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

Lesson 07 Seneor Calibration 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

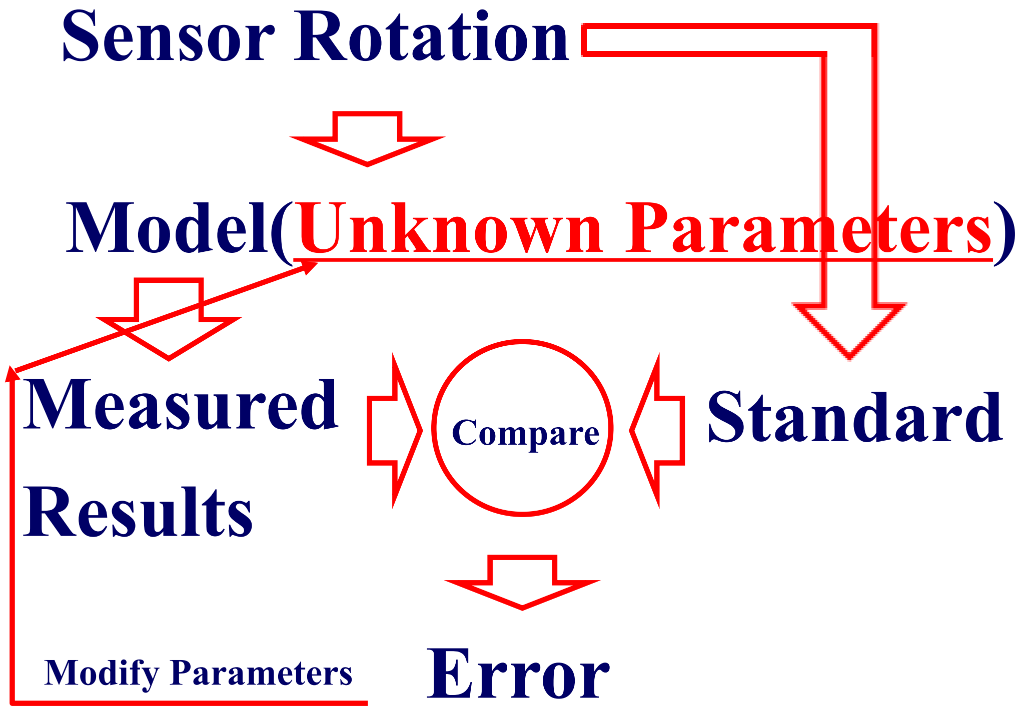
Outline

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

Preliminary

☐ Calibration

(1) Calibration


Sensor Rotation

Model(Unknown Parameters)

Measured Results

Compare

Standard

Error

Modify Parameters

(2) Automatic calibration

- ● General calibration:

  need external calibration equipment, but accurate.

- ● Automatic calibration:

  do not require external calibration devices, simple, slightly poor precision

Preliminary

□ Three-Axis Accelerometer

(1) Error Model

There are always some deviations in the three-axis accelerometer during production and installation. Therefore, the relationship between the calibrated specific force and the specific force without calibration needs to be established. The error model is as follows

$$ {}^{\mathrm{b}}\mathbf{a}_{\mathrm{m}} = \mathbf{T}_{\mathrm{a}}\mathbf{K}_{\mathrm{a}}\left({}^{\mathrm{b}}\mathbf{a}'_{\mathrm{m}} + \mathbf{b}'_{\mathrm{a}}\right) $$

The calibrated specific force

$$ \mathbf{T}_{\mathrm{a}} = \begin{bmatrix} 1 & \Delta\psi_{\mathrm{a}} & -\Delta\theta_{\mathrm{a}} \\ -\Delta\psi_{\mathrm{a}} & 1 & \Delta\phi_{\mathrm{a}} \\ \Delta\theta_{\mathrm{a}} & -\Delta\phi_{\mathrm{a}} & 1 \end{bmatrix} $$

The tiny tilt in the process of mounting sensors

$$ \mathbf{K}_{\mathrm{a}} = \begin{bmatrix} s_{\mathrm{ax}} & 0 & 0 \\ 0 & s_{\mathrm{ay}} & 0 \\ 0 & 0 & s_{\mathrm{az}} \end{bmatrix} $$

The scale factor

The specific force without calibration

$$ \mathbf{b}'_{\mathrm{a}} = \begin{bmatrix} b'_{\mathrm{ax}} \\ b'_{\mathrm{ay}} \\ b'_{\mathrm{az}} \end{bmatrix} $$

The bias

Preliminary

□ Three-Axis Accelerometer

(2) Calibration Principle

To calibrate an accelerometer, the following unknown parameters need to be estimated

$$\boldsymbol{\Theta}_{\mathrm{a}}=\begin{bmatrix}\Delta\psi_{\mathrm{a}} & \Delta\theta_{\mathrm{a}} & \Delta\phi_{\mathrm{a}} & s_{\mathrm{ax}} & s_{\mathrm{ay}} & s_{\mathrm{az}} & b'_{\mathrm{ax}} & b'_{\mathrm{ay}} & b'_{\mathrm{az}}\end{bmatrix}^{T}$$

Then, it can be written as a function, as follows

$${}^{\mathrm{b}}\mathbf{a}_{\mathrm{m}}=\mathbf{h}_{\mathrm{a}}\left(\Theta_{\mathrm{a}},{}^{\mathrm{b}}\mathbf{a}'_{\mathrm{m}}\right)=\mathbf{T}_{\mathrm{a}}\mathbf{K}_{\mathrm{a}}\left({}^{\mathrm{b}}\mathbf{a}'_{\mathrm{m}}+\mathbf{b}'_{\mathrm{a}}\right)$$

Principle: The calibration principle is that the magnitude of specific force keeps constant with different attitude of accelerometers, i.e. the local gravity, denoted as g.

The calibration principle is that the magnitude of specific force keeps constant with different attitude of accelerometers, i.e. the local gravity, denoted as g.

$$\boldsymbol{\Theta}_{\mathrm{a}}^{*}=\arg\min_{\boldsymbol{\Theta}_{\mathrm{a}}}\sum_{k=1}^{M}\left(\left\|\mathbf{h}_{\mathrm{a}}\left(\boldsymbol{\Theta}_{\mathrm{a}},{}^{\mathrm{b}}\mathrm{a}'_{\mathrm{m},k}\right)\right\|-g\right)^{2}$$

arg min {} denotes the minimum value of the variable in the objective function

Preliminary

☐ Three-Axis Magnetometer

(1) Error Model

There are always some deviations in the three-axis magnetometer during production and installation. Therefore, the relationship between the magnetic induction value after calibration and he magnetic induction value without calibrationneeds to be established. The error model is as follows

$$
{}^{\mathrm{b}}\mathbf{m}_{\mathrm{m}} = \mathbf{T}_{\mathrm{m}}\mathbf{K}_{\mathrm{m}}\left({}^{\mathrm{b}}\mathbf{m}'_{\mathrm{m}} + \mathbf{b}'_{\mathrm{m}}\right)
$$

The magnetic induction value after calibration

$$
\mathbf{T}_{\mathrm{m}} = \begin{bmatrix} 1 & \Delta\psi_{\mathrm{m}} & -\Delta\theta_{\mathrm{m}} \\ -\Delta\psi_{\mathrm{m}} & 1 & \Delta\phi_{\mathrm{m}} \\ \Delta\theta_{\mathrm{m}} & -\Delta\phi_{\mathrm{m}} & 1 \end{bmatrix}
$$

The tiny tilt in the process of mounting sensors

$$
\mathbf{K}_{\mathrm{m}} = \begin{bmatrix} s_{\mathrm{mx}} & 0 & 0 \\ 0 & s_{\mathrm{my}} & 0 \\ 0 & 0 & s_{\mathrm{mz}} \end{bmatrix}
$$

The scale factor

The magnetic induction value without calibration.

$$
\mathbf{b}'_{\mathrm{m}} = \begin{bmatrix} b'_{\mathrm{mx}} \\ b'_{\mathrm{my}} \\ b'_{\mathrm{mz}} \end{bmatrix}
$$

The bias

Preliminary

□ Three-Axis Magnetometer

(2) Calibration Principle

Principle: In normal, the magnetic induction keeps constant with different attitude of magnetometer. Here, the magnetic induction value is normalized $\left\| {}^{\mathrm{b}}\mathbf{m}_{\mathrm{m},k} \right\|^2 = 1, k = 1, 2, \ldots, \mathrm{M}$

To calibrate a magnetometer, the following unknown parameters need to be estimated.

$$\boldsymbol{\Theta}_{\mathrm{m}} \triangleq \begin{bmatrix} \Delta\psi_{\mathrm{m}} & \Delta\theta_{\mathrm{m}} & \Delta\phi_{\mathrm{m}} & s_{\mathrm{mx}} & s_{\mathrm{my}} & s_{\mathrm{mz}} & b'_{\mathrm{mx}} & b'_{\mathrm{my}} & b'_{\mathrm{mz}} \end{bmatrix}^{\mathrm{T}}$$

Then, it can be written as a function, as follows

$$\mathbf{h}_{\mathrm{m}}(\boldsymbol{\Theta}_{\mathrm{m}}, {}^{\mathrm{b}}\mathbf{m}'_{m}) \triangleq \mathbf{T}_{\mathrm{m}}\mathbf{K}_{\mathrm{m}}({}^{\mathrm{b}}\mathbf{m}'_{\mathrm{m}} + \mathbf{b}'_{\mathrm{m}})$$

According to this principle, the following optimization is given

$$\boldsymbol{\Theta}^{*}_{\mathrm{m}} = \arg\min_{\boldsymbol{\Theta}_{\mathrm{m}}} \sum_{k=1}^{M} \left(\left\| \mathbf{h}_{\mathrm{m}}\left(\boldsymbol{\Theta}_{\mathrm{m}}, {}^{\mathrm{b}}\mathbf{m}'_{\mathrm{m},k}\right) \right\| - 1 \right)^2$$

Preliminary

In order to make this chapter self-contained, the preliminary is from Chapter. 7 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, QGroundControl(QGC), Instruction Package “e3.1”(https://rflysim.com/course);

(3) Data for calibration are prepared in Instructional Package “e3.1” for readers without hardware to collect data.

■ Objectives

Repeat the given calibration steps to calibrate an accelerometer in the given Pixhawk autopilot system. Subsequently, make a comparison between the calibrated results and uncalibrated results.

Basic Experiment

☐ Experimental Procedure

(1) Step1： Obtain accelerometer data via the Pixhawk autopilot system

1) Hardware connection

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

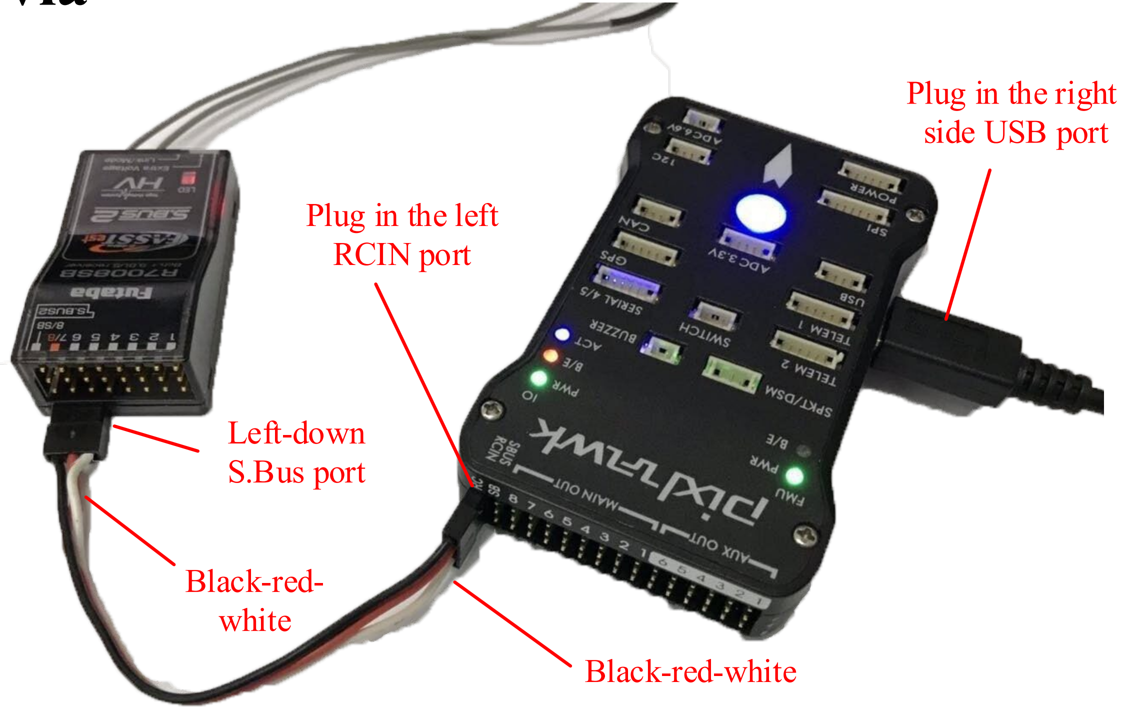


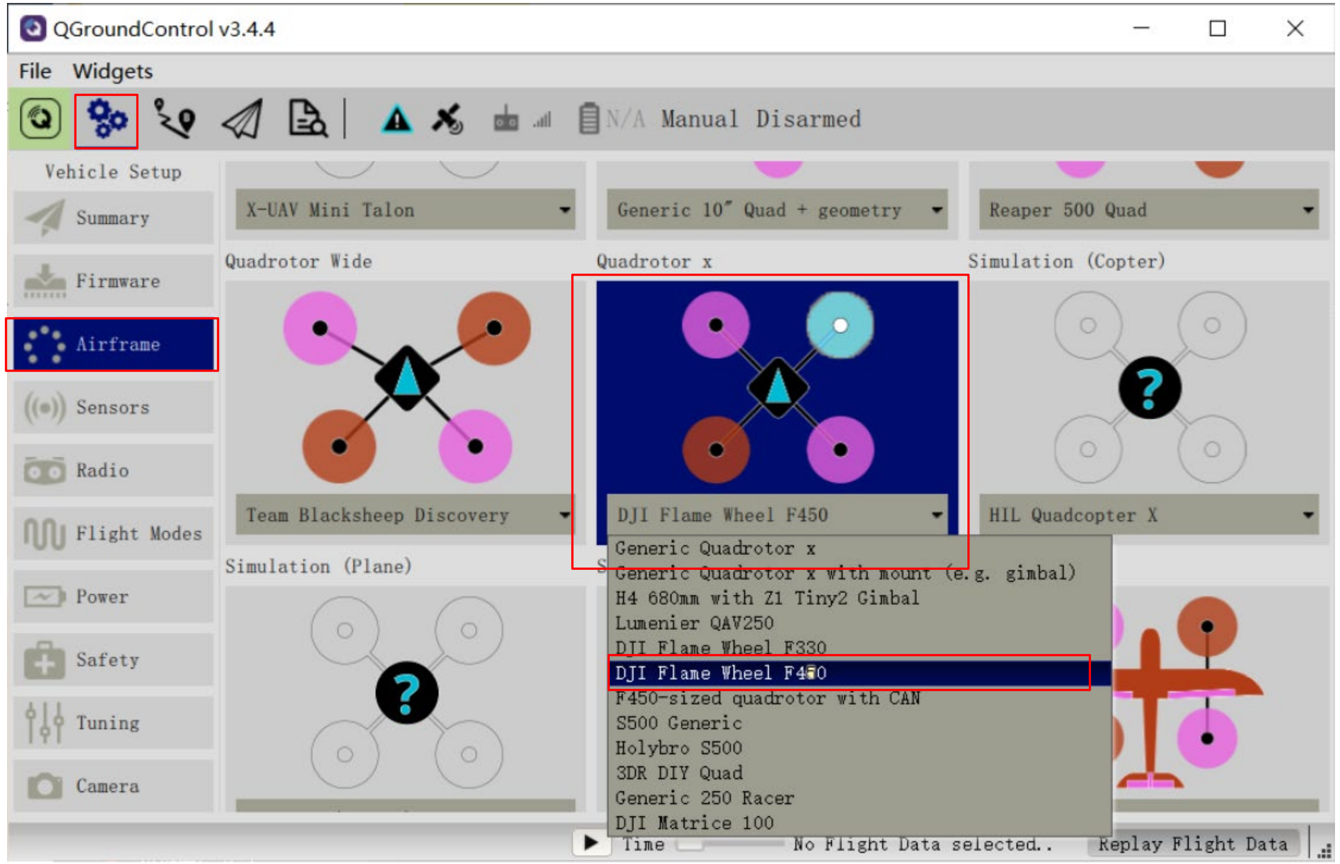
Figure. Pixhawk and RC transmitter connection

Basic Experiment

(1) Step1：Obtain accelerometer data via the Pixhawk autopilot system

1) Hardware connection

Open the QGC, as shown in the right figure and then select “Vehicle Setup” – “Airframe” –“Quadcopter x” – “DJI Flame Wheel F450”.

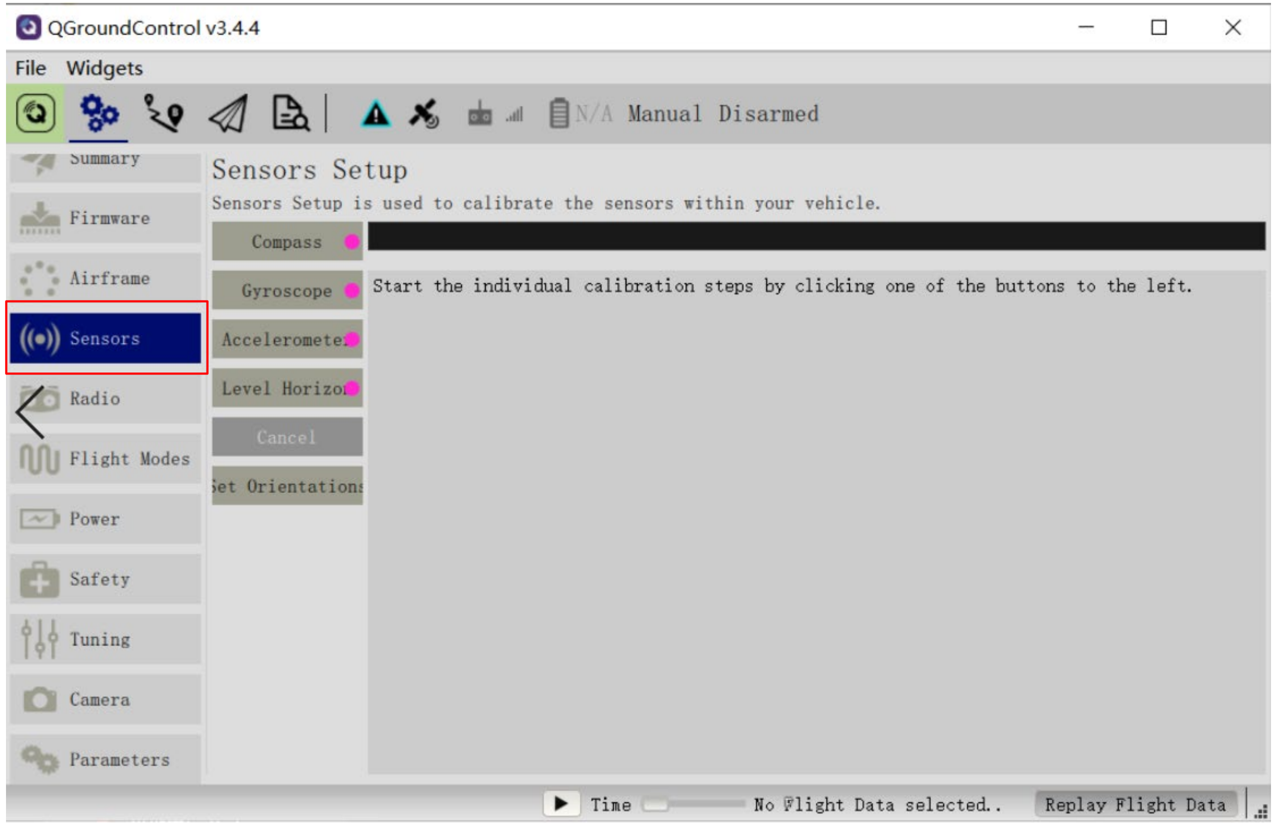


Basic Experiment

(1) Step1：Obtain accelerometer data via the Pixhawk autopilot system

1) Hardware connection

Finally, click the “Apply” button and the autopilot will automatically reboot to make the adopted configuration available. If the adopted configuration is modified, the system will require another re-calibration of the sensor data, as shown in the figure.



Sensors Setup

Sensors Setup is used to calibrate the sensors within your vehicle.

Start the individual calibration steps by clicking one of the buttons to the left.

No Flight Data selected..

Basic Experiment

☐ Accelerometer Calibration Procedure

(1) Step1： Obtain accelerometer data via the Pixhawk autopilot system

2) Open the data collection model

To record the sensor and Radio Controller (RC) data in the Pixhawk microSD card, a file “acquire_data_ag.slx” is created, as shown in the right figure.

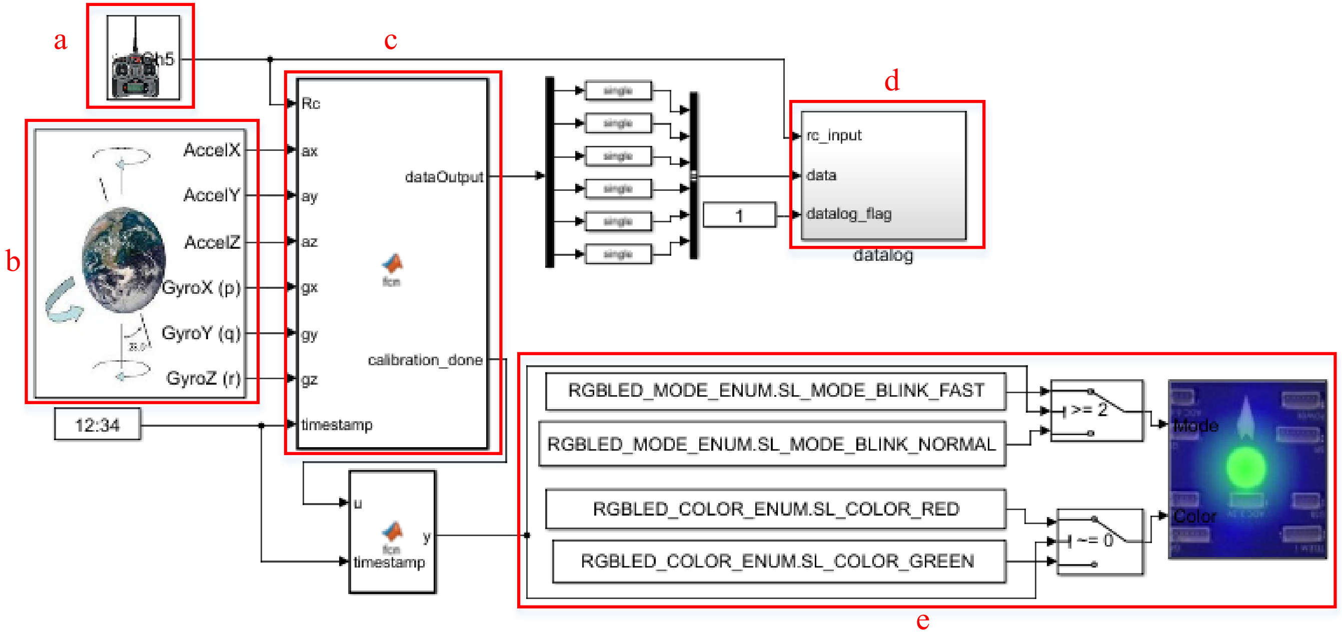


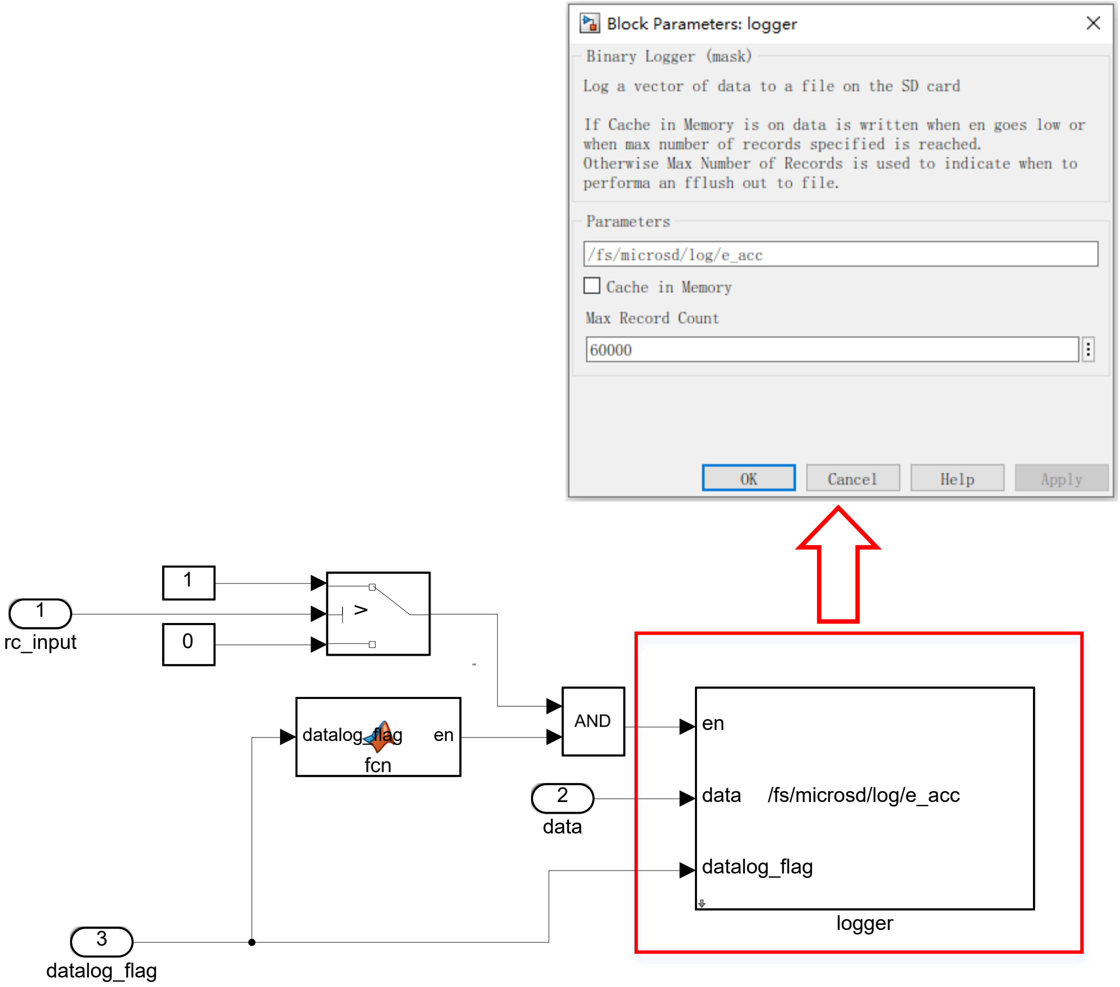
Figure. Accelerometer data logging, Simulink model “acquire_data_ag.slx”

Basic Experiment

(1) Step1： Obtain accelerometer data via the Pixhawk autopilot system

2) Open the data collection model

Write data to the microSD card. Double-click block “binary_logger”, as shown in the right figure. The first three path names of “fs/microsd/log/e3” cannot be changed, whereas the last path name “e3” is the name of the file to the log data, and can be changed as desired.



Block Parameters: logger

Binary Logger (mask)

Log a vector of data to a file on the SD card

If Cache in Memory is on data is written when en goes low or when max number of records specified is reached.
Otherwise Max Number of Records is used to indicate when to performa an fflush out to file.

Parameters

/fs/microsd/log/e_acc

☐ Cache in Memory

Max Record Count

60000

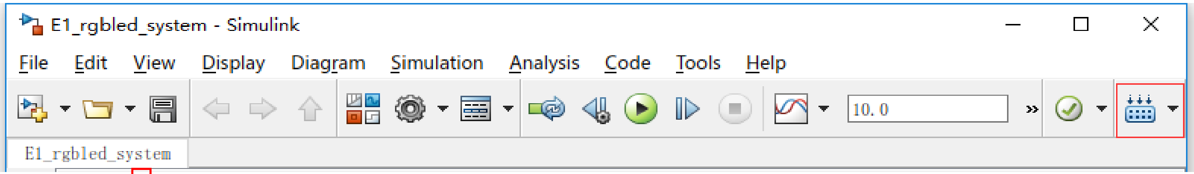
Figure. Logging block in Simulink model “acquire_data_ag.slx”Click

Basic Experiment

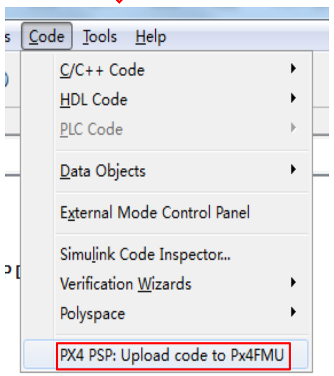
(1) Step1： Obtain accelerometer data via the Pixhawk autopilot system

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

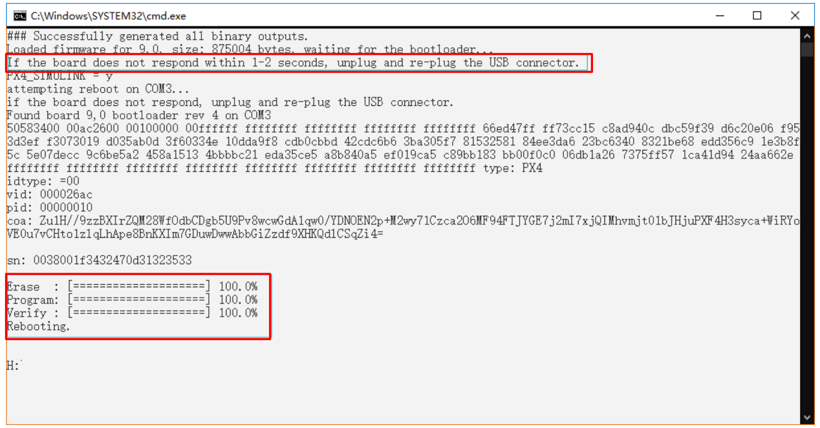
Data can be logged automatically, by placing the upper-left stick (CH5) in the corresponding position.



Click to build



Click to download



```
### Successfully generated all binary outputs.
Loaded firmware for 9,0, size: 875004 bytes, waiting for the bootloader...
If the board does not respond within 1-2 seconds, unplug and re-plug the USB connector.
PX4_SIMULINK = y
attempting reboot on COM3...
if the board does not respond, unplug and re-plug the USB connector.
Found board 9,0 bootloader rev 4 on COM3
50583400 00ac2600 00100000 00ffffff ffffffff ffffffff ffffffff ffffffff 66ed47ff ff73cc15 c8ad940c dbc59f39 d6c20e06 f95
3d3ef f3073019 d035ab0d 3f60334e 10dda9f8 cdb0cbbd 42cdc6b6 3ba305f7 81532581 84ee3da6 23bc6340 8321be68 edd356c9 1e3b8f
5c 5e07decc 9c6be5a2 458a1513 4bbbbc21 eda35ce5 a8b840a5 ef019ca5 c89bb183 bb00f0c0 06db1a26 7375ff57 1ca41d94 24aa662e
ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff type: PX4
idtype: =00
vid: 000026ac
pid: 00000010
coa: Zu1H//9zzBXIrZQM28Wf0dbCDgb5U9Pv8wcwGdA1qw0/YDNOEN2p+M2wy71Czca206MF94FTJYGE7j2mI7xjQIMhvmjt01bJHjuPXF4H3syca+WiRYo
VE0u7vCHto1z1qLhApe8BnKXIm7GDuwDwwAbbGiZzdf9XHKQd1CSqZi4=

sn: 0038001f3432470d31323533

Erase  : [===================] 100.0%
Program: [===================] 100.0%
Verify : [===================] 100.0%
Rebooting.


H:
```

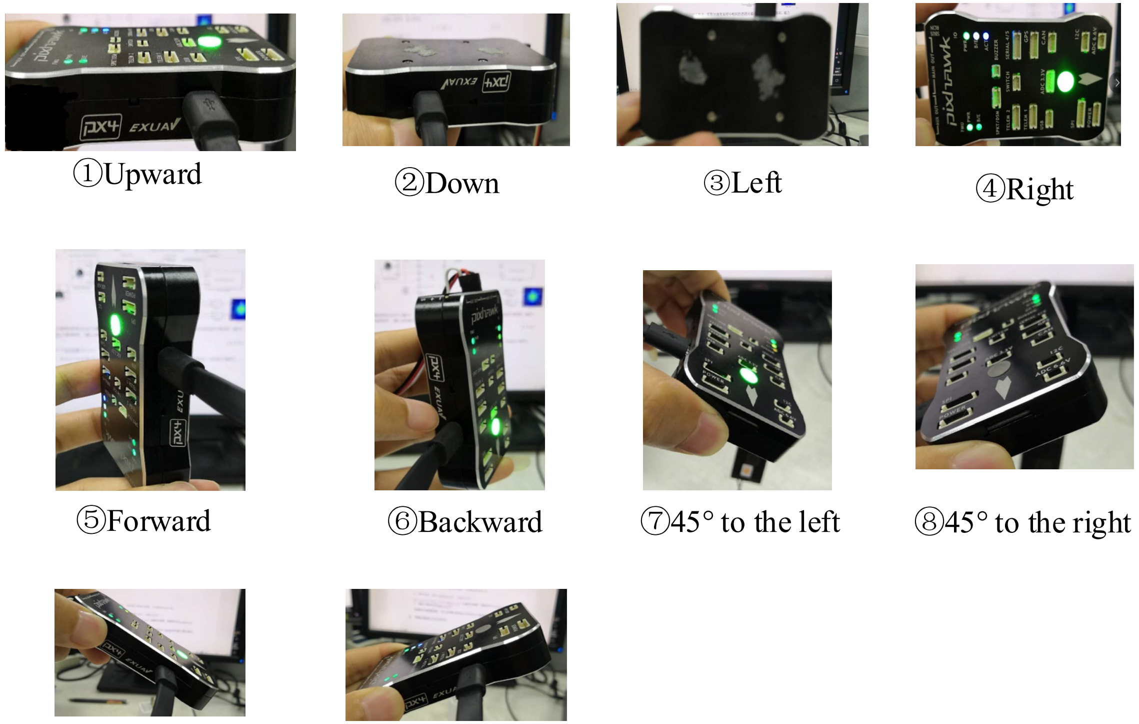
Download completed

Basic Experiment

(1) Step1：Obtain accelerometer data via the Pixhawk autopilot system

4) Rotate the Pixhawk autopilot to log data

Pull back the upper-left switch corresponding to CH5>1500 , to start writing data to the SD card. Place the Pixhawk autopilot as guided by the right figure and hold the Pixhawk autopilot still with each orientation for a period of time. Meanwhile, the Pixhawk autopilot logs data to a file called “e3_A.bin” on the SD card. Once a feature point is collected, the Pixhawk LED status light will slowly blink in red. By recalling the feature point collection method, one feature point corresponds to one orientation that the Pixhawk is placed at. Repeat the logging process for all orientations. Once ten feature points corresponding to ten orientations are collected, the Pixhawk LED status light will begin quickly blinking in red. Then, pull forward the upper-left.



①Upward ②Down ③Left ④Right

⑤Forward ⑥Backward ⑦45° to the left ⑧45° to the right

⑨45° to the forward ⑩45° to the backward

Figure. Ten different orientations of Pixhawk autopilot

Basic Experiment

(1) Step1：Obtain accelerometer data via the Pixhawk autopilot system

5) Read data

Take out the SD card, read the data by a card reader, copy the file “e3_A.bin”to the folder “e3\e3.1”. Use the function

[datapoints, numpoints] = px4_read_binary_file(‘e_acc1_A.bin’)

to decode the data. The data are saved in "datapoints" and the number of the data is saved in "numpoints". The x-axis accelerometer sampling data and feature data are shown in the right figure.

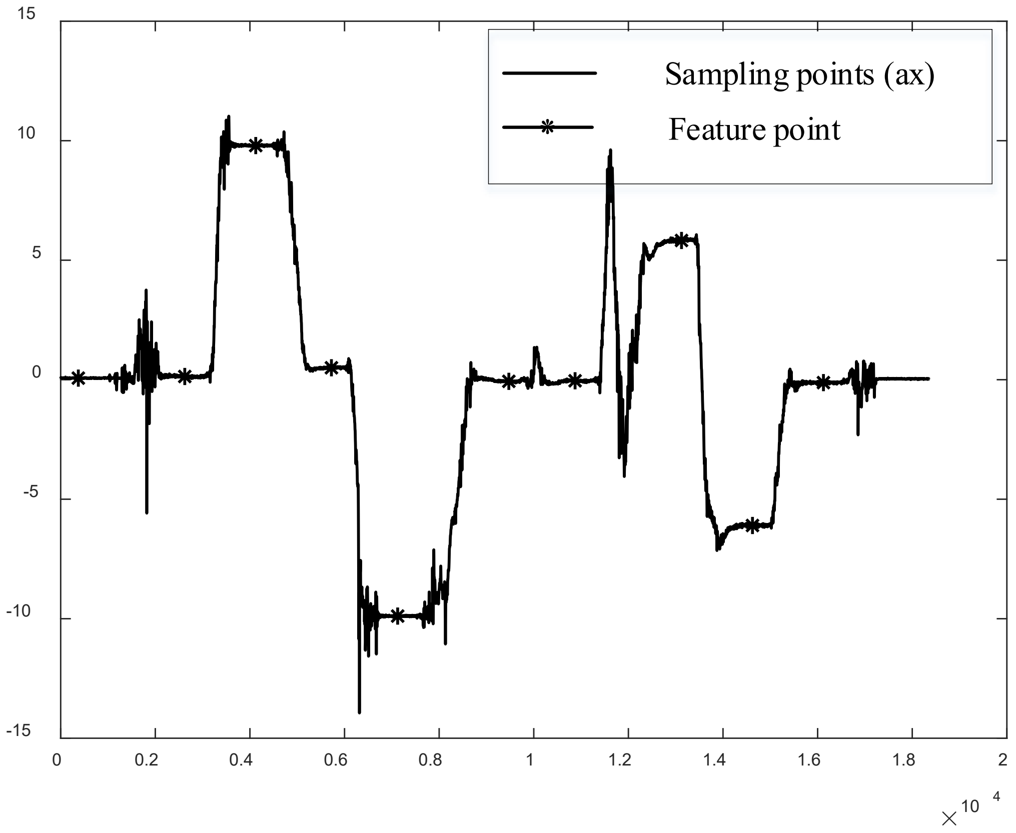
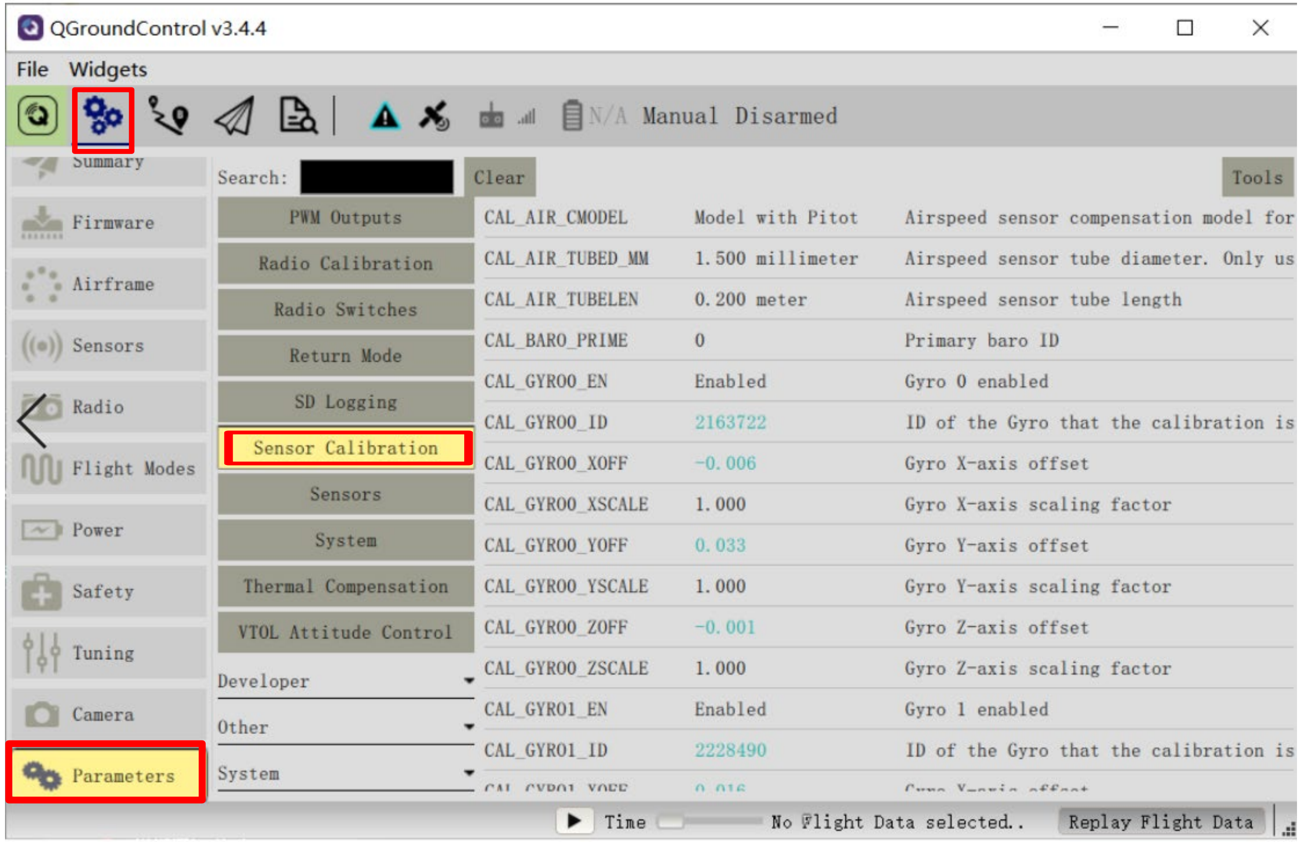


Figure. x-axis accelerometer sampling data and feature data

Basic Experiment

(2) Step2：Parameter calibration

To obtain uncalibrated original data, the accelerometer calibration parameters in the Pixhawk autopilot are first read by QGC. In QGC, select “Parameters” – “Sensor Calibration” to obtain the accelerometer calibration parameters, as shown in the right figure. Then, the data are restored to uncalibrated data, based on the obtained accelerometer calibration parameters from QGC.



CAL_AIR_CMODEL	Model with Pitot	Airspeed sensor compensation model for
CAL_AIR_TUBED_MM	1.500 millimeter	Airspeed sensor tube diameter. Only us
CAL_AIR_TUBELEN	0.200 meter	Airspeed sensor tube length
CAL_BARO_PRIME	0	Primary baro ID
CAL_GYRO0_EN	Enabled	Gyro 0 enabled
CAL_GYRO0_ID	2163722	ID of the Gyro that the calibration is
CAL_GYRO0_XOFF	-0.006	Gyro X-axis offset
CAL_GYRO0_XSCALE	1.000	Gyro X-axis scaling factor
CAL_GYRO0_YOFF	0.033	Gyro Y-axis offset
CAL_GYRO0_YSCALE	1.000	Gyro Y-axis scaling factor
CAL_GYRO0_ZOFF	-0.001	Gyro Z-axis offset
CAL_GYRO0_ZSCALE	1.000	Gyro Z-axis scaling factor
CAL_GYRO1_EN	Enabled	Gyro 1 enabled
CAL_GYRO1_ID	2228490	ID of the Gyro that the calibration is

Figure. Calibration parameter in QGC

Basic Experiment

(2) Step2： Parameter calibration

1) The usage of function

p = lm(func,p,x,y_dat,dp,p_min,p_max)

- **Input parameters：**
- **func function name y_hat = func(x,p)， the functional relationship is** $\left\|\mathbf{T}_{\mathrm{a}}\mathbf{K}_{\mathrm{a}}\left({}^{\mathrm{b}}\mathbf{a}'_{\mathrm{m}}+\mathbf{b}'_{a}\right)\right\|$ **, where** $\mathbf{T}_{\mathrm{a}}=\mathbf{I}_3$ **.**
- **p: the initial value of the parameter to be estimated, which denotes the initial value in optimization;**
- **x: feature points;**
- **dp:related to the Jacobian matrix;**
- **p_min： the minimum norm of the unknown parameter, the default is -100*abs(p);**
- **p_max: the maximum norm of the unknown parameter, the default is 100*abs(p);**

- **Output**
- **p: The estimated parameter value by the algorithm iteration, i.e.** $\boldsymbol{\Theta}_{\mathrm{a}}$ **.**

Basic Experiment

(2) Step2：Parameter calibration

2) Calibrate the accelerometer by the LM algorithm. The main code is shown in following table.

```
%File Description:
% According to the accelerometer error model, the accelerometer error model parameters are calculated using the lm optimization algorithm.
close all
clc
clear

load AccRaw  %Load uncalibrated accelerometer data
g = 9.8;
m = length(AccRaw);

y_dat = g*ones(m, 1);  %Expected gravitational acceleration data
p0 = [1 1 1 0 0 0]';
p_init = [1.0 1.0 1.0 0.1 0.1 0.1]';  %Accelerometer error model parameter initial data
y_raw = calFunc(AccRaw, p0);  %2−norm of uncalibrated accelerometer dara
y_raw = y_raw(:);
r_raw = y_dat - y_raw;   %The difference between the uncalibrated gravitational acceleration measured by the accelerometer and the standard gravitational acceleration
p_fit = lm('calFunc', p_init, AccRaw, y_dat);
y_lm = calFunc(AccRaw, p_fit);  %2−norm of calibrated accelerometer
value
y_lm = y_lm(:);
r_lm = y_dat - y_lm;
kx = p_fit(1);
ky = p_fit(2);
kz = p_fit(3);
bx = p_fit(4);
by = p_fit(5);
bz = p_fit(6);
Ka9_8 = [kx 0 0; 0 ky 0; 0 0 kz]
ba9_8 = [bx by bz]'
save('calP9_8', 'Ka9_8', 'ba9_8')
```

Basic Experiment

(2) Step2：Parameter calibration

Moreover, as shown in the following figure, the optimization objective converges to zero very quickly as the iterative number is increased with the calibrated parameters obtained as

$$\mathbf{K}_{\mathrm{a}}^{*}=\begin{bmatrix}0.9912 & 0 & 0\\ 0 & 0.9974 & 0\\ 0 & 0 & 0.9947\end{bmatrix}$$

$$\mathbf{b}_{\mathrm{a}}^{*}=\begin{bmatrix}0.0168\\ 0.2691\\ 0.1253\end{bmatrix}$$

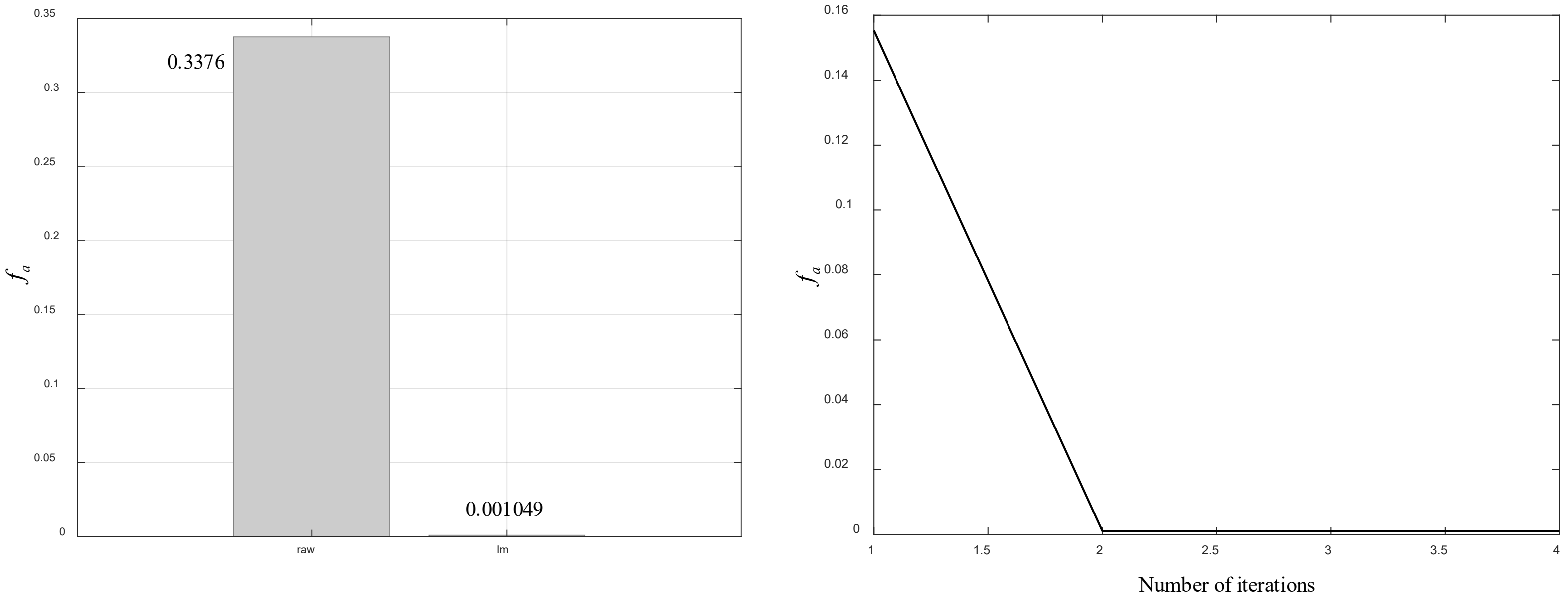


Figure. Value of f_{a} of calibrated and uncalibrated accelerameters

Analysis Experiment

☐ Experimental Objective

■ Things to prepare

The restored acceleration data collected from the basic experiments.

■ Objectives

Change the value of the gravitational acceleration from 9.8 to 1. Calibrate the accelerometer parameters again; with the calibrated data, calculate the pitch angle. Compare and analyze the calibration parameters and pitch angles computed based on different calibrated parameters when the gravitational acceleration is set from 9.8 to 1.

Analysis Experiment

□ Experimental Analysis

The optimization of $\boldsymbol{\Theta}_{\mathrm{a}}^{*}$ is the desired objective. Then

$$\left.\frac{\partial\sum_{k=1}^{M}\left(\left\|\mathbf{T}_{\mathrm{a}}\mathbf{K}_{\mathrm{a}}^{*}\left({}^{\mathrm{b}}\mathbf{a}'_{\mathrm{m}}+\mathbf{b}'_{\mathrm{m}}\right)\right\|-g\right)^{2}}{\partial\boldsymbol{\Theta}_{\mathrm{a}}}\right|_{\boldsymbol{\Theta}_{\mathrm{a}}=\boldsymbol{\Theta}_{\mathrm{a}}^{*}}=\mathbf{0}$$

where $\mathbf{T}_{\mathrm{a}}=\mathbf{I}_{3}$. In that regard, the gravitational acceleration changes. For example, $\alpha g,\alpha>0$ is changed to g. Multiplying the left and right sides of

The equation by α yields

$$\left.\frac{\partial\sum_{k=1}^{M}\left(\left\|\mathbf{T}_{\mathrm{a}}\alpha\mathbf{K}_{\mathrm{a}}^{*}\left({}^{\mathrm{b}}\mathbf{a}'_{\mathrm{m}}+\mathbf{b}'_{\mathrm{m}}\right)\right\|-\alpha g\right)^{2}}{\partial\boldsymbol{\Theta}_{\mathrm{a}}}\right|_{\boldsymbol{\Theta}_{\mathrm{a}}=\boldsymbol{\Theta}_{\mathrm{a}}^{*}}=\mathbf{0}$$

This implies that, after the gravitational acceleration is changed, the calibration parameters are $\mathbf{K}_{\mathrm{a}}^{*\prime}=\alpha\mathbf{K}_{\mathrm{a}}^{*}$ and $\mathbf{b}_{\mathrm{a}}^{*\prime}=\mathbf{b}_{\mathrm{a}}^{*}$

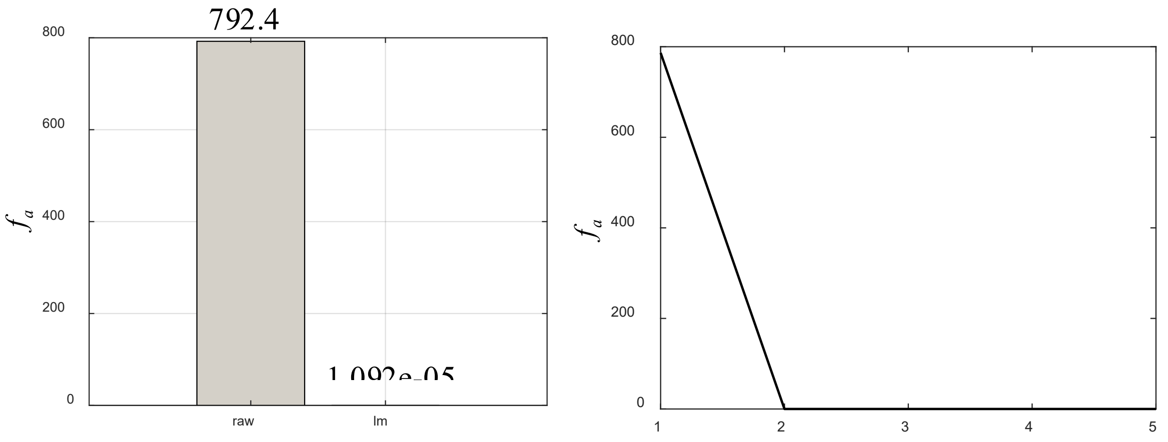
Analysis Experiment

☐ Experimental Procedure

(1) Step1: Open the file “calLM.m” and change the value of gravitational acceleration “g” from 9.8 to 1.

(2) Step2: Copy the sensor data obtained in the basic experiment for file “calLM.m”.

(3) Step3: Run the file “calLM.m” to obtain the calibration results and curves.



(a) Values of optimization objective f_a using calibrated and uncalibrated accelerameters

(b) Value of optimization objective f_a changes with number of iterations

Figure. Calibration results when “g” is 1

Analysis Experiment

□ Calibration Procedure

(3) Step3: Run the file “calLM.m” to obtain the calibration results and curves

- g=9. 8

$$\mathbf{K}_a^{*\prime} = \begin{bmatrix} 0.9912 & 0 & 0 \\ 0 & 0.9974 & 0 \\ 0 & 0 & 0.9947 \end{bmatrix} \quad \mathbf{b}_a^{*\prime} = \begin{bmatrix} 0.0168 \\ 0.2691 \\ 0.1253 \end{bmatrix}$$

- g=1

$$\mathbf{K}_a^{*\prime} = \begin{bmatrix} 0.1012 & 0 & 0 \\ 0 & 0.1017 & 0 \\ 0 & 0 & 0.1014 \end{bmatrix} \quad \mathbf{b}_a^{*\prime} = \begin{bmatrix} 0.0123 \\ 0.2771 \\ 0.1456 \end{bmatrix}$$

One can conclude that when “g” is 1 and 9.8, $\mathbf{K}_a^{*\prime}$ is reduced to approximately 1/9.8 of $\mathbf{K}_a^{*}$, which is consistent with the theoretical analysis.

Analysis Experiment

☐ Calibration Procedure

(3) Step3: Run the file "calLM.m" to obtain the calibration results and curves

The three-axis accelerometer is fixed to the multicopter and aligned with the aircraft-body coordinate frame. To better illustrate the pitch angle in a continuous process, a set of accelerometer data is logged again with the Pixhawk autopilot being slowly turned, as shown in the right figure. One can conclude that the two calibration solutions result in the same angle. This implies that the pitch angle measurement is independent of the acceleration of gravity.

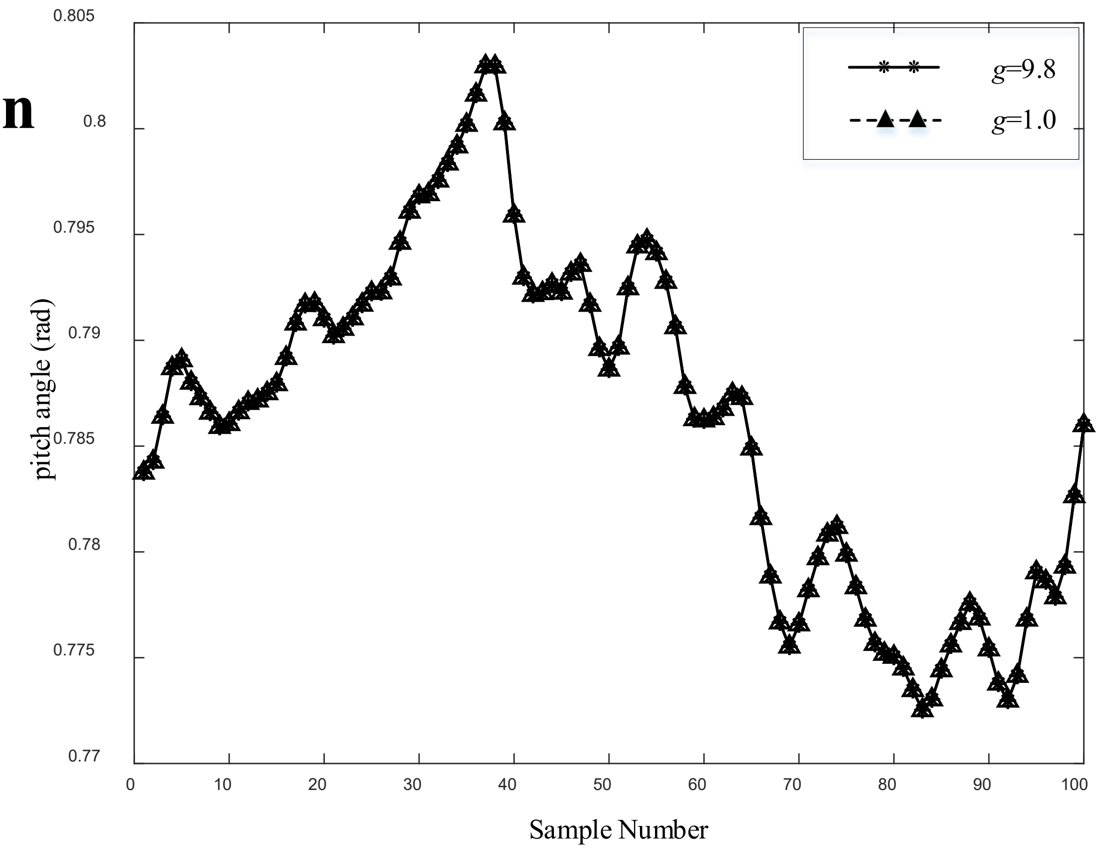


Figure. Pitch angle with respect two different "g" values

Design Experiment

□ Experimental Objective

■ Things to prepare

(1) Hardware: Pixhawk Autopilot System;

(2) Software: MATLAB R2017b or above, Pixhawk Support Package(PSP) Toolbox, QGroundControl(QGC) and Instruction Package “e3.3” (https://rflysim.com/course);

(3) Data for calibration are prepared in Instructional Package “e3.3” for readers without hardware to collect data.

■ Objectives

Design the magnetometer data logging block, following the procedure in the basic experiment. With the obtained data, calibrate the magnetometer, and compare the calibrated and uncalibrated results.

Design Experiment

☐ Experimental Procedure

(1) Step1: Obtain magnetometer data from Pixhawk Autopilot

1) Data logging block

Create a new Simulink file and drag out the corresponding modules from the Simulink PSP Toolbox, as shown in the right figure. To log data, use the corresponding blocks in “Pixhawk Target Blocks” that log data from the inertial sensor and RC transmitter. That data can be saved into the Pixhawk SD card.

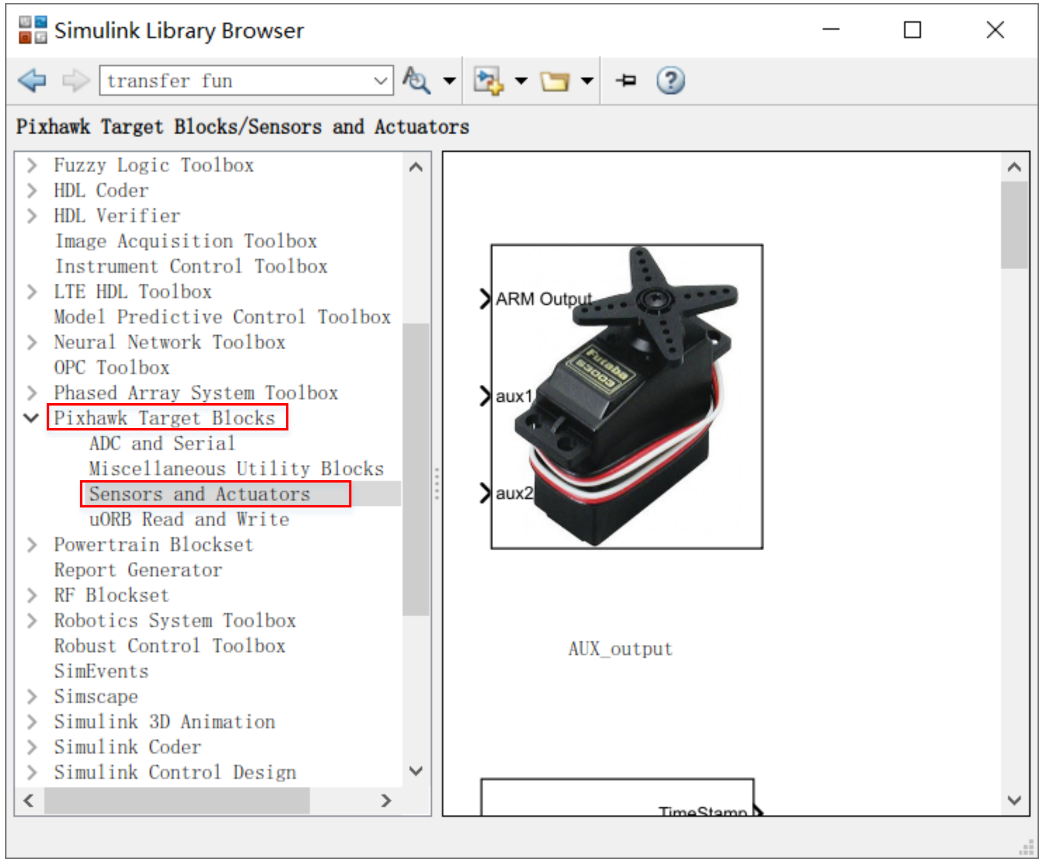


Figure. PSP toolbox in Simulink library browser

Design Experiment

□ Experimental Procedure

(1) Step1: Obtain magnetometer data from Pixhawk Autopilot

1) Data logging block

An appropriate model given in the file "acquire_data_mag.slx" is shown in the right figure.

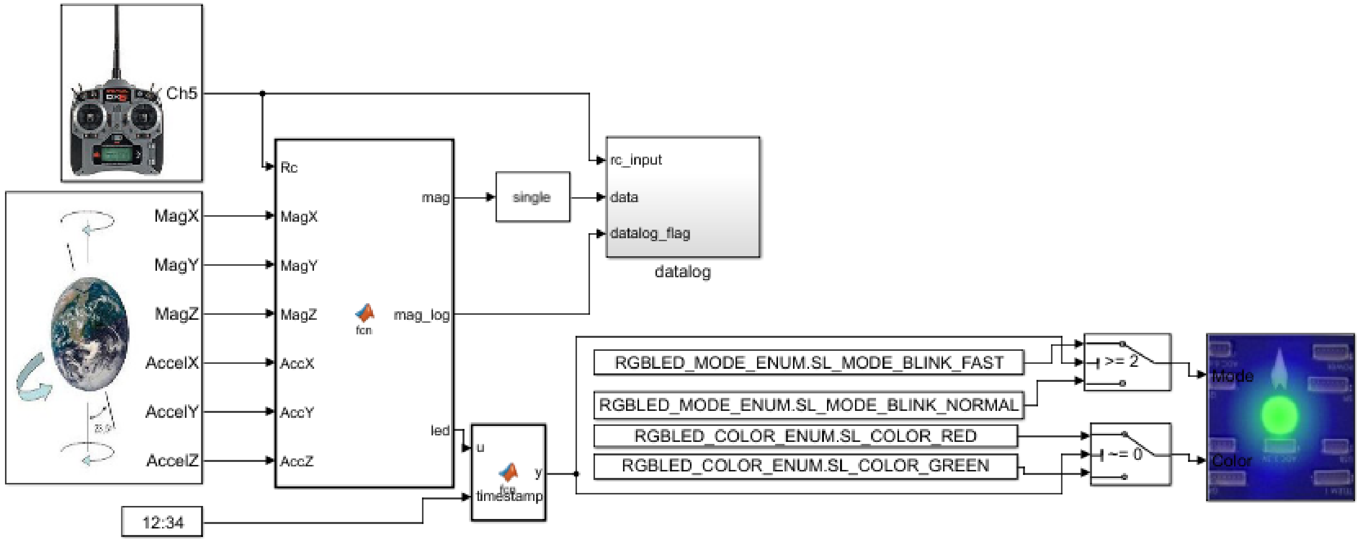


Figure. Magnetometer data logging, Simulink model "acquire_data_mag.slx"

Design Experiment

□ Experimental Procedure

(1) Step1: Obtain magnetometer data from Pixhawk Autopilot

2) Hardware connection

The connection between the RC receiver and the Pixhawk autopilot can be determined, as is shown in the right figure.

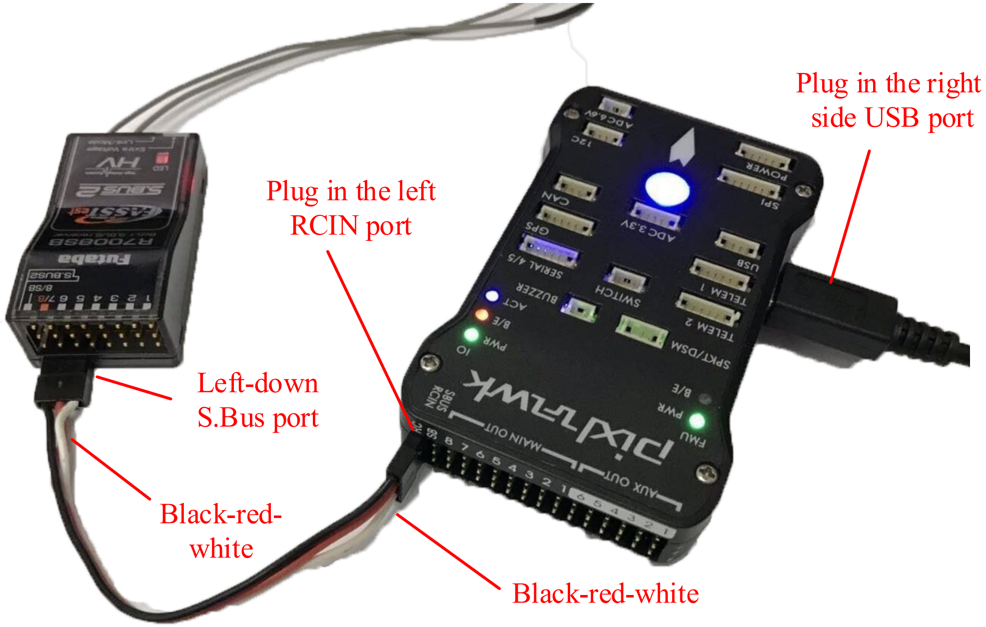


Figure. Pixhawk and RC transmitter connection

Design Experiment

□ Experimental Procedure

(1) Step1: Obtain magnetometer data from Pixhawk Autopilot

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

4) Log the data

When the data of the magnetometer is logged, the average data is not used to eliminate data noise; rather, a significant amount of raw data is used to eliminate noise. The reason for this is that when the data of the magnetometer is logged, the Pixhawk autopilot should be shaken, which will cause extra acceleration, but will not change the magnetic field around the autopilot. Thus, to obtain sufficient magnetometer data, the Pixhawk autopilot can be randomly shaken.

Design Experiment

□ Experimental Procedure

(1) Step1: Obtain magnetometer data from Pixhawk Autopilot

4) Log the data

Pull back the upper-left switch corresponding to CH5>1500, to start writing data to the SD card. Place the Pixhawk autopilot as guided by the right figure. Starting at each orientation, rotate the Pixhawk autopilot a circle clockwise or counterclockwise around its principal axes of the moment of inertia, where 40 sampling data are logged. Meanwhile, the Pixhawk autopilot logs data to a file called “e3_m_A.bin” on the SD card. Once the process of logging is completed for the current orientation, the Pixhawk LED status light will be slowly blinking in red. Then, repeat the logging process for all orientations. Once all data corresponding to all six orientation is collected, the Pixhawk LED status light will be quickly blinking in red and a total 240 sampling data are logged. Then, pull forward the upper-left switch (CH5<1500) to stop writing data to the SD card.

①Upward

②Down

③Left

④Right

⑤Upward

⑥Down

Figure. Pixhawk autopilot placement facing six different directions

Design Experiment

（1） Step1: Obtain magnetometer data from Pixhawk Autopilot

5) Read the data

First, remove the microSD card from Pixhawk autopilot. Read the data using a card reader. Copy the file “e3_m_A.bin” to the folder “e3\e3.3” and save it. Use the function

[datapoints, numpoints] = px4_read_binary_file(‘e3_m_A.bin’)

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

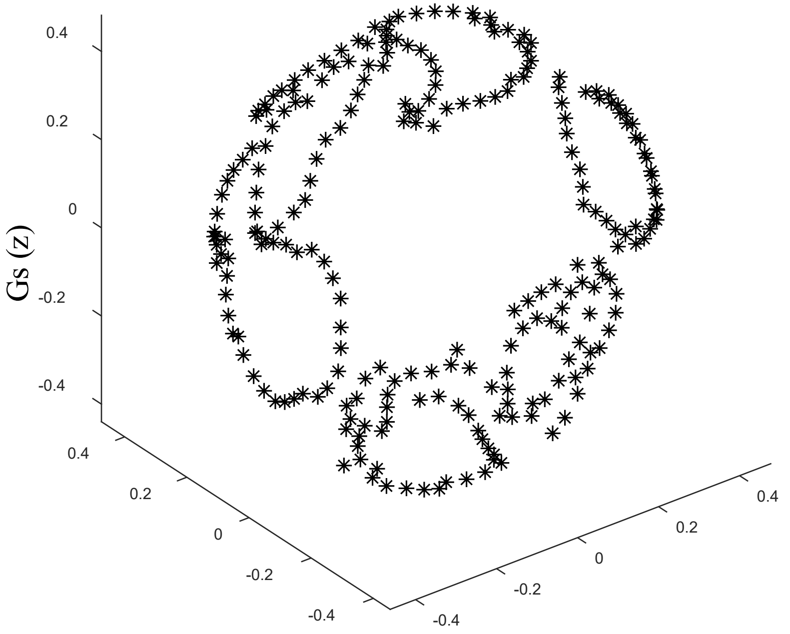


Figure. Magnetometer calibration sampling point

Design Experiment

Calibrate the accelerometer by the LM algorithm. The main code is shown in following table.

```
close all
clc
clear
load MagRaw.mat
CAL_MAG_SCALE = [1, 1, 1]';  %Calibration value in autopilot
CAL_MAG_OFF = [0.064, 0.014, -0.053]';
MagRaw = (mag + CAL_MAG_OFF)./CAL_MAG_SCALE;  %Original magnetometer data
m = length(MagRaw);
MagSum = 0;
for k = 1 : m
   MagSum = MagSum + norm(MagRaw(:, k));
end
MagAver = MagSum/m;  %Estimated magnetic field strength
Vdata = MagRaw/MagAver;  %Normalization

y_dat = ones(m, 1);
p0 = [1 1 1 0 0 0]';

p_init = [1 1 1 0.01 0.01 0.01]';  %Initial value of the parameter to be estimated
y_raw = calFunc(Vdata, p0);  %2−norm of uncalibrated magnetometer value
y_raw = y_raw(:);
r_raw = y_dat - y_raw;
p_fit = lm('calFunc', p_init, Vdata, y_dat, 0.001);
y_lm = calFunc(Vdata, p_fit);  %2−norm of calibrated magnetometer value
y_lm = y_lm(:);
r_lm = y_dat - y_lm;
y_px4 = calFunc(mag/MagAver, p0);  %2−norm of PX4 Calibrated magnetometer value
y_px4 = y_px4(:);
r_px4 = y_dat - y_px4;
kx = p_fit(1);
ky = p_fit(2);
kz = p_fit(3);
bx = p_fit(4);
by = p_fit(5);
bz = p_fit(6);
Km = [kx 0 0;0 ky 0;0 0 kz]
bm = [bx by bz]'
```

Design Experiment

□ Experimental Procedure

(2) Parameter calibration

As shown in the following figure, the optimization objective becomes smaller than that for uncalibrated parameters, and the optimization objective converges to 0.5 very quickly as the iterative number is increased. Finally, the calibrated parameters are obtained as

$$\mathbf{K}_{\mathrm{m}}^{*}=\begin{bmatrix}0.9853 & 0 & 0\\ 0 & 1.0202 & 0\\ 0 & 0 & 1.0004\end{bmatrix}$$

$$\mathbf{b}_{\mathrm{m}}^{*}=\begin{bmatrix}-0.1448\\ -0.0334\\ -0.0898\end{bmatrix}$$

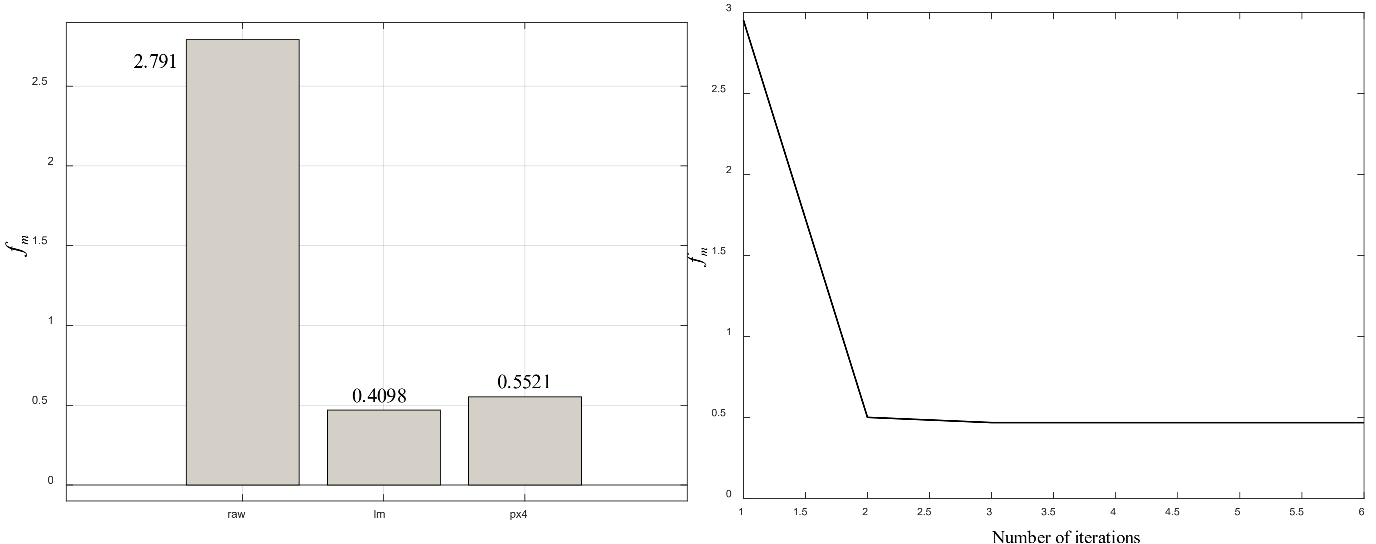


Figure. Values of optimization objective f_{m} using calibrated, uncalibrated magnetometers and PX4

Summary

(1) An accelerometer calibration model and a magnetometer calibration model are illustrated using the multicopter sensor calibration experiment, where the PSP Toolbox is used for data logging.

(2) When recording accelerometer data, to avoid additional non-gravity acceleration, it is necessary to hold the Pixhawk still. To reduce the external acceleration noise, some extracted feature points are used for calibration optimization, rather than all sample points. The calibration results are satisfactory.

(3) After the gravity acceleration “g” is changed from 9.8 to 1, the same code from the basic experiment is executed again. The results show that the scale factor Ka narrows 1/9.8, but the attitude angle is consistent with that when “g” is 9.8. This implies that the pitch angle measurement is independent of the acceleration of gravity.

(4) As for recording the magnetometer data, the Pixhawk autopilot is rotated starting from six different orientations. The calibration results are satisfactory.

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