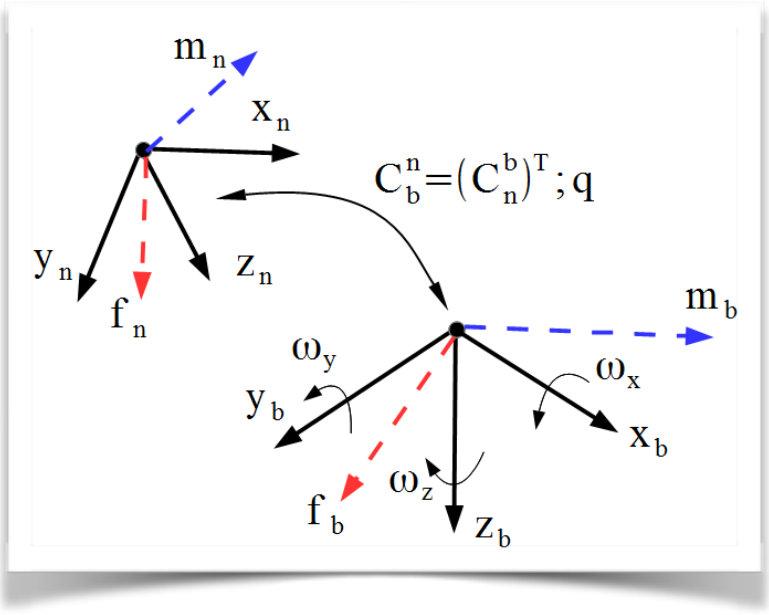
GyroLib - Matlab AHRS Library

GyroLib is a free, open-source Attitude and Heading Reference System (AHRS) library for the Matlab. It includes basic algorithms that allow to determine the orientation of the device equipped with the inertial sensors - accelerometers and gyroscopes and with vector magnetometer.

Axes definitions. It is assumed that the device is equipped with three-axis accelerometer, gyroscope and vector magnetometer with mutually orthogonal axes: x axis is looking "forward"; z axis is looking "downwards"; y axis completes the right orthogonal system of axes.

Sensors measurements. Gyroscopes measure positive angular rates being rotated counter-clockwise (if one watches from the end of the corresponding axis). The magnetometer is assumed to measure the Earth's magnetic field vector. Accelerometer "measures" the gravity force (it actually doesn't - it can only measure active forces, and when it sits on the table it measures the force with which the table acts on the accelerometer body preventing it from falling right through the lid of the table). Gyroscopes measurements are in rad/sec, accelerometers measurements are in m/sec^2.

Order of rotations. The standard ZYX order is used - first we rotate counter-clockwise around the z axis, then around the new y axis and finally around the new x axis.



Attitude determination. What AHRS is basically doing is that it finds the attitude of the device, equipped with sensors relative to some other pre-defined frame. We call the first frame "body" frame and latter "navigation" frame. We can define the attitude of the body using two non-collinear vectors attached to it.

TRIAD and **QUEST.** These two algorithms (*TRIAD.m* and *QUEST.m*) determine the attitude using information from two sensors, that measure the projections of two non-collinear vectors on the axes of the body frame. We can use Earth's magnetic field vector and gravity vector, or two direction vectors to the distant stars, etc. We also need to know the projections of the same two vectors on the axes of the navigation frame. TRIAD and QUEST algorithms employ the described principle and return the estimations of the direction cosine matrix and attitude quaternion respectively. Algorithms in action can be seen by calling *test_TRIAD.m* and *test_QUEST.m* functions. Here is the typical output of one of them (type test_TRIAD in the command line):

```
>> test_TRIAD
Measurements in b-frame:
fb=[0.181717, 0.619803 -0.763429]
mb=[0.539385, 0.199423 -0.818104]

Reference DCM:
   0.8547   -0.3957   -0.3360
  -0.5097   -0.7625   -0.3985
  -0.0985    0.5118   -0.8534

Vectors in n-frame:
fn=[0.166534, -0.261050 0.950852]
mn=[0.656958, -0.100994 0.747132]

Estimated DCM:
   0.8547   -0.3957   -0.3360
  -0.5097   -0.7625   -0.3985
  -0.0985    0.5118   -0.8534

Test - should give the identity matrix:
   1.0000    0.0000         0
        0    1.0000    0.0000
   0.0000         0    1.0000

Errors = [0.0000000000, -0.0000000000 0.0000000000] deg.
```

AHRS algorithms. The problem with attitude determination using magnetic field vector and "gravity" vector is that these two vectors can easily be disturbed (accelerated motion, magnetic field sources nearby) which leads to erroneous attitude estimates. AHRS algorithm uses the information from the gyroscopes (integrates angular rates) to determine the attitude and periodically employs information from the accelerometer and magnetometers to correct the attitude. AHRS algorithms in GyroLib was built using classical scheme, where the complimentary Kalman filter estimates the errors in calculated attitude and gyroscopes biases. Two types of AHRS are implemented: *ahrs_dcm.m* uses the direction cosine matrix (DCM) to represent the attitude, while *ahrs_quat.m* uses the quaternion.

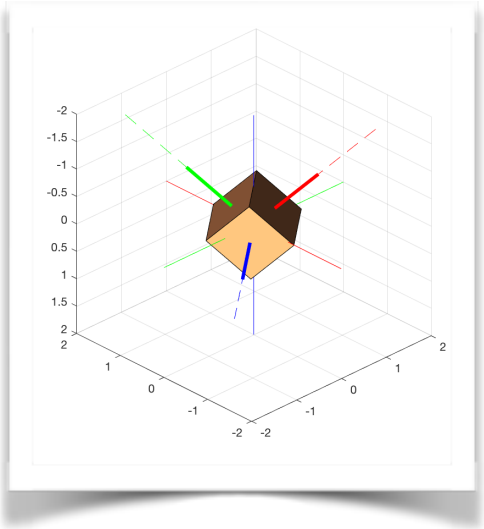
Reference data for AHRS algorithm testing are generated by calling the following function:

[dWb, q, dt, q_init, fb, mb, fn, mn, bw] = reference_data_spin_cone;

This function creates reference accelerometer, magnetometer and gyroscopes measurements alongside with reference values of attitude quaternion. SPIN-CONE algorithm is used (Paul G. Savage, Strapdown System Performance Analysis), so the reference attitude as well as all the sensors measurements are exact.
One can assess the performance of TRIAD, QUEST, angular rate integration algorithms and visualize the reference motion of the body frame by running the following function:

test_spin_cone(dWb, fb, mb, fn, mn, q, q_init, dt);

Note how the attitude, calculated by integration of angular rates, slowly drifts because of the body frame coning motion.



Finally, to estimate how AHRS algorithms work add some noise to the sensors measurements: sensors noises, sensors biases and biases random walk and generate noisy data by rerunning *reference_data_spin_cone* function. You can also rerun *test_spin_cone* to se how sensors noises influence the attitude estimations. To assess the performance of the AHRS algorithms call the following functions:

[Euler_, bw_] = test_ahrs_dcm(dWb, fb, mb, q, q_init, fn, mn, dt);

[Euler_, bw_] = test_ahrs_quat(dWb, fb, mb, q, q_init, fn, mn, dt);

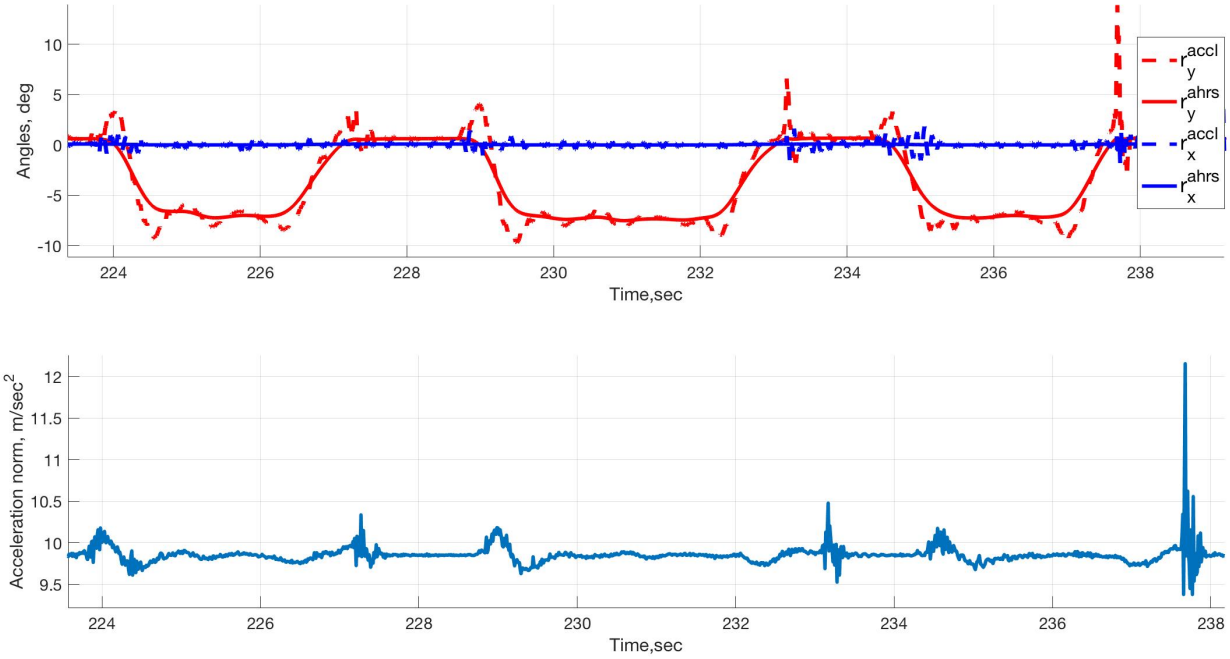
Using GyroLib with real sensors data

To demonstrate how the AHRS algorithms works with real sensors data we collected recordings from two different MEMS IMU modules: Analog Devices' ADIS 16485 and Fairchild's FMT 1010. Test data can be found in **/data** folder.

Recording from the ADIS 16485 module (adis_16485.mat) should first be imported to the workspace. Note that this module doesn't include vector magnetometer, so the AHRS algorithm is unable to provide long-term stability of the heading channel. Performance of the DCM-based AHRS algorithm can be evaluated by calling the following test function (we substitute the acceleration for magnetic field vector here for convenience, it doesn't influence the estimation results):

[Euler_, bw_, Angles_] = test_ahrs_dcm_data(dWb, Fb, Fb, fn, fn, dt);

Among the resulting plots note one that compares the angles, estimated by the AHRS and by the accelerometers. Note how AHRS keeps the attitude smooth when accelerometers are disturbed with movements - bottom plot represents the acceleration norm:



Recording from the FMT 1010 module (fmt_1010.mat) should be imported to the workspace. This module includes the full set of required sensors, so the DCM-based AHRS algorithm should be called like this:

[Euler_, bw_, Angles_] = test_ahrs_dcm_data(dWb, Fb, Mb, fn, mn, dt);