2013 International Conference on Unmanned Aircraft Systems (ICUAS)
May 28-31, 2013, Grand Hyatt Atlanta, Atlanta, GA

Experimental Validation of a Quaternion-based Attitude Estimation with Direct Input to a Quadcopter Control System

Øyvind Magnussen, Morten Ottestad and Geir Hovland

Abstract— **This paper presents a method to calculate the attitude quaternion of a quadcopter with few calculations. The quaternion calculation is based on accelerometers and gyroscopes from an Inertial Measurement Unit (IMU). The quaternion from the accelerometer is calculated as the shortest rotation arc from the gravity vector in the navigation frame. The quaternion from the gyroscope is calculated based on equations of the quaternion derivative. A complementary filter is combining the two quaternions with a componentwise comparison. The attitude estimation is calculated without any trigonometric functions. The quaternion is directly used as an input to the attitude controller. The attitude controller is a PD controller running at 400Hz. A model of the quadcopter in Matlab verified that the control system worked as intended. The estimator was verified with a Stewart platform, by mounting the quadcopter on top of it and comparing the angles from the Stewart platform with the angles from the filter. Finally the algorithms were implemented on a quadcopter controller board, and the attitude estimator were compared with the attitude estimation from a high-end IMU from MicroStrain. The complete control system was also tested on a 8-bit microcontroller running at 16 MHz. The relatively slow processor on the microcontroller was also able to do every calculations within 2.5ms.**

I. INTRODUCTION

Quadcopters are popular for hobby-enthusiasts as well as academic research in control of Unmanned Aerial Vehicle (UAVs) due to their simple and low-cost design. Industrial use is currently limited, but potential applications are for example search-and-rescue, surveillance, movie recording, etc.

The quadcopter is an interesting test-bed for academic research since the dynamics is open-loop unstable and a feedback controller with an attitude estimator is required to stabilize and operate it. Such a controller must be able to handle all six degrees of freedom (DOF). A common 6-DOF representation consists of the X,Y,Z positions as well as the roll, pitch and yaw (RPY) angles relative to ground. However, the RPY angles suffer from singularities which may become an issue with quadcopters due to the possibility of acrobatic motions with large angles. An alternative 6-DOF representation is to use the X,Y,Z positions combined with a quaternion to represent the orientation. Quaternions are also used due to the reduced number of floating-point calculations which is a benefit on low-cost, low-weight microcontroller hardware.

Quaternion estimation of the attitude is often done by a variety of Kalman filters. The attitude estimation in [2] is computed using a multiplicative extended Kalman filter. The estimation is based on information from accelerometers, gyroscopes and a GPS. In [1] an unscented Kalman filter is proposed. Both of the filters are efficient, but also require significant of computational power.

The authors are with Faculty of Engineering and Science, Department of Engineering, University of Agder, Norway oyvind.magnussen@uia.no

Attitude estimation is also used in human body motion tracking [11], [12]. These filters rely on slow moving objects and are not suited for UAVs.

In [8] a quaternion estimation for a quadcopter is presented. This is similar to the one presented in this paper, but involves more complicated mathematics. Their control system is also similar to the one in this paper, but involves trigonometric functions to generate the angle setpoints. A complimentary filter design for attitude estimation is also presented in [3]. An advantage of that approach is that the bias of the gyroscope is also estimated, but the filter can be complicated to implement on a microcontroller.

In [9], [10] a new quaternion attitude controller for a quadcopter is proposed. As noted the proposed controller is based upon the compensation of the Coriolis and gyroscopic torques and the use of a PD^2 feedback structure, where the proportional action is in terms of the vector quaternion and the two derivative actions are in terms of the airframe angular velocity and the vector quaternion velocity. The quaternion is extracted from the rotation matrix, while in this paper the quaternion is directly generated from the accelerometer and gyroscopes.

In this paper a new approach is taken, which is more computationally efficient to compared other methods. The estimator is based on the combination of sensor data from a gyroscope and an accelerometer combined with a complementary filter. The new approach for estimating the quaternion is used directly in a quaternion-based attitude controller, without the need for any trigonometric functions.

II. QUADCOPTER TESTBED

The quadcopter used for testing is a DJI Flame Wheel F450 quadcopter, Fig. 1. The total lifting capacity of the DJI is approximately 1600 grams. The control system was implemented MultiWii Mega, a low-cost flight controller from diymulticopter.com.

The flight controller is equipped with an MPU6050 IMU. The accelerometer is measuring the acceleration of the quadcopter including the gravity. It is assumed that the gravity is much greater than the body accelerations of the quadcopter, and can be used to estimate the attitude.

Fig. 1. *DJI quadcopter used for testing*

The roll and pitch setpoint angles of the DJI are controlled by a RC transmitter, where the receiver is directly connected to the flight controller.

III. MODELLING

The quadcopter reference frame is the body frame denoted with a subscript 'b'. The navigation frame is a fixed frame denoted with a subscript 'n'. Positive axes and motor torques are shown in Fig. 2. The quadcopter is modeled using

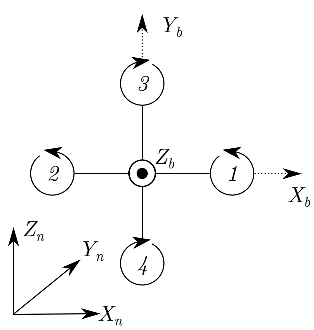

Fig. 2. *Quadcopter model with positive axes and motor torques*

quaternions described in [7]. The quaternion has the benefit of fast calculations and singularity-free representation. The quaternion is defined as:

$$q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos(\frac{\beta}{2}) \\ \sin(\frac{\beta}{2})\, u_1 \\ \sin(\frac{\beta}{2})\, u_2 \\ \sin(\frac{\beta}{2})\, u_3 \end{bmatrix} \tag{1}$$

where β is the angle to rotate, and $u = [u_1\ u_2\ u_3]^T$ is the unit vector describing the axis to rotate about. It's inverse is defined as

$$q^{-1} = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \end{bmatrix}^T \tag{2}$$

Multiplication of two quaternions q and r is denoted with the symbol $\otimes$:

$$q \otimes r = \begin{bmatrix} r_0q_0 - r_1q_1 - r_2q_2 - r_3q_3 \\ r_0q_1 + r_1q_0 - r_2q_3 + r_3q_2 \\ r_0q_2 + r_1q_3 + r_2q_0 - r_3q_1 \\ r_0q_3 - r_1q_2 + r_2q_1 + r_3q_0 \end{bmatrix} \tag{3}$$

A 3×1 vector v can be rotated in 3D space to the vector v' using a quaternion by extending v to a 4×1 vector by adding a leading zero.

$$v' = q \otimes \begin{bmatrix} 0 \\ v \end{bmatrix} \otimes q^{-1} \tag{4}$$

where the new rotated vector v' is the last three elements of the multiplication result.

The derivative of the quaternion is defined as

$$\dot{q} = \frac{1}{2} q \otimes \begin{bmatrix} 0 \\ \omega_b \end{bmatrix} \tag{5}$$

where ω_b is a 3×1 vector describing angular velocities in rad/sec.

A conversion from a given quaternion to Euler Angles, roll (ϕ), pitch (θ), yaw (ψ) is defined as

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \tan^{-1}(\frac{2(q_0q_1+q_2q_3)}{1-2(q_1^2+q_2^2)}) \\ \sin^{-1}(2(q_0q_2 - q_3q_1)) \\ \tan^{-1}(\frac{2(q_0q_3+q_1q_2)}{1-2(q_2^2+q_3^2)}) \end{bmatrix} \tag{6}$$

Forces acting on the quadcopter are the gravity F_g in the Z-direction of the navigation frame, propeller thrust F_{bz} in the body Z-direction and the gyroscopic force from the angular velocity ω_b. Blade flapping, wake interaction and any other effects caused by the translational velocity are ignored. Summing forces F_b in the body coordinate system yields:

$$\sum F_b = m\dot{V}_b$$

$$\dot{V}_b = \frac{1}{m} \begin{bmatrix} 0 \\ 0 \\ F_{bz} \end{bmatrix} + q^{-1} \otimes \begin{bmatrix} 0 \\ 0 \\ 0 \\ g \end{bmatrix} \otimes q - \omega_b \times V_b \tag{7}$$

$$\dot{V}_n = q \otimes \dot{V}_b \otimes q^{-1} \tag{8}$$

$$\tag{9}$$

where m is the mass of the quadcopter, $\dot{V}$ denotes the acceleration and V denotes the velocity in the defined frames.

Moments acting on the quadcopter are the one from different motor thrust, rotational torque from the motor due to the propeller and the gyroscopic force. The induced motor torque due to motor acceleration is neglected since it is difficult to measure. Summing the moments M_b in the body coordinate system yields:

$$\sum M_b = I_b\dot{\omega}_b$$

$$\dot{\omega}_b = I_b^{-1} \left(\begin{bmatrix} M_{bx} \\ M_{by} \\ M_{bz} \end{bmatrix} - \omega_b \times (I_b\omega_b) \right) \tag{10}$$

where I_b is the inertia of the quadcopter and M_b is the moment about each body coordinate axis from the propeller thrust:

$$\begin{bmatrix} M_{bx} \\ M_{by} \\ M_{bz} \end{bmatrix} = \begin{bmatrix} 0 & 0 & d_m & -d_m \\ -d_m & d_m & 0 & 0 \\ T_q & T_q & -T_q & -T_q \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} \tag{11}$$

where d_m is the distance from the quadcopters center of gravity to the center of the motors. There is a linear relationship between the thrust of the propellers and the torque which is produced [6], T_q is the coefficient describing this relationship.

IV. ATTITUDE ESTIMATION

The proposed attitude estimator is only estimating the roll and pitch angles. The roll and pitch angles are also the most critical angles to control for a stable flight. The attitude estimator is a complementary filter that combines two different quaternions, one from the gyroscopes and one from the accelerometers, into one filtered quaternion.

A. Gyroscope Quaternion

The quaternion from the gyro, q^g, is calculated as a time integration of the quaternion derivative from (5)

$$q^g_{k+1} = q_k + t_s \dot{q}^g_k \tag{12}$$

where k denotes the discrete time step, q_k denotes the estimated quaternion from the complementary filter, t_s is the sample rate of the system and $\dot{q}^g_k$ is the time derivative of the quaternion calculated as in (5). The initial value of q^g_0 has to be set to a normalized quaternion, i.e. $[1\ 0\ 0\ 0]^T$, or ideally the quaternion from the accelerometer since this will result in a more correct initial value.

B. Accelerometer Quaternion

The accelerometer is attached to the quadcopter body frame and its values describes the quadcopter body orientation vector relative to the gravity in the navigation frame, given a steady state such as hovering. Aggressive maneuovers may affect the accelerometer with other forces like centripetal force and give a false reading of the gravity vector. For any given vectors v_1 and v_2, where $v_1 \neq -v_2$ and $||v_1|| = ||v_2|| \neq 0$, the quaternion for the shortest rotation arc between those vectors can be described as [5]:

$$q_0 = \frac{1}{||v_1||}\sqrt{\frac{||v_1||^2 + v_1^T v_2}{2}} \tag{13}$$

$$q_{1:3} = \frac{1}{||v_1||}\sqrt{\frac{1}{2(||v_1||^2 + v_1^T v_2)}} \cdot v_2 \times v_1 \tag{14}$$

By using the normalized accelerometer vector and the fixed gravity vector in the navigation frame $r_{nz} = [0\ 0\ 1]^T$, (13) and (14) can be used to calculate the quaternion of the quadcopter relative to the fixed navigation frame. The method which calculates the shortest rotation arc between the two vectors encounters a singularity for a 180 degrees rotation. At this situation there is no shortest rotation arc and the calculation results in dividing by zero. A workaround can be implemented using if-else statements.

Inserting the normalized accelerometer vector $\widetilde{a}_b$ and the gravity vector $r_{nz} = [0\ 0\ 1]^T$ in (13) and (14) yields

$$q^a_0 = \sqrt{\frac{1 + \widetilde{a}_{nz}}{2}} \tag{15}$$

$$q^a_{1:3} = \frac{1}{\sqrt{2(1 + \widetilde{a}_{bz})}} \cdot \begin{bmatrix} \widetilde{a}_{by} \\ -\widetilde{a}_{bx} \\ 0 \end{bmatrix} \tag{16}$$

C. Filtered Quaternion

The gyroscope can provide a quaternion with fast and smooth angle updates. The quaternion from the gyroscope, q^g is calculated as the rotation from its initial position based on the angular velocity, and is not referred to a fixed reference frame. The gyroscope is also affected by a bias which means that the quaternion q^g will drift over time if not compensated for.

The accelerometer however will always have a fixed reference frame i.e. the navigation frame. But the accelerometer is sensitive to noise and airframe vibrations and will not be able to estimate the attitude smooth and fast enough in order to stabilize the quadcopter.

The proposed filter will combine q^g with q^a and compensate for both the drifting and the noise. Assume that the output quaternion is mainly based on q^g. As q^g starts to drift q^g will be significant different than q^a. q^g should then be rotated back towards q^a. Rotating q^g completely back to q^a in one operation makes q^g equal to q^a. Since q^a is has a lot of noise, q^g should only be rotated enough to compensate for the drifting. Quaternion rotations as described in (4) can be time consuming if implemented on a microcontroller. A much simpler way of doing it is to componentwise compare the two quaternions q^g and q^a and calculate the difference q^e:

$$q^e = q^a - q^g \tag{17}$$

$$q^e = \begin{bmatrix} q^a_0 - q^g_0 \\ q^a_1 - q^g_1 \\ q^a_2 - q^g_2 \\ q^a_3 - q^g_3 \end{bmatrix} \tag{18}$$

q_e is not a quaternion saying how much to rotate q^g in order to get to q^a. It is only the componentwise difference and is the correction needed to be added to q^g in order to become q^a. By implementing a scaling factor P_{err}, the total difference to compensate for at each cycle is reduced. This reduces the noise from the accelerometer.

The estimated quaternion q of the filter becomes:

$$\begin{aligned} q &= q^g + P_{err} q^e \\ q &= q^g + P_{err}(q^a - q^g) \\ q &= q^g \cdot (1 - P_{err}) + q^a \cdot P_{err} \end{aligned} \tag{19}$$

where $P_{err} \in [0, 1]$ has to be set as low as possible to reduce the noise from the accelerometer, but high enough to correct for the drifting of the gyroscope. Setting $P_{err} = 0$ eliminates the accelerometer and $P_{err} = 1$ eliminates the gyroscope. The needed value of P_{err} will be dependent on factors such as gyroscope bias, cycle time of the system, accelerometer noise etc. An initial value of P_{err} can be found by monitoring the estimated quaternion with no movement. P_{err} should then be reduced until the estimated quaternion starts to drift and then increased back to no drifting. A block diagram of the filter is presented in Fig. 3.

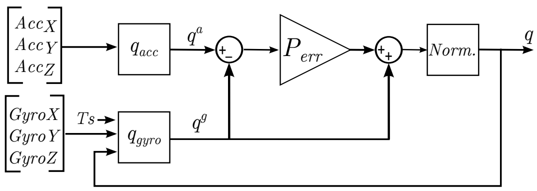

Fig. 3. *Attitude estimator*

where the normalization block is given by

$$q_{norm} = \frac{q}{||q||} \tag{20}$$

and the blocks q_{gyro} and q_{acc} are given by (12) and (15)-(16) respectively. The normalization is very important as numerical errors will make the length of the quaternion different from one over time.

V. CONTROL ARCHITECTURE

For a XYZ rotation matrix the quaternion can be written as:

$$\mathbf{q} = \begin{bmatrix} \cos(\frac{\phi}{2})\cos(\frac{\theta}{2})\cos(\frac{\psi}{2}) + \sin(\frac{\phi}{2})\sin(\frac{\theta}{2})\sin(\frac{\psi}{2}) \\ \sin(\frac{\phi}{2})\cos(\frac{\theta}{2})\cos(\frac{\psi}{2}) - \cos(\frac{\phi}{2})\sin(\frac{\theta}{2})\sin(\frac{\psi}{2}) \\ \cos(\frac{\phi}{2})\sin(\frac{\theta}{2})\cos(\frac{\psi}{2}) + \sin(\frac{\phi}{2})\cos(\frac{\theta}{2})\sin(\frac{\psi}{2}) \\ \cos(\frac{\phi}{2})\cos(\frac{\theta}{2})\sin(\frac{\psi}{2}) - \sin(\frac{\phi}{2})\sin(\frac{\theta}{2})\cos(\frac{\psi}{2}) \end{bmatrix} \tag{21}$$

Equation (21) clearly shows that each element in the quaternion q is dependent of a rotation around every of the three axes. By assuming small angles the sine- and cosine function can be linearized by the use of these formulas:

$$\cos(\sigma) = 1 \tag{22}$$

$$\sin(\sigma) = \sigma \tag{23}$$

$$\sin(\sigma)\sin(\sigma) = 0 \tag{24}$$

$$\cos(\sigma)\cos(\sigma) = 1 \tag{25}$$

Applying the rules of (22)-(25) in (21) yields

$$q_{1:3} = \begin{bmatrix} \frac{\phi}{2} \\ \frac{\theta}{2} \\ \frac{\psi}{2} \end{bmatrix} \tag{26}$$

Small angles are normally in the range of ± 15 degrees. In this case the sine and cosine is for half the angle, hence this approximation is valid for angles up to ± 30 degrees. By use of this linearization $q_{1:3}$ can be used as a control input directly. Since the attitude controller is not estimating the yaw, ψ will always be set to zero. The linearization also shows that it is possible to compare desired angle setpoints directly with the quaternion vector $q_{1:3}$ without generating a new quaternion from the angle setpoints.

The attitude of the quadcopter is controlled by angle setpoints to the controller in terms of a desired quaternion q_d. The controller is a proportional cascade controller with angle and angular velocity feedback as illustrated in Fig. 4. The controller is often used for quadcopters, also for demanding situations [4].

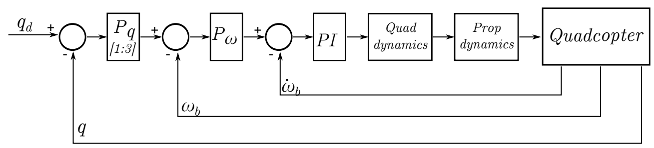

Fig. 4. *Attitude controller*

where q_d is the desired quaternion, q is the estimated attitude quaternion of the quadcopter, ω_b is the angular velocity of the quadcopter's body frame. The block $P_q[1:3]$ extracts the vector part of the quaternion and multiplies it with a gain P_q. The output of the block is the setpoint for the angular velocity. P_ω is the gain of the difference of the angular velocity setpoint and the measured value. "Quad Dynamics" converts the desired setpoint from the controller into forces for each motor using a slightly modified version of (11):

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & d_m & -d_m \\ -d_m & d_m & 0 & 0 \\ T_q & T_q & -T_q & -T_q \\ 1 & 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} M_x \\ M_y \\ M_Z \\ T_h \end{bmatrix} \tag{27}$$

where T_h is the desired total propeller thrust. By adding T_h, the matrix in (11) becomes invertible and the propeller thrusts $F_{1:4}$ can be calculated as in (27).

VI. SIMULATIONS

To verify the controller a quadcopter was modeled in Matlab/Simulink based on equation (7) - (10). The model is shown in Fig. 5 The verification was conducted to see

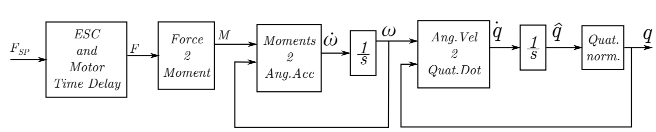

Fig. 5. *Quadcopter model in Matlab/Simulink*

how the quadcopter would behave if the quaternion was used directly as an input, and compare the results with extracted Euler angles. The quadcopter was excited with a step input to the controller with a quaternion equal to $[20, 30, 0]^T$ degrees.

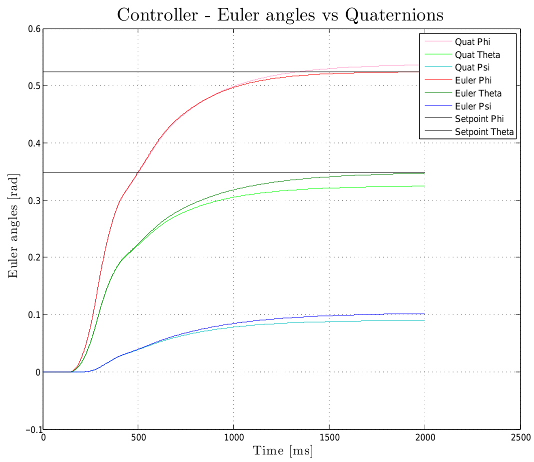

Fig. 6. *Comparison of a quaternion and Euler angle as setpoints to the controller*

Fig. 6 shows that there is a small deviation in the response of the two different input signals. As the angle increases the quaternion set point decreases, showed in (21).

The reduction of the angle is due to the linearization in (21)-(25), but is not critical for the operation of the quadcopter. Angles different from zero are normally only used in transient responses, while the steady-state situation is normally hovering where the angles are zero.

VII. EXPERIMENTS

A. Stewart Platform

The attitude estimator was verified using a Bosch-Rexroth Stewart platform. The Stewart platform has previously been verified by a FARO laser tracker to have approximately 10μ meter in position error. It is therefore assumed that the angle error of the Stewart platform is also very small. Prior to the testing, the offset of the gyros were calculated and compensated for. With little gyroscope offset the value of P_q can be very small. The selected value was $P_q = 0.002$. The low value of P_q results in a smooth signal with low noise. The draw back is that the smaller P_q is, the longer it takes to correct for initial errors.

The quadcopter was placed on top of the Stewart platform, Fig. 7 with aligned axes. The Stewart platform was set to run a sine with an amplitude of 0.17 radians and a frequency of 0.2 Hz. The frequency was set low due to restrictions of the Stewart platform.

The quadcopter attitude was initialized with aligned axes with the navigation frame, $q_0 = [1\ 0\ 0\ 0]^T$. The Stewart platform started its motions prior to starting the quadcopter to induce an offset error. The error was induced to verify that the filter could compensate for it. The estimated quaternion was converted into Euler angles for comparison using (6) and the result is shown in Fig. 8.

After approximately 500ms the estimated attitude of the quadcopter was equal to the attitude of the Stewart platform.

Fig. 7. *Stewart platform used for testing*

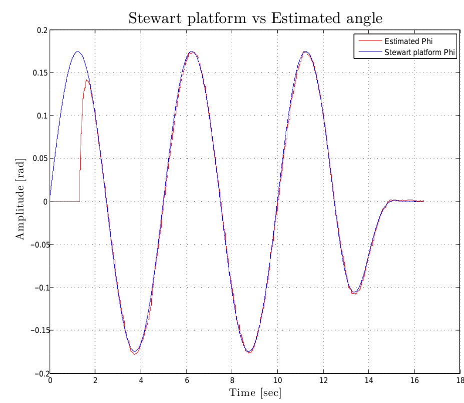

Fig. 8. *Dynamic response of the filter tested on a Stewart platform*

During the whole test the estimator was able to track the attitude of the Stewart platform. It must be noted that this was a very ideal test with very small amplitudes of noise and other disturbances. The test did however verify that the filter works as intended, and gave good results.

B. Real flight comparison

The filter was designed to be good and simple to implement on a low cost controller board for a multicopter. Unlike the the Stewart Platform a multicopter introduces a lot of vibrations to the airframe, and hence the IMU. The IMU and especially the accelerometers, are affected by those vibrations. The MPU6050 has an integrated filter which reduces the unwanted vibrations. A drawback of the MPU6050 is that it does not support individual filter settings for the accelerometers and gyroscopes. The bandwidth of the gyroscopes were set to 42Hz and with that setting the bandwidth of the accelerometers were automatically set to 44Hz. In general it is desired to filter the accelerometer even some more. Since this was not possible to do on the MPU6050 a second filter was implemented on the microcontroller in order to get decent values from the accelerometers. The filter is a first order low pass filter:

$$\hat{a}_{b(n)} = \hat{a}_{b(n-1)} \cdot G_a + a_{(n)} \cdot (1 - G_a) \tag{28}$$

where ˆ denotes the filtered accelerometer value, (n) and $(n-1)$ denotes the new and previous value and $G_a \in [0,1]$ denotes the filter gain. Increasing G_a increases the filter of the accelerometer. A more advanced filter is not desired, or needed, as this complicates the implementation on the microcontroller.

The attitude estimation filter and attitude controller algorithms were implemented on the MultiWii Mega. The efficient algorithms made the loop frequency as high as 400Hz, which includes sensor readings and scaling, attitude estimation, attitude controller and update of the motor speed.

In lack of a motion capture system, such as IR-cameras, the proposed filter was compared with one of the best high-end IMU's available, the 3DM-GX3-25 from MicroStrain. Both sensors were installed on the quadcopter and aligned as good as possible in order to get equal angle values. Aligning three axes with relative primitive equipment is almost impossible. With even a small rotation about one of any of the axes will result in different measurement. Also the two sensors were placed on foam dampers to reduce the mechanical vibrations. The dampers might also result in different movements. A log of the two sensors from a acrobatic real flight are shown in Fig. 9.

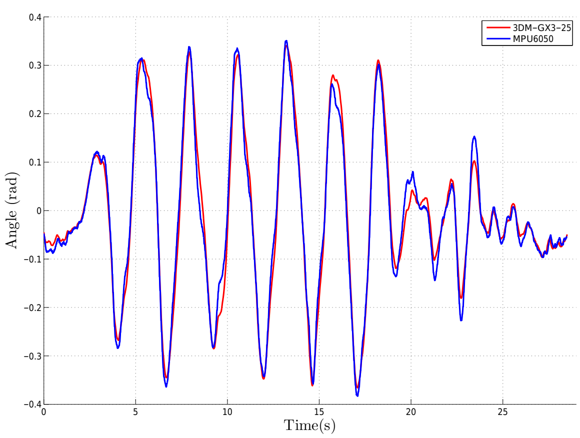

Fig. 9. *Pitch from a flight log with the proposed quaternion estimator implemented on a MultiWii Mega using MPU6050 (blue) with the high-end 3DM-GX3-25 IMU (red).*

Fig. 9 is a log from a real flight from take-off to landing. It shows that the proposed algorithm calculates the attitude close to the the 3DM-GX3-25 sensor. Even flying the quadcopter over a longer period of time shows no indication of drifting of the proposed filter.

VIII. CONCLUSIONS

In this paper a new method to estimate the attitude (roll and pitch) of a UAV was described. The attitude was quaternion based and a calculated quaternion from accelerometers was filtered together with the quaternion from the gyroscopes using a complementary filter. The attitude of the accelerometer was calculated using a formula for the shortest rotation arc from the gravity reference vector. The quaternion from the gyroscope was calculated using time integration of the derivative of the quaternion which is calculated by the angular velocities measured by the gyroscopes. The quaternion were completely estimated without the need of trigonometric functions. The estimated filter is able to dynamically correct errors, and was verified experimentally by a test on a Stewart platform and compared with a high-end IMU from MicroStrain. The gyroscope bias has to be calculated prior to the flight, as the attitude estimator does not estimate it. An enhancement of the proposed filter is to dynamically adapt the filter gain P_{err} for the two quaternions q^a and q^g in flight. One approach is to reduce P_{err} if the module of the sensed accelerometer vector is different from one.

The quaternion was also used for direct input to the control system. For small roll and pitch angles the quaternion vector is a representation of the Euler angles of the quadcopter, given a yaw angle equal to zero. The quadcopters are usually flying for a position setpoint, hence this approximation should not affect the total system. If necessary the approximation can be compensated for by solving two equations from (21).

The proposed filter may not be the best filter when it comes to noise cancellation, but the required computational effort required to calculate the attitude quaternion is very low, and in all a good attitude estimator to stabilize a quadcopter.

REFERENCES

[1] H.G. de Marina, F.J. Pereda, J.M. Giron-Sierra, and F. Espinosa. Uav attitude estimation using unscented kalman filter and triad. *Industrial Electronics, IEEE Transactions on*, 59(11):4465 –4474, nov. 2012.

[2] J.K. Hall, N.B. Knoebel, and T.W. McLain. Quaternion attitude estimation for miniature air vehicles using a multiplicative extended kalman filter. In *Position, Location and Navigation Symposium, 2008 IEEE/ION*, pages 1230 –1237, may 2008.

[3] R. Mahony, T. Hamel, and J.-M. Pflimlin. Complementary filter design on the special orthogonal group so(3). In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, pages 1477 – 1484, dec. 2005.

[4] D. Mellinger, N. Michael, and V. Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. In *Proceedings of the International Symposium on Experimental Robotics*, Dec 2010.

[5] A. Roberts and A. Tayebi. Adaptive position tracking of vtol uavs. *Robotics, IEEE Transactions on*, 27(1):129 –142, feb. 2011.

[6] J. Roskam and C.T.E. Lan. *Airplane Aerodynamics and Performance*. Airplane design and analysis. DARcorporation, 1997.

[7] M.J Sidi. *Spacecraft Dynamics and Control A practical engineering approach*. Cambridge Unversity Press, 2002.

[8] E. Stingu and F. Lewis. Design and implementation of a structured flight controller for a 6dof quadrotor using quaternions. In *Control and Automation, 2009. MED '09. 17th Mediterranean Conference on*, pages 1233 –1238, june 2009.

[9] A. Tayebi and S. McGilvray. Attitude stabilization of a four-rotor aerial robot. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 2, pages 1216 – 1221 Vol.2, dec. 2004.

[10] A. Tayebi and S. McGilvray. Attitude stabilization of a vtol quadrotor aircraft. *Control Systems Technology, IEEE Transactions on*, 14(3):562 – 571, may 2006.

[11] Yun Xiaoping, E.R. Bachmann, and R.B. McGhee. A simplified quaternion-based algorithm for orientation estimation from earth gravity and magnetic field measurements. *Instrumentation and Measurement, IEEE Transactions on*, 57(3):638 –650, march 2008.

[12] Xiaoping Yun and E.R. Bachmann. Design, implementation, and experimental results of a quaternion-based kalman filter for human body motion tracking. *Robotics, IEEE Transactions on*, 22(6):1216 –1227, dec. 2006.