Slerp

From Wikipedia, the free encyclopedia

In computer graphics, **Slerp** is shorthand for **s**pherical **l**inear int**erp**olation, introduced by Ken Shoemake in the context of quaternion interpolation for the purpose of animating 3D rotation. It refers to constant-speed motion along a unit-radius great circle arc, given the ends and an interpolation parameter between 0 and 1.

Geometric Slerp [edit]

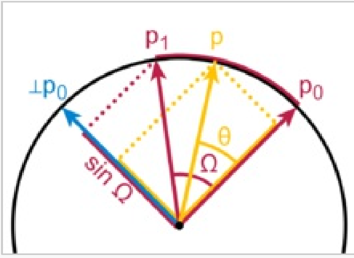
Slerp has a geometric formula independent of quaternions, and independent of the dimension of the space in which the arc is embedded. This formula, a symmetric weighted sum credited to Glenn Davis, is based on the fact that any point on the curve must be a linear combination of the ends. Let p_0 and p_1 be the first and last points of the arc, and let t be the parameter, $0 \le t \le 1$. Compute Ω as the angle subtended by the arc, so that $\cos\Omega = p_0 \cdot p_1$, the n-dimensional dot product of the unit vectors from the origin to the ends. The geometric formula is then

$$\mathrm{Slerp}(p_0, p_1; t) = \frac{\sin[(1-t)\Omega]}{\sin\Omega} p_0 + \frac{\sin[t\Omega]}{\sin\Omega} p_1.$$

The symmetry can be seen in the fact that $\mathrm{Slerp}(p_0, p_1; t) = \mathrm{Slerp}(p_1, p_0; 1 - t)$. In the limit as $\Omega \to 0$, this formula reduces to the corresponding symmetric formula for linear interpolation,

$$\mathrm{Slerp}(p_0, p_1; t) = (1-t)p_0 + tp_1.$$

A Slerp path is, in fact, the spherical geometry equivalent of a path along a line segment in the plane; a great circle is a spherical geodesic.



Oblique vector rectifies to Slerp factor.

More familiar than the general Slerp formula is the case when the end vectors are perpendicular, in which case the formula is $p_0 \cos\theta + p_1 \sin\theta$. Letting $\theta = t\pi/2$, and applying the trigonometric identity $\cos\theta = \sin(\pi/2 - \theta)$, this becomes the Slerp formula. The factor of $1/\sin\Omega$ in the general formula is a normalization, since a vector p_1 at an angle of Ω to p_0 projects onto the perpendicular $\perp p_0$ with a length of only $\sin\Omega$.

Some special cases of Slerp admit more efficient calculation. When a circular arc is to be drawn into a raster image, the preferred method is some variation of Bresenham's circle algorithm. Evaluation at the special parameter values 0 and 1 trivially yields p_0 and p_1, respectively; and bisection, evaluation at ½, simplifies to $(p_0 + p_1)/2$, normalized. Another special case, common in animation, is evaluation with fixed ends and equal parametric steps. If p_{k-1} and p_k are two consecutive values, and if c is twice their dot product (constant for all steps), then the next value, p_{k+1}, is the reflection $p_{k+1} = c\,p_k - p_{k-1}$.

Quaternion Slerp [edit]

When Slerp is applied to unit quaternions, the quaternion path maps to a path through 3D rotations in a standard way. The effect is a rotation with uniform angular velocity around a fixed rotation axis. When the initial end point is the identity quaternion, Slerp gives a segment of a one-parameter subgroup of both the Lie group of 3D rotations, SO(3), and its universal covering group of unit quaternions, S^3. Slerp gives a straightest and shortest path between its quaternion end points, and maps to a rotation through an angle of 2Ω. However, because the covering is double (q and $-q$ map to the same rotation), the rotation path may turn either the "short way" (less than 180°) or the "long way" (more than 180°). Long paths can be prevented by negating one end if the dot product, $\cos\Omega$, is negative, thus ensuring that $-90° \le \Omega \le 90°$.

Slerp also has expressions in terms of quaternion algebra, all using exponentiation. Real powers of a quaternion are defined in terms of the quaternion exponential function, written as e^q and given by the power series equally familiar from calculus, complex analysis and matrix algebra:

$$e^q = 1 + q + \frac{q^2}{2} + \frac{q^3}{6} + \cdots + \frac{q^n}{n!} + \cdots.$$

Writing a unit quaternion q in versor form, $\cos\Omega + \mathbf{v} \sin\Omega$, with $\mathbf{v}$ a unit 3-vector, and noting that the quaternion square $\mathbf{v}^2$ equals −1 (implying a quaternion version of Euler's formula), we have $e^{\mathbf{v}\Omega} = q$, and $q^t = \cos t\Omega + \mathbf{v} \sin t\Omega$. The identification of interest is $q = q_1 q_0^{-1}$, so that the real part of q is $\cos\Omega$, the same as the geometric dot product used above. Here are four equivalent quaternion expressions for Slerp.

$$\begin{aligned} \mathrm{Slerp}(q_0, q_1, t) &= q_0(q_0^{-1}q_1)^t \\ &= q_1(q_1^{-1}q_0)^{1-t} \\ &= (q_0q_1^{-1})^{1-t}q_1 \\ &= (q_1q_0^{-1})^t q_0 \end{aligned}$$

The derivative of $\mathrm{Slerp}(q_0, q_1; t)$ with respect to t, assuming the ends are fixed, is $\log(q_1q_0^{-1})$ times the function value, where the quaternion natural logarithm in this case yields half the 3D angular velocity vector. The initial tangent vector is parallel transported to each tangent along the curve; thus the curve is, indeed, a geodesic.

In the tangent space at any point on a quaternion Slerp curve, the inverse of the exponential map transforms the curve into a line segment. Slerp curves not extending through a point fail to transform into lines in that point's tangent space.

Quaternion Slerps are commonly used to construct smooth animation curves by mimicking affine constructions like the de Casteljau algorithm for Bézier curves. Since the sphere is not an affine space, familiar properties of affine constructions may fail, though the constructed curves may otherwise be entirely satisfactory. For example, the de Casteljau algorithm may be used to split a curve in affine space; this does not work on a sphere.

The two-valued Slerp can be extended to interpolate among many unit quaternions, but the extension loses the fixed execution-time of the Slerp algorithm.

External links [edit]

- Ken Shoemake. Animating rotation with quaternion curves.
- Erik B. Dam, Martin Koch, Martin Lillholm. Quaternions, Interpolation and Animation
- Understanding Slerp, Then Not Using It
- Brian Martin on quaternion animation

Categories: Computer graphics algorithms | Quaternions | Interpolation | Rotation in three dimensions | Spherical geometry