

Codeanalyse Kalman-Filter von Rotomotion Autopilot Version 2.5

$$Q = \begin{pmatrix} 0.00015 & 0 & 0 & 0 \\ 0 & 0.00015 & 0 & 0 \\ 0 & 0 & 0.00015 & 0 \\ 0 & 0 & 0 & 0.00015 \end{pmatrix}$$

$$R = \begin{pmatrix} 0.086 & 0 & 0 \\ 0 & 0.086 & 0 \\ 0 & 0 & 0.086 \end{pmatrix}$$

$$X = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

ahrs_init(0, 0, 1, 0);

⇒ **accel2euler(euler_angles, ax, ay, az, heading);**

⇒ **= accel2euler(euler_angles, 0, 0, 1, 0);**

○ initiales setzen der Eulerwinkel phi, theta, psi

$$g = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

$$\varphi = e_0 = \arctan\left(\frac{a_y}{a_z}\right)$$

$$\theta = e_1 = \arcsin\left(\frac{a_x}{-g}\right)$$

$$\psi = e_2 = 0$$

⇒ **euler2quat:** Umrechnen der Eulerwinkel in Quaternionen

$$q_0 = \cos\left(\frac{\varphi}{2}\right) \cdot \cos\left(\frac{\theta}{2}\right) \cdot \cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\varphi}{2}\right) \cdot \sin\left(\frac{\theta}{2}\right) \cdot \sin\left(\frac{\psi}{2}\right)$$

$$q_1 = -\cos\left(\frac{\varphi}{2}\right) \cdot \sin\left(\frac{\theta}{2}\right) \cdot \sin\left(\frac{\psi}{2}\right) + \sin\left(\frac{\varphi}{2}\right) \cdot \cos\left(\frac{\theta}{2}\right) \cdot \cos\left(\frac{\psi}{2}\right)$$

$$q_2 = \cos\left(\frac{\varphi}{2}\right) \cdot \sin\left(\frac{\theta}{2}\right) \cdot \cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\varphi}{2}\right) \cdot \cos\left(\frac{\theta}{2}\right) \cdot \sin\left(\frac{\psi}{2}\right)$$

$$q_3 = \cos\left(\frac{\varphi}{2}\right) \cdot \cos\left(\frac{\theta}{2}\right) \cdot \sin\left(\frac{\psi}{2}\right) - \sin\left(\frac{\varphi}{2}\right) \cdot \sin\left(\frac{\theta}{2}\right) \cdot \cos\left(\frac{\psi}{2}\right)$$

Schleife {

ahrs_step(angles, 0.1, 0, 0, 0, 0, 0, 1, 0);

printf("Angles: %lf %lf %lf\n",

angles[0],

angles[1],

angles[2]);

}

ahrs_step(angles_out, dt, p, q, r, ax, ay, az, heading):

⇒ **rotate2omega(p, q, r):** Construct the quaternion omega matrix

$$\circ A = \begin{pmatrix} 0 & -\frac{p}{2} & -\frac{q}{2} & -\frac{r}{2} \\ \frac{p}{2} & 0 & \frac{r}{2} & -\frac{q}{2} \\ \frac{q}{2} & -\frac{r}{2} & 0 & \frac{p}{2} \\ \frac{r}{2} & \frac{q}{2} & -\frac{p}{2} & 0 \end{pmatrix}$$

$$\Rightarrow \dot{X} = AX$$

$$\Rightarrow X = X + \dot{X} \cdot dt$$

$$\Rightarrow AT = \text{transpose}(A)$$

$$\Rightarrow P = A \cdot P \cdot AT + Q$$

$$\Rightarrow \mathbf{C} = \text{compute_C}(X):$$

C is the measurement matrix that has the measured state and the estimated state.

$$C = \begin{bmatrix} [dphi/dq0 \ dhpi/dq1 \ \dots] \\ [dtheta/qd0 \ dtheta/dq1 \ \dots] \\ [dpsi/dq0 \ dpsi/dq1 \ \dots] \\] \end{bmatrix}$$

$$\circ \mathbf{D} = \text{quat2dcm}(X)$$

This will construct a direction cosine matrix from quaternions in the standard rotation sequence [phi][theta][psi]

$$D = \begin{pmatrix} 1 - 2(x_2^2 + x_3^2) & 2(x_1x_2 + x_0x_3) & 2(x_1x_3 - x_0x_2) \\ 2(x_1x_2 - x_0x_3) & 1 - 2(x_1^2 + x_3^2) & 2(x_2x_3 + x_0x_1) \\ 2(x_1x_3 + x_0x_2) & 2(x_2x_3 - x_0x_1) & 1 - 2(x_1^2 + x_2^2) \end{pmatrix}$$

$$\circ \mathbf{F} = \text{compute_f}(D)$$

F is a temp vector of d(arcsin(X)) to reduce the complexity of the C matrix calculations

$$F = \begin{pmatrix} \frac{2}{d_{2,2}^2 + d_{1,2}^2} \\ -1 \\ \sqrt{1 - d_{0,2}^2} \\ \frac{2}{d_{0,0}^2 + d_{0,1}^2} \end{pmatrix}$$

$$\circ \text{Compute the estimated state}$$

$$C = \begin{pmatrix} f_0 x_1 d_{2,2} & f_0(x_0 d_{2,2} + 2x_1 d_{1,2}) & f_0(x_0 d_{2,2} + 2x_1 d_{1,2}) & f_0 x_2 d_{2,2} \\ -2f_1 x_2 & 2f_1 x_3 & -2f_1 x_0 & 2f_1 x_1 \\ f_2 x_3 d_{0,0} & f_2 x_2 d_{0,0} & f_2(x_1 d_{0,0} + 2x_2 d_{0,1}) & f_2(x_0 d_{0,0} + 2x_3 d_{0,1}) \end{pmatrix}$$

$$\Rightarrow \mathbf{X}_m = \text{accel2euler}(ax, ay, az, \text{heading}):$$

$$g = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

$$\varphi = e_0 = \arctan\left(\frac{a_y}{a_z}\right)$$

$$\theta = e_1 = \arcsin\left(\frac{a_x}{-g}\right)$$

$$\psi = e_2 = \text{heading}$$

⇒ **Xe = quat2euler(X):**

$$\varphi = e_0 = \arctan\left(\frac{2(x_2 x_3 + x_0 x_1)}{1 - 2(x_1^2 + x_2^2)}\right)$$

$$\theta = e_1 = -\arcsin\left(2(x_1 x_3 - x_0 x_2)\right)$$

$$\psi = e_2 = \arctan\left(\frac{2(x_1 x_2 + x_0 x_3)}{1 - 2(x_2^2 + x_3^2)}\right)$$

⇒ **kalman_update(P, R, C, X, Xe, Xm):**

$$E = C P C' + R$$

$$K = C P' \text{inv}(E)$$

$$X = X + K(Xm - Xe)$$

$$P = P - K C P$$

⇒ **angles = quat2euler(X)** (siehe oben)