Notes on an aviation-capable AHRS algorithm for GPS and IMU sensor fusion and its open source implementation

Jiri Pittner[*]

February 23, 2020

Abstract

These notes give some information about an aviation-capable AHRS algorithm for GPS and IMU sensor fusion, which was employed in my hobby project described in Ref.[13], which I made public under the GPL license.

1 Introduction

In recent years, MEMS accelerometers and rotation sensors as well as magnetic field sensors became widely available as cheap integrated circuits and are employed in smartphones, tablets etc. In 2010, Madgwick published an algorithm for "sensor fusion" which computes orientation from the raw sensor data, correcting for measurement noise and drift and provided also an open source implementation[1]. In 2017, Admiraal et al. [2] reformulated this algorithm in a more clear manner, basically as an integration of a differential equation for time evolution of the orientation quaternion, corrected by gradient descent algorithm minimizing the deviation of the orientation with respect to the gravitational and magnetic reference vectors. Later, Wilson et al.[3] improved the algorithm by taking the second reference to be a vector product of gravitation with magnetic field, so that the two references are orthogonal and the respective gradient descent steps do not interfere. This is a similar idea to the improvement from the steepest descent to conjugate gradient function minimization algorithm and also solves the problem of non-zero and varying z-component of the magnetic field, which would otherwise possibly spoil the pitch and roll angle computation.

However, the aforementioned papers were not concerned with application to aviation, so they omitted the fact that in a non-inertial reference system the accelerometer does not measure gravity, but the vector sum of gravity with non-inertial forces. Similarly, Android interface to IMU sensors provides "gravity vector" as a low-passed accelerometer readout. Therefore any "attitude indicator" or "artificial horizon" smartphone app, which relies on the orientation from the standard library is doomed to fail during turning flight in a plane (or when driving a car at a roundabout). In this note I combined the transformation from Earth reference frame (approximated to be inertial) to the non-inertial moving frame of the sensor to compute the "pure" gravity vector to be used as reference for the AHRS algorithm. This transformation is of course well known in aviation [4], but maybe less well known in other application areas of IMU sensors. I also provide a GPL open source implementation of the algorithm in C++ and open hardware design of an experimental attitude/airspeed/altitude indicator [13].

[*]jiri@pittnerovi.com

2 Algorithm and its implementation

2.1 AHRS algorithm summary

Rotations in three-dimensional space are conveniently described by normalized quaternions, which form a group isomorphic to SU(2) and homomorphic to the group of proper rotations SO(3) ($\boldsymbol{q}$ and $-\boldsymbol{q}$ yield the same rotation matrix) [6, 7] The vectors subject to rotation are represented as quaternions with zero real component and the rotation of a vector $\boldsymbol{x}$ by a normalized quaternion $\boldsymbol{q}$ is performed as

$$\boldsymbol{x}' = \boldsymbol{q}^{-1}\boldsymbol{x}\boldsymbol{q} = \boldsymbol{q}^{*}\boldsymbol{x}\boldsymbol{q} \tag{1}$$

The (non-inertial) plane reference frame is centered in the plane's center of mass and oriented according the the usual convention in aviation [4], with x-axis from back to front of the plane, y-axis from left to right wing, and z axis pointing "down" from the pilot's point of view. The Earth reference frame is approximated to be inertial and is defined so that a plane sitting at the earth with its tail-to-cockpit line towards north has its axes aligned with the Earth frame. When considering rotational orientation, one can assume this frame translated at each time instant to become centered in the center of mass of the plane, so that the two frames are related by rotation only. In accord with [2], we define the orientation quaternion so that

$$\boldsymbol{r}_{\text{plane}} = \boldsymbol{q}^{*}\boldsymbol{r}_{\text{earth}}\boldsymbol{q} \tag{2}$$

where $\boldsymbol{r}$ is position vector in quaternion representation. When rotating with angular velocity $\boldsymbol{\omega}$ measured in the plane's system, the time development of the orientation is

$$\dot{\boldsymbol{q}}(t) = \frac{1}{2}\boldsymbol{q}(t)\boldsymbol{\omega}(t) \tag{3}$$

If the angular velocity sensor had ideal accuracy, no noise and drift, it would be enough to integrate this equation numerically to keep track of the orientation

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}(t) + \dot{\boldsymbol{q}}(t)\Delta t \tag{4}$$

However, unlike the ring laser gyros, the MEMS sensors are far from ideal and even when not moving, they give nonzero readout of ω. Part of it is a constant bias and can be calibrated out, but part of it depends on temperature, time and acceleration and cannot be simply subtracted. To prevent the errors due to this nonideal sensor behavior to accumulate over time, the algorithm performs corrections of the orientation to minimize the difference between a reference vector known in the earth-system rotated to the plane-system by the current orientation quaternion $\boldsymbol{q}(t)$ and its counterpart measured in the plane-system

$$\begin{aligned} F(\boldsymbol{q}) &= \boldsymbol{f}(\boldsymbol{q})^{T}\cdot\boldsymbol{f}(\boldsymbol{q}) \overset{!}{=} \min \\ \boldsymbol{f}(\boldsymbol{q}) &= \boldsymbol{q}^{*}\boldsymbol{v}_{\text{ref,earth}}\boldsymbol{q} - \boldsymbol{v}_{\text{ref,plane}} \end{aligned} \tag{5}$$

This minimization can be achieved iteratively using a gradient descent

$$\boldsymbol{q}_{n+1} = \boldsymbol{q}_{n} - \beta'\nabla_{\boldsymbol{q}}F(\boldsymbol{q}) \tag{6}$$

where $\beta' > 0$ is a step size parameter. Since the minimization need not be completed at one time instant, as the orientation changes anyway, it is more practical to join the time propagation (3,4) and minimization (6) into a single orientation update equation, which for N_{ref} reference vectors yields

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}(t) + \Delta t\left[\frac{1}{2}\boldsymbol{q}(t)\boldsymbol{\omega}(t) - \sum_{i=1}^{N_{\text{ref}}}\beta_i\nabla_{\boldsymbol{q}}F_i(\boldsymbol{q}(t))\right] \tag{7}$$

Subsequently $\boldsymbol{q}(t+\Delta t)$ is normalized at each step, to prevent numerical deviation from a normalized quaternion which represents a rotation.

In order to determine the orientation uniquely, at least two reference vectors are needed. One of them is the normalized gravity vector $\boldsymbol{v}_{\text{ref}_1,\text{earth}} \equiv \boldsymbol{g}_{\text{earth}} = (0,0,-1)$. The negative sign corresponds to gravity force in $+z$ direction being locally equivalent to acceleration in $-z$ direction, which corresponds to what is measured. For determining the corresponding $\boldsymbol{g}_{\text{plane}}$ see the next section. The second reference is provided by the normalized magnetic field vector $\boldsymbol{m}_{\text{earth}} = (m_x, 0, m_z)$. However, since this vector should only determine the azimuth (yaw) and fluctuations in its m_z component could actually spoil pitch and roll values, it is better to take as a second reference its vector product with gravity $\boldsymbol{v}_{\text{ref}_2,\text{earth}} = \boldsymbol{g}_{\text{earth}} \times \boldsymbol{m}_{\text{earth}}$, as suggested in [3]. In principle, one could derive the orientation from the two references only, not taking the gyroscope into account at all, but this would be unreliable during turning maneuvers due to neglect of $d\boldsymbol{\omega}(t)/dt$ and other approximations below. A combination of the integration of $\boldsymbol{\omega}(t)$ and gradient correcting steps is thus needed, as given by (7) possibly with a dynamic scaling of the gradient step parameters β_i, giving more weight to the gradient term of gravity reference in straight flight and less weight when aerobatic maneuvers are detected (large $\boldsymbol{\omega}(t)$, large pitch and roll angles).

In order to compute the orientation update (7) numerically, one has to determine the respective gradients of the target functions (5)

$$\nabla_{\boldsymbol{q}} F(\boldsymbol{q}) = 2f(\boldsymbol{q})^T \cdot \nabla_{\boldsymbol{q}} f(\boldsymbol{q}) = 2(\nabla_{\boldsymbol{q}} f(\boldsymbol{q}))^T \cdot f(\boldsymbol{q}) \tag{8}$$

The rotation by a normalized quaternion $\boldsymbol{q}$ can equivalently be expressed by a rotation matrix [9]

$$\boldsymbol{q}^* \boldsymbol{v} \boldsymbol{q} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{v} \tag{9}$$

where

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} (q_0^2+q_1^2-q_2^2-q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_1q_3+q_0q_2) \\ 2(q_1q_2+q_0q_3) & (q_0^2-q_1^2+q_2^2-q_3^2) & 2(q_2q_3-q_0q_1) \\ 2(q_1q_3-q_0q_2) & 2(q_2q_3+q_0q_1) & (q_0^2-q_1^2-q_2^2+q_3^2) \end{bmatrix} \tag{10}$$

Then

$$f(\boldsymbol{q}) = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{v}_{\text{ref,earth}} - \boldsymbol{v}_{\text{ref,plane}} \tag{11}$$

and

$$\nabla_{\boldsymbol{q}} f(\boldsymbol{q}) = \nabla_{\boldsymbol{q}} \boldsymbol{M}(\boldsymbol{q}) \cdot \boldsymbol{v}_{\text{ref,earth}} \tag{12}$$

where the quaternion-gradient of the rotation matrix $\nabla_{\boldsymbol{q}} \boldsymbol{M}(\boldsymbol{q})$ can be considered as a matrix of quaternions or a quaternion with matrix-valued components, where the latter is more convenient in the templated C++ implementation. Taking derivatives of (10) one easily gets

$$\frac{\partial \boldsymbol{M}(\boldsymbol{q})}{\partial q_0} = \begin{bmatrix} q_0 & q_3 & -q_2 \\ -q_3 & q_0 & q_1 \\ q_2 & -q_1 & q_0 \end{bmatrix} \tag{13}$$

$$\frac{\partial \boldsymbol{M}(\boldsymbol{q})}{\partial q_1} = \begin{bmatrix} q_1 & q_2 & q_3 \\ q_2 & -q_1 & q_0 \\ q_3 & -q_0 & -q_1 \end{bmatrix} \tag{14}$$

$$\frac{\partial \boldsymbol{M}(\boldsymbol{q})}{\partial q_2} = \begin{bmatrix} -q_2 & q_1 & -q_0 \\ q_1 & q_2 & q_3 \\ q_0 & q_3 & -q_2 \end{bmatrix} \tag{15}$$

$$\frac{\partial \boldsymbol{M}(\boldsymbol{q})}{\partial q_3} = \begin{bmatrix} -q_3 & q_0 & q_1 \\ -q_01 & -q_3 & q_2 \\ q_1 & q_2 & q_3 \end{bmatrix} \tag{16}$$

$$\tag{17}$$

Finally, the quaternion gradient of the minimized function is

$$\begin{aligned}\nabla_{\boldsymbol{q}} F(\boldsymbol{q}) &= 2f(\boldsymbol{q})^T \cdot \nabla_{\boldsymbol{q}} \boldsymbol{M}(\boldsymbol{q}) \cdot \boldsymbol{v}_{\text{ref,earth}} \\ &= 2[\boldsymbol{M}(\boldsymbol{q})\boldsymbol{v}_{\text{ref,earth}} - \boldsymbol{v}_{\text{ref,plane}}]^T \cdot \nabla_{\boldsymbol{q}} \boldsymbol{M}(\boldsymbol{q}) \cdot \boldsymbol{v}_{\text{ref,earth}}\end{aligned}\tag{18}$$

In the articles [2, 3] the authors developed explicit formulas for the special cases of sparse reference vectors (z-only and y-only components, respectively) to save floating operations. As it turned out that on STM32F373 at 72MHz I can achieve high enough update frequency with a general (and more elegant) implementation, I did not implement the special cases.

2.2 Coordinate frames and acceleration in the non-inertial system

The papers [1, 2, 3] did not consider application of their algorithm in aviation and thus supposed that the accelerometer measures, except for accelerations of very short duration, only the gravity. This is not the case during flight, where long-time accelerations in a flight along a curved path occur and cannot be neglected. In order to account for the non-inertial forces during turning flight, it is instructive to consider an auxiliary non-inertial frame which only rotates with respect to the (translated) earth frame and in which the plane is fixed at a given instant. In other words, the distance of the center of this system from the plane will be the turning radius given as reciprocal of the plane's trajectory at a given point of its trajectory. Then the whole speed of the plane with respect to earth system $\boldsymbol{v}_{\text{i}}$ comes from the rotation of this auxiliary system

$$\boldsymbol{v}_{\text{i}} = \boldsymbol{\omega} \times \boldsymbol{r}_{\text{turning}}\tag{19}$$

where the angular speed $\boldsymbol{\omega}$ is identical to the one measured in the plane's coordinate system, which is just translated with respect to the auxiliary system, the speed of the plane with respect to the auxiliary system $\boldsymbol{v}\text{r}$ vanishes by definition, and $\boldsymbol{r}_{\text{turning}}$ is the position of the plane in the auxiliary system. Then one can employ the formula for non-inertial acceleration in a rotating frame [5]

$$\boldsymbol{a}_{\text{r}} = \boldsymbol{a}_{\text{i}} - 2\boldsymbol{\omega} \times \boldsymbol{v}_{\text{r}} - \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \boldsymbol{r}_{\text{turning}}) - \frac{d\boldsymbol{\omega}}{dt} \times \boldsymbol{r}_{\text{turning}}\tag{20}$$

Considering $\boldsymbol{v}_{\text{r}} = 0$, using (19) and neglecting $d\boldsymbol{\omega}/dt$ one obtains accelerations in the rotating system

$$\boldsymbol{a}_{\text{r}} = \boldsymbol{a}_{\text{i}} - \boldsymbol{\omega} \times \boldsymbol{v}_{\text{i}}\tag{21}$$

where the inertial accelerations $\boldsymbol{a}_{\text{i}}$ combines gravity and linear acceleration $\boldsymbol{a}_{\text{linear}}$ of the plane. Notice, however, that the accelerometer readout gives $\boldsymbol{a}\text{measured} = -\boldsymbol{a}\text{r}$ since it aims to provide the accelerations of the non-inertial system, which are in opposite direction to the apparent non-inertial forces experienced by objects moving with the system. The measured acceleration thus adds the centripetal acceleration

$$\boldsymbol{a}_{\text{measured}} = \boldsymbol{g} + \boldsymbol{a}_{\text{linear}} + \boldsymbol{\omega} \times \boldsymbol{v}_{\text{i}}\tag{22}$$

in accordance with the result in [4]. Gravity is locally indistinguishable from acceleration in opposite direction, so the the gravity reference vector corresponding to non-moving accelerometer readout is $\boldsymbol{g}_{\text{earth}} = (0, 0, -g)$ in the Earth frame. In the plane's coordinate system, it can thus be recovered as

$$\boldsymbol{g}_{\text{plane}} = \boldsymbol{a}_{\text{measured}} - \boldsymbol{a}_{\text{linear}} + \boldsymbol{\omega} \times \boldsymbol{v}_{\text{i}}\tag{23}$$

where all vectors are expressed in the plane's reference system. The quantities $\boldsymbol{a}_{\text{measured}}$ and $\boldsymbol{\omega}$ are directly measured in plane's system, while $\boldsymbol{a}_{\text{linear}}$ and $\boldsymbol{v}_{\text{i}}$ are derived from time change of the GPS position and have to be transformed from the earth reference frame to the current orientation of the plane.

3 Results

Reference [13] gives an open source implementation of the described algorithm and an open hardware experimental device, presently at “alpha-stage” of development. In a later version of these notes I plan to give results of the performance of the algorithm/device during a flight in a light sport plane and compare to results of previous works by Ma et al. [11] and Johansen et al. [12].

References

[1] Sebastian O.H. Madgwick: An efficient orientation filter for inertial and inertial/magnetic sensor arrays, 2010

[2] Marcel Admiraal, Samuel Wilson, Ravi Vaidyanathan: Improved Formulation of the IMU and MARG Orientation Gradient Descent Algorithm for Motion Tracking in Human-Machine Interfaces, in 2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2017)

[3] Samuel Wilson, Henry Eberle, Yoshikatsu Hayashi, Sebastian O.H. Madgwick, Alison McGregor, Xingjian Jing, Ravi Vaidyanathan: Formulation of a new gradient descent MARG orientation algorithm: Case study on robot teleoperation, Mechanical Systems and Signal Processing 130, 183-200 (2019)

[4] R.P.G. Collinson, ”Introduction to Avionics Systems”, 3rd edition, Springer, Dordrecht Heidelberg London New York 2011 ISBN 978-94-007-0707-8 DOI 10.1007/978-94-007-0708-5.

[5] https://en.wikipedia.org/wiki/Rotating_reference_frame

[6] Simon L. Altmann: Rotations, quaternions, and double groups, Oxford University Press 1986

[7] https://en.wikipedia.org/wiki/Quaternion

[8] https://en.wikipedia.org/wiki/Euler_angles

[9] https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19770024290.pdf

[10] https://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles

[11] Der-Ming Ma, Jaw-Kuen Shiau, I.Chiang Wang, Yu-Heng Lin: Attitude determination using a MEMS-based flight information measuring unit, Sensors **12**, 1–23 (2012).

[12] Tor A. Johansen, Andrea Cristofaro, Kim Sorensen, Jakob M. Hansen, Thor I. Fossen: On estimation of wind velocity, angle-of-attack and sideslip angle of small UAVs using standard sensors

[13] http://www.pittnerovi.com/jiri/hobby/electronics/avionics