

کمارینی مل

ناؤبری تائیع

اڈدیں سر کار ل

تاریخ تحول : ۱۴۰۹، ۲، ۲۰

۴۰۳۱۵۹۹||

امدادی احمد

۱- الگوریتم کالیبراسیون با روش حداقل مربعات یا Least Square را برای یک IMU که دارای سه کanal شتابنگ و سه کanal ژایروسکوپ میباشد پیاده سازی کنید.

۲- عملکرد الگوریتم پیادهسازی شده را با استفاده از ۳ سناریو متشکل از ضرایب، بایاس‌ها و نویزها با واریانس‌های مختلف بررسی کنید. در تمامی سناریوهای ضرایب مربوط به Misalignment در حسگرها، متفاوت از سناریو قبلي و غير صفر در نظر گرفته شوند. انتخاب ضرایب، بایاس‌ها و نویزها در هر سناريو بر عهده دانشجو مي باشد.

الگوریتم می‌بایست در هر ۳ سناریو موفق به شناسایی تمامی پارامترهای کالیبراسیون با خطای کمتر از ۱ درصد، باشد.

در انتخاب ضرایب، موارد زیر مد نظر قرار گیرد:

- مرتبه ضرایب misalignment و scale factor error از  $10^{-5}$  بیشتر نشود.
  - مرتبه بایاس‌های شتاب‌سنج‌ها از  $10^{-3}$  و ژاکوبی‌ها از مقدار  $10^{-9}$  بیشتر نشود.
  - نویزها به صورت گوسی-سفید با مقدار  $3\sigma$  کمتر از مقدار بایاس‌ها انتخاب شوند.
  - هم مقادیر مثبت و هم مقادیر منفی لحاظ شوند.

$$l = 66^\circ$$

$$\lambda = \omega d^o$$

$$h = 1000$$

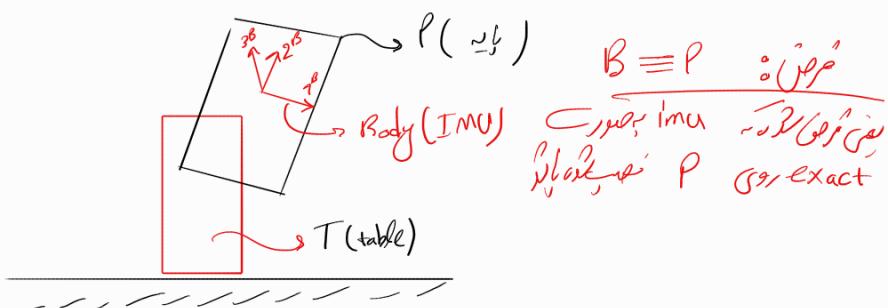
خواصی از جمله مخصوصیت میتواند در مکانیزم این روش کاربرد داشته باشد.

$$\text{Forward T pitch: } \left[ \begin{array}{c} \sin \theta \\ -\cos \theta \\ 0 \end{array} \right]$$

$$T = N \text{ over } D_L$$

$$[T]^N = [T]^E =$$

$$\left[ \begin{array}{ccc} \sin \alpha & -\sin \beta & \cos \gamma \\ \sin \beta & \cos \alpha & 0 \\ -\cos \alpha & -\cos \beta & \sin \gamma \end{array} \right]$$



جیسا کو Yaw → roll ہے جو سے یہ ہے۔

$$[T]^{BT} = [T] = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}$$

لما زادت قوى المقاومة في المقطع المركب  $[f_B^I]^B$  فيكون

$$D^I v_B^I = f_B^I + G$$

$$v_B^I = D^I S_{BI} = D^E S_{BI} + \Omega^{EI} S_{BI} = v_B^E + \Omega^{EI} S_{BE}$$

$$D^I v_B^I = D^E v_B^E + \Omega^{EI} v_B^E$$

$$= D^E v_B^E + D^E \{ \Omega^{EI} S_{BE} \} + \Omega^{EI} v_B^E + \Omega^{EI} \Omega^{EI} S_{BE}$$

$$= D^E v_B^E + \Omega^{EI} v_B^E + \Omega^{EI} \Omega^{EI} S_{BE} = f_B^I + G$$

$$\rightarrow D^E v_B^E + \Omega^{EI} v_B^E = f_B^I + G - \underbrace{\Omega^{EI} \Omega^{EI} S_{BE}}_g$$

$$\rightarrow D^E v_B^E + \Omega^{EI} v_B^E = f_B^I + g$$

$$D^E v_B^E = D^N v_B^E + \Omega^{NE} v_B^E$$

$$\rightarrow D^N v_B^E = f_B^I + g - \Omega^{EI} v_B^E - \Omega^{NE} v_B^E$$

$$\rightarrow [D^N v_B^E]^\sim = [\bar{T}]^{NB} [f_B^I]^\beta + [g]^\sim - [\bar{T}]^{NE} [\Omega^{EI}]^\beta [\bar{T}]^{NE} [v_B^E]^\sim [-\Omega^{NE}]^\sim [v_B^E]^\sim$$

لما زادت قوى المقاومة في المقطع المركب  $v_B^E = 0$

$$0 = [\bar{T}]^{NB} [f_B^I]^\beta + [g]^\sim \Rightarrow [\bar{T}]^{NB} [f_B^I]^\beta = -[g]^\sim$$

$$\underbrace{[\bar{T}]^{NB}}_I \underbrace{[\bar{T}]^{NB} [f_B^I]^\beta}_{-} = -[\bar{T}]^{NB} [g]^\sim$$

$$[\bar{T}]^{NB} [f_B^I]^\beta = -[\bar{T}]^{TB} [g]^\sim = -[\bar{T}]^{BT} [g]^\sim = \begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix}$$

$\therefore$   $N \equiv T$   $\therefore$   $T$   $\therefore$   $T$   $\therefore$   $T$

مقدمة في الميكانيكا المنشآت

$$\begin{cases} \ddot{a}_n = \ddot{a}_n + \mu_{xy} \ddot{a}_y + \mu_{xz} \ddot{a}_z + S_{xx} \ddot{a}_n + n_x + b_n \\ \ddot{a}_y = \ddot{a}_y + \mu_{yn} \ddot{a}_n + \mu_{yz} \ddot{a}_z + S_{yy} \ddot{a}_y + n_y + b_y \\ \ddot{a}_z = \ddot{a}_z + \mu_{zx} \ddot{a}_n + \mu_{zy} \ddot{a}_y + S_{zz} \ddot{a}_z + n_z + b_z \end{cases}$$

$$\begin{bmatrix} \ddot{a}_n \\ \ddot{a}_y \\ \ddot{a}_z \end{bmatrix} = \underbrace{\begin{bmatrix} 1+S_{nn} & \mu_{ny} & \mu_{nz} \\ \mu_{yn} & 1+S_{yy} & \mu_{yz} \\ \mu_{zn} & \mu_{zy} & 1+S_{zz} \end{bmatrix}}_M \begin{bmatrix} \ddot{a}_n \\ \ddot{a}_y \\ \ddot{a}_z \end{bmatrix} + \begin{bmatrix} b_n \\ b_y \\ b_z \end{bmatrix} + \begin{bmatrix} n_n \\ n_y \\ n_z \end{bmatrix}$$

الحالات المستقرة  $\ddot{a}_n = 0$

$$Q = 1, 11, 15, 19, 25$$

الحالات غير المستقرة  $\ddot{a}_n \neq 0$

```

clear;
clc;

%% define latitude and longitude
lat = deg2rad(35);
lon = deg2rad(55);
h = 1000;

%% define yaw and phi position
si_positions = [10, 110, 250, 270, 30, 150];
phi_positions = [-60, -30, 30, 60];
num_samples_in_each_si_phi_pair = 1000000;
recorded_accel_ouptuts = zeros(num_samples_in_each_si_phi_pair, 3);
measured_accel_outputs = zeros(length(si_positions) * length(phi_positions), 3);
true_accel_outputs = zeros(length(si_positions) * length(phi_positions), 3);

index = 1;

for i = 1:length(si_positions)
    for j = 1:length(phi_positions)
        %% define si and phi position
        si = si_positions(i);
        phi = phi_positions(j);

        %% calculating the [g]
        R0 = 6378137;
        Rp = 6356752.31425;
        wei = 2 * pi / (24 * 3600);
        e = 0.0818191908425;
        f = 1 / (298.257223563);
        mu = 3.986004418 * 1e14;
        Rn = R0 / sqrt(1 - e^2 * sin(lat)^2);
        Rm = R0 * (1 - e^2) / (1 - e^2 * sin(lat)^2)^(3/2);
        g0 = 9.7803253359 * (1 + 0.001931853 * sin(lat)^2) / sqrt(1 - e^2 * sin(lat)^2);
        q = 1 + (wei^2 * R0^2 * Rp / mu) + (1 - 2 * sin(lat)^2) * f;
        p = 1 - (2 * q * h / R0) + (3 * h^2 / R0^2);
        g = [-8.08 * 1e-9 * h * sin(2 * lat);
              0;
              p * g0];

        %% calculating ideal accel output
        T_bt = [cos(si)      sin(si)      0;
                 -sin(si)    cos(si)      0;
                 0           0           1] * [1           0           0;
                                         0           cos(phi)    sin(phi);
                                         0           -sin(phi)   cos(phi)];
        f_bi_b = -T_bt * g;

        %% simulating uncalibrated accel output
        M = [(1 + 7e-5),      -5e-5,      7e-5;
               -5e-5,      (1 - 9e-5),    -8e-5;
               7e-5,      -8e-5,      (1 + 8e-5)];

        b_a_b = [-9e-3;
                  -8e-3;
                  -7e-3];

        for m = 1:num_samples_in_each_si_phi_pair
            n_a_b = randn(3, 1) .* [1e-3;
                                     2e-3;
                                     2e-3];
            recorded_accel_ouptuts(m, :) = transpose(M * f_bi_b + b_a_b + n_a_b);
        end
        measured_accel_outputs(index, :) = mean(recorded_accel_ouptuts);
        true_accel_outputs(index, :) = transpose(f_bi_b);
        index = index + 1;
    end
end

```

```

%% solve the calibration problem for three channels of accel
H = [ones(size(true_accel_outputs, 1), 1), true_accel_outputs];
pesudu_inverse = inv(transpose(H) * H) * transpose(H);

% bias vector and M matrix
calibration_matrix = pesudu_inverse * measured_accel_outputs;
bias_calibration = calibration_matrix(1, :);
M_calibration = calibration_matrix(2:4, :);

% Compute percentage error for bias
bias_error_percent = 100 * (bias_calibration - b_a_b') ./ b_a_b';

% Compute percentage error for each element of M
M_error_percent = 100 * (M_calibration - M) ./ M;

% Display results
disp('Accel Bias estimation error in percentage:');
disp(array2table(bias_error_percent, 'VariableNames', {'X', 'Y'});

disp('Accel M matrix estimation error in percentage:');
disp(array2table(M_error_percent, ...
    'VariableNames', {'X', 'Y', 'Z'}, ...
    'RowNames', {'X_row', 'Y_row', 'Z_row'}));

```

$$M = \begin{bmatrix} (1 + \gamma x l_0^{-\alpha}) & -\tilde{r} x l_0^{-\alpha} & -\gamma x l_0^{-\alpha} \\ -\tilde{r} x l_0^{-\alpha} & (1 - \Delta x l_0^{-\alpha}) & \gamma x l_0^{-\alpha} \\ -\gamma x l_0^{-\alpha} & \tilde{r} x l_0^{-\alpha} & (1 + \sqrt{\gamma} x l_0^{-\alpha}) \end{bmatrix}$$

$$[b_a]^B = \begin{bmatrix} q \times l_o^{-r} \\ -\lambda \times l_o^{-r} \\ -V \times l_o^{-r} \end{bmatrix}$$

### Accel Bias estimation error in percentage:

X Y Z

-0.00021683 -0.0030034 0.0099253

Accel M matrix estimation error in percentage

X Y Z

X_row	-1.0083e-05	0.14925	-0.25176
Y_row	-0.25999	7.3992e-07	0.48644
Z_row	0.29213	-0.018968	-8.7338e-06

و مقداری در سایر این موارد که در روش این مکارهای طنزی نموده اند.

: (1) > (2)

$$M = \begin{bmatrix} (1 + \kappa x l_0^{-\alpha}) & \kappa x l_0^{-\alpha} & -\varepsilon x l_0^{-\alpha} \\ \kappa x l_0^{-\alpha} & (1 + \varepsilon x l_0^{-\alpha}) & \nu x l_0^{-\alpha} \\ -\varepsilon x l_0^{-\alpha} & \nu x l_0^{-\alpha} & (1 - \Delta x l_0^{-\alpha}) \end{bmatrix}$$

$$\begin{bmatrix} b_a \end{bmatrix}^B = \begin{bmatrix} \kappa x l_0^{-\alpha} \\ -\gamma x l_0^{-\alpha} \\ \eta x l_0^{-\alpha} \end{bmatrix} \quad \begin{bmatrix} n_a \end{bmatrix}^B \sim \begin{bmatrix} \mathcal{N}(0, (\frac{\kappa x l_0^{-\alpha}}{\sigma_{a1}})^2) \\ \mathcal{N}(0, (\frac{\nu x l_0^{-\alpha}}{\sigma_{a2}})^2) \\ \mathcal{N}(0, (\frac{1 - \Delta x l_0^{-\alpha}}{\sigma_{a3}})^2) \end{bmatrix}$$

• Col 1 > Col 2 > Col 3, so Q > P

Accel Bias estimation error in percentage:

X	Y	Z
-0.0076772	0.0046387	0.00061143

Accel M matrix estimation error in percentage:

X	Y	Z	
X_row	-2.1328e-06	-0.083891	-0.16702
Y_row	-0.10477	1.8697e-06	0.043018
Z_row	0.056845	-0.053802	3.324e-07

$$M = \begin{bmatrix} (1 + \sqrt{x l_0^{-\alpha}}) & -\Delta x l_0^{-\alpha} & +\sqrt{\nu x l_0^{-\alpha}} \\ -\Delta x l_0^{-\alpha} & (1 - \gamma x l_0^{-\alpha}) & -\eta x l_0^{-\alpha} \\ +\sqrt{\nu x l_0^{-\alpha}} & -\eta x l_0^{-\alpha} & (1 + \lambda x l_0^{-\alpha}) \end{bmatrix}$$

: (P) > (Q)

$$\begin{bmatrix} b_a \end{bmatrix}^B = \begin{bmatrix} -\lambda x l_0^{-\alpha} \\ -\eta x l_0^{-\alpha} \\ \sqrt{\nu x l_0^{-\alpha}} \end{bmatrix} \quad \begin{bmatrix} n_a \end{bmatrix}^B \sim \begin{bmatrix} \mathcal{N}(0, (\frac{-\lambda x l_0^{-\alpha}}{\sigma_{a1}})^2) \\ \mathcal{N}(0, (\frac{-\eta x l_0^{-\alpha}}{\sigma_{a2}})^2) \\ \mathcal{N}(0, (\frac{\sqrt{\nu x l_0^{-\alpha}}}{\sigma_{a3}})^2) \end{bmatrix}$$

Accel Bias estimation error in percentage:

X	Y	Z
7.2276e-05	-0.0060068	0.0099253

Accel M matrix estimation error in percentage:

X	Y	Z	
X_row	-3.3609e-06	0.1791	0.2158
Y_row	-0.051998	1.4799e-06	-0.12161
Z_row	-0.083465	0.009484	-8.7337e-06

مقدمة في الميكانيكا المنشآتية

الوحدة الأولى: المنشآت المترابطة

$$\omega_{BI} = \underbrace{\omega_{BP}}_{\circlearrowleft} + \underbrace{\omega_{PT}}_{\circlearrowleft} + \underbrace{\omega_{TN}}_{\circlearrowleft} + \underbrace{\omega_{NE}}_{\circlearrowleft} + \underbrace{\omega_{EI}}_{\circlearrowleft}$$

$$\rightarrow \omega_{BI} = \omega_{PT}^{PT} + \omega_{EI}^{EI} = \omega_{BT}^{BT} + \omega_{EI}^{EI}$$

$$[\omega_{BI}]^B = [\omega_{PT}]^B + \underbrace{[T]}_I \underbrace{[T]}_I \underbrace{[T]}_{PT} \underbrace{[T]}_{TN} \underbrace{[T]}_{NE} \underbrace{[\omega_{EI}]}_E$$

$$[T] = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & \dots & \dots \\ 0 & \text{case shear} & \\ 0 & -\text{shear case} & \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ 0 \\ \omega_{EI} \end{bmatrix} \xrightarrow{\text{أولاً}} \begin{bmatrix} -\sin\lambda \cos\theta & -\sin\lambda \sin\theta & \cos\lambda \\ -\sin\theta & \cos\theta & 0 \\ -\cos\lambda \cos\theta & -\cos\lambda \sin\theta & -\sin\lambda \end{bmatrix}$$

$$[\omega_{PT}]^B \xrightarrow{\text{ثانياً}} \begin{bmatrix} \dot{\psi} & \dot{\alpha} & \dot{\tau} \end{bmatrix}^T$$

$$\omega_{PT} = \dot{\psi} \beta^P + \dot{\alpha} \tau^P$$

$$[T]^P [x]^P$$

$$[\omega_{PT}]^P = \dot{\psi} [\beta^P]^P + \dot{\alpha} [\tau^P]^P$$

$$= \dot{\psi} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \dot{\alpha} \underbrace{\begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}}_{\begin{bmatrix} \cos\psi \\ -\sin\psi \\ 0 \end{bmatrix}} = \begin{bmatrix} \dot{\psi} \cos\psi \\ -\dot{\alpha} \sin\psi \\ \dot{\psi} \end{bmatrix}$$

$$\rightarrow [\omega_{BI}]^B = \begin{bmatrix} \dot{\psi} \cos\psi \\ -\dot{\alpha} \sin\psi \\ \dot{\psi} \end{bmatrix} + \underbrace{[T]}_I \underbrace{[T]}_{PT} \underbrace{[T]}_{TN} \underbrace{[T]}_{NE} \underbrace{[\omega_{EI}]}_E$$

لذلك،  $\dot{\psi}$  هو المكون المترابط للدوران حول محور  $x$ ،  $\dot{\alpha}$  هو المكون المترابط للدوران حول محور  $y$ ،  $\dot{\tau}$  هو المكون المترابط للدوران حول محور  $z$ .

$$\dot{\psi} = [\pm 1^\circ, \pm 2^\circ, \pm 3^\circ] \frac{\text{deg}}{\text{s}}$$

$$\dot{\alpha} = [\pm 1^\circ, \pm 2^\circ, \pm 3^\circ] \frac{\text{deg}}{\text{s}}$$

$$\begin{cases} \dot{\psi}_{k+1} = \dot{\psi}_k + \dot{\psi} \cdot T_s \\ \dot{\alpha}_{k+1} = \dot{\alpha}_k + \dot{\alpha} \cdot T_s \\ T_s = 0.01 \end{cases}$$

مختصر طلب المنشآت المترابطة

مکانیزم روتوری ایکل ریومنی مکانیزم  $\omega^{BI}$

$$\begin{cases} \ddot{\omega}_x = \hat{\omega}_n + \mu_{xy} \hat{\omega}_y + \mu_{xz} \hat{\omega}_z + S_{xx} \hat{\omega}_x + n_x + b_x \\ \ddot{\omega}_y = \hat{\omega}_y + \mu_{yn} \hat{\omega}_n + \mu_{yz} \hat{\omega}_z + S_{yy} \hat{\omega}_y + n_y + b_y \\ \ddot{\omega}_z = \hat{\omega}_z + \mu_{zn} \hat{\omega}_n + \mu_{zy} \hat{\omega}_y + S_{zz} \hat{\omega}_z + n_z + b_z \end{cases}$$

$$\begin{bmatrix} \ddot{\omega}_x \\ \ddot{\omega}_y \\ \ddot{\omega}_z \end{bmatrix} = \begin{bmatrix} (1+S_{an}) & \mu_{ay} & \mu_{az} \\ \mu_{ayn} & (1+S_{ay}) & \mu_{ayz} \\ \mu_{azn} & \mu_{azy} & (1+S_{az}) \end{bmatrix} \begin{bmatrix} \hat{\omega}_n \\ \hat{\omega}_y \\ \hat{\omega}_z \end{bmatrix} + \begin{bmatrix} b_{\omega_x} \\ b_{\omega_y} \\ b_{\omega_z} \end{bmatrix} + \begin{bmatrix} n_{\omega_n} \\ n_{\omega_y} \\ n_{\omega_z} \end{bmatrix}$$

• عوامل  $\left\{ \dot{\psi} = [\pm 1^\circ, \pm 2^\circ, \pm 3^\circ] \frac{\text{deg}}{\text{sec}}$  داده شده تا در اینجا درست باشند،

$$\left\{ \dot{\phi} = [\pm 1^\circ, \pm 2^\circ, \pm 3^\circ] \frac{\text{deg}}{\text{sec}}$$

```

clear;
clc;
% define earth angular velocity
wei = 2 * pi / (24 * 3600);
Wei_e = [0;
          0;
          wei];
Wei_e

%% define lattitude and longitude
lat = deg2rad(35);
lon = deg2rad(55);

%% define yaw and phi position
si_dot_values = [10, 20, 30, -10, -20, -30]; % deg/sec
phi_dot_values = [10, 20, 30, -10, -20, -30]; % deg/sec
num_samples_in_each_sidot_phidot_pair = 1000; % number of samples that will be record in each si dot and phi dot pairs
Ts = 0.001; % sample time of getting gyro data and table data

measured_gyro_outputs = zeros(length(si_dot_values) * length(phi_dot_values) * num_samples_in_each_sidot_phidot_pair, 3);
true_gyro_outputs = zeros(length(si_dot_values) * length(phi_dot_values) * num_samples_in_each_sidot_phidot_pair, 3);

index = 1;

for i = 1:length(si_dot_values)
    for j = 1:length(phi_dot_values)
        %% define si_dot and phi_dot and their initail values
        si = 30;
        phi = 30;
        si_dot = si_dot_values(i);
        phi_dot = phi_dot_values(i);

        for m = 1:num_samples_in_each_sidot_phidot_pair
            %% calculate the w_pt_b
            w_pt_b = [phi_dot * cos(si);
                      -phi_dot * sin(si);
                      si_dot];

            %% calculating DCM of T to P coordinate
            T_pt = [cos(si)           sin(si)           0;
                    -sin(si)         cos(si)           0;
                    0                 0                 1] * [1           0           0;
                                                       0           cos(phi)       sin(phi);
                                                       0           -sin(phi)      cos(phi)];

            %% calculating DCM of N to E coordinate
            T_ne = [-sin(lat) * cos(lon), -sin(lat) * sin(lon), cos(lat);
                     -sin(lon),           cos(lon),           0;
                     -cos(lat) * cos(lon), -cos(lat) * sin(lon), -sin(lat)];

            %% calculate the true gyro output
            w_bi_b = w_pt_b + T_pt * T_ne * Wei_e;

            %% simulating uncalibrated gyro output
            M = [(1 + 8e-5), 7e-5, 6e-5;
                  7e-5, (1 + 9e-5), -7e-5;
                  6e-5, -7e-5, (1 + 7e-5)];

            b_a_b = [9e-9;
                      7e-9;
                      -9e-9];

            n_a_b = randn(3, 1) .* [1e-9;
                                      1e-9;
                                      2e-9];
            uncalibrated_gyro_output = M * w_bi_b + b_a_b + n_a_b;

            %% save the uncalibrated and true values of gyro
            measured_gyro_outputs(index, :) = transpose(uncalibrated_gyro_output);
            true_gyro_outputs(index, :) = transpose(w_bi_b);

            %% updating si and phi
            si = si + si_dot * Ts;
            phi = phi + phi_dot * Ts;
            index = index + 1;
        end
    end
end

```

```

%% solve the calibration problem for three channels of gyro
H = [ones(size(true_gyro_outputs, 1), 1), true_gyro_outputs];
pesudu_inverse = inv(transpose(H) * H) * transpose(H);

% bias vector and M matrix
calibration_matrix = pesudu_inverse * measured_gyro_outputs;
bias_calibration = calibration_matrix(1, :);
M_calibration = calibration_matrix(2:4, :);

% Compute percentage error for bias
bias_error_percent = 100 * (bias_calibration - b_a_b') ./ b_a_b';

% Compute percentage error for each element of M
M_error_percent = 100 * (M_calibration - M) ./ M;

% Display results
disp('Gyro Bias estimation error in percentage:');
disp(array2table(bias_error_percent, 'VariableNames', {'X', 'Y', 'Z'}));

disp('Gyro M matrix estimation error in percentage:');
disp(array2table(M_error_percent, ...
    'VariableNames', {'X', 'Y', 'Z'}, ...
    'RowNames', {'X_row', 'Y_row', 'Z_row'}));

```

$$M = \begin{bmatrix} (1 - \gamma \times 10^{-9}) & -\gamma \times 10^{-9} & \gamma \times 10^{-9} \\ -\gamma \times 10^{-9} & (1 + \gamma \times 10^{-9}) & -\gamma \times 10^{-9} \\ \gamma \times 10^{-9} & -\gamma \times 10^{-9} & (1 - \gamma \times 10^{-9}) \end{bmatrix}$$

$$[b_a]^B = \begin{bmatrix} 0 \times 10^{-9} \\ -\gamma \times 10^{-9} \\ -\gamma \times 10^{-9} \end{bmatrix} \quad [n_a]^B \sim \begin{bmatrix} N(0, (\gamma \times 10^{-9})^2) \\ N(0, (10^{-9})^2) \\ N(0, (\gamma \times 10^{-9})^2) \end{bmatrix}$$

دالة الگوریتم

لهم اذن علیک

Gyro Bias estimation error in percentage:

X	Y	Z
-0.10928	0.061428	-0.061102

Gyro M matrix estimation error in percentage:

	X	Y	Z
X_row	1.4367e-10	-2.6708e-06	-4.9428e-07
Y_row	4.9422e-06	-3.628e-11	2.8744e-07
Z_row	8.4062e-07	2.1038e-07	-1.0303e-11

لهم اذن علیک

$$M = \begin{bmatrix} (1 - q_x l_o^{-\alpha}) & -q_x l_o^{-\alpha} & -r_x l_o^{-\alpha} \\ -q_x l_o^{-\alpha} & (1 - r_x l_o^{-\alpha}) & p_x l_o^{-\alpha} \\ -r_x l_o^{-\alpha} & p_x l_o^{-\alpha} & (1 - p_x l_o^{-\alpha}) \end{bmatrix}$$

⇒  $\rho \gg \rho_g / \rho$

$$\begin{bmatrix} b_a \end{bmatrix}^B = \begin{bmatrix} -q_x l_o^{-\alpha} \\ -r_x l_o^{-\alpha} \\ p_x l_o^{-\alpha} \end{bmatrix} \quad \begin{bmatrix} n_a \end{bmatrix}^B \sim \begin{bmatrix} N(0, (q_x l_o^{-\alpha})^2) \\ N(0, (r_x l_o^{-\alpha})^2) \\ N(0, (p_x l_o^{-\alpha})^2) \end{bmatrix}$$

⇒  $Cov(b_a, b_a) \approx Cov(n_a, n_a)$

Gyro Bias estimation error in percentage:

X	Y	Z
0.024481	-0.003907	-0.1141

Gyro M matrix estimation error in percentage:

	X	Y	Z
X_row	-9.2479e-11	2.4646e-06	-4.3519e-07
Y_row	1.6651e-06	-2.2606e-11	9.0543e-07
Z_row	4.3133e-07	-2.6551e-06	-1.2824e-11

$$M = \begin{bmatrix} (1 + q_x l_o^{-\alpha}) & \sqrt{q_x l_o^{-\alpha}} & p_x l_o^{-\alpha} \\ \sqrt{q_x l_o^{-\alpha}} & (1 + r_x l_o^{-\alpha}) & -\sqrt{r_x l_o^{-\alpha}} \\ p_x l_o^{-\alpha} & -\sqrt{r_x l_o^{-\alpha}} & (1 + p_x l_o^{-\alpha}) \end{bmatrix}$$

⇒  $\rho \ll \rho_g / \rho$

$$\begin{bmatrix} b_a \end{bmatrix}^B = \begin{bmatrix} q_x l_o^{-\alpha} \\ \sqrt{r_x l_o^{-\alpha}} \\ -q_x l_o^{-\alpha} \end{bmatrix} \quad \begin{bmatrix} n_a \end{bmatrix}^B \sim \begin{bmatrix} N(0, (q_x l_o^{-\alpha})^2) \\ N(0, (r_x l_o^{-\alpha})^2) \\ N(0, (p_x l_o^{-\alpha})^2) \end{bmatrix}$$

Gyro Bias estimation error in percentage:

X	Y	Z
-0.018248	0.0019032	-0.028497

Gyro M matrix estimation error in percentage:

	X	Y	Z
X_row	-1.8872e-11	4.3691e-07	3.4476e-07
Y_row	-6.2109e-07	-4.0986e-11	2.1e-07
Z_row	-1.8115e-07	6.2368e-08	-2.4024e-11

۳- در یک سناریو دلخواه از ۳ سناریو بررسی شده در بخش ۲، ضرایب کالیبراسیون بدست آمده را به خروجی حسگرهای IMU (خروجی‌های واقعی و دارای خطای اعمال کرده و دقت کالیبراسیون را بررسی کنید.

مدل‌های در نظر گرفته شده برای حسگرهای به صورت زیر می‌باشند:

$$\begin{aligned}\tilde{a}_x &= \hat{a}_x + S_{ax} \hat{a}_x + M_{axy} \hat{a}_y + M_{axz} \hat{a}_z + b_{ax} + n_{ax} & \tilde{\omega}_x &= \hat{\omega}_x + S_{gx} \hat{\omega}_x + M_{gxy} \hat{\omega}_y + M_{gxz} \hat{\omega}_z + b_{gx} + n_{gx} \\ \tilde{a}_y &= \hat{a}_y + S_{ay} \hat{a}_y + M_{ayx} \hat{a}_x + M_{ayz} \hat{a}_z + b_{ay} + n_{ay} & \tilde{\omega}_y &= \hat{\omega}_y + S_{gy} \hat{\omega}_y + M_{gyx} \hat{\omega}_x + M_{gyz} \hat{\omega}_z + b_{gy} + n_{gy} \\ \tilde{a}_z &= \hat{a}_z + S_{az} \hat{a}_z + M_{azx} \hat{a}_x + M_{azy} \hat{a}_y + b_{az} + n_{az} & \tilde{\omega}_z &= \hat{\omega}_z + S_{gz} \hat{\omega}_z + M_{gzx} \hat{\omega}_x + M_{gzy} \hat{\omega}_y + b_{gz} + n_{gz}\end{aligned}$$

که به عنوان مثال  $\tilde{a}_x$  مقدار شتاب صحیح اعمال شده به شتاب سنج کانال X و  $\tilde{a}_x$  مقدار شتاب قرائت شده از شتاب سنج کانال X می‌باشد. بقیه کانال‌ها نیز به صورت مشابه می‌باشند.

پس از اینجا ماتریس‌های تحقیق مارکس  $M$  و ماتریس ارتصاف‌لرزی  $S$  که با استفاده از مراحل زیر

محاسبه خواهد شد

$$\begin{aligned}\tilde{a}_n &= \hat{a}_n + M_{ayx} \hat{a}_y + M_{azx} \hat{a}_z + S_{ax} \hat{a}_x + n_{ax} + b_{ax} \\ \tilde{a}_j &= \hat{a}_j + M_{ayx} \hat{a}_n + M_{ayz} \hat{a}_z + S_{ay} \hat{a}_y + n_{ay} + b_{ay} \\ \tilde{a}_z &= \hat{a}_z + M_{azx} \hat{a}_x + M_{azy} \hat{a}_y + S_{az} \hat{a}_z + n_{az} + b_{az}\end{aligned}$$

$$\begin{bmatrix} \tilde{a}_n \\ \tilde{a}_j \\ \tilde{a}_z \end{bmatrix} = \begin{bmatrix} (1+S_{an}) & M_{ayx} & M_{azx} \\ M_{ayx} & (1+S_{aj}) & M_{ajz} \\ M_{azx} & M_{azy} & (1+S_{az}) \end{bmatrix} \begin{bmatrix} \hat{a}_n \\ \hat{a}_j \\ \hat{a}_z \end{bmatrix} + \begin{bmatrix} b_{an} \\ b_{aj} \\ b_{az} \end{bmatrix} + \begin{bmatrix} n_{an} \\ n_{aj} \\ n_{az} \end{bmatrix}$$

$$\begin{bmatrix} \tilde{\omega}_n \\ \tilde{\omega}_j \\ \tilde{\omega}_z \end{bmatrix} = \bar{M} \left( \begin{bmatrix} \tilde{a}_n \\ \tilde{a}_j \\ \tilde{a}_z \end{bmatrix} - \begin{bmatrix} b_{an} \\ b_{aj} \\ b_{az} \end{bmatrix} - \begin{bmatrix} n_{an} \\ n_{aj} \\ n_{az} \end{bmatrix} \right) \underset{\text{چون اندیزه نویز را از اندیزه نویز از جزوی حذف کردیم}}{\approx} \bar{M} \left( \begin{bmatrix} \tilde{a}_n \\ \tilde{a}_j \\ \tilde{a}_z \end{bmatrix} - \begin{bmatrix} b_{an} \\ b_{aj} \\ b_{az} \end{bmatrix} \right) \rightarrow \text{حذف نمودیم}$$

با آنچه به صورت ماتریس  $M$  از ماتریس  $\bar{M}$  می‌باشد مطابقه خواهد بود.

$$\begin{aligned}\tilde{\omega}_n &= \hat{\omega}_n + M_{ayx} \hat{\omega}_y + M_{ayz} \hat{\omega}_z + S_{ay} \hat{\omega}_x + n_{ay} + b_{ay} \\ \tilde{\omega}_j &= \hat{\omega}_j + M_{ayx} \hat{\omega}_n + M_{ayz} \hat{\omega}_z + S_{aj} \hat{\omega}_j + n_{aj} + b_{aj} \\ \tilde{\omega}_z &= \hat{\omega}_z + M_{ayz} \hat{\omega}_x + M_{ajz} \hat{\omega}_j + S_{az} \hat{\omega}_z + n_{az} + b_{az}\end{aligned}$$

$$\begin{bmatrix} \tilde{\omega}_n \\ \tilde{\omega}_j \\ \tilde{\omega}_z \end{bmatrix} = \begin{bmatrix} (1+S_{jn}) & M_{ayx} & M_{ayz} \\ M_{ayx} & (1+S_{jj}) & M_{ajz} \\ M_{ayz} & M_{ajz} & (1+S_{jz}) \end{bmatrix} \begin{bmatrix} \hat{\omega}_n \\ \hat{\omega}_j \\ \hat{\omega}_z \end{bmatrix} + \begin{bmatrix} b_{ay} \\ b_{aj} \\ b_{az} \end{bmatrix} + \begin{bmatrix} n_{ay} \\ n_{aj} \\ n_{az} \end{bmatrix}$$

$$\begin{bmatrix} \hat{\omega}_n \\ \hat{\omega}_j \\ \hat{\omega}_z \end{bmatrix} = \bar{M} \left( \begin{bmatrix} \tilde{\omega}_n \\ \tilde{\omega}_j \\ \tilde{\omega}_z \end{bmatrix} - \begin{bmatrix} b_{ay} \\ b_{aj} \\ b_{az} \end{bmatrix} - \begin{bmatrix} n_{ay} \\ n_{aj} \\ n_{az} \end{bmatrix} \right) \underset{\text{چون اندیزه نویز را از اندیزه نویز از جزوی حذف کردیم}}{\approx} \bar{M} \left( \begin{bmatrix} \tilde{\omega}_n \\ \tilde{\omega}_j \\ \tilde{\omega}_z \end{bmatrix} - \begin{bmatrix} b_{ay} \\ b_{aj} \\ b_{az} \end{bmatrix} \right)$$

جامعة الملك عبد الله للعلوم والتقنية، كلية الهندسة، قسم الكهرباء والطاقة  
 والطاقة المتجددة، إنشاءات الطاقة، قسم الطاقة المتجددة، مختبر الطاقة المتجددة، قسم الطاقة المتجددة

```

clear;
clc;

%% do accel calibration process
accel_calibration

%% convert accel output data and correct them using calibration matrix
calibrated_accel_output = (inv(M_calibration) * (measured_accel_outputs - bias_calibration)');

%% compare calibrated output to true value of accel
error_percentage = 100 * abs(calibrated_accel_output - true_accel_outputs) ./ true_accel_outputs;

%% Create the table with each combination of si and phi
si_len = length(si_positions);
phi_len = length(phi_positions);
num_rows = si_len * phi_len;

% Initialize arrays for si and phi values
si_array = repmat(si_positions, phi_len, 1);
phi_array = repmat(phi_positions', si_len, 1);

% Convert the arrays to column vectors for the table
si_array = si_array(:);
phi_array = phi_array(:);

% Create the table with units in the column names
T = table(si_array, phi_array, error_percentage(:, 1), error_percentage(:, 2), error_percentage(:, 3), ...
    'VariableNames', {'si_angle_deg', 'phi_angle_deg', 'ax_error_percent', 'ay_error_percent', 'az_error_percent'});

% Display the table
disp(T);

```

جامعة الملك عبد الله للعلوم والتقنية، كلية الهندسة، قسم الكهرباء والطاقة  
 والطاقة المتجددة، إنشاءات الطاقة، قسم الطاقة المتجددة، مختبر الطاقة المتجددة، قسم الطاقة المتجددة

si_angle_deg	phi_angle_deg	ax_error_percent	ay_error_percent	az_error_percent
10	-60	3.9568e-05	0.00011633	3.8318e-06
10	-30	7.5792e-06	9.1336e-06	-0.00015539
10	30	-2.1601e-05	-2.9466e-05	-0.00016258
10	60	-0.00010566	-7.4208e-05	7.4923e-06
110	-60	0.0014638	9.128e-07	6.3054e-06
110	-30	0.00011515	9.2585e-06	-0.000188
110	30	-0.00024212	-3.195e-05	-9.7813e-06
110	60	-0.0027695	-3.4637e-05	2.0857e-06
250	-60	0.00010632	-0.00033316	2.9334e-06
250	-30	2.3008e-05	-0.00013343	-0.000246
250	30	-1.4463e-05	5.921e-05	-0.00014954
250	60	-6.209e-05	0.0001541	8.651e-06
270	-60	0.00070921	-3.19e-06	5.6983e-06
270	-30	0.00017427	-2.2747e-05	-0.00015198
270	30	-9.7741e-05	1.9847e-05	-6.7133e-05
270	60	-0.0006773	6.2889e-05	1.0765e-05
30	-60	6.6893e-05	-0.00031178	1.6977e-05
30	-30	4.9888e-06	-4.2813e-05	-0.00012164
30	30	-1.2782e-05	0.00015832	-3.7972e-05
30	60	-1.035e-05	0.00044447	1.8103e-05
150	-60	4.9207e-05	-2.8067e-05	7.0661e-06
150	-30	1.073e-05	-8.9256e-06	-0.00018421
150	30	-2.036e-05	5.4652e-06	-0.00020724
150	60	-2.8734e-05	0.00012951	1.3874e-05

که میتواند باعث افزایش تراویح و ایجاد خطاها شود. بنابراین توانایی کنترل خطاها بسیار مورد توجه قرار دارد.

```

clear;
clc;

%% do accell calibration process
gyro_calibration

%% convert accell output data and correct them using calibration matrix
calibrated_gyro_output = (inv(M_calibration) * (measured_gyro_outputs - bias_calibration)');

%% compare calibrated output to true value of accell
error_percentage = 100 * abs(calibrated_gyro_output - true_gyro_outputs) ./ true_gyro_outputs;
num_blocks = size(error_percentage, 1) / num_samples_in_each_sidot_phidot_pair;
error_percentage_downsampled = zeros(num_blocks, 3);
% Loop through each block and compute the mean
for i = 1:num_blocks
    idx_start = (i-1)*num_samples_in_each_sidot_phidot_pair + 1;
    idx_end = i*num_samples_in_each_sidot_phidot_pair;
    block = error_percentage(idx_start:idx_end, :);
    error_percentage_downsampled(i, :) = mean(block, 1); % mean across rows
end

%% Create the table with each combination of si and phi
si_dot_len = length(si_dot_values);
phi_dot_len = length(phi_dot_values);
num_rows = si_dot_len * phi_dot_len;

% Initialize arrays for si and phi values
si_dot_array = repmat(si_dot_values, phi_dot_len, 1);
phi_dot_array = repmat(phi_dot_values', si_dot_len, 1);

% Convert the arrays to column vectors for the table
si_dot_array = si_dot_array(:);
phi_dot_array = phi_dot_array(:);

% Create the table with units in the column names
T = table(si_dot_array, phi_dot_array, error_percentage_downsampled(:, 1), error_percentage_downsampled(:, 2), error_percentage_downsampled(:, 3), ...
    'VariableNames', {'si_dot_deg/sec', 'phi_dot_deg/sec', 'gx_error_percent', 'gy_error_percent', 'gz_error_percent'});

% Display the table
disp(T);

```

جذب و توجيه الماء من مياه الأمطار

si_dot_deg/sec	phi_dot_deg/sec	gx_error_percent	gy_error_percent	gz_error_percent
10	10	-1.6078e-07	-4.1476e-09	1.6625e-08
10	20	-8.1106e-08	-4.9576e-10	1.6244e-08
10	30	-2.2645e-07	-4.5483e-09	1.6022e-08
10	-10	-2.4822e-07	-1.922e-09	1.5675e-08
10	-20	-2.3814e-07	-4.6140e-09	1.6340e-08
10	-30	6.4655e-09	-6.0773e-09	1.6296e-08
20	10	1.1397e-09	-1.1775e-08	7.9506e-09
20	20	5.6052e-09	-1.0521e-08	8.1081e-09
20	30	1.7201e-09	-2.6202e-09	8.1293e-09
20	-10	1.5942e-09	-1.1781e-08	8.0351e-09
20	-20	3.0902e-09	-1.5015e-08	7.8562e-09
20	-30	3.7445e-09	-7.0015e-09	7.8613e-09
30	10	-1.1197e-07	-1.5392e-09	5.3681e-09
30	20	-9.2039e-08	5.7723e-10	5.5685e-09
30	30	-3.5525e-08	2.0762e-10	5.5445e-09
30	-10	-6.0243e-08	-1.5475e-09	5.2562e-09
30	-20	-8.0771e-08	1.6132e-09	5.2116e-09
30	-30	-1.3999e-07	5.0486e-09	5.217e-09
-10	10	-4.2469e-08	2.1452e-09	-1.6028e-08
-10	20	-3.8486e-08	-1.9002e-09	-1.5594e-08
-10	30	-2.1619e-08	7.4971e-09	-1.6086e-08
-10	-10	-2.7141e-08	1.2966e-08	-1.6271e-08
-10	-20	-1.5788e-08	7.7693e-09	-1.6002e-08
-10	-30	-4.0925e-08	5.2712e-09	-1.6007e-08
-20	10	-2.6299e-08	2.1162e-09	-8.266e-09
-20	20	-3.5437e-08	-2.3533e-09	-7.6508e-09
-20	30	-6.7994e-09	-1.3244e-09	-8.0837e-09
-20	-10	-3.9585e-09	-1.9229e-09	-8.0712e-09
-20	-20	-2.1118e-09	3.4103e-10	-7.9653e-09
-20	-30	-1.7678e-08	-2.5929e-09	-8.0073e-09
-30	10	-1.0183e-09	5.2942e-09	-5.2948e-09
-30	20	1.4962e-09	4.6035e-09	-5.2602e-09
-30	30	-2.6305e-10	3.8603e-09	-5.361e-09
-30	-10	-7.8568e-09	4.7867e-09	-5.5298e-09
-30	-20	-7.7234e-09	4.106e-09	-5.3269e-09
-30	-30	-6.3736e-09	3.4331e-09	-5.3214e-09