Deriving 2D and 3D Rotations

Kenneth E. Hoff III

Jan 22, 2000

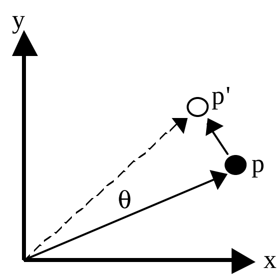
Introduction

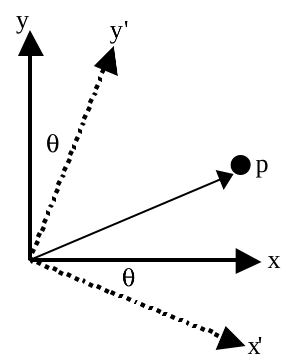
In this tech report, I describe how to derive 2D rotations about the origin and 3D rotations about the principal axes of a right-handed cartesian coordinate system. We will only require knowledge of basic principles such as the relationship between cos and sin and coordinates on the unit circle, and the dot product between two vectors. The 2D derivation will be used to construct the 3D rotations.

2D Rotations

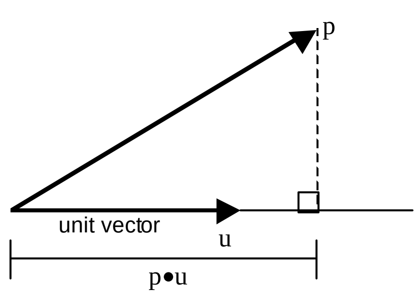
Our first goal is to construct a 2x2 matrix (pre-multiplication, column vectors) that will rotate a 2D point about the origin by some angle θ where positive rotation is counter-clockwise. Given a point p in a 2D cartesian coordinate system, our goal is to find p' in the same coordinate system rotated θ radians about the origin. Equivalently, we can simply find the point p in a new coordinate frame that has been rotated $-\theta$ with respect to the original frame containing p. The coordinates of p in this rotated frame will be the coordinates p' in the original frame.

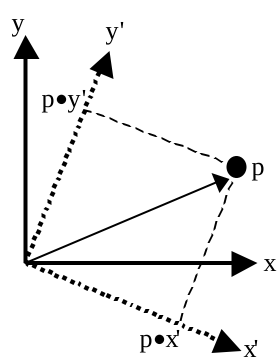
So, how do we find the coordinates of p in the rotated coordinate frame defined by the axes x' and y'? We can use



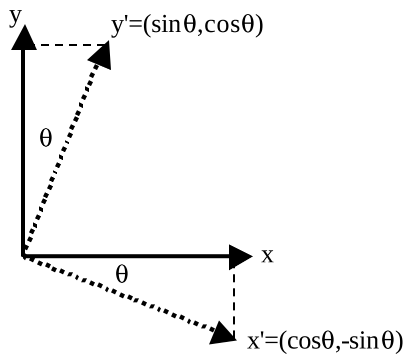


the dot product between the vector to point p and the rotated frame axes. Coordinate frame axes are normalized (unit length) and the dot product between an arbitrary vector to p and a unit length vector equals the distance between the origin and the point p projected onto the line defined by the unit length vector. So, the coordinates of p in the rotated coordinate system are $(p \bullet x', p \bullet y')$.





Now if we just show how to compute the rotated frame axes x' and y' given a rotation angle θ, we will have a complete derivation of 2D rotations. The construction of a 2x2 rotation matrix will come directly from the unit axis vectors x' and y'. The axes of the original coordinate frame are unit vectors, so we can calculate x' and y' using simple trigonometry based on the right triangles formed between between x and x', and y and y'.



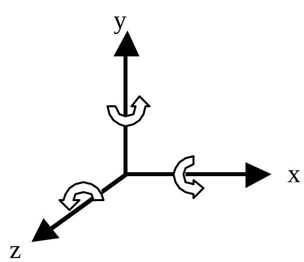
p' = (p•x',p•y') = (p•(cosθ,-sinθ),p•(sinθ,cosθ)) = ((px,py)•(cosθ,-sinθ),(px,py)•(sinθ,cosθ))

OR

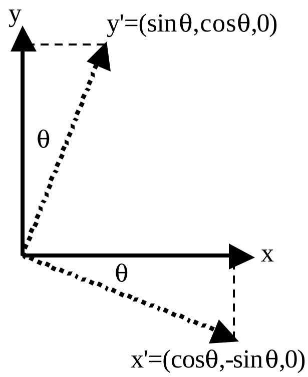
$$\begin{bmatrix} p'_x \\ p'_y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \bullet \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

3D Rotations

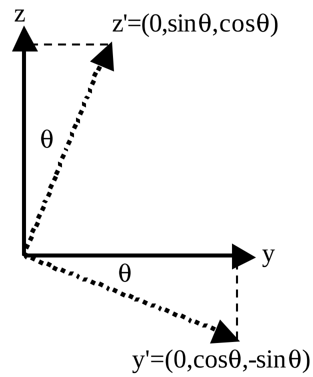
We will build a 3x3 matrix for rotation about each principal axis. We will adopt the right-hand rule for determining the direction of positive rotation: curling your right hand fingers from the x-axis to the y-axis will point your thumb down the positive z-axis, so positive rotations about the z-axis will go in the direction from the x-axis to the y-axis. Here is a summary of the axis positive rotation directions: about z (x-axis to y-axis), about x (y-axis to z-axis), and about y (z-axis to x-axis).



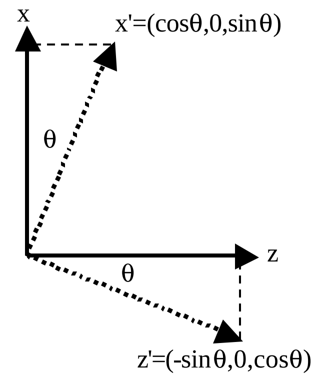
We will construct a rotated frame for each rotation in 3D. One axis will be the same as an axis of the original frame since one coordinate will remain unaffected in each rotation. For example, rotations about the z-axis will not affect the z-coordinate of the rotated point. We will construct a 3x3 matrix for each rotation.



z-axis rot: z'=(0,0,1)



x-axis rot: x'=(1,0,0)



y-axis rot: y'=(0,1,0)

As in the 2D rotations, we can find the point rotated 3D point p' from p by using the dot product: p'=(p•x',p•y',p•z'). Z and X axis rotations are very similar to the 2D rotation, but the Y-axis rotation is somewhat special since the coordinates appear to be swapped. It is important to write the axes coordinates in a consistent order (x,y,z) even though the axes appear to swap in the Y-axis rotation. We must maintain the original unrotated right-handed coordinate system. Now we can construct the rotation matrices directly from the rotated coordinate frames. We will have 3 separate 3x3 matrices corresponding to each axis rotation:

$$\text{Z-axis: } \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

$$\text{X-axis: } \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \bullet \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

$$\text{Y-axis: } \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \bullet \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

Orthonormal Matrix

The rotation matrices that we have derived show that we can simply describe a rotation with the coordinates of the rotated frame as rows in the matrix. Each row of the matrix corresponds to the vectors of the principal axes of the rotated coordinate frame. This matrix is called an orthonormal matrix since the vectors formed by the rows or columns form the axes of a rotated coordinate system. These axis vectors are normalized and orthogonal meaning they are unit length and are perpendicular to each other respectively.

In our derivation, we were trying to derive a particular type of rotation (e.g. around a principal axis) so we simply fit a rotated frame that is the inverse of the rotation we desire. We then simply create a rotation matrix by using the axes of the rotated frame as rows in the matrix. By creating this orthonormal matrix from the rotated frame and applying it to a point in world (or unrotated) space, we effectively reallign the rotated frame with the world frame and bring the point along with it. The inverse rotation is simply the transpose of this matrix.