_1} N_1^1(x) + \frac{y_3 - x}{y_3 - y_2} N_2^1(x) \\ &= (x - 1)N_1^1(x) + (3 - x)N_2^1(x) \\ &= \begin{cases} x - 1, 1 \leq x < 2 \\ 3 - x, 2 \leq x < 3 \\ 0, \ en\ otro\ caso \end{cases} \end{aligned}$$

   Análogamente tendremos que

$$N_2^2(x) = \begin{cases} x - 2, 2 \leq x < 3 \\ 4 - x, 3 \leq x < 4 \\ 0, \ en\ otro\ caso \end{cases}$$

   Con lo que $N_i^2(x) = N_0^2(x - i)$

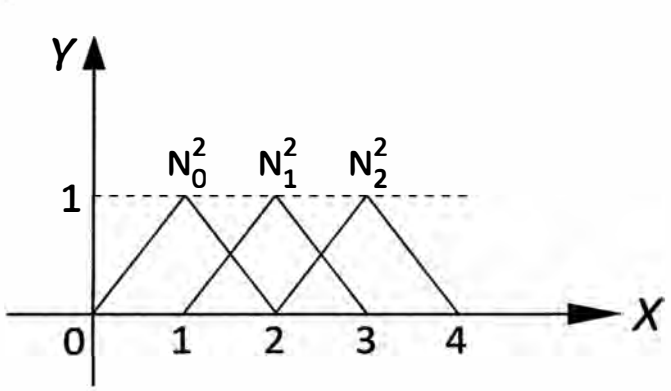

Figura 2.2: B-Spline Lineales

3. B-Splines Normalizados Cuadráticos

   Para $m = 3$ obtendremos

$$
\begin{aligned}
N_0^3(x) &= \frac{x - y_0}{y_2 - y_0} N_0^2(x) + \frac{y_3 - x}{y_3 - y_1} N_1^2(x) \\
&= \frac{x}{2} N_0^2(x) + \frac{3 - x}{2} N_1^2(x) \\
&= \begin{cases} \frac{x^2}{2}, 0 \leq x < 1 \\ \frac{3}{4} - (x - \frac{3}{2})^2, 1 \leq x < 2 \\ \frac{(3-x)^2}{2}, 2 \leq x < 3 \\ 0, \ \textit{en otro caso} \end{cases}
\end{aligned}
$$

   Además $N_i^3(x) = N_0^3(x - i)$

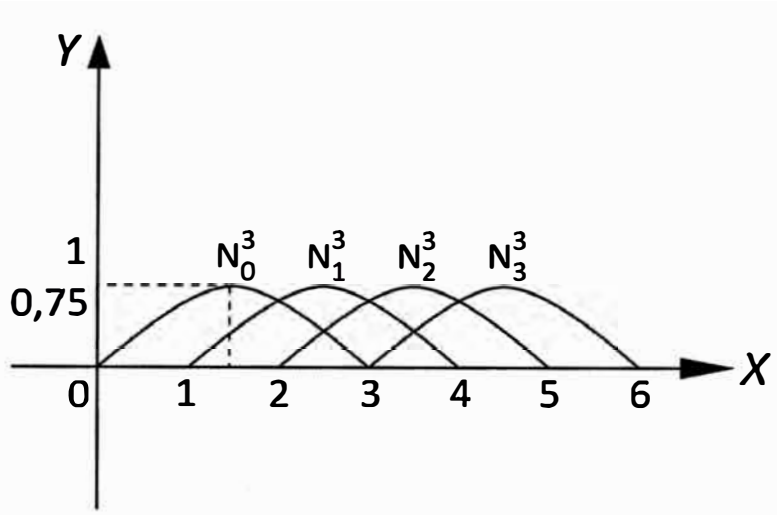

Figura 2.3: B-Spline Cuadráticos

4. B-Splines Normalizados Cúbicos

   Para $m = 4$ obtendremos

$$
\begin{aligned}
N_0^4(x) &= \left(\frac{x - y_0}{y_3 - y_0}\right) N_0^3(x) + \left(\frac{y_4 - x}{y_4 - y_1}\right) N_1^3(x) \\
&= \frac{x}{3} N_0^3(x) + \frac{4 - x}{3} N_1^3(x) \\
&= \begin{cases}
\frac{x^3}{6}, 0 \leq x < 1 \\
\frac{1}{6}(-3x^3 + 12x^2 - 12x + 4), 1 \leq x < 2 \\
\frac{1}{6}(3x^3 - 24x^2 + 60x - 44), 2 \leq x < 3 \\
\frac{1}{6}(4 - x)^3, 3 \leq x < 4 \\
0, \ \textit{en otro caso}
\end{cases} \\
&= \begin{cases}
\frac{x^3}{6}, 0 \leq x < 1 \\
\frac{1}{6}(3(2 - x)^3 - 6(2 - x)^2 + 4), 1 \leq x < 2 \\
\frac{1}{6}(3(x - 2)^3 - 6(x - 2)^2 + 4), 2 \leq x < 3 \\
\frac{1}{6}(4 - x)^3, 3 \leq x < 4 \\
0, \ \textit{en otro caso}
\end{cases}
\end{aligned}
$$

Con lo que

$$N_0^4(x) = \begin{cases} f(1-x), 0 \leq x < 1 \\ g(2-x), 1 \leq x < 2 \\ g(x-2), 2 \leq x < 3 \\ f(x-3), 3 \leq x < 4 \\ 0, \ en\ otro\ caso \end{cases}$$

Donde

$$\begin{aligned} f(x) &= \frac{1}{6}(1-x)^3 \\ g(x) &= \frac{1}{6}(3x^3 - 6x^2 + 4) \end{aligned}$$

$N_0^4(x)$ es simétrica con respecto a $x = 2$.

Definición 2.7. Una curva B-Spline $c(x)$, es definida por

$$c(x) = \sum_{i=0}^{n} p_i N_i^m(x), x \in [y_{m-1}, y_{n+1}[$$

donde

1. $\{p_i : i = 0, ..., n\}$ son los puntos de control o puntos de Boor.
2. m es el orden de los B-Splines normalizados y $m - 1$ su grado.
3. $\{y_i : i = 0, ..., n + m\}$ es el conjunto de nodos con $y_i \leq y_{i+1}$
4. $$N_i^1(x) = \begin{cases} 1, x \in [y_i, y_{i+1}[\\ 0, \ en\ otro\ caso \end{cases}$$

   Si $m > 1$

$$N_i^m(x) = \left(\frac{x - y_i}{y_{i+m-1} - y_i}\right) N_i^{m-1}(x) + \left(\frac{y_{i+m} - x}{y_{i+m} - y_{i+1}}\right) N_{i+1}^{m-1}(x)$$

Si algún denominador es cero entonces toda la fracción debe ser considerada como cero.

Para la elección de los nodos y con la finalidad de interpolar en los puntos extremos elegiremos el vector de nodos estándar o de Clamped, definida por

$$y_i = \begin{cases} 0, i < m \\ i - m + 1, m \leq i \leq n \\ n - m + 2, i > n \end{cases}$$

Donde

$$0 \leq i \leq m + n$$

$$0 \leq x \leq n - m + 2$$

con lo cual vemos que necesitamos $n + m + 1$ nodos y $n + 1$ puntos de control para tener una curva de grado $m - 1$, de la desigualdad anterior se tiene que $n + 1 \geq m$.

Teorema 2.5. (Algoritmo de De Boor)

Dada la curva B-Spline

$$c(x) = \sum_{i=0}^{n} p_i N_i^m(x), x \in [y_{m-1}, y_{n+1}[$$

si definimos

$$p_i^{(j)}(x) = \begin{cases} p_i; j = 0 \\ \left(\frac{y_{i+m-j} - x}{y_{i+m-j} - y_i}\right) p_{i-1}^{(j-1)}(x) + \left(\frac{x - y_i}{y_{i+m-j} - y_i}\right) p_i^{(j-1)}(x); j \geq 1 \end{cases}$$

Entonces

$$c(x) = \sum_{i=m-1}^{n} p_i^{(m-1)}(x) N_i^1(x), x \in [y_{m-1}, y_{n+1}[$$

Si $x \in [y_k, y_{k+1}[, k = m - 1, ..., n$ entonces $c(x) = p_k^{(m-1)}(x)$

Demostración:

Sea $c(x) = \sum_{i=r}^{n} p_i^{(r)}(x) N_i^{m-r}(x)$ donde $r = 0, 1, ..., m-1$

Probemos por inducción, si $r = 0$ entonces $c(x) = \sum_{i=0}^{n} p_i(x) N_i^m(x)$.

Supongamos ahora que se cumple para $r = m-2$, es decir

$$
\begin{aligned}
c(x) &= \sum_{i=m-2}^{n} p_i^{(m-2)}(x) N_i^2(x) \\
c(x) &= \sum_{i=m-2}^{n} p_i^{(m-2)}(x) \left[\left(\frac{x - y_i}{y_{i+1} - y_i} \right) N_i^1(x) + \left(\frac{y_{i+2} - x}{y_{i+2} - y_{i+1}} \right) N_{i+1}^1(x) \right] \\
c(x) &= p_{m-2}^{(m-2)}(x) \left(\frac{x - y_{m-2}}{y_{m-1} - y_{m-2}} \right) N_{m-2}^1(x) + \sum_{i=m-1}^{n} p_i^{(m-2)}(x) \left(\frac{x - y_i}{y_{i+1} - y_i} \right) N_i^1(x) + \\
&\quad \sum_{i=m-2}^{n-1} p_i^{(m-2)}(x) \left(\frac{y_{i+2} - x}{y_{i+2} - y_{i+1}} \right) N_{i+1}^1(x) + \left(\frac{y_{n+2} - x}{y_{n+2} - y_{n+1}} \right) N_{n+1}^1(x)
\end{aligned}
$$

Como $x \in [y_{m-1}, y_{n+1}[$ se tiene que $N_{m-2}^1(x) = 0$ y $N_{n+1}^1(x) = 0$, además si cambiamos de índices tendremos

$$
\begin{aligned}
c(x) &= \sum_{i=m-1}^{n} p_i^{(m-2)}(x) \left(\frac{x - y_i}{y_{i+1} - y_i} \right) N_i^1(x) + \sum_{i=m-1}^{n} p_{i-1}^{(m-2)}(x) \left(\frac{y_{i+1} - x}{y_{i+1} - y_i} \right) N_i^1(x) \\
c(x) &= \sum_{i=m-1}^{n} \left[p_i^{(m-2)}(x) \left(\frac{x - y_i}{y_{i+1} - y_i} \right) + p_{i-1}^{(m-2)}(x) \left(\frac{y_{i+1} - x}{y_{i+1} - y_i} \right) \right] N_i^1(x) \\
c(x) &= \sum_{i=m-1}^{n} p_i^{(m-1)}(x) N_i^1(x); x \in [y_{m-1}, y_{n+1}[
\end{aligned}
$$

Como

$$
\begin{aligned}
N_k^1(x) &= \begin{cases} 1, & x \in [y_k, y_{k+1}[, k = m-1, ..., n \\ 0, & \textit{en otro caso} \end{cases} \\
c(x) &= p_i^{(m-1)}(x), x \in [y_k, y_{k+1}[
\end{aligned}
$$

■

Propiedad 2.2. Las curvas B-Splines cumplen lo siguiente:

1. Interpolación en los puntos extremos

$$c(y_0) = p_0; c(y_n) = p_n$$

Según la definición (2.7)

$$c(x) = p_0 N_0^m(x) + ... + p_{n-m+1} N_{n-m+1}^m(x) + ... + p_n N_n^m(x)$$

por teorema (2.4) y considerando el soporte de los B-Splines tendremos

$$c(y_0) = p_0; c(y_n) = p_{n-m+1}$$

si consideramos el vector de nodos estándar(Clamped) en donde los nodos extremos son de multiplicidad m entonces

$$c(y_n) = p_{n-m+1} = p_n$$

En particular, si el soporte $\{0, ..., 0, 1, ..., 1\}$ con m ceros y m unos, la curva B-Spline es una curva de Bezier, ya que

$$N_i^k(t) = \begin{pmatrix} k-1 \\ i \end{pmatrix} t^i (1-t)^{k-1-i} = B_i^{k-1}(t), i = 0, ..., k-1$$

2. Control Local

Sea la curva B-Spline con puntos de control $p_0, p_1, ..., p_n$

$$c(x) = \sum_{i=0}^{n} p_i N_i^m(x)$$

y sea $\bar{c}(x)$ la curva obtenida cambiando un único punto de control, digamos p_k

$$\bar{c}(x) = \sum_{i=0}^{n} \bar{p}_i N_i^m(x), \bar{p}_i = p_i, i \neq k$$

la diferencia entre ambas curvas será

$$\bar{c}(x) - c(x) = (\bar{p}_k - p_k)N_k^m(x)$$

Según la propiedad (2.1) tenemos que $N_k^m(x) = 0, \forall x \notin [y_k, y_{k+m}[$ entonces

$$\bar{c}(x) - c(x) = 0, \forall x \notin [y_k, y_{k+m}[$$

esto significa que ambas curvas son idénticas salvo en el intervalo $[y_k, y_{k+m}[$. Sabemos que $0 \leq N_i^m(x) \leq 1, \forall x \in [y_0, y_{n+m}]$ con lo cual

$$|| \bar{c}(x) - c(x) || = || \bar{p}_k - p_k || N_k^m(x) \leq || \bar{p}_k - p_k ||$$

3. Combinación afín

$$c(x) = \sum_{i=j+1-m}^{j} p_i N_i^m(x); j \geq m-1; x \in [y_j; y_{j+1}[$$

es una combinación afín de sus puntos de control ya que por el teorema (2.4)

$$\sum_{i=j+1-m}^{j} N_i^m(x) = 1$$

4. Invarianza afín

Por teorema (1.1), la curva B-Spline normalizada

$$c(x) = \sum_{i=j+1-m}^{j} p_i N_i^m(x); j \geq m-1; x \in [y_j; y_{j+1}[$$

es afin invariante, es decir, siendo

$$\phi : R^n \longrightarrow R^n$$

una transformación afín

$$\phi(c(x)) = \sum_{i=j+1-m}^{j} \phi(p_i) N_i^m(x)$$

5. Combinación convexa

$$c(x) = \sum_{i=j+1-m}^{j} p_i N_i^m(x); j \geq m-1; x \in [y_j; y_{j+1}[$$

es una combinación convexa ya que es un combinación afín y $N_i^m(x) \geq 0$.

Definición 2.8. Una superficie B-Spline es definido por

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} p_{ij} N_i^p(u) N_j^q(v)$$

donde $N_i^p(u)$, $N_j^q(v)$ son B-Spline de orden p y q respectivamente y p_{ij} son los puntos de control.

Propiedad 2.3. En una superficie B-Spline se cumple las siguientes propiedades:

1. Interpolación en las esquinas de la malla de control, es decir

$$S(0,0) = p_{00}$$

$$S(0,1) = p_{0m}$$

$$S(1,0) = p_{n0}$$

$$S(1,1) = p_{nm}$$

esto se obtiene si consideramos los soportes

$$S_u = \{u_0, ..., u_r\}$$

$$S_v = \{v_0, ..., v_s\}$$

siendo ambos no periódicos, es decir con los nodos extremos de multiplicidad $p+1$ y $q+1$ respectivamente.

2. Derivadas parciales en las esquinas de la malla de control

$$\begin{aligned}\frac{\delta S(u,v)}{\delta u} &= \sum_{i=0}^{n}\sum_{j=0}^{m} p_{ij}(\frac{dN_i^p(u)}{du})N_j^q(v) \\ &= \sum_{j=0}^{m}[\sum_{i=0}^{n} p_{ij}(\frac{dN_i^p(u)}{du})]N_j^q(v)\end{aligned}$$

para $u = 0, u = 1$ tendremos

$$\begin{aligned}\frac{\delta S(0,v)}{\delta u} &= \frac{p}{u_{p+1} - u_1}\sum_{j=0}^{m}(p_{1j} - p_{0j})N_j^q(v) \\ \frac{\delta S(1,v)}{\delta u} &= \frac{p}{u_{r-1} - u_n}\sum_{j=0}^{m}(p_{nj} - p_{n-1,j})N_j^q(v)\end{aligned}$$

Para $v = 0, v = 1$ tendremos

$$\begin{aligned}\frac{\delta S(0,0)}{\delta u} &= \frac{p}{u_{p+1} - u_1}(p_{10} - p_{00}) \\ \frac{\delta S(0,1)}{\delta u} &= \frac{p}{u_{p+1} - u_1}(p_{1m} - p_{0m}) \\ \frac{\delta S(1,0)}{\delta u} &= \frac{p}{u_{r-1} - u_n}(p_{n0} - p_{n-1,0}) \\ \frac{\delta S(1,1)}{\delta u} &= \frac{p}{u_{r-1} - u_n}(p_{nm} - p_{n-1,m})\end{aligned}$$

esto significa que las curvas isoparamétricas $v = 0$ y $v = 1$ son tangentes a los segmentos primero y último de los polígonos formados por $p_{00}, p_{10}, \cdots, p_{nm}$ y $p_{0m}, p_{1m}, \ldots, p_{nm}$ respectivamente.

3. Control Local

   Si un punto de control p_{kh} es modificado a $\bar{p}_{kh}$ entonces

$$S(u,v) - \bar{S}(u,v) = (p_{kh} - \bar{p}_{kh}) N_k^p(u) N_h^q(v)$$

   donde $N_k^p(u) \neq 0$ para $u \in [u_k, u_{k+p}[$ y $N_h^q(v) \neq 0$ para $v \in [v_h, v_{h+q}[$, es decir las superficies sólo se diferencian en $[u_k, u_{k+p}[\times [v_h, v_{h+q}[$.

4. Se encuentra en la cápsula convexa

   Sea $u = \bar{u} \in [u_k, u_{k+1}[, v = \bar{v} \in [v_h, v_{h+1}[$ los unicos B-Splines no nulos son $N_{k-p}^p(u), N_{k-p+1}^p(u), ..., N_k^p(u)$ y $N_{h-q}^q(v), N_{h-q+1}^q(v), ..., N_h^q(v)$ con lo cual

$$S(\bar{u},\bar{v}) = \sum_{i=k-p}^{k} \sum_{j=h-q}^{h} p_{ij} N_i^p(\bar{u}) N_j^q(\bar{v})$$

$$con \sum_{i=k-p}^{k} \sum_{j=h-q}^{h} N_i^p(u) N_j^q(v) = 1; N_i^p(u) N_j^q(v) \geq 0$$

   es una combinación convexa sólo de los puntos de control p_{ij} con $k-p \leq i \leq k, h-q \leq j \leq h$, en consecuencia $S(\bar{u},\bar{v})$ se encuentra en la cápsula convexa de sus puntos de control.

5. Invarianza afín

   Ya que $S(\bar{u},\bar{v}) = \sum_{i=k-p}^{k} \sum_{j=h-q}^{h} N_i^p(\bar{u}) N_j^q(\bar{v}) = 1$ entonces $S(u,v)$ es una combinación afín, por lo tanto para efectuar una transformación afín basta con realizarla sobre los puntos de control y combinar los transformados con las funciones base para obtener la superficie transformada.

6. Superficies de Bezier como B-Splines

   Si los soportes no contienen nodos interiores, es decir, $S_u = \{0, ..., 0, 1, .., 1\}$, 0 y 1 de multiplicidad p y $S_v = \{0, ..., 0, 1, .., 1\}$, 0 y 1 de multiplicidad q entonces la superficie B-Spline es una superficie

de Bezier de orden pxq, porque los B-Spline asociados a cada soporte son los polinomios de Bernstein.

2.3. Curvas Racionales

Las curvas polinomiales son incapaces de modelar circunferencias y todas las secciones cónicas ya que no pueden representarse de forma paramétrica como un polinomio, por ejemplo una circunferencia de centro $(0,0)$ y radio 1 es parametrizada por

$$\begin{aligned} x &= \frac{1-t^2}{1+t^2} \\ y &= \frac{2t}{1+t^2} \end{aligned}$$

2.4. Curvas Racionales como Proyecciones

Una curva racional en el plano es aquella con ecuación paramétrica

$$c(t) = \begin{cases} x = \frac{P(t)}{W(t)} \\ y = \frac{Q(t)}{S(t)} \end{cases}$$

donde $t \in [a,b]$ y $P(t), Q(t), W(t)$ y $S(t)$ son polinomios.
Supongamos que $W(t) = S(t)$ y reemplazando en la curva $c(t)$ tendremos que

$$c(t) = \begin{cases} x = \frac{P(t)}{W(t)} \\ y = \frac{Q(t)}{W(t)} \end{cases}$$

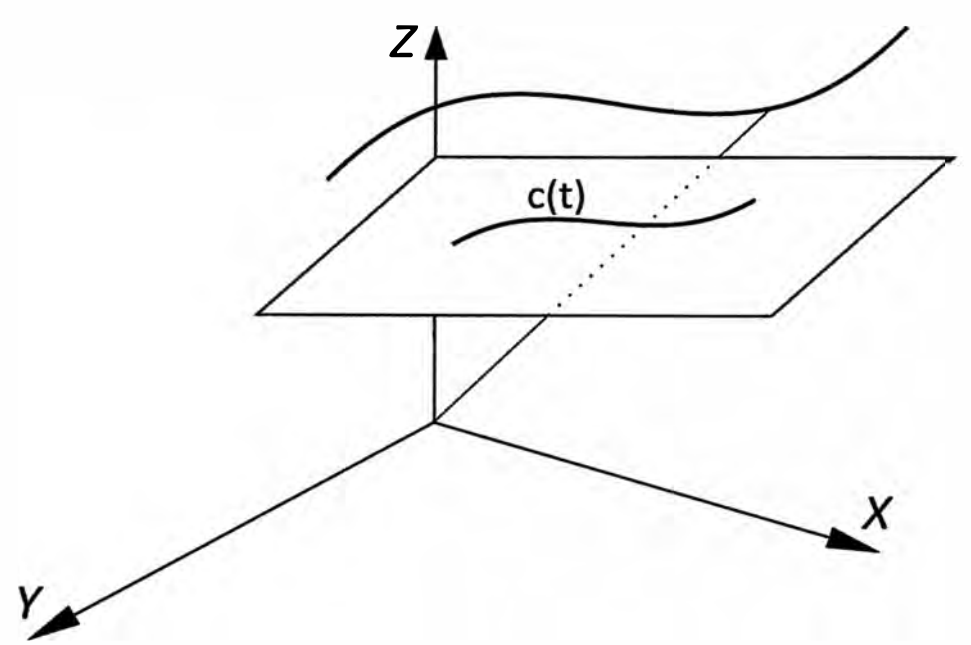

Figura 2.4: Proyección sobre el plano $Z = 1$

Consideremos ahora la curva polinomial $\bar{c}(t)$ en R^3 donde

$$\bar{c}(t) = \begin{cases} \bar{x} = P(t) \\ \bar{y} = Q(t) \\ \bar{z} = W(t) \end{cases}$$

Si proyectamos la curva sobre el plano $Z = 1$ utilizando el origen como centro de la proyección, es decir si $(\bar{x}_0, \bar{y}_0, \bar{z}_0)$ es un punto de $\bar{c}(t)$, entonces la proyección es

$$x_0 = \frac{\bar{x}_0}{\bar{z}_0}$$
$$y_0 = \frac{\bar{y}_0}{\bar{z}_0}$$

Por lo tanto la proyección sobre el plano $Z = 1$ de la curva polinomial $\bar{c}(t)$ definida en R^3 es la curva racional $c(t)$.

2.5. Curva Racional de Bezier

Definición 2.9. Sean p_k para $k = 0, ..., n$ los puntos de control, entonces

$$c(t) = \sum_{k=0}^{n} p_k R_k^n(t)$$

es llamado Curva Racional de Bezier, y para $k = 0, ..., n$ los

$$R_k^n(t) = \frac{w_k B_k^n(t)}{\sum_{k=0}^{n} w_k B_k^n(t)}$$

son las funciones base de la Curva Racional de Bezier, $B_k^n(t)$ son los polinomios de Bernstein y los $w_k > 0$ son llamados pesos.

Propiedad 2.4. Las Curvas Racionales de Bezier cumplen las siguientes propiedades:

1. Interpolación de los puntos de control extremos.

$$c(0) = p_0$$

$$c(1) = p_n$$

2. Tangencia al poligono de control en sus segmentos extremos.

$$\begin{aligned} c'(0) &= n\frac{w_1}{w_0}(p_1 - p_0) \\ c'(1) &= n\frac{w_{n-1}}{w_n}(p_n - p_{n-1}) \end{aligned}$$

3. Las funciones base son una partición de la unidad, es decir

$$\sum_{k=0}^{n} R_k^n(t) = 1$$

4. La curva se encuentra en la cápsula convexa de sus puntos de control ya que

$$R_k^n(t) \geq 0, k = 0, ..., n$$

esto debido a que los pesos w_k son positivos.

5. Control Pseudolocal

Si se modifica un punto de control digamos p_i para $i = 0, ..., n$ la nueva curva se diferencia de la original en un múltiplo de $R_i^n(t)$

$$c(t) - \bar{c}(t) = (p_i - \bar{p}_i)R_i^n(t)$$

La funcion $R_i^n(t)$ alcanza unicamente un máximo en $[0, 1]$, por lo tanto, en valores del parámetro t lejanos a este máximo la diferencia es pequeña.

6. Invarianza Afín

Para efectuar una transformación afín, basta con aplicarla sobre los puntos de control y obtener con ellos la curva transformada, esto debido a que las funciones base forman una partición de la unidad.

7. Secciones Cónicas

Todas la secciones cónicas (parábolas, elipses e hipérbolas) son curvas racionales de Bezier formado por una división de polinomios de segundo grado.

8. Si los pesos son $w_k = 1, k = 0, ..., n$ obtenemos la curva de Bezier, esto debido a que el denominador de $R_k^n(t)$ es uno

$$\sum_{k=0}^{n} w_k B_k^n(t) = \sum_{k=0}^{n} B_k^n(t) = 1$$

2.6. Superficie Racional de Bezier

Definición 2.10. Sean los p_{ij} con $i = 0, ..., n$ y $j = 0, ..., m$ los puntos de control, entonces

$$c(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} p_{ij} R_{ij}^{nm}(u, v)$$

es llamado Superficie Racional de Bezier con

$$R_{ij}^{nm} = \frac{w_{ij} B_i^n(u) B_j^m(v)}{\sum_{i=0}^{n} \sum_{j=0}^{m} w_{ij} B_i^n(u) B_j^m(v)}$$

denominado función base de la Superficie Racional de Bezier, $B_i^n(u), B_j^m(v)$ son polinomios de Bernstein y $w_{ij} > 0$ son los pesos.

Propiedad 2.5. Las Superficies Racionales de Bezier cumplen las siguientes propiedades:

1. Las funciones base son una partición de la unidad.

$$\sum_{i=0}^{n} \sum_{j=0}^{m} R_{ij}^{nm}(u, v) = 1$$

2. Es una combinación convexa de sus puntos de control.
3. Si $w_{ij} = 1$ tendremos la Superficie de Bezier.
4. Es invariante bajo transformaciones afines.

Si en particular $v = \bar{v}$ sería una curva contenida en la superficie.
Las curvas racionales más usadas son las conocidas como NURBS incluyen un peso al igual que las Curvas Racionales de Bezier que le da al usuario un grado de libertad adicional para tener un mayor control sobre la curva.

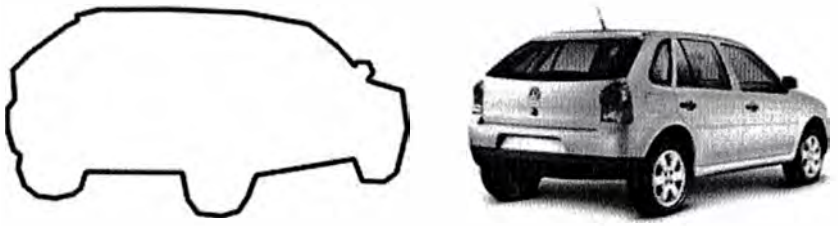

Figura 2.5: Contorno

2.7. Curva NURBS

Definición 2.11. Sean p_k para $k = 0, ..., n$ los puntos de control, entonces

$$c(t) = \sum_{k=0}^{n} p_k Q_k^m(t)$$

es llamado Curva NURBS, y para $k = 0, ..., n$ los

$$Q_k^m(t) = \frac{w_k N_k^m(t)}{\sum_{k=0}^{n} w_k N_k^m(t)}$$

son las funciones base de la curva NURBS, $N_k^m(t)$ son los polinomios de B-Spline Normalizados y los $w_k > 0$ son llamados pesos.

Propiedad 2.6. Las curvas NURBS cumplen lo siguiente:

1. Interpolación en los puntos extremos

$$c(0) = \frac{w_0 p_0}{w_0} = p_0$$

$$c(1) = \frac{w_n p_n}{w_n} = p_n$$

2. Tangencia al polígono de control en sus segmentos extremos

   Expresando las curvas NURBS como división de curvas B-Spline polinomiales

con puntos de control $\{w_k p_k\}$ y $\{w_k\}$:

$$\begin{aligned} c(t) &= \frac{S(u)}{W(u)} \\ c'(t) &= \frac{W(t)S'(t) - W'(t)S(t)}{W^2(t)} \end{aligned}$$

De donde

$$\begin{aligned} c'(0) &= \frac{W(0)S'(0) - W'(0)S(0)}{W^2(0)} \\ &= \frac{w_0 S'(0) - W'(0) w_0 p_0}{w_0^2} \\ &= \frac{S'(0) - W'(0) p_0}{w_0} \end{aligned}$$

Por derivada a de B-Spline no racional

$$\begin{aligned} S'(0) &= c(w_1 p_1 - w_0 p_0), c \in R \\ W'(0) &= c(w_1 - w_0), c \in R \end{aligned}$$

Entonces

$$\begin{aligned} c'(0) &= c\frac{(w_1 p_1 - w_0 p_0) - (w_1 - w_0) p_0}{w_0}, c \in R \\ &= c\frac{w_1}{w_0}(p_1 - p_0) \end{aligned}$$

y analogamente

$$c'(1) = r\frac{w_{n-1}}{w_n}(p_n - p_{n-1}), r \in R$$

Por lo tanto la curva es tangente al primer y último segmento del polígono de control.

3. Control Local

   Si el punto de control p_j es modificado a $\bar{p}_j$, la diferencia entre las curvas es

$$c(t) - \bar{c}(t) = (p_j - \bar{p}_j)Q_j^m(t) = (p_j - \bar{p}_j)\frac{w_j N_j^m(t)}{\sum_{j=0}^{n} w_i N_i^m(t)}$$

   entonces ambas curvas son iguales para $t \notin [t_j, t_{j+n}[$ ya que $N_j^n(t) = 0$ para $t \notin [t_j, t_{j+n}[$, los mismo ocurre si se modifica un peso $w_j, j = 0, ..., n$.

4. Las funciones base son una partición de la unidad, es decir

$$\sum_{k=0}^{n} Q_k^m(t) = 1$$

5. La curva se encuentra en la cápsula convexa de sus puntos de control ya que

$$Q_k^m(t) \geq 0, k = 0, ..., n$$

   esto debido a que los pesos w_k son positivos.

6. Si los pesos $w_i = 1, \forall i = 0, ..., n$ entonces la curva NURBS es igual a la curva B-Spline.

7. En un nodo de multiplicidad r_k, una curva NURBS es de clase C^{m-r_k-1}.

2.8. Superficie NURBS

Definición 2.12. Sean los p_{ij} con $i = 0, ..., n$ y $j = 0, ..., m$ los puntos de control, entonces

$$c(u, v) = \sum_{i=0}^{n}\sum_{j=0}^{m} p_{ij} Q_{ij}^{pq}(u, v)$$

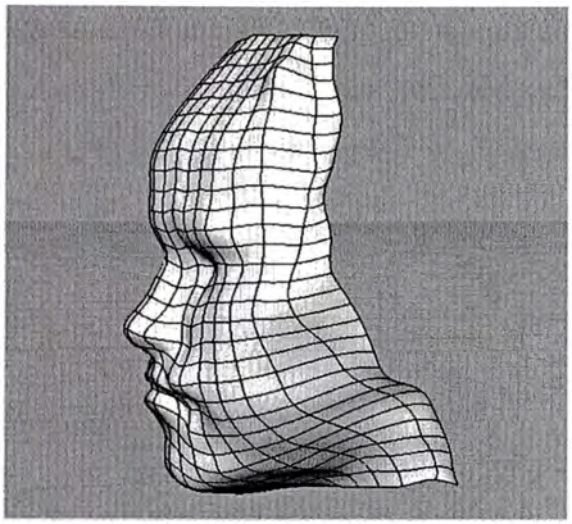

Figura 2.6: Superficie

es llamado Superficie NURBS con

$$Q_{ij}^{nm} = \frac{w_{ij} N_i^p(u) N_j^q(v)}{\sum_{i=0}^{n} \sum_{j=0}^{m} w_{ij} N_i^p(u) N_j^q(v)}$$

denominado función base de la Superficie NURBS, $N_i^p(u), N_j^q(v)$ son polinomios de B-Spline Normalizados y $w_{ij} > 0$ son los pesos.

Propiedad 2.7. Las Superficies NURBS cumplen lo siguiente:

1. Las funciones base son una partición de la unidad.

$$\sum_{i=0}^{n} \sum_{j=0}^{m} Q_{ij}^{nm}(u, v) = 1$$

2. Es una combinación convexa de sus puntos de control.
3. Si $w_{ij} = 1$ tendremos la Superficie de B-Spline.
4. Es invariante bajo transformaciones afines.

Las superficies NURBS son generalizaciones de las superficies racionales y polinomiales. Si en particular $u = \bar{u}$ sería una curva NURBS contenida en la superficie NURBS.

Mediante NURBS se puede modelar todo tipo de contornos que se pueden hacer mediante B-Spline pero cuando se desea obtener una circunferencia o una cónica es necesario usar NURBS ya que su parametrización es mediante la siguiente división de dos polinomios cuadráticos

$$c(t) = \frac{p_0(1-t^2) + 2wp_1t(1-t) + p_2t^2}{(1-t^2) + 2wt(1-t) + t^2}$$

Donde el peso $w > 0$ determina el tipo de cónica.

Ejemplo 2.5. Circunferencia como composición de tres arcos
Necesitaremos siete puntos de control ($n = 6$) y como es una división de dos polinomios cuadráticos, el orden es tres ($m = 3$), por lo tanto se necesita $m+n+1 = 10$ nodos, consideremos los siguiente nodos

$$0, 0, 0, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, 1, 1, 1$$

con puntos

$$p_0 = \left(\frac{\sqrt{3}}{2}, \frac{1}{2}\right), p_1 = (0, 2), p_2 = \left(-\frac{\sqrt{3}}{2}, \frac{1}{2}\right), p_3 = (-\sqrt{3}, -1), p_4 = (0, -1), p_5 = (\sqrt{3}, -1)$$

además $p_6 = p_0$ lo cual nos permite cerrar la curva, y por cada punto de control asignemos los siguientes pesos

$$w_0 = 1, w_1 = \frac{1}{2}, w_2 = 1, w_3 = \frac{1}{2}, w_4 = 1, w_5 = \frac{1}{2}, w_6 = 1$$

Ejemplo 2.6. Circunferencia como composición de cuatro arcos
Necesitaremos nueve puntos de control ($n = 8$) y orden tres ($m = 3$), por lo tanto se necesita $m + n + 1 = 12$ nodos, consideremos los siguientes nodos

$$0, 0, 0, \frac{1}{4}, \frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}, \frac{3}{4}, 1, 1, 1$$

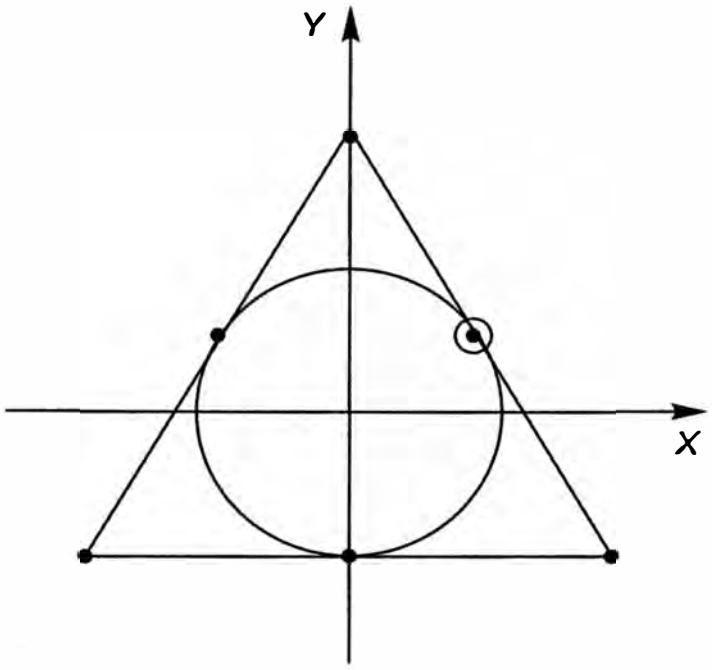

Figura 2.7: Circunferencia con siete puntos de control

con puntos

$$p_0 = (1,0), p_1 = (1,1), p_2 = (0,1), p_3 = (-1,1), p_4 = (-1,0), p_5 = (-1,-1)$$

$$p_6 = (0,-1), p_7 = (1,-1), p_8 = p_0$$

y por cada punto de control asignemos los siguientes pesos

$$w_0 = 1, w_1 = \frac{\sqrt{2}}{2}, w_2 = 1, w_3 = \frac{\sqrt{2}}{2}, w_4 = 1, w_5 = \frac{\sqrt{2}}{2}, w_6 = 1, w_7 = \frac{\sqrt{2}}{2}, w_8 = 1$$

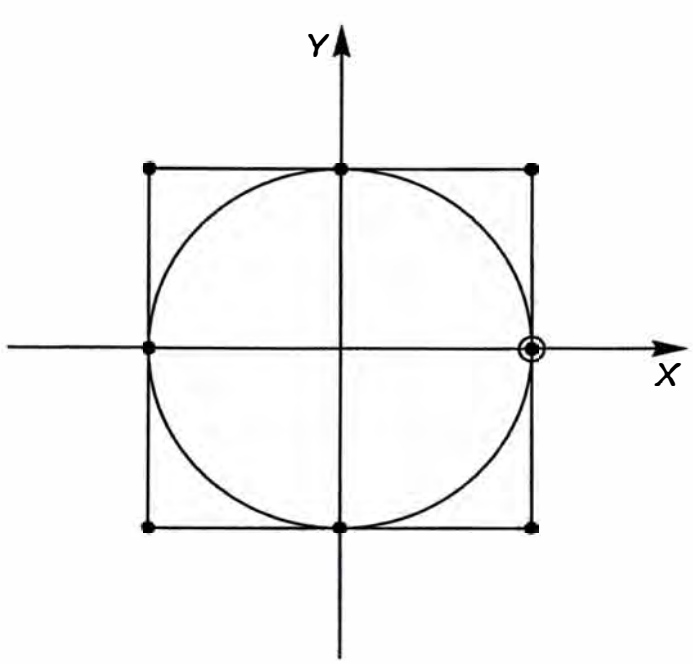

Figura 2.8: Circunferencia con nueve puntos de control

2.9. Aproximación del Contorno de un Objeto

Si tenemos el contorno original de un objeto representado por la figura (2.9), en el diseño del contorno muchas veces se presentan casos de deformaciones o desgastes en los cuales es necesario contar con curvas que nos permitan corregir este defecto pero sin perder la suavidad de esta, veamos dos casos:

2.9.1. Caso 1: Contorno con diferente concavidad

Un primer caso de esta prueba sería el contorno dado en la figura (2.10) en el cual se muestra un contorno alterado que puede ser producto de algún desgaste o rotura del objeto y el las figuras (2.11), (2.12) , (2.13) y (2.14) diferentes aproximaciones utilizando la curva NURBS.

La curva NURBS permite la asignación de pesos a cada punto de control, en este caso se observa que para pesos menores a la unidad la curva se aleja de los puntos, además es necesario que en la zona de falla se tenga un número limitado de puntos

ya que cuando existen muchos puntos de control no es posible alejar la curva de dichos puntos, el cambio en el grado también es un dato a tener en cuenta ya que influye en la forma de la curva.

2.9.2. Caso 2: Contorno con la misma concavidad

Consideremos ahora el caso en el que el contorno obtenido sea producto de una deformación del objeto pero con la misma concavidad que presenta el diseño de la curva original como se observa en la figura (2.15) los puntos de control se han tomado de tal manera que se tenga el mínimo número en la zona deformada.
De acuerdo a esta disposición, los pesos de cada punto se deben elegir de tal manera que la curva se aleje lo necesario de los puntos de control para así aproximarla al contorno y de la falla, ahora ambas tienen la misma concavidad, para ello se han realizados varias pruebas como se observa en las figuras del (2.16) al (2.20).

2.9.3. Conclusiones

1. La aplicación de las curvas NURBS constituyen un buen método de aproximación al momento de diseñar contornos con desgaste o deformacionnes. La aplicación de los métodos de ajuste es del tipo local.
2. La falla y la zona del contorno involucrada deben tener la misma concavidad para garantizar un mejor resultado.
3. Si las zonas de falla y presentan la misma concavidad de la curva original podemos aplicar el método de ajuste con NURBS, se debe tomar el menor número de puntos posible dentro de la zona de falla y definir puntos de control en el inicio y fin de la misma.
4. La suavidad de la curva se consigue también seleccionando un mayor grado para el polinomio y dándole pesos más bajos a aquellos puntos que deben alejarse más de los puntos de control.

Aproximación con Curva B-Spline sin interrupción de contorno

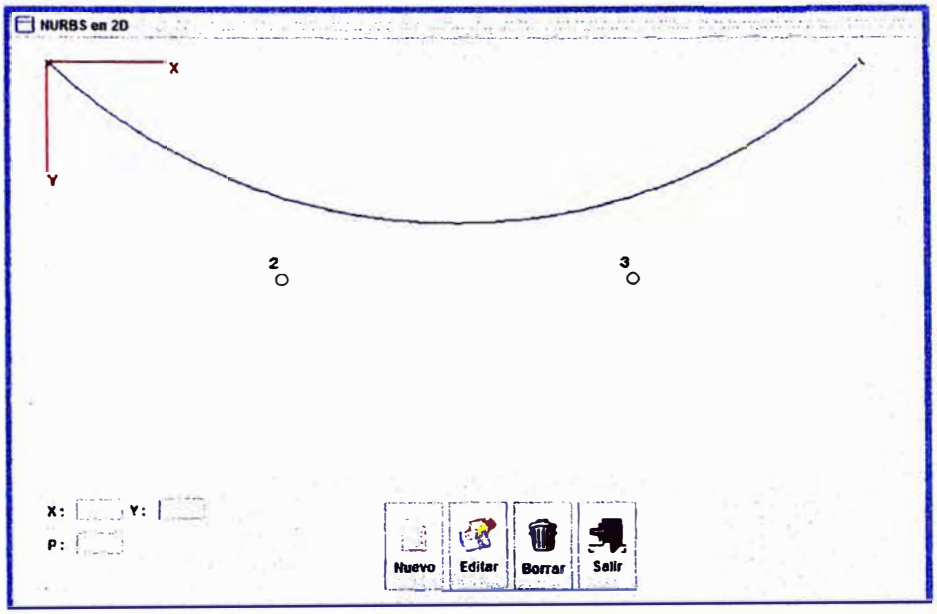

Figura 2.9: contorno de un objeto

Aproximación con Curva B-Spline con Interrupción de Contorno con Diferente Concavidad

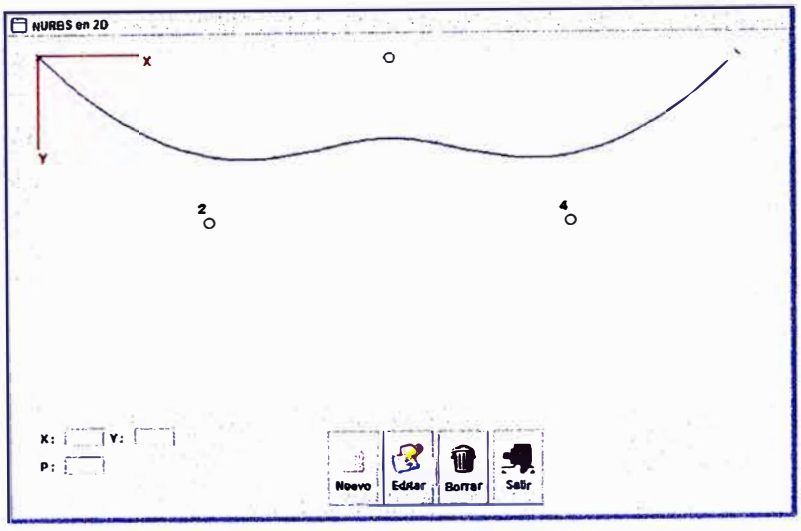

Figura 2.10: Caso 1 de contorno de un objeto

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=5)con pesos: [1 1 0.01 1 1] y la interrupción tiene diferente concavidad

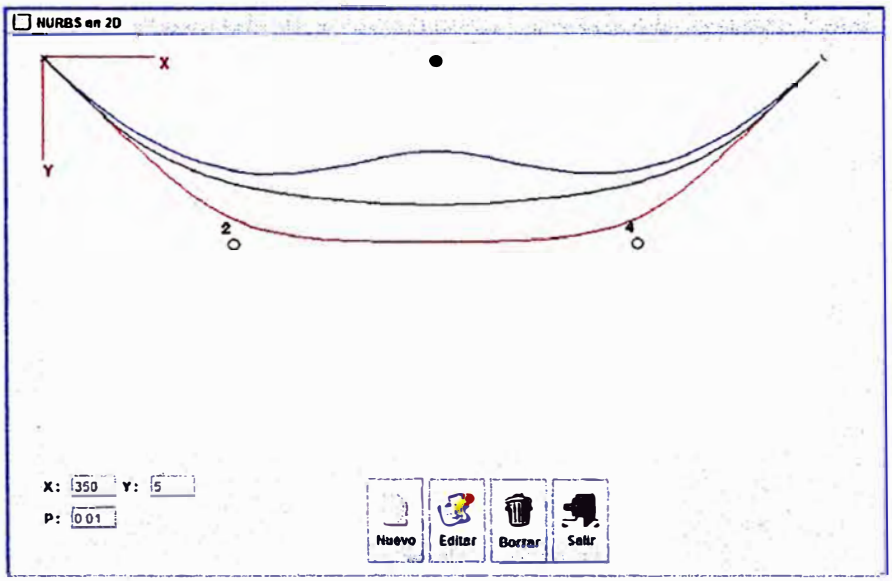

Figura 2.11: Primera Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=5) con pesos: [1 0.5 0.01 0.01 0.01 0.5 1] y diferente concavidad

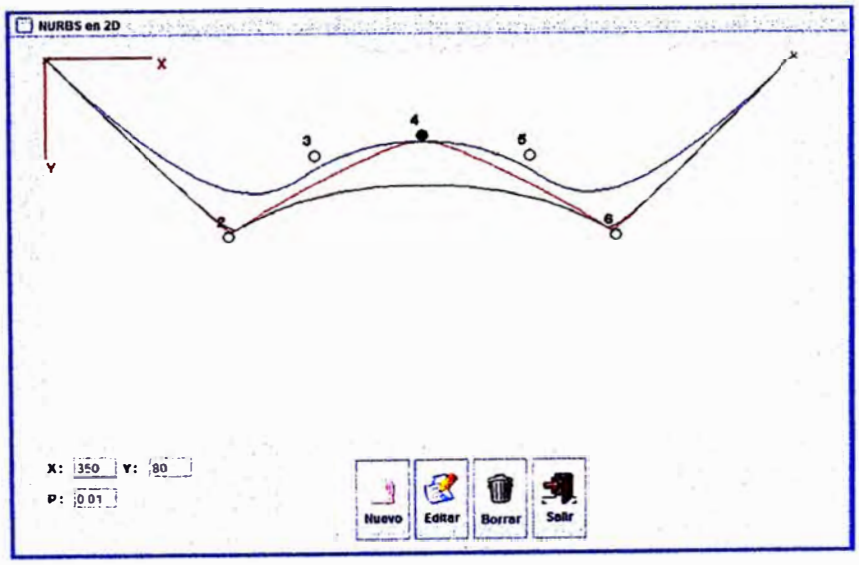

Figura 2.12: Segunda Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=7) con pesos: [1 0.5 0.01 0.01 0.01 0.5 1] y diferente concavidad

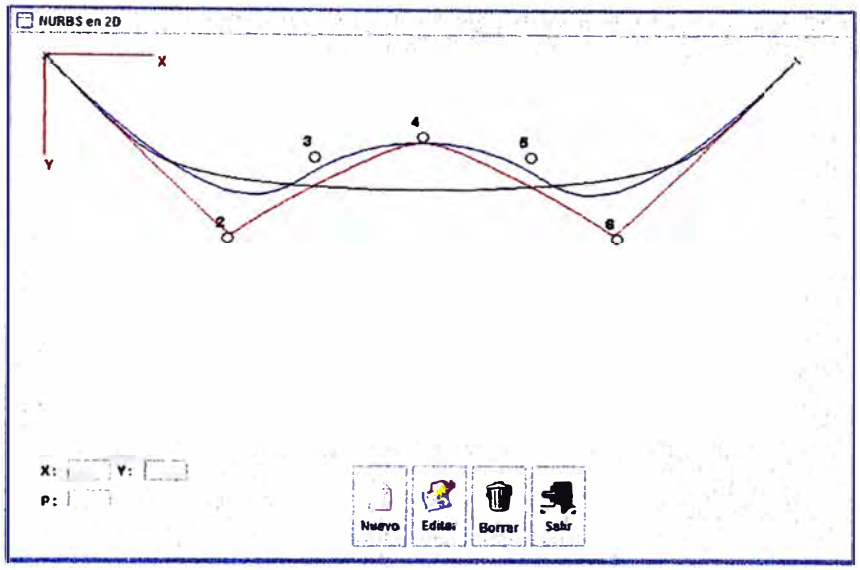

Figura 2.13: Tercera Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=5) con pesos: [1 1 2 1 1] y la interrupción tiene diferente concavidad

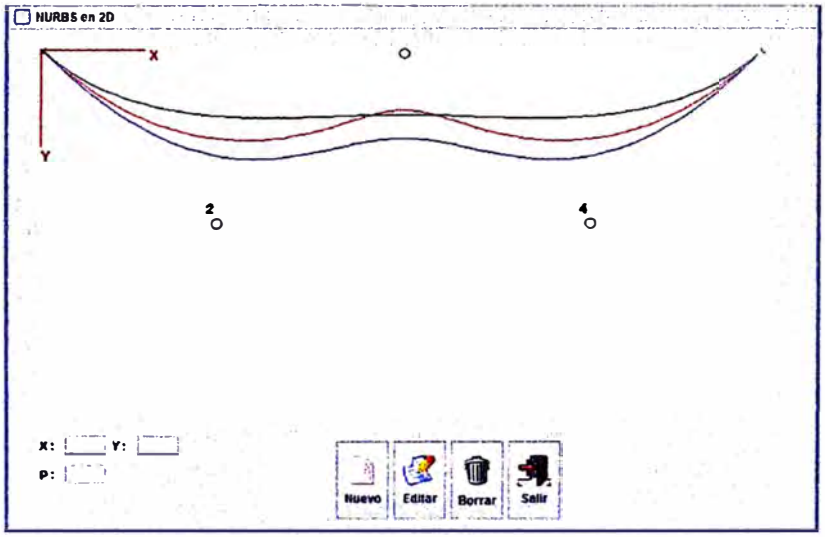

Figura 2.14: Cuarta Aproximación

Aproximación con Curva B-Spline con Interrupción de Contorno con la Misma Concavidad

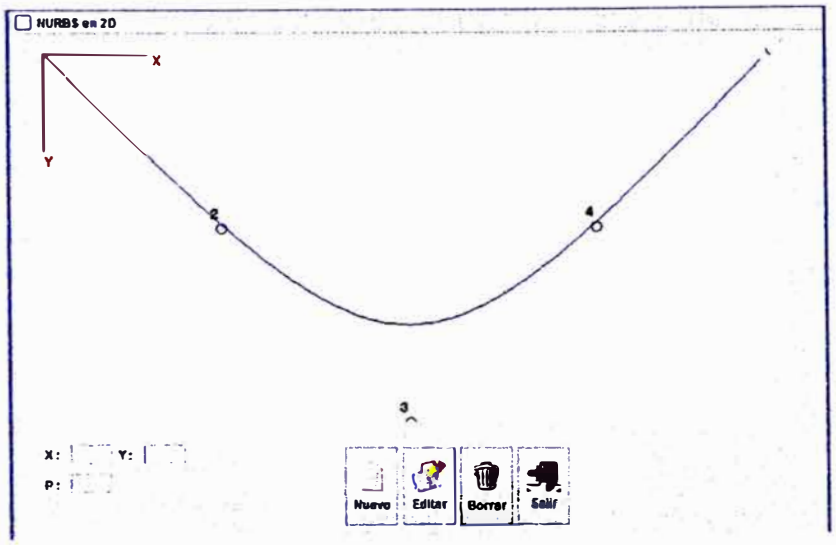

Figura 2.15: Caso 2 de contorno de un objeto

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4) con pesos: [1 1 0.2 1 1] y la interrupción tiene la misma concavidad

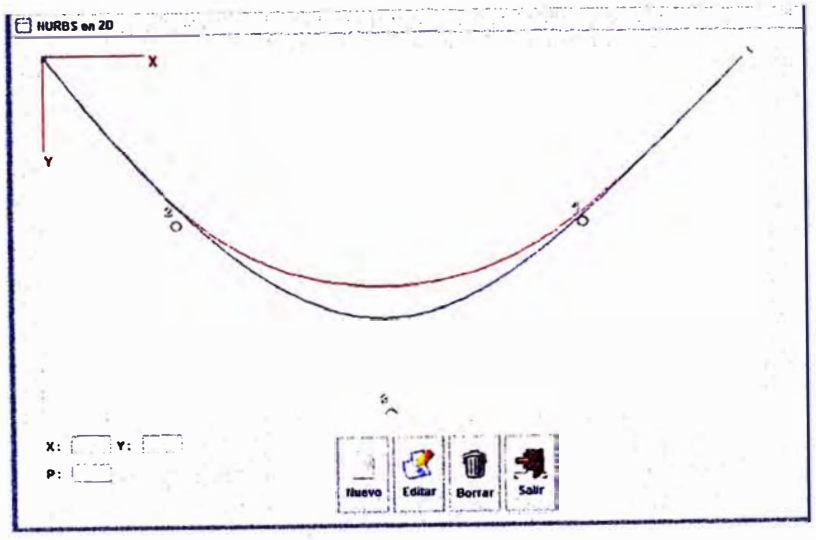

Figura 2.16: Quinta Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=5) con pesos: [1 0.2 0.2 0.2 1] y la interrupción tiene la misma concavidad

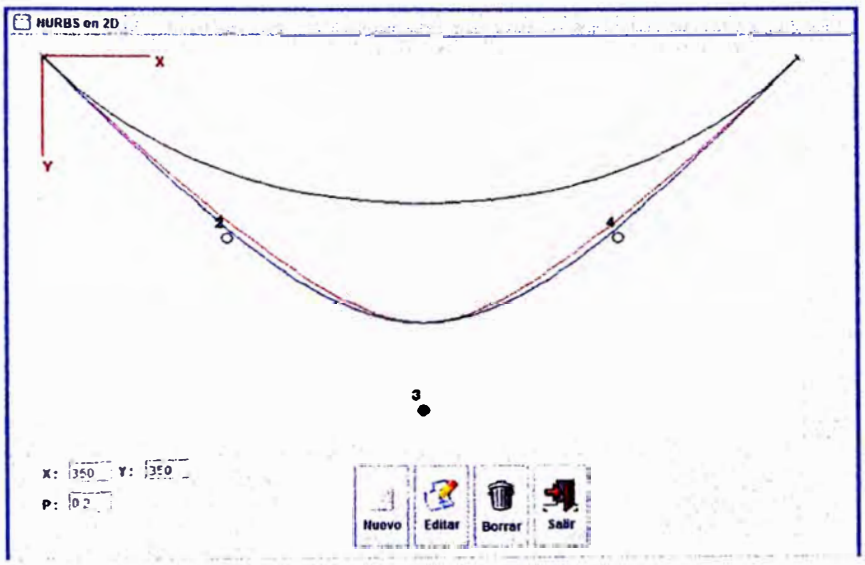

Figura 2.17: Sexta Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=5) con pesos: [1 1 2 1 1] y la interrupción tiene la misma concavidad

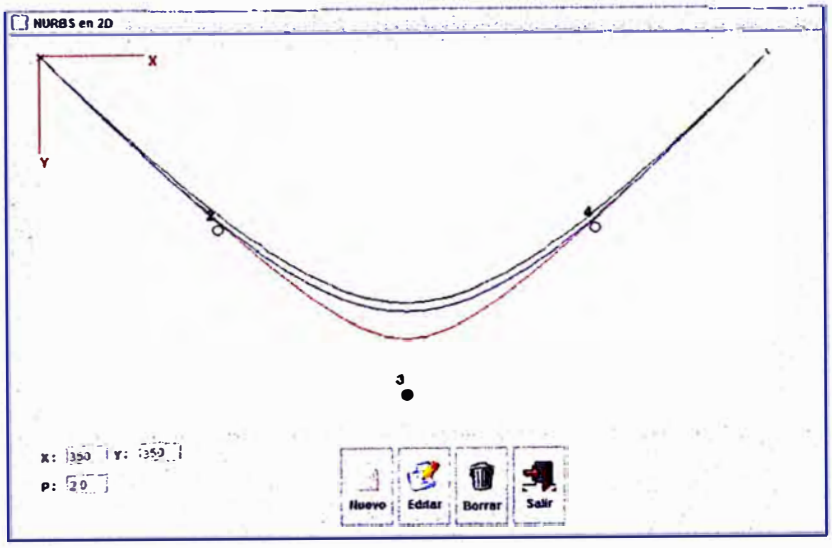

Figura 2.18: Sétima Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=5) con pesos: [1 1 1 0.2 1 1 1] y la interrupción tiene la misma concavidad

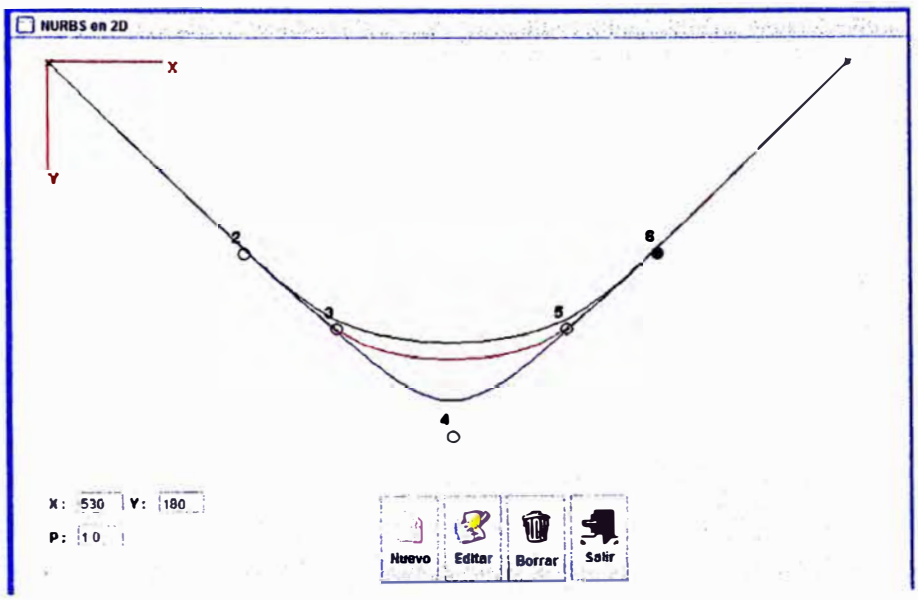

Figura 2.19: Octava Aproximación

Curva B-Spline(AZUL, m=4); NURBS(ROJO, m=4); NURBS(NEGRO, m=7) con pesos: [1 0.2 0.01 2 0.01 0.2 1] y la misma concavidad

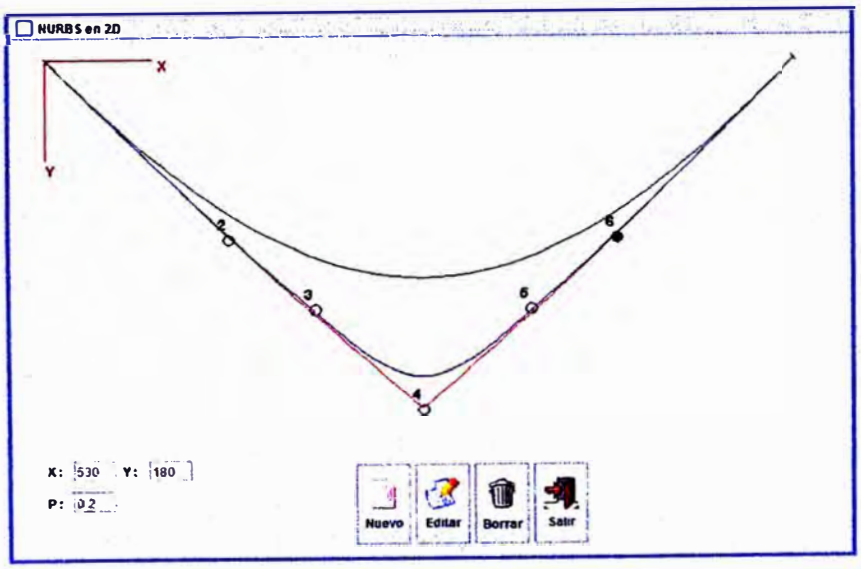

Figura 2.20: Novena Aproximación

3 CONSTRUCCIÓN DEL SISTEMA

Para construir el sistema primero debemos realizar el análisis y diseño de los algoritmos, para ello vamos a utilizar el diagrama N-S (Nassi - Schneiderman), luego para la implementación hemos elegido herramientas de software libre(Lenguaje de Programación Java y el Gestor de Base de Datos MYSQL) y para la programación la técnica de la programación orientada a objetos (POO).
El sistema debe permitir através de una interfaz amigable ingresar puntos de control y sus pesos respectivos, luego podremos elegir que tipo de curva o superficie graficar.

3.1. Algoritmos

Los principales algoritmos para la construcción del sistema son los que nos permiten graficar las curvas de Bezier, la curva B-Spline y la curva NURBS ya que para la gráfica de superficies lo que haremos es proyectar estas curvas en los ejes X, Y del espacio tridimensional y graficar curvas paralelas a la curva obtenida en el eje Z.

3.1.1. Para graficar la curva de Bezier

Dada la curva de Bezier

$$c(t) = \sum_{k=0}^{n} p_k B_k^n(t), n \in N, t \in [0,1], k \in \{0, ..., n\}$$

Donde p_k son denominados los puntos de control de la curva de Bezier y $B_k^n(t)$ son los polinomios de Bernstein.

definidos por

$$B_k^n(t) = \begin{pmatrix} n \\ k \end{pmatrix} t^k(1-t)^{n-k}, t \in [0,1], k = 0, 1, ..., n$$

donde

$$\begin{pmatrix} n \\ k \end{pmatrix} = \begin{cases} \frac{n!}{k!(n-k)!}, k \in \{0, 1, ..., n\} \\ 0, \ en\ caso\ contrario \end{cases}$$

El algoritmo permitirá para valores de t que irán de 0 a 1 con intervalos de longitud muy pequeños obtener puntos en el plano los cuales se unirán mediante segmentos de recta y formarán la gráfica de la Curva de Bezier.

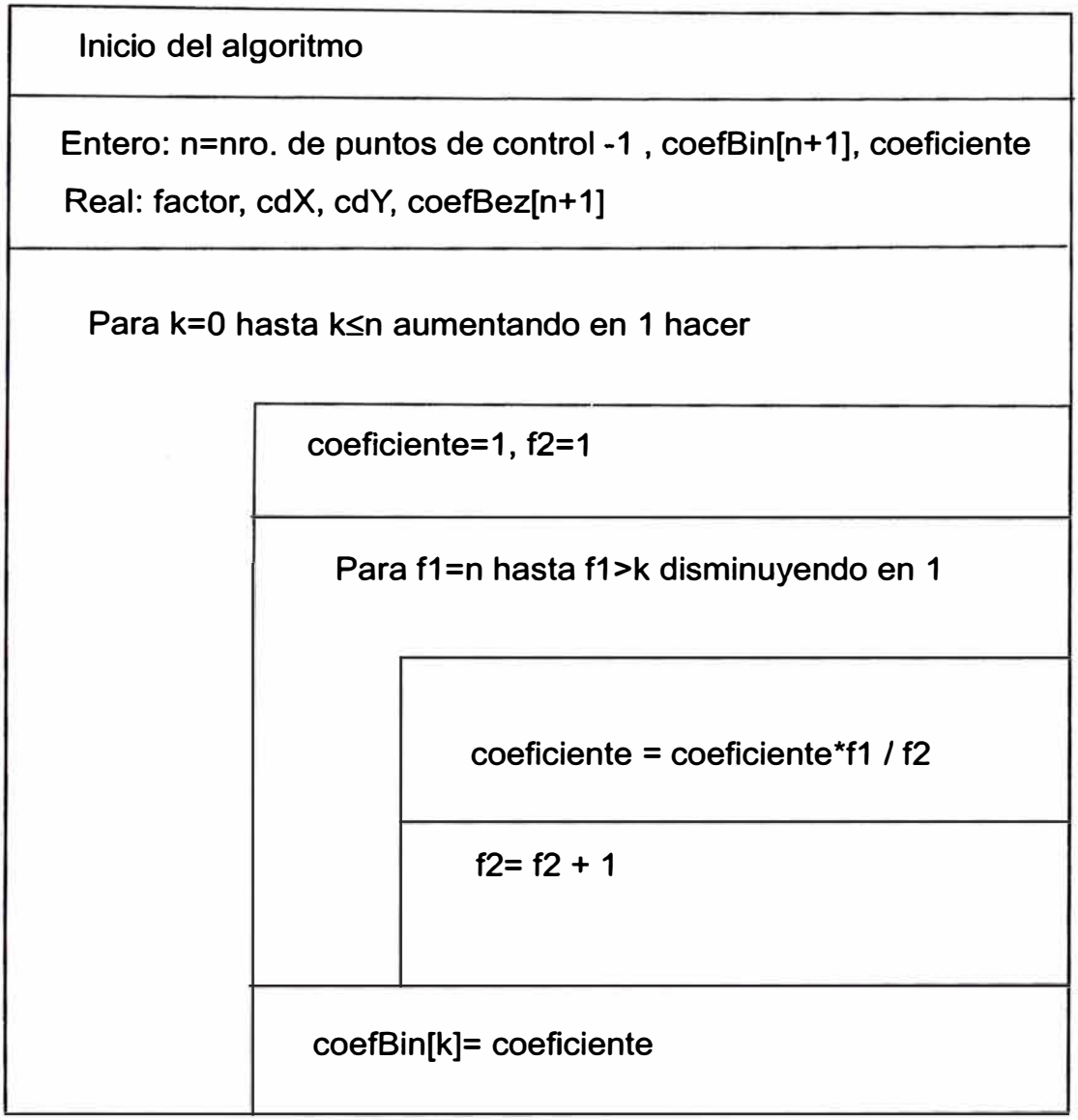

Figura 3.1: Diagrama N-S para la Curva de Bezier

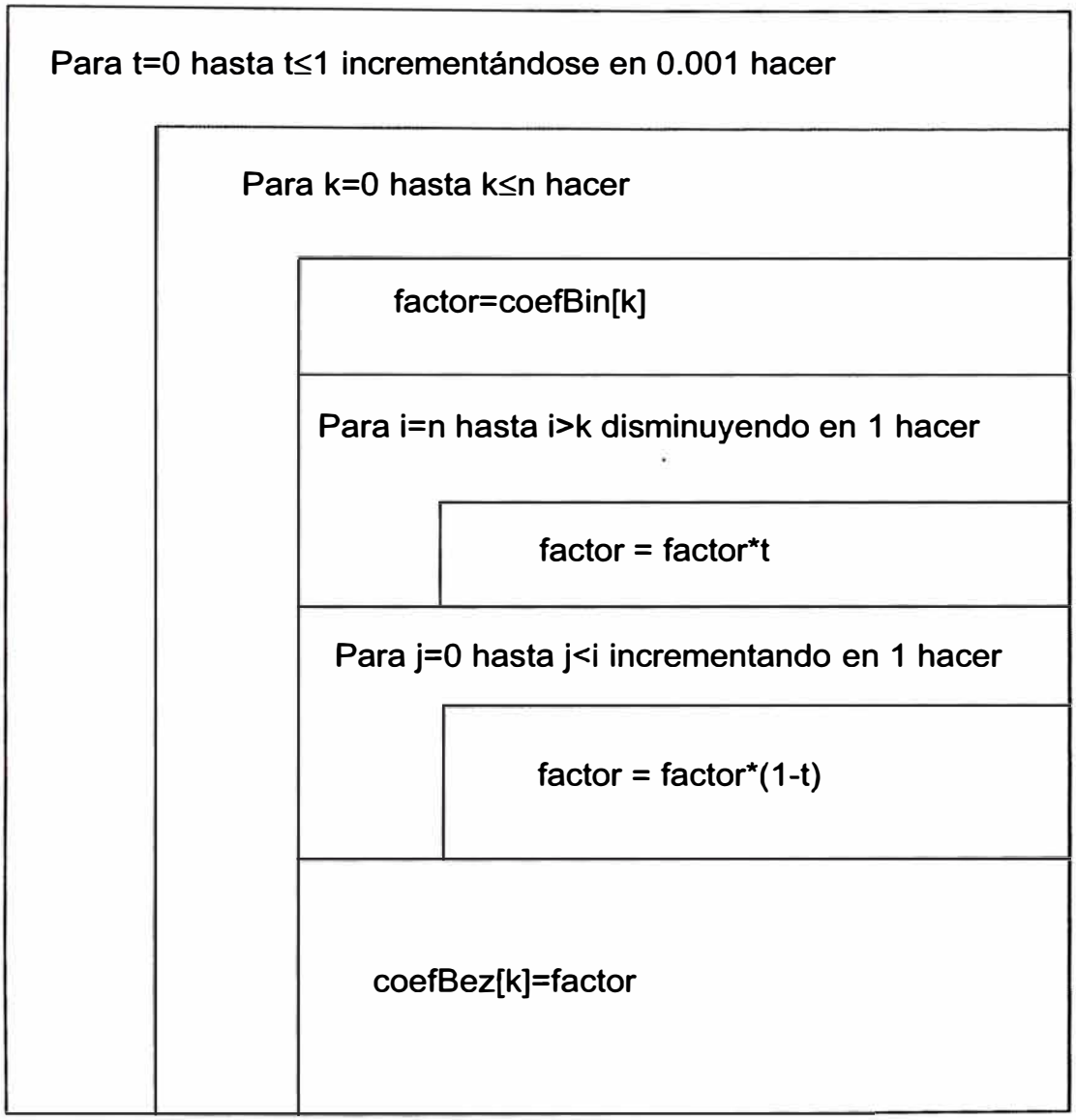

Figura 3.2: Diagrama N-S para la Curva de Bezier

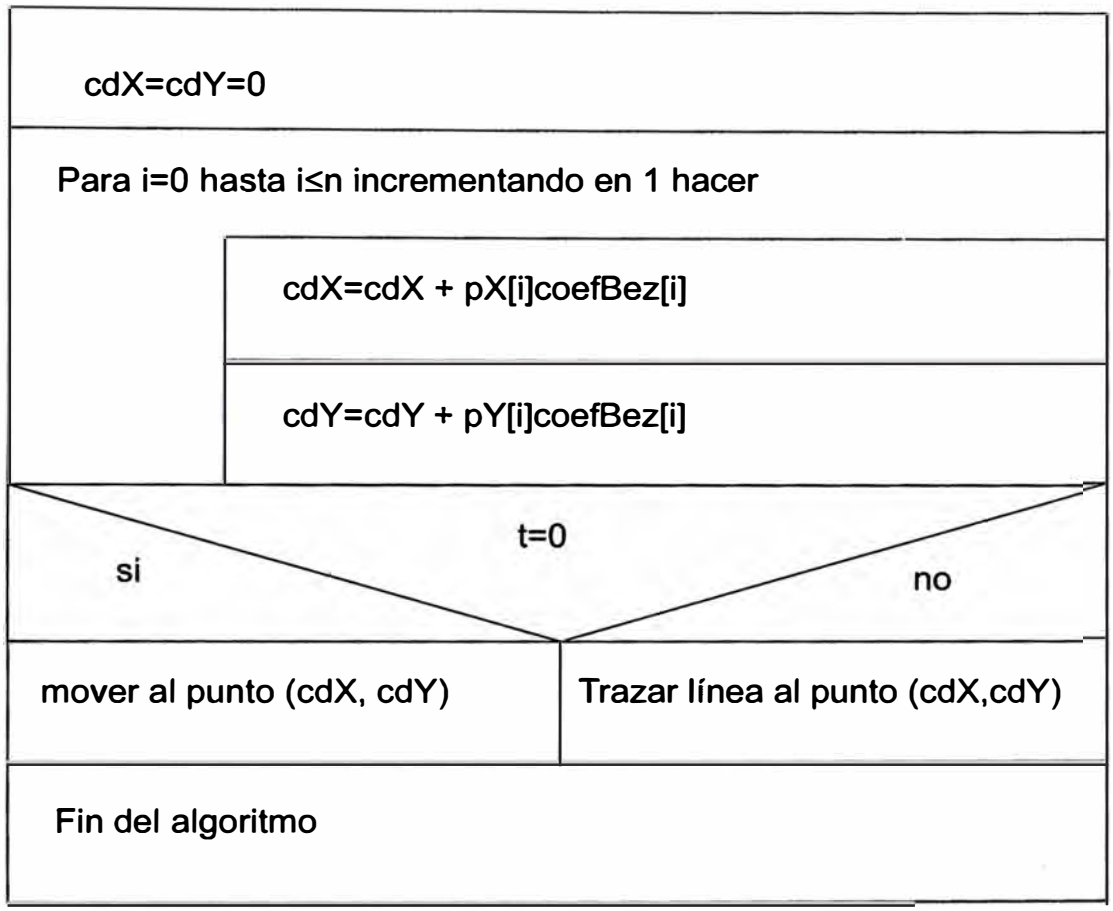

Figura 3.3: Diagrama N-S para la Curva de Bezier

3.1.2. Para graficar la curva B-Spline

Recuerde que una curva B-Spline $c(t)$, es definida por

$$c(t) = \sum_{i=0}^{n} p_i N_i^m(t), t \in [y_{m-1}, y_{n+1}[$$

donde

- $\{p_i : i = 0, ..., n\}$ son los puntos de control o puntos de Boor.
- m es el orden de los B-Splines normalizados y $m - 1$ su grado.
- $\{y_i : i = 0, ..., n + m\}$ es el conjunto de nodos con $y_i \leq y_{i+1}$
-

$$N_i^1(t) = \begin{cases} 1, t \in [y_i, y_{i+1}[\\ 0, \ en \ otro \ caso \end{cases}$$

Si $m > 1$

$$N_i^m(t) = \left(\frac{t - y_i}{y_{i+m-1} - y_i}\right) N_i^{m-1}(t) + \left(\frac{y_{i+m} - t}{y_{i+m} - y_{i+1}}\right) N_{i+1}^{m-1}(t)$$

Si algún denominador es cero entonces toda la fracción debe ser considerada como cero.

El algoritmo permite primero obtener los nodos luego para los B-Spline Normalizados utilizaremos algoritmos recursivos y finalmente con intervalos de longitud pequeña obtener los puntos en el plano que serán unidas por segmentos de recta para generar la curva B-Spline.

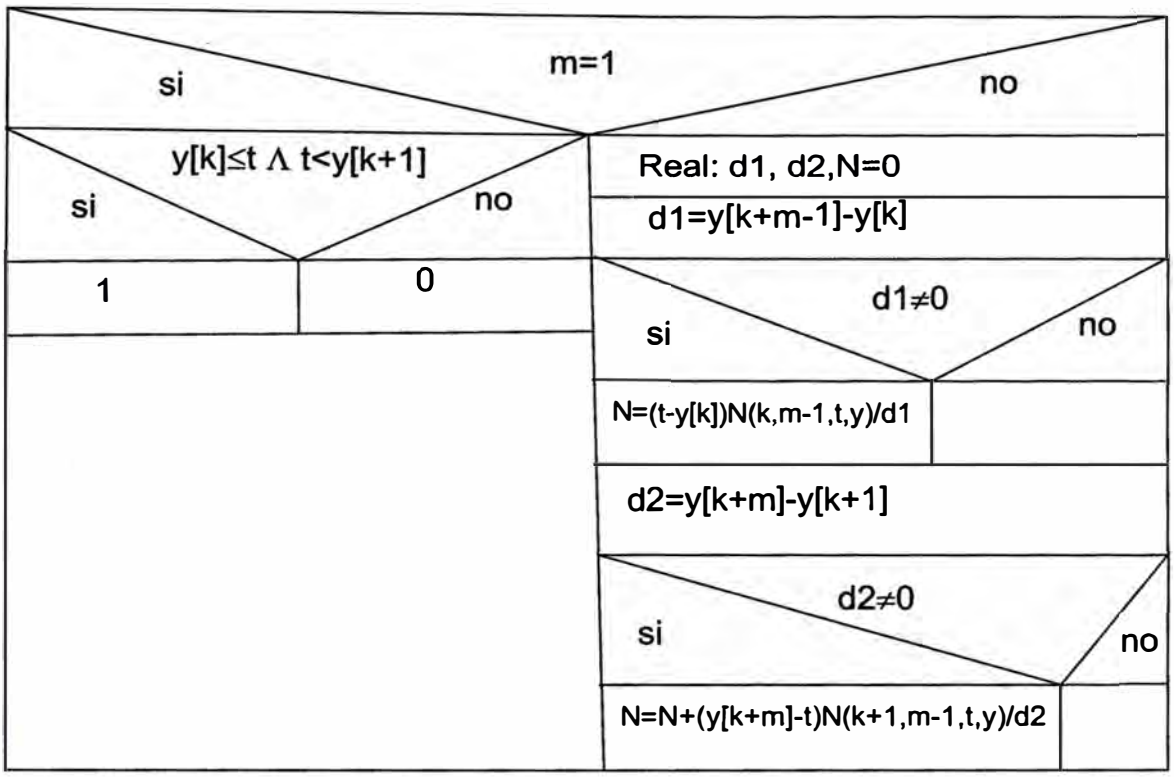

Figura 3.4: Diagrama N-S para el Método N(k,m,t,y)

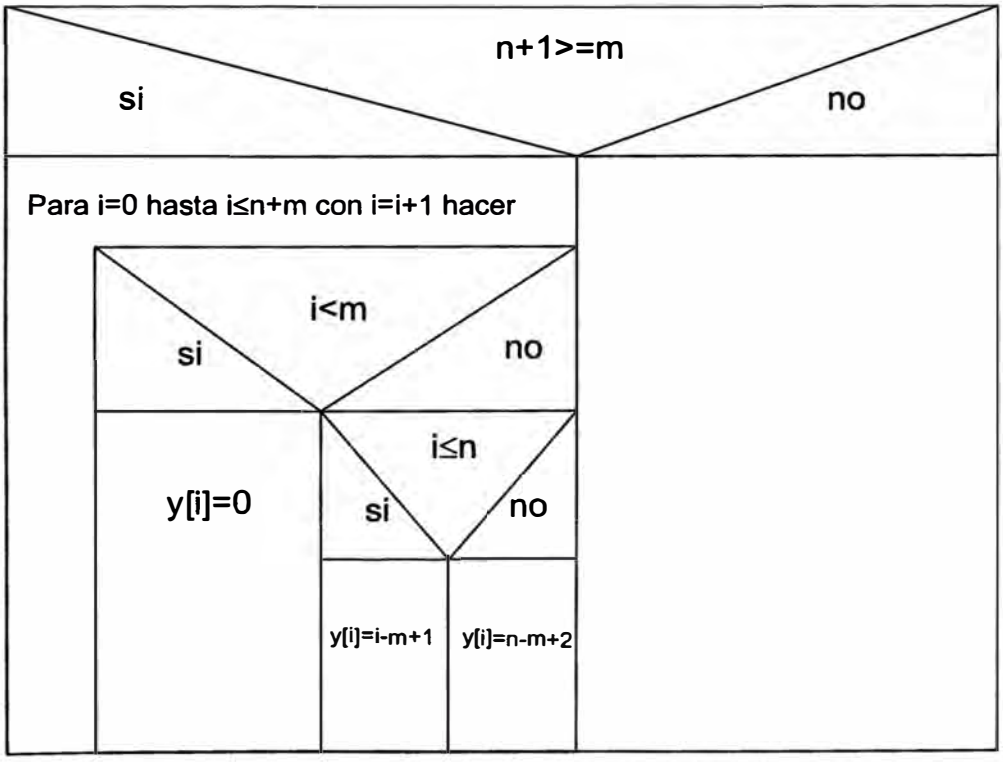

Figura 3.5: Diagrama N-S para el Método generaNodos(n,m,y)

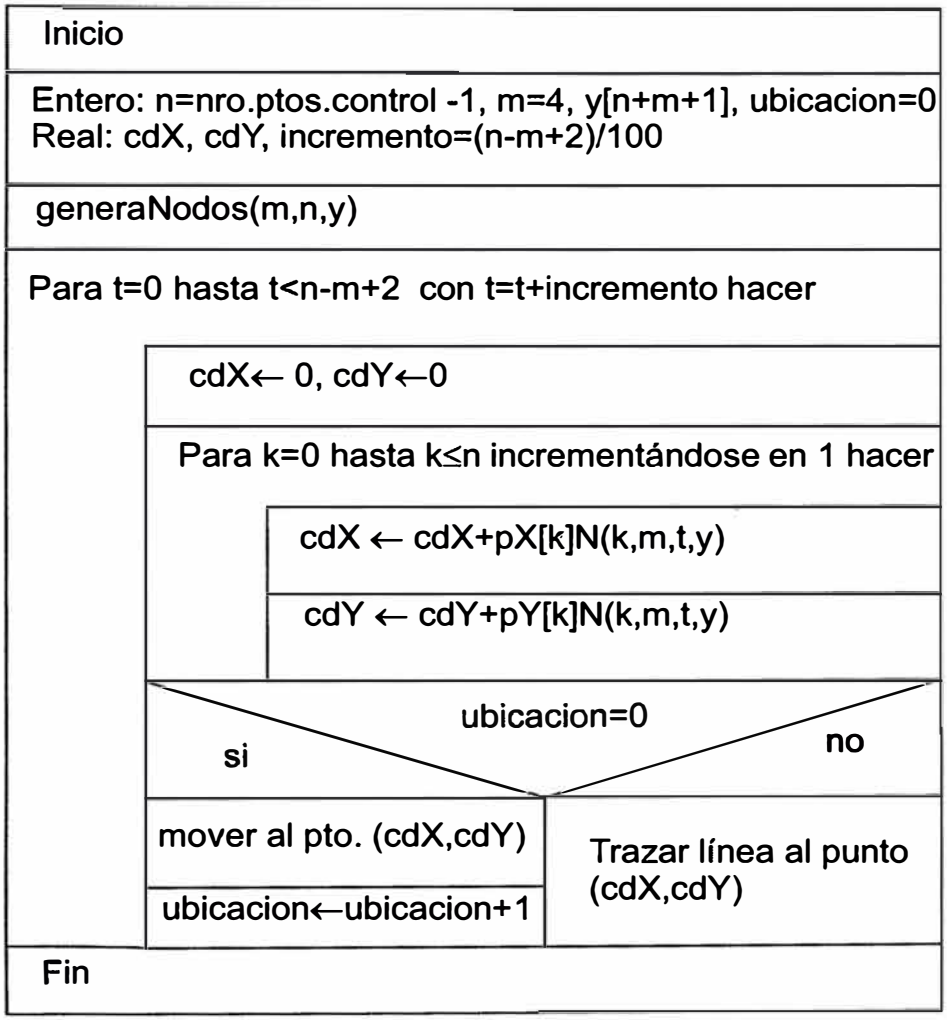

Figura 3.6: Diagrama N-S para la Curva B-Spline

3.1.3. Para graficar la curva NURBS

Recuerde que una curva NURBS $c(t)$, es definida por: Sean p_k para $k = 0, ..., n$ los puntos de control, entonces

$$c(t) = \sum_{k=0}^{n} p_k Q_k^m(t)$$

es llamado Curva NURBS, y para $k = 0, ..., n$ los

$$Q_k^m(t) = \frac{w_k N_k^m(t)}{\sum_{k=0}^{n} w_k N_k^m(t)}$$

son las funciones base de la curva NURBS, $N_k^m(t)$ son los polinomios de B-Spline Normalizados y los $w_k > 0$ son llamados pesos.

Para graficar esta curva haremos uso de los métodos creados para generar nodos y para calcular los B-Spline Normalizados, la diferencia de esta curva con respecto a la curva B-Spline es que está nos permite asignarle pesos distintos a la unidad que nos permitirá acercar o alejar la curva de los puntos de control.

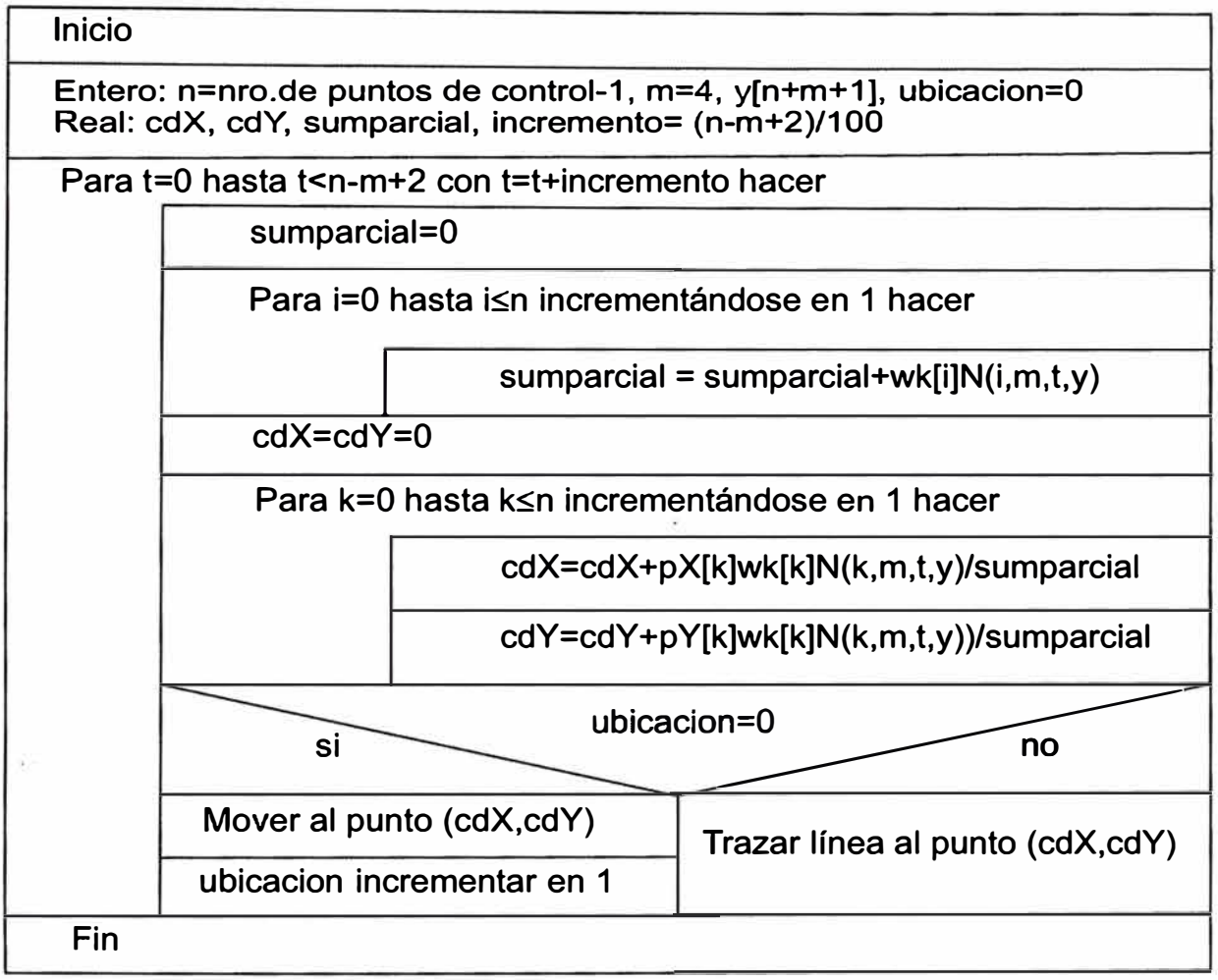

Figura 3.7: Diagrama N-S para la Curva NURBS

3.2. Herramientas de Desarrollo

Para la implementación del sistema se optó por utilizar software libre ya que nos brinda cuatro libertades:

1. La libertad de usar el programa, con cualquier propósito.
2. La libertad de estudiar cómo funciona el programa, y adaptarlo a cualquier necesidad.
3. La libertad de distribuir copias del mismo.
4. La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie.

3.2.1. Lenguaje de Programación

Se eligió el lenguaje de programación Java ya que es Open Source, fue creado en 1991 por un grupo de trabajo llamado el equipo verde, con la finalidad de comunicar varios dispositivos con diferentes unidades de procesamiento, originalmente el proyecto se llamó Oak y luego con el surgimiento de la red mundial el equipo de desarrolló visualizó que Java es un lenguaje ideal para desarrollar aplicaciones para internet

Características:

1. **Orientado a Objetos**.-Java es un lenguaje de programación orientado a objetos debido a que su objetivo principal es comunicar objetos con otros objetos para la solución de un problema.

2. **Distribuido**.-Es distribuido ya que provee soporte para el desarrollo de aplicaciones basadas en tecnologías como lo es RMI (Remote Method Invocation) y CORBA(Common Object Request Broker Architecture)
3. **Simple**.-No deja la administración de la memoria al desarrollador ya que posee un recolector de basura y está diseñado totalmente orientado a objetos, por lo que si el programador está familiarizado con el paradigma, no resulta complicado entenderlo.
4. **Multihilo**.-Java permite desarrollar aplicaciones que ejecuten varias tareas simultáneamente.
5. **Seguro**.-No permite el uso de punteros para la manipulación de la memoria, en aplicaciones distribuidas no se permite escribir y leer en la máquina de los clientes por medio de los applets.
6. **Independiente**.-Java permite que las aplicaciones puedan ejecutarse en diversas plataformas.

Plataforma Java

La plataforma para la ejecución de un programa en JAVA está formada por la Máquina Virtual Java (JVM) y las APIs de JAVA.
Las API (Application Programming Interface) son un conjunto de bibliotecas de clases estándar, que están organizadas en paquetes, formando un gran árbol de herencia.

Java Virtual Machine

La JAVA Virtual Machine es una máquina hipotética que emula por software a una máquina real en el cual el compilador genera bytecodes (instrucciones de

código de máquina para la JVM), y el intérprete ejecuta y traduce los bytecodes para cada máquina específica.

3.2.2. Gestor de Base de Datos

MySQL, es el sistema de gestión de bases de datos SQL Open Source más popular, lo desarrolla, distribuye y soporta MySQL AB. MySQL AB es una compañía comercial, se estableció originalmente en Suecia por David Axmark, Allan Larsson, y Michael "Monty"Widenius. Es una compañía Open Source de segunda generación que une los valores y metodología Open Source con un exitoso modelo de negocio.

1. **MySQL es un sistema de gestión de bases de datos**
   Una base de datos es una colección estruturada de datos. Puede ser cualquier cosa, desde una simple lista de compra a una galería de pintura o las más vastas cantidades de información en una red corporativa. Para añadir, acceder, y procesar los datos almacenados en una base de datos, necesita un sistema de gestión de base de datos como MySQL Server. Al ser los computadores muy buenos en tratar grandes cantidades de datos, los sistemas de gestión de bases de datos juegan un papel central en computación, como aplicaciones autónomas o como parte de otras aplicaciones.
2. **MySQL es un sistema de gestión de bases de datos relacionales**
   Una base de datos relacional almacena datos en tablas separadas en lugar de poner todos los datos en un gran almacén. Esto añade velocidad y flexibilidad. La parte SQL de MySQL se refiere a Structured Query Language. SQL es el lenguaje estandarizado más común para acceder a bases de datos y está definido por el estándar ANSI/ISO SQL. El estándar SQL ha evolucionado desde 1986 y existen varias versiones.

3. **MySQL software es Open Source**

   Open Source significa que es posible para cualquiera usar y modificar el software. Cualquiera puede bajar el software MySQL desde internet y usarlo sin pagar nada. Si lo desea, puede estudiar el código fuente y cambiarlo para adapatarlo a sus necesidades. El software MySQL usa la licencia GPL (General Public License), http://www.fsf.org/licenses/, para definir lo que puede y no puede hacer con el software en diferentes situaciones. Si no se encuentra cómodo con la GPL o necesita añadir código MySQL en una aplicación comercial, puede comprar una licencia comercial(http://www.mysql.com/company/legal/licensing/).

4. **El servidor de base de datos MySQL es muy rápido, fiable y fácil de usar**

   El servidor MySQL también tiene una serie de características prácticas desarrolladas en cooperación con los usuarios. MySQL Server se desarrolló originalmente para tratar grandes bases de datos mucho más rápido que las soluciones existentes y ha sido usado con éxito en entornos de producción de alto rendimiento durante varios años. MySQL Server ofrece hoy en día una gran cantidad de funciones. Su conectividad, velocidad, y seguridad hacen de MySQL Server altamente apropiado para acceder a bases de datos a través de Internet.

5. **MySQL Server trabaja en entornos cliente/servidor o incrustados**

   El software de bases de datos MySQL es un sistema cliente/servidor que consiste en un servidor SQL multi-threaded que trabaja con diferentes programas, bibliotecas cliente, herramientas administrativas y un amplio abanico de interfaces de programación para aplicaciones (APIs).

   También proporciona el MySQL Server como biblioteca incrustada multi-threaded que puede lincar en su aplicación para obtener un producto más pequeño, rápido y fácil de administrar.

3.3. Código del Sistema

Hemos definido las siguientes clases:

1. **Conexión**.- Define los métodos que permiten la conexión a la base de datos PC creada en mysql.
2. **BotonesMantenimiento**.- Se crea un panel de botones que permiten agregar, modificar y eliminar puntos de control.
3. **ProyecciónZ**.-Se define las sentencias que permite la proyección de las curvas definidas en el plano XY al plano XYZ.
4. **Bezier2D**.-Aquí se implementa el algoritmo para la curva de Bezier.
5. **Spline2D**.-Aquí se implementa el algoritmo para la curva B-Spline.
6. **NURBS2D**.-Aquí se implementa el algoritmo para la curva NURBS.
7. **Bezier3D**.-Aquí se implementa el algoritmo para la superficie de Bezier en base a la curva de Bezier.
8. **Spline3D**.-Aquí se implementa el algoritmo para la superficie B-Spline en base a la curva B-Spline.
9. **NURBS3D**.-Aquí se implementa el algoritmo para la superficie NURBS en base a la curva NURBS.
10. **Canva**.-Es un lienzo en el cual se graficarán las curvas y superficies en base a los puntos de control ingresados previamente.
11. **Ventana**.-Muestra una interfaz que permite observar una instancia de la clase Canva y otra de la clase BotonesMantenimiento.
12. **Inicio**.- Muestra una interfaz que permitirá ingresar el usuario y password, está clase valida los datos ingresados.
13. **Principal**.- Elabora un menú de opciones que permite elegir a que ventana desea acceder, para ello hace uso de la clase Ventana y de la clase Conexión.

El sistema solicita un usuario y un password que lo verificará con el que se encuentra registrado en la base de datos PC creada en el el motor de base de datos MySql.

Figura 3.8: Ventana de Seguridad

Si el usuario y password son correctos ingresará al menú principal donde podremos elegir menús que nos permitirán ingresar los puntos de control,observar las curvas y superficie que estos generan.

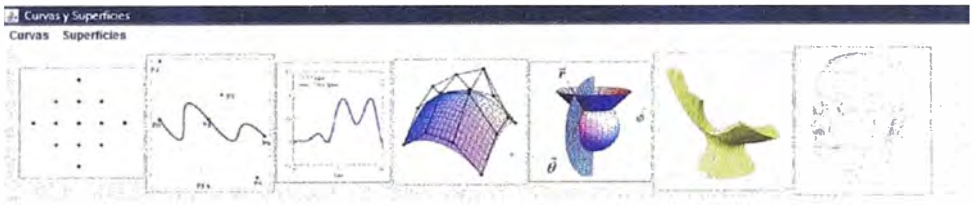

Figura 3.9: Menú Principal

En esta ventana podemos mediante la elección de los botones ingresar, modificar o eliminar un punto de control que se almacenarán en la tabla punto 2d de la base de datos PC creado en MySql,

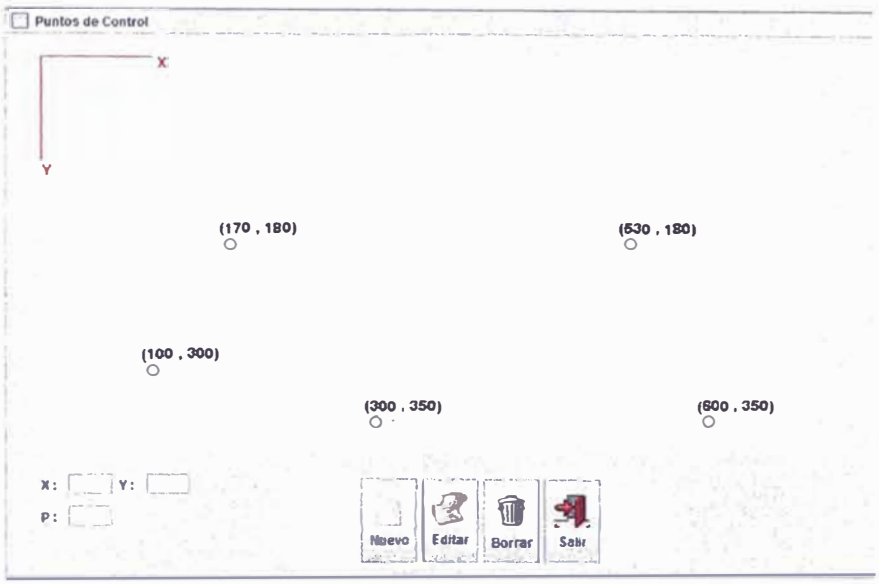

Figura 3.10: Puntos de Control

Base de datos PC creada con dos tablas: usuario para saber los usuarios autorizados para el uso del sistema y puntos2d en MySql en el cual se guardará los puntos de control.

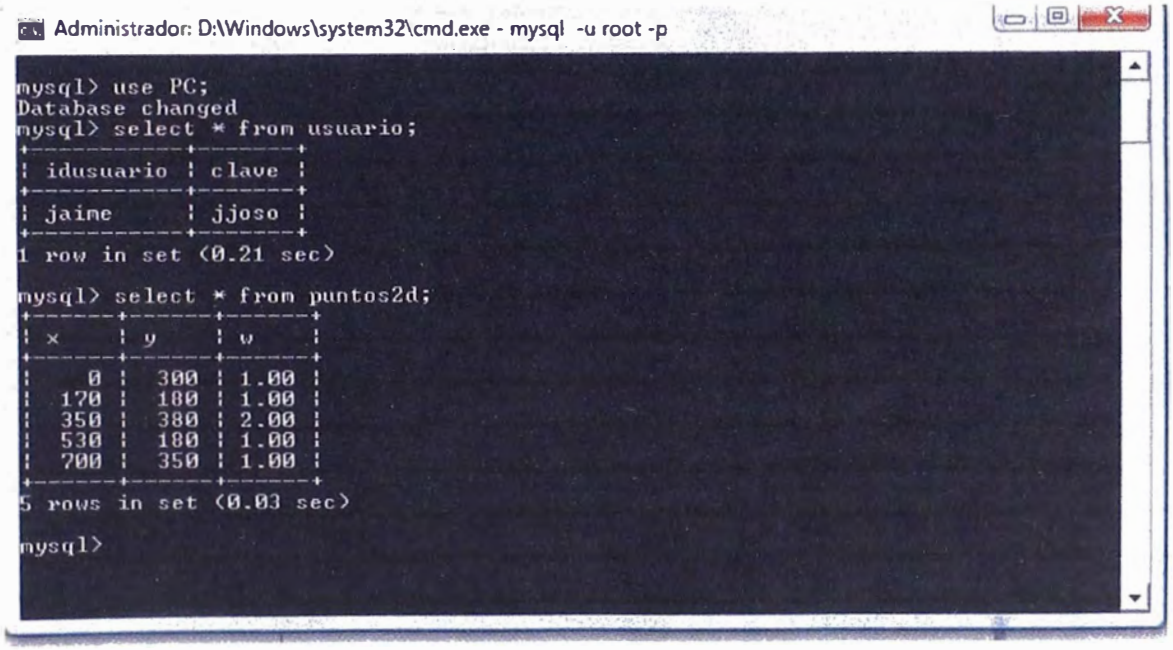

Figura 3.11: Base de datos PC

La curva de Bezier generado por los puntos de control ingresados anteriormente es

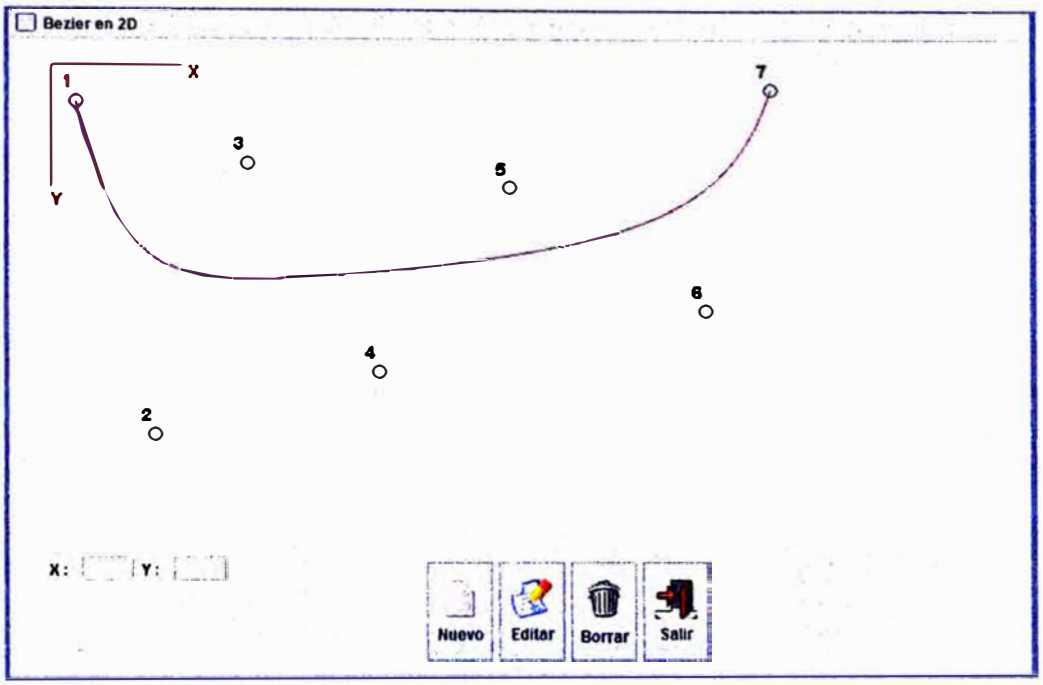

Figura 3.12: Curva de Bezier

La curva B-Spline para los mismos puntos de control

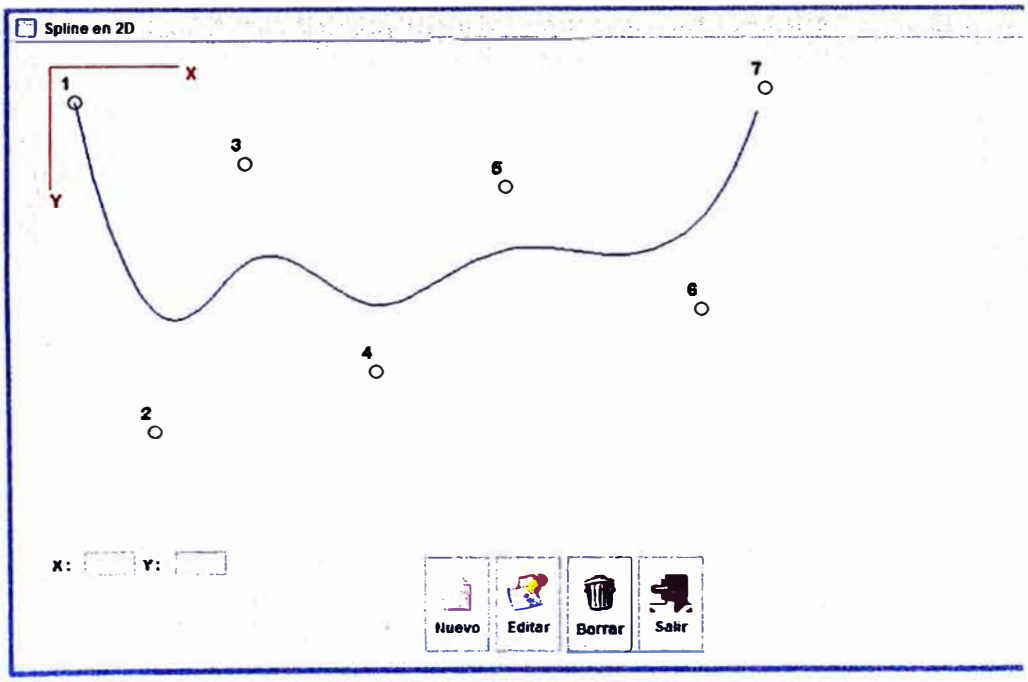

Figura 3.13: Curva B-Spline

Asignado pesos a los puntos de control, B-Spline(azul)-NURBS(rojo)

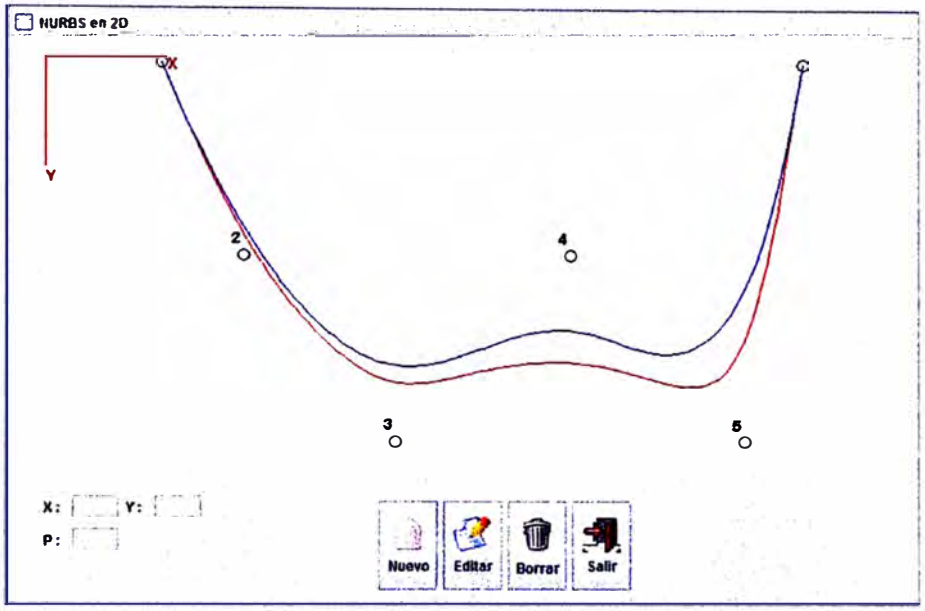

Figura 3.14: Curva NURBS

Apartir de la curva de Bezier se genera la superficie de Bezier mediante la proyección de los puntos x e y y haciendo que z se incremente mediante una constante.

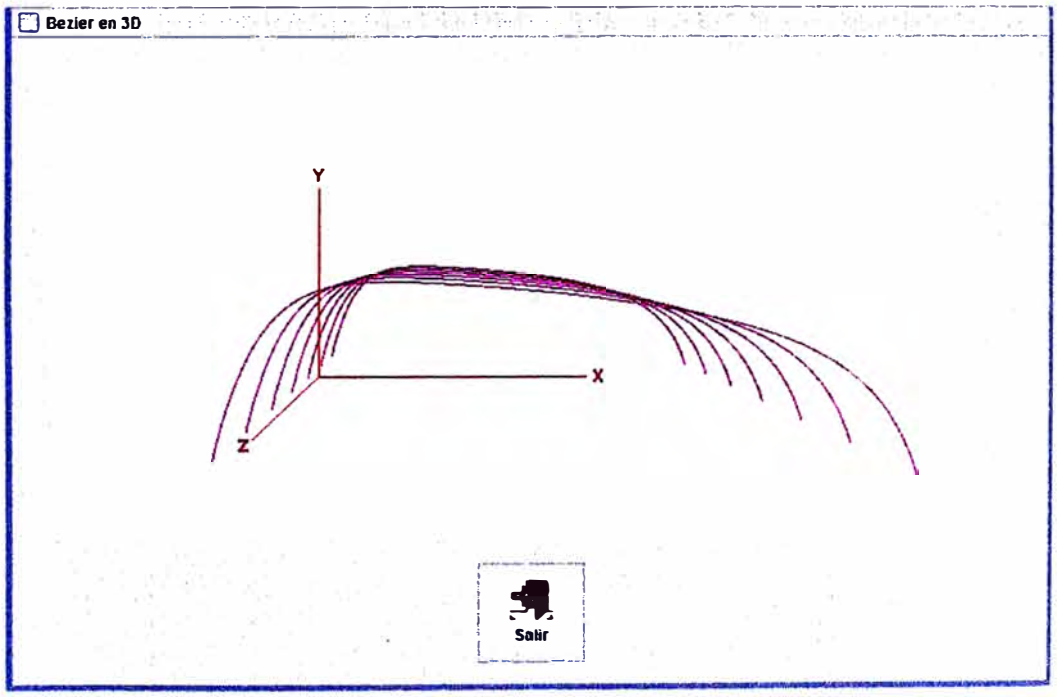

Figura 3.15: Superficie de Bezier

Con los mismos puntos generados x,y,z se obtiene la superficie B-Spline

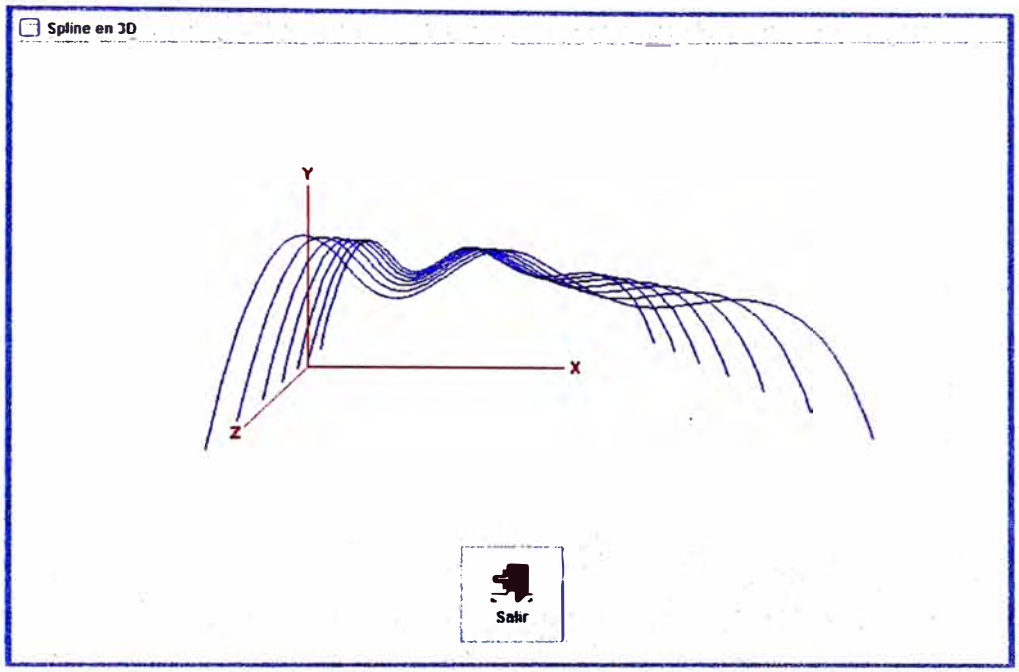

Figura 3.16: Superficie B-Spline

Asignado pesos a los puntos de control y proyectando la curva NURBS(rojo),B-Spline(azul)

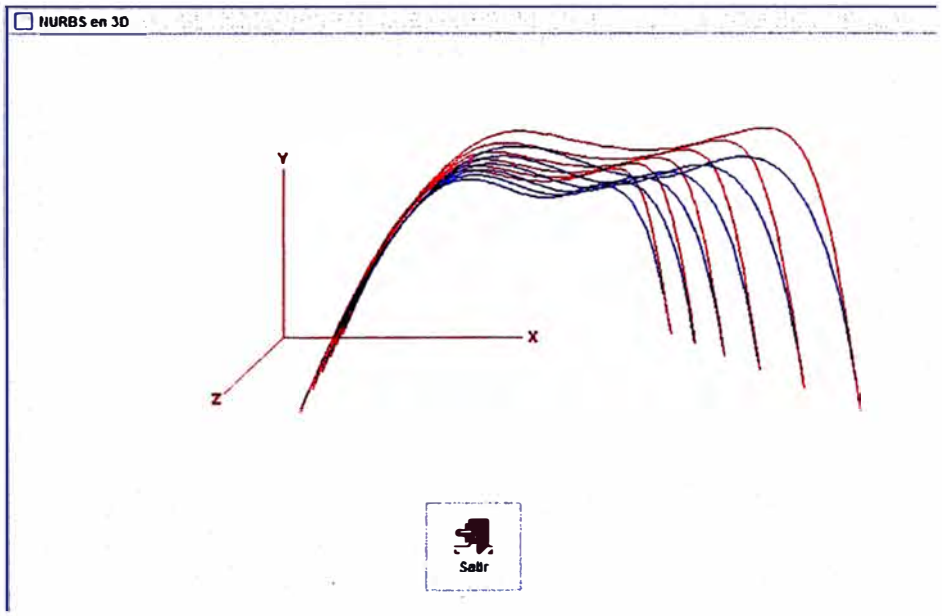

Figura 3.17: Superficie NURBS

4 CONCLUSIONES

Para las curvas y superficies de Bezier los puntos de control son el epicentro de su generación ya que constituyen el punto de partida, modelan la curva y superficie, definen el grado del polinomio y permiten la deformación de la misma de forma automática.

Pero las curvas y superficies NURBS se constituyen como la técnica de aproximación más utilizada ya que el grado de los polinomios que la forman se puede establecer de manera independiente de la cantidad de puntos de control y permiten un mayor control local sobre la forma de una curva o superficie, en caso de deformaciones o desgaste nos permite aproximarnos mediante la asignación de pesos por cada punto de control.

Bibliografía

[1] Paluszny-Prautzsch-Boehm, Métodos de Bezier y B-Splines, Universitätsverlag Karlsruhe, 2005

[2] Deitel,Harvey M., Cómo programar en Java, México D.F. Pearson Education, 2004

[3] Horstmann, Cay S., Core Java 2 Volumen 1, Sun Microsystems Press, 2003

[4] Horstmann, Cay S., Core Java 2 Volumen 2, Sun Microsystems Press, 2003

[5] Kenneth I. Joy, Visualization Graphics Research Group, Departament of Computer Science, University of California, 2000

[6] Larry L. Schumaker, Spline Functions Basic Theory, Krieger Publishing Company, Malabar, Florida, 1993

[7] Juan Manuel Cordero Valle, José Cortés Parejo, Curvas y Superficies para Modelado Geométrico, Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, 2002

[8] Peter Lancaster and Kestutis Salkauskas, Curve and Surface Fitting, Departament of Mathematics and Statistics, University of Calgary, 1990

[9] Gerald Farin, Curves and Surfaces for Computer Aided Geometric Design, Departament of Computer Science, Arizona State University, 1990

[10] Charles K. Chui, Multivariate Splines, Texas A&M University, 1988