Sigma-Point Kalman Filters for Integrated Navigation

Rudolph van der Merwe and Eric A. Wan

Adaptive Systems Lab, OGI School of Science & Engineering, Oregon Health & Science University

BIOGRAPHY

Rudolph van der Merwe is a Senior Research Associate in the Department of Computer Science & Engineering, OGI School of Science & Engineering at OHSU. He holds a BS degree (1995) and a MS degree (1998) in Electrical Engineering from the University of Stellenbosch, South Africa. In 2004 he received a Ph.D. degree in Electrical and Computer Engineering from the OGI School of Science & Engineering at OHSU. His broad range of research interests include probabilistic inference & machine learning as well as neural, adaptive, and machine learning approaches to signal processing, pattern recognition, robotics and autonomous control.

Eric A. Wan is an Associate Professor in the Department of Computer Science and Engineering, OGI School of Science and Engineering at OHSU. He received his BS (1987), MS (1988), and Ph.D. (1994) in Electrical Engineering from Stanford University. His research is in algorithms and architectures for adaptive signal processing and machine learning. He has ongoing projects in autonomous unmanned aerial vehicles, estimation and probabilistic inference, integrated navigation systems, time series prediction and modeling, and speech enhancement. He holds several patents in adaptive signal processing and has authored over 50 technical papers in the field.

ABSTRACT

Core to integrated navigation systems is the concept of fusing noisy observations from GPS, Inertial Measurement Units (IMU), and other available sensors. The current industry standard and most widely used algorithm for this purpose is the *extended Kalman filter* (EKF) [6]. The EKF combines the sensor measurements with predictions coming from a model of vehicle motion (either dynamic or kinematic), in order to generate an estimate of the current navigational state (position, velocity, and attitude). This paper points out the inherent shortcomings in using the EKF and presents, as an alternative, a family of improved derivativeless nonlinear Kalman filters called *sigma-point Kalman filters* (SPKF). We demonstrate the improved state estimation performance of the SPKF by applying it to the problem of loosely coupled GPS/INS integration. A novel method to account for latency in the GPS updates is also developed for the SPKF (such latency compensation is typically inaccurate or not practical with the EKF). A UAV (rotor-craft) test platform is used to demonstrate the results. Performance metrics indicate an approximate 30% error reduction in both attitude and position estimates relative to the baseline EKF implementation.

INTRODUCTION

In a typical integrated GPS/INS system, an EKF combines rate-gyro and accelerometer data (from an IMU) with a kinematic or dynamic model of a vehicle movement. Other sensors such as barometric altimeter or magnetic compass may also be integrated. GPS position and velocity measurements are then used to correct INS errors using the same EKF. The navigational state of the vehicle to be estimated include position, attitude, velocities, as well as INS sensor biases. In addition (for loosely coupled implementations), the GPS receiver may employ its own EKF to solve position and velocity estimates (and timing) from satellite pseudorange, phase, and Doppler data. Alternatively, in a tightly coupled approach, a single EKF may be used to combine raw satellite signals with the IMU and other sensor measurements to make an optimal estimation of the vehicles navigational state.

A central and vital operation performed in all Kalman filters is the propagation of a Gaussian random variable (GRV) through the system dynamics. In the EKF, the system state distribution and all relevant noise densities are approximated by GRVs, which are then propagated analytically through a first-order linearization of the nonlinear system. This can introduce large errors in the true posterior mean and covariance of the transformed GRV, which may lead to sub-optimal performance and sometimes divergence of the filter. The SPKF addresses this problem by using a deterministic sampling approach. The state distribution is again approximated by a GRV, but is now represented using a minimal set of carefully chosen weighted sample points. These sample points completely capture the true mean and covariance of the GRV, and when propagated through the true nonlinear system, captures the posterior mean and covariance accurately to the 3rd order (Taylor series expansion) for any nonlinearity. The EKF, in contrast, only achieves first-order accuracy. Remarkably, the computational complexity of the SPKF is the same order as that of the EKF. Furthermore, implementation of the SPKF is often substantially easier and requires no analytic derivation or Jacobians as in the EKF. SPKF methods have proven to be far superior to standard EKF based estimation approaches in a wide range of applications in the areas of nonlinear state estimation, parameter estimation (system identification) as well as dual estimation (machine learning) [20, 10, 25]. In this paper, we apply the SPKF framework to the problem of nonlinear estimation and sensor fusion for the GPS/INS integration.

In the first part of the paper we review the general state-estimation framework employed by all Kalman filters, after which we highlight the basic assumptions and flaws with using the EKF. We then introduce and review the fundamental development of the SPKF family of algorithms. This presentation is based on the general sigma-point approach for the calculation of the posterior statistics of a random variables that undergoes a nonlinear transformation. The actual algorithmic specification of different SPKF variants such as the *unscented Kalman filter* (UKF) [9], *central difference Kalman filter* (CDKF) [14], and numerically efficient and stable square-root implementations [22, 23] are deferred to the appendices at the end of this paper.

In the second part of the paper, we then focus on the application of the SPKF to the integrated navigation problem. We specifically detail the development of a loosely coupled implementation for integrating GPS measurements with an IMU and altimeter within the context of autonomous UAV guidance, navigation and control. We report experimental results generated using both a high-fidelity UAV simulation system (for ground truth comparison) as well as on real flight data using a fully instrumented XCell-90 RC helicopter platform.

THE EKF AND ITS FLAWS

The general Kalman framework involves estimation of the state of a discrete-time nonlinear dynamic system,

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{v}_k) \quad (1)$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{n}_k) \quad (2)$$

where $\mathbf{x}_k$ represent the unobserved state of the system and $\mathbf{y}_k$ is the only observed signal. The *process* noise $\mathbf{v}_k$ drives the dynamic system, and the *observation* noise is given by $\mathbf{n}_k$. Note that we are not assuming additivity of the noise sources. The system dynamic model $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ are assumed known. In state-estimation, the EKF is the standard method of choice to achieve a recursive (approximate) maximum-likelihood estimation of the state $\mathbf{x}_k$. Given the noisy observation $\mathbf{y}_k$, the recursive estimation for $\mathbf{x}_k$ can be expressed in the following form [12]:

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_k^-\right) \quad (3)$$
$$\mathbf{P}_{\mathbf{x}_k} = \mathbf{P}_{\mathbf{x}_k}^- - \mathbf{K}_k \mathbf{P}_{\tilde{\mathbf{y}}_k} \mathbf{K}_k^T \quad (4)$$

where $\hat{\mathbf{x}}_k^-$ is the *optimal prediction* of the state at time k conditioned on all of the observed information up to and including time $k-1$, and $\hat{\mathbf{y}}_k^-$ is the *optimal prediction* of the observation at time k. $\mathbf{P}_{\mathbf{x}_k}^-$ is the covariance of $\hat{\mathbf{x}}_k^-$, and $\mathbf{P}_{\tilde{\mathbf{y}}_k}$ is the covariance of $\tilde{\mathbf{y}}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k^-$, termed the *innovation*. The optimal terms in this recursion are given by

$$\hat{\mathbf{x}}_k^- = E\left[\mathbf{f}\left(\hat{\mathbf{x}}_{k-1}, \mathbf{v}_{k-1}\right)\right] \quad (5)$$
$$\hat{\mathbf{y}}_k^- = E\left[\mathbf{h}\left(\hat{\mathbf{x}}_k^-, \mathbf{n}_k\right)\right] \quad (6)$$
$$\begin{aligned}\mathbf{K}_k &= \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}\mathbf{P}_{\tilde{\mathbf{y}}_k}^{-1} &(7)\\ &= E\left[(\mathbf{x}_k - \hat{\mathbf{x}}_k^-)(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)^T\right] \times \\ &\quad E\left[(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)(\mathbf{y}_k - \hat{\mathbf{y}}_k^-)^T\right]^{-1} &(8)\end{aligned}$$

where the optimal prediction $\hat{\mathbf{x}}_k^-$ corresponds to the expectation of a nonlinear function of the random variables $\hat{\mathbf{x}}_{k-1}$ and $\mathbf{v}_{k-1}$ (see Eq. (5)). Similar interpretation holds for the optimal prediction of the observation $\hat{\mathbf{y}}_k^-$ in Eq. (6). The optimal gain term $\mathbf{K}_k$ is expressed as a function of posterior covariance matrices in Eq. (7). Note these terms also require taking expectations of a nonlinear function of the prior state estimate RVs. This recursion provides the optimal minimum mean-squared error (MMSE) linear estimator of $\mathbf{x}_k$ assuming all relevant random variables in the system can be efficiently and consistently modeled by maintaining their first and second order moments, i.e., they can be accurately modeled as Gaussian random variables (GRVs). We need not assume linearity of the system model $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$.

The Kalman filter calculates the optimal quantities in Eqs. (5), (6) and (7) exactly in the linear case, and can be viewed as an efficient method for analytically propagating a GRV through linear system dynamics. For nonlinear models, however, the extended Kalman filter (EKF) approximates the optimal terms as:

$$\hat{\mathbf{x}}_k^- \approx \mathbf{f}\left(\hat{\mathbf{x}}_{k-1}, \bar{\mathbf{v}}\right) \quad (9)$$
$$\hat{\mathbf{y}}_k^- \approx \mathbf{h}\left(\hat{\mathbf{x}}_k^-, \bar{\mathbf{n}}\right) \quad (10)$$
$$\mathbf{K}_k \approx \hat{\mathbf{P}}_{\mathbf{x}_k\mathbf{y}_k}^{lin}\left(\hat{\mathbf{P}}_{\tilde{\mathbf{y}}_k}^{lin}\right)^{-1} \quad (11)$$

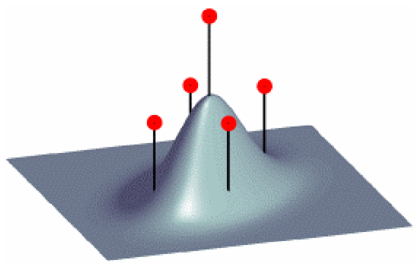

Figure 1 Weighted sigma-points for a 2 dimensional Gaussian random variable (RV). These sigma-points lie along the major eigen-axes of the RV's covariance matrix and complete captures the first and second order statistics of the RV. The height of each sigma-point indicates its relative weight.

where predictions are approximated as simply the function of the prior *mean* value for estimates (no expectation taken). The noise means are denoted by $\bar{\mathbf{v}}$ and $\bar{\mathbf{n}}$, and are usually assumed to equal to zero. Furthermore, the covariances $\hat{\mathbf{P}}^{lin}_{\mathbf{x}_k\mathbf{y}_k}$ and $\hat{\mathbf{P}}^{lin}_{\tilde{\mathbf{y}}_k}$ are determined by linearizing the system model, Eqs. (1) and (2), around the current estimate of the state and then determining (approximating) the posterior covariance matrices analytically for the linear system (see [12] for exact equations). This is equivalent to applying the linear Kalman filter covariance update equations to the first-order linearization of the nonlinear system. As such, the EKF can be viewed as providing "first-order" approximations to the optimal terms in Eq. (3). As mentioned earlier, these approximations used in the EKF can result in large errors in the estimates and even divergence of the filter.

THE SIGMA-POINT KALMAN FILTER

The sigma-point Kalman filter address the approximation issues of the EKF. This is achieved through a fundamentally different approach for calculating the posterior 1st and 2nd order statistics of a random variable that undergoes a nonlinear transformation. The state distribution is again represented by a GRV, but is now specified using a minimal set of deterministically chosen weighted sample points (See Fig. 1). These samples, called *sigma-points*, completely capture the true mean and covariance of the prior random variable, and when propagated through the *true* nonlinear system, captures the posterior mean and covariance accurately to the 2nd order (Taylor series expansion) for *any* nonlinearity (3rd order accuracy is achieved if the prior random variable has a symmetric distribution, such as the exponential family of pdfs.) The basic *sigma-point approach* (SPA) can be described as follows [9, 21]:

The Sigma-point Approach (SPA)

1. *A set of weighted samples (sigma-points) are deterministically calculated using the mean and square-root decomposition of the covariance matrix of the prior random variable. As a minimal requirement the sigma-point set must completely capture the first and second order moments of the prior random variable. Higher order moments can be captured, if so desired, at the cost of using more sigma-points.*

2. *The sigma-points are propagated through the true nonlinear function using functional evaluations alone, i.e., no analytical derivatives are used, in order to generate a posterior sigma-point set.*

3. *The posterior statistics are calculated (approximated) using tractable functions of the the propagated sigma-points and weights. Typically these take on the form of simple weighted sample mean and covariance calculations of the posterior sigma-points.*

To be more specific: Consider propagating a random variable $\mathbf{x} \in \mathbb{R}^L$ through an arbitrary nonlinear function $\mathbf{y} = \mathbf{g}(\mathbf{x})$. Assume $\mathbf{x}$ has mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P_x}$. To calculate the statistics of $\mathbf{y}$, we form a set of $2L+1$ sigma-points $\{\boldsymbol{\mathcal{X}}_i; {\scriptstyle i=0,\dots,2L}\}$ where $\boldsymbol{\mathcal{X}}_i \in \mathbb{R}^L$. The sigma-points are calculated using the following general selection scheme:

$$\begin{aligned} \boldsymbol{\mathcal{X}}_0 &= \bar{\mathbf{x}} \\ \boldsymbol{\mathcal{X}}_i &= \bar{\mathbf{x}} + \zeta\left(\sqrt{\mathbf{P_x}}\right)_i \qquad {\scriptstyle i=1,\dots,L} \\ \boldsymbol{\mathcal{X}}_i &= \bar{\mathbf{x}} - \zeta\left(\sqrt{\mathbf{P_x}}\right)_i \qquad {\scriptstyle i=L+1,\dots,2L} \end{aligned} \tag{12}$$

where ζ is a scalar scaling factor that determines the spread of the sigma-points around $\bar{\mathbf{x}}$ and $\left(\sqrt{\mathbf{P}}\right)_i$ indicates the ith column of the matrix square-root of the covariance matrix $\mathbf{P}$. Once the sigma-points are calculated from the prior statistics as shown above, they are propagated through the nonlinear function,

$$\boldsymbol{\mathcal{Y}}_i = \mathbf{g}\left(\boldsymbol{\mathcal{X}}_i\right) \quad {\scriptstyle i=0,\dots,2L} \tag{13}$$

and the mean and covariance of $\mathbf{y}$ are approximated using a weighted sample mean and covariance of the posterior sigma-points,

$$\bar{\mathbf{y}} \approx \sum_{i=0}^{2L} w_i^m \boldsymbol{\mathcal{Y}}_i \tag{14}$$

$$\mathbf{P_y} \approx \sum_{i=0}^{2L}\sum_{j=0}^{2L} w_{ij}^c \boldsymbol{\mathcal{Y}}_i \boldsymbol{\mathcal{Y}}_j^T \tag{15}$$

$$\mathbf{P_{xy}} \approx \sum_{i=0}^{2L}\sum_{j=0}^{2L} w_{ij}^c \boldsymbol{\mathcal{X}}_i \boldsymbol{\mathcal{Y}}_j^T\ , \tag{16}$$

where w_i^m and w_{ij}^c are scalar weights. Note, all weights need not be positive valued. In fact, depending on the specific sigma-point approach at hand, certain weights on the cross-terms are set equal to zero, i.e., $w_{ij} = 0$ for certain $\{i, j; i \neq j\}$. The specific values of the weights (w) and the scaling factors (ζ) depend on the *type* of sigma-point approach used: These include the *unscented transformation*

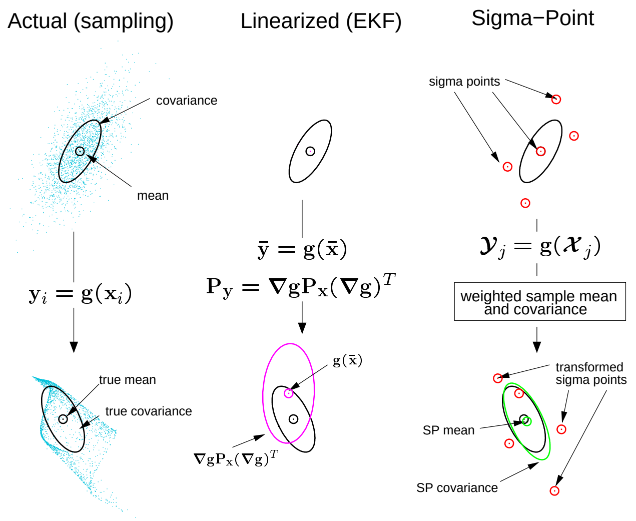
Actual (sampling)
Linearized (EKF)
Sigma−Point
covariance
mean
sigma points
$\mathbf{y}_i = \mathbf{g}(\mathbf{x}_i)$
$\bar{\mathbf{y}} = \mathbf{g}(\bar{\mathbf{x}})$
$\mathbf{P_y} = \nabla\mathbf{g}\mathbf{P_x}(\nabla\mathbf{g})^T$
$\boldsymbol{\mathcal{Y}}_j = \mathbf{g}(\boldsymbol{\mathcal{X}}_j)$
weighted sample mean and covariance
true mean
true covariance
$\mathbf{g}(\bar{\mathbf{x}})$
$\nabla\mathbf{g}\mathbf{P_x}(\nabla\mathbf{g})^T$
transformed sigma points
SP mean
SP covariance

Figure 2 2D example of sigma-point approach.

[9] and the Stirling-interpolation based *central difference transformation* [14] to name but two.

Note that the sigma-point approach differs substantially from general stochastic sampling techniques such as Monte-Carlo integration which require orders of magnitude more sample points in an attempt to propagate an accurate (possibly non-Gaussian) distribution of the state. The deceptively simple sigma-point approach results in posterior approximations that are accurate to the third order for Gaussian inputs for all nonlinearities. For non-Gaussian inputs, approximations are accurate to at least the second-order, with the accuracy of third and higher order moments determined by the specific choice of weights and scaling factors [21]. Furthermore, no analytical Jacobians of the system equations need to be calculated as is the case for the EKF. This makes the sigma-point approach very attractive for use in "black box" systems where analytical expressions of the system dynamics are either not available or not in a form which allows for easy linearization.

A simple comparative example of the sigma-point approach is shown in Figure 2 for a 2-dimensional system: the left plot shows the true mean and covariance propagation using Monte Carlo sampling; the center plots show the results using a linearization approach as would be done in the EKF; the right hand plots show the performance of the sigma-point approach (note, only 5 sigma-points are needed for the 2D case). The superior performance of the sigma-point approach is clearly evident.

Implementing the SPKF Algorithm

The sigma-point Kalman filter is a straightforward extension of the sigma-point approach to the recursive estimation in Eqs. (3)-(8), where the state RV is redefined as the concatenation of the original state and noise variables: $\mathbf{x}_k^a = [\ \mathbf{x}_k^T \ \ \mathbf{v}_k^T \ \ \mathbf{n}_k^T\ ]^T$. The sigma-point selection scheme (Equation 12) is applied to this new augmented state RV to calculate the corresponding sigma-point set, $\{\boldsymbol{\mathcal{X}}_{k,i}^a; i{=}0,\ldots,2L\}$ where $\boldsymbol{\mathcal{X}}_{k,i}^a \in \mathbb{R}^{L_\mathbf{x}+L_\mathbf{v}+L_\mathbf{n}}$. The pseudo-code for the SPKF is given below:

- *Initialization:*

$$\begin{aligned}
\hat{\mathbf{x}}_0 &= E[\mathbf{x}_0]\ \ ,\ \ \mathbf{P}_{\mathbf{x}_0} = E[(\mathbf{x}_0-\hat{\mathbf{x}}_0)(\mathbf{x}_0-\hat{\mathbf{x}}_0)^T] \\
\hat{\mathbf{x}}_0^a &= E\left[\mathbf{x}_0^a\right] = [\ \hat{\mathbf{x}}_0^T \ \ \hat{\mathbf{v}}_0^T \ \ \hat{\mathbf{n}}_0^T\ ]^T \\
\mathbf{P}_0^a &= E\left[(\mathbf{x}_0^a-\hat{\mathbf{x}}_0^a)(\mathbf{x}_0^a-\hat{\mathbf{x}}_0^a)^T\right] \\
&= \begin{bmatrix} \mathbf{P}_{\mathbf{x}_0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R_v} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R_n} \end{bmatrix}
\end{aligned}$$

- *For* $k = 1,\ldots,\infty$:

  1. Set $t = k-1$
  2. Calculate sigma-points:

$$\boldsymbol{\mathcal{X}}_t^a = \left[\ \hat{\mathbf{x}}_t^a \quad \hat{\mathbf{x}}_t^a + \zeta\sqrt{\mathbf{P}_t^a} \quad \hat{\mathbf{x}}_t^a - \zeta\sqrt{\mathbf{P}_t^a}\ \right]$$

  3. Time-update equations:

$$\begin{aligned}
\boldsymbol{\mathcal{X}}_{k|t}^x &= \mathbf{f}\left(\boldsymbol{\mathcal{X}}_t^x, \boldsymbol{\mathcal{X}}_t^v, \mathbf{u}_t\right) \\
\hat{\mathbf{x}}_k^- &= \sum_{i=0}^{2L} w_i^m \boldsymbol{\mathcal{X}}_{i,k|t}^x \\
\mathbf{P}_{\mathbf{x}_k}^- &= \sum_{i=0}^{2L}\sum_{j=0}^{2L} w_{ij}^c \left(\boldsymbol{\mathcal{X}}_{i,k|t}^x\right)\left(\boldsymbol{\mathcal{X}}_{j,k|t}^x\right)^T
\end{aligned}$$

  4. Measurement-update equations:

$$\begin{aligned}
\boldsymbol{\mathcal{Y}}_{k|t} &= \mathbf{h}\left(\boldsymbol{\mathcal{X}}_{k|t}^x, \boldsymbol{\mathcal{X}}_t^n\right) \\
\hat{\mathbf{y}}_k^- &= \sum_{i=0}^{2L} w_i^m \boldsymbol{\mathcal{Y}}_{i,k|t} \\
\mathbf{P}_{\tilde{\mathbf{y}}_k} &= \sum_{i=0}^{2L}\sum_{j=0}^{2L} w_{ij}^c \left(\boldsymbol{\mathcal{Y}}_{i,k|t}\right)\left(\boldsymbol{\mathcal{Y}}_{i,k|t}\right)^T \\
\mathbf{P}_{\mathbf{x}_k\mathbf{y}_k} &= \sum_{i=0}^{2L}\sum_{j=0}^{2L} w_{ij}^c \left(\boldsymbol{\mathcal{X}}_{i,k|t}^x\right)\left(\boldsymbol{\mathcal{Y}}_{i,k|t}\right)^T \\
\mathbf{K}_k &= \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}\mathbf{P}_{\tilde{\mathbf{y}}_k}^{-1} \\
\hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_k^-\right) \\
\mathbf{P}_{\mathbf{x}_k} &= \mathbf{P}_{\mathbf{x}_k}^- - \mathbf{K}_k\mathbf{P}_{\tilde{\mathbf{y}}_k}\mathbf{K}_k^T
\end{aligned}$$

- *Parameters:* $\mathbf{x}^a = [\ \mathbf{x}^T \ \ \mathbf{v}^T \ \ \mathbf{n}^T\ ]^T$, $\boldsymbol{\mathcal{X}}^a = [\ (\boldsymbol{\mathcal{X}}^x)^T \ \ (\boldsymbol{\mathcal{X}}^v)^T \ \ (\boldsymbol{\mathcal{X}}^n)^T\ ]^T$, ζ is scaling parameter that determines the spread of the sigma-points around the prior mean, L is the dimension of the augmented state, $\mathbf{R_v}$ is the process-noise covariance, $\mathbf{R_n}$ is the observation-noise covariance, and w_i^m and w_{ij}^c are the scalar weights.

The specific type of resulting SPKF is determined by the choice of sigma-point selection scheme (weights & scaling factors) as well as the specific method by which the propagated sigma-points are combined in order to calculate the posterior covariance matrices. In Appendix A we summarize two specific SPKF approaches, the *square-root unscented Kalman filter* (SRUKF) [23] and the *square-root central difference Kalman filter* (SRCDKF) [14]. The *square-root* implementations propagate (and update) directly the square-root of the state covariance matrix, thus avoiding the need to perform a direct matrix square-root operation at each time step. This provides increased computational efficiency as well as robust numerical stability. Other variations include efficient implementations when the noise is assumed additive (allowing fewer sigma-points to be used), or for special state-transition structures (as with pure parameter estimation) [7, 21]. Note that the overall computational complexity of the SPKF is the same as that of the EKF.

SPKF BASED GPS/INS INTEGRATION

We now describe the application of the SPKF to the problem of loosely coupled GPS/INS integration for guidance, navigation and control (GNC) of an unmanned aerial vehicle (UAV). Our UAV research platform (software simulator, hardware-in-the-loop simulator & flight vehicle) is based on a fully instrumented XCell-90 R/C helicopter (see Figure 3), originally designed by MIT's *Laboratory for Information and Decision Systems* [3]. The avionics package includes an Inertial Sciences ISIS MEMS based IMU, an Ashtech G12 10Hz GPS, a barometric altimeter and a DSP Design TP400 PC104 based flight computer running QNX-4. Our nonlinear control system (which requires state-estimates) is based on an efficient *state-dependent Ricatti-equation* (SDRE) framework that has proven to be significantly superior and more robust than standard LQR methods[2, 1].

The existing GPS/INS navigation filter was based on an MIT designed high-performance hand-tuned EKF implementation [4]. Our proposed estimator simply replaced the EKF in MIT's system with a SPKF based estimator (SRCDKF). All our experimental results in later sections will use the original EKF based navigation filter as a baseline reference. As a further extension, we also implemented a SPKF based sensor latency compensation technique. We compared our SPKF based system performance to the baseline system with specific focus on: 1) Improved six-degrees-of-freedom (6DOF) state estimation accuracy, 2) SPKF based compensation for GPS latency, 3) Evaluation of improved closed-loop control envelope.

We will next discuss the UAV specific system process and observation (measurement) models used inside our SPKF based system.

Process Model

Even though we used a high-fidelity (70 parameters, 43 states) nonlinear dynamic model of UAV movement [2] for

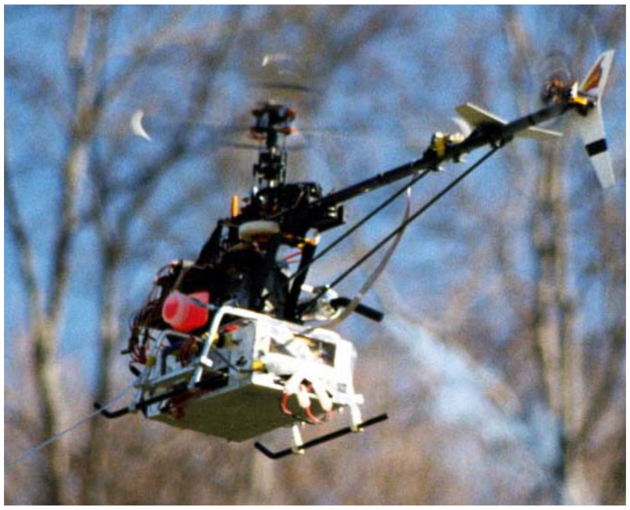

Figure 3 Instrumented X-Cell-90 helicopter in flight.

our UAV simulators and control system design, due to its high computational complexity it is not ideally suited for use within the navigation filter loop. For this reason we opted for the standard IMU driven kinematic process model formulation that comprises an INS mechanization component [16, 17] and a IMU sensor error model component. Because low cost MEMS based IMUs such as the one used in our avionics system have large bias and scale factor errors we included these components into our state vector to be estimated. The estimated values of these error components are then used to correct the raw IMU acceleration and gyro-rate measurements before they are used inside the INS mechanization equations of the process model. The 16 dimensional state vector of our system is given by,

$$\mathbf{x} = \begin{bmatrix} \mathbf{p}^T & \mathbf{v}^T & \mathbf{e}^T & \mathbf{a}_b^T & \boldsymbol{\omega}_b^T \end{bmatrix} \tag{17}$$

where $\mathbf{p} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ and $\mathbf{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$ are the position and velocity vectors of the vehicle in the navigation frame, $\mathbf{e} = \begin{bmatrix} e_0 & e_1 & e_2 & e_3 \end{bmatrix}^T$ is the unity norm vehicle attitude quaternion, $\mathbf{a}_b = \begin{bmatrix} a_{x_b} & a_{y_b} & a_{z_b} \end{bmatrix}^T$ is the vector of IMU acceleration biases, and $\boldsymbol{\omega}_b = \begin{bmatrix} p_b & q_b & r_b \end{bmatrix}^T$ is the IMU gyro rate bias vector. Note that we could have include a separate *scale* factor in addition to the *bias* term in the state vector. However, in our experiments, we found it sufficient to model the combined effect of the bias and scale error terms as a single *time-varying* bias term.

The continuous time *kinematic* navigation equations (INS mechanization equations and error model) operating on this state vector and driven by the error corrected IMU measurements are given below:

$$\begin{aligned}
\dot{\mathbf{p}} &= \mathbf{v} & (18)\\
\dot{\mathbf{v}} &= \mathbf{C}_b^n\left(\bar{\mathbf{a}} - \mathbf{a}_{\tilde{\mathbf{r}}_{imu}}\right) + \left[\begin{array}{ccc} 0 & 0 & 1 \end{array}\right]^T g & (19)\\
\dot{\mathbf{e}} &= -\frac{1}{2}\tilde{\mathbf{\Omega}}_{\bar{\omega}}\mathbf{e} & (20)\\
\dot{\mathbf{a}}_b &= \mathbf{w}_{\mathbf{a}_{\mathbf{b_k}}} & (21)\\
\dot{\boldsymbol{\omega}}_b &= \mathbf{w}_{\boldsymbol{\omega}_{\mathbf{b_k}}} & (22)
\end{aligned}$$

$\mathbf{C}_b^n$ is the direction cosine matrix (DCM) transforming vectors from the body frame to the navigation frame. The DCM is a nonlinear function of the current attitude quaternion and is given by

$$\mathbf{C}_b^n = \left(\mathbf{C}_n^b\right)^T$$
$$= 2\left[\begin{array}{ccc} 0.5 - e_2^2 - e_3^2 & e_1e_2 - e_0e_3 & e_1e_3 + e_0e_2 \\ e_1e_2 + e_0e_3 & 0.5 - e_1^2 - e_3^2 & e_2e_3 - e_0e_1 \\ e_1e_3 - e_0e_2 & e_2e_3 + e_0e_1 & 0.5 - e_1^2 - e_2^2 \end{array}\right] . \quad (23)$$

The term g is the gravitational acceleration component and $\bar{\mathbf{a}}$ and $\bar{\boldsymbol{\omega}}$ are the bias and noise corrected IMU accelerometer and gyro rate measurements, i.e.,

$$\begin{aligned}
\bar{\mathbf{a}} &= \tilde{\mathbf{a}} - \mathbf{a}_b - \mathbf{n}_{\mathbf{a}} & (24)\\
\bar{\boldsymbol{\omega}} &= \tilde{\boldsymbol{\omega}} - \boldsymbol{\omega}_b - \mathbf{C}_n^b\boldsymbol{\omega}_c - \mathbf{n}_{\boldsymbol{\omega}} \ . & (25)
\end{aligned}$$

In the above equations $\tilde{\mathbf{a}}$ and $\tilde{\boldsymbol{\omega}}$ are the raw measurements coming from the IMU, $\mathbf{n}_{\mathbf{a}}$ and $\mathbf{n}_{\boldsymbol{\omega}}$ are the IMU acceleration and gyro-rate measurement noise terms, and $\boldsymbol{\omega}_c$ is the rotational rate of the earth as measured in the navigation frame (Coriolis effect). In general, $\boldsymbol{\omega}_c$ is a function of the location of the navigational frame relative to the earth frame and hence is time-varying as the navigation frame moves relative to the earth frame. However, for our purposes (aggressive autonomous UAV flight within a very small airspace volume) we assumed the navigation frame does not change relative to the earth frame resulting in a constant $\boldsymbol{\omega}_c$ for a given origin location (lat/long) of our navigation frame. $\tilde{\mathbf{\Omega}}_{\bar{\omega}}$ is a 4×4 skew-symmetric matrix [19] composed of the error corrected IMU gyro-rate measurements, i.e.,

$$\tilde{\mathbf{\Omega}}_{\bar{\omega}} = \left[\begin{array}{cccc} 0 & \bar{\omega}_p & \bar{\omega}_q & \bar{\omega}_r \\ -\bar{\omega}_p & 0 & -\bar{\omega}_r & \bar{\omega}_q \\ -\bar{\omega}_q & \bar{\omega}_r & 0 & -\bar{\omega}_p \\ -\bar{\omega}_r & -\bar{\omega}_q & \bar{\omega}_p & 0 \end{array}\right] . \quad (26)$$

In Eq. (19), $\mathbf{a}_{\tilde{\mathbf{r}}_{imu}}$ is the IMU-lever-arm coupling component due to the IMU not being located at the center of gravity of the vehicle. This component can be ignored if the navigation filter computes the state estimate at the IMU location. This IMU centric navigation solution can then simply be transformed to the center of gravity location after the fact as needed by the vehicle control system.

The final components of the process model, Eqs. (21) and (22) models the time-varying nature of the IMU sensor bias error terms. Usually, sensor error in an INS are modelled as a zero-mean, stationary, first-order Gauss-Markov process [13]. Since the biases and scale factors of low cost MEMS based IMU sensors exhibit non-zero mean and non-stationary behaviour, the errors are modelled as a *random walk*, in order to improve the tracking of these time-varying errors by the navigation filter. This does however require that the effect of these errors be *observable* through the specific choice of measurement model.

The position and velocity discrete-time updates are calculated by the following simple first-order Euler update

$$\begin{aligned}
\mathbf{p}_{k+1} &= \mathbf{p}_k + \dot{\mathbf{p}}_k \cdot dt & (27)\\
\mathbf{v}_{k+1} &= \mathbf{v}_k + \dot{\mathbf{v}}_k \cdot dt \ , & (28)
\end{aligned}$$

where $\dot{\mathbf{p}}_k$ and $\dot{\mathbf{v}}_k$ are calculated using Eqs. (18) and (19) and dt is the integration time-step of the system (in our system this was dictated by the IMU rate, i.e., $dt = 10ms$). The quaternion propagation equation can be discretized with an analytical calculation of the exponent of the skew-symmetric matrix given by Stevens [19]. The discrete-time update can be written as

$$\mathbf{e}_{k+1} = \exp\left(-\frac{1}{2}\tilde{\mathbf{\Omega}} \cdot dt\right)\mathbf{e}_k \ . \quad (29)$$

If we further denote

$$\begin{aligned}
\Delta\phi &= \bar{\omega}_p \cdot dt & (30)\\
\Delta\theta &= \bar{\omega}_q \cdot dt & (31)\\
\Delta\psi &= \bar{\omega}_r \cdot dt \ , & (32)
\end{aligned}$$

as effective rotations around the (body frame) roll, pitch and yaw axes undergone by the vehicle during the time period dt, assuming that the angular rates $\bar{\omega}_p, \bar{\omega}_q$ and $\bar{\omega}_r$ remained constant during that interval, we can introduce the 4×4 skew-symmetric matrix

$$\begin{aligned}
\mathbf{\Phi}_\Delta &= \tilde{\mathbf{\Omega}} \cdot dt \\
&= \left[\begin{array}{cccc} 0 & \Delta\phi & \Delta\theta & \Delta\psi \\ -\Delta\phi & 0 & -\Delta\psi & \Delta\theta \\ -\Delta\theta & \Delta\psi & 0 & -\Delta\phi \\ -\Delta\psi & -\Delta\theta & \Delta\phi & 0 \end{array}\right] . & (33)
\end{aligned}$$

Using the definition of the matrix exponent and the skew symmetric property of $\mathbf{\Phi}_\Delta$, we can write down the following closed-form solution:

$$\exp\left(-\frac{1}{2}\mathbf{\Phi}_\Delta\right) = \mathbf{I}\cos(s) - \frac{1}{2}\mathbf{\Phi}_\Delta\frac{\sin(s)}{s} \ , \quad (34)$$

where

$$\begin{aligned}
s &= \frac{1}{2}\left\|\left[\begin{array}{ccc} \Delta\phi & \Delta\theta & \Delta\psi \end{array}\right]\right\| \\
&= \frac{1}{2}\sqrt{(\Delta\phi)^2 + (\Delta\theta)^2 + (\Delta\psi)^2} \ . & (35)
\end{aligned}$$

See [21] for a proof of this closed-form solution. Eqs. (29) and (34) ensure (at least theoretically) that the updated quaternion $\mathbf{e}_{k+1}$ has a unit norm. However, a small Lagrange multiplier term can be added to the first component of Equation 34 to further maintain numerical stability and the unity norm of the resulting quaternion. The resulting final solution for the time-update of the quaternion vector is given by

$$\mathbf{e}_{k+1} = \left[\mathbf{I}\left(\cos(s) + \eta \cdot dt \cdot \lambda\right) - \frac{1}{2}\mathbf{\Phi}_{\Delta}\frac{\sin(s)}{s}\right]\mathbf{e}_k \ . \tag{36}$$

where $\lambda = 1 - \|\mathbf{e_k}\|^2$ is the deviation of the square of the quaternion norm from unity due to numerical integration errors, and η is the factor that determines the convergence speed of the numerical error. These factors serve the role of the above mentioned Lagrange multiplier that ensures that the norm of the quaternion remains close to unity [15]. The constraint on the speed of convergence for stability of the numerical solution is $\eta \cdot dt < 1$ [4].

Finally, the discrete time random-walk process for the IMU sensor error terms are given by

$$\mathbf{a}_{b_{k+1}} = \mathbf{a}_{b_k} + dt \cdot \mathbf{w}_{\mathbf{a}_{b_k}} \tag{37}$$
$$\boldsymbol{\omega}_{b_{k+1}} = \boldsymbol{\omega}_{b_k} + dt \cdot \mathbf{w}_{\boldsymbol{\omega}_{b_k}} \ , \tag{38}$$

where $\mathbf{w}_{\mathbf{a}_{b_k}}$ and $\mathbf{w}_{\boldsymbol{\omega}_{b_k}}$ are zero-mean Gaussian random variables.

Note that these navigation equations are considered a *direct* formulation, as opposed to the alternative *indirect* (error) formulation. This choice was made for consistency with the MIT EKF implementation. The trade-offs between direct versus indirect formulations with the SPKF are currently being investigated.

Observation Models

Our system made use of 2 independent avionic sensors to aid the INS: a 10Hz, 50ms latency GPS (Ashtech G12) and a barometric altimeter that measures absolute altitude as a function of ambient air pressure. The observation models used in our system for these sensors (see below) are highly nonlinear, making the use of the SPKF framework again preferable to an EKF solution.

GPS: Since our GPS antenna is not located at the same location in the body frame as the IMU, it not only observes the bodies position and velocity in the navigation frame, but also the body's attitude relative to the navigation frame due to the "lever-arm effect". More specifically, the GPS observation model is given by:

$$\mathbf{p}_k^{GPS} = \mathbf{p}_{k-N} + \mathbf{C}_b^n \tilde{\mathbf{r}}_{gps} + \mathbf{n}_{p_k} \tag{39}$$
$$\mathbf{v}_k^{GPS} = \mathbf{v}_{k-N} + \mathbf{C}_b^n \boldsymbol{\omega}_{k-N} \times \tilde{\mathbf{r}}_{gps} + \mathbf{n}_{v_k} \ , \tag{40}$$

where $\mathbf{p}_{k-N}$ and $\mathbf{v}_{k-N}$ are the time-delayed (by N samples due to sensor latency) 3D navigation-frame position and velocity vectors of the vehicle, $\tilde{\mathbf{r}}_{gps}$ is the location of the GPS antenna in the body frame (relative to the IMU location), $\boldsymbol{\omega}_{k-N}$ are the true rotational rates of the vehicle at time $k-N$, and $\mathbf{n}_{p_k}$ and $\mathbf{n}_{v_k}$ are stochastic measurement noise terms. Here the noise terms are modeled as being time-dependent. This is due to the fact that the accuracy of observations vary over time according to the current PDOP value of the loosely coupled GPS solution. Since the DCM, $\mathbf{C}_b^n$, in Eqs. (39) and (40) are a function of the attitude quaternion, the GPS measurements provides information not only of the vehicles position and velocity, but also of its attitude. This removes the need for an absolute attitude sensor such as a magnetic compass or tilt-sensor. However, this will also result in the non-observability of the IMU sensor errors during prolonged periods of GPS outages, which in turn can lead to significant INS drift.

The time delay (N samples) in the GPS model equations is due to the internal GPS processing latency inherent to all loosely coupled GPS solutions. This implies that the latest GPS measurement relates to the state of the vehicle as it was a number of samples in the past. If the specific latency of the GPS is small, it can (and often is) ignored. However, if the latency is significant, care must be taken when fusing this lagged information with the current estimate of the vehicle's state in the measurement update step of the Kalman filter.

Barometric altimeter: Ambient air pressure provides an accurate source of sea-level altitude information. Important sources of error are sensor quantization and measurement noise. We used a high-end altimeter with $10^{-3}psi$ (0.6 meters) resolution. The measurement noise was assumed to be zero-mean, white and Gaussian. The observation model that incorporates these effects are:

$$z_k^{alt} = -\frac{1}{\varphi}\ln\left[\frac{\rho_0^q \left\lfloor\left(\rho_0 \exp\left(\varphi \cdot z_k\right) + n_{z_a}\right)/\rho_0^q\right\rfloor}{\rho_0}\right] \tag{41}$$

where ρ_0 is the nominal air pressure at sea-level, φ is the pressure decay rate with altitude constant ($1.16603 \times 10^{-4} psi/m$), z_k is the current navigation-frame z-axis position of the vehicle, ρ_0^q is the air pressure quantization resolution of the altimeter ($10^{-3}psi$), z_k^{alt} is the altimeter output and $\lfloor\cdot\rfloor$ is the integer flooring function. This model is not only a nonlinear function of the state, but the measurement noise also effects the output altitude measurement in a non-additive fashion. Again, for such a model the use of the SPKF not only allows for a much simpler implementation than the EKF (no analytical derivatives need to be calculated), but will also results in more accurate estimation results.

SPKF Based Sensor Latency Compensation

As mentioned in the previous section, when fusing latency delayed measurements with the current best prediction of the vehicle's state, care must be taken to incorporate this information in a mathematically correct fashion. Previous approaches to deal with this problem either store all of the state estimates and observations during the latency period and then *re-run* the complete filter when the latency lagged observation finally arrives, or apply accumulated correction terms to the state estimate based on a time-convolved linearized approximation of the system [11]. The first approach, although accurate incurs an exceedingly high computational penalty, precluding its use in real-time systems. The second approach on the other hand can be highly inaccurate if the system process and measurement equations are significantly nonlinear.

For our SPKF based navigation filter, we derived a new approach to deal with the latency issue based on accurately maintaining the relevant *cross-covariance matrices across time.* These terms are needed to formulate a modified Kalman gain matrix, which is used to fuse the current prediction of the state with an observation related to a prior (lagged) state of the system. The system process model is first augmented such that a copy of the prior system state is maintained across time. The observation model is also adapted to relate the current GPS observation to this *lagged* (but-maintained) state. The correct gain terms are then automatically calculated inside the SPKF filter. The SPKF allows for such a simple solution due to the fact that it does not need to linearize the system equations when calculating the relevant posterior statistics. For a more detailed exposition of this method, see Appendix B and [21, 24].

EXPERIMENTAL RESULTS

This section presents a number of experimental results comparing our proposed SPKF based GPS/INS system with an similar system built around an EKF implementation. The first set of experiments were all performed in simulation using the high-fidelity MIT-Draper-XCell-90 model based UAV simulator platform [4]. All relevant avionic sensors (IMU, GPS, altimeter, etc.) as well as all actuators were accurately modeled, including effects such as GPS latency and IMU sensor bias errors and drift. The purpose of the simulation based experiments were both to compare the performance of our new proposed SPKF approaches to that of the existing EKF approach in a controlled (repeatable) environment where the *ground truth* state information is available. This allows for objective comparison of estimation accuracy.

The second set of experiments were performed on real flight data using telemetry recordings of actual autonomous flights performed by the UAV. Although ground truth information is not available for these experiments to judge absolute accurate, it still allows for direct qualitative comparison between the EKF and SPKF based systems. Specific performance issues related to real world events such as GPS outages were investigated.

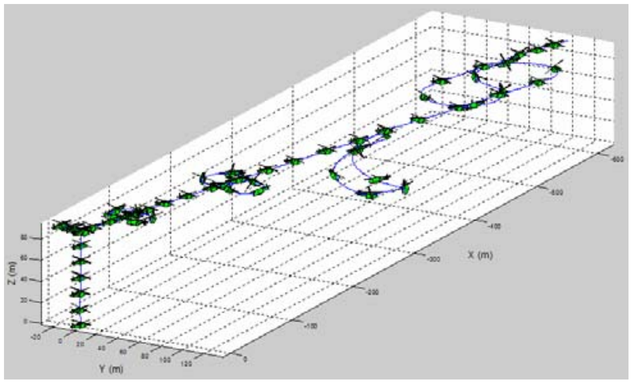

Figure 4 Simulated UAV trajectory used for state estimation experiments.

Simulation experiments

The first simulated experiment performed was used to provide quantitative comparisons between the EKF, SPKF, and latency compensated SPKF. The helicopter was flown (in simulation) along a complex trajectory that increased in “aggressiveness” over time. Figure 4 shows a 3D representation of this flight-plan trajectory with the helicopter’s true attitude superimposed at certain intervals. The simulated flight included complex acrobatic maneuvers such as *rapid-rise-and-hover, figure-eights, split-s,* etc. For this experiment we did not “close the loop” for the flight control system. In other words, the control system used the true known states of vehicle for the online calculation of the control law. The SPKF or EKF estimated state was not fed back to the control system. This was done to ensure that the helicopter flew exactly the same flight profile when comparing the performance of the different estimators. Again, the repeatability and access to the ground truth is what makes the high-fidelity simulation environment so attractive for these initial investigations.

Table 1 compares the average root-mean-square (RMS) estimation errors for the three different state estimators. We also show (in brackets) the relative error reduction percentage for each of the two SPKF estimators compared to the EKF. The normal SPKF is able to reduce the 3D position and velocity estimation errors by about 10% and the roll and pitch angle estimation errors by about 20%.

The biggest improvement over the EKF, 55%, is in the estimation of the yaw (heading) angle. The GPS latency compensated SPKF goes even further with a 33% reduction in position, velocity, roll angle and pitch angle errors. The yaw angle error reduction is again the highest at 65%. We repeated this experiment numerous times with different initializations and realizations of measurement noise as well as flying different flight trajectories and all of the results consistently confirmed the same relative performance between the different estimators as presented in this experiment. Clearly, even though the normal SPKF already out-

Table 1 UAV state estimation results : EKF vs. SPKF (with and without GPS latency compensation). The table reports average (over complete flight trajectory) root-mean-square (RMS) estimation errors for the EKF, SPKF (without GPS latency compensation) and SPKF (with GPS latency compensation) for the simulated flight shown in Figure 4. The estimation error reduction percentages are shown for all filters (relative to EKF).

Algorithm	**Average RMS Error**				
	position	velocity	Euler angles (degrees)		
	(m)	(m/s)	roll	pitch	yaw
EKF	2.1	0.57	0.25	0.32	2.29
SPKF (without latency compensation)	1.9 (10%)	0.52 (9%)	0.20 (20%)	0.26 (19%)	1.03 (55%)
SPKF (with latency compensation)	1.4 (32%)	0.38 (34%)	0.17 (32%)	0.21 (34%)	0.80 (65%)

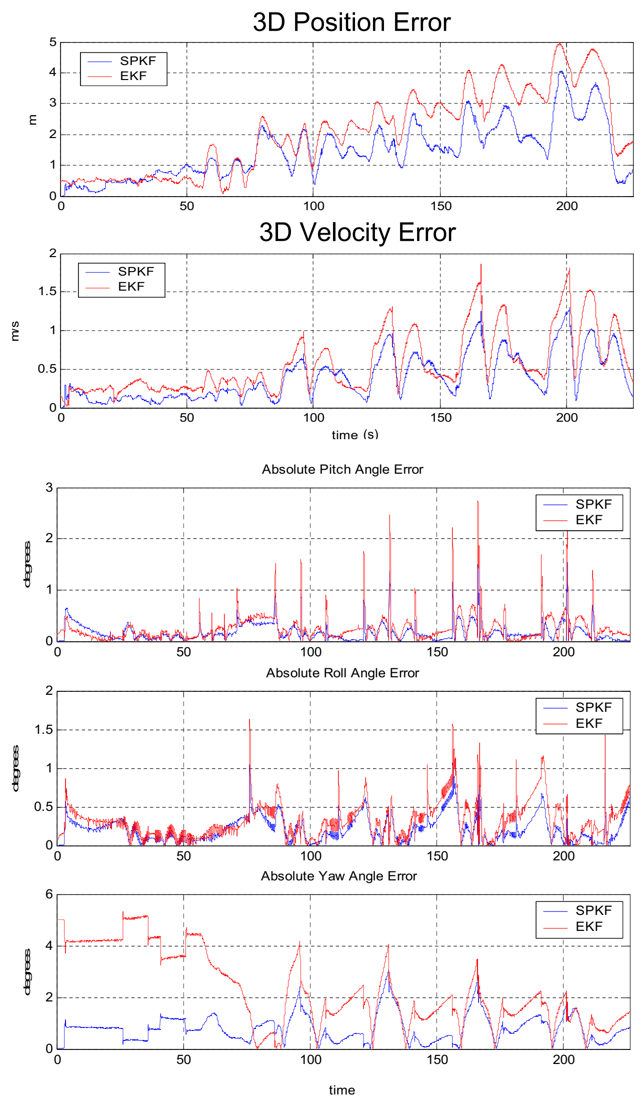

Figure 5 State estimation results - SPKF vs EKF (RMS error): [top] 3D position (32% error reduction). [2nd from top] 3D velocity (34% error reduction). [middle] pitch (34% error reduction). [2nd from bottom] roll (32% error reduction). [bottom] yaw (65% error reduction).

performs the EKF (as expected), correctly accounting for GPS latency is well worth the extra effort.

In order to clearly illustrate the difference in estimation performance between the EKF and the (latency compensated) SPKF we present the results of another run of the same simulation experiment, this time only showing the EKF and latency-compensated SPKF implementation plots. The position and velocity estimation errors are shown in the top two plots of Figure 5 and the Euler angle estimation errors are shown in the bottom three plots. As before the SPKF clearly outperforms the EKF with the largest improvement again evident in the yaw (heading) angle estimation error. Figure 5 indicates how the EKF has a very large error in the yaw estimate for the first 80 seconds of the flight. This is due to a significant initial error in the underlying IMU bias error estimates. Even though the EKF and SPKF filters were initialized with exactly the same initial state estimates, the SPKF was able to converge to the true biases in the IMU measurements much quicker and then track them more accurately. This result has been corroborated independently in [18] (experiments focused on in-flight IMU *alignment*). This contributes (among other things) to more accurate Euler angle estimates. Although the average yaw estimate error improvement for the SPKF over the whole trajectory is 65%, this value does not accurately reflect the expected steady-state (after bias convergence) performance of the SPKF. Discounting this period, the average error improvement after bias convergence ($t > 80$s) is 43%. The steady-state error improvement of the SPKF over the EKF is thus 32%, 34% and 43% respectively for the roll, pitch and yaw angle estimates.

Another interesting performance characteristic to note from the Euler angle estimates in Figure 5 are the frequent high peaks in the EKF's estimation error plots. These coincide with the onsets of aggressive maneuvers (banking, turns, rapid climbs, etc.) that pushes the vehicle into regimes of increased nonlinear response. The linearization errors of the EKF will therefore be more severe at these times resulting in poor estimation performance and increase estimation error. In contrast the SPKF is able to deal with these increased nonlinearities quite satisfactorily.

In the second set of simulated experiments we "closed the loop" in the GNC system by feeding the estimated states back to the SDRE control system. In other words, the vehicle control commands will now be a function of the estimates generated by either the EKF or SPKF estimator and not of the "true" vehicle states. This mimics (in simulation) the true interdependency between the estimation and control system as would occur in the real flight hardware during a fully autonomous flight. The helicopter is commanded to perform an aggressive high speed nose-in turn. This maneuver requires the helicopter to fly along an imaginary circular trajectory while constantly pointing its nose towards the exact center of the circle. Accurate position, velocity and especially yaw angle estimates are needed to follow the desired flight plan with the desired attitude. Figure 6 shows the results of this experiment for both the EKF and SPKF. The desired flight trajectory is indicated by the red curve, the true realized trajectory in blue and the estimated trajectory in green. The true attitude of the helicopter is indicated by periodic renderings of the vehicle itself along the flight path. Clearly for the SPKF case the estimated trajectory is not only close to the true trajectory (small estimation error), but the true trajectory is close to the *desired* trajectory which indicated good control performance. The EKF plots clearly shows worse performance according to both these criteria. Also evident from the plots is the much improved yaw angle tracking performance of the SPKF system compared to the EKF system. The helicopter renderings for the EKF indicate that the nose is not consistently pointing at the true center of the desired circle. The SPKF system, on the other hand, does much better in estimating and realizing the correct yaw attitude for this maneuver. EKF.

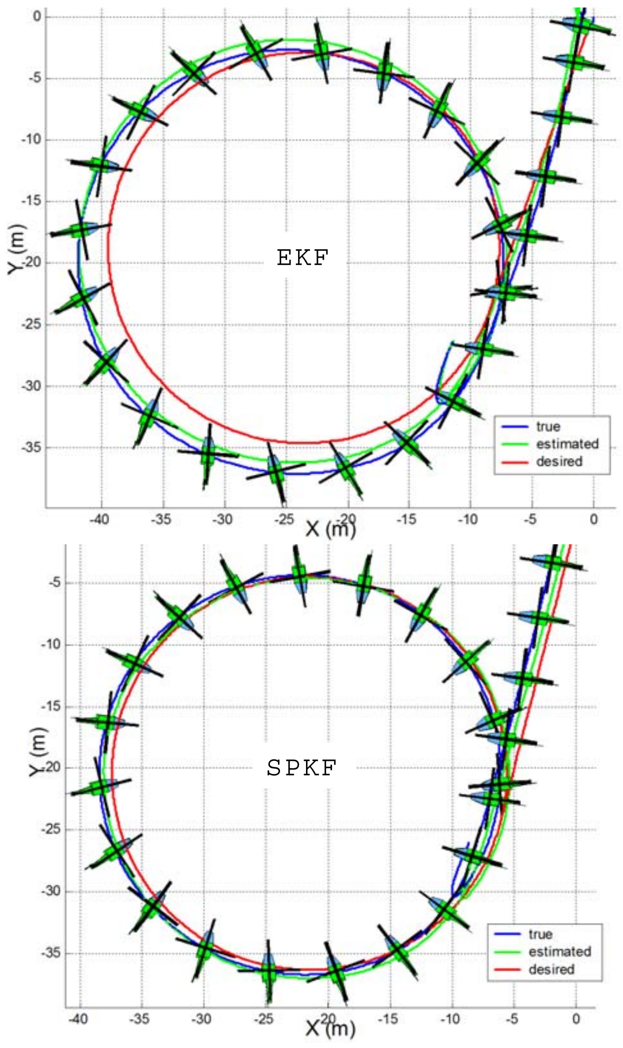

Figure 6 Closed-loop control performance comparison.

Real flight data experiments

Figure 7 shows the estimation results of the SPKF compared to the EKF based system on real flight telemetry. The UAV was flown under pilot guidance to a specified altitude at which point the system was switched over to fully autonomous flight. The autonomous flight plan was as follows: First the UAV held steady in hover for a number of seconds, after which it flew a square trajectory at a constant altitude of about 55-60 meters. Since no ground truth signal is available for absolute error comparison, we need to evaluate the results on more subjective terms. For this purpose, a top-down (2D) projection of the estimation results is quite insightful (see Figure 8).

Notice the significant number of GPS outages that occurred during the pilot guided ascent to the hovering altitude (s-shaped curve). Clearly the SPKF appears to more accurately track the (assumed) true underlying trajectory during this outage period. The EKF generated position estimate exhibits an erratic jump just before the GPS measurements becomes available again (see Figure 8 at coordinates

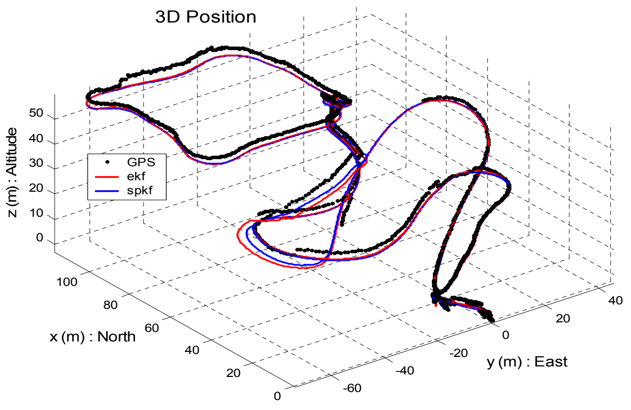

Figure 7 Estimated 3D position of test flight. The UAV lifted off and flew a complex sweeping S-maneuver until it reached its hover altitude at about 50m. At this point it hovered for a number of seconds after which it attempted to fly a horizontal square-profile. After the square was completed it hovered again for a number of seconds and then landed.

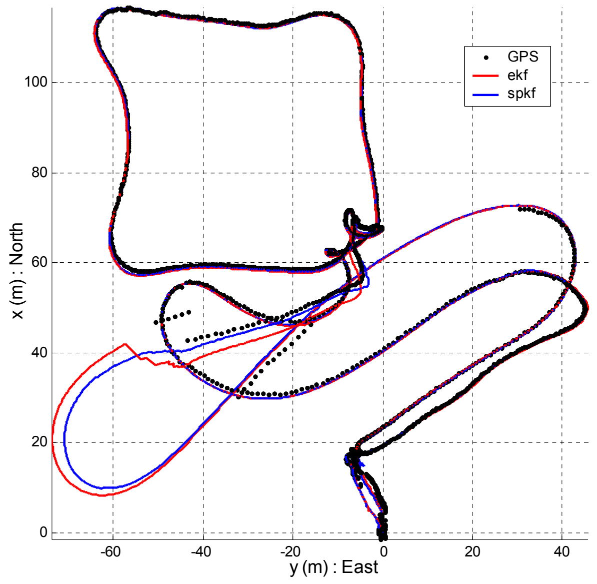

Figure 8 Estimated 2D position of test flight (top view)

$\{40, -60\}$). Figure 9 shows the effect of this on the separate north, east and down components of the estimated 3D position. This error is due to the inherent nature of the INS solution (derived from integrating the bias compensated IMU gyro and accelerometer data) to drift during periods of GPS outage. Since the SPKF performs a more accurate time-update during these periods than the EKF, and possibly also more accurately tracks the underlying IMU biases, the resulting SPKF estimates appear more robust to GPS outages in general. We are still investigating these claims further.

CONCLUSIONS

In this paper we presented a method for integrated navigation based on the sigma-point Kalman filter. The SPKF provides superior performance over the current industry standard, EKF, by better accounting for nonlinearities and accommodating asynchronous and lagged sensor measurements. The computational complexity of the SPKF is equivalent to the EKF. While performance comparisons were based on a specific UAV rotor craft platform, the general implementation of the navigation filter and SPKF approach makes it applicable to general integrated navigation systems, with performance gains expected independent of the vehicle or specific sensors used.

We continue to investigate trade-offs between direct and indirect navigation equations, alternative quaternion representations, ability to track scale and bias, as well as robustness to GPS outages. Additional extensions include a tightly-coupled integration approach and additional sensor augmentations. We are also investigating the utility of replacing the higher-end IMU in our INS system with a low-cost IMU (<$1000). Such IMUs typically have worse error performance (higher bias, scale-factor & drift), which can hopefully be compensated through the enhanced estimation performance of the SPKF based system.

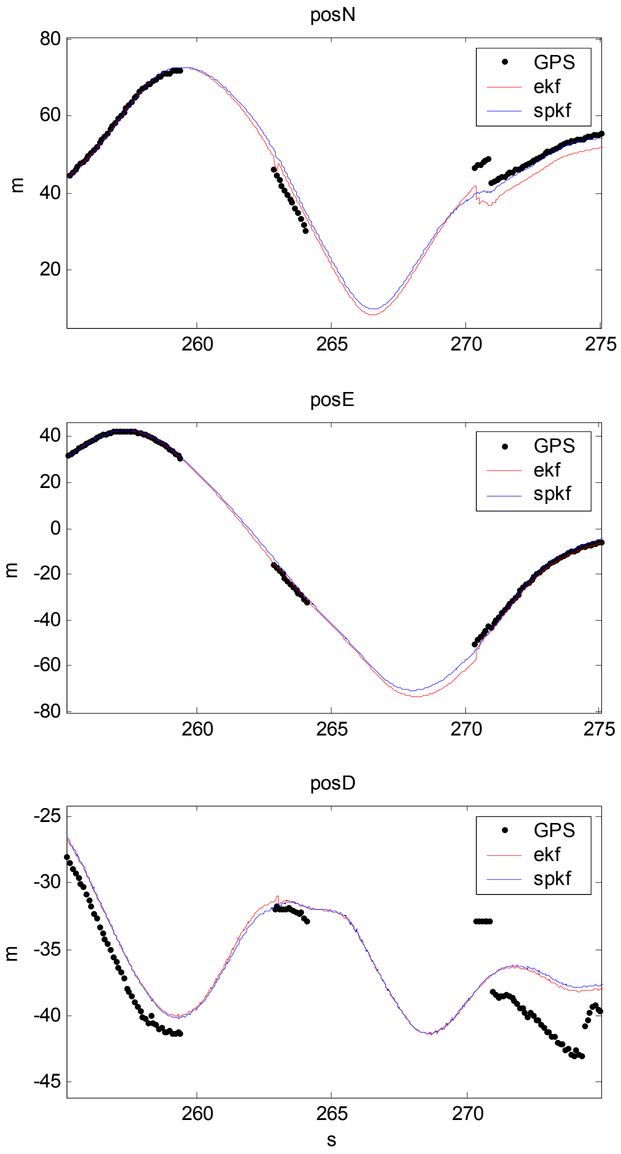

Figure 9 Estimated position (North, East and Down) during test flight (EKF vs. SPKF).

ACKNOWLEDGEMENTS

This work was supported in part by the following grants: NSF ECS-0083106 & ONR-FNC N0014-02-C-0248. The authors would like to thank Geoff Harvey and Dr. Alexander Bogdanov for assistance with the UAV platform and data collection.

APPENDIX A : SPKF VARIANT PSEUDO-CODE

This section provides the algorithmic pseudo-code for two different numerical robust and efficient *square-root* SPKF implementations [22, 23] . The first is based on the *unscented transformation* (a SPA scheme proposed by Julier & Uhlman[9]) and is called the *square-root unscented Kalman filters* (SR-UKF). The second SPKF is based on the *central-difference transformation* (a SPA scheme proposed separately by Norgaard et al. [14] and Ito et al.[8]) and is called the *square-root central difference Kalman filters* (SR-CDKF).

Square-Root UKF (SRUKF)

- *Initialization:*

$$\begin{aligned}\hat{\mathbf{x}}_0 &= E[\mathbf{x}_0]\ ,\ \mathbf{S}_{\mathbf{x}_0} = \sqrt{E[(\mathbf{x}_0-\hat{\mathbf{x}}_0)(\mathbf{x}_0-\hat{\mathbf{x}}_0)^T]}\\ \hat{\mathbf{x}}_0^a &= E[\mathbf{x}^a] = \begin{bmatrix}\hat{\mathbf{x}}_0 & \bar{\mathbf{v}} & \bar{\mathbf{n}}\end{bmatrix}^T\\ \mathbf{S}_0^a &= \sqrt{E[(\mathbf{x}_0^a-\hat{\mathbf{x}}_0^a)(\mathbf{x}_0^a-\hat{\mathbf{x}}_0^a)^T]}\\ &= \begin{bmatrix}\mathbf{S}_{\mathbf{x}_0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{S}_\mathbf{v} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{S}_\mathbf{n}\end{bmatrix}\end{aligned}$$

- *For* $k = 1, \ldots, \infty$:

  1. Set $t = k - 1$
  2. Calculate sigma-points:

$$\boldsymbol{\mathcal{X}}_t^a = \begin{bmatrix}\hat{\mathbf{x}}_t^a & \hat{\mathbf{x}}_t^a + \gamma\mathbf{S}_{\mathbf{x}_t}^a & \hat{\mathbf{x}}_t^a - \gamma\mathbf{S}_{\mathbf{x}_t}^a\end{bmatrix}$$

  3. Time-update equations:

$$\begin{aligned}\boldsymbol{\mathcal{X}}_{k|t}^x &= \mathbf{f}(\boldsymbol{\mathcal{X}}_t^a, \boldsymbol{\mathcal{X}}_t^v, \mathbf{u}_t)\\ \hat{\mathbf{x}}_k^- &= \sum_{i=0}^{2L} w_i^m \boldsymbol{\mathcal{X}}_{i,k|t}^x\\ \mathbf{S}_{\mathbf{x}_k}^- &= \mathsf{qr}\left\{\left[\sqrt{w_1^c}\left(\boldsymbol{\mathcal{X}}_{1:2L,k|t}^x - \hat{\mathbf{x}}_k^-\right)\right]\right\}\\ \mathbf{S}_{\mathbf{x}_k}^- &= \mathsf{cholupdate}\left\{\mathbf{S}_{\mathbf{x}_k}^-, \boldsymbol{\mathcal{X}}_{0,k|t}^x - \hat{\mathbf{x}}_k^-, w_0^{(c)}\right\}\\ \boldsymbol{\mathcal{Y}}_{k|t} &= \mathbf{h}(\boldsymbol{\mathcal{X}}_{i,k|t}^x, \boldsymbol{\mathcal{X}}_t^n)\\ \hat{\mathbf{y}}_k^- &= \sum_{i=0}^{2L} w_i^m \boldsymbol{\mathcal{Y}}_{i,k|t}\end{aligned}$$

  4. Measurement-update equations:

$$\begin{aligned}\mathbf{S}_{\tilde{\mathbf{y}}_k} &= \mathsf{qr}\left\{\left[\sqrt{w_1^c}\left(\boldsymbol{\mathcal{Y}}_{1:2L,k|t} - \hat{\mathbf{y}}_k^-\right)\right]\right\}\\ \mathbf{S}_{\tilde{\mathbf{y}}_k} &= \mathsf{cholupdate}\left\{\mathbf{S}_{\tilde{\mathbf{y}}_k}, \boldsymbol{\mathcal{Y}}_{0,k|t} - \hat{\mathbf{y}}_k^-, w_0^{(c)}\right\}\\ \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k} &= \sum_{i=0}^{2L} w_i^c\left(\boldsymbol{\mathcal{X}}_{i,k|t}^x - \hat{\mathbf{x}}_k^-\right)\left(\boldsymbol{\mathcal{Y}}_{i,k|t} - \hat{\mathbf{y}}_k^-\right)^T\\ \mathbf{K}_k &= \left(\mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}/\mathbf{S}_{\tilde{\mathbf{y}}_k}^T\right)/\mathbf{S}_{\tilde{\mathbf{y}}_k}\\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_k^-\right)\\ \mathbf{U} &= \mathbf{K}_k\mathbf{S}_{\tilde{\mathbf{y}}_k}\\ \mathbf{S}_{\mathbf{x}_k} &= \mathsf{cholupdate}\left\{\mathbf{S}_{\mathbf{x}_k}^-, \mathbf{U}, -1\right\}\end{aligned}$$

- *Weights & parameters:* $\gamma = \sqrt{L+\lambda}$, $w_0^m = \lambda/(L+\lambda)$, $w_0^c = w_0^m + (1-\alpha^2+\beta)$, $w_i^c = w_i^m = 1/[2(L+\lambda)]$ for $i = 1, \ldots, 2L$. $\lambda = \alpha^2(L+\kappa) - L$ is a compound scaling parameter, L is the dimension of the augmented state-vector, $0 < \alpha \le 1$ is the primary scaling factor determining the extent of the spread of the sigma-points around the prior mean. Typical range for α is $1e-3 < \alpha \le 1$. β is a secondary scaling factor used to emphasize the weighting on the zeroth sigma-point for the posterior covariance calculation. β can be used to minimize certain higher-order error terms based on known moments of the prior RV. For Gaussian priors, $\beta = 2$ is optimal. κ is a tertiary scaling factor and is usually set equal to 0. In general, the optimal values of these scaling parameters will be problem specific. For more detail on how to choose them, see [10].

- *General notes:* The augmented state vector and sigma-point vector is given by$\mathbf{x}^a = \begin{bmatrix}\mathbf{x}^T & \mathbf{v}^T & \mathbf{n}^T\end{bmatrix}^T$, $\boldsymbol{\mathcal{X}}^a = \begin{bmatrix}(\boldsymbol{\mathcal{X}}^x)^T & (\boldsymbol{\mathcal{X}}^v)^T & (\boldsymbol{\mathcal{X}}^n)^T\end{bmatrix}$. $\mathbf{S}_\mathbf{v} = \sqrt{\mathbf{R}_\mathbf{v}}$ and $\mathbf{S}_\mathbf{n} = \sqrt{\mathbf{R}_\mathbf{n}}$ where $\mathbf{R}_\mathbf{v}$ and $\mathbf{R}_\mathbf{n}$ are the process-noise and observation-noise covariance matrices.

- *Linear-algebra operators* (See [5, 21] for more detail): $\sqrt{\cdot}$: matrix square-root using lower triangular Cholesky decomposition. $\mathsf{qr}(\mathbf{A})$: lower-triangular part of $\mathbf{R}$ matrix resulting from economy QR decomposition of data-matrix $\mathbf{A}$. $\mathsf{cholupdate}\{\mathbf{R}, \mathbf{U}, \pm\nu\}$: N consecutive rank-1 Cholesky up(down)dates of the lower-triangular Cholesky factor $\mathbf{R}$ by the N columns of $\sqrt{\nu}\mathbf{U}$. $/$: Efficient least-squares pseudo inverse implemented using triangular QR decomposition with pivoting.

Square-Root CDKF (SRCDKF)

- *Initialization:*

$$\hat{\mathbf{x}}_0 = E[\mathbf{x}_0]\ ,\ \mathbf{S}_{\mathbf{x}_0} = \sqrt{E[(\mathbf{x}_0-\hat{\mathbf{x}}_0)(\mathbf{x}_0-\hat{\mathbf{x}}_0)^T]}$$

- *For* $k = 1, \ldots, \infty$:

  1. Set $t = k - 1$
  2. Calculate sigma points for time-update:

$$\begin{aligned}\hat{\mathbf{x}}_t^{av} &= \begin{bmatrix}\hat{\mathbf{x}}_t & \bar{\mathbf{v}}\end{bmatrix}\ ,\ \mathbf{S}_t^{av} = \begin{bmatrix}\mathbf{S}_{\mathbf{x}_t} & \mathbf{0}\\ \mathbf{0} & \mathbf{S}_\mathbf{v}\end{bmatrix}\\ \boldsymbol{\mathcal{X}}_t^{av} &= \begin{bmatrix}\hat{\mathbf{x}}_t^{av} & \hat{\mathbf{x}}_t^{av} + h\mathbf{S}_t^{av} & \hat{\mathbf{x}}_t^{av} - h\mathbf{S}_t^{av}\end{bmatrix}\end{aligned}$$

  3. Time-update equations:

$$\begin{aligned}\boldsymbol{\mathcal{X}}_{k|t}^x &= \mathbf{f}(\boldsymbol{\mathcal{X}}_t^x, \boldsymbol{\mathcal{X}}_t^v, \mathbf{u}_t)\\ \hat{\mathbf{x}}_k^- &= \sum_{i=0}^{2L} w_i^m \boldsymbol{\mathcal{X}}_{i,k|t}^x\\ \mathbf{S}_{\mathbf{x}_k}^- &= \mathsf{qr}\left\{\left[\sqrt{w_1^{c_1}}\left(\boldsymbol{\mathcal{X}}_{1:L,k|t}^x - \boldsymbol{\mathcal{X}}_{L+1:2L,k|t}^x\right)\right.\right.\\ &\qquad \left.\left.\sqrt{w_1^{c_2}}\left(\boldsymbol{\mathcal{X}}_{1:L,k|t}^x + \boldsymbol{\mathcal{X}}_{L+1:2L,k|t}^x - 2\boldsymbol{\mathcal{X}}_{0,k|t}^x\right)\right]\right\}\end{aligned}$$

4. Calculate sigma-points for measurement update:

$$\begin{aligned}\hat{\mathbf{x}}^{a_n}_{k|t} &= \begin{bmatrix} \hat{\mathbf{x}}^-_k & \bar{\mathbf{n}} \end{bmatrix} \quad , \quad \mathbf{S}^{a_n}_{k|t} = \begin{bmatrix} \mathbf{S}^-_{\mathbf{x}_k} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{\mathbf{n}} \end{bmatrix} \\ \boldsymbol{\mathcal{X}}^{a_n}_{k|t} &= \begin{bmatrix} \hat{\mathbf{x}}^{a_n}_{k|t} & \hat{\mathbf{x}}^{a_n}_{k|t} + h\mathbf{S}^{a_n}_{k|t} & \hat{\mathbf{x}}^{a_n}_{k|t} - h\mathbf{S}^{a_n}_{k|t} \end{bmatrix}\end{aligned}$$

5. Measurement-update equations:

$$\begin{aligned}\boldsymbol{\mathcal{Y}}_{k|t} &= \mathbf{h}\left(\boldsymbol{\mathcal{X}}^x_{k|t}, \boldsymbol{\mathcal{X}}^n_{k|t}\right) \\ \hat{\mathbf{y}}^-_k &= \sum_{i=0}^{2L} w^m_i \boldsymbol{\mathcal{Y}}_{i,k|t} \\ \mathbf{S}_{\tilde{\mathbf{y}}_k} &= \mathsf{qr}\Big\{\Big[\sqrt{w^{c_1}_1}\left(\boldsymbol{\mathcal{Y}}_{1:L,k|t} - \boldsymbol{\mathcal{Y}}_{L+1:2L,k|t}\right) \\ &\qquad \sqrt{w^{c_2}_1}\left(\boldsymbol{\mathcal{Y}}_{1:L,k|t} - \boldsymbol{\mathcal{Y}}_{L+1:2L,k|t} - 2\boldsymbol{\mathcal{Y}}_{0,k|t}\right)\Big]\Big\}\end{aligned}$$

$$\begin{aligned}\mathbf{P}_{\mathbf{x}_k\mathbf{y}_k} &= \sqrt{w^{c_1}_1}\mathbf{S}^-_{\mathbf{x}_k}\left[\boldsymbol{\mathcal{Y}}_{1:L,k|t} - \boldsymbol{\mathcal{Y}}_{L+1:2L,k|t}\right]^T \\ \mathbf{K}_k &= \left(\mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}/\mathbf{S}^T_{\tilde{\mathbf{y}}_k}\right)/\mathbf{S}_{\tilde{\mathbf{y}}_k} \\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}^-_k + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}^-_k\right) \\ \mathbf{U} &= \mathbf{K}_k\mathbf{S}_{\tilde{\mathbf{y}}_k} \\ \mathbf{S}_{\mathbf{x}_k} &= \mathsf{cholupdate}\left\{\mathbf{S}^-_{\mathbf{x}_k}, \mathbf{U}, -1\right\}\end{aligned}$$

- *Weights:* $w^m_0 = (h^2 - L)/h^2$, $w^m_i = 1/(2h^2)$, $w^{c_1}_i = 1/(4h^2)$ and $w^{c_2}_i = (h^2-1)/(4h^4)$ for $i=1,\ldots,2L$ where $h \geq 1$ is the scalar central difference interval size which is optimally set equal to the square-root of the kurtosis of the prior random variable [14]. For Gaussian prior RVs, the optimal value is $h = \sqrt{3}$. The scaling factor h in the CDKF plays the same role of α in the UKF, i.e., it determines the spread of the sigma-points around the prior mean. L is the dimension of the augmented state vector.

- *Other parameters:* $\mathbf{S}_{\mathbf{v}} = \sqrt{\mathbf{R}_{\mathbf{v}}}$ and $\mathbf{S}_{\mathbf{n}} = \sqrt{\mathbf{R}_{\mathbf{n}}}$ where $\mathbf{R}_{\mathbf{v}}$ and $\mathbf{R}_{\mathbf{n}}$ are the process-noise and observation-noise covariance matrices.

- *General note*: Here we again augment the system state with the process noise and observation noise vectors ($\mathbf{v}_k$ and $\mathbf{n}_k$) as we did for the UKF. For the CDKF, however, we split this augmentation between the time-update and measurement-update, i.e., for the time-update the augmented state vector and augmented covariance matrix is given by

$$\mathbf{x}^{a_v}_k = \begin{bmatrix} \mathbf{x}^T_k & \mathbf{v}^T_k \end{bmatrix}^T \quad , \quad \mathbf{P}^{a_v}_k = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_k} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathbf{v}} \end{bmatrix} ,$$

and by

$$\mathbf{x}^{a_n}_k = \begin{bmatrix} \mathbf{x}^T_k & \mathbf{n}^T_k \end{bmatrix}^T \quad , \quad \mathbf{P}^{a_n}_k = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_k} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathbf{n}} \end{bmatrix} ,$$

for the measurement-update. Accordingly the sigma-point vectors are given by: $\boldsymbol{\mathcal{X}}^{a_v} = \begin{bmatrix} (\boldsymbol{\mathcal{X}}^x)^T & (\boldsymbol{\mathcal{X}}^v)^T \end{bmatrix}^T$ and $\boldsymbol{\mathcal{X}}^{a_n} = \begin{bmatrix} (\boldsymbol{\mathcal{X}}^x)^T & (\boldsymbol{\mathcal{X}}^n)^T \end{bmatrix}^T$. Note: $(\cdot)^{\mathbf{2}}$ is shorthand for the vector outer product, i.e., $\mathbf{a}^{\mathbf{2}} \doteq \mathbf{a}\mathbf{a}^T$.

APPENDIX B: SPKF BASED LATENCY COMPENSATION

In order to accurately fuse a N-sample *lagged* innovation vector $\tilde{\mathbf{y}}_{k-N} = \mathbf{y}_{k-N} - \hat{\mathbf{y}}^-_{k-N}$ with the *current* prediction of the system state $\hat{\mathbf{x}}^-_k$, the Kalman update formulation of Eq. (3) is re-written as

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}^-_k + \tilde{\mathbf{K}}_{k,N}\tilde{\mathbf{y}}_{k-N} \ . \tag{42}$$

In Eq. (42) the Kalman gain is again expressed in terms of the correct covariance terms, i.e.,

$$\tilde{\mathbf{K}}_{k,N} = \mathbf{P}_{\mathbf{x}_k\tilde{\mathbf{y}}_{k-N}}\mathbf{P}^{-1}_{\tilde{\mathbf{y}}_{k-N}} \ , \tag{43}$$

where $\mathbf{P}_{\mathbf{x}_k\tilde{\mathbf{y}}_{k-N}}$ and $\mathbf{P}_{\tilde{\mathbf{y}}_{k-N}}$ are calculated by propagating sigma-points drawn from $\mathbf{P}_{\mathbf{x}_k\mathbf{x}_{k-N}}$ and $\mathbf{P}_{\mathbf{x}_{k-N}}$ (and corresponding lagged mean $\hat{\mathbf{x}}_{k-N}$) through the observation function $\mathbf{h}(\cdot)$ and applying the standard SPKF covariance calculation formulation of Eqs. (16) and (15). The optimal lagged observation prediction is given by

$$\hat{\mathbf{y}}^-_{k-N} = E\left[\mathbf{h}\left(\hat{\mathbf{x}}^-_{k-N}, \mathbf{n}_{k-N}\right)\right] \tag{44}$$

which is also calculated using the standard sigma-point propagation technique of Eq. (14) after sigma-points were drawn from $\{\hat{\mathbf{x}}_{k-N}, \mathbf{P}_{\mathbf{x}_{k-N}}\}$.

The key insight here is that we need to accurately maintain the lagged state estimate $\hat{\mathbf{x}}_{k-N}$ as well as the correct lagged covariance and cross-covariance estimates $\mathbf{P}_{\mathbf{x}_k\mathbf{x}_{k-N}}$ and $\mathbf{P}_{\mathbf{x}_{k-N}}$ within the SPKF as the system evolves from time $k-N$ to the current time k. Note that $\mathbf{P}_{\mathbf{x}_k\mathbf{x}_{k-N}}$ corresponds to the cross-covariance-over-time between the system state at time $k-N$ and the current time k, i.e., $\mathbf{P}_{\mathbf{x}_k\mathbf{x}_{k-N}} = E\left[\left(\mathbf{x}_k - \hat{\mathbf{x}}^-_k\right)\left(\mathbf{x}_{k-N} - \hat{\mathbf{x}}^-_{k-N}\right)^T\right]$. This can be achieved within the SPKF framework by augmenting the state vector at time $k-N$ with the lagged state $\mathbf{x}_{lag} = \mathbf{x}_{k-N}$, i.e.,

$$\mathbf{x}^{(a)}_{k-N} = \begin{bmatrix} \mathbf{x}_{k-N} \\ \mathbf{x}_{lag} \end{bmatrix} \tag{45}$$

and then redefining the process models as

$$\begin{aligned}\mathbf{x}^{(a)}_{k-N+1} &= \breve{\mathbf{f}}\left(\mathbf{x}^{(a)}_{k-N}, \mathbf{v}_{k-N}\right) \\ &= \begin{bmatrix} \mathbf{f}\left(\mathbf{x}_{k-N}, \mathbf{v}_{k-N}\right) \\ \mathbf{x}_{lag} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{x}_{k-N+1} \\ \mathbf{x}_{lag} \end{bmatrix}\end{aligned} \tag{46}$$

and the observation model (which is only valid at time k) by

$$\begin{aligned}\mathbf{y}^*_k &= \breve{\mathbf{h}}\left(\mathbf{x}^{(a)}_k, \mathbf{n}_{lag}\right) \\ &= \mathbf{h}\left(\mathbf{x}_{lag}, \mathbf{n}_{lag}\right) \\ &= \mathbf{h}\left(\mathbf{x}_{k-N}, \mathbf{n}_{k-N}\right) \ .\end{aligned} \tag{47}$$

Note that $\mathbf{y}_k^* = \mathbf{y}_{k-N}$, i.e., an observation of the system state at time $k-N$ which is received at time k. Using this redefined state and process model from time $k-N$ to k within the normal SPKF framework will result in the following prediction of the state mean and covariance at time k, just before the lagged measurement is fused:

$$\hat{\mathbf{x}}_k^{(a)-} = \begin{bmatrix} \hat{\mathbf{x}}_k^- \\ \hat{\mathbf{x}}_{lag}^- \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}_k^- \\ \hat{\mathbf{x}}_{k-N}^- \end{bmatrix}$$

$$\begin{aligned} \mathbf{P}^-_{\mathbf{x}_k^{(a)}} &= \begin{bmatrix} \mathbf{P}^-_{\mathbf{x}_k} & \mathbf{P}^-_{\mathbf{x}_k\mathbf{x}_{lag}} \\ \mathbf{P}^-_{\mathbf{x}_{lag}\mathbf{x}_k} & \mathbf{P}^-_{\mathbf{x}_{lag}} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{P}^-_{\mathbf{x}_k} & \mathbf{P}^-_{\mathbf{x}_k\mathbf{x}_{k-N}} \\ \mathbf{P}^-_{\mathbf{x}_{k-N}\mathbf{x}_k} & \mathbf{P}^-_{\mathbf{x}_{k-N}} \end{bmatrix} \end{aligned}$$

Sigma-points are now drawn from this prior distribution of the augmented state and propagated through the redefined observation model (Eq. (47)) in order to calculate

$$\mathbf{P}_{\mathbf{x}_k^{(a)}\tilde{\mathbf{y}}_{k-N}} = \begin{bmatrix} \mathbf{P}^T_{\mathbf{x}_k\tilde{\mathbf{y}}_{k-N}} & \mathbf{P}^T_{\mathbf{x}_{k-N}\tilde{\mathbf{y}}_{k-N}} \end{bmatrix}^T$$

and $\mathbf{P}_{\tilde{\mathbf{y}}_{k-N}}$ using the standard SPKF framework of Eqs. (16) and (15). These terms can then be used to compute the correct Kalman gain

$$\begin{aligned} \tilde{\mathbf{K}}_{k,N} &= \mathbf{P}_{\mathbf{x}_k^{(a)}\tilde{\mathbf{y}}_{k-N}}\mathbf{P}^{-1}_{\tilde{\mathbf{y}}_{k-N}} \\ &= \begin{bmatrix} \mathbf{P}_{\mathbf{x}_k\tilde{\mathbf{y}}_{k-N}}\mathbf{P}^{-1}_{\tilde{\mathbf{y}}_{k-N}} \\ \mathbf{P}_{\mathbf{x}_{k-N}\tilde{\mathbf{y}}_{k-N}}\mathbf{P}^{-1}_{\tilde{\mathbf{y}}_{k-N}} \end{bmatrix} \end{aligned}$$

needed to fuse the lagged measurement when it is received at time k.

REFERENCES

[1] A. A. Bogdanov and E. A. Wan. SDRE Control with Nonlinear Feedforward Compensation for a Small Unmanned Helicopter. In *Proceedings of the 2nd AIAA Unmanned Unlimited Systems, Technologies, and Operations Conference*, San Diego, CA, September 2003. AIAA Paper Number: 2003-6512.

[2] A. A. Bogdanov, E. A. Wan, M. Carlsson, R. Kieburtz, G. Harvey, J. Hunt, and R. van der Merwe. State-Dependent Ricatti Equation Control of a Small Unmanned Helicopter. In *Proceedings of the AIAA Guidance Navigation and Control Conference*, Austin, TX, August 2003. AIAA Paper Number: 2003-5672.

[3] E. Feron. Aerial robotics project : Laboratory for information and decision systems mit. `http://gewurtz.mit.edu/research/heli.htm`.

[4] V. Gavrilets. *Autonomous Aerobatic Maneuvering of Miniature Helicopters: Modeling and Control*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[5] G. H. Golub and van Loan C. F. *Matrix Computations*. Johns Hopkins University Press, 1996.

[6] M. Grewal, L. R. Weil, and A. P. Andrews. *Global Positioning Systems, Inertial Navigation, and Integration*. John Wiley & Sons, 1 edition, 2001.

[7] S. Haykin, editor. *Kalman Filtering and Neural Networks*, chapter 7 - The Unscented Kalman Filter, E. A. Wan and R. van der Merwe, pages 221–280. Adaptive and Learning Systems for Signal Processing, Communications, and Control. Wiley, 2001.

[8] Kazufumi Ito and Kaiqi Xiong. Gaussian Filters for Nonlinear Filtering Problems. *IEEE Transactions on Automatic Control*, 45(5):910–927, May 2000.

[9] S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proceedings of the American Control Conference*, pages 1628–1632, 1995.

[10] S. J. Julier and J. K. Uhlmann. Unscented Filtering and Nonlinear Estimation. *Proceedings of the IEEE*, 92(3):401–422, March 2004.

[11] T. D. Larsen, N. A. Andersen, and O. Ravn. Incorporation of Time Delayed Measurements in a Discrete-time Kalman Filter. In *Proceedings of the 37th IEEE Conference on Decision & Control*, pages 3972–3977, Tampa, Florida, USA, Dec 1998.

[12] Frank L. Lewis. *Optimal Estimation*. John Wiley & Sons, Inc., New York, 1986.

[13] S. Nassar, K. Schwarz, and N. El-Sheimy. INS and INS/GPS Accuracy Improvement Using Autoregressive (AR) Modeling of INS Sensor Errors. In *Proceedings of ION-NTM*, pages 936–944, San Diego, Jan 2004.

[14] M. Norgaard, N. Poulsen, and O. Ravn. New Developments in State Estimation for Nonlinear Systems. *Automatica*, 36(11):1627–1638, November 2000.

[15] J. M. Rolfe and K. J. Staples. *Flight Simulation*. Cambridge University Press, United Kingdom, 1986.

[16] P. G. Savage. Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms. *Journal of Guidance, Control, and Dynamics*, 21(1):19–28, Jan 1998.

[17] P. G. Savage. Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms. *Journal of Guidance, Control, and Dynamics*, 21(2):208–221, March 1998.

[18] Eun-Hwan Shin and Naser El-Sheimy. An Unscented Kalman Filter for In-Motion Alignment of Low-Cost IMUs. In *Proceedings of the IEEE Frames Conference*, pages 273–279, Monterey, California, April 2004.

[19] B. Stevens and F. Lewis. *Aircraft Control and Simulation*. Wiley, New York, NY, 1992.

[20] R. van der Merwe. Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models. In *Workshop on Advances in Machine Learning*, Montreal, June 2003. `http://www.iro.umontreal.ca/~kegl/CRMWorkshop/program.html`.

[21] R. van der Merwe. *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models*. PhD thesis, OGI School of Science & Engineering at Oregon Health & Science University, Portland, OR, April 2004.

[22] R. van der Merwe and E. Wan. Efficient Derivative-Free Kalman Filters for Online Learning. In *Proceedings of the 9th European Symposium on Artificial Neural Networks (ESANN)*, pages 205–210, Bruges, Belgium, Apr 2001.

[23] R. van der Merwe and E. Wan. The Square-Root Unscented Kalman Filter for State- and Parameter-Estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 6, pages 3461–3464, Salt Lake City, UT, May 2001.

[24] R. van der Merwe, E. Wan, and S. J. Julier. Sigma-Point Kalman Filters for Nonlinear Estimation and Sensor-Fusion: Applications to Integrated Navigation. In *Proceedings of the AIAA Guidance, Navigation & Control Conference*, Providence, RI, Aug 2004.

[25] E. A. Wan and R. van der Merwe. The Unscented Kalman Filter for Nonlinear Estimation. In *Proceedings of IEEE Symposium on Adaptive Systems for Signal Processing Communications and Control (AS-SPCC)*, pages 153–158, Lake Louise, Alberta, Canada, October 2000.