NOTES

Using Quadratic Forms to Correct Orientation Errors in Tracking

JACK GOLDFEATHER
Carleton College
Northfield, MN 55057

Introduction The motivation for this note comes from a problem in tracking a moving object in virtual reality systems. For example, in so-called "see-through" virtual reality environments a moving computer-generated "virtual" image is integrated into a real environment, and the placement of the image is based on computing the position and orientation of the observer. There are many techniques for tracking three-dimensional motion, but all of them depend in one way or another on gathering data from the environment which are used as coefficients in some system of equations with the position and orientation parameters as the unknowns. Typically, orientation is expressed as a 3×3 orthogonal matrix, R, interpreted as the change of basis from the fixed world coordinate system to the moving coordinate system.

Errors occur when the coefficients are "noisy," (e.g., when the measuring equipment is capable of returning results accurate to only a few significant digits), and noisy coefficients can produce an R which is not quite orthogonal.

For example, the matrix

$$R = \begin{pmatrix} -0.97451771 & 0.02041436 & 0.03124792 \\ 0.02372552 & 0.97924131 & 0.00034581 \\ -0.03188555 & 0.00102279 & -0.95477235 \end{pmatrix}$$

was actually computed by a tracking system in the graphics lab at the University of North Carolina as the orientation of an ultrasound transducer being used in a trial experiment to create real-time 3D images of a human fetus. Unfortunately, the lengths of the column vectors of R are 0.97532782, 0.97945461, and 0.95528362, and the angles between column vectors are 89.8016998, 90.000003, and 89.999996 degrees, i.e., it is not quite orthogonal. This can create a number of problems for tracking systems. For example, R is used to transform various geometrically-defined data sets to a user's point of view and a non-orthogonal R can skew the resulting image. Even worse, some tracking systems use dynamic models to predict future orientation over short time intervals when there is insufficient time to collect or process data. Integrating over time using a non-orthogonal matrix as an initial condition can produce badly skewed results.

In this note, a method is described for computing an orthogonal matrix that is "nearest" to an "almost orthogonal" matrix. The method has been used successfully in ultrasound transducer tracking to correct skewing errors due to noise.

A Simple Tracking Example Imagine a camera (for simplicity let's assume a pinhole camera) that moves around, and is able to "see" points P in the world whose

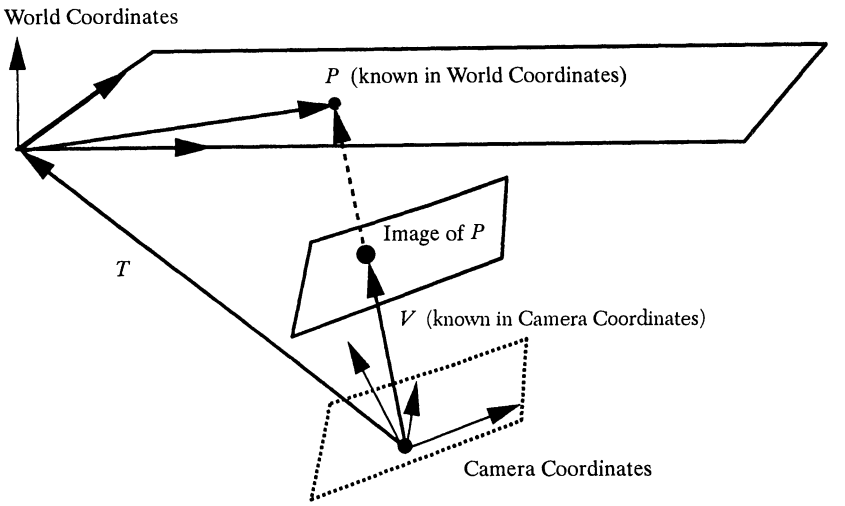

FIGURE 1

fixed world coordinates are known (see FIGURE 1). The camera has its own internal coordinate system, with two orthogonal axes in the image plane; the third orthogonal axis points in the direction the camera is aimed. The camera registers the point P as a vector V drawn from its origin to the image of P on its image plane. We would like to find (i) the camera's position, expressed as a vector T from the camera origin to the world origin; and (ii) the camera's orientation, expressed as a 3×3 orthogonal matrix R that rotates world coordinates into camera coordinates.

The vector equation $T + RP = kV$ relates P to V. It can be understood as follows:

(1) Multiplying P by the unknown rotation matrix R transforms P from the world coordinate system to the camera coordinate system.
(2) The vector sum $T + RP$ is the vector from the camera origin to P.
(3) The same vector can be found by scaling V by an unknown scalar k.

If the camera can see several fixed points, $P_1, P_2, \ldots, P_n$, and n is sufficiently large, the resulting system

$$T + RP_i = k_i V_i,\ i = 1 \ldots n$$

(which we will refer to as the tracking equations) can usually be solved for the unknowns T, R, k_i. How big n has to be is an interesting question and lies at the heart of the next section. For the moment, however, note that both P_i and V_i are not known exactly because both are measured by imprecise instruments. In particular, points look fuzzy through a camera, so their precise position is difficult to determine. It is this "noisy" kind of data that leads to problems.

We will describe two approaches to solving the tracking equations.

The Linear System Approach The simplest way to solve the tracking equations is to treat all 9 entries in

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

as independent parameters. Of course, in an orthogonal matrix the 9 parameters are not independent (e.g. $r_{11}^2 + r_{21}^2 + r_{31}^2 = 1$ and $r_{11}r_{12} + r_{21}r_{22} + r_{31}r_{32} = 0$), but it turns out that there is no need to impose these additional (non-linear) constraints right away. Instead, we impose the constraint that the points P_i all lie in a horizontal plane. In this case, each P_i is of the form $(x_i, y_i, 0)$ which eliminates the third column of R from the tracking equations. (This column can be recovered later by taking the cross product of the first two columns.) The tracking system in the graphics lab at UNC uses tiny lights embedded in a (nearly-planar) ceiling as the tracking points, so the coplanarity constraint is not just a theoretical one. Note, however, that small errors in the planarity of the ceiling lead to more noise in the data.

Once the third column of R is eliminated, and provided that n is sufficiently large, the resulting linear system can be solved (for either a unique solution if the system is consistent or a best fit solution if it is not) by a variety of efficient computer algorithms. It can be shown that $n = 4$ suffices provided that no three of the P_i are collinear, but in practice it is better to overconstrain the system by choosing $n > 4$.

So the good news is that a value for R can be found using a fast computer algorithm. The bad news is that noisy data will produce an R that is not quite orthogonal.

The Quaternion Approach Another approach to solving the tracking equations is to use quaternion parameters (w, x, y, z) and write R in the form

$$(*) \qquad R = \begin{pmatrix} w^2 + x^2 - y^2 - z^2 & 2xy - 2wz & 2xz + 2wy \\ 2xy + 2wz & w^2 + y^2 - x^2 - z^2 & 2yz - 2wx \\ 2xz - 2wy & 2yz + 2wx & w^2 + z^2 - x^2 - y^2 \end{pmatrix}$$

where $w^2 + x^2 + y^2 + z^2 = 1$. The name derives from the fact that if $Q = (w, x, y, z)$ is thought of as a unit quaternion and V is a vector in 3-space, then RV, i.e., "rotating" V by R, is the same as the vector part of the quaternion product QVQ^{-1}. (See **[1]**, **[2]** for more details.) Quaternion parameters are a good choice for keeping matrices orthogonal, because if the computed $Q_1 = (w_1, x_1, y_1, z_1)$ is not quite a unit vector (remember, coefficients in equations will be noisy), it can be replaced with $Q_2 = Q_1/|Q_1|$. It is not hard to show that among all unit quaternions, Q_2 minimizes

$$f(w, x, y, z) = (w - w_1)^2 + (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2.$$

However, quaternions may be a bad choice because this makes the tracking equations quadratic, not linear, and quadratic systems are much less amenable to a quick computer solution. Indeed, there may not be *any* tractable method for solving certain quadratic systems.

So in many cases, R must be computed directly using the r_{ij} as parameters, and then corrected to an orthogonal matrix.

A Quadratic Forms Solution to Correcting *R* Although the tracking equations may be hard to solve directly using quaternion parameters, we can use them to correct an R found by the linear system method. Suppose

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

has been computed from noisy data and is not orthogonal. We will say an orthogonal matrix

$$S = \begin{pmatrix} s_{11} & s_{12} & s_{13} \\ s_{21} & s_{22} & s_{23} \\ s_{31} & s_{32} & s_{33} \end{pmatrix}$$

is *nearest* to R if $m = \sum_{i,j}(r_{ij} - s_{ij})^2$ is minimized by S. Since the columns of S have unit length, expanding m gives

$$\begin{aligned} m(S) &= \sum_{i,j} r_{ij}^2 - 2\sum_{i,j} r_{ij}s_{ij} + \sum_{i,j} s_{ij}^2 \\ &= \sum_{i,j} r_{ij}^2 - 2\sum_{i,j} r_{ij}s_{ij} + 3. \end{aligned}$$

Hence, minimizing m is equivalent to maximizing

$$M(S) = \sum_{i,j} r_{ij}s_{ij}.$$

Using the quaternion parameters (w, x, y, z) in $(*)$ for S, multiplying everything out, and rearranging terms, we obtain

$$\begin{aligned} M(w, x, y, z) = M(S) &= \sum_{i,j} r_{ij}s_{ij} \\ &= (r_{11} + r_{22} + r_{33})w^2 + (r_{11} - r_{22} - r_{33})x^2 + (-r_{11} + r_{22} - r_{33})y^2 \\ &\quad + (-r_{11} - r_{22} + r_{33})z^2 + 2(r_{32} - r_{23})wx + 2(r_{13} - r_{31})wy \\ &\quad + 2(r_{21} - r_{12})wz + 2(r_{21} + r_{12})xy + 2(r_{31} + r_{13})xz + 2(r_{32} + r_{23})yz. \end{aligned}$$

This is a quadratic form in (w, x, y, z) and can be written as $M(X) = X^TAX$ where $X = \begin{pmatrix} w \\ x \\ y \\ z \end{pmatrix}$, X^T denotes the transpose of X, and

$$A = \begin{pmatrix} r_{11} + r_{22} + r_{33} & r_{32} - r_{23} & r_{13} - r_{31} & r_{21} - r_{12} \\ r_{32} - r_{23} & r_{11} - r_{22} - r_{33} & r_{21} + r_{12} & r_{31} + r_{13} \\ r_{13} - r_{31} & r_{21} + r_{12} & -r_{11} + r_{22} - r_{33} & r_{32} + r_{23} \\ r_{21} - r_{12} & r_{31} + r_{13} & r_{32} + r_{23} & -r_{11} - r_{22} + r_{33} \end{pmatrix}$$

Note that since X is a unit vector, we have converted the problem of finding S to that of maximizing the quadratic form X^TAX on the unit sphere in R^4. A sketch of the solution method follows (see [3] for more details). Since A is a symmetric matrix it can be written in the form

$$A = B^TDB$$

where

$$D = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 \\ 0 & 0 & 0 & \lambda_4 \end{pmatrix}$$

is the diagonal matrix of (real) eigenvalues of A and $B^T = B^{-1}$. Letting

$$BX = Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}$$

we obtain

$$M(X) = X^T AX = X^T B^T DBX = Y^T DY = \lambda_1 y_1^2 + \lambda_2 y_2^2 + \lambda_3 y_3^2 + \lambda_4 y_4^2.$$

Assuming (without loss of generality) that λ_1 is the largest positive eigenvalue, the last expression is maximized when $Y = (1,0,0,0)$ with maximum value λ_1, so the unit eigenvector $X = B^T Y$ is the solution we seek. Substituting $X = (w, x, y, z)$ into $(*)$ produces the nearest orthogonal matrix to R.

Note that if R is already orthogonal so that $r_{ij} = s_{ij}$, then

$$M = \sum_{i,j} r_{ij} s_{ij} = \sum_{i,j} r_{ij}^2 = 3$$

so that the largest eigenvalue will be 3. Hence $3 - \lambda_1$ is a measure of how far R is from being orthogonal.

If this method is applied to the matrix in the introduction, the "corrected" orthogonal matrix

$$\begin{pmatrix} -0.99921004 & 0.02256809 & 0.03271062 \\ 0.02259201 & 0.99974470 & 0.00036199 \\ -0.03269410 & 0.00110071 & -0.99946480 \end{pmatrix}$$

is obtained. The associated largest eigenvalue is $\lambda = 2.91006313$.

REFERENCES

1. Patrick Du Val, *Homographies, Quaternions, and Rotations*, Oxford Mathematical Monographs, Oxford University Press, Oxford, UK, 1964, pp. 33–40.
2. Bryant A. Julstrom, Using real quaternions to represent rotations in three dimensions, *UMAP Modules in Undergraduate Mathematics and Its Applications: Module* 652, COMAP, Inc., Lexington, MA, 1992.
3. Ben Noble and James W. Daniel, *Applied Linear Algebra*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1977, pp. 429–430.