Weekly Report

John Anglo

May 17, 2012

1 More on rotations around an arbitrary axis

Two approaches to rotating a vector $\vec{v} = x, y, z$ around an axis described by the unit vector $\hat{n} = (a, b, c)$ by an angle θ are known. It turns out that the rotation matrix used in the first approach can be recovered by expressing the operations performed in the alternative, vector approach in terms of matrix multiplications. The rotated vector $\vec{v}'$ is found as

$$\vec{v}' = \vec{v}_{\parallel} + \vec{v}_{\perp}\cos\theta + \vec{u}\sin\theta \tag{1}$$

where

$$\vec{v}_{\parallel} = \hat{n}(\hat{n}\cdot\vec{v}) \tag{2a}$$
$$\vec{v}_{\perp} = \vec{v} - \vec{v}_{\parallel} \tag{2b}$$
$$\vec{u} = \hat{n}\times\vec{v} \tag{2c}$$

$\vec{v}_{\parallel}$ can be written explicitly as a vector from Equation 2a:

$$\begin{aligned}\vec{v}_{\parallel} &= \hat{n}(ax+by+cz)\\ &= \begin{bmatrix} a^2x+aby+acz\\ abx+b^2y+bcz\\ acx+bcy+c^2z\end{bmatrix}\end{aligned} \tag{3}$$

The same result is obtained by multiplying $\vec{v}$ by the matrix

$$\mathrm{D} = \begin{bmatrix} a^2 & ab & ac\\ ab & b^2 & bc\\ ac & bc & c^2\end{bmatrix} \tag{4}$$

So $\vec{v}_{\parallel} = D\vec{v}$. Consequently, $\vec{v}_{\perp}$ can be expressed as $\vec{v} - \vec{v}_{\parallel} = (\mathbb{1} - \mathrm{D})\vec{v}$, where $\mathbb{1}$ is the identity operator. $(\mathbb{1} - \mathrm{D})$ can also be expressed as a single matrix:

$$(\mathbb{1} - \mathrm{D}) = \begin{bmatrix} 1-a^2 & -ab & -ac\\ -ab & 1-b^2 & -bc\\ -ac & -bc & 1-c^2\end{bmatrix} \tag{5}$$

As with $\vec{v}_{\parallel}$, $\vec{u}$ is written explicitly as a column vector, from Equation 2c:

$$\vec{u} = \det \begin{bmatrix} \hat{i} & \hat{j} & \hat{k} \\ a & b & c \\ x & y & z \end{bmatrix} = \begin{bmatrix} -cy + bz \\ cx - az \\ -bx + ay \end{bmatrix} \tag{6}$$

Which is also given by multiplying $\vec{v}$ by the matrix

$$\mathrm{C} = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix} \tag{7}$$

Using all of these together, Equation 1 can be expressed exclusively in terms of multiplying $\vec{v}$ by matrices and scalar quantities, as $\vec{v}' = (\mathrm{D}\vec{v} + (\cos\theta)(\mathbb{1} - \mathrm{D})\vec{v} + (\sin\theta)\mathrm{C}\vec{v})$. The multiplicands beside $\vec{v}$ can all be combined into a single matrix A:

$$\begin{aligned} \mathrm{A} &= (\mathrm{D} + (\cos\theta)(\mathbb{1} - \mathrm{D}) + (\sin\theta)\mathrm{C}) \\ &= \begin{bmatrix} a^2 + (1 - a^2)\cos\theta & ab(1 - \cos\theta) - c\sin\theta & ac(1 - \cos\theta) + b\sin\theta \\ ab(1 - \cos\theta) + c\sin\theta & b^2 + (1 - b^2)\cos\theta & bc(1 - \cos\theta) - a\sin\theta \\ ac(1 - \cos\theta) - b\sin\theta & bc(1 - \cos\theta) + a\sin\theta & c^2 + (1 - c^2)\cos\theta \end{bmatrix} \end{aligned} \tag{8}$$

2 IDL work

2.1 Euler rotation matrix

I wrote a program to compute the rotation matrix given by equation 4-46 in the Goldstein book, and compared its performance to the function included with the `skymap_vector` class definition. Appropriately, having to compute only one matrix instead of three meant the new program finished in about one third of the `skymap_vector` function's time.

2.2 Rotations around an arbitrary axis

I also wrote programs to perform each of the methods mentioned in Section 1. It was expected that the single matrix technique would be optimal, as a more straightforward calculation; the alternative technique is more intuitive but appears to require more operations. Instead my implementations performed almost identically, with the vector technique instead finishing in at most 3% less time. To me it appears that it might be possible to further optimize the matrix function as I've implemented it, more so than the vector technique, but I don't expect the performance of the former to improve significantly.