Robotics &
Internet-of-Things

ROS

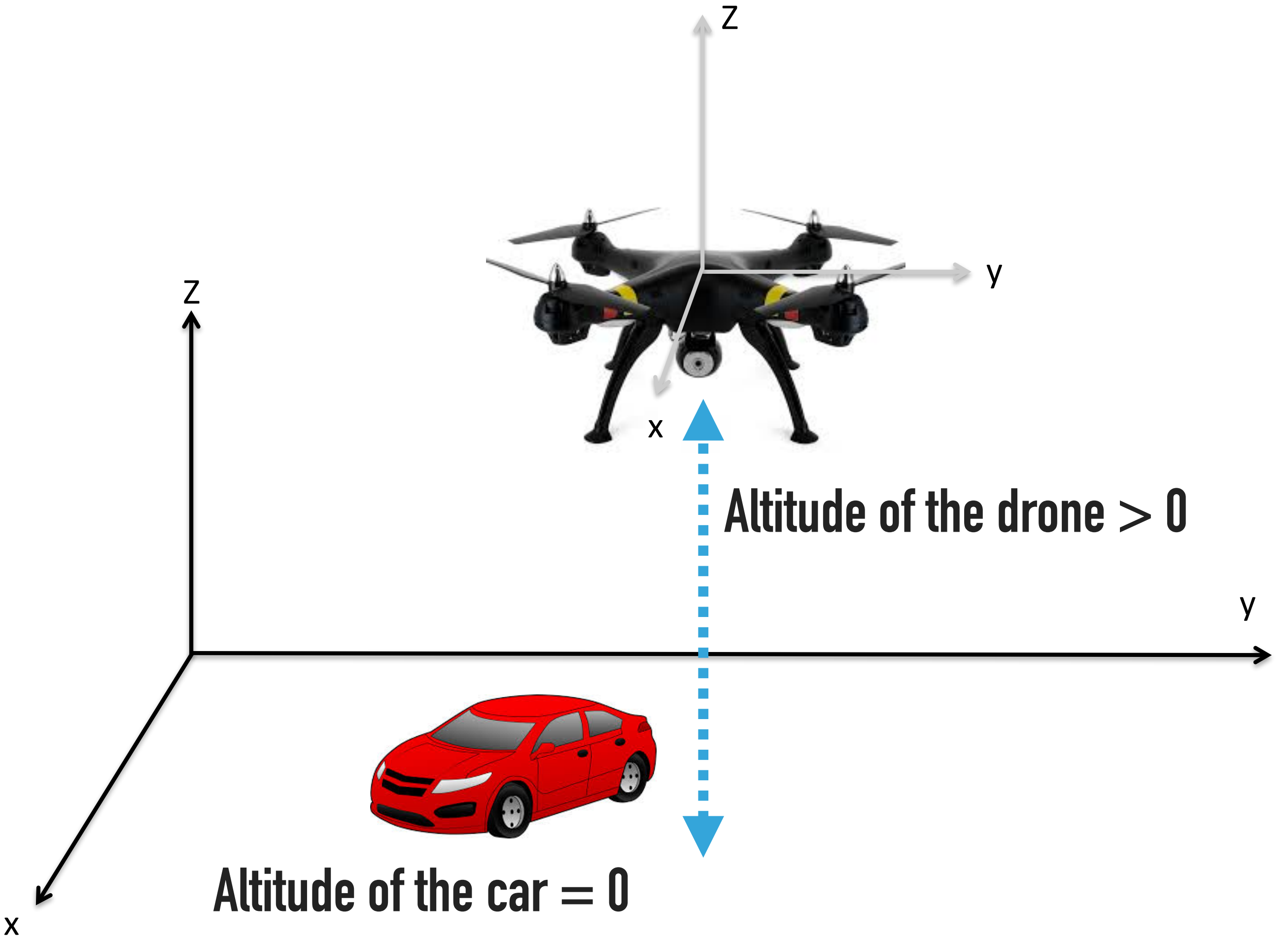
Udemy Course
Robot Operating System (Part II)
Navigation and SLAM

Prof. Anis Koubaa

Section
Frames and 3D Transformations

https://www.udemy.com/user/anis-koubaa/

3D Coordinate System



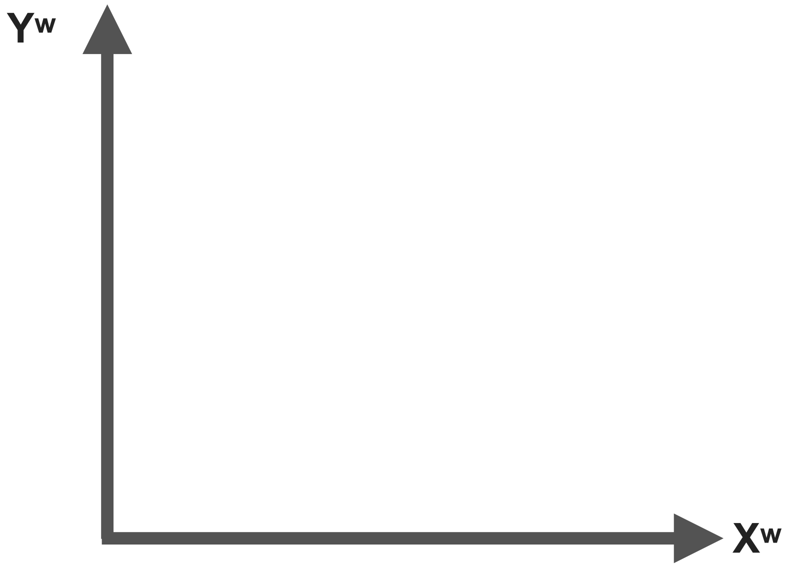
Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

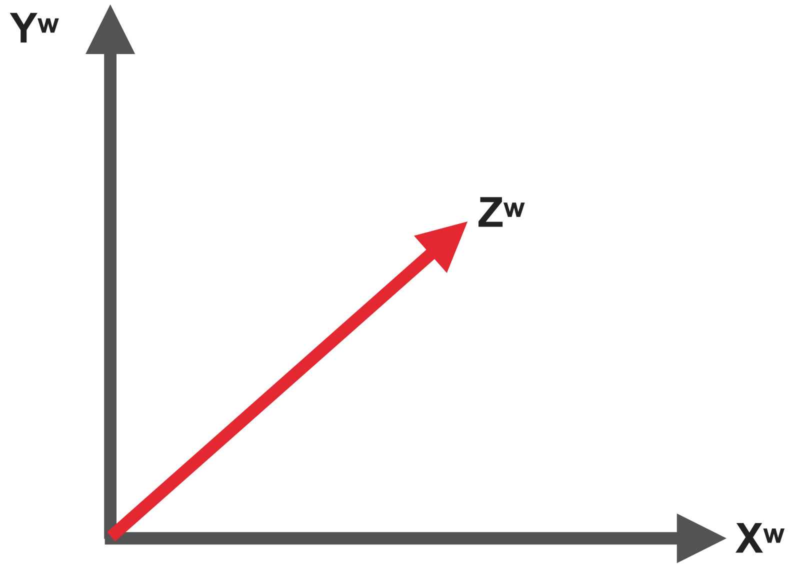
Right-Hand Rule

https://www.udemy.com/user/anis-koubaa/

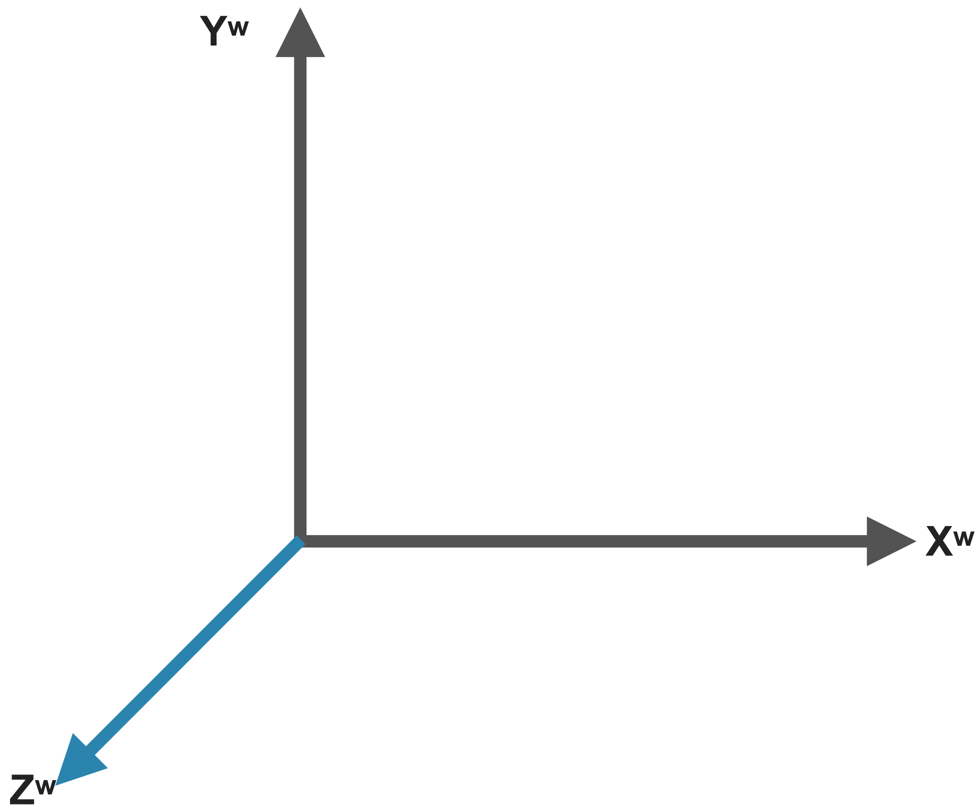
3D Coordinate Frame



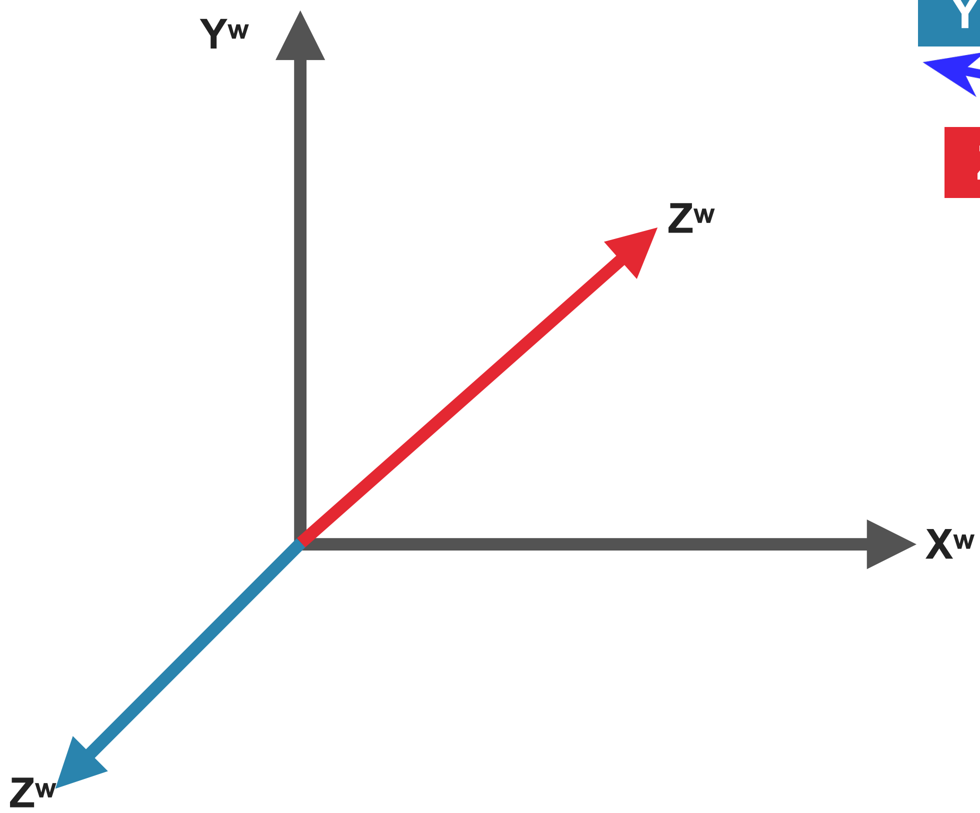
3D COORDINATE FRAME

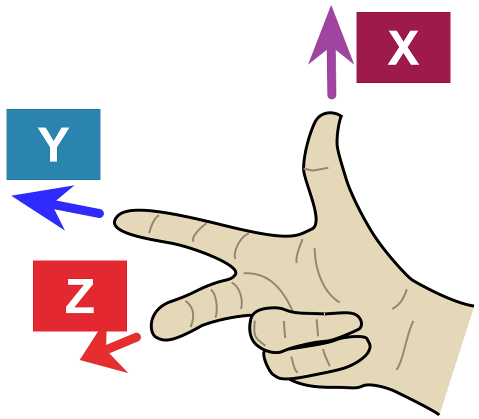


3D COORDINATE FRAME

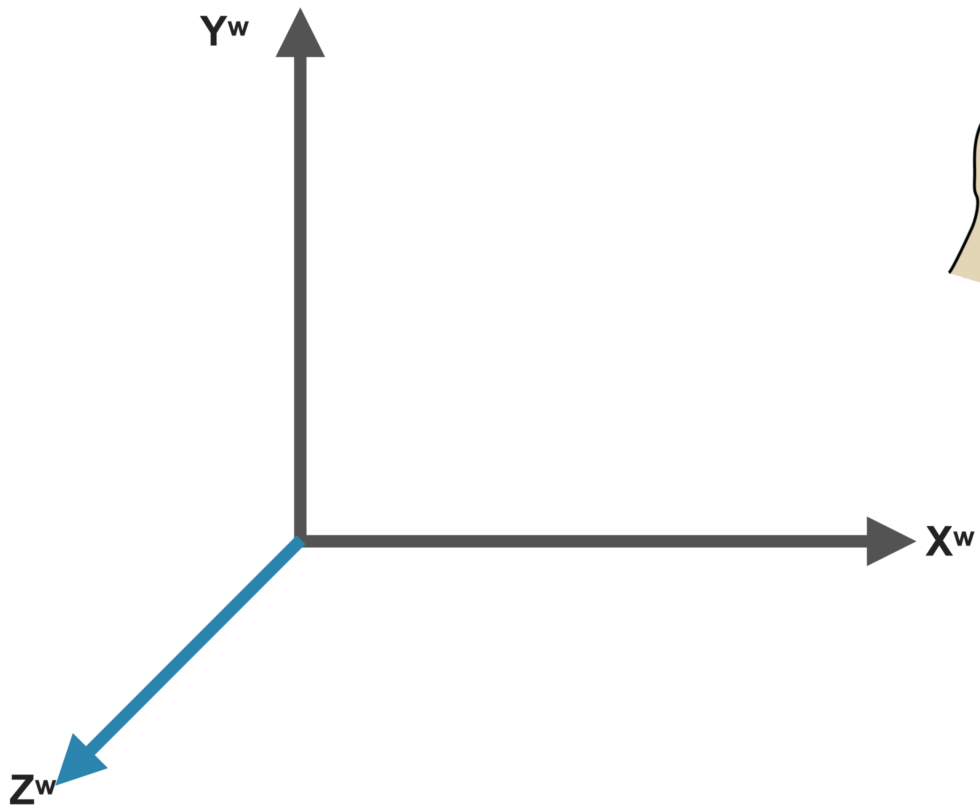


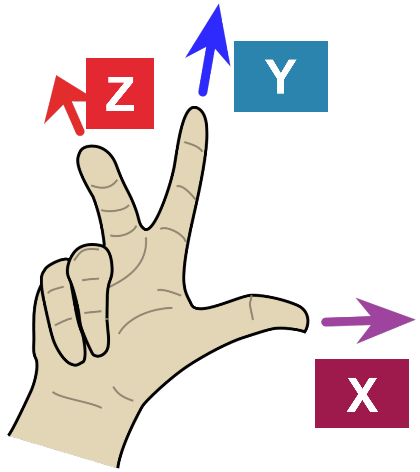
3D COORDINATE FRAME



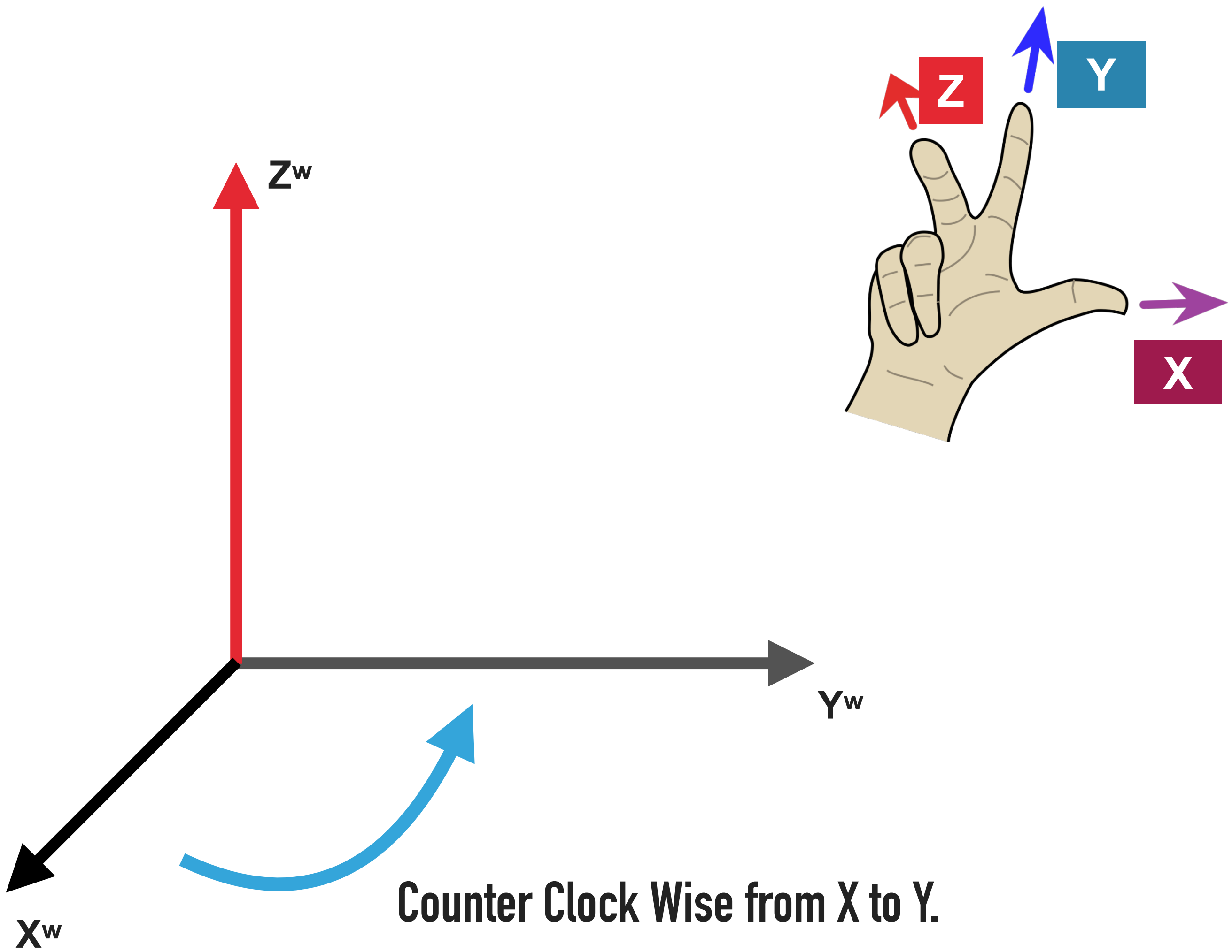


3D Coordinate Frame





3D COORDINATE FRAME



Robotics & Internet-of-Things

ROS

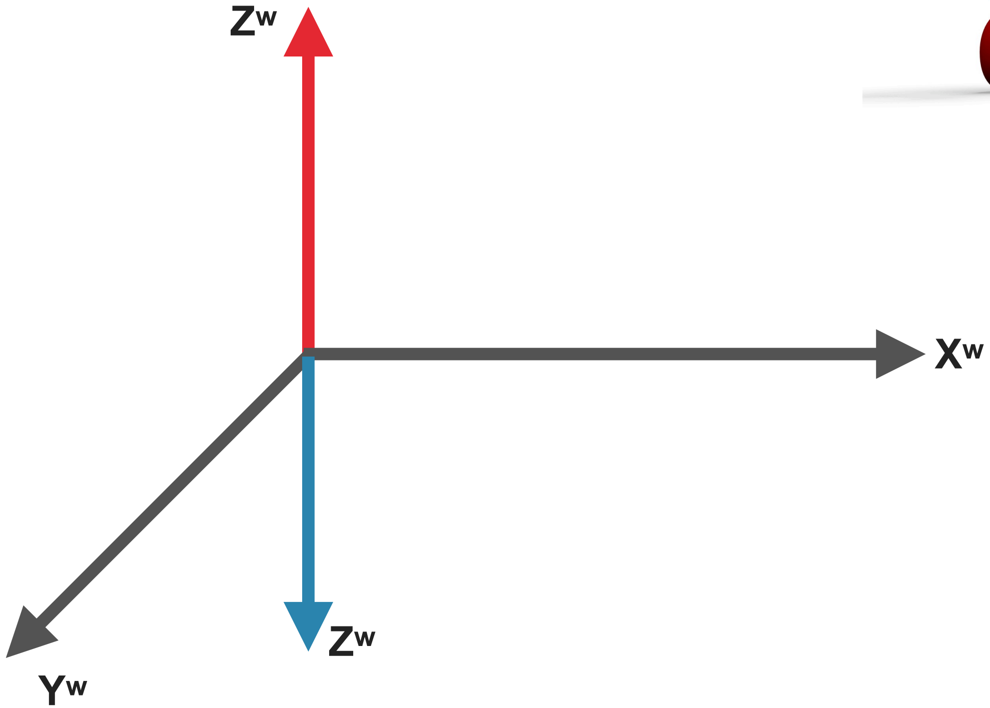
Udemy Course
Robot Operating System (Part II)
Navigation and SLAM

Prof. Anis Koubaa

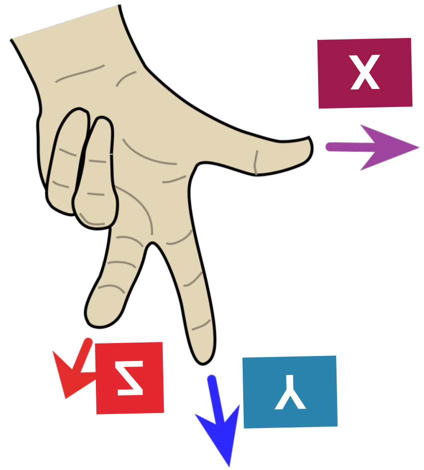
Right-Hand Rule: Exercise

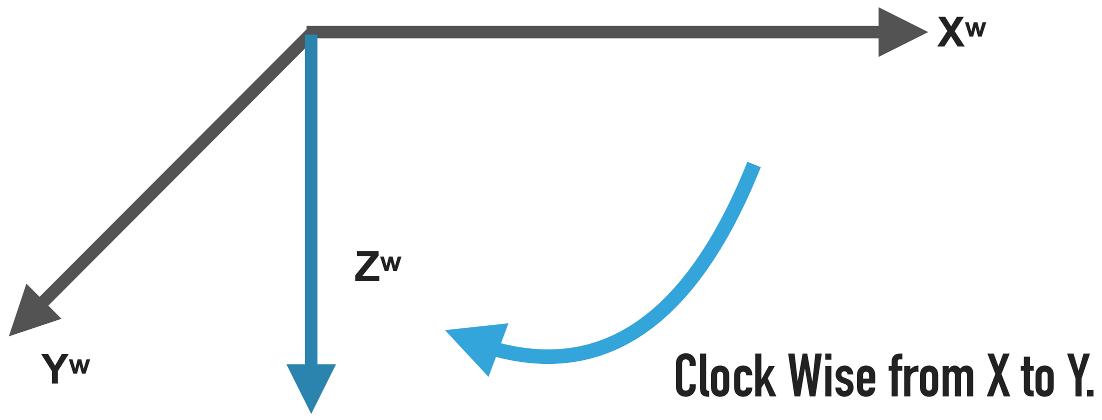
https://www.udemy.com/user/anis-koubaa/

Right-Hand Rule



RIGHT-HAND RULE





Robotics &
Internet-of-Things

ROS

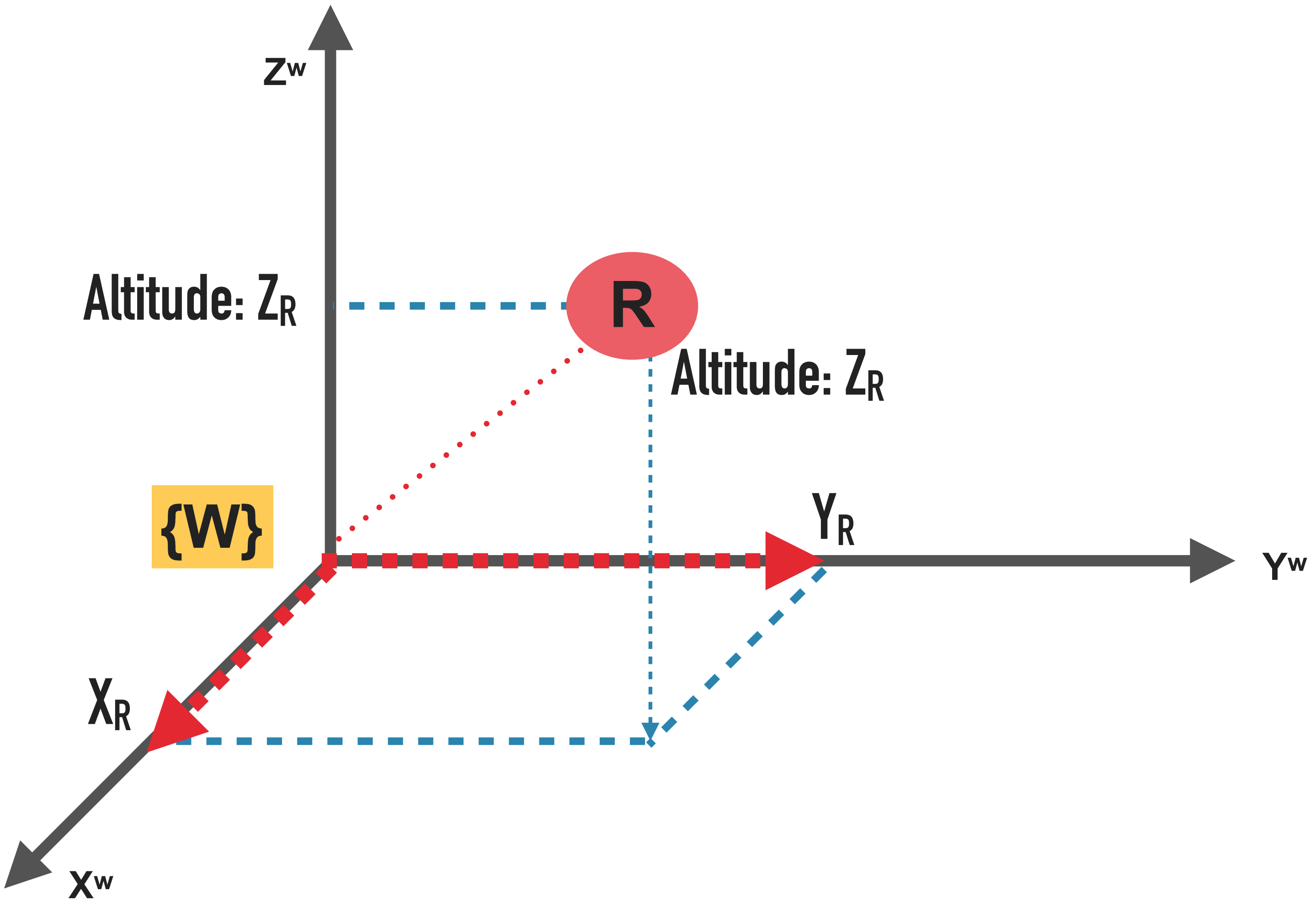
Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

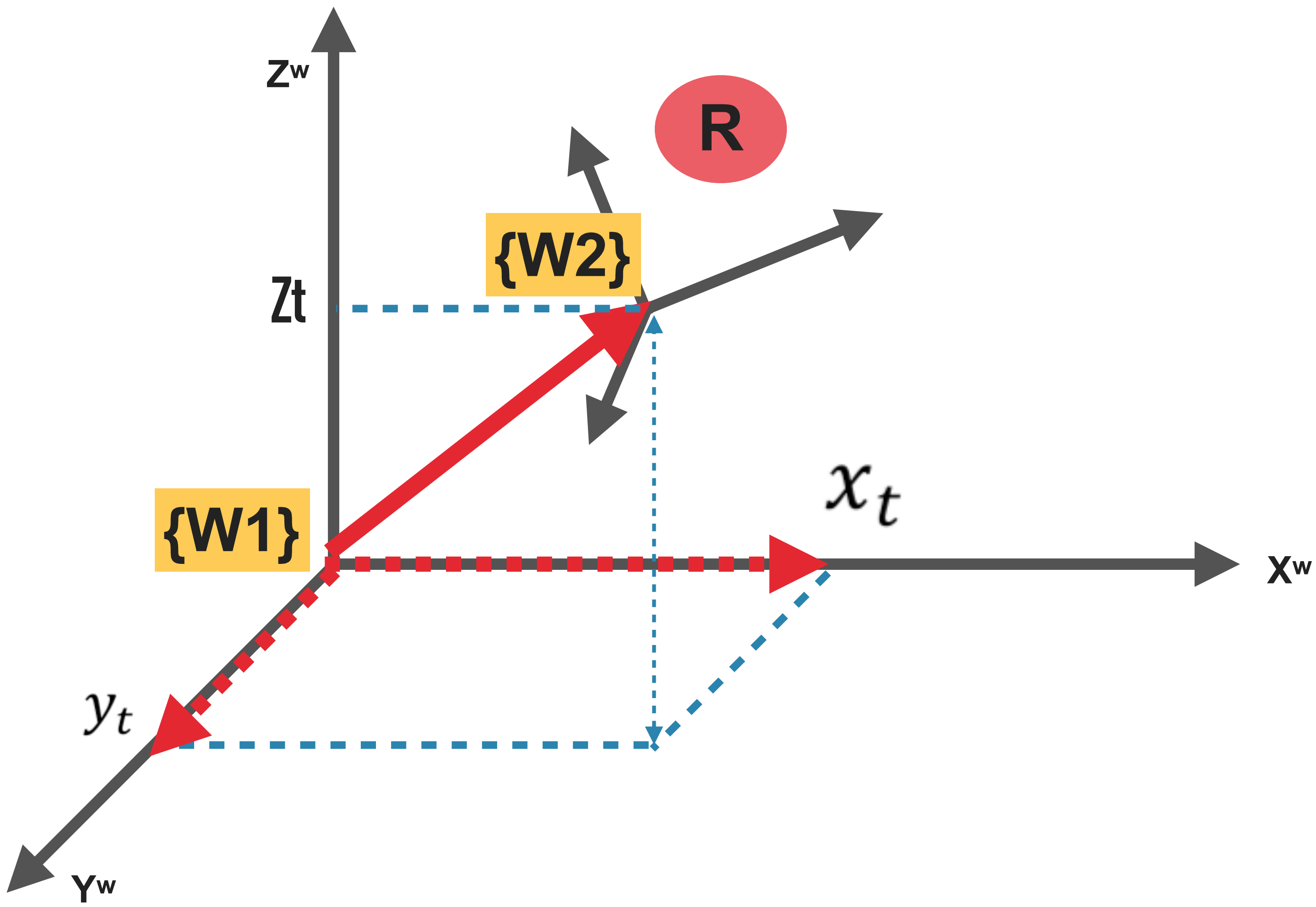
3D Transformation

https://www.udemy.com/user/anis-koubaa/

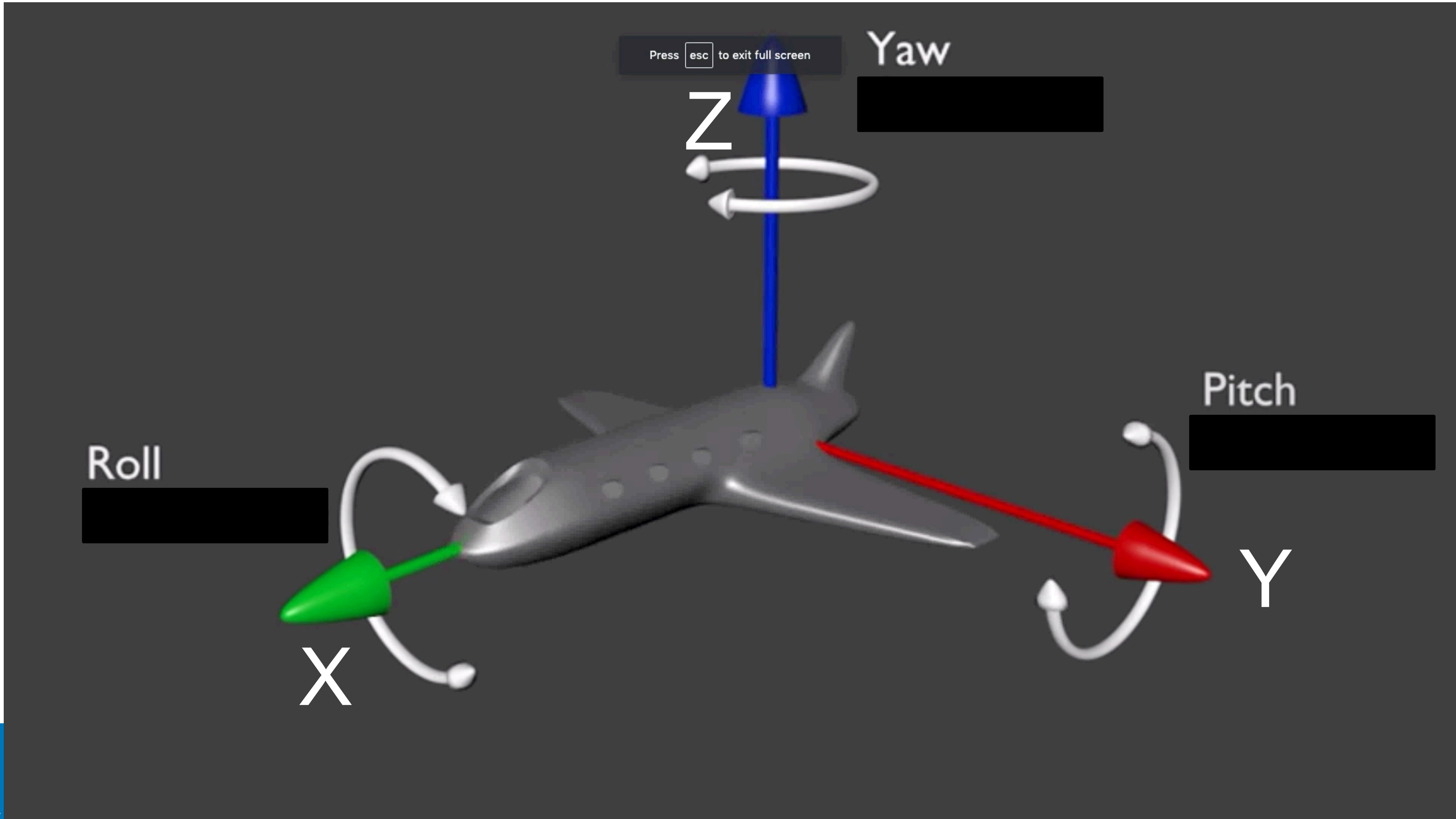
3D Coordinate Frame



3D Coordinate Frame

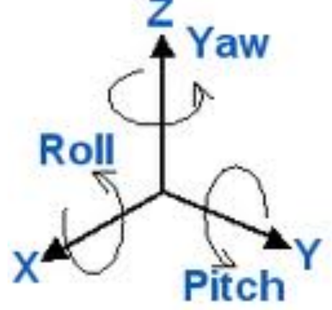


ROLL-PITCH-YAW (X,Y,Z)

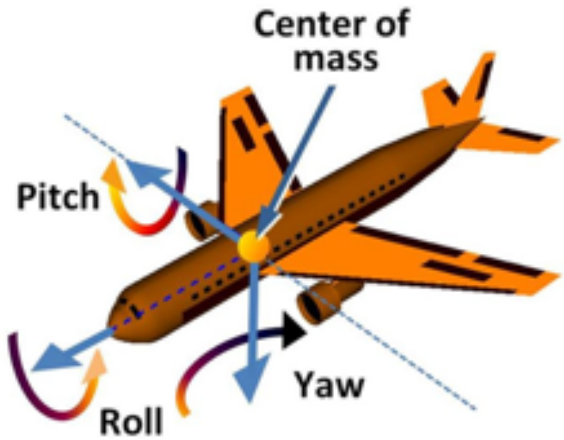


ROLL-PITCH-YAW (X,Y,Z)

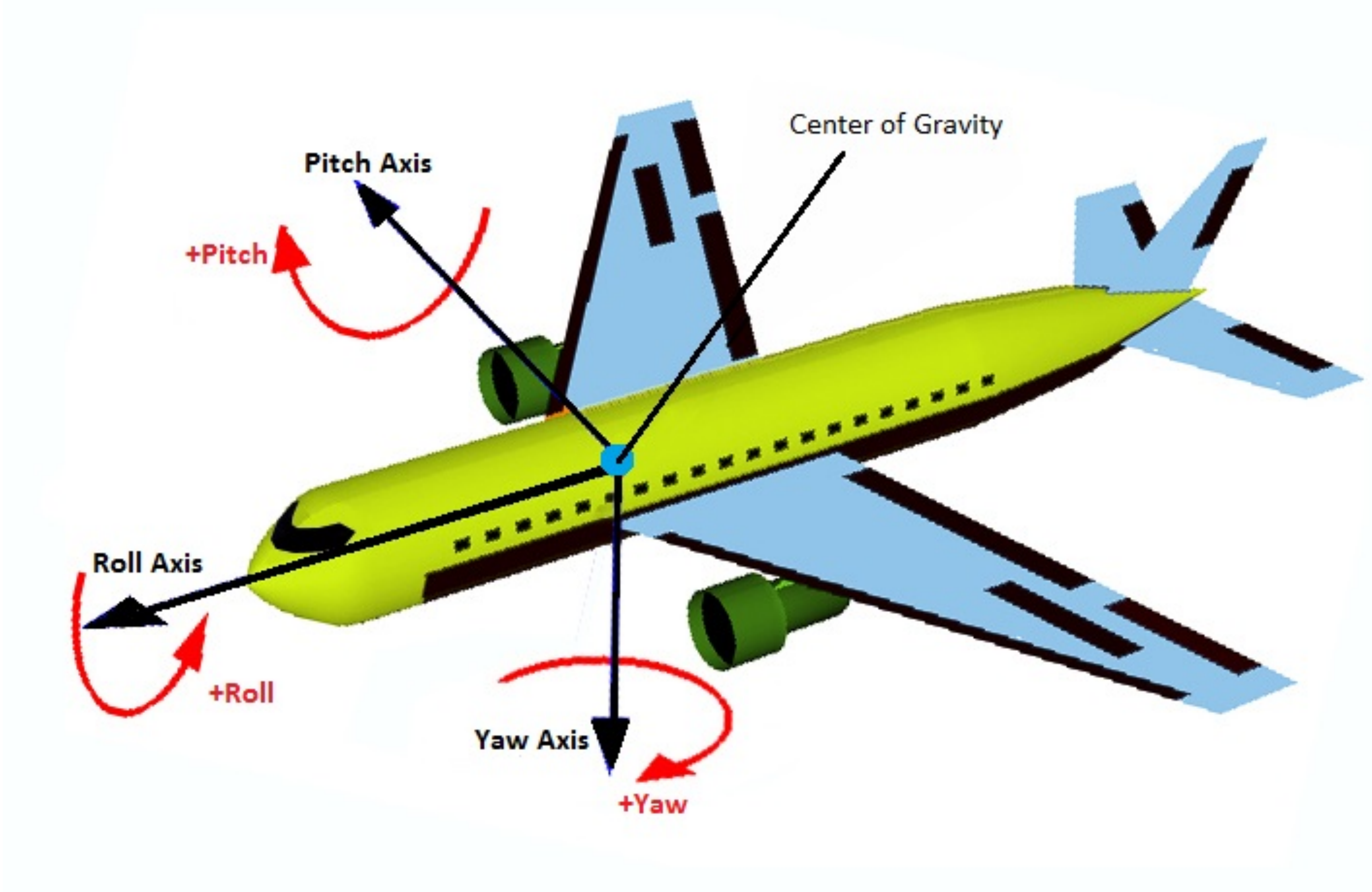
For aerospace and ground vehicles



- **Roll:** bank
  - Rotation on x-axis
- **Pitch:** attitude
  - Rotation on y-axis
- **Yaw:** heading
  - Rotation on z-axis



COORDINATE FRAME



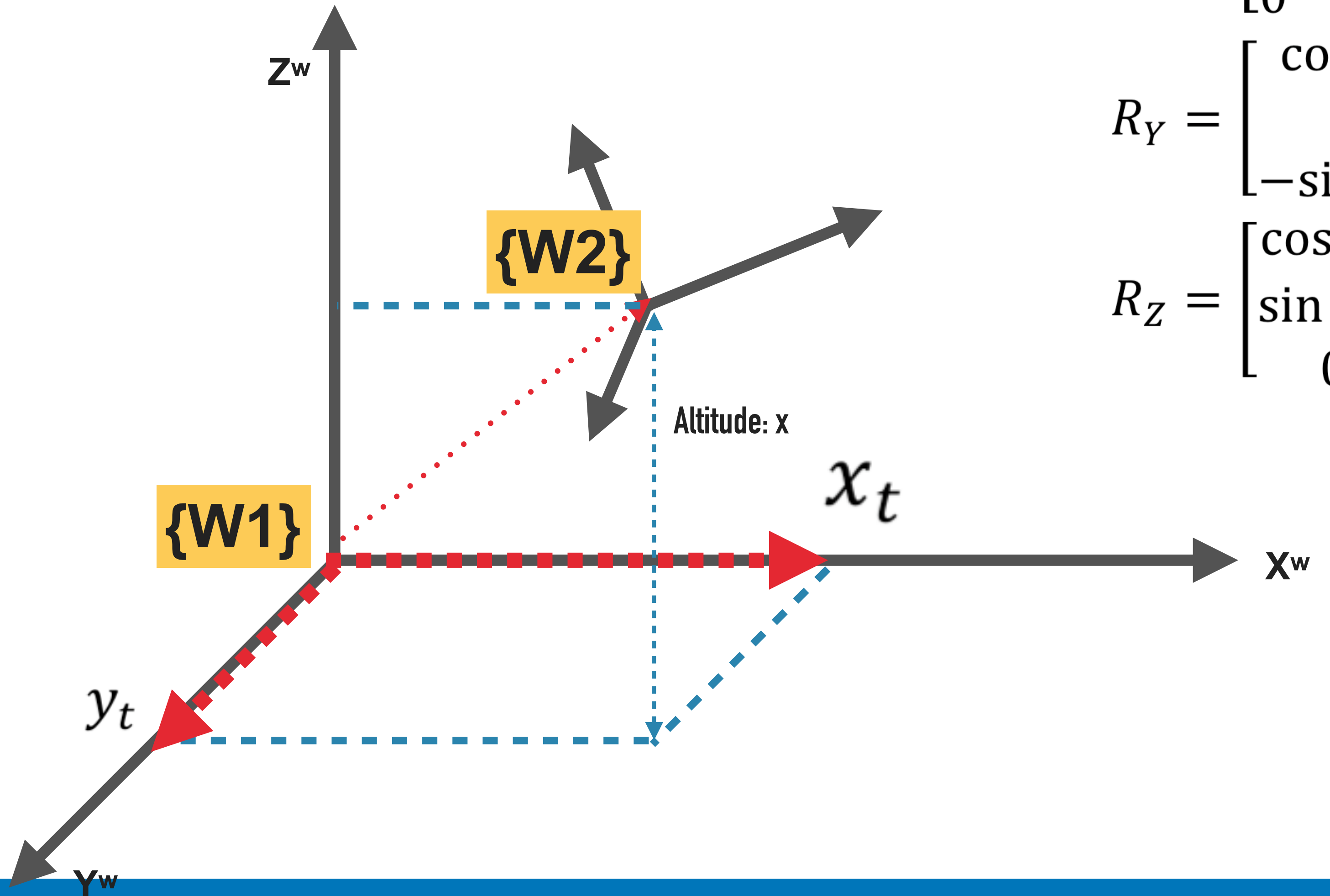
http://elektromot.com/how-to-control-an-aircraft/

3D Rotation Transformation

$$R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$R_Y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 - \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

$$R_Z = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



General Rotation Matrix

- The general rotation matrix can be obtained by a matrix multiplication
- For example, if we perform a rotation around the Z-axis with angle APLHA, then around the Y-axis with angle BETA and then around the X-axis with the angle GAMMA, the resulting rotation matrix will be.

$$R = R_z(\alpha)R_y(\beta)R_x(\gamma)$$

Coordinate Frame 3D Transformation

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} r11 & r12 & r13 & x_t \\ r21 & r22 & r23 & y \\ r31 & r32 & r33 & z_t \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

Coordinate Frame 3D Transformation

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{W1}R_{W2} & t \\ 0_{1x3} & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

Robotics &
Internet-of-Things

ROS

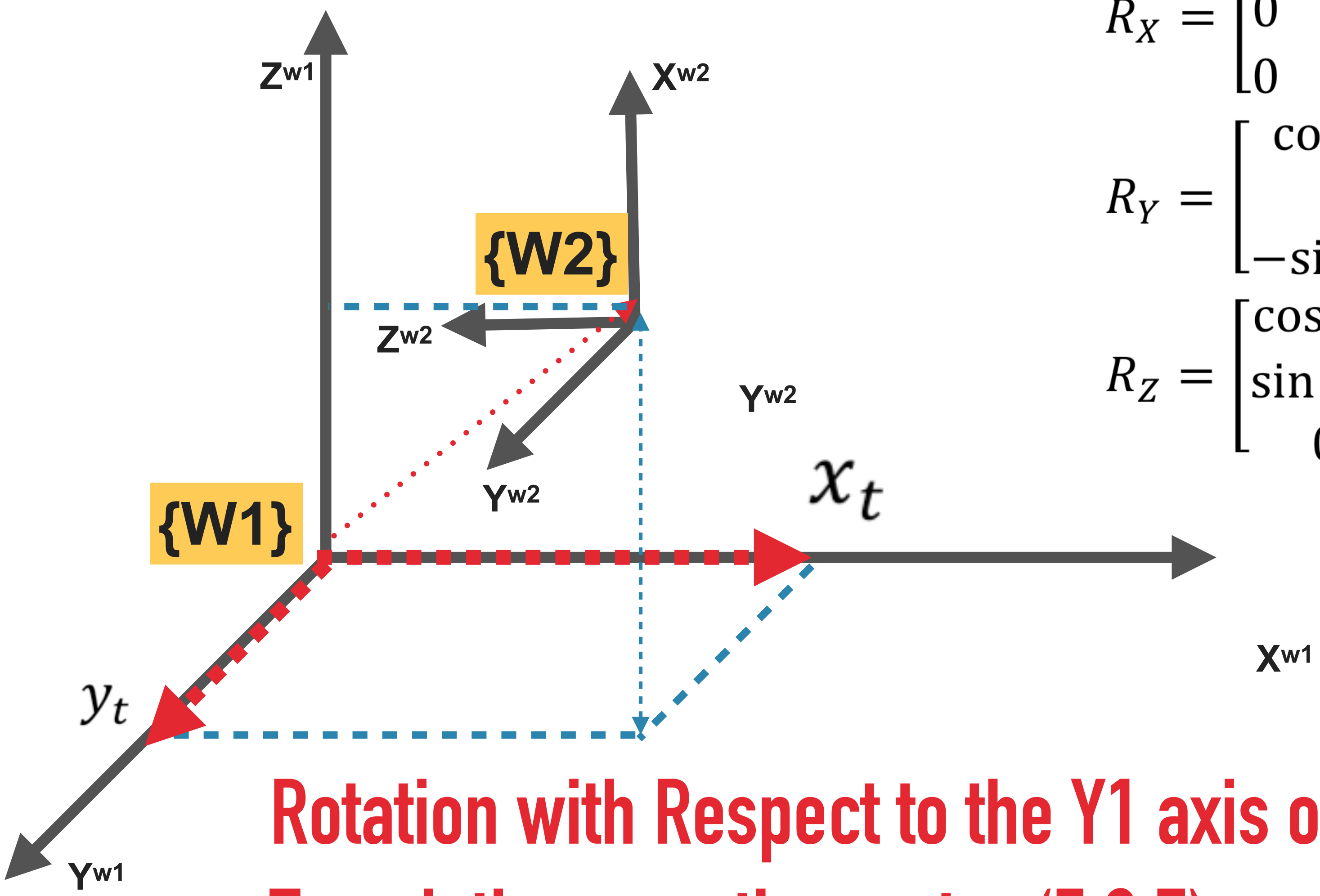
Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

3D Transformation: Application

https://www.udemy.com/user/anis-koubaa/

ROTATION EXAMPLE



$$R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$R_Y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0\,- \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

$$R_Z = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation with Respect to the Y1 axis of 90 degrees

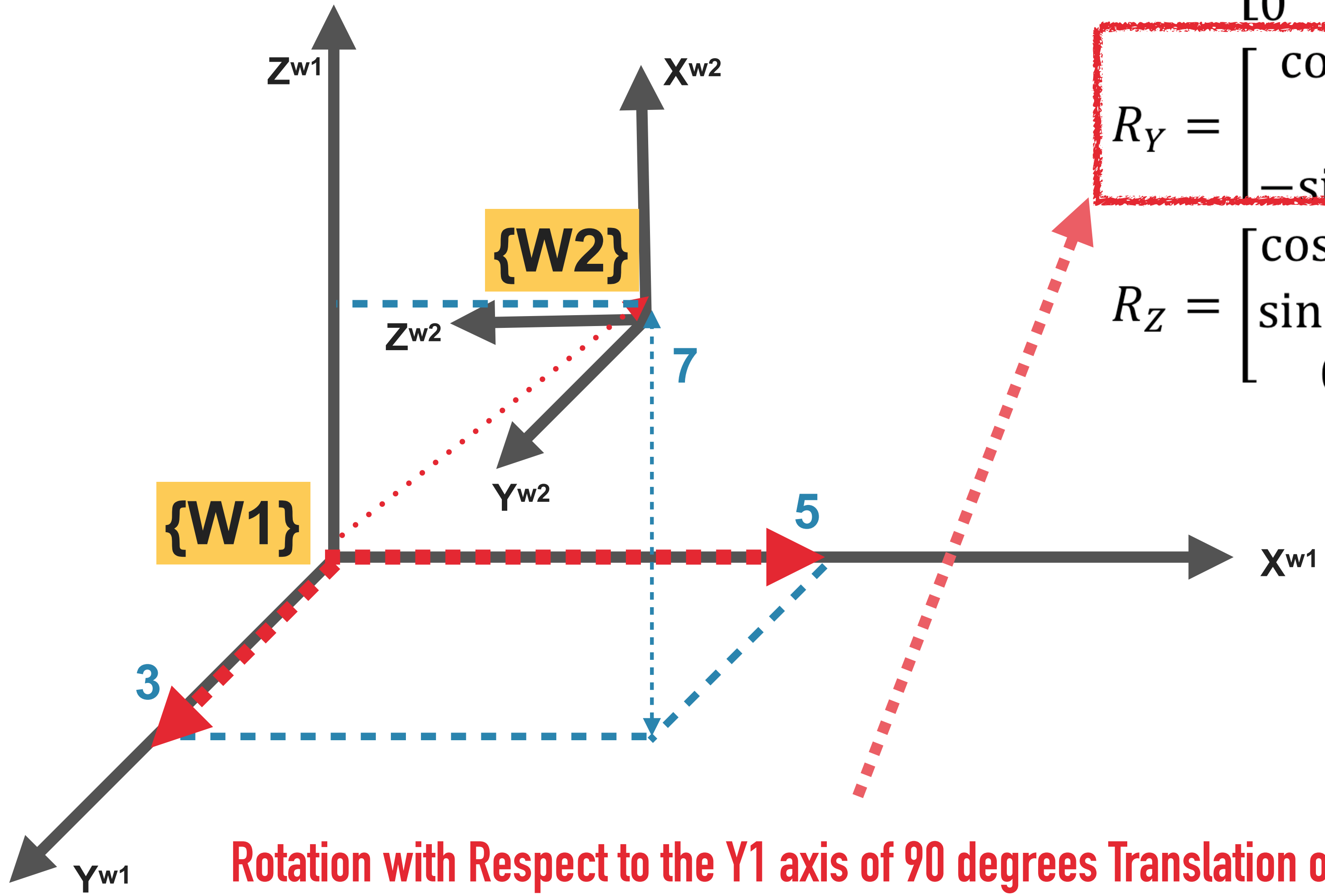
Translation over the vector (5,3,7)

3D Transformation Example

$$R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$\boxed{R_Y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 - \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}}$$

$$R_Z = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Rotation with Respect to the Y1 axis of 90 degrees Translation over the vector (5,3,7)

Coordinate Frame 3D Transformation

$$R = R_y(\theta)$$

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(90) & 0 & \sin(90) & 5 \\ 0 & 0 & 0 & 3 \\ -\sin(90) & 0 & \cos(90) & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 3 \\ -1 & 0 & 0 & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

Coordinate Frame 3D Transformation

$$R = R_y(\theta)$$

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(90) & 0 & \sin(90) & 5 \\ 0 & 0 & 0 & 3 \\ -\sin(90) & 0 & \cos(90) & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 3 \\ -1 & 0 & 0 & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

Coordinate Frame 3D Transformation

Rotation Matrix **Translation Vector**

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(90) & 0 & \sin(90) & 5 \\ 0 & 0 & 0 & 3 \\ -\sin(90) & 0 & \cos(90) & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} {}^{W1}x \\ {}^{W1}y \\ {}^{W1}z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 3 \\ -1 & 0 & 0 & 7 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} {}^{W2}x \\ {}^{W2}y \\ {}^{W2}z \\ 1 \end{bmatrix}$$

Robotics & Internet-of-Things

ROS

Udemy Course Robot Operating System (Part II) Navigation and SLAM

Prof. Anis Koubaa

3D Orientation Models

https://www.udemy.com/user/anis-koubaa/

Rotation Representation Methods

- Three-Angle Representation
  - Euler Rotation vs. Cardan Rotation
- Rotation about Arbitrary Vector
- Quaternions

Robotics & Internet-of-Things

ROS

Udemy Course
Robot Operating System (Part II) Navigation and SLAM

Prof. Anis Koubaa

Three-Angle Representation

https://www.udemy.com/user/anis-koubaa/

Three-Angle Representation: Euler Rotation

- **Euler Rotation Sequence:** involves repetition, but not successive, of rotations about one particular axis
- **Cardan Rotation Sequence**: characterized by rotations about all three axes

XYX, YXY, YZY

ZYZ, XZX, XZX

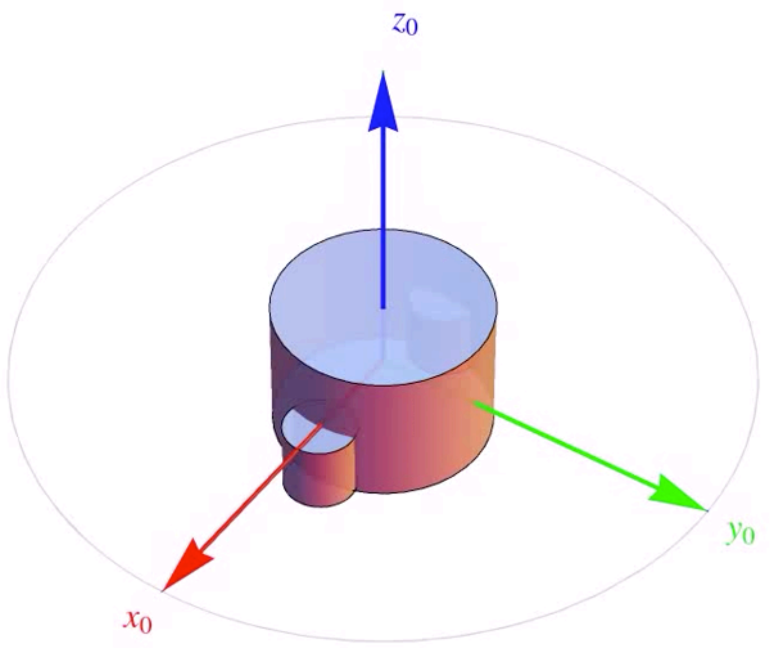
XYZ, XZY, YXZ

YZX, ZYX, ZXY

THREE-ANGLE REPRESENTATION: EULER ROTATION

- Any **two** independent **orthonormal coordinate frames** can be related by a sequence of rotations (not more than three) about coordinate axes, where **no two successive** rotations may be about the same axis.

Example of Euler Rotation: Z, X, Z



Robotics & Internet-of-Things

ROS

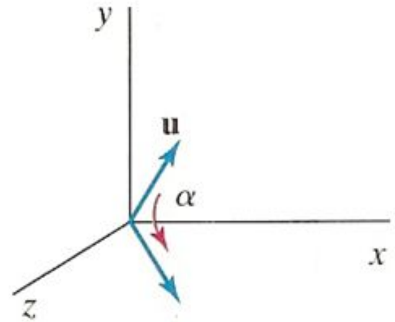
Udemy Course
Robot Operating System (Part II) Navigation and SLAM

Prof. Anis Koubaa

Rotation about Arbitrary Vector

https://www.udemy.com/user/anis-koubaa/

EULER ROTATION THEOREM



- In 3D space, orientation can be expressed as a rotation about some axis that runs through a fixed point on the rigid body.
- This means, for any rotation, there exists some axis at certain angle around which the rotation occurs.
- Rodrigues Formula

$$R = I\cos\theta + \sin\theta[\mathbf{u}]_{\times} + (1-\cos\theta)\mathbf{u}\otimes\mathbf{u},$$

EULER ROTATION THEOREM

- Given a unit vector u=(ux, uy, uz), the matrix of rotation by an angle THETA about the axis of direction u is

$$R = \begin{bmatrix} \cos\theta + u_x^2\left(1-\cos\theta\right) & u_x u_y\left(1-\cos\theta\right) - u_z\sin\theta & u_x u_z\left(1-\cos\theta\right) + u_y\sin\theta \\ u_y u_x\left(1-\cos\theta\right) + u_z\sin\theta & \cos\theta + u_y^2\left(1-\cos\theta\right) & u_y u_z\left(1-\cos\theta\right) - u_x\sin\theta \\ u_z u_x\left(1-\cos\theta\right) - u_y\sin\theta & u_z u_y\left(1-\cos\theta\right) + u_x\sin\theta & \cos\theta + u_z^2\left(1-\cos\theta\right) \end{bmatrix}.$$

Udemy Course
Robot Operating System
(Part II)
Navigation and SLAM

Prof. Anis Koubaa

Quaternion

https://www.udemy.com/user/anis-koubaa/

QUATERNION

- Quaternion are another way of representing rotation
- It is written as a scalar and a vector

$$q = s\langle v_1 | v_2 | v_2 \rangle$$

$$q = s + v$$

$$q = s + v_1 i + v_2 j + v_3 k$$

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

$$\begin{cases} q_0 = q_w = s \\ q_1 = q_x = v_1 \\ q_2 = q_y = v_2 \\ q_3 = q_z = v_3 \end{cases}$$

In ROS: the notation used is

(x,y,z,w)

Equivalent Rotation Matrix

- The rotation matrix corresponding to a clockwise/left-handed rotation by the unit quaternion axis

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

https://www.euclideanspace.com/maths/geometry/rotations/conversions/quaternionToMatrix/index.htm

EULER ANGLES TO QUATERNION CONVERSION

- We can also obtain the quaternion from Euler angles using the following conversion:

$$
\begin{aligned}
\mathbf{q}_{IB} &= \begin{bmatrix} \cos(\psi/2) \\ 0 \\ 0 \\ \sin(\psi/2) \end{bmatrix} \begin{bmatrix} \cos(\theta/2) \\ 0 \\ \sin(\theta/2) \\ 0 \end{bmatrix} \begin{bmatrix} \cos(\phi/2) \\ \sin(\phi/2) \\ 0 \\ 0 \end{bmatrix} \\
&= \begin{array}{cl} \left[\begin{matrix} \cos(\phi/2)\cos(\theta/2)\cos(\psi/2) - \sin(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \sin(\phi/2)\cos(\theta/2)\cos(\psi/2) + \cos(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\sin(\theta/2)\cos(\psi/2) - \sin(\phi/2)\cos(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\cos(\theta/2)\sin(\psi/2) + \sin(\phi/2)\sin(\theta/2)\cos(\psi/2) \end{matrix}\right] & \begin{matrix} \mathrm{S{=}Q0} \\ \mathrm{V1} \\ \mathrm{V2} \\ \mathrm{V3} \end{matrix} \end{array}
\end{aligned}
$$

Quaternion to Euler Angles Conversion

- The Euler angles can be obtained from the quaternions via the relations

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \operatorname{atan2}(2(q_0 q_1 + q_2 q_3), 1 - 2(q_1^2 + q_2^2)) \\ \operatorname{asin}(2(q_0 q_2 - q_3 q_1)) \\ \operatorname{atan2}(2(q_0 q_3 + q_1 q_2), 1 - 2(q_2^2 + q_3^2)) \end{bmatrix}$$

Quaternion to Euler Angles Conversion

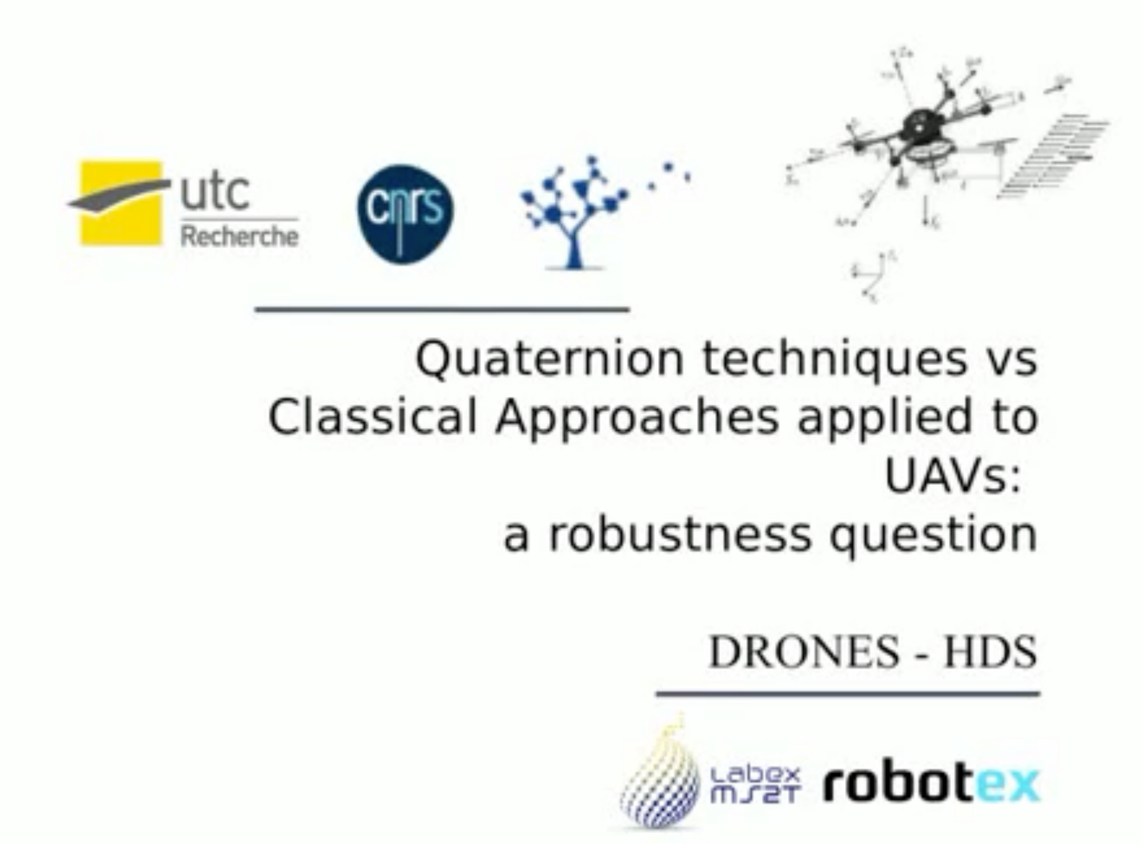
- Example of Python Code

```
import math

def quaternion_to_euler_angle(w, x, y, z):

    t0 = +2.0 * (w * x + y * z)
    t1 = +1.0 - 2.0 * (x * x + y * y)
    X = math.degrees(math.atan2(t0, t1))

    t2 = +2.0 * (w * y - z * x)
    t2 = +1.0 if t2 > +1.0 else t2
    t2 = -1.0 if t2 < -1.0 else t2
    Y = math.degrees(math.asin(t2))

    t3 = +2.0 * (w * z + x * y)
    t4 = +1.0 - 2.0 * (y * y + z * z)
    Z = math.degrees(math.atan2(t3, t4))

    return X, Y, Z
```

https://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles

WHY QUATERNION?

WHY QUATERNION?



https://www.youtube.com/watch?v=0VAc_G79POE

Quaternions Benefits

- Compared to Euler angles they are simpler to compose and avoid the problem of gimbal lock.
- Compared to rotation matrices they are more compact, more numerically stable, and more efficient.
- Quaternions have applications in computer graphics computer vision, robotics, navigation, molecular dynamics, flight dynamics, orbital mechanics of satellites and crystallographic texture analysis.

https://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation