Vol. 4, No. 4: 1–4

# Efficiently Building a Matrix to Rotate One Vector to Another

Tomas Möller
Chalmers University of Technology

John F. Hughes
Brown University

**Abstract.** We describe an efficient (no square roots or trigonometric functions) method to construct the $3 \times 3$ matrix that rotates a unit vector $\mathbf{f}$ into another unit vector $\mathbf{t}$, rotating about the axis $\mathbf{f} \times \mathbf{t}$. We give experimental results showing this method is faster than previously known methods. An implementation in C is provided.

## 1. Introduction

Often in graphics, we have a unit vector, $\mathbf{f}$, that we wish to rotate to another unit vector, $\mathbf{t}$, by rotation in a plane containing both; in other words, we seek a rotation matrix $\mathbf{R}(\mathbf{f}, \mathbf{t})$ such that $\mathbf{R}(\mathbf{f}, \mathbf{t})\mathbf{f} = \mathbf{t}$. This paper describes a method to compute the matrix $\mathbf{R}(\mathbf{f}, \mathbf{t})$ from the coordinates of $\mathbf{f}$ and $\mathbf{t}$, without square root or trigonometric functions. Fast and robust C code can be found on the accompanying Web site.

## 2. Derivation

Rotation from $\mathbf{f}$ to $\mathbf{t}$ could be generated by letting $\mathbf{u} = \mathbf{f} \times \mathbf{t}/||\mathbf{f} \times \mathbf{t}||$, and then rotating about the unit vector $\mathbf{u}$ by $\theta = \arccos(\mathbf{f} \cdot \mathbf{t})$. A formula for the matrix that rotates about $\mathbf{u}$ by $\theta$ is given in Foley et al. [Foley et al. 90],



1086-7651/99 $0.50 per page

namely

$$\begin{pmatrix} u_x^2 + (1-u_x^2)\cos\theta & u_x u_y(1-\cos\theta) - y_z \sin\theta & u_x u_z + u_y \sin\theta \\ u_x u_y(1-\cos\theta) + u_z \sin\theta & u_y^2 + (1-u_y^2)\cos\theta & u_y u_z(1-\cos\theta) - u_x \sin\theta \\ u_x u_z(1-\cos\theta) - u_y \sin\theta & u_y u_z(1-\cos\theta) + u_x \sin\theta & u_z^2 + (1-u_x^2)\cos\theta \end{pmatrix}$$

The above involves $\cos(\theta)$, which is just $\mathbf{f} \cdot \mathbf{t}$, and $\sin(\theta)$, which is $||\mathbf{f} \times \mathbf{t}||$.

If we instead let

$$\begin{aligned} \mathbf{v} &= \mathbf{f} \times \mathbf{t} \\ c &= \mathbf{f} \cdot \mathbf{t} \\ h &= \frac{1-c}{1-c^2} = \frac{1-c}{\mathbf{v} \cdot \mathbf{v}} \end{aligned}$$

then, after considerable algebra, one can simplify the matrix to

$$\mathbf{R}(\mathbf{f}, \mathbf{t}) = \begin{pmatrix} c + hv_x^2 & hv_x v_y - v_z & hv_x v_z + v_y \\ hv_x v_y + v_z & c + hv_y^2 & hv_y v_z - v_x \\ hv_x v_z - v_y & hv_y v_z + v_x & c + hv_z^2 \end{pmatrix} \tag{1}$$

Note that this formula for $\mathbf{R}(\mathbf{f}, \mathbf{t})$ has no square roots or trigonometric functions.

When $\mathbf{f}$ and $\mathbf{t}$ are nearly parallel (i.e., $|\mathbf{f} \cdot \mathbf{t}| > 0.99$), the computation of the plane that they define (and the normal to that plane, which will be the axis of rotation) is numerically unstable; this is reflected in our formula by the denominator of $h$ becoming close to zero.

In this case, we observe that a product of two reflections (angle-preserving transformations of determinant $-1$) is always a rotation, and that reflection matrices are easy to construct: For any vector $\mathbf{u}$, the Householder matrix [Golub, Van Loan 96]

$$\mathbf{H}(\mathbf{u}) = \mathbf{I} - \frac{2}{\mathbf{u} \cdot \mathbf{u}} \mathbf{u}\mathbf{u}^t$$

reflects the vector $\mathbf{u}$ to $-\mathbf{u}$, and leaves fixed all vectors orthogonal to $\mathbf{u}$. In particular, if $\mathbf{p}$ and $\mathbf{q}$ are unit vectors, then $\mathbf{H}(\mathbf{q} - \mathbf{p})$ exchanges $\mathbf{p}$ and $\mathbf{q}$, leaving $\mathbf{p} + \mathbf{q}$ fixed.

With this in mind, we choose a unit vector $\mathbf{p}$ and build two reflection matrices: one that swaps $\mathbf{f}$ and $\mathbf{p}$, and the other that swaps $\mathbf{t}$ and $\mathbf{p}$. The product of these is a rotation that takes $\mathbf{f}$ to $\mathbf{t}$.

To choose $\mathbf{p}$, we determine which coordinate axis ($x$, $y$, or $z$) is most nearly orthogonal to $\mathbf{f}$ (the one for which the corresponding coordinate of $\mathbf{f}$ is smallest in absolute value) and let $\mathbf{p}$ be a unit vector along that axis. We then build $\mathbf{A} = \mathbf{H}(\mathbf{p} - \mathbf{f})$, and $\mathbf{B} = \mathbf{H}(\mathbf{p} - \mathbf{t})$, and the rotation we want is $\mathbf{R} = \mathbf{BA}$.

That is, if we let

$$\begin{aligned} \mathbf{p} &= \begin{cases} \hat{\mathbf{x}}, \text{ if } |f_x|<|f_y| \text{ and } |f_x|<|f_z| \\ \hat{\mathbf{y}}, \text{ if } |f_y|<|f_x| \text{ and } |f_y|<|f_z| \\ \hat{\mathbf{z}}, \text{ if } |f_z|<|f_x| \text{ and } |f_z|<|f_y| \end{cases} \\ \mathbf{u} &= \mathbf{p}-\mathbf{f} \\ \mathbf{v} &= \mathbf{p}-\mathbf{t}, \end{aligned}$$

then the entries of $\mathbf{R}$ are given by

$$r_{ij} = \delta_{ij} - \frac{2}{\mathbf{u}\cdot\mathbf{u}}u_i u_j - \frac{2}{\mathbf{v}\cdot\mathbf{v}}v_i v_j + \frac{4\mathbf{u}\cdot\mathbf{v}}{(\mathbf{u}\cdot\mathbf{u})(\mathbf{v}\cdot\mathbf{v})}v_i u_j \tag{2}$$

where $\delta_{ij} = 1$ when $i = j$ and $\delta_{ij} = 0$ when $i \neq j$.

## 3. Performance

We tested the new method for performance against all previously known (by the authors) methods for rotating a unit vector into another unit vector. A naive way to rotate $\mathbf{f}$ into $\mathbf{t}$ is to use quaternions to build the rotation directly: Letting $\mathbf{u} = \mathbf{v}/||\mathbf{v}||$, where $\mathbf{v} = \mathbf{f} \times \mathbf{t}$, and letting $\phi = (1/2)\arccos(\mathbf{f}\cdot\mathbf{t})$, we define $\mathbf{q} = (\sin(\phi)\mathbf{u}; \cos\phi)$ and then convert the quaternion $\mathbf{q}$ into a rotation via the method described by Shoemake [Shoemake 85]. This rotation takes $\mathbf{f}$ to $\mathbf{t}$, and we refer to this method as Naive. The second is called Cunningham and is just a change of bases [Cunningham 90]. Goldman [Goldman 90] gives a routine for rotating around an arbitrary axis: in our third method we simplified his matrix for our purposes; this method is denoted Goldman. All three of these require that some vector be normalized; the quaternion method requires normalization of $\mathbf{v}$; the Cunningham method requires that one input be normalized, and then requires normalization of the cross-product. Goldman requires the normalized axis of rotation. Thus, the requirement of unit-vector input in our algorithm is not exceptional.

For the statistics below, we used 1,000 pairs of random normalized vectors $\mathbf{f}$ and $\mathbf{t}$. Each pair was fed to the matrix routines 10,000 times to produce accurate timings. Our timings were done on a Pentium II 400 MHz with compiler optimizations for speed on.

| **Routine:** | Naive | Cunningham | Goldman | New Routine |
|---|---|---|---|---|
| **Time (s):** | 18.6 | 13.2 | 6.5 | 4.1 |

The fastest of previous known methods (Goldman) still takes about 50% more time than our new routine, and the naive implementation takes almost 350%

more time. Similar performance can be expected on most other architectures, since square roots and trigonometric functions are expensive to use.

**Acknowledgements.** Thanks to Eric Haines for encouraging us to write this.

## References

[Cunningham 90] Steve Cunningham. "3D Viewing and Rotation using Orthonormal Bases." In *Graphics Gems*,edited by Andrew S. Glassner, pp. 516–521, Boston: Academic Press, Inc., 1990.

[Foley et al. 90] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics—Principles and Practice*, Second Edition, Reading, MA: Addison-Wesley, 1990.

[Goldman 90] Ronald Goldman. "Matrices and Transformations." In *Graphics Gems*, edited by Andrew S. Glassner, pp. 472–475, Boston: Academic Press, Inc., 1990.

[Golub, Van Loan 96] Gene Golub and Charles Van Loan, *Matrix Computations*, Third Edition, Baltimore: Johns Hopkins University Press, 1996.

[Shoemake 85] Ken Shoemake. "Animating Rotation with Quaternion Curves," *Computer Graphics (Proc. SIGGRAPH '85)*, 19(3): 245–254 (July 1985).

**Web Information:**

http://www.acm.org/jgt/papers/MollerHughes99

Tomas Möller, Chalmers University of Technology, Department of Computer Engineering, 412 96 Gothenburg, Sweden (tompa@acm.org) *Currently at U.C. Berkeley.*

John F. Hughes, Brown University, Computer Science Department, 115 Waterman Street, Providence, RI 02912 (jfh@cs.brown.edu)

Received June 28, 1999; accepted in revised form January 14, 2000.