BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

MODELACIÓN DE SISTEMAS COMPLEJOS USANDO B-SPLINE

TESIS PRESENTADA PARA OBTENER EL TÍTULO DE:

LICENCIATURA EN MATEMÁTICAS APLICADAS

PRESENTA: MARIA DE LOURDES MORALES SÁNCHEZ

ASESOR: DR. W. FERMIN GUERRERO SÁNCHEZ

COASESOR: DR. CARLOS IGNACIO ROBLEDO SÁNCHEZ

PUEBLA, PUE

JUNIO 2014

MODELACIÓN DE SISTEMAS DINÁMICOS COMPLEJOS USANDO B-SPLINE

MARIA DE LOURDES MORALES SANCHEZ

23 de julio de 2014

RESUMEN

En este trabajo de tesis se pretende modelar y simular sistemas físicos reales con la mayor perfección posible. En el mundo real en el que vivimos y el que hemos creado artificialmente lo que menos impera son las líneas rectas, los objetos que la naturaleza ha formado al igual que los sistemas artificiales que el humano ha creado a semejanza, la belleza de los objetos en 2D y 3D está dado por las curvas y superficies. El problema con el cual nos enfrentaremos es que no existe un modelo matemático del objeto a estudiar, la solución que se propone en este trabajo de tesis se basara en curvas simples que se obtienen a partir de curvas parametrizadas los algoritmos que se construirán están inspirados en el trabajo de Bézier, la formulación que se usara será los B-Spline similar a las funciones Bézier, aquí en lugar de tener polinomios de Bernstein, tendremos funciones de base donde el grado (k) de estas funciones definen la precisión y suavidad de la curva. En este trabajo no se tocara a un tipo de curvas llamadas Racionales o NURBS[1] (Non Uniform Rational B-Spline) que son el caso más general ya que los B-Spline y las curvas de Bézier son casos particulares de los NURBS, las aplicaciones de los B-Spline son muy amplias va desde el diseño de vehículos, aviones, etc... hasta el tratamiento de datos en el procesamiento de imágenes. El enfoque que tendrá esta tesis está planeado como un problema inverso el cual consiste en la obtención de un modelo matemático a partir de la información que nos presenta el objeto real, esta información se puede obtener de diferentes formas como puede ser con un perfilometro, un escáner etc... Los algoritmos basados en B-Spline se construirán en Mathematica y se pretende que se apliquen en proyectos para diseñar aviones y barcos a escala, mismos que posteriormente se convertirán en UAV´s[2] y USV´s [3]

[1]B-Spline racionales no uniformes (acrónimo inglés de non-uniform rational B-spline)

[2]Vehículo aéreo no tripulado (acrónimo inglés de unmanned aerial vehicle)

[3]Vehículos de superficie no tripulado (acrónimo inglés de unmanned surface vehicles)

Índice general

Capítulo 1

INTRODUCCIÓN:

La mayor parte de los objetos existentes en el mundo real presentan formas continuas y suaves, y para nada pueden asemejarse a formas poligonales. La mayoría de las aplicaciones graficas necesitan modelar objetos presentes en el mundo real, y por lo tanto se hace necesario poder generar curvas y superficies de una forma más exacta que una simple sucesión de segmentos rectos cortos como es el caso de los fractales. La necesidad de poder representar curvas y superficies viene por dos motivos. En primer lugar para modelar objetos construidos por la naturaleza o por el humano (montañas, arboles, ríos etc.), en donde una descripción matemática del objeto puede no estar disponible. Por supuesto, podemos usar para modelar las coordenadas de los infinitos puntos del objeto, lo cual no es nada viable para esto se usa un escáner o un perfilometro, etc. También podemos representarlo aproximadamente usando planos, esferas u otras primitivas sencillas, fáciles de describir matemáticamente, pero seguiríamos sin lograr una representación adecuada.

En segundo lugar para modelar algo que aún no exista físicamente, por ejemplo cuando se está diseñando un nuevo prototipo de coche o de avión. Para crearlo, el usuario puede ir esculpiéndolo interactivamente, o bien especificar su descripción matemática, o dar una descripción aproximada para luego ir retocándola.

La importancia de la transición suave de un punto a otro, es sin duda un importante problema en el campo de la matemática aplicada, así podemos encontrar que dependiendo del tipo de problema, existen soluciones diferentes; por ejemplo para un conjunto de puntos P(x, y) puede ser de relevante importancia el que se haga pasar una curva por todos ellos utilizando en este caso un método de interpolación, uno de los más conocidos es la interpolación polinomial de Lagrange. Otro problema es encontrar la curva sin que forzosamente se pase por todos los puntos, sino que ésta describa el comportamiento general de los datos, tal es el caso del ajuste de curvas usando el método de Mínimos Cuadrados. Pero cuando el problema es trazar una curva suave desde un punto inicial hasta un punto final, y que ésta sea afectada en su trayectoria por un conjunto de puntos

que describen un polígono de apoyo, entonces el problema puede resolverse con las Curvas de Bézier.

Pierre Bézier, francés, ingeniero de profesión, en 1960 resolvió el problema numérico en el trazado de curvas y superficies interpolantes, que parten y llegan a un punto dado, y su trayectoria es afectada por un conjunto de puntos de apoyo. Su utilización desde entonces ha sido en la fabricación de vehículos y en la aeronáutica, áreas en la que él ha incursionado exitosamente, son sin duda algunas de las más valiosas aplicaciones que se hacen de esta herramienta.

Entonces al definir las curvas de Bézier, decimos que éstas se apoyan en un conjunto de puntos ordenados llamados puntos de control; los cuales afectan la trayectoria de la curva. La curva pasa obligatoriamente solo por el primer y el último punto. Al polígono que se forma al unir secuencialmente los puntos de control se le llama polígono de apoyo o de control. La curva de Bézier que se obtiene como resultado, está en la cubierta convexa del polígono de apoyo.

En esta tesis trabajaremos con elementos conocidos como Splines que son curvas polinómicas por trozos continuamente diferenciables hasta un orden prescrito. El ejemplo más sencillo es el spline C^0, o sea, lineal por trozos. Este spline es simplemente una poligonal en el plano o en el espacio. Otro ejemplo son los splines cúbicos C^2.

El nombre "spline" es una palabra en idioma ingles que significa "listón elástico". Estos listones eran usados por artesanos para crear curvas, que describen superficies a construir, como cascos de barcos y fuselajes de aviones. Constreñidos por pesos, estos listones elásticos o splines asumen una forma que minimiza su energía elástica, propiedad que heredan en forma aproximada los splines matemáticos C^2 de grado tres.

La herramienta de los splines se desarrolla para solventar las limitaciones de las curvas de Bézier: falta de control local, la laboriosidad requerida para imponer continuidad C^2 y el hecho de que el número de puntos de control de una curva de Bézier impone su grado. Los elementos que se usaran en esta tesis son conocidos como B-Spline, la B de B-Spline se refiere a "base": en este modelo se utiliza una base, en general polinómica, a diferencia de los intentos originales de simular una varilla flexionada mediante "puntos de control", usando las ecuaciones diferenciales de la elasticidad. Las B-Spline ganaron la batalla sobre todos los otros tipos definidos (¡muchos!), las curvas de Bézier y los B-Spline se consideran como un caso particular de los NURBS. Los B-Spline se trata de curvas no−interpolantes, en general cúbicas, unidas con continuidad C^2 y cuyos puntos de control poseen control local. Pueden pensarse como un método de unión de curvas de Bézier de grado n con continuidad C^{n-1}, pero ahorrándonos muchos puntos de control innecesarios como los que se introdujeron al interpolar con curvas de Bézier.

Normalmente, las B-Spline y las NURBS se definen mediante las blending functions (como los polinomios de Bernstein para las curvas de Bézier) usando el algoritmo recursivo de Cox−De Boor para calcularlas. De ese modo, se dan una serie de métodos algebraicos, sin contenido geométrico.

Existen muchas formas de abordar el tema. En esta tesis las estudiaremos mediante la forma polar y blossoming, lo que implica un cambio de notación que parece bastante rebuscado, pero finalmente hace mucho más fácil "entender" el tema. Para programar, en cambio, cualquier receta es adecuada. Pero uno de nuestros objetivos en esta tesis es construir superficies. Los algoritmos que se diseñaran y se programaran en esta tesis se basaran en un método conocido como superficies obtenidas usando producto tensorial, esta superficie se obtienen al mover cada uno de los puntos de control de una curva del tipo:

$$P(t) = \sum_i b_i N_i(t)$$

A lo largo de otra curva de expresión similar. La superficie viene controlada por una malla rectangular.

En 1966, Bézier desarrolla la construcción de superficies a partir de polinomios de Bernstein. Se llaman superficies de Bézier de producto tensorial. Las Superficies de B-Spline de producto tensorial, incluyen a las superficies de Bézier de producto tensorial, y a las superficies a trozos de Bézier como casos particulares. Dependen de dos parámetros independientes, u y v.

Ventajas:

- Los grados en u y en v se escogen a priori
- Tienen carácter local: cada vértice de control va asociado a una única función de base, que tiene soporte local.

La superficie se obtiene al mover una curva B-Spline de grado p en la dirección u a lo largo de una curva B-spline de grado q en la dirección v:

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

donde

$$u \in [0,1]$$

$$v \in [0,1]$$

Las nuevas funciones de base son:

$$N_i^p(u)N_i^q(v) \leftarrow \begin{cases} u \in [u_i, u_{i+1}] \\ v \in [v_j, v_{j+1}] \end{cases}$$

En esta tesis existen varias razones para representar un objeto mediante un modelo de superficie:

- Cuando el objeto mismo es una superficie que podemos suponer sin grosor. Este tipo de representación nos permite visualizar superficies abiertas, mientras que los sólidos se caracterizarán por tener su superficie necesariamente cerrada sobre sí misma.
- Cuando tan sólo nos interesa visualizar su aspecto visual externo, sin detalles sobre su estructura interna, aunque el objeto ocupe un cierto volumen.
- Cuando deseamos realizar una visualización en tiempo real, y para ello utilizamos hardware o software gráfico que está sólo preparado para visualizar polígonos.

En cualquiera de estos casos es conveniente utilizar una representación de la superficie del objeto. En principio la información sobre una superficie debe dar cuenta de su geometría, de sus propiedades visuales (cómo se comporta frente a la luz) y quizás también de alguna propiedad física (como la elasticidad) si se va a efectuar una simulación física sobre el objeto.

Si la totalidad del objeto consta de diferentes partes (por ejemplo, varios polígonos o varios trozos definidos por diferentes ecuaciones), entonces podemos añadir también información topológica, es decir, sobre cómo estas diferentes partes se conectan entre sí para formar la superficie. Con esta información adicional el modelo se conoce como una representación de frontera del objeto. Esta representación de frontera se utiliza frecuentemente en combinación con un modelo sólido, ya que una de ellas se puede reconstruir a partir de la otra.

Una característica que suele exigirse a las superficies representadas en esta tesis es que sean variedades bidimensionales, es decir, que no existan puntos singulares donde la superficie se intersecte consigo misma o se abra en varias hojas.

Para representar un objeto complejo se puede utilizar una descomposición de la superficie en trozos o parches (patches) triangulares o cuadrangulares, cada uno de los cuales se suele modelar con superficies paramétricas (superficies de Bézier, B-Spline, NURBS...). Esta representación permite modificar la forma de la superficie con mucha facilidad: en lugar de cambiar las coordenadas de un gran número de vértices, solo necesitamos cambiar unos cuantos parámetros en las ecuaciones. Una vez terminado el modelado si queremos hacer una visualización en tiempo real siempre resultará posible convertir estas superficies en una aproximación poligonal.

El problema que plantea la representación continúa a trozos son, por una parte, la dificultad de controlar que la forma de la superficie sea exactamente la que queremos y, por otro, la complejidad de mantener las restricciones (continuidad, diferenciabilidad) en las intersecciones de los trozos.

Para definir una superficie curvada y suave de forma exacta debemos recurrir a una representación analítica, es decir, mediante ecuaciones. Dependiendo de la forma de las ecuaciones habrá que evaluar la superficie (hallar sus puntos) utilizando distintos métodos. En general tenemos los siguientes tipos:

- Ecuación implícita: p.ej. $f(x, y, z) = 0$
- Ecuación explícita: p.ej. $y = f(x, z)$
- Ecuación paramétrica : p.ej. una superficie $(x, y, z) = f(u, v)$

En tres dimensiones una ecuación explícita siempre puede transformarse en paramétrica.

Superficies Paramétricas: Superficies de Control.

En esta tesis se van a examinar algunas de las formas de proceder con una superficie paramétrica, que mostrarán lo simple que resulta su utilización en ciertos algoritmos y su visualización. Si se emplean las ecuaciones paramétricas para definir una superficie, es posible hallar puntos que pertenezcan a ella (por ejemplo, para poder dibujarla) sin más que ir dando valores a los parámetros u y v, de tal forma que se recorre la superficie de forma exhaustiva. En muchas representaciones se normaliza la ecuación de manera que los parámetros u y v solamente tomen valores en el rango [0,1].

Además se pueden calcular muy fácilmente algunos parámetros interesante, tales como:

- Los vectores tangentes para los parámetros u,v y el vector normal a la superficie.
- Calcular la distancia recorrida en un camino a lo largo de la superficie.
- Calcular la curvatura (relacionada con las segundas derivadas) en un punto cualquiera de la superficie.
- Calcular los parámetros u ,v de un punto desconocido (por ejemplo, el punto donde una recta intersecta a la superficie) mediante aproximaciones sucesivas dando valores a u y v.El problema de esta aproximación es que la complejidad de las ecuaciones suele aumentar enormemente cuando imponemos muchas restricciones (por ejemplo, que la superficie pase por más y más puntos que nos interesan). También aparecen efectos indeseados; las superficies resultantes pueden tener una forma extraña aunque les forcemos a pasar por ciertos puntos de control.

Una curva Splines es una secuencia de segmentos de curva que se conectan entre sí para formar una sola curva continua. Por ejemplo, una colección de trozos de curvas Bézier, extremo conectado a extremo, puede ser llamada una curva Splines. Spline se define como un polinomio de grado n a trozos cuyos segmentos son de clase C^{n-1}.

El modelo computacional basado en el método B-Spline se adapta a superficies complejas, ya que trabaja con una base de datos, la cual la captura y crea una manta suave. Por este motivo es que tiene muchas aplicaciones y en diversos campos, como en la ingeniería, arquitectura, en la industria de construcción naval, etc.

Capítulo 2

PRELIMINARES MATEMÁTICOS

2.1. ESPACIOS AFINES

Definición. Sea E un espacio no vacío. Se dice que E es un espacio afín asociado a un espacio vectorial V , si existe una aplicación: $E \times E \to V$

Verificando las propiedades:

1. $\forall\, p \in E$ y $\forall\, \bar{v} \in V$ existe un único $q \in E$ verificando que $\bar{v} = \overline{pq}$ es decir, existe un único vector que es igual a la distancia entre p y q.
2. $\forall\, p, q, r \in E$ se verifica que $\overline{pq} + \overline{qr} = \overline{pr}$ (relación de Chasles).

A los elementos de un espacio afín los llamaremos puntos.

COMBINACIONES AFINES

Los $m+1$ puntos $p_0, \ldots, p_m$ de un espacio afín E se llaman afínmente independientes si los m vectores $(\overline{p_1p_0}) \ldots, (\overline{p_mp_0})$ son linealmente independientes.

Sea n la dimensión de E, $p_0, \ldots, p_n$ son $n+1$ puntos afinmente independientes entonces cada q en E se puede expresar como:

$$q = p_0 + (\overline{p_1p_0})x_1 + \cdots + (\overline{p_np_0})x_n = p_0x_0 + \cdots + p_nx_n \tag{2.1}$$

Donde $1 = x_0 + \cdots + x_n$. Los coeficientesx_i son las coordenadas baricéntricas de q con respecto al marco $p_0 \ldots p_n$.

La secuencia ordenada $p_0, \ldots, p_n$ se denomina marco.

Nótese que $x_0, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n$ son las coordenadas afines de q con respecto al origen p_j y los n vectores $(\overline{p_i p_j}), i \neq j$.

APLICACIONES AFINES

Sean A y B espacios afines, U y V sus espacios vectoriales subyacentes, de dimensión m y n, respectivamente **[4]**. Una función $\Phi : A \to B$ se denomina aplicación afín si se puede representar, con respecto a alguno de nuestros sistemas de coordenadas, a través de una matriz A, de dimensión $n \times m$ tal que

$$y = \Phi(x) = a + \mathrm{A}x \tag{2.2}$$

Donde a es la imagen del origen.

La aplicación lineal $\varphi : U \to V$ dada por

$$v = \varphi(u) = Au \tag{2.3}$$

Se denomina la aplicación lineal subyacente de Φ. Usando coordenadas extendidas, ambas aplicaciones tienen la misma representación matricial

$$\begin{bmatrix} y \\ 1 \end{bmatrix} = \begin{bmatrix} A & a \\ o^t & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix}, \quad \begin{bmatrix} v \\ 0 \end{bmatrix} = \begin{bmatrix} A & a \\ o^t & 1 \end{bmatrix} \begin{bmatrix} u \\ 0 \end{bmatrix}$$

Lo cual se puede rescribir de manera más compacta como:

$$y = Ax, \qquad v = Au$$

Las siguientes dos propiedades son consecuencia de la representación matricial:

Una aplicación afín Φ conmuta con las combinaciones afines, es decir

$$\Phi(\sum a_i \alpha_i) = \sum \Phi(a_i)\alpha_i$$

Además, una aplicación afín está completamente determinada por un marco de dimensión $dimA + 1$ $p_0 \ldots p_m$ y su marco imagen $q_0 \ldots q_m$.

2.2. CURVAS Y SUPERFICIES PARAMÉTRICAS

Sea $x_1, ..., x_n$ funciones reales continuas definidas en el intervalo $[a, b]$,el vector $x \in \mathbb{R}^n$ define una curva parametrica para el parámetro $t \in [a, b]$.

$$x(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix} \tag{2.4}$$

Usualmente pensamos en $x(t)$ como una curva de puntos. En particular si las funciones coordenadas $x_i(t)$ son de grado menor o igual que n entonces $x(t)$ es una curva polinómica de grado n en t **[2]**.

La grafica de una función $x(t)$ es una curva polinómica de la siguiente forma:

$$X(t) = \begin{bmatrix} t \\ x(t) \end{bmatrix}$$

Ejemplos:

$$x(t) = \begin{bmatrix} cos(t) \\ sen(t) \end{bmatrix}$$

$$t \in [0, 2\pi]$$

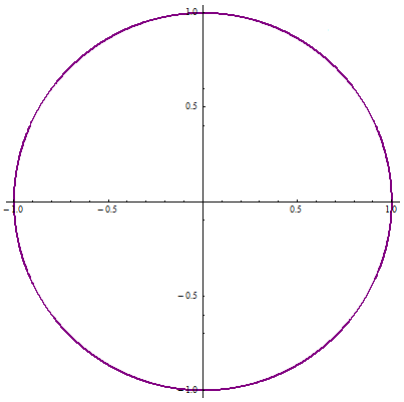


Figura 2.1: curva parametrizada (circunferencia)

$$x(t) = \begin{bmatrix} tcos(t) \\ tsen(t) \end{bmatrix}$$

$$t \in [0, 6\pi]$$

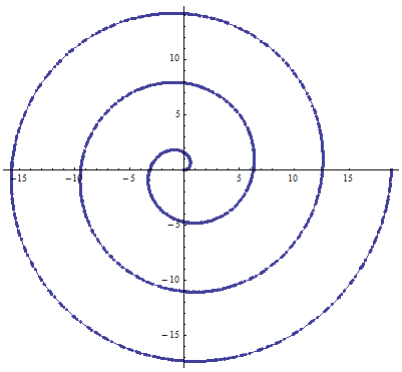


Figura 2.2: La espiral como una función paramétrica.

En el siguiente ejemplo como se puede observar el grosor de la curva va aumentando, eso nos lleva a deducir que es una superficie, además de que está en función de dos variables.

$$x(u,v) = \begin{bmatrix} (v+1)ucos(u) \\ (v+1)usen(u) \end{bmatrix}$$

$$u \in [0, 6\pi]$$

$$v \in [0.5, 1]$$

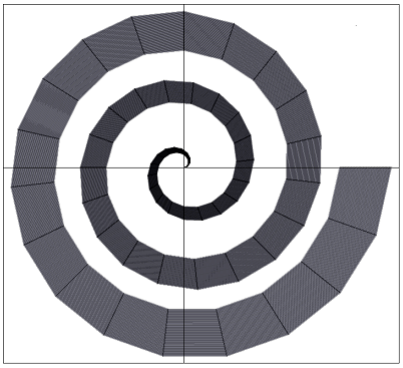


Figura 2.3: Superficie como una función paramétrica

Análogamente una superficie paramétrica se puede describir como una función x que depende de dos parámetros s y t.

$$x(s,t) = \begin{bmatrix} x_1(s,t) \\ \vdots \\ x_n(s,t) \end{bmatrix} \tag{2.5}$$

$$s \in [a,b]$$

$$t \in [c,d]$$

Observación, una superficie podría ser degenerada a una curva o a un punto.

La superficie se denomina polinómica de grado total n si los x_i son polinomios de grado menor o igual que n en s y t, por lo menos uno de los x_i tiene grado n. La grafica de una función polinómica $x(s,t)$ es una superficie dada de la siguiente forma:

$$x(s,t) = \begin{bmatrix} s \\ t \\ x(s,t) \end{bmatrix}$$

Ejemplos:

$$x(s,t) = \begin{bmatrix} Cos[s]Sen[t] \\ Sen[t]Sen[t] \\ Cos[t] \end{bmatrix}$$

$$s \in [-\pi, \pi]$$

$$t \in [0, \pi]$$

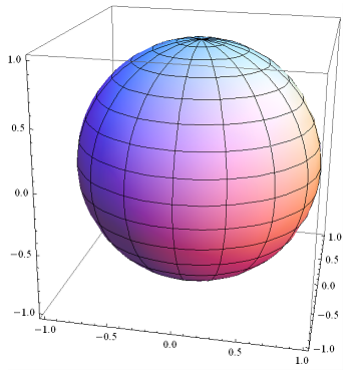


Figura 2.4: Esfera

$$x(s,t) = \begin{bmatrix} (2+Cos[t])Cos[t] \\ (2+Cos[t])Sen[t] \\ Sen[t] \end{bmatrix}$$

$$s \in [0, 2\pi]$$

$$t \in [0, 2\pi]$$

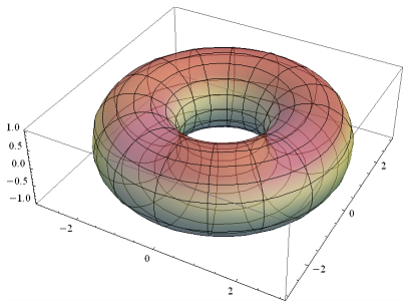


Figura 2.5: Toro o dona

2.3. POLINOMIOS DE BERNSTEIN

Se obtienen a partir del cálculo de la expansión binomial

$$1 = (u + (1-u))^n = \sum_{i=0}^{n} \binom{n}{i} u^i (1-u)^{n-i} \tag{2.6}$$

Nos permite introducir los polinomios de Bernstein de grado n que se definen como:

$$B_i^n(u) := \left(\begin{array}{c} n \\ i \end{array} \right) u^i (1-u)^{n-i}$$
$$i = 0, \ldots, 1$$

Las siguientes propiedades de los polinomios de Bernstein de grado n son importantes para nuestros propósitos:

- Son linealmente independientes.

De hecho, dividiendo

$$\sum_{i=0}^{n} p_i u^i (1-u)^{n-1} = 0 \tag{2.7}$$

por $(1-u)^n$ y usando$s = u/(1-u)$

Obtenemos

$\sum_{i=0}^{n} p_i s^i = 0$ lo cual implica que $p_0 = p_1 = ... = p_n = 0$.

- Son simétricos,

$$B_i^n(u) = B_{n-i}^n(1-u) \tag{2.8}$$

Las únicas raíces son 0 y 1,

$$B_i^n(0) = B_{n-i}^n(1) = \left\{ \begin{array}{ll} 1 & i = 0 \\ 0 & i > 0 \end{array} \right. \tag{2.9}$$

- Forman una partición de la unidad

$$\sum_{i=0}^{n} B_i^n(u) = 1 \quad \forall\, u \in R$$

- Satisfacen la relación de recurrencia

$$B_i^{n+1}(u) = u B_{i-1}^n(u) + (1-u) B_i^n(u) \tag{2.10}$$

Donde $B_{-1}^n(u) \equiv B_{n+1}^n(u) \equiv 0$ y $B_0^0(u) \equiv 1$

2.4. ALGORITMO DE CASTELJAU

El algoritmo de Casteljau consiste en la aplicación reiterada de la interpolación al polígono de control de la curva de Bézier.

$$p_i^1(t) = (1-t)p_i + t p_{i+1}\ ,\ i = 0, \ldots, n-1 \tag{2.11}$$

De modo que en cada paso disminuye el número de vértices en una unidad;

$$p_i^r(t) = (1-t)p_i^{r-1}(t) + tp_{i+1}^{r-1}(t)\ ,\ i = 0, \ldots, n-r, r = 1, \ldots n$$

nota: el superíndice$r$ corresponde al paso o nivel del poligono de control.

Hasta llegar a un único punto de iteración n-ésima. Dicho punto es precisamente el valor de la parametrización de la curva de Bézier en t, con $t \in [0,1]$

$$C(t) = p_0^n(t)$$

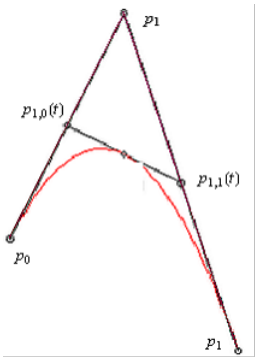


Figura 2.6: Curva Bézier con tres punto de control

Las ventajas del algoritmo de Casteljau son muy sencillas por el hecho de que involucra tan sólo sumas, ya que todos los términos son positivos.

2.5. EL ALGORITMO DE BOOR

El algoritmo de Boor es una generalización del algoritmo de Casteljau. Este proporciona una manera rápida y estable numéricamente para encontrar un punto en una curva B-Spline dado un u en el dominio.

Consideremos la combinación lineal:

$$s(u) = \sum_i p_i^0 N_i^n(u)$$

De B-Spline de grado n sobre una secuencia de nodos (u_i). Sin pérdida de generalidad podemos suponer que la secuencia de nodos y la sumatoria es en todo $\mathbb{R}$.

Supongamos que $u \in [u_n, u_{n+1})$, entonces

$$s(u) = \sum_{i=0}^{n} p_i^0 N_i^n(u) \tag{2.12}$$

Usando la relación de recurrencia obtenemos que:

$$s(u) = \sum_{i=1}^{n} p_i^0 N_i^{n-1}(u)$$

$$\vdots$$

$$s(u) = \sum_{i=n}^{n} p_i^0 N_i^0(u) = p_n^n$$

Donde los p_i^r están dados por las combinaciones afines

$$p_i^r(t) = (1-t)p_i^{r-1}(t) + tp_{i+1}^{r-1}(t) \tag{2.13}$$

$$t = t_i^{n-r} = \frac{(u - u_i)}{u_{i+n+1-r} - u_i} \tag{2.14}$$

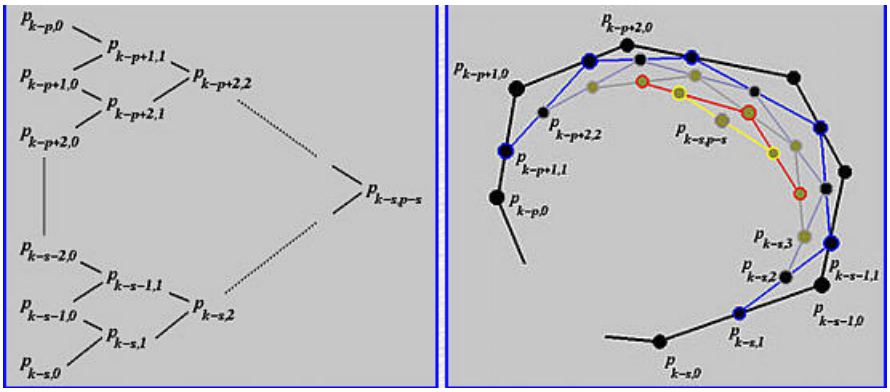


Figura 2.7: proceso de corte de esquina

En el siguiente diagrama de la Figura 2.7, todos los $p_{i,0}$ están en la columna de la izquierda. Desde la columna cero y coeficientes $u_{i,1}$, se puede calcular $p_{i,1}$. A partir de esta primera columna y los coeficientes de $u_{i,2}$, la segunda columna se calcula y así sucesivamente. Puesto que hay $(k-s)-(k-p) = p+s+1$ puntos en el cero-ésima columna y puesto que cada columna tiene un punto menos que el anterior, se necesita $p-s$ columnas para reducir el número de puntos en que la columna a 1. Por ello, el último punto es $p_{k-s,p-s}$.

Aunque este proceso se parece a la que se obtiene a partir del algoritmo de Casteljau, en realidad son muy diferentes Primero, en el algoritmo de Casteljau, los puntos de división se calculan con un par de números $1-u$ y u que nunca cambian durante todo el procedimiento de cálculo, mientras que bajo el algoritmo de de Boor estos pares de números son diferentes y dependen del número de columna y el número de punto de control. En segundo lugar, el algoritmo de Casteljau se puede utilizar para la subdivisión curva, mientras que los puntos de control intermedios generados por el algoritmo de Boor no son suficientes para este propósito. En tercer lugar, en el algoritmo de Boor sólo $p+1$ puntos de control están involucrados en el cálculo, mientras que el algoritmo de Casteljau utiliza todos los puntos de control. Desde los puntos de control p_{k-1} a p_k definen un casco convexo que contiene el segmento de la curva en nudo escala $[u_i, u_{i+1}]$, el cálculo del algoritmo de Boor se lleva a cabo dentro de la envolvente convexa correspondiente.

Capítulo 3

CURVAS DE BÉZIER

Los polinomios de Bernstein $B_i^n(u)$ de grado n forman una base para el espacio vectorial de polinomios de grado menor o igual que n. Por tanto toda curva polinómica $B(u)$ de grado $\leq n$ tiene una única representación de Bézier

$$B(u) = \sum_{i=0}^{n} c_i B_i^n(u) \tag{3.1}$$

La transformación afín

$$u = a(1-t) + bt \quad a \neq b$$

Deja invariante el grado de B, por lo tanto$B(u(t))$ también tiene una única representación de grado n, en términos de los $B_i^n(t)$

$$B(u(t)) = \sum_{i=0}^{n} p_i B_i^n(t)$$

Los coeficientes p_i en $\mathbb{R}^d$ se denominan puntos Bézier y son los vértices del polígono de Bézier de $B(u)$ sobre el intervalo [a,b]. Nos referimos a t como el parámetro local y a u como el parámetros global de B.

La representación de Bézier de la curva polinómica sobre el intervalo $[a, b]$ hereda las propiedades de los polinomios de Bernstein.

- La simetría de los polinomios de Bernstein implica

$$B(u) = \sum_{i=0}^{n} p_i B_i^n(t) = \sum_{i=0}^{n} p_{n-i} B_i^n(s) \tag{3.2}$$

Donde

$$u = a(1-t) + bt = b(1-s) + as$$

- Los extremos del segmento de curva $[a, b]$ son

$$B(a) = p_0 \; y \; B(b) = p_n$$

Como los polinomios de Bernstein suman 1

- $B(u)$ es una combinación afín de sus puntos Bézier.
- La representación de Bézier es afinmente invariante, es decir, dada una aplicación afín Φ, la curva $\Phi(B)$ tiene a lo mas $\Phi(p_i)$ como puntos de control, sobre $[a, b]$

Como los polinomios de Bernstein son no negativos en [0,1]

- Para todo $u \in [a, b]$, $B(u)$ es una combinación convexa de los p_i.

3.1. REPRESENTACIÓN DE LAS CURVAS DE BÉZIER

Se denomina curva de Bézier a un método de definición de una curva en serie de potencias. El método consiste en definir algunos puntos de control, a partir de los cuales se calculan los puntos de la curva**[2]**.

Describiremos el método de construcción recursivo conocido como algoritmo de Casteljau. Para dos puntos, la curva es un segmento recto, definido en forma paramétrica por interpolación de los puntos extremos:

$$P = (1-t)P_0 + tP_1$$

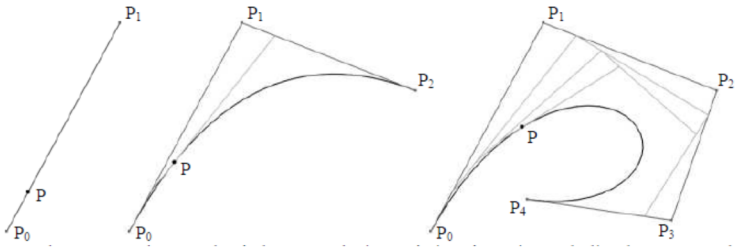


Figura 3.1: Proceso iterado de la curva Bézier

Donde los P_i son los puntos de control y $t \in [0, 1]$es un parámetro.

Agregando un punto de control más la curva adquiere más interés: se interpola linealmente en cada uno de los segmentos y luego entre los puntos resultantes, siempre con el mismo valor del parámetro. Para más puntos de control, el proceso se repite en forma iterativa.

La poligonal que forman los puntos de control se conoce como polígono de control.

Cambiemos ligeramente la notación para favorecer la interpretación de las propiedades y el código fuente. Indicaremos con un superíndice el nivel de iteración: llamemos P_i^0 a los puntos de control dados (nivel 0) y aumentemos en uno el superíndice con cada iteración:

$$P_1^{j+1} = (1-t)P_1^j + tP_{i+1}^j$$

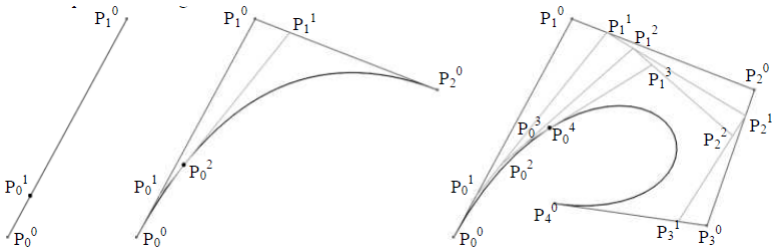


Figura 3.2:

Con esta notación, si hay $n+1$ puntos de control (0 a n) el punto de la curva es el P_0^n, y la figura 3.1 queda como la fugura 3.2:

Por razones históricas se llama orden de una curva de Bézier a la cantidad de puntos de control. Para dos puntos de control, la curva es de segundo orden y primer grado, por cada punto de control que se agrega, se agrega además un paso de interpolación, en donde los términos en u quedan multiplicados por $t\,o\,(1-t)$ y por lo tanto se aumenta en uno el grado del polinomio. Nosotros no utilizaremos el orden sino el grado pero es necesario definirlo.

$o = n+1$ (o :=Orden = grado+1)

A modo de ejemplo, veamos el desarrollo para una curva de 3er grado (recordar que el superíndice de u indica potencia):

$$P_0^3 = (1-t)P_0^2 + tP_1^2$$

$$= (1-t)[(1-t)P_0^1 + tP_1^1] + t[(1-t)P_1^1 + tP_2^1]$$

$$= (1-t)^2[(1-t)P_0^0 + tP_1^0] + 2(1-t)t[(1-t)P_1^0 + tP_2^0] + t^2[(1-t)P_2^0 + tP_3^0]$$

$$= (1-t)^3P_0^0 + 3(1-t)^2tP_1^0 + 3(1-t)t^2P_2^0 + t^3P_3^0$$

Esa es la forma en que se realiza el cálculo recursivo de sucesivos puntos de la curva, de la definición $P_n^0 = P_n$, entonces la función paramétrica de una curva Bézier con los puntos de control P_i se escribe como:

$$C(u) = \sum_{i=0}^{i=n} B_i P_i t \in [0,1] \tag{3.3}$$

Nuevamente cambiando un poco la notación y usando el algoritmo de Casteljau que se deduce del algoritmo geométrico recursivo.

3.2. CURVAS DE BÉZIER EXPRESADAS EN TÉRMINOS DE LOS POLINOMIOS DE BERNSTEIN

Se definen los polinomios de Bernstein $B_i^n(t)$ como:

$$B_i^n(u) := \begin{pmatrix} n \\ i \end{pmatrix} u^i(1-u)^{n-i} \tag{3.4}$$

$$i = 0, \ldots, 1$$

Donde se recuerda que

$$\begin{pmatrix} n \\ i \end{pmatrix} = \begin{cases} \frac{n!}{i!(n-i)!} & 0 \leq i \leq n \\ 0 & \textit{en caso contrario} \end{cases} \tag{3.5}$$

Los polinomios de Bernstein cumplen las siguientes propiedades:

Recursividad

$$B_i^n(t) = (1-t)B_i^{n-1}(t) + tB_{i-1}^{n-1}(t)$$

Con

$$\begin{cases} B_0^0(t) \equiv 1 \\ B_j^n \equiv 0 \quad para\, j \notin [0, \ldots, n]) \end{cases}$$

El conjunto forma una partición de la unidad.

$$\sum_{i=0}^{n} B_i^n(t) = 1 \tag{3.6}$$

Dadas estas definiciones, podemos expresar una curva de Bézier $P^n(t)$ como:

$$P^n = P_0^n(t) = \sum_{i=0}^{n} P_i B_i^n(t)$$

Los coeficientes $B_i^n(t)$ son los polinomios de Bernstein de grado n, que pueden ser considerados como la base de un espacio donde el punto se identifica mediante $n+1$ coordenadas (vectores y no escalares) que son los puntos de control. Los polinomios de Bernstein se pueden graficar para observar algunas de sus propiedades**[1].**

Los polinomios de BERNSTEIN de grado 1 están dados por

$$B_0(t) = 1 - t$$

$$B_1(t) = t$$

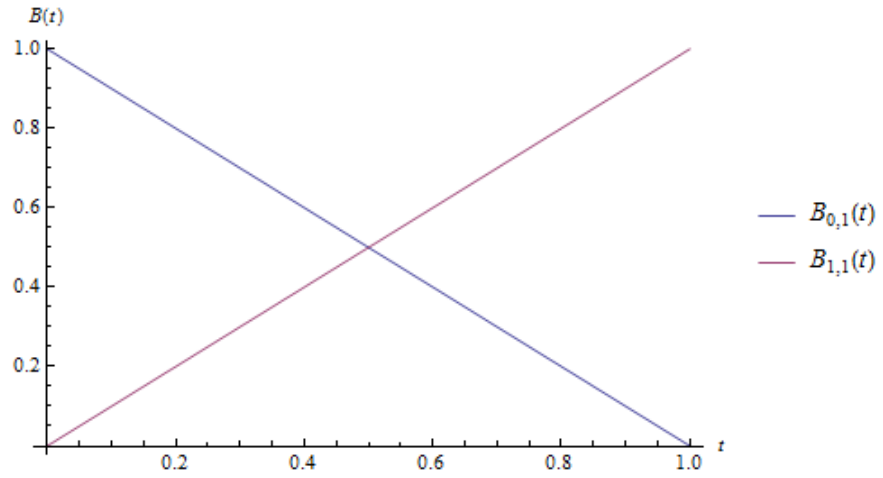


Figura 3.3: Polinomios de Bernstein de primer grado

La curva de Bézier está dada de la siguiente forma $(1-t)P_0 + tP_1$ con puntos de control P_0 y P_1 Los polinomios de BERNSTEIN de grado 2:

$$B_0[t] = (1-t)^2$$

$$B_1[t] = 2(1-t)t$$

$$B_2[t] = t^2$$

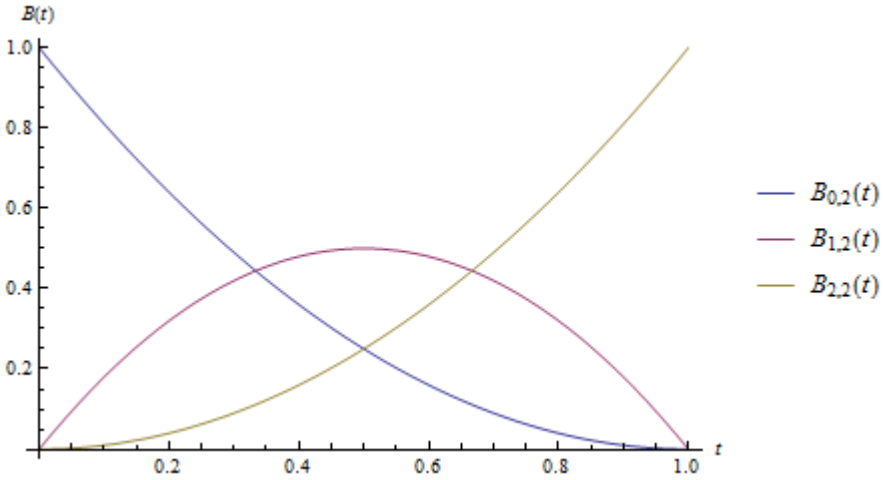


Figura 3.4: Polinomios de Bernstein de segundo grado

La curva de Bézier $(1-t)^2P_0 + 2(1-t)tP_1 + t^2P_2$ con puntos de control P_0, P_1 y P_2.

La combinación convexa de puntos de control implica que la curva se encuentra dentro de su envolvente convexo.

Los polinomios de BERSTEIN de grado 3

$$B_0[t] = (1-t)^3$$

$$B_1[t] = 3(1-t)^2 t$$

$$B_2[t] = 3(1-t)t^2$$

$$B_3[t] = t^3$$

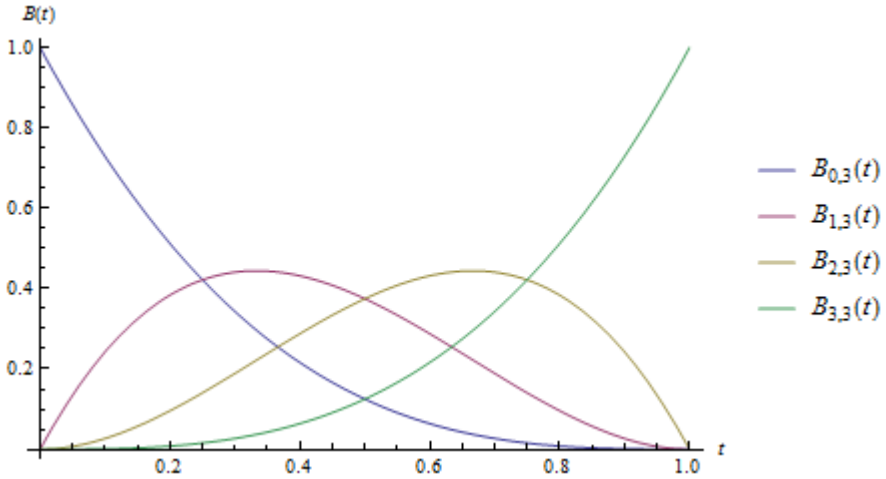


Figura 3.5: Polinomios de Bernstein de tercer grado

La curva de Bézier $(1-t)^3P_0 + 3(1-t)^2tP_1 + 3(1-t)t^2P_2 + t^3P_3$ con puntos de control P_0, P_1, P_2 y P_3.

Los polinomios de BERSTEIN de grado 4

$$B_0[t] = (1-t)^4$$

$$B_1[t] = 4(1-t)^3 t$$

$$B_2[t] = 6(1-t)^2t^2$$

$$B_3[t] = 4(1-t)t^3$$

$$B_4[t] = t^4$$

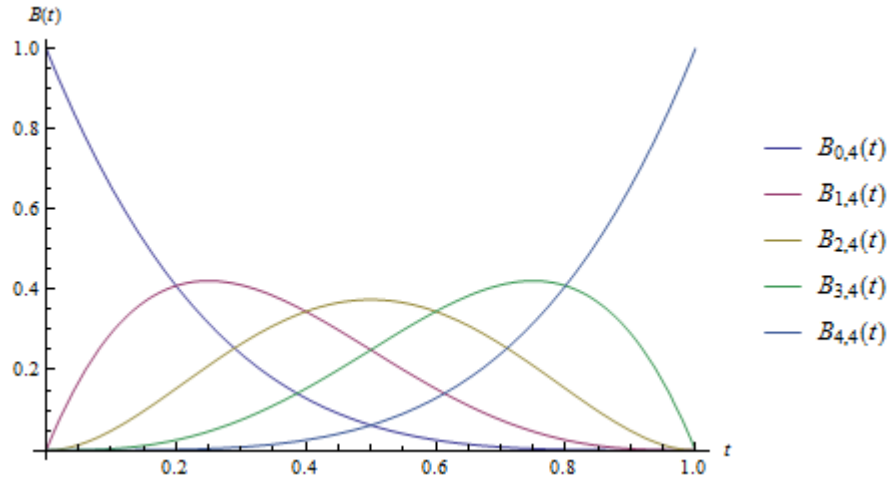


Figura 3.6: Polinomios de Bernstein de cuarto grado

La curva de Bézier $(1-t)^4P_0+4(1-t)^3tP_1+6(1-t)^2t^2P_2+4(1-t)t^3P_3+t^4P_4$ con puntos de control P_0, P_1, P_2, P_3 y P_4 .

Cada punto de control se corresponde con un polinomio de Bernstein que actúa entonces como función de forma, aunque en este contexto suelen denominarse blending functions (funciones mezcladoras) y miden la influencia de cada punto de control en cada punto de la curva. B_0 Toma el valor 1 en $u = 0$ mientras que $B_n = 1$ en $u = 1$ eso implica que la curva interpolada pasa por los puntos de control inicial y final. Además ningún B toma el valor unitario en ningún otro valor de u, por lo que P_1 y P_n son los únicos puntos de control interpolados.

La curva entera cambia al mover un solo punto de control, el punto variable es afectado por todos los puntos de control, esto se llama control global. Esta es una propiedad indeseable en CAGD, donde siempre se pretende el control local de la curva, es decir que la curva varíe sólo en las cercanías del punto de control movido. Más adelante veremos cómo se arregla esto con las B-Spline.

Un punto de la curva es combinación afín convexa de los puntos de control, Podemos verlo de dos modos: a) Por ser una cadena de combinaciones convexas b) Mediante el binomio de Newton, podemos ver que los polinomios de Bernstein suman uno.

$$\sum B_i = \sum_{i=0}^{n} \left(\begin{array}{c} n \\ i \end{array} \right) u^i (1-u)^{n-i} = ((1-u)+u)^n = 1 \tag{3.7}$$

Cualquier punto (y por lo tanto, toda la curva) estará entonces dentro o en el límite del convex-hull de los puntos de control. La curva de Bézier, es combinación afín convexa de los puntos de control.

Siendo una combinación afín, tienen invariancia afín. Como ya hemos visto, la transformación afín mantiene las combinaciones afines, para transformar la curva solo hay que transformar sus puntos de control y construirla, en lugar de tener que transformar una miríada de puntos de la curva.

Unicidad: Dada una curva de Bézier de grado n hay un solo conjunto de $n+1$ puntos de control que la definen. La demostración se hace por el absurdo, y requiere una demostración de la independencia lineal de los polinomios de Bernstein, que es la de las series de potencias.

3.3. PROPIEDADES DE LAS CURVAS DE BÉZIER

Una construcción muy útil basada en el algoritmo de Casteljau es el blossom, que denominaremos polarización de la parametrización, debido a que está relacionado con esta del mismo modo que una forma cuadrática con su forma bilineal asociada, también llamada polarización.

La idea es sencilla; en vez de interpolar siempre para el mismo valor de t en los n pasos del algoritmo o también llamado nivel del polígono de control, interpolamos en el valor t_1 en el primer paso, en el valor t_2 en el segundo y en el valor t_n en el último nivel del polígono de control. Obtenemos así un punto que denotaremos $c[t_1, .., t_n]$. No se corresponde con ningún punto de la curva, salvo en el caso trivial en el que todos los t_i son iguales.

$$c_i^1(t_1) := (1-t_1)c_i + t_1 c_{i+1}, i = 0, \ldots, n-1$$

$$c_i^r(t_1,\ldots,t_r) := (1-t_r)c_i^{r-1}(t_1,\ldots,t_{r-1}) + t_r c_{i+1}^{r-1}(t_1,\ldots,t_{r-1})$$

$$c(t_1,\ldots,t_n) := c_0^n(t_1,\ldots,t_n) \qquad i = 0,\ldots,n-r \qquad r = 1,\ldots,n$$

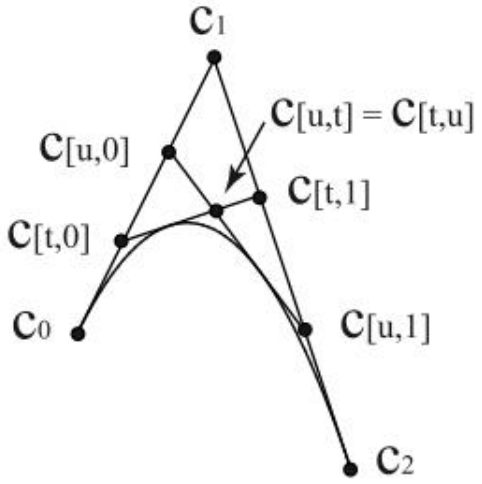


Figura 3.7: Curva Bézier con tres puntos de control

Una propiedad importante de la polarización es que permite reconstruir el polígono de la curva de Bézier, a diferencia de la parametrización, que solo proporciona los vértices inicial y final. Así para calcular el polígono de control hemos de emplear la polarización, que requiere a su vez el polígono de control. Sin embargo, es útil para obtener polígonos de control de curvas que se obtienen a partir de una dada, de la cual si conocemos el polígono. La polarización es un resultado muy potente, ya que permite calcular el polígono de control de cualquier tramo de una curva de Bézier.

3.4. DERIVADAS

La derivada de un polinomio de Bernstein de grado n es la siguiente.

De la definición de los polinomios de Bernstein se obtiene:

$$\frac{d}{dt}B_i^n(t) = n(B_{i-1}^{n-1}(t) - B_i^{n-1}(t)) \tag{3.8}$$

para $i = 0,\ldots,n$.

Para unificar la notación hemos supuesto, que $B_{-1}(t) \equiv B_n^n(t) \equiv 0$

Entonces

$$B(u) = \sum_{i=0}^{n} p_i B_i^n(t), \qquad t = \frac{(u-a)}{(b-a)}$$

$$\frac{d}{dt}B(u) = \frac{n}{b-a}\sum_{i=0}^{n-1} \triangle p_i B_i^{n-1}(t)$$

Donde

$$\Delta p_i = p_{i+1} - p_i.$$

Si $B(u)$ se considera un punto, entonces $B'(u)$ es un vector. Al sumarle un punto a $B'(u)$ se obtendrá un punto. En particular $0 + B'(u)$ se denomina el primer hodógrafo de B.

Obtenemos la r-ésima derivada de B

$$B^r = \frac{n!}{(n-r)!(b-a)^r}\sum_{i=0}^{n-r} \triangle^r p_i B_i^{n-r}(t)$$

Donde $\triangle^r p_i = \triangle^{r-1} p_{i+1} - \triangle^{r-1} p_i$.

A partir de las fórmulas de las derivadas y de la propiedad de interpolación de los puntos de control en los extremos se obtiene el siguiente resultado de Pierre Bézier.

Las derivadas r-ésima hasta el orden n, de $B(u)$ en $t = 0$ $(t = 1)$ depende de los primeros (últimos) $r + 1$ puntos de Bézier. El recíproco de esta observación también es cierto.

3.5. INTEGRACIÓN

Tenemos por definición que:

$$B(u) = \sum_{i=0}^{n} p_i B_i^n(t), \quad t = \frac{u-a}{b-a}$$

Integrando tenemos que

$$c(u) = \int B(u)du = \sum_{i=0}^{n+1} c_i B_i^{n+1}(t) \tag{3.9}$$

Donde

$$c_i = c_{i-1} + \frac{b-a}{n+1} p_{i-1}$$

$$= c_0 + \frac{b-a}{n+1}(p_0 + \cdots + p_{i-1}), \qquad i = n+1, \ldots, 1$$

Y p_0 es una constante de integración. Esto puede verificarse facilmenete derivando respecto a $c(u)$.

A partir de la formula anterior y de que $B(u)$ interpola los extremos p_0 y p_n se deduce la siguiente igualdad

$$\int_a^b B(u)du = \frac{b-a}{n+1}(p_0 + \cdots + p_n)$$

Y en particular,

$$\int_0^1 B_i^n(t)dt = \frac{1}{n+1} i = 0, \ldots, n \tag{3.10}$$

3.6. ELEVACIÓN DE GRADO

En algunos casos, nos puede interesar cambiar el grado de la curva para tener más grados de libertad a la hora de modificarla. Esto es muy fácil de realizar en la base canónica, bastaba añadir un termino t^{n+1} y la parametrización formalmente seria de grado $n+1$. La representación de Bézier también permite elevar el grado de manera sencilla. Basta multiplicar la parametrización por $1 = (1 - t + t)$,

$$c(t) = \sum_{i=0}^{n} c_i B_i^n(t)$$

$$= \sum_{i=0}^{n} c_i \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}(1-t+t)$$

$$= \sum_{i=0}^{n} c_i \frac{n!}{i!(n-i)!} (t^{i+1}(1-t)^{n-i} + t^i(1-t)^{n+1-i})$$

$$= \sum_{i=0}^{n} c_i \frac{i+1}{n+1} B_{i+1}^{n+1}(t) + (n+1-i)/(n+1) B_i^{n+1}(t))$$

$$= \sum_{i=0}^{n} \frac{i}{n+1} c_i + \frac{n+1-i}{n+1} c_i) B_i^{n+1}(t)$$

$$= \sum_{i=0}^{n+1} c_i^1 B_i^{n+1}(t)$$

E identificando los nuevos vértices,$c_0^1, \ldots, c_{n+1}^1$

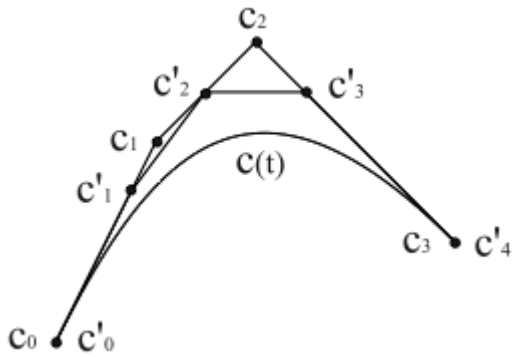


Figura 3.8: Curva Bézier con cuatro puntos de control

$$c_i^1(1-\frac{i}{n+1})c_i+\frac{i}{n+1}c_{n+1}$$

En la figura anterior se muestra los polígonos $c_0,\ldots,c_3$, $c'_0,\ldots,c'_4$ corresponden a la misma curva de Bézier.

3.7. INTERPOLACIÓN

Las curvas de Bézier pueden utilizarse para interpolar entre varios puntos, conocidos los valores que les corresponden del parámetro t. Supongamos que tenemos $n+1$ puntos $a_0,\ldots,a_n$ y que queremos obtener una curva $c(t)$ definida en el intervalo$[0,1]$, que verifique

$$c(t_0)=a_0$$

$$c(t_n)=a_n$$

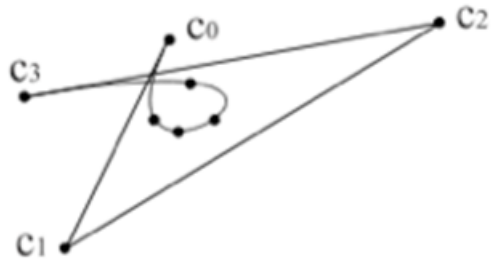


Figura 3.9: Cúbica interpolante de cuatro puntos

Para unos valores $t_0, \ldots, t_n$ del parámetro

Para resolver el problema de interpolación, debemos encontrar el polígono de control de la curva de grado n que verifique

$$\sum_{i=0}^{n} c_i B_i^n(t_j) = a_j \qquad j = 0, \ldots, n \tag{3.11}$$

Es decir, se trata de resolver el sistema lineal de $n+1$ ecuaciones

$$\begin{pmatrix} B_0^0(t_0) & \cdots & B_n^0(t_0) \\ \vdots & \ddots & \vdots \\ B_0^n(t_n) & \cdots & B_n^n(t_n) \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} \tag{3.12}$$

Este sistema está determinado y tiene solución única, como se comprueba, por ejemplo, expresándolo en una base más cómoda, la canónica

$$\begin{pmatrix} 1 & \cdots & t_n^0 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & t_n^n \end{pmatrix} \begin{pmatrix} \overline{c}_0 \\ \vdots \\ \overline{c}_n \end{pmatrix} = \begin{pmatrix} \overline{a}_0 \\ \vdots \\ \overline{a}_n \end{pmatrix}$$

Ya que el determinante de la matriz del sistema de ecuaciones lineales es conocido de Vandermonde.

$$\begin{vmatrix} 1 & \cdots & t_n^0 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & t_n^n \end{vmatrix} = (t_1 - t_0)\cdots(t_n - t_0)(t_2 - t_1)\cdots(t_n - t_1)\cdots(t_n - t_{n-1})$$

Que se anula si y sólo si al menos dos valores t_i,t_j son iguales, que no es nuestro caso. Luego el rango de la matriz del sistema es $n+1$ y el sistema está bien determinado.

En notación matricial, nuestro sistema es $BC = A$ donde B es la matriz de los valores de los polinomios de Bernstein en $t_0, .., t_n$ y C, A son las matrices cuyas filas son los componentes, respectivamente, de los vértices del polígono de control y de los datos del problema. La solución formal seria $C = B^{-1}A$, aunque la manera eficiente de obtenerla será por algún algoritmo de resolución numérica.

3.8. APROXIMACIÓN

Lo más común, sin embargo, no es que tengamos $n+1$ puntos y queramos obtener una curva de grado n que pase por todos ellos, ya que, como ya se ha

dicho, los grados altos presentan oscilaciones espureas. Es más interesante el caso en el que tenemos $m+1$ puntos y queremos obtener la curva de grado n que más se aproxime a nuestro conjunto de puntos. Luego matizaremos que quiere decir "mejor aproximación".

Si lo planteáramos como un problema de interpolación, no habría en general solución,

$$\begin{pmatrix} B_0^0(t_0) & \cdots & B_n^0(t_0) \\ \vdots & \ddots & \vdots \\ B_0^n(t_n) & \cdots & B_n^n(t_n) \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix}$$

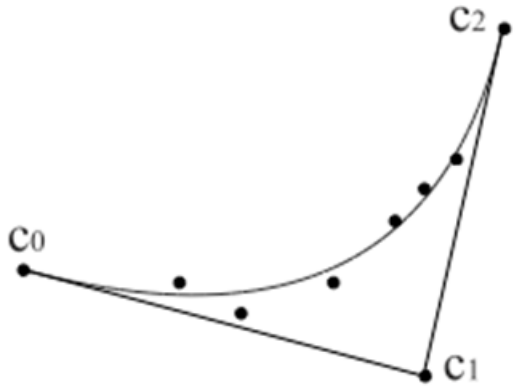


Figura 3.10: Parábola aproximante de seis puntos

Ya que obtendríamos un sistema sobre determinado si, como es de esperar, $m > n$. Pero podemos obtener una pseudoinversa mediante el siguiente artificio. En vez de resolver el sistema $BC = A$ de $m+1$ecuaciones y $n+1$ incógnitas, resolveremos el sistema $B^{-1}\ BC = B^tA$, que es un sistema de $n+1$ ecuaciones y $n+1$ incógnitas. Como la matriz del nuevo sistema $B^{-1}B$ es definida positiva y simétrica, tiene determinante positivo y el problema tiene solución única $C = (B^tB)^{-1}B^tA$.

Ahora bien, ¿Qué hemos resuelto realmente? Porque el problema de interpolación sabemos a ciencia que no tiene solución. Lo que hemos resuelto es el problema de aproximación por mínimos cuadrados, es decir, la desviación de la parametrización de la curva solución $c(t)$ de los valores $a_i = c(t_i)$ es mínima. Denotando por$\|\|$la longitud de un vector y por $\langle , \rangle$, el producto escalar y desarrollando la longitud $\| v - w \|^2 = \| v \|^2 + \| w \|^2 - 2\langle v, w \rangle$

$$L(c_0, .., c_n) := \sum_{i=0}^{m} \| c(t_i) - a_i \|^2$$

$$= \sum_{i=0}^{m} \| \sum_{j=0}^{n} c_j B_j^n(t_i) - a_i \|^2$$

$$= \sum_{i=0}^{m} \left\langle \sum_{j=0}^{n} c_j B_j^n(t_i), \sum_{k=0}^{n} c_k B_k^n(t_i) \right\rangle + \langle a_i, a_i \rangle - 2 \left\langle \sum_{k=0}^{n} c_k B_k^n(t_i), a_i \right\rangle$$

El extremo de la desviación se obtiene derivando respecto a cada componente de las variables c_k e igualando a cero.

$$0 = 2 \sum_{i=0}^{m} \sum_{j=0}^{n} c_j B_j^n(t_i) - a_i B_k^n(t_i)$$

Que en notación matricial es precisamente $B^{-1}BC - B^tA$ con lo cual la construcción heurística del comienzo de la sección proporciona la curva de grado n mejor aproximación por mínimos cuadrados a los datos del problema.

3.9. ELECCIÓN DE LOS NUDOS

En los problemas de interpolación y aproximación que hemos tratado hasta el momento, hemos dado por sentado que conocemos una sucesión de nudos, $t_0, .., t_m$, que determinan el momento en el cual la curva pasa por los puntos dados del problema $c(t_i) = a_i$.

Este planteamiento es el adecuado cuando en un experimento vamos aobteniendo datos para distintos valores, por ejemplo, del tiempo, es decir, cuando interesa la parametrización, la ordenación temporal de los puntos de la gráfica.

Sin embargo, el caso común en el diseño es distinto. En el diseño importa exclusivamente la gráfica de la curva, es decir, su forma, no la manera en la que esta es recorrida. La parametrizazión es un artificio nuestro para describirla.

Por tanto, lo más frecuente es que no dispongamos de una sucesión de valores t_i privilegiada, sino tan sólo de un conjunto de puntoa, y tal vez tangentes, por los cuales queremos que pase nuestra curva.

Una elección ingenua consistiría en elegir la partición uniforme, es decir aquella en la que las diferencias $\triangle t_i = k$ s on iguales. Notese que, por la invariancia bajo transformaciones afines del parámetro $\bar{t} = at + b$, no importa ni el origen de la partición, ni la superación entre nudos.

Esta partición puede ser útil cuando los datos están distribuidos de manera homogenea, pero, obviamente, no será útil cuando la separación entre los puntos sea muy dispar.

Una segunda solución al problema consiste en suponer que la curva está parametrizada aproximadamente por su longitud de arco, s,

$$s(t) = \int_a^b dt \parallel c'(t) \parallel$$

Y tomar como estimación de esta medida la longitud de la cuerda entre los puntos dados.

$$\triangle t_i = k \parallel \triangle a_i \parallel$$

Donde k es una constante de escala.

Esta solución del problema mejora los resultados de la partición uniforme, pero fracasa si la t está alejada de la longitud de arco o si esta difiere mucho de la longitud de la cuerda entre los punto. Esto sucede, por ejemplo, si la curva presenta mucha variación.

3.10. SUPERFICIE DE BÉZIER

La manera más sencilla de construir una superficie consiste en barrer una curva en el espacio, en la representación de Bézier. Los puntos de control de esta curva se mueven a su vez siguiendo curvas de Bézier, cuyos puntos de control definen la superficie.

La representación de la superficie por medio de estos puntos de control tiene propiedades similares a la representación de Bézier univariada. Por esta razón se puede trabajar con estas superficies aplicando los algoritmos para curvas. También, se pueden construir volúmenes multidimensionales barriendo una superficie o un volumen en el espacio de manera que sus puntos de control se mueven a lo largo de diversas curvas. Análogamente se obtienen mallas de control con propiedades similares de las de representaciones de curvas.

PRODUCTOS TENSORIALES

Para mostrar cómo se construye una superficie producto tensorial a partir de curvas, sean

$$A_0(u), \ldots, A_m(u) \quad y \quad B_0(v), \ldots, B_n(v)$$

dos conjuntos de funciones independientes construyamos la curva

$$p(u) = \sum_{i=0}^{m} a_i A_i(u)$$

Si cada uno de los puntos de control de esta curva yace sobre una curva de Bézier, esto es si interpolamos una curva de Bézier y pedimos que pase por los puntos de control se obtiene

$$a_i = a_i(v) = \sum_{j=0}^{n} b_{i,j} B_j(v)$$

sustituyendo esta expresión en $p(u)$ se obtiene La superficie p dada por:

$$p(u,v) = \sum_i \sum_j b_{i,j} A(u) B_j(v)$$

se denomina superficie producto tensorial. Resumiendo podemos decir que la superficie

Se construye con productos de los polinomios $B_i = \begin{pmatrix} n \\ i \end{pmatrix} u^i (1-u)^{n-i}$ y una familia de puntos de control $P_{i,j}$:

$$B(u,v) = \sum_i \sum_j P_{i,j} B_i(u) B_j(v) \qquad u, v \in [0,1] \tag{3.13}$$

Una superficie polinómica de Bézier de bigrado (m,n) se define por una malla de control,

$$\begin{pmatrix} P_{0,0} & \cdots & P_{0,n} \\ \vdots & \ddots & \vdots \\ P_{m,0} & \cdots & P_{m,n} \end{pmatrix}$$

Y la función paramétrica está dada por:

$$B(u,v) = \sum_{i=1}^{m} \sum_{j=1}^{n} B_i(u) B_j(v) P_{i,j} \tag{3.14}$$

$$u \in [0,1], \;\; v \in [0,1]$$

En donde

$$B_i(u) = \begin{pmatrix} m \\ i \end{pmatrix} u^i (1-u)^{m-i}$$

$$B_j(u) = \begin{pmatrix} n \\ j \end{pmatrix} u^j (1-u)^{n-j}$$

Y

$$\begin{pmatrix} m \\ i \end{pmatrix} = \frac{m!}{i!(m-i)!}$$

$$\begin{pmatrix} n \\ j \end{pmatrix} = \frac{n!}{j!(n-j)!}$$

Interpolación bilineal, observación

$$B(u,v) = \sum_{i=1}^{m} B_i(u)(\sum_{j=1}^{n} B_j(v)P_{ij})$$

$$= \sum_{j=1}^{n} B_j(u)(\sum_{i=1}^{m} B_i(v)P_{ij})$$

Además también se cumple que:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} B_i(u)B_j(v) = 1 \tag{3.15}$$

SUPERFICIES PRODUCTO TENSORIAL DE BÉZIER

En esta sección se mostrara como extender a las superficies tensoriales de Bézier las técnicas desarrolladas para curvas. Esto facilitara la extensión de las técnicas de B-Spline al caso de superficies**[2]**.

Una superficie polinómica $b(u) = b(u,v)$ es de grado $m = (m,n)$ si es de grados m y n en u y v, respectivamente. La pareja de transformaciones afines.

$$u = c_1(1-s) + d_1 s \qquad , \qquad v = c_2(1-t) + d_2 t$$

mantiene invariante el grado de b, es decir, $b(s,t) = b(u(s,t))$ es también de grado m en $s = (s,t)$. El polinomio $b(s,t)$ se puede considerar como un polinomio de grado$n$ en t cuyos coeficientes son polinomios de grado m en s. Entonces $b(s,t)$ tiene la representación de Bézier

$$b(s,t) = \sum_i b_i(t)B_i^m(s) = \sum_i \sum_j b_{ij}B_i^m(s)B_j^n(t)$$

o en notación más compacta

$$b(s) = \sum_i b_i B_i^m(s)$$

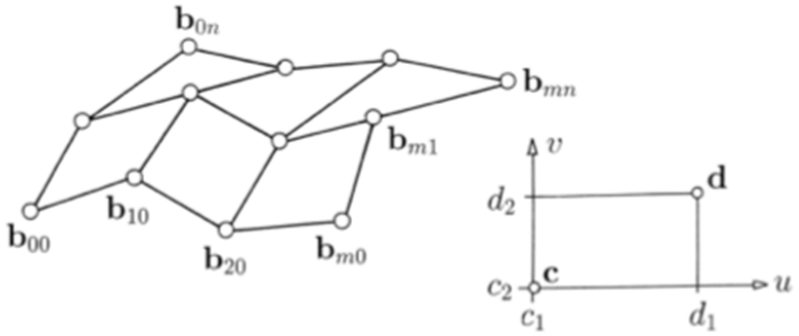


Figura 3.11: Malla de Bézier

donde $i = (i, j)$.

Los coeficientes b_i se denominan los puntos de Bézier de $b(u)$ sobre el intervalo$[c, d] = [c_1, d_1] \times [c_2, d_2]$. Cuando estos puntos se conectan como indica la Figura 3.11 nos referimos a la configuración resultante como la malla de Bézier de b. La variable s se denomina parámetro local y u es el parámetro global.

Como en el caso de las curvas, las superficies tensoriales de Bézier heredan las propiedades de los polinomios de Bernstein.

La simetría de los polinomios de Bernstein implica

$$b(s) = \sum_{i,j} b_{ij} B_{ij}^m(s) = \sum_i b_{m-1,j} B_{ij}^m(\widetilde{s}) \tag{3.16}$$

Donde $s = (1 - \widetilde{s}, t)$, etc.

La utilidad práctica de la simetría consiste en que para parametrizar el mismo parche tomando las filas o columnas, o de ambas.

Como una curva pasa por su primer y último punto Bézier, se tiene que el borde de la malla de Bézier determina las cuatro curvas fronteras del parche$b[c, d]$.

Por ejemplo, se tiene

$$b(c_1, v) = \sum_j b_{0j} B_j^n(v)$$

En particular, resulta que

- Las cuatro esquinas del parche de su malla Bézier coinciden, esto es:

$b(c_1, d_1) = b_{00}, b(c_1, d_2) = b_{0n}$. etc

Como los polinomios de Bernstein suman uno;

- Sus productos también forman una partición de la unidad.

$$\sum_i B_i^m = \sum_i 1 \cdot B_j^m = 1 \tag{3.17}$$

Por lo tanto $b(u)$ es una combinación afín de sus puntos de Bézier y la representación de Bézier afínmente invariante.

Como los polinomios de Bernstein no son negativos $[0, 1]$, se tiene que:

- Para todo $u \in [c, d]$, $b(u)$ es una combinación convexa de los b_i.

Por lo tanto

- El parche $b[c, d]$ está contenido en la cápsula convexa de sus puntos de Bézier

Observación: Usando la propiedad de la cápsula convexa para cada componente se obtiene una caja de acotación para el parche $b[c, d]$

$$b(u) \in \begin{bmatrix} \underset{i}{minb_i} & , \underset{i}{maxb_i} \end{bmatrix} \qquad para \quad u \in [c, d]$$

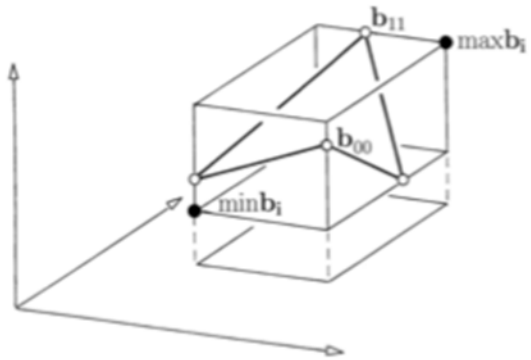


Figura 3.12: Caja de acotación

Donde $minb_i$ es el punto cuya primera coordenada es el mínimo de las primeras coordenadas de las b_i y de modo similar se definen las demás coordenadas de los $minb_i$. La definición de $maxb_i$ es análoga.

3.11. ALGORITMO PARA CONSTRUIR UNA SUPERFICIE DE BÉZIER EN MATHEMATICA

SUPERFICIES DE BEZIER

Se construyen con productos de los polinomio

$B_i\,(t) = \binom{n}{i}\, t^i\,(1-t)^{n-i}$

y una familia de puntos de control P_{ij} :

$$B\,(u, v) = \sum_{i=0}^{n}\sum_{j=0}^{n} B_i\,(u)\,B_j\,(v)\,P_{ij}$$

$u, v \in [0, 1]$

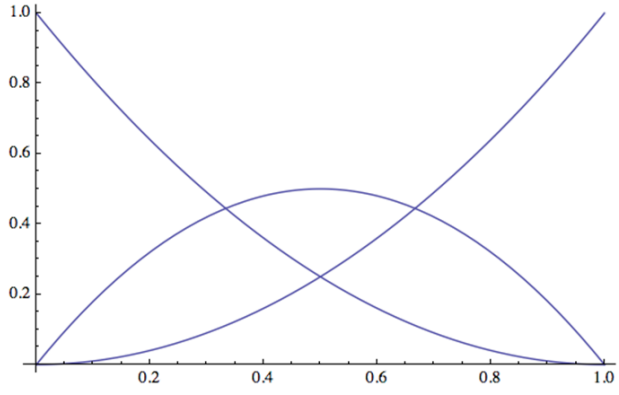
```
n = 2; (*Da el numero de puntos mas uno en R^2*)
```

▪ Polinomios de Berstein

```
Do[B_i[t_] = n!/(i! (n - i)!) * t^i * (1 - t)^(n-i), {i, 0, n}] (*Polinomios de Berstein*)
Print["Polinomios de Berstein"]
Do[Print[B_i[t]], {i, 0, n}]
Print["Grafica de los polinomios de Berstein"]
g = Plot[B_0[t], {t, 0, 1}];
Do[g_i = Plot[B_i[t], {t, 0, 1}]; g = Show[g_i, g], {i, 1, n}]
```

```
Polinomios de Berstein
(1 - t)^2
2 (1 - t) t
t^2
Grafica de los polinomios de Berstein
```

Out[8]=



- **Superficie de Bezier**

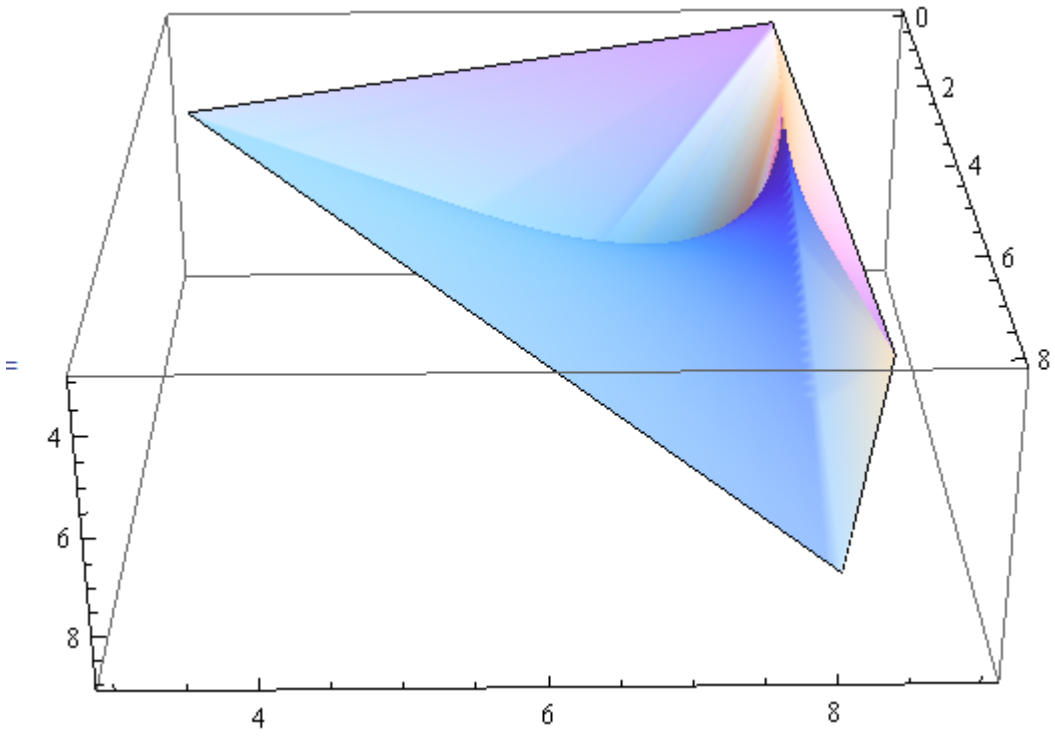
```
B[u_, v_] = Sum_{i=0}^{n} Sum_{j=0}^{n} B_i[u] * B_j[v] * P_{i,j} (*Superficies de Bezier sus coeficientes son los polinomios de Berstein*)
```

```
9]= (1 - u)^2 (1 - v)^2 P_{0,0} + 2 (1 - u)^2 (1 - v) v P_{0,1} + (1 - u)^2 v^2 P_{0,2} + 2 (1 - u) u (1 - v)^2 P_{1,0} +
  4 (1 - u) u (1 - v) v P_{1,1} + 2 (1 - u) u v^2 P_{1,2} + u^2 (1 - v)^2 P_{2,0} + 2 u^2 (1 - v) v P_{2,1} + u^2 v^2 P_{2,2}
```

- **Puntos de control**

```
Do[Do[P_{i,j} = {Random[Integer, {0, 10}], Random[Integer, {0, 10}],
    Random[Integer, {0, 10}]}, {j, 0, n}], {i, 0, n}] (* Se Genera puntos aleatorios en R^3*)
p = {};
Do[Do[p1 = {P_{i,j}}; p = Join[p1, p], {j, 0, n}], {i, 0, n}]
Print["Los puntos generado aleatoriamente son:"]
p
```

```
Los puntos generado aleatoriamente son:
{{9, 2, 9}, {8, 0, 3}, {3, 0, 5}, {8, 8, 7}}
```


=
0
2
4
6
8
4
6
8
4
6
8

Capítulo 4

REPRESENTACIÓN DE LOS B- SPLINE

4.1. B-SPLINE FUNCIONES BASE

Las curvas que consisten en solo una función polinomial son insuficientes. Algunos inconvenientes que tienen son:

Los altos grados son requeridos para satisfacer un gran número de restricciones; por ejemplo, $(n-1)$- grados son necesarios para hacer una curva Bézier que pasa por n puntos. Sin embargo, curvas de altos grados son ineficientes para procesar e inestable numéricamente.

Los altos grados requieres de una gran precisión para algunas superficies complejas.

Las curvas de un solo segmento no son muy adecuados para el diseño forma interactiva, aunque las curvas de Bézier se pueden formar a través de sus puntos de control, pero el control no es suficientemente local.

La solución para desventajas es una curva polinomial por trozos. En la siguiente imagen se muestra un ejemplo de una curva $C(u)$ que es construida con segmentos de polinomios de grado m $(m = 3)$.

$C(u)$es definida en $u \in [0, 1]$. Los valores de los parámetros $u_0 = 0 < u_1 < u_2 < u_3 = 1$ son llamados puntos de interrupción. Se asignan a los puntos finales de los tres segmentos. Nos indican los segmentos por $C_i(u), \quad 1 \leq i \leq m$.

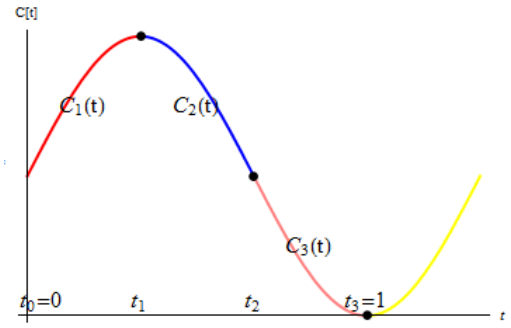


Figura 4.1: Curva que es construida con segmentos de polinomios de grado 3

Los segmentos están construidos de manera que se unen con un cierto nivel de continuidad (no necesariamente la misma en cada punto de interrupción). Sea C_i^j denotada como la j-ésima derivada de C_i, $C(u)$ se dice que es C^k continua en el punto de interrupción $u_i$si $C_i^j(u_i) = C_{i+1}^j(u_i) \qquad \forall 0 \leq j \leq k$.

Cualquiera de las formas polinómicas estándar se puede utilizar para representar $C_i(u)$.

P_i^j denota el i-ésimo punto de control del j-ésimo segmento.

Si el grado es igual a tres y los puntos de interrupción $U = \{u, u_1, u_2, u_3\}$ siendo fijo, y si permitimos que los doce puntos de control P_i^j, variar arbitrariamente, se obtiene el espacio vectorial V, compuesto por todas las curvas polinómicas a trozos cúbicos de U. V tiene dimensión doce, y una curva en V pueda ser discontinua en u_1 o u_2.

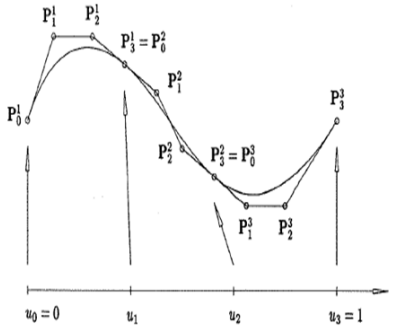


Figura 4.2: La figura muestra la curva de cúbica de Bézier.

Ahora supongamos que especificamos $P_3^1 = P_0^2$ y $P_3^2 = P_0^3$ como en se muestra en la figura de la siguiente. Esto da lugar a V^0, el espacio vectorial de todas las curvas polinómicas a trozos cúbicos en U que son al menos C^0 es decir continuas en todas partes, V^0 tiene dimensión diez, entonces $V^0 \subset V$.

La imposición de continuidad C^1 es un poco más complicada. Consideremos $u = u_1$. Supongamos que $P_3^1 = P_0^2$.

Sea

$$v = \frac{u - u_0}{u_1 - u_0}$$

y

$$w = \frac{u - u_1}{u_2 - u_1}$$

Los parámetros locales en los intervalos $[u_0 - u_1]$ y $[u_1 - u_2]$, respectivamente. Entonces $0 \leq v$, $w \leq 1$. C^1 continua en u_1 implica

$$\frac{1}{u_1 - u_0} C_1^1(v = 1) = C_1^1(u_1) = C_2^1(u_1) = \frac{1}{u_2 - u_1} C_2^1(w = 0)$$

Entonces

$$\frac{3}{u_1 - u_0}(P_3^1 - P_2^1) = \frac{3}{u_2 - u_1}(P_1^2 - P_0^2)$$

Así

$$P_3^1 = \frac{(u_2 - u_1)P_2^1 + (u_1 - u_0)P_1^2}{u_2 - u_0}$$

La ecuación anterior dice que P_3^1 y P_3^2 pueden ser escritos en términos de P_2^1 , P_1^2 y P_2^2, P_1^3, respectivamente. Por lo tanto, V^1, el espacio vectorial de toda C^1 curva polinómica continua cúbica a trozos en U, tiene dimensión ocho, y $V^1 \subset V^0 \subset V$.

Esto deja claro que el almacenamiento y la manipulación de los segmentos de polinomio individuales de una curva polinomial a trozos no es el método ideal para el manejo de tales curvas. En primer lugar, los datos redundantes se deben almacenar: doce coeficientes, donde sólo ocho se requiere para C^2 curvas cúbicas continuas y sólo seis de C^1 curvas cúbicas continuas. En segundo lugar, por la forma de Bézier la continuidad de $C(u)$ depende de las posiciones de los puntos de control, por lo tanto, hay poca flexibilidad en los puntos de control de posicionamiento, mientras que el mantenimiento de la continuidad. Si un diseñador quiere continuidad C^1 y está satisfecho con los segmentos $C_1(u)$ y $C_3(u)$, pero quiere modificar la forma de $C_2(u)$, esta sin suerte, ninguno de los puntos de control de $C_2(u)$ se puede modificar. En tercer lugar, la determinación de la continuidad de una curva requiere de cálculos**[1]**.

Queremos una representación de la curva de la forma

$$C(u) = \sum_{i=0}^{n} f_i(u) P_i \tag{4.1}$$

Donde P_i son los puntos de control, y $\{f(u), i = 0, \ldots, n\}$ son funciones polinómicas a trozos que forman una base para el espacio vectorial de todas las funciones polinómicas a trozos de grado deseado y continuidad (por una secuencia de puntos de parada $U = \{u\}$,$0 \leq i \leq m$. Tenga en cuenta que la continuidad está determinada por las funciones de base, por lo tanto, los puntos de control se pueden modificar sin alterar la continuidad de curvas. Además, la f_i debe tener las buenas propiedades "habituales" de análisis. Esto asegura que las curvas definidas por la ecuación (32) tienen propiedades agradables geométricamente similares a las curvas de Bézier, por ejemplo, envolvente convexa, variación decreciente, invariancia transformación. Otra propiedad importante que buscamos en nuestras funciones de base es el de apoyo local, lo que implica que cada $f_i(u)$ es distinto de cero sólo en un número limitado de subintervalos, no en todo el dominio $[u_0, u_m]$. Puesto que P_i se multiplica por $f_i(u)$, moviéndose P_i afecta forma de la curva sólo en los subintervalos donde $f_i(u)$ es distinto de cero.

Por último, teniendo en cuenta las funciones de base polinómicas a trozos adecuados, podemos construir curvas racionales a trozos

$$C^w(u) = \sum_{i=0}^{n} f_i(u) P_i^w$$

y superficies de productos tensoriales no racionales y racionales

$$S(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} f_i(u) g_j(v) P_{i,j}$$

$$S^w(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} f_i(u) g_j(v) P_{i,j}^w$$

4.2. DEFINICIÓN Y PROPIEDADES DE FUNCIONES BASE DE B-SPLINE

Hay un número de maneras de definir las funciones base B-Spline y para demostrar sus propiedades importantes, por ejemplo, por divisiones de funciones de potencia truncadas, y por una fórmula de recurrencia debido a la Boor, Cox y Mansfield. Usamos la fórmula de recurrencia, ya que es el más útil para la aplicación ordenador**[1]**.

Sea $U = \{u_0, \ldots, u_m\}$ una secuencia no decreciente de números reales. Es decir $u_i \leq u_{i+1}, \quad i = 0, \ldots, m-1$. Los u_i son llamados nudos, y U es el vector de nudos. La i-ésima función base B-Spline de grado p (orden $p+1$) denotada por $N_{i,p}(u)$ es definida como

$$N_{i,0}(u) = \begin{cases} 1 & u_i \leq u < u_{i+1} \\ 0 & c.o.f \end{cases} \tag{4.2}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \tag{4.3}$$

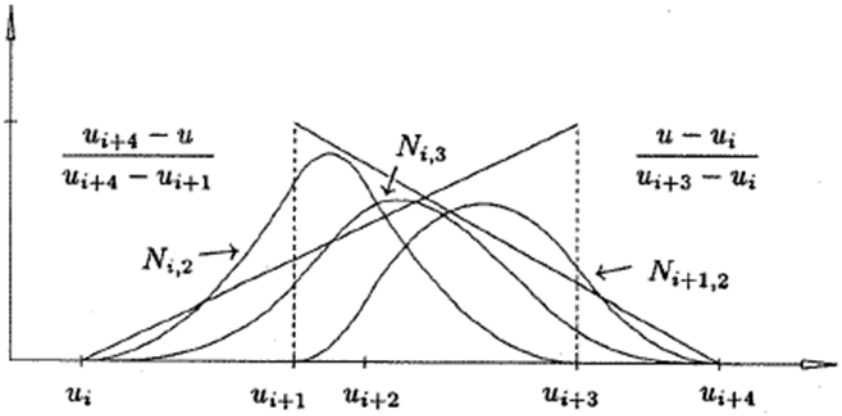


Figura 4.3: Recursividad de las funciones base de B-Spline

Notar que

- $N_{i,0}(u)$ es una función escalada, igual a cero en todas partes excepto en el intervalo semiabierto $u \in [u_i, u_{i+1})$
- Para $p > 0$, $N_{i,p}(u)$ es una combinación lineal de dos funciones base de grado $p-1$.
- El cálculo del conjunto de la función base requiere de la especificación del vector de nudos, U y del grado p.
- Los $N_{i,p}(u)$ son polinomios a trozos, definidos en toda la recta real; generalmente solo en el intervalo$[u_0, u_m]$.
- El intervalos semi-abierto $[u_i, u_{i+1})$, es llamado el i-ésimo nudo; este puede tener longitud cero, los nudos no necesitan ser distintos.
- El cálculo de las funciones de grado p genera una tabla triangular truncada

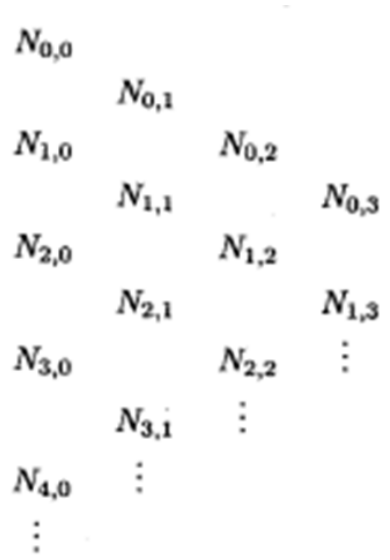


Figura 4.4:

Por razones de brevedad escribimos a menudo $N_{i,p}$ en lugar de $N_{i,p}(u)$.

2.2.1 Ejemplo.

Sea $U = \{u_0 = 0, u_1 = 0, u_2 = 0, u_3 = 1, u_4 = 1, u_5 = 1\}$ y $p = 2$. Ahora calculamos las funciones base B-Spline de grado 0,1 y 2.

$$N_{0,0} = N_{1,0} = 0 \qquad -\infty < u < \infty$$

$$N_{2,0}(u) = \begin{cases} 1 & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{3,0} = N_{4,0} = 0$$

$$N_{0,1} = \frac{u-0}{0-0}N_{0,0} + \frac{0-u}{0-0}N_{1,0} = 0 \qquad -\infty < u < \infty$$

$$N_{1,1} = \frac{u-0}{0-0}N_{1,0} + \frac{1-u}{1-0}N_{2,0} = \begin{cases} 1-u & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{2,1} = \frac{u-0}{1-0}N_{2,0} + \frac{1-u}{1-1}N_{3,0} = \begin{cases} u & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{3,1} = \frac{u-1}{1-1}N_{3,0} + \frac{1-u}{1-1}N_{4,0} = 0 \qquad -\infty < u < \infty$$

$$N_{0,2} = \frac{u-0}{0-0}N_{1,0} + \frac{1-u}{1-1}N_{1,1} = \begin{cases} (1-u)^2 & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{1,2} = \frac{u-0}{1-0}N_{1,1} + \frac{1-u}{1-0}N_{2,1} = \begin{cases} 2u(1-u) & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{2,2} = \frac{u-0}{1-0}N_{2,1} + \frac{1-u}{1-1}N_{3,1} = \begin{cases} u^2 & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

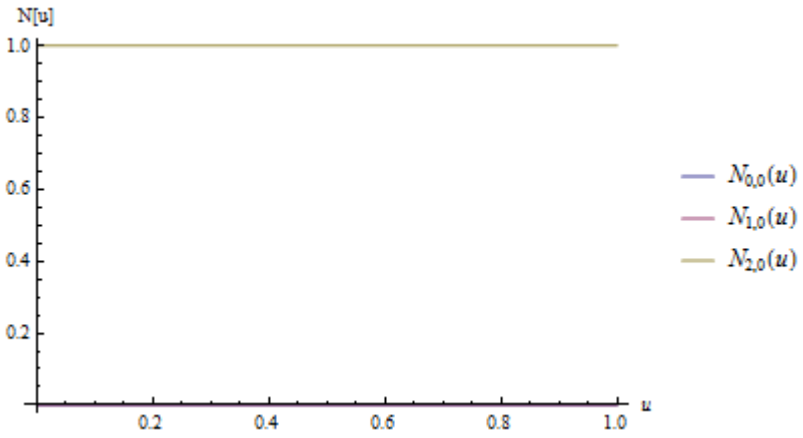


Figura 4.5: Funciones base B-Spline de grado 0.

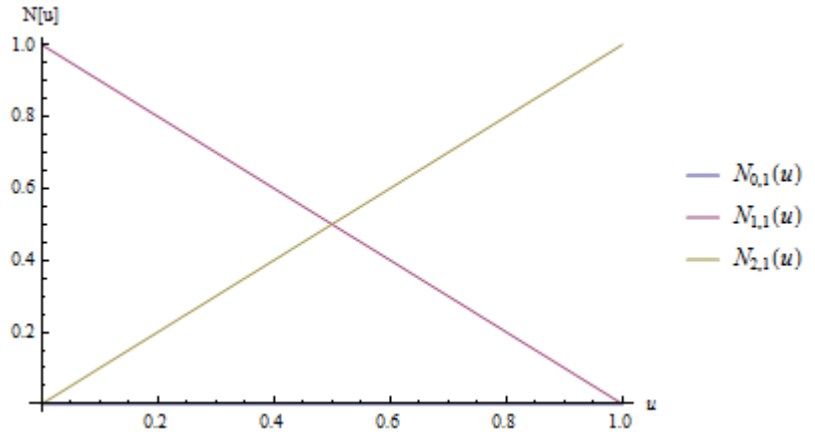


Figura 4.6: Funciones base B-Spline de grado1.

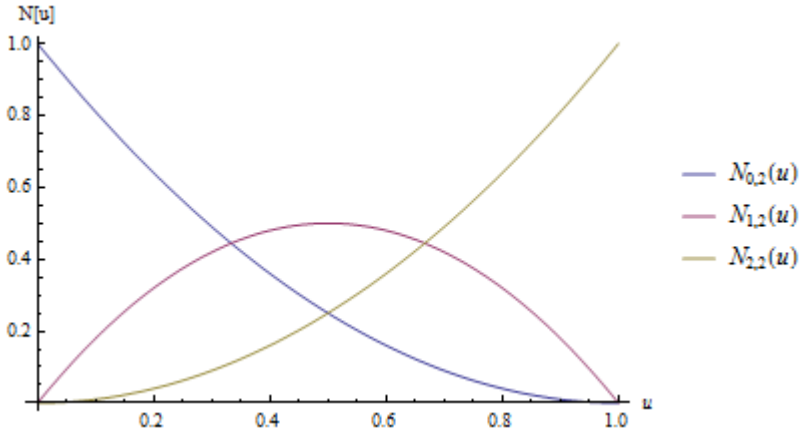


Figura 4.7: Funciones base B-Spline de grado 3

Gráficamente podemos visualizar las funciones base B-Spline de la siguiente manera

Notar que $N_{i,2}$, restringidos al intervalo $u \in [0,1]$, son los polinomios de Bernstein cuadráticos. Por esta razón la representación de los B-Spline con un vector nudo de la forma

$$U = \{\underbrace{0\ldots,0}_{p+1},\underbrace{1\ldots,1}_{p+1}\} \tag{4.4}$$

Es la generalización de la representación Bézier.

2.2.2. Ejemplo.

Sea $U = \{u_0 = 0, u_1 = 0, u_2 = 0, u_3 = 1, u_4 = 2, u_5 = 3, u_6 = 4, u_7 = 5, u_8 = 5, u_9 = 5\}$ y $p = 2$. Ahora calculamos las funciones base B-Spline de grado 0,1 y 2.

$$N_0, 0 = N_1, 0 = N_7, 0 = N_8, 0 = 0 \qquad -\infty < u < \infty$$

$$N_{2,0}(u) = \begin{cases} 1 & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{3,0}(u) = \begin{cases} 1 & 1 \leq u < 2 \\ 0 & c.o.f \end{cases}$$

$$N_{4,0}(u) = \begin{cases} 1 & 2 \leq u < 3 \\ 0 & c.o.f \end{cases}$$

$$N_{5,0}(u) = \begin{cases} 1 & 3 \leq u < 4 \\ 0 & c.o.f \end{cases}$$

$$N_{6,0}(u) = \begin{cases} 1 & 4 \leq u < 5 \\ 0 & c.o.f \end{cases}$$

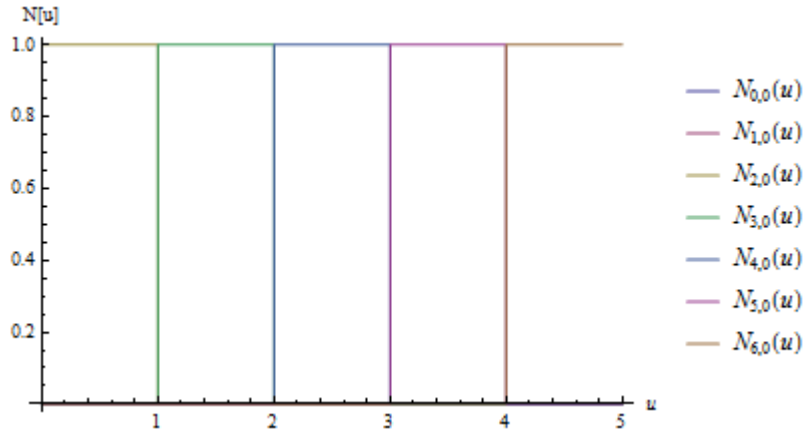


Figure 4.8: Funciones base B-Spline de grado 0

Las funciones base de grado 0 se representan:

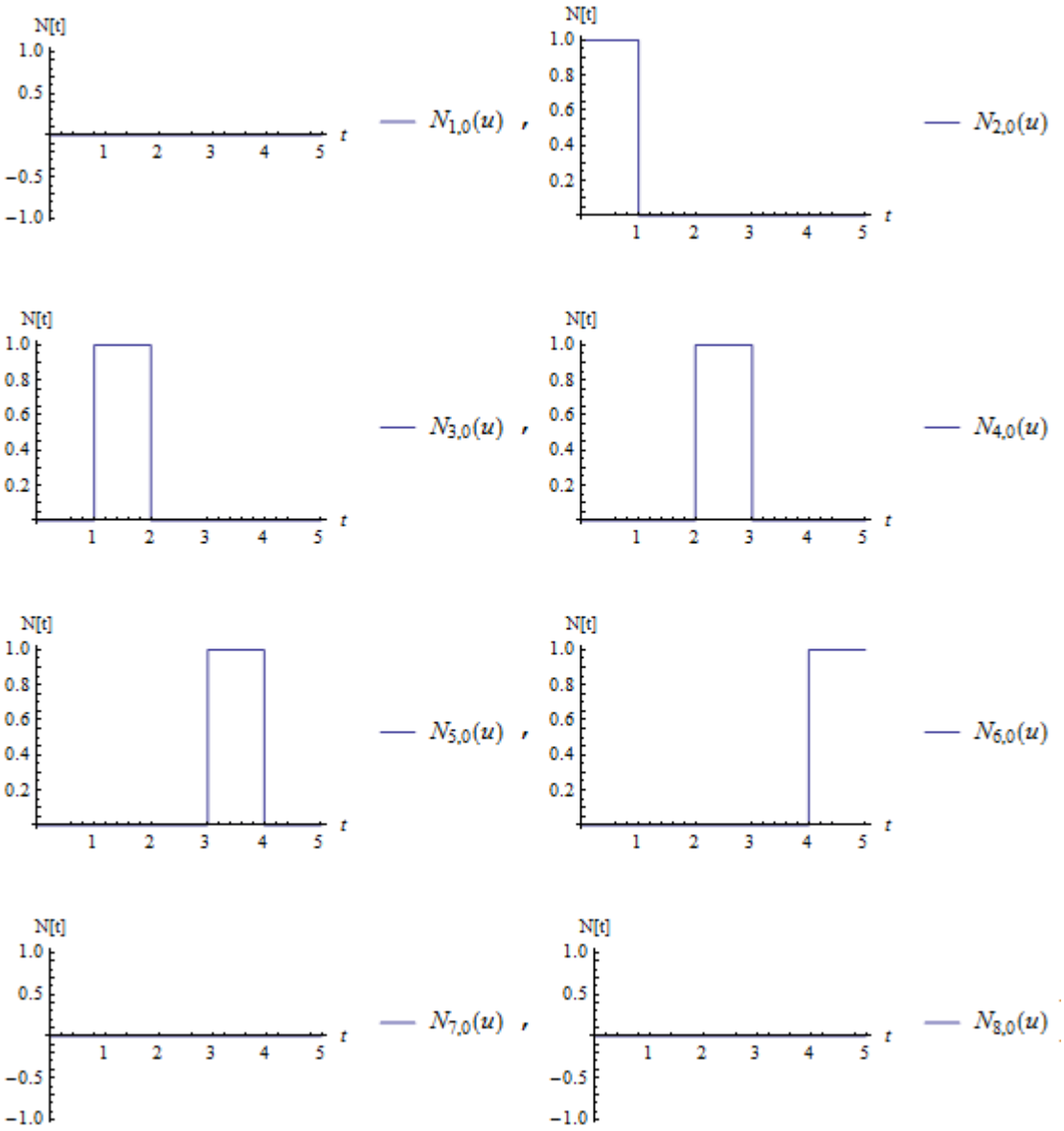


Figure 4.9: Funciones base de grado cero

Para observar mejor el comportamiento de las funciones base de grado cero se muestran en la figura 4.9 de manera individual.

$$N_{0,1} = \frac{u-0}{0-0}N_{0,0} + \frac{0-u}{0-0}N_{1,0} = 0 \qquad -\infty < u < \infty$$

$$N_{1,1} = \frac{u-0}{0-0}N_{1,0} + \frac{1-u}{1-0}N_{2,0} = \begin{cases} 1-u & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

$$N_{2,1} = \frac{u-0}{1-0}N_{2,0} + \frac{2-u}{2-1}N_{3,0} = \begin{cases} u & 0 \leq u < 1 \\ 2-u & 1 \leq u < 2 \\ 0 & c.o.f \end{cases}$$

$$N_{3,1} = \frac{u-1}{2-1}N_{3,0} + \frac{3-u}{3-2}N_{4,0} = \begin{cases} u-1 & 1 \leq u < 2 \\ 3-u & 2 \leq u < 3 \\ 0 & c.o.f \end{cases}$$

$$N_{4,1} = \frac{u-2}{3-2}N_{4,0} + \frac{3-u}{3-2}N_{5,0} = \begin{cases} u-2 & 2 \leq u < 3 \\ 4-u & 3 \leq u < 4 \\ 0 & c.o.f \end{cases}$$

$$N_{5,1} = \frac{u-3}{4-3}N_{5,0} + \frac{4-u}{4-3}N_{6,0} = \begin{cases} u-3 & 3 \leq u < 4 \\ 5-u & 4 \leq u < 5 \\ 0 & c.o.f \end{cases}$$

$$N_{6,1} = \frac{u-4}{5-4}N_{6,0} + \frac{5-u}{5-5}N_{7,0} = \begin{cases} u-4 & 0 \leq u < 1 \\ 0 & c.o.f \end{cases}$$

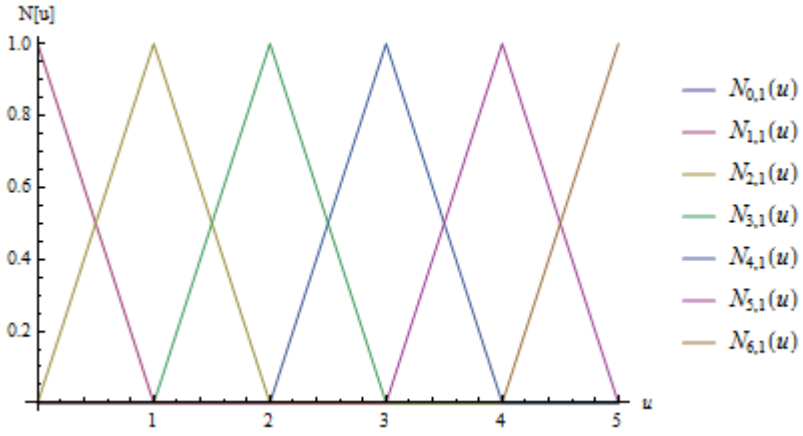


Figure 4.10: Funcines base B-Spline de grado 1

$$N_{0,2} = \frac{u-0}{0-0}N_{0,1} + \frac{1-u}{1-0}N_{1,1} = (1-u)^2 \qquad -\infty < u < \infty$$

$$N_{1,2} = \frac{u-0}{1-0}N_{1,2} + \frac{2-u}{2-0}N_{2,2} = \begin{cases} 2u - \frac{3}{2}u^2 & 0 \leq u < 1 \\ \frac{1}{2}(2-u)^2 & 1 \leq u < 2 \end{cases}$$

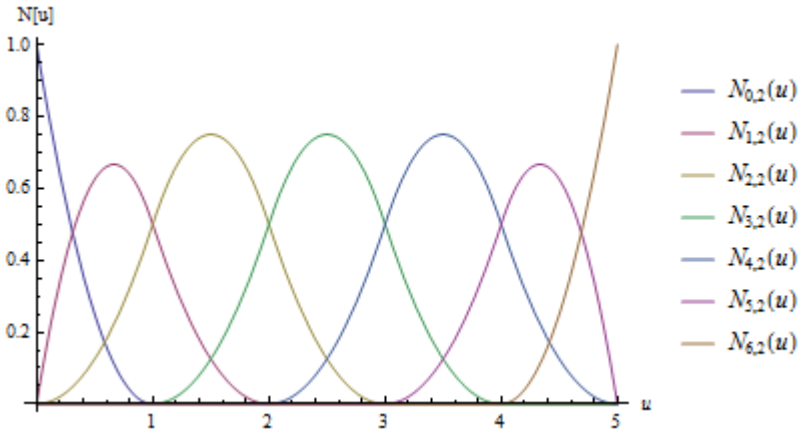


Figure 4.11: Funcines base B-Spline de grado 2

Continuando recursivamente con los cálculos, podemos observar las funciones base de grado 2:

Ahora enumeramos una serie de unas importantes propiedades de las funciones de base B-Spline. Supongamos de grado p y un vector de nudo $U = \{u_0, \ldots, u_m\}$

- $N_{i,p}(u) = 0$ si u esta fuera del intervalo $[u_i, u_{i+p+1})$. Esto es ilustrado por el esquema triangular de la figura 4.12 que se muestra. Notar que $N_{1,3}$ es combinación de $N_{1,0}$ $N_{2,0}$,$N_{3,0}$ y $N_{4,0}$. Por tanto $N_{1,3}$ es diferente de cero solo para $u \in [u_1, u_5)$.

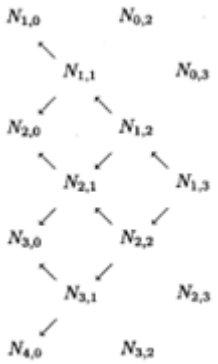


Figure 4.12:

- En cualquier paso de un nudo dado $[u_j, u_{j+1})$, a lo sumo $p+1$ de $N_{i,p}$ son diferente de cero, es decir las funciones $N_{j-p,p}, \ldots, N_{j,p}$ no son cero. En $[u_3, u_4)$ solo la función de grado cero que es diferentes de cero es $N_{3,0}$.

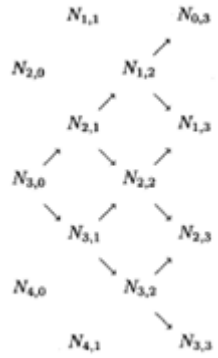


Figure 4.13:

- $N_{i,p}(u) \geq 0 \quad \forall i, p, u.$

Esta prueba es por inducción en p es claro que es verdad para $p = 0$; Asumimos que se cumple para $p-1$, $p \geq 0$ con i y u arbitrarios. Por definición

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \qquad (4.5)$$

Y por la primera propiedad enunciada anteriormente $N_{i,p-1}(u) = 0$ si $u \notin [u_i, u_{i+p})$. Pero $u \in [u_i, u_{i+p})$. Implica que

$$\frac{u - u_i}{u_{i+p} - u_i}$$

Es no negativo. Por asumir que $N_{i,p-1}(u)$ es no negativo, y entonces el primer término de la ecuación (4.5) es no negativo. Lo mismo se cumple para el segundo término, y por lo tanto $N_{i,p}(u)$ es no negativo.

Para cualquier paso de un nudo dado $[u_j, u_{j+1})$,

$$\sum_{j=i-p}^{i} N_{j,p}(u) = 1 \qquad \forall u \in [u_i, u_{i+1}) \tag{4.6}$$

Para probar esto, consideramos

$$\sum_{j=i-p}^{i} N_{j,p}(u) = \sum_{j=i-p}^{i} \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \sum_{j=i-p}^{i} \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

Cambiamos en la sumatoria la variable de la segunda suma de $i - p$ a $i - p + 1$ y considerar que $N_{i-p,p-1}(u) = N_{i+1,p-1}(u) = 0$, tenemos que

$$\sum_{j=i-p}^{i} N_{j,p}(u) = \sum_{j=i-p+1}^{i} \left[\frac{u - u_j}{u_{j+p} - u_j} + \frac{u_{j+p+} - u}{u_{j+p} - u_{i+1}} \right] N_{j,p-1}(u) + \sum_{j=i-p+1}^{i} N_{j,p-1}(u)$$

Aplicando el mismo concepto de recursividad

$$\sum_{j=i-p}^{i} N_{j,p}(u) = \sum_{j=i-p+1}^{i} N_{j,p-1}(u) = \sum_{j=i-p+2}^{i} N_{j,p-2}(u) = \sum_{j=i}^{i} N_{j,0}(u) = 1$$

Todas las derivadas de $N_{i,p}(u)$ existen en el interior de un lapso de nudo (donde es un polinomio). En el nudo $N_{i,p}(u)$ es $p-k$ veces continuamente diferenciable, donde k es la multiplicidad del nudo. Por lo tanto, el aumento del grado aumenta la continuidad, y el aumento de la multiplicidad nudo disminuye la continuidad.

Excepto para el caso $p = 0$, $N_{i,p}(u)$ obtiene exactamente un valor máximo.

4.3. DEFINICIÓN Y PROPIEDADES DE UNA CURVA B-SPLINE

Una curva B-Spline de grado p es definida por:

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \qquad a \leq u \leq b$$

Donde $\{P_i\}$ son los puntos de control, y las $\{N_{i.p}\}(u)$ son las funciones base de B-Spline de grado p definidas en el vector nudo no periódico y uniforme:

$$U = \{\underbrace{a \ldots, a}_{p+1}, u_{p+1}, \ldots, u_{m-p-1}, \underbrace{b \ldots, b}_{p+1}\} \tag{4.7}$$

A menos que se indique lo contrario, asumimos que $a = 1$ y $b = 1$. Los polígonos formados por los $\{P_i\}$ son llamados polígonos de control**[1]**.

Se requieren tres pasos para calcular un punto en una curva B-Spline como valor fijo u.

- Encontrar el lapso nudo en la que se encuentra u
- Calcular las funciones de base distintos de cero
- Multiplicar los valores de las funciones bases distintos de cero con los puntos de control correspondientes

Considere $U = \{0, 0, 0, 1, 2, 3, 4, 5, 5, 5\} u = \frac{5}{2}$ y $p = 2$. Entonces $u \in [u_4, u_5)$ y

$$N_{2,2}\left(\frac{5}{2}\right) = \frac{1}{8} \qquad N_{3,2}\left(\frac{5}{2}\right) = \frac{6}{8} \qquad N_{4,2}\left(\frac{5}{2}\right) = \frac{1}{8}$$

Multiplicando los puntos de control

$$C\left(\frac{5}{2}\right) = \frac{1}{8}P_2 + \frac{6}{8}P_3 + \frac{1}{8}P_4$$

Sea

$$C(u) = \sum_{i=0}^{n} N_{i,p}(u) P_i \qquad a \leq u \leq b$$

Tenemos las siguientes propiedades:

- Si $n = p$ y $U = \{0, \ldots, 0, 1, \ldots, 1\}$ entonces $C(u)$ es una curva Bézier.
- $C(u)$ es una curva polinomial por pedazos de grado p , número de puntos de control $n+1$, y número de nodos $m+1$, son relacionados por $m = n+p+1$
- $C(0) = P_0$ y $C(1) = P_n$
- Invariancia Afín: una transformación afine se aplica a la curva mediante la aplicación a los puntos de control. Sea r un punto en ε^3(espacio euclidiano tridimensional). Una transformación afín, denotado por Φ, mapas de ε^3 en ε^3 y tiene la forma
- $\Phi(r) = Ar + v$

Donde A es una matriz 3×3 y v es un vector. La transformación afín incluye traslaciones, rotaciones, escalas y cortes. La propiedad invariante afín para las curvas B-Spline se desprende de la partición de la unidad de la propiedad $N_{i.p}(u)$. En consecuencia, sea $r = \sum \alpha_i p_i$ donde $p_i \in \varepsilon^3$ y $\sum \alpha_i = 1$. Entonces

$$\Phi(r) = \Phi(\sum \alpha_i p_i) = A(\sum \alpha_i p_i) + v = \sum \alpha_i A p_i + \sum \alpha_i v = \sum \alpha_i (A p_i + v) = \sum \alpha_i \Phi(p_i)$$

- Fuerte propiedad envolvente convexa: la curva es una figura en la envolvente convexa de su polígono de control, de hecho, si $u \in (u_i, u_{i+1}]$, $p \leq i < m - p - 1$, entonces $C(u)$ es la envolvente convexa de los puntos de control $P_{i-p}, \ldots, P_i$, como se muestra en la siguiente figura. Así se desprende de la no negatividad y la partición de la unidad de las propiedades de $N_{i.p}(u)$ y de las propiedades de $N_{j,p}(u) = 0$ para $j < i - p$ y $j > i$ cuando $u \in (u_i, u_{i+1}]$.

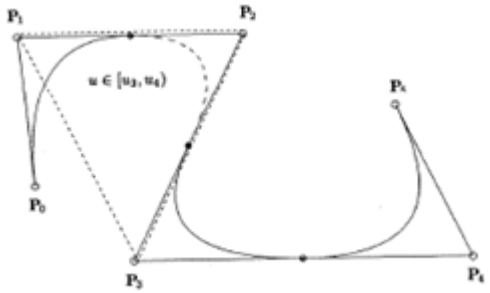


Figura 4.14: Propiedad envolvente convexa

- Sistema de modificación: moviendo P_i cambia $C(u)$ solo en el intervalo $[u_i, u_{i+p+1})$

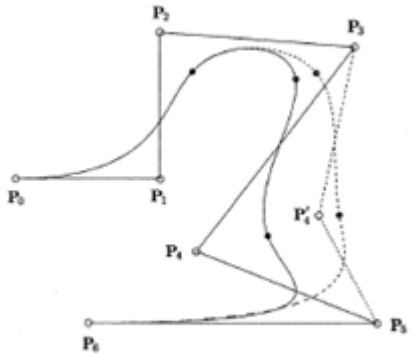


Figura 4.15: La curva solo es modificada en un intervalo

- El polígono de control representa una aproximación lineal a trozos a la curva, su aproximación se mejora mediante la inserción nudo o grados de elevación. Es una regla general, cuanto más bajo sea el nivel, más cerca de una curva B-Spline sigue su polígono de control.
- Mudanza a lo largo de la forma de la curva $u = 0$ a $u = 1$, las funciones $N(u)$ actúan como interruptores, como u se mueve más allá de un nudo, una $N(u)$ (y por tanto la correspondiente P_i) se apaga, y el siguiente en los interruptores.

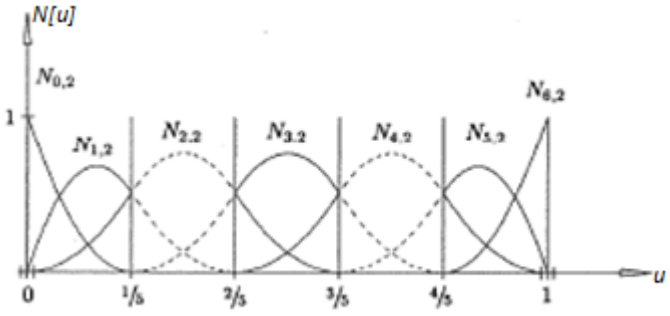


Figura 4.16: Funciones base de B-spline

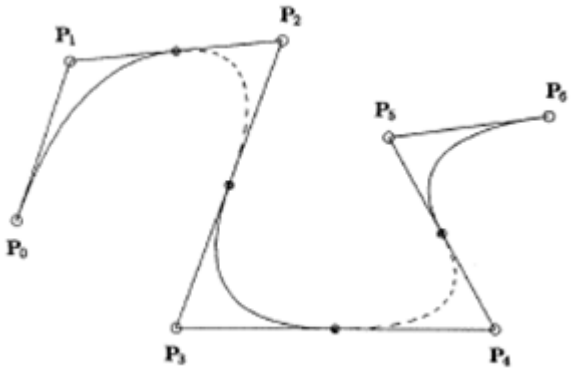


Figura 4.17:

- Variación decreciente propiedad: ningún plano tiene más intersecciones con la curva que con el polígono de control (sustituir el plano con la línea, para las curvas de dos dimensiones)
- La continuidad y diferenciabilidad de $C(u)$ se derivan de la de la $N_{i,p}(u)$ (ya que $C(u)$ es una combinación lineal de $N_{i,p}(u)$). Por lo tanto, $C(u)$ es infinitamente diferenciable en el interior de los intervalos de nudo, y es por lo menos veces $p-k$ continuamente diferenciable en nudo de multiplicidad k.

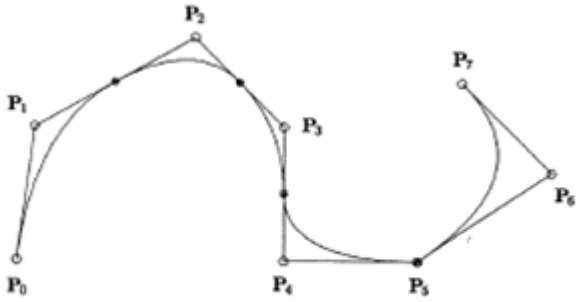


Figura 4.18: Cúbica con ocho puntos de control

4.4. ENVOLVENTE CONVEXA FUERTE

Cada punto de $C(t)$ pertenece a la envolvente convexa de $k+1$ vértices de control. De hecho, si $u_0 \in [x_i, x_{i+1}]$, entonces $C(u_0)$ se construye utilizando

las funciones de base $N_{i-k.k}(u), \ldots, N_{i.k}(u)$ son las únicas con soporte en el intervalo (x_i, x_{i+1}):

$$C(u_0) = \sum_{j=i-k}^{i} N_{j,k}(u) P_j \qquad x_i \leq u_0 \leq x_{i+1}$$

Por tanto, $C(u_0)$ está contenido en la envolvente convexa de los vértices de control $P_{i-k}, \ldots, P_i$.

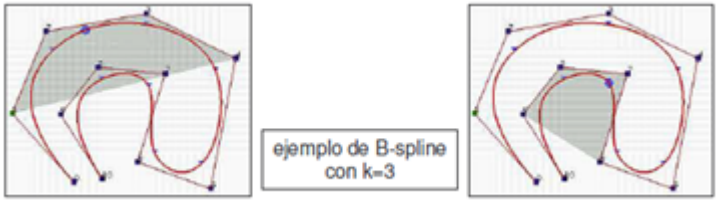


Figura 4.19:

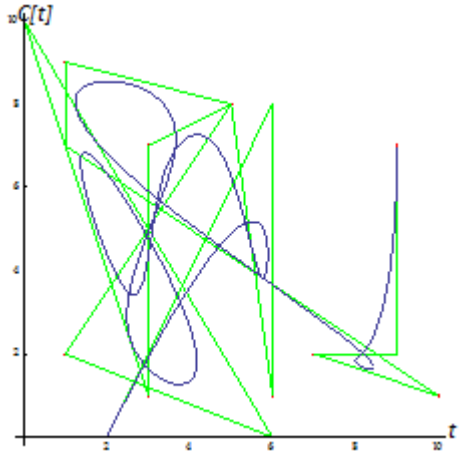


Figura 4.20: Ejemplo de una curva B-spline

En consecuencia, toda la curva $C(t)$ está contenida en la unión de las envolventes convexas de $k+1$ vértices de control consecutivos.

EJEMPLO DE CURVA B-SPLINE

Grado de la curva es 3

El vector nudo {0 ,0 ,0 ,0 ,1/2,1 ,1 ,1 ,1}

Puntos de control son 5 : {4 ,6}{2 ,9}{1 ,5}{5 ,8}{3 ,7}

Las funciones $N_{i,j}$ son las siguientes

Funciones $N_{i,j}$

$$N_{0,0}(u) = 0$$

$$N_{1,0}(u) = 0$$

$$N_{2,0}(u) = 0$$

$$N_{3,0}(u) = \text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right]$$

$$N_{4,0}(u) = \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]$$

$$N_{0,1}(u) = 0$$

$$N_{1,1}(u) = 0$$

$$N_{2,1}(u) = 2\left(\frac{1}{2} - u\right) \text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right]$$

$$N_{3,1}(u) = 2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]$$

$$N_{4,1}(u) = 2\left(-\frac{1}{2} + u\right) \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]$$

$$N_{0,2}(u) = 0$$

$$N_{1,2}(u) = 4\left(\frac{1}{2} - u\right)^2 \text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right]$$

$$N_{2,2}(u) = 4\left(\frac{1}{2} - u\right) u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + (1-u)\left(2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]\right)$$

$$N_{3,2}(u) = 4(1-u)\left(-\frac{1}{2} + u\right) \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right] + u\left(2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]\right)$$

$$N_{4,2}(u) = 4\left(-\frac{1}{2} + u\right)^2 \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]$$

$$N_{0,3}(u) = 8\left(\frac{1}{2} - u\right)^3 \text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right]$$

$$N_{1,3}(u) = 8\left(\frac{1}{2} - u\right)^2 u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + (1-u)\left(4\left(\frac{1}{2} - u\right) u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + (1-u)\left(2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]\right)\right)$$

$$N_{2,3}(u) = u\left(4\left(\frac{1}{2} - u\right) u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + (1-u)\left(2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]\right)\right) + (1-u)\left(4(1-u)\left(-\frac{1}{2} + u\right) \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right] + u\left(2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]\right)\right)$$

$$N_{3,3}(u) = 8(1-u)\left(-\frac{1}{2} + u\right)^2 \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right] + u\left(4(1-u)\left(-\frac{1}{2} + u\right) \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right] + u\left(2u\,\text{If}\left[0 \le u < \frac{1}{2}, 1, 0\right] + 2(1-u)\,\text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]\right)\right)$$

$$N_{4,3}(u) = 8\left(-\frac{1}{2} + u\right)^3 \text{If}\left[\frac{1}{2} \le u < 1, 1, 0\right]$$

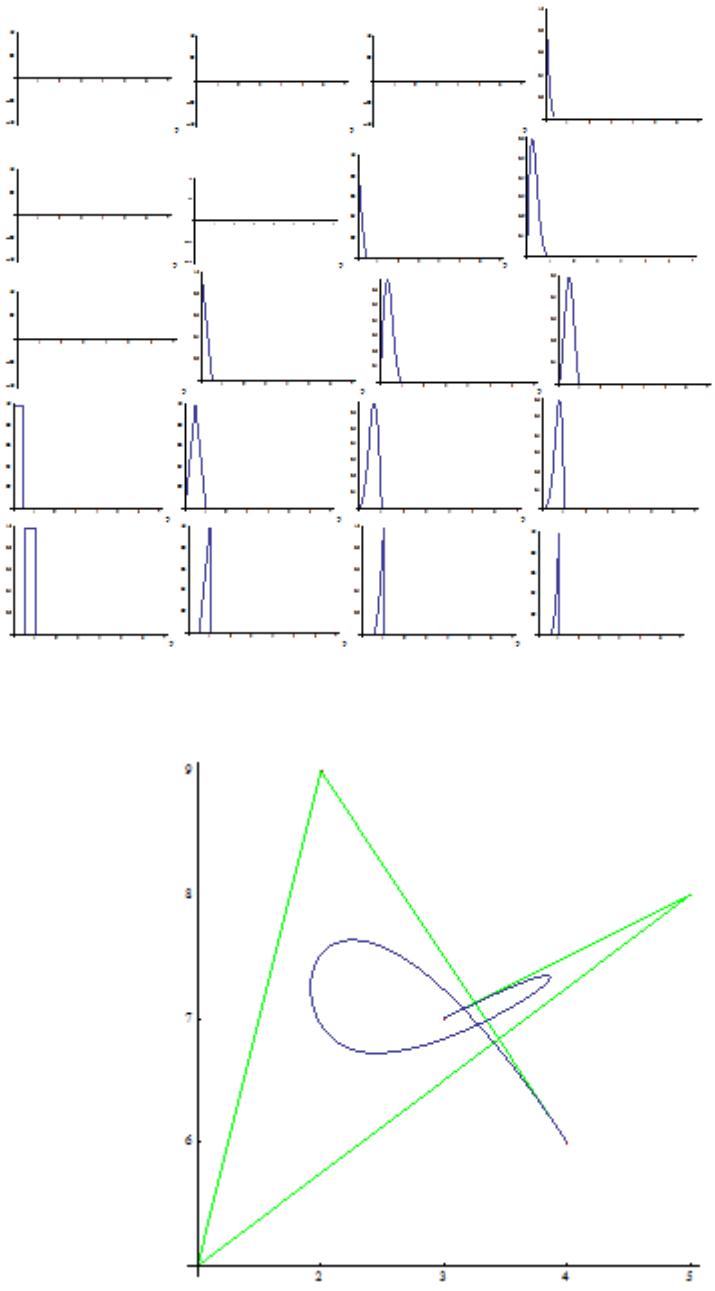


Figura 4.21: Curva B-spline con puntos de control {4 ,6}{2 ,9}{1 ,5}{5 ,8}{3 ,7} y de grado 3

La curva se muestra en la figura 4.21.

$$C(u) =$$
$$\{32(\frac{1}{2}-u)^3 If[0 \le u < \frac{1}{2}, 1, 0] + 24(-\frac{1}{2}+u)^3 If[\frac{1}{2} \le u < 1, 1, 0] + u(4(\frac{1}{2}-u)u If[0 \le u < \frac{1}{2}, 1, 0] + (1-u)(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0])) + (1-u)(4(1-u)(-\frac{1}{2}+u) If[\frac{1}{2} \le u < 1, 1, 0] + u(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0])) + 2(8(\frac{1}{2}-u)^2 u If[0 \le u < \frac{1}{2}, 1, 0] + (1-u)(4(\frac{1}{2}-u)u If[0 \le u < \frac{1}{2}, 1, 0] + (1-u)(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0]))) + 5(8(1-u)(-\frac{1}{2}+u)^2 If[\frac{1}{2} \le u < 1, 1, 0] + u(4(1-u)(-\frac{1}{2}+u) If[\frac{1}{2} \le u < 1, 1, 0] + u(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0]))), 48(\frac{1}{2}-u)^3 If[0 \le u < \frac{1}{2}, 1, 0] + 56(-\frac{1}{2}+u)^3 If[\frac{1}{2} \le u < 1, 1, 0] + 9(8(\frac{1}{2}-u)^2 u If[0 \le u < \frac{1}{2}, 1, 0] + (1-u)(4(\frac{1}{2}-u)u If[0 \le u < \frac{1}{2}, 1, 0] + (1-u)(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0]))) + 5(u(4(\frac{1}{2}-u)u If[0 \le u < \frac{1}{2}, 1, 0] + (1-u)(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0])) + (1-u)(4(1-u)(-\frac{1}{2}+u) If[\frac{1}{2} \le u < 1, 1, 0] + u(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0]))) + 8(8(1-u)(-\frac{1}{2}+u)^2 If[\frac{1}{2} \le u < 1, 1, 0] + u(4(1-u)(-\frac{1}{2}+u) If[\frac{1}{2} \le u < 1, 1, 0] + u(2u If[0 \le u < \frac{1}{2}, 1, 0] + 2(1-u) If[\frac{1}{2} \le u < 1, 1, 0])))\}$$

4.5. DEFINICIÓN Y PROPIEDADES DE SUPERFICIES B-SPLINE

Una superficie B-Spline es obtenida al tomar una red bidireccional de puntos de control, dos vectores nudo, y los productos de las funciones B-Spline univariadas

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,q}(v) P_{i,j} \tag{4.8}$$

Con

$$U = \{\underbrace{0\ldots,0}_{p+1}, u_{p+1}, \ldots, u_{r-p-1}, \underbrace{1\ldots,1}_{p+1}\}$$

$$V = \{\underbrace{0\ldots,0}_{q+1}, v_{q+1}, \ldots, v_{s-q-1}, \underbrace{1\ldots,1}_{q+1}\}$$

U tiene $r+1$ nudos, y V tiene $s+1$.

$$r = n + p + 1$$

y

$$s = m + q + 1$$

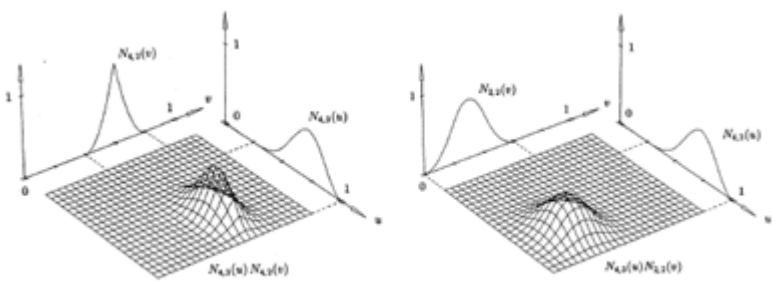


Figura 4.22: Producto tensorial de las funciones base

Sea $U = \{0, 0, 0, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1, 1, 1, 1\}$ el vector nudo y $\{N_{i,3}(u)\}$ funciones bases cúbicas, $\{N_{j,2}\}$ las funciones base cuadráticas en $V = \{0, 0, 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1, 1, 1\}$. En la figura de abajo se muestra el producto tensorial de las funciones base $N_{4,3}(u)\ N_{4,2}(u)$ y $N_{4,3}(u)\ N_{2,2}(u)$ respectivamente.

Se requieren cinco pasos para calcular un punto de la superficie B-Spline en fijo (u, v) valores de los parámetros:

1. Encontrar el vector nudo donde u se encuentra, es decir $u \in [u_i, u_{i+1})$
2. Calcular las funciones de base distintas de cero $N_{i-p,p}(u), ..., N_{i,p}(u)$
3. Encontrar el vector nudo donde v se encuentra, es decir $v \in [v_i, v_{i+1})$
4. Calcular las funciones bases distintas de cero $N_{i-p,p}(v), ..., N_{i,p}(v)$
5. Multiplicar los valores de las funciones de base distintos de cero con los puntos de control correspondientes.

El último paso toma forma

$$S(u,v) = [N_{k,p}(u)]^T [P_{k,l}][N_{l,q}(v)]$$

$$i - p \leq k \leq i, \qquad j - q \leq l \leq j$$

Notar que $[N_{k,p}(u)]^T$ es un vector fila $l \times (p+1)$ de escalares, $[P_{k,l}]$ es una matriz de puntos de control de $(p+1) \times (q+1)$, y $[N_{l,q}(v)]$ es un vector columna de $(q+1) \times 1$ de escalares.

Ejemplo:

Sea $p = q$ y $\sum_{i=0}^{4} \sum_{j=0}^{5} N_{i,2}(u) N_{j,2}(v) P_{i,j}$

con

$$U = \{0, 0, 0, \frac{2}{5}, \frac{3}{5}, 1, 1, 1\}$$

$$V = \{0, 0, 0, \frac{1}{5}, \frac{1}{2}, \frac{4}{5}, 1, 1, 1\}$$

.

Calculamos $S(\frac{1}{5}, \frac{3}{5})$ entonces $\frac{1}{5} \in [v_4, v_5)$ y

$$S\left(\frac{1}{5}, \frac{3}{5}\right) = \left[N_{0,3}\left(\frac{1}{5}\right) \quad N_{1,2}\left(\frac{1}{5}\right) \quad N_{2,2}\left(\frac{1}{5}\right)\right] \times \begin{bmatrix} P_{0,2} & P_{0,3} & P_{0,4} \\ P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,2} & P_{2,3} & P_{2,4} \end{bmatrix} \times \begin{bmatrix} N_{2,2}\left(\frac{3}{5}\right) \\ N_{3,2}\left(\frac{3}{5}\right) \\ N_{4,2}\left(\frac{3}{5}\right) \end{bmatrix}$$

Las propiedades del producto tensorial de las funciones base son las siguientes, correspondientes propiedades de las funciones base univariadas **[1]**.

- No negatividad $N_{i,p}(u) N_{j,q}(v) \geq 0$ para toda i, j, p, q, u, v
- Partición de unidad $\sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,q}(v) = 1$ para toda $(u, v) \in [0,1] \times [0,1]$
- Si $n = p, \quad m = q$,$U = \{0, \ldots, 0, 1, \ldots 1\}$ y $V = \{0, \ldots, 0, 1, \ldots 1\}$ entonces $N_{i,p}(u) N_{j,q}(v) = B_{i,n}(u) B_{j,m}(v)$ para toda i, j , esto es, el producto de funciones b-spline degeneradas a productos de polinomios de Bernstein.
- $N_{i,p}(u) N_{j,q}(v) = 0$ si (u, v) esta fuera del rectángulo $[u_i, u_{i+p+1}) \times [v_j, v_{j+q+1})$
- En cualquier rectángulo dado, $[u_{i_0}, u_{i_{0+p+1}}) \times [v_{j_0}, v_{j_{0+q+1}})$, a lo mas $(p+1)(q+1)$ funciones base no son cero. En particular la $N_{i,p}(u) N_{j,q}(v)$ para $j_{0-q} \leq j \leq j_0$

- Si $p > 0$ y $q > 0$, entonces $N_{i,p}(u)N_{j,q}(v)$ obtiene un valor máximo.
- Interior de los rectángulos formados por las líneas de nudo u y v, donde la función es un polinomio de dos variables, existen todas las derivadas parciales de $N_{i,p}(u)N_{j,q}(v)$; en el nodo u (nodo v) es $p - k(q - k)$ veces diferenciable en dirección de $u(v)$, donde k es la multiplicidad del nudo. En superficies B-spline se tienen las siguientes propiedades:
- Si $n = p$, $m = q$,$U = \{0, ..., 0, 1, ..., 1\}$ y $V = \{0, ..., 0, 1, ..., 1\}$ entonces $S(u, v)$ es una superficie Bézier.
- La interpolación de superficie de cuatro esquinas como puntos de control $S(0, 0) = P_{0,0},\ S(1, 0) = P_{n,0},\ S(0, 1) = P_{0,m}$ y $S(1, 1) = P_{n,m}$, tenemos que el producto tensorial es 1.

$$N_{0,p}(0)N_{0,q}(0) = N_{n,p}(1)N_{0,q}(0) = N_{0,p}(0)N_{m,q}(1) = N_{n,p}(1)N_{m,q}(1) = 1$$

- Invariancia afín: una transformación afín es aplicada a la superficie por una aplicación de los puntos de control.
- Propiedad de envolvente convexa fuerte: si $(u, v) \in [u_{i_0}, u_{i_{0+1}}) \times [v_{j_0}, v_{j_{0+1}})$ entonces $S(u.v)$ en una envolvente convexa de los puntos de control $P_{i,j}$, $i_0 - p \leq i \leq i_0$ y $j_0 - q \leq j \leq j_0$ver la siguiente figura:

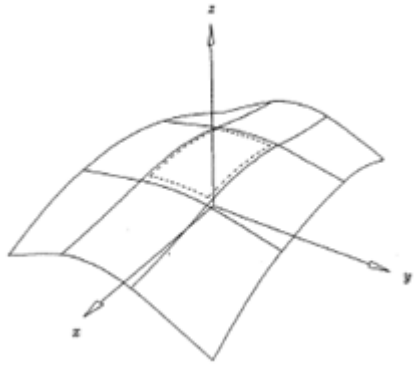


Figura 4.23: En esta imagen se muestra la envolvente convexa

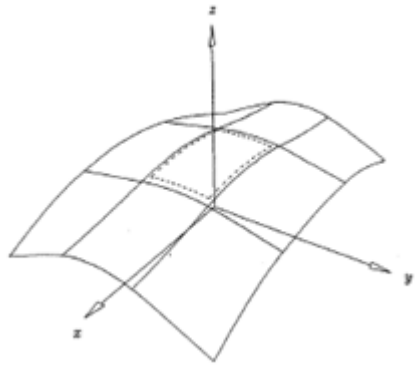


Figura 4.24: Se ilustran los puntos de control

- Si trianguladas, forma de la red de control de un plano de aproximación por tramos a la superficie; como es el caso de curvas, menor será el grado mejor será la aproximación.

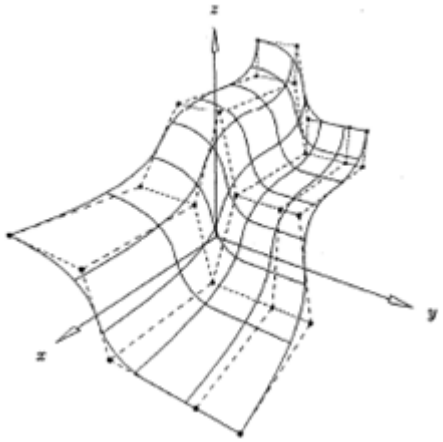


Figura 4.25: Superficie bicuadrada

Superficie bicuadrada, es decir $p = 2$,$q = 2$

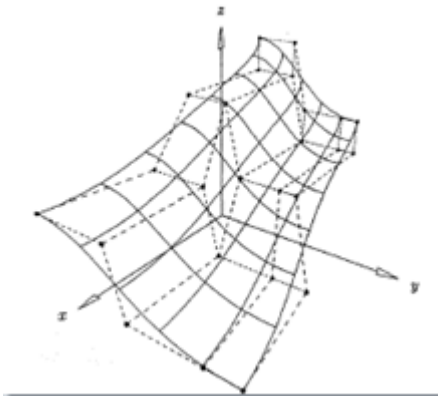


Figura 4.26: Superficie bicuarta

Superficie bicuarta, es decir $p = 4$, $q = 4$, con los mismos puntos de control que la imagen anterior

- Esquema de modificación Local: Si $P_{i,j}$ es movida esta afecta a la superficie sólo en el rectángulo $[u_i, u_{i+p+1}) \times [v_j, v_{j+q+1})$.

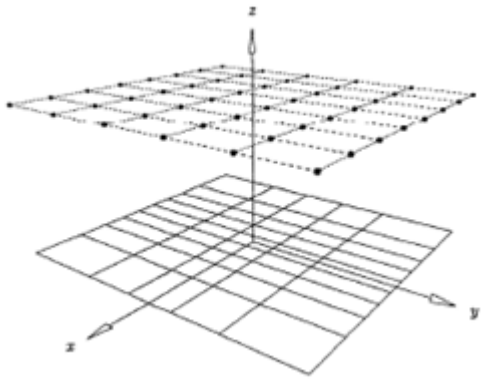


Figura 4.27: La superficie es plana porque todos los puntos de control se encuentran en un plano común; la red de control está desplazada de la superficie para una mejor visualización

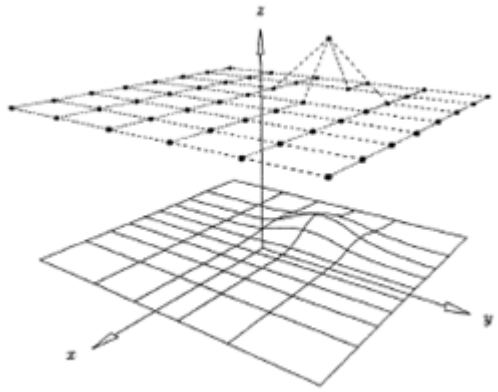


Figura 4.28: Ahora se mueve un punto de control y solo afecta la superficie en un rectángulo, que es en el que se encuentra el punto que se movió.

- La continuidad y diferenciabilidad de $S(u, v)$ sigue de las funciones base. En particular, $S(u, v)$ es $p - k(q - k)$ veces diferenciable en $u(v)$ en la dirección a $u(v)$ nudo de multiplicidad k. La Figura 4.29 muestra una superficie *cuadrática* $\times$ *cúbica* definida en los vectores de nudo $U = \{0, 0, 0, \frac{1}{2}, \frac{1}{2}, 1, 1, 1\}$ y $V = \{0, 0, 0, 0, \frac{1}{2}, 1, 1, 1, 1\}$. Observe el pliegue en la superficie, que corresponde a la línea del nudo $u = \frac{1}{2}$.

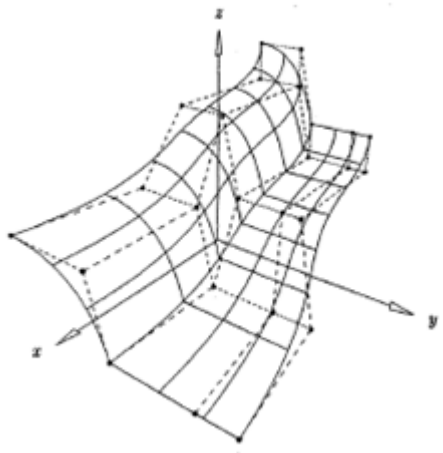


Figura 4.29: superficie cuadrática x cúbica

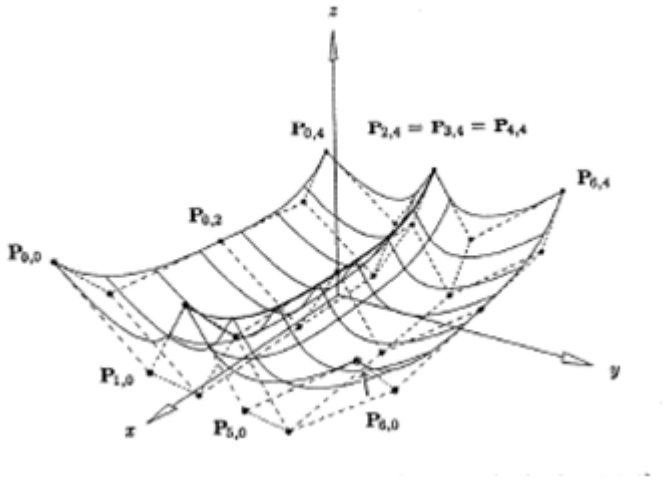


Figura 4.30: Superficie bicúbica

Por supuesto, como es el caso para las curvas, es posible colocar los puntos de control de tal manera que se cancelan las discontinuidades en las funciones base. Mediante el uso de puntos de control se multiplican coincidentes, discontinuidades visuales pueden ser creados cuando no hay discontinuidades correspondientes en las funciones de base. En la figura 4.30 se muestra una superficie de este tipo, que es bicúbica sin múltiples nudos. Por lo tanto, las segundas derivadas parciales están en todas partes continuas. El pliegue es debido a los múltiples puntos de control.

Destacamos aquí que no se conoce la variación decreciente propiedad para superficies B-Spline.

Isoparamétricas curvas en $S(u, v)$ se obtuvieron de una manera análoga a la de la superficie de Bézier. Fijamos $u = u_0$

$$C_{u_0}(v) = S(u_0, v)$$

$$= \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u_0) N_{j,q}(v) P_{i,j}$$

$$= \sum_{j=0}^{m} N_{j,q}(v) \sum_{i=0}^{n} N_{i,p}(u_0) P_{i,j}$$

$$= \sum_{j=0}^{m} N_{j,q}(v) Q_j(u_0)$$

Donde

$$Q_j(u_0) = \sum_{i=0}^{n} N_{i,p}(u_0) P_{i,j}$$

Análogamente

$$C_{v_0}(u) = \sum_{i=0}^{n} N_{i,p}(u) Q_i(v_0)$$

Donde

$$Q_i(v_0) = \sum_{j=0}^{m} N_{j,q}(v_0) P_{i,j}$$

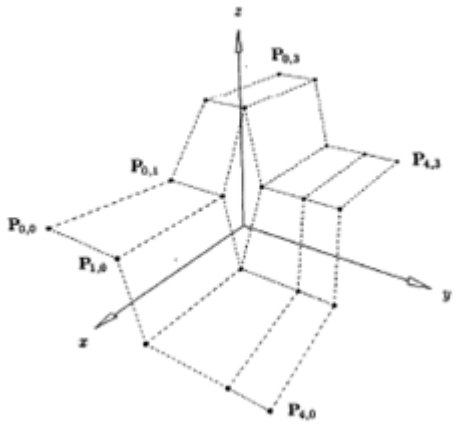


Figura 4.31: Malla de puntos de control

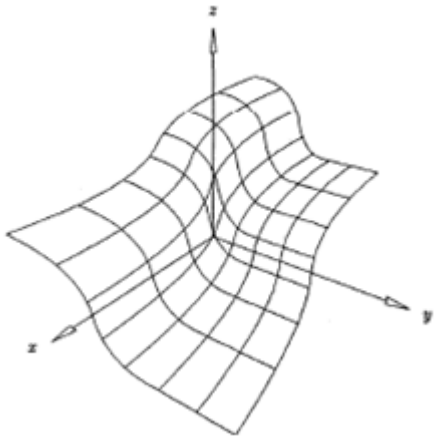


Figura 4.32: Superficie B-Spline

es un isocurva u en $S(u,v)$. $C_{u_0}(v)$ es una curva B-Spline de grado q-ésimo en V, y $C_{v_0}(u)$ es una curva B-Spline de grado p-ésimo en U. $S(u_0,v_0)$ es la en la intersección de $C_{v_0}(u)$ y $C_{v_0}(u)$. Todas las líneas que aparecen en la superficie de figura abajo, son isolíneas.

EJEMPLO DE UNA SUPERFICIE B-SPLINE

El vector nudo $\{0,0,0,0,1,1,1,1\}$

Puntos de control 16

$$\begin{array}{cccc} \{\{\{5,12,24\}, & \{5,19,22\}, & \{5,13,26\}, & \{2,12,20\}\}, \\ \{\{3,18,28\}, & \{1,16,24\}, & \{1,18,23\}, & \{0,12,22\}\}, \\ \{\{0,19,21\}, & \{0,14,20\}, & \{1,19,27\}, & \{5,16,28\}\}, \\ \{\{5,20,20\}, & \{4,12,24\}, & \{1,15,26\}, & \{1,15,26\}\}\} \end{array}$$

Las funciones $N_{i,j}$ son las siguientes

Funciones $N_{i,j}$

$N_{0,0}(u) = 0$

$N_{1,0}(u) = 0$

$N_{2,0}(u) = 0$

$N_{3,0}(u) = \text{If}[0 \leq u < 1, 1, 0]$

$N_{0,1}(u) = 0$

$N_{1,1}(u) = 0$

$N_{2,1}(u) = (1 - u)\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{3,1}(u) = u\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{0,2}(u) = 0$

$N_{1,2}(u) = (1 - u)^2\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{2,2}(u) = 2\ (1 - u)\ u\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{3,2}(u) = u^2\ \text{If}[0 \leq u < 1, 1, 0]$

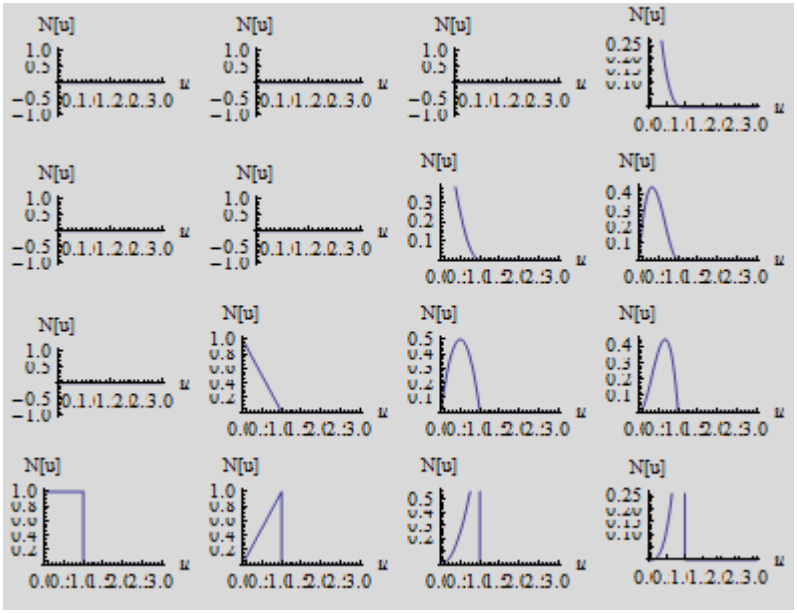
$N_{0,3}(u) = (1 - u)^3\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{1,3}(u) = 3\ (1 - u)^2\ u\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{2,3}(u) = 3\ (1 - u)\ u^2\ \text{If}[0 \leq u < 1, 1, 0]$

$N_{3,3}(u) = u^3\ \text{If}[0 \leq u < 1, 1, 0]$

Se muestran gráficamente:



El producto tensorial de las funciones base:

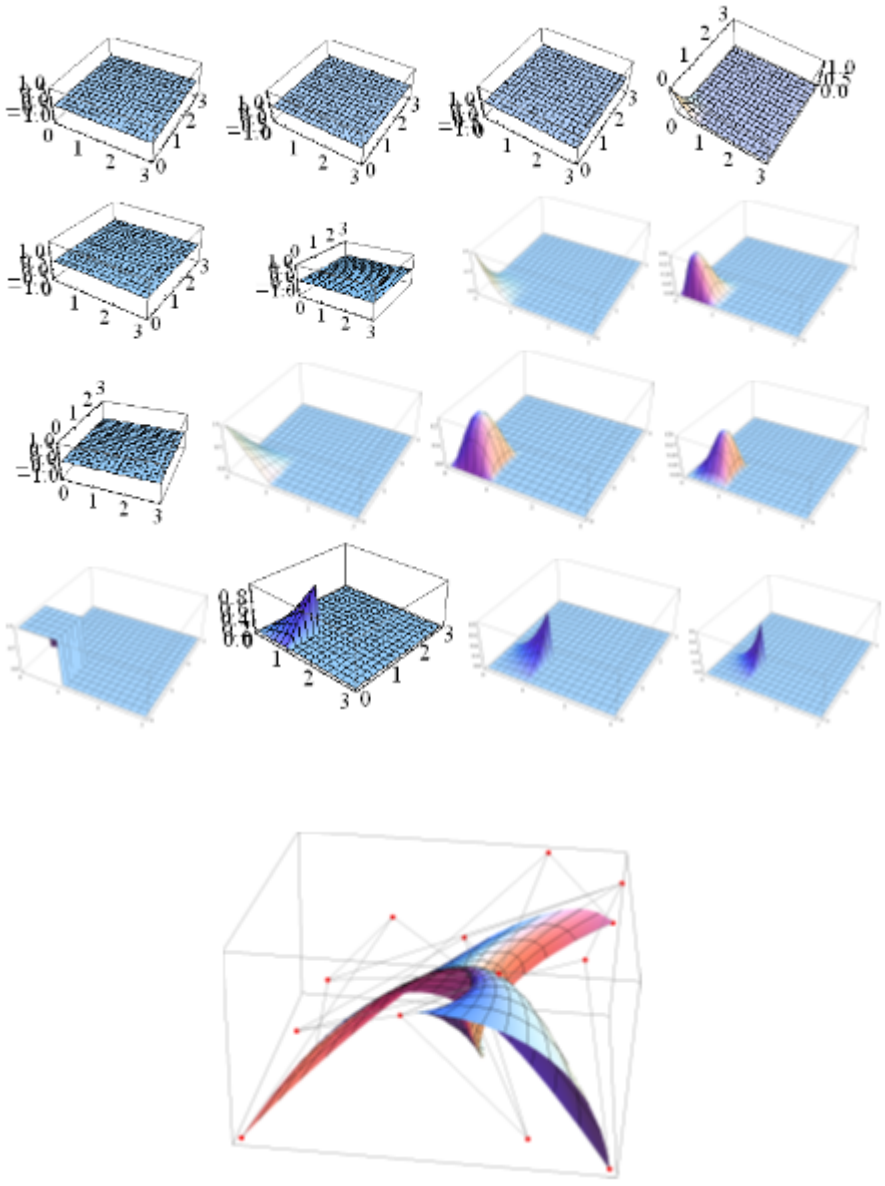


Figura 4.33: Superficie B-Spline

Capítulo 5

PLANTEAMIENTO CÓMO CONSTRUIR SUPERFICIES BASADAS EN B-SPLINE

Una superficie B-Spline es obtenida al tomar una red bidireccional de puntos de control, dos vectores nudo, y los productos de las funciones B-Spline univariadas

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u) N_{j,q} P_{i,j} \tag{5.1}$$

Con

$$U = \{\underbrace{0,\ldots,0}_{p+1}, u_{p+1}, \ldots, u_{r-p-1}, \underbrace{1,\ldots,1}_{p+1}\}$$

$$V = \{\underbrace{0,\ldots,0}_{q+1}, v_{q+1}, \ldots, v_{s-q-1}, \underbrace{1,\ldots,1}_{q+1}\}$$

U tiene $r+1$ nudos, y V tiene $s+1$.

$$r = n + p + 1$$

y

$$s = m + q + 1$$

Recordemos que:

Sea $U = \{u_0, \ldots, u_m\}$ una secuencia no decreciente de números reales. Es decir $u_i \leq u_{i+1}, \quad i = 0, \ldots, m-1$. Los u_i son llamados nudos, y U es el vector de nudos. La i-ésima función base B-Spline de grado p (orden $p+1$) denotada por $N_{i,p}(u)$ es definida como

$$N_{i,0}(u) = \begin{cases} 1 & u_i \leq u < u_{i+1} \\ 0 & c.o.f \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

Análogamente sea $V = \{v_0, \ldots, v_s\}$ una secuencia no decreciente de números reales. La j-ésima función base B-Spline de grado q (orden $q+1$) denotada por $N_{j,q}(u)$ es definida como

$$N_{j,0}(v) = \begin{cases} 1 & v_j \leq v < v_{j+1} \\ 0 & c.o.f \end{cases}$$

$$N_{j,q}(v) = \frac{v - v_j}{v_{j+q} - v_j} N_{j,q-1}(v) + \frac{v_{j+q+1} - v}{v_{j+q+1} - v_{j+1}} N_{j+1,q-1}(v)$$

5.1. DISEÑO DE SUPERFICIES.

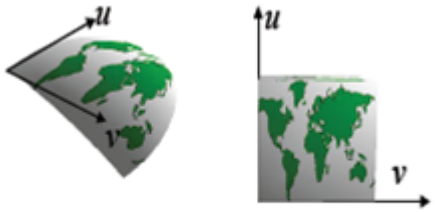


Figura 5.1:

Una superficie se describe matemáticamente usando dos parámetros, que establecen un sistema de coordenadas sobre ella, permitiéndonos recorrerla. La superficie se puede definir directamente a partir de una malla de puntos de control, al igual que una curva, con la única diferencia de que los puntos de control forma una distribución bidimensional y que las funciones de forma dependen de dos parámetros, $B_{i,j}(u, v)$. Alternativamente, es posible definir una superficie a partir de una o varias curvas. Por otra parte, es cada vez más frecuente utilizar mallas de triángulos como una aproximación a la superficie. En esta sección se

aborda la representación y diseño de superficies, comenzando con la aproximación por mallas de triángulos, la construcción a partir de curvas y la generación directa usando una malla de puntos de control.

5.2. ALGORITMO

- Declarar variables:

Dos enteros positivos n y m para definir el número de puntos de control $((n+1)(m+1))$.

Dos enteros positivos p y q que determinan los grados de la función bivariada.

Nota $n \geq p$,

el número de puntos de control debe ser mayor que el grado ya que no se podría construir una curva de grado tres con dos puntos de control. Lo mismo para m, $m \geq q$.

- Construcción de nodos.

Como U tiene $r+1$ nudos, y V tiene $s+1$ nudos entonces se asignan:

$$r = n+p+1 \; y \; s = m+q+1$$

Los nodos son de la forma

$$U = \{\underbrace{0,\ldots,0}_{p+1}, u_{p+1},\ldots,u_{r-p-1}, \underbrace{1,\ldots,1}_{p+1}\}$$

$$V = \{\underbrace{0,\ldots,0}_{q+1}, v_{q+1},\ldots,v_{s-q-1}, \underbrace{1,\ldots,1}_{q+1}\}$$

Veamos que

$$(r+1)-p-1-p-1 = (n+p+1+1)-p-1-p-1 = n-p$$

Ahora para formar U partimos al intervalo $[0,1]$ de forma equitativa en $n-p+1$ lugares para construir el siguiente nodo con $r+1$ componentes.

$$U = \{\underbrace{0\ldots,0}_{p+1}, \frac{2}{n-p+1}, \frac{1}{n-p+1}, \ldots, \frac{n-p-1}{n-p+1}, \frac{n-p}{n-p+1}, \underbrace{1\ldots,1}_{p+1}\}$$

Una opción para construir este nudo es hacer tres vectores y después unirlos.

- Puntos de control.

El número de puntos de control es $((n+1)(m+1))$. Los puntos $P_{i,j}$ están dados en el espacio entonces están compuestos de tres componentes donde $i = \overline{0,n}$,$j = \overline{0,m}$.

- Funciones base de b-Spline.

Para $p = 0$ se asigna de la siguiente manera.

$$N_{i,0}(u) = \begin{cases} 1 & u_i \leq u < u_{i+1} \\ 0 & c.o.f \end{cases}$$

Para $p \geq 0$ con un ciclo se definen las funciones.

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

Superficie b-Spline

Por último la función bivariada

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

5.3. SELECCIÓN DE PUNTOS DE CONTROL DE UNA CURVA

Figura 5.2: Curva de grado 3

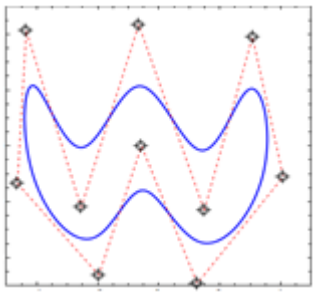


Figura 5.3: Curva de grado 2

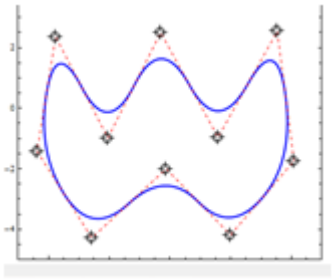
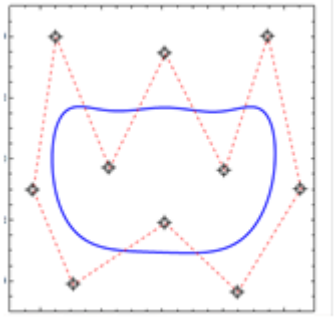


Figura 5.4: Curva de grado 7



Dada una curva específica los puntos de control son los puntos máximos o mínimos de una curva en un intervalo. La distancia entre la curva y el punto de control depende del grado de la curva en la primer imagen de abajo se muestra una curva B-Spline de grado 3 en la siguiente de grado 2, cuando en grado aumenta la distancia entre en punto de control y la curva crece.

5.4. SELECCIÓN DE PUNTOS DE CONTROL DE UNA SUPERFICIE

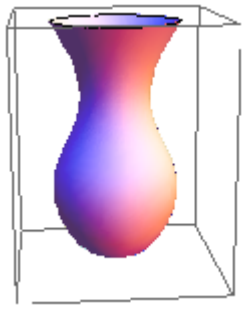


Figura 5.5: florero

Para construir una superficie es análogo al procedimiento de selección de puntos de una curva en este caso se tiene la unión de varias curvas, es decir curvas sobre "x" y "y" en diferente altura. Imaginemos que estamos haciendo un florero de barro en un torno de alfarero como se muestra en la figura siguiente;

Primero con el barro se forma un cono, el siguiente paso es darle forma, entonces se empieza desde la parte inferior, mientras el torno gira la distancia entre las manos determina el ancho de la figura. La posición de las manos al inicio es casi cerrada después va aumentando su distancia entre ellas luego disminuye y después nuevamente aumenta y la figura está terminada.

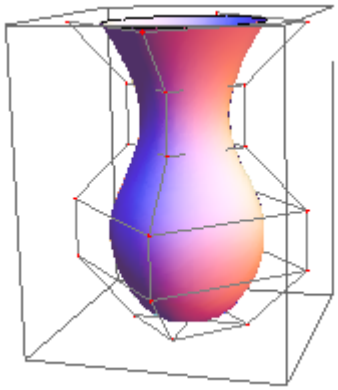


Figura 5.6: florero con puntos de control

Esto ayuda a calcular los puntos de control, que serían algunos puntos de la posición de nuestras manos.

En la figura 5.6 se muestra el florero indicando sus puntos de control.

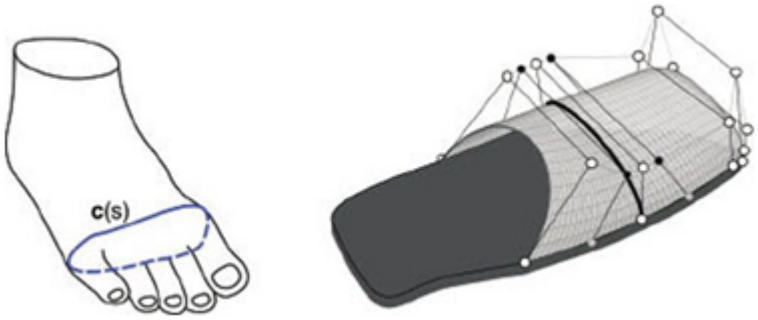


Figura 5.7:

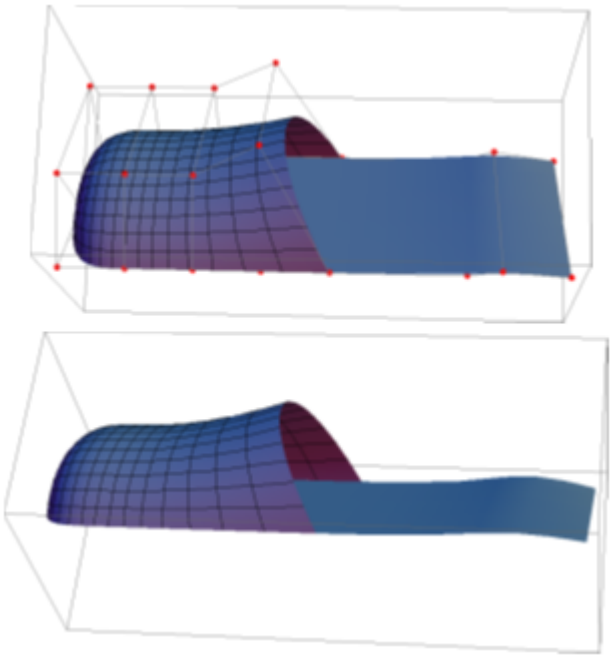


Figura 5.8: Sandalia

Es apreciable observar que con pocos puntos se pueda construir una superficie.

Capítulo 6

RESULTADOS Y SUS APLICACIONES

6.0.1. FLORERO

Construcción de un florero, se eligieron adecuadamente los puntos de control para formar una base de datos y así con el programa elaborado (Apéndice A, programa A4) construir la superficie.

La lista de puntos que se necesita para construir un florero como el que se muestra, es la siguiente:

0 0 0	0 0 0	0 0 0	0 0 0
1 0 0	0 1 0	−1 0 0	0 −1 0
2 0 1	0 2 1	−2 0 1	0 −2 1
2 0 2	0 2 2	−2 0 2	0 −2 2
1 0 3	0 1 3	−1 0 3	0 −1 4
1 0 4	0 2 5	−1 0 4	0 −1 3
2 0 5	0 2 2	−2 0 5	0 −2 5

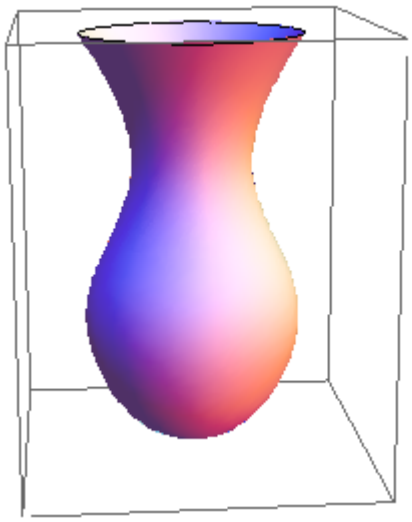


Figura 6.1: florero

El orden de los puntos es importante, como se puede observar en los primeros siete puntos se fija la coordenada y en 0 y las otras componentes se mueven. Despues la coordenada x es la que esta fija, enseguida nuevamente es la coordenada y pero ahora las coordenada es x son negativas, lo mismo pasa con el ultimo grupo de siete puntos, la coordenada x es fija y los negativos son en la coordenda y.

El florero es una superficie bicúbica Se define a las funciones base de grado tres (se podría escoger cualquier grado). Ahora la lista de datos se ordena en forma de tabla para realizar el producto tensorial.

$$\{\{\{0,0,0\},\{1,0,0\},\{2,0,1\},\{2,0,2\},\{1,0,3\},\{1,0,4\},\{2,0,5\}\},$$

$$\{\{0,0,0\},\{0,1,0\},\{0,2,1\},\{0,2,2\},\{0,1,3\},\{0,1,4\},\{0,2,5\}\},$$

$$\{\{0,0,0\},\{-1,0,0\},\{-2,0,1\},\{-2,0,2\},\{-1,0,3\},\{-1,0,4\},\{-2,0,5\}\},$$

$$\{\{0,0,0\},\{0,-1,0\},\{0,-2,1\},\{0,-2,2\},\{0,-1,3\},\{0,-1,4\},\{0,-2,5\}\}\}$$

Ahora el vector nudo $U = \{0,0,0,0,1,1,1,1\}$ y $V = \{0,0,0,0,\frac{1}{4},\frac{1}{2},\frac{3}{4},1,1,1,1\}$
Del primer nudo U obtenemos las siguientes funciones base:

$N1_{0,0}(u) = 0$

$N1_{1,0}(u) = 0$

$N1_{2,0}(u) = 0$

$N1_{3,0}(u) = \text{If}[0 \le u < 1, 1, 0]$

$N1_{0,1}(u) = 0$

$N1_{1,1}(u) = 0$

$N1_{2,1}(u) = (1-u)\,\text{If}[0 \le u < 1, 1, 0]$

$N1_{3,1}(u) = u\,\text{If}[0 \le u < 1, 1, 0]$

$N1_{0,2}(u) = 0$

$N1_{1,2}(u) = (1-u)^2\,\text{If}[0 \le u < 1, 1, 0]$

$N1_{2,2}(u) = 2\,(1-u)\,u\,\text{If}[0 \le u < 1, 1, 0]$

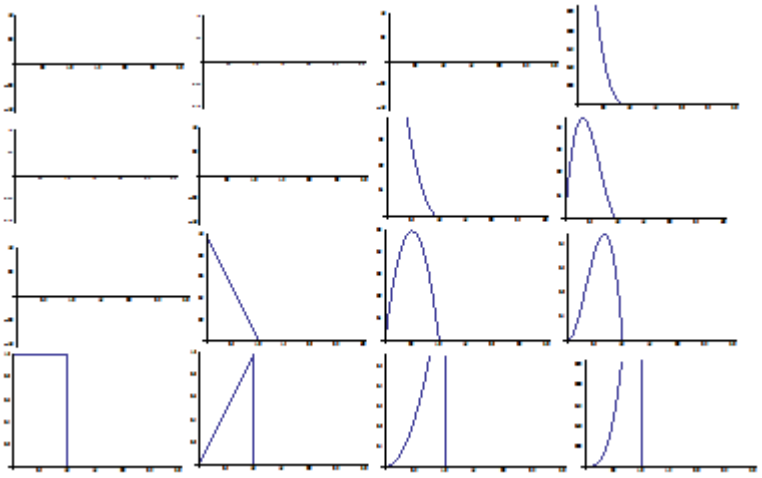
$N1_{3,2}(u) = u^2\,\text{If}[0 \le u < 1, 1, 0]$

$N1_{0,3}(u) = (1-u)^3\,\text{If}[0 \le u < 1, 1, 0]$

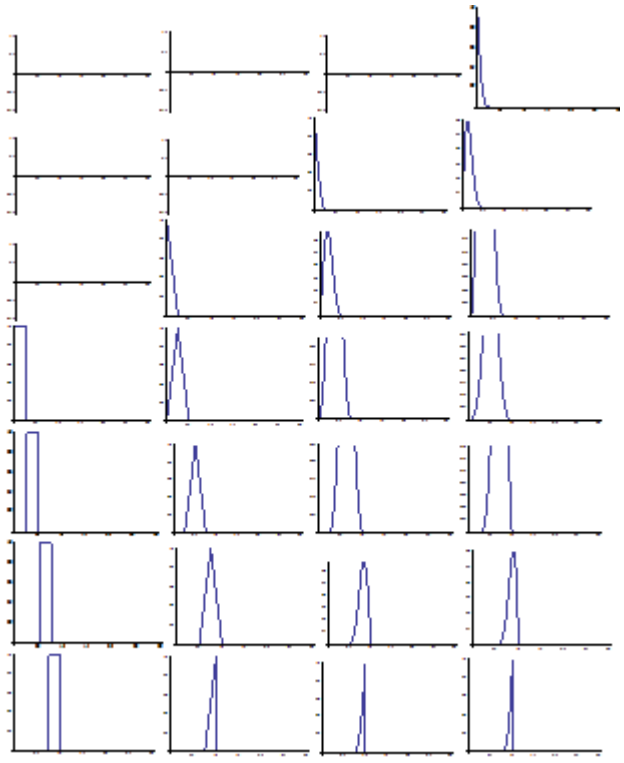
$N1_{1,3}(u) = 3\,(1-u)^2\,u\,\text{If}[0 \le u < 1, 1, 0]$

$N1_{2,3}(u) = 3\,(1-u)\,u^2\,\text{If}[0 \le u < 1, 1, 0]$

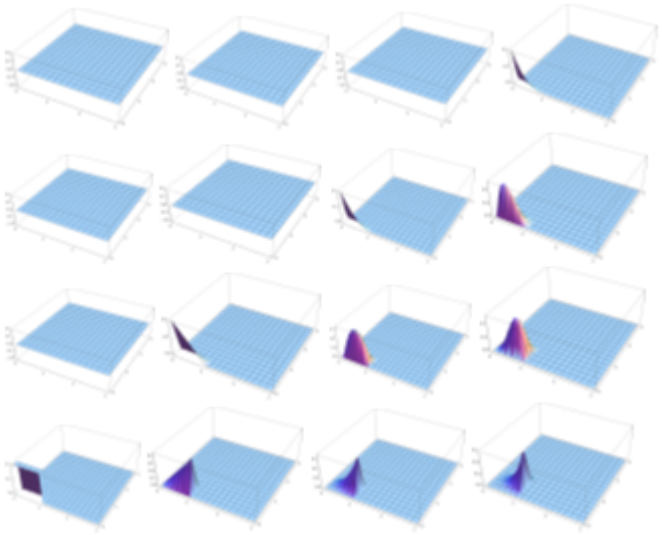
$N1_{3,3}(u) = u^3\,\text{If}[0 \le u < 1, 1, 0]$



Del segundo nudo V se tienen las siguientes funciones base:



El producto tensorial se ilustra gráficamente



Los restantes productos tensoriales son cero ya que $i = \overline{0,3}$. Por último para obtener la superficie

$$S(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} N_{i,p}(u)N_{j,q}(v)P_{i,j}$$

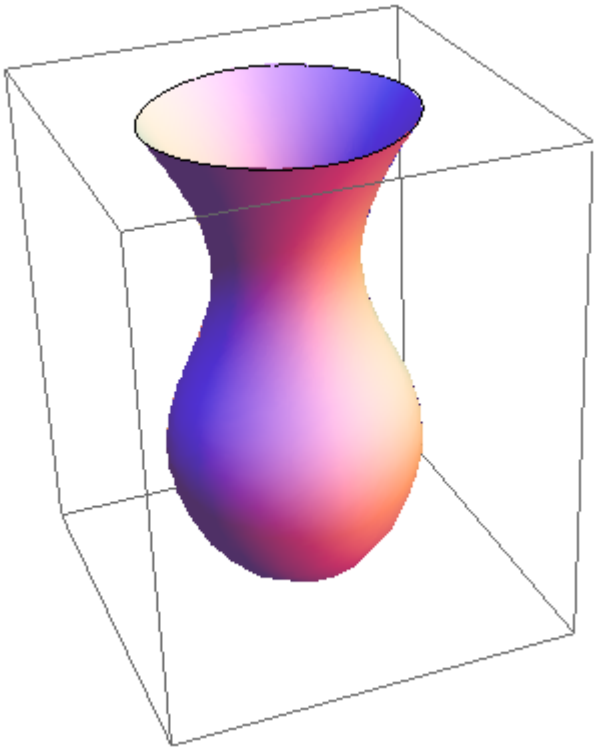


Figura 6.2:

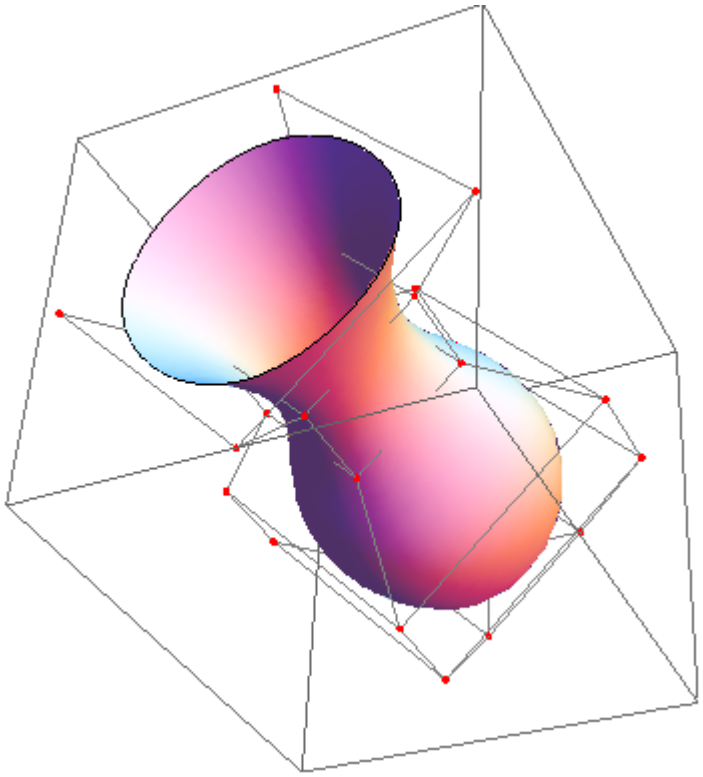


Figura 6.3:

6.0.2. TAZA DE TÉ

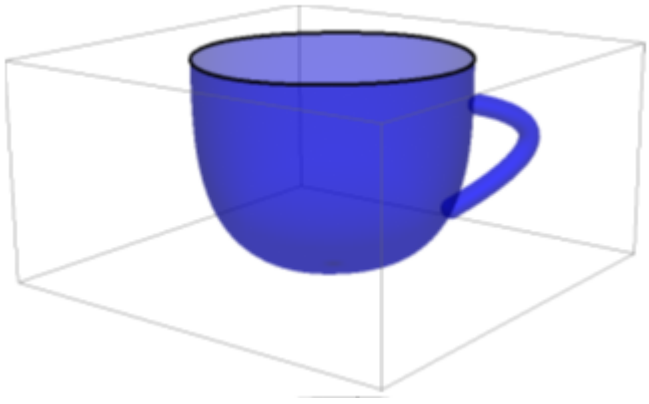


Figura 6.4: Superficie contruida con funciones B-Spline

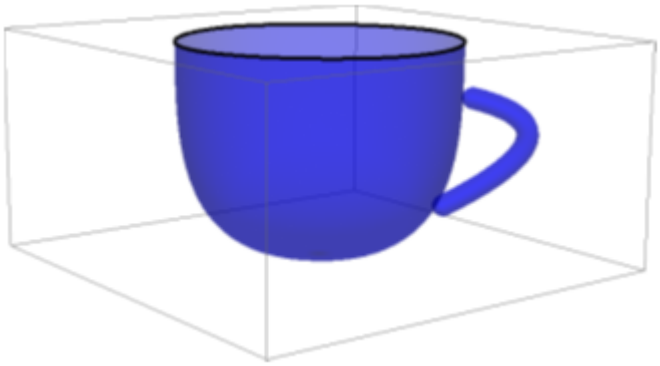


Figura 6.5: taza de té

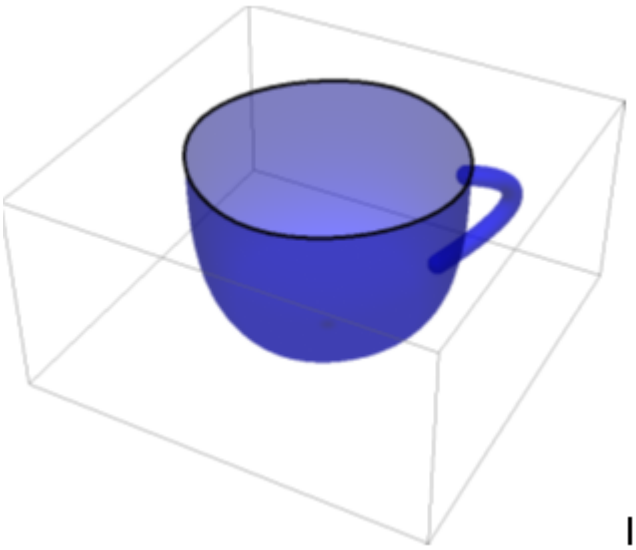


Figura 6.6:

Esta taza es la unión de dos superficies B-Spline lo que es el tazón y la oreja.

6.0.3. TETERA

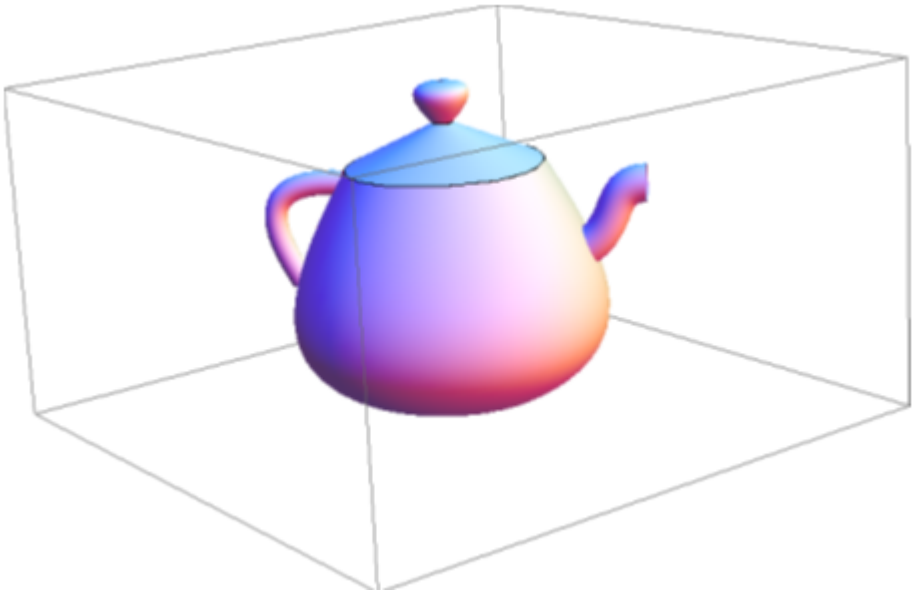


Figura 6.7: Superficie construida con funciones B-Spline

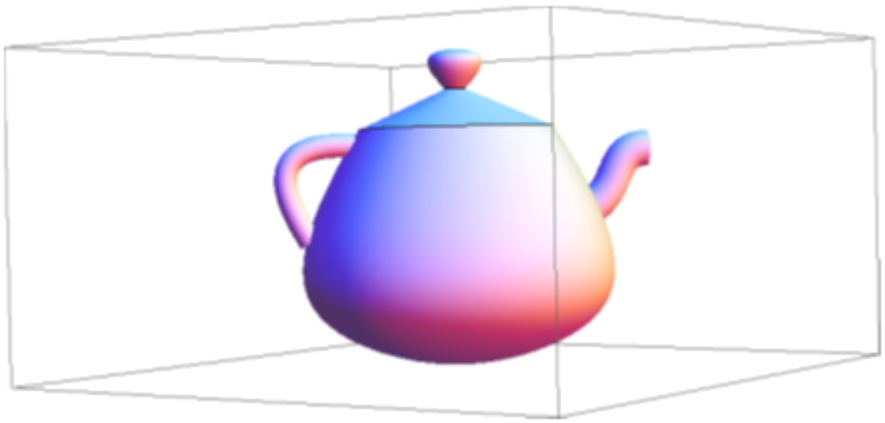


Figura 6.8: tetera

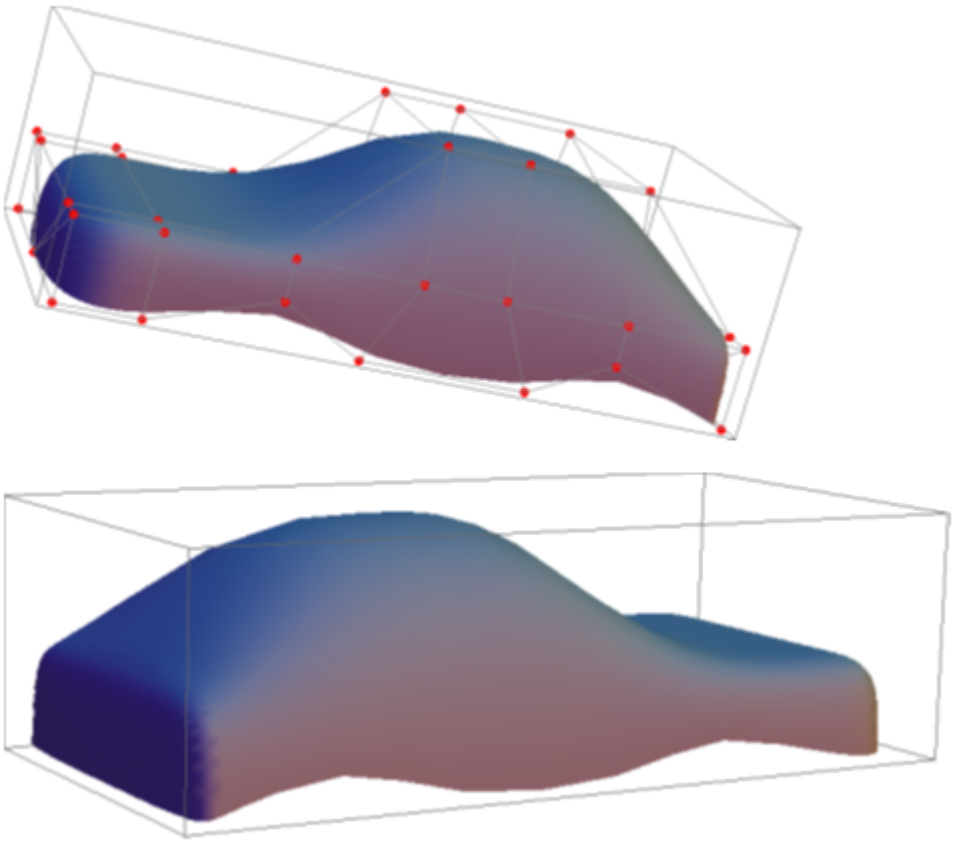


Figura 6.9: Carcasa de un auto construida con superficies B-Spline

Para construir esta figura se unen cinco superficies B-Spline lo que es la base, la oreja, el pitorro, la tapa y el silbato.

6.1. APLICACIONES

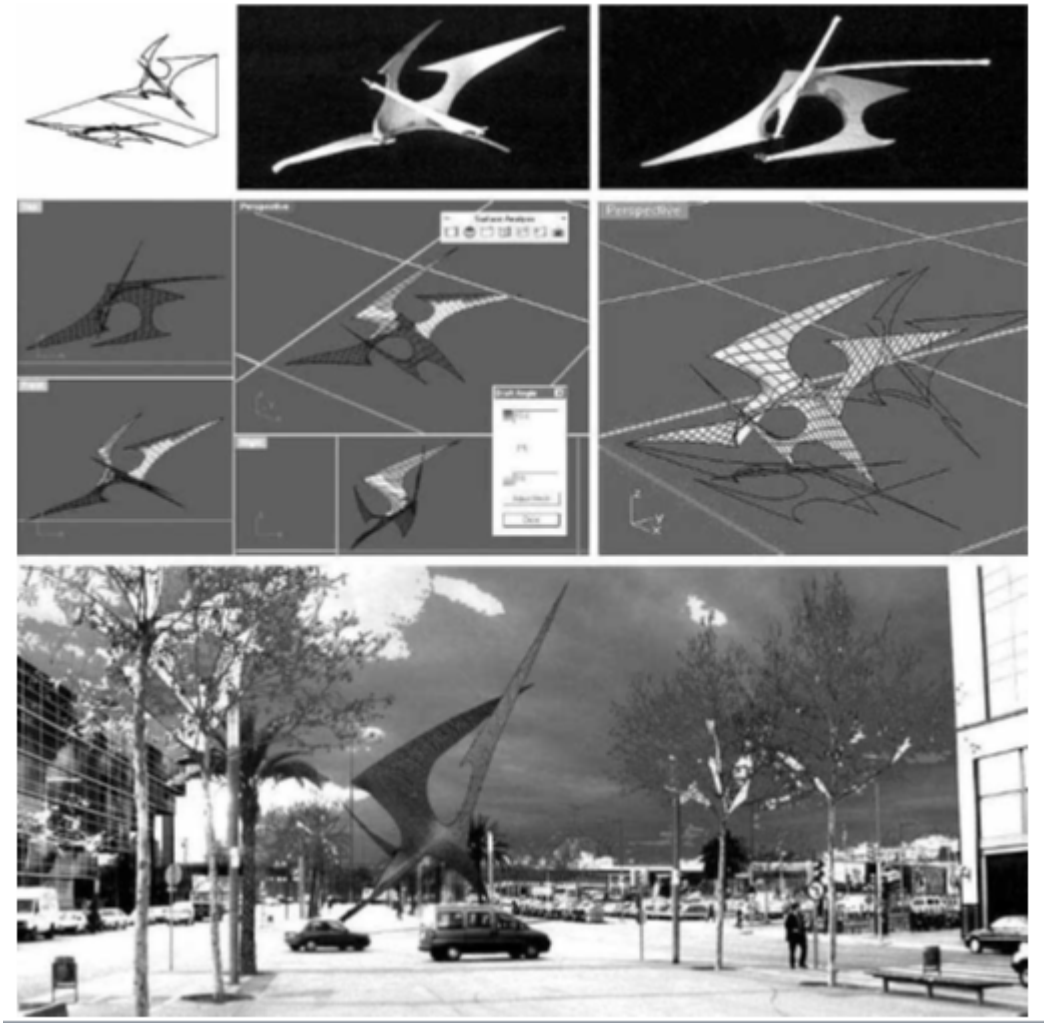
En el pensamiento gráfico arquitectónico, la geometría aparece habitualmente como soporte instrumental de la especulación proyectual. Los procedimientos geométricos se presentan como recursos de representación para la comprobación gráfica de la reflexión y la exposición de las ideas con la intención de construir un orden lógico, tanto en la representación como en la pre-figuración formal. La geometría aparece tradicionalmente tanto en las etapas iniciales como en las finales del proceso generativo del diseño operando como instrumento de orden y síntesis, reafirmando la definición de componentes tecnológicos y espaciales.

La incorporación de las superficies paramétricas a las tradicionales geometrías euclidianas se manifiesta en el taller de arquitectura como una necesidad de los estudiantes para abordar la manipulación de geometrías de formas libres en una marcada intención de asumir una espacialidad del presente o simplemente por una seducción ingenua de las posibilidades formales de los nuevos instrumentos de modelado digital.

Los sistemas de ideación digitales potencian las capacidades de pensamiento de los estudiantes hacia una dirección que los obliga a forzar muchas veces los sistemas constructivos tradicionales o a modificar los resultados gráficos para lograr una correcta materialización del proyecto arquitectónico.

En el Taller de Arquitectura la pedagogía debe asumir las tradiciones del oficio y participar de los impulsos de una época innovadora y cargada de desafíos. Ante la preocupación de una cierta utilización ingenua de los recursos digitales es que se propone definir unas estrategias que acompañen las exigencias del proceso heurístico y las expectativas proyectuales evitando que se constituyan en instrumentos autónomos en donde la representación usurpe la identidad de lo representado.

Asumiendo la metáfora de estar en el umbral entre dos épocas (industrial-informacional, analógica-digital, material-virtual) y el desafío de los nuevos instrumentos conceptuales y operacionales que dispone nuestra disciplina, nos inclinamos a trabajar en la mixtura, sin exclusiones ni substituciones recíprocas a través de las alternativas de trabajo sugeridas para el abordaje de la problemática planteada desde el Taller de Arquitectura**[3]**.



Inversa por fotogrametría. Escultura Urbana Estudiante: Paulo Chiarella

Los arcos son creados con los poderosos nurbs y tres puntos de control.

La contrucción de vías para el tren son un ejemplo de la aplicación de los B-Spline

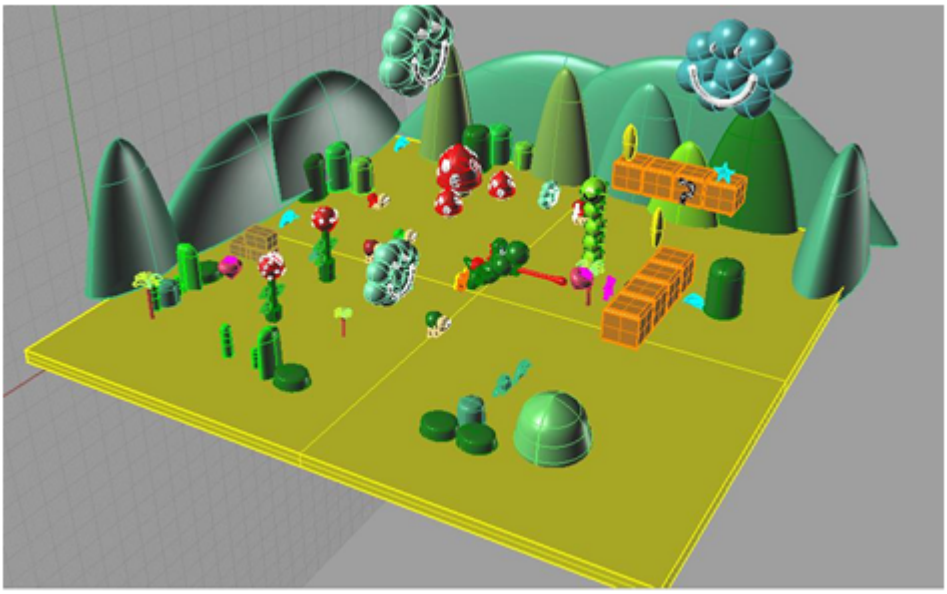
Parte I

CONCLUSIONES

CONCLUSIONES

Las funciones B-Spline son ajustables, es el mejor método para reconstruir superficies, ya que los puntos de control no se alejan tanto de la forma que queremos modelar, por lo que podemos asemejarnos más al modelo sin necesidad de utilizar muchos puntos de control. Ademas que se controla el grado de la curva.

Las curvas B-Spline proporcionan una manera rápida de definir una curva razonablemente cercana a los puntos de control (de hecho, empieza en el primero y acaba en el último), siendo además menos rígida que la utilización de curvas Bézier. Una curva B-Spline es una curva definida a trozos que puede ser de un grado relativamente bajo frente al grado que precisaríamos si quisiéramos usa un único segmento de Bézier. Ademas, dichos trozos enlazan entre ellos con una continuidad muy alta en relación con el grado con el que se está trabajando.



Con B-Spline se puede recrear un mundo virtual.

Parte II

REFERENCIAS

Bibliografía

[1] Piegl, Tiller - The NURBS Book (Springer 1997) segunda ediciónISBN 3-540-6145-8

[2] Paluszny, Prautzsch,Boehm,Métodos de Bézier y B-Spline, Universitatsverlag Karlsruhe 2005 ISBN 3-937300-47-3

[4] Eugenia Rosado, Espacio Afín, Departamento de Matemática Aplicada a la Edificación al Medio Ambiente y al Urbanismo, Universisdad Politecnica de Madrid

[3] Mauro Chiarella, Superficies Paramétricas y Arquitectura: conceptos, educación y desarrollo, Universisdad Nacional de Litoral FADU, Ciudad Universitaria- Santa Fe- Argentina

Parte III

APÉNDICE

APÉNDICE

PROGRAMA A1.

Programa para construir superficies de Bézier de forma aleatoria en Mathematica. Define el grado del polinomio y los puntos de control de forma aleatoria. El número de puntos de control depende del grado del polinomio. Si el grado de el polinomio es "n" entonces el número de puntos de control es $(n+1)^2$.

n=Random[Integer,{1,4}]

Print["polinomios de BERSTEIN de grado ", n]

Do[Print[Bi[λ_]=n!/(i! (n-i)!)*(1-λ)n-i* λi],{i,0,n}]

Print["Puntos de control"]

dat=Table[$P_{i,j}$={Random[Integer,{0,10}],Random[Integer,{0,10}],

Random[Integer,{0,10}]},{i,0,n},{j,0,n}]

Print["x(u,v)= ",] Print["y(u,v)= ",] Print["z(u,v)= ",]

Print["El número de puntos de control es ",$(n+1)^2$]

Print["La funcion de Bézier es de la siguiente manera"]

Bézier[u_,v_]=

$$\sum_{i=0}^{n}\sum_{j=0}^{n} B_i(u)B_j(v)P_{i,j}$$

Show[Graphics3D[{PointSize[Medium],Red,Map[Point,dat]}],

Graphics3D[{Gray,Line[dat],Line[Transpose[dat]]}],

ParametricPlot3D[**Bézier**[u,v],{u,0,1},{v,0,1},Mesh→None]]

PROGRAMA A2

Este programa construye la superficie de Bézier en base a los datos que ingresa el usuario, el programa solicita el grado del polinomio y en base a este pide un número de puntos de control, estos puntos se ingresan por cada entrada.

Clear[n]

n=Input["Escribe el grado del polinomio"]

Print["polinomios de BERSTEIN de grado ", n]

Do[Print[B_i[λ_]=n!/(i! (n-i)!)*(1-λ)n-i* λi],{i,0,n}]

Print["El número de puntos de control es ",$(n+1)^2$]

Print["Puntos de control"]

datos=Table[$Q_{i,j}$={Input["coordenada de la primera columna"],

Input["coordenada de la segunda columna"],

Input["coordenada de la tercera columna"]},{i,0,n},{j,0,n}]

Print["x(u,v)= ", $\sum_{i=0}^{n}\sum_{j=0}^{n} B_i(u)B_j(v)P_{i,j}$]

Print["y(u,v)= ", $\sum_{i=0}^{n}\sum_{j=0}^{n} B_i(u)B_j(v)P_{i,j}$]

Print["z(u,v)= ", $\sum_{i=0}^{n}\sum_{j=0}^{n} B_i(u)B_j(v)P_{i,j}$]

Bézier[u_,v_]=

$$\sum_{i=0}^{n}\sum_{j=0}^{n} B_i(u)B_j(v)Q_{i,j}$$

Show[Graphics3D[{PointSize[Medium],Red,Map[Point,datos]}],

Graphics3D[{Gray,Line[datos],Line[Transpose[datos]]}],

ParametricPlot3D[**Bézier**[u,v],{u,0,1},{v,0,1},Mesh→None]]

PROGRAMA A3

CONTRUCCIÓN DE CURVAS B-SPLINE

Programa que manda a llamar una dirección y después de la dirección anterior se importa un archivo el cual contiene datos con los cuales se construirá una curva B-Spline

(*Se ubica la dirección*)

SetDirectory["C:\\Users\\user\\Dropbox\\practicas_profesionales\\B-Spline"
];

Import["dat.txt"]; (*Se importa en archivo de datos*)

pts=ReadList["dat.txt",{Number,Number}]

(*Se forma una lista con los datos*)

n:=(Length[pts]-1)

d:=3(*grado del polinomio es d-1*)

Print[" Número de puntos control ", n+1]

Print[" puntos de control en cada segmento ", d]

(*El número de nudos está dado por m *)

m:=n+d+1

nudos={0,0,0,0,1,1,1,1};

(* (m+1)-(2*d+2)=(n+d+1+1)-d-d-1-1=n-d *)

nudos2=Table[i/(n-d+1),{i,1,n-d}];

nudos=Flatten[Insert[nudos,nudos2,5]];

p=Length[nudos]; nudos1=Table[t_{i-1}=nudos[[i]],{i,1,p}]

Print["Puntos de control" ,n+1]

Table[P_{i-1}=pts[[i]],{i,1,n+1}];

Do[$B_{i,0}$[u_]=If[$t_i \leq u < t_{i+1}$,1,0],{i,0,m-2}]

Do[Do[$B_{i,j}$[$u_]$=

If[$(t_{i+j} - t_i) = 0, 0, (u - t_i)/(t_{i+j} - t_i) * B_{i,j-1}[u]$]+

If[$(ti + j + 1 - t_{i+1}) = 0, 0, (t_{i+j+1} - u)/(t_{i+j+1} - t_{i+1}) * B_{i+1,j-1}[u]$] ,{i,0,m-1}],{j,1,d}]

Print["Funciones $B_{i,j}$"] tabla:=Table[Plot[$B_{i,j}[u]$,{u,0,m-1}] ,{i,0,n},{j,0,d}]

MatrixForm[tabla]

Print[" La función B-Spline queda en la siguiente forma"]

s[u_]=

$$\sum_{i=0}^{n} P_i B_{i,d}[u]$$

Show[Graphics[{Red,Point[pts],Green,Line[pts]},Axes→True],

ParametricPlot[s[u],{u,0,.9999}]]

PROGRAMA A4

Programa que manda a llamar una dirección y después de la dirección anterior se importa un archivo el cual contiene datos con los cuales se construirá una superficie B-Spline

(*Se ubica la dirección*)

SetDirectory["C:\\Users\\user\\Dropbox\\practicas_profesionales\\B-Spline"];

Import["florero.txt"]; (*Se importa en archivo de datos*)

FilePrint["florero.txt"]; (*Se imprime los datos para verificar*)

```
pts=ReadList["florero.txt",{Number,Number,Number}]

(*Se forma una lista con los datos*)

numero=Length[pts];

div=3;

ban=0;

While[div<=numero&&ban==0,If[Mod[numero,div]==0,ban=1];div++]

n:=(div-1)-1(*Se resta 1 a div ya que al salir del ciclo suma 1*)

m:=(numero/(div-1))-1

If[n≥3,d=3,d=n];

If[m≥3,q=3,q=m];

(* n debe ser mayor o igual que d (n≥d)

m debe ser mayor o igual que p (m≥p)*)

Print[" la primera función base es de grado ", d," La segunda es de ",q]

(*El número de nudos está dado por m *)

r:=n+d+1

s:=m+q+1

nud11=Table[0,{i,0,d}];

nud12=Table[1,{i,0,d}];

nudos1=Flatten[{nud11,nud12}];

nud21=Table[0,{i,0,q}];

nud22=Table[1,{i,0,q}];
```

nudos2=Flatten[{nud21,nud22}];

(* (m+1)-(2*d+2)=(n+d+1+1)-d-d-1-1=n-d *)

nudos10=Table[i/(n-d+1),{i,1,n-d}];

nudos1=Flatten[Insert[nudos1,nudos10,d+2]];

p1=Length[nudos1]; nudos13=Table[$t1_{i-1}$=nudos1[[i]],{i,1,p1}]

nudos20=Table[i/(m-q+1),{i,1,m-q}];

nudos2=Flatten[Insert[nudos2,nudos20,q+2]];

p2=Length[nudos2]; nudos23=Table[$t2_{i-1}$=nudos2[[i]],{i,1,p2}]

Print["Puntos de control " ,(n+1)(m+1)]

If[n≥m, dat=Table[$P_{i,j}$={pts[[(n+1)*i+1+j,1]],pts[[(n+1)*i+1+j,2]],

pts[[(n+1)*i+1+j,3]]},{i,0,n},{j,0,m}],

dat=Table[$P_{i,j}$={pts[[(m+1)*i+1+j,1]],pts[[(m+1)*i+1+j,2]],

pts[[(m+1)*i+1+j,3]]},{i,0,n},{j,0,m}]]

Do[$B1_{i,0}$[u_]=If[$t1_i \leq u < t1_{i+1}$,1,0],{i,0,r}]

Do[Do[$B1_{i,j}[u_]$=If[$(t1_{i+j} - t1_i) = 0, 0, (u - t1_i)/(t1_{i+j} - t1_i) * B1_{i,j-1}[u]$]+

If[$(t1_{i+j+1} - t1_{i+j} = 0, 0, (t1_{i+j+1} - u)/(t1_{i+j+1} - t1_{i+1}) * B1_{i+1,j-1}[u]$] ,{i,0,n+1}],{j,1,d}]

Do[$B2_{i,0}[v_] = If[t2_i \leq v < t2_{i+1}, 1, 0]$,{i,0,s}]

Do[Do[$B2_{i,j}[v_]$=If[$(t2_{i+j} - t2_i) = 0, 0, (v - t2_i)/(t2_{i+j} - t2_i) * B2_{i,j-1}[v]$]+

If[$(t2_{i+j+1} - t2_{i+j}) = 0, 0, (t2_{i+j+1} - v)/(t2_{i+j+1} - t2_{i+1}) * B2_{i+1,j-1}[v]$] ,{i,0,m+1}],{j,1,q}]

S[u_,v_]:=

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B1_{i,d}(u) * B2_{j,q}[v] * P_{i,j}$$

```
Show[Graphics3D[{PointSize[Medium],Red,Map[Point,dat]}],
Graphics3D[{Gray,Line[dat],Gray,Line[Transpose[dat]]}],
ParametricPlot3D[S[u,v],{u,0,1},{v,0,1}]]
Print[" La función B-Spline queda en la siguiente forma"]
S[u_,v_]=
```

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B1_{i,d}(u) * B2_{j,q}[v] * P_{i,j}$$