9901 Geometric Computing Lecture 4 - 01/01/2010

Interpolation Methods

Lecturer: Abel Gomes *Scribe: A. Gomes*

Summary: In this lecture we review the fundamental concept of *interpolation* and its possible applications in computer-aided geometric design, and start considering basic constructive methods for curves and surfaces. We discuss curves and surfaces in more detail in future lectures.

Contents:

4.1 Introduction
4.1.1 Linear Interpolation
4.1.2 Cosine Interpolation
4.1.3 Cubic Interpolation
4.1.4 Hermite Interpolation

4.2 Linear Interpolation
4.2.1 Linear Interpolation and Barycentric Coordinates
4.2.2 Linear Interpolation and Geometric Ratios
4.2.3 Linear Interpolation over [a,b]

4.3 Piecewise Linear Interpolation

4.4 Repeated Linear Interpolation

4.5 de Casteljau Algorithm

4.6 Final Remarks

4.1 Introduction

We discuss here a number of interpolation methods that we commonly find in computer graphics and geometric modeling. Interpolation means to calculate a point or several points between two given points. For a given sequence of points, this means to estimate a curve that passes through every single point.

4.1.1 Linear Interpolation

Linear interpolation is the simplest interpolation method. Applying linear interpolation to a sequence of points results in a polygonal line where each straight line segment connects two consecutive points of the sequence. Therefore, every segment (P, Q) is interpolated independently as follows:

$$P(t) = (1 - t)\,.P + t\,.Q \tag{1}$$

where $t \in [0, 1]$. By varying t from 0 to 1 we get all the intermediate points between P and Q. Note that $P(t) = P$ for $t = 0$ and $P(t) = Q$ for $t = 1$. For values of $t < 0$ and $t > 1$ result in extrapolation, that is, we get points on the line defined by P, Q, but outside the segment (P, Q).

4.1.2 Cosine Interpolation

As shown in Fig. **??**, the curve resulting from Linear interpolation has discontinuities at each point. In certain circumstances, we need a smoother interpolating function, that is a function that allows for a smooth transition between consecutive segments. The cosine interpolation carries out a transition that looks smooth, though every segment is interpolated independently too.

http://local.wasp.uwa.edu.au/ pbourke/miscellaneous/interpolation/

4.1.3 Cubic Interpolation

...

4.1.4 Hermite Interpolation

...

4.2 Linear Interpolation

Let us now define linear interpolation in more mathematical terms.

Definition 1. *A* linear interpolation $f : [0, 1] \to \mathbb{R}^n$, $t \mapsto f(t) = (f_1(t), \ldots, f_n(t))$ *is an affine transformation from an unit interval* $[0, 1]$ *to a straight line segment in* $\mathbb{R}^n$*, where* $f_1(t), \ldots, f_n(t)$ *are the function components of* f *along each coordinate axis.*

See Lecture 1 for more details on affine transformations. By definition, an affine transformation preserves barycentric combinations. Therefore, if $t \in [0, 1]$ is defined as a barycentric combination of the points $0, 1 \in \mathbb{R}$

$$t = \alpha_0\,.0 + \alpha_1\,.1, \quad \text{with } \alpha_0 + \alpha_1 = 1$$

then,

$$f(t) = \alpha_0\,.f(0) + \alpha_1\,.f(1), \quad \text{with } f(0), f(1) \in \mathbb{R}^n$$

with $\alpha_0 = 1-t$ and $\alpha_1 = t$. This is illustrated in Fig.??, where we intuitively see that the linear interpolation preserves the ratio $\frac{f(t)-f(0)}{f(1)-f(t)} = \frac{t-0}{1-t}$.

4.2.1 Linear Interpolation and Barycentric Coordinates

Let us first see the relation between collinearity and barycentric coordinates. Let P_0, P, P_1 be three collinear points in $\mathbb{R}^3$. Then, P is the barycentric combination of P_1 and P_2 given as follows:

$$P = \alpha_0 P_0 + \alpha_1 P_1, \quad \text{with} \quad \alpha_0 + \alpha_1 = 1$$

where α_0 and α_1 are the barycentric coordinates of P with respect to P_0 and P_1, that is

$$\alpha_0 = \frac{D(P, P_1)}{D(P_0, P_1)} \quad \text{and} \quad \alpha_1 = \frac{D(P_0, P)}{D(P_0, P_1)}$$

where $D(,)$ denotes the signed Euclidean distance between two points.

We are now at a position that allows to show that the linear interpolation is given by Eq. (1). Taking into consideration the above expressions for α_0 and α_1, and the fact that a linear interpolation preserves barycentric coordinates, we have:

$$P(t) = \alpha_0 P_0 + \alpha_1 P_1, \quad \text{with} \quad \alpha_0 + \alpha_1 = 1$$

where

$$\alpha_0 = \frac{t-1}{0-1} = 1 - t \quad \text{and} \quad \alpha_1 = \frac{0-t}{0-1} = t$$

that is,

$$P(t) = (1-t)\,.P_0 + t\,.P_1$$

4.2.2 Linear Interpolation and Geometric Ratios

By definition, the ratio of three collinear points P_0, P, and P_1 is given by

$$r(P_0, P, P_1) = \frac{D(P_0, P)}{D(P, P_1)}$$

Taking into account the expressions of the barycentric coordinates α_0, α_1 given above, we have

$$r(P_0, P, P_1) = \frac{\alpha_1}{\alpha_0}$$

We know that an affine transformation $f : [0, 1] \to \mathbb{R}^n$ preserves barycentric coordinates; as a consequence, the ratio of barycentric coordinates is also preserved. Therefore, $r(P_0, P, P_1)$ remains unchanged by affine transformations, that is,

$$r(f(P_0), f(P), f(P_1)) = \frac{\alpha_1}{\alpha_0}$$

In short, an affine transformation preserves the geometric ratio of collinear points, that is, the image of a straight line segment is a straight line segment.

4.2.3 Linear Interpolation over $[a, b]$

The interval $[a, b]$ can be obtained from the affine transformation of the interval $[0, 1]$. With $t \in [0, 1]$ and $u \in [a, b]$, this affine trnsformation is given by

$$t = \frac{u-a}{b-a}$$

so, replacing the expression of t into

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

we have

$$P(u) = \frac{b-u}{b-a} P_0 + \frac{u-a}{b-a} P_1$$

Because a, u, b and $0, t, 1$ have the same geometric ratio as P_0, P, P_1, we end up showing that the linear interpolation is invariant under affine domain mappings. By affine domain mapping we mean an affine transformation from the real line to itself.

4.3 Piecewise Linear Interpolation

Piecewise linear interpolation involves not two points but a sequence of points $P_0, P_1, \ldots, P_N \in \mathbb{R}^n$ such that a linear interpolation is applied to two consecutive points of this sequence. The result is a polyline $\mathbf{P}$, called *piecewise linear interpolant* of all points $P_0, P_1, \ldots, P_N$. This is illustrated in FIg. **??**, where we determine a point in each segment for every $t \in [0, 1]$.

If the points $P_0, P_1, \ldots, P_N$ are on a curve $\mathcal{C}$, we say that the resulting polyline $\mathbf{P}$ is a *piecewise linear interpolant* of the curve $\mathcal{C}$; symbolically, we write $\mathbf{P} = \mathbf{P}(\mathcal{C})$.

The piecewise linear interpolation enjoys two properties, as described in the sequel.

Property L4.1 (*Affine Invariance*)

If a curve $\mathcal{C}$ is subject to an affine transformation f, then a piecewise linear interpolant of $f(\mathcal{C})$ is an affine transformation of the original piecewise linear interpolant, that is,

$$\mathbf{P}(f(\mathcal{C})) = f(\mathbf{P}(\mathcal{C}))$$

Property L4.2 (*Variance Diminishing*)

Let $\mathbf{P}(\mathcal{C})$ be a piecewise linear interpolant of the curve $\mathcal{C}$, and π an arbitrary hyperplane that intersects both $\mathcal{C}$ and $\mathbf{P}(\mathcal{C})$. Then, we have

$$\#\big(\pi \cap (\mathbf{P}(\mathcal{C}))\big) \leq \#\big(\pi \cap (\mathcal{C})\big)$$

that is, the number of intersection points between the plane π and the interpolant is less or equal to the number of points resulting from the intersection between π and the curve.

This is so because, unlike a straight line segment of the interpolant, the curve segment passing through the two endpoints of such a straight line segment is not necessarily convex.

The Menelaus Theorem

Let us now have a look at an important theorem in the context of piecewise linear interpolation.

Theorem 2. *Let $A, B, C \in \mathbb{R}^2$ be three points defining two straight lines that meet at B, and D, E, F points in the lines defined by (B, C), (A, C), and (A, B), respectively, each one of which is distinct from the vertices of the triangle ΔABC. Then, the points D, E, F are said to be* collinear *if and only if*

$$\frac{\delta(A,F)}{\delta(F,B)} . \frac{\delta(B,D)}{\delta(D,C)} . \frac{\delta(C,E)}{\delta(E,A)} = -1$$

Proof Let us consider the piecewise linear interpolant of the points P_0, P_1, P_2. Let us apply the same linear interpolation to two points $t, u \in [0, 1] \subset \mathbb{R}$ in a way we get two image points $P(t), P(u)$ in the straight line segment (P_0, P_1), and other two image points $Q(t), Q(u)$ in the straight line segment (P_1, P_2) in $\mathbb{R}^2$, as illustrated in Fig. **??**.

We intend to prove that

$$\frac{\delta(Q(u),Q(t))}{\delta(Q(t),P_1)} \cdot \frac{\delta(P_1,P(t))}{\delta(P(t),P(u))} \cdot \frac{\delta(P(u),P)}{\delta(P,Q(u))} = -1$$

For that purpose, we have only to determine the unknown third ratio $\frac{\delta(P(u),P)}{\delta(P,Q(u))}$, that is, we have to determine the barycentric coordinates of P. Taking into account that P is a barycentric combination of both straight line segments $(P(u),Q(u))$ and $(P(t),Q(t))$, we have

$$\begin{cases} P = \alpha_0 P(t) + \alpha_1 Q(t) & \text{with} \quad \alpha_0 + \alpha_1 = 1 \\ P = \alpha'_0 P(u) + \alpha'_1 Q(u) & \text{with} \quad \alpha'_0 + \alpha'_1 = 1 \end{cases} \tag{2}$$

Now, let us substitute the expressions of

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

$$Q(t) = (1-t)P_1 + tP_2$$

and

$$P(u) = (1-u)P_0 + uP_1$$

$$Q(u) = (1-u)P_1 + uP_2$$

into (7) so that we get

$$\begin{cases} P = \alpha_0(1-t)P_0 + [\alpha_0 t + \alpha_1(1-t)]P_1 + \alpha_1 tP_2 \\ P = \alpha'_0(1-u)P_0 + [\alpha'_0 u + \alpha'_1(1-u)]P_1 + \alpha'_1 uP_2 \end{cases} \tag{3}$$

By combining (7) and ((3)), we have

$$\begin{cases} \alpha_0(1-t) = \alpha'_0(1-u) \\ \alpha_1 t = \alpha'_1 u \\ \alpha_0 + \alpha_1 = 1 \\ \alpha'_0 + \alpha'_1 = 1 \end{cases} \tag{4}$$

that is,

$$\begin{cases} \alpha_0 = \alpha'_0 \frac{1-u}{1-t} \\ \alpha_1 = \alpha'_1 \frac{u}{t} \\ (1-\alpha'_1)\frac{1-u}{1-t} + \alpha'_1 \frac{u}{t} = 1 \\ \alpha'_0 = 1 - \alpha'_1 \end{cases} \tag{5}$$

or, equivalently,

$$\begin{cases} \alpha_0 = 1-u \\ \alpha_1 = u \\ \alpha'_1 = t \\ \alpha'_0 = 1-t \end{cases} \tag{6}$$

Substituting these barycentric coordinates in (7), we obtain

$$\begin{cases} P = (1-u)P(t) + uQ(t) \\ P = (1-t)P(u) + tQ(u) \end{cases} \tag{7}$$

Finally, we can write down

$$\frac{\delta(Q(u),Q(t))}{\delta(Q(t),P_1)} . \frac{\delta(P_1,P(t))}{\delta(P(t),P(u))} . \frac{\delta(P(u),P)}{\delta(P,Q(u))} = \frac{u-t}{t} . \frac{1-t}{-(u-t)} . \frac{-t}{-(1-t)} = -1$$

□

4.4 Repeated Linear Interpolation

Let us now see how *repeated linear interpolation* allows for a procedure to construct parabolas. As we will see in lectures to come, the generalization of this procedure leads us to the construction of Bézier curves.

So, let P_0, P_1, P_2 be three points in $\mathbb{R}^2$. Using piecewise linear interpolation, we determine two points for a given $t \in \mathbb{R}$, one in the straight line defined by (P_0, P_1) and another defined by (P_1, P_2), as follows:

$$\begin{cases} P_0^1(t) = (1-t)P_0 + tP_1 \\ P_1^1(t) = (1-t)P_1 + tP_2 \end{cases} \tag{8}$$

where the exponent 1 indicates the degree of the polynomial. By applying the *same* linear interpolation at $t \in \mathbb{R}$ to the new segment $(P_0^1(t), P_1^1(t))$, we have

$$P_0^2(t) = (1-t)P_0^1(t) + tP_1^1(t) \tag{9}$$

Substituting the expressions of $P_0^1(t)$ and $P_1^1(t)$ given by (8) into (9), we obtain

$$P_0^2(t) = (1-t)^2 P_0 + 2t(1-t)P_1 + t^2 P_2 \tag{10}$$

which is a degree-2 polynomial representing a parabola. This construction procedure for parabolas uses repeated linear interpolation.

Property L4.3 (*Convex Hull*)

The construction of a parabola using repeated linear interpolation enjoys the following the convex hull property, because
$P_0^2(0) = P_0$ for $t = 0$;
$P_0^2(1) = P_1$ for $t = 1$; and
$P_0^2(t)$ is within the convex hull of the points P_0, P_1, P_2 for $t \in]0, 1[$.

Property L4.4 (*Affine Invariance*)

Taking into consideration the ratios

$$r(P_0, P_0^1, P_1) = r(P_1, P_1^1, P_2) = r(P_0', P_0^2, P_1^1) = \frac{t}{1-t}$$

we conclude that the above construction of a parabola is invariant under affine transformations because the piecewise linear interpolation is affine invariant.

4.5 de Casteljau Algorithm

The previous construction for parabolas in $\mathbb{R}^2$ can be generalized to degree-N polynomial curves even in $\mathbb{R}^3$, using the well-known algorithm due to de Casteljau.

Let us then outline the de Casteljau algorithm for curves in $\mathbb{R}^3$. Given the points $P_0, P_1, \ldots, P_N \in \mathbb{R}^3$, the degree-$d$ curve point at t is calculated as follows:

$$P_i^d(t) = (1-t)P_i^{d-1}(t) + tP_{i+1}^{d-1}(t) \tag{11}$$

where $d = 1, \ldots, N$ and $i = 0, \ldots, N-d$, with $P_i^0(t) = P_i$. As we will see in a later lecture, this means that $P_0^N(t) = P_i$ is a point of a degree-N Bézier curve. The vertices $P_0, \ldots, P_N$ are known as *control points* (or *Bézier points*), and the corresponding polygon is called *control polygon* (or *Bézier polygon*).

The de Casteljau scheme behind the computation of the coefficients $P_i^d(t)$ can be organized into a triangular matrix of points. For example, a cubic curve ($N = 3$) depicted in Fig. **??** has the following de Casteljau scheme:

$$\begin{array}{llll}
P_0 & & & \\
P_1 & P_0^1 & & \\
P_2 & P_1^1 & P_0^2 & \\
P_3 & P_2^1 & P_1^2 & P_0^3
\end{array}$$

This suggests the use of a 2-dimensional array in the implementation of the algorithm, but this would be a waste of memory space for higher-dimensional curves. Therefore, instead of using a 2-dimensional array to store a triangular matrix, we use a column vector to accommodate the non-null elements of such a matrix.

So, given the column array P[ ] containing a number $N+1$ of control points P_i, the C program to compute the degree-N curve point at t makes use of Eq. 11 as follows:

```
float deCasteljau(int N, float P[], float t)
{
        int d, i;

        for (d=1;d<=N;d++)
                for (i=0;i<=N-d;i++)
                        P[i]=(1-t)*P[i]+t*P[i+1];
        return P[0];
}
```

This program returns the curve point P_0^N given the control points $P_0, P_1, \ldots, P_N$.

4.6 Final Remarks

References

[1] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, M. and J. Crowcroft, "XORs in the air: practical wireless network coding", *IEEE/ACM Transactions on Networking,*, vol. 16, no. 3, pp. 497–510, 2008.

[2] H. Rahul, N. Kushman, D. Katabi, C. Sodini, and F. Edalat, "Learning to Share: Narrowband-Friendly Wideband Wireless Networks", *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 147–158, 2008.