Simon Honigmann

Sensor Orientation

November 7, 2018

Lab 6: Coarse alignment of a high accuracy IMU

Task 1: Data Collection and Processing

Data was collected from a (locally) stationary XSea navigation grade IMU over approximately 2 minutes with a sampling rate of 200 Hz for 24000 data points per axis per sensor. The IMU was relatively level (pitch and roll were roughly 0) and a yaw of approximately 200 degrees was set with respect to the NWD local frame. The localization reported by the iXRepeater : Airins software was noted to be used to compare calculations against. Initially, the IMU data was reported in the NWU frame and the reference localization was given in the NWD frame. Both collections of data were converted to the NED frame for all subsequent analysis.

First, the data was imported and the appropriate subset corresponding to my angle was selected. This data is shown in Figure 1 for the accelerometer and gyroscope readings.

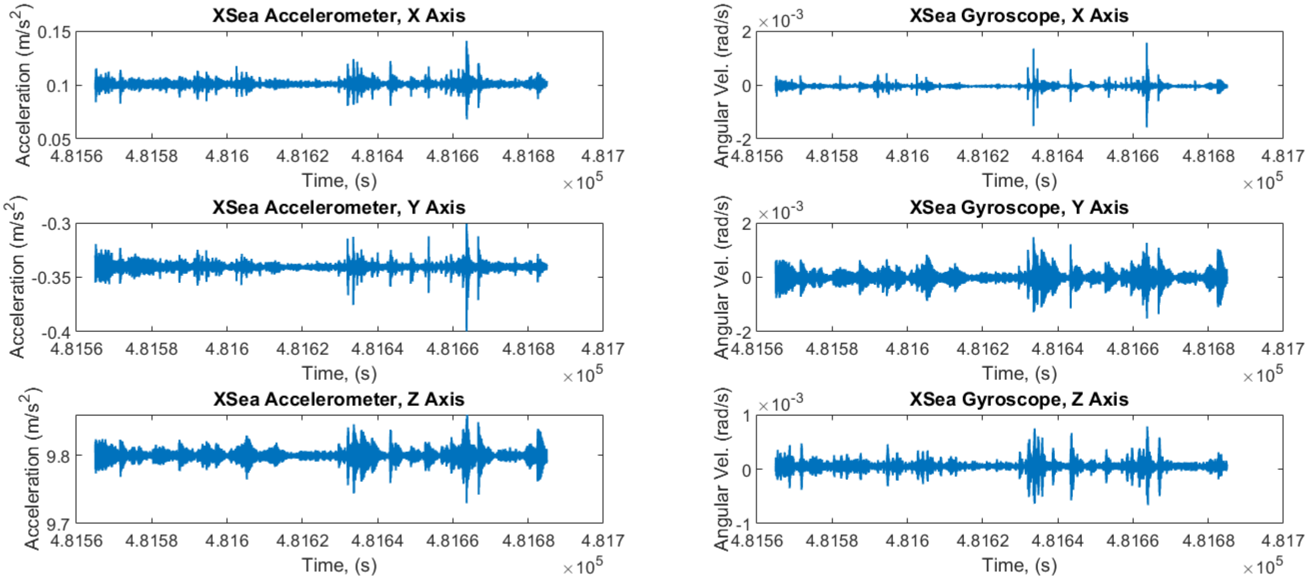

Figure 1: Raw Accelerometer and Gyroscope Data

No data points needed to be removed from the 2 minute span. The data was then averaged for all 6 data series (accelerometer and gyro triads). Averaging over the 24000 data points helps reduce the impact of noise on the coarse alignment. After averaging, the raw data was converted into the NED frame by applying a transformation matrix. The average values post-transformation are included in Table 1 below.

Table 1: Average Sensor Data

Axis:	Accelerometer (m/s^2):	Gyroscope (rad/s):
X	0.1008	-4.636e-5
Y	0.3405	2.003e-5
Z	-9.7994	-5.280e-5

Task 2 & 3: Calculating the Triad Norms and Comparing to the Reference

The averaged accelerometer and gyro data could the be normalized to find the vector magnitude. For a stationary IMU, the accelerometer magnitude is expected to equal the gravity vector. Similarly the vector magnitude of the gyroscope components is expected to equal the rate of rotation of the Earth (~360 deg/day). Table 2 shows the results of these norm calculations and calculates the error with respect to the given reference values.

Table 2: Accelerometer and Gyro Norms and Errors with Respect to Reference Values

	Accelerometer (m/s^2):	Gyroscope (rad/s):
IMU 3 Axis Norm:	9.8058	7.307e-5
Accepted Value:	9.8055	7.292e-5
Error:	0.0003	1.4e-7
Percent Error:	0.003%	0.19%

The questions for these two tasks are answered below:

1. A difference of only 1.4e-7 rad/s was measured between the accepted value of earths rotational speed and that measured by the gyroscope. This corresponds with a 0.19% error.
2. Because earth's rotational rate is very well defined, it is a rather convenient reference to compare the accuracy of gyroscopes. The earth never stops spinning and does so fairly consistently. That said, there is more to quantifying sensor accuracy than just measuring an average compared to a known reference. Sensor drift and noise are also both important factors. Noise, because filtering isn't free when it comes to control applications, and drift because the apparent accuracy can decrease over time. Earth's rotation also isn't the best choice for quantifying accuracy on lower performance sensors, as the angular rate is relatively small (~70 microrad/s). Lower performance sensors, including most MEMS gyros would likely have trouble distinguishing earth's rotation from other sources of error.
3. A difference of only 3e-4 m/s^2 was measured between accepted value of gravitational acceleration at EPFL and the measured value. This corresponds with a miniscule 0.003% error.
4. Similarly to using earth's rotation for gyro accuracy comparisons, gravity can be used to estimate accuracy of accelerometers. While this is a more powerful approach due to the relative size of gravitational acceleration (typically on the same order of magnitude as the experienced accelerations of vehicles), consideration still needs to be taken regarding local variations in gravitational field to use this method appropriately. Again, noise and drift are also important considerations.

Task 4: Accelerometer Leveling

The next step in coarse alignment, after collecting and averaging data, is to calculate the roll and pitch of the IMU based on the accelerometer values. Using the x and y accelerometer values, the pitch and roll can be determined. This is because the pitch and roll act about the x and y axes respectively. Therefore, only the roll angle determines the vertical component of the x axis, and similarly only the pitch angle determines the vertical component of the y axis. By taking the arcsin of the ratio between the x and y measurements and the gravity magnitude, the pitch and roll angles can be determined respectively. These are included in Table 3 below.

Table 3: IMU Pitch and Roll Calculations and Comparison to Reference Values

	Pitch Angle	Roll Angle
Calculated (rad)	0.0103	-0.0347
Reference (rad)	0.0103	-0.0346
Error (rad)	2.2e-5	-0.0001
Percent Error (%)	0.21%	0.30 %

Questions 6 and 7:

The roll and pitch values measured using accelerometer levelling correspond extremely well with the reference values given by the real-time solution during acquisition. Less than 1% error was noted for both angles, which is particularly impressive given how small the angles themselves are. Errors amounted to less than half an arcminute in both cases.

Task 5: Gyrocompassing

Finally, coarse alignment can be completed using gyrocompassing, a technique where the yaw is determined by transforming the gyro vectors into a pseudo-local frame and then relating them to Earth's rotation rate. First the rotation matrices R1 and R2 can be determined by using the previously calculated roll and pitch angles. Azimuth can be assumed to be 0 here as this method is independent of Azimuth rotation. This simplifies math as well and R3 becomes the identity matrix. With the original gyro vectors transformed into this pseudo local frame, the azimuth can then be calculated by relating its x and y vectors. With Azimuth, the proper local frame can then be calculated by determining the proper transformation matrix, R3(Az). With the local frame calculated, the latitude can then be calculated, again based on the gyro vector norm or earth's rotation rate. Table 4 below outlines the results from this task.

Table 4: Gyrocompassing to Calculate Azimuth and Latitude

	Azimuth	Latitude
Calculated (deg)	-201.192	46.139
Reference (deg)	-200.983	46.521
Error (deg)	0.209	-0.3824
Percent Error (%)	0.104 %	-0.82 %

Questions:

8. The yaw was calculated to be -201.192 degrees, compared to the reference of -200.983 degrees. This corresponds with an error of 0.2 degrees
9. Depending on the situation, single epoch alignment could be sufficient. For instance, if yaw is unimportant and only pitch and roll are required to initialize a controller, a quick initialization would likely be appropriate. For autonomous mapping or navigation with a dead-reckoning approach, data averaging is likely merited. This is because small angular errors will accrue over larger distances, and noise in initial readings will result in quite large positional changes when latitude is calculated.
10. But of course. After calculating Azimuth, it can be used to transform the gyro axes into the local frame (along with the roll and pitch transformations previously performed). The north axis of the local frame can be related to the rate of rotation of the Earth to find the latitude. Latitude was found to be 46.139 degrees, compared to a reference value of 46.521 degrees. This corresponds with an error of approximately 0.4 degrees, which seems impressive on the surface, but corresponds with (very) roughly 45 km in practice.

Appendix A: Matlab Code

```
%% Simon Honigmann
%  Sensor Orientation, Lab 6
%  Nov 9, 2018

%% Cleanup
clear
clc

graphs = 0; %1 to graph output, 0 to turn graphing off
```

```
%% Reference Values
mean_earth_rotation = 7.2921150e-5; %mean earths rotation, rad/s (NED)
grav_epfl = (980000+550)*10^-5; %gravity at epfl, m/s^2
%IMU Reported Values
lat_IMU = 46.52093816; %IMU reported latitude, deg
long_IMU = 6.56585746; %IMU reported longitude, deg
h_IMU = -200.983; %IMU reported heading, rad
r_IMU = -1.984*pi/180; %IMU reported roll, rad
p_IMU = 0.588*pi/180; %IMU reported pitch, rad (NED)
alt_IMU = 410.258; %IMU reported Altitude, m
speed_IMU = 0; %IMU reported speed norm, m/s

%% Lab Formatting:
set(groot,'DefaultAxesFontSize',14);
set(groot,'DefaultLineLineWidth',1.5);

%% Load and Preprocess Data for XSEA navigation sensor
load('lab6data.mat');
%data = readimu('1109_1407_PostProBinaryDecoded.imu');

t = data(:,1); %time
sample_rate = 1/(t(2)-t(1)); %Hz

start_time = 481565; %(s) taken from photo
[~, start_index] = min(abs(t-start_time));

%start_index = 489114; %found by looking at all data and seeing where my angle was...
photo time didn't work for me...
stop_index = floor(2*60*sample_rate+start_index);

t=t(start_index:stop_index);
gyro = data(start_index:stop_index,2:4); %x y and z measurements of gyro
accel = data(start_index:stop_index,5:7); %x y and z measurements of accelerometer

%% Averaging over the time interval:
gyro_avg = mean(gyro);
accel_avg = mean(accel);

%% Calculate the Norm of the Gyros
gyro_avg(:,4) = sqrt(gyro_avg(1).^2+gyro_avg(2).^2+gyro_avg(3).^2);
accel_avg(:,4) = sqrt(accel_avg(1).^2+accel_avg(2).^2+accel_avg(3).^2);

%% Plotting Raw Data
axisLabels = ['X','Y','Z','Norm'];
if graphs==1
    for i=1:3
        figure(1);
        subplot(3,1,i);
        plot(t,gyro(:,i));
        title(['XSea Gyroscope, ',axisLabels(i),' Axis']);
        xlabel('Time, (s)');
        ylabel('Angular Velocity (rad/s)');

        figure(2);
        subplot(3,1,i);
        plot(t,accel(:,i));
        title(['XSea Accelerometer, ',axisLabels(i),' Axis']);
        xlabel('Time, (s)');
        ylabel('Acceleration (m/s^2)');
    end
end
%% Converting Reference Frame
%IMU Uses Front Left Up = X Y Z
R = [1 0 0; 0 -1 0; 0 0 -1]; %transformation matrix from IMU to NED

gyro_avg(1:3) = R*gyro_avg(1:3)'; %convert from NWU to NED
accel_avg(1:3) = R*accel_avg(1:3)';%convert from NWU to NED

%% Calculate the Errors (Q2.1, Q3.1)
err_earth_rotation = gyro_avg(4)-mean_earth_rotation;
p_err_earth_rotation = (gyro_avg(4)-mean_earth_rotation)/mean_earth_rotation*100;
err_g = accel_avg(4)-grav_epfl;
p_err_g = (accel_avg(4)-grav_epfl)/grav_epfl*100;

tableQ21 = [gyro_avg(4),mean_earth_rotation,err_earth_rotation, p_err_earth_rotation]
tableQ34 = [accel_avg(4),grav_epfl,err_g, p_err_g]

%% Accelerometer Leveling
%Calculated Values
g = -accel_avg(4);
r = asin(accel_avg(2)/g); % roll, rad
p = asin(-accel_avg(1)/g); % pitch, rad

%errors
err_r = r-r_IMU;
p_err_r = err_r/r_IMU*100;
err_p = p-p_IMU;
p_err_p = err_p/p_IMU*100;

tableQ4 = [r,r_IMU,err_r, p_err_r; ...
           p,p_IMU,err_p, p_err_p]

%creating local level frame (NED) from body frame (front left up):
Az = 0; %for now
R1 = [1     0       0; ...
      0 cos(r)   sin(r); ...
      0 -sin(r)  cos(r)];

R2 = [cos(p)  0 -sin(p); ...
      0       1     0;  ...
      sin(p)  0 cos(p)];

%rotating to local frame with arbitrary azimuth
omega_l = ((R1*R2)'*gyro_avg(1:3)')

%% Gyrocompassing
Az = atan2(omega_l(2),omega_l(1));
 if Az>0
     Az = Az-2*pi;
 end
Az_deg = Az*180/pi
Az_err = Az_deg - h_IMU

% properly determining the local frame now:
R3 = [cos(Az)  sin(Az) 0; ...
      -sin(Az)  cos(Az) 0; ...
      0         0      1];
```

```
%rotating to local frame with arbitrary azimuth
omega_l = (R3*R2*R1*gyro_avg(1:3)');

%% Determining Latitude
lat = acosd(omega_l(1)/gyro_avg(4)); %from eq: wN = wEcos(phi) on slide 1
err_lat = lat-lat_IMU;
p_err_lat = err_lat/lat_IMU*100;
tableQ5 = [lat,lat_IMU,err_lat,p_err_lat]
```