Quaternion (쿼터니언, 사원수)

최근에는 3차원 공간에서의 회전을 표현할 때 Quaternion을 사용한다.

Quaternion으로 회전을 표현할 때는 하나의 회전축에 대한 회전각도로 표현이 가능해 Gimbal Lock 같은 문제를 원천적으로 피할수 있다.

핵심 요약

1. 3차원 회전을 숫자 4개로 표현
   - 3차원 공간의 회전을 표현할 때 사용하는 Quaternion은 절대값이 1이 되어야 한다는 제약 조건이 있어 4자유도가 아닌 3자유도다.
2. 회전 행렬과 호환
   - Graphics Pipeline은 행렬 계산에 유리하므로 Quaternion을 직접 사용하기 보단 Matrix → Quaternion으로 바꾸거나 계산된 Quaternion을 Matrix로 변환해 사용한다.
3. 회전의 보간(Interpolation)에 유리

$$q = xi + yj + zk + w = (u, w)$$

Quaternion 계산 핵심

핵심 1 : 벡터 v를 회전축 n에 대해서 θ만큼 회전시키고 싶다면

1. Quaternion $p = (v, 0)$을 만든다., Vector → Quaternion 변환
2. Quaternion $q = \left(n \sin\left(\frac{\theta}{2}\right), \cos\left(\frac{\theta}{2}\right)\right)$을 만든다., 회전 표현
3. Conjugate(켤레) Quaternion $q^* = \left(-n \sin\left(\frac{\theta}{2}\right), \cos\left(\frac{\theta}{2}\right)\right)$
4. Quaternion $p' = (v', 0) = q p q^*$로 회전시킨다.

   ㄴ 회전된 Quaternion

핵심 2 : Quaternion으로부터 회전행렬을 만들어서 쓴다.
회전행렬을 Quaternion으로 만들어 사용할 수 있다.

$q = (q_1, q_2, q_3, q_4)$라고 할 때

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} = \begin{bmatrix} 1-2q_2^2-2q_3^2 & 2q_1q_2+2q_3q_4 & 2q_1q_3-2q_2q_4 \\ 2q_1q_2-2q_3q_4 & 1-2q_1^2-2q_3^2 & 2q_2q_3+2q_1q_4 \\ 2q_1q_3+2q_2q_4 & 2q_2q_3-2q_1q_4 & 1-2q_1^2-2q_2^2 \end{bmatrix}$$

그래픽스 파이프라인은 행렬계산에 최적화되어 있기에 Quaternion을 직접 사용하지 않고 행렬로 바꿔 사용하는 방식도 많이 사용된다.
경우에 따라 메모리를 아끼기 위해 Quaternion으로 저장하기도 한다.

쿼터니언의 구성

$q = xi + yj + zk + w = (u, w)$, i, j, k는 허수

$i^2 = j^2 = k^2 = ijk = -1$

$ij = k = -ji$

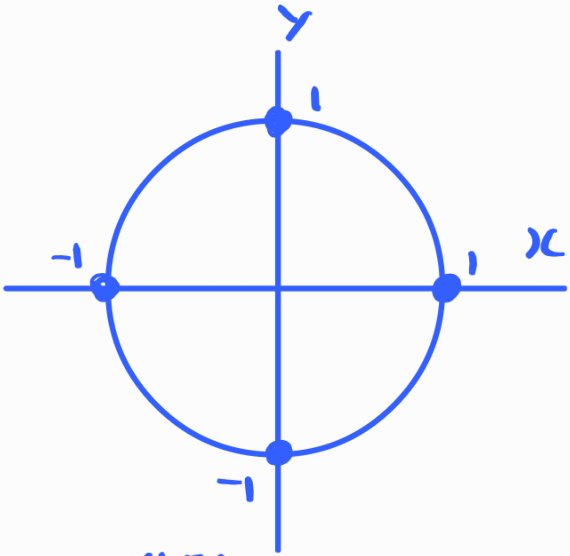
$jk = i = -kj$

$ki = j = -ik$

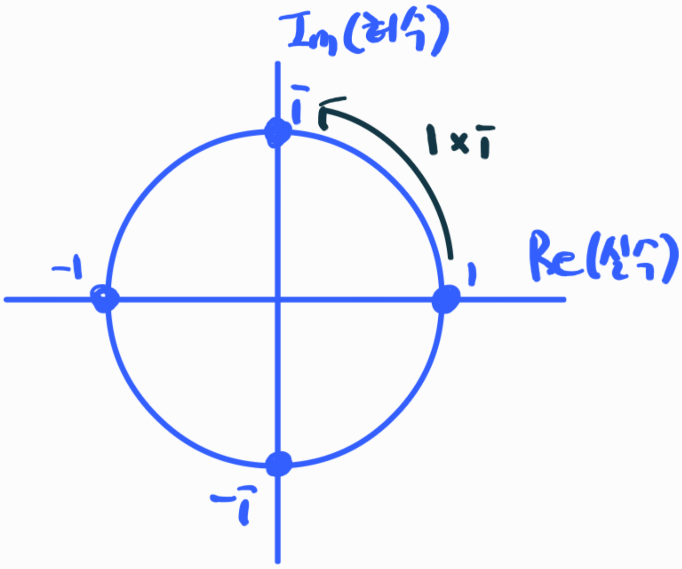
실수 : real number (x, y, z, w)

허수 : imaginary number (i, j, k)

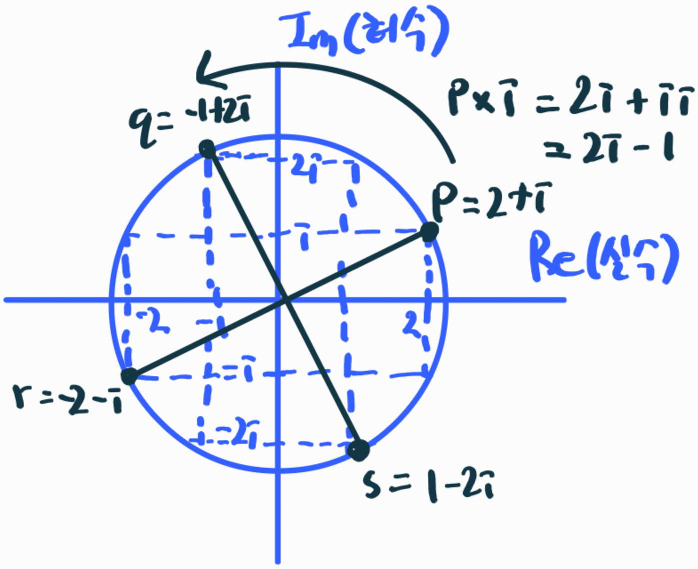
복소수 : Complex number (q , $a+bi$ 형태)



일반적인 2D 좌표계



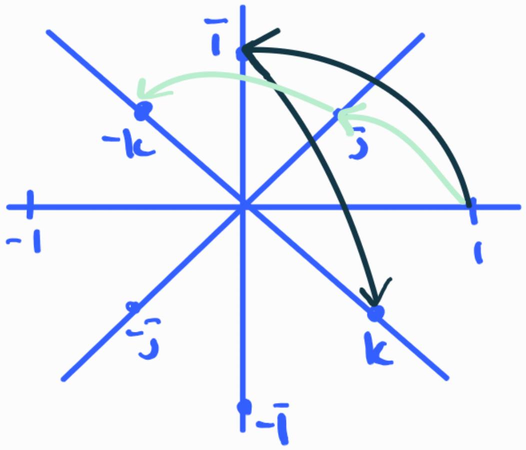
실수와 허수를 이용



$p \times i = 2i + ii$

$= 2i - 1 = -1 + 2i = q$ (90도 회전한 것과 같다)

원하는 각도로 회전하고 싶을 때
그 회전에 대응하는 복소수를 찾아서
곱해주면 cos, sin 대신 복소수끼리의
곱셈으로 복소수를 회전시키는 것과
동일한 효과를 얻을 수 있다.



$ij=k$, i가 되서 j축을 기준으로 회전해 k가 된다

$ji=-k$, j가 되서 i축을 기준으로 회전해 -k가 된다.

$ij=-ji$

기본 연산

$u=v$ 이고 $a=b$ 이면 $(u,a)=(v,b)$, Quaternion의 4개의 수가 같으면 같은 Quaternion

$(u,a)\pm(v,b)=(u\pm v, a\pm b)$, 각각 ± 계산

$q = xi + yj + zk + w = (u,w)$, i,j,k는 허수

$i^2 = j^2 = k^2 = ijk = -1$

$ij = k = -ji$

$jk = i = -kj$

$ki = j = -ik$

곱하기

$$
\begin{aligned}
q_a q_b &= (a, w_a)(b, w_b) = ((x_a, y_a, z_a), w_a)((x_b, y_b, z_b), w_b) \\
&= (x_a i + y_a j + z_a k + w_a)(x_b i + y_b j + z_b k + w_b) \\
&= (w_a w_b - x_a x_b - y_a y_b - z_a z_b) \\
&\quad + (w_a x_b + w_b x_a + y_a z_b - y_b z_a) i \\
&\quad + (w_a y_b + w_b y_a + z_a x_b - z_b x_a) j \\
&\quad + (w_a z_b + w_b z_a + x_a y_b - x_b y_a) k
\end{aligned}
$$

$q_a q_b = (w_a b + w_b a + a \times b, w_a w_b - a \cdot b)$

$a \cdot b = x_a x_b + y_a y_b + z_a z_b$

$a \times b = (y_a z_b - y_b z_a) i + (z_a x_b - z_b x_a) j + (x_a y_b - x_b y_a) k$

절대값

Quaternion Norm

$$\|q\| = \|xi + yj + zk + w\| = \|(u, w)\|$$
$$= \sqrt{x^2 + y^2 + z^2 + w^2} = \sqrt{\|u\|^2 + w^2}$$

Inverse Quaternion

Conjugate(켤레) Quaternion

$q = (u, w)$의 conjugate인 $q^* = (-u, w)$

Inverse

$$qq^* = \|q\|^2 = \|q^*\|^2$$
$$q^{-1} = \frac{q^*}{\|q\|^2}$$

만약 $\|q\| = 1$인 단위 Quaternion이면 $q^{-1} = q^*$

회전 - 벡터 v를 회전시킬 때 $v \rightarrow P = (v, 0)$

$$qPq^{-1} = qPq^*$$

q = 회전 Quaternion(단위 쿼터니언)

q^* = 회전 Quaternion의 Conjugate

$$= (u, w)(v, 0)(-u, w)$$
$$= (u, w)(wv - v \times u, v \cdot u)$$, Quaternion 곱셈
$$= ((w^2 - u \cdot v)v + 2(u \cdot v)u + 2w(u \times v), 0)$$

$q = (n \sin(\frac{\theta}{2}), \cos(\frac{\theta}{2}))$, $\|n\| = 1$인 경우 q로 unit Quaternion이 된다.

n = 회전축, θ = 회전각도

$$qPq^* = \cos(\theta)v + (1 - \cos(\theta))(n \cdot v)n + \sin(\theta)(n \times v)$$

앞에서 유도한 회전과 동일

$$R_n(v) = proj_n(v) + R_n(v_\perp)$$
$$= (n \cdot v)n + \cos\theta v_\perp + \sin\theta(n \times v)$$
$$= (n \cdot v)n + \cos\theta(v - (n \cdot v)n) + \sin\theta(n \times v)$$
$$= \cos\theta v + (1 - \cos\theta)(n \cdot v)n + \sin\theta(n \times v)$$