Types of Splines

CS116B
Chris Pollett
Feb 7, 2005.
Outline

• Spline Representations
• Cubic-Spline Interpolation
Spline Representations

• Last day we said how to represent a spline using its parametric cubic \((x(u), y(u))\) where \(u\) is in \([0,1]\), the endpoint values \((x(0), y(0)), (x(1), y(1))\) and derivatives at the endpoints \((x'(0), y'(0))\) and \((x'(1), y'(1))\). Here \(x(u)\) looks like: \(a_x u^3 + b_x u^2 + c_x u + d_x\). and \(y(u)\) is similar.

• Today we’ll look at different ways to represent this same equations
Other Representations

• Note could write \( x(u) \) as:
  \[
  [u^3 \; u^2 \; u \; 1][a_x \; b_x \; c_x \; d_x]^t = U \cdot C \quad \text{(the little } t \text{ is for transpose)}
  \]

• One can also write \( C \) as \( M_{\text{spline}} \cdot M_{\text{geom}} \) where
  \( M_{\text{spline}} \) is the boundary values on the spline and
  \( M_{\text{geom}} \) contains a 4x4 matrix based on the
  coordinates of the control points.

• Thus, \( x(u) \) is sometimes written as \( U \cdot M_{\text{spline}} \cdot M_{\text{geom}} \) which is called the basis matrix.

• Finally, sometimes see splines represented in terms of coordinates of control points as
  \( x(u) = \sum_{k=0}^{3} g_k \cdot BF_k \{u\} \) where \( BF_k \) are called blending functions.
Spline Surfaces

- The splines discussed so far were 2D. Can also specify surfaces.
- To do this need to specify sets of orthogonal spline curves using a mesh of control points.
- Might have an equation like:

\[ P(u,v) := \sum_{i,j} p_{i,j} B_{F_i}(u) B_{F_j}(v) \]
Trimming Spline Surfaces

• Sometimes in CAD applications it is useful to be able to trim out sections from a spline surface.

• Some graphics packages have facilities to generate trimming curves to support this
Cubic-Spline Interpolation Methods

- Rather than using general splines of arbitrary degree, cubic splines are often used to design objects because they are reasonably flexible and can be computed and stored efficiently.
- Consider the curve
  
  \[ p_0, p_2, \ldots, p_k, \ldots, p_n \]

- One wants to be able to approximate this curve with cubic splines.
More on Cubic Spline Interpolation

• Between each pair of control points \( p_k, p_{k+1} \) we will try to find a best approximating cubic spline. \((x(u), y(u), z(u))\) where \(x(u), y(u), z(u)\) are cubics in \(u\).

• To do this we need to set enough boundary conditions at the endpoints of a segment so this spline is uniquely determined.

• Next few slides discuss different ways to do this
Natural Cubic Splines

• In these kind of spline, if have $n+1$ control points then we specify $n$ cubic splines.

• We specify the values of the spline, its first and second derivative, at each of its endpoints.

• We require adjacent splines to have matching values at the endpoints.

• To complete the description usual set the first and second derivative of $p_0$ and $p_n$ to be 0.
Hermite Interpolation

• Natural splines are hard to locally update
• For Hermite splines we specify the value of the tangent at each control point rather than say that this value must be equal among adjacent curves.
• For example, if want to specify curve $P$ between two points $p_k, p_{k+1}$, would use equations:

$P(0) = p_k$
$P(1) = p_{k+1}$
$P'(0) = Dp_k$ Here $Dp_k$ is some fixed value like 4.
$P'(1) = Dp_{k+1}$
More on Hermite Interpolation

Now $P(u) = au^3 + bu^2 + cu + d$ for $0 \leq u \leq 1$

- That is $P(u) = [u^3 \ u^2 \ u \ 1][a \ b \ c \ d]^t$
- Taking the derivative we have $P'(u) := [3u^2 \ 2u \ 1 \ 0] \ [a \ b \ c \ d]^t$
- Substituting values 0 and 1 for $u$ in the above gives us the matrix equation:

$$\begin{bmatrix}
  p_k \\
  p_{k+1} \\
  p_k \\
  p_{k+1}
\end{bmatrix} =
\begin{bmatrix}
  0 & 0 & 0 & 1 \\
  1 & 1 & 1 & 1 \\
  0 & 0 & 1 & 0 \\
  3 & 2 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
  a \\
  b \\
  c \\
  d
\end{bmatrix}$$

Call the inverse matrix $M_H$. Can solve now for $a \ b \ c \ d$ by taking $M_H$. 
Yet More on Hermite Interpolation

• So $M_H$ is:

\[
\begin{bmatrix}
2 & -2 & 1 & 1 \\
-3 & 3 & -2 & -1 \\
0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}
\]

• $P(u) = [u^3 \ u^2 \ u \ 1] \ M_H [p_k \ p_{k+1} \ Dp_k \ Dp_{k+1}]^t$

• From which we can calculate the blending functions:

\[
P(u) = p_k(2u^3 - 3u^2 + 1) + p_{k+1}(-2u^3 + 3u^2) + Dp_k(u^3 - 2u^2 + u) + Dp_{k+1}(u^3 - u^2)
\]

\[
P(u) = p_kH_0(u) + p_{k+1}H_1(u) + Dp_kH_2(u) + Dp_{k+1}H_3(u)
\]