Análisis Numérico
Interpolación
Eddy Herrera Daza

1. Introducción

En interpolación son varios los métodos que existen para modelar curvas y superficies. No obstante, surge la pregunta como realizar a aproximaciones de objetos reales, como se ve en la Figura 1., teniendo en cuenta que el modelo matemático sea lo más cercano al objeto real

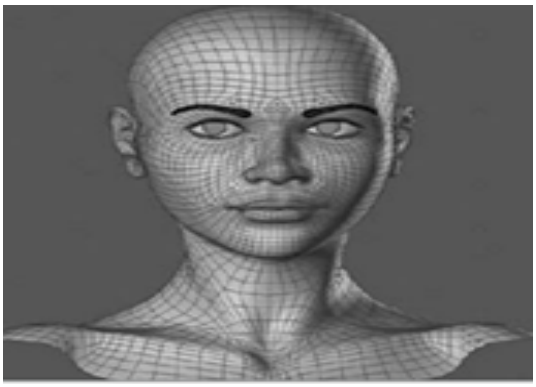

Figura 1: *Objeto real*

Por su parte, el uso de los splines conlleva a reproducir la curva o superficie a tramos pequeños, que relaciona de forma funcional relacione la información de forma matemática por ejemplo, las relaciones de primer y segundo orden. Por su parte, los polinomios de Lagrange que si pueden reproducir un tramo largo o completo de una curva,necesitan de polinomios de grado alto para obtener una buena aproximación. No obstante, una pequeña modificación de los puntos de interpolación hace muy probable que ya no nos sirve el mismo polinomio y es necesario calcularlo de nuevo.
Lo anterior, indica que la interpolación por splines cúbicos o Lagrange como metodos de interpolación polinómica no son suficientes en todos los casos. Un ejemplo de esto son las fuentes de computadora, que además son una gran demostración de arte y estética combinada sin problemas con la tecnología.

Al respecto, un joven licenciado en matemáticas llamado Paul de F. de Casteljau decide probar suerte dentro de una conocida compañía dedicada a la fabricación y venta de automóviles y el ingeniero Pierre E. Bézier, comenzaron a plantearse también el problema del diseño asistido por ordenador, y así su trabajo trascendio en todo tipo de programas de diseño (Autocad, Corel, Adobe Illustrator . . .), el lenguaje Postscript esta basado también en las curvas de Bézier, y en definitiva cualquier arquitecto o profesional del diseño conoce esta herramienta matemática

La mayoría de las fuentes en uso hoy en día se pueden cambiar a un tamaño arbitrario y aún se ven bien y sin problemas. Este tipo de fuentes se denominan fuentes de contorno (o fuentes vectoriales), a diferencia

Figura 2: *Fuente letra n*

de las fuentes de mapa de bits ya obsoletas. Tenga en cuenta que la forma de cada fuente se compone de líneas rectas y curvas. Varias fuentes utilizan principalmente dos tipos principales de esquema: esquema Truetype y esquemaPpostscript. Los contornos TrueType se representan con curvas Bézier cuadráticas, mientras que los contornos PostScript se representan con curvas Bézier cúbicas.

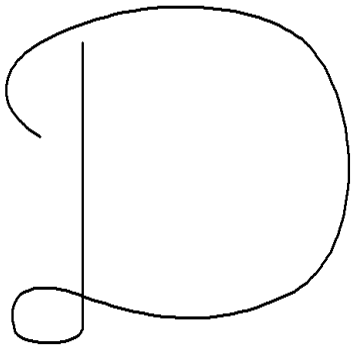

Figura 3: *Fuente letra D*

2. Curva y Superficies de Bézier

Dadas a conocer en los años 60 del siglo XX a través del trabajo de Pierre Bézier en Renault y Paul de Faget de Casteljau en Citroën. Se pueden ver como splines que, a partir de unos puntos de control, o polígono de control, permiten al usuario controlar las pendientes en esos puntos y modelizar curvas a voluntad, escalándolas sin límite. Su aplicación inicial era el diseño de carrocerías de automóviles, barcos, hélices de barcos.
Las curvas de Bézier, es un instrumento matemático para la modelización de curvas y superficies, nacieron como una aplicación concreta en el seno de la industria automovilística. Las curvas Bézier pueden ser útiles para interpolar movimiento cuando el objeto exhibe formas de movimiento curvilíneo (Long 2015). Por ejemplo, los movimientos de los mamíferos marinos a menudo exhiben esto propiedad (Tremblay et al. 2006).

Una curva de Bézier es una curva paramétrica basada en cuatro puntos de control. La curva comienza en el primer punto de control con su pendiente tangente a la línea entre los dos primeros puntos de control y la curva termina en el cuarto punto de control con su pendiente tangente a la línea entre los dos últimos puntos de control. En la Figura 3, los cuatro círculos grises son puntos de control y la línea negra es una curva de Bezier en relación con esos puntos de control.

Definición :Una curva de Bézier de grado n se especifica por una secuencia de $n+1$ puntos $P_0, ...P_1 \in R^2, R^3$ que se conocen como puntos de control(dados por el escaneo) El polígono que se obtiene al unir los puntos de control con segmentos de línea se denomina polígono de control.

Figura 4: *Curva Bezier*

Spline de Bezier
Una spline de Bezier es una curva que consta de varias curvas de Bezier unidas entre sí. Por ejemplo, en la Figura 4., hay siete puntos de control, pero el cuarto punto de control de la primera curva es también el primer punto de control de la segunda curva. Los dos últimos puntos de la primera curva sean colineales con los dos primeros puntos de la segunda curva.
Esto significa que la spline de Bezier es suave, porque las pendientes de las dos curvas de Bezier son las mismas donde se encuentran.

Polinomios de Bernstein
Las funciones de base de las curvas de Bézier son los polinomios de Bernstein —conocidos desde 1912—, formulados por Sergei Natanovich Bernstein, Rusia, 1880.
Una curva de Bézier lineal que pasa por dos puntos se define por

$$B_i^1(t) = (1-t)P_0 + tP_1 \tag{1}$$

Una cuadrática curva de Bézier, que pasa por tres puntos esta dada por:

$$B_i^2(t) = (1-t)^2P_0 + (1-t)tP_1 + t^2P_2 \tag{2}$$

Se define el polinomios de Bernstein de grado n de la siguiente manera:

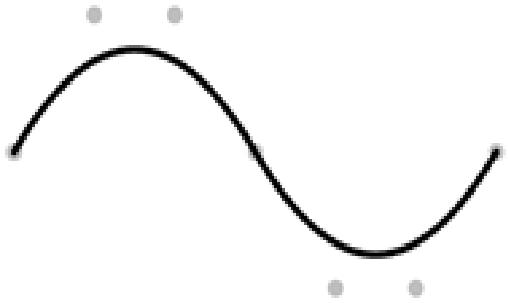

Figura 5: *Spline Bezier*

Con estos polinomios se tiene entonces, la curva (paramétrica) de Bezier dada por

$$\alpha(t) = \sum_{i=0}^{n} B_i^n(t) P_i; \forall t \in [0,1] \tag{3}$$

Los polinomios de Bernstein son también utilizados para efectuar aproximaciones e interpolaciones de funciones como la curva de Beizer, funciones de densidad de probabilidad,entre otras.

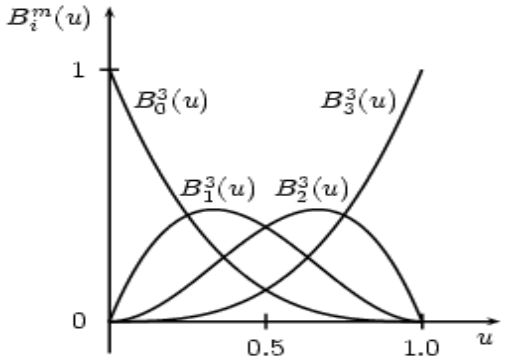

Figura 6: *Polinomios Bernstein*

Curvas de Bezier

Las curvas de Bézier son curvas paramétricas cuyas formas están controladas por un parámetro t y algunos puntos de curva de encendido y apagado. En esta parte, la forma de las curvas cuadráticas de Bézier está determinada por dos puntos en curva (o puntos finales) y un punto fuera de curva. El punto fuera de curva se usa para controlar la forma de la curva.

Supongamos que tenemos tres puntos:$(A_x, A_y); (B_x, B_y); (C_x, C_y)$ entonces,

$$P_x = (1-t)^2 A_x + 2t(1-t)B_x + t^2 C_x \tag{4}$$

$$P_y = (1-t)^2 A_y + 2t(1-t)B_y + t^2 C_y \tag{5}$$

Por lo tanto toda curva de Bezier $\alpha(t)$ de grado $\leq n$ tiene una única representación de Bezier como se ve Figura 6.

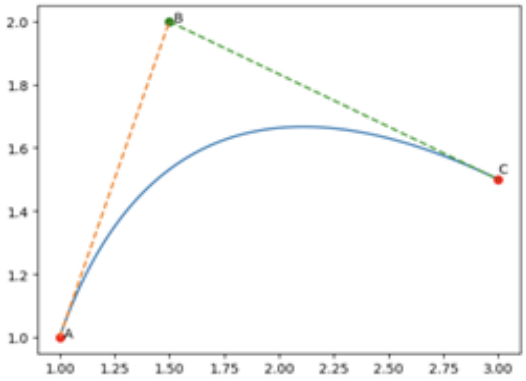

Figura 7: *Curva Cuadrática de bezier*

$$\alpha(t) = \sum_{i=0}^{n} B_i^n(t) P_i; \forall t \in [0,1] \tag{6}$$

Por su parte, la Figura 7 ilustra una curva de grado 3 con sus puntos de control y su polígono de control.

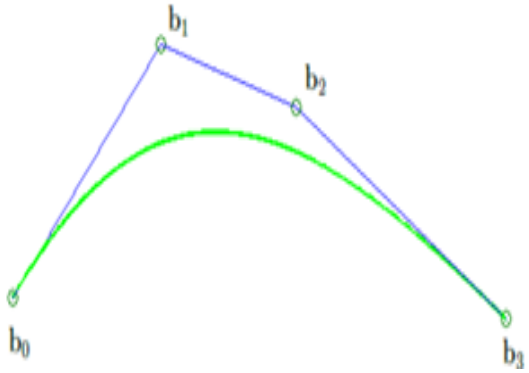

Figura 8: *Curva Bezier de grado 3*

2.1. Propiedades

1. **Interpolación de los puntos frontera**
   $\alpha(0) = P_0; \alpha(1) = P_n$

2. **Envoltura Convexa**
   La curva esta contenida en la envoltura convexa de los puntos control

3. **Invarianza Afín**
   Esta propiedad nos indica que es posible aplicar ciertas transformaciones, homotecias, traslaciones y rotaciones geométricas a $\alpha(t)$, con sólo aplicarlas a los puntos de control

4. **Invarianza bajo transformación de parametros afines**

5. **Simetría** Si invertimos el orden de los puntos de control, la curva de B´ezier resultante tras el cambio es la misma pero recorrida en sentido inverso

2.2. Aplicación

En la imagen, el carácter *e* se compone de dos contornos. Los puntos rojos son los puntos finales de las curvas B y los símbolos de cruz son los puntos de control. Puede ver que para los personajes complejos, el contorno consta de una docena de pequeñas curvas.

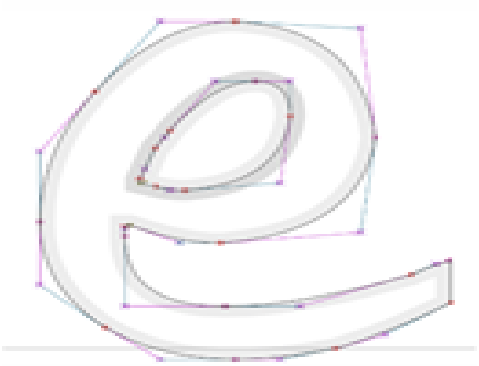

Figura 9: *Curva Bezier de la fuente e*

Un campo ejemplar en el que se exhiben y corporeizan sobremanera las implicaciones estéticas de los gráficos vectoriales es el de la tipografía digital. La explosión digital de las últimas décadas ha expandido y actualizado de forma radical el significado tradicional del término tipografía (tipo: «carácter», grafos: «escritura»). Para el arte tipográfico, la digitalización ha comportado unas posibilidades ilimitadas de creación, gestión, composición y experimentación con tipos, ahora intangibles y escalable

2.3. Superficies

Dados dos enteros positivos n, m y un conjunto de puntos de control $P_{ij} \in R^2, R^3$ una Superficie de Bezier de grado n en dirección de u y de grado m en dirección de v esta dada por:

$$\alpha(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^m(u)B_j^n(v)P_{i,j}; \forall u, v \in [0,1]X[0,1] \tag{7}$$

Un ejemplo de una superficie generada esta en la Figura 4

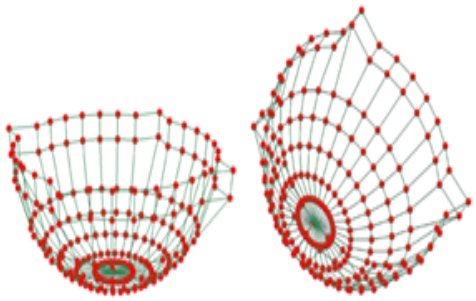

Figura 10: *Grafica de los nodos (8 niveles, cada uno representado por 32 puntos).*

3. Problema de Aproximación

El problema natural que se plantea una vez definida una curva o una superficie de Bézier, es el siguiente: Dada una curva o superficie arbitrarias, cómo podemos obtener una curva o superficie de Bézier lo más cercana posible a la original. Es decir, nos planteamos un problema de aproximación. Una vez conocidos las técnicas de aproximación como los splines cúbicos, los polinomios de Lagrange o mínimos cuadrados, estas tienen sus ventajas y desventajas. El denominador común en todas ellas es que se toma una muestra no aleatoria y finita de puntos del objeto a aproximar y se intenta conseguir un objeto similar aproximándose a estos puntos.
Luego, dada una lista de $k+1$ puntos $Q_0, .., Q_k$ del plano real se busca una curva polinómica $\alpha[0,1] \longrightarrow R_2$ talque se cumpla $||Q_i - \alpha(t_i)|| < \varepsilon$ en otras palabras, dados el vector de los puntos de control **P** se debe solucionar el sistema $MP = Q$

Reto 2A de Interpolación con Curvas-Superficies de Bézier
El objetivo propuesto es conseguir la recontrucción que se muestra en la Figura 7 un jarrón y sus curvas de nivel, que se pueden utilizarar para reconstruir del jarrón, usando superficies de Bezier y/o otro método (BSplines). Para ello se puede utilizar R(PathInterpolatR, gridBezier,vwline) o Python(griddata, matplotlib). También, se puede utilizar el algoritmo Marching Cube

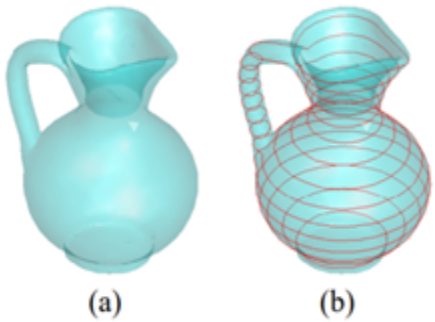

Figura 11: *a. Jarro. b. Curvas de nivel*

Sugerencia

I Se puede utilizar la herramienta MATHEMATICA,con la función Graphics3D, teniendo como parámetro la función BSplineSurface que representa las B-splines racionales no uniformes (NURBS) definidas por un arreglo que contiene los puntos de control en el plano XYZ. Esta función usa por defecto splines bicúbicos con grado.

II Se suguiere dividir la figura (utilizar la heuristica de divide y venceras), de manera que una vez construido uno, el resto puede representarse realizando rotaciones o con estiramientos, adiciones etc.

III La figura no puede representarse mediante una unica y la superficie se puede dividir de manera eficiente.

IV Para el caso de superficies la derivada es obviamente direccional, pero la idea es la misma.

V Adicionalmente,para la captura de los puntos de control existen varias opciones, entre ellas estaba la de modelar una imagen de la figura para poder así obtener los puntos de control, siendo los más apropiados Blender y AutoCAD. Blender una herramienta gratuita con código abierto que cuenta con una de las mayores y más activas comunidades para la creación de modelos en 3D.

Referencias

[1] Long, JA (2015) Kinematic interpolation of movement data. International Journal of Geographical Information Science. DOI: 10.1080/13658816.2015.1081909.

[2] Tremblay, YC et al. (2006) Interpolation of animal tracking data in a fluid environment. Journal of Experimental Biology. 209(1): 128-140.
[3] W.E. Lorensen and H. E. Cline, A high resolution 3D surface construction Algorithm, Proceedings of SIGGRAPH'87, Vol 21, No. 4, pp. 163-169.
[4] Blender. Recuperado de https://www.blender.org/
[5] PyCharm. Recuperado de https://www.jetbrains.com/pycharm/
[6] Wolfram Language System Documentation Center Graphics3D.
Recuperado de https://reference.wolfram.com/language/ref/Graphics3D.html
[7] Wolfram Language System Documentation Center BSplineSurface. Recuperado de
https://reference.wolfram.com/language/ref/BSplineSurface.html
[8]Wolfram Language System Documentation Center Show.
Recuperado de https://reference.wolfram.com/language/ref/Show.html
[9] Wolfram Language System Documentation Center Manipulate.
Recuperado de https://reference.wolfram.com/language/ref/Manipulate.html
[10] Wolfram Language System Documentation Center BSplineFunction. Recuperado de
https://reference.wolfram.com/language/ref/BSplineFunction.html [11] MATHEMATICA DE WOLFRAM.
Recuperado de https://www.wolfram.com/mathematica

Anexos

```
#"Construccion de letras"

library(grid)
library(gridBezier)
x1 <-c(0.23, 0.23 , 0.21 , 0.1 )
y1 <- c(0.57, 0.4 , 0.3 ,0.1)
x2 <- c(0.1,0.6,0.7,0.53)
y2 <- c(0.1,0.7,0.6,0.3)
x3<-c(0.53,0.4,0.45,0.5)
y3<-c(0.3,0,0,0.1)
grid.newpage()
grid.bezier(x1,y1,gp=gpar(lwd = 3, fill="black"))
grid.bezier(x2,y2,gp=gpar(lwd = 3, fill="black"))
grid.bezier(x3,y3,gp=gpar(lwd = 3, fill="black"))
```

Figura 12: *Implementación en R de la letra n*

```
library(gridBezier)
x <- c(.2, .2, .8, .8, .8, .2)/3
y <- c(.5, .8, .8, .5, .2, .2)
grid.circle(x, y, r=unit(1, "mm"), gp=gpar(col=NA,
                                           fill="grey"))
grid.Bezier(x[1:4], y[1:4], gp=gpar(lwd=3))
grid.Bezier(x[c(1:6, 1)] + 1/3, y[c(1:6, 1)],
            gp=gpar(lwd=3))
grid.Bezier(x[1:6] + 2/3, y[1:6], open=FALSE,
            gp=gpar(lwd=3,
                                    fill="grey"))

###Gruesa
library(vwline)
x <- c(.1, .2, .3, .4, .5, .6, .7)
y <- c(.4, .7, .7, .4, .1, .1, .4)
grid.offsetBezier(x, y, w=unit(0:1, "cm"))
```

Figura 13: *Implementación en R de curvas de Bezier*