Slerp—Spherical Linear Interpolation

R. Saucier

August 2015 (Revised June 2016)

The rotation that takes the unit vector $\hat{\mathbf{u}}_1$ to the unit vector $\hat{\mathbf{u}}_2$ is the unit quaternion

$$q \equiv \cos\frac{\theta}{2} + \hat{\mathbf{n}}\sin\frac{\theta}{2}, \tag{1}$$

where θ is the angle between $\hat{\mathbf{u}}_1$ and $\hat{\mathbf{u}}_2$, and

$$\hat{\mathbf{n}} \equiv \frac{\hat{\mathbf{u}}_1 \times \hat{\mathbf{u}}_2}{||\hat{\mathbf{u}}_1 \times \hat{\mathbf{u}}_2||} \tag{2}$$

is the unit vector along the axis of rotation. This means that

$$\hat{\mathbf{u}}_2 = q\hat{\mathbf{u}}_1 q^{-1}. \tag{3}$$

Now let us parametrize the angle as $t\theta$, where $0 \le t \le 1$, and let

$$q(t) \equiv \cos\frac{t\theta}{2} + \hat{\mathbf{n}}\sin\frac{t\theta}{2}. \tag{4}$$

Then an intermediate unit vector $\hat{\mathbf{u}}(t)$ that runs along the arc on the unit circle from $\hat{\mathbf{u}}_1$ to $\hat{\mathbf{u}}_2$ is given by

$$\hat{\mathbf{u}}(t) = q(t)\hat{\mathbf{u}}_1 q(t)^{-1} = \left(\cos\frac{t\theta}{2} + \hat{\mathbf{n}}\sin\frac{t\theta}{2}\right)\hat{\mathbf{u}}_1\left(\cos\frac{t\theta}{2} - \hat{\mathbf{n}}\sin\frac{t\theta}{2}\right). \tag{5}$$

Treating $\hat{\mathbf{u}}_1$ as the pure quaternion $(0, \hat{\mathbf{u}}_1)$ and using the fact that $\hat{\mathbf{u}}_1 \cdot \hat{\mathbf{n}} = \hat{\mathbf{u}}_2 \cdot \hat{\mathbf{n}} = 0$, $\hat{\mathbf{n}} \cdot (\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1) = 0$, and $||\hat{\mathbf{u}}_1 \times \hat{\mathbf{u}}_2|| = \sin\theta$, we can carry out the quaternion multiplication to get

$$\begin{aligned}
\hat{\mathbf{u}}(t) &= \left(\cos\frac{t\theta}{2} + \hat{\mathbf{n}}\sin\frac{t\theta}{2}\right)\hat{\mathbf{u}}_1\left(\cos\frac{t\theta}{2} - \hat{\mathbf{n}}\sin\frac{t\theta}{2}\right) \\
&= \left(\cos\frac{t\theta}{2}\hat{\mathbf{u}}_1 + \hat{\mathbf{n}} \times \hat{\mathbf{u}}_1 \sin\frac{t\theta}{2}\right)\left(\cos\frac{t\theta}{2} - \hat{\mathbf{n}}\sin\frac{t\theta}{2}\right) \\
&= \cos^2\frac{t\theta}{2}\hat{\mathbf{u}}_1 + \cos\frac{t\theta}{2}\sin\frac{t\theta}{2}\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1 + \cos\frac{t\theta}{2}\sin\frac{t\theta}{2}\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1 - \sin^2\frac{t\theta}{2}(\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1) \times \hat{\mathbf{n}} \\
&= \cos^2\frac{t\theta}{2}\hat{\mathbf{u}}_1 + 2\cos\frac{t\theta}{2}\sin\frac{t\theta}{2}\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1 + \sin^2\frac{t\theta}{2}\hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1).
\end{aligned} \tag{6}$$

Now

$$\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1 = -\hat{\mathbf{u}}_1 \times \hat{\mathbf{n}} = -\frac{\hat{\mathbf{u}}_1 \times (\hat{\mathbf{u}}_1 \times \hat{\mathbf{u}}_2)}{\sin\theta} = -\frac{\hat{\mathbf{u}}_1(\hat{\mathbf{u}}_1 \cdot \hat{\mathbf{u}}_2) - \hat{\mathbf{u}}_2(\hat{\mathbf{u}}_1 \cdot \hat{\mathbf{u}}_1)}{\sin\theta} = \frac{\hat{\mathbf{u}}_2 - \cos\theta\,\hat{\mathbf{u}}_1}{\sin\theta} \tag{7}$$

and

$$\hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \hat{\mathbf{u}}_1) = \hat{\mathbf{n}}(\hat{\mathbf{n}} \cdot \hat{\mathbf{u}}_1) - \hat{\mathbf{u}}_1(\hat{\mathbf{n}} \cdot \hat{\mathbf{n}}) = -\hat{\mathbf{u}}_1 \tag{8}$$

so that

$$\begin{aligned}
\hat{\mathbf{u}}(t) &= \left(\cos^2\frac{t\theta}{2} - \sin^2\frac{t\theta}{2}\right)\hat{\mathbf{u}}_1 + 2\cos\frac{t\theta}{2}\sin\frac{t\theta}{2}\left(\frac{\hat{\mathbf{u}}_2 - \cos\theta\,\hat{\mathbf{u}}_1}{\sin\theta}\right) \\
&= \cos t\theta\,\hat{\mathbf{u}}_1 + \sin t\theta\left(\frac{\hat{\mathbf{u}}_2 - \cos\theta\,\hat{\mathbf{u}}_1}{\sin\theta}\right) \\
&= \frac{\cos t\theta\sin\theta\,\hat{\mathbf{u}}_1 + \sin t\theta\,\hat{\mathbf{u}}_2 - \sin t\theta\cos\theta\,\hat{\mathbf{u}}_1}{\sin\theta} \\
&= \frac{\sin(\theta - t\theta)\,\hat{\mathbf{u}}_1 + \sin t\theta\,\hat{\mathbf{u}}_2}{\sin\theta} \\
&= \frac{\sin(1-t)\theta}{\sin\theta}\hat{\mathbf{u}}_1 + \frac{\sin t\theta}{\sin\theta}\hat{\mathbf{u}}_2.
\end{aligned} \tag{9}$$

Slerp Formula

Thus, the spherical linear interpolation of the unit vector on the arc of the unit sphere from $\hat{\mathbf{u}}_1$ to $\hat{\mathbf{u}}_2$ is given by

$$\boxed{\hat{\mathbf{u}}(t) = \frac{\sin(1-t)\theta}{\sin\theta}\hat{\mathbf{u}}_1 + \frac{\sin t\theta}{\sin\theta}\hat{\mathbf{u}}_2}, \tag{10}$$

where $0 \le t \le 1$.

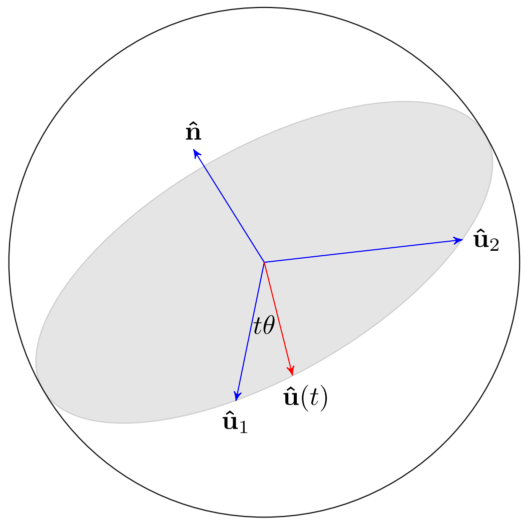

Figure 1. Spherical linear interpolation over the unit sphere.

This gives us the C++ implementation in Listing 1.

Listing 1. slerp.cpp

```
// slerp.cpp: original slerp formula

#include "Vector.h"
#include <iostream>
#include <cstdlib>

int main( int argc, char* argv[] ) {

   va::Vector i( 1., 0., 0. ), j( 0., 1., 0. ), k( 0., 0., 1. );

   va::Vector u1 = i;
   va::Vector u2 = j;
   if ( argc > 1 ) { // specify initial and final vectors on the command line

      u1 = va::Vector( atof( argv[1] ), atof( argv[2] ), atof( argv[3] ) );
      u2 = va::Vector( atof( argv[4] ), atof( argv[5] ), atof( argv[6] ) );
   }

   const int N = 1000;
   const double TH = acos( u1 * u2 );
   const double A = 1. / sin( TH );
   const double DELTA = TH / double( N-1 );
   va::Vector u = u1;

   double th = 0.;
   for ( int n = 0; n < N; n++ ) {

      u = A * ( sin( TH - th ) * u1 + sin( th ) * u2 ) ;
      th += DELTA;
      std::cout << u << std::endl;
   }

   return EXIT_SUCCESS;
}
```

Fast Incremental Slerp

Starting with Eq. 10, using the double angle formula, and

$$\hat{\mathbf{u}}_1 \cdot \hat{\mathbf{u}}_2 = \cos\theta, \tag{11}$$

we have

$$
\begin{aligned}
\hat{\mathbf{u}}(t) &= \frac{[\sin\theta\cos(t\theta) - \cos\theta\sin(t\theta)]\,\hat{\mathbf{u}}_1 + \sin t\theta\,\hat{\mathbf{u}}_2}{\sin\theta} \\
&= \cos(t\theta)\,\hat{\mathbf{u}}_1 + \frac{\sin(t\theta)}{\sin\theta}[\hat{\mathbf{u}}_2 - \cos\theta\hat{\mathbf{u}}_1] \\
&= \cos(t\theta)\,\hat{\mathbf{u}}_1 + \sin(t\theta)\left[\frac{\hat{\mathbf{u}}_2 - \cos\theta\hat{\mathbf{u}}_1}{\sqrt{1-\cos^2\theta}}\right] \\
&= \cos(t\theta)\,\hat{\mathbf{u}}_1 + \sin(t\theta)\left[\frac{\hat{\mathbf{u}}_2 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)\hat{\mathbf{u}}_1}{\sqrt{1-(\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)^2}}\right].
\end{aligned} \tag{12}
$$

Now consider the term in square brackets. The numerator is $\hat{\mathbf{u}}_2$ minus the projection of $\hat{\mathbf{u}}_2$ onto $\hat{\mathbf{u}}_1$, and thus is orthogonal to $\hat{\mathbf{u}}_1$. Also, the denominator is the norm of the numerator, since

$$
[\hat{\mathbf{u}}_2 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)\hat{\mathbf{u}}_1]\cdot[\hat{\mathbf{u}}_2 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)\hat{\mathbf{u}}_1] = 1 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)^2 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)^2 + (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)^2 = 1 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)^2. \tag{13}
$$

Thus, the term in square brackets is a unit vector that is tangent to $\hat{\mathbf{u}}_1$, which we label $\hat{\mathbf{u}}_0$:

$$
\boxed{\hat{\mathbf{u}}_0 \equiv \frac{\hat{\mathbf{u}}_2 - (\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)\hat{\mathbf{u}}_1}{\sqrt{1-(\hat{\mathbf{u}}_1\cdot\hat{\mathbf{u}}_2)^2}}}. \tag{14}
$$

Therefore, Eq. 12 can be written as

$$
\hat{\mathbf{u}}(t) = \cos(t\theta)\,\hat{\mathbf{u}}_1 + \sin(t\theta)\,\hat{\mathbf{u}}_0. \tag{15}
$$

We want to evaluate $\hat{\mathbf{u}}$ incrementally, so let us discretize this equation by setting $\delta\theta = \theta/(N-1)$ and let $x = \delta\theta$. Then Eq. 15 becomes

$$
\boxed{\hat{\mathbf{u}}[n] = \cos(nx)\,\hat{\mathbf{u}}_1 + \sin(nx)\,\hat{\mathbf{u}}_0} \quad \text{for} \quad n = 0, 1, 2, \ldots, N-1. \tag{16}
$$

Now we make use of the trigonometric identities

$$
\cos(n+1)x + \cos(n-1)x = 2\cos nx\cos x \tag{17}
$$

and

$$
\sin(n+1)x + \sin(n-1)x = 2\sin nx\cos x. \tag{18}
$$

Or, changing $n \to n-1$ and rearranging,

$$
\cos nx = 2\cos x\cos(n-1)x - \cos(n-2)x \tag{19}
$$

and

$$
\sin nx = 2\cos x\sin(n-1)x - \sin(n-2)x. \tag{20}
$$

Substituting these into Eq. 16 results in a simple recurrence relation:

$$
\begin{aligned}
\hat{\mathbf{u}}[n] &= [2\cos x\cos(n-1)x - \cos(n-2)x]\,\hat{\mathbf{u}}_1 + [2\cos x\sin(n-1)x - \sin(n-2)x]\,\hat{\mathbf{u}}_0 \\
&= 2\cos x[\cos(n-1)x\,\hat{\mathbf{u}}_1 + \sin(n-1)x\,\hat{\mathbf{u}}_0] - [\cos(n-2)x\,\hat{\mathbf{u}}_1 + \sin(n-2)x\,\hat{\mathbf{u}}_0] \\
&= 2\cos x\,\hat{\mathbf{u}}[n-1] - \hat{\mathbf{u}}[n-2].
\end{aligned} \tag{21}
$$

It is also easy to evaluate the first two values directly from Eq. 16:

$$
\hat{\mathbf{u}}[0] = \hat{\mathbf{u}}_1 \quad \text{and} \tag{22}
$$

$$
\hat{\mathbf{u}}[1] = \cos x\,\hat{\mathbf{u}}_1 + \sin x\,\hat{\mathbf{u}}_0. \tag{23}
$$

Putting this all together gives us the C++ implementation in Listing 2.

Listing 2. slerp2.cpp

```
// slerp2.cpp: using the recurrence formula to replace the trig functions in the loop

#include "Vector.h"
#include <iostream>
#include <cstdlib>

int main( int argc, char* argv[] ) {

   va::Vector i( 1., 0., 0. ), j( 0., 1., 0. ), k( 0., 0., 1. );

   va::Vector u1 = i;
   va::Vector u2 = j;
   if ( argc > 1 ) { // specify initial and final vectors on the command line

      u1 = va::Vector( atof( argv[1] ), atof( argv[2] ), atof( argv[3] ) );
      u2 = va::Vector( atof( argv[4] ), atof( argv[5] ), atof( argv[6] ) );
   }

   double u12 = u1 * u2;
   va::Vector u0 = ( u2 - u12 * u1 ) / sqrt( ( 1. - u12 ) * ( 1. +  u12 ) );

   const int N = 1000;
   const double TH = acos( u12 );
   double th = TH / double( N-1 );
   va::Vector u_2, u_1, u;

   u_2 = u1;                                 // n = 0 will become u at n - 2;
   u_1 = cos( th ) * u1 + sin( th ) * u0;   // n = 1 will become u at n - 1

   std::cout << u_2 << std::endl;
   std::cout << u_1 << std::endl;

   const double C = 2. * cos( th );

   for ( int n = 2; n < N; n++ ) {

      u   = C * u_1 - u_2;
      u_2 = u_1;
      u_1 = u;
      std::cout << u << std::endl;
   }

   return EXIT_SUCCESS;
}
```

Removing the trigonometric functions from the inner loop results in a speedup of about 12 times over the original slerp formula in Eqs. 10 or 16.

References

1. Shoemake K. Animating rotation with quaternion curves. ACM SIGGRAPH. 1985;245–254.

2. Barrera T, Hast A, Bengtsson E. Incremental spherical linear interpolation. SIGRAD 2004. 2005;7–10.

3. Hast A, Barrera T, Bengtsson E. Shading by spherical linear interpolation using De Moivre's formula. WSCG'03. 2003;Short Paper;57–60.

4. Eberly D. A fast and accurate algorithm for computing SLERP. Journal of Graphics, GPU, and Game Tools. 2011;15:3;161–176.

5. Li X. iSlerp: An incremental approach to slerp. Journal of Graphics Tools. 2007;12:3;1–6.