IAC-11.B2.5.8

A COMPARISON OF ATTITUDE DETERMINATION METHODS: THEORY AND EXPERIMENTS

Kristian Lindgård Jenssen

Department of Engineering Cybernetics

Norwegian University of Science and Technology(NTNU)

N-7491 TRONDHEIM, NORWAY

kristian.jenssen@gmail.com

Kaan Huseby Yabar*, Jan Tommy Gravdahl†

Abstract

In this paper two attitude determination methods are derived, implemented and compared. The QUaternion ESTimator (QUEST) algorithm is extended to include non-vectorized measurements, whereas the well known Extended Kalman Filter has been implemented for performance comparison. The methods have been developed for the Norwegian University Test Satellite (NUTS), as a part of the CubeSat project at the Norwegian University of Science and Technology (NTNU). Due to the specifications of CubeSats, both methods are customized for satellites with limited weight-, size- or financial budgets. The attitude estimation is based on two vectorized measurements and data from a gyroscope. Both methods have been developed and simulated in MATLAB. The code have been rewritten using C language. The methods are compared both theoretically and experimentally with implementation and testing on an AVR microcontroller.

Testing indicates that the EKF provides a smoother estimation than the newly developed EQUEST. In contrast to EQUEST, the EKF is able to estimate sensor biases. However, the EQUEST has significantly faster settling time and is less computational costly. Compared to the EKF, EQUEST runs more than 5 times faster. It also requires only 8% of the arithmetic operations of the EKF. Another disadvantage with the EKF is tracking problems that occur when the two vectorized measurements are close to parallel. With vectors close to parallel, the mathematical formulation of the EKF makes tracking of a rotation around the parallel axis extremely difficult. These difficulties are hardly observed in the EQUEST algorithm, which makes it very attractive for attitude estimation.

For small satellites the magnetic field of the Earth is often used for attitude determination. A substantial number of these satellites use magnetorquers for attitude control, affecting the local magnetic field. Hence, control and estimation should not be done simultaneously, resulting in the estimation and control switching on and off. For this reason, the long settling time of the EKF makes the EQUEST even more attractive.

1. BACKGROUND

Attitude determination is an important subsystem in satellites of all sizes. Knowledge of the satellites orientation is crucial to perform space missions such as nadir pointing control. Two common methods to estimate the orientation are the Kalman Filter (KF) and the Quaternion Estimator (QUEST) [1][2][3]. The KF tries to fuse system dynamics, input data and sensor measurements to estimate the best possible attitude, whereas the QUEST calculates the attitude by minimizing a cost function relating sensor measurements with known references.

The equipment used for estimating the attitude vary with price, power consumption and physical size. The attitude estimation systems in this paper are based on both the QUEST and the KF, designed

*Department of Engineering Cybernetics - NTNU, Norway, kayabar@gmail.com

†Department of Engineering Cybernetics - NTNU, Norway, jan.tommy.gravdahl@itk.ntnu.no

especially for CubeSats [4]. A CubeSat is a cube shaped picosatellite where the dimensions has been standardized to 10×10×10 cm, with a weight limited to 1.3 kg. NTNU takes part in a student CubeSat project, designing a double cube - NUTS [5]. As the name indicates, a double cube is 10×10×20 cm, with a weight limited to 2.6 kg. The NUTS (NTNU Test Satellite) project was started in September 2010. The project is part of the Norwegian student satellite program run by NAROM (Norwegian Centre for Space-related Education). The projects goal is to design, manufacture and launch a double CubeSat by 2014. As main payload an IR-camera is planned, as well as a short-range RF experiment. The satellite will fly two transceivers in the amateur radio bands. Ten students were involved in the project first half of 2011.

Two satellites have been developed at NTNU earlier, each containing attitude estimation [6] . Svartveit [7] estimated the attitude by using a discrete Kalman filter based on measurements from magnetometer and sun sensor. The solar panels were used as a crude sun sensor. The experiment Svartveit did, indicated that the sun sensor seems to be inaccurate mainly due to the Earths albedo effect. Ose [8] made further work in order to implement the extended Kalman filter (EKF) in MATLAB. Due to the complexity of the EKF, only a linearized version was implemented on a microcontroller. With respect to all the challenges the previous NTNU students unveiled, this paper base the attitude determination system on a different approach. The estimation is based on two vectorized measurements as well as data from a gyroscope. Sabatini [9] developed an extended Kalman filter based on accelerometer, gyroscope and magnetometer, for use in biomedical engineering. The extended Kalman filter developed in this project is adapted from the work done by Sabatini to fit the satellite.

Attitude can also be estimated using quaternion estimation (QUEST). Markley [3] describes how two vectorized measurements can be used to estimate orientation. However, the method can only be used for vectorized measurements, which makes the gyroscope unsuited. Psiaki [1] has extended the QUEST in order to handle an arbitrary dynamic model and to estimate errors such as rate-gyro biases. The extended QUEST (EQUEST) developed in this paper, is based on the work done by Psiaki and Markley, with focus on integrating the nonvectorized gyroscope measurements.

2. THEORY

2.1. *Coordinate Frames*

The following frames have been used to describe the attitude of the satellite

BODY: This frame is attached to the satellite. In the BODY frame the axes coincide with the principle axes of inertia, and the positive z-axis is defined as the vector pointing outwards from the quadratic side of the satellite. The x- and y-axis are orthogonal to the rectangular sides of the cube.

NED: North-East-Down (NED) is a local reference frame defined relative to Earth with the x-axis pointing towards north, z-axis downwards perpendicular to Earth's reference ellipsoid. The y-axis completes a right handed orthogonal coordinate system, with the positive y-axis pointing towards East. Earth's reference ellipsoid is a mathematically defined surface fitted to approximate the shape of the Earth.

2.2. *Rotation Matrix*

A rotation matrix R is used to describe a rotation between two coordinate frames. It is common to describe the rotation from one frame to another as three consecutive rotations. First a rotation ψ around the z-axis, then a rotation θ around the rotated y-axis and finally a new rotation ϕ around the current x-axis. The entire rotation from frame "a" to frame "b", is described as

$$R_b^a = R_z(\psi)R_y(\theta)R_x(\phi) \qquad [2.1]$$

The rotation matrix can be represented using quaternions [10][11]

$$R(q) = (q_4^2 - \|q_{13}\|^2)I_{3\times3} + 2q_{13}q_{13}^\top - 2q_4[q_{13}\times] \qquad [2.2]$$

A rotation matrix has the following properties

$$det(R) = 1 \qquad [2.3]$$

$$R^\top = R^{-1} \qquad [2.4]$$

A vector in NED frame k^n can be written in BODY frame k^b using rotation matrix as given below

$$k^b = R_n^b k^n \qquad [2.5]$$

2.3. Extended Kalman Filter (EKF)

The Kalman filter is only applicable for linear systems. However, it is possible to extend the filter to also deal with nonlinear problems, but the filter will then lose some of its properties. The EKF is adapted for nonlinear problems through a linearization of the nonlinear equations for each iteration. Due to the linearization, the EKF is not necessarily optimal and can diverge if initial errors are too large or if the system model is inaccurate. The EKF is an iterative method where the estimates are based on several measurements, as well as a process model.

Defining the discrete nonlinear system as

$$x_k = f(x_{k-1}, u_{k-1}) + w_{k-1} \quad [2.6]$$

$$z_k = h(x_k) + v_k \quad [2.7]$$

where x is state vector, $f(.)$ describes the system dynamics, u is control input, w is process noise, h is the measurment model and v is measurement noise. Both measurement and process noise are assumed to be zero mean Gaussian. The subscript k denotes discrete time. Using the system described above, the equations for the extended Kalman filter can be written as [12]

Predict:

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}) \quad [2.8]$$

$$P_k^- = F_k P_k F_k^T + Q_{kal} \quad [2.9]$$

Update:

$$K_k = P_k^- H_k^\top (H_k P_k^- H_k^\top + R_{kal})^{-1} \quad [2.10]$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_k \hat{x}_k^-) \quad [2.11]$$

$$P_k = (I - K_k H_k) P_k^- \quad [2.12]$$

where $\hat{x}$ denotes estimated state vector, P is error covariance matrix, K is calculated Kalman gain and

$$F_k = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1}, u} \quad [2.13]$$

is the derivative of the nonlinear system with respect to the states, and

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k-1}} \quad [2.14]$$

is the derivative of the measurement equations with respect to the states.

As these equations show, the nonlinear model is still used in predicting the new states, whereas a linearized model is used for comparing the measurements with the current states in the update phase and for computing the covariance matrix and the Kalman gain. R_{kal} is the measurement covariance matrix, and Q_{kal} is the process covariance.

2.4. Quaternion Estimation

Another method for estimation of the rotation matrix based on the sensor measurements is the QUaternion ESTimator (QUEST) [1]. QUEST will minimize the cost function defined as

$$J(q) = \frac{1}{2}\sum_{j=1}^{n} \frac{1}{\sigma_j^2}(b_j - R_b^i(q) r_j)^\top (b_j - R_b^i(q) r_j)$$

$$= \frac{1}{2}\sum_{j=1}^{n} \frac{1}{\sigma_j^2}(b_j^T b_j - 2 b_j^\top R_b^i(q) r_j + r_j^\top r_j) \quad [2.15]$$

where r_j are known unit vectors in the NED frame, and b_j are unit vectors of the measured observations in the body-fixed frame. σ_j are the standard deviation of the measurement error. However, both r_j and b_j are unit vectors, and this reduces the equation to

$$J(q) = \sum_{j=1}^{n} \frac{1}{\sigma_j^2}(1 - b_j^\top R_b^i(q) r_j) \quad [2.16]$$

Minimizing J is equivalent to maximizing

$$J_{QUEST}(q) = \sum_{j=1}^{n} \frac{1}{\sigma_j^2} b_j^\top R_b^i(q) r_j \quad [2.17]$$

The QUEST does not depend on initial conditions, which is a great advantage. Another advantage is that the algorithm can be solved exactly by solving an eigenvalue problem. However, the QUEST algorithm can only estimate the attitude quaternions.

2.5. Attitude Estimation Using EKF

The theory of the EKF was explained in Section 2.3. There are several ways to configure the EKF for attitude estimation based on vectorized measurements and a gyroscope. Here, the state vector was chosen to consist of the quaternions q and the biases from both vectorized measurements b_{v1} and b_{v2}. The state vector x is given below

$$x = \begin{bmatrix} q \; b_{v1} \; b_{v2} \end{bmatrix}^\top$$

where

$$q = \begin{bmatrix} q_1 \; q_2 \; q_3 \; q_4 \end{bmatrix}^\top$$

$$b_{v1} = \begin{bmatrix} b_{v1,x} \; b_{v1,y} \; b_{v1,z} \end{bmatrix}^\top$$

$$b_{v2} = \begin{bmatrix} b_{v2,x} \; b_{v2,y} \; b_{v2,z} \end{bmatrix}^\top$$

The great advantage of estimating the biases of the two vectorized measurements is that the EKF will subtract them from the measurements during the next iteration. This will increase the precision of the measurements and give a more accurate attitude estimate. Since all the measurements are done in the BODY frame and the reference model is given in the NED frame, the sensor model includes a rotation matrix $R(q)$. The rotational matrix introduce a non-linearity in the measurement matrix. The measurement model z is given by the two vectorized measurements as

$$z = \begin{bmatrix} v_1^b \; v_2^b \end{bmatrix}^\top \quad [2.18]$$

where

$$v_1^b = R_n^b(q) v_{1,real}^n + b_{v1}^b \quad [2.19]$$

$$v_2^b = R_n^b(q) v_{2,real}^n + b_{v2}^b \quad [2.20]$$

In order to include angular velocity as a state, a dynamical model of the satellite can be used. An alternative is to use the measurements from the gyroscope to estimate the dynamical model for the quaternions. The kinematic differential equation for unit quaternions can be written as [9][10]

$$\dot{q} = \frac{1}{2} \begin{bmatrix} \omega\times & \omega \\ -\omega^\top & 0 \end{bmatrix} q \quad [2.21]$$

where

$$\omega\times = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \quad [2.22]$$

with

$$\omega = \begin{bmatrix} \omega_1 \; \omega_2 \; \omega_3 \end{bmatrix}^\top$$

being the angular velocity measured in the body frame. Further, constant bias can be assumed from the two vectorized measurements:

$$\dot{b}_{v1} = 0 \quad [2.23]$$

$$\dot{b}_{v2} = 0 \quad [2.24]$$

The differential equation for the entire system is therefore [9]

$$\dot{x} = \begin{bmatrix} \dot{q} \; \dot{b}_{v1} \; \dot{b}_{v2} \end{bmatrix}^\top$$

$$= \begin{bmatrix} \frac{1}{2}\begin{bmatrix} \omega\times & \omega \\ -\omega^\top & 0 \end{bmatrix} & 0 \; 0 \\ 0 & 0 \; 0 \\ 0 & 0 \; 0 \end{bmatrix} \begin{bmatrix} q \\ b_{v1} \\ b_{v2} \end{bmatrix} \quad [2.25]$$

Note that the system equation is linear, making the computation of the EKF less complicated. However, an EKF is still necessary due to the nonlinearity in the measurement model.

The system in Equation [2.25], is given in continuous time. For computer implementation, discretization is required. The system is discretized using zero-order hold with sampling time T_s. The discrete system will be given as

$$x_{k+1} = \begin{bmatrix} F & 0 & 0 \\ 0 & I_{3x3} & 0 \\ 0 & 0 & I_{3x3} \end{bmatrix} x_k \quad [2.26]$$

where

$$F = expm\left(\frac{1}{2}\begin{bmatrix} \omega\times & \omega \\ -\omega^\top & 0 \end{bmatrix} T_s\right) \quad [2.27]$$

with $expm$ being the matrix exponential.

Note that the angular velocity is not a part of the measurement vector, nor the state vector. As mentioned earlier, the measurements from the gyroscope will be indirectly integrated in the EKF through the dynamical model of the quaternions.

The vectors explained above are used in the equations in Section 2.3 to iteratively compute the attitude of the satellite.

2.6. Extended Quaternion Estimator

As mentioned in Section 2.4, the QUEST algorithm is not able to utilize the measurements from the gyroscope. However, it is possible to extend the QUEST and implement an Extended QUaternion ESTimator (EQUEST) in order to include the gyroscope measurements. The main idea behind the EQUEST is to modify the cost function. This is done by adding another term, containing the gyroscope measurements.

By tracking the rotation based on gyroscope measurements, it is possible to penalize deviations from the rotation matrix estimated by the gyroscope alone.

$$J_{gyro}(q) = \frac{1}{2}(q - \hat{q}_{gyro})^{\top} D(q - \hat{q}_{gyro}) \qquad [2.28]$$

here $\hat{q}_{gyro}$ is the estimated attitude quaternion based on gyroscope tracking, and D is a diagonal weighting matrix. The main idea by using the term $q - \hat{q}_{gyro}$ is to minimize the cost function. Note that the term $q - \hat{q}_{gyro}$ is not an attitude quaternion.

The EQUEST can be expanded further by adding a prediction term. The prediction is most suitable for applications where it is possible to forecast upcoming orientation based on previous behavior. It can also be used to filter out noise. The slow and predictable change of attitude for the satellite makes it possible to use previous attitude calculations to estimate future orientation. For a short period of time, the attitude change will be minimal. However, as several attitude calculations are done in this period it is possible to establish a linear relation between time and change of attitude. This is illustrated in Figure 2.1. A deviation from the predicted term can be penalized in the cost function by adding the following term

$$J_{pre}(q) = \frac{1}{2}(q - \hat{q}_{pre})^{\top} S(q - \hat{q}_{pre}) \qquad [2.29]$$

where q vector contains attitude quaternions, $\hat{q}_{pre}$ is the predicted attitude based on previous observations, S is the state weight matrix.

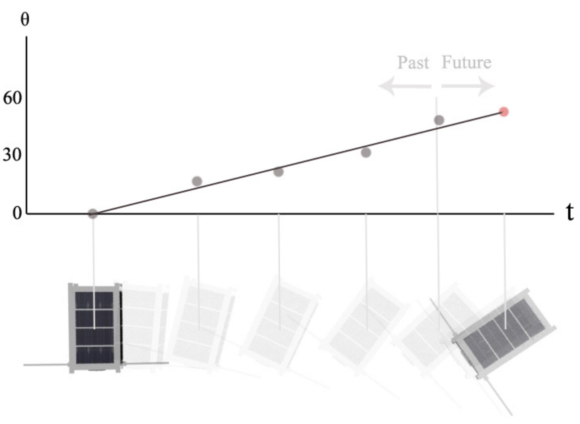

Figure 2.1: Linear prediction based on past orientations. Note that the motion is exaggerated for illustrational purposes.

Extending Equation [2.17] with the two terms described in [2.28] and [2.29] gives

$$\begin{aligned} J_{EQUEST} &= J_{QUEST} + J_{gyro} + J_{pre} \\ &= \frac{1}{2}\sum_{j=1}^{n}\left\{\frac{1}{\sigma_j^2}(b_j - R_b^i(q)r_j)^{\top}(b_j - R_b^i(q)r_j)\right\} \\ &+ \frac{1}{2}(q - \hat{q}_{gyro})^{\top} D(q - \hat{q}_{gyro}) \\ &+ \frac{1}{2}(q - \hat{q}_{pre})^{\top} S(q - \hat{q}_{pre}) \end{aligned} \qquad [2.30]$$

subject to

$$q^{\top} q = 1 \qquad [2.31]$$

It is possible to rewrite both extensions to quadratic terms. By writing the entire equation in quadratic form, the cost function can be minimized using well-known optimization techniques [13].

Note that the EQUEST is still not able to estimate the biases. It is possible to estimate the biases using other methods, and then subtract them from the measurements used in EQUEST. Due to computational costs, this is not considered here.

2.7. Solving the Extended Quaternion Estimator

One option for solving the minimization problem given in [2.30] and [2.31] is to use the Lagrangian multiplier method. However, the cost function has to be written on the special form

$$J(x) = \frac{1}{2}x^{\top} Gx + x^{\top} c \qquad [2.32]$$

where G is a positive definite matrix and c is constant with respect to x. The original QUEST criterion in [2.15] can be posed in the quadratic form [3]

$$g(q) = -q^{\top} Vq \qquad [2.33]$$

where V is a symmetric matrix given by

$$V = \begin{bmatrix} U - \varphi I_{3\times3} & Z \\ Z^\top & \varphi \end{bmatrix} \quad [2.34]$$

with

$$U = L + L^\top \quad [2.35]$$

$$L = \sum_{j=1}^{n} \frac{1}{\sigma_j^2}(b_j r_j^\top) \quad [2.36]$$

$$Z = \begin{bmatrix} L_{23} - L_{32} \\ L_{31} - L_{13} \\ L_{12} - L_{21} \end{bmatrix} \quad [2.37]$$

$$\varphi = trace(L) \quad [2.38]$$

The gyroscope tracking needs to be written in the same quadratic form in order to solve EQUEST with Lagrangian multipliers:

$$J_{gyro}(q) = \frac{1}{2}(q - \hat{q}_{gyro})^\top D(q - \hat{q}_{gyro}) \quad [2.39]$$

$$= \frac{1}{2}(q^\top Dq - q^\top D\hat{q}_{gyro} - \hat{q}_{gyro}^\top Dq + \hat{q}_{gyro}^\top D\hat{q}_{gyro}). \quad [2.40]$$

Since

$$(\hat{q}_{gyro}^\top Dq)^\top = \hat{q}_{gyro}^\top Dq = q^\top D\hat{q}_{gyro} \quad [2.41]$$

we can rewrite [2.39] as

$$J_{gyro}(q) = \frac{1}{2}(q^\top Dq - 2q^\top D\hat{q}_{gyro} + \hat{q}_{gyro}^\top D\hat{q}_{gyro}) \quad [2.42]$$

Further, the term $\hat{q}_{gyro}^T D\hat{q}_{gyro}$ will be constant with respect to q. This term will not affect the minimization problem, hence it can be removed. Now the gyroscope part of the cost function can be written in quadratic form as

$$J_{gyro}(q) = \frac{1}{2}q^\top Dq - q^\top D\hat{q}_{gyro} \quad [2.43]$$

Exactly the same as above can be done to write the prediction term in quadratic form

$$J_{pre}(q) = \frac{1}{2}q^\top Sq - q^\top S\hat{q}_{pre} \quad [2.44]$$

By adding [2.43] and [2.44] with [2.33], the entire EQUEST can be written in quadratic form as:

$$J_{EQUEST}(q) = \frac{1}{2}q^\top(D + S - V)q + q^\top(-D\hat{q}_{gyro} - S\hat{q}_{pre}) \quad [2.45]$$

Introducing new variables

$$\kappa = D + S - V \quad [2.46]$$

$$\xi = -D\hat{q}_{gyro} - S\hat{q}_{pre} \quad [2.47]$$

the problem will be to minimize

$$J_{EQUEST}(q) = \frac{1}{2}q^\top\kappa q + q^\top\xi \quad [2.48]$$

subject to

$$q^\top q = 1 \quad [2.49]$$

The Lagrangian equation is now given as

$$\mathcal{L} = \frac{1}{2}q^\top\kappa q + q^\top\xi + \frac{\lambda}{2}(q^\top q - 1) \quad [2.50]$$

The q that minimizes [2.48] is found as

$$\frac{d\mathcal{L}}{dq} = \kappa q + \xi + \lambda Iq = 0 \quad [2.51]$$

$$q = -(\kappa + \lambda I)^{-1}\xi. \quad [2.52]$$

By combining [2.49] and [2.52], the constraint can be written as

$$\xi^\top(\kappa + \lambda I)^{-2}\xi = 1 \quad [2.53]$$

The largest positive real λ will give the global minimum for Equation [2.48][1]. Further κ is a symmetric matrix, hence it can be factorized as

$$\kappa = M \begin{bmatrix} -\lambda_1 & 0 & 0 & 0 \\ 0 & -\lambda_2 & 0 & 0 \\ 0 & 0 & -\lambda_3 & 0 \\ 0 & 0 & 0 & -\lambda_4 \end{bmatrix} M^\top \quad [2.54]$$

where λ_i are the eigenvalues of κ, and M is an orthogonal eigenvector matrix. By introducing

$$c = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = M^\top\xi \Leftrightarrow \xi^\top = c^\top M^\top, \quad [2.55]$$

[2.53] can be written as

$$c^\top M^\top \left\{ M \begin{bmatrix} -\lambda_1 & 0 & 0 & 0 \\ 0 & -\lambda_2 & 0 & 0 \\ 0 & 0 & -\lambda_3 & 0 \\ 0 & 0 & 0 & -\lambda_4 \end{bmatrix} M^\top + \begin{bmatrix} \lambda & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & \lambda \end{bmatrix} \right\}^{-2} Mc - 1 = 0 \quad [2.56]$$

Because the matrix is diagonal, and the inverse of an orthogonal matrix is the transposed of the orthogonal matrix, this can be simplified to

$$c^\top M^\top \Upsilon^{-2} M c - 1 = 0 \qquad [2.57]$$

where

$$\Upsilon = M \begin{bmatrix} \lambda-\lambda_1 & 0 & 0 & 0 \\ 0 & \lambda-\lambda_2 & 0 & 0 \\ 0 & 0 & \lambda-\lambda_3 & 0 \\ 0 & 0 & 0 & \lambda-\lambda_4 \end{bmatrix} M^\top \qquad [2.58]$$

Now since M is orthogonal, $M^\top M = I$. Further, the inverse of a diagonal matrix can be computed element by element. Equation [2.57] can now be written as

$$c^\top \begin{bmatrix} \frac{1}{(\lambda-\lambda_1)^2} & 0 & 0 & 0 \\ 0 & \frac{1}{(\lambda-\lambda_2)^2} & 0 & 0 \\ 0 & 0 & \frac{1}{(\lambda-\lambda_3)^2} & 0 \\ 0 & 0 & 0 & \frac{1}{(\lambda-\lambda_4)^2} \end{bmatrix} c - 1 = 0 \qquad [2.59]$$

or

$$\frac{c_1^2}{(\lambda-\lambda_1)^2} + \frac{c_2^2}{(\lambda-\lambda_2)^2} + \frac{c_3^2}{(\lambda-\lambda_3)^2} + \frac{c_4^2}{(\lambda-\lambda_4)^2} - 1 = 0 \qquad [2.60]$$

The optimal λ (λ_{opt}) will be larger than the smallest eigenvalue[1]. After λ_{opt} is identified, it can be substituted back into Equation [2.52] to find the q that minimizes the cost function.

In contrast to EKF, EQUEST is not able to estimate the sensor biases. The state vector is therefore chosen to be the quaternion

$$x = q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \qquad [2.61]$$

Equation [2.21] is used to estimate the $\hat{q}_{gyro}$ term in EQUEST. We propose to calculate the $\hat{q}_{pre}$ term by using simple linear regression with a window size of 10 samples. The next quaternion vector is predicted by using the 10 latest samples, and fitting them to an equation for a line

$$y(t) = b_0 t + b_1 \qquad [2.62]$$

b_0 will represent the slope of the line, while b_1 is the measured value at $t = 0$. With n observations, b_0 and b_1 can be found by solving the following formulas [15]

$$b_0 = \frac{n\sum_{i=1}^{n} y(t_i)t_i - \sum_{i=1}^{n} t_i \sum_{i=1}^{n} y(t_i)}{n\sum_{i=1}^{n}(t_i^2) - \left(\sum_{i=1}^{n} t_i\right)^2} \qquad [2.63]$$

$$b_1 = \frac{\sum_{i=1}^{n} y(t_i) \sum_{i=1}^{n} t_i^2 - \sum_{i=1}^{n} t_i \sum_{i=1}^{n} t_i y(t_i)}{n\sum_{i=1}^{n} t_i^2 - \left(\sum_{i=1}^{n} t_i\right)^2} \qquad [2.64]$$

The next $\hat{q}_{pre}$ is estimated using the linear relation found with the last 10 samples using the values b_0 and b_1 as slope parameters. Note that the prediction term does not have to be linear. However, it is chosen to be linear for this particular case. In case of less strict restrictions on calculations power or time, the prediction term might even be an EKF.

3. IMPLEMENTATION

3.1. Microcontroller

In order to test the estimation methods, both EKF and EQUEST were implemented in MATLAB. Both methods were later implemented on microcontrollers. During testing, an 8-bit microcontroller from Atmel was used. In addition to the estimation methods, several important functions were configured. A watchdog timer was implemented to reset the microcontroller in case the software run into an endless loop. Sleep mode for the microcontroller were also configured. The satellite does not have to estimate the attitude all the time. When no attitude data is required, activating the sleep mode can save considerable amount of power.

3.2. Implementing the Kalman Filter in C

In order to implement the extended Kalman filter on the microcontroller, it was written using C language. This introduces some difficulties, as several mathematical operations are not supported in C. First of all, C does not support matrix multiplication which is an essential part of the Kalman filtering. Secondly the matrix inverse operation does not exist. The third challenge is to find the transpose of a matrix. Since the module should be used on-board a satellite, the memory usage should be as optimal as possible. Hence, it is not desirable to implement an entire math library in order to perform the matrix operations. Therefore only the necessary methods have been implemented on the microcontroller.

C supports two-dimensional arrays which looks very similar to a matrix. Using double arrays makes the code more complex. Therefore, matrices were transformed into one dimensional arrays before implementation. An example is given below.

$$\begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix} \Rightarrow \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix}$$

Using 3 for-loops, matrix multiplication can be implemented. For each iteration in the Kalman filter, two matrices must be transposed. These matrices have dimensions 10×10 and 6×10, and can easily be transposed using two for-loops.

Computing the inverse of a matrix is only possible if the matrix is positive-definite. The matrix which is to be inverted in the Kalman filter has properties making it always invertible. It is a 6×6 symmetric matrix, meaning it is Hermetian, and since every eigenvalue in Hermetian matrices is positive, the matrix is positive definite. An efficient way to compute the inverse of a matrix is by performing an LU decomposition. The algorithm for inverse operation using LU decomposition can be found in [16]. The code in this book is optimized for personal computers implying minor changes to adapt it for a microcontroller.

3.3. Implementing the EQUEST in C

The EQUEST was also written using C language. Two challenges arise when the code is rewritten from MATLAB to C: one matrix inversion and one eigenvalue problem. However, it is only a matrix inverse of size 4-by-4. Since the matrix is so small, it is more efficient to invert it using the adjoint method than the LU decomposition. For larger matrices, the adjoint method tends to be computational costly as the number of operations increase as O(n!). To solve the EQUEST, it is necessary to identify the smallest eigenvalue and all eigenvectors of a symmetric 4-by-4 matrix. This is done by implementing the cyclic Jacobi method which returns all eigenvalues and the corresponding eigenvectors of the input matrix [17].

4. RESULTS AND SIMULATIONS

4.1. Extended Kalman Filter

The performance of the algorithms have been evaluated both in expended run time and number of arithmetic operations. The expended run time is found by setting a flag at the start of each cycle, and then resetting the flag after the execution. The run time of the EKF is about 200 milliseconds. By introducing a global counter in the algorithm, it is possible to detect how many arithmetic operations each cycle executes. The linearization contains quite a large amount of numerical operations. On average EKF required about 40 000 operations.

4.2. Extended Quaternion Estimation

Compared to the EKF, the EQUEST requires less matrix multiplications, and only a 4-by-4 matrix inversion. However, the eigenvalues and eigenvectors of a 4-by-4 matrix must be found. The EQUEST algorithm, does not require any linearization. The number of arithmetic operations for the EQUEST was found to be about 3200, which is only 8% of the EKF's operations. The run time for EQUEST is about 40ms. This means that EQUEST is approximately 5 times faster than the EKF. The linear prediction term will have a low-pass filtering effect, as high frequent changes in position will be suppressed. This is illustrated in Figure 4.1. The figure clearly indicates that the EQUEST with linear prediction is much smoother than without prediction.

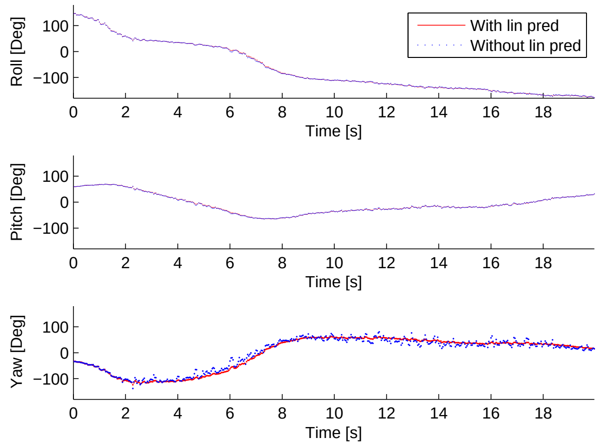

Figure 4.1: EQUEST with and without linear prediction

4.3. Experimental Comparison of the Two Methods

Hardware configuration

For testing purposes, a prototype was designed. The prototype was based on an CHIMU Micro AHRS IMU consisting of a three-axes accelerometer, magnetometer and gyroscope. The accelerometer can not be used onboard the satellite. However, due to test simplicity it was chosen as one of the vectorized measurements for the prototype. Data from the IMU is sent to an AVR ATMEGA2561 microcontroller, where both attitude estimation methods are implemented. A picture of the prototype is given in Figure 4.2.

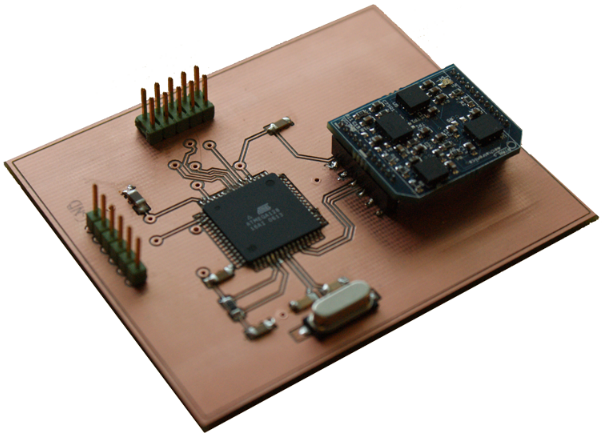

Figure 4.2: Designed prototype

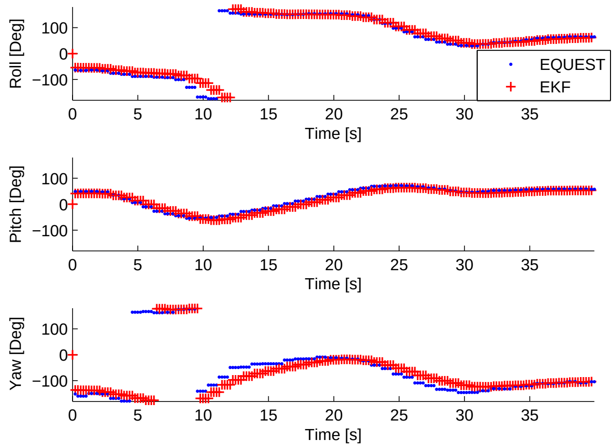

Figure 4.3: EKF vs EQUEST.

The following experiments are done using the prototype running both methods simultaniously. The estimated attitude from the prototype were continiously sent to the computer for plotting through serial communication.

Tracking

To compare the performance of the newly developed EQUEST with the well-known EKF, both methods were implemented on the same microcontroller. They had the same input data and ran simultaneously sending the estimated attitude to a computer. Figure 4.3 indicates that both methods are able to track arbitrary rotations. The figure shows that the estimated orientation is almost identical for both EKF and EQUEST. However, it can be observed that the EQUEST has faster tracking than the EKF. EQUEST and EKF solve the attitude problem in quite different ways. Whereas EQUEST solves the problem in one iteration, the EKF iteratively calculates the solution. Hence, the EQUEST provides a faster estimation for quick orientational changes.

Start-up

One of the greatest differences between EKF and EQUEST is observed during the start-up phase. The EKF uses many iterations to converge towards the correct attitude. For each iteration, the EKF will improve the estimated orientation. The number of iterations used by the EKF will be dependent on its tuning parameters. In addition the start-up phase of the EKF will be greatly influenced by the dissipation of the vectorized measurements. Perpendicular measurement vectors will give the fastest start-up phase. However, the EQUEST will solve an eigenvalue problem to achieve the correct attitude in one iteration. An example of the initial start-up is showed in Figure 4.4. The figure clearly indicates the start-up differences of the methods.

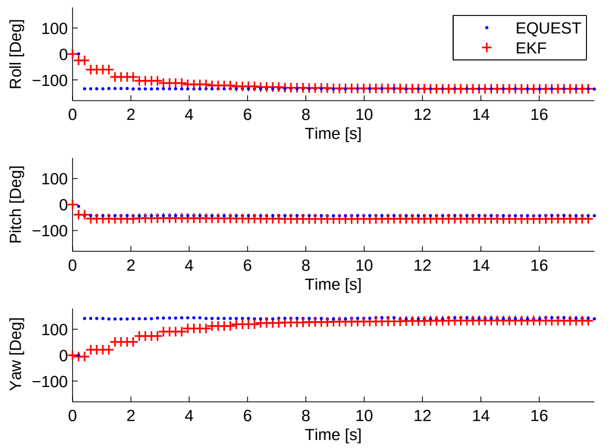

Figure 4.4: Start-up phase of the two algorithms.

Parallel input vectors

The performance of both methods are strictly dependent on the two directional vectors. For optimal results, the vectors should be close to perpendicular. However, in some cases the two vectorized measurements may be close to parallel. This is observed in Figure 4.5, where an accelerometer and a magnetometer were used as vectorized measurements for testing. The normalized acceleration measured in NED coordinates is

$$g = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad [4.1]$$

and the normalized geomagnetic field vector in Trondheim, Norway (N $63.24°$, E $010, 24°$) is

$$B = \begin{bmatrix} 0.2631 \\ 0.0086 \\ 0.9647 \end{bmatrix} \quad [4.2]$$

with the Down-axis being the dominating value. Both vectors are now close to parallel with the down axis. Performing a rotation around this axis, will cause problems for the EKF due to the mathematical formulation. The rotational matrix for rotations around the down axis can be written as

$$R_{down} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad [4.3]$$

Remember that the state update in EKF is given as

$$x_{k+1} = x_k + K\left(z_k - \begin{bmatrix} Rg^\top \\ RB^\top \end{bmatrix} - \begin{bmatrix} b_{v1} \\ b_{v2} \end{bmatrix}\right) \quad [4.4]$$

Here, the state update is a product of the Kalman gain, K, and the deviation from the measurements tracked by the rotational term. A rotation around the down axis is described above. Since the gravitational vector is close to parallel with the magnetic field, the rotational term in Equation [4.4] will now be described as

$$Rg^\top = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad [4.5]$$

$$RB^\top \approx \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad [4.6]$$

clearly, the rotational term fails to identify the rotational angle around the down-axis due to the magnetic field being almost parallel with the gravitational vector. As the measurements are B and g, there will be very small deviations from the measurements to the rotated position.

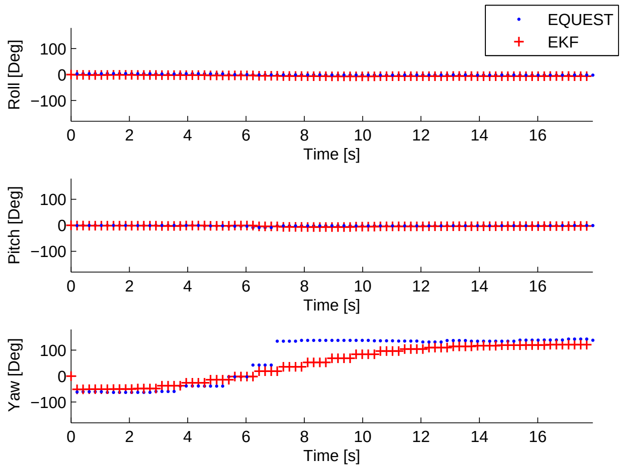

Figure 4.5: The two methods response to a rotation around the down-axis in the NED frame.

5. CONCLUSION

In this paper, two methods were derived, tested and compared. The QUEST algorithm has been developed further to EQUEST, to include non-vectorized measurements in the estimations. An Extended Kalman Filter was fitted for the CubeSat project at NTNU, and implemented with a sensor fusion model instead of a dynamical model of the satellite.

Testing indicates that both methods are able to estimate the orientation using a single microcontroller. The methods were compared theoretically and experimently. In contrast to EKF, the EQUEST is unable to estimate the sensor biases. However, the EQUEST uses only 8% of the arithmetical operations required by the EKF. The runtime of the EQUEST was measured to be more than 5 times faster than the EKF.

NUTS will most likely be designed with a magnetometer as one of the vectorized measurements. As the satellite will be controlled by magnetourqers, the local magnetic field will be greatly affected by the attitude control. Hence, it is important to seperate the attitude estimation and the attitude control. This is done through a turn-based switching between estimation and control. This implies that a short start-up phase for the estimation is preferred. As the EQUEST solves the attitude problem in one iteration, it is more attractive for satellites using this technique than the iterative EKF.

REFERENCES

[1] Psiaki,M.L., "Attitude-Determination Filtering via Extended Quaternion Estimation", Journal of Guidance, Control and Dynamics, Vol.23, No.2, March-April 2000.

[2] Cheng,Y. and Shuster,M.D, "Robustness and Accuracy of the Quest Algorithm", presented as paper AAS 07-102, 17th AAS/AIAA Space Flight Mechanics Meeting, Sedona, Arizona, January 28 - February 2, 2007.

[3] Markley,L.F. and Mortari,D., "Quaternion Attitude Estimation Using Vector Observations", The Journal of the Astronautical Sciences, Vol.48, April-September 2000.

[4] CubeSat Design Specification Rev. 12, The CubeSat Program, Cal Poly SLO, Retrieved May 2011.

[5] NTNU Test Satellite, Retrieved May 2011, http://nuts.iet.ntnu.no

[6] Gravdahl,J.T., Skavhaug,A., Eide,E., Svartveit,K., Fauske.K.M. and Indergaard,F.M. "Three axis Attitude Determination and Control System for a picosatellite: Design and implementation", presented as paper IAC-03-A.5.07, IAC, Bremen, Germany, 2003.

[7] Svartveit,K., "Attitude determination of the NCUBE satellite", Master Thesis, Department of Engineering Cybernetics, NTNU, June 2003.

[8] Ose, S.S., " Attitude determination for the Norwegian student satellite nCube", Master Thesis, Department of Engineering Cybernetics, NTNU, June 2004.

[9] Sabatini,A.M., "Quaternion-Based Extended Kalman Filter for Determining Orientation by Inertial and Magnetic Sensing", IEEE Transaction on Biomedical Engineering, Vol.53, No. 7, July 2006.

[10] Hughes,P.C., "Spacecraft Attitude Dynamics", Dover Publications, 2004.

[11] Crassidis,J.L., Markley,F.L., and Cheng,Y., "Survey of Nonlinear Attitude Estimation Methods", Journal of Guidance, Control, and Dynamics, Vol.30, No.1, January-February 2007.

[12] Ribeiro,M.I., "Kalman and Extended Kalman Filters: Concept, Derivation and Properties", Institute for Systems and Robotics, Lisboa, Portugal, February 2004.

[13] Nocedal,J. and Wright,S.J. "Numerical Optimization", Springer Series in Operations Research, 1999.

[14] Crassidis, Crassidis,J.L. and Markley,F.L., "Attitude Estimation Using Modified Rodrigues Parameters", Proceedings of the Flight Mechanics/Esitmation Theory Symposium, Cp-1996-3333, NASA Goddard Space Flight Center, Greenbelt, MD, 1996.

[15] Weisstein,E.W., "Least Squares Fitting", From MathWorld-A Wolfram Web Resource, Retrieved May 2011, mathworld.wolfram.com/LeastSquaresFitting.html.

[16] Press,W.H., Teukolsky,S.A., Vetterling,W.T. and Flannery,B.P., "Numerical Recipes in C - The Art of Scientific Computation", Second Edition, Cambridge University Press, pp. 43-49, 1992.

[17] Jacobi,C.G.J., "Über ein leichtes Verfahren, die in der Theorie der Säkularstörungen vorkommenden Gleichungen numerisch aufzulösen", Crelle's Journal 30, pp. 51-94, 1846.