Review of Splines and Introduction to Subdivision

Lecture #12: Thursday, Nov 6 1997
Lecturer: Denis Zorin
Scribe: Xiaomei Zhu

In this lecture we will review some simple facts about splines and introduce subdivision. Next few lectures will be about subdivision.

1 The Idea of Subdivision.

Splines and subdivision are closely related. Splines use piecewise polynomials to interpolate curves. There are many ways to evaluate splines. One such method is the de Casteljau algorithm; for quadratic splines it is especially simple. Start with the control polygon of a spline curve; insert two new control points on each edge connecting two old control points. Each new control point divides the edge in the ratio 3:1. Connect the new control points and discard the old.

This algorithm can be generalized: note that each new control point is a linear combination of old control points with weights 1/4 and 3/4. Rather then using these specific weights, we can use other weights. This is the idea of subdivision: use algorithms similar to those that are used for spline evaluation, but modify these algorithms so that they generate curves with different properties (which becomes apparent in the one-dimensional case) and so that they handle arbitrary control meshes for surfaces (which becomes an issue in the two-dimensional case).

2 Subdivision and other modeling methods.

We briefly compare subdivision to several free-form surface and curve representations used in geometric modeling: splines, implicit surfaces, and variational surfaces. All of these representations are primarily used for smooth surfaces.

We can compare these representations from a number of points of view.

1. *Efficiency.* Computational cost is an important aspect of a modeling method. Subdivision is easy to implement and is computationally efficient; the efficiency is a property inherited from splines since the algorithms for evaluation are essentially the same. From the point of view of efficiency, implicit surfaces are not so good. An algorithm such as marching cubes is required to generated the polygonal approximation needed for rendering. Variational surfaces are even worse: a global optimization problem has to be solved each time a control of the surface is changed.

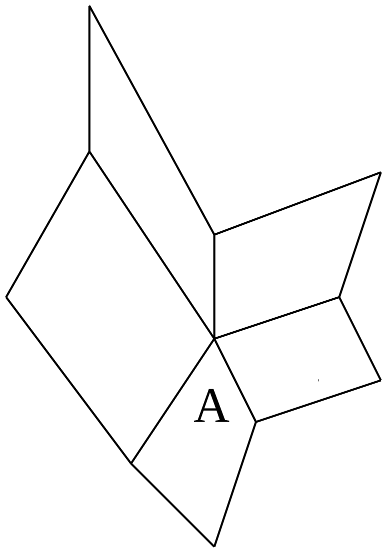

Figure 1: A mesh with an extraordinary vertex: number of faces meeting at A is not 4. Special efforts are required to guarantee smoothness between spline patches meeting at the extraordinary point; subdivision handles such situations in a natural way.

2. *Arbitrary topology.* It is important to be able to produce smooth surfaces with control meshes of arbitrary topology. "Arbitrary topology" means two things: first, the topological genus of the mesh and associated surface can be arbitrary. Second, the structure of the graph formed by the edges and vertices of the mesh can be arbitrary; specifically, each vertex may have an arbitrary degree.

   These two aspects are related: for example, any control mesh for a sphere (topological genus 0) necessarily has vertices of different degrees. Only closed surfaces of genus 1 can be represented using a mesh with all vertices having the same degree.

   From a practical point of view arbitrary control meshes have a considerable advantage: a user of a modeling system would not want to worry about degrees of control vertices when defining a shape.

   When rectangular spline patches are adapted for arbitrary control meshes, enforcing smoothness at extraordinary vertices becomes difficult and considerably increases the complexity of the representation. Implicit surfaces can be of arbitrary topological genus, but the genus, precise location, and even connectivity of a surface are difficult to control typically. Variational surfaces can handle arbitrary topology better than any other representation, but the computational cost has been thus far prohibitive. Subdivision canhandle arbitrary topology quite well without losing efficiency; this is one of its key advantages.

3. *Surface features.* Subdivision allows more flexible controls than is possible with splines. In addition to choosing locations of control points, one can manipulate

the coefficients of subdivision to achieve such effects such as the the sharpness of creases or the control behavior of the boundary curves. Implicit surfaces, on the other hand, are very difficult to control. Variational surface, however, are the best for creating features.

4. *Complex geometry.*

   For interactive applications, efficiency is crucial. Subdivision-based representations for complex geometry are very efficient and have an advantage in situations when the geometry has to be modified (editing, animation). For applications that only require the visualization of the geometry, other representations, such as H. Hoppe's progressive meshes, are likely to be more suitable.

3 Review of Splines

3.1 Piecewise-Polynomial Curves

Splines are piecewise polynomial curves. In the case of quadratic splines, each polynomial segment of a curve can be written as

$$\begin{aligned} x(t) &= a_2t^2 + a_1t + a_0 \\ y(t) &= b_2t^2 + b_1t + b_0 \end{aligned}$$

Where a_2, a_1, a_0, b_2, b_1, and b_0 are constant coefficients. Spline curves are typically specified by *control points* rather then by the coefficients. The curve can be modified by moving the control points. Moving a control point has the greatest effect on the part of the curve near that control point. For B-splines, the effect of changing the position of a single control point is actually local: only a finite portion of the curve is affected.

Specifying the curve with control points amounts to rewriting the equations above in the following form.

$$\begin{aligned} x(t) &= \sum C_x^i B_i(t) \\ y(t) &= \sum C_y^i B_i(t) \end{aligned}$$

$B_i(t)$ are called the basis functions. The collection of basis functions for $i = 0, 1, 2, ..., n$ is the basis for the space of all curves that we can represent. For uniform splines, the basis functions satisfy

$$B_i(t) = B(t - i)$$

We choose the basis function $B_i(t)$ in such a way that the resulting curves are smooth and that the influence of the control points is local. One way to ensure smoothness is to use smooth basis functions. [1] Since polynomials themselves are inifinitely smooth, we only have to make sure that derivatives match at the points where two polynomial segments meet. The higher the degree of the polynomial, the more derivatives we are able to match. We also want the influence of a control point to be maximal at regions of the curve close to it, and for this influence to decrease as we move away along the curve, and for it to disappear completely at some distance. Finally, we would like the basis functions to be piecewise polynomial, and we should be able to represent any piecewise-polynomial curve of a given degree with the apropriate basis. In a moment, we will show how to construct basis functions satisfying all these requirements for polynomial curves of arbitrary degree.

When we talk about curves, it is important to distinguish the curve itself and the graphs of the coordinate functions of the curve (which also can be thought of as curves). For example, a curve can be described by equations $x(t) = sin(t)$, $y(t) = cos(t)$. The curve itself is a circle, but the coordinate functions are sinusoids. For the moment, we are going to concentrate on representing the coordinate functions.

3.2 B-Splines Basis Functions

We start with the simplest case: piecewise constant coordinate functions. Any piecewise constant function can be written as

$$x(t) = \sum c_i^x U_i(t)$$

where $U_i(t)$ is the box function defined as

$$\begin{aligned} U(t) &= 1 \quad \text{if} \quad 0 \leq t < 1 \\ &= 0 \quad \text{otherwise} \end{aligned}$$

The functions $U_i(t) = U(t-i)$ are translates of U(t). Furthermore, let us represent The continuous convolution of two functions $f(t)$and $g(t)$ with

$$(f \otimes g)(t) = \int f(s)g(t-s)ds$$

Now a B-spline basis function of degree n can be obtained by convolving the basis function of degree $n-1$ with the box $U(t)$. For example, the B-spline basis function of degree 1 is defined as the convolution $U(t)$ with itself. We need to compute

$$\int U(s)U(t-s)ds$$

Graphically, this convolution can be evaluated by sliding one box function along the coordinate axis from minus to plus infinity while keeping the second box fixed. The value

[1] The smoothness of the basis functions guarantees the smoothness of the coordinate functions of the curve. However, it does not guarantee the geometric smoothness of the curve. We will return to this distinction in our discussion of subdivision surfaces.

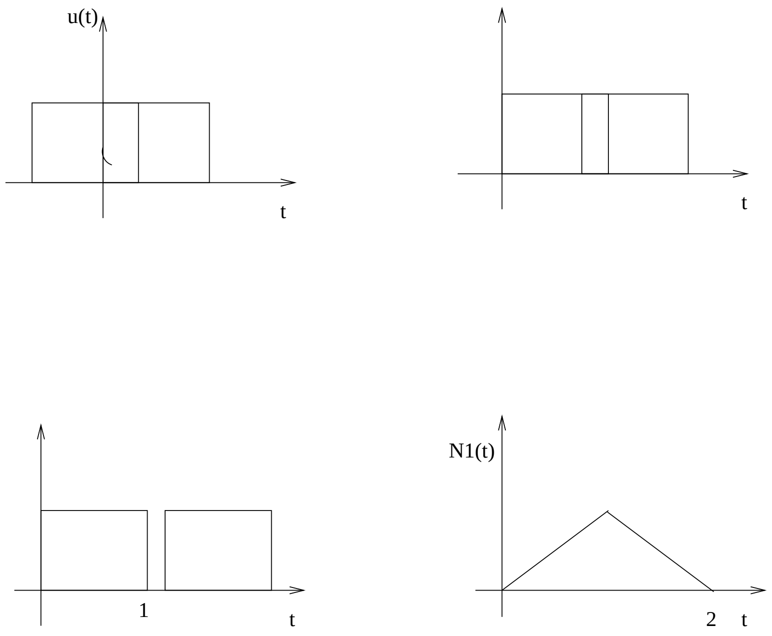

Figure 2: The definition of degree 1 B-Spline basis function through convolution of $U(t)$ with itself.

of the convolution for a given position of the moving box is the area under the product of the boxes, which is just the length of the interval where both boxes are non-zero. At first the two graphs do not have common support. Once the moving box reaches 0, there is a growing overlap between the supports of the graphs. The value of the convolution grows with t until $t = 1$. Then the overlap starts decreasing, and the value of the convolution decreases down to zero at $t = 2$. The function $N_1(t) = \int U(s)U(t-s)ds$ is the linear hat function as shown in Figure 2.

We can compute the B-spline basis function of degree 2 convolving $N_1(t)$ with the box $U(t)$ again.

$$N_2(t) = \int N_1(s)U(t-s)ds$$

In this case, the resulting curve consists out of three quadratic segments defined on intervals $(0,1)$, $(1,2)$ and $(2,3)$. This is illustrated in Figure 3.

In general, by convolving j times, we can get a B-spline basis function of degree j

$$N_j(t) = \int N_{j-1}(s)U(t-s)ds$$

Now convolution has a remarkable property

Theorem 1. *If $f(t)$ is C^k-continuous, then $U(t) \otimes f(t)$ is C^{k+1}-continuous.*

It follows from this theorem that the B-spline basis function of degree n is C^{n-1} continuous because the basis function of degree 1 is C^0-continuous. Defining B-spline basis functions

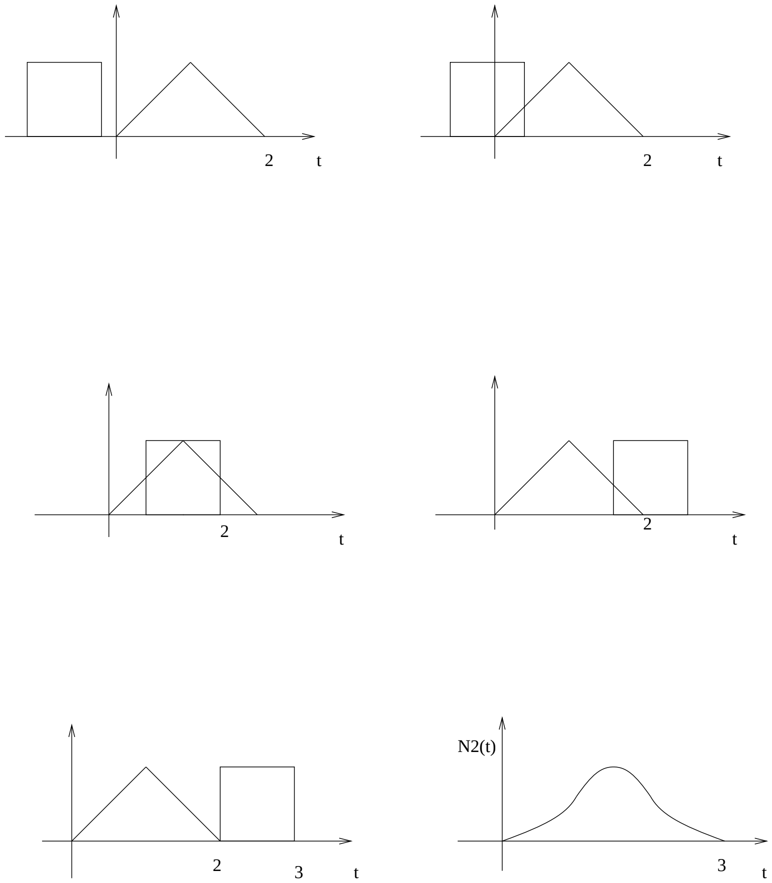

Figure 3: $N_2(t)$ as convolution of $N_1(t)$ with $U(t)$.

through convolution leads us to a scaling equation for splines similar to the dilation equation that we had for scaling functions of wavelet bases. Recall that $U(t)$ is Haar scaling function and that it satisfies the dilation (scaling) equation

$$U(t) = U(2t) + U(2t-1) \tag{1}$$

We have defined the B-spline basis function of degree j as

$$N_j(t) = \otimes_{i=0}^{j} U(t) \tag{2}$$

We can obtain a scaling equation for the basis function $N_j(t)$, which is actually a special case of deBoor's formula. Observe that convolution is a linear operation

$$\begin{aligned} f \otimes (g+h) &= f \otimes g + f \otimes h \\ f \otimes (ag) &= a(f \otimes g) \end{aligned}$$

where f and g are functions and a is a constant. By substituting Equation 1 into Equation 2 and expanding, we obtain an expression of the form

$$N_j(t) = \sum s_k N_j(2t - k)$$

where s_k are some constants. Next time we will show that the coefficients s_k allow us to compute successive piecewise-linear approximations to the limit curve starting with the control polygon.