We represent a point and a vector with an array of size four. The fourth entry is 1 for a point and 0 for a vector. This has the natural advantage that point minus point is vector, point plus vector is point, vector plus vector is vector, and–for the moment–point plus point is invalid.

Similarly, the matrix is represented using an array of size 16 which is thought of as a 4 by 4 matrix. The i, j entry is stored at $i + 4j$. This is assuming zero-based indexing $0 \le i, j < 4$.

What happens if we put values t_x, t_y, t_z into the rightmost column and multiply it times a point $p = (p_x, p_y, p_z)$?

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} p_x + t_x \\ p_y + t_y \\ p_z + t_z \\ 1 \end{bmatrix}.$$

Or a vector $v = (v_x, v_y, v_z)$?

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

The matrix is a *translation*! It translates points and vectors properly. Since vectors can't be translated, what you get here is the right answer.

So we can represent both rotations and translations using a 4 by 4 matrix. It turns out we can do more than that–if the bottom row isn't $0, 0, 0, 1$–but that comes later.

What about adding points? That doesn't work because the fourth coordinate becomes 2. But it does if we modify getX(), getY(), and getZ() to divide by the fourth coordinate. That way the geometric point (p_x, p_y, p_z) is represented by any *homogeneous* coordinate (q_x, q_y, q_z, q_w) such that

$$p_x = q_x/q_w, \qquad p_y = q_y/q_w, \qquad p_z = q_z/q_w.$$

So multiplying a point times a scalar leaves it the same point. Adding two points with fourth coordinate 1 yields–their midpoint! In general a linear combination of two homogeneous coordinates q_1 and q_2 is a coordinate on the line through q_1 and q_2, and a linear combination of three points lies on their plane.

So how do we scale a point? Can you come up with a matrix to do it? If it is all ss along the diagonal, it actually does nothing!

A vector has a zero fourth coordinate, so it is like a "point at infinity".

One more thing. A plane is the set of points p satisfying

$$ap_x + bp_y + cp_z + d = 0.$$

Create a homogeneous coordinate $e = [a, b, c, d]$. Then the equation of a line is $e \cdot p = 0$. Here the use of homogeneous points is natural: if you multiply a, b, c, d by a scalar, you know it is an equivalent plane equation.

We can represent points and vectors in 2D as complex numbers: $x + yi$. Rotating by θ is the same as multiplying by $\cos\theta + \sin\theta i$. Try it!

Quaternions are another way to represent rotations in 3D. A quaternion looks like this,

$$a + bi + cj + dk$$

where

$$i^2 = j^2 = k^2 = -1, \qquad ij = k, \qquad ji = -k, \qquad jk = i, \qquad kj = -i, \qquad kj = j, \qquad jk = -j.$$

You can think of a quaternion as the sum of a scalar and a vector:

$$a + (b, c, d) = a + bi + cj + dk.$$

In this way, quaternion multiplication combines scalar (scalar or vector), dot, and cross product:

$$(s + u)(t + v) = st + sv + tu - u \cdot v + u \times v.$$

Quaternions have conjugates just like complex numbers,

$$\overline{s + u} = s - u,$$

and the product with the conjugate is the square of the magnitude,

$$(s + u)(s - u) = s^2 + u \cdot u.$$

Here is the formula for rotating v by θ about a (unit axis vector) u ($|u| = 1$):

$$(\cos\theta/2 + \sin\theta/2u)v(\cos\theta/2 - \sin\theta/2u).$$

First try plugging in $v = u$. You get u back again, which is right.

Then try plugging in v with $v \cdot u = 0$

$$(\cos\theta/2 + \sin\theta/2u)v(\cos\theta/2 - \sin\theta/2u) = (\cos\theta/2 + \sin\theta/2u)(\cos\theta/2v - \sin\theta/2v \times u),$$

$$= (\cos\theta/2 + \sin\theta/2u)(\cos\theta/2v + \sin\theta/2u \times v),$$

$$= (\cos\theta/2 + \sin\theta/2u)(\cos\theta/2 + \sin\theta/2u)v.$$

But $uu = -1$, so $(\cos\theta/2 + \sin\theta/2u)$ acts like the complex number $(\cos\theta/2 + \sin\theta/2i)$, and rotating by $\theta/2$ twice is the same as rotating by θ.

$$(\cos\theta/2 + \sin\theta/2u)v(\cos\theta/2 - \sin\theta/2u) = (\cos\theta + \sin\theta u)v = \cos\theta v + \sin\theta uxv.$$

Is this what you should get when you rotate a vector v perpendicular to u?

Since it works for u and vectors perpendicular to u, it works for everything.

So what? So you can interpolate smoothly from one quaternion to another $q(t) = (1 - t)q_1 + tq_2$. An intermediate quaternion is not a rotation because it does not have unit length, but you can unitize it: $q(t)/|q(t)|$.

My father used this to program a paint spraying robot for Ford. Also notice that you don't need a square root:

$$(q/|q|)v(q/|q|) = (qvq)(q\overline{q}).$$

So you can take an integer quaternion and use to to make a rational rotation matrix. He and I wrote a journal paper about this. Cool to publish a paper with my dad!