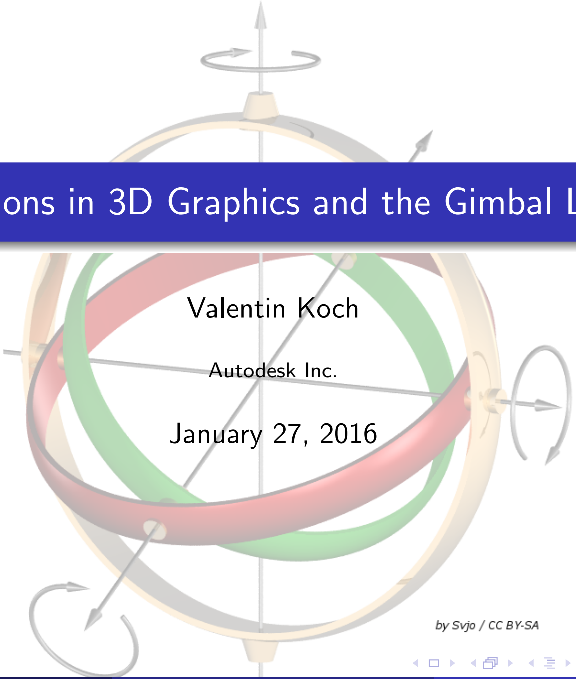
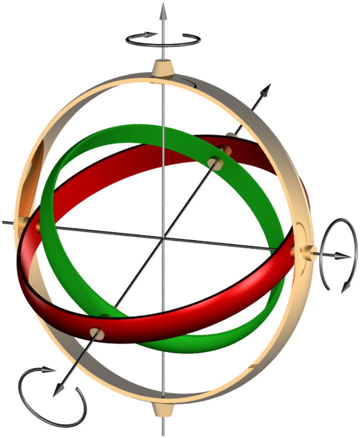
Rotations in 3D Graphics and the Gimbal Lock

Valentin Koch

Autodesk Inc.

January 27, 2016



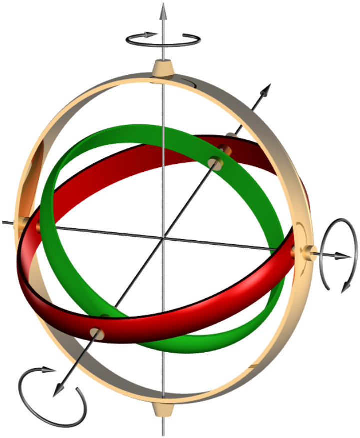


About me

- MSc. Mathematics with thesis in Mathematical Optimization
- Principal Research Engineer at Autodesk, Inc.
- Infrastructure Optimization
- 3D environment, InfraWorks (similar to Sim City).
- Encountered Gimbal Lock using numerical optimization algorithms.

Presentation Road Map

1. Introduction
2. Rotation Matrices in $\mathbb{R}^2$
3. Rotation Matrices in $\mathbb{R}^3$
4. Gimbal Lock
5. Quaternions

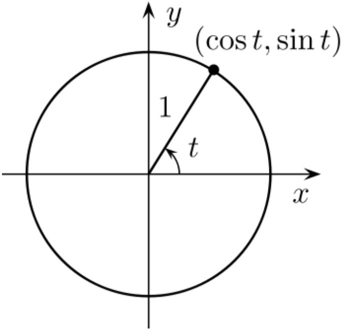


A vector (x, y) of magnitude r, is rotated by an angle t about the origin. We recall that

$$\cos t = \frac{x^*}{r},$$
$$\sin t = \frac{y^*}{r},$$

and if $r = 1$, we obtain

$$x^* = \cos t,$$
$$y^* = \sin t.$$



Given a point (x, y) at an angle α on the unit circle. We want to rotate it by angle β. Hence

$$x^* = \cos(\alpha + \beta).$$

Using the trig identities, we see that

$$\begin{aligned} x^* &= \cos(\alpha + \beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta \\ &= x\cos\beta - y\sin\beta. \end{aligned}$$

Similarly, we have

$$y^* = y\cos\beta + x\sin\beta.$$

Since

$$
\begin{aligned}
x^* &= x\cos\beta - y\sin\beta,\\
y^* &= y\cos\beta + x\sin\beta,
\end{aligned}
$$

we write

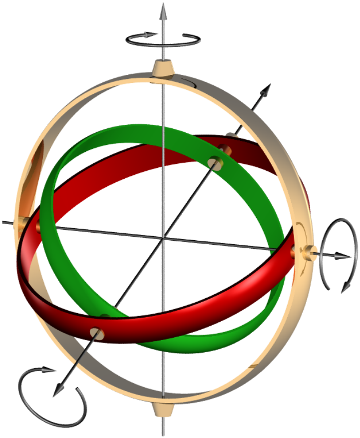
$$
\begin{pmatrix} x^* \\ y^* \end{pmatrix} = \begin{pmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}
$$

and our rotation matrix in $\mathbb{R}^2$ is

$$
R = \begin{pmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{pmatrix}.
$$

Presentation Road Map

1. Introduction
2. Rotation Matrices in $\mathbb{R}^2$
3. **Rotation Matrices in $\mathbb{R}^3$**
4. Gimbal Lock
5. Quaternions



Rotation Matrices in $\mathbb{R}^3$

Major axis approach

How to extend previous result to $\mathbb{R}^3$? General idea:

- $\mathbb{R}^2$ rotation around major axis x, y, and z
- Extend the 2x2 matrix to 3x3

Question

Why selecting 3 axis to rotate an object in $\mathbb{R}^3$?

Rotation Matrices in $\mathbb{R}^3$

Example rotation about z-axis

Example

Step 1 Take 2x2 rotation matrix

$$R = \begin{pmatrix} \cos\gamma & -\sin\gamma \\ \sin\gamma & \cos\gamma \end{pmatrix}$$

.

Step 2 Extend to 3x3 by adding z-axis, and keep z value unchanged

$$R_z = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Rotation Matrices in $\mathbb{R}^3$

Basic rotation matrices

Given angles α, β, and γ, we obtain the *basic* rotations about the x-axis

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix},$$

the y-axis

$$R_y = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix},$$

and the z-axis

$$R_z = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Why are basic rotation matrices bad?

- Expensive to correct matrix *drifting*.
- Hard to interpolate nicely between two rotations.
- **Gimbal Lock!**

- Matrix drift happens when multiple matrices are concatenated.
- Round off errors happens on some of the matrix elements, resulting in *sheared* rotations.
- Need to *orthonormalize* the matrix.
- Gram-Schmidt process is computationally expensive!

Example

Linearly interpolate between identity I and rotation A, which is $\frac{\pi}{2}$ around x-axis.

$$R = 0.5 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + 0.5 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & -0.5 & 0.5 \end{pmatrix}$$

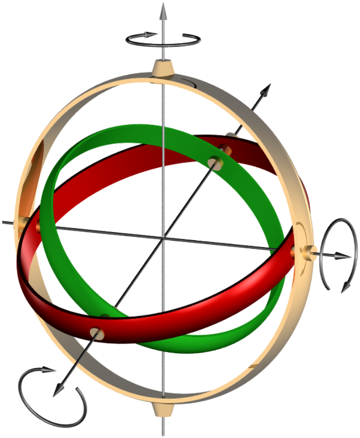
Let $u = (0, 1, 0)^T$. Then $\|Iu\| = \|Au\| = 1$.

But $\|Ru\| = \|(0, 0.5, -0.5)^T\| = \sqrt{0.5}$.

So R is **not a rotation matrix!**

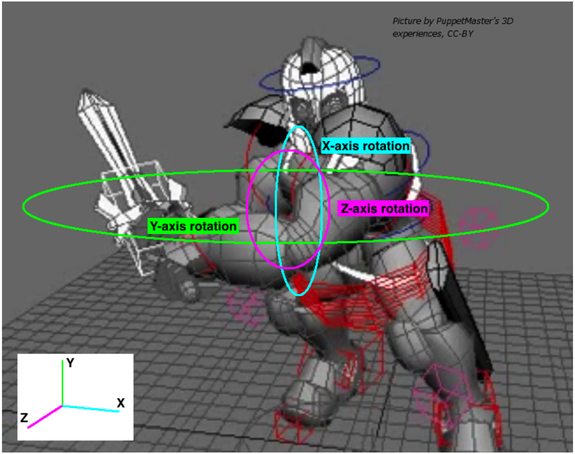
You need a *Spherical Linear Interpolation* (SLERP) for the rotational parts, and LERP for the other parts. Complicated.

Presentation Road Map



Gimbal Lock
Sword movement

We want to rotate a sword by $\beta = \pi$ about the y-axis, and use rotations about the z and y-axis to move the sword down and up again.



We first want to rotate to $\beta = \frac{\pi}{2}$. Let $R = R_x R_y R_z$, which is

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} \cos\gamma & -sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Since $\cos\frac{\pi}{2} = 0$, and $\sin\frac{\pi}{2} = 1$, the above becomes

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \cos\gamma & -sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

which equals

$$R = \begin{pmatrix} 0 & 0 & 1 \\ \sin\alpha\cos\gamma + \cos\alpha\sin\gamma & -\sin\alpha\sin\gamma + \cos\alpha\cos\gamma & 0 \\ -cos\alpha\cos\gamma + \sin\alpha\sin\gamma & \cos\alpha\sin\gamma + \sin\alpha\cos\gamma & 0 \end{pmatrix}.$$

Using the facts that

$$\begin{aligned}\sin(\alpha \pm \gamma) &= \sin\alpha\cos\gamma \pm \cos\alpha\sin\gamma \\ \cos(\alpha \pm \gamma) &= \cos\alpha\cos\gamma \mp \sin\alpha\cos\gamma\end{aligned}$$

we obtain

$$R = \begin{pmatrix} 0 & 0 & 1 \\ \sin(\alpha+\gamma) & \cos(\alpha+\gamma) & 0 \\ -\cos(\alpha+\gamma) & \sin(\alpha+\gamma) & 0 \end{pmatrix}$$

Gimbal Lock

Loosing a degree of freedom

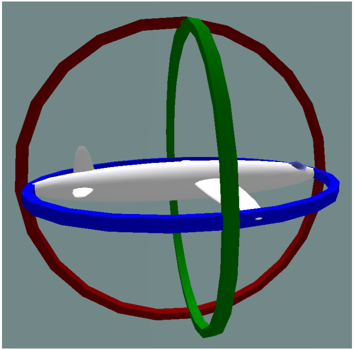
From $\beta = 0$ to $\frac{\pi}{2}$, we now want to rotate $\frac{-\pi}{6}$ about the z-axis, and from $\beta = \frac{\pi}{2}$ to π, we want to rotate $\frac{-\pi}{6}$ about the x-axis. But since

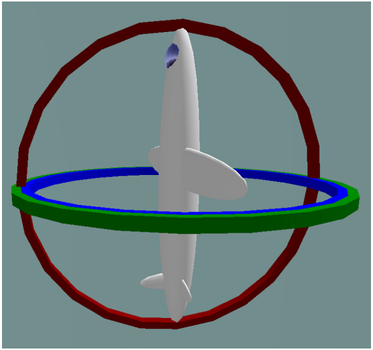
$$R = \begin{pmatrix} 0 & 0 & 1 \\ \sin(\alpha+\gamma) & \cos(\alpha+\gamma) & 0 \\ -\cos(\alpha+\gamma) & \sin(\alpha+\gamma) & 0 \end{pmatrix},$$

is a rotation matrix **about the z-axis**, changing α or γ has the same effect! The angle $\alpha + \gamma$ may change, but the rotation happens always about the z-axis with unexpected results.

Watch this ▸ Video by PuppetMaster's 3D experiences, CC-BY.

Using Euler angles to steer pitch, roll, and yaw.





Pictures by MathsPoetry, CC BY-SA

Rotation Alternatives

How to avoid Gimbal Lock?

There are alternative rotations:

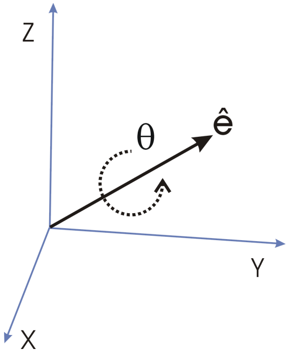
- Euler angles
- Axis-angle representation
- Quaternions

Axis angle representation
Euler vector

A rotation vector

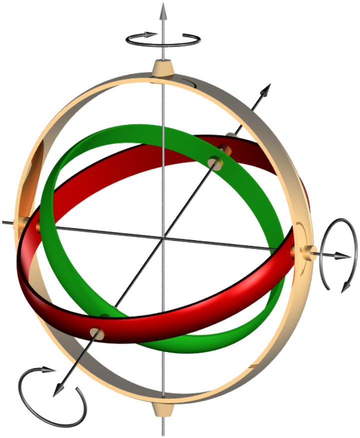
$$r = \theta \hat{e},$$

where $\hat{e} = (e_x, e_y, e_z)^T$ is an arbitrary unit vector, and θ is the angle of rotation about the axis $\hat{e}$.



- No Gimbal Lock.
- Combining two rotations defined by Euler vectors is not simple.
- Cannot use LERP to interpolate.

Presentation Road Map



Quaternions

The holy grail of 3D rotations

- Introduced by William Rowan Hamilton in 1843
- An extension to complex numbers from two dimensions to three
- $i^2 = j^2 = k^2 = ijk = -1$

by JP, CC BY-SA 2.0

Definition

A *quaternion* is a quadruple formed by a *scalar* w, and a *vector* $\mathbf{v} = (x, y, z)$. We write it as

$$\mathbf{q} = (w, \mathbf{v}).$$

Any rotation in $\mathbb{R}^3$ can be represented by a **unit quaternion**.

Definition

A *unit quaternion* is a quaternion $\mathbf{q}$, such that $\|\mathbf{q}\| = 1$, where $\|\mathbf{q}\| = w^2 + x^2 + y^2 + z^2$.

All unit quaternions form a *hypersphere* in $\mathbb{R}^4$. Rotations happen on the surface of this hypersphere.

Given an axis defined by **unit vector**

$$\mathbf{u} = (u_x, u_y, u_z),$$

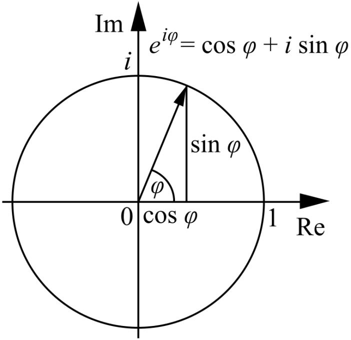
and an angle

$$\theta.$$

How do I obtain a rotation quaternion?

Any point in $\mathbb{R}^2$ can be represented by a complex number, where the *real* part lays on the x-axis, and the *imaginary* part on the y-axis.

Euler's formula relates the trigonometric functions to the *exponential function*



Let $\mathbf{i} = (1,0,0)^T, \mathbf{j} = (0,1,0)$ and $\mathbf{k} = (0,0,1)$. Given $\mathbf{u}$ and θ, an extension to Euler's Formula says that

$$\mathrm{e}^{w\mathbf{v}} = \mathrm{e}^{\frac{\theta}{2}(u_x\mathbf{i}+u_y\mathbf{j}+u_z\mathbf{k})} = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}(u_x\mathbf{i} + u_y\mathbf{j} + u_z\mathbf{k}).$$

Conversion of an axis-angle to a rotation quaternion

Given a unit vector $\mathbf{u} = (u_x, u_y, u_z) \in \mathbb{R}^3$ and a rotation angle θ about $\mathbf{u}$, the corresponding unit quaternion $\mathbf{q}$ is

$$\mathbf{q} = (\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\mathbf{u}).$$

Example

We want to rotate 90° about the y-axis using a quaternion.

We know $\mathbf{q} = (\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\mathbf{u})$. The y-axis can be represented by $\mathbf{u} = (0, 1, 0)$, and $\theta = \frac{\pi}{2}$. We obtain the unit quaternion

$$\mathbf{q} = (\cos\frac{\pi}{4}, 0, \sin\frac{\pi}{4}, 0).$$

Great. What do we do with this rotation quaternion now?

Rotation

Any vector $\mathbf{v} \in \mathbb{R}^3$ can be rotated using a rotation quaternion $\mathbf{q}$ by

$$\mathbf{q}\mathbf{p}\mathbf{q}^{-1},$$

where $\mathbf{p} = (0, \mathbf{v})$, using the Hamilton product.

What is the Hamilton product and how do you take the inverse of a quaternion?

Hamilton Product

Definition

Let $\mathbf{q_0} = (w_0, \mathbf{v_0})$ and $\mathbf{q_1} = (w_1, \mathbf{v_1})$. The *Hamilton Product* is defined as

$$\mathbf{q_0}\mathbf{q_1} = (w_0 w_1 - \mathbf{v_0} \cdot \mathbf{v_1}, w_0\mathbf{v_1} + w_1\mathbf{v_0} + \mathbf{v_0} \times \mathbf{v_1}).$$

Using transposes $\mathbf{q_0} = (w_0, x_0, y_0, z_0)^T$ and $\mathbf{q_1} = (w_1, x_1, y_1, z_1)^T$, we can write

$$\mathbf{q_0}\mathbf{q_1} = \begin{pmatrix} w_0 w_1 - x_0 x_1 - y_0 y_1 - z_0 z_1 \\ w_0 x_1 + x_0 w_1 + y_0 z_1 - z_0 y_1 \\ w_0 y_1 - x_0 z_1 + y_0 w_1 + z_0 x_1 \\ w_0 z_1 + x_0 y_1 - y_0 x_1 + z_0 w_1 \end{pmatrix}.$$

Warning

Quaternion multiplication is not commutative, $\mathbf{q_0}\mathbf{q_1} \neq \mathbf{q_1}\mathbf{q_0}$

Quaternion Inverse

Definition

The inverse of a quaternion $\mathbf{q} = (w, \mathbf{v})$ is computed as

$$\mathbf{q}^{-1} = \frac{(w, -\mathbf{v})}{\|\mathbf{q}\|^2}.$$

Addition and subtraction is the same as with complex numbers.

Quaternions
Rotation matrix

Given a rotation quaternion $\mathbf{q} = (w, x, y, z)$, we can rotate any vector $\mathbf{v} \in \mathbb{R}^3$ using the

Rotation matrix

$$R_q = \begin{pmatrix} 1 - 2y^2 - 2z^2 & 2xy - 2zw & 2xz + 2yw \\ 2xy + 2zw & 1 - 2x^2 - 2z^2 & 2yz - 2xw \\ 2xz - 2yw & 2yz + 2xw & 1 - 2x^2 - 2y^2 \end{pmatrix}.$$

Given a rotation defined by $\mathbf{q_0}$, followed by $\mathbf{q_1}$. They can be composed as

$$\mathbf{q}^* = \mathbf{q_1}\mathbf{q_0}.$$

Example

Sword movement We rotate $\beta = \frac{\pi}{2}$ about the y-axis $v_0 = (0,1,0)$, followed by $\alpha = \frac{-\pi}{6}$ about the x-axis $v_1 = (1,0,0)$. We obtain the quaternions

$$\mathbf{q_0} = (\cos\frac{\pi}{4}, 0, \sin\frac{\pi}{4}, 0), \qquad \mathbf{q_1} = (\cos\frac{-\pi}{12}, \sin\frac{-\pi}{12}, 0, 0).$$

Compose them $\mathbf{q} = \mathbf{q_1}\mathbf{q_0} = (w, x, y, z)$, we get

$$w = \cos\frac{-\pi}{12}\cos\frac{\pi}{4}, \qquad x = \sin\frac{-\pi}{12}\cos\frac{\pi}{4},$$
$$y = \cos\frac{-\pi}{12}\sin\frac{\pi}{4}, \qquad z = \sin\frac{-\pi}{12}\sin\frac{\pi}{4}.$$

The same as with complex numbers.

- LERP

$$\mathbf{q}_t = (1 - t)\mathbf{q}_0 + t\mathbf{q}_1$$

Links:

- Gimbal Lock
- Rotating Objects Using Quaternions
- Understanding Quaternions