MATH 223. Rotation about an axis. Richard Anstee

In my first summer job (in an office in 1973) having finished a linear algebra course I was asked to consider the problem of placing 100 points on the surface of a sphere (for a goniophotometer to measure light radiation from light bulbs). I determined the 100 points by an approximate process, namely simulating a repulsive force between the points and running it for a while until the closest distance between points had roughly stabilized. At that point I wanted to show the person who had asked the question, what my solution looked like.

I decided to take each of the 100 points and draw a circle of maximum radius around the point P (on the surface of the sphere) by rotating the original point by an appropriate angle to begin the circle at a point Q and then rotating Q about the axis given by OP by angles of 10°. We have the rotation matrix in 2-dimensions

$$R(\theta) = \begin{bmatrix} \cos(\theta & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

We can extend to rotation in 3-dimensions about the z-axis:

$$R = \begin{bmatrix} \cos(\theta & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This isn't what we want. We want rotation in 3-dimensions about an arbitrary axiss $\mathbf{d} = (d_1, d_2, d_3)^T$. So we employ a change of basis that replaces $(0,0,1)^T$ by $\mathbf{d}/||\mathbf{d}||$ and of course use an orthonormal basis $\mathbf{u}, \mathbf{v}, \mathbf{d}$ for $\mathbf{R}^3$. We want a circle to remain a circle and not get squished into an ellipse. We can do this easily.

The matrix we want

$$M = \left[\begin{array}{c|c|c} & & d_1 \\ \mathbf{u} & \mathbf{v} & d_2 \\ & & d_3 \end{array}\right]$$

will take the $\mathbf{u}, \mathbf{v}, \mathbf{d}$ coordinates to standard coordinates. Then M^{-1} takes standard coordinates to $\mathbf{u}, \mathbf{v}, \mathbf{d}$ coordinates. We imagine R as the rotation matrix with respect to $\mathbf{u}, \mathbf{v}, \mathbf{d}$, namely rotation around $(0,0,1)^T$ in $\mathbf{u}, \mathbf{v}, \mathbf{d}$ coordinates which is $\mathbf{d}$ when interpreted in $\mathbf{u}, \mathbf{v}, \mathbf{d}$ coordinates. Then we return from $\mathbf{u}, \mathbf{v}, \mathbf{d}$ coordinates to standard coordinates using M. Thus MRM^{-1} is our desired matrix. The rotation matrix around $\mathbf{d}$ becomes

$$MRM^{-1} = M \begin{bmatrix} \cos(\theta & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} M^{-1}$$

All my calculations had to be done on a machine that allowed me to store 400 memory locations including the code, so .4K. Space was so limited that I stored the 100 points in polar coordinates so they only needed 200 memory locations. To obtain the orthonormal basis I just found one vector perpendicular to $\mathbf{d}$, then computed a vector orthogonal to both using the cross product formula.

The actual plotting was done using a mechanical plotter that could plot line segments using a felt tipped pen. These line segments were joining two points one rotated by a further 10° using MRM^{-1}. You might well ask whether my points moved consisitent with right hand rule about $\mathbf{d}$ but that didn't matter to me. The sign of the determinant of R would have told me about the orientation. Of course I then had to finish using 3D projection. The date (1973) for all this is relevant.