Multicopter Design and Control Practice
—— A Series Experiments Based on MATLAB and Pixhawk

Lesson 08 State Estimation and Filter Design Experiment

Quan Quan, Associate Professor, qq_buaa@buaa.edu.cn
School of Automation Science and Electrical Engineering,
BeihangUniversity, Beijing 100191, China.

北航可靠飞行控制研究组
BUAA Reliable Flight Control Group

Outline

1. Preliminary
2. Basic Experiment
3. Analysis Experiment
4. Design Experiment
5. Summary

Measurement Principle

The three-axis accelerometer is fixed to the multicopter, aligned with the aircraft-body coordinate frame. Therefore, the observation of low-frequency pitch and roll angle acquired by accelerometer measurement illustrated as

$$\theta_{\mathrm{m}} = \arcsin\left(\frac{a_{x_{\mathrm{b}}\mathrm{m}}}{g}\right)$$

$$\phi_{\mathrm{m}} = -\arcsin\left(\frac{a_{y_{\mathrm{b}}\mathrm{m}}}{g\cos\theta_{\mathrm{m}}}\right)$$

where ${}^{\mathrm{b}}\mathbf{a}_{\mathrm{m}} = [a_{x_{\mathrm{b}}\mathrm{m}} \quad a_{y_{\mathrm{b}}\mathrm{m}} \quad a_{z_{\mathrm{b}}\mathrm{m}}]^{\mathrm{T}}$ denotes the measurement from the accelerometer.

Measurement Principle

Several further considerations are as follows

(1) It is better to eliminate the slow time-varying drift of the accelerometer to obtain a more accurate angle.

(2) If the amplitude of the vibration is large, $a_{x_{\mathrm{b}}\mathrm{m}}, a_{y_{\mathrm{b}}\mathrm{m}}$ are polluted by noise severely and further affect the estimation of $\theta_{\mathrm{m}}, \phi_{\mathrm{m}}$. Thus, the vibration damping is very important. Additionally, the attitude rates $\dot{\theta}, \dot{\phi}, \dot{\psi}$ and angular velocity ${}^{\mathrm{b}}\boldsymbol{\omega}$ exhibit the following relationship

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \tan\theta\sin\phi & \tan\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} \omega_{x_{\mathrm{b}}} \\ \omega_{y_{\mathrm{b}}} \\ \omega_{z_{\mathrm{b}}} \end{bmatrix}$$

Multicopters typically work under condition that θ, ϕ are small, and thus the above equation is approximated as follows.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \approx \begin{bmatrix} \omega_{x_{\mathrm{b}}} \\ \omega_{y_{\mathrm{b}}} \\ \omega_{z_{\mathrm{b}}} \end{bmatrix}$$

According to the working principle, the attitude is estimated by the accelerometers and magnetometers with large noise but small drifts.

Preliminary

Linear Complementary Filter

Take pitch angle as an example to deduce the linear complementary filtering in detail. The Laplace transform of pitch angle θ is expressed as follows

$$\theta(s)=\frac{1}{\tau s+1}\theta(s)+\frac{\tau s}{\tau s+1}\theta(s)$$

A low-pass filter, $\tau \in \mathbb{R}_+$ denotes a time constant

A high-pass filter $\frac{\tau s}{\tau s+1}=1-\frac{1}{\tau s+1}$

θ denotes the true value of the pitch angle.

1) Given that the pitch angle obtained by an accelerometer has high noise but a low drift; for simplicity, it is modeled as follows

$$\theta_{\mathrm{m}}=\theta+n_\theta$$

where n_θ denotes high-frequency noise, and θ denotes the true pitch angle.

2) Given that the pitch angle estimated by integrating angular velocity exhibits a little noise but a large drift, and the integration is modeled as follows

The true value of the pitch angle

$$\frac{\omega_{y_b\mathrm{m}}(s)}{s}=\theta(s)+c\frac{1}{s}$$

The Laplace transform of the integration of angular velocity

The Laplace transform of the constant drift

Linear Complementary Filter

Take pitch angle as an example to deduce the linear complementary filtering in detail. The Laplace transform of pitch angle θ is expressed as follows

$$\theta(s)=\frac{1}{\tau s+1}\theta(s)+\frac{\tau s}{\tau s+1}\theta(s)$$

A low-pass filter, $\tau\in\mathbb{R}_{+}$ denotes a time constant

A high-pass filter $\frac{\tau s}{\tau s+1}=1-\frac{1}{\tau s+1}$

θ denotes the true value of the pitch angle.

The pitch angle, the standard form of a linear complementary filter is expressed as follows

$$\hat{\theta}(s)=\frac{1}{\tau s+1}\theta_{\mathrm{m}}(s)+\frac{\tau s}{\tau s+1}\left(\frac{1}{s}\omega_{y_{\mathrm{b}}\mathrm{m}}(s)\right)$$

The true value of the pitch angle by accelerometer

The integration of angular velocity by gyroscope

Linear Complementary Filter

Take pitch angle as an example to deduce the linear complementary filtering in detail. The Laplace transform of pitch angle θ is expressed as follows

$$\theta(s)=\frac{1}{\tau s+1}\theta(s)+\frac{\tau s}{\tau s+1}\theta(s)$$

A low-pass filter, $\tau\in\mathbb{R}_+$ denotes a time constant

A high-pass filter $\frac{\tau s}{\tau s+1}=1-\frac{1}{\tau s+1}$

θ denotes the true value of the pitch angle.

The pitch angle, the standard form of a linear complementary filter is expressed as follows

$$\hat{\theta}(s)=\frac{1}{\tau s+1}\theta_{\mathrm{m}}(s)+\frac{\tau s}{\tau s+1}\left(\frac{1}{s}\omega_{y_{\mathrm{b}}\mathrm{m}}(s)\right)$$

The true value of the pitch angle by accelerometer

The integration of angular velocity by gyroscope

$$\Rightarrow \quad \hat{\theta}(s)=\theta(s)+\boxed{\left[\frac{1}{\tau s+1}n_{\theta}(s)+\frac{\tau s}{\tau s+1}c\frac{1}{s}\right]} \quad \Rightarrow \quad \hat{\theta}(s)\approx\theta(s)$$

be attenuated near zero

□ Linear Complementary Filter

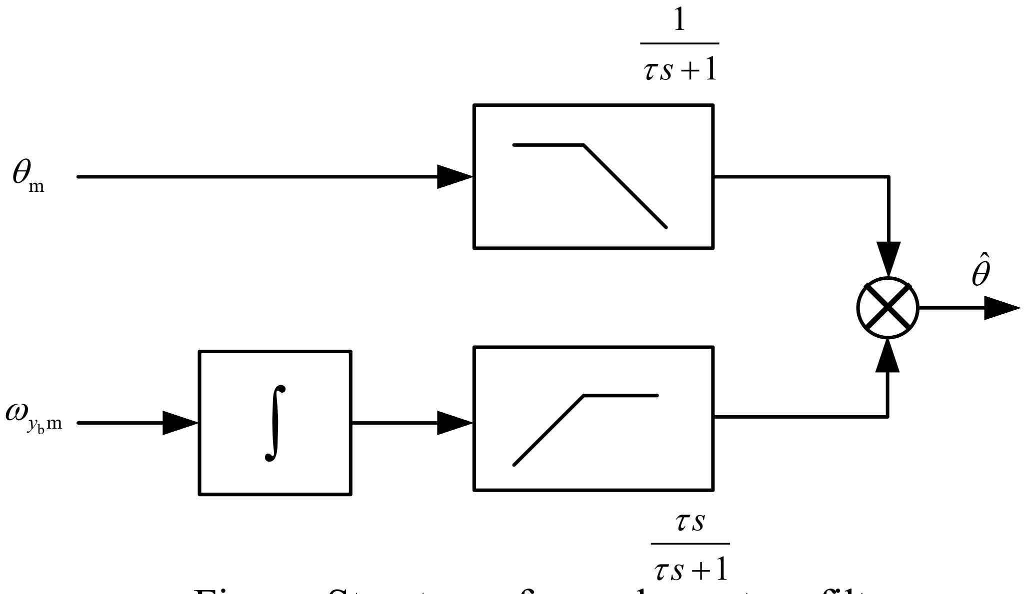


Figure. Structure of complementary filter

During the process, the low-frequency filter exhibits the advantage that θ_{m} has a small drift; while the high-pass filter maintains the advantage that $\omega_{y_b\mathrm{m}}(s)/s$ has a little noise.

In order to realize the filter with digital computers, the filter should be transformed into a discrete-time differential form

$$\hat{\theta}(s)=\frac{1}{\tau s+1}\theta_{\mathrm{m}}(s)+\frac{\tau s}{\tau s+1}\left(\frac{1}{s}\omega_{y_b\mathrm{m}}(s)\right)$$

Through the first-order backward difference, s is expressed as

$$s=\left(1-z^{-1}\right)/T_{\mathrm{s}}$$

$T_{\mathrm{s}}\in\mathbb{R}_{+}$ denotes the sampling period

Further

$$\hat{\theta}(z)=\frac{1}{\tau\frac{1-z^{-1}}{T_{\mathrm{s}}}+1}\theta_{\mathrm{m}}(z)+\frac{\tau}{\tau\frac{1-z^{-1}}{T_{\mathrm{s}}}+1}\omega_{y_b\mathrm{m}}(z)$$

The above equation is further transformed into a discrete-time difference form as follows

$$\hat{\theta}(k)=\frac{\tau}{\tau+T_s}(\hat{\theta}(k-1)+T_s\omega_{y_bm}(k))+\frac{T_s}{\tau+T_s}\theta_m(k)$$

Preliminary

Kalman Filter

The"truth"model for discrete-time cases is given as follows

$$\mathbf{x}_k = \mathbf{\Phi}_{k,k-1}\mathbf{x}_{k-1} + \mathbf{u}_{k-1} + \mathbf{\Gamma}_{k,k-1}\mathbf{w}_{k-1}$$

$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k$$

Where $\mathbf{w}_{k-1}$ and $\mathbf{v}_k$ are assumed as zero-mean Gaussian whitenoise processes. This means that the errors are uncorrelated forward or backward in time such that

Autocorrelation coefficient	$\mathbf{R}_{\mathrm{ww}}$
Correlation coefficient	$\mathbf{R}_{\mathrm{wv}}$
the system noise covariance matrix	$\mathbf{Q}_k \geqslant \mathbf{0}_{n\times n}$
the measurement noise covariance matrix	$\mathbf{R}_k > \mathbf{0}_{m\times m}$

$$\delta_{kj} = \begin{cases} 1, & k = j \\ 0, & k \neq j \end{cases}$$

$$\mathrm{E}\left(\mathbf{w}_{k-1}\right) = \mathbf{0}_{n\times 1}, \mathrm{E}\left(\mathbf{v}_k\right) = \mathbf{0}_{m\times 1}, \mathbf{R}_{\mathrm{wv}}\left(k, j\right) = \mathbf{0}_{n\times m}$$

$$\mathbf{R}_{\mathrm{ww}}\left(k, j\right) = \mathrm{E}\left(\mathbf{w}_k\mathbf{w}_j^{\mathrm{T}}\right) = \mathbf{Q}_k\delta_{kj} = \begin{cases} \mathbf{Q}_k, k = j \\ \mathbf{0}_{n\times n}, k \neq j \end{cases}$$

Uncorrelated!

$$\mathbf{R}_{\mathrm{vv}}\left(k, j\right) = \mathrm{E}\left(\mathbf{v}_k\mathbf{v}_j^{\mathrm{T}}\right) = \mathbf{R}_k\delta_{kj} = \begin{cases} \mathbf{R}_k, k = j \\ \mathbf{0}_{m\times m}, k \neq j \end{cases}$$

Preliminary

Kalman Filter

The"truth"model for discrete-time cases is given as follows

$$\mathbf{x}_k = \mathbf{\Phi}_{k,k-1}\mathbf{x}_{k-1} + \mathbf{u}_{k-1} + \mathbf{\Gamma}_{k,k-1}\mathbf{w}_{k-1}$$
$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k$$

Suppose that the initial condition of state $\mathbf{x}_0$ satisfies following expression

$$\mathrm{E}(\mathbf{x}_0) = \hat{\mathbf{x}}_0, \mathrm{cov}(\mathbf{x}_0) = \mathbf{P}_0$$

where, $\mathrm{cov}(\cdot)$ denotes covariance

Besides, $\mathbf{x}_0, \mathbf{u}_k$ and $\mathbf{w}_{k-1}, \mathbf{v}_k, k \geq 1$ are uncorrelated, and $\mathbf{w}_{k-1}$ and $\mathbf{v}_k$ are uncorrelated that the following expression is obtained

$$\mathbf{R}_{\mathbf{xw}}(0,k) = \mathrm{E}\left(\mathbf{x}_0\mathbf{w}_k^{\mathrm{T}}\right) = \mathbf{0}_{n\times n}$$
$$\mathbf{R}_{\mathbf{xv}}(0,k) = \mathrm{E}\left(\mathbf{x}_0\mathbf{v}_k^{\mathrm{T}}\right) = \mathbf{0}_{n\times m}$$
$$\mathbf{R}_{\mathbf{uw}}(k,j) = \mathrm{E}\left(\mathbf{u}_k\mathbf{w}_j^{\mathrm{T}}\right) = \mathbf{0}_{n\times n}$$

Uncorrelated!

Preliminary

Summary of the Kalman filter

1. Step1: Process model

$$\mathbf{x}_k = \mathbf{\Phi}_{k-1}\mathbf{x}_{k-1} + \mathbf{u}_{k-1} + \mathbf{\Gamma}_{k-1}\mathbf{w}_{k-1},\ \mathbf{w}_k \sim \mathcal{N}\left(\mathbf{0}_{n\times 1}, \mathbf{Q}_k\right)$$

Measurement model:

$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k,\ \mathbf{v}_k \sim \mathcal{N}\left(\mathbf{0}_{m\times 1}, \mathbf{R}_k\right)$$

2. Step2: Initial state

$$\hat{\mathbf{x}}_0 = \mathrm{E}\left(\mathbf{x}_0\right)$$

$$\mathbf{P}_0 = \mathrm{E}\left[\left(\mathbf{x}_0 - \mathrm{E}\left(\mathbf{x}_0\right)\right)\left(\mathbf{x}_0 - \mathrm{E}\left(\mathbf{x}_0\right)\right)^{\mathrm{T}}\right]$$

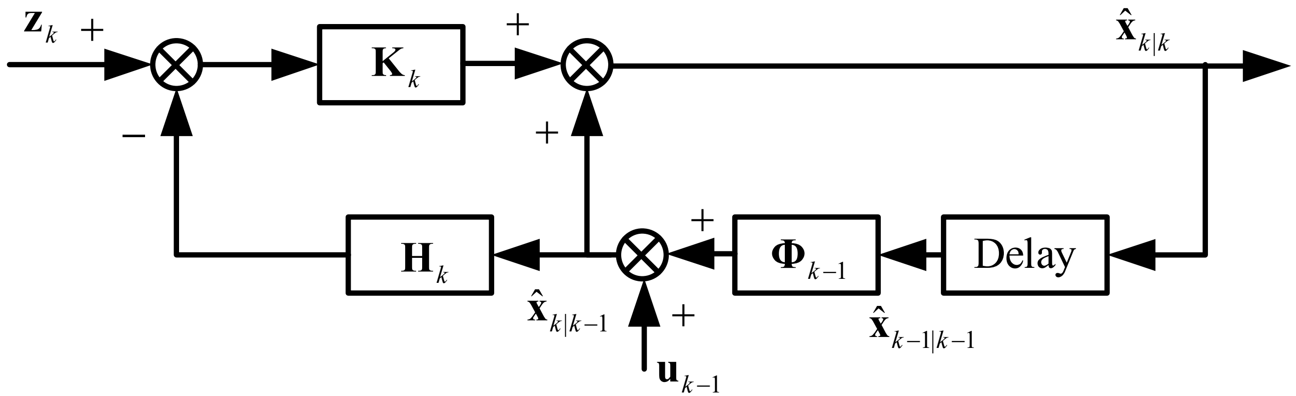


Figure. Structure of Kalman filter algorithm

Preliminary

Summary of the Kalman filter

3. Step3: For $k=0$, set

$$\mathbf{P}_{0|0}=\mathbf{P}_0,\hat{\mathbf{x}}_{0|0}=\hat{\mathbf{x}}_0$$

4. Step3: $k=k+1$

5. Step5: State estimate propagation

$$\mathbf{x}_k=\mathbf{\Phi}_{k-1}\mathbf{x}_{k-1}+\mathbf{u}_{k-1}$$

6. Step6: Error covariance propagation

$$\mathbf{P}_{k|k-1}=\mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^{\mathrm{T}}+\mathbf{\Gamma}_{k-1}\mathbf{Q}_{k-1}\mathbf{\Gamma}_{k-1}^{\mathrm{T}}$$

7. Step7: Kalman gain matrix

$$\mathbf{K}_k=\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}+\mathbf{R}_k\right)^{-1}$$

8. Step8: State estimate update

$$\hat{\mathbf{x}}_{k|k}=\hat{\mathbf{x}}_{k|k-1}+\mathbf{K}_k\left(\mathbf{z}_k-\hat{\mathbf{z}}_{k|k-1}\right)$$

where, $\hat{\mathbf{z}}_{k|k-1}=\mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$

9. Step9: Error covariance update

$$\mathbf{P}_{k|k}=\left(\mathbf{I}_n-\mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_{k|k-1}$$

10. Step10: Go back to Step4.

Preliminary

(1) It is observed that the error covariance matrix $\mathbf{P}_{k|k}$ can be obtained using the filter, which represents the estimation accuracy. Additionally, it can be used to evaluate the health of sensors.

(2) Generally speaking, if a reasonable sampling time is adopted and the continuous-time system is observable, then the corresponding discrete-time system is also observable. Conversely, the system can also lose controllability and observability when an improper sampling time is adopted. Thus, it is necessary to check the observability of the discrete system after sampling.

(3) The matrix $\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}+\mathbf{R}_k$ needs to be non-singular. Otherwise, the solution expressed by $\mathbf{K}_k=\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}+\mathbf{R}_k\right)^{-1}$ does not make sense.

(4) If the system $(\mathbf{\Phi}_{k,k-1},\mathbf{H}_k)$ is unobservable, the filter also works without causing numerical problems. Only, the unobservable mode will not be corrected. In an extreme case, the whole system is completely unobservable if $\mathbf{H}_k=\mathbf{0}_{m\times n}$. Subsequently, the filter gain $\mathbf{K}_k=\mathbf{0}_{n\times m}$. Thus, the Kalman filter degenerates as follows

$$\hat{\mathbf{x}}_{k|k}=\mathbf{\Phi}_{k,k-1}\hat{\mathbf{x}}_{k-1|k-1}+\mathbf{u}_{k-1}$$
$$\mathbf{P}_{k|k}=\mathbf{\Phi}_{k,k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k,k-1}^{\mathrm{T}}+\mathbf{\Gamma}_{k,k-1}\mathbf{Q}_{k-1}\mathbf{\Gamma}_{k,k-1}^{\mathrm{T}}$$

□ Extended Kalman Filter

The main idea of EKF denotes the linearization of nonlinear functions, which **ignores the higher order terms**. The nonlinear problem is transformed into a linear problem through Taylor expansion and first order linear truncation. the EKF is **a suboptimal filter** since the processing of linearization will cause an additional error.

Preliminary

Extended Kalman Filter

The following general nonlinear system is first considered that is described as follows

$$\mathbf{x}_k = \mathbf{f}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right)$$
$$\mathbf{z}_k = \mathbf{h}\left(\mathbf{x}_k, \mathbf{v}_k\right)$$

Where the random vector $\mathbf{w}_{k-1}$ captures uncertainties in the system model and $\mathbf{v}_k$ denotes the measurement noise, both of which are temporally uncorrelated (white noise), zero-mean random sequences with covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively.

In the derivation of the EKF, $\mathbf{f}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right)$ and $\mathbf{h}\left(\mathbf{x}_k, \mathbf{v}_k\right)$ are expanded via Taylor expansion.

By ignoring the higher-order terms, Taylor series is as follows

$$\begin{aligned}\mathbf{f}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right) &= \mathbf{f}\left(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}, \mathbf{0}_{n\times1}\right) \\ &+ \left.\frac{\partial \mathbf{f}\left(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{w}\right)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{w}=\mathbf{0}_{n\times1}} \left(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}\right) \\ &+ \left.\frac{\partial \mathbf{f}\left(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{w}\right)}{\partial \mathbf{w}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{w}=\mathbf{0}_{n\times1}} \mathbf{w}_{k-1}\end{aligned}$$

$$\begin{aligned}\mathbf{h}\left(\mathbf{x}_k, \mathbf{v}_k\right) &= \mathbf{h}\left(\hat{\mathbf{x}}_{k|k-1}, \mathbf{0}_{m\times1}\right) \\ &+ \left.\frac{\partial \mathbf{h}\left(\mathbf{x}, \mathbf{v}\right)}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{v}=\mathbf{0}_{m\times1}} \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\right) \\ &+ \left.\frac{\partial \mathbf{h}\left(\mathbf{x}, \mathbf{v}\right)}{\partial \mathbf{v}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{v}=\mathbf{0}_{m\times1}} \mathbf{v}_k.\end{aligned}$$

Preliminary

Extended Kalman Filter

In order to simplify the expression of the EKF, the following notation is defined

$$\mathbf{\Phi}_{k-1} \triangleq \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u}_{k-1},\mathbf{w})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1},\mathbf{w}=\mathbf{0}_{n\times 1}}$$

$$\mathbf{H}_{k} \triangleq \left.\frac{\partial \mathbf{h}(\mathbf{x},\mathbf{v})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1},\mathbf{v}=\mathbf{0}_{m\times 1}}$$

$$\mathbf{\Gamma}_{k-1} \triangleq \left.\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u}_{k-1},\mathbf{w})}{\partial \mathbf{w}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1},\mathbf{w}=\mathbf{0}_{n\times 1}}$$

$$\mathbf{u}'_{k-1} \triangleq \mathbf{f}(\hat{\mathbf{x}}_{k-1|k-1},\mathbf{u}_{k-1},\mathbf{0}_{n\times 1}) - \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1|k-1}$$

$$\mathbf{z}'_{k} \triangleq \mathbf{z}_{k} - \mathbf{h}\left(\hat{\mathbf{x}}_{k|k-1},\mathbf{0}_{m\times 1}\right) + \mathbf{H}_{k}\hat{\mathbf{x}}_{k|k-1}$$

$$\mathbf{v}'_{k} \triangleq \left.\frac{\partial \mathbf{h}(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1},\mathbf{v}=\mathbf{0}_{m\times 1}} \mathbf{v}_{k}$$

Simplified model $\longrightarrow$

$$\mathbf{x}_{k} = \mathbf{\Phi}_{k-1}\mathbf{x}_{k-1} + \mathbf{u}'_{k-1} + \mathbf{\Gamma}_{k-1}\mathbf{w}_{k-1}$$

$$\mathbf{z}'_{k-1} = \mathbf{H}_{k}\mathbf{x}_{k} + \mathbf{v}'_{k-1}$$

where, $\mathbf{v}'_{\mathrm{k}}$ is $\mathrm{E}\left(\mathbf{v}'_{k}\right)=\mathbf{0}_{m\times 1}$ and

$$\mathbf{R}_{\mathbf{v}'\mathbf{v}'}(k,j) \triangleq \mathrm{E}\left(\mathbf{v}'_{k}\mathbf{v}'^{\mathrm{T}}_{j}\right) = \begin{cases} \mathbf{R}'_{k}, k=j \\ \mathbf{0}_{m\times m}, k\neq j \end{cases}$$

where

$$\mathbf{R}'_{k} \triangleq \left.\frac{\partial \mathbf{h}(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1},\mathbf{v}=\mathbf{0}_{m\times 1}} \mathbf{R}_{k} \left(\left.\frac{\partial \mathbf{h}(\mathbf{x},\mathbf{v})}{\partial \mathbf{v}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1},\mathbf{v}=\mathbf{0}_{m\times 1}}\right)^{\mathrm{T}}$$

Preliminary

□ Summary of the Extended Kalman filter

1. Step1: Process model

$$\mathbf{x}_k = \mathbf{f}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right),\ \mathbf{w}_k \sim \mathcal{N}\left(\mathbf{0}_{n\times 1}, \mathbf{Q}_k\right)$$

Measurement model

$$\mathbf{z}_k = \mathbf{h}\left(\mathbf{x}_k, \mathbf{v}_k\right),\ \mathbf{v}_k \sim \mathcal{N}\left(\mathbf{0}_{m\times 1}, \mathbf{R}_k\right)$$

2. Step 2: Initial state

$$\hat{\mathbf{x}}_0 = \mathrm{E}\left(\mathbf{x}_0\right)$$

$$\mathbf{P}_0 = \mathrm{E}\left[\left(\mathbf{x}_0 - \mathrm{E}\left(\mathbf{x}_0\right)\right)\left(\mathbf{x}_0 - \mathrm{E}\left(\mathbf{x}_0\right)\right)^{\mathrm{T}}\right]$$

Preliminary

Summary of the Extended Kalman filter

3. Step3: For $k=0$, set

$$\mathbf{P}_{0|0}=\mathbf{P}_0,\hat{\mathbf{x}}_{0|0}=\hat{\mathbf{x}}_0$$

4. Step4: $k=k+1$

5. Step5: State estimate propagation

$$\hat{\mathbf{x}}_{k|k-1}=\mathbf{f}\left(\hat{\mathbf{x}}_{k-1|k-1},\mathbf{u}_{k-1},\mathbf{0}_{n\times 1}\right)$$

6. Step6: Error covariance propagation

$$\mathbf{P}_{k|k-1}=\mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^{\mathrm{T}}+\mathbf{\Gamma}_{k-1}\mathbf{Q}_{k-1}\mathbf{\Gamma}_{k-1}^{\mathrm{T}}$$

7. Step7: Kalman gain matrix

$$\mathbf{K}_k=\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}+\mathbf{R}_k\right)^{-1}$$

8. Step8: State estimate update

$$\hat{\mathbf{x}}_{k|k}=\hat{\mathbf{x}}_{k|k-1}+\mathbf{K}_k\left(\mathbf{z}_k-\hat{\mathbf{z}}_{k|k-1}\right)$$

where, $\hat{\mathbf{z}}_{k|k-1}=\mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$

9. Step9: Error covariance update

$$\mathbf{P}_{k|k}=\left(\mathbf{I}_n-\mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_{k|k-1}$$

10. Step10: Go back to Step4.

Preliminary

In order to make this chapter self-contained, the preliminary is from Chapter. 9 of “**Quan Quan. *Introduction to Multicopter Design and Control*. Springer, Singapore, 2017**”.

Basic Experiment

□ Experimental Objective

■ Things to prepare

(1) Hardware: Pixhawk autopilot system;

(2) Software: MATLAB R2017b or above, Pixhawk Support Package(PSP) Toolbox, Instructional Package “e4.1”(https://rflysim.com/course);

(3) Data for experiment are prepared in Instructional Package “e4.1” for readers without hardware to collect data.

■ Objectives

Repeat the given steps to log accelerometer and gyroscope data via the given Pixhawk autopilot system. Subsequently, run the offered code to compare the estimated attitude of complementary filter with that of raw data and the self-contained filter in PX4 software of the Pixhawk autopilot(the estimate by the PX4 software is taken as ground truth).

Basic Experiment

□ Experimental Procedure

(1) Step1: Log data from accelerometers and gyroscopes

1) Hardware and software connection.The connection between the Radio Controller (RC) receiver and the Pixhawk autopilot is shown in the right figure.

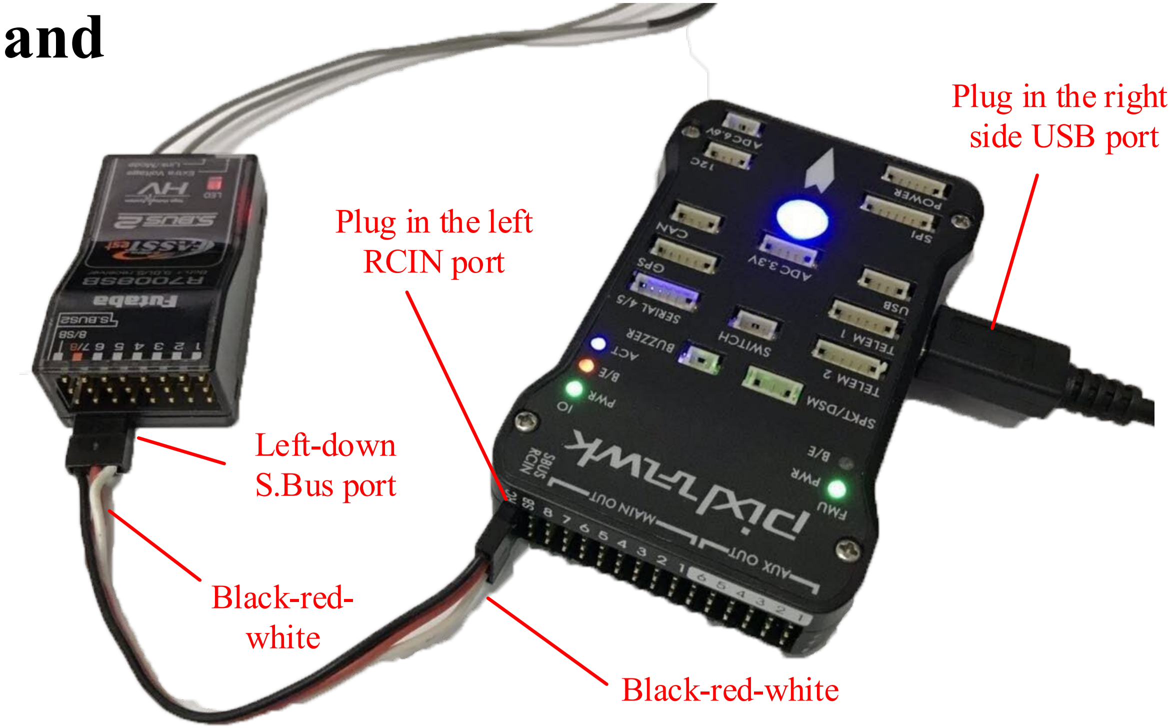


Figure. Pixhawk and RC transmitter connection diagram

Experimental Procedure

(1) Step1: Log data from accelerometers and gyroscopes

2) Open file "log_data.slx" as shown in the right figure. which can obtain data including acceleration, angular velocity, time stamp, and attitude in the Pixhawk autopilot. The data logged is saved in the Pixhawk SD card via placing the upper-left stick (CH5) at a corresponding position.

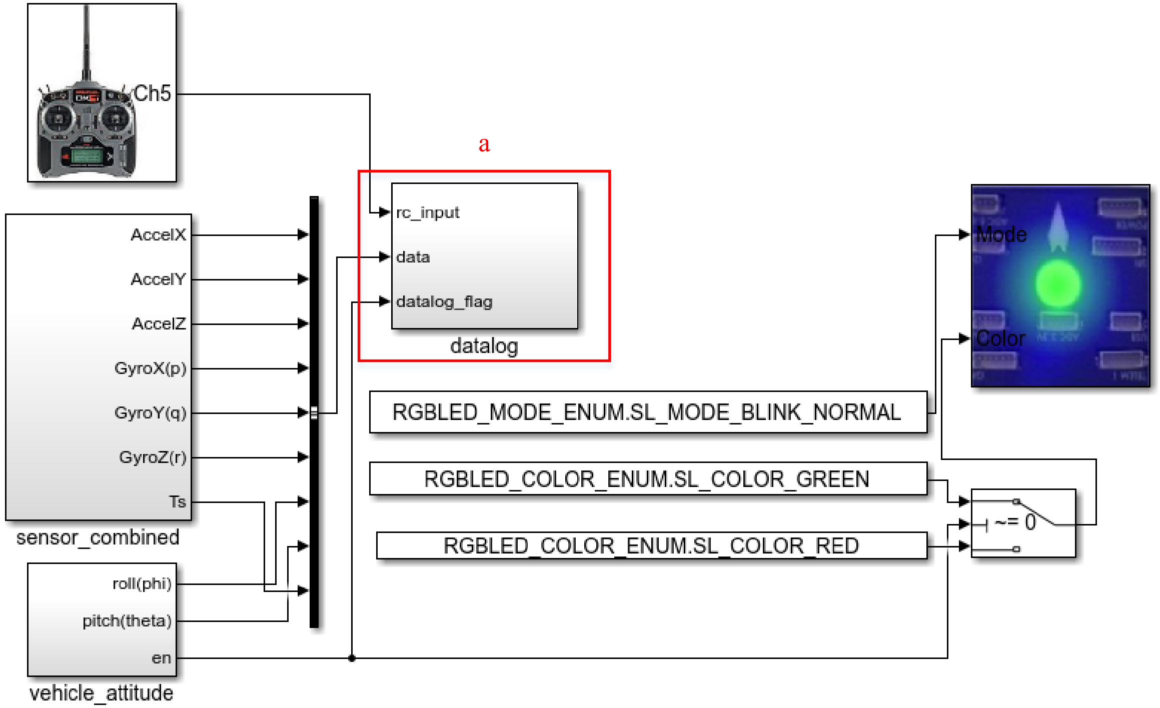


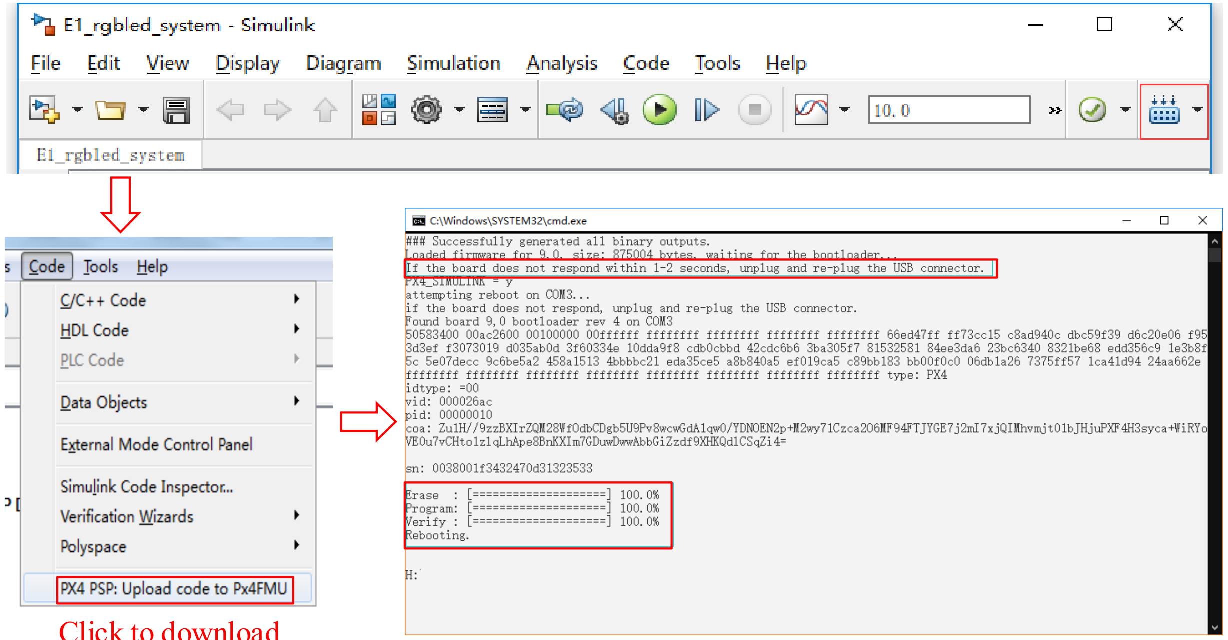
Figure. Data logging, Simulink model "log_data.slx"

Basic Experiment

(1) Step1： Log data from accelerometers and gyroscopes

3) Compile the file “log_data.slx” and upload it to the Pixhawk autopilot

Click to compile



Click to download

Download completed

Figure. Process of compiling and uploading

Basic Experiment

(1) Step1: Log data from accelerometers and gyroscopes

4) Log data. The Pixhawk LED status light lighting red denotes that PX4 software does not work. Thus, after connecting the RC receiver to the Pixhawk autopilot, wait for a while until Pixhawk LED status light gets green (if Pixhawk LED status light does not get green, the Pixhawk is required to be re-plugged). Pull back the upper-left switch corresponding to CH5>1500 to start writing data to the SD card. Subsequently, manually shake the Pixhawk autopilot. After data logging finished, pull forward the upper-left switch (CH5<1500) to stop writing data to the SD card.

5) Read data. First, take the SD card out from the Pixhawk autopilot. Read the data by a card reader. Copy the file "e4_A.bin" to folder "e4\e4.1". Use the function

[datapoints, numpoints] = px4_read_binary_file('mw_A.bin')

to decode the data. The data are saved in "datapoints" and the number of data is saved in "numpoints".

Basic Experiment

Experimental Procedure

(2) Step2： Design a complementary filter

1) The procedure of a complementary filter in MATLAB is in file “Attitude_cf.m”, as shown in the following table. “theta_am” and phi_am” denotes the pitch and roll angles calculated by the raw data from the accelerometer, respectively. theta_cf” and “phi_cf” denotes the pitch and roll angles filtered by the complementary filter, respectively.

```
function [ phi_cf, theta_cf ] = Attitude_cf(dt, z, phi_cf_k, theta_cf_k, tao)
%Function description:
%  Complementary filter for attitude estimation.
%Input:
%  dt: sampling period, unit: s
%  z: three-axis angle gyroscope and three-axis accelerometer measurements, [gx, gy, gz, ax, ay, az]’, unit: rad/s, m/s2
% phi_cf_k, theta_cf_k: Angle value of the previous moment, unit: rad
% tao: Filter parameter
%Output:
%  phi_cf, theta_cf:  Attitude angle, unit: rad

    gx = z(1); gy = z(2);
    ax = z(4); ay = z(5); az = z(6);

    %Calculate the attitude angle using accelerometer measurement
    g = sqrt(ax*ax + ay*ay + az*az);
    theta_am = asin(ax/g);
    phi_am = -asin(ay/(g*cos(theta_am)));

    %Complementary filtering
    theta_cf = tao/(tao + dt)*(theta_cf_k + gy*dt) + dt/(tao + dt)*theta_am;
    phi_cf = tao/(tao + dt)*(phi_cf_k + gx*dt) + dt/(tao + dt)*phi_am;
end
```

☐ Experimental Procedure

(3) Step3： Analyze filtering results

1) Two sets of sensor data are given, where the file “e4_A.bin” stores the data directly from the Pixhawk autopilot rotating by hands, and the file “logdata.mat” stores the data from a practical flight of a quadcopter.

2) Run the file “Attitude_estimator0.m” with the results, where “gyro” corresponds to angle velocity from the gyroscope, “acc” corresponds to the raw data from the accelerometer, “cf” corresponds to the complementary filter and “px4” corresponds to the data from the self-contained filter in PX4 software of the Pixhawk autopilot.

Basic Experiment

◻ Experimental Procedure

(3) Step3： Analyze filtering results

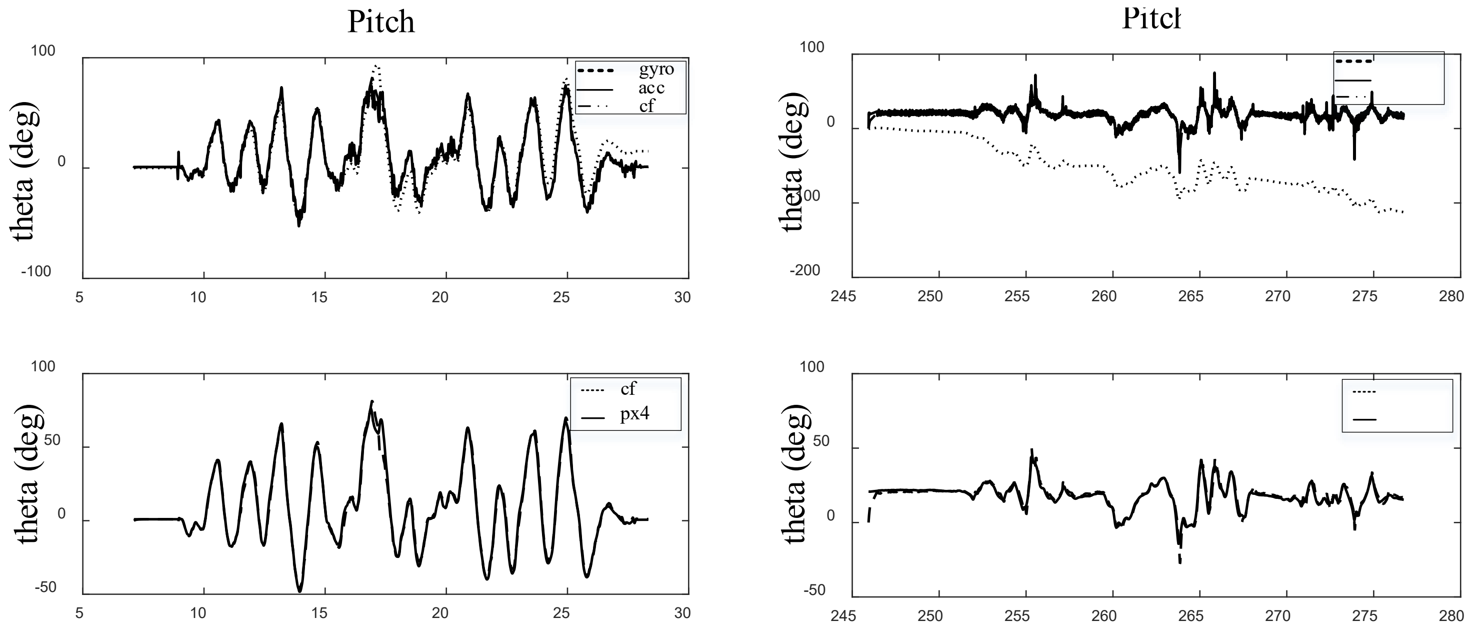


Figure. Estimate comparison of gyro,acc,cf and px4

Several observations are obtained as follows:

1) cumulative errors of the pure integral of the angle velocity from the gyroscope are large and diverge;

2) based on the raw data from the accelerometer, the estimate does not diverge although the noise is larger with obvious peaks and, especially using the data from a practical flight;

3) by using the complementary filter, the estimate is smooth and exhibits less cumulative error.

Analysis Experiment

□ Experimental Objective

■ Things to prepare

Data logged in the basic experiment and Instructional Package "e4.2" (https://rflysim.com/course);

■ Objectives

Based on the basic experiment, change the value of the parameter τ in the complementary filter

$$\hat{\theta}(k)=\frac{\tau}{\tau+T_s}(\hat{\theta}(k-1)+T_s\omega_{y_bm}(k))+\frac{T_s}{\tau+T_s}\theta_m(k)$$

to obseve the filtered result, and analyze the function of the parameter τ in the complementary filter.

Analysis Experiment

Experimental Procedure

Write a program, where the parameter τ in equation

$$\hat{\theta}(k)=\frac{\tau}{\tau+T_s}(\hat{\theta}(k-1)+T_s\omega_{y_bm}(k))+\frac{T_s}{\tau+T_s}\theta_m(k)$$

corresponding to “tao” is modified. Subsequently, compare the estimate with respect to different values of the parameter τ. Obtain the estimate by the file “Attitude_cf_tao.m” with τ corresponding to 0.01, 0.1, and 1, respectively.

```
%The influence of the parameter tao on the filtering performance
clear;
load logdata
n = length(ax);  %Number of data collected
Ts = zeros(1,n);  %Sampling time

Ts(1) = 0.004;

for k =1:n-1
  Ts(k+1) = (timestamp(k + 1) - timestamp(k))*0.000001;
end
theta_cf = zeros(1, n);  %Roll obtained from complementary filtering
phi_cf = zeros(1, n);  %Pitch obtained from complementary filtering
tao = 0.001;

for i = 1 : 3
   tao = tao*10;
   for k = 2 : n
      g = sqrt(ax(k)*ax(k) + ay(k)*ay(k) + az(k)*az(k));
      theta_am = asin(ax(k)/g);
      phi_am = -asin(ay(k)/(g*cos(theta_am)));

      theta_cf(i, k) = tao/(tao + Ts(k))*(theta_cf(i, k - 1) + gy(k)*Ts(k)) + Ts(k)/(tao + Ts(k))*theta_am;
      phi_cf(i,k) = tao/(tao + Ts(k))*(phi_cf(i, k-  1) + gx(k)*Ts(k)) + Ts(k)/(tao + Ts(k))*phi_am;
   end
end
```

Analysis Experiment

Experimental Procedure

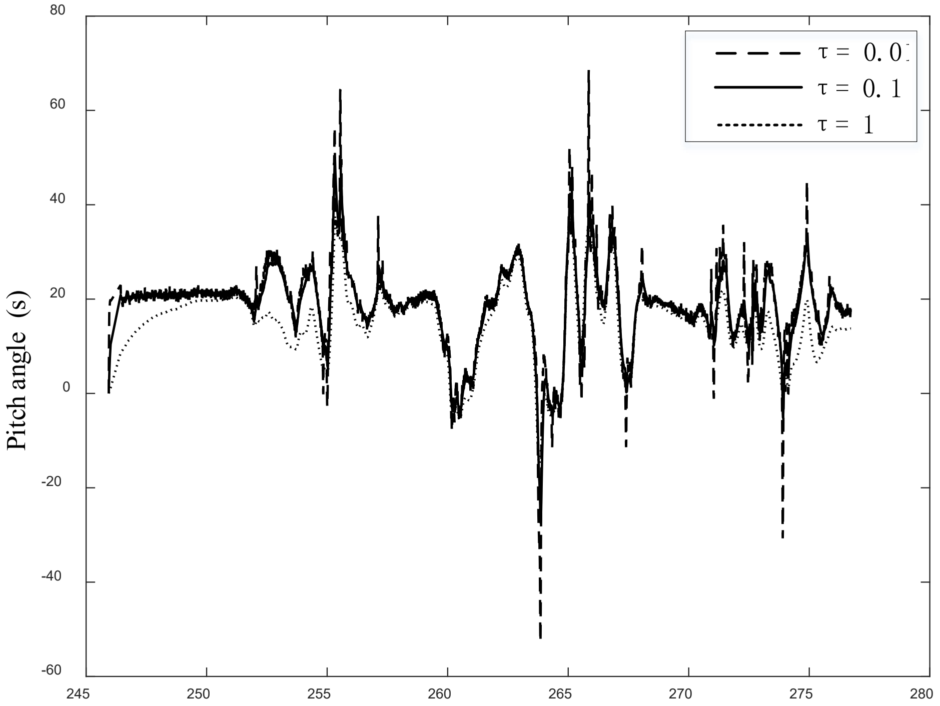


Figure. Pitch estimate with respect to parameter τ

The larger the parameter τ is, the more high-frequency noise is filtered. In particular, when $\tau \gg Ts$, namely

$$\frac{\tau}{\tau+T_s} \approx 1, \frac{T_s}{\tau+T_s} \approx 0$$

Complementary filter is

$$\begin{cases} \hat{\theta}(k) \approx \hat{\theta}(k-1) + T_s \omega_{y_b m}(k) \\ \hat{\phi}(k) \approx \hat{\phi}(k-1) + T_s \omega_{x_b m}(k) \end{cases}$$

This implies that the gyroscope no longer contributes to the estimate and only the calculation of the angle velocity from the accelerometer is used.

Design Experiment

□ Experimental Objective

■ Things to prepare

(1) Hardware: Pixhawk autopilot system;

(2) Software: MATLAB R2017b or above, Pixhawk Support Package(PSP) toolbox, gyroscope and accelerometer data logged in basic experiment and Instructional Package "e4.3"(https://rflysim.com/course);

(3) Data for experiment are prepared in the instructional package "e4.3" for readers without hardware to collect data.

Design Experiment

□ Experimental Objective

■ Objectives

With the obtained data, design a Kalman filter to estimate the pitch and roll angles, and compare the estimated attitude of Kalman filter with that of raw data and the self-contained filter in PX4 software of the Pixhawk autopilot (the estimate by the PX4 software is taken as ground truth).

Design Experiment

□ Experimental Design

(1) Step1： Kalman filter for attitude estimation

$$\dot{\mathbf{R}}_{\mathrm{b}}^{\mathrm{e}}=\mathbf{R}_{\mathrm{b}}^{\mathrm{e}}\left[{}^{\mathrm{b}}\boldsymbol{\omega}\right]_{\times}\Rightarrow\dot{\mathbf{R}}_{\mathrm{e}}^{\mathrm{b}}=-\left[{}^{\mathrm{b}}\boldsymbol{\omega}\right]_{\times}\mathbf{R}_{\mathrm{e}}^{\mathrm{b}}$$

$$\mathbf{x}=\begin{bmatrix}-\sin\theta\\ \sin\phi\cos\theta\\ \cos\phi\cos\theta\end{bmatrix}\xrightarrow{\text{The third column}}\dot{\mathbf{x}}=-\left[{}^{\mathrm{b}}\boldsymbol{\omega}\right]_{\times}\mathbf{x}$$

Process model

The reading of a three-axis accelerometer

$${}^{\mathrm{b}}\mathbf{a}_{\mathrm{m}}=-g\mathbf{x}+\mathbf{n}_{\mathrm{a}}\longleftarrow$$

Measurement model

where $\mathbf{n}_{\mathrm{a}}\in\mathbb{R}^3$ is noise.

Further considering the drift model of gyroscope, the process model of the Kalman filter is established as follows

$$\begin{cases}\dot{\mathbf{x}}=-\left[{}^{\mathrm{b}}\boldsymbol{\omega}_{\mathrm{m}}-\mathbf{b}_{\mathrm{g}}-\mathbf{w}_{\mathrm{g}}\right]_{\times}\mathbf{x}\\ \dot{\mathbf{b}}_{\mathrm{g}}=\mathbf{w}_{\mathbf{b}_{\mathrm{g}}}\end{cases}$$

Design Experiment

☐ Experimental Design

(2) Step2：Design Kalman filter

In order to run Kalman filtering on a computer, a discretization form should be used. So, first of all, the above equations are transformed into the discrete-time difference form through a first-order backward difference that

■ **Process model**

$$\begin{bmatrix} \mathbf{b}_{g,k} \\ \mathbf{x}_{k} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{g,k-1} + \mathbf{w}_{b_g,k-1} T \\ (\mathbf{I} - [{}^{b}\boldsymbol{\omega}_{m,k} - \mathbf{b}_{g,k-1} - \mathbf{w}_{g,k-1}]_{\times} T)\mathbf{x}_{k-1} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{g,k-1} \\ (\mathbf{I} - [{}^{b}\boldsymbol{\omega}_{m,k} - \mathbf{b}_{g,k-1}]_{\times} T)\mathbf{x}_{k-1} \end{bmatrix} + \begin{bmatrix} \mathbf{w}_{b_g,k-1} T \\ [\mathbf{w}_{g,k-1}]_{\times} T)\mathbf{x}_{k-1} \end{bmatrix}$$

■ **Measurement model**

$${}^{b}\mathbf{a}_{m,k} = \begin{bmatrix} \mathbf{0} & -g\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{b}_{g,k} \\ \mathbf{x}_{k} \end{bmatrix} + \mathbf{n}_{a,k}$$

Design Experiment

□ Experimental Design

(2) Step2： Design Kalman filter

Using Taylor expansion for the process model, you can further obtain the information required by the Kalman filter.

The transition matrix is

$$\mathbf{\Phi}_{\text{k-1}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -[\mathbf{x}_{\text{k-1}}]_{\times}\text{T} & (\mathbf{I} - [{}^{\text{b}}\boldsymbol{\omega}_{\text{m,k}} - \mathbf{b}_{g,k-1}]_{\times}\text{T}) \end{bmatrix}$$

The system noise matrix is

$$\mathbf{\Gamma}_{k-1} = \begin{bmatrix} T*\mathbf{I} & \mathbf{0} \\ 0 & -[\mathbf{x}_{\text{k-1}}]_{\times}\text{T} \end{bmatrix}$$

The measurement matrix is

$$\mathbf{H}_{\text{k}} = \begin{bmatrix} \mathbf{0} & \text{-g}\mathbf{I} \end{bmatrix}$$

Design Experiment

□ Experimental Design

(3) Step3：Kalman filtering step

1) State estimate propagation

$$\begin{bmatrix} \mathbf{b}_{g,k} \\ \mathbf{x}_k \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{g,k-1} \\ (\mathbf{I}_3 - [{}^{b}\omega_{m,k} - \mathbf{b}_{g,k-1}]_\times T_s)\mathbf{x}_{k-1} \end{bmatrix}$$

Calculate state transition matrix and system noise matrix

$$\mathbf{\Phi}_{k-1} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_{3\times3} \\ -[\ \mathbf{x}_{k-1}\ ]_\times T_s & (\mathbf{I}_3 - [{}^{b}\omega_{m,k} - \mathbf{b}_{g,k-1}]_\times T_s) \end{bmatrix}$$

$$\mathbf{\Gamma}_{k-1} = \begin{bmatrix} T_s\mathbf{I}_3 & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & -[\ \mathbf{x}_{k-1}\ ]_\times T_s \end{bmatrix}$$

where ${}^{b}\boldsymbol{\omega}_{m,k}$ denotes the current measurement value of the gyroscope, and $\mathbf{x}_{k-1}$ denotes the former state estimate.

Design Experiment

◻ Experimental Design

(3) Step3：Kalman filtering step

2) Error covariance propagation

$$\mathbf{P}_{k|k-1} = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{\Gamma}_{k-1}\mathbf{Q}_{k-1}\mathbf{\Gamma}_{k-1}^{\mathrm{T}}$$

where $\mathbf{Q}_{k-1}$ denotes the variance of system noise.

3) Kalman gain matrix

$$\mathbf{K}_{k} = \mathbf{P}_{k|k-1}\mathbf{H}_{k}^{\mathrm{T}}(\mathbf{H}_{k}\mathbf{P}_{k|k-1}\mathbf{H}_{k}^{\mathrm{T}} + \mathbf{R}_{k})^{-1}$$

where $\mathbf{R}_{k-1}$ denotes the variance of measurement noise.

Design Experiment

Experimental Design

(3) Step3: Kalman filtering step

4) State estimate update

$$\begin{bmatrix} \mathbf{b}_{g,k} \\ \mathbf{x}_k \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{g,k-1} \\ \mathbf{x}_{k-1} \end{bmatrix} + \mathbf{K}_k \left(\mathbf{z}_k - \mathbf{H}_k \begin{bmatrix} \mathbf{b}_{g,k-1} \\ \mathbf{x}_{k-1} \end{bmatrix} \right)$$

where $\mathbf{z}_k$ denotes the accelerometer measurement.

5) Error covariance update

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}$$

Design Experiment

□ Experimental Design

The procedure of Kalman filter is in file "Attitude_ekf.m", as shown in the following table.

```
function [ x_aposteriori, P_aposteriori, roll, pitch] =
 Attitude_ekf( dt, z, q, r, x_aposteriori_k, P_aposteriori_k)
%unction description:
%  Extended Kalman Filtering Method for State Estimation
%nput:
%  dt: Sampling period
%  z: Measured value
%  q:System noise,r:Measuring noise
%  x_aposteriori_k: State estimate at the last moment
%  P_aposteriori_k: Estimate covariance at the last moment
%Output:
%  x_aposteriori：  State estimate of current time
%  P_aposteriori：  Estimated covariance at the current moment
%  roll,pitch：  Euler angle, unit: rad

w_m = z(1:3);  %Angular velocity measurement
a_m = z(4:6);  %Acceleration measurement
g = norm(a_m,2);   %Gravitational acceleration
%   w_x_=[ 0,-(wz-bzg, wy-byg;
%        wz-bzg, 0 ,-(wx-bxg);
%        -(wy-byg), wx-bxg, 0];
w_x_ = [0, -(w_m(3) - x_aposteriori_k(3)), w_m(2) -x_aposteriori_k(2);
     w_m(3) - x_aposteriori_k(3), 0, -(w_m(1) - x_aposteriori_k(1));
     -(w_m(2) - x_aposteriori_k(2)), w_m(1) - x_aposteriori_k(1), 0];
bCn = eye(3, 3) - w_x_*dt;
% Predict
% The state estimate propagation
x_apriori = zeros(1, 6);
x_apriori(1: 3) = x_aposteriori_k(1 : 3);  %The drift model of
gyroscope
x_apriori(4 : 6) = bCn*x_aposteriori_k(4 : 6);   %Acceleration
normalized value
%[x]x
x_aposteriori_k_x = [0, -x_aposteriori_k(6), x_aposteriori_k(5);
                    x_aposteriori_k(6), 0, -x_aposteriori_k(4);
                   -x_aposteriori_k(5), x_aposteriori_k(4), 0];
% Update state transition matrix
PHI = [eye(3, 3), zeros(3, 3);
        -x_aposteriori_k_x*dt, bCn];
```

Design Experiment

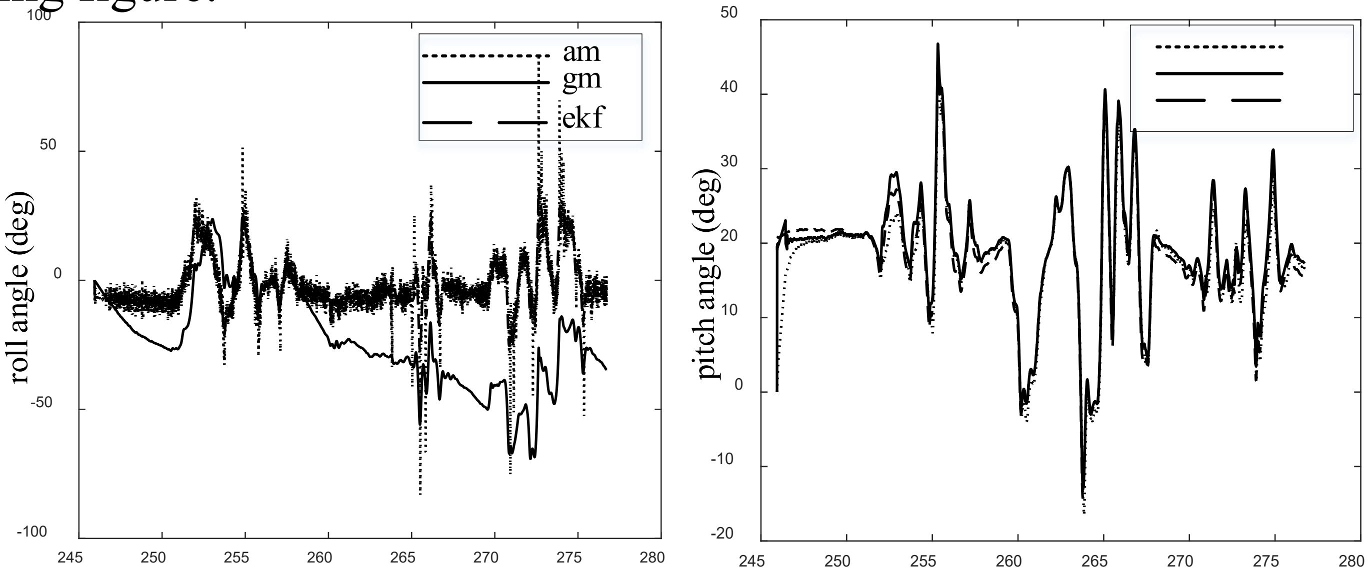
```
GAMMA = [eye(3, 3)*dt, zeros(3, 3); % System noise matrix
      zeros(3, 3), -x_aposteriori_k_x*dt];

Q = [eye(3, 3)*q(1), zeros(3, 3);
   zeros(3, 3), eye(3, 3)*q(2)];
%  Error covariance propagation matrix
 P_apriori = PHI*P_aposteriori_k*PHI' + GAMMA*Q*GAMMA';
% Update
R = eye(3, 3)*r(1);
H_k = [zeros(3, 3), -g*eye(3, 3)];
%Kalman gain
K_k = (P_apriori*H_k')/(H_k*P_apriori*H_k' + R);
% State estimation matrix
x_aposteriori = x_apriori' + K_k*(a_m - H_k*x_apriori');
% Estimation error covariance
P_aposteriori = (eye(6, 6) - K_k*H_k)*P_apriori;
% Calculate roll, pitch
k = x_aposteriori(4 : 6) /norm(x_aposteriori(4 : 6), 2);

roll = atan2(k(2), k(3));
pitch = -asin(k(1));
end
```

Design Experiment

□ Simulation Procedure

(1) Step1：Algorithm simulation and verification

Run the file “Attitude_estimator.m” in file folder “e4.3” to obtain the estimate shown in the following figure.



An observation is that the estimate by the Kalman filter algorithm is better than that by the complementary filtering when the data is from an actual flight.

Figure. Roll and pitch estimation comparison

Design Experiment

The main code in file “Attitude_estimator.m” is as follows

```
clear;
load logdata n = length(ax);  %Number of data collected
Ts = zeros(1,n);  %Sampling time

Ts(1)  =0.004;

for k = 1 : n-1
   Ts(k+1) = (timestamp(k + 1) - timestamp(k))*0.000001;
end

theta_am = zeros(1, n);  %Roll calculated from acceleration
phi_am = zeros(1, n);  %Pitch calculated from acceleration
theta_gm = zeros(1, n);  %Roll from the gyroscope
phi_gm = zeros(1, n);  %Pitch from the gyroscope
theta_cf = zeros(1, n);  %Roll obtained from complementary filtering
phi_cf = zeros(1, n);  %Pitch obtained from complementary filtering
phi_ekf = zeros(1, n);
theta_ekf = zeros(1, n);

tao = 0.3;
w = [0.08, 0.01];  %System noise
v = 50;  %Measurement noise
P_aposteriori = zeros(6, 6, n);
P_aposteriori(:, :, 1)=eye(6, 6)*100;  %P0
x_aposteriori = zeros(6, n);
x_aposteriori(:, 1) = [0, 0, 0, 0, 0, -1];  %X0

for k = 2 : n
  %Calculate Euler angles using accelerometer data
  g = sqrt(ax(k)*ax(k) + ay(k)*ay(k) + az(k)*az(k));
  theta_am(k) = asin(ax(k)/g);
  phi_am(k) = -asin(ay(k)/(g*cos(theta_am(k))));
  %Calculate Euler angles using gyroscope data
  theta_gm(k) = theta_gm(k - 1) + gy(k)*Ts(k);
  phi_gm(k) = phi_gm(k - 1) + gx(k)*Ts(k);
 %Complementary filtering and EKF
  z = [gx(k), gy(k), gz(k), ax(k), ay(k), az(k)];
 [ phi_cf(k), theta_cf(k) ] = Attitude_cf(Ts(k), z', phi_cf(k - 1), theta_cf(k - 1), tao);
  [x_aposteriori(1 : 6, k), P_aposteriori(1 : 6, 1 : 6, k), phi_ekf(k), theta_ekf(k)] = Attitude_ekf(Ts(k), z', w, v, x_aposteriori(1 : 6, k - 1), P_aposteriori(1 : 6, 1 : 6, k - 1));
end
t = timestamp*0.000001;
rad2deg = 180/pi;
```

Design Experiment

□ Simulation Procedure

(2) Step2: Design the model for the HIL simulation

Based on the linear complementary filter and Kalman filter designed above, design a Simulink model termed as “ekf_cf.slx” as shown in the right figure, which includes the two filters and can save the data to the SD card of the Pixhawk autopilot.

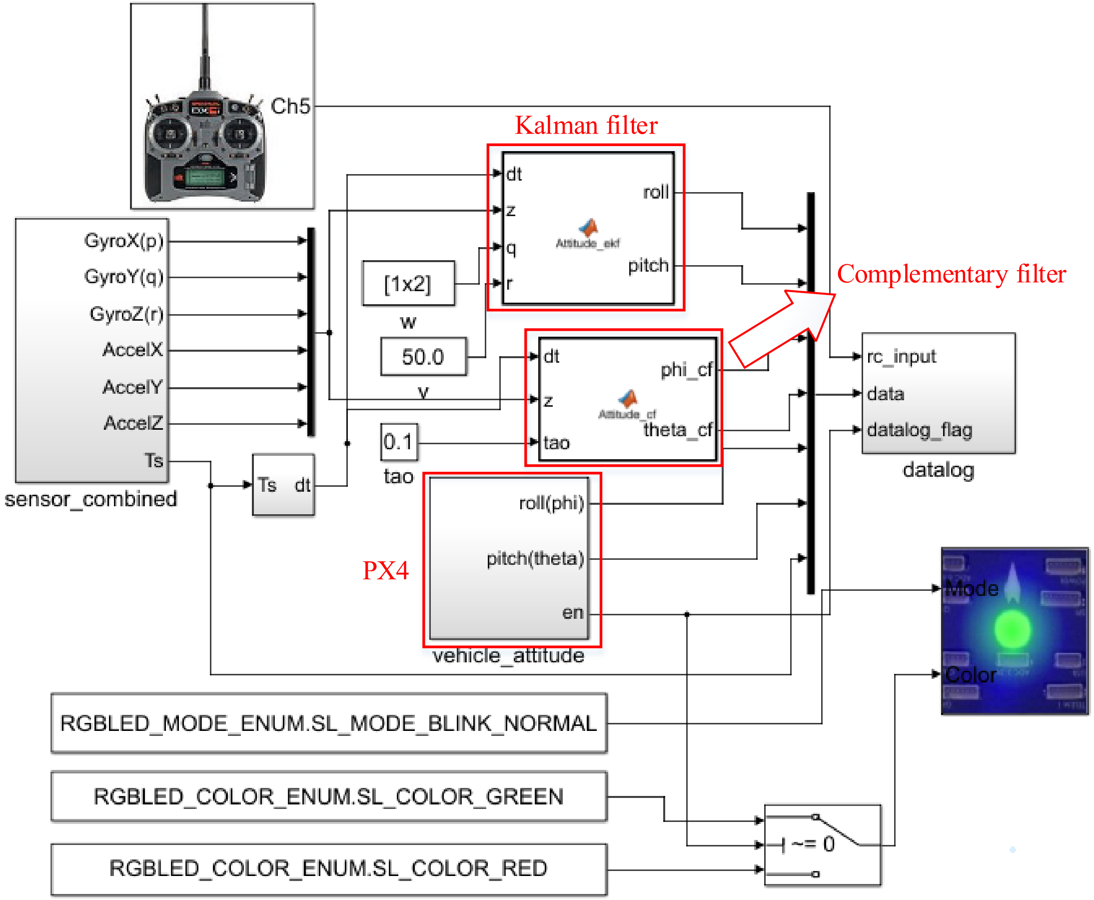


Figure. Kalman filter and complementary filter comparison, Simulink model “ekf_cf.slx”

Simulation Procedure

(3) Step3： Hardware connection

The connection between the RC receiver and the Pixhawk autopilot is shown in the right figure.

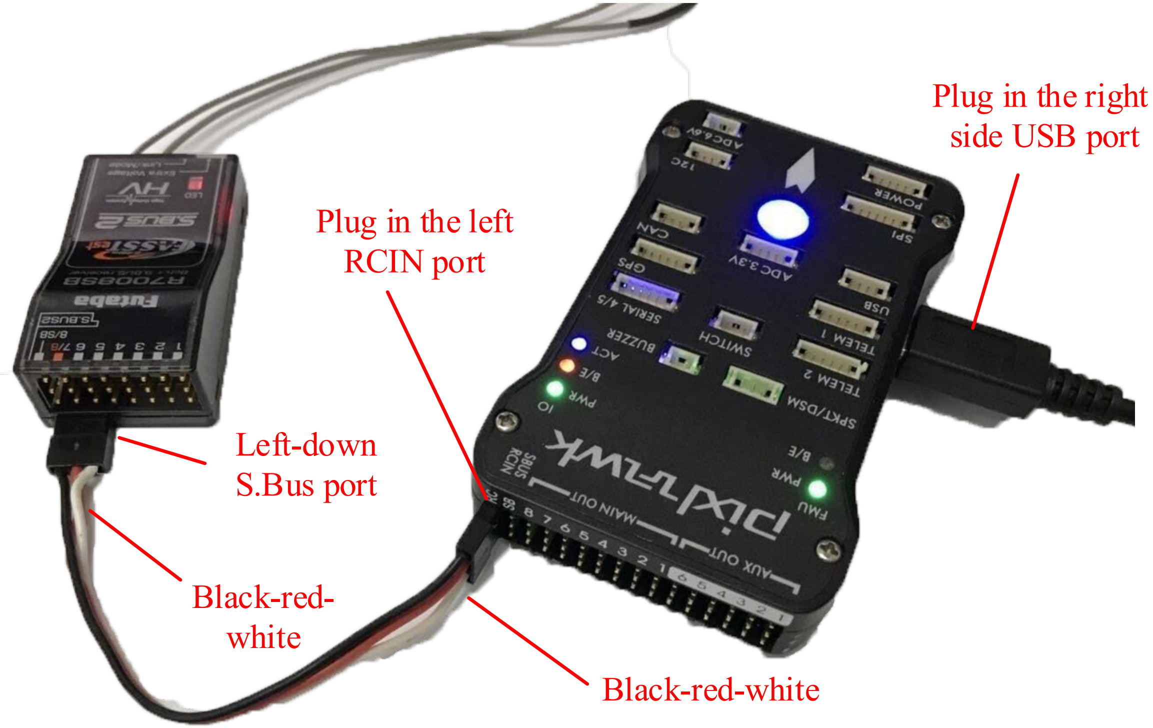


Figure. Pixhawk and RC transmitter connection diagram

Design Experiment

□ Simulation Procedure

(4) Step4: Compile and upload the codes

Compile the file “ekf_cf.slx” and upload it to the Pixhawk autopilot.

(5) Step5: Data logging

The Pixhawk LED status light being in red denotes that PX4 software does not work. Hence, after connecting the RC receiver and the Pixhawk autopilot, wait for a while until Pixhawk LED status light gets green (if Pixhawk LED status light does not get green, the Pixhawk requires to be re-plugged). Pull back the upper-left switch corresponding to Ch5>1500 to start writing data to the SD card. Subsequently, manually rotate the Pixhawk autopilot. After data logging finished, pull forward the upper-left switch (CH5<1500) to stop writing data to the SD card.

Design Experiment

◻ Simulation Procedure

(6) Step6: Read data

Take out the SD card, read the data by a card reader, copy the file “ekf1_A.bin” to folder “e4\e4.3”.

(7) Step7: Draw data curves

Run the file “plot_filter.m” to get the curves, as shown in the right figure.

The observation is made that, during the first half time when the Pixhawk autopilot is rotated slowly, the estimate results by the complementary filter, Kalman filter and the self-contained filter in PX4 software of the Pixhawk autopilot are similar. During the second half time, the Pixhawk autopilot is rotated quickly. The estimate by the complementary filter is evidently bad although, the estimates by the designed Kalman filter and PX4 software in the Pixhawk autopilot are similar.

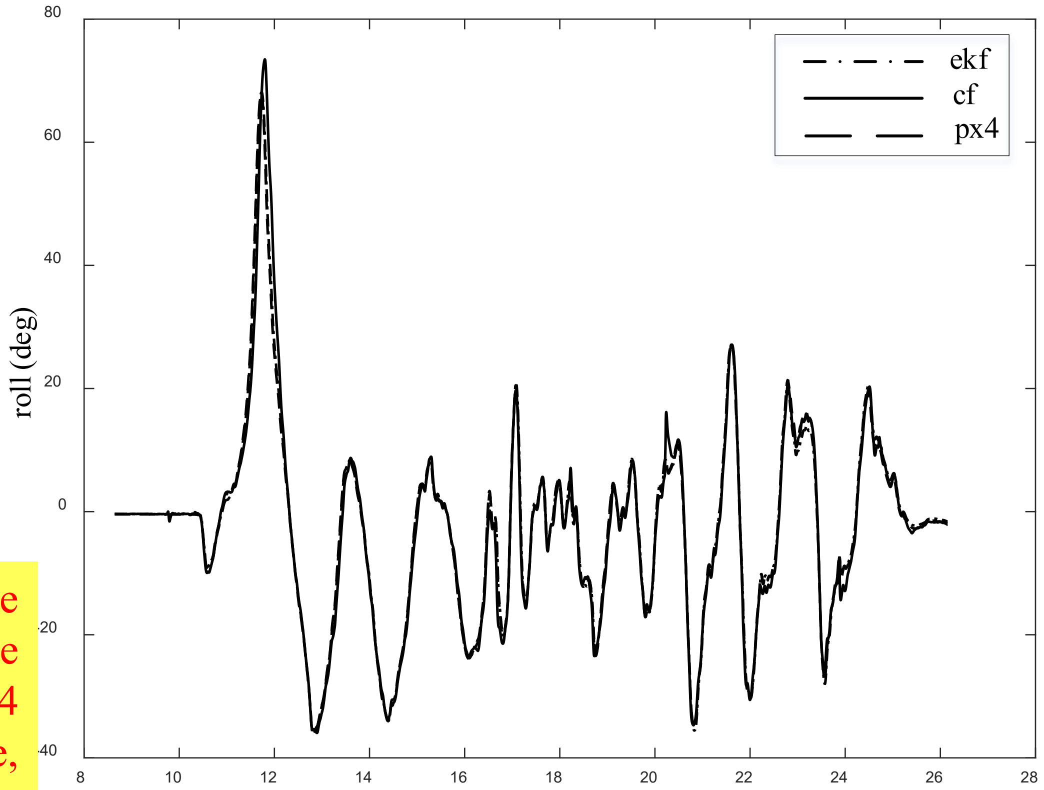


Figure. Comparison among complementary filter(cf), Kalman filter(ekf) and filter in PX4(px4)

Summary

(1) In order to obtain an accurate attitude angle, a complementary filter is designed to fuse the data from the gyroscope and accelerometer. This filter is equivalent to a high-pass filter for the gyroscope and a low-pass filter for the accelerometer, which effectively eliminates the measurement noise and improve accuracy.

(2) The contribution of the gyroscope and accelerometer in the complementary filter is controlled by parameter τ. Hence, the value of the parameter τ affects the complementary filter performance. When the parameter τ is high, the gyroscope plays a major role. In contrast, the accelerometer contributes more when the parameter τ is small.

(3) Design a Kalman filter including the process model and the measurement model. The experimental results indicate that the Kalman filter is better than the complementary filter and is similar to the self-contained filter in PX4 software of the Pixhawk autopilot.

Resource

All course PPTs, videos, and source code will be released on our website

https://rflysim.com/en

For more detailed content, please refer to the textbook:

Quan Quan, Xunhua Dai, Shuai Wang. *Multicopter Design and Control Practice*. Springer, 2020

https://www.springer.com/us/book/9789811531378

If you encounter any problems, please post question at Github page

https://github.com/RflySim/RflyExpCode/issues

If you are interested in RflySim advanced platform and courses for rapid development and testing of UAV Swarm/Vision/AI algorithms, please visit:

https://rflysim.com/en/4_Pro/Advanced.html

Thanks