UNIVERSIDAD DE LA RIOJA

**Facultad de Ciencia y Tecnología**

**TRABAJO FIN DE GRADO**

Grado en Matemáticas

# Esquemas de subdivisión y animación gráfica

Realizado por:

Alicia Burgos Carrascón

Tutelado por:

José Manuel Gutiérrez Jiménez

**Logroño, 7 de julio de 2021**

# Índice general

# Abstract

Computer graphics is an appealing example of the wide variety of applications of Mathematics in other fields. In this dissertation, we introduce some of the mathematical tools and algorithms behind curve and surface representations in the two dimensional plane and in the tree dimensional space. More specifically, we intend to analyze the subdivision techniques applied in computer-aided design.

The path we will follow towards the goal of curve and surface modelling has its starting point in the Bernstein polynomials and Bézier curves and surfaces, rather than in any other representations, such as splines or B-splines. The main reason is that their properties make them a simple and adequate element, not only analitically, but also graphically.

# Resumen

La animación gráfica es un atractivo ejemplo de aplicación de las Matemáticas en otras disciplinas. En este trabajo presentamos algunas de las herramientas matemáticas y algoritmos que se emplean para representar curvas y superficies definidas en el plano y en el espacio tridimensional. En concreto, pretendemos analizar las técnicas de subdivisión que hay detrás de los proyectos de diseño asistido por ordenador.

El camino que seguiremos hacia el objetivo final del modelado mediante curvas o superficies tiene como punto de partida el estudio de los polinomios de Bernstein y las curvas de Bézier, en lugar de otras representaciones como los splines o B-splines. El principal motivo es que sus propiedades hacen de ellas un elemento favorable a nivel analítico y, por ende, a nivel gráfico.

# Introducción

El diseño asistido por ordenador y el diseño gráfico son, en la actualidad, dos métodos fundamentales, ya sea en ciencia, tecnología o ingeniería. El objetivo de este Trabajo es analizar las técnicas de aproximación mediante las curvas de Bézier definidas en el plano, comenzando por los conceptos más esenciales, e ilustrarlas mediante ejemplos en el área del diseño.

El desarrollo de las curvas de Bézier surgió en la década de los años 50 del siglo pasado, de la mano de ingenieros franceses, debido a los problemas de diseño a los que se enfrentaba la industria automovilística. Ocurría que, cuando se quería realizar alguna modificación sobre el diseño original de las piezas fabricadas, por ínfimo que fuese, se debía reproducir el dibujo completo de la misma, lo cual resultaba ser una gran inversión de tiempo y de dinero. Motivado por el carácter obsoleto de las técnicas clásicas y la aparición de los primeros ordenadores, se produjo un periodo de gran innovación en este campo.

En aquel tiempo, ya se conocían algunas técnicas de implementación informática para representar curvas en el plano, como los splines cúbicos o los polinomios de interpolación de Lagrange. A pesar de ello, no resultaban del todo efectivos: los splines, al igual que las plantillas de diseño, requerían reproducir la curva a lo largo de tramos muy pequeños; y los polinomios de Lagrange eran muy sensibles a las variaciones en los puntos de interpolación y requerían un número grande de evaluaciones, complicando su representación gráfica.

Faltaba por introducir una propiedad de control pseudo-local o de pseudo-control local, una característica muy interesante de las curvas de Bézier, que detallaremos más adelante. Se basa en trazar una curva que pase por determinados puntos de la forma más aproximada posible, haciendo que la curva solución se asemeje al aspecto de la figura que resulta de unir los puntos mediante segmentos. Es decir, si los puntos son colineales, obtendremos una recta, y la variación en alguno de estos puntos hace que la curva obtenida se comporte de forma similar en los puntos que se alejan.

Las curvas de Bézier son, en esencia, polinomios de Bernstein y heredan sus propiedades, a todos los efectos. Esta familia de funciones matemáticas fue investigada por el matemático ruso Sergei Natanovich Bernstein, cuyas publicaciones al respecto están citadas en 1912.

El matemático Paul de F. De Casteljau y el ingeniero Pierre E. Bézier fueron pioneros en esta investigación. Trabajaban como ingenieros en las empresas automovilísticas francesas *Renault* y *Citröen*, respectivamente, y ambos resolvieron, de forma paralela pero indepentiente, el problema de diseño asistido por ordenador mediante splines. De Casteljau fue el primero en hacerlo; sin embargo, el trabajo de Bézier fue publicado primero, promoviendo el uso de las curvas y superficies que a día de hoy llevan su nombre.

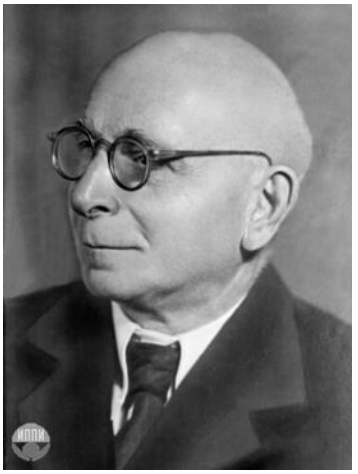


Figura 1: El matemático Sergei Natanovich Bernstein(1880–1968) y los ingenieros franceses Pierre Bézier (1919–1999) y Paul De Casteljau (1930–), pioneros del diseño gráfico asistido por ordenador.

Este tipo de curvas pronto trascendieron el ámbito donde surgieron, y a día de hoy son la base de gran cantidad de programas de diseño, como AutoCAD, Adobe Illustrator, o SolidWorks. Además, su formalización también es posible mediante programas de cálculo como Wolfram Mathematica, Matlab o Sagemath.

El entorno TikZ, de LaTeX, permite trabajar con este tipo de curvas. En concreto, el comando

```
\draw (P) .. controls (C) and (D).. Q;
```

dibuja una curva de este tipo, para la que se necesitan cuatro puntos, los extremos $P$ y $Q$, y un par de puntos de control. Además, con una mínima modificación en cualquiera de ellos, se puede generar una gran variedad de animaciones.

También es posible definir un nodo en una coordenada determinada mediante

```
\path (0,0)
    node[draw,shape=circle] (v0)
    {$v_0$};
```

que determina un nodo llamado $v_0$ centrado en el origen, de forma circular y con su correspondiente etiqueta. Mediante la opción `draw`, se dibujaría el nodo asociado. De hecho, las gráficas que se encuentran en el presente documento han sido generadas mediante los programas de cálculo y el entorno mencionados.

En este Trabajo de Fin de Grado, nos acercaremos a una de las aplicaciones más a la orden del día de la aproximación mediante curvas, intentando reflejar el esquema:

$$\text{Polinomios de Bernstein} \rightarrow \text{Curvas de Bézier}$$

como alternativa al esquema interpolatorio:

$$\text{Splines} \rightarrow \text{B-Splines} \rightarrow \text{NURBS},$$

ya que el estudio de sus propiedades matemáticas, basadas en el concepto de la subdivisión, resultan más interesantes a nivel analítico.

En el Capítulo 1, se definen los polinomios de Bernstein en el intervalo $[0, 1]$, además de en el intervalo arbitrario $[a, b]$. Gracias al estudio de sus propiedades más significativas, se obtienen algunos resultados relevantes, de entre los cuales destacamos la demostración del Teorema de Weierstrass, que establece que cada función continua sobre un compacto puede ser aproximada uniformemente por polinomios. Así, se evidencia la adecuación de dichos polinomios en la aproximación de cualquier función que cumpla ciertas condiciones de continuidad.

En el Capítulo 2, se utiliza la base de los polinomios de Bernstein para definir la representación de Bézier de una curva polinomial en el plano. Además de enunciar sus propiedades más fundamentales, se introducen la noción de vértice de una curva de Bézier y el concepto de polinomio parcial de Bézier, con el fin de entender el algoritmo de De Casteljau, esencial en la construcción recursiva de curvas de Bézier. El capítulo finaliza con algunos enunciados relativos a las derivadas de dichas curvas.

En el Capítulo 3, se estudian las transformaciones que más habitualmente se realizan a las curvas de Bézier, como son: la elevación del grado (que hace posible describir una curva de Bézier de grado $n + 1$ a partir de una de grado $n$); la subdivisión (que permite definir el trazado de una curva de Bézier generando nuevos nodos a partir de los nodos iniciales) y la conexión de curvas de Bézier (que establece una serie de condiciones de continuidad y derivabilidad para que dos segmentos de Bézier puedan ser renderizados de forma idónea). Para concluir, se hace ver la relación de dichas curvas de Bézier con los Splines, justificando la decisión de centrarnos en el estudio de las primeras, en vez de en las segundas.

En el Capítulo 4, nos encaminamos a las aplicaciones de las curvas de Bézier en el campo de la animación gráfica por medio del análisis de las superficies de Bézier. Vemos que cuentan con características muy similares a las de las curvas, lo cual hace posible que extendamos el ya estudiado algoritmo de De Casteljau a las superficies de Bézier. Por último, observamos que los procesos de elevación del grado y de derivación no se diferencian de los estudiados para curvas polinómicas.

Finalmente, el Capítulo 5, destacamos el papel que juegan las curvas de Bézier en la animación y el modelado de objetos por ordenador. Nos centraremos en el modelo de la tetera de Utah, objeto de referencia en la comunidad de la computación gráfica, además de en los esquemas de subdivisión de superficies de Bézier. Concretamente, estudiaremos el esquema de Catmull-Clark, empleado por la compañía de animación *Pixar* en algunas de sus películas. Además de introducir el esquema recursivo de dicho algoritmo, presentamos varios ejemplos de superficies de Catmull-Clark mediante MATHEMATICA.

Todo ello no habría sido posible sin la dedicación de mi tutor, José Manuel Gutiérrez, "Guti", al que dedico estas líneas a modo de agradecimiento. Con su buen hacer y sus consejos no solo he tratado un tema diferente, curioso y nuevo para mí, sino que también he disfrutado del proceso.

# Capítulo 1

# Polinomios de Bernstein

La forma de las curvas de Bézier puede expresarse de forma explícita, en función de los polinomios de Bernstein. Es por eso que, en este capítulo, vamos a definir y estudiar algunas propiedades sobre ellos.

Este tema ha sido tratado en destacadas publicaciones, de las cuales podemos citar: [5], [10] o [16], algunas de ellos con una clara variante analítica.

## 1.1. Definición

Con la creciente importancia de la construcción por computador, a partir de los años 70 del siglo pasado se abrió un interesante camino en la teoría de la interpolación y aproximación, donde algunos aspectos geométricos obtuvieron una importancia decisiva. Una curva o una superficie tiene que ser representada en un ordenador de forma que pueda dibujarse o manipularse de forma inmediata. Con este fin, se usan parametrizaciones de objetos geométricos.

En este caso, consideraremos las situaciones más simples y, en particular, nos restringiremos a las curvas polinómicas, como pueden ser los objetos geométricos unidimensionales. Comenzaremos con una generalización de aquellos polinomios que toman valores reales.

**Definición 1.1.** *Un polinomio (o curva polinomial) de grado $n$ en $\mathbb{R}^d$ es una función $P$ de la forma $P : \mathbb{R} \longrightarrow \mathbb{R}^d$, tal que*

$$P(t) = \sum_{i=0}^{n} a_i t^i \quad con \quad a_0, a_1, a_2, \ldots, a_n \in \mathbb{R}^d, \quad a_n \neq 0.$$

El espacio de los polinomios de grado menor o igual que $n$ en $\mathbb{R}^d$ se denota por $\mathbb{P}_n^d$. Para el caso $d = 1$, los denotaremos por $\mathbb{P}_n$.

A medida que las técnicas interpolatorias fueron evolucionando, surgieron distintas bases del espacio de polinomios $\mathbb{P}_n^d$. Si bien la más conocida es la base de monomios $\{1, t, t^2, \ldots, t^n\}$, en algunas referencias como [10] encontramos que, mediante el polinomio interpolador de Lagrange, se introdujo una nueva base $\{L_0(t), L_1(t), \ldots, L_n(t)\}$, además de la base de Newton $\{\omega_0(t), \omega_1(t) \ldots, \omega_n(t)\}$. A diferencia de la primera, estas dos últimas dependen de $n+1$ nodos, $t_0, \ldots, t_n$, lo cual las hace adecuadas para representaciones locales de polinomios.

Empezaremos considerando el intervalo cerrado entre los puntos $a$ y $b$, denotado por $[a, b]$

$$[a, b] := \{\lambda a + (1 - \lambda)b; \quad \lambda \in [0, 1]\},$$

el cual podemos transformar en el intervalo unitario $[0, 1]$, considerando la aplicación afín

$$\lambda : [a, b] \to [0, 1]$$

$$t \longmapsto \lambda(t) = \frac{t - a}{b - a}. \tag{1.1}$$

Ahora, utilizando el Teorema Binomial, podemos representar la unidad como

$$1 = \big((1 - \lambda) + \lambda\big)^n = \sum_{i=0}^{n} \binom{n}{i} (1 - \lambda)^{n-i} \lambda^i.$$

Los términos de esta partición de la unidad son nada menos que los polinomios básicos de Bernstein con respecto al intervalo $[0, 1]$. Junto con la transformación afín anterior, obtenemos los polinomios básicos de Bernstein con respecto al intervalo $[a, b]$.

**Definición 1.2.** *Se conoce como el $i$-ésimo polinomio básico de Bernstein $B_i^n \in \mathbb{P}_n^d$, de grado $n$ con respecto al intervalo $[0, 1]$ a:*

$$B_i^n(\lambda) = \binom{n}{i} (1 - \lambda)^{n-i} \lambda^i \quad 0 \leq \lambda \leq 1, \tag{1.2}$$

*donde los coeficientes binomiales vienen dados por*

$$\binom{n}{i} = \begin{cases} \frac{n!}{i!(n-i)!} & si \quad 0 \leq i \leq n \\ 0 & en\ otro\ caso. \end{cases}$$

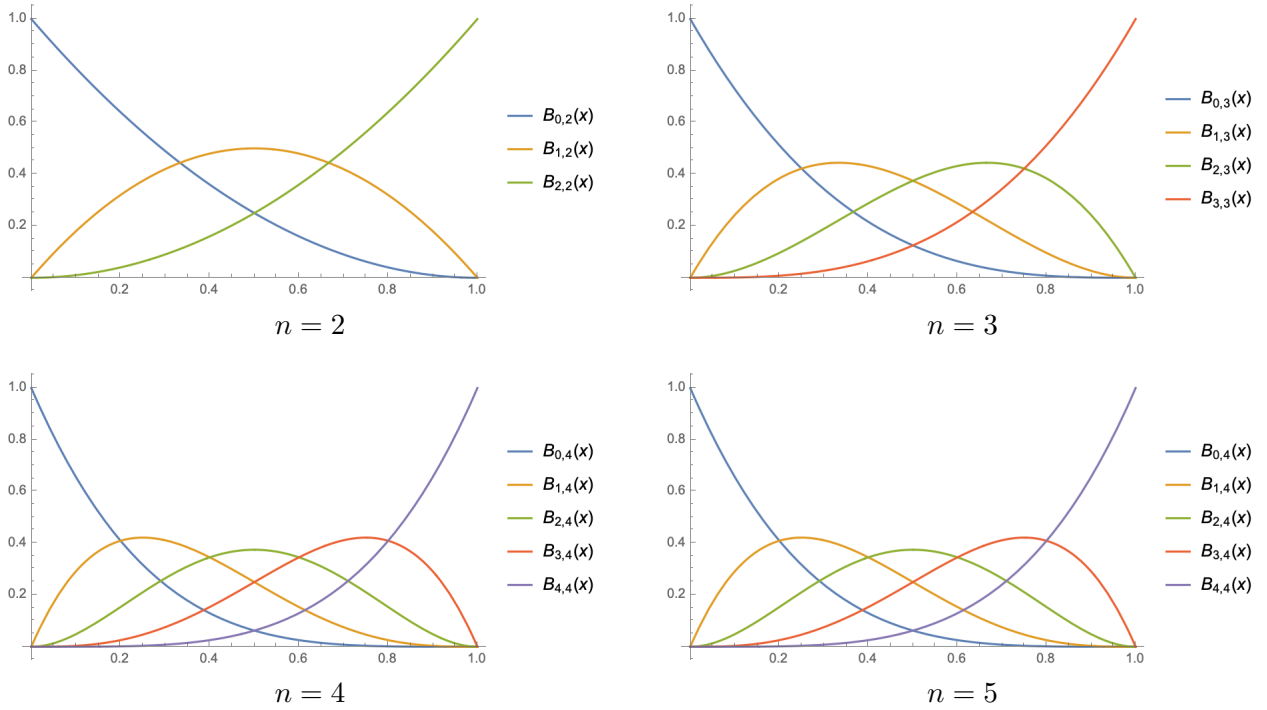



Figura 1.1: Gráficas de algunos polinomios básicos de Bernstein para $n = 2, 3, 4, 5$.

**Definición 1.3.** *Se define el $i$-ésimo polinomio básico de Bernstein de grado $n$ con respecto al intervalo $[a, b]$, con $a \leq b$, $a, b \in \mathbb{R}$ como*

$$B_i^n(t; a, b) = B_i^n(\lambda(t)) = \binom{n}{i}\Big(1 - \frac{t-a}{b-a}\Big)^{n-i}\Big(\frac{t-a}{b-a}\Big)^i$$

$$= \frac{1}{(b-a)^n}\binom{n}{i}(b-t)^{n-i}(t-a)^i \tag{1.3}$$

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

**Definición 1.4.** *Se conoce como el $n$-ésimo polinomio de Bernstein a la combinación lineal de los polinomios básicos de Bernstein de grado $n$ en un intervalo $[a, b]$:*

$$B_n(t) = \sum_{i=0}^{n} \beta_i B_i^n(t) \quad para \quad t \in \mathbb{R}, \quad a \leq t \leq b, \tag{1.4}$$

*con $\beta_i \in \mathbb{R}$.*

Por último, vamos a introducir un concepto similar asociado a una función $f$ dada, que se utilizará más adelante para justificar la aproximación de la función $f$ por polinomios de Bernstein.

**Definición 1.5.** *El polinomio de Bernstein de una función $f$ definida en $[a, b]$ se define de la siguiente manera:*

$$B_n(t; f) = \sum_{i=0}^{n} f\left(\frac{i}{n}\right) B_i^n(t) \quad para \quad t \in \mathbb{R}, \quad a \leq t \leq b. \tag{1.5}$$

## 1.2. Propiedades

A continuación, vamos a enunciar y probar los resultados correspondientes a polinomios de Bernstein respecto a $[0, 1]$, pero se podrían generalizar a cualquier intervalo $[a, b]$, dada la aplicación vista anteriormente.

**Teorema 1.1.** *Los polinomios básicos de Bernstein $B_i^n(\lambda)$, con $0 \leq i \leq n$, definidos en la fórmula* (1.2) *cumplen:*

*(i)* $\lambda = 0$ *es una raíz de multiplicidad* $i$ *de* $B_i^n$.

*(ii)* $\lambda = 1$ *es una raíz de multiplicidad* $n - i$ *de* $B_i^n$.

*(iii) Simetría:* $B_i^n(\lambda) = B_{n-i}^n(1 - \lambda)$.

*(iv)* $(1 - \lambda)B_0^n = B_0^{n+1}$ *y* $\lambda B_n^n = B_{n+1}^{n+1}$.

*(v) Relación de recurrencia*

$$B_i^n(\lambda) = \lambda B_{i-1}^{n-1}(\lambda) + (1-\lambda)B_i^{n-1}(\lambda) \tag{1.6}$$

*para* $i = 1, \ldots, n$, $\lambda \in \mathbb{R}$.

*(vi) Los polinomios básicos de Bernstein son no negativos en* $[0,1]$ *y forman una partición de la unidad; es decir,*

$$B_i^n(\lambda) \geq 0 \quad para \quad 0 \leq \lambda \leq 1 \quad y \quad \sum_{i=0}^{n} B_i^n(\lambda) = 1 \quad para \quad \lambda \in \mathbb{R}. \tag{1.7}$$

*(vii) Máximo local:* $B_i^n(\lambda)$ *(con* $i \neq 0$*) tiene un único máximo local en* $\lambda = i/n$.

*(viii) Los polinomios básicos de Bernstein forman una base* $\mathcal{B} := \{B_0^n, \ldots, B_n^n\}$ *de* $\mathbb{P}_n$. *Se la conoce como la base de Bernstein.*

*(ix) Las derivadas de los polinomios básicos de Bernstein* $B_i^n(\lambda)$ *en* $[0,1]$ *vienen dadas por*

$$\frac{d}{d\lambda}B_0^n(\lambda) = -nB_0^{n-1}(\lambda),$$

$$\frac{d}{d\lambda}B_i^n(\lambda) = n\Big(B_{i-1}^{n-1}(\lambda) - B_i^{n-1}(\lambda)\Big) \quad i = 1, \ldots, n-1, \tag{1.8}$$

$$\frac{d}{d\lambda}B_n^n(\lambda) = nB_{n-1}^{n-1}(\lambda).$$

*Demostración.* Vamos a ir probando cada uno de estos apartados.

(i) y (ii) son triviales.

(iii) Se demuestra usando la Definición 1.2,

$$B_{n-i}^n(1-\lambda) = \binom{n}{n-i}\big(1-(1-\lambda)\big)^{n-(n-i)}(1-\lambda)^{n-i} = \binom{n}{i}(1-\lambda)^{n-i}\lambda^i = B_i^n(\lambda).$$

(iv) $B_0^n(\lambda) = (1-\lambda)^n$ y $B_n^n(\lambda) = \lambda^n$. Entonces,

$$B_0^{n+1}(\lambda) = (1-\lambda)^{n+1} = (1-\lambda)B_0^n(\lambda).$$

$$B_{n+1}^{n+1}(\lambda) = \lambda^{n+1} = \lambda B_n^n(\lambda).$$

(v) La relación de recurrencia viene dada aplicando la propiedad sobre los números combinatorios

$$\binom{n}{i} = \binom{n-1}{i-1} + \binom{n-1}{i}.$$

Así,

$$\lambda B_{i-1}^{n-1}(\lambda) + (1-\lambda)B_i^{n-1}(\lambda) = \binom{n-1}{i-1}(1-\lambda)^{n-i}\lambda^i + \binom{n-1}{i}(1-\lambda)^{n-i}\lambda^i$$

$$= (1-\lambda)^{n-i}\lambda^i\left[\binom{n-1}{i-1} + \binom{n-1}{i}\right] = (1-\lambda)^{n-i}\lambda^i\binom{n}{i} = B_i^n(\lambda).$$

(vi) Es claro por (1.2) y por el teorema del binomio de Newton.

(vii) Una condición necesaria para la existencia de extremo relativo en el dominio de una función es que éste anule su primera derivada. Para $i = 1, \ldots, n$,

$$\frac{d}{d\lambda}B_i^n(\lambda) = \binom{n}{i}(1-\lambda)^{n-i-1}\lambda^{i-1}(i-n\lambda) = 0 \Longleftrightarrow \begin{cases} \lambda = 1 \\ \lambda = 0 \\ \lambda = \frac{i}{n}. \end{cases}$$

Estudiemos ahora el signo que toman los puntos críticos hallados en la segunda derivada, que naturalmente existe.

$$\frac{d}{d\lambda}\left(\frac{d}{d\lambda}B_i^n(\lambda)\right) = \binom{n}{i}\Big(i^2 + (n-1)n\lambda^2 + i\big(-1-2(n-1)\big)\lambda\Big)(1-\lambda)^{n-i-2}\lambda^{i-2}.$$

Si dado un valor de $\lambda$, el signo de la segunda derivada en dicho punto es positivo, estamos ante un mínimo relativo. Si por el contrario, el signo es negativo, se trata de un máximo relativo. Ahora bien,

$$\begin{cases} \dfrac{d^2B_i^n}{d\lambda^2}(1) = 0, \\ \dfrac{d^2B_i^n}{d\lambda^2}(0) = 0, \\ \dfrac{d^2B_i^n}{d\lambda^2}\left(\dfrac{i}{n}\right) < 0. \end{cases}$$

Por lo tanto, en $\lambda = i/n$ se alcanza el máximo relativo, que además es absoluto, ya que, evaluando en los extremos del intervalo de definición ($[0,1]$) se tiene que

$$\begin{cases} B_i^n(1) = 0 \\ B_i^n(0) = 0 \\ B_i^n\big(\frac{i}{n}\big) \geq 0. \end{cases}$$

(viii) Para verlo, basta con probar que los $n+1$ polinomios $B_i^n(\lambda)$ son linealmente independientes. Está claro que en un espacio vectorial de dimensión $n+1$, un conjunto de $n+1$ vectores linealmente independientes también es generador.
Veamos pues que, para $b_0, \ldots, b_n \in \mathbb{R}$,

$$\sum_{i=0}^{n} b_i B_i^n(\lambda) = 0 \Leftrightarrow b_0 = \cdots = b_n = 0.$$

La implicación a izquierda es obvia. Para probar la implicación a derecha, utilizaremos el Teorema del Binomio de Newton, reescribiendo $B_i^n(\lambda)$ de la siguiente forma:

$$B_i^n(\lambda) = \binom{n}{i}(1-\lambda)^{n-i}\lambda^i = \binom{n}{i}\lambda^i\sum_{j=0}^{n-i}(-1)^j\binom{n-i}{j}\lambda^j = \sum_{j=0}^{n-i}(-1)^j\binom{n}{i}\binom{n-i}{j}\lambda^{i+j}$$

$$= \sum_{j=i}^{n} (-1)^{j-i} \binom{n}{i} \binom{n-i}{j-i} \lambda^j = \sum_{j=i}^{n} (-1)^{j-i} \binom{n}{j} \binom{j}{i} \lambda^j, \quad 0 \leq \lambda \leq 1.$$

Por hipótesis,

$$0 = \sum_{i=0}^{n} b_i B_i^n(\lambda) = b_0 B_0^n(\lambda) + b_1 B_1^n(\lambda) + \cdots + b_n B_n^n(\lambda) = b_0 \sum_{j=0}^{n} (-1)^j \binom{n}{j} \binom{j}{0} \lambda^j$$

$$+ b_1 \sum_{j=1}^{n} (-1)^{j-1} \binom{n}{j} \binom{j}{1} \lambda^j + b_2 \sum_{j=2}^{n} (-1)^{j-2} \binom{n}{j} \binom{j}{2} \lambda^j + \cdots + b_n \sum_{j=n}^{n} (-1)^{j-n} \binom{n}{j} \binom{j}{n} \lambda^j$$

$$= \sum_{i=0}^{n} \lambda^i \left[ \sum_{j=0}^{i} (-1)^{i-j} \binom{n}{i} \binom{i}{j} b_j \right].$$

Ahora, la base de potencias es un conjunto linealmente independiente, por lo que necesariamente se debe dar

$$\begin{cases} b_0 = 0, \\ \sum_{j=0}^{1} b_j \binom{n}{1} \binom{1}{j} = 0, \\ \sum_{j=0}^{2} b_j \binom{n}{2} \binom{2}{j} = 0, \\ \vdots \\ \sum_{j=0}^{n} b_j \binom{n}{n} \binom{n}{j} = 0. \end{cases}$$

lo cual implica, mediante una sustitución progresiva, que

$$b_0 = b_1 = \cdots = b_n = 0.$$

$b_0$ es claramente cero, por la primera ecuación; llevándolo a la segunda ecuación, tenemos que $b_1 = 0$; sustituyendo en la tercera ecuación obtenemos que $b_2 = 0$, y así sucesivamente.

(ix) es inmediata derivando la Definición 1.2 y empleando la Propiedad (iv), pues

$$\frac{d}{d\lambda} B_i^n(\lambda) = \binom{n}{i} \Big( i(1-\lambda)^{n-i} \lambda^{i-1} - (n-i)(1-\lambda)^{n-1-i} \lambda^i \Big) \quad i = 0, \ldots, n.$$

□

En el siguiente resultado vamos a enunciar una proposición relacionada con la forma de la envolvente de un polinomio de Bernstein. Este problema fue propuesto como "*Problem 10990*" en la revista *American Mathematical Monthly, 110, no.1 (2003)*. Su demostración fue realizada por el profesor Rick Mabry y se encuentra en su página personal de la Louisiana State University, véase [13].

Recordemos que, en geometría, una envolvente de una familia o haz de curvas en el plano es una curva tangente a cada elemento, en algún punto. Estos puntos de tangencia, que son no singulares, conforman la envolvente completa.

**Proposición 1.1.** *La expresión de la envolvente $f_n(t)$ de los polinomios de Bernstein $B_n(t)$ para $i = 0, 1 \ldots, n$ viene dada por*

$$f_n(t) = \frac{1}{\sqrt{2\pi nt(1-t)}}.$$

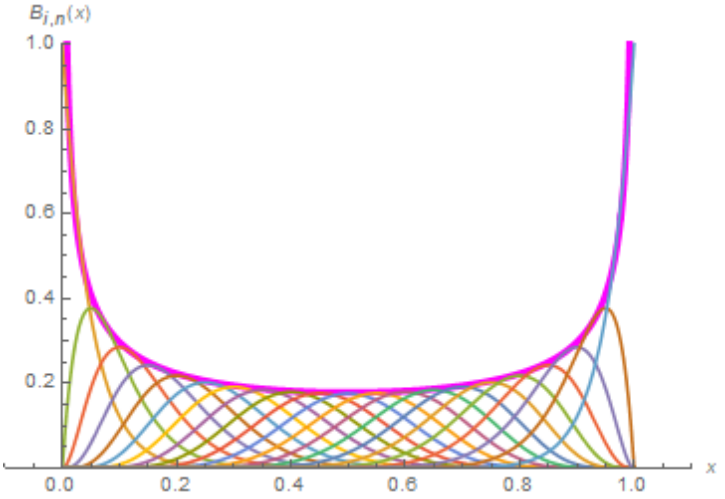



Figura 1.2: Gráfico de la envolvente de los polinomios de Bernstein para $n = 20$.

**Teorema 1.2.** *(Elevación del grado) Cualquier polinomio básico de Bernstein de grado menor que n puede expresarse como una combinación lineal de polinomios básicos de Bernstein de grado n. En particular, cualquier polinomio de Bernstein de grado $n-1$ puede escribirse como combinación lineal de polinomios de Bernstein de grado n.*

*Demostración.* Primero, notemos que

$$\lambda B_i^n(\lambda) = \binom{n}{i}\lambda^{i+1}(1-\lambda)^{n-i} = \binom{n}{i}\lambda^{i+1}(1-\lambda)^{(n+1)-(i+1)} = \frac{\binom{n}{i}}{\binom{n+1}{i+1}}B_{i+1}^{n+1}(\lambda) = \frac{i+1}{n+1}B_{i+1}^{n+1}(\lambda).$$

Además,

$$(1-\lambda)B_i^n(\lambda) = \binom{n}{i}\lambda^i(1-\lambda)^{n+1-i} = \frac{\binom{n}{i}}{\binom{n+1}{i}}B_i^{n+1}(\lambda) = \frac{n-i+1}{n+1}B_{i+1}^{n+1}(\lambda).$$

Finalmente,

$$\frac{1}{\binom{n}{i}}B_i^n(\lambda) + \frac{1}{\binom{n}{i+1}}B_{i+1}^n(\lambda) = \lambda^i(1-\lambda)^{n-i} + \lambda^{i+1}(1-\lambda)^{n-(i+1)}$$

$$= \lambda^i(1-\lambda)^{n-i-1}\big[(1-\lambda)+\lambda\big] = \lambda^i(1-\lambda)^{n-i-1} = \frac{1}{\binom{n-1}{i}}B_i^{n-1}(\lambda).$$

Usando esta última ecuación, podemos escribir un polinomio de Bernstein arbitrario en término de polinomios de Bernstein de orden superior. Esto es,

$$B_i^{n-1}(\lambda) = \binom{n-1}{i}\left[\frac{1}{\binom{n}{i}}B_i^n(\lambda) + \frac{1}{\binom{n}{i+1}}B_{i+1}^n(\lambda)\right]$$

$$= \binom{n-i}{n} B_i^n(\lambda) + \binom{i+1}{n} B_{i+1}^n(\lambda),$$

que expresa un polinomio de Bernstein de orden $n-1$ en términos de combinaciones lineales de polinomios de Bernstein de grado $n$.

□

Para finalizar esta sección, daremos una demonstración constructiva del Teorema de Weierstrass, uno de los más importantes en teoría de aproximación, y veremos el papel que juegan estos polinomios en él.

Karl Weierstrass, el padre del análisis moderno, probó el Teorema de aproximación de Weierstrass en 1885, a la edad de 70 años. Sin embargo, durante el siglo XX emergieron otras demostraciones alternativas. Entre ellas, la de Sergei Bernstein de 1911, en la cual se da un algoritmo explícito para aproximar funciones usando sus propios polinomios.

Comenzaremos presentando un lema previo, del cual haremos uso en la demostración posterior.

**Lema 1.1.** *Para $\lambda \in [0,1]$, se cumplen las siguientes identidades:*

$$\sum_{i=0}^{n} \frac{i}{n} B_i^n(\lambda) = \lambda. \tag{1.9}$$

$$\sum_{i=0}^{n} \left(\frac{i}{n}\right)^2 B_i^n(\lambda) = \left(\frac{n-1}{n}\right)\lambda^2 + \frac{\lambda}{n}. \tag{1.10}$$

*Demostración.* Vamos a probar la primera de estas igualdades.

$$\sum_{i=0}^{n} \frac{i}{n} B_i^n(\lambda) = \sum_{i=1}^{n} \frac{i}{n} \binom{n}{i} (1-\lambda)^{n-i} \lambda^i$$

$$= \sum_{i=1}^{n} \binom{n-1}{i-1} (1-\lambda)^{n-i} \lambda^{i-1} \lambda = \lambda \sum_{j=0}^{n-1} \binom{n-1}{j} (1-\lambda)^{n-1-j} \lambda^j$$

$$= \lambda \sum_{j=0}^{n-1} B_j^{n-1}(\lambda) = \lambda.$$

Para probar la segunda igualdad, haremos uso del Binomio de Newton. Dado que

$$(p+q)^n = \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k},$$

podemos derivar a ambos lados de esta igualdad respecto a $p$ para obtener

$$n(p+q)^{n-1} = \sum_{k=0}^{n} k \binom{n}{k} p^{k-1} q^{n-k},$$

es decir,

$$(p+q)^{n-1} = \sum_{k=0}^{n} \frac{k}{n} \binom{n}{k} p^{k-1} q^{n-k}.$$

Volviendo a derivar a ambos lados,

$$(n-1)(p+q)^{n-2} = \sum_{k=0}^{n} \frac{k}{n}\binom{n}{k}(k-1)p^{k-2}q^{n-k}.$$

Con lo que

$$p^2(p+q)^{n-2} = \sum_{k=0}^{n} \frac{k}{n}\binom{n}{k}\frac{k-1}{n-1}p^k q^{n-k}.$$

Ahora, tomando $p = \lambda$ y $q = 1-\lambda$ obtenemos una expresión para $\lambda^2$:

$$\lambda^2 = \sum_{k=0}^{n} \binom{n}{k}\frac{k(k-1)}{n(n-1)}\lambda^k(1-\lambda)^{n-k}.$$

Volviendo a la ecuación que queremos probar, reescribimos

$$\Big(\frac{n-1}{n}\Big)\lambda^2 + \frac{\lambda}{n} = \frac{n-1}{n}\sum_{k=0}^{n}\binom{n}{k}\frac{k(k-1)}{n(n-1)}\lambda^k(1-\lambda)^{n-k}$$

$$+\frac{1}{n}\sum_{k=0}^{n}\frac{k}{n}\binom{n}{k}\lambda^k(1-\lambda)^{n-k} = \sum_{k=0}^{n}\binom{n}{k}\lambda^k(1-\lambda)^{n-k}\left(\frac{k(k-1)}{n^2}+\frac{k}{n^2}\right)$$

$$= \sum_{k=0}^{n}\binom{n}{k}\lambda^k(1-\lambda)^{n-k}\left(\frac{k}{n}\right)^2.$$

□

**Teorema 1.3.** *(Weierstrass) Dada una función continua $f$ definida en el intervalo $[0,1]$, la secuencia de polinomios*

$$p_n(\lambda) = \sum_{i=0}^{n} f\left(\frac{i}{n}\right) B_i^n(\lambda), \quad n \geq 1$$

*converge uniformemente a $f$.*

*Demostración.* Recordemos que los polinomios de Bernstein forman una partición de la unidad, es decir,

$$\sum_{i=0}^{n} B_i^n(\lambda) = \sum_{i=0}^{n}\binom{n}{i}\lambda^i(1-\lambda)^{n-i} = 1 \quad \text{para} \quad \lambda \in \mathbb{R}, \quad 0 \leq \lambda \leq 1.$$

Por lo tanto, podemos escribir la diferencia entre $f$ y su $n$-ésimo polinomio de Bernstein, teniendo en cuenta la Definición (1.5), como:

$$B_n(\lambda; f) - f(\lambda) = \sum_{i=0}^{n} f\left(\frac{i}{n}\right) B_i^n(\lambda) - \sum_{i=0}^{n} f(\lambda) B_i^n(\lambda) = \sum_{i=0}^{n}\left[f\left(\frac{i}{n}\right) - f(\lambda)\right] B_i^n(\lambda).$$

Ahora, tomamos el supremo con respecto al valor absoluto de la ecuación para así obtener, después de aplicar la desigualdad triangular,

$$||B_n(\cdot; f) - f(\cdot)|| \leq \sup_{0\leq\lambda\leq 1}\left[\sum_{i=0}^{n}\left|f\left(\frac{i}{n}\right) - f(\lambda)\right| B_i^n(\lambda)\right].$$

**Observación 1.1.** *Aquí usamos $B_n(\lambda; f)$ para denotar el polinomio de Bernstein en $\lambda$, y $B_n(\cdot; f)$ para denotar la correspondiente función polinómica.*

Como $f$ es continua en un compacto, por el Teorema de Heine es uniformemente continua (véase [1]). Es decir, dado $\varepsilon > 0$, existe un $\delta > 0$ tal que

$$|x - y| < \delta \Rightarrow |f(x) - f(y)| < \varepsilon, \quad \forall x, y \in [0, 1].$$

Para estimar el supremo, vamos a dividir nuestros términos en los dos conjuntos siguientes:

$$I(\lambda) = \{i;\, 0 \leq i \leq n \quad \text{y} \quad |\lambda - i/n| < \delta\},$$

$$J(\lambda) = \{i;\, 0 \leq i \leq n \quad \text{y} \quad |\lambda - i/n| \geq \delta\}.$$

Teniendo en cuenta la desigualdad triangular y que la norma de una función está acotada por su supremo, combinamos las tres últimas expresiones y obtenemos:

$$||B_n(\cdot; f) - f(\lambda)|| \leq \sup_{0\leq\lambda\leq 1} \left[\sum_{i\in I(\lambda)} B_i^n(\lambda)\right]\varepsilon + \sup_{0\leq\lambda\leq 1} \left[\sum_{i\in J(\lambda)} \left|f\left(\frac{i}{n}\right) - f(\lambda)\right| B_i^n(\lambda)\right]$$

$$\leq \varepsilon + 2||f|| \sup_{0\leq\lambda\leq 1} \left[\sum_{i\in J(\lambda)} B_i^n(\lambda)\right].$$

Como $\left[\lambda - \frac{i}{n}\right]^2 \geq \delta^2$, para $i \in J(\lambda)$ podemos estimar el término de la derecha de la inecuación anterior de la siguiente manera:

$$\sup_{0\leq\lambda\leq 1} \left[\sum_{i\in J(\lambda)} B_i^n(\lambda)\right] \leq \frac{1}{\delta^2} \sup_{0\leq\lambda\leq 1} \left[\sum_{i\in J(\lambda)} \left(\lambda - \frac{i}{n}\right)^2 B_i^n(\lambda)\right]$$

$$\leq \frac{1}{\delta^2} \sup_{0\leq\lambda\leq 1} \left[\sum_{i=0}^{n} \left(\lambda^2 - \frac{2i\lambda}{n} + \frac{i^2}{n^2}\right) B_i^n(\lambda)\right]$$

$$= \frac{1}{\delta^2} \sup_{0\leq\lambda\leq 1} \left[\lambda^2 \sum_{i=0}^{n} B_i^n(\lambda) - 2\lambda \sum_{i=0}^{n} \frac{i}{n} B_i^n(\lambda) + \sum_{i=0}^{n} \left(\frac{i}{n}\right)^2 B_i^n(\lambda)\right].$$

Gracias a las ecuaciones (1.9) y (1.10) del Lema previo, la expresión anterior puede expresarse como

$$\frac{1}{\delta^2} \sup_{0\leq\lambda\leq 1} \left[\lambda^2 - 2\lambda^2 + \left(\frac{n-1}{n}\right)\lambda^2 + \frac{\lambda}{n}\right] = \frac{1}{\delta^2} \sup_{0\leq\lambda\leq 1} \left[\frac{\lambda}{n}(1 - \lambda)\right].$$

Como el máximo de la función $\lambda(1 - \lambda)$ se alcanza para $\lambda = 1/2$, podemos estimar que $\lambda(1 - \lambda) \leq \frac{1}{4}$, y

$$\|B_n(\cdot; f) - f(\lambda)\| \leq \varepsilon + 2\|f\| \frac{1}{\delta^2} \frac{1}{4n} = \varepsilon + \frac{\|f\|}{2n\delta^2}.$$

Tomando su límite superior, obtenemos

$$\limsup_{n\to\infty} \|B_n(\cdot; f) - f(\lambda)\| \leq \varepsilon.$$

Debido a que $\varepsilon$ es arbitrario, concluimos que el límite anterior es igual a cero, lo cual implica que $p_n = B_n(\cdot; f)$ converge uniformemente a $f$.

□

Este resultado es de gran utilidad, pues puede ser generalizado para cualquier valor de $t \in [a, b]$, arbitrario. Si quisiéramos crear un polinomio que se aproxime a los valores de interpolación $(i/n, y_i)$ para $0 \leq i \leq n$, podemos usar $\sum_{i=0}^{n} B_i^n(t) y_i$ en primer lugar, y tras examinar la curva resultante, ajustar los coeficientes para modificar su forma. Cambiar el valor de $y_i$ modificará la curva, principalmente en los alrededores de $i/n$, debido al carácter local de los polinomios $B_i^n$.

Otra forma en la que se pueden usar estos polinomios es a la hora de crear curvas que no solo sean gráficas de una función $f$ de $\mathbb{R}$ en $\mathbb{R}$. Por ejemplo, podemos convertir lo anterior a forma vectorial. Si tenemos $n+1$ vectores $v_0, v_1, \ldots v_n$ en $\mathbb{R}^2$ o $\mathbb{R}^3$, la expresión

$$u(t) = \sum_{i=0}^{n} v_i B_i^n(t) \tag{1.11}$$

tiene sentido, pues el lado derecho de la igualdad es, para cada $t$, una combinación de los vectores $v_i$. Conforme $t$ recorre el intervalo $[a, b]$, el vector $u(t)$ describe una curva en el espacio donde están situados los vectores $v_i$.

Para ilustrar este procedimiento, en la Figura 1.3 hemos tomado seis puntos en el plano y hemos dibujado la curva generada por la expresión (1.11), es decir, por $u(t)$. El conjunto de puntos es el siguiente: $\{(1,1),(2,4),(3,3),(4,1),(4,5),(5,2)\}$. También se muestra la curva $u^*(t)$ obtenida al modificar uno de ellos, ya que de $(3,3)$ se pasa a $(3,4)$.

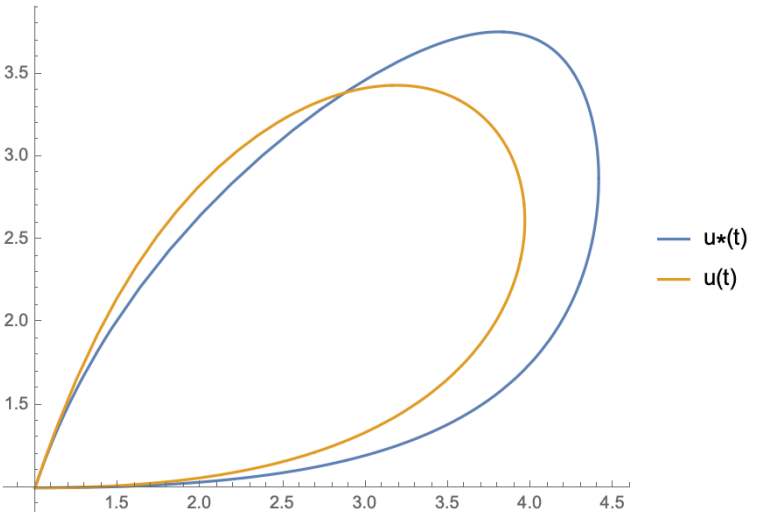



Figura 1.3: Curva generada según el vector $u(t)$ para seis puntos en el plano. Se modifica ligeramente uno de ellos para obtener $u^*(t)$.

# Capítulo 2

# Introducción a las curvas de Bézier

En diseño asistido por ordenador, resulta muy útil tener un procedimiento que produzca una curva que pasa por ciertos puntos de control, o una curva que pueda ser manipulada fácilmente para darle una forma deseada. Esto es, precisamente, en lo que radica la aportación de Bézier-De Casteljau. La teoría que introducimos a continuación se encuentra desarrollada, entre otros, en [7], [9] o [10]. Además, las animaciones han sido creadas en base a [14].

## 2.1. Representación de Bézier de un polinomio de $\mathbb{P}_n^2$

Dado que los polinomios de Bernstein forman una base de los polinomios de grado $n$ o inferior, podemos representar las curvas polinómicas de grado $n$ como combinación de dichos polinomios.

En primer lugar, abordaremos la relación entre los polinomios de Bernstein y las curvas de Bézier.

**Definición 2.1.** *Dado un polinomio $P \in \mathbb{P}_n^2$, se definen los puntos de control, vértices, nodos o puntos de Bézier de $P$ como los coeficientes $b_0, b_1, \ldots, b_n \in \mathbb{R}^2$ de la representación de $P$ en base $\mathcal{B} = \{B_0^n, \ldots, B_n^n\}$ (polinomios de Bernstein), en el intervalo $[a, b]$.*

$$P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b), \quad t \in [a, b].$$

*Decimos que la representación de $P$ en dicha base es la representación de Bézier de $P$. Además, al polígono resultante de unir los puntos de control mediante segmentos se llama polígono de control de $P$.*

Todos estos coeficientes son puntos del plano o espacio afín, dependiendo si la curva $P$ es plana o espacial. Una curva de grado $n$ tiene un polígono de control o de Bézier de $n + 1$ vértices.

**Observación 2.1.** *Como ya hemos indicado con anterioridad, normalmente hablaremos de puntos de control como la representación de $P$ en base $\mathcal{B} = \{B_0^n(\lambda), \ldots, B_n^n(\lambda)\}$, con $B_i^n(\lambda)$ definidos en $[0, 1]$, en vez de en el intervalo genérico $[a, b]$.*

**Definición 2.2.** *Una curva de Bézier definida en el intervalo* $[0,1]$ *(de grado* $n$*, con puntos de control* $b_0, b_1, b_2, \ldots, b_n$*) puede escribirse como*

$$P(t) = \sum_{i=0}^{n} B_i^n(t) b_i = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i b_i \tag{2.1}$$

$$= (1-t)^n b_0 + \binom{n}{1}(1-t)^{n-1} t b_1 + \cdots + \binom{n}{n-1}(1-t) t^{n-1} b_{n-1} + t^n b_n \quad 0 \leq t \leq 1.$$

De esta forma, notamos que las curvas de Bézier quedan definidas en función del $i$-ésimo polinomio de Bernstein de orden $n$.

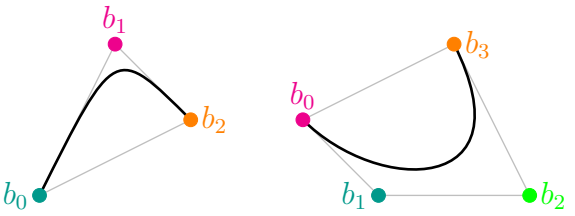



Figura 2.1: Construcción de dos curvas de Bézier mediante TikZ. La primera de ellas pasa por los puntos $b_0$ y $b_2$, teniendo como punto de control $b_1$. La segunda, tiene a $b_1$ y $b_2$ por puntos de control y pasa por $b_0$ y $b_3$.

Sin embargo, vemos en [11] o en [18] que existen múltiples formas de representar una curva de Bézier, alternativas a (2.1), de entre las cuales cabe destacar:

(i) *Suma de potencias de* $t$:

$$P(t) = \sum_{i=0}^{n} t^i \left[ \sum_{j=0}^{i} (-1)^{i-j} \binom{n}{i} \binom{i}{j} b_j \right].$$

(ii) *Operador de diferencias finitas*:

$$P(t) = \sum_{i=0}^{n} \binom{n}{i} t^i \Delta^i b_0, \quad \text{con} \quad \Delta^i b_k = \sum_{j=0}^{i} (-1)^{i-j} \binom{i}{j} b_{k+j}.$$

(iii) *Forma matricial.* Si intentamos expresar una curva de Bézier como producto de vectores básicos,

$$P(t) = \begin{bmatrix} B_0^n(t) & B_1^n(t) & \cdots & B_n^n(t) \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} t^n & t^{n-1} & \cdots & t & 1 \end{bmatrix} M \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}$$

donde $M$ es una matriz $(n+1)(n+1)$, que transforma vectores de la base de potencias en vectores de la base de Bernstein, de tal forma que $M = [m_{ij}]_{i,j=0}^{n}$ y

$$m_{ij} = \begin{cases} (-1)^{n-i-j} \binom{n}{n-i} \binom{n-i}{j}, & 0 \leq i+j \leq n, \\ 0 \quad \text{en otro caso.} \end{cases}$$

## 2.2. Propiedades de las curvas de Bézier

A continuación, vamos a ver algunas de las propiedades de las curvas de Bézier. Varias de ellas han sido descritas ya en la sección anterior. Para profundizar en este apartado, se puede consultar [7].

1. *Vértices* $b_0$ *y* $b_n$. Dos de los vértices del polígono de control tienen una interpretación inmediata. En vista de la expresión de los polinomios de Bernstein, tenemos que $B_i^n(0) = 0$ para cualquier grado $n$, excepto para $i = 0$, para el cual toma el valor $B_0^n(0) = 1$. De igual manera, $B_i^n(1) = 0$, excepto para $i = n$, donde $B_n^n(1) = 1$.
   Entonces, la curva pasa por los vértices $b_0$ y $b_n$,

$$b_0 = \sum_{i=0}^{n} B_i^n(0) b_i, \qquad b_n = \sum_{i=0}^{n} B_i^n(1) b_i.$$

   En particular, son los únicos vértices del polígono de control por los que pasa la curva.

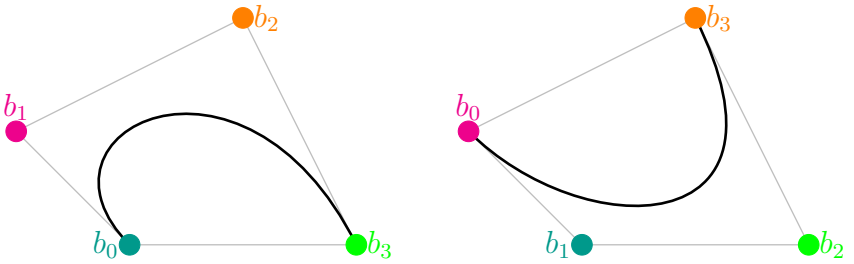



Figura 2.2: Vértices $b_0$ y $b_n$. Una curva de Bézier siempre pasa por sus vértices primero y último.

2. *Simetría*. Si invertimos el orden de los puntos de control, la curva de Bézier resultante es la misma que la original, recorrida en sentido inverso. Es decir,

$$\sum_{j=0}^{n} b_j B_i^n(t) = \sum_{j=0}^{n} b_{n-j} B_i^n(1-t).$$

   Esto se debe que los polinomios de Bernstein verifican la propiedad de simetría $B_i^n(t) = B_i^n(1-t)$. Decimos que son simétricos con respecto a $t$ y a $1-t$.

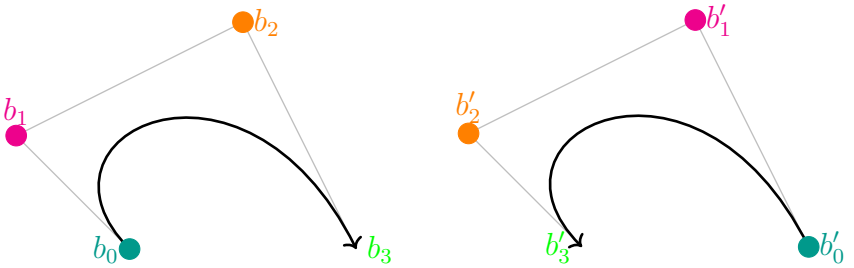



Figura 2.3: Simetría. Para invertir el sentido de una curva de Bézier, basta invertir el orden de su polígono de control.

3. *Pseudo-contol local.* Esta propiedad es, quizás, la de mayor relevancia. El nombre del polígono de control hace referencia a su utilidad, pues sirve para controlar la forma de la curva. Sin embargo, este control no es local, pues desplazando solo un vértice, la curva entera quedará desplazada. Esto se debe a que el máximo del $i$-ésimo polinomio de Bernstein se encuentra en $t = i/n$.
   En otras palabras, si movemos un solo punto de control $b_i$ a $b'_i$, todos los puntos de la curva se moverán respecto de la curva original en la dirección del vector $b_i\vec{b}'_i$. Esta modificación no afecta a todos los puntos de la misma manera, ya que se acentúa en la región de la curva más cercana al valor $i/n$.

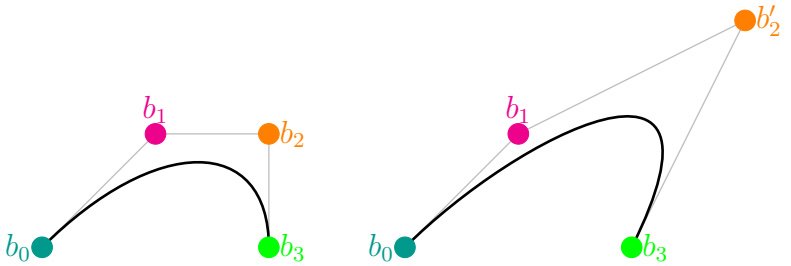



Figura 2.4: Control local. Al mover el vértice $b_2$ a $b'_2$, se deforma en mayor medida la parte más próxima a él.

4. *Envoltura convexa.* Una curva de Bézier siempre está completamente contenida en la envoltura convexa de los puntos de control. De forma intuitiva, tomando la expresión vectorial de una curva de Bézier, la curva $u(t)$ debe estar contenida en la envoltura convexa de los vectores $v_i$, pues $u(t)$ es una combinación convexa lineal de los $v_i$.
   Recordemos que la la envoltura convexa de $A \in \mathbb{R}^d$ es el conjunto de todas las combinaciones convexas de los puntos $a_1, \ldots, a_m$ de $A$, o el menor convexo que contiene a $A$:

$$co(A) = \bigcap \{B \subset \mathbb{R}^d | B \text{ convexo, con } A \subset B\}$$

$$= \Big\{ \sum_{i=1}^{m} \lambda_i a_i | m \in \mathbb{N}, \quad a_i \in A, \quad \lambda_i \geq 0, \quad \sum_{i=1}^{m} \lambda_i = 1 \Big\}.$$

   La prueba de esta propiedad es inmediata a partir de la propiedad de partición de la unidad de los polinomios de Bernstein y de la representación de Bézier de $P$.

5. *Invarianza afín.* Una curva de Bézier no varía al aplicarle una transformación afín, una homotecia, traslación o rotación. Por lo tanto, dada una curva de Bézier $P(t)$ de los puntos $\{b_i\}_{i=0}^n$ y dada una función afín $f : \mathbb{R}^2 \to \mathbb{R}^2$, entonces $f(P(t))$ es la curva de Bézier de los puntos $\{f(b_i)\}_{i=0}^n$.

6. *Disminución de la variación.* El número de intersecciones entre la curva de Bézier y una recta cualquiera es menor o igual al número de intersecciones de la misma con su polígono de control.

## 2.3. Algoritmo de De Casteljau

Aunque la base teórica de las Curvas de Bézier se formuló en 1912, no fue hasta 1959 que se introdujeron al diseño, cuando Paul de Casteljau desarrolló un algoritmo estable numéricamente para evaluar las curvas, y fue el primero en implementarlo en el diseño asistido por computador de la automovilística francesa *Citroën*.

En él, se construyen curvas de Bézier mediante combinaciones convexas de un número finito y ordenado de puntos de control, de forma que la curva une el primero y el último de los puntos, y utiliza el resto para producir una deformación suave de la misma. Además, el mismo algoritmo puede usarse para subdividir la curva de Bézier en segmentos. Aplicando esta partición a los segmentos, la secuencia de los polígonos de Bézier converge de forma muy rápida a la curva (exponencialmente, cuando se divide el intervalo por la mitad). Comenzaremos la sección dando una definición de los polinomios parciales de $P$.

**Definición 2.3.** *Sea $P(\lambda) = \sum_{i=0}^{n} b_i B_i^n(\lambda)$ un polinomio en representación de Bézier con respecto a $[0,1]$. Definimos los polinomios parciales $b_i^k \in \mathbb{P}_k^d$ de $P$ para $i = 0, \ldots, n-k$ como*

$$b_i^k(\lambda) := \sum_{j=0}^{k} b_{i+j} B_j^k(\lambda) = \sum_{j=i}^{i+k} b_j B_{j-i}^k(\lambda).$$

*Para un polinomio $P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b)$ en representación de Bézier con respecto a $[a, b]$, los polinomios parciales $b_i^k$ se definen de manera similar,*

$$b_i^k(t; a, b) := b_i^k(\lambda(t)) = \sum_{j=0}^{k} b_{i+j} B_j^k(t; a, b).$$

Así, el polinomio parcial $b_i^k \in \mathbb{P}_k^d$ es precisamente el polinomio de Bézier, determinado por los puntos $b_i, \ldots b_{i+k}$. Si no hay lugar a confusión, escribiremos $b_i^k(t)$ en lugar de $b_i^k(t; a, b)$.

En particular, se cumple que $b_0^n(t) = P(t)$ es el polinomio original, y los $b_i^0(t) = b_i$ son sus puntos de Bézier para todo $t \in \mathbb{R}$. Además, en los puntos correspondientes a los extremos, se tiene que $b_i^k(a) = b_i$ y $b_i^k(b) = b_{i+k}$.

Presentamos ahora una relación de recurrencia que constituye la base del algoritmo de De Casteljau.

**Lema 2.1.** *Los polinomios parciales $b_i^k(t)$ de $P(t) = \sum_{i=0}^{n} b_i B_i^n(t; a, b)$ satisfacen la relación*

$$b_i^k = (1-\lambda) b_i^{k-1} + \lambda b_{i+1}^{k-1}, \quad \text{con} \quad \lambda = \lambda(t) = \frac{t-a}{b-a}$$

*para $k = 0, \ldots, n$ y $i = 0, \ldots, n-k$.*

*Demostración.* Introduciendo la relación de recurrencia dada por 1.6 en la definición de los polinomios parciales obtenemos

$$b_i^k = \sum_{j=0}^{k} b_{i+j} B_j^k = b_i B_0^k + b_{i+k} B_k^k + \sum_{j=1}^{k-1} b_{i+j} B_j^k$$

$$= b_i(1-\lambda)B_0^{k-1} + b_{i+k}\lambda B_{k-1}^{k-1} + \sum_{j=1}^{k-1} b_{i+j}\big((1-\lambda)B_j^{k-1} + \lambda B_{j-1}^{k-1}\big)$$

$$= \sum_{j=0}^{k-1} b_{i+j}(1-\lambda)B_j^{k-1} + \sum_{j=1}^{k} b_{i+j}\lambda B_{j-1}^{k-1} = (1-\lambda)b_i^{k-1} + \lambda b_{i+1}^{k-1}.$$

□

Debido a que $b_i^0(t) = b_i$, $i = 0, \ldots, n$, $t \in [a, b]$ podemos computar el valor de $P(t) = b_0^n(t)$ mediante sucesivas combinaciones convexas de los puntos de Bézier. Este proceso se conoce con el nombre de algoritmo de De Casteljau, cuyo esquema presentamos a continuación.

$$\begin{array}{ccccccccc}
b_n & = & b_n^0 & & & & & & \\
 & & & \searrow & & & & & \\
b_{n-1} & = & b_{n-1}^0 & \to & b_{n-1}^1 & & & & \\
 & \vdots & & & & \ddots & & & \\
b_1 & = & b_1^0 & \to & \ldots & \to & b_1^{n-1} & & \\
 & & & \searrow & & & & \searrow & \\
b_0 & = & b_0^0 & \to & \ldots & \to & b_0^{n-1} & \to & b_0^n
\end{array}$$

En la Figura 2.5, hemos construido una curva de Bézier de grado tres mediante sucesivas combinaciones convexas de los vértices iniciales, con $t = 0.5$. Partiendo de los nodos $b_0, b_1, b_2$ y $b_3$ (en azul), se han determinado en un primer paso los vértices $b_0^1, b_1^1, b_2^1$ y $b_3^1$ (en rosa). Tras ello, se computan los puntos $b_0^2$ y $b_1^2$ (en naranja). La iteración final consiste en determinar el punto $b_0^3$, que como observamos, es un punto de la curva de Bézier original.

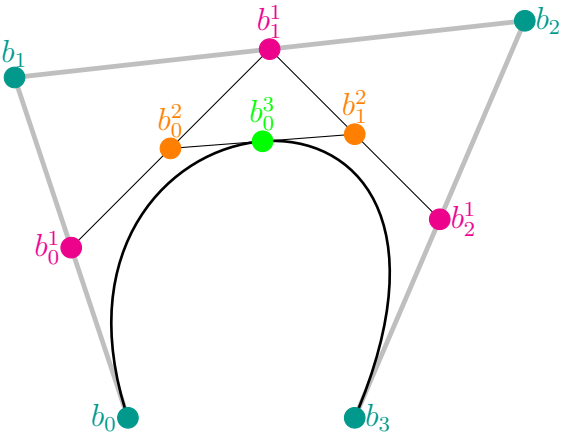



Figura 2.5: Algoritmo de De Casteljau para $t = 0.5$. Construcción del caso cúbico mediante sus polinomios parciales.

En definitiva, puede que este sea el algoritmo más fundamental en el campo del diseño de curvas y superficies. La interacción entre el álgebra y la geometría hacen que, de una construcción geométricamente intuitiva, se dé lugar a una sólida teoría. En la siguiente sección veremos que las derivadas de $P$ juegan un papel fundamental en ella.

## 2.4. Derivada de una curva de Bézier

La derivada una curva de Bézier, es decir, el campo tangente a la curva, se obtiene de manera sencilla gracias a las propiedades de los polinomios de Bernstein. Además, no es descabellado sospechar que la forma de la curva esté estrechamente relacionada con el polígono de Bézier. En esta sección daremos un significado geométrico a los puntos de control y estudiaremos los resultados pertinentes para comprobar que las derivadas de $P$ están detrás de, entre otros, el esquema de De Casteljau.

**Teorema 2.1.** *(Derivada k-ésima de P) Sea $P(\lambda) = \sum_{i=0}^{n} b_i B_i^n(\lambda)$ un polinomio en representación de Bézier con respecto a $[0,1]$. Entonces, la derivada $k-$ésima de $P$ satisface*

$$P^{(k)}(\lambda) = \frac{n!}{(n-k)!}\sum_{i=0}^{n-k}\Delta^k b_i B_i^{n-k}(\lambda), \quad con \quad \begin{cases}\Delta b_i = b_{i+1} - b_i \\ \Delta^k b_i = \Delta^{k-1}b_{i+1} - \Delta^{k-1}b_i\end{cases}$$

*operador de diferencias progresivas.*

*Demostración.* Se prueba por inducción sobre $k$.

- Para el caso $k = 1$, nos ayudaremos de la ecuación (1.8), de tal manera que

$$P'(\lambda) = -nb_0B_0^{n-1}(\lambda) + \sum_{i=0}^{n-1} nb_i\Big(B_{i-1}^{n-1}(\lambda) - B_i^{n-1}(\lambda)\Big) + nb_nB_{n-1}^{n-1}(\lambda)$$

$$= n\Bigg[b_nB_{n-1}^{n-1}(\lambda) - b_0B_0^{n-1}(\lambda) + b_1B_0^{n-1}(\lambda) - b_{n-1}B_{n-1}^{n-1}(\lambda)$$

$$+\sum_{i=1}^{n-2}\Big(b_{i+1} - b_i\Big)B_i^{n-1}(\lambda)\Bigg] = n\sum_{i=0}^{n-1}(b_{i+1} - b_i)B_i^{n-1}(\lambda) = n\sum_{i=0}^{n-1}\Delta b_i B_i^{n-1}(\lambda).$$

- Supongamos que $P^{(k)}(\lambda) = \dfrac{n!}{(n-k)!}\displaystyle\sum_{i=0}^{n-k}\Delta^k b_i B_i^{n-k}(\lambda)$ y probemos que

$$P^{(k+1)}(\lambda) = \frac{n!}{(n-k-1)!}\sum_{i=0}^{n-k-1}\Delta^{k+1}b_i B_i^{n-k-1}(\lambda).$$

Derivando $P^{(k)}$ de forma análoga a la anterior,

$$P^{(k+1)}(\lambda) = \frac{n!}{(n-k)!}\sum_{i=0}^{n-k}\Delta^k b_i\frac{d}{d\lambda}B_i^{n-k}(\lambda) = \frac{n!}{(n-k)!}\Bigg[-(n-k)\Delta^k b_0B_0^{n-k-1}(\lambda)$$

$$+(n-k)\Delta^k b_{n-k}B_{n-k-1}^{n-k-1}(\lambda) + (n-k)\sum_{i=0}^{n-k-1}\Delta^k b_i\Big(B_{i-1}^{n-k-1}(\lambda) - B_i^{n-k-1}(\lambda)\Big)\Bigg]$$

$$= \frac{n!}{(n-k-1)!}\Bigg[-\Delta^k b_0B_0^{n-k-1}(\lambda) + \Delta^k b_{n-k}B_{n-k-1}^{n-k-1}(\lambda)$$

$$+\sum_{i=0}^{n-k-1}\Delta^k b_i\Big(B_{i-1}^{n-k-1}(\lambda)-B_i^{n-k-1}(\lambda)\Big)\Bigg]$$

$$=\frac{n!}{(n-k-1)!}\Bigg[\sum_{i=0}^{n-k-1}\Big(\Delta^k b_{i+1}-\Delta^k b_i\Big)B_i^{n-k-1}(\lambda)\Bigg]$$

$$=\frac{n!}{(n-k-1)!}\sum_{i=0}^{n-k-1}\Delta^{k+1}b_i B_i^{n-k-1}(\lambda).$$

□

**Corolario 2.1.** *Para los puntos extremos $\lambda=0,1$ se obtienen los valores*

$$P^{(k)}(0)=\frac{n!}{(n-k)!}\Delta^k b_0 \quad y \quad P^{(k)}(1)=\frac{n!}{(n-k)!}\Delta^k b_{n-k},$$

*en particular,*

*(i)* $P(0)=b_0$ *y* $P(1)=b_n$.

*(ii)* $P'(0)=n(b_1-b_0)$ *y* $P'(1)=n(b_n-b_{n-1})$.

*(iii)* $P''(0)=n(n-1)(b_2-2b_1+b_0)$ *y* $P''(1)=n(n-1)(b_n-2b_{n-1}+b_{n-2})$.

Gracias al resultado que acabamos de probar, sabemos que, en un punto extremo, la curva está determinada hasta la derivada $k$-ésima por los $k$ puntos de control más próximos. A continuación, veremos que la derivada $k$-ésima también se obtiene a partir de la columna $n-k$ del esquema de De Casteljau.

**Teorema 2.2.** *Sea $P(\lambda)=\sum_{i=0}^{n}b_iB_i^n(\lambda)$ un polinomio en representación de Bézier con respecto a $[0,1]$. Entonces, las derivadas $P^{(k)}(\lambda)$ para $k=0,\ldots,n$ pueden obtenerse a partir de los polinomios parciales $b_i^k(\lambda)$, a través de la relación*

$$P^{(k)}(\lambda)=\frac{n!}{(n-k)!}\Delta^k b_0^{n-k}(\lambda), \quad con \quad \Delta b_i^k=b_{i+1}^k-b_i^k.$$

*Demostración.* De acuerdo con el teorema anterior y dado que el operador de diferencias progresivas conmuta con la suma,

$$P^{(k)}(\lambda)=\frac{n!}{(n-k)!}\sum_{i=0}^{n-k}\Delta^k b_i B_i^{n-k}(\lambda)=\frac{n!}{(n-k)!}\Delta^k\sum_{i=0}^{n-k}b_iB_i^{n-k}(\lambda)$$

$$=\frac{n!}{(n-k)!}\Delta^k b_0^{n-k}(\lambda).$$

□

**Corolario 2.2.** *En particular, para cualquier valor de* $t \in [a, b]$

*(i)* $P(t) = b_0^n$.

*(ii)* $P'(t) = n(b_1^{n-1} - b_0^{n-1})$.

*(iii)* $P''(t) = n(n-1)(b_2^{n-2} - 2b_1^{n-2} + b_0^{n-2})$.

Hasta ahora, solo hemos considerado la representación de Bézier de un único polinomio respecto a un intervalo de referencia fijo. Aquí, la pregunta sobre cómo se transforman los puntos de control a medida que cambiamos el intervalo de referencia, queda abierta. Sería interesante saber si se pueden juntar varios trozos de curva de forma continua o suave, además de la posibilidad de subdividir las curvas (subdividiendo el intervalo y computando los puntos de Bézier para los subintervalos).

En virtud de la Propiedad 4 de las curvas de Bézier, la curva está contenida en la envoltura convexa de los puntos de control. Entonces, está claro que los polígonos de Bézier se aproximan más a la curva a medida que se produce una subdivisión refinada. Las cuestiones anteriores no solo están relacionadas, sino que se pueden resolver de forma favorable en el contexto de las técnicas de Bézier, siendo su factor común los polinomios parciales.

En realidad, el Corolario 2.1 también implica el recíproco: los valores de $P$ hasta la $k$-ésima derivada en $\lambda = 0$ determinan los puntos de Bézier $b_0, \ldots, b_k$. Esto también se cumple con los polinomios parciales $b_0^0(\lambda), \ldots b_0^k(\lambda)$, aunque no lo probaremos.

# Capítulo 3

# Transformaciones en una curva de Bézier

En este capítulo vamos a ahondar en las transformaciones que más se aplican a las curvas de Bézier. Hemos seguido [7], [11] y [18] como referencia. Empezaremos por la elevación del grado, continuaremos por la subdivisión y finalmente veremos conexión de dos segmentos de curvas de Bézier.

## 3.1. Elevación del grado en una curva de Bézier

Supongamos que intentamos realizar un diseño mediante una curva de Bézier $P(t)$ de grado $n$, sin obtener un buen ajuste. Después de modificar el polígono de control, puede ocurrir que dicha curva no sea lo suficientemente flexible como para modelar la forma deseada. Consideremos el problema de elevar formalmente el grado de un segmento de curva de Bézier sin que su trazo varíe, con el fin de no desperdiciar el trabajo realizado.

Este proceso se basa en buscar una curva con vértices $b_0^{(1)}, b_1^{(1)}, \ldots, b_{n+1}^{(1)}$ que describan la misma curva que el polígono original $b_0, b_1, \ldots, b_n$. Se comienza reescribiendo la curva de grado $n$ de la siguiente forma:

$$\begin{aligned} P(t) &= (1-t)P(t) + tP(t) \\ &= \sum_{i=0}^{n} b_i \binom{n}{i} t^i (1-t)^{n+1-i} + \sum_{i=0}^{n} b_i \binom{n}{i} t^{i+1}(1-t)^{n-i} \\ &= \sum_{i=0}^{n} b_i \binom{n+1}{i} \frac{n+1-i}{n+1} t^i (1-t)^{n+1-i} + \sum_{i=0}^{n} b_i \binom{n+1}{i+1} \frac{i+i}{n+1} t^{i+1}(1-t)^{n-i} \\ &= \sum_{i=0}^{n} b_i \frac{n+1-i}{n+1} B_i^{n+1}(t) + \sum_{i=0}^{n} b_i \frac{i+i}{n+1} B_{i+1}^{n+1}(t) \\ &= \sum_{i=0}^{n} b_i \frac{n+1-i}{n+1} B_i^{n+1}(t) + \sum_{i=1}^{n+1} b_{i-1} \frac{i}{n+1} B_i^{n+1}(t). \end{aligned}$$

Por lo tanto, añadiendo un término nulo a cada uno de los sumatorios de la expresión anterior, tenemos:

$$P(t) = \sum_{i=0}^{n+1} b_i \frac{n+1-i}{n+1} B_i^{n+1}(t) + \sum_{i=0}^{n+1} b_{i-1} \frac{i}{n+1} B_i^{n+1}(t).$$

Si ahora combinamos ambas sumas y comparamos coeficientes, damos con el resultado buscado

$$b_i^{(1)} = \frac{i}{n+1} b_{i-1} + \left(1 - \frac{i}{n+1}\right) b_i, \quad \text{para} \quad i = 0, \ldots, n+1. \tag{3.1}$$

Notemos que los nuevos vértices $b_i^{(1)}$ han sido obtenidos del polígono dato mediante interpolación lineal; $\lambda b_{i-1} + (1-\lambda) b_i$, con $\lambda = \frac{i}{n+1}$.

Las aplicaciones de la elevación del grado son numerosas en el diseño de superficies asistido por ordenador: hay ciertos algoritmos que producen superficies a partir de curvas dadas, siendo necesario que todas ellas sean del mismo grado. Una solución para este problema sería elevar el grado de todas las curvas hasta el de aquella de mayor grado, haciendo uso de (3.1).

Como se puede ver en [7], esto hace que el nuevo polígono de control (llamémosle $\mathfrak{EP}$) sea el más próximo a $P(t)$ y esté contenido en la envoltura convexa del polígono original, $\mathfrak{P}$. El proceso de elevación del grado asigna un polígono $\mathfrak{EP}$ a un polígono original $P$. Si repetimos el proceso, obtenemos una secuencia de polígonos $\mathfrak{P}, \mathfrak{EP}, \mathfrak{E}^2\mathfrak{P}$, etc. Tras $r$ elevaciones, el polígono tiene como vértices los puntos $b_0^{(r)}, \ldots, b_{n+r}^{(r)}$, y cada $b_i^{(r)}$ viene dado explícitamente por

$$b_i^{(r)} = \sum_{j=0}^{n} b_j \binom{n}{j} \frac{\binom{r}{i-j}}{\binom{n+r}{i}}.$$

Si continuamos con dicho proceso de forma reiterada, los polígonos $\mathfrak{E}^r\mathfrak{P}$ convergen a la curva que todos ellos definen. La prueba es bastante extensa y se puede consultar en distintas fuentes, entre las que destacamos [7] y [18].

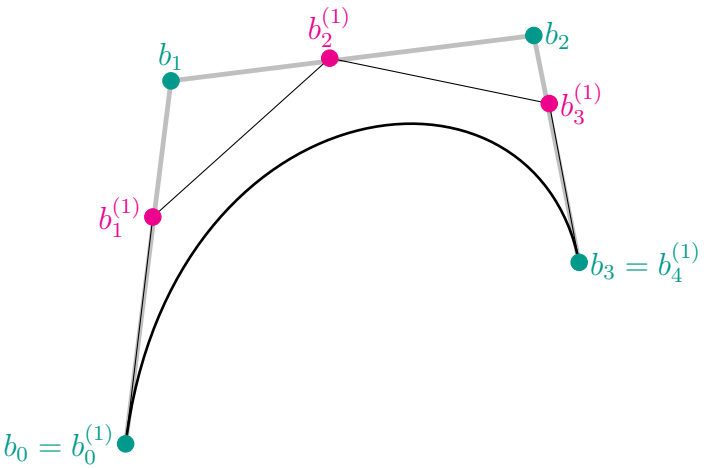



Figura 3.1: Elevación del grado de un segmento cúbico de Bézier a uno de grado cuatro. Ambos polígonos definen la misma curva (de grado tres).

Concluimos esta sección con la Figura 3.1, en la que hemos elevado el grado de una curva de Bézier de grado tres a grado cuatro. De acuerdo con la ecuación (3.1), se establecen nuevos puntos de control (en rosa) a partir de los originales (en azul), de forma que el trazado de la curva de definen ambos polígonos es invariante, siendo el número de nuevos vértices mayor en una unidad, en cada paso.

## 3.2. Subdivisión de una curva de Bézier

Una curva de Bézier normalmente se define sobre el intervalo $[0, 1]$, su dominio de definición. Sin embargo, también puede hacerse sobre cualquier intervalo $[a, b]$ con $a < b$. En particular, pensemos en $[0, \lambda]$, con $0 \leq \lambda \leq 1$. La parte de la curva original que corresponde a $[0, \lambda]$ puede determinarse también a partir de un polígono de Bézier. El proceso de encontrar dicho polígono se conoce como la subdivisión de una curva de Bézier.

Queremos obtener los puntos de Bézier $c_i$, del intervalo $[0, \lambda]$. Esto se puede efectuar mediante $i$ pasos en el esquema de De Casteljau con respecto a $\lambda$, de la siguiente forma:

$$c_i = b_0^i(\lambda), \tag{3.2}$$

también llamada fórmula de subdivisión para curvas de Bézier.

Por lo tanto, resulta que el algoritmo de De Casteljau no solo computa el punto $b_0^n$, evaluado en $\lambda$, sino que también nos proporciona los vértices de la curva correspondiente al intervalo $[0, \lambda]$. Además, gracias a la propiedad de simetría, sabemos que los puntos de control de la parte correspondiente a $[\lambda, 1]$ vienen dados por $b_i^{n-i}(\lambda)$, como puede verse en la Figura 3.2: partiendo de los nodos originales del intervalo $[0, 1]$ (dibujados en azul), obtenemos los nuevos nodos correspondientes a la parte del intervalo $[0, \lambda]$ (dibujados en rosa). Notemos que la curva que generan los nuevos nodos es una parte del trazado de la curva original.

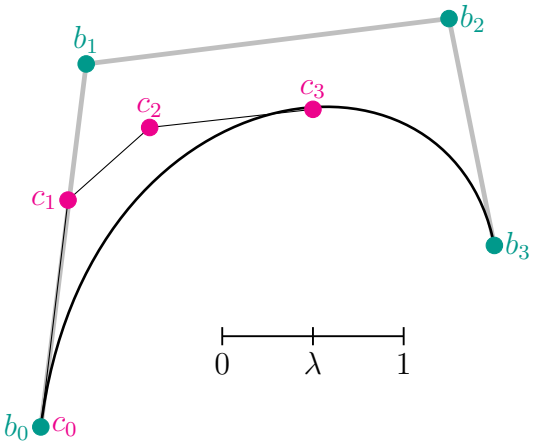



Figura 3.2: Subdivisión: dos polígonos de Bézier que describen la misma curva. Uno de ellos $(b_i)$ está asociado con el intervalo $[0, 1]$, el otro $(c_i)$, con $[0, \lambda]$.

En otras palabras, usando el algoritmo de De Casteljau, podemos encontrar los puntos en los cuales dividir una curva de Bézier en otras curvas más pequeñas, de forma que juntas conformen la original.

## 3.3. Conexión de curvas de Bézier y relación con los Splines

Los segmentos de curvas de Bézier tienen múltiples aplicaciones en softwares CAD, como ya hemos explicado en la introducción. Sin embargo, cuando queremos representar una curva de trazo complejo por segmentos de Bézier, necesariamente tendríamos que aproximarla elevando su grado de forma considerable. Esto dificulta la forma de su polígono en términos de control local, debido a la naturaleza global de la que está dotado. Por lo tanto, si realizamos esta conexión con las restricciones adecuadas, conseguiremos limitar la influencia de un segmento sobre los otros. Si además encontramos una relación entre dichas condiciones, ejerceremos un control local del trazado sin olvidarnos las condiciones de continuidad, y así crear el diseño deseado.

Nos encontramos ahora ante el problema de conectar dos curvas genéricas, no necesariamente de Bézier, $P_1(t)$ y $P_2(t)$, con $t$ variando entre 0 y 1. Abordaremos esta sección desde la teoría de curvas y superficies de Frenet, que damos por sabida. Para una lectura más completa, se pueden consultar [2], [11] y [18].

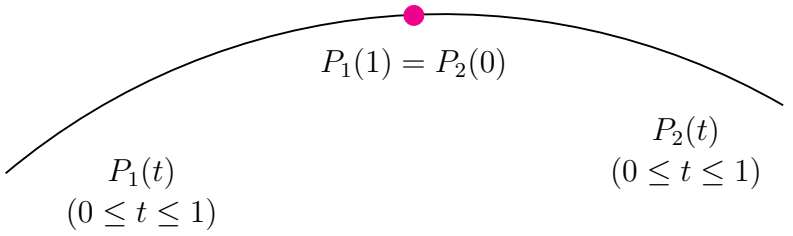



Figura 3.3: Condiciones de continuidad para curvas

En primer lugar, debemos estudiar las condiciones que garanticen la continuidad de sendas curvas. Empezaremos por la condición de continuidad de la posición

$$P_2(0) = P_1(1). \tag{3.3}$$

Si las magnitudes de los vectores tangentes de $P_1(t)$ y $P_2(t)$ en el punto de conexión son $\alpha_1$ y $\alpha_2$, respectivamente, entonces la condición de continuidad de la pendiente es:

$$\frac{1}{\alpha_2} P_2'(0) = \frac{1}{\alpha_1} P_1'(1) = t,$$

que podemos expresar como:

$$P_2'(0) = \frac{\alpha_2}{\alpha_1} P_1'(1) = \mu P_1'(1), \quad \mu = \frac{\alpha_2}{\alpha_1}. \tag{3.4}$$

Ahora, estudiemos la condición para que los centros de curvatura de ambos segmentos varíen de forma continua en el punto de conexión. La ecuación de la curvatura viene dada por

$$P'' = \kappa n,$$

donde $\kappa$ es la magnitud de la curvatura y $n$ es el vector normal unitario, que apunta al centro de curvatura. De la ecuación anterior, se asume que es necesario que $\kappa$ y $n$ sean continuas.

Si el vector binormal $b = t \times n$ y $t$ son continuos, está claro que $n$ también lo es. En consecuencia, para encontrar la condición de $\kappa$ y $n$ continuas, necesitamos una condición para $\kappa$ y $b$ continuos.

Como

$$\kappa b = \frac{P' \times P''}{|P'|^3},$$

para que el centro de curvatura varíe de forma continua en un punto de conexión, debemos tener

$$\frac{P_2'(0) \times P_2''(0)}{|P_2'(0)|^3} = \frac{P_1'(1) \times P_1''(1)}{|P_1'(1)|^3}.$$

Sustituyendo la ecuación (3.4) en la expresión anterior y reagrupando, obtenemos

$$P_1'(1) \times \left[ P_2''(0) - \left(\frac{\alpha_2}{\alpha_1}\right)^2 P_1''(1) \right] = 0.$$

A partir de esto, concluimos que

$$\begin{aligned} P_2''(0) &= \left(\frac{\alpha_2}{\alpha_1}\right)^2 P_1''(1) + \beta P_1'(1) \\ &= \mu^2 P_1''(1) + \beta P_1'(1), \end{aligned} \tag{3.5}$$

con $\mu = \alpha_2/\alpha_1$ y $\beta$ un escalar arbitrario. Notemos que esta es la condición buscada entre los dos segmentos para poder ser conectados.

Supongamos ahora que queremos conectar al segmento de curva siguiente

$$P_1(t) = \sum_{i=0}^{m} \binom{m}{i} (1-t)^{m-i} t^i b_{1,i} \quad (0 \le t \le 1)$$

un segundo segmento, llamémosle

$$P_2(t) = \sum_{j=0}^{n} \binom{n}{j} (1-t)^{n-j} t^j b_{2,j} \quad (0 \le t \le 1).$$

con el fin de que la combinación resultante sea también una curva de Bézier, donde el vector curvatura se modifique de forma continua.

**Observación 3.1.** *Notemos que hemos modificado la notación de los puntos de control ligeramente, incluyendo un primer subíndice que indica si un vértice pertenece a la primera o a la segunda curva, para no dar lugar a confusión.*

Para empezar, de (3.3) sabemos que la condición de continuidad el en punto de conexión es:

$$P_1(1) = P_2(0) \Rightarrow b_{2,0} = b_{1,m}. \tag{3.6}$$

Usando el apartado (ii) del Corolario 2.1 en la condición de continuidad de la pendiente dada por (3.4):

$$\frac{n}{\alpha_2}(b_{2,1} - b_{2,0}) = \frac{m}{\alpha_1}(b_{1,m} - b_{1,m-1}) = t, \tag{3.7}$$

donde $\alpha_1$ y $\alpha_2$ son las magnitudes de los vectores tangentes $P_1'(1)$ y $P_2'(0)$, respectivamente. La ecuación (3.7) requiere que los tres puntos siguientes; $b_{1,m-1}$, $b_{1,m} = b_{2,0}$ y $b_{2,1}$ sean colineales.

De la condición de continuidad de los vectores curvatura en el punto de unión (3.5) y de la condición (iii) del Corolario 2.1, tenemos que la segunda derivada en el punto buscado cumple:

$$P_1''(1) = m(m-1)(b_{1,m} - 2b_{1,m-1} + b_{1,m-2})$$
$$P_2''(0) = n(n-1)(b_{2,2} - 2b_{2,1} + b_{2,0}).$$

Sustituyendo lo anterior en (3.4),

$$n(n-1)(b_{2,2} - 2b_{2,1} + b_{2,0}) = m(m-1)\left(\frac{\alpha_2}{\alpha_1}\right)^2 (b_{1,m} - 2b_{1,m-1} + b_{1,m-2}) + m\beta(b_{1,m} - b_{1,m-1}).$$

Ahora, en virtud de (3.6) y (3.7), eliminamos $b_{2,0}$ y $b_{2,1}$ de forma que

$$b_{2,2} = \frac{m(m-1)}{n(n-1)}\Big(\frac{\alpha_2}{\alpha_1}\Big)^2 b_{1,m-2}$$

$$-\left[\frac{2m(m-1)}{n(n-1)}\Big(\frac{\alpha_2}{\alpha_1}\Big)^2 + \frac{2m}{n}\frac{\alpha_2}{\alpha_1} + \frac{m}{n(n-1)}\beta\right] b_{1,m-1}$$

$$-\left[\frac{m(m-1)}{n(n-1)}\Big(\frac{\alpha_2}{\alpha_1}\Big)^2 + \frac{2m}{n}\frac{\alpha_2}{\alpha_1} + 1 + \frac{m}{n(n-1)}\beta\right] b_{1,m} \tag{3.8}$$

determina el valor de $b_{2,2}$ que hará continuos los vectores curvatura. Podemos reagrupar la ecuación de la siguiente manera:

$$b_{2,2} - b_{1,m} = \frac{m(m-1)}{n(n-1)}\Big(\frac{\alpha_2}{\alpha_1}\Big)^2 (b_{1,m-2} - b_{1,m-1})$$

$$+\left[\frac{m(m-1)}{n(n-1)}\Big(\frac{\alpha_2}{\alpha_1}\Big)^2 + \frac{2m}{n}\frac{\alpha_2}{\alpha_1} + \frac{m}{n(n-1)}\beta\right](b_{1,m} - b_{1,m-1}), \tag{3.9}$$

lo cual nos dice que $b_{2,2}$ se encuentra en el mismo plano que $b_{1,m-2}$, $b_{1,m-1}$, $b_{1,m} = b_{2,0}$ y $b_{2,1}$.

De esta manera, hemos encontrado una relación geométrica entre las restricciones impuestas, garantizando un control local del trazado, para así poder modelar el diseño deseado.

**Observación 3.2.** *La representación vectorial de una curva mediante la base de splines, que normalmente se conoce como B-spline, tiene las características que acabamos de describir. En bibliografía, se suele introducir la representación mediante splines en primer lugar. Sin embargo, se necesita mayor bagaje matemático y el cálculo numérico de las curvas y superficies de splines es mucho más complejo a la hora de evaluar trazos o formas distintos, en*

*comparación con las curvas y superficies de Bézier. Esto se debe a que los splines se definen de forma recursiva, mientras que la definición de las curvas de Bézier es analítica, simple y posee una interpretación gráfica clara.*

*Además, como una base de splines consiste en trozos de curvas polinomiales, cada segmento puede expresarse como una curva de Bézier definida por un polígono de control. Es por eso por lo que, en este trabajo, hemos tomado un enfoque diferente para sintetizar una curva o superficie, que verdaderamente están compuestas por curvas o, como estudiaremos a continuación, mallas de Bézier.*

# Capítulo 4

# Superficies de Bézier

En este capítulo, veremos que muchas de las herramientas y algoritmos que hemos desarrollado para las curvas van a seguir siendo útiles para las superficies. Su diseño no es mucho más complejo que el diseño de curvas que hemos estudiado hasta el momento.

Hoy en día tenemos muchas variantes para construir superficies. La más elemental es mediante el producto cartesiano o tensorial de curvas de Bézier o de NURBS *(non uniform rational B-splines)*. Para definir una superficie de esta manera, hemos de permitir que los vértices del polígono de control se muevan a lo largo de curvas de Bézier parametrizadas. Es decir, debemos asumirla como una función de dos parámetros, que en este caso vamos a llamar $(u, v)$. Será, además, el enfoque que seguiremos en la introducción a las superficies de Bézier, en las que nos centraremos principalmente, debido a su sencillez en comparación con las superficies racionales o polinómicas a trozos. En los apartados que se desarrollan a continuación, hemos seguido la referencia de [7], un texto que trata sobre los métodos más usados en el diseño de curvas y superficies y se basa en el movimiento de la teoría a la práctica.

## 4.1. Definición

De acuerdo con lo comentado, nos interesa que los vértices evolucionen a lo largo de curvas fáciles de modelar, en este caso, curvas de Bézier de grado $n$ de polígonos $\{b_{i,0}, \ldots, b_{i,n}\}$ respectivamente.

Así, dado un conjunto finito de puntos de $\mathbb{R}^3$ que llamaremos red o malla de control $\{b_{i,j}\}_{(i,j)=0}^{(m,n)}$, definimos la superficie de Bézier $S : [0,1] \times [0,1] \to \mathbb{R}^3$ como:

$$S(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{i,j} B_i^m(u) B_j^n(v), \quad (u,v) \in [0,1] \times [0,1].$$

En esta representación, contamos con una malla rectangular de $(n+1)(m+1)$ puntos en el espacio. Es conocida como producto tensorial, ya que la base de funciones es el producto de las bases de polinomios en $u$ y $v$.

También puede ser parametrizada en forma matricial como:

$$P(u,v) = \begin{bmatrix} B_0^m(u) & B_1^m(u) & \cdots & B_m^m(u) \end{bmatrix} \begin{bmatrix} b_{0,0} & b_{0,1} & \cdots & b_{0,n} \\ b_{1,0} & \cdots & \cdots & b_{1,n} \\ \vdots & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \vdots \\ b_{m,0} & b_{m,1} & \cdots & b_{m,n} \end{bmatrix} \begin{bmatrix} B_0^n(v) \\ B_1^n(v) \\ \vdots \\ \vdots \\ B_n^n(v) \end{bmatrix}.$$

Las $(n+1)(m+1)$ funciones empleadas son el producto de dos bases que ya conocemos: la base de polinomios de grado $m$ en la variable $u$ y la de grado $n$ en la variable $v$, es decir, $\{B_0^m(u)B_0^n(v), \ldots, B_m^m(u)B_n^n(v)\}$. Como el nuevo espacio de polinomios es un producto de los espacios anteriores,

$$\mathbb{R}_{m,n}[u,v] = \mathbb{R}_m[u]\mathbb{R}_n[v],$$

característica de las denominadas superficies producto tensorial bigrado, en este caso $(m,n)$. En la Figura 4.1, se muestra un ejemplo de superficie bigrado para $m = 4$ y $n = 3$.

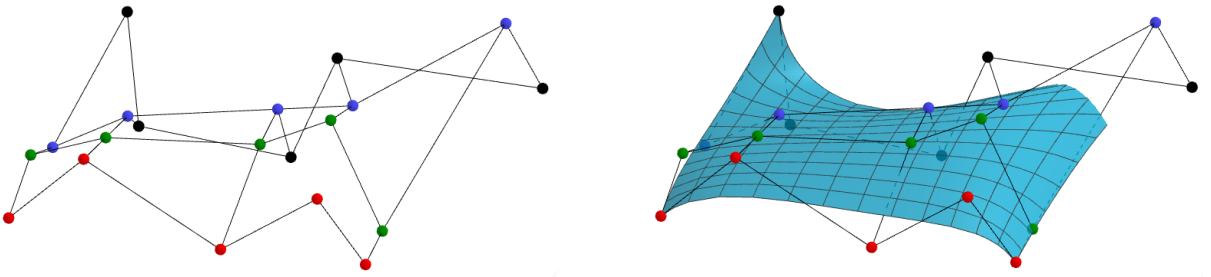


Figura 4.1: Malla de control y superficie de Bézier bigrado $(4,3)$, generadas mediante la calculadora gráfica GeoGebra.

De esta manera, si fijamos el parámetro $u$, obtenemos

$$S(u_0,v) = \sum_{j=0}^{n} \left( \sum_{i=0}^{m} b_{i,j} B_i^m(u_0) \right) B_j^n(v) \quad v \in [0,1],$$

que son curvas de Bézier de grado $n$ y polígono de vértices $b_j = (\sum_{i=0}^{m} b_{i,j} B_i^m(u_0))$. Las curvas resultantes de fijar $v$;

$$S(u,v_0) = \sum_{i=0}^{m} \left( \sum_{j=0}^{n} b_{i,j} B_j^n(v_0) \right) B_i^m(u) \quad u \in [0,1],$$

son curvas de Bézier de grado $m$ y polígono $b_i = \left( \sum_{j=0}^{n} b_{i,j} B_j^n(v_0) \right)$.

Notemos que la base de funciones es una partición de la unidad, pues

$$\sum_{i=0}^{m} \sum_{j=0}^{n} B_i^m(u) B_j^n(v) = 1,$$

lo cual permite que las superficies hereden las buenas propiedades de las curvas.

## 4.2. Propiedades

Las siguientes propiedades de las funciones $B_i^m(u)$, $B_j^n(v)$ y la superficie $S(u,v)$ se deducen de forma clara de los polinomios de Bernstein y las curvas de Bézier.

1. *No negatividad.* Como las superficies de Bézier son una generalización de las curvas polinómicas ya estudiadas, esta propiedad se mantiene:

$$B_i^m(u)B_j^n(v) \geq 0, \forall i,j,u,v.$$

2. *Partición de la unidad.* Al ser nuestra parametrización una combinación convexa de los vértices de la malla de control y, dado que los polinomios de Bernstein forman una partición de la unidad,

$$\sum_{i=0}^{m}\sum_{j=0}^{n} B_i^m(u)B_j^n(v) = 1.$$

   Esto implicará que se cumplan otras buenas propiedades de las curvas, como puede ser la invarianza ante transformaciones afines.

3. *Invarianza afín.* La imagen de una superficie de malla $\{b_{i,j}\}_{(i,j)=0}^{(m,n)}$ a través de una aplicación afín $f$ es una superficie de malla $\{f(b_{i,j})\}_{(i,j)=0}^{(m,n)}$, es decir,

$$f\left(S(u,v)\right) = \sum_{i=0}^{m}\sum_{j=0}^{n} f(b_{i,j})B_i^m(u)B_j^n(v) \quad (u,v) \in [0,1]\times[0,1].$$

4. *Envolvente convexa.* La superficie de Bézier sigue estando comprendida en la envolvente convexa de la malla de control, el menor poliedro convexo que contiene a todos los vértices de la malla. Esto proporciona una estimación inicial sobre la posición de dicha superficie.

5. *Extremos y bordes.* De igual manera que las curvas de Bézier pasan por los vértices extremos, las superficies solo pasan por las esquinas de la malla de control:

$$b_{0,0} = S(0,0), \quad b_{m,0} = S(1,0), \quad b_{0,n} = S(0,1), \quad b_{m,n} = S(1,1).$$

   En el caso de las curvas, contábamos con dos puntos extremos, mientras que ahora tenemos cuatro puntos y cuatro curvas que describen el borde de la superficie. El polígono límite está formado por estas cuatro curvas y sus respectivos cortes, llamados vértices extremos.
   Es decir, las filas y columnas del borde de la malla describen el borde de la superficie:

   - La curva $u = 0$ tiene por polígono $\{b_{0,0},\ldots,b_{0,n}\}$, que es la primera fila de la malla de control de la superficie:

$$S(0,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} b_{i,j}B_i^m(0)B_j^n(v) = \sum_{j=0}^{n} b_{0,j}B_j^n(v), \quad v \in [0,1].$$

   - La curva $v = 0$ tiene por polígono $\{b_{0,0},\ldots,b_{m,0}\}$, la primera columna de la malla:

$$S(u,0) = \sum_{i=0}^{m}\sum_{j=0}^{n} b_{i,j}B_i^m(u)B_j^n(0) = \sum_{i=0}^{m} b_{i,0}B_i^m(u), \quad u \in [0,1].$$

- De manera análoga, la curva $u = 1$ tiene por polígono $\{b_{m,0}, \ldots, b_{m,n}\}$, que es la última fila de la malla de control.
- La curva $v = 1$ tiene por polígono $\{b_{0,n}, \ldots, b_{m,n}\}$, última columna de la malla.

Por lo tanto, el borde de la superficie se corresponde con el borde de la malla.

6. *Control local.* Un vértice de la malla de control afecta a lo sumo a $(m+1)(n+1)$ tramos de la superficie. En la Figura 4.2 mostramos un ejemplo gráfico de esta propiedad, al desplazar un vértice (de color verde) de forma exagerada. Vemos que la superficie queda ligeramente deformada en el entorno del punto afectado.

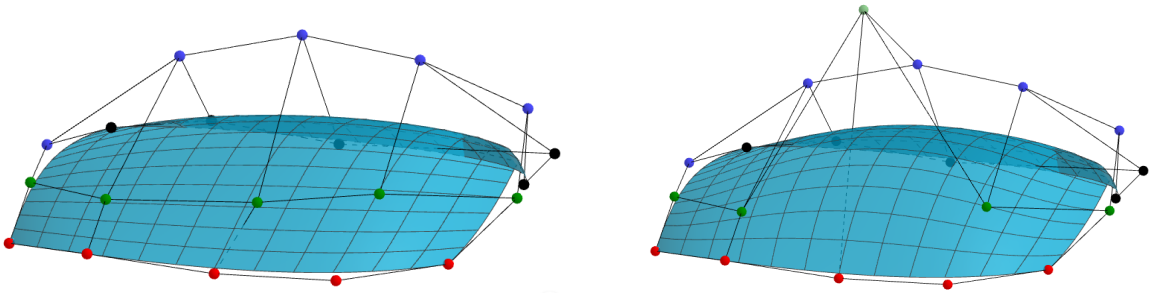


Figura 4.2: Control local de superficies de Bézier. Se observa el efecto producido al desplazar un vértice de una superficie bigrado $(4, 3)$.

Sin embargo, las superficies no ofrecen una generalización sencilla para la propiedad de disminución de la variación. Es fácil probar que una recta puede cortar al poliedro descrito por la malla de control en menos puntos de los que corta a la superficie.

Como parece natural, si queremos definir la parametrización en un recinto rectangular $[a, b]\times[c, d]$ en vez del usado hasta ahora, $[0, 1]\times[0, 1]$, podemos realizar una reparametrización afín de los parámetros $u$ y $v$ sin que la malla de control o el trazo de la superficie se vean afectados:

$$u(\tilde{u}) = \frac{\tilde{u} - a}{b - a} \quad \tilde{u} \in [a, b], \qquad v(\tilde{v}) = \frac{\tilde{v} - c}{d - c} \quad \tilde{v} \in [c, d].$$

Finalmente, es importante destacar que la representación en forma de producto tensorial funciona bien para superficies abiertas. En cambio, presenta ciertos inconvenientes en algunas topologías cerradas, porque suelen requerir mallas degeneradas, en las que los vértices se solapan. Otras representaciones como los triángulos de Bézier son más versátiles y evitan los problemas mencionados. Para un mayor estudio del papel de la topología de superficies en este tema, se pueden consultar los apuntes de sendos cursos, además de [2].

## 4.3. Algoritmo de De Casteljau

Dada la simetría que ofrece la parametrización de las superficies de Bézier, podemos tratar de forma prácticamente independiente a las variables $u$ y $v$. En el momento que fijamos una de ellas, somos capaces de recuperar el formalismo de las curvas polinómicas. Por esta razón, extender el algoritmo de De Casteljau a superficies parece intuitivo.

No solamente lo es, sino que la clave está en aplicarlo a cada una de las columnas de la malla de control por separado, como si fuesen polígonos de curvas de grado $m$. De acuerdo

con el Lema 2.1, aplicando

$$b_{i,j}^{k,0}(u) = (1-u)b_{i,j}^{k-1,0}(u) + ub_{i+1,j}^{k-1,0}(u)$$

para $i = 0, \ldots, m-k, \quad k = 1, \ldots, m \quad$ y $\quad j = 0, \ldots, n$, daríamos con un único punto en la iteración $m$-ésima de cada columna; $\{b_{0,0}^{m,0}(u), \ldots, b_{0,n}^{m,0}(u)\}$, que podríamos tratar como un polígono de control de una curva de grado $n$, al cual aplicaríamos de nuevo el algoritmo,

$$b_{0,j}^{m,r}(u,v) = (1-v)b_{0,j}^{m,r-1}(u,v) + vb_{0,j+1}^{m,r-1}(u,v)$$

para $j = 0, \ldots, n-r \quad$ y $\quad r = 1, \ldots, n$, hasta llegar al último paso del esquema, que nos proporciona el punto $S(u,v)$ de la superficie:

$$S(u,v) = b_{0,0}^{m,n}(u,v).$$

La Figura 4.3 muestra la aplicación del algoritmo de De Casteljau a una superficie, para los valores $u_0 = 0.5$ y $v_0 = 0.25$. Cada punto de la superficie se obtiene mediante interpolación bilineal repetida, empezando por la variable $u$. Fijado el valor de $u_0$ se obtiene la curva de color rojo, y fijado $v_0$ obtenemos la de color verde. Su intersección da lugar a un valor fijo de $u$ y de $v$. La animación se ha obtenido con MATHEMATICA y puede verse en `https://demonstrations.wolfram.com/DeCasteljauAlgorithmForATensorProductBezierSurface/`.

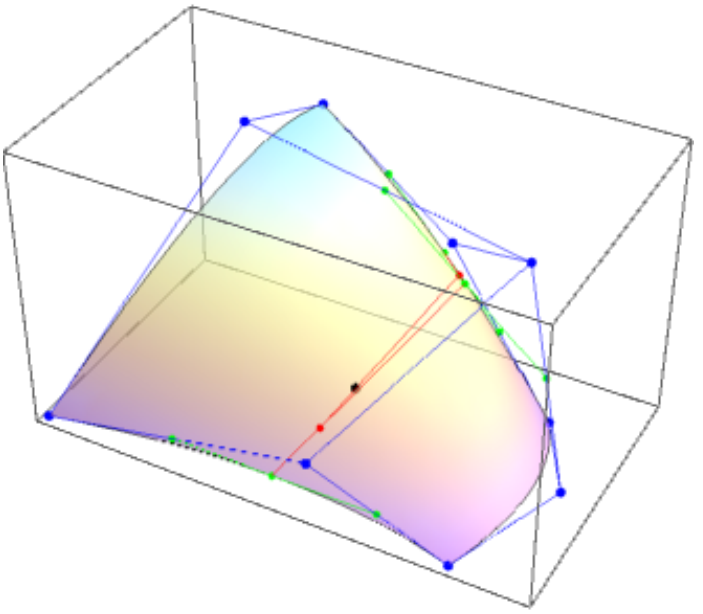


Figura 4.3: Algoritmo de Casteljau para $u_0 = 0.5, v_0 = 0.25$ aplicado en superficies de Bézier.

Observemos que este proceso es simétrico, podíamos haberlo aplicado en primer lugar a las filas, considerándolas polígonos de curvas de grado $n$ y repetirlo sobre el polígono obtenido, de grado $m$. La idea subyace en aplicar el algoritmo de De Casteljau de forma sucesiva a las filas y columnas de la malla de control.

## 4.4. Elevación del grado en una superficie de Bézier

El proceso de elevación del grado de una superficie de Bézier no se diferencia del estudiado para curvas polinómicas. Supongamos que, dada una malla de control bigrado $(m, n)$, $\{b_{0,0}, \ldots, b_{m,n}\}$, deseamos expresar la superficie en términos de una malla bigrado $(m+1, n)$. Bastaría con aplicar el algoritmo de elevación del grado, dado en (3.1), a las $n+1$ columnas de la malla de control de manera independiente:

$$S(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} b_{i,j}B_i^m(u)B_j^n(v) = \sum_{j=0}^{n}\left[\sum_{i=0}^{m+1} b_{i,j}^{(1,0)}B_i^{m+1}(u)\right]B_j^n(v), \quad \text{con}$$

$$b_{i,j}^{(1,0)} = \left(1 - \frac{i}{m+1}\right)b_{i,j} + \frac{i}{m+1}b_{i-1,j}; \quad i = 0, \ldots, m+1, \quad j = 0, \ldots, n.$$

De igual forma, si queremos elevar el bigrado a $(m, n+1)$, aplicamos la operación a las $m+1$ filas de la malla de control por separado:

$$S(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} b_{i,j}B_i^m(u)B_j^n(v) = \sum_{i=0}^{m}\left[\sum_{j=0}^{n+1} b_{i,j}^{(0,1)}B_j^{n+1}(v)\right]B_i^m(u), \quad \text{con}$$

$$b_{i,j}^{(0,1)} = \left(1 - \frac{j}{n+1}\right)b_{i,j} + \frac{j}{n+1}b_{i,j-1}.$$

## 4.5. Derivadas de una superficie de Bézier

Debido a que la base de funciones que empleamos, $B_i^m(u)B_j^n(v)$, cuenta con variables $u, v$ separadas, las derivadas parciales de la parametrización no presentan cambios significativos en comparación con los estudiados en la sección de curvas de Bézier.

Las derivadas de la superficie $S(u, v)$ con respecto a las variables $u$ y $v$ satisfacen:

$$\frac{\partial S(u,v)}{\partial u} = \sum_{j=0}^{n}\frac{\partial}{\partial u}\left(\sum_{i=0}^{m} b_{i,j}B_i^m(u)\right)B_j^n(v)$$

$$= m\sum_{j=0}^{n}\sum_{i=0}^{m-1}\Delta^{1,0}b_{i,j}B_i^{m-1}(u)B_j^n(v).$$

$$\frac{\partial S(u,v)}{\partial v} = n\sum_{i=0}^{m}\sum_{j=0}^{n-1}\Delta^{0,1}b_{i,j}B_i^m(u)B_j^{n-1}(v),$$

donde las diferencias vienen dadas en los subíndices:

$$\Delta^{i,j}b_{i,j} = \Delta^{i-1,j}b_{i+1,j} - \Delta^{i-1,j}b_{i,j} = \Delta^{i,j-1}b_{i,j+1} - \Delta^{i,j-1}b_{i,j},$$

y los valores iniciales se obtienen de la generalización de las diferencias vistas en la sección correspondiente a las curvas. El superíndice $(1, 0)$ indica que la diferencia se hace solo en el primer subíndice:

$$\Delta^{1,0}b_{i,j} = b_{i+1,j} - b_{i,j}, \qquad \Delta^{0,1}b_{i,j} = b_{i,j+1} - b_{i,j}.$$

La expresión de las derivadas de orden superior es inmediata, pues

$$\frac{\partial^{k+r} S(u,v)}{\partial u^k \partial v^r} = \frac{m!n!}{(m-k)!(n-r)!} \sum_{i=0}^{m-k} \sum_{j=0}^{n-r} \Delta^{k,r} b_{i,j} B_i^{m-k}(u) B_j^{n-r}(v).$$

Además, son de especial interés las expresiones de las derivadas en los bordes de la superficie, correspondientes a $u = 0$, $u = 1$, $v = 0$, $v = 1$, que juegan un papel fundamental en el proceso de unión de superficies de Bézier, dado que deben ser diferenciables. Llegados a este punto, no desarrollaremos dichas técnicas para las superficies, pero pueden ser consultadas en algunos textos como [7], [8] y [11].

# Capítulo 5

# Aplicaciones en animación gráfica

En este capítulo, vamos a centrarnos en las aplicaciones de las curvas de Bézier en la animación de objetos. También introduciremos los esquemas de subdivisión, en particular los relacionados con las superficies. En primer lugar, daremos a conocer el modelo de la *Tetera de Utah*, en el que se emplean curvas de Bézier de grado tres. También veremos las razones que llevaron a la compañía de animación *Pixar Animation Studios* a emplear los esquemas de subdivisión en sus últimos proyectos, con especial atención al esquema de Catmull-Clark, empleado en aplicaciones de la animación 3D como son el sombreado o la texturización de imágenes. Finalmente, presentamos una serie de ejemplos gráficos de subdivisión obtenidos tras implementar el algoritmo de Catmull-Clark en MATHEMATICA. Las fuentes consultadas para la elaboración de este capítulo final son extensas. Destacamos las publicaciones [12], [15], [19] en el caso de la tetera; y [6], [20], [22] para el esquema de Catmull-Clark y su relación con *Pixar*.

## 5.1. Tetera de Utah

La tetera de Utah, *Utah teapot o Newell teapot,* es una prueba de modelado $3D$ que se ha convertido en un objeto estándar de referencia dentro de la comunidad de computación gráfica. Es un modelo matemático de una tetera de apariencia sólida, cilíndrica, y parcialmente convexa. La tetera está considerada el equivalente de un programa *"hello, world"*, como manera de crear una sencilla figura 3D con un modelo sencillo. Tanto es así que su primer ejemplar se encuentra en el *Computer History Museum* en Mountain View, California, de-

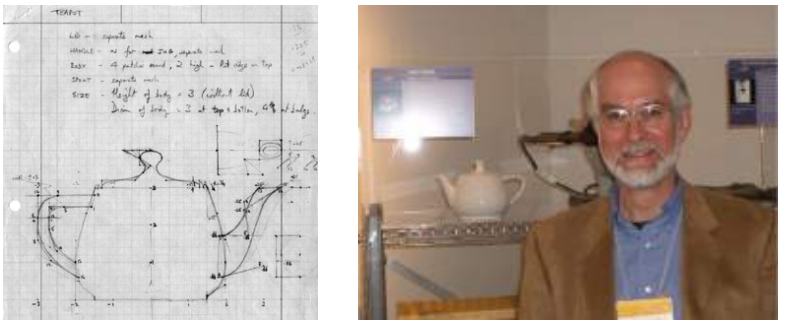
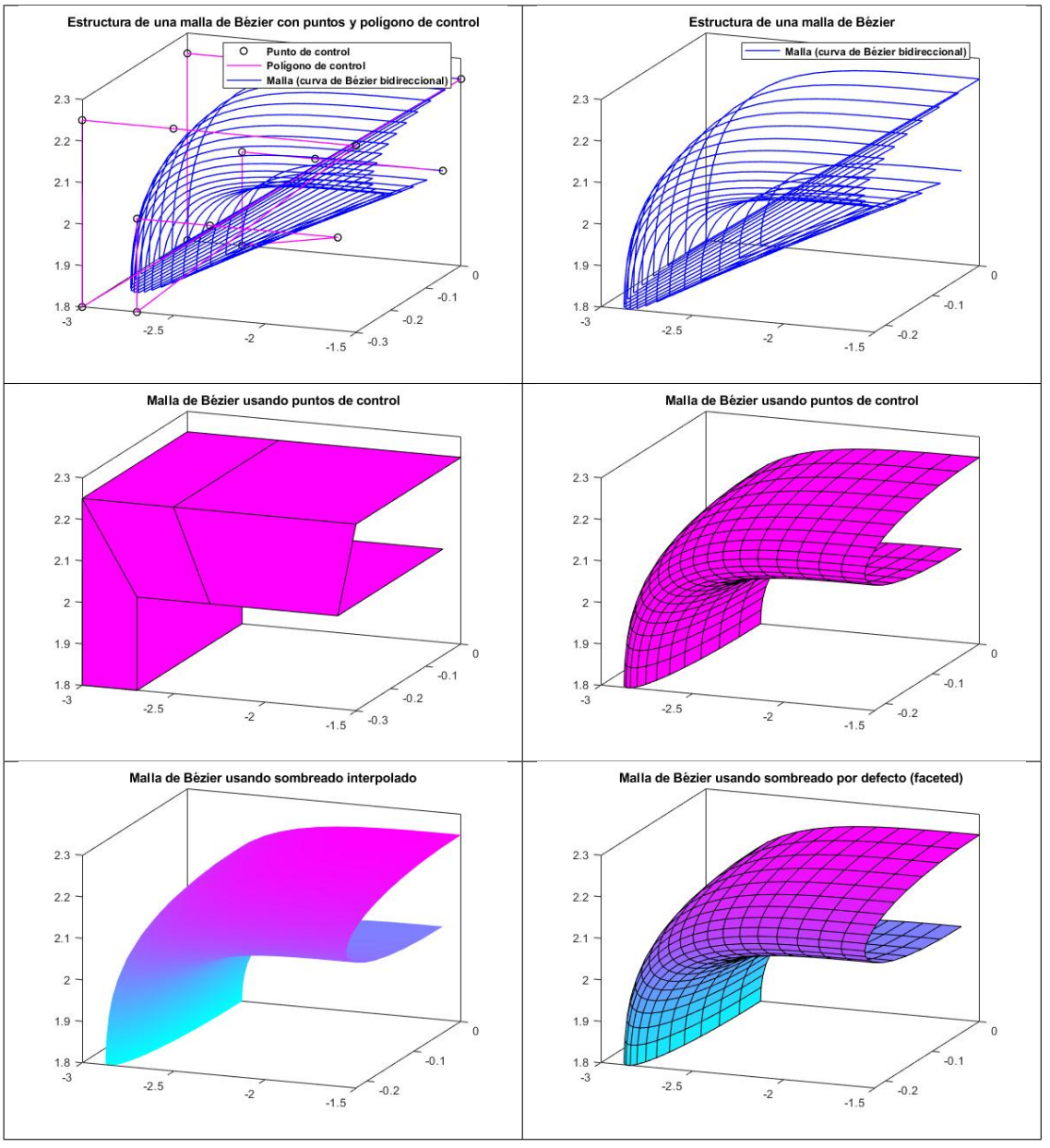


Figura 5.1: Martin Newell posando con su famosa tetera en el *Computer History Museum*, California. Newell desarrolló un modelo tridimensional tras una tarde de té con su mujer.

dicado a la preservación de herramientas de la era digital, y la exploración de la revolución computacional y su impacto en nuestras vidas.

El modelo de la tetera fue creado en 1975 por Martin Newell, investigador en computación gráfica y miembro de la Universidad de Utah. Fue pionero en su campo, pues, durante décadas, los programadores lo trataron como un aliciente a la hora de explorar nuevas técnicas de luz, sombreado y profundidad para dotar a los objetos de un mayor realismo.

Nuestro cometido será intentar generar dicha tetera mediante interpolación de curvas de Bézier cúbicas. Para ello, mostramos las imágenes del proceso de interpolar 16 puntos de control de una malla de Bézier para crear el dibujo tanto del asa como de la taza completa, en los que hemos utilizado dos tipos de sombreado. El software empleado ha sido MATLAB y su código puede consultarse en [12]. Sin embargo, algunas bibliotecas de programación, como OpenGL cuentan con funciones dedicadas exclusivamente al dibujo de teteras.



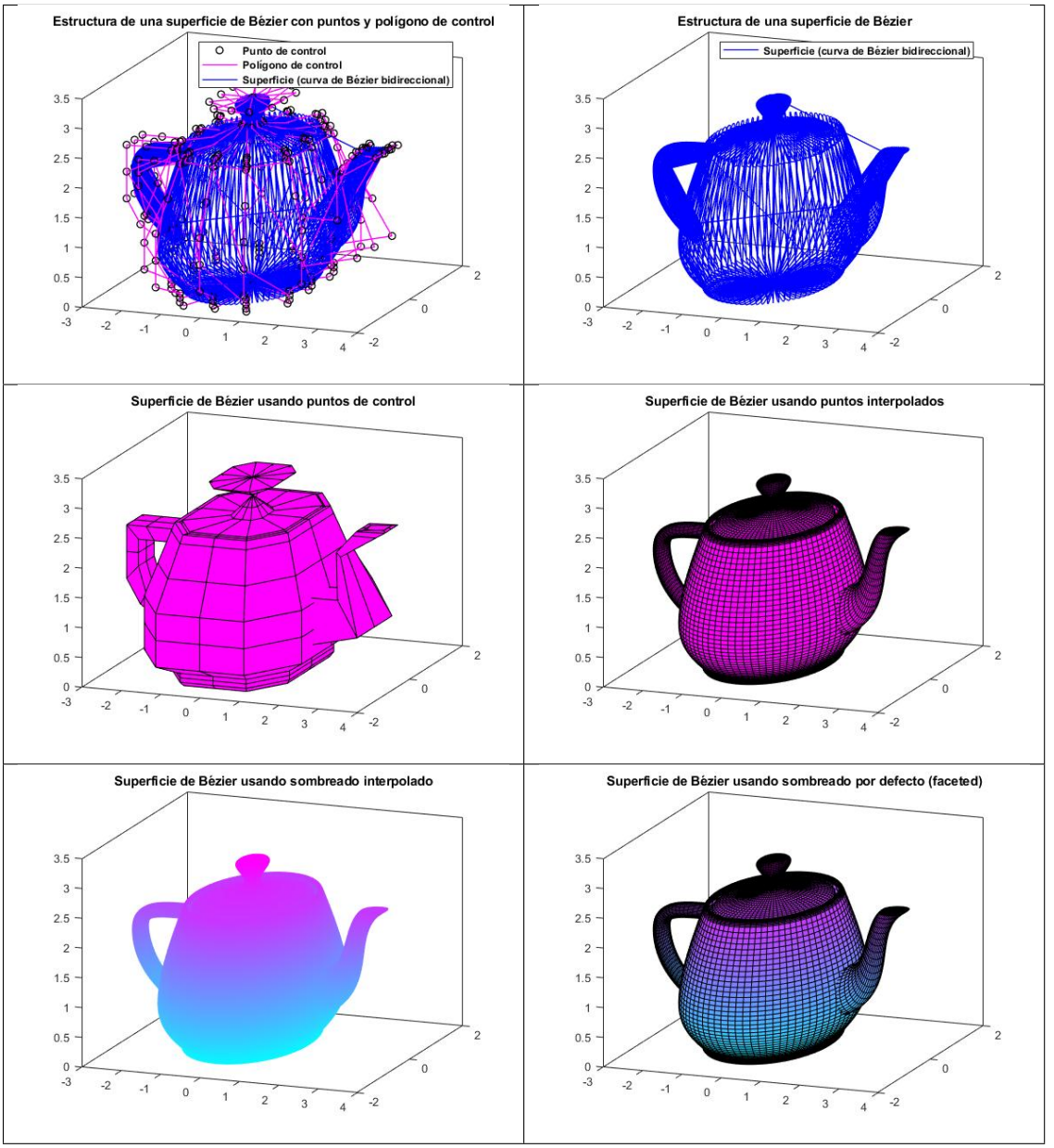



Figura 5.2: Generación de una superficie de Bézier cúbica a partir de curvas de Bézier cúbicas, conocida como la tetera de Utah.

Como curiosidad, añadiremos que la tetera de Utah contó con tanta popularidad que distintos estudios de animación la han incluido en sus proyectos para que los más avispados pudieran conocer su historia. Por ejemplo, apareció en un capítulo de *The Simpsons* y en una de las primeras y más populares películas de animación, *Toy Story*.

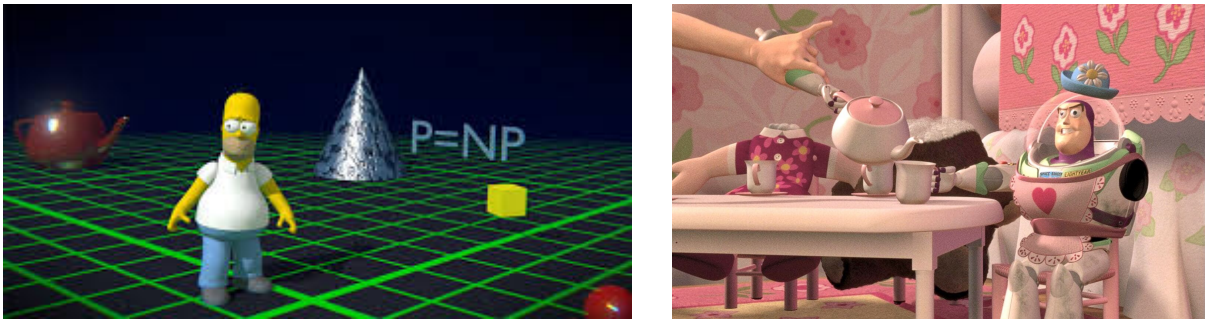



Figura 5.3: En distintos proyectos de animación podemos encontrar un guiño a la tetera de Utah.

## 5.2. Esquema de Catmull-Clark

Antes de detallar este algoritmo, es necesario que introduzcamos de forma breve el concepto de superficie de subdivisión. Una superficie de este tipo es un método de diseño gráfico que permite representar una superficie suave por medio de una malla polinomial. Su origen se remonta a 1978, con los trabajos de D. Doo y M. Sabin y E. Catmull y J. Clark, basados en las publicaciones de G. Chaikin. No gozaron de gran popularidad hasta el desarrollo de la computación gráfica de los años 90.

Una superficie de subdivisión se obtiene tras un proceso de refinamientos sucesivos de una superficie inicial, produciendo en cada paso más superficies con más caras. En el límite, lo que se obtiene es una superficie diferenciable. Entre las ventajas de este tipo de construcciones, destacamos las siguientes:

- Su eficiencia. Para pasar de un paso al siguiente, se descarta la información sobre un gran número de puntos, lo cual hace que sea de fácil implementación y bajo coste computacional.
- La posibilidad que ofrece de representar superficies de topologías arbitrarias.
- Su capacidad para producir una superficie límite suave y controlable, en particular en ciertos aspectos relativos a la forma, como bordes, pliegues o crestas.

### Pixar y los esquemas de Catmull-Clark

El algoritmo de Catmull-Clark es una técnica usada para crear superficies curvas utilizando el modelado de superficies de subdivisión. Fue ideado por Ed Catmull y Jim Clark como una generalización del algoritmo del *B-spline bicúbico*, frente a los dos problemas que presentaban las NURBS:

- El primero, que eran susceptibles a presentar errores numéricos.
- El segundo, que dificultaban que la superficie aproximada fuese suave.

Hasta entonces, el uso de la tecnología no era muy común en animación. Esta tendencia cambió a partir del año 1998, cuando la compañía decidió sacar a la luz los patrones de subdivisión empleados en sus últimos proyectos, con la esperanza de que una era marcada por la geometría se abriese paso en la industria de la animación.

En 2005, Catmull y Clark recibieron un Óscar, junto con Tony DeRose, por su invención y aplicación en la animación del personaje de la película *Geri's game*, de *Pixar*. A día de

hoy, la compañia cuenta con OpenSubdiv, un conjunto de librerías de código abierto para la evaluación del desempeño de superficies de subdivisión en arquitecturas de CPU y GPU.

Figura 5.4: A los lados, figuras de los personajes de Geri (*Geri's game*) y Woody (*Toy Story*) como superficies de Catmull-Clark. En el centro, la mano de Geri, en la que se han usado pliegues infinitamente refinados.

### Esquema de refinamiento de un poliedro

Las superficies de Catmull-Clark se definen de forma recursiva, usando un esquema de refinamiento, que detallamos a continuación para el caso de una pirámide cuadrangular. Para encontrar los fundamentos matemáticos más rigurosos se pueden consultar los trabajos originales de E. Catmull y J. Clark [4] y de J. Stam [17].

1. Se parte de los puntos originales, que son los vértices de la malla inicial. Aparecen en color negro y etiquetados con $A_i$, $i = 1, \ldots, 5$ en la parte izquierda de la Figura 5.5.
2. Se introducen los puntos de cara: $F_i$, $i = 1, \ldots, 5$. Aparecen en color rojo y etiquetados con $F_i$, $i = 1, \ldots, 5$ en la parte derecha de la Figura 5.5. Cada punto cara es el promedio de los vértices de la cara correspondiente. Por ejemplo, el punto $F_1$ correspondiente a la cara frontal, se calcula como

$$F_1 = \frac{1}{3}(A_1 + A_2 + A_5).$$

   El punto $F_5$ correspondiente a la cara del "suelo" de la figura, se calcula como

$$F_5 = \frac{1}{4}(A_1 + A_2 + A_3 + A_4).$$

3. Se introducen los puntos de arista: $E_i$, $i = 1, \ldots, 8$. Aparecen en color azul y etiquetados con $E_i$, $i = 1, \ldots, 8$ en la parte izquierda de la Figura 5.6. Cada punto de arista se calcula como el promedio de los dos vértices de la arista y los puntos de cara de las caras adyacentes. Por ejemplo, el punto $E_1$ correspondiente a la arista que une los vértices $A_1$ y $A_5$, se calcula como

$$E_1 = \frac{1}{4}(A_1 + A_5 + F_1 + F_4).$$

4. Se introducen los nuevos vértices: $V_i$, $i = 1, \ldots, 5$. Aparecen en gris y etiquetados con $V_i$, $i = 1, \ldots, 5$ en la parte derecha de la Figura 5.6. Por cada vértice original, $A_i$ se toma la media ($Q_i$) de los $n$ puntos de cara colindantes y creados recientemente.

Posteriormente, se calcula la media $R_i$ de los $n$ puntos medios de las aristas que tocan $A_i$. El nuevo vértice $V_i$ se calcula como la media ponderada

$$V_i = \frac{1}{n}(Q_i + 2F_i + (n-3)A_i).$$

**Observación 5.1.** *La elección de esta media ponderada está basada más bien en criterios estéticos, apoyados en la experiencia de los creadores del algoritmo, que en deducciones matemáticas. De hecho, tal y como se explica en [22] existen sucesivas modificaciones de este algoritmo en las que se consideran distintos pesos de los originales,*

$$\left\{\frac{1}{n}, \frac{2}{n}, \frac{n-3}{n}\right\}.$$

5. Se conecta cada punto cara con los puntos arista adyacentes.
6. Se conecta cada nuevo vértice con los puntos arista adyacentes.
7. Se obtiene, por subdivisión, un nuevo polígono con 16 caras, como se muestra en la Figura 5.7. Notemos que los nuevos vértices que se han hecho visibles aparecen coloreados en negro. El vértice $V_4$, que queda en la parte trasera de la figura, se mantiene coloreado en gris.

Notemos que con este algoritmo, pasamos de un poliedro con $v_0 = 5$ vértices, $c_0 = 5$ caras y $e_0 = 8$ aristas a un poliedro con $v_1 = v_0 + c_0 + e_0 = 18$ vértices, $c_1 = 3 \times 4 + 1 \times 4 = 16$ caras y $e_1 = 2 \times 8 + 4 \times 3 + 1 \times 4 = 32$ caras. Además, en ambos casos se cumple la fórmula de Euler

$$c_i + v_i = e_i + 2, \quad i = 0, 1.$$

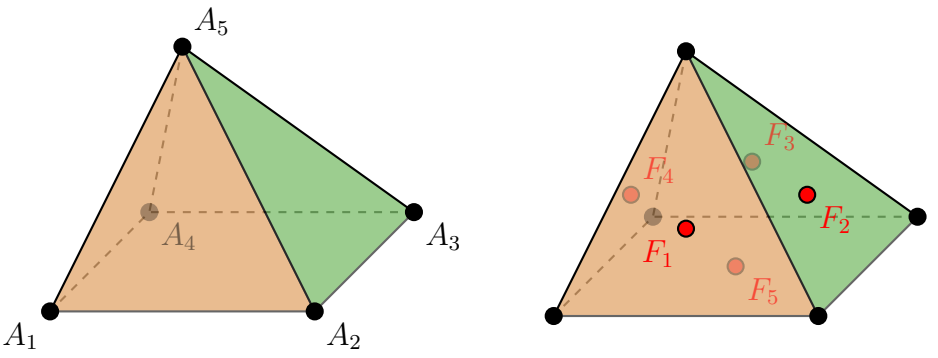



Figura 5.5: Algoritmo de Catmull-Clark: puntos originales (en negro) y puntos cara (en rojo).

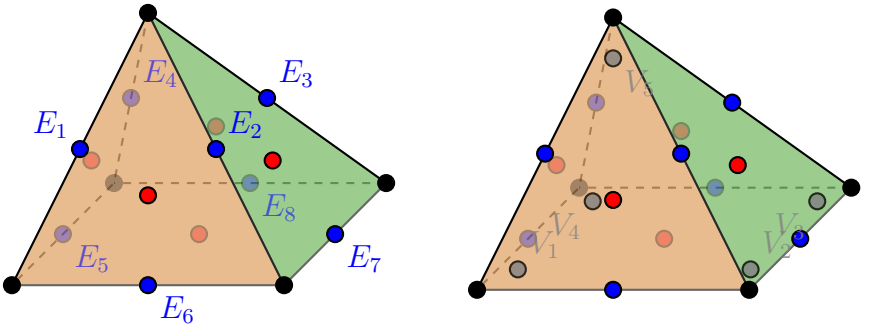



Figura 5.6: Algoritmo de Catmull-Clark: puntos de arista (en azul) y nuevos vértices (en gris).

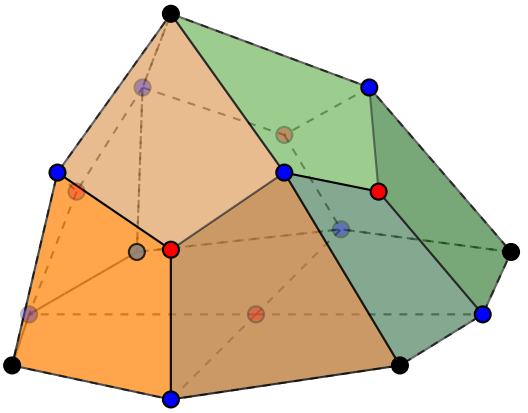


Figura 5.7: Algoritmo de Catmull-Clark: nuevo polígono con 16 caras.

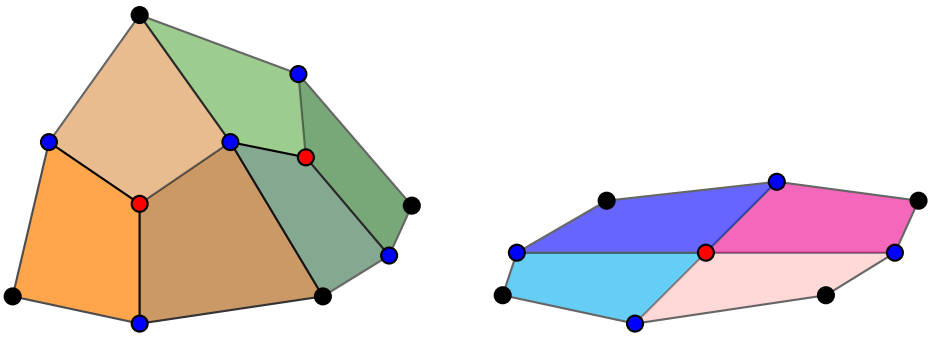


Figura 5.8: Algoritmo de Catmull-Clark: a la izquierda, vista frontal del nuevo polígono. A la derecha, el suelo del nuevo polígono.

### Ejemplos de superficies de Catmull-Clark generadas mediante MATHEMATICA

Para finalizar, mostramos las figuras obtenidas tras la implementación del algoritmo en MATHEMATICA, proporcionada por [21]. Para ello, ha sido necesario contar con una versión reciente del programa para generar una corriente de trabajo con C++, ejecutando los compiladores externos del ordenador mediante MICROSOFT VISUAL STUDIO.

A la hora de la evaluación, los vértices de la malla inicial se pueden obtener principalmente de dos formas: o bien definiéndolos manualmente mediante una lista de puntos, o bien cargando paquetes de puntos con los que cuenta el software.

En la primera línea de la Figura 5.9 mostramos tres iteraciones del algoritmo de Catmull-Clark aplicadas a una pirámide cuadrangular, obtenidas con MATHEMATICA. Vemos cómo en el primer paso obtenemos el poliedro que aparece en la Figura 5.7, de apariencia poco refinada todavía. Con un par de iteraciones, se obtiene una superficie mucho más suave. En la segunda línea mostramos el mismo proceso partiendo de una pirámide cuadrangular a la que se le ha eliminado la base.

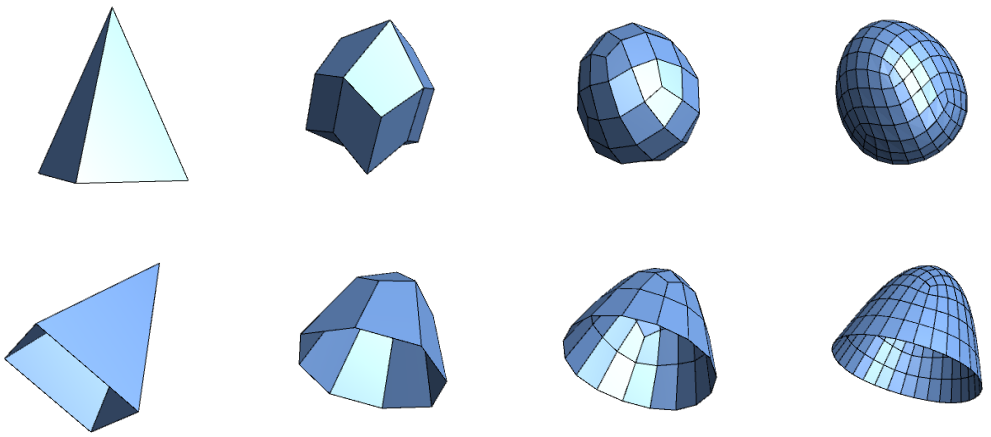


Figura 5.9: Aplicaciones del algoritmo de Catmull-Clark a una pirámide cerrada (primera línea) y a una pirámide abierta por la base (segunda línea).

En la Figura 5.10 mostramos tres ejemplos de aplicación del algoritmo de Catmull-Clark. En el primero de ellos, partimos de un cubo y, tras cuatro iteraciones, llegamos a una superficie casi esférica. Es evidente que en el límite llegaríamos a superficie totalmente esférica. Aún así, con tan solo cuatro iteraciones obtenemos un resultado satisfactorio gráficamente. En los otros dos ejemplos, se parte de un cubo abierto por una de sus caras. Se han aplicado dos algoritmos con pesos diferentes, notándose la influencia de los mismos en el resultado final.

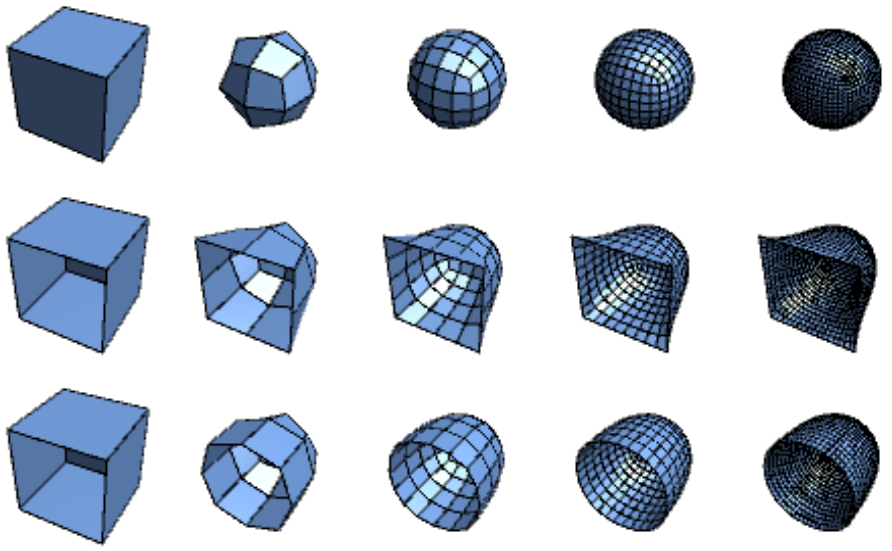


Figura 5.10: Aplicaciones del algoritmo de Catmull-Clark a un cubo cerrado (primera línea) y a un cubo abierto con variación de los pesos (líneas segunda y tercera).

Además de partir de un conjunto de puntos definidos por el usuario, podemos partir también de bases de puntos disponibles en MATHEMATICA. Es bien conocida la representación del triceratops que mostramos en la Figura 5.11, junto con la del icosaedro.

Figura 5.11: Aplicaciones del algoritmo de Catmull-Clark a un icosaedro (primera línea) y a un triceratops (segunda línea).

El algoritmo de Catmull-Clark y otros usados en subdivisión merecerían un estudio matemático más profundo. Además de los artículos originales, existen publicaciones exhaustivas sobre el tema como [3] y vídeos como [22] o [23], que pueden conformar un curso de universidad más vanguardista.

# Conclusiones

En este apartado mostraremos, por una parte, las aportaciones realizadas en este Trabajo Fin de Grado y por otra, algunas cuestiones que quedan abiertas y ofrecen nuevas vías de estudio. En cuanto a las aportaciones, destacamos las siguientes:

- Hemos trabajado con referencias que presentan varias opciones para abordar el tema de la interpolación con curvas en el caso unidimensional, como pueden ser los splines, B-splines o las curvas NURBS. En este caso, nos hemos decantado por el estudio de los polinomios de Bernstein y las curvas de Bézier, todo ello desde un enfoque analítico. Destacamos en este sentido la demostración obtenida del Teorema de Weierstrass, que asegura la convergencia de la aproximación realizada con dichos elementos matemáticos. En definitiva, hemos comprobado matemáticamente la adecuación de este tipo de curvas para los aspectos de la matemática más aplicada, como es el diseño asistido por ordenador.
- Se han estudiado también algunas de las propiedades gráfico–numéricas de las curvas de Bézier, como son la simetría, la disminución de la variación o el control local, que hacen de ellas una herramienta idónea en el campo del diseño gráfico. Muchas de las operaciones que podemos realizar con ellas, como la subdivisión o la elevación del grado son recursivas, lo que las hace fácilmente implementables desde el punto de vista informático.
- Una vez vistas las propiedades de las curvas, hemos pasado a generalizar, mediante una herencia de propiedades, las bondades de las superficies de Bézier. Todo esto se debe al estudio exhaustivo que habíamos realizado en los capítulos previos.
- En el último capítulo, que es el más práctico, hemos pasado a ver algoritmos con los que subdividir una superficie y las aplicaciones que esto tiene en el campo de la animación por ordenador. Hemos hecho uso de un lenguaje ampliamente usado, pero con el que no hemos trabajado durante el grado, como es MATLAB. Además hemos explorado funciones de MATHEMATICA que nos han permitido crear un entorno de compilación para conseguir los propósitos establecidos en este TFG.

Llegados a este punto, se abre un amplio abanico de posibilidades para seguir profundizando en estos temas:

- Una vía abierta es la del estudio formal de las propiedades que hemos obtenido. Sobre todo, en el último capítulo nos hemos centrado en la implementación algorítmica de los esquemas de subdivisión, sin considerar los resultados teóricos que soportan esta teoría. Esta es una vía de indudable interés que se puede considerear en estudios posteriores.

- Otra puede ser el estudio de las implementaciones de dichos algoritmos de refinamiento. En el desarrollo del TFG se ha utilizado software conocido como GEOGEBRA, MATLAB o MATHEMATICA. No obstante, existe la posibilidad de utilizar lenguajes como C++, PHYTON, JULIA o incluso LuaTEX. Aún así, el campo de la animación está en auge y cuenta con programas exclusivamente dedicados al modelado 3D, que llevan incorporados este tipo de algoritmos.
- En línea con el punto anterior, se podrían considerar programas de diseño mucho más específicos y sofisticados, como por ejemplo AUTOCAD, SOLIDWORKS o BLENDER.

Nuestro verdadero trabajo ha sido el de conseguir una pequeña transición entre el enfoque más clásico del problema de aproximación con polinomios y curvas, a otro más vanguardista, interdisciplinar y abierto que se encuentra en constante evolución.

# Bibliografía


[1] T. Apostol. *Mathematical Analysis; 2nd ed.* Addision-Wesley, Reading, 1974.

[2] M. P. do Carmo. *Geometría Diferencial de Curvas y Superficies.* Alianza Universidad Textos, Madrid, 1994.

[3] N. Carter. *Introduction to the Mathematics of Computer Graphics.* American Mathematical Society, Washington DC, 2016.

[4] E. Catmull, J. Clark. *Recursively generated B-spline Surfaces on Arbitrary Topological Meshes.* I.P.C. Business Press, Berkeley, 1978.

[5] W. Cheney, D. Kincaid. *Numerical Mathematics and Computing.* Thomson Brooks/Cole, Belmont, 2008.

[6] T. DeRose, M. Kass, T. Truong. *Subdivision Surfaces in Character Animation.* Pixar Animation Studios, Emeryville, 1998.
`https://graphics.pixar.com/library/Geri/paper.pdf`.

[7] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design.* Academic Press, San Diego, 1993.

[8] L. Fernández Jambrina. *Curvas y Superficies en el Diseño Geométrico Asistido por Ordenador.* E.T.S.I Navales, Universidad Politécnica de Madrid, 2013.
`https://dcain.etsin.upm.es/~leonardo/`.

[9] O. Galdames. *Modelización con Curvas y Superficies de Bézier.* Modelling in Science Education and Learning. Vol. 4, Issue 14, pp. 181–193, 2011.

[10] A. Hohmann, P. Deuflhard. *Numerical Analysis in Modern Scientific Computing.* Springer, Nueva York, 2003.

[11] M. Hosaka. *Modeling of Curves and Surfaces in CAD/CAM.* Springer, Nueva York, 1992.

[12] M. Khan. *Construction of cubic Bézier Patch and Surface.* MATLAB Central File Exchange, 2021.
`https://www.mathworks.com/matlabcentral/fileexchange/37876-\`
`construction-of-cubic-bezier-patch-and-surface`.

[13] R. Mabry. *10990.* The American Mathematical Monthly, Vol. 110, no.1, pp. 59–59, 2003.
`https://lsusmath.rickmabry.org/rmabry/bernsteinpolys-10990.pdf`

[14] A. Mertz, W.Slough. *Graphics with PGF and TikZ*. Department of Mathematics and Computer Science, Easter Illinois University, 2007.
https://www.tug.org/pracjourn/2007-1/mertz/mertz.pdf.

[15] M. Piper. *Whatever happened to the ubiquitous digital Utah Teapot?*. The Salt Lake Tribune, 2016.
https://archive.sltrib.com/article.php?id=4660789&itype=CMSID.

[16] V. Quian, M.D. Riedel, I. Rosenberg. *Uniform Approximation and Bernstein Polynomials with Coefficients in the Unit Interval*. European Journal of Combinatorics. Vol. 32, Issue 3, pp. 448–463, 2011.

[17] J. Stam. *Exact evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values*. SIGGRAPH '98: Proceedings of the 25th annual conference on Computer Graphics and iterative techniques. pp. 395–404, Toronto, 1998.

[18] F. Yamaguchi. *Curves and Surfaces in Computer Aided Geometric Design*. Springer, Berlín, 1988.

[19] Página web del Computer History Museum. *The Utah Teapot*.
https://www.computerhistory.org/revolution/computer-graphics-music-and-art/15/206.

[20] Página web de FX Guide. *Pixar's OpenSubdiv V2: a detailed look*.
https://www.fxguide.com/fxfeatured/pixars-opensubdiv-v2-a-detailed-look/.

[21] Página web de Stack Exchange. *Catmull-Clark and Doo-Sabin Subdivision Implementations*.
https://mathematica.stackexchange.com/questions/195468/catmull-clark-\and-doo-sabin-subdivision-implementations.

[22] Vídeo de YouTube del canal de *Numberphile*. B. Haran, 2014. *Math and Movies (Animation at Pixar)*.
https://www.youtube.com/watch?v=mX0NB9IyYpU.

[23] Vídeo de YouTube del canal de *U.C. Davis Academics*. K. Joy, 2014. *Subdivision Surfaces Part 2*.
https://www.youtube.com/watch?v=THiF7-QxKXk.