M.Sc. thesis in Extended Kalman filtration for attitude and orbital determination of satellites

Daniel Lindqvist
daniel.per.lindqvist@gmail.com

School of Innovation, Design and Technology
Mälardalens University

Supervisor: Fredik Bruhn
Examiner: Giacomo Spampinato

April 26, 2015

Abstract

One of the most valuable aspects when controlling a satellite is knowing its current location and orientation. When controlling the movement of a satellite, it is vital to have accurate measurements of its position and orientation as well as the rate of change of these variables. In order to approximate the state based on different measurements available an approach based on the extended Kalman filter is suggested. The systems for determining position and the orientation has been developed separately with the modelling of the positioning system being done using the Cartesian coordinates of the Euclidean space, the modelling of the orientation was based on quaternions.

The fraction of the noise from the measurement that affects the filtered value is the main aspect being considered when evaluating the performance of the filter system. This fraction of noise that affects the filtered value is increasing rapidly when the measurement error reduces, while the overall noise still gets reduced with reduced noise on the measurements.

Contents

Chapter 1

Introduction

A artificial satellite is an artificial object which has intentionally been placed in orbit around Earth, these artificial objects are mainly used for communication with various devices on Earth. The first artificial satellite was sent into orbit 1957 with many more to come thereafter. The price of launching satellites into orbit is one of the main reason that the number of satellites has been kept down to the low numbers that they currently are in, combined with precautions to ensure that they are functional for a long time. In order to ensure that the system stays operational for its specified lifetime, safety critical sections of the software must be developed using approaches to ensure that no critical errors could occur capable of causing damage to the satellite. Another major part is to allow the satellite to make its decision with regards to the correct data. If the satellite's attitude determination system came to the wrong conclusion at a point when thrusters were used to change the orbit of the satellite, the results could be catastrophic. This is one aspect that makes the reliability of the positioning and orientation filters a key concern. A reliable positioning and orientation system allows the satellites accurate and precise estimation in order for the satellite's control system to reach the most optimal orientation. The most optimal orientation is dependent on the current position in order to optimize the performance with regards to communication or for the satellite to reach the optimal angle of charging using solar panels.

When data with regards to position and orientation has been retrieved within a certain margin of error, combined with a guarantee that the information is not old appropriate actions could be done. This is where the real time aspect comes in hand, as it is not enough that it has been tested to execute in time but rather that it should be proven that the filter will always respond in time.

1.1 System for evaluating the real position and orientation

The aspects that can and will be useful when evaluating the satellites position and orientation are both the behaviour of the system with regards to how it moves but also the different outer affects causes impact on the satellite. This is combined with the different signals and other indications that can be used to evaluate position and orientation such as communication to ground, the magnetic field and the light emitted or bent around different sources such as the edge of the Earth, the sun and stars.

The different sensors that are used for evaluating the position with regards to these different aspects all have their advantages and disadvantages, either with regards to power consumptions, partial information, margin of error and may be unable to operate under certain situations. The sensors for the aspects described above will be combined with gyroscopes either from an inertial navigation system or using separate gyroscopes, to determine the angular velocity.

1.1.1 Outer effects on the satellite

The satellite in orbit around Earth will as described by being in orbit around Earth mainly be affected by gravity between the satellite and Earth. Gravity is considering the distance between center of Earth and the satellite, affecting all the parts of the satellite in an identical way. As the effect of the gravity can be seen as a function of distance pointing towards the mass center of Earth. For the satellites orbit around Earth the velocity that the satellite is put in to orbit around the Earth is based on the force of gravity and the radius of the orbit around Earth and the field of gravity, which is also based on the distance to the Earth's center of mass.

An additional outer effect would be debris which could collide with the satellite which could have fatal consequences, but if they aren't causing fatal consequences they could still cause rotation and accelerations that should be cancelled out in order for the satellite to be able to function optimally.

The magnetic field is one of the other outer parameters to take under consideration, though this one has mainly affect when it comes to orientation. This is an aspect that can be controlled as electromagnets can be rotated around to ensure that the satellite rotates in a way to reach the desired orientations. These kind of methods only work for near Earth orbits as the magnetic field decays with increasing distance from Earth, another disadvantage is that this disturbs the magnetic field, which could otherwise be used for determining orientation.

1.1.2 Sensors that could be used for position and orientation measurements

The different sensors that are considered due to their utilities on a satellite which has been showed with previous approaches for stabilizing approximation of state of the satellite using them. The sensors will be evaluated with regards to their functionality to show how their measurements can be used to determine the satellites overall state.

Global Positioning System

The Global Positioning System (GPS) is a system set up of two control stations, 12 command and control antennas and approximately 24 GPS satellites. The control antennas are used for calibrating the GPS satellites positioning which are later used as control points for various devices on Earth and close to Earth for determining their positions. The signals used for communication are sent with information of time and position of the satellite that sent the information, this data is then used to determine the distance using the time it takes the signal to reach the receiver. With distance known to atleast three objects with known locations results in an accurate estimation of the position of the receiver using triangulation.

Magnetometer

Magnetometers have been used on several space crafts and satellites over the years, especially ring core fuxgate magnetometers. This approach was used on Apollo 16 in 1972. The flux gate magnetometers used in many space crafts since Apollo 16 give out the vector of the magnetic field. This means that magnetometers can be used as a 2 Degrees of Freedom (DoF) orientation measurement, given an already known magnetic field at a specific location. A location that would have to be evaluated by positioning systems with inputs such as the GPS. The magnetic field is a force that diminishes with distance to earth, limiting its usability for evaluating satellite orientation at higher altitudes.

One of the disadvantages of using the magnetic field for measuring orientation is that noise from electronic systems, electrical motors and especially electromagnets for creating angular acceleration present on satellites, can give faulty values on the magnetometers which has to be taken under consideration.

Sun sensor

Sun sensors are as described by the name, sensors that locates the direction of the sun by the light emitted by it. This restricts the usability of the sensor with regards to the fact that the sensors would not be able to function at all points in time points when the line of sight to the sun is blocked by mainly Earth but also possibly other object. Sensors used for detecting the angle of the sun can be limited with regards to its field of view which for example is generally in the ranges of about 60 degrees with a high accuracy of approximately of less than a degree on the 2 axis measurement used to define the orientation. They are small and lightweight which makes them suitable for smaller satellites just like the larger once.

Horizon sensor

Horizon sensors are either passive IR (Infra Red) temperature sensors, or using the visible light to see the contrast between Earth and open space. The passive IR sensor works by detecting the difference in temperature between the surface of Earth and the temperature of the cold open space behind it. Using either of these two approaches the different horizon sensors can pin point the center of the Earth with high accuracy at a fair update frequency at higher than 1 Hz.

Star tracker

Star trackers are widely used for the attitude control of satellites, they are comparing the locations of the stars with a predefined map of the locations of the stars in order to determine the orientation of the sensor. Using this approach models ranging from larger super accurate devices used in large satellites to smaller models, lacking the extreme high accuracy but still able to pin point the orientation with an accuracy along all axes within angles of roughly 0.01 degrees. The smaller star trackers also perform this task with a lower power consumption and are able to calculate the orientation within a reasonable time of a few seconds.

The inertial measurement unit and gyroscopes

An inertial measurement unit (IMU) uses accelerometers and gyroscopes in order to determine accelerations and angular velocities, which can then be integrated to velocities, position and orientation. The disadvantage of these kind of sensor systems is that they are prone to drift, as the margin of error from the measurement gets integrated creating drift when it comes to orientation, velocity and position. The drift of gyroscopes varies from the military grade with a angular drift of less than 0.005 degrees per hour, navigation grade with less than 0.01 degrees per hour, the a lot smaller mems systems having their best models with a drift of roughly 1 degree per hour, and other more inaccurate models capable of drifting a full rotation of 360 degrees in just a few hours. The acceleration measurements are though as expected not taking gravity into account, this is as the body frame and the measurement sensor are both getting affected by gravity in the same way. A common misconception is that gravitation can be detected on an object being still on the ground, instead, the force that counteracts gravity is what is actually being measured in the accelerometer which is what the IMU systems generally use for their evaluation of the orientation.

Chapter 2

Problem description

The objectives of this thesis is to evaluate different approaches used for filtering sensor data and to decide the best models to use both for orbital determination and the attitude determination. These evaluations are done in order to develop a safety critical real time system to be implemented in ADA using the Ravenscar profile.

After the evaluation of the filtration approaches and the models to be used for the filtration, the development of simulators for creating measurement data and filters that are generic to the degree where the filter is not limited to specific measurements but rather generic enough that different types of measurements could be used to improve the estimated state. For the orientation measurement this includes any vector measurements to either sun earth or magnetic field and complete orientation measurements such as the one gained from star sensors.

The aim of this thesis is summarized as follows:

- Evaluations of currently existing algorithms.
- Evaluations of models for both prediction and representation of the system.
- Implementation of the model and the algorithm to create a filter for the sensor fusion.
- Create the design of the safety critical system.
- Implementation in ADA.

Chapter 3

Related work

The related work for the attitude and orbital determination of satellites was divided between the two main parts of this project, which is the attitude determination and the orbital determination. The attitude determination and the orbital determination use similar methods for filtering the data. Though with different systems both the optimization process of the filter along with the models varies.

3.1 Attitude determination

The state of the art for attitude determination of satellites mainly revolve around the adaptation of the Kalman filter for non-linear systems combined, with the utilization of quaternions for describing the orientation. One additional aspect that most of the approaches have in common is the use of gyroscopes for measuring the angular velocity. The parts where the approaches differs are seen in either the way they chose to optimize the adaptation for Kalman filters for non-linear systems such as the once described in the papers [1][2][3][4], or to adapt the system to new measurement approaches.

3.1.1 Measurement methods

The measurement approaches that were considered for the related work were reviewed for this application includes optical measurement, magnetometers, star trackers, GNSS noise measurements and GNSS phase measurements.

A standard measurement approach such as the optical measurement for the angle to certain heavenly bodies such as earth sun and moon was improved in [5]. The improvement was knowing both the position of the sun and another object such as the moon to give additional orientation information when viewing how the moon is lit up by the sun.

More traditional measurement methods include magnetic sensors as used in [6] and star trackers described in [1].

The usage of Global Navigation Satellite System (GNSS) for the orientation is one of the newer approaches, by using multiple antennas for receiving the signals sent out by the different Satellites in the GNSS network the difference between the input signals on each of the antennas can be used to determine the orientation. The approaches that were considered for evaluating the orientation are based on the difference of the input signals. The approach described in [7] uses the noise of the input signal to determine the direction that the signal is coming from. The other approach described in [8] [9] are based on the phase of the signals on the different antennas, and uses this information to determine the direction towards the GNSS satellite sending the data.

When it comes to orientation data the last aspect that is considered is the angular velocity, the angular velocity which a gyroscope is able to measure with a high accuracy. Though there are approaches that works well for estimating the angular velocities based on accurate orientation measurements such as the one described in [10].

3.1.2 Filtration approaches

The approaches for filtration are mainly based of the Kalman filters and are all adapted for non-linear filters. These approaches include versions of the extended Kalman filter, the unscented Kalman filter and federal filtering.

The most traditional approach of filtering based on prediction models for non-linear systems is the Extended Kalman filter. With smaller changes on the Extended Kalman filter a combined gyroscope and star sensor combination showed promising results in [1].

The Unscented Kalman filter is another approach for adapting the Kalman filter to non-linear systems. The paper [2] describes an improvement of the Unscented Kalman Filter, called the Adaptive Unscented Kalman Filter. This approach is used to reduce the noise from external disturbance compared to the Unscented Kalman Filter. This is done by using adaptive factors that makes the filter converge faster. Another attempt to improve the Unscented Kalman Filter is described in [3] where a smoothing algorithm is used to improve the overall accuracy of the system which reduces the convergence time as well as reducing the error when reaching convergence.

The Federal filtering is the last approach considered in this section, Federal Filter is also an adaptation of the Kalman filter for non-linear system. The improved Federal Filter described in [4] was adapted to the attitude determination and was shown to reach improvement in performance compared to the original Federal Filter.

3.2 Orbital determination

The orbital determination revolves around the two main topics, which are the measurement methods used for determining the position and the filtration methods used to determine the most accurate position based on the measurements.

3.2.1 Measurement methods used

This section will cover the measurement approaches used in work related to the orbital determination. This includes the usage of Global Navigation Satellite System satellites such as the Global Positioning Systems satellites, together with the utilization of ground beacons and magnetic sensors.

The use of GNSS for low orbit satellites is one of the more common approaches for measuring position, seen in filtering approaches such as [11][12]. The reason for it being one of the more common approach is because of the fact that the GNSS can accurately pin point the position of objects close to earth or on earth, which a low orbit satellite is considered to be.

One of the other more common approaches are the utilization of ground beacons. Ground beacons are generally used for the positioning of the GNSS satellites for accurately pin pointing their location such as for the Global Positioning System [13]. Papers such as [14] also describe the way of using ground beacons for measuring the position of satellites.

A less common way of measuring the position of a satellite was described in [15], which uses magnetic sensors for estimating the position.

3.2.2 Filtration approaches

The filter approaches for orbital determination revolves mainly around the adaptation of the Kalman filter for non-linear systems. Two of these Kalman filters that will be considered in this related work is the Extended Kalman Filter and Unscented Kalman Filter.

An Extended Kalman filter approach for the orbital determination is described in [12]. This simple approach is shown to get a significant improvement on the measurement data from the GPS receiver for position and velocity.

The Unscented Kalman filter is the second approach which is also called Sigma Point Kalman Filter. This filter is evaluated in comparison to the Extended Kalman filter for orbital determination systems in [11]. In this paper the Unscented Kalman filter performs equal or better than the Extended Kalman filter in all regards, especially when it comes to reaching convergence.

Chapter 4

Background

There are three major parts when it comes to the evaluation of the current state. Firstly it is the decision of the sensors, where the aim should be to reach good enough performance to a relatively cheap cost. Secondary is modelling of the system with regards to how the satellite moves around, the margin of error for the sensors and the information that the sensors actually yields. Thirdly is the filtration algorithm using the data from the modelling combined with the sensor data in order to get the best possible estimations of the current state with regards to position, velocity and orientation.

This section will be divided into three parts, where the three parts are the modelling of the attitude determination, orbital determination and the filtration algorithms. The reason for this is that the filtration algorithms suggested for the two different systems are the same with the same benefits and deficits. While on the modelling part the attitude and the orbital determination systems are not in any way connected and thereby split into the sections modelling of the attitude determination system and modelling of the orbital system.

4.1 Modelling of attitude determination system

The first part of modelling is determining how to represent the state being the orientation. The determination of the representation of the state will be one of the key aspect of the modelling of the filter. The aim of the representation is to find a compact way of describing the orientation that lives up to certain criteria, such as being close to linear and making the representation easy to use when it comes to both corrections from additional measurements and to predictions of the future.

4.1.1 Representation of orientation

An orientation is a describing the rotation between one set of coordinate axes in another frame of coordinate axis. Taking the coordinate axes of the satellite and describing them in the global scope of coordinates. For the case of orientation the scopes are always in the same size and dimension, a vectors length doesn't change when it is being rotated nor is parts of the object in the three dimensional space entering a dimension which is not a part of the three dimensions. This means that the rotation can be described using a square transformation matrix transforming the object from a three dimensional space to a three dimensional space with the determinants being one. The matrix describing the transition from the local scope to the global scope can be seen as the three vectors that describe how the X axis of the local scope is defined in the global scope and the same approach for the local Y and Z axis in the global scope. In this way the transformation matrix or the so called rotation matrix when it comes to pure rotation is given. When using rotation matrices for the numerical integration of the orientation. The orientation of the object in mind would be multiplied to the left by the rotation matrix describing the delta rotation based on the angular velocities to achieve the new orientation.

$$[Rotation(\mathbf{w} * \delta t)] * [CurrentOrientation] = [NewOrientation] \tag{4.1}$$

The definition of a new state as described in equation (4.1) only works for the rotation matrices. Another approach of describing the orientation in a more generic way would be to describe it as a function of its derivative and previous state such as in equation (4.1).

$$R_t = R_{t-1} + \dot{R_{t-1}} * \Delta t \tag{4.2}$$

Where R_t describes the Current rotation of the object and R_{t-1} describes the previous orientation and the derivative of R_{t-1} multiplied by δt the change in orientation between those two points in time. This formula is a approximate formula and is only accurate for small values on δt.

Euler angles

Any rotation can be described using three Euler angles, the Euler angles are simply the rotation around one axis followed by a rotation around the second axis followed by a rotation around the third axis. This means that a state could always be used to describe the state using the three angles. The transformation from Euler angles to rotation matrices are shown in equations (4.4), (4.5) and (4.6). The equations are simply sine cosine functions describing the x, y and z axis being rotated around either axis. Though there is not an intuitive way to see the change in the angles as the object is rotating, since the order of the axis is a key aspect.

$$[RotX(a)] * [RotY(b)] * [RotZ(c)] \tag{4.3}$$

$$RotX(a) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(a) & -sin(a) \\ 0 & sin(a) & cos(a) \end{bmatrix} \tag{4.4}$$

$$RotY(b) = \begin{bmatrix} cos(b) & 0 & sin(b) \\ 0 & 1 & 0 \\ -sin(b) & 0 & cos(b) \end{bmatrix} \quad (4.5)$$

$$RotZ(c) = \begin{bmatrix} cos(c) & -sin(c) & 0 \\ sin(c) & cos(c) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.6)$$

Rotation matrix

When using the rotation matrix to describe the approach described in equation (4.1) could be used. This approach would then in some way create rotation matrix based on the change in orientation and one way of doing that is using the Euler angles. This simple approach which assumes that at small angles the order of the Euler angles are not that important. This is true for the angles zero of course as all the angles would be the same but start increasing from that point as the angles increase. With angles larger than a few degrees the performance starts to drop and the approach would be completely useless when going above those few degrees. In order to reach a higher accuracy the dependency of the Euler angles have to be taken into consideration, since a rotation around x followed by a rotation around z is not the same as a rotation around z followed by a rotation around x even though it has a small effect at smaller angles. An approach that is described in "Rotation About an Arbitrary Axis in 3 Dimensions" [16] uses transformation matrices to rotate the local coordinate system so that the rotation axis and the z axis of the coordinate system matches up and then does inverse transformation in order to transform that coordinate system back to the coordinate system of the object to then get an accurate rotation. Even though this approach could be used to accurately predict the upcoming steps, it has one feature which limits it which is that it uses the rotation matrix as the state variables. The rotation matrix has plenty of duplicated information seen as any rotation matrix can be described with three Euler angles. For statistical filters the preference would be to find a representation with as low amount of state variables as possible with a derivative that is easy to calculate and is as close to linear as possible.

Unit Quaternion

The unit quaternion [17] is built up using 4 state variables where the quaternion is defined as described in equations (4.7), (4.8) and (4.9). The transformation from a unit quaternion to a rotation matrix is shown in equation (4.10).

$$q = (q0 + q1 * i + q2 * j + q3 * k) \quad (4.7)$$

$$i^2 = j^2 = k^2 = ijk = -1 \quad (4.8)$$

$$i * j = k = -j * i, j * k = i = -k * j, k * i = j = -i * k \quad (4.9)$$

$$R(q) = \begin{bmatrix} 1-2q_2^2-2q_3^2 & 2q_1q_2-2q_3q_0 & 2q_1q_3+2q_2q_0 \\ 2q_1q_2+2q_3q_0 & 1-2q_1^2-2q_3^2 & 2q_2q_3-2q_1q_0 \\ 2q_1q_3-2q_2q_0 & 2q_2q_3+2q_1q_0 & 1-2q_1^2-2q_2^2 \end{bmatrix} \tag{4.10}$$

Equation (4.10) is the transformation to the rotation matrix, which is one of the key aspects in order to be able to transfer it back to the matrix that can be multiplied by any vector to change the vectors orientation. This does though require the quaternion to be in the unit quaternion form, i.e. that the length of the quaternion vector is one, hence unit a quaternion will be used to describe rotation. They key benefits that makes this notation such a good notation for describing the rotations is the fact that it doesn't have any singularities which will be shown when retrieving the derivation of the quaternion. As the quaternions describes rotations the definition of a quaternion multiplication is the same as the definition of a rotation matrix multiplication, which is one rotation followed by another. This means that multiplying two quaternions and then transforming the result into a rotation matrix is the same as transforming the quaternions to rotation matrices and then multiplying the rotation matrices shown in equation (4.11).

$$R(qa * qb) = R(qa) * R(qb) \tag{4.11}$$

Where the quaternion is described using formula (4.7) and follows the rules described in formulas (4.8) and (4.9). The creation of a quaternion used for a rotation around an arbitrary axis can be defined as a quaternion based on a vector e which is a unit vector parallel with the angular velocity vector, as shown in equation (4.12).

$$q = \begin{bmatrix} \mathbf{e} * sin(\Delta B/2) \\ cos(\Delta B/2)) \end{bmatrix} \tag{4.12}$$

Where the e vector defined with the i component in the x direction, j component in y and k component in the z direction. To complement the equation the value delta B is the angle that the rotation is defined by. This means if a rotation is defined by a rotation vector than the delta B component would be dependent on the length of the rotation vector.

4.1.2 Determining the derivative of the rotation

The two main approaches for creating a formulation such as that one is either using an approach based on the Euler angles or by an approach using quaternion. Both approaches were previously described and this section will show how their derivatives are derived. Neither of the resulting models are exact estimates and they both use approximations of different degrees to get the derivative of the orientation.

Euler angles

The time derivative of the Euler angles can be found from derivation of the rotation matrix found by doing rotation around the three axis, either RotX, RotY, RotZ in which ever order is found most suitable. This could also be done by rotating the matrix around any axis followed by any other axis and then followed by the same axis as the first rotation was rotated around again. For the following example the rotations used would be following the rotation order z-x-z. This would yield the overall rotation matrix when multiplying the three rotations matrices together. The multiplication would start off by having the first rotation around the z axis. This would be multiplied to the left by the rotation matrix of the rotation around the x axis. The results of this would in the end then be multiplied to the left by the last rotation around the z axis. The sub rotation matrices would then be used for evaluating the vector which the next rotation would be done around, by multiplying the rotation matrix by the unit vector to figure out where the vector that will be rotated around is currently located. These vectors are shown in equations (4.13), (4.14) and (4.15).

$$RotZ(\theta) * RotX(\phi) * [0,0,1]^T = K \tag{4.13}$$

$$RotZ(\theta) * [0,1,0]^T = L \tag{4.14}$$

$$I_3x3 * [1,0,0]^T = M \tag{4.15}$$

The rotation which is summarized as the angular velocity **w** is summarized in equation (4.16).

$$\begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} = K * \dot{\beta} + L * \dot{\phi} + M * \dot{\theta} = \begin{bmatrix} \dot{\beta} * sin(\theta) * sin(\phi) + \dot{\phi} * cos(\theta) \\ \dot{\beta} * cos(\theta) * sin(\phi) - \dot{\phi} * cos(\theta) \\ \dot{\beta} * cos(\phi) + \dot{\theta} \end{bmatrix} \tag{4.16}$$

This could be rewritten where the derivative vector as it is only linear combinations as a matrix depending on the orientation multiplied by the angular velocity to achieve the derivative of the Euler angles. This approach even though it is quite simple to understand and only using three state variables it has some key issues, which could quits easy be seen in the matrices in the formula above. This happens when the orientation is the identity matrix. This is when all angles are zero and the vector shown in equation (4.16) becomes equation (4.17) as $sin(0) = 0$ and $cos(0) = 1$.

$$\begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ -\dot{\phi} \\ \dot{\beta} + \dot{\theta} \end{bmatrix} \tag{4.17}$$

Here it is quite obvious to see that all angular velocities are not possible as Wx must be equal to -Wy. Which clearly does not have to be the case for a rotating object just because it is in a certain orientation. This issue is a commonly known issue with the Euler angles and it is called Gimbal lock as the object is locked from performing all its possible rotations that it should be able to. Evaluating the change in Euler angles close to any Gimbal lock situation will cause numerical integration issues. The Euler angles are intuitive and easily understandable and uses only three state variables to describe the rotations. Though Gimbal lock is such a huge disadvantage of this method which doesn't make it viable to pursue any further.

Quaternion

The issues with Gimbal lock when using the Euler angles was one of the key aspects that led to the quaternion representation for rotation. The quaternion described in the previous section have several ways to gain the derivative, the one shown is one out of many and was considered the most intuitive by the author. The quaternion derivative seen in [17] will shortly summarized in equations (4.18) to (4.23).

$$q(t + \Delta t) = (\Delta q) * q(t) \tag{4.18}$$

$$\Delta q = \begin{bmatrix} \mathbf{e} * sin(\Delta B/2) \\ cos(\Delta B/2) \end{bmatrix} = \begin{bmatrix} \mathbf{e} * \Delta B/2 \\ 1 \end{bmatrix} = \begin{bmatrix} (1/2 * \mathbf{w} * \Delta t \\ 1 \end{bmatrix} \tag{4.19}$$

$$\mathbf{w} = w_x * \mathbf{i} + w_y * \mathbf{j} + w_z * \mathbf{k} \tag{4.20}$$

$$q(t + \Delta t) = \begin{bmatrix} \frac{1}{2} * \mathbf{w} * \Delta t \\ 1 \end{bmatrix} * q(t) = (1 + 1/2 * \mathbf{w} * \Delta t) * q(t) \tag{4.21}$$

$$q(t + \Delta t) - q(t) = 1/2 * \mathbf{w} * \Delta t * q(t) \tag{4.22}$$

$$q(t) = (q(t + \Delta t) - q(t))/\Delta t = 1/2 * \mathbf{w} * q(t) \tag{4.23}$$

To explain these formulas, the quaternion in equation (4.18) at time point $t + \delta t$ can be written as the product of the quaternion of the rotation at time t multiplied on the left by the quaternion based on the rotation of δt. This is because of the fact that the delta movement is applied after in the global coordinate frame. The quaternion of δt is seen to be small and the square of delta is assumed to be zero. This means that the delta quaternion which can be seen to be a rotation around the angular velocity vector with a step with a rotation of delta B follow the formula shown in the start of equation (4.19). At the following part of equation (4.19) sine of a small angle is assumed to be the sine parameter, and the value of cosine of a small angle is assumed to be one. This is coming from their first derivative at the point zero where sine is 1 and cosine is zero, making this a valid approximation. Following this the vector $e * \delta B$ is also known as the vector $w * \delta t$ where w follow the description shown in equation (4.20). Splitting up delta q as the quaternion can be seen as the sum of its components it can easily be split up using the normal math rules combined with the non commutative multiplication shown in equations (4.9). This together with real numbers acting as they normally would allows for equation (4.21) to be simplified into equation (4.23), where the left side of the equation becomes the definition of the derivation which it is said to be. The benefits of this approach is that any angular velocity will be possible to be entered regardless of the orientation, which wasn't the case with Euler angles. It also lacks the Euler angles deficit with increased derivative scaling near the Gimbal lock making the Euler angles less linear and therefore more likely to generate errors when close to the Gimbal lock. The main disadvantage of this approach is that it is harder to grasp as quaternions don't have an simple visual representation making them less intuitive. The other disadvantage compared to Euler angles is that they require four state variables to represent the orientation compared to the three required by the Euler angles. The last part is that the derivation of the quaternion is like taking line on a circle, even with the shortest time steps it will start to slide of the circle with radius one in its four dimensions. Though this is easily fixed by normalizing the vector after integrating it.

4.1.3 Modelling the angular velocities

Modelling for the angular acceleration is done using Euler's equation of motion, which describes the motion of a solid object affected by different torques. The relationship that Euler put together is between the inertia matrix of the object, the current angular velocity, the torque and the angular acceleration.

$$I\hat{\mathbf{w}} = -\mathbf{w}I * \mathbf{w} + T \tag{4.24}$$

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \tag{4.25}$$

$$\mathbf{w} = \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix}, \dot{\mathbf{w}} = \begin{bmatrix} \dot{w_x} \\ \dot{w_y} \\ \dot{w_z} \end{bmatrix} \tag{4.26}$$

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{4.27}$$

Here the inertia matrix I is describing the inertia of the object in mind. The inertia describes how much force is required to create rotation around an arbitrary axis. The vector w is the angular velocity of the object and w is the derivative of the angular velocity. The last parameter which is T describes the vector of the torque applied to the body frame. What can be seen in formula one is that the rotation that is created by a torque around an arbitrary axis doesn't have to create an angular velocity around that specific arbitrary axis. The only case in which the of the torque and the axis of the angular velocity would always be parallel would be for an object with an inertia matrix defined as a constant times the identity matrix.

4.2 Modelling of orbital determination system

The approach for determining the position of the satellite is done using a Global Navigation Satellite System (GNSS). Among the two GNSS systems up and running, the USA's Global Positioning System is the most commonly used one for satellite position determination. As most approaches use the exact same inputs making the filtering algorithms more key together with the models. Modelling of the sensor reading is expected to be similar for the different approaches while modelling of the motion of the satellite is expected to vary more at higher orbits where gravity from Earth has a smaller effect and the effect of the gravity from other bodies such as Sun and the Moon is greater, among other factors.

4.2.1 Representation of state variables

When it comes to representing the state variables of a moving object, the two main features to take into account will be position and velocity. There are several different options for how the velocity and the position should be described. Firstly it is the choice between a local and a global reference frame, where the global frame would be a frame that does not rotate and does not move around the Earth. Finding such a reference frame in space where the knowledge of what is moving and what is not moving proves to be a challenge. What it all comes down to is if the frame should be rotating, if it should follow the earth most likely through the center of Earth in its movement around the sun or not. If the position is described in an as local reference frame as the position based on the center of Earth, then the other main feature of describing the position appears, and that is how should the state variables be for representing the current position. Here one of the more simple approaches is to describe it as position in an x dimension, in a y dimension and in a z dimension. Relative to a set reference where the x, y and z axis are always pointing in the same direction and is not rotating with the Earth. Another approach is to use polar coordinates, polar coordinates would describe the position using two angles and one radius. Polar coordinates struggles with being nonlinear at its two poles. Though as an orbiting object around a field of gravity would generally move in a planar ellipsoid. The reference frame can therefore be chosen to avoid highly nonlinear changes in the state variables.

4.2.2 Laws of gravity

The main force when it comes to acceleration of a satellite excluding debris colliding with the satellite is gravity, and the main part of the gravity affecting near Earth orbiting satellites is the gravity from Earth. This means that with the equations of gravity a model of the movement for the satellite can be created based on the knowledge of the mass of Earth.

$$F1 = F2 = G * M1 * M2/R^2 \tag{4.28}$$

The universal gravity constant $G = 6.67384 * 10^{-11} Nm^2/kg^2$ and the mass of Earth $= 5.97219 * 10^{24}$ kilograms.

4.3 Approaches for filtration

When it comes to filtering there is one main feature to take under consideration and that is if knowledge from previous states can and should be used for evaluating the current state. The main way of describing this is as either an iterative approach or a non-iterative approach, the choice of approach will impact the types of sensors that should be chosen and the way that the algorithm and filter system are developed. For example using gyroscopes with a non iterative method will only allow it to yield the state variables of the angular velocities, while with an iterative approach it will be capable of yielding a change in orientation since the last measurement to compliment the knowledge achieved from orientation measurements and still yield the current angular velocity. Another advantage that the iterative approaches have are that it is possible to use angular velocities measurements to predict estimates of orientation, which leads to the fact that gyroscopes combined with orientation sensors such as sun sensor and horizon sensors compliments each other well using an iterative approach. The fact that iterative methods using the knowledge from previous states perform well for attitude determination in spacecraft but also on ground shows as [4][6]. With the two main approaches for filtering orientation data uses either versions of the Kalman filter or the particle filters. The only benefit of not taking the history into account when making new estimates would be if the current state was not dependent on the previous state or if no relation between the current state and the previous state could be made. Another issue with the iterative approaches is the risk of the result diverging, especially when it comes to problems that are not in any way close to linear with many local extreme points. This is mainly caused by incorrect estimates due to the linearisation of the accuracy of the current measurements but can also be an issue with the part taking new measurements into account. Correction could then due to linearisation make the evaluation based on the measurement worse while still calculating the accuracy to be better. With the usage of close to linear systems with state variable representations that make the integrations and corrections as close to linear as possible, then the issues of divergence can be overcome.

When it comes to achieving good results when filtering data it is mainly related to the algorithm used and the model with the state variables and the relation between the state variables and both the integration to upcoming steps and the measurements to be made. With the model part and the descriptions of the beneficial properties of the quaternion already been described the main focus on this part will be mainly based on the different algorithms used to filter data and more specifically for filtering position and orientation data.

The most commonly used filter algorithms for position and orientation data will be described including the different types of Kalman filters with the unscented, extended and the central divided-difference Kalman filter. The different Kalman filters will be compared to some other statistical approaches and the idea of using artificial intelligence based approaches.

4.3.1 Kalman filter

The Kalman filters are all iterative approaches that are named after Rudolf E. Kalman, who was the one who developed the Kalman filter based on already existing statistical approaches for filtering and combined them to create the Kalman filter. This filter which is an optimal filter with regards to minimizing square error based on the sensor readings and the variance of the sensor readings. The original Kalman filter that with all those properties have one major flaw which doesn't make it usable for this application and that is that it can't be used for anything except linear systems. This drawback makes it not usable for most of the problems as only a fraction of the systems that would be filtered are linear. Position, velocity or orientation are not a part of the fraction of systems that are linear. Due to the reason that the Kalman filter only works for linear systems, different types of versions of the original Kalman filter have been used to implement the Kalman filter on problems that are close to linear while still achieving close to optimal performance, though doing this means that the filter is no longer optimal and that it can suffer from divergence if the system is highly nonlinear meaning that it have one or several local extreme points that can cause the divergence either when it comes to corrections and integration. As neither the models for the orientation system nor the second order positioning system can be represented in a linear formula the original Kalman filter can't be used but is still mentioned since most of the filters used for filtering orientation is based on it.

4.3.2 Extended Kalman filter

The extended Kalman filter is as it is described by its name an extension to the original Kalman filter. This extension means that it is linearising the systems so that it extends the Kalman filter from just being used for linear systems. The extended Kalman filter would work exactly like the original Kalman filter if it was used on a linear problem as all the linearisation of a linear problem would make the systems identical to how it would already be in the original Kalman filter. The benefits of using the extended Kalman filter is in its simplicity, compared to other approaches for applying the Kalman filter to nonlinear systems. This results in code that is possible to be written more transparently because of the algorithm not being as complex, helping when proving that the algorithm is critically safe. Another benefit with a less complex algorithm for the filtering is the fact that it generally will have a shorter execution time compared to the more complex versions of the Kalman filter for the nonlinear systems. The shorter execution time can prove to be one of the key aspects when proving safety critical execution for the implementation on micro controller units with limited hardware which is often the case. In a less general view about the filter and with more aim towards what it is actually planned to be used for, then multiple orientation based filters or filters for attitude determination which is one of the more commonly used names for it. Then the extended Kalman filter is one of the more used once with several implementations for both attitude determination in the Inertial measurement unit and the inertial navigation unit systems but also used frequently for sensor fusion with regards to attitude determination in satellites which is where the main focus lies in here.

4.3.3 Unscented Kalman filter

One of the main issues with the Extended Kalman filter is how it performs when the second derivite increases. The main reason why the extended Kalman filter struggles in more nonlinear problems was as described in section 3.3.2 that both the predictions and corrections are based on the first order Jacobians. This means that the derivatives are considered to be constant during the change in state variables both for when evaluating the new state but also when evaluating the variance-covariance matrices.

The Unscented Kalman filters [18] solution to this is to instead of using Jacobians which can be hard to calculate analytically, instead using sigma points. What this means is that using a relatively few points to represent the current state and the current variance of the state variables. This is done by the mean of the weighted points is the current state and the variance of these weighted points is the variance covariance of the current state. These sigma points are then used as current points for the nonlinear equation used to describe the prediction of the next state. When this has been done for all the sigma points, then the new point cloud will be used to re evaluate the current state by its mean and the current state variance-covariance by the variance-covariance of these points. This is repeated over and over for each prediction then using the result to evaluated the upcoming state. For the correction part sigma points are also being used, where the sigma points are created based on the current state and the current covariance. These points are then transformed into the predicted measurements and then combined to one predicted measurement based on the weights of the prediction of the sigma points. The predicted point is then used in combination with the measurement in order to evaluate the unscented Kalman filters gain which is what the state variable and variance covariance gains are based on. The benefits of this approach gets described straight away with the reasoning for using this approach, the downside is a slightly bit more complex code compared to the extended Kalman filter combined with the calculations having to be computed for several different points resulting in a risk of requiring extra computational power. The orientation described in the quaternion coordinates as state variables will not have any extreme points making the system close to linear within a short span on the whole surface on its unit hyper-sphere. Though this is still a used method for evaluating orientation for quaternion which can be seen in articles such as [6].

4.3.4 Federal Kalman filter

The Federal Kalman filter is a sensor fusion structure based on the Kalman filter, it uses several sub-filters for the different input parameters with their corresponding equations in order to evaluate the most likely state as well as the variance-covariance of that state. The most likely state is based on the previous state and the specific measurement or measurements that are corresponding to that specific sub-system. The data from each subsystem is in the end taken into a sensor fusion module which tries to fuse all the positioning data together in order to get the most accurate available estimates of the state variables with the lowest possible variance. As the goal with all the filters is to reduce the variance and covariance of the current state while keeping the variance and covariance as accurate as possible. Though in the case of this filter is achieving the results using substantially more complex algorithms than the extended and unscented Kalman filter at a cost that not only would affect the execution times of the filter but also making the system less transparent when trying to prove that it is safety critical. One other deficit with this method is that it has been tested out less and with a more specific and complex implementation to the specific problem. Even so tests and implementations of this approach when it comes to attitude estimations have shown success in articles such as [4].

4.3.5 Particle filter

The particle filter is, just like the Kalman filters, an iterative statistical method that uses the knowledge that the measurement is only dependent on the current state and noise. The particle filter as suggested in its name uses particles to represent the current state and the current variance and covariance. For the particle filters is that it instead of using the same approach as the extended Kalman filter does with linearisation the action used calculate the variance and covariance, uses multiple particles to represent the current state and variance. The prediction of the next step with regards to the action function is applied to each particle. The result of each particle then receives added noise of the action, which creates additional particles based on the noise. After completing the prediction step, the expected measurement of each of the particles are evaluated, these values are then used for comparison with the real measurement to calculate a weight value of each particle. When all particles are given a weight depending on their accuracy a new set of particles are generated based on the weighting of the different particles from the previous state. This algorithm suffers when it comes to performance against computational power. As the more particles that are used the better the results tend to become. The amount of particles required to achieve high performance on single state variables are already quite high and when the amount of state variables increase so does the amount of particles required to keep the performance high.

4.3.6 Artificial intelligence based approaches

When it comes to implementing a real time safety critical system, two parts that are key are for the first part to see good performance, the other is as described the transparency of the system[19]. If a system is built up using machine learning and artificial intelligence, the transparency is close to none as the system would in a sense be created by itself. This results in an algorithm that seems to work good in most of the cases, as the artificial algorithms tries to improve of every set of data. The problem with the approach is that it lacks any hard proofs that the algorithm created by the algorithm actually works as expected. If an algorithm that was created by a neural network actually could be proven to work in all cases within a certain degree of error then the machine learning would have to stop at such a point otherwise a new proof would have to be derived. Generic algorithms suffer when it comes to the safety critical aspect just as the neural network but for slightly different reasons. Generic algorithms with a fitness value that took the safety criticality under consideration could in theory evolve a system that lives up to those criteria. But defining a fitness value that could classify how well a system performs with regards to the critical safety in mind is hard and the achievable performance doesn't make it a suitable approach either. Articles that discusses how an artificially intelligent based approach could be proven to be safety critical includes "A Scenario-Based Method for Safety Certification of Artificial Intelligent Software" [20]. Less complex systems could under certain situations be shown to be safe. Though as described with transparency and artificial intelligence, the few transparent approaches are mainly the less complex AI algorithms, such as fuzzy logic, which does not tend to give promising results compared to the more traditional solutions.

Chapter 5

Method

This section describes the decision process of which methods to use based on the related work and background within this and related fields. The section will be split into the two different sub parts of the assignment which is the position determination and the attitude determination. Even though the filters that are considered for the two systems are the same set of filters, the decision process for which of the filters to choose for each system are different and do not need to coincide.

5.1 Position determination

The methods used to describe the position determinations are split into two major parts and followed with a last part for the methods used to evaluate the performance. The first is the modelling and the second is the filtering algorithm. The last part will be the method that will be used to evaluate the filter.

5.1.1 Modelling

The modelling of the system includes the decision of state variables combined with the equations describing the fundamental properties of the system. When it comes to a satellite orbiting in space the fundamental properties would be the force of gravity causing acceleration the satellite towards the center of the planet that it is orbiting. This would then be combined with the fact that acceleration is the derivative of the velocity and that velocity is the derivative of position. The final part that is key when modelling is to find the connection between the measurement data and the state variables so that corrections can be made on the state based on the correction.

5.1.2 State Variables

The choice of the state variables to describe the position and velocity of a satellite at a certain point of time. The choice to use the center of Earth as the reference point was quite a simple decision as the GPS position measurements are based on the position of Earth, meaning that when the planet orbits around the sun the same position on the planet would still result in the same measurement. This means basing the reference point on anything else would increase the complexity and also the margin of error of the measurements, while not yielding any benefits as the results in the end could be transformed if needed to another point of reference. The second part is if the reference system should rotate around with the rotation of Earth. Meaning that the same point on Earth should always be described with the same coordinates. Here the decision to keep the model as simple as possible by having a non-rotating coordinate frame while transforming measurements and results of position was decided upon. Lastly the decision that the state variables for describing the coordinates would be based on the three dimensional orthogonal space with the position in the x, y and z dimension. This was decided upon over the polar coordinates for simplicity reasons both due to time limitations but also to keep the code as simple and transparent as possible due to the safety criticality of the system.

$$X_t = \begin{bmatrix} PosX \\ PosY \\ PosZ \\ VelX \\ VelY \\ VelZ \end{bmatrix} \tag{5.1}$$

This results in the state variable X being described as the three position variable and the three velocity variables where both the position and velocity is described in a non-rotating coordinate frame with its origin at the center of Earth.

5.1.3 Prediction

The approach for the integration is simple numeric prediction. This means that the position is simply the result of adding the previous position with the velocity multiplied by the delta time for each of the three axis. To integrate the velocity over time the knowledge that the acceleration of the satellite is purely coming from the force of gravity if no thrusters are used nor is the satellite hit by debris at that specific point in time. This means that the acceleration can be calculated straightly from the equation of gravity stating that the force of gravity on an object is described in the following formula.

$$F = G * M1 * M2/R12^2 \tag{5.2}$$

Where F is the magnitude of the force on the satellite, G the universal gravity constant, M1 mass of the satellite and M2 the mass of Earth. And R12 the distance between the center of the objects. Since the acceleration is the force divided by the mass of the satellite this equation can be rewritten to equation (5.3).

$$a = G * M2/(R12)^2 \tag{5.3}$$

Where a is the magnitude of the acceleration directed towards the center of Earth of the satellite. Which in turn can be written as the acceleration components using the knowledge of the position of the satellite into equation (5.4).

$$\mathbf{a} = \frac{G * M2}{(PosX^2 + PosY^2 + PosZ^2)^{3/2}} * \begin{bmatrix} -PosX \\ -PosY \\ -PosZ \end{bmatrix} \tag{5.4}$$

As the square root of the sum squared positions is the radius and then multiplying that with the negated unit vector of the position being the negated vector divided by the norm of the position being the same as the radius. This equation for the acceleration can then simply be numerically integrated into velocity by adding the previous velocity to the acceleration multiplied by the delta time. The end result being the following formula (5.5).

$$\begin{bmatrix} PosX_t \\ PosY_t \\ PosZ_t \\ VelX_t \\ VelY_t \\ VelZ_t \end{bmatrix} = X_t = F(X_{t-1}) = \begin{bmatrix} X_{t-1}(1) + X_{t-1}(4) * dt \\ X_{t-1}(2) + X_{t-1}(5) * dt \\ (X_{t-1}(3) + X_{t-1}(6) * dt \\ X_{t-1}(4) - \frac{G*M*X_{t-1}(1)}{(X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt \\ X_{t-1}(5) - \frac{G*M*X_{t-1}(2)}{(X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt \\ X_{t-1}(6) - \frac{G*M*X_{t-1}(3)}{(X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt) \end{bmatrix} \tag{5.5}$$

5.1.4 Correction

The corrections are based on positional measurements, this is because of how exact the positional measurements are based on the GPS technology. The measurement would then be evaluated to the non-rotating measurement system by rotating it around the rotation axis of Earth based on the current time. At this point the measurement could simply be used for correction of the system based on the difference between the measured position and the filters position among other things and using the relationship between the velocity and the position to improve the evaluation of the velocity.

$$Meassurement = \begin{bmatrix} MeasurementX \\ MeasurementY \\ MeasurementZ \end{bmatrix} = \begin{bmatrix} X(1) \\ X(2) \\ X(3) \end{bmatrix} \tag{5.6}$$

5.1.5 Filter Algorithm Extended Kalman filter

The choice of which filter algorithm was one of the key parts when it came to decide the approach that the filter would be developed in. The Extended Kalman filter got decided upon due to its simplicity while still yielding good results for systems that are close to linear. That the system is close to linear can clearly be seen as there are linear relations between the position and the velocity data, this combined with the rate of change of the acceleration which is close to zero over a period of a a few minutes for an orbiting satellite. These properties resulted in the fact that none of the other more complex filters should have benefits that would yield any major improvements if any at all at the cost of a more complex algorithm and longer execution times. Since the extended Kalman filter uses the Jacobians for analysing the change in the error term during integration and the correction state. Making that an analytic evaluation of these functions a key aspect in keeping the extended Kalman filter a computationally easy approach.

Derivatives of the prediction

Firstly in order to get the derivatives of the prediction part of the extended Kalman filter the integration formula should be taken into account. Which is how to describe the upcoming step using the action of what is happening and the previous state. Which is done using the equation (5.7) also described in the related work section.

$$\begin{bmatrix} PosX_t \\ PosY_t \\ PosZ_t \\ VelX_t \\ VelY_t \\ VelZ_t \end{bmatrix} = X_t = F(X_{t-1}) = \begin{bmatrix} X_{t-1}(1) + X_{t-1}(4) * dt \\ X_{t-1}(2) + X_{t-1}(5) * dt \\ (X_{t-1}(3) + X_{t-1}(6) * dt \\ X_{t-1}(4) - \frac{G*M*X_{t-1}(1)}{(X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt \\ X_{t-1}(5) - \frac{G*M*X_{t-1}(2)}{(X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt \\ X_{t-1}(6) - \frac{G*M*X_{t-1}(3)}{(X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt) \end{bmatrix} \tag{5.7}$$

The Jacobian of this equation would then be evaluated by taking each of the six equation rows and evaluating the derivative based on each of the six state variables. The Jacobian of the position prediction shown in equation (5.8) is then a 6x6 matrix of first order derivatives.

$$\begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ B & A*X(1) & A*X(1) & 1 & 0 & 0 \\ A*X(2) & C & A*X(2) & 0 & 1 & 0 \\ A*X(3) & A*X(3) & D & 0 & 0 & 1 \end{bmatrix} \tag{5.8}$$

$$A = -3/2 * (X_1^2 + X_2^2 + X_3^2)^{-5/2}] * dt * M * G \tag{5.9}$$

$$B = [(X_1^2 + X_2^2 + X_3^2)^{-3/2} - 3 * X_1^2 * (X_1^2 + X_2^2 + X_3^2)^{-5/2}] * dt * M * G \tag{5.10}$$

$$C = [(X_1^2 + X_2^2 + X_3^2)^{-3/2} - 3 * X_2^2 * (X_1^2 + X_2^2 + X_3^2)^{-5/2}] * dt * M * G \tag{5.11}$$

$$D = [(X_1^2 + X_2^2 + X_3^2)^{-3/2} - 3 * X_3^2 * (X_1^2 + X_2^2 + X_3^2)^{-5/2}] * dt * M * G \quad (5.12)$$

Where A, B, C and D are variables that are only used in order to save space inside the matrix and to improve readability. The other thing to point out is that the results from A, B, C and D have such a small magnitude that they won't have any major impact on the actual performance of the system but when the evaluation to confirm that was done the evaluations had already been made so there were no benefits of removing it.

Derivatives of the corrections

The equation used to describe the measurement is significantly less complex than the one used for describing the prediction. The equation of the measurement which can be shown to be linearly dependent on the state variables shown in equation (5.13).

$$Meassurement = \begin{bmatrix} MeasurementX \\ MeasurementY \\ MeasurementZ \end{bmatrix} = \begin{bmatrix} X(1) \\ X(2) \\ X(3) \end{bmatrix} \quad (5.13)$$

This yields a Jacobian (5.14) with a dimension of 3x6 only containing ones and zeroes.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (5.14)$$

After the analytical evaluating of these Jacobians the rest of the adapting of the extended Kalman filter to the system is implementation parts and analysis of results.

5.2 Attitude determination

This section describes the methods used for creating the attitude determination system. The major parts that will be gone through is the modelling with the key aspects including the choice of state variables. The other key part is the filter algorithm with motivation for why the specific algorithm was used combined with the key analytic parts that are not in itself a part of the implementation. When it comes to modelling the two key aspects is how to represent the current state and how the equations of prediction and correction will be based on those specific state variables used for representing the states.

5.2.1 State Variables

The state variables should explicitly tell the current state, meaning that with a certain set of state variables there is just one possible solution for the current state of the system. The secondary part for the choice of state variables is to make the modelling equations as simple and close to linear as possible in order to reduce the risk of local extreme points that create risks of inaccuracy in filters. The choice to use unit quaternion was quite the obvious one as it is the currently most used representation for orientation, especially for filter algorithms. The high usage of unit quaternions does not come without reasons such as the fact that it keeps a simple representation with few state variables while keeping the system of rotations as simple and close to linear as possible. Secondly, the choice to keep the representation of the state variables on a non-rotating coordinate frame was decided upon due to the fact that the limitations of doing so would deny the possibility to use the system for other purposes while still not yield any favourable aspects as the measurement towards the sun and the surface of the Earth would not benefit from a more local coordinate frame while the models would be considerably more complex and would require positional measurements for testing thereby adding to the complexity. The decision not to use angular velocity as state variables comes from the fact that the angular velocity measurements in cheap gyroscopes are very accurate. So to run prediction and corrections for that part would not be considered worth the computational time for the possibility of a slight improvement of the velocity of the orientation system.

$$X_t = \begin{bmatrix} QuaternionComponent0 \\ QuaternionComponent1 \\ QuaternionComponent2 \\ QuaternionComponent3 \end{bmatrix} \tag{5.15}$$

Where Quaternion Component 0 is the scalar part and the quaternion components 1, 2 and 3 are the vector part of the axis of rotation. This leaves the current state as shown in the equation above.

5.2.2 Prediction

The prediction of the next state for a quaternion is exactly the same as the time derivative of the quaternion. This was one of the main reasons as described in Chapter 3, that the quaternion was such a good option to use as state variable.

$$\dot{q}(t) = 1/2 * \mathbf{w} * q(t) \tag{5.16}$$

The first derivative of a quaternion is described where q is the quaternion and w is a quaternion with the scalar part of 0 followed by a vector part of omega. This means that the upcoming state quaternion could be written as in equation (5.17).

$$q(t+1) = q(t) + \dot{q}(t) * dt \tag{5.17}$$

Which in turn for the system with its state variables would correlate with the two equation (5.16) and (5.17) into equation (5.18).

$$X_{t+1} = X_t + 1/2 * \mathbf{w} * X_t * dt \tag{5.18}$$

Equation (5.18) could then be transformed from the quaternion operations into the matrix operations in equation (5.19).

$$\begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix}_{t+1} = \begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix}_{t} + \frac{1}{2} * \begin{bmatrix} 0 & -wx & -wy & -wz \\ wx & 0 & wz & -wy \\ wy & -wz & 0 & wx \\ wz & wy & -wz & 0 \end{bmatrix} * \begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix}_{t} * dt \quad (5.19)$$

5.2.3 Correction

The correction part revolves around finding a way using the current state in order to calculate how the measurement should be if the current state was completely correct. The ways that the system should be able to correct from should be either from a quaternion giving the current orientation, or from a vector towards a certain object or aligned with a certain force field. For the quaternion part the correction is quite simple. Since the state is described as a quaternion this simply becomes the quaternion itself.

$$QMeasurement = X \quad (5.20)$$

Where X is the current state and Qmeasurement is the measurement quaternion. In order to correct the current state with a vector the equation is slightly more complex. This requires the quaternion to be converted to the inverse rotation matrix, as the measurement will be local the global vector has to be multiplied by the inverse rotation matrix. The inversion is simply done by negating the first term of the quaternion before using the function to create a rotation matrix based on the state quaternion. The creation of a rotation matrix from quaternion was described in more detail in the related work section for quaternions.

$$R(q) = \begin{bmatrix} 1 - 2 * q_2^2 - 2 * q_3^2 & 2 * q_1 * q_2 - 2 * q_3 * q_0 & 2 * q_1 * q_3 + 2 * q_2 * q_0 \\ 2 * q_1 * q_2 + 2 * q_3 * q_0 & 1 - 2 * q_1^2 - 2 * q_3^2 & 2 * q_2 * q_3 - 2 * q_1 * q_0 \\ 2 * q_1 * q_3 - 2 * q_2 * q_0 & 2 * q_2 * q_3 + 2 * q_1 * q_0 & 1 - 2 * q_1^2 - 2 * q_2^2 \end{bmatrix} \quad (5.21)$$

Where R(q) is the rotation matrix based on the unit quaternion q.

$$inv(R(q)) = \begin{bmatrix} 1 - 2 * q_2^2 - 2 * q_3^2 & 2 * q_1 * q_2 + 2 * q_3 * q_0 & 2 * q_1 * q_3 - 2 * q_2 * q_0 \\ 2 * q_1 * q_2 - 2 * q_3 * q_0 & 1 - 2 * q_1^2 - 2 * q_3^2 & 2 * q_2 * q_3 + 2 * q_1 * q_0 \\ 2 * q_1 * q_3 + 2 * q_2 * q_0 & 2 * q_2 * q_3 - 2 * q_1 * q_0 & 1 - 2 * q_1^2 - 2 * q_2^2 \end{bmatrix} \quad (5.22)$$

The inverse of the rotation matrix can be seen by negating the q0 term. The inverse rotation quaternion should then be multiplied to the right with the global knowledge of the direction to the object represented by a unit vector. This result in the following formula to compare the measurements.

$$h(Q) = \begin{bmatrix} 1-2*q_2^2-2*q_3^2 & 2*q_1*q_2+2*q_3*q_0 & 2*q_1*q_3-2*q_2*q_0 \\ 2*q_1*q_2-2*q_3*q_0 & 1-2*q_1^2-2*q_3^2 & 2*q_2*q_3+2*q_1*q_0 \\ 2*q_1*q_3+2*q_2*q_0 & 2*q_2*q_3-2*q_1*q_0 & 1-2*q_1^2-2*q_2^2)] \end{bmatrix} * \begin{bmatrix} K \\ L \\ M \end{bmatrix} \tag{5.23}$$

Where h(Q) is the formula used to describe the vector measurement as a function of the current state and the unit vector [K, L, M].

5.2.4 Filter Algorithm Extended Kalman filter

This section will describe the reasons of why the Algorithm of the Extended Kalman filter was chosen for this specific problem as well as describe the methods used for adapting the algorithm to the specific system that it is used for, the satellite orientation using quaternions. The reason for going with the extended Kalman filter is that the system is still quite close to linear allowing the extended Kalman filter to perform well. This is under a few circumstances though, which in the main part is that the rotation angle between prediction needs to be kept reasonably low in order to keep the system as close to linear as possible. This means that the satellite either has to rotate slowly or the update frequency needs to be high. As the satellite is not expected to rotate at a high velocity, and that the update frequency of the prediction can be done at a high enough rate without suffering from lack of execution time, the angles between each update even up at a few hundred revolutions per minute (RPM) can be considered low enough that the linearity wouldn't suffer to any major degree. This combined with the fact that the extended Kalman filter is one of the simpler and therefore more transparent making it easier to prove critically safe compared to some of the other methods.

Derivatives of prediction

The derivation of the prediction step for the extended Kalman filter is simply done by derivation of every part of the prediction formula which is described in equation (5.24).

$$\begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix}_{t+1} = \begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix}_{t} + \frac{1}{2} * \begin{bmatrix} 0 & -wx & -wy & -wz \\ wx & 0 & wz & -wy \\ wy & -wz & 0 & wx \\ wz & wy & -wz & 0 \end{bmatrix} * \begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix}_{t} * dt \tag{5.24}$$

The Jacobians which is the first order derivative of the function will be a 4x4 matrix as there are four equations and there are four state variables.

The prediction step will always be a square matrix as the evaluation of each state variable is being derivative with regards to each state variable. Doing the derivation like that results in the Jacobian shown in equation (5.25).

$$Prediction\_Jacobian = \begin{bmatrix} 2*\Delta t & -wx & -wy & -wz \\ wx & 2*\Delta t & wz & -wy \\ wy & -wz & 2*\Delta t & wx \\ wz & wy & -wz & 2*\Delta t \end{bmatrix} * \frac{\Delta t}{2} \tag{5.25}$$

The representation with the multiplication with $\delta t/2$ on the outside making the diagonally one is to prevent the matrix to become to large when representing it.

Derivatives of correction

The derivatives used for evaluating the variance-covariance of the correction of the state variables will be based on the formulas for the corrections. The two possible corrections are the one from a quaternion and from a vector. The analytic derivation of the two Jacobian are not of any higher degree of complexity. The Jacobian for the correction with regards from quaternion can easily be extracted from the formula for the measurement.

$$QMeasurement = X \tag{5.26}$$

Where it is clear that the Jacobian is an identity matrix of size 4x4 as both the quaternions have four components. The correction Jacobian for the correction with regards to a vector isn't quite as simple. The correction Jacobian can be evaluated from the following formula (5.27).

$$h(Q) = \begin{bmatrix} 1-2*q_2^2-2*q_3^2 & 2*q_1*q_2+2*q_3*q_0 & 2*q_1*q_3-2*q_2*q_0 \\ 2*q_1*q_2-2*q_3*q_0 & 1-2*q_1^2-2*q_3^2 & 2*q_2*q_3+2*q_1*q_0 \\ 2*q_1*q_3+2*q_2*q_0 & 2*q_2*q_3-2*q_1*q_0 & 1-2*q_1^2-2*q_2^2)] \end{bmatrix} * \begin{bmatrix} K \\ L \\ M \end{bmatrix} \tag{5.27}$$

The Jacobian (5.28) of the correction formula h can then be described with the following derivatives from the definition of the Jacobian, where h_1 is the first row of the equation, h_2 the second row and h_3 the third row. These derivatives are then calculated separately in equations (5.29).

$$\begin{bmatrix} \frac{dh_1}{dq_0} & \frac{dh_1}{dq_1} & \frac{dh_2}{dq_2} & \frac{dh_3}{dq_3} \\ \frac{dh_2}{dq_0} & \frac{dh_1}{dq_1} & \frac{dh_2}{dq_2} & \frac{dh_3}{dq_3} \\ \frac{dh_3}{dq_0} & \frac{dh_1}{dq_1} & \frac{dh_2}{dq_2} & \frac{dh_3}{dq_3} \end{bmatrix} \tag{5.28}$$

$$
\begin{aligned}
\frac{dh_1}{dq_0} &= 0 + 2q3 * l - 2q2 * m \\
\frac{dh_1}{dq_1} &= 0 + 2 * q2 * l + 2 * q3 * m \\
\frac{dh_2}{dq_2} &= -4q2 * k + 2q1 * l - 2q0 * m \\
\frac{dh_3}{dq_3} &= -4q3 * k + 2 * q0 * l + 2 * q1 * m \\
\frac{dh_2}{dq_0} &= -2 * q3 * k + 0 + 2 * q1 * m \\
\frac{dh_1}{dq_1} &= 2 * q2 * k - 4 * q1 * l + 2 * q0 * m \\
\frac{dh_2}{dq_2} &= 2 * q1 * k + 0 + 2 * q3 * m \\
\frac{dh_3}{dq_3} &= -2 * q0 * k - 4 * q3 * l + 2 * q2 * m \\
\frac{dh_3}{dq_0} &= 2 * q2 * k - 2 * q1 * l + 0 \\
\frac{dh_1}{dq_1} &= 2 * q3 * k - 2 * q0 * l - 4 * q1 * m \\
\frac{dh_2}{dq_2} &= 2 * q0 * k + 2 * q3 * l - 4 * q2 * m \\
\frac{dh_3}{dq_3} &= 2 * q1 * k + 2 * q2 * l + 0
\end{aligned}
\tag{5.29}
$$

This analytic evaluation of the Jacobian sums up the methods used for adapting the extended Kalman filter to the problem of the attitude determination system.

5.2.5 Evaluation of error in quaternion space

The result of a the extended Kalman filter is the current state quaternion combined with the matrix describing the variance and covariance of the state variables. For most the comparison between two quaternions or the variance-covariance in the quaternion coordinates doesn't tell much. For further analysis of how the system actually would perform, two main parts of evaluations was to be considered, the first being the real error. With getting an angle of rotation between the two quaternions. The second is to analyse the variance-covariance matrix in order to achieve an angle to represent the standard deviation.

Error angle from quaternion

The aim is to achieve an error angle based on the real and the estimated quaternion. This can simply be done by inverting the quaternion used to represent the real orientation. Then multiplying the inverse of the real quaternion with the estimated quaternion. The result of this comes to rotate an object in the way the estimated quaternion determines, then rotating it opposite to how the real quaternion says that it should be rotated. This means that the resulting quaternion will describe the rotation in between the two quaternions. Then as the first element in a quaternion determines how much the quaternion is being rotated, the angle of rotation can easily be calculated from the value. The formula (5.30) that the calculation comes from is that the first variable is cosines of half the rotation angle.

$$Angle_{error} = 2 * arccos([Q_{est} * inv(Q_{real}]_0) \tag{5.30}$$

Where the 0 after the quaternion bracket refers to the value and index 0 being the scalar part of the quaternion.

Analysing quaternion Variance-Covariance

The variance-covariance matrix that the extended Kalman filter has to determine the normal distribution of the evaluated state. One approach for evaluating this is to get the Jacobian of the transfer function between the quaternion coordinates for example the roll pitch yaw coordinates. Then use the Jacobians to transfer the quaternion coordinates to the roll pitch yaw coordinates. This approach only works well if the transformation is close to linear which doubtfully would be the case and that several percent in evaluation would go wrong. Another approach that will be used is to create a point cloud of a uniform hyper sphere. Then use all these points to multiply with the covariance. When this is done adding the change to the current orientation. The problem here is that this might not be a unit quaternion. To correct for this the first term should stay the same as it describes the rotation while the vector part will be scaled to make the quaternion yet again a unit quaternion. After this the quaternion gets rotated back by multiplication of the inverse state quaternion and in the end use the transfer function between the quaternion and the roll pitch yaw coordinates. This transformation is done by converting the quaternion to rotation matrix and then the rotation matrix to Euler angles.

$$R(q) = \begin{bmatrix} 1 - 2 * q_2^2 - 2 * q_3^2 & 2 * q_1 * q_2 - 2 * q_3 * q_0 & 2 * q_1 * q_3 + 2 * q_2 * q_0 \\ 2 * q_1 * q_2 + 2 * q_3 * q_0 & 1 - 2 * q_1^2 - 2 * q_3^2 & 2 * q_2 * q_3 - 2 * q_1 * q_0 \\ 2 * q_1 * q_3 - 2 * q_2 * q_0 & 2 * q_2 * q_3 + 2 * q_1 * q_0 & 1 - 2 * q_1^2 - 2 * q_2^2 \end{bmatrix} \tag{5.31}$$

The formula to convert a rotation matrix to roll pitch yaw angles can be described using the equation (5.32).

$$RPY(R) = \begin{bmatrix} arctan(R(3,2), R(3,3)) \\ arctan(-R(3,1), sqrt(R(3,2)^2 + R(3,3)^2)) \\ arctan(R(2,1), R(1,1)) \end{bmatrix} \tag{5.32}$$

The set of data that is generated by this will be a point cloud representing the standard deviation of the variance of the state. Which means that the variance in the roll pitch yaw coordinate frame can easily be evaluated. The smaller the areas between the points on the hyper sphere is the better the generated data will be, and to keep it evenly distributed is key to allow the variance to be recalculated based on the points. The major disadvantage of adding more points is of course the added computational time needed while giving improved accuracy of the results.

The evaluation requires a uniform distribution on a hyper sphere or at least a hyper sphere that is close to uniform. Generating points uniformly on a hyper sphere is a well-known issue. Though good estimations that are close to exact exist [21]. That approach tries to spread the points as uniform as possible, was done by evenly splitting the hyper sphere into spheres, splitting the spheres to circles and evenly splitting the circles into points. Where all the splitting is done using parallel hyper planes, where the distance between the planes are based on the amount of points that should be generated uniformly on the surface combined with the size of that specific sphere or circle.

5.3 Simulation

To simulate the reality for which the filter should be tested on should live up to the following requirements: The simulator should integrate at a significantly higher frequency than the filter to ensure that errors due to numeric integration show up as they would in reality. The measurements should be created using Gaussian noise as it is the most common way that real measurements have their measured values. The noise should be represented using a variance matrix. The simulator should be able to analyse the data coming from the filter and show the error of the estimates. For the interface towards the tester, then the simulator should be able to show a visible representation of how the object is moving or rotating, combined with graphs that clearly show the error with regards to the angle of error for orientation and the magnitude of the error in position as well as the magnitude of the error in velocity. To complement this the real time feed of data should be able to be turned off in order for the simulator to be able to run longer duration tests without being slowed down by the interface. The simulator should live up to these requirements while being simply implemented both for the argument of it being transparent but also to ensure that long simulations can be run efficiently.

5.3.1 Position

This means that the positioning system will merely use the equation of gravity and numeric integrations to simulate the movement of the satellite around the sun, taking measurements at the defined time frames and save the data down in order to allow for testing of the filtration.

5.3.2 Orientation

The orientation system on the other hand will use Euler's equations of motions to calculate the torque coming from the angular velocity as well as an addition of random torques. This is then used in combination with the moment of inertia to get the angular velocity that can further be integrated into rotation.

Chapter 6

Implementation Matlab

The Matlab implementation was the key testing phase in order to both give the author a better understanding of the filter algorithm as well as allowing for the key testing to ensure that the filter worked as intended. This section includes a simple implementation of the Kalman filter that the extended Kalman filter is based off as well as the implementation of the quaternion based rotation filter system and the positioning system.

6.1 1-D problem for testing of the Kalman filter

To start of the implementation part it was suggested by the thesis examiner to start off with something simple to allow for easy testing in order to deal with possible misunderstandings with a problem that has a low complexity level. The system used for this testing was to try and filter an angle of a line. The two measurements available would be a measurement of the angular velocity of the line and a measurement of the actual angle of the line. These measurement would both be yielded a certain amount of noise and the filter should process the measurements to follow the true angle of the line to as high of a degree as possible.

6.1.1 Simulation

For the simulations this line would be described using two variables, the angular velocity and the current angle. This line would then be affected by a torque which based on the inertia of the line changes the angular velocity which in turn generates rotations of the line. To ensure that this test case for the filter was not good just for any specific movement pattern random variable was added into the formula of the torque to create a less predictable environment. The errors added to the measurements of the angular velocity and the current angles was Gaussian since most errors on sensor readings are close to Gaussian. The error used in this specific test set up was close to Gaussian with normal distribution of roughly

```
%predict
X_Predict = A*X_Correct+B*U;
P_Predict=A*P_Correct*transpose(A)+B*V*transpose(B)

K=P_Predict*transpose(C)*pinv(C*P_Predict*transpose(C)+W);

X_Correct = X_Predict + K *(Y-C*X_Predict);
P_Correct = P_Predict-K*C*P_Predict
```

Figure 6.1: The algorithm used for the filtering of the 1-d filtering

5-10 degrees. The measurement frequency was in this example the same as the refresh rate of the system which was at 100 Hz. Choosing the same frequency on this might be considered an unfair advantage for the filtering algorithm, but was still used for simplicity and as the implementation was mainly done to get a better understanding of the filter.

6.1.2 Filtration

The filter that was meant to solve the created problem is the Kalman filter with a one state variable representation. The Kalman filter used was using the simplified formula based on the fact that the gain is defined according to the formula describing the optimal Kalman gain.

In the algorithm represented in figure 6.1, the matrices A and B are in this case 1x1 matrices, they describe the correlation between the new state and the previous state. Where in this case the new state will be the previous state plus the angular velocity times the time frame, making the $A = 1$ and $B = \delta t$. It starts with making a prediction of how the state is, the angle of the line, dependent on the previous state and an input which in this case is the angular velocity. This makes X_Correct the angle of the previously corrected value and U the input angular velocity. The P_Predict is the predicted value of how accurate the value achieved by the prediction of the state was. This is based on the previously corrected accuracy P_Correct the matrices that describe how big of a part this variable has in the new prediction. This is combined with the error of the measured angular velocity. The variable K is the optimal Kalman gain described by the formula after, where C is the matrix that transforms measurement to the state, in the problem formulation above C=1. The variable W should be approximately the normal distribution of the error of the angular measurement approximated to 0.1 from a graph of the errors. After this the correction of the X value is achieved by finding the difference between the measured angle and the predicted angle and transitioned by the matrix C to get Y and X to the same state. This difference is then multiplied to the left by the square matrix k describing the optimal Kalman gain and the result is added to the prediction of the state. The last correction is done on the evaluation of how good the value is in the term P_Correct, where the predicted value of P gets

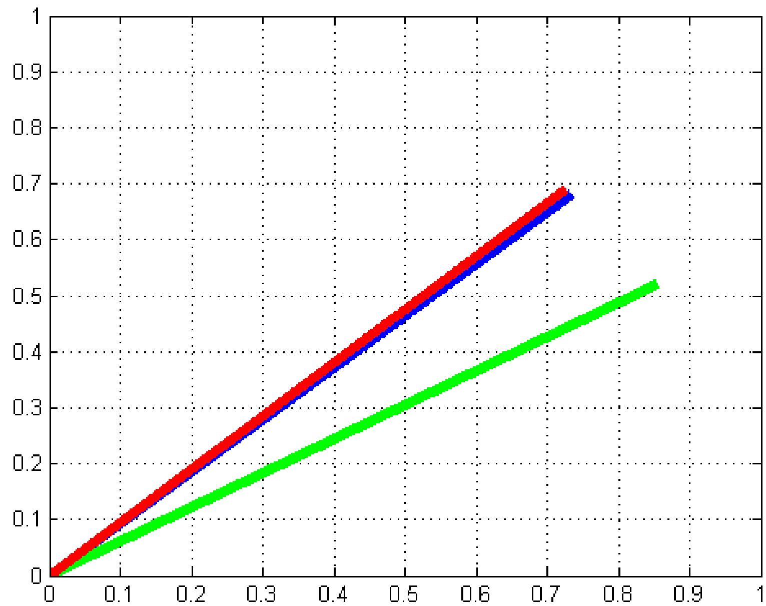

Figure 6.2: Representation of the angle with measurement green, real value red and estimated value blue

subtracted by the Kalman gain K times the transformation matrix C which is then multiplied by P_Predict.

6.1.3 Results of filtering

The results can be seen in figure 6.2, where the real value shown with the blue line and the filtered value with the red line was kept within roughly two degrees. The results were not measured in a scientific way nor were the length of the tests long enough or the sample pool big enough to guarantee any performance from the filter. The main result achieved here was to give the author a better understanding of how the Kalman filter works. In order to get a better understanding of the simple case before starting to implementation of the extended Kalman filter with more state variables and more complex models.

6.2 Attitude system using Extended Kalman filter

The implementation of the extended Kalman filter for the filtering of quaternions to describe rotations, was written taking the knowledge of the simpler problems

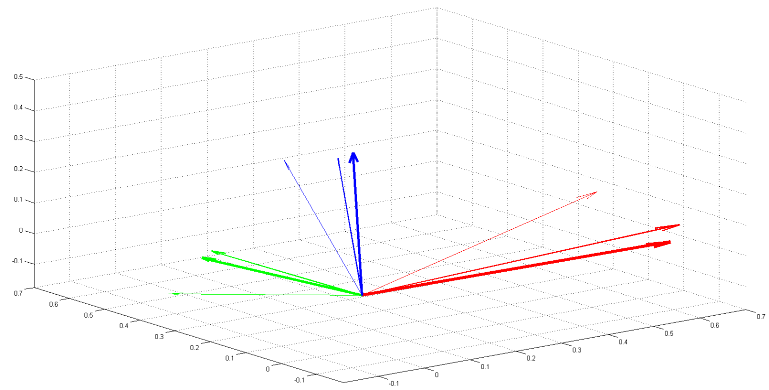

Figure 6.3: Representation of orientation for quaternion filtration

Kalman filtering.

6.2.1 Quaternion functions

The implementation of elementary quaternion functions and analysis tools for quaternions is implemented even though existing functions already exist in Matlab. The reason for this is to make sure that these key parts are already tested properly before the implementation in Ada. This to have the capability of living up to the requirements that specific hardware can have that denies the usage of generically written algorithms for general computers. Another aspect that makes this important is that it will allow the implementation in ADA to live up to the requirements set by the Ravenscar profile which is a requirement product before use. The quaternion functions include the algorithms used for transferring a quaternion to a rotation matrix, multiplying quaternions and getting the angle between two quaternions. These were implemented and used instead of already existing Matlab functions for quaternions for testing purposes.

6.2.2 Simulation and representation

The simulation shown in figure 6.3 is done using numerical integration. In the simulation the state is described by a quaternion for the orientation and a vector is used to describe the angular velocity. The torque affecting the system is created using a random number generator combined with friction for slowing the rotation down. For each state of integration the quaternion is used to generate a rotation matrix. From this rotation matrix the vectors for showing the current rotation is taken combined with using the rotation matrix to achieve any vector based measurements to compliment the measurements taken based on the quaternion.

```
%Prediction

X_Predict = X_Correct+1/2*OmegaMatrix*X_Correct*dt; %=f(x,u);
X_Predict = X_Predict/norm(X_Predict);
P_Predict=F_Jacobian*P_Correct*transpose(F_Jacobian)+Qk;

%Correction Vector
K = P_Predict * transpose(H_Jacobian) * pinv (H_Jacobian * P_Predict *
transpose(H_Jacobian)+Rk);
X_Correct = X_Predict + K*(Y-h(X_Predict));
X_Correct = X_Correct/norm(X_Correct);
P_Correct = P_Predict-K*H_Jacobian*P_Predict;


%Correction Quaternion
K = P_Predict * pinv(P_Predict + Rk);
X_Correct = X_Predict + K*(Y-X_Predict);
X_Correct = X_Correct/norm(X_Correct);
P_Correct = P_Predict-K*P_Predict;
```

Figure 6.4: The algorithm used for the attitude filtration

To visually represent an object in three dimensional space, three axes are used to represent the x, the y and the z axis of the object. This gives a simple yet informative representation that doesn't have to be reprocessed to give the viewer a rough estimate of the performance of the filter. Similar to the one dimensional problem described in the section 6.1 the data represented in the figures are the real values, the measured values and the filtered values. In this case the real values are represented by the three thickest lines of each color, the measurements with the thinnest lines and the filtered results are represented with the last set of three lines. The visual representation was as seen in figure 6.3 done using three vectors, which are obtained as the columns of a rotation matrix. Though this is a conversion done before representing the data because using rotation matrices for for calculations is not an efficient approach neither for the simulation nor for the filtering. The reasons for this is described in Section 5.2.1 where the Hamilton quaternions were found the most efficient way.

6.2.3 Filtration

The state variables for the filtering is the four values in the quaternion used to represent the orientation. The idea behind the extended Kalman filter is quite quickly described with the code represented in figure 6.4.

Where in figure 6.4 the variables are as following: X_Predict is the predicted value of the state quaternion, P_Predict is the accuracy of the predicted value, K is the Kalman gain, X_Correct is the corrected values of the state quaternion, P_Correct is the accuracy of the corrected values, Y is the measurement of the unit quaternion for the complete rotation.

The OmegaMatrix is the matrix that is a representation of the matrix that would be multiplied by a quaternion Y to the right, which would be equal to multiplying the quaternion [0;Wx;Wy;Wz] to the left with the same quaternion Y.

$$OmegaMatrix = \begin{bmatrix} 0 & -wx & -wy & -wz \\ wx & 0 & wz & -wy \\ wy & -wz & 0 & wx \\ wz & wy & -wz & 0 \end{bmatrix} \tag{6.1}$$

F_Jacobian is the Jacobian, meaning a square matrix of derivatives, of the function used for the prediction of the new state variables derivated from the current state variables. The matrix used to define df/dq where f is the function for the prediction is simply calculated given the omega matrix and the state variables.

$$F\_Jacobian = \begin{bmatrix} 2*\Delta t & -wx & -wy & -wz \\ wx & 2*\Delta t & wz & -wy \\ wy & -wz & 2*\Delta t & wx \\ wz & wy & -wz & 2*\Delta t \end{bmatrix} * \frac{\Delta t}{2} \tag{6.2}$$

For the vector correction the function h is the function used to describe the observed vectorial measurement based on the current state quaternion.

$$h(Q) = \begin{bmatrix} 1-2*q_2^2-2*q_3^2 & 2*q_1*q_2+2*q_3*q_0 & 2*q_1*q_3-2*q_2*q_0 \\ 2*q_1*q_2-2*q_3*q_0 & 1-2*q_1^2-2*q_3^2 & 2*q_2*q_3+2*q_1*q_0 \\ 2*q_1*q_3+2*q_2*q_0 & 2*q_2*q_3-2*q_1*q_0 & 1-2*q_1^2-2*q_2^2)] \end{bmatrix} * \begin{bmatrix} K \\ L \\ M \end{bmatrix} \tag{6.3}$$

H_Jacobian is the Jacobian to the function, this Jacobian is shown in the method section and is to large for repetition in this section.

For the quaternion based correction the formula can be reduced due to the fact that the h function is the state variables themselves. This making the Jacobian into a identity matrix with the dimension four times four.

The norm functions is simply the length of the vectors, and is always used for normalizing the vectors to ensure that the unit quaternions stays as unit quaternions as they might drift in length due to numeric integrations or filter corrections on the surface of a hypersphere.

6.2.4 Data analysis tools

The analysis of the error of the estimate, calculated using the results of the correction combined with the real value of the orientation is described in the method part.

$$Angle_{error} = 2*arccos([Q_{est}*inv(Q_{real}]_0) \tag{6.4}$$

```
if y1(1)>1
   y1(1) = 1;
end
if y1(1)<-1
    y1(1) = -1;
end;
v = [y1(2),y1(3),y1(4)];
v = v/norm(v);
v = v * sin(acos(y1(1)));

y2 = [y1(1);v(1);v(2);v(3)];
y3 =  quaternionmultiplication(Quaternion_Inv(X_Correct),y2)
```

Figure 6.5: The algorithm for analysing error angle of quaternions

The quaternion variance analysis uses the method described in the section 5.2.5.

The idea for the approach shown in figure 6.5 is done to preserve the scalar part that defines the rotation angle, though if the magnitude of the scalar part is more than one the this has to be corrected as a unit quaternion can't have values with a magnitude larger than one. The next part is to scale down the vector part of the quaternion based on the magnitude of the scalar part, keeping the direction of the vector. The last part is to multiply the quaternion with the inverse state to rotate it into a quaternion describing the error.

6.3 Positional system using Extended Kalman filter

This section will cover the simulation, representation extraction of measurement data and the filtration of the measurement.

6.3.1 Simulation and representation

The simulation is based on the formulas described in Chapter 5.3, using numeric integration to integrate over time. To visualize the current state of the simulation, a three dimensional real time plot was used to show all the locations that the satellite had been at and at which points that the filter assumed the position to be at. To note is that no smaller errors would be visible on such a plot as the distance from center of Earth to the satellite is in the size of millions of meters and the error is not expected to be near that magnitude. Therefore the plots of the magnitude of the error is in its own graph was added to allow for better analysis of the results. The error representation is quite simple for this simulation as it simply plots the magnitude of the difference in position in a two dimensional graph and the magnitude of the error of velocity in another two dimensional graph with the axis being magnitude of error and time.

```
%Predict

X_Predict = X_Correct +[X_Correct(4) ; X_Correct(5) ; X_Correct(6);
-Acceleration * X_Correct(1) / Radius;
-Acceleration * X_Correct(2) / Radius;
-Acceleration * X_Correct(3) / Radius]*delta;

P_Predict = F_Jacobian*P_Correct*transpose(F_Jacobian)+Qk;


%Correct

H_Matrix = [1,0,0,0,0,0;
    0,1,0,0,0,0;
    0,0,1,0,0,0];

S_Covariance = H_Matrix*P_Predict*transpose(H_Matrix)+RkP;
K_Gain = P_Predict * transpose(H_Matrix)*pinv(S_Covariance);
X_Correct = X_Predict + K_Gain*(Y-X_Predict(1:3));
P_Correct = P_Predict-K_Gain*H_Matrix*P_Predict;
```

Figure 6.6: The algorithm used for positional filtration

6.3.2 Filtration

For filtration of the positional data the extended Kalman filter will be used with the state variables being the vector pointing to the position combined with the velocity vector in the same coordinate frame. For evaluating these state variables the only available measurement would be the GPS. Since the only available measurement of the satellite is the approximated positions using GPS. This means for the extended Kalman filter will base its predictions on a model of the acceleration, which in turn is based on the position of the satellite.

In the algorithm shown in figure 6.6 the state variable X_predict and X_correct are variables describing the current state. The current state is as described in Chapter 5.2.1 the position and the velocities.

$$X_t = \begin{bmatrix} PosX \\ PosY \\ PosZ \\ VelX \\ VelY \\ VelZ \end{bmatrix} \tag{6.5}$$

In Matlab, array and matrix indexing starts at one which means that as the position starts in the x axis X_Predict(1) and X_Correct(1) is prediction and correction of the position along the x axis. The prediction part consist of the first four lines of the Matlab code shown in figure 6.6. The prediction formula was evaluated in the method section for the model of the positioning system, resulting in equation (6.6).

$$X_t = F(X_{t-1}) = \begin{bmatrix} X_{t-1}(1) + X_{t-1}(4) * dt \\ X_{t-1}(2) + X_{t-1}(5) * dt \\ (X_{t-1}(3) + X_{t-1}(6) * dt \\ X_{t-1}(4) - \frac{G*M*X_{t-1}(1)}{X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt \\ X_{t-1}(5) - \frac{G*M*X_{t-1}(2)}{X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt \\ X_{t-1}(6) - \frac{G*M*X_{t-1}(3)}{X_{t-1}(1)^2+X_{t-1}(2)^2+X_{t-1}(3)^2)^{3/2}} * dt) \end{bmatrix} \tag{6.6}$$

The F_Jacobian is the first derivative Jacobian of the function F with regards to the state. This is evaluated in the method section and the results requires too much data to allow for short and efficient representation leaving a recommendation to check section 5.2.2. The F_Jacobian is used for calculating the transformation of the variance of the prediction. Qk is the noise that appears from integration, this includes both the noise of the measurement and the noise of the numeric integration. Due to the fact that the extended Kalman filter risks underestimating the noise when predicting and correcting this value is significantly larger then it in reality should be in order to make sure that the filter does not under estimate the error. The correction terms are linear as the state variables are represented in the same coordinated frame as the input is with regards to position making the corrections linearly dependent with a scalar one to the input variables. This creates the H function of the state to simply be the positional state variables.

$$H_Positional(X) = \begin{bmatrix} X_{t-1}(1) \\ X_{t-1}(2) \\ X_{t-1}(3) \end{bmatrix} \tag{6.7}$$

The Jacobian of the measurement which was simply shown in the method part is just the identity matrix appended by a three by three matrix of zeroes, with the results as shown in equation (6.8).

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{6.8}$$

Chapter 7

Implementation Ada

The previous implementation in Matlab revolved around not using any predefined functions that would not be available as standard conventions in ADA which will help as the system would have to be proven to be critically safe. There are some exceptions for this and it is with regards to the matrix implementation combined with certain matrix inverse algorithms. For the implementation for general computers, rather than embedded system, the generic matrix functions of ADA were used. This includes `Ada.Numerics.Generic_Real_Arrays` and `Ada.Numerics.Generic_Real_Matrices`, the implementation part with the regards to the usage of these predefined matrix operations was done mainly as a proof of concept.

7.1 Design for the attitude determination

The design of the implementation is based on the utilization of protected objects, this means that the integration and all the corrections based on measurements will be evaluated within the protected object. The reason for implementing the filter algorithm in a protected object is to ensure the safe communication between the several different tasks used for gathering measurements and the tasks utilizing the results from the filter. The main idea behind this implementation approach is that several different tasks used for measuring the gyroscope data, the data from the sun sensor, horizon sensor and other available sensors, will in a safe way be able to receive and use their measurements to improve the evaluation of the orientation. The tasks will use that data as input to the different available procedures such as integration of gyroscope data, correction for vector measurement and correction from a unit quaternion used to represent the complete orientation. The integration of gyroscope data is a simple numeric integration and will push the current time forward with the numeric integration of the state and increasing the variance based on how good the numeric integration is assumed to be. The correction from vector measurement will use the variance of the sensor combined with the measured vector to the object in mind

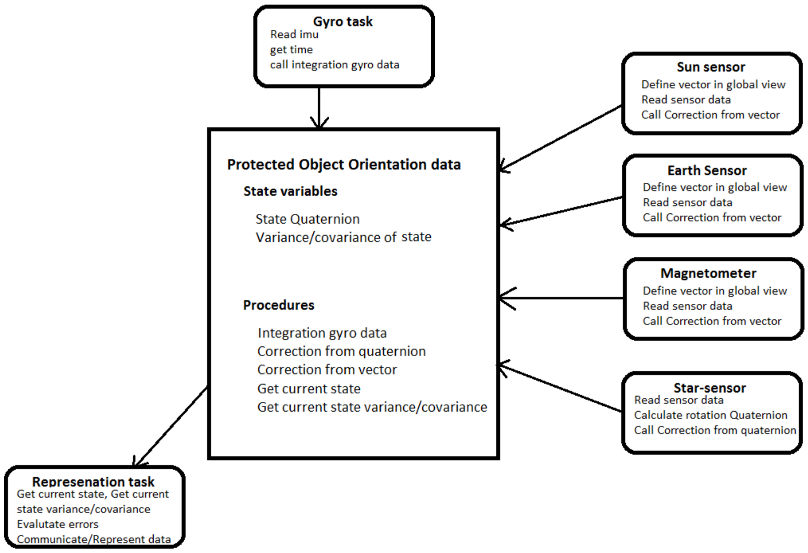

Figure 7.1: Implementation design for the attitude determination

and lastly combined with the global direction of the measurement. These three sets of data combined will allow the filter to make corrections to its current state and reduce the variance covariance of the current state. The last kind of measurement that the filter can correct with regards to is a measurement of the current orientation of the satellite. This measurement is taken in as a quaternion combined with the variance covariance of the measurement quaternion.

The main focus of the implementation will be on the protected object and the features of the procedures used by the protected object. The system design of the different tasks and their communication with the sensors will be left as future work. To show that the filtration within the protected object works as intended and that the algorithm used for the filtration yields good results are the key parts that will be implemented and tested using a single task reading sensor data from a text file.

7.2 Design positioning system

The positioning filters state variables are the position and velocity of the satellite given in xyz coordinates. This is combined with the variance covariance that describes how accurate the current state is. Though being a second order problem the evaluation of the variance covariance suffers from linearisation. The implementation design for the positioning system is quite similar to the filtration system of for the orientation. It is implemented within a protected object with the algorithm built into the procedures and functions of protected object.

The motivation to use the protected object for the positioning system is based on allowing multiple systems running parallel for positioning measurements to complement each other to get more data to filter with, though at this point the focus is on the implementation for one input task for the positioning filter. The main functionality for the positioning system is that when a task, that measures the position, gets an evaluation of the position it calls the filter with the data received and the accuracy of the measurement and the time point for the measurement. The filter then integrates the state variables to that point in time using its estimate of the velocity and formula of acceleration. After this is achieved the correction based on the current position will be used to correct the current position and current velocity based on the relation between the position and the velocity.

Chapter 8

Results

This section covers the results for the tests of the filtered data, the filtration is done using the ADA implementation based on measurements from the simulations ran in Matlab. The simulation's integration are always run with several steps of integration in the simulator before data is fed to the filter. One small set of error that is not introduced which realistically would be is that the simulations and the filters are executed based on the same models of movement. This means for the positioning system that the movement is based on gravity, where both the global constants and the formula for gravity is used for both for the filtering and simulation. This means that these constants and functions are considered to be perfect by the filter which does not have to be the case. The results from running filtrations in both the Matlab implementation and the ADA implementation yielded almost identical results to the degree where indications that rounding of variables were the major affect causing the differences, this gave an indication that identical algorithms were in fact implemented in both implementation.

8.1 Positioning system

The results of the positioning systems will be shown using simulation measurement data of 10 hertz, both representing the velocity error as well as the positional error. The standard deviation of the error of the positional measurement along each axis was set as 50 meters with close to Gaussian distribution.

The results after filtration can be seen in figure 8.1. The graph shows error as a function of time. The error can be seen to never reach above 15 meters and with a standard deviation of 6 meters after the filtering stabilizes.

The error of the angular velocity is shown in figure 8.2, also as a function of time. In the graph the error bellow one meter per second can be extracted, though the improvement of this data can't really be compared to the measurements due to different units of measurement. Though adding the error from the angular velocity for a shorter period of time can be seen to have a small

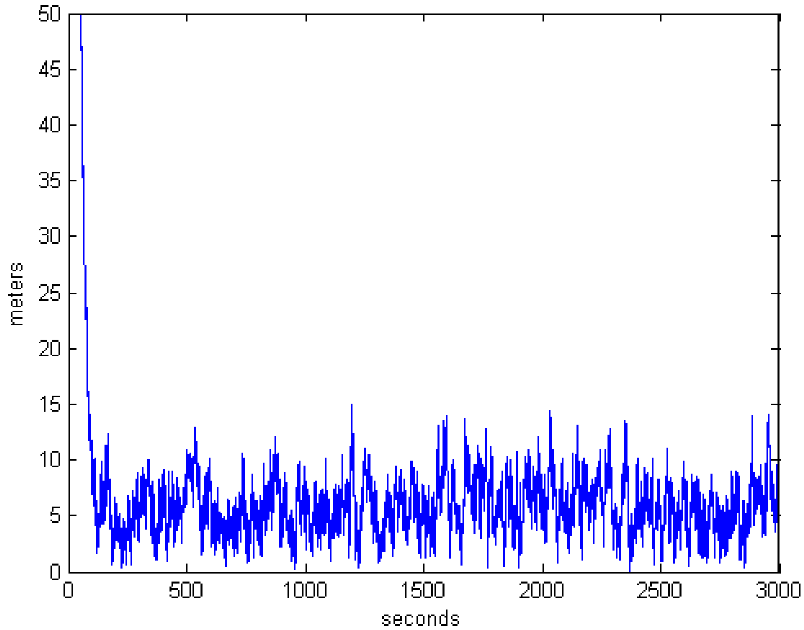

Figure 8.1: The figure shows the magnitude of the positional error over time in meters.

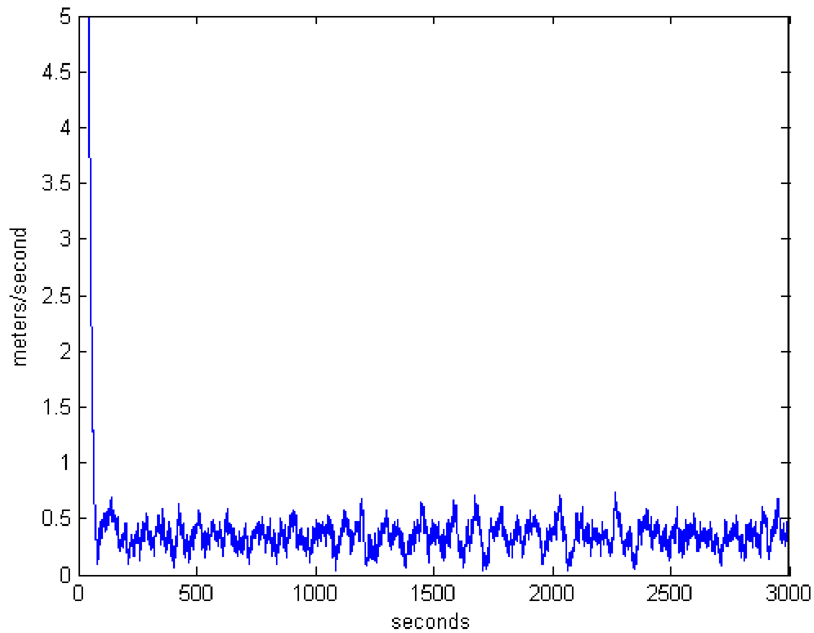

Figure 8.2: The figure shows the magnitude of the error in velocity over time in meters per second

impact on the actual position making predictions for where the satellite will be within reasonable time limits possible. One issue with this implementation of the positioning system was the tuning of parameters for the variance of predictions. The variance of a prediction considering the change in acceleration and considering the small error in velocity should be quite small. Though using that magnitude of variance for the prediction yielded diverging results, as the filter yielded variances stating there was no variance in the prediction. One possible reason for that is the linearisation of the extended Kalman filter making the predictions seem better in the linearised system than they were in reality. To make up for this more variance was added into the predictions to ensure that the variance matrix never reached zero allowing more recent measurements to have a higher impact than those taken at an earlier time points.

8.2 Orientation system

The result for the orientation is split into the simulations run with three different test setups, used to evaluate the performance of the system. These different setups contain two setups with two perpendicular vectors. The measurements on those two simulations was simulated with two and twenty degrees standard deviation of the measurements error. This is combined with one test setup to test the performance of the quaternion correction.

The simulation was done for two perpendicular vectors with a measurement error with a standard deviation of 2 degrees. The results which can be seen in figure 8.3 has a significant improvement on the measured data, the largest error is roughly half the same as the average was on the measurement. The resulting standard deviation for the filtration of the simulation data has a standard deviation of 0.45 degrees approximately a quarter of the original error.

With increased standard deviation of the measurements in the simulations run in figure 8.3. The simulations were run with the same frequency of 10 hertz. The results of this simulation can be seen to improve from a measurement with the standard deviation of 20 degrees to a standard deviation of 0.75 degrees on the results. This means that the measurement error is roughly 30 times larger than the filtered value yielding a significant improvement on the measurement data. The main thing to note here is how the performance of the filter increases when the error gets larger, if the integration set up stays the same.

The quaternion correction can be seen in figure 8.5. When comparing the quaternion correction to the other measurements of correction it is clear how the error is significantly less noisy. What can be state for this simulation is how well the correction is doing in comparison to the vector corrections. The standard deviation of the error in measurement angle for the quaternion correction is at 30 degrees, though this is with uniform variance in the quaternion space which might not be realistic. The results after filtration stays constantly at an significantly lower level than the standard deviation of 30 degrees with not reaching above 0.1 degree.

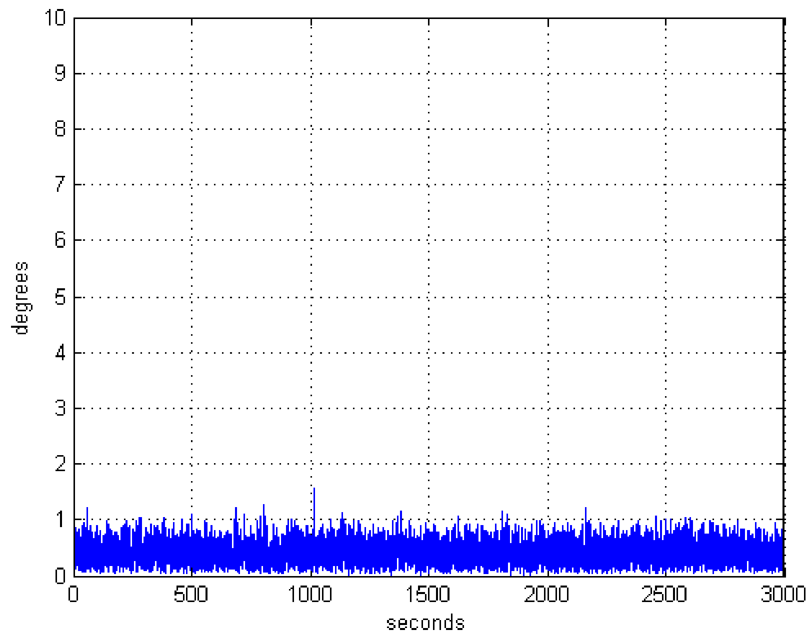

Figure 8.3: Quaternion angle error over time, vector measurement standard deviation 2 degrees

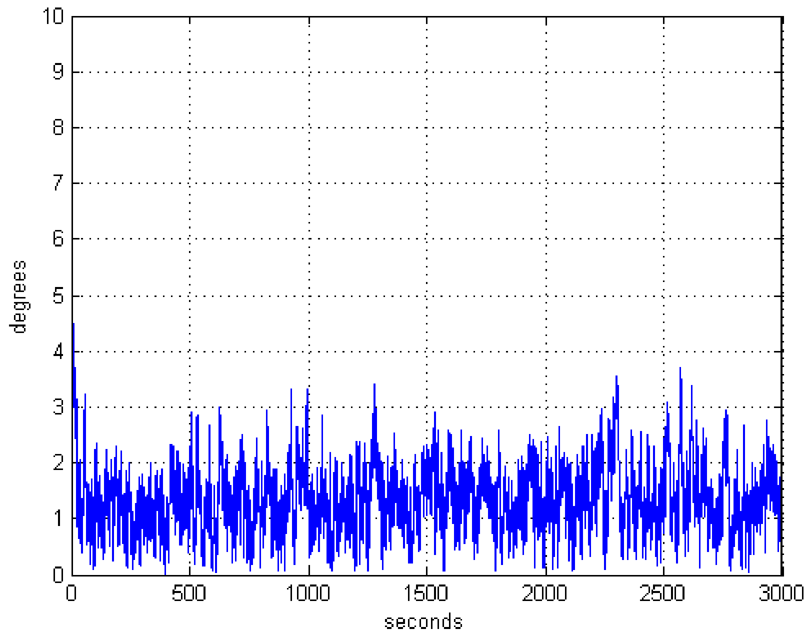

Figure 8.4: Quaternion angle error over time, vector measurement standard deviation 20 degrees

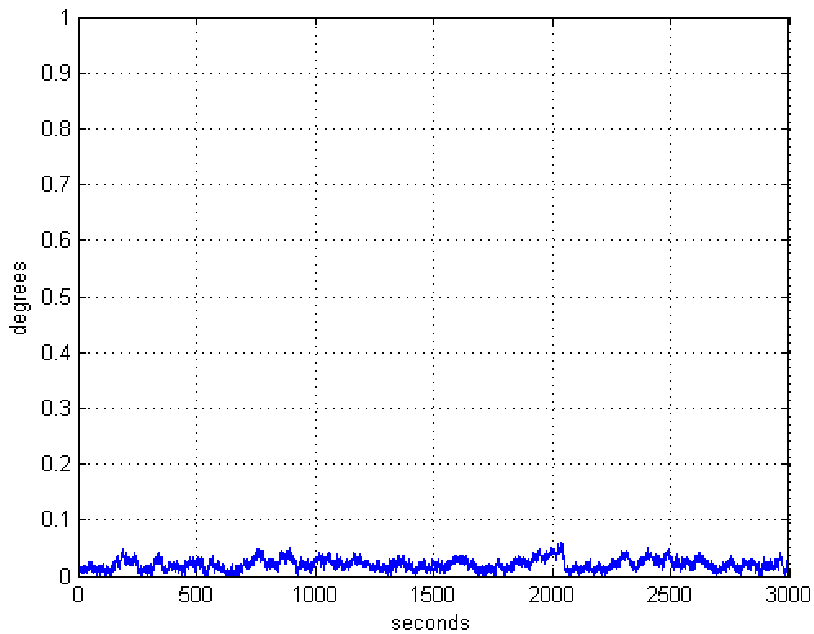

Figure 8.5: Quaternion angle error over time, quaternion measurement standard deviation 30 degrees

Chapter 9

Conclusion

The conclusions that can be made based on the results shown in Chapter 8 is that the filtration with the extended Kalman filter gives significant improvement in results when comparing it with the raw data input for both position and orientation.

The results of the positional filtering yielded an estimation of the position that would be roughly 10 times smaller than the error of the measurement used for the filtration. These results of the simulations show that the system is linear enough to yield significant improvements with the extended Kalman filter.

The results of the vector corrections are generally rather noisy while the quaternion correction had significantly smaller error. One reason for this is that the quaternion correction is linear making the correction optimal while the vector correction is not linear and therefore not optimal. Though the correction of the quaternion correction is only optimal if the variance describing the state is correct.

Chapter 10

Discussion

A discussion topic for the utilization of the extended Kalman filter is with regards to how close to linear is enough for a system to get good results using the Extended Kalman filter. The prediction which can be shown in the orientation filtration is clearly linear enough to give reasonably good results, though the vector correction is not linear enough to give high performance results using the extended Kalman filter.

For the filtration of the position the velocity is accurately being determined, this is one of the main aspects in order for the filtration to determine the position. One of the problem with this is that the outgoing velocity is not following a normal distribution which makes using it for updating the position less optimal but can still be used for filtration of the position where it gets significantly better results than the raw data.

For the orientation prediction step is the common sources of error that exists, which as the quaternion correction is yielding significantly better results than the vector correction should only have minor importance. This starts with the fact that the simulated gyroscope is only read every 0.1 second. Yielding errors that are larger than necessary as most gyroscopes can run up to 200 hertz or more. Another possible source of error is the derivative of the quaternion and the prediction step, using this approach only works if the rotation per calculation is small. This would generally be the case when running a gyroscope at 200 hertz, though it can be further improved by numerically integrating the prediction more than one time per gyroscopes measurement.

The main feature that the author has drawn from this is the importance of linearity in the system when using extended Kalman filters combined that with the fact that particle filters and the unscented Kalman filter are more stable and suitable when the systems are not close to linear.

10.1 Future work

Future work would include improvements in the prediction steps, as that is at least for the orientation system where the author believes that the most significant errors come from. This improvement would also affect the prediction steps derivation and variance. Other future work is on the implementation of the system to the specific hardware that it would be run on. This would include making all code compatible with the Ravenscar profile as well as proving that the code works correctly.

Bibliography

[1] L. Zhao, C. Huang, X. Wang, Y. Hao, and X. Yan, "Study on satellite attitude determination based on hybrid extended kalman," in *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pp. 1110–1114, July 2010.

[2] M. Li, S. yan Wang, B. long Zhu, Y. chun Zhang, and H. yi Li, "Micro-satellite attitude determination algorithm based on adaptive ukf," in *Mechatronic Sciences, Electric Engineering and Computer (MEC), Proceedings 2013 International Conference on*, pp. 2829–2833, Dec 2013.

[3] Y. Jing and L. Xuan, "Satellite attitude determination in post-processing based on urts optimal smoother," in *Control, Automation and Systems (ICCAS), 2012 12th International Conference on*, pp. 267–272, Oct 2012.

[4] L. Cao, W. Yang, X. Chen, and Y. Huang, "Application of multi-sensors data fusion based on improved federal filtering in micro-satellite attitude determination," in *Multi-Platform/Multi-Sensor Remote Sensing and Mapping (M2RSM), 2011 International Workshop on*, pp. 1–5, Jan 2011.

[5] Y. Sugawara, K. Nagasawa, and N. Kobayashi, "An attitude determination method for satellite by the use of image data of a phase of a heavenly body," in *SICE Annual Conference (SICE), 2012 Proceedings of*, pp. 1640–1645, Aug 2012.

[6] H.-N. Shou and C.-T. Lin, "Micro-satellite attitude determination: Using kalman filtering of magnetometer data approach," in *Computer Communication Control and Automation (3CA), 2010 International Symposium on*, vol. 2, pp. 195–198, May 2010.

[7] P. Axelrad and C. Behre, "Satellite attitude determination based on gps signal-to-noise ratio," *Proceedings of the IEEE*, vol. 87, pp. 133–144, Jan 1999.

[8] N. Nadarajah, P. Teunissen, and P. Buist, "Attitude determination of leo satellites using an array of gnss sensors," in *Information Fusion (FUSION), 2012 15th International Conference on*, pp. 1066–1072, July 2012.

[9] N. Abbas, L. Y. Jun, and M. Fiaz, “Attitude determination of small satellite using phase and code measurements of global navigation satellite system: Design, simulation and comparison,” in *Electronics, Communications and Photonics Conference (SIECPC), 2013 Saudi International*, pp. 1–6, April 2013.

[10] M. Grewal and M. Shiva, “Application of kalman filtering to gyroless attitude determination and control system for environmental satellites,” in *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, vol. 2, pp. 1544–1552 vol.2, Dec 1995.

[11] P. P. C. P. M., K. H. K., and V. d. M. R., “Robustness assessment between sigma point and extended kalman filter for orbit determination.” `http://www2.dem.inpe.br/hkk/2011/JAESA-04-PardalKugaMoraes.pdf`, 2011. Accessed: 2014-11-09.

[12] C. Ana Paula Marins, K. Hlio Koiti, and B. d. A. P. Antonio Fernando, “Onboard and real-time artificial satellite orbit determination using gps.” `http://www.hindawi.com/journals/mpe/2013/530516/`, Jan 2013. Accessed: 2014-11-09.

[13] “Nasa gps info.” `http://ilrs.gsfc.nasa.gov/missions/satellite_missions/past_missions/gp35_general.html`. Accessed: 2014-11-09.

[14] C. Zhiguo, C. Pei, and H. Chao, “An algorithm for orbit determination of navigation satellites based on a ground beacon,” in *Measurement, Information and Control (MIC), 2012 International Conference on*, vol. 2, pp. 1051–1055, May 2012.

[15] G. Shorshi and I. Bar-Itzhack, “Satellite autonomous navigation and orbit determination using magnetometers,” in *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*, pp. 542–548 vol.1, 1992.

[16] G. Murray, “Rotation about an arbitrary axis in 3 dimensions.” `http://inside.mines.edu/fs_home/gmurray/ArbitraryAxisRotation`, June 2013. Accessed: 2014-09-02.

[17] “Rotations using quaternions.” `http://www.ecsutton.ece.ufl.edu/ens/handouts/quaternions.pdf`. Accessed: 2014-09-03.

[18] G. A. Terejanu, “Unscented kalman filter tutorial.” `http://inside.mines.edu/fs_home/gmurray/ArbitraryAxisRotation`. Accessed: 2014-09-03.

[19] “Transparency in safety-critical systems.” `http://intelligence.org/2013/08/25/transparency-in-safety-critical-systems/`. Accessed: 2014-09-03.

[20] G. Li, M. Lu, and B. Liu, "A scenario-based method for safety certification of artificial intelligent software," in *Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference on*, vol. 3, pp. 481–483, Oct 2010.

[21] L. Lovisolo, "Uniform distribution of points on a hyper-sphere with applications to vector bit-plane encoding." `http://www02.smt.ufrj.br/~eduardo/papers/ri15.pdf`. Accessed: 2014-09-03.