A QUANTISED STATE SYSTEMS APPROACH FOR JACOBIAN FREE EXTENDED KALMAN FILTERING

Lars Alminde, Jan D. Bendtsen and Jakob Stoustrup *

* *Department of Electronic Systems, Section of Automation and Control, Aalborg University,*
{*alminde, dimon, jakob*}*@es.aau.dk*

Abstract: Model based methods for control of intelligent autonomous systems rely on a state estimate being available. One of the most common methods to obtain a state estimate for non-linear systems is the Extended Kalman Filter (EKF). In order to apply the EKF an expression must be available for the Jacobian of the driving function; for complex systems this can be difficult to obtain. This paper presents an EKF variation that makes use of integrated quantised state simulation to propagate the state and obtain a backward difference estimate of the Jacobian at a small computational cost. A simulation study of a deep space probe is presented.

Keywords: non-linear systems, quantised state systems, Kalman filtering

1. INTRODUCTION

Model based methods for control of intelligent autonomous systems rely on a state estimate being available. One of the most common methods to obtain a state estimate for non-linear systems is the Extended Kalman Filter (EKF) (Grewal and Andrews 1993). In order to apply the EKF an expression must be available for the Jacobian of the driving function; for complex systems this can be difficult to obtain.

This paper presents an EKF variation that makes use of integrated Quantised State Simulation (QSS) to propagate the state and obtain a backward difference estimate of the Jacobian at a small computational cost.

A simulation case study involving a spinning deep space probe is presented which must infer its attitude and angular rates from vector-observations of the sun and a fixed star. The simulation confirms that the QSS filter operates consistently and demonstrates the advantages of the approach.

A companion paper, see (Alminde *et al.* 2007), describes how the QSS approach can also be used for control of non-linear autonomous systems.

The paper is organised as follows: In section (2) an introduction to quantised state simulation is given, followed by a review of the standard formulation of extended Kalman filtering in section (3). Hereafter it is described how QSS and the EKF can be combined for efficient Jacobian free EKF estimation in section (4). Section (5) describes the case study and section (6) gives the simulation results together with a discussion of the results. Conclusions are given in section (7).

2. QUANTISED STATE SYSTEMS

Quantised State Systems is a recent approach for propagating ordinary differential equations by decoupling the states and transforming the system into a discrete event system, where event times are dynamically decided using a quantum separation criteria between two models of different order

for each state. The approach was developed by Kofman (2002). This paper will make use of the QSS2 algorithm in which the event time is decided using the difference between a first order and second order model. This technique is reviewed in the remainder of this section, and further details can be found in Kofman (2002).

The QSS2 algorithm simulates systems of the following form, with state vector $\mathbf{x}$ of dimension n and input vector $\mathbf{u}$ of dimension m:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{1}$$

$$\mathbf{y} = \mathbf{g}(\mathbf{x}) \tag{2}$$

The QSS2 algorithm integrates the state and produces the output $\mathbf{g}(\mathbf{x})$ by decoupling the system into event-passing software entities representing functions and integrators respectively, see fig. 1.

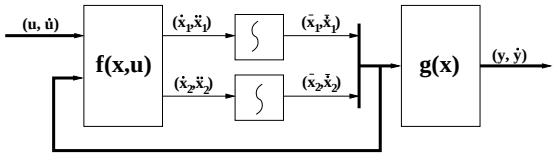

Fig. 1. Structure of a QSS2 simulation. Thick lines represent vector signals. This figure represents a system with two states

The following describes how the integrator blocks and function blocks work, respectively.

2.1 QSS2 Integrators

The integrators maintain a first order and second order model of the state trajectory as a function of time since the last update, $\tau = t - t_k$.

$$\bar{x}_i(\tau) = \bar{x}_i(t_k) + \bar{\dot{x}}_i \tau \tag{3}$$

$$x_i(\tau) = x_i(t_k) + \dot{x}_i \tau + \frac{1}{2}\ddot{x}_i \tau^2 \tag{4}$$

where $\bar{x}_i \in \mathbb{R}$ and $x_i \in \mathbb{R}$ are the i'th states in the first- and second order models, respectively.

The second model is updated whenever an external event occurs, i.e. new information from the block representing $\mathbf{f}(\mathbf{x}, \mathbf{u})$, while the first model $\bar{x}_i(\tau)$, is updated when the difference between the two models exceeds a preset *quantum*, i.e. when:

$$|\bar{x}_i(\tau) - x_i(\tau)| > \Delta Q \tag{5}$$

where ΔQ is the chosen quantum. At each event the next internal event time is calculated by solving for τ in eq. (5).

The above scheme is sketched in fig. 2, which provides an example trajectory. Here, it can be seen how the models for $\bar{x}_i(\tau)$ and $x_i(\tau)$ are allowed to evolve independently whereafter upon reaching the difference ΔQ are reset to the same condition.

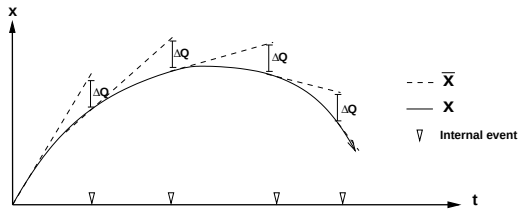

Fig. 2. A sample QSS2 trajectory

With this formulation one can think of $(\bar{\mathbf{x}}, \bar{\dot{\mathbf{x}}})$ as an operating point, or rather an *operating trajectory*, with the guarantee that it is correct to within the chosen quantum within the time interval until the next event.

2.2 State and Output Maps

Whenever an integrator produces a new operating trajectory, $(\bar{\mathbf{x}}, \bar{\dot{\mathbf{x}}})$, this is communicated to the blocks that are connected to the output of the integrator. For fig. 1 this means $\mathbf{f}(\mathbf{x}, \mathbf{u})$ and $\mathbf{g}(\mathbf{x})$. These function blocks then calculate new outputs which depend on the input variable [1].

In order to calculate the second derivative; the block numerically derives a matrix with second order information as in the first order Taylor expansion of $\mathbf{f}(\mathbf{z})$ with $\mathbf{z} = [\mathbf{x^T}\ \mathbf{u^T}]^\mathbf{T}$ (equivalently for $\mathbf{g}(\mathbf{x})$):

$$\mathbf{f}(\mathbf{z}(\tau)) = \mathbf{f}(\mathbf{z}(t_k)) + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{z}}(\mathbf{z}(t_k))\right] \dot{\mathbf{z}}\tau \tag{6}$$

Each time there is an integrator event or an input event the corresponding column in $\frac{\partial \mathbf{f}}{\partial \mathbf{z}}$ is updated.

2.3 QSS2 Properties

The properties of the QSS2 approach to ODE propagation have been established in Kofman (2002). Most importantly it is proven that the QSS2 simulation converges to a region around the equilibrium of the original continuous system, where the size of the region is a function of the selected quantum size. For practical applications, the quantum can be selected small enough to make the region small compared to system noise. For the application of QSS2 in this paper the benefits are:

- it is an efficient way to propagate a non-linear model on-line

[1] At initialisation, a static coupling analysis of the equation set is performed

- a byproduct of state propagation is a Jacobian matrix for the system that can be used for extended Kalman filtering.
- the QSS2 framework inherently supports events at arbitrary times as compared to a sample-based implementation

The QSS2 algorithm has been implemented in a Java-based library for execution of discrete event models, see (Alminde *et al.* 2006), which is based on the Discrete EVent Specification (DEVS) (Zeigler *et al.* 2000).

3. REVIEW OF EXTENDED KALMAN FILTERING

Kalman filtering and Extended Kalman Filtering (EKF) are some of the most widely used methods for estimation in linear and non-linear systems respectively. This section presents the classical EKF algorithm for non-linear systems, whereas the next sections describes modifications to the algorithm for use with quantised state systems.

3.1 Extended Kalman Filtering

The EKF algorithm propagates the state and associated covariance in intervals where no measurements are available. When a measurement is available the algorithm calculates a state update and reduced covariance matrix. Under the assumption that noise is Gaussian and has zero mean then the EKF is optimal in the sense that it minimises the expected prediction error. The starting point is a continuous non-linear model:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{w}(t) \quad \mathbf{w} \sim N(0, \mathbf{Q}(t)) \tag{7}$$

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}, t) + \mathbf{v}(t) \quad \mathbf{v} \sim N(0, \mathbf{R}(t)) \tag{8}$$

where $\mathbf{w}(t)$ and $\mathbf{v}(t)$ represent process noise and measurement noise respectively, which is assumed to follow a Gaussian distribution with zero mean and which is assumed to be independent of the process. For covariance propagation the EKF algorithm requires partial derivatives of the process and measurements:

$$\mathbf{F}(\mathbf{x}, t) = \frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} \quad \text{and} \quad \mathbf{G}(\mathbf{x},t) = \frac{\partial \mathbf{g}(\mathbf{x}, t)}{\partial \mathbf{x}} \tag{9}$$

A covariance matrix $\mathbf{P}$ is associated with the process, which is propagated in discrete time:

$$\mathbf{P}(k+1) = \mathbf{\Phi}(k)\mathbf{P}(k)\mathbf{\Phi}^T(k) + \mathbf{Q}(k) \tag{10}$$

where $\mathbf{\Phi}(k)$ is the state transition matrix associated with $\mathbf{F}(\hat{\mathbf{x}}, t)$ evaluated in the current state estimate, $\hat{\mathbf{x}}$. The state-estimate is propagated from the last sample point using the non-linear model, e.g. using the well-known Runge-Kutta algorithm. The discrete process noise term, $\mathbf{Q}(k)$, for the equation above, is obtained by integration:

$$\mathbf{Q}(k) = \mathbf{\Phi}(k)\mathbf{Q}(kTs)\mathbf{\Phi}^T(k) = \int_0^{Ts} \mathbf{F}(\hat{\mathbf{x}}, kTs+\tau)\mathbf{Q}(kTs+\tau)\mathbf{F}^T(\hat{\mathbf{x}}, kTs+\tau)d\tau \tag{11}$$

When a new measurement, $\mathbf{y}$, is available then the state is updated according to (where superscript "+" indicates the value after the update):

$$\hat{\mathbf{x}}^+(k) = \hat{\mathbf{x}(k)} + \mathbf{K}\left(\mathbf{y}(k) - \mathbf{g}(\hat{\mathbf{x}}, k)\right) \tag{12}$$

where $\mathbf{K}$ is the Kalman gain which is calculated according to:

$$\mathbf{K} = \mathbf{P}(k)\mathbf{G}(\hat{\mathbf{x}}, k)\left(\mathbf{G}(\hat{\mathbf{x}}, k)\mathbf{P}(k)\mathbf{G}^{\mathbf{T}}(\hat{\mathbf{x}}, k) + \mathbf{R}\right)^{-1} \tag{13}$$

after the state correction the covariance is updated to represent the increased knowledge of the state inferred from the measurement:

$$\mathbf{P}^+(k) = \left[\mathbf{I} - \mathbf{K}\mathbf{G}(\hat{\mathbf{x}}, k)\right]\mathbf{P}(k) \tag{14}$$

3.2 EKF Temporal Flow

Typical implementations of the EKF algorithm assumes a constant sample time with measurements arriving precisely at these sample times. The propagation of the state and the covariance in time is sketched on fig. 3.

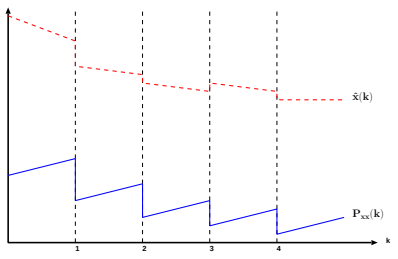

Fig. 3. Propagation of state (dashed) and covariance (solid) in the EKF algorithm. New measurements arrive at sample times.

As time progresses the covariance and state equations are propagated using eq. (7) and (10) respectively. At sample times new measurements are processed according to eqs. (12), (13), and (14), i.e. a state correction is calculated and applied and the covariance is updated to reflect the added state knowledge.

4. ADOPTION OF QSS2 BASED STATE/COVARIANCE PROPAGATION

The EKF functionality is included as an extra block in a QSS2 simulation, see fig. 4 compared to fig. 1, where it can be seen how this block is connected to the other blocks of the QSS simulation. The figure shows an example for a system with two states and one input.

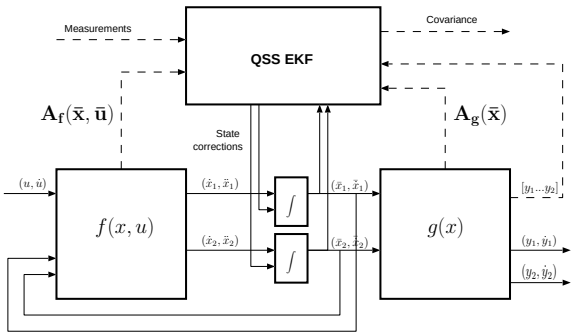

Fig. 4. Block diagram for a QSS based EKF implementation. Dashed lines are vector/matrix signals.

The central difference as compared to the sample-based implementation is that state/covariance propagation and measurement updates are not synchronised at equally spaced sample times. This is sketched on fig. 5, which should be compared to the sample based implementation, see fig. 3.

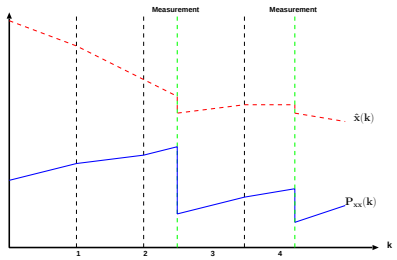

Fig. 5. Propagation of state (dashed) and covariance (solid) in the EKF algorithm. Green vertical lines are measurement events

The following subsections relate the figure to the EKF algorithm and explain how the EKF equations are handled in the QSS/DEVS framework.

4.1 State and Covariance Propagation

The state is propagated using the QSS2 simulation. This is fully analogous to typical implementation of the EKF where a forward Euler or Runge-Kutta approach is utilised for state propagation between discrete points in time.

Propagation of the covariance in the EKF requires that partial derivative with respect to the state can be found as indicated in eq. (9). This requires that an expression for $\mathbf{F}(\mathbf{x}, t)$ can be found analytically With the system model being propagated as a QSS2 model this information is already maintained and can be applied by letting:

$$\mathbf{F}(\mathbf{x}, t) = \mathbf{A_f}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \simeq \frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} \tag{15}$$

where $\mathbf{A_f}(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ is taken from the corresponding QSS2 simulation, see eq. 6. This allows the EKF to be easily applied to models where it is difficult or impossible to obtain analytical expressions. Eq. (10) can be calculated each time a new $\mathbf{A_f}(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ matrix is received in the EKF block.

4.2 Measurement Update

When a new measurement is available a measurement update is performed, this entails:

(1) The covariance is propagated to the current time, as described above
(2) The Kalman gain is calculated as in eq. (13)
(3) State correction is performed using eq. (12)
(4) The covariance is updated due to the measurement, i.e. eq. (14)

In order to calculate the Kalman gain the partial derivative of the measurement equations with respect to the state must be available; the same approach as used for covariance propagation is used here by letting:

$$\mathbf{G}(\mathbf{x},t) = \mathbf{A_g}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \simeq \frac{\partial \mathbf{g}(\mathbf{x}, t)}{\partial \mathbf{x}} \tag{16}$$

The state corrections are calculated by eq. (12) and are applied to the integrator blocks as a state reset event, i.e. the integrator adopts the value and solve eq. (5) for the next event-time or set the event-time to zero if the correction is large enough to make the difference exceed the criteria.

4.3 Implementation

Implementation of the quantised filter requires a QSS2 simulation to be setup, see fig. 1, with the process model as the driving function, $\mathbf{f}(\mathbf{x})$, and the sensor model as the output function, $\mathbf{g}(\mathbf{x})$. The added EKF functionality is achieved by three function calls; first the EKF block is constructed:

```
EKF ekf=new EKF(double cT ,int nM,
                Matrix P, Matrix Q,
                Qss2Static mMap);
```

where `cT` is a guaranteed minimum time between covariance propagation, `nM` is the number of associated measurements, `P` is the initial covariance matrix, `Q` is a matrix of continuous time process noise variances, and `mMap` is a reference to the

function, $\mathbf{g}(\mathbf{x})$. Hereafter references to integrator blocks in the model is supplied:

```
ekf.registerState(Qss2ResetIntegrator[]
                                    ints);
```

where `ints` are integrator references. Measurements are registered using the call for each:

```
ekf.registerMeasurement(int[] rows,
                        Matrix R);
```

where `ints` are row-indexes for $\mathbf{g}(\mathbf{x})$ for the corresponding measurement and `R` is an associated matrix of measurement noise variances.

The discussion of the function calls above serves to demonstrate how simple it is to add estimation to a QSS2 model; This allows a user to concentrate on the modelling part of the task rather than implementation of standard algorithms.

5. SIMULATION CASE DESCRIPTION

As a case study we will consider attitude and angular rate determination for a Deep Space Probe (DSP), which is equipped with two sensors that produce a unit vector in the body frame of the craft to the sun and a fixed bright star respectively. See fig. 6.

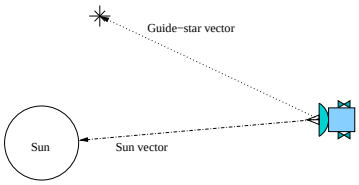

Fig. 6. Attitude information by measuring the direction to the sun and a bright guide star

5.1 Dynamical and Kinematical Model

The dynamical model has the following form:

$$\mathbf{J}\dot{\omega} = -[\omega\times]\mathbf{J}\omega + \mathbf{n}_{dist} \tag{17}$$

where $\omega = [\omega_1\ \omega_2\ \omega_3]^T$ are the body rates, $[\omega\times]$ is a matrix representing the gyro-scopic coupling:

$$[\omega\times] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{18}$$

and the parameter $\mathbf{J}$ is the inertia matrix:

$$\mathbf{J} = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{19}$$

The kinematical description of the DSP will be based on Euler angles, i.e.: (s for $\sin(\cdot)$ and c for $\cos(\cdot)$)

$$\dot{\theta} = \frac{1}{c\theta_2}\begin{bmatrix} c\theta_2 & s\theta_1 s\theta_2 & c\theta_1 s\theta_2 \\ 0 & c\theta_1 c\theta_2 & -s\theta_1 c\theta_2 \\ 0 & s\theta_1 & c\theta_1 \end{bmatrix}\omega \tag{20}$$

The disturbances, $\mathbf{n}_{dist}$, are assumed Gaussian distributed with standard deviation of $\sigma_m = 10^{-4}$.

5.2 Sensor Model

The DSP utilises a simple sun-sensor and star-sensor in cruise mode which are mounted on the x- panel of the spacecraft, see fig. 6. Noise for both sensors are assumed to be Gaussian distributed with zero mean and with the following variances on all axises with the stated update rate.

- Sun-sensor: $\sigma_{s_1} = 0.1^o$ @ 0.2Hz
- Star-sensor: $\sigma_{s_2} = 1^o$ @ 1Hz

A sensor model for both sensors can be described by the relation:

$$\mathbf{y} = \mathbf{C}_{321}(\theta)\cdot\left(\frac{\mathbf{X}_{DSP} - \mathbf{X}_{target}}{|\mathbf{X}_{DSP} - \mathbf{X}_{target}|}\right) + \mathbf{v} \tag{21}$$

where $\mathbf{C}_{321}(\theta)$ is the direction cosine matrix associated with a 3-2-1 Euler rotation sequence that rotates a vector from an assumed inertial frame to the body frame, $\mathbf{X}_{DSP}$ is the position of the probe and $\mathbf{X}_{DSP}$ is the position of the sun or guide star respectively. $\mathbf{v}$ is following a Gaussian distribution with standard deviations σ_{s_1} and σ_{s_2} respectively.

It should be noted that each sensor does not provide full observe-ability since, for vector measurements, the rotation around the sensor boresight cannot be inferred from a single measurement.

6. SIMULATION RESULTS

This section presents simulation results for the deep space probe. A truth-model was implemented in Simulink and the simulated measurements were processed by the QSS/EKF filter and then compared to the truth-model states.

The estimator has an initial attitude error of $\theta_{err} = [-0.1\ 0\ 0]rad$ and an initial angular rate error of $\omega_{err} = [0.002\ 0\ 0.005]rad/s$. The QSS2 implementation of the DSP model utilises quanta of $\Delta Q = 10^{-5}$ rad for the attitude states and $\Delta Q = 10^{-6}$ rad/s for angular velocity states.

Fig. 7 and 8 show the estimated values versus truth values for attitude and angular rates respectively. It is clear from the figure that the attitude estimates converge quickly to the truth-model and that the angular rates also converge albeit somewhat slower as expected due to poorer observe-ability.

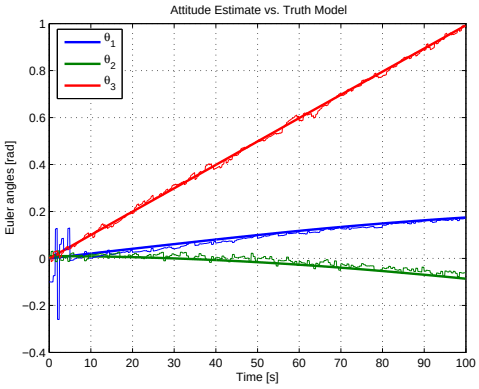

Fig. 7. Attitude estimation. Thick lines are from the truth model and thin lines are estimator outputs

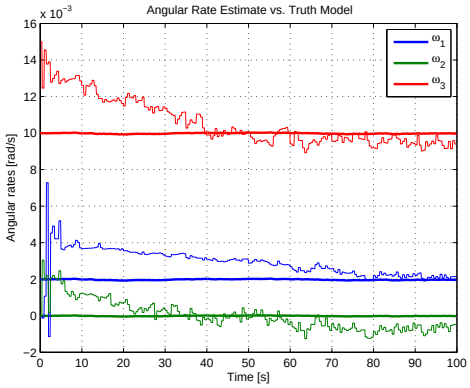

Fig. 8. Angular rate estimation. Thick lines are from the truth model and thin lines are estimator outputs

Fig. 9 and 10 show a comparison between 2-sigma bounds of the propagated covariance vs. the absolute of the estimation error for attitude and angular rates respectively.

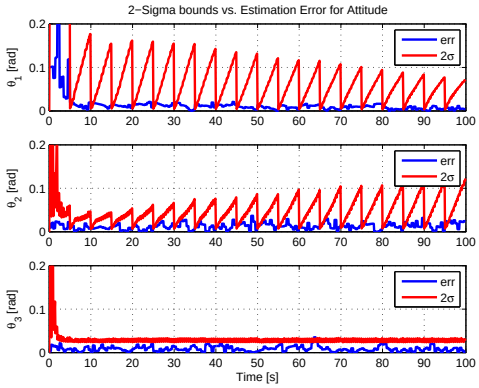

Fig. 9. Performance for attitude estimation. 2-sigma bounds vs absolute estimation error

It is evident that the filter provides estimates that are consistent with the predicted covariance of the filter. The estimation task executes in approximately 250 times real time on a contemporary

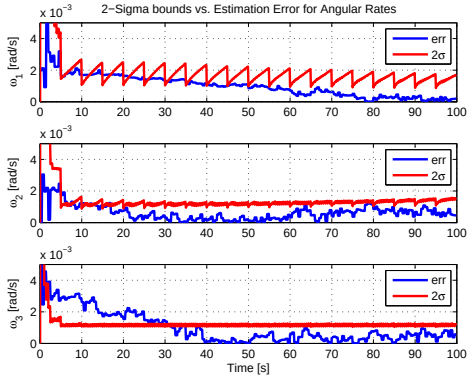

Fig. 10. Performance for angular rate estimation. 2-sigma bounds vs. absolute estimation error

computer. By varying the quanta it has been found that the quantum for each state should be chosen one magnitude lower than the expected covariance of that state in order to guarantee consistent results.

7. CONCLUSIONS

This paper has presented how quantised state simulation can be used for extended Kalman filtering and how it enables Jacobian free estimation for non-linear models. A case study of a deep space probe has demonstrated that the filter operates consistently and has demonstrated how easy it is to setup the filter in the QSS2/DEVS framework once a system model is available.

REFERENCES

Alminde, Lars, Jan D. Bendtsen and Jakob Stoustrup (2006). *A Quantized State Approach to On-line Simulation for Spacecraft Autonomy*. AIAA. In 2006 Modeling and Simulation Technologies Conference Proceedings. American Institute of Aeronautics and Astronautics, Keystone, Colorado, August 2006.

Alminde, Lars, Jan D. Bendtsen, Jakob Stoustrup and Kristin Y. Pettersen (2007). *Objective Directed Control using Local Minimisation for an Autonomous Underwater Vehicle*. IFAC. In proceedings of IAV2007, 3-5. September 2007, Toulouse, France.

Grewal, M. S. and A. P. Andrews (1993). *Kalman Filtering Theory and Practice*. Prentice Hall.

Kofman, Ernesto (2002). *A Second Order Approximation for DEVS Simulation of Continous Systems*. SIAM. Journal of Simulation, issue 78, p. 76-89.

Zeigler, Bernard P., Herbert Praehofer and Tag Gon Kim (2000). *Theory of Modelling and Simulation*. John Wiley and Sons. 2nd edition.