VIII.4
USING GEOMETRIC CONSTRUCTIONS TO INTERPOLATE ORIENTATION WITH QUATERNIONS

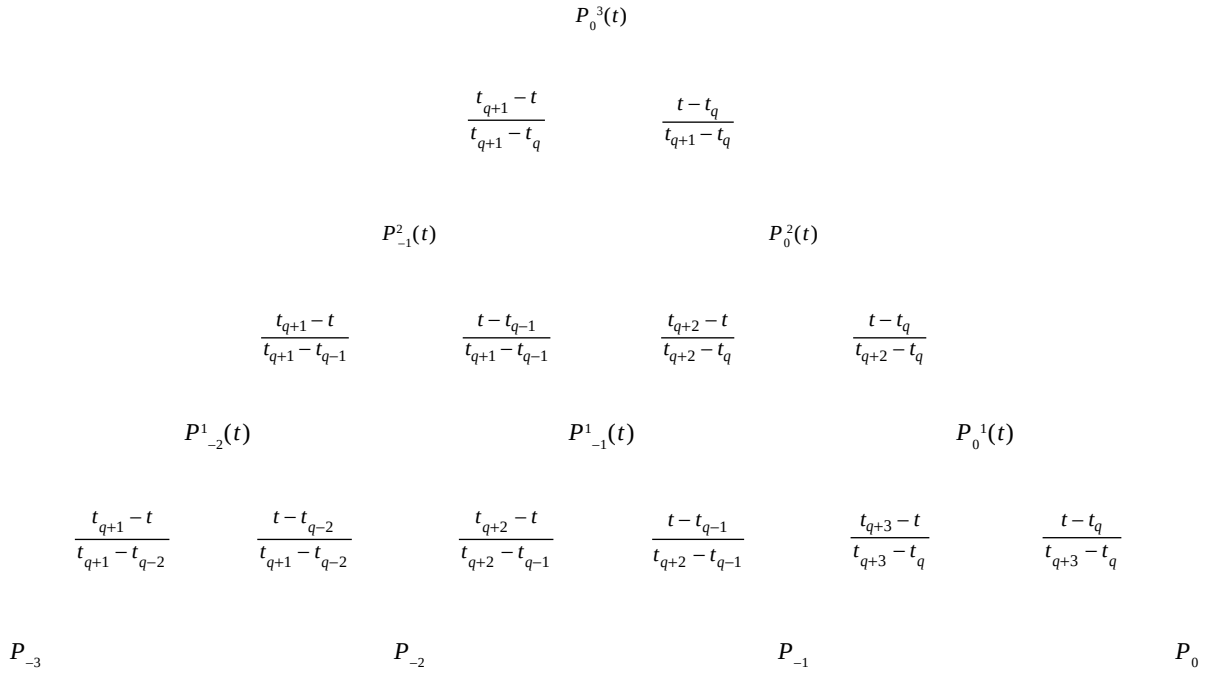
John Schlag
MacroMind, Inc.
San Francisco, California

Quaternions have been established as a useful representation for interpolating 3D orientation in computer animation. In keeping with traditional computer animation practices, we would like both interpolating and approximating splines. These can be derived easily by applying the geometric constructions known for linear splines.

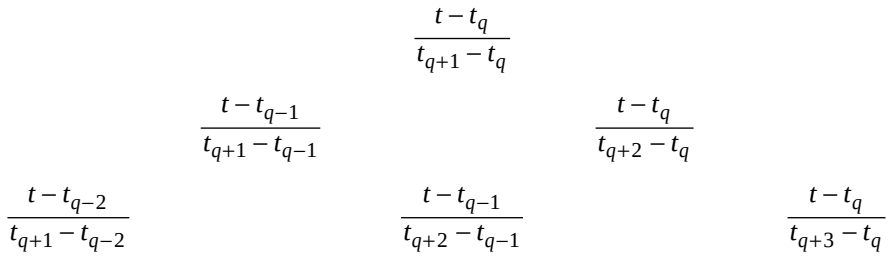
Shoemake (1985) provided a scheme for deriving Bézier control points from a sequence of quaternions. This provides an interpolating spline for quaternions, but the construction is somewhat more complicated than necessary. The most common interpolating spline in use probably is the *Catmull-Rom spline*. In 1988, Barry and Goldman (1988) obligingly provided a geometric construction for Catmull-Rom splines. This produces an interpolating spline directly from the control points, without the construction of auxiliary Bézier points. For an approximating spline, a geometric construction for *B*-splines is well-known (de Boor, 1972).

Barry and Goldman represent geometric constructions as a triangle with the four control points at the bottom, the result point at the top, and weighting functions for the intermediate results on each arc; for example,

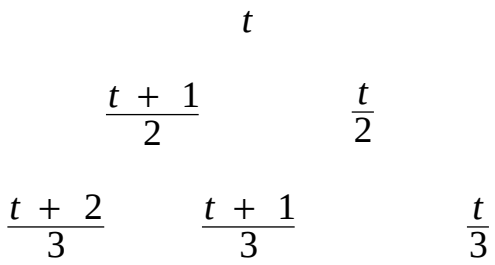
the nonuniform *B*-spline construction:



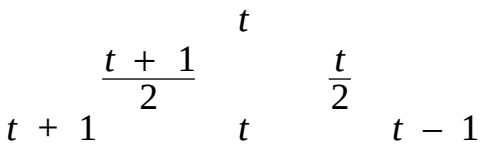
This notation can be compressed considerably, since each pair of points is blended with coefficients that sum to one (i.e., the points are linearly interpolated, or *lerped).* By collapsing such coefficients and deleting the points, we get the six-entry triangle:



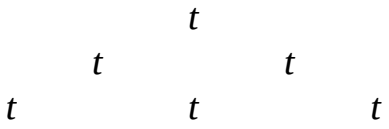
In the uniform case (with knot values $t_i = i$), this reduces to:



Similarly, the Catmull-Rom construction is



and the Bezier construction is simply



For our purposes, the most useful consequence of expressing the constructions as lerping pairs of points together is that they can be readily applied to any domain where an analog to *lerp* exists. For quaternions, the analog to lerp is *slerp*, for spherical linear interpolation (Shoemake, 1985):

$$\text{slerp}(q_1,q_2,u) = \frac{\sin(1-u)\theta}{\sin\theta} q_1 + \frac{\sin u\theta}{\sin\theta} q_2, \text{ where } \cos\theta = q_1 \cdot q_2.$$

For example:

```
function qCatmullRom(
        q00, q01, q02, q03: quaternion; t: real
): quaternion;
        q10, q11, q12, q20, q21: quaternion;
begin
        q10 ← slerp(q00, q01, t + 1);
        q11 ← slerp(q01, q02, t);
        q12 ← slerp(q02, q03, t − 1);
        q20 ← slerp(q10, q11, (t + 1)/2);
        q21 ← slerp(q11, q12, t/2);
        return [slerp(q20, q21, t)];
endproc qCatmullRom
```

The implementation of the other constructions differs only in the third arguments to slerp().

See also VII.6 Quaternions and 4 × 4 Matrices, Ken Shoemake; *(498) Using Quaternions for Coding 3D Transformations, Patrick-Gilles Maillot*