Table of Contents

Setting up

```
clc
clear;

addpath('.\sensordata\')
addpath('.\Helper functions\')

run startup.m
```

Establishing connection

```
[xhat, meas] = filterTemplate();
```

Plotting

```
clf;
figure()
plot(meas.t, meas.acc)
title('Accelerometer measuremnts')
legend('x', 'y', 'z')
xlabel('Time'), ylabel('y_{acc}')
% print('./Images/Acc_meas_flat','-dpng')

figure()
plot(meas.t, meas.gyr)
title('Gyroscope Measurements')
legend('x', 'y', 'z')
xlabel('Time'), ylabel('y_{gyr}')
% print('./Images/Gyro_meas_flat','-dpng')
```

```
figure()
plot(meas.t, meas.mag)
title('Magnetometer Measurements')
legend('x', 'y', 'z')
xlabel('Time'), ylabel('y_{mag}')
% print('./Images/Mag_meas_flat','-dpng')
```

Computing mean and covariance

Accelerometer measurements

```
mean_acc = mean(meas.acc(:, ~any(isnan(meas.acc), 1)), 2)
cov_acc = cov(meas.acc(:, ~any(isnan(meas.acc), 1)).')
histogram(meas.acc(3, ~any(isnan(meas.acc),
 1)), 'Normalization','probability')
title('Accelerometer measurements')
xlabel('y_{acc}^z'), ylabel('Probability')
print('./Images/hist_acc_z','-dpng')

% Gyroscope measurements
mean_gyr = mean(meas.gyr(:, ~any(isnan(meas.gyr), 1)), 2)
cov_gyr = cov(meas.gyr(:, ~any(isnan(meas.gyr), 1)).')
histogram(meas.gyr(2, ~any(isnan(meas.gyr),
 1)), 'Normalization','probability')
title('Gyrometer measurements')
xlabel('y_{gyr}^y'), ylabel('Probability')
print('./Images/hist_gyr_y','-dpng')

% Magnetometer measurements
mean_mag = mean(meas.mag(:, ~any(isnan(meas.mag), 1)), 2)
cov_mag = cov(meas.mag(:, ~any(isnan(meas.mag), 1)).')
histogram(meas.mag(3, ~any(isnan(meas.mag),
 1)), 'Normalization','probability')
title('Magenetometer measurements')
xlabel('y_{mag}^z'), ylabel('Probability')
print('./Images/hist_mag_z','-dpng')

% Saving flat_data for reference
save flat_data.mat
```

Task 4

Implemented time update function Test

```
x = ones(4,1);
P = rand(4);

omega = rand(3,1);
T = 0.1;
Rw = rand(3);

[x, P] = tu_qw(x, P, omega, T, Rw);
```

Task 5

```
[xhat5, meas5] = Myfilter();
```

Plotting the results

```
Estimated_euler5 = quat2eul(xhat5.x');
Orientation_euler5 = quat2eul(meas5.orient');

T5 = xhat5.t(2:end);
figure()
subplot(3,1,1)
plot(T5, Estimated_euler5(:,1)')
hold on
plot(T5, Orientation_euler5(:,1)')
hold off
legend('Estimated', 'Google')
title('Yaw')
xlabel('Time'), ylabel('\psi (radians)')

subplot(3,1,2)
plot(T5, Estimated_euler5(:,2)')
hold on
plot(T5, Orientation_euler5(:,2)')
hold off
legend('Estimated', 'Google')
title('Pitch')
xlabel('Time'), ylabel('\theta')

subplot(3,1,3)
plot(T5, Estimated_euler5(:,3)')
hold on
plot(T5, Orientation_euler5(:,3)')
hold off
legend('Estimated', 'Google')
title('Roll')
xlabel('Time'), ylabel('\phi')

% print('./Images/task5_timeUpdate_eul','-dpng') % Rotations alongs
 all axes

% print('./Images/task5_tilted_start_eul','-dpng')  % XY axis
 perpendicular to the groud

% print('./Images/task5_shake_eul','-dpng') % Shake
```

Task 6

Implemented the acceleromater measurement update

```
x = rand(4,1);
P = rand(4);
```

```
yacc = rand(3,1);
Ra = cov_acc;
g0 = [0;0;9.81];

[x, P] = mu_g(x, P, yacc, Ra, g0)
```

Task 7

```
[xhat7, meas7] = Myfilter();
```

Plotting after the accelerometer measurement update

```
Estimated_euler7 = quat2eul(xhat7.x');
Orientation_euler7 = quat2eul(meas7.orient');

T7 = xhat7.t(2:end);
figure()
subplot(3,1,1)
plot(T7, Estimated_euler7(:,1)')
hold on
plot(T7, Orientation_euler7(:,1)')
hold off
legend('Estimated', 'Google')
title('Yaw')
xlabel('Time'), ylabel('\psi (radians)')

subplot(3,1,2)
plot(T7, Estimated_euler7(:,2)')
hold on
plot(T7, Orientation_euler7(:,2)')
hold off
legend('Estimated', 'Google')
title('Pitch')
xlabel('Time'), ylabel('\theta')

subplot(3,1,3)
plot(T7, Estimated_euler7(:,3)')
hold on
plot(T7, Orientation_euler7(:,3)')
hold off
legend('Estimated', 'Google')
title('Roll')
xlabel('Time'), ylabel('\phi')

% print('./Images/task7_timeUpdate_eul','-dpng') % Rotations alongs
 all axes

% print('./Images/task7_tilted_start_eul','-dpng')  % XY axis
 perpendicular to the groud
```

```
% print('./Images/task7_shake_eul','-dpng') % Shake

% print('./Images/task8_outlier_rej_eul','-dpng') % Outlier rejection
```

Task 9

Implemented measurement update using magnetometer measurements Test

```
x = rand(4,1);
P = rand(4);
mag = rand(3,1);
m0 = mean_mag;
Rm = cov_mag;

[x, P] = mu_m(x, P, mag, m0,Rm)
```

Task 10

```
[xhat10, meas10] = Myfilter();
```

Plotting with magnetometer measurement update

```
Estimated_euler10 = quat2eul(xhat10.x');
Orientation_euler10 = quat2eul(meas10.orient');

T10 = xhat10.t(2:end);
figure()
subplot(3,1,1)
plot(T10, Estimated_euler10(:,1)')
hold on
plot(T10, Orientation_euler10(:,1)')
hold off
legend('Estimated', 'Google')
title('Yaw')
xlabel('Time'), ylabel('\psi (radians)')

subplot(3,1,2)
plot(T10, Estimated_euler10(:,2)')
hold on
plot(T10, Orientation_euler10(:,2)')
hold off
legend('Estimated', 'Google')
title('Pitch')
xlabel('Time'), ylabel('\theta')

subplot(3,1,3)
plot(T10, Estimated_euler10(:,3)')
hold on
plot(T10, Orientation_euler10(:,3)')
```

```
hold off
legend('Estimated', 'Google')
title('Roll')
xlabel('Time'), ylabel('\phi')

% print('./Images/task10_timeUpdate_eul','-dpng') % Rotations alongs
 all axes

% print('./Images/task10_facing_eul','-dpng') % Facing a different
 direction other than north

% print('./Images/task10_tilted_start_eul','-dpng')  % XY axis
 perpendicular to the groud

% print('./Images/task10_mag_dist_eul','-dpng') % Magnetic Disturbance
 with single pitch rot

% print('./Images/task11_outlier_rej_eul','-dpng') % Outlier rejection
```

Task 12

Implemented the outlier rejection for magnetometer reading

```
[xhat12, meas12] = Myfilter();
Estimated_euler12 = quat2eul(xhat12.x');
Orientation_euler12 = quat2eul(meas12.orient');

T12 = xhat12.t(2:end);
figure()
subplot(3,1,1)
plot(T12, Estimated_euler12(:,1)')
hold on
plot(T12, Orientation_euler12(:,1)')
hold off
legend('Estimated', 'Google')
title('Yaw')
xlabel('Time'), ylabel('\psi (radians)')

subplot(3,1,2)
plot(T12, Estimated_euler12(:,2)')
hold on
plot(T12, Orientation_euler12(:,2)')
hold off
legend('Estimated', 'Google')
title('Pitch')
xlabel('Time'), ylabel('\theta')

subplot(3,1,3)
plot(T12, Estimated_euler12(:,3)')
hold on
plot(T12, Orientation_euler12(:,3)')
hold off
legend('Estimated', 'Google')
```

```
title('Roll')
xlabel('Time'), ylabel('\phi')

% print('./Images/task12_acc_mag','-dpng') % only acc and mag
% print('./Images/task12_acc_mag2','-dpng') % only acc and mag

% print('./Images/task12_gyr_mag','-dpng') % only gyr and mag
% print('./Images/task12_gyr_mag2','-dpng') % only gyr and mag
```

Published with MATLAB® R2020a