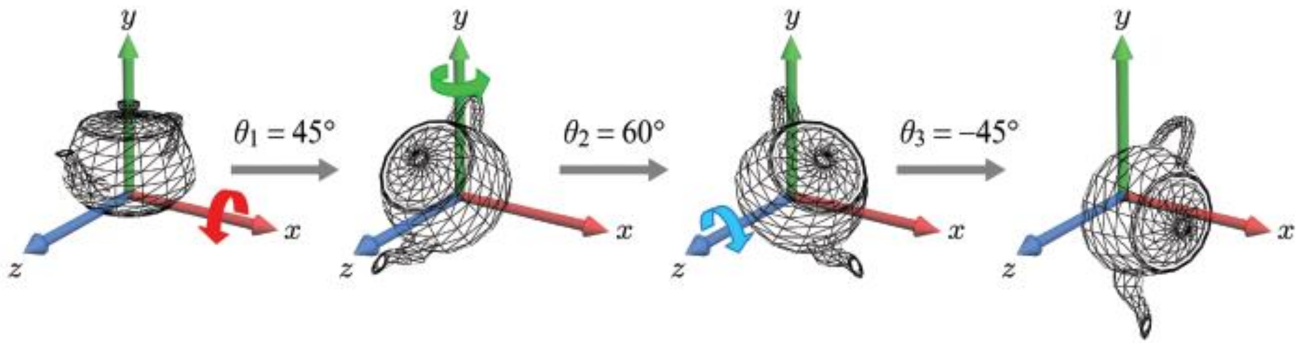
Quaternions

심주용
숙명여자대학교
기계시스템학부

Euler Transforms

- When we ‘successively’ rotate an object about the principal axes, the object acquires an arbitrary orientation. This ‘method’ of determining an object’s orientation is called *Euler transform*. The rotation angles are called the *Euler angles*, which are often denoted as $(\theta_1,\theta_2,\theta_3)$ or $(\theta_x,\theta_y,\theta_z)$.

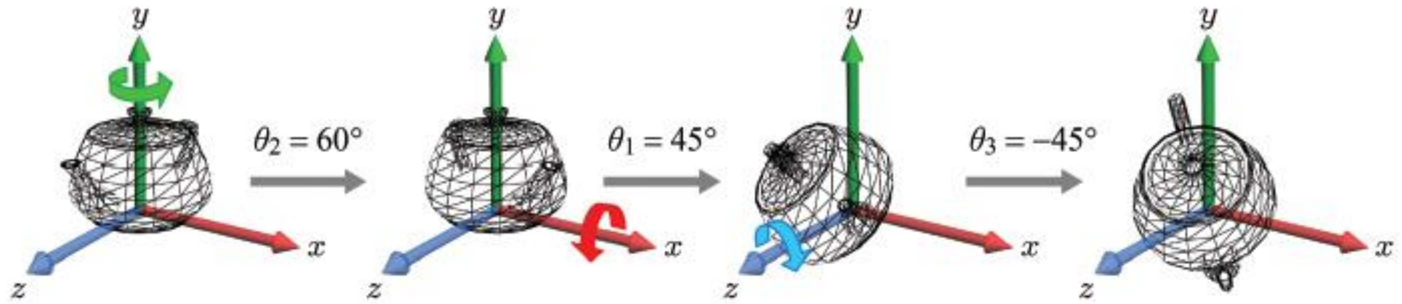


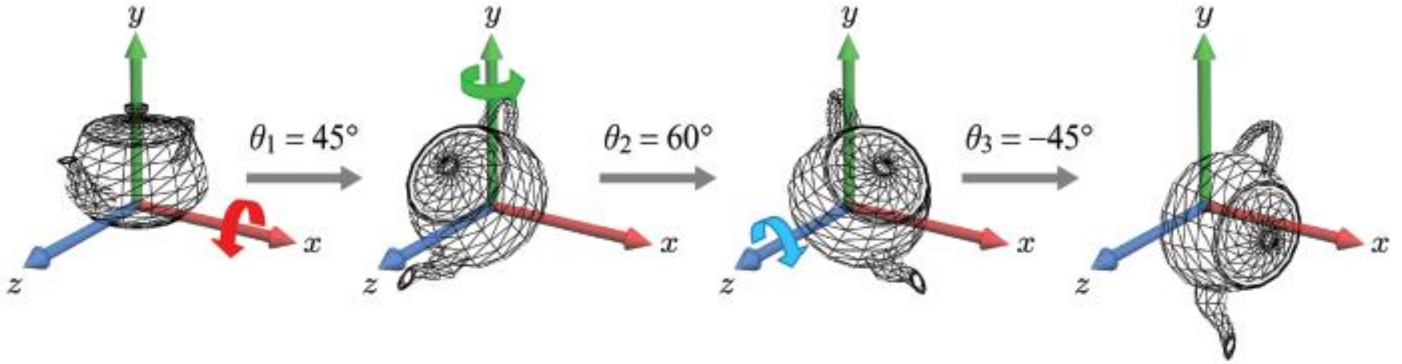
- Concatenating the matrices produces a single ‘rotation’ matrix that defines an arbitrary orientation.

$$
\begin{aligned}
R_z(-45^\circ)R_y(60^\circ)R_x(45^\circ) &= \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{2} & 0 & \frac{\sqrt{3}}{2} \\ 0 & 1 & 0 \\ -\frac{\sqrt{3}}{2} & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix} \\
&= \begin{pmatrix} \frac{\sqrt{2}}{4} & \frac{2+\sqrt{3}}{4} & \frac{-2+\sqrt{3}}{4} \\ -\frac{\sqrt{2}}{4} & \frac{2-\sqrt{3}}{4} & \frac{-2-\sqrt{3}}{4} \\ -\frac{\sqrt{3}}{2} & \frac{\sqrt{2}}{4} & \frac{\sqrt{2}}{4} \end{pmatrix}
\end{aligned}
$$

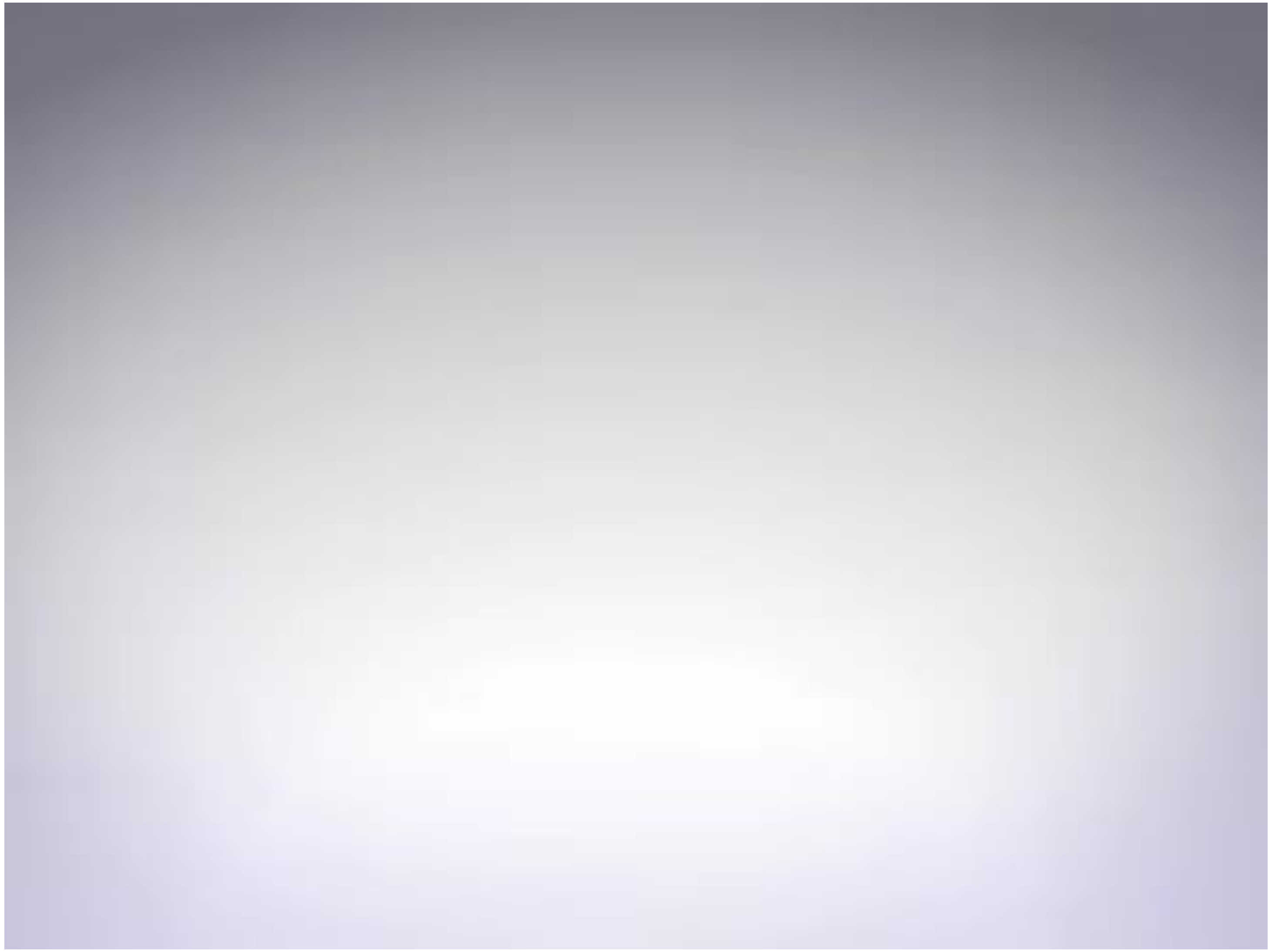
Euler Transforms (cont'd)

- The rotation axes are not necessarily taken in the order of *x*, *y* and *z*. Shown below is the order of *y*, *x* and *z*. Observe that the teapot has a different orientation from the previous one.





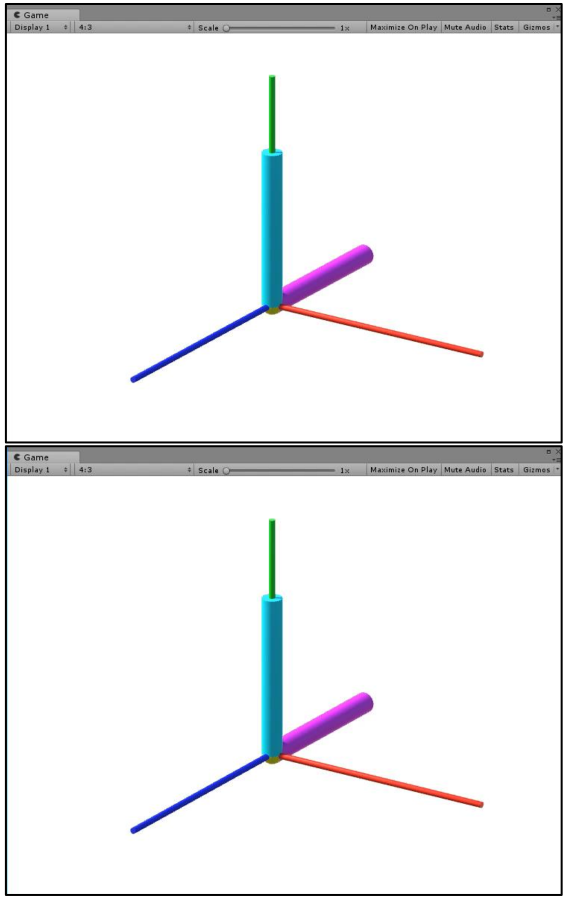
A Problem of Euler Angles: Gimbal Lock

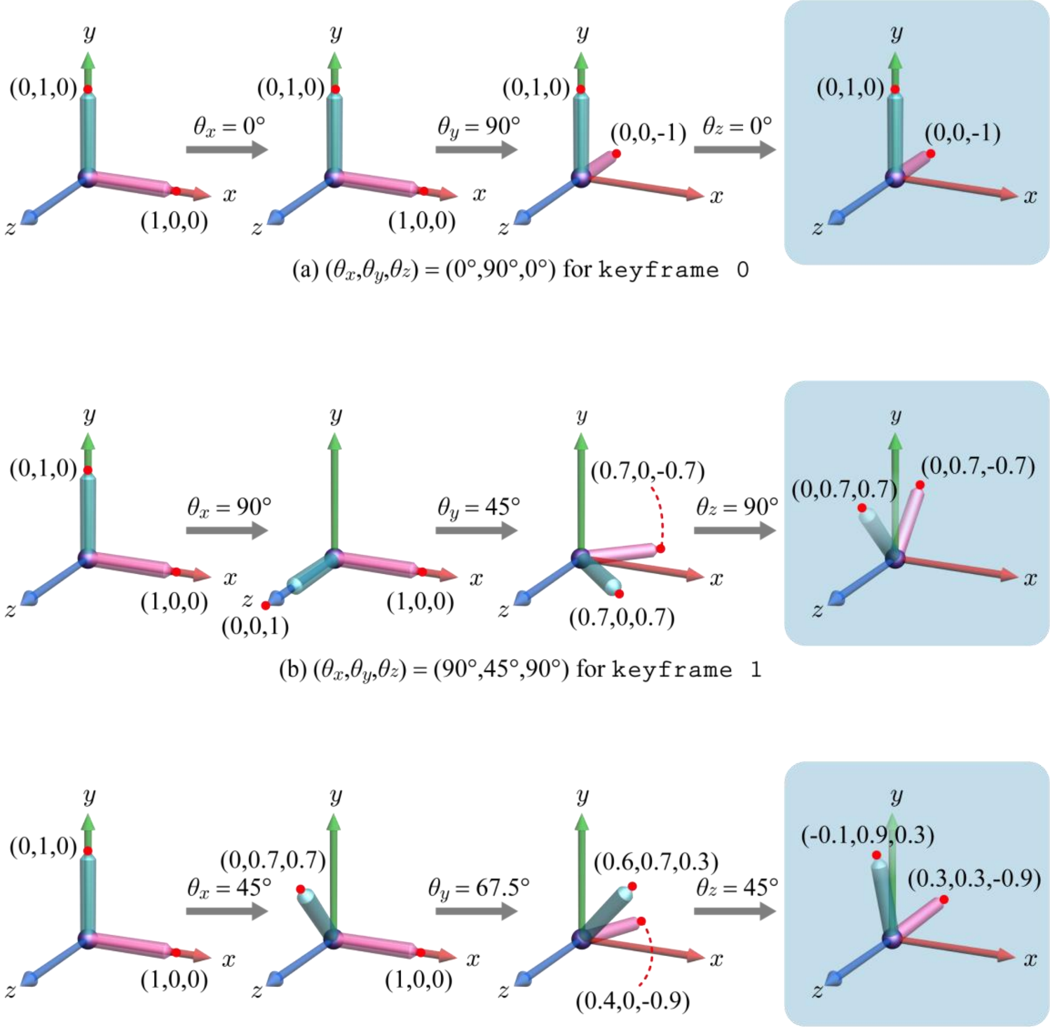


https://www.youtube.com/watch?v=zc8b2Jo7mno

A Problem of Euler Angles: Interpolation

- Euler angles are not always correctly interpolated and so are not suitable for keyframe animation.





(a) $(\theta_x,\theta_y,\theta_z) = (0°,90°,0°)$ for `keyframe 0`

(b) $(\theta_x,\theta_y,\theta_z) = (90°,45°,90°)$ for `keyframe 1`

(c) Interpolated Euler angles $(\theta_x,\theta_y,\theta_z) = (45°,67.5°,45°)$

- The alternative is quaternions.

Source: Introduction to Computer Graphics with OpenGL ES (J. Han)

Quaternion Definition

- A quaternion is an extended complex number.

$$q_x i + q_y j + q_z k + q_w = (q_x, q_y, q_z, q_w) = (\mathbf{q}_v, q_w)$$

$$\begin{aligned} i^2 &= j^2 = k^2 = -1 \\ ij &= k, ji = -k \\ jk &= i, kj = -i \\ ki &= j, ik = -j \end{aligned}$$

$$\begin{aligned} \mathbf{p} &= (p_x, p_y, p_z, p_w) \\ \mathbf{q} &= (q_x, q_y, q_z, q_w) \end{aligned}$$

$$\begin{aligned} \mathbf{pq} = &(p_x i + p_y j + p_z k + p_w)(q_x i + q_y j + q_z k + q_w) \\ = &(p_x q_w + p_y q_z - p_z q_y + p_w q_x)i + \\ &(-p_x q_z + p_y q_w + p_z q_x + p_w q_y)j + \\ &(p_x q_y - p_y q_x + p_z q_w + p_w q_z)k + \\ &(-p_x q_x - p_y q_y - p_z q_z + p_w q_w) \end{aligned}$$

- Conjugate: $$\begin{aligned} \mathbf{q}^* &= (-\mathbf{q}_v, q_w) \\ &= (-q_x, -q_y, -q_z, q_w) \\ &= -q_x i - q_y j - q_z k + q_w \end{aligned}$$

- It is easy to show that (**pq**)*=**q***p***.
- Magnitude: If the magnitude of a quaternion is 1, it's called a *unit quaternion.*

$$\|\mathbf{q}\| = \sqrt{q_x^2 + q_y^2 + q_z^2 + q_w^2}$$

3D Rotation through Quaternions

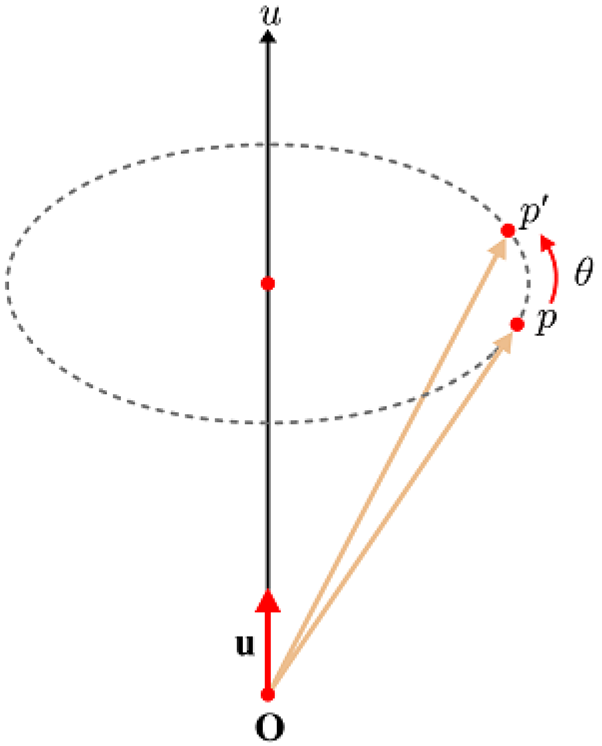
- As extended complex numbers, quaternions can be used to describe 3D rotation.
- Consider rotating a 3D vector, *p*, about an axis, *u*, by θ.
- Represent both “the vector to be rotated” and “the rotation” in quaternions.

Define a quaternion **p** using *p*.

$$\mathbf{p} = (\mathbf{p}_v, p_w) = (p, 0)$$

Define a *unit quaternion* **q** using *u* and θ, **|u|**=1

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

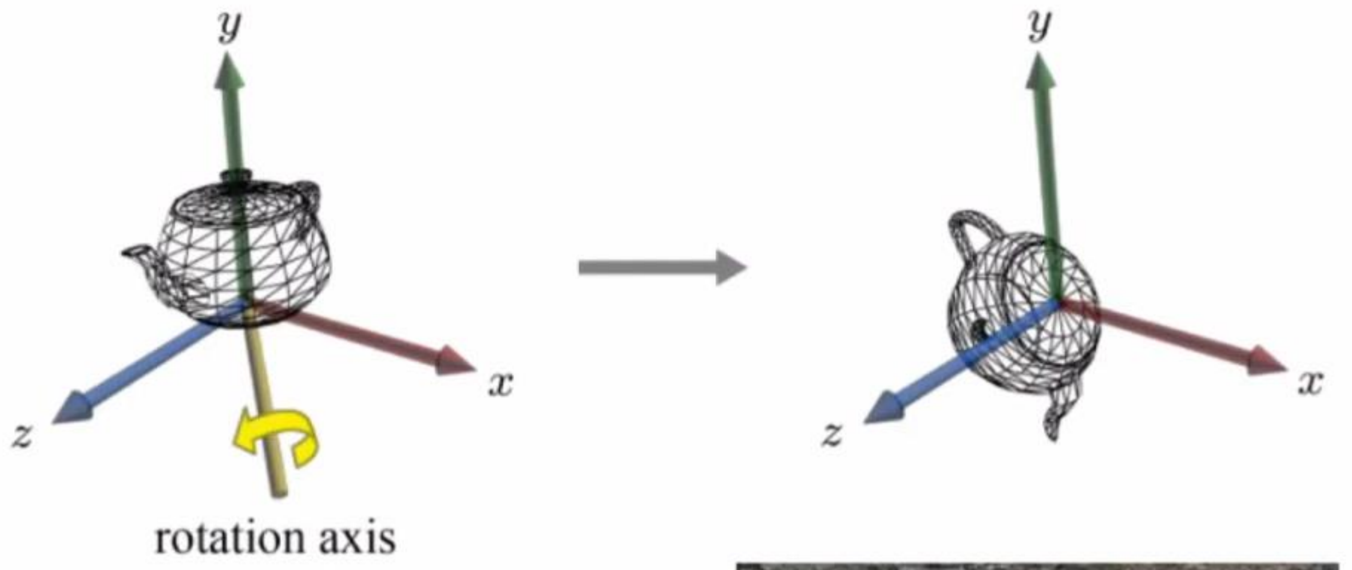


- Compute **qpq***. Then, its *imaginary part* represents the rotated vector.

3D Rotation through Quaternions

Unit quaternion **q** using *u* and θ, **|u|**=1.

$$
\begin{aligned}
\mathbf{q} &= (\mathbf{q}_v, q_w) \\
&= (sin\frac{\theta}{2}\mathbf{u}, cos\frac{\theta}{2})
\end{aligned}
$$



Hamilton

Quaternion and Rotation Matrix

- All transforms we have learned so far are represented in matrices so that multiple transforms can be concatenated to a single matrix. How about quaternions? Fortunately, any quaternion can be converted to a rotation matrix.
- Before presenting that, let's make an important observation.

- Let **q** denote "rotation about **u** by θ," i.e., $\mathbf{q} = (sin\frac{\theta}{2}\mathbf{u}, cos\frac{\theta}{2})$.
- Let **s** denote "rotation about **u** by 2π+θ." Obviously, **q** = **s**.

$$
\begin{aligned}
\mathbf{s} &= (sin\frac{2\pi+\theta}{2}\mathbf{u}, cos\frac{2\pi+\theta}{2}) \\
&= (sin(\pi+\frac{\theta}{2})\mathbf{u}, cos(\pi+\frac{\theta}{2})) \\
&= (-sin\frac{\theta}{2}\mathbf{u}, -cos\frac{\theta}{2})
\end{aligned}
$$

- The above shows that **s** = -**q**.
- It is found that **q** = -**q**, i.e., $(q_x, q_y, q_z, q_w) = (-q_x, -q_y, -q_z, -q_w)$.

Quaternion and Rotation Matrix (cont'd)

- A quaternion **q** representing a rotation can be converted into a matrix form. If **q** = (q_x,q_y,q_z,q_w), the rotation matrix is defined as follows:

$$\begin{pmatrix} 1-2(q_y^2+q_z^2) & 2(q_xq_y-q_wq_z) & 2(q_xq_z+q_wq_y) & 0 \\ 2(q_xq_y+q_wq_z) & 1-2(q_x^2+q_z^2) & 2(q_yq_z-q_wq_x) & 0 \\ 2(q_xq_z-q_wq_y) & 2(q_yq_z+q_wq_x) & 1-2(q_x^2+q_y^2) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Conversely, given a rotation matrix, we can compute its quaternion. It requires us to extract $\{q_x,q_y,q_z,q_w\}$ given the above matrix.
  - Compute the sum of all diagonal elements.

$$4-4(q_x^2+q_y^2+q_z^2) = 4-4(1-q_w^2) = 4q_w^2$$

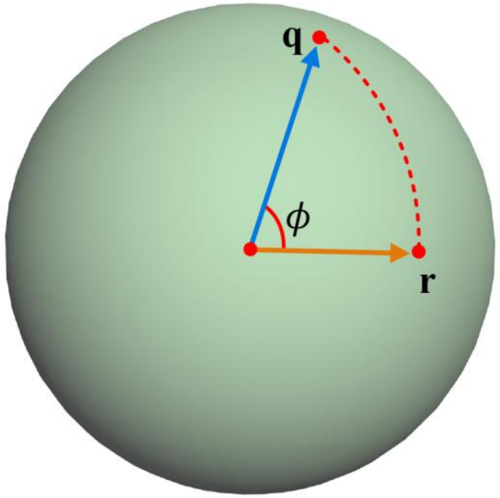
  - So, we obtain q_w (one positive, the other negative; d and $-d$).
  - Subtract m_{12} from m_{21} of the above matrix.

$$m_{21}-m_{12} = 2(q_xq_y+q_wq_z)-2(q_xq_y-q_wq_z) = 4q_wq_z$$

  - As we know q_w, we can compute q_z. Similarly, we can compute q_x and q_y.
  - Two quaternions are obtained, (a, b, c, d) and ($-a$, $-b$, $-c$, $-d$), which are proven to be identical.

Quaternion Interpolation

- The set of all possible quaternions makes up a 4D unit sphere. Consider two unit quaternions, **q** and **r**. Let ϕ denote the angle between **q** and **r**.



$$\begin{aligned} \mathbf{q}\cdot\mathbf{r} &= (q_x, q_y, q_z, q_w)\cdot(r_x, r_y, r_z, r_w) \\ &= q_x r_x + q_y r_y + q_z r_z + q_w r_w \\ \mathbf{q}\cdot\mathbf{r} &= \|\mathbf{q}\|\|\mathbf{r}\| cos\phi = cos\phi \end{aligned}$$

$$cos\phi = q_x r_x + q_y r_y + q_z r_z + q_w r_w$$

$$\phi = arccos(q_x r_x + q_y r_y + q_z r_z + q_w r_w)$$

- The interpolated quaternion must lie on the shortest arc connecting **q** and **r**. It is defined using parameter *t* in the range of [0,1]:

$$\frac{sin(\phi(1-t))}{sin\phi}\mathbf{q} + \frac{sin(\phi t)}{sin\phi}\mathbf{r}$$

- This is called *spherical linear interpolation* (slerp)