Zoltán Belső, Balázs Gáti, István Koller, Péter Rucz, Antal Turóczi

DESIGN OF A NONLINEAR STATE ESTIMATOR FOR NAVIGATION OF AUTONOMOUS AERIAL VEHICLES

The aim of this paper is to present a novel approach for the design and implementation of an onboard nonlinear state estimation system for an autonomous aerial vehicle. The tasks of such a system include collection of measurement data from different navigation sensors, and estimation of all the quantities describing the system's state of motion based on the system dynamics and the measurement data. The widely accepted method for such a sensor fusion problem is the Kalman filter in the case of a linear system. The dynamics of a rigid body inherently involves nonlinearity, therefore a nonlinear extension of the Kalman filter is needed. The proposed solution relies on the Unscented Kalman Filter (UKF) technique with making use of both the quaternion and Rodrigues parameters representations of the attitude. The developed method is tested in MATLAB and implemented in the onboard embedded system in C code. The applicability of the proposed method is demonstrated in this paper by means of various examples.

Keywords: *UAV, sensor fusion, Unscented Kalman Filter (UKF), quaternion, Rodrigues parameters*

INTRODUCTION

The task of navigation and control of an autonomous aerial vehicle (sometimes called UAV) is the task of the robotic pilot or autopilot. The autopilot system consists of sensors for the detection of the state of motion of the system, actuators for applying forces and an embedded microprocessor and appropriate software to execute the control algorithm. The control algorithm is responsible for producing the appropriate driving signals for the actuators in order to keep the vehicle on the desired trajectory. This is accomplished by comparing the actual state (more precisely the estimation of the actual state) with the desired reference signal. The reference of the state variables is produced by the guidance algorithm based on the (estimation of the) actual state and the desired flight path. The estimation of the actual state of motion is produced by the state estimation algorithm from the navigation sensor measurements (Figure 1). This paper focuses only on the state estimation algorithm. The control and guidance algorithms are discussed in more detail in the accompanying paper [1].

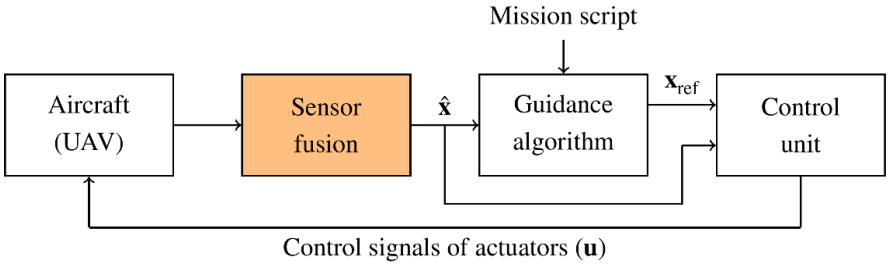


Figure 1. The autopilot in an UAV system

The first step in the control of an autonomous robotic vehicle is determining its position and state of motion. The vehicle as a rigid body has six degrees of freedom (6DoF): the position of its center of mass in some reference frame (three degrees) and its orientation (another three degrees).

In order to describe the state of motion of the vehicle we must choose a reference frame. One obvious choice would be the one attached to fixed stars. Its greatest advantage is that it is an inertial frame of reference, so the laws of mechanics take their simplest forms. In case of a vehicle working in the geographical vicinity of its starting position it would be natural to choose the starting point as the origin of the reference frame and attach the frame to the Earth. This is not an inertial frame of reference in strict sense,[1] but in our case the difference is negligible. Thus, the most common choice is the north-east-down (NED) system, called the Earth frame.

One of the difficulties using the Earth frame is that the quantities describing the vehicle's moment of inertia are changing as the vehicle turns (changes its orientation). We can attach the axes of the coordinate system to the principal axes of the vehicle body: the x-axis points in the forward direction, the y-axis points to the right and the z-axis points downwards. We could fix the frame's origin at the vehicle's center of mass, but in that case the position would always be [0, 0, 0], which is rather useless. Therefore we define the body frame as follows: the origin of the frame fixed at the same point as the reference Earth frame origin, but the orientation of the axes are attached to the principal axes of the vehicle's body [1][3].

The motion of the center of mass obeys Newton's laws. These are a set of linear differential equations expressed in the Earth frame. However, the description of the orientation which is the rotation between the reference frame and the frame attached to the body is inherently nonlinear. Rotation can be represented in terms of rotation angles, but use of angle values involves the usage of trigonometric functions which are not linear. There are descriptions of rotation in which not the angle is the state variable but some quantity related to its sinusoid. In these cases the trigonometric functions may be eliminated but in exchange other nonlinear functions (like square and square root) are involved. These issues are examined in the next section in detail.

During the processing some quantities are expressed in Earth frame while others are expressed in body frame. Time to time we need to transform the quantities from body frame to Earth frame and back, which involves multiplying state variables with each other. This is again a nonlinear operation, so we need to deal with nonlinear techniques [4][5].

DESCRIPTION OF ORIENTATION

The orientation (also called angular position or attitude) is a part of the description of the state of motion of a rigid body. This is the transformation that rotates the Earth frame into the body frame.

The orientation of the vehicle must be incorporated into the state space in some way, therefore its representation must be chosen appropriately. The orientation being a rotation in three-dimensional space can be described in several ways. In the following we summarize the advantages and drawbacks of the most common representations.

[1] We need to calculate with the Coriolis force caused by the Earth's rotation and some more complex transformation if we want to take into consideration the Earth's rather complex motion relative to the fixed stars.

Rotation matrix

A rotation transformation can be described by a rotation matrix (or direction cosine matrix, DCM). A rotation matrix in a three-dimensional space is a 3 by 3 matrix with nine real elements. The DCMs are orthonormal matrices, so their inverse coincides with their transpose.

The greatest advantage of DCMs is the ease of calculating the transformed version of a given vector: obtained by a vector by matrix multiplication.

The disadvantage of the DCM representation is redundancy. It has nine elements for describing the three degrees of freedom of the 3D rotation operation. The orthogonal property means a strong correlation between the elements which must be preserved during transformations. If we include the redundant DCM representation in the state space the covariance matrix of the state will be singular, and the system we get is not controllable [6].

Euler angles

Another natural description of a rotation in three-dimensional space is by three rotations around the three main axes of the body. Unfortunately the order of the rotations does matter, altogether there are 12 possibilities. The most commonly used order is rotation around the x, then the y, and finally the z-axis. These are the roll, pitch and yaw angles, respectively.

The Euler angles have three dimensions for describing the three degree of freedom system. For small angles (in the vicinity of level straight flight) it has a nice linear behavior by the approximation of $\sin(x)$ by x.

From the Euler angles description we can always get the rotation matrix description by:

$$\mathbf{R}_{\mathrm{DCM}} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ -\cos\theta\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix}, \tag{1}$$

where ϕ, θ, ψ are roll, pitch, yaw, respectively.

The Euler angle representation has some well-known drawbacks. First of all it has a discontinuity at ±180°, which is one of the most severe non-linearity one can face with. The second problem is the so-called gimbal lock: For some orientations the description is not unique. For the roll–pitch–yaw order this occurs when the pitch angle is ±90°, in this case the roll and yaw angles describe a rotation around the same axis. This way we lose one degree of freedom, which leads to a singularity in the system equations [6].

Quaternions

The orientation is a rotation, hence it can be represented by a rotation axis and a rotation angle. Let the rotation axis be the unit vector k and the rotation angle be α. Hence we can represent an orientation (and any rotation) by a unit quaternion:

$$\mathbf{q} = \begin{bmatrix} \cos\frac{\alpha}{2} \\ k_x \sin\frac{\alpha}{2} \\ k_y \sin\frac{\alpha}{2} \\ k_z \sin\frac{\alpha}{2} \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}. \tag{2}$$

This way for every orientation there are two unit quaternion representations: the sing of k and α are interchangeable. If we restrict our description having only positive angles then the representation is unique.

Quaternions are similarly easy to use as rotation matrices: two rotations in series is the quaternion product of the two representing quaternions, and rotating a vector is expressed as a vector by quaternion product. The quaternion representation is also free from the discontinuity problem the Euler angles representation has. A quaternion represents a three degrees of freedom system by four values, therefore the same redundancy problem arises as in the case of DCMs [6][7].

Rodrigues parameters

The Rodrigues parameters are a three-dimensional representation of a rotation projecting the four-dimensional hypersphere of unit quaternions onto a three-dimensional hyperplane. By placing the projection point and the hyperplane into different locations different variants of the Rodrigues parameters can be constructed. Choosing the projection focal point to be [-1, 0, 0, 0] and the first coordinate of the hyperplane to be 0 (Figure 2) we get a singularity free parameter representation up to a principal rotation of ±360°, also free from the gimbal lock phenomenon.

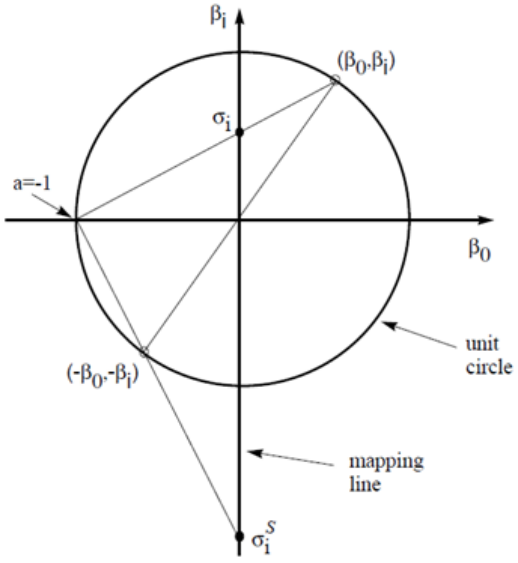


Figure 2. The Rodrigues projection demonstated in 2 dimensions. [8]

The Rodrigues parameters s_i are derived from the quaternion elements q_i with the following formula:

$$s_i = \frac{q_i}{1 + q_0}, \qquad i = 1, 2, 3. \tag{3}$$

The transformation in the opposite direction reads as:

$$q_0 = \frac{1 - s^{\mathrm{T}} s}{1 + s^{\mathrm{T}} s}, \qquad q_i = \frac{2 s_i}{1 + s^{\mathrm{T}} s}, \qquad i = 1,\ 2,\ 3. \tag{4}$$

The rotation matrix representation of a rotation described by a set of Rodrigues parameters can be calculated as follows:

$$\mathbf{R}_{\mathrm{DCM}} = \frac{1}{(1+n)^2} \begin{bmatrix} 4\left(s_1^2 - s_2^2 - s_3^2\right) + u^2 & 8 s_1 s_2 + 4 s_3 u & 8 s_1 s_3 - 4 s_2 u \\ 8 s_2 s_1 - 4 s_3 u & 4\left(-s_1^2 + s_2^2 - s_3^2\right) + u^2 & 8 s_2 s_3 + 4 s_1 u \\ 8 s_3 s_1 + s_2 u & 8 s_3 s_2 - 4 s_1 u & 4\left(-s_1^2 - s_2^2 + s_3^2\right) + u^2 \end{bmatrix}, \tag{5}$$

where $n = \mathbf{s}^T \mathbf{s}$ is the norm square of the Rodrigues parameters vector and $u = 1 - n$.

One additional advantage of using Rodrigues parameters for describing the orientation is that for small angles they are approximately proportional to the Euler angles, so similar linear approximation apply to them.

This way for the unit quaternions whose first element is positive (this is one half of all the unit quaternions, representing all possible rotations) we get Rodrigues parameters inside the unit sphere. The other half of the unit quaternions are mapped to the Rodrigues parameters outside the unit sphere, having counterparts (representing the same rotation) inside the unit sphere. This way the transition at the ±180° rotation boundary is smooth, but in case of large angles we are far from the linear approximation [8].

NAVIGATION SENSORS AND SENSOR FUSION

Navigation is based on data coming from various sensors. For some state variables there are direct measurements available. For example, the GPS sensor can measure the absolute position in the Earth frame. Some sensors measure in the body frame. For some other state variables there are no direct measurements, these states can only be obtained by derivation which may cause increased uncertainty or accumulation of errors.

In the following the advantages and disadvantages of different sensors are addressed.

Absolute measurement sensors

There are some sensors that can measure some of the state variables in our reference frame, that is the Earth frame. These measurements are sometimes called absolute measurements (opposing to relative or difference measurements, which measure the rate of change of some quantity).

The most commonly used absolute measurement is GPS position. It provides the absolute geographical position in the Earth frame. By differentiation we could get the velocity in the Earth frame, but due to the inaccuracy (the noise) of the position measurement, it would be practically useless. Most GPS receiver units provide an independent velocity measurement in the Earth frame based on the Doppler principle.

An other type of absolute measurement is the altitude measurement based on barometric pressure. Although the values must be corrected by the pressure measured on the reference

height, during the typical operation time window of an autonomous device the measurement can be regarded as an absolute altitude above the surface.

Inertial sensors

Inertial sensors are sensing the change of the object's state of motion. The two most commonly used inertial sensors are the accelerometer, and the angular velocity sensor (called gyroscope in this context). The third sensor widely used in conjunction with the previous two is a magnetic field sensor. Although it has nothing to do with the change of the object's state of motion, as a sensor it has very similar properties to true inertial sensors, and the three types of sensors are commonly used together for attaining the object's state of motion. A system composed of inertial sensors is sometimes called inertial measurement unit or IMU.

In a strapdown navigation system all of these sensors measure in the body frame. The measurements have some errors. The measured values are corrupted by noise and other sensor errors. The noise can be well considered as zero mean additive white Gaussian noise. The offset and scale error can theoretically be eliminated by calibration [9][10]. The problem here is that the error may change over time and temperature, so the calibration procedure must be repeated frequently.

Utilizing these measurements we can derive the attitude in several ways. By integrating the angular velocity over time from a starting orientation we may get the attitude. The inherent and inevitable problem here is that we also integrate the noise added to the measurement. By integrating a zero mean noise we get a random walk process. This problem can only be eliminated by some kind of absolute measurement.

The accelerometer measures both the specific force and the gravitational acceleration. If the body (the center of mass) does not accelerate we get the gravitational vector alone. Together with the Earth's magnetic vector measured by the magnetometer we may get an absolute measurement for the orientation.

On the other hand if the orientation is known the gravitational acceleration can be subtracted from the measured acceleration to get the acceleration of the center of mass. By integration we can get the linear velocity, and by one more integration we get the position. The same problem applies here as in the case of angular velocity and orientation: integrating the noise added to the measurement causes a random walk. Double integration makes the problem even more severe.

Sensor fusion problem

As we have seen previously all of the sensors have some errors. Not all of the state variables can be measured directly, but for most of the state variables we have several derivations, each having its own weaknesses and advantages. The method of taking different measurements and derived quantities for the same value and combining (averaging) them together is called sensor fusion.

In the case of a 6DoF system a special problem has to be overcome: some of the measurements are available in the Earth frame, some others are available in the body frame. Since these are measurements for the same principal value, they should be fusioned together. We must choose one of the two frames and transform the values that are given in the other. The transformation is the rotation representing the orientation, which is part of the state space itself. That is, we do

not know the exact value of the transformation, we only have the estimation of it. That is extremely problematic when the measurements (directly or indirectly) are used for the estimation of the orientation. The system could be underdetermined and may not converge. This situation must be detected and some change of the system model may be required based on the available measurements. In the sequel a possible solution for that problem is proposed.

THE KALMAN FILTER

Linear optimal estimation problem

The Kalman filter is the optimal solution for state space estimation of linear systems based on measurements corrupted by additive zero mean Gaussian white noise with known covariance. The filter is optimal in the sense of minimizing the mean square of the estimation error. The Kalman filter can be regarded as a special case of the least mean square method.

Instead of giving the exact derivation of the Kalman filter (which can be found in many good books, e.g. [9][12]) we just state the most common form. We restrict the description to the discrete time version.

Given a linear system with $\mathbf{x}(n)$ as the state vector and $\mathbf{u}(n)$ as the input vector for every n time instance:

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) + \mathbf{B}\mathbf{u}(n) + \mathbf{w}(n). \tag{6}$$

Here the matrices $\mathbf{A}$ and $\mathbf{B}$ together represent the linear system of equations describing the state transition through the T_s sampling time interval. $\mathbf{w}(n)$ is the so-called process noise, representing all the effects and errors not exactly modeled by the system equations. We suppose that the process noise comes from many independent sources and these are small enough to be described by a set of independent zero mean normally distributed random variables with covariance matrix $\mathbf{Q}(n)$.

The outputs of the system are the quantities for which we have direct measurements, denoted by the vector $\mathbf{y}(n)$:

$$\mathbf{y}(n) = \mathbf{C}\mathbf{x}(n) + \mathbf{v}(n). \tag{7}$$

Here the matrix $\mathbf{C}$ represents a linear mapping (function) between the states and the measurements, and $\mathbf{v}(n)$ denotes to the noise of the measurement, described by a set of independent zero mean normal distribution random variables with covariance matrix $\mathbf{R}(n)$.

The exact value of $\mathbf{x}(n)$ is not known, it can only be estimated. Let us denote the latter by $\hat{\mathbf{x}}(n)$, which is a random variable itself with normal distribution. The expected value of $\hat{\mathbf{x}}(n)$ is $\mathbf{x}(n)$, and the covariance is denoted by $\mathbf{P}(n)$. This is the uncertainty we know about $\mathbf{x}(n)$.

The Kalman filter works in two steps. The first is called 'innovation', that is when the system equations are applied:

$$\mathbf{x}^{-} = \mathbf{A}\hat{\mathbf{x}}(n) + \mathbf{B}\mathbf{u}(n). \tag{8}$$

That is the expected value of the transformed state vector. From that we can make a prediction about the output values (the measurements):

$$\hat{\mathbf{y}}(n) = \mathbf{C}\mathbf{x}^{-}(n). \tag{9}$$

After performing the actual measurements we can make a difference between the prediction and the reality:

$$\mathbf{d}(n) = \mathbf{y}(n) - \hat{\mathbf{y}}(n). \tag{10}$$

Here comes the second step, the so-called 'correction':

$$\mathbf{x}^{+} = \mathbf{x}^{-} + \mathbf{K}_n\mathbf{d}(n). \tag{11}$$

This way we get an estimation of the state vector in the next time instance:

$$\hat{\mathbf{x}}(n+1) = \mathbf{x}^{+}. \tag{12}$$

The block diagram of the Kalman filter can be seen on Figure 3.

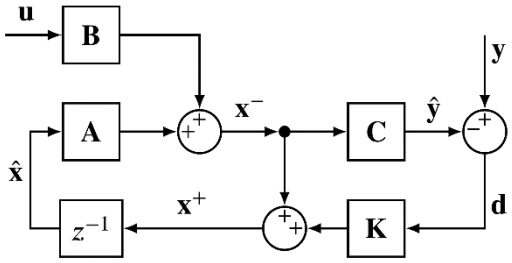


Figure 3. The Kalman filter

The $\mathbf{K}_n$ feedback matrix is calculated based on the covariance of the estimation and the measurement noise so that we get the new state estimation as a weighted average of the predicted and the measured values.

The covariance (uncertainty) of the state vector after the innovation is expressed as:

$$\mathbf{P}_n^{-} = \mathbf{A}\mathbf{P}_{n-1}\mathbf{A}^{\mathrm{T}} + \mathbf{Q}_n. \tag{13}$$

This is obtained by calculating the covariance of the transformation of a normal distribution random variable added to the process noise covariance.

$\mathbf{K}_n$ is calculated as:

$$\mathbf{K}_n = \mathbf{P}_n^{-}\mathbf{C}^{\mathrm{T}}\left(\mathbf{C}\mathbf{P}_n^{-}\mathbf{C}^{\mathrm{T}} + \mathbf{R}_n\right)^{-1} = \mathbf{P}_n^{+}\mathbf{C}^{\mathrm{T}}\mathbf{R}_n^{-1}. \tag{14}$$

Note that $\mathbf{K}_n$ depends only on the noise covariance matrices (**R** and **Q**) and the system matrices (**A** and **C**) and of course on $\mathbf{P}_0$. In a linear system it is independent of the actual measurement and the values of the state variables.

The covariance (uncertainty) of the state vector after the correction is expressed as:

$$\begin{aligned}\mathbf{P}_n = \mathbf{P}_n^{+} &= \left(\mathbf{I} - \mathbf{K}_n\mathbf{C}\right)\mathbf{P}_n^{-}\left(\mathbf{I} - \mathbf{K}_n\mathbf{C}\right)^{\mathrm{T}} + \mathbf{K}_n\mathbf{R}_n\mathbf{K}_n^{\mathrm{T}} \\ &= \left(\mathbf{P}_n^{-1} + \mathbf{C}^{\mathrm{T}}\mathbf{R}_n^{-1}\mathbf{C}\right)^{-1} = \left(\mathbf{I} - \mathbf{K}_n\mathbf{C}\right)\mathbf{P}_n^{-}.\end{aligned} \tag{15}$$

In the case of a linear system with time invariant (constant) noise covariance matrices the uncertainty matrix (**P**) and the feedback matrix (**K**) have a steady state value, the calculations in

equations (13)–(15) do not need to be performed in every time step. The steady state **K** can be precalculated and equations (8)–(12) applied in every time step.

Extended Kalman Filter

When the system is nonlinear we can use linear approximation and apply the Kalman filter on the latter. One way of doing so is the method of linearization about an operating point. We can calculate a different linear approximation based on the Jacobian method and use a different system matrix for every time instance.

This method is called extended Kalman filter (EKF). In this case there is no steady state value for the feedback matrix, the equations (8)–(15) must be calculated for every time step.

The linearization can be made by other methods too. One other method is called 'State Dependent Coefficients' (SDC) or 'State Dependent Riccati Equation' (SDRE). The nonlinear system equation

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{16}$$

is factorized into a state dependent coefficients form as:

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x})\mathbf{x} + \mathbf{B}(\mathbf{x})\mathbf{u}, \tag{17}$$

and the Kalman filter equations (8)–(15) are applied to the state dependent matrices.

The problem of factorization of the system equations is discussed in more detail in the accompanying paper [1].

Unscented Kalman Filter

In the Kalman filter the feedback matrix (**K**) is calculated using the covariance matrices by equation (14). The noise covariance matrices (**R** and **Q**) are considered given, but the state covariance matrix (**P**) is calculated based on the linear transformation of a normally distributed random variable.

In case of a nonlinear system there is no general formula for calculating the expected value and the covariance of the transformed random variable (furthermore the transformed random variable does not necessarily follow the normal distribution).

By approximating some way the transformed mean and covariance of the state variables ($\hat{\mathbf{x}}$) and the predicted output ($\hat{\mathbf{y}}$) and the crosscorrelation between these two we can calculate the feedback matrix.

This approximation for a random vector **x** with covariance **P** can be made by carefully choosing sigma points ($\mathbf{x}^{\sigma}$) around the vector **x** in a way that the mean of the sigma points gives the **x** and covariance of these sigma points gives **P** (

Figure 4). Then, by making the nonlinear transformation on all sigma points the mean and covariance of the transformed sigma points serve as an estimation of the transformed mean and covariance of the random variable **x**.

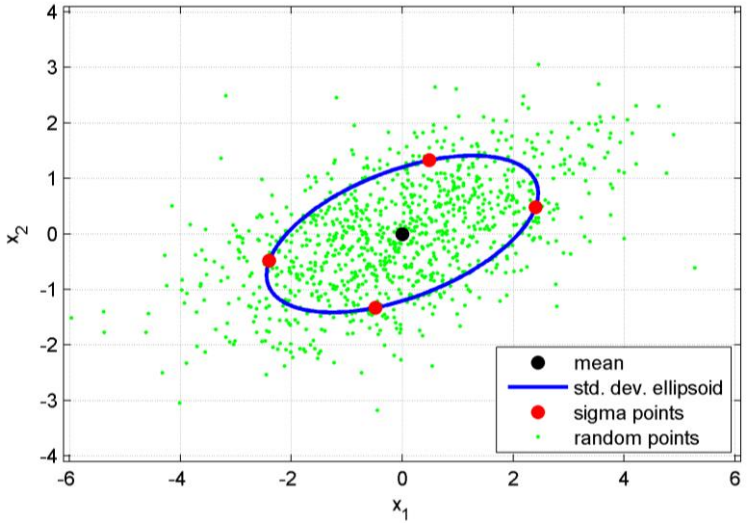


Figure 4. Generation of sigma points

The method is the following (denote the state vector by $\mathbf{x}(n)$, its dimension by N, its covariance by $\mathbf{P}_n$):

1. In each timestep n for the state vector $\mathbf{x}(n)$ choose $2N$ sigma points $\mathbf{x}_i^\sigma$ as:

$$\mathbf{x}_i^\sigma, \mathbf{x}_{N+i}^\sigma = \mathbf{x}_n \pm \boldsymbol{\sigma}_i, \quad i = 1 \dots N, \tag{18}$$

where $\boldsymbol{\sigma}_i$ is chosen as the rows of the $\sqrt{N\mathbf{P}_n}$ matrix. This way the statistics of the sigma points gives the statistics of the $\mathbf{x}(n)$ random variable.

2. For these sigma points apply the system equation. Denote the transoformed sigma points by $\mathbf{x}_i^{\sigma*}$. The mean of these transformed points will be the estimate for the new state vector:

$$\hat{\mathbf{x}}^- = \frac{1}{2N}\sum_{i=1}^{2N} \mathbf{x}_i^{\sigma*}, \tag{19}$$

and the covariance of the trasformed points is:

$$\hat{\mathbf{P}}^- = \frac{1}{2N}\sum_{i=1}^{2N} \left(\mathbf{x}_i^{\sigma*} - \hat{\mathbf{x}}^{\sigma-}\right)\left(\mathbf{x}_i^{\sigma*} - \hat{\mathbf{x}}^{\sigma-}\right)^{\mathrm{T}}. \tag{20}$$

Adding the covariance of the process noise ($\mathbf{Q}$) we get:

$$\mathbf{P}^- = \hat{\mathbf{P}}^- + \mathbf{Q}. \tag{21}$$

3. In order to estimate the output vector ($\hat{y}$) and its covariance create sigma points around $\hat{\mathbf{x}}^-$ with the help of $\mathbf{P}^-$. Apply the output equation on these sigma points and denote the result by $\mathbf{y}_i^\sigma$. The mean of these vectors will be the estimation $\hat{\mathbf{y}}$. Calcuate the covariance of the $\mathbf{y}_i^\sigma$ vectors ($\mathbf{P}_{yy}$) and the crosscorrelation with the $\mathbf{x}_i^{\sigma*}$ vectors:

$$\mathbf{P}_{yy} = \frac{1}{2N}\sum_{i=1}^{2N} \left(\mathbf{y}_i^\sigma - \hat{\mathbf{y}}\right)\left(\mathbf{y}_i^\sigma - \hat{\mathbf{y}}\right)^{\mathrm{T}}, \tag{22a}$$

$$\mathbf{P}_{xy} = \frac{1}{2N}\sum_{i=1}^{2N} \left(\mathbf{x}_i^{\sigma*} - \hat{\mathbf{x}}^{\sigma-}\right)\left(\mathbf{y}_i^\sigma - \hat{\mathbf{y}}\right)^{\mathrm{T}}. \tag{22b}$$

4. Calculate the feedback matrix as:

$$\mathbf{K}_n = \mathbf{P}_{xy} \cdot \mathbf{P}_{yy}^{-1}. \tag{23}$$

Calculate the estimator of the state vector and its covariance after correction:

$$\mathbf{x}_{n+1} = \hat{\mathbf{x}}^{+} = \mathbf{x}^{-} + \mathbf{K}_n \left(\mathbf{y}_n - \hat{\mathbf{y}}\right), \tag{24}$$

$$\mathbf{P}_{n+1} = \mathbf{P}^{+} = \mathbf{P}^{-} + \mathbf{K}_n \left(\mathbf{P}_{yy} + \mathbf{R}\right) \mathbf{K}_n^{\mathrm{T}}. \tag{25}$$

The blockdiagram of the whole process is depicted on Figure 5.

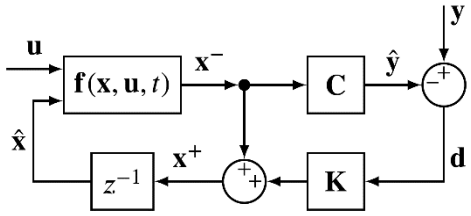


Figure 5. The Kalman filter with nonlinear system function

State space augmentation

The method above assumes that the process and measurement noises act in a linear manner (as additive white Gaussian noise) to the otherwise nonlinear system. Colored noise (not following Gaussian distribution and/or not being independent) can be handled as appropriately filtered white noise. The system equations and the state space can be extended by the filter equations and internal states.

In many practical cases there is no better assumption for the noises but the white Gaussian noise, but in the system equations the noises do not act in a linear way. This is a case for example when two state variables (both with their process noise added) are multiplied.

In this case the state space does not need to be truly extended, only the system equations are written as if the noise values would be incorporated into the state vector. Sigma points need to be generated around the zero noise vector and the system equations are applied to them. The statistics of the transformed $\mathbf{x}_i^{\sigma *}$ vectors yield the estimation $\hat{\mathbf{x}}^{-}$ and its covariance $\widehat{\mathbf{P}}^{-}$, but the $\mathbf{Q}$ matrix does not need to be added to $\widehat{\mathbf{P}}^{-}$ as it has already been taken into consideration in the augmentation procedure.

IMPLEMENTATION OF UKF

In this section the implementation choices for the Kalman filter are investigated. The implementation is based on the Unscented Kalman Filter algorithm with state space augmentation. For handling the gimbal lock problem a special state space choice is applied. For handling the severe nonlinearities arising during attitude measurement a virtual measurement procedure is defined.

State space choice

The state space must represent the state of motion of the vehicle including its three degrees of freedom translation and also its orientation and angular velocity. For the linear motion there are measurements available for the position and the acceleration, therefore the position, the velocity and the acceleration

should all be included in the state space. The estimation of the offset of the gyroscope sensor is also included in the hope of the successful compensation of this disturbing effect.

The proposed choice of state space is the following:

$$\mathbf{x} = \begin{bmatrix} \mathbf{r} \\ \boldsymbol{\beta}^{(\mathrm{B})} \\ \boldsymbol{\omega}^{(\mathrm{B})} \\ \mathbf{a}^{(\mathrm{B})} \\ \mathbf{v}^{(\mathrm{E})} \\ \mathbf{p}^{(\mathrm{E})} \end{bmatrix}, \tag{26}$$

where **p**, **v** and **a** are the position, velocity and acceleration vectors, **ω** is the angular velocity vector and **β** is its estimated offset, **r** is the Rodrigues parameter triplet, while the upper index $^{(\mathrm{B})}$ or $^{(\mathrm{E})}$ means that the quantity is represented in the body or the Earth frame, respectively.

The Rodrigues parameters vector is a minimal representation of the orientation behaving nicely for small angles but as discussed in the navigation sensors section it has discontinuity, therefore it is not applicable for representing arbitrary angles. The quaternion representation is free from discontinuities but it is a redundant representation resulting in a singular covariance matrix which causes numerical difficulties. The absolute orientation is chosen to be stored in a separate quaternion **q** while the Rodrigues parameters representing the difference to the quaternion (delta Rodrigues parameters) are stored in state space. The covariance matrix (**P**) is updated based on the Rodrigues parameters.

The system equations for the integration step are written using quaternions. The delta Rodrigues parameters for each sigma point are converted to delta quaternions ($\delta\mathbf{q}_i$) using formula (4) and multiplied by the stored quaternion:

$$\mathbf{q}_0 = \mathbf{q}^+ \tag{27a}$$

$$\mathbf{q}_i = \delta\mathbf{q}_i \otimes \mathbf{q}^+, \tag{27b}$$

where $\mathbf{q}^+$ is from the previous step and $\otimes$ denotes to quaternion multiplication.

The integration is as follows:

$$\mathbf{q}_i^{\mathrm{tr}} = \mathbf{q}_i + \frac{1}{2} T_{\mathrm{s}} \cdot \boldsymbol{\Xi}_i \boldsymbol{\omega}, \tag{28}$$

where

$$\boldsymbol{\Xi} = \begin{bmatrix} -q_1 & -q_2 & -q_3 \\ q_0 & -q_3 & q_2 \\ q_3 & q_0 & -q_1 \\ -q_2 & q_1 & q_0 \end{bmatrix}. \tag{29}$$

From the transformed quaternions we compute the transformed delta qutaternions:

$$\delta q_i^{\mathrm{tr}} = q_i^{\mathrm{tr}} \otimes \overline{q_0^{\mathrm{tr}}}, \tag{30}$$

and finally the transformed delta quaternions are converted to transformed delta Rodrigues parameters.

The other state variables are integrated as usual:

$$\boldsymbol{\beta}^{(\mathrm{B})\mathrm{tr}} = \boldsymbol{\beta}^{(\mathrm{B})}, \tag{31a}$$

$$\boldsymbol{\omega}^{(\mathrm{B})\mathrm{tr}} = \boldsymbol{\omega}^{(\mathrm{B})}, \tag{31b}$$

$$\mathbf{a}^{(\mathrm{B})\mathrm{tr}} = \mathbf{a}^{(\mathrm{B})}, \tag{31c}$$

$$\mathbf{v}^{(\mathrm{E})\mathrm{tr}} = \mathbf{v}^{(\mathrm{E})} + T_\mathrm{s}\mathbf{a}^{(\mathrm{E})}, \tag{31d}$$

$$\mathbf{p}^{(\mathrm{E})\mathrm{tr}} = \mathbf{p}^{(\mathrm{E})} + T_\mathrm{s}\mathbf{v}^{(\mathrm{E})} + \tfrac{1}{2}T_\mathrm{s}^2\mathbf{a}^{(\mathrm{E})}. \tag{31e}$$

The statistics for the covariance matrix **P** is calculated from the state vector containing the delta Rodrigues parameters according to the rules presented in the previous section.

The output equation is the following:

$$\mathbf{y} = \begin{bmatrix} \mathbf{r} \\ \boldsymbol{\omega}^{*(\mathrm{B})} \\ \mathbf{a}^{*(\mathrm{B})} \\ \mathbf{p}^{(\mathrm{E})} \end{bmatrix}, \tag{32}$$

where $\mathbf{a}^{*(B)}$ and $\boldsymbol{\omega}^{*(B)}$ are formed to be directly comparable to the measurable values:

$$\mathbf{a}^{*(\mathrm{B})} = \mathbf{a}^{(\mathrm{B})} + \mathbf{g}^{(\mathrm{B})}, \tag{33a}$$

$$\boldsymbol{\omega}^{*(\mathrm{B})} = \boldsymbol{\omega}^{(\mathrm{B})} + \boldsymbol{\beta}, \tag{33b}$$

where **g** denotes to the gravitational acceleration.

The **r** Rodrigues vector is not directly measurable, but a virtual measurement is defined for the attitude measurement based on vector measurements. This virtual measurement is discussed in detail in the following subsection. After the correction step the stored quaternion ($\mathbf{q}^+$ in the previous step) is updated by the delta Rodrigues parameters resulting in the new $\mathbf{q}^+$.

Virtual attitude measurements

In many navigation applications, the attitude determination is based on vector or direction observations. In the case of small UAVs, these reference directions are usually the Earth's magnetic field vector and the gravity vector. Measuring the known reference vectors in the airplane-fixed body frame the orientation of the airplane can be estimated. Almost all attitude determination algorithms using vector observations are based on the minimization of the loss function:

$$L(\mathbf{A}) = \frac{1}{2}\sum_i a_i \left|\mathbf{b}_i - \mathbf{A}\mathbf{r}_i\right|^2 \tag{34}$$

or in alternative form:

$$L(\mathbf{A}) = \sum_i {a_i}^2 - \mathrm{tr}(\mathbf{A}\mathbf{B}^\mathrm{T}), \quad \text{where} \quad \mathbf{B} = \sum_i a_i \mathbf{b}_i \mathbf{r}_i^\mathrm{T}, \tag{35}$$

proposed by G. Wahba [13]. In equation (34) and (35) $\mathbf{b}_i$ denotes a set of unit vectors measured in the body frame, $\mathbf{r}_i$ are the corresponding unit vectors in a reference frame, and a_i are non-negative weights. The problem is to find the orthogonal matrix **A** that minimizes the above loss function. There are several different algorithms to find **A**, but they have different robustness, speed and accuracy properties. The most robust yet computationally most intensive algorithm is based on the singular value decomposition of **B**:

$$\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} = \mathbf{U}\,\langle \Sigma_{11}, \Sigma_{22}, \Sigma_{33} \rangle\,\mathbf{V}^{\mathrm{T}}, \tag{36}$$

where **U** and **V** are orthogonal matrices and the singular values obey the inequalities $\sum_{11} \geq \sum_{22} \geq \sum_{33} \geq 0$. The optimal orientation defined by the optimal rotation matrix can be calculated as [14]:

$$\mathbf{A}_{\mathrm{opt}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} = \mathbf{U}\,\langle 1, \quad 1, \quad \det(\mathbf{U}) \cdot \det(\mathbf{V}) \rangle\,\mathbf{V}^{\mathrm{T}}. \tag{37}$$

This rotation matrix can be converted to the orientation representation used in the state space.

Switching between models

In a real environment special conditions can occur that must be handled properly. The robustness of the system means that it can react and behave properly in cases outside the normal operating conditions.

In our system the position and attitude estimation is based both on the GPS and IMU sensors. The interaction between the GPS and the IMU sensors is twofold:

1. The GSP sensor produces measurement data less frequently than the IMU. In the cycles between two GPS sensor measurement the position data is updated by integration of the acceleration estimate.
2. The gravity vector measurement is based on the accelerometer sensor by subtracting the linear acceleration, which is corrected by the position data from the GPS sensor.

The GPS signal is sometimes lost due to various disturbances. The navigation system must be prepared to be able to work without GSP data. In our system this case is handled by switching to a reduced system model whenever the position estimate becomes uncertain. The problems that arise in case of GPS signal outages:

1. The acceleration data obtained by the accelerometer sensor is not corrected by the GPS position data become uncertain, which causes instability in the gravity vector measurement.
2. Large uncertainty values in the **P** covariance matrix can cause numerical instabilities at the matrix square root calculation during the generation of sigma points.

In the reduced model the following simplifications apply:

1. The accelerometer is assumed to be measuring solely the $\mathbf{g}^{(B)}$ vector. This approximation is correct when the linear acceleration of the center of mass is small enough ($|\mathbf{a}| \ll |\mathbf{g}|$).
2. In the **P** covariance matrix the values belonging to the $\mathbf{v}^{(E)}$ velocity and the $\mathbf{p}^{(E)}$ position vectors are not updated but kept constantly equal to the initial $\mathbf{P}_0$ values. These $\mathbf{P}_0$ values are large enough to allow fast convergence should the GPS data become available again, but at the same time small enough to guarantee numerical stability.

3. All the feedback from the position data is disabled by zeroing the columns of the $\mathbf{K}_n$ matrix belonging to the position measurement.

Matlab simulation and embedded implementation in C

For the quantitative analysis of the proposed algorithm a Matlab simulation model is created. The input of the simulation is the ideal measurement data corrupted by artificial noise and other errors. The input data can be created by different sources. For simple cases a Simulink model of the 6DoF rigid body is used. For a more realistic case the mathematical model of a real life UAV and the airplane simulation method introduced in [15] are utilized. Finally, the same Matlab model is used to test the algorithm on collected real sensor readings. The Matlab code is created using only basic linear algebraic operations (no special toolbox is used) in order to facilitate the translation to C code.

The goal is to create a realtime application running on an embedded ARM processor under Linux operating system. The performance of the system depends highly on the efficient implementation of the linear algebraic operations. Some of the latter are quite complicated, like the Householder transformation or the QR decomposition. For this matter the LAPACK library [16] is utilized. The C version of the algorithm is created in such manner that the Matlab input data can also be used for its testing. The estimation data produced by the C algorithm can be imported into Matlab and the results can easily be compared. Finally, the implementation is tested realtime, fed by real sensor data, and both the raw measurements and the estimation data are saved for later analysis in Matlab.

CASE STUDIES

In the sequel, we demonstrate the capabilities of the implemented UKF algorithm by means of simulated measurement data. The preprocessing (input data file generation) and the postprocessing (output data visualization) were carried out using Matlab.

The case study presented here simulates forces and moments acting on a 6DoF rigid body. Let us suppose there are inertial sensors and GPS fixed to the center of mass of the simulated body. The motion of the body and the ideal (noiseless and errorless) measurement readings were simulated in Simulink. Artificial band limited white Gaussian noise and other errors were added to the ideal measurements according to the datasheet parameters of real sensors used in our experimental system.

Loss of GPS signal

In the following the behavior of the presented implementation will be investigated in the case of GPS signal loss. The sampling time of the algorithm and the inertial sensors is $T_S^{\mathrm{IMU}} = 0.01\mathrm{s}$ while the sampling time of the GPS measurements is $T_S^{\mathrm{GPS}} = 0.1$ s. The simulation time interval is $T_{\mathrm{sim}} = 200$ s. During the simulation GPS outages were assumed in the time intervals $20\ \mathrm{s} < t \leq 60\ \mathrm{s}$ and $100\ \mathrm{s} < t \leq 160\ \mathrm{s}$.

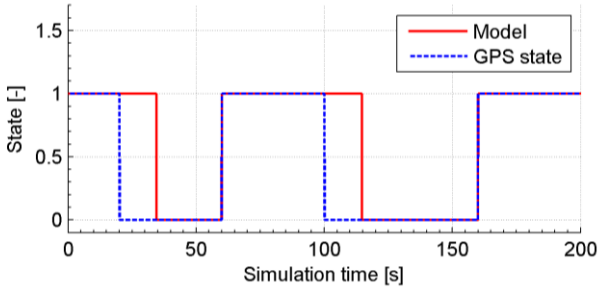


Figure 6. Model switch and GPS outages

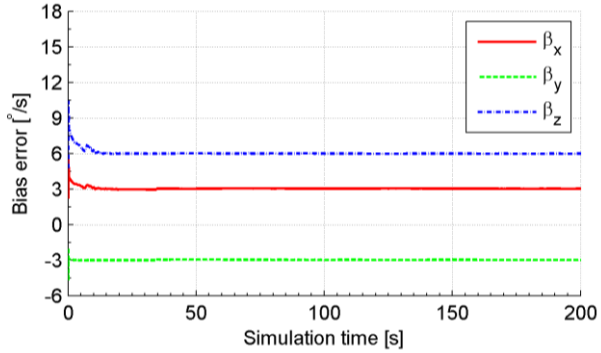


Figure 7. Gyroscope bias estimation

In Figure 6 the system model switch and GPS signal outages can be seen. The value 1 means normal system model and strong GPS signal, while the value 0 means reduced system model and GPS outages, respectively. Selecting an appropriate value for the maximal acceptable position covariance in $\mathbf{P}_{\max}$ the normal system model double integrating the acceleration can provide usable position information up to 13 s after GPS outage. Selecting an appropriate value for the minimal acceptable position covariance in $P_{\min}$ an instant system model switch (in a single sampling interval) can be achieved when GPS measurement becomes available again.

In the simulation the modeled gyroscope measurement data is corrupted by noise and bias error. The knowledge or the proper estimation of the bias term is crucial in attitude estimation. The bias term is set to $[\beta_x\ \beta_y\ \beta_z]^{\mathrm{T}} = [+3 \quad -3 \quad +6]^{\mathrm{T}}$ °/s. As can be seen in Figure 7 the bias converges fast (in couple of seconds) to the correct value. The GPS outages do not affect the bias estimation process.

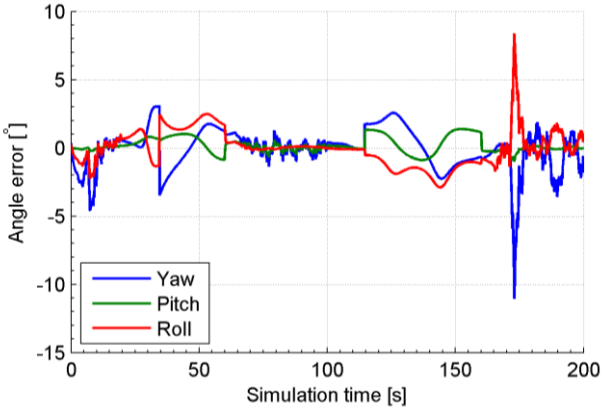


Figure 8. Orientation estimation error in Euler angles

In Figure 8 the error of the orientation estimation can be seen. During GPS outages the error variation is smoother yet higher. At $t \approx 170$ s a high Euler angle error can be observed, but it is a false error signal due to the gimbal lock effect. The variance of the Euler angle error calculated in the simulation time interval is

$$\boldsymbol{\sigma}_{\text{Euler}}^{\text{norm}} = \begin{bmatrix} \sigma_\psi & \sigma_\theta & \sigma_\phi \end{bmatrix}^{\text{T}} = \begin{bmatrix} 1.56° & 0.27° & 1.03° \end{bmatrix}^{\text{T}} \tag{38a}$$

$$\boldsymbol{\sigma}_{\text{Euler}}^{\text{redu}} = \begin{bmatrix} \sigma_\psi & \sigma_\theta & \sigma_\phi \end{bmatrix}^{\text{T}} = \begin{bmatrix} 1.61° & 0.92° & 1.74° \end{bmatrix}^{\text{T}}, \tag{38b}$$

where $\boldsymbol{\sigma}_{\text{Euler}}^{\text{norm}}$ and $\boldsymbol{\sigma}_{\text{Euler}}^{\text{redu}}$ are variances belonging to the normal and reduced system models respectively. The simulation results show the GPS measurement based correction has a significant influence on the orientation estimation when the body is subject to a long-term acceleration. However, in the present simulation the attitude estimation error remained in an acceptable range even in the case of the reduced system model.

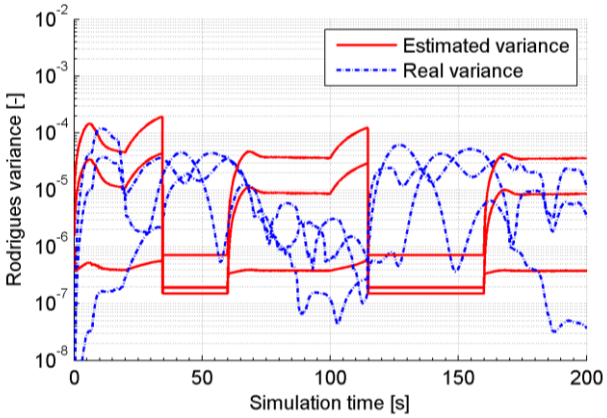


Figure 9. Time function of the variation of the Rodigues parameters

Beside the state estimation, the Kalman filter also provides an estimation of the covariance matrix of the state variables. In Figure 9 the time function of the diagonal elements of the covariance matrix can be seen corresponding to the Rodrigues parameters. To enhance the visibility a sliding window averaging filter is applied. The results show that the estimated and real variances are close using the normal system model; however, in case of the reduced system model the estimated variances are lower than the real ones.

As can be seen in Figure 10 the position estimation in the x–y plane starting from the $(x, y, z) = (0, 0, 0)$ point is close to the ideal trajectory. However, the error of the position estimation is continuously increasing during GPS outages. The error of the position estimation from the ideal trajectory can be characterized by the calculated variances:

$$\boldsymbol{\sigma}_{\text{pos}}^{\text{norm}} = \begin{bmatrix} \sigma_x & \sigma_y & \sigma_z \end{bmatrix}^{\text{T}} = \begin{bmatrix} 1.85\,\text{m} & 2.45\,\text{m} & 2.86\,\text{m} \end{bmatrix}^{\text{T}} \tag{39a}$$

$$\boldsymbol{\sigma}_{\text{pos}}^{\text{redu}} = \begin{bmatrix} \sigma_x & \sigma_y & \sigma_z \end{bmatrix}^{\text{T}} = \begin{bmatrix} 106.0\,\text{m} & 51.64\,\text{m} & 13.41\,\text{m} \end{bmatrix}^{\text{T}}. \tag{39b}$$

In the absence of GPS measurements, the variance of the position estimation with the reduced system model is clearly unusable. It should be noted that this situation is identified based on the operating mode of the filter and the end user can be notified.

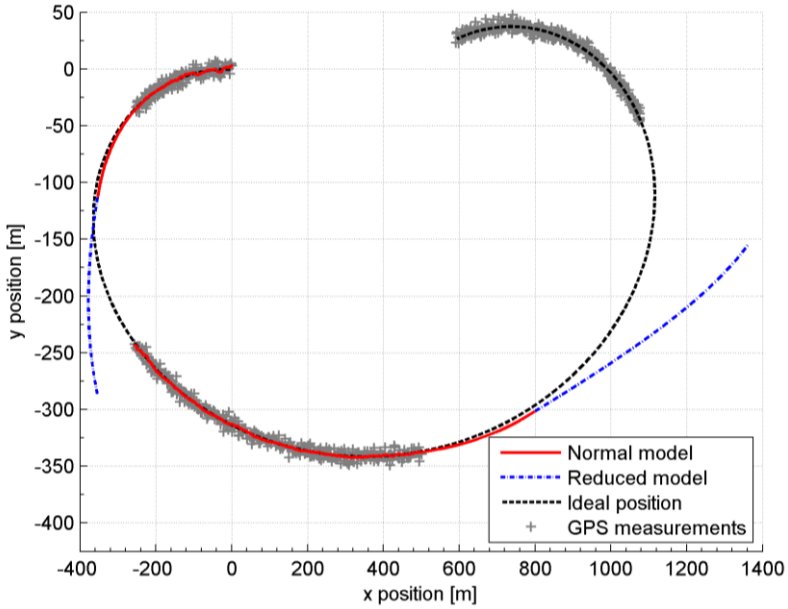
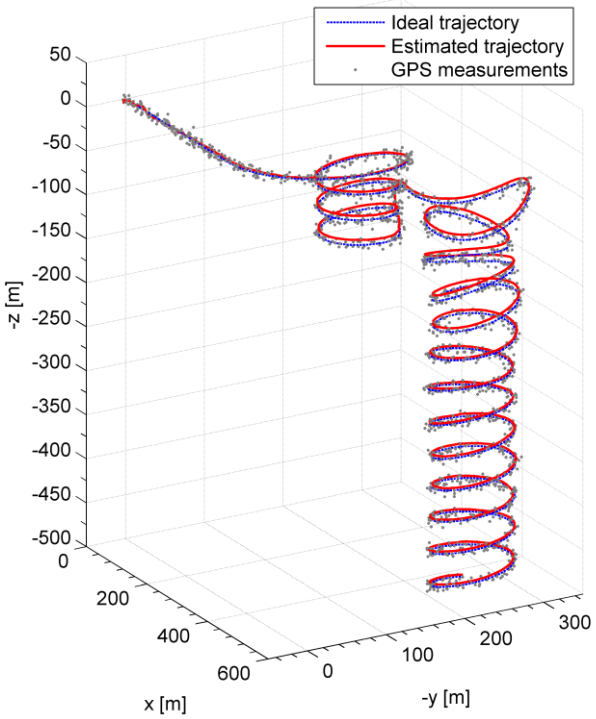


Figure 10. Position estimation using normal and reduced models

Airplane in a descending spin

It is a hard task to estimate the state of an airplane when it is exposed to large accelerations and angular velocities during aggressive maneuvers. To demonstrate the capabilities of the proposed state estimation algorithm, sensor (gyroscope, accelerometer, magnetometer and GPS) measurement readings of a spinning UAV were simulated using the mathematical model of a real life UAV and the airplane simulation method introduced in [15].



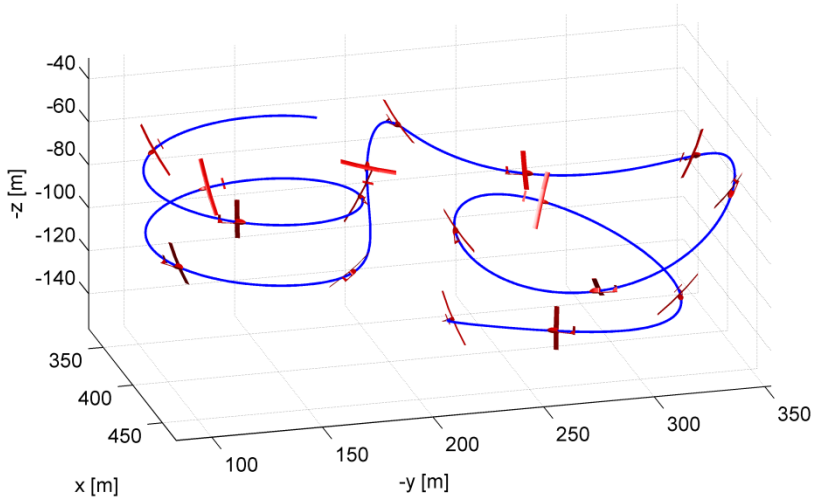


Figure 11. Position measurement and estimation (top) and a part of the simulated trajectory (bottom)

As seen in Figure 11 after a short straight flight, the plane was driven into a descending spin. Beside the gravity vector the onboard accelerometer measurement is also affected by the acceleration of the plane. The difficulty of the orientation estimation arises when the acceleration of the airplane is comparable to the gravity vector that is the case in the simulation results presented in Figure 12. The part of the trajectory where the largest accelerations occur is illustrated on the bottom panel of Figure 11. As can be seen on Figure 12 the estimation of the acceleration is quite accurate.

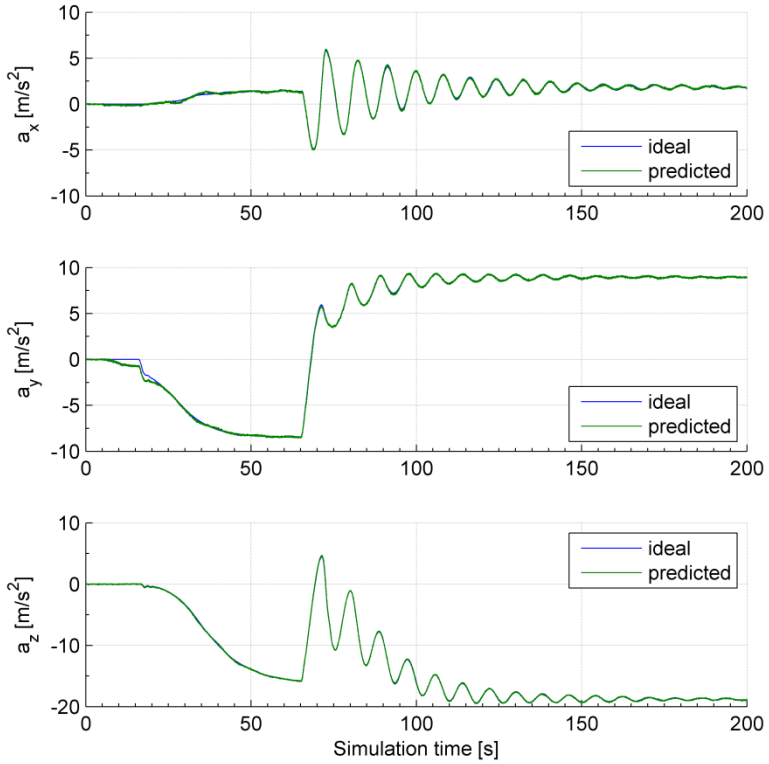


Figure 12. The acceleration during simulation

Theoretically, in order to acquire the gravity vector for the virtual measurement of the orientation the accelerometer readings are corrected by the GPS measurement. This correction is incorporated into our UKF implementation. As can be seen in Figure 13 the proposed state estimation algorithm can estimate the orientation of the UAV with acceptable error. The largest attitude estimation errors observable in the first interval of the simulation are due to the initial converging state of the filter.

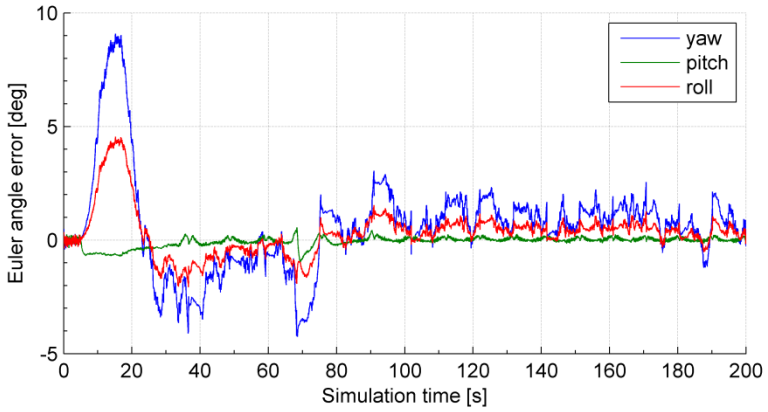


Figure 13. The orientatin error

The variance of the Euler angle, the acceleration, the velocity and the position estimation errors calculated in the simulation time interval are:

$$\boldsymbol{\sigma}_{\text{pos}} = \begin{bmatrix} \sigma_x & \sigma_y & \sigma_z \end{bmatrix}^{\text{T}} = \begin{bmatrix} 1.26\,\text{m} & 1.42\,\text{m} & 3.87\,\text{m} \end{bmatrix}^{\text{T}} \tag{40a}$$

$$\boldsymbol{\sigma}_{\text{vel}} = \begin{bmatrix} \sigma_x & \sigma_y & \sigma_z \end{bmatrix}^{\text{T}} = \begin{bmatrix} 0.52\,\text{m/s} & 0.81\,\text{m/s} & 0.50\,\text{m/s} \end{bmatrix}^{\text{T}} \tag{40b}$$

$$\boldsymbol{\sigma}_{\text{acc}} = \begin{bmatrix} \sigma_x & \sigma_y & \sigma_z \end{bmatrix}^{\text{T}} = \begin{bmatrix} 0.095\,\text{m/s}^2 & 0.17\,\text{m/s}^2 & 0.081\,\text{m/s}^2 \end{bmatrix}^{\text{T}} \tag{40c}$$

$$\boldsymbol{\sigma}_{\text{Euler}} = \begin{bmatrix} \sigma_\psi & \sigma_\theta & \sigma_\phi \end{bmatrix}^{\text{T}} = \begin{bmatrix} 2.23° & 0.23° & 1.10° \end{bmatrix}^{\text{T}}. \tag{40d}$$

Performance of the implementation

The execution time of our algorithm was 15 s in a today's average PC meaning 7.5% CPU load in a real time implementation. In contrast, the execution time was 60 s in our embedded experimental system meaning 30% CPU load of a single core of the four core system with real time measurements. Based on these results it can be stated that this highly complicated sensor fusion algorithm can be used in practice on a relatively small and cheap embedded system.

CONCLUSIONS AND OUTLOOK

In this paper the task of state estimation of unmanned aerial vehicles was discussed. The nonlinear behavior of the orientation description was introduced and the advantages and drawbacks of various possible attitude representations were discussed. After surveying the properties of some navigation sensors the Kalman filter and some of its nonlinear extensions were presented. A possible solution of the problem based on the Unscented Kalman Filter algorithm was presen-

ted. A Matlab simulation and a C implementation of the proposed algorithm were also discussed. The implementation utilizes the LAPACK library and runs effectively in an embedded onboard computer with ARM processor and Linux operating system. Several scenarios were tested during the study, including the outage of the GPS signal, the elimination of the gyroscope offset and the effect of severe accelerations during aggressive maneuvers. The results presented in the case studies section of the paper indicate that the proposed sensor fusion algorithm could also be applied in real-life UAV flight scenarios.

There are several aspects of the implementation that could benefit from further study. The effect of magnetic anomalies corrupting the measurements of the magnetic sensor is out of the scope of this paper. However, these anomalies inevitably occur in real environments. Such distortions can be caused by either soft and hard iron part of the vehicles or the currents running in the vicinity of the sensors. Such anomalies must be eliminated or at least reduced in a successful application by means of a sophisticated calibration procedure. Another possibility for further development is optimizing the computational effort of the algorithm. Although it was shown that the computational power of such an embedded system is capable of running the implementation, further optimization of the algorithm can speed up the execution time providing safety and offloaded CPU.

ACKNOWLEDGEMENT

The Hungarian government, in the form of a KMR_12-1-2012-0121 Research and Developement Project, named “AMORES” (Autonomous Mobile Remote Sensing), supported the research presented in this paper. The authors gratefully acknowledge this financial support.

LITERATURE

[1] P. RUCZ, Z. BELSŐ, B. GÁTI, I. KOLLER, A. TURÓCZI: Design and implementation of nonlinear control systems for rotary and fixed wing UAVs, submitted for publication into the same volume
[2] S. H. STOVALL: Basic Inertial Navigation, Naval Air Warfare Center Weapons Division, 1997. (Online) url: http://www.globalsecurity.org/space/library/report/1997/basicnav.pdf (2015.11.27)
[3] C.-P. FRITZEN: Machine dynamics and system dynamics, 2014. (Online) url: http://www.mb.uni-siegen.de/imr3/lehre/dynamics/skript_2014/md_script_ss2014.pdf (2015.11.27)
[4] H. K. KHALIL: Nonlinear systems. Prentice Hall, Upper Saddle River, New Jersey, third edition edition, 2001.
[5] A. ISIDORI: Nonlinear control systems, Volume I. Springer-Verlag, London, third edition edition, 1995.
[6] Y.-B. JIA: Quaternions and rotations: Problem solving techniques for applied computer science, 2013. Lecturer Notes. (Online) url: http://web.cs.iastate.edu/~cs577/handouts/quaternion.pdf (2015.11.27)
[7] N. TRAWNY, S. I. ROUMELIOTIS: Indirect Kalman filter for 3D attitude estimation: A tutorial for quaternion algebra. Technical Report 2005-002, Rev. 57, University of Minnesota, March 2005. (Online) url: http://www-users.cs.umn.edu/~trawny/Publications/Quaternions_3D.pdf (2015.11.27)
[8] H. SCHAUB, J. L. JUNKINS: Stereographic orientation parameters for attitude dynamics: A generalization of the Rodrigues parameters. Journal of Astronautical Sciences, 44(1):1–19, 1996.
[9] D. GEBRE-EGZIABHER, G. H. ELKAIMY, J. D. POWELLZ, B. W. PARKINSONX: A non-linear, two-step estimation algorithm for calibrating solid-state strapdown magnetometers. Department of Aeronautics and Astronautics, Stanford University. (Online) url: http://waas.stanford.edu/papers/Gebre_ICINS_2001_ins201.pdf (2015.11.27)
[10] S. Bonnet, C. Bassompierre, C. Godin, S. Lesecq, A. Barraud: Calibration methods for inertial and magnetic sensors. Sensors and Actuators A 156 (2009) 302–311. url: http://www-ist.cea.fr/publicea/exl-doc/200900000986.pdf (2015.11.27)

[11] P. S. MAYBECK: Stochastic models, estimation and control: Volume 1. Academic Press, 1982.
[12] D. SIMON: Optimal state esimation. Wiley & Sons, 2006.
[13] WAHBA, GRACE: A Least Squares Estimate of Spacecraft Attitude, SIAM Review, Vol. 7, No. 3, July 1965, p. 409.
[14] F. L. MARKLEY, D. MORTARI: How to estimate attitude from vector observations. In AAS/AIAA Astrodynamics Specialist Conference, Girdwood, Alaska, August 1999.
[15] T. GAUSZ, B. GÁTI. Merevszárnyú és többrotoros légi eszközök modellje precíziós repülési feladatok szimulációjához, 2013. AMORES projekt, kutatási jelentés.
[16] NETLIB Repository at UTK and ORNL. (online) url: http://netlib.org/ (2015.11.27)

NEMLINEÁRIS ÁLLAPOTBECSLŐ TERVEZÉSE ROBOTREPÜLŐGÉPEK NAVIGÁCIÓJÁHOZ

Jelen cikk célja egy új módszer bemutatása robotrepülőgépek nemlineáris fedélzeti állapotbecslő rendszerének tervezéséhez és megalósításához. A feladat magában foglalja különböző navigációs szenzorokból származó mérési adatok gyűjtését és a rendszer dinamikája valamint a gyűjtött mérési adatok alapján a rendszer mozgásállapotát leíró értékek becslését. Ilyen szenzorfúziós feladat megoldására lineáris rendszerek esetében a széleskörben elterjedt megoldás a Kálmán-szűrő alkalmazása. Mivel a kiterjedt merev testek dinamikai leírása eredendően nemlineáris, a Kálmán-szűrő nemlineáris kiterjesztésére van szükség. Az általunk javasolt módszer az úgynevezett Unscented Kalman Filter (UKF) technikán alapul. Az orientáció leírására kvaterniók és Rodrigues-paraméterek egyidejű használatát javasoljuk. A kifejlesztett módszert MATLAB-ban teszteltük, majd egy, a fedélzeti beágyazott környezetben futó C kódú implementációt készítettünk. A módszer alkalmazásának lehetőségeit a cikkben példákkal is szemléltetjük.

Kulcsszavak: *UAV, szenzorfúzió, Unscented Kalman Filter (UKF), kvaternió, Rodrigues-paraméterek*

http://www.repulestudomany.hu/folyoirat/2015_3/2015-3-19-0241_Belso_Z_et_al.pdf