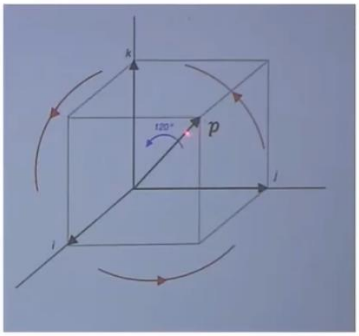
Theorem of Quaternion - Rotation of a Vector

Given a vector $p = x_1\vec{i} + x_2\vec{j} + x_3\vec{k}$, it can be converted into a quaternion from as

$$p = \begin{bmatrix} 0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix},$$

where the scalar part is 0.

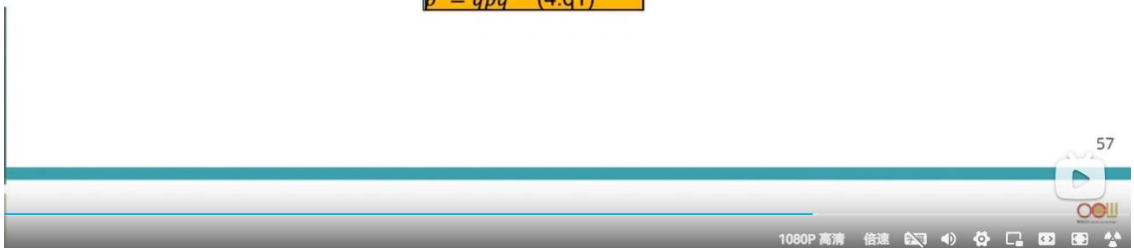


Now an arbitrary axis of rotation $\vec{n}$ expressed by a quaternion is given by

$$q = cos\left(\frac{\theta}{2}\right) + sin\left(\frac{\theta}{2}\right)\vec{n}.$$

Then vector p rotated by quaternion q can be expressed as

$$p' = qpq^{-1} \text{ (4.q1)}$$

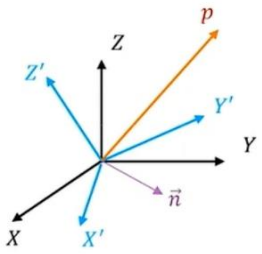


情况一：旋转向量 坐标系不变

Theorem of Quaternion- Rotation of a Coordinate System

Conversely, given a vector $p = x_1\vec{i} + x_2\vec{j} + x_3\vec{k}$ in the coordinate system XYZ, the XYZ coordinate system is rotated by a quaternion q to $X'Y'Z'$, and vector p expressed in the $X'Y'Z'$ coordinate system denoted as p' is expressed as

$$p' = q^{-1}pq \text{ (4.q3)}$$



情况二：旋转坐标系，在旋转后的坐标系中表示原来那个向量

在另一篇文章里，给出的公式也是相同的

四元数可以在某个坐标系中，表示一个向量坐标旋转的变化

Theorem 1 *For any unit quaternion*

$$q = q_0 + \boldsymbol{q} = \cos\frac{\theta}{2} + \boldsymbol{u}\sin\frac{\theta}{2}, \tag{4}$$

and for any vector $\boldsymbol{v} \in \mathbb{R}^3$ *the action of the operator*

$$L_q(\boldsymbol{v}) = q\boldsymbol{v}q^*$$

on $\boldsymbol{v}$ *is equivalent to a rotation of the vector through an angle* θ *about* $\boldsymbol{u}$ *as the axis of rotation.*

给定一个四元数q，由角度和轴u定义

坐标系（coordinate frame）绕着轴u旋转一个角度 v不旋转

得到的是旋转后的坐标系中的v的表达

Theorem 2 *For any unit quaternion*

$$q = q_0 + \boldsymbol{q} = \cos\frac{\theta}{2} + \boldsymbol{u}\sin\frac{\theta}{2},$$

and for any vector $\boldsymbol{v} \in \mathbb{R}^3$ *the action of the operator*

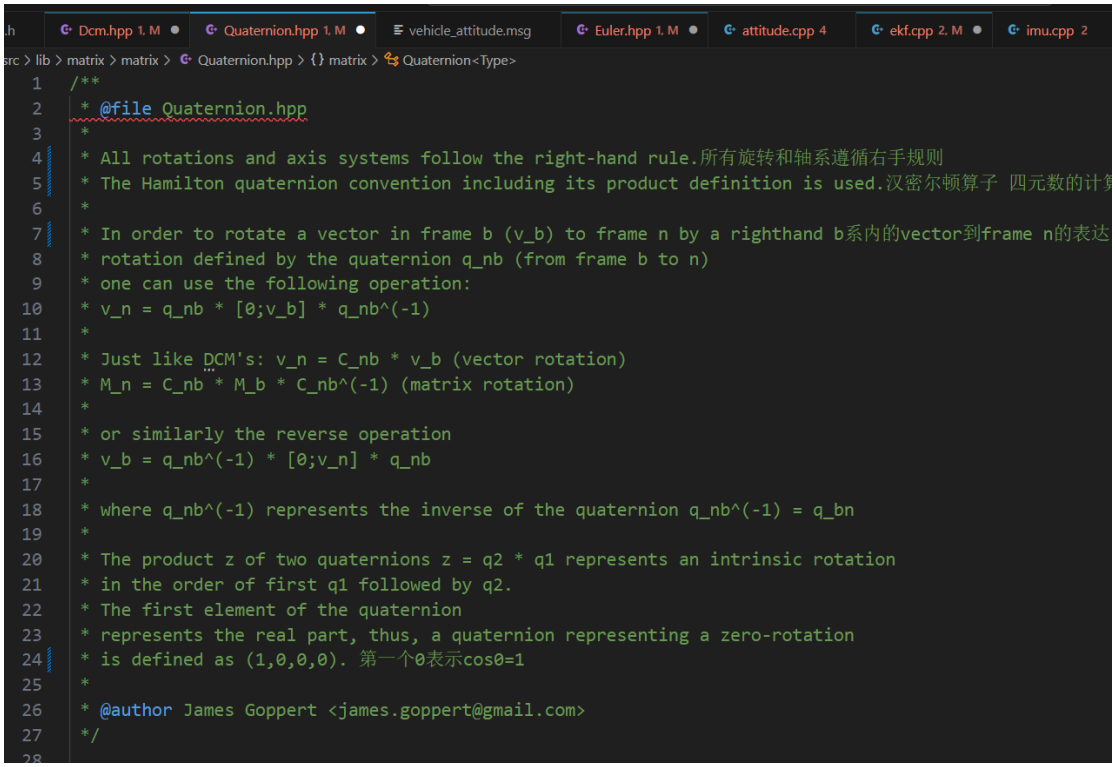
$$L_{q^*}(\boldsymbol{v}) = q^*\boldsymbol{v}(q^*)^* = q^*\boldsymbol{v}q$$

is a rotation of the coordinate frame about the axis $\boldsymbol{u}$ *through an angle* θ *while* $\boldsymbol{v}$ *is not rotated.*

Equivalently, the operator L_{q^*} rotates the vector $\boldsymbol{v}$ with respect to the coordinate frame through an angle $-\theta$ about $\boldsymbol{q}$.

The quaternion operator $L_q(\boldsymbol{v}) = q\boldsymbol{v}q^*$ may be interpreted as a *point or vector rotation* with respect to the (fixed) coordinate frame. The quaternion operator $L_{q^*}(\boldsymbol{v}) = q^*\boldsymbol{v}q$ may be interpreted as a *coordinate frame rotation* with respect to the (fixed) space of points.

但是在 PX4 里



可以看到

```
In order to rotate a vector in frame b (v_b) to frame n by a righthand
rotation defined by the quaternion q_nb (from frame b to n)
one can use the following operation:
v_n = q_nb * [0;v_b] * q_nb^(-1)
```

旋转向量 vector
在 frame b 中用 v_b 表示　在 frame n 中用 v_n 表示
然后四元数 quaternion q_nb 表示 frame b to n
那么按照前面两类四元数旋转的公式 这里应该是坐标系旋转后，在旋转后的坐标系中表示向量 v，应该用第二个公式
$V_n = q^* \times v_b \times q$

但是这里是反的

这里也是一样，比如 conjugate
Q_21 描述 frame 1 到 2 的旋转
那么我觉得 v_2 应该是 q_21^-1 * v_1 * q_21

```
/**
 * Rotates vector v_1 in frame 1 to vector v_2 in frame 2
 * using the rotation quaternion q_21
 * describing the rotation from frame 1 to 2
 * v_2 = q_21 * v_1 * q_21^-1
 *
 * @param vec vector to rotate in frame 1 (typically body frame)
 * @return rotated vector in frame 2 (typically reference frame)
 */
Vector3<Type> conjugate(const Vector3<Type> &vec) const
{
        const Quaternion &q = *this;
        Quaternion v(Type(0), vec(0), vec(1), vec(2));
        Quaternion res = q * v * q.inversed();
        return Vector3<Type>(res(1), res(2), res(3));
}

/**
 * Rotates vector v_2 in frame 2 to vector v_1 in frame 1
 * using the rotation quaternion q_21
 * describing the rotation from frame 1 to 2
 * v_1 = q_21^-1 * v_2 * q_21
 *
 * @param vec vector to rotate in frame 2 (typically reference frame)
 * @return rotated vector in frame 1 (typically body frame)
 */
Vector3<Type> conjugate_inversed(const Vector3<Type> &vec) const
{
        const Quaternion &q = *this;
        Quaternion v(Type(0), vec(0), vec(1), vec(2));
        Quaternion res = q.inversed() * v * q;
        return Vector3<Type>(res(1), res(2), res(3));
}
```

2.1.3 Unit quaternion

Every rotation can be parametrized by a single rotation about a fixed axis described by the unit vector $\vec{k}$ and a rotation angle α. This is known as eigenaxis rotation and embodies the shortest rotation between two orientations. Based on this, a unit quaternion is defined as

$$\mathbf{q} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^\top = \begin{bmatrix} q_0 \\ \mathbf{q}_{1:3} \end{bmatrix} = \begin{bmatrix} \cos(\frac{\alpha}{2}) \\ \vec{k}\sin(\frac{\alpha}{2}) \end{bmatrix}. \tag{10}$$

The adjoint, norm and inverse of a quaternion $\mathbf{q}$ are

$$\bar{\mathbf{q}} = \begin{bmatrix} q_0 \\ -\mathbf{q}_{1:3} \end{bmatrix}, \tag{11}$$

$$\|\mathbf{q}\| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}, \tag{12}$$

$$\mathbf{q}^{-1} = \frac{\bar{\mathbf{q}}}{\|\mathbf{q}\|}. \tag{13}$$

The multiplication of two quaternions q and p is then defined[2] by

$$\mathbf{q} \cdot \mathbf{p} := Q(\mathbf{q})\mathbf{p} \tag{14}$$

where

$$Q(\mathbf{q}) := \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix}. \tag{15}$$

Note that the quaternion, which corresponds to a rotation matrix $\mathbf{I}$ is given by

$$\mathbf{q}_\mathbf{I} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{16}$$

The rotation of a vector $\vec{r}$ by a quaternion $\mathbf{q}$ is computed by

$$\mathbf{p}(\vec{r}_r) = \mathbf{q} \cdot \mathbf{p}(\vec{r}) \cdot \bar{\mathbf{q}} \tag{17}$$

where $\vec{r}_r$ is the rotated vector and $\mathbf{p}(\cdot)$ is the quaternion representation of a vector:

$$\mathbf{p}(\vec{r}) = \begin{bmatrix} 0 \\ \vec{r} \end{bmatrix}. \tag{18}$$

The mapping from Euler angles to unit quaternions for the ZYX-sequence is

$$\mathbf{q}(\psi, \theta, \phi) = \begin{bmatrix} c_{\frac{\phi}{2}} c_{\frac{\theta}{2}} c_{\frac{\psi}{2}} + s_{\frac{\phi}{2}} s_{\frac{\theta}{2}} s_{\frac{\psi}{2}} \\ -c_{\frac{\phi}{2}} s_{\frac{\theta}{2}} s_{\frac{\psi}{2}} + c_{\frac{\theta}{2}} c_{\frac{\psi}{2}} s_{\frac{\phi}{2}} \\ c_{\frac{\phi}{2}} c_{\frac{\psi}{2}} s_{\frac{\theta}{2}} + s_{\frac{\phi}{2}} c_{\frac{\psi}{2}} s_{\frac{\psi}{2}} \\ c_{\frac{\phi}{2}} c_{\frac{\theta}{2}} s_{\frac{\psi}{2}} - s_{\frac{\phi}{2}} c_{\frac{\psi}{2}} s_{\frac{\theta}{2}} \end{bmatrix}. \tag{19}$$

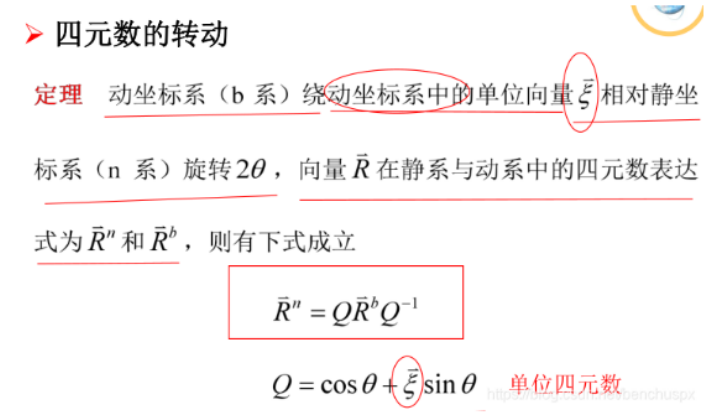
[2]Note: For ease of interpretation, the quaternion multiplication used in this report differs from the one defined in [3]. With the definition in (14), the quaternion multiplication $\mathbf{q} \cdot \mathbf{p}$ corresponds to first a rotation of $\mathbf{q}$ and afterwards a rotation of $\mathbf{p}$ in the new coordinate system.

这是 ETH 写的关于四旋翼控制的一篇文章，应该就是开发 PX4 的团队

（17）这个公式是在一个坐标系中旋转向量的公式，和 px4 里面的不是一类情况

四元数的用途

顺便提一下，四元数的实际意义还是用来表示向量旋转或者坐标系转换比较方便，这主要是利用四元数三角式的性质



只有一篇文章里截图的公式看起来和 PX4 中一样

旋转坐标系，q 表示动坐标系（机体系）绕单位向量、旋转某个角度到静坐标系（惯性系）的四元数，在静系 n 系中 Rn 的表达，和 PX4 中的公式一样，但是和前面其他资料的公式都不一样