COMPARISON OF STOCHASTIC FILTERING TECHNIQUES FOR GPS/INS FUSION OF SOUNDING ROCKETS

Eduardo J. de A. J. Filho*, Marcos R. O. A. Maximo*, Júlio C. F. Filho*

*Aeronautics Institute of Technology
Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900
São José dos Campos, SP, Brazil

Emails: eduardojourdan92@gmail.com, mmaximo@ita.br, ita.julio.cesar@gmail.com

Abstract— This paper considers three different stochastic filtering techniques for the navigation problem of a sounding rocket: extended Kalman filter, unscented Kalman filter and particle filter. Aided navigation is used with global position obtained by low rate GPS devices as well as linear accelerations and angular velocities measured by a low-cost inertial sensor unit with high levels of noise. Flight data are obtained with a high-fidelity six degrees of freedom simulation considering the rocket characteristics of ITA Rocket Design. Euler angles are considered for angular kinematics and they are defined in order to avoid the singularity at 90 degrees of pitch. A comparison is made regarding root mean square error and computational cost. Extended and unscented Kalman filters show better performance in both criteria and the particle filter presents problems related to particle impoverishment. Furthermore, direct roughening technique is used to improve the particle filter.

Keywords— Navigation, GPS/INS Fusion, Sensor Fusion.

Resumo— Este trabalho considera 3 técnicas distintas de filtragem estocástica para o problema de navegação de um foguete de sondagem: Filtro de Kalman Estendido, Unscented Kalman Filter e Filtro de Partículas. Navegação auxiliada utiliza-se de coordenadas globais obtidas pelo GPS de baixa taxa de aquisição, bem como das acelerações lineares e velocidades angulares dadas pelos sensores da unidade inercial de baixo custo, cumuladas de grande quantidade de ruído. Dados de voo são obtidos com um simulador de seis graus de liberdade de alta fidelidade que considera o foguete característico da ITA Rocket Design. Ângulos de Euler são utilizados para a cinemática angular e são definidos de forma a evitar a singularidade de 90° de arfagem. Uma comparação foi feita com relação à raiz quadrada da média do erro quadrático médio e ao custo computacional. Os filtros de Kalman estendido e Unscented mostraram melhor performance em ambos os critérios e o filtro de partículas apresentou problemas relacionados a empobrecimento de partículas. Então, a técnica "direct roughening" foi usada para melhorar sua performance.

Palavras-chave— Navegação, Fusão GPS/INS, Fusão de Sensores

1 Introduction

Navigation is the process of estimating the vehicle's position, velocity and attitude at a given time. Given the vehicle's initial states, its states at a given time may be estimated by integrating accelerations and angular velocities measured using an inertial measurement unit (IMU). A system that performs this process is termed an inertial navigation system (INS).

Nevertheless, errors in acceleration and angular velocity accumulate over time in position and the position estimate quickly drifts. Therefore, inertial systems require high accuracy and fast update rates, which are not possible on hobby-grade sensors. On the other hand, Global Positioning System (GPS) provide driftless global position measurements, but low-cost GPS devices have low sampling rates and significant noise levels.

GPS/INS fusion is a well-known strategy for Navigation where the strengths of both sensors are combined. This sensor fusion may be accomplished using stochastic filtering techniques, which consider stochastic mathematical models for the sensors and the vehicle's dynamics to recursively estimate the vehicle's states. Given linear dynamics, linear measurement model and gaussian additive noises, the Kalman Filter is the optimal quadratic estimator (Bruno, 2013). However, six degrees of freedom (6DOF) rigid-body equations are intrinsically nonlinear, thus nonlinear techniques are needed for the navigation of an aerospace vehicle.

We are interested in developing a GPS/INS fusion algorithm for the navigation system of the ITA Rocket Design's sounding rocket. ITA Rocket Design is a rocket team composed by undergraduate students from the Aeronautics Institute of Technology (ITA). Considering the trade-off between filtering performance and computational cost, the best stochastic filtering technique is usually problem dependent, thus we evaluated three popular nonlinear techniques: Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), and Particle Filter (PF). The EKF and UKF are nonlinear extensions of the Kalman Filter, while the PF is based on Monte Carlo methods (Bruno, 2013).

Other works have compared stochastic filtering techniques. Gross et al. (2010) compared EKF, UKF, and PF for an unmanned aerial vehicle (UAV) navigation system focusing on attitude estimation . In this problem, the filtering performances of the EKF and UKF were close, while the PF with 200 particles showed the worst results. Then, the EKF may be regarded as the best choice

ISSN 2525-8311

for UAV navigation due to its very low computational cost. Similar conclusions were found for a problem where mobile robots are localized using an overhead camera (Pinto et al., 2015). However, the PF usually shows superior performance experimentally when strong nonlinearities and multimodal probability distributions are present (Silva and Bruno, 2006).

The remaining of this paper is organized as follows. In Section 2, a mathematical background about the stochastic filtering techniques is presented. Section 3 shows the equations involved in the dynamics and observation models of a sounding rocket. In Section 4, we compare the three techniques using data generated by a six degrees of freedom rocket flight simulator. Finally, Section 5 concludes and shares our ideas for future work.

2 Stochastic Filtering

A generic stochastic dynamical system may be modeled by the following prediction and observation equations:

$$\mathbf{x}_k = f_k\left(\mathbf{x}_{k-1}, \boldsymbol{\varepsilon}_{k-1}, \mathbf{w}_{k-1}\right) \tag{1}$$

$$\mathbf{z}_k = h_k\left(\mathbf{x}_k, \mathbf{v}_k\right) \tag{2}$$

where $\mathbf{w}_{k-1}$ and $\mathbf{v}_k$ are random noises and f_k and h_k are generic functions, possibly nonlinear. The stochastic filtering problem consists of determining the probability density function (*pdf*) for the state vector $\mathbf{x}_k$ at every time instant t_k, given the initial state distribution $\mathbf{x}_0 \sim p(\mathbf{x})$ and the history of inputs $\boldsymbol{\varepsilon}_i$ and observations $\mathbf{z}_i$, $i = 1, \ldots, k$.

We adopted the usual assumption that the random variables $\{\mathbf{x}_0, \mathbf{w}_{0:k-1}, \mathbf{v}_{1:k}\}$ are mutually independent and jointly Gaussian. Furthermore, we considered that $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{M}_k)$ and $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$. Given these assumptions and further assuming that the functions f_k and h_k are linear, the Kalman filter is the optimal quadratic estimator to solve the stochastic filtering problem.

Given an estimate with *pdf* $\mathcal{N}\left(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1}\right)$ at time step t_{k-1}, the prediction phase of the Kalman filter propagates the estimate to time step t_k, yielding an estimate with *pdf* $\mathcal{N}\left(\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}\right)$. Then, the observation is incorporated in the update phase to yield an estimate with *pdf* $\mathcal{N}\left(\hat{\mathbf{x}}_{k|k}, \mathbf{P}_{k|k}\right)$. These prediction and update phases use the knowledge contained in equations (1) and (2). If the functions f_k and h_k are nonlinear, alternative approaches may be employed, such as the ones presented herein.

2.1 Extended Kalman Filter

As said before, the Kalman Filter only works for linear systems. However, the majority of engineering problems has nonlinearities. Therefore, to solve this kind of problem, an extension of the Kalman Filter was introduced. That extension is known as the Extended Kalman Filter (EKF) and is described in Ljung (1979). It considers that still holds the Gaussianity hypothesis for the random noises and for the initial Gaussian distribution and that f_k and h_k are differentiable functions. Then, taking a first order expansion using Taylor series for the functions f_k and h_k, the algorithm for the EKF is shown in Algorithm 1.

Algorithm 1 Extended Kalman Filter

function Initialize($\hat{\mathbf{x}}_0, \mathbf{P}_0$)
  $\hat{\mathbf{x}}_{0|0} = \hat{\mathbf{x}}_0$
  $\mathbf{P}_{0|0} = \mathbf{P}_0$
end function

function Predict($\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1}, \boldsymbol{\varepsilon}_{k-1}$)
  $\hat{\mathbf{x}}_{k|k-1} = f_k\left(\hat{\mathbf{x}}_{k-1|k-1}, \boldsymbol{\varepsilon}_{k-1}, \mathbf{0}\right)$
  $\mathbf{F}_k = \left.\frac{\partial f_k}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{k-1|k-1}, \boldsymbol{\varepsilon}=\boldsymbol{\varepsilon}_{k-1}, \mathbf{w}=\mathbf{0}}$
  $\mathbf{L}_k = \left.\frac{\partial f_k}{\partial \mathbf{w}}\right|_{\mathbf{x}=\mathbf{x}_{k-1|k-1}, \boldsymbol{\varepsilon}=\boldsymbol{\varepsilon}_{k-1}, \mathbf{w}=\mathbf{0}}$
  $\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k|k-1} \mathbf{F}_k^T + \mathbf{L}_k \mathbf{M}_k \mathbf{L}_k^T$
end function

function Update($\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}, \mathbf{z}_k$)
  $\mathbf{H}_k = \left.\frac{\partial h_k}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_{k|k-1}, \mathbf{v}=\mathbf{0}}$
  $\mathbf{G}_k = \left.\frac{\partial h_k}{\partial \mathbf{v}}\right|_{\mathbf{x}=\mathbf{x}_{k|k-1}, \mathbf{v}=\mathbf{0}}$
  $\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{G}_k \mathbf{R}_k \mathbf{G}_k^T$
  $\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}$
  $\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - h\left(\hat{\mathbf{x}}_{k|k-1}\right)\right)$
  $\mathbf{P}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\right) \mathbf{P}_{k|k-1}$
end function

2.2 Unscented Kalman Filter

The EKF may perform poorly when the prediction and observation functions are highly nonlinear. The UKF uses a deterministic sampling technique, known as the unscented transform (UT). The UT carefully picks samples (called sigma points) around the mean and propagate them through the nonlinear function, from which the mean and covariance of the estimate are computed. This method better approximates the posterior's true mean and covariance (Julier and Uhlmann, 2004). Therefore, the UKF often provides better estimates than the EKF with similar computational cost. Additionally, the UKF does not require Jacobians computations, which may be a hard task in very complex models. Algorithm 2 shows the predict and update functions for the UKF.

In this pseudocode, α, β, and λ are constants, where α determines the spread of the sigma points around the mean and is usually set to a small value (we use $\alpha = 10^{-3}$), $\beta = 2$ is the optimal value for gaussian distributions, and $\lambda = \alpha^2\left(n + \kappa\right)$, where

κ is an auxiliary constant usually set to 0. The notation $\sqrt{\mathbf{P}}$ means the square-root matrix of $\mathbf{P}$, i.e. the matrix $\mathbf{F}$ such as $\mathbf{F}\mathbf{F}^T = \mathbf{P}$. The square-root of a matrix may be computed using the Cholesky decomposition. Moreover, $\mathbf{F}_{:,i}$ is the i-th column of the matrix $\mathbf{F}$.

2.3 Particle Filter

Particle Filter (PF) is the most general method for approximating the true Bayesian optimal estimate between the three discussed in Section 2. Different from EKF and UKF, PF is fully capable of handling non-linear models and is based on non-deterministic sampling techniques (Arulapalam et al., 2002). As can be seen in following pseudocode, the state vector $\mathbf{x}_{k-1}$ at each instant t_{k-1} is represented by a set $\boldsymbol{\chi}_{k-1}$ of N random samples called particles. The prediction phase corresponds to estimate $\hat{\mathbf{x}}_{k|k-1}$ by simply propagating each particle using the equations of motion described in Section 3. The input $\boldsymbol{\varepsilon}_{k-1}^{(n)}$ associated with each sample n has a *pdf* $\mathcal{N}(\boldsymbol{\varepsilon}_{k-1}, \mathbf{M})$.

The update phase corresponds to determine the a posteriori estimative $\hat{\mathbf{x}}_{k|k}$.If there are new observations then it is done by considering that each particle n has a weight $w_k^{(n)}$ proportional to the likelihood of its states.

A well known problem existing in PF is particle degeneration as can be seen in Arulapalam et al. (2002). After few iterations, it is very common that some particles have very small weights and so contributing very little to the estimative $\hat{\mathbf{x}}_{k|k}$ in spite of spending a lot of computational resources. A well known solution to this is to resample the particles after the update phase by choosing the particles with more likelihood and discarding the others. Reference (Arulapalam et al., 2002) presents a sequential stochastic process to do such task that is $O(N)$ and therefore has a practical computational cost. After resampling, the particles have the same weight $w_k^{(n)} = 1/N$ (Arulapalam et al., 2002). This variant of PF is called Sampling Importance Resampling (SIR).

Another problem regarding PF is called particle's impoverishment, when the particle's cloud is too small and does not reflect the true density (Li et al., 2014). The degeneracy converts to impoverishment as a direct result of resampling and it is more common and critical as the number of states grows. There are a few ways to deal with this problem as described in Li et al. (2014). The procedure chosen in the present work was the one called direct roughening. It consists of directly applying more random noise to each particle input $\boldsymbol{\varepsilon}_{k-1,n}$ in the prediction phase and therefore helping to spread the particle‘s cloud. The amount of extra noise cannot be too large or it will blur the states estimates out of the range of the observations. More sophisticated procedures which do not destroy PF‘s characteristics are being considered to be done in future works.

Algorithm 2 Unscented Kalman Filter

function DrawSigmaPoints($\mathbf{x}^a$, $\mathbf{P}^a$)
  $\boldsymbol{\chi}^{(0)} = \mathbf{x}^a$
  $w_m^{(0)} = \frac{\lambda}{n+\lambda}$
  $w_c^{(0)} = \frac{\lambda}{n+\lambda} + \left(1 - \alpha^2 + \beta\right)$
  for $i = 1, \ldots, n$ **do**
    $\boldsymbol{\chi}^{(i)} = \mathbf{x}^a + \left(\sqrt{(n+\lambda)\,\mathbf{P}^a}\right)_{:,i}$
    $w_m^{(i)} = w_c^{(i)} = \frac{1}{2(n+\lambda)}$
  end for
  for $i = n+1, \ldots, 2n$ **do**
    $\boldsymbol{\chi}^{(i)} = \mathbf{x}^a - \left(\sqrt{(n+\lambda)\,\mathbf{P}^a}\right)_{:,i-n}$
    $w_m^{(i)} = w_c^{(i)} = \frac{1}{2(n+\lambda)}$
  end for
end function

function Initialize($\hat{\mathbf{x}}_0, \mathbf{P}_0$)
  $\hat{\mathbf{x}}_{0|0} = \hat{\mathbf{x}}_0$
  $\mathbf{P}_{0|0} = \mathbf{P}_0$
end function

function Predict($\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1}, \boldsymbol{\varepsilon}_{k-1}$)
  $\mathbf{x}^a = \begin{bmatrix}\hat{\mathbf{x}}_{k-1|k-1}^T & \mathbf{0}_m^T\end{bmatrix}^T$
  $\mathbf{P}^a = \mathrm{diag}(\mathbf{P}_{k-1|k-1}, \mathbf{M}_k)$
  DrawSigmaPoints($\mathbf{x}^a, \mathbf{P}^a$)
  for $i = 0, \ldots, 2n$ **do**
    $\boldsymbol{\chi}_{k|k-1}^{(i)} = f_k(\boldsymbol{\chi}_x^{(i)}, \boldsymbol{\varepsilon}_{k-1}, \boldsymbol{\chi}_w^{(i)})$
  end for
  $\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} w_m^{(i)} \boldsymbol{\chi}_{k|k-1}^{(i)}$
  $\mathbf{P}_{k|k-1} = \sum_{i=0}^{2n} \left\{ w_c^{(i)} \left(\boldsymbol{\chi}_{k|k-1}^{(i)} - \mathbf{x}_{k|k-1}\right) \times \left(\boldsymbol{\chi}_{k|k-1}^{(i)} - \mathbf{x}_{k|k-1}\right)^T \right\}$
end function

function Update($\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}, \mathbf{z}_k$)
  $\mathbf{x}^a = \begin{bmatrix}\hat{\mathbf{x}}_{k|k-1}^T & \mathbf{0}_r^T\end{bmatrix}^T$
  $\mathbf{P}^a = \mathrm{diag}(\mathbf{P}_{k|k-1}, \mathbf{R}_k)$
  DrawSigmaPoints($\mathbf{x}^a, \mathbf{P}^a$)
  for $i = 0, \ldots, 2n$ **do**
    $\boldsymbol{\gamma}_k^{(i)} = h_k(\boldsymbol{\chi}_x^{(i)}, \boldsymbol{\chi}_v^{(i)})$
  end for
  $\hat{\mathbf{z}}_k = \sum_{i=0}^{2n} w_m^{(i)} \boldsymbol{\gamma}^{(i)}$
  $\mathbf{P}_{zz} = \sum_{i=0}^{2n} w_c^{(i)} \left(\boldsymbol{\gamma}_k^{(i)} - \hat{\mathbf{z}}_k\right)\left(\boldsymbol{\gamma}_k^{(i)} - \hat{\mathbf{z}}_k\right)^T$
  $\mathbf{P}_{xz} = \sum_{i=0}^{2n} \left\{ w_c^{(i)} \left(\boldsymbol{\chi}_{k|k-1}^{(i)} - \mathbf{x}^a\right) \times \left(\boldsymbol{\gamma}_k^{(i)} - \hat{\mathbf{z}}_k\right)^T \right\}$
  $\mathbf{K}_k = \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}$
  $\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \hat{\mathbf{z}}_k\right)$
  $\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{P}_{zz}\mathbf{K}_k^T$
end function

Algorithm 3 Particle Filter: SIR approach

function InitializeParticles($\hat{\mathbf{x}}_0, \mathbf{P}_0$)
  for $n = 1, \ldots, N$ **do**
    $\boldsymbol{\chi}_{0|0}^{(n)} = \mathcal{N}\left(\hat{\mathbf{x}}_0, \mathbf{P}_0\right)$
  end for
end function

function Predict($\boldsymbol{\chi}_{k-1|k-1}, \boldsymbol{\varepsilon}_{k-1}$)
  $\mathbf{M} = diag\left(\mathbf{M_{acc}}, \mathbf{M_{gyr}}\right)$
  for $n = 1, \ldots, N$ **do**
    $\boldsymbol{\varepsilon}_{k-1}^{(n)} = \mathcal{N}\left(\boldsymbol{\varepsilon}_{k-1}, \mathbf{M}\right)$
    $\boldsymbol{\chi}_{k|k-1}^{(n)} = f_k(\boldsymbol{\chi}_{k-1|k-1}^{(n)}, \boldsymbol{\varepsilon}_{k-1}^{(n)})$
  end for
  $\hat{\mathbf{x}}_{k|k-1} = \sum_{n=1}^{N} w_{k-1}^{(n)} \boldsymbol{\chi}_{k|k-1}^{(n)}$
end function

function Update($\boldsymbol{\chi}_{k|k-1}, \mathbf{z}_k$)
  for $n = 1, \ldots, N$ **do**
    $p(\mathbf{z}_k|\boldsymbol{\chi}_{k|k-1}^{(n)}) = \prod_{j=1}^{3} exp\left[-\frac{\left(z_{j,k}-\chi_{j,k|k-1}^{(n)}\right)^2}{2\sigma_j^2}\right]$
    $w_k^{(n)} = \frac{w_{k-1}^{(n)} p(\mathbf{z}_k|\boldsymbol{\chi}_{k|k-1}^{(n)})}{\sum_{n=1}^{N} w_k^{(n)}}$
  end for
  $\boldsymbol{\chi}_{k|k} = Resample(\boldsymbol{\chi}_{k|k-1}, \mathbf{w}_k^{(1:n)})$
  $\hat{\mathbf{x}}_{k|k} = \frac{\sum_{n=1}^{N} \boldsymbol{\chi}_{k|k}^{(n)}}{N}$
end function

function $\boldsymbol{\chi}_{k|k} =$ Resample($\boldsymbol{\chi}_{k|k-1}, \mathbf{w}_k^{(1:n)}$)
  $r = Uniform(0, 1/N)$
  $c = w_k^{(1)}$
  $i = 1$
  for $n = 1, \ldots, N$ **do**
    $U = r + (n-1)/N$
    while $U > c$ **do**
      $i = i + 1$
      $c = c + w_k^{(i)}$
    end while
    $resampleIndex(n) = i$
  end for
  for $n = 1, \ldots, N$ **do**
    $w_k^{(n)} = 1/N$
    $\boldsymbol{\chi}_{k|k}^{n} = \boldsymbol{\chi}_{k|k-1}^{resampleIndex(n)}$
  end for
end function

3 Sounding Rocket Modeling

The sounding rocket model used in the present work uses the flat-Earth system of reference where the origin is located in the launch base and the movement is related to the local plane tangent to the Earth in that same point. For the purpose of this study that system can be considered inertial as both flight duration and envelope are relative small to Earth's radius and angular velocity. The x axis is in the horizontal plane and points to the launch direction. The z axis is pointed downwards to the center of the Earth and the y axis completes the system respecting the right hand rule. Note that with this definition positive altitude reflects in a negative component z of position and the position (0,0,0) is exactly the launch site.

Another reference system used was the body system, where the origin is in the vehicle's center of mass. When the rocket is in the horizontal plane, the x axis is aligned with the longitudinal direction of the rocket pointing forward, the z axis is pointing downward and the y axis is pointing right.

The translational states are the position (x, y, z) and linear velocity (u, v, w) of the body reference system (rocket) relative to the inertial system (flat-Earth). The linear velocity components are described in the body reference system, not the inertial one.

The order chosen to rotate the inertial frame into the body reference system was 2-3-1, different from the conventional 3-2-1 chosen for aircraft simulations. The first rotation about the y axis defines the pitch angle θ, the second rotation defines the yaw angle ψ and the third one defines the roll angle ϕ. With these different definitions, the singularity occurs not when $\theta = 90^o$ but when $\psi = 90^o$, a rather impossible event to achieve for an unguided sounding rocket. The final rotation matrix from body to inertial $\boldsymbol{M_{ib}}$ can be found in Henderson (1977). The angular velocity of the body frame relative to the inertial system has the components (p, q, r) described in the body reference system. They are measured by the three gyrometers located in the vehicle.

The following expressions are the kinematics equations of motion, in which linear and angular velocities are related to translational position and Euler angles.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \boldsymbol{M_{ib}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3}$$

$$\boldsymbol{M_{ib}} = \begin{bmatrix} \mathrm{c}\psi\mathrm{c}\theta & \mathrm{s}\phi\mathrm{s}\theta - \mathrm{c}\phi\mathrm{c}\theta\mathrm{s}\psi & \mathrm{c}\phi\mathrm{s}\theta + \mathrm{s}\phi\mathrm{c}\theta\mathrm{s}\psi \\ \mathrm{s}\psi & \mathrm{c}\phi\mathrm{c}\psi & -\mathrm{s}\phi\mathrm{c}\psi \\ -\mathrm{c}\psi\mathrm{s}\theta & \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi + \mathrm{s}\phi\mathrm{c}\theta & \mathrm{c}\phi\mathrm{c}\theta - \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi \end{bmatrix} \tag{4}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\psi} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} p - q * \cos\phi * \tan\psi + r * \sin\phi * \tan\psi \\ \left(q * \cos\phi - r * \sin\phi\right) / \cos\psi \\ q * \sin\phi + r * \cos\phi \end{bmatrix} \tag{5}$$

The following expressions are the translational dynamics equations in which the accelerations (a_x, a_y, a_z) described in the body axis are related to the change of linear velocities (u, v, w). The forces that affects the vehicle's motion are the motor thrust, aerodynamic forces and the gravity. The first two are measured by the accelerometers.

It is interesting to notice that the angular dynamics equations are not necessary since the angular velocity is directed measured by the gyrometers.

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} a_x - gc\psi s\theta + rv - qw \\ a_y + g\left(c\phi s\theta s\psi + s\phi c\theta\right) + pw - ru \\ a_x + g\left(c\phi c\theta - s\phi s\theta s\psi\right) + qu - pv \end{bmatrix} \tag{6}$$

We may summarize the vehicle's navigation states as $[x, y, z, u, v, w, \theta, \psi, \phi]$. The measurements made by accelerometers and gyrometers $[a_x, a_y, a_z, p, q, r]$ are used as input $\boldsymbol{\varepsilon}$ in equation (1) and the GPS data $[x, y, z]$ as update measurements in the Equation (2).

4 Simulation Experiments

To generate test data, we used a 6DOF rocket flight simulator developed in C++ using the ITA Rocket Design's sounding rocket model. This simulator integrates the 6DOF flight dynamics equations using the two-step Adams-Bashforth method using high-fidelity models for the physical processes, such as aerodynamics and propulsion. The sample time used for simulation was 1 ms. The total simulation time was 25 s. The motor ignites at $t = 0$ s and burns until $t = 4.5$ s. Although the sounding rocket model used in the filters use Euler angles as described in Section 3, the 6DOF simulator works with quaternions. Before the experiments, the 6DOF output data had to be converted into the Euler angles described in Section 3 using simple formulas as presented in (Henderson, 1977).

Values sampled from Gaussian distributions were added to the trajectory computed by the simulator to simulate sensor noise. The parameters for the sensors were set to values close to the ones found in the datasheets of the real modules. Therefore, the IMU and GPS sample rates are 200 Hz and 5 Hz, respectively. The accelerometer and gyrometer covariances were set to $\mathbf{M}_{acc} = \text{diag}(0.229^2, 0.229^2, 0.377^2)$ m^2s^{-4} and $\mathbf{M}_{gyr} = \left(0.03\pi\sqrt{200^{-1}}/180\right)^2 \mathbf{I}_3$ rad^2s^{-2}. The GPS covariance is $\mathbf{R} = \text{diag}(3, 3, 5^2)$ m^2.

We implemented the three filtering techniques in MATLAB following the algorithms presented in Section 2. To integrate the model's equations, we used Euler for EKF and 4th order Runge-Kutta for UKF and the PF. The filtering begins as soon as the simulation starts. To initialize the filters, we sample a mean $\hat{\mathbf{x}}_0 \sim \mathcal{N}(\mathbf{x}_0, \mathbf{P}')$ and set $\mathbf{P}_0 = \mathbf{P}'$, where $\mathbf{x}_0$ is the initial states vector, $\mathbf{P}' = \text{diag}(\mathbf{R}, \mathbf{V}_0, \mathbf{\Phi}_0)$, $\mathbf{V}_0 = \text{diag}(0.229^2, 0.229^2, 0.377^2)$ m^2s^{-2}, and $\mathbf{\Phi}_0 = \left(0.03\pi\sqrt{200^{-1}}/180\right)^2 \mathbf{I}_3$ rad^2s^{-2}. Additionally, we used 100 particles for the particle filter (PF). To illustrate, Figure 1 shows the X-Z trajectory of the rocket in a single simulation where the UKF was used to compute the estimate.

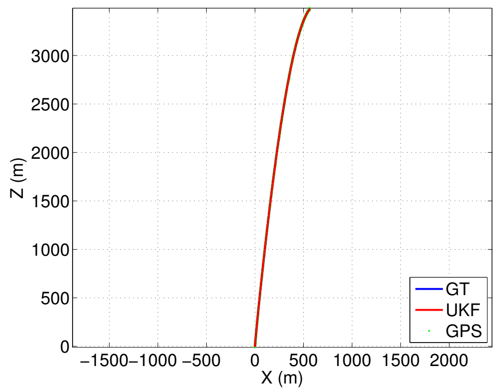

Figure 1: X-Z trajectory of the rocket showing: ground truth (GT) data from the simulator in blue, the trajectory estimated by the UKF in red, and GPS measurements as green dots.

To make the comparison less sensible to randomness, we ran Monte Carlo simulations and computed root mean square errors (RMSE) for each time instant. Given estimates $q_k^{(i)}$ at time instant $t = t_k$ of a certain quantity $q_k^{(i)}$ obtained in the i-th Monte Carlo run, we compute the RMSE of $\hat{q}_k$ using:

$$RMSE\left(\hat{q}_k\right) = \sqrt{\frac{1}{N_{mc}} \sum_{i=1}^{N_{mc}} \left(\hat{q}_k^{(i)} - q_k^{(i)}\right)^2} \tag{7}$$

where N_{mc} is the number of Monte Carlo runs. Figure 2 show the linear position, linear speed, and pitch angle RMSEs (using $N_{mc} = 1000$). The same trend is observed over the 3 figures: the PF (with 100 particles) shows the worst filtering performance, while EKF and UKF results are very similar. In Figure 2(b), an interesting effect occurs around 4.5 s: the speed RMSE sudden increases due to the motor stopping burning.

Given that navigation involves real-time computations, we are also concerned about computational cost. Hence, Table 1 shows processing time statistics for the prediction and update phases of the three filters. As expected, the PF is more computationally costly than the UKF, which is slower than the EKF. Thus, the results indicate that the EKF is the best choice for this problem. We used a Intel Core i5 2.6 GHz dual-core notebook for this comparison.

5 Conclusion

The main contribution of this work is showing that, although less complex, the first order nonlinear extension of the Kalman filter (EKF) outperforms Particle Filter and has results similar to the Uscented Kalman Filter for the model of a

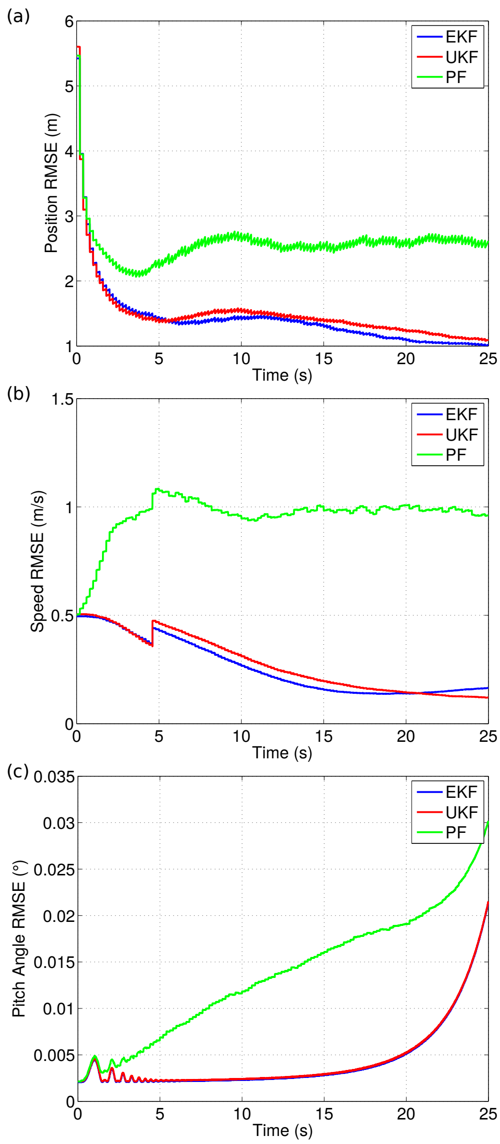

Figure 2: RMSE results considering the three filters (EKF, UKF, and PF with 100 particles): (a) linear position, (b) linear speed, and (c) pitch angle.

Table 1: Processing time statistics for each filter phase.

Function	Mean (ms)	Std. Dev. (ms)
EKF Prediction	0.05	0.02
EKF Update	0.1	0.08
UKF Prediction	11.31	2.23
UKF Update	2.84	0.54
PF Prediction	24.07	3.88
PF Update	3.15	1.91

sounding rocket. This may be explained by the following: the model does not present strong nonlinearities and we did not use a large number of particles for the PF. Thus, since the processing time of the EKF is the smallest of the three techniques, for real-time applications, it is the more qualified option.

This work can be extended in many directions. We may extend the sounding rocket model to use quaternions, thus removing the possibility of singularity related to Euler angles. We may also include bias effects in the accelerometer and the gyrometer models. Another path is to use other sensors present in the IMU, such as the barometer and the magnetometer, in the update phase.

References

Arulapalam, S., M., Gordon, N. and T., C. (2002). A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking, *IEEE Transactions on Signal Processing* **50**(02): 157–173.

Bruno, M. G. S. (2013). *Sequential Monte Carlo Methods for Nonlinear Discrete-Time Filtering*, Synthesis Lectures on Signal Processing.

Gross, J., Gu, Y., Gururajan, S., Seanor, B. and Napolitano, M. R. (2010). A Comparison of Extended Kalman Filter, Sigma-Point Kalman Filter, and Particle Filter in GPS/INS Sensor Fusion, *Proceedings of the 2010 AIAA Guidance, Navigation, and Control Conference*.

Henderson, D. M. (1977). Shuttle Program: Euler Angles, quaternions and transformations matrices working relationships, NASA- Technical Memorandum.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented Filtering and Nonlinear Estimation, *Proceedings of the IEEE*, Vol. 92.

Li, T., Sun, S., Sattar, T. P. and Corchado, J. M. (2014). Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches, *Expert Systems with Applications* **41**(08): 3944–3954.

Ljung, L. (1979). Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems, *IEEE Transactions on Automatic Control* **24**(1): 36 – 50.

Pinto, S. C., Maximo, M. R. O. A. and Bruno, M. G. S. (2015). Comparison of Nonlinear Stochastic Filters Performance for Differential Drive Robots Localization, *Proceedings of the 2015 Simpósio Brasileiro de Automação Inteligente (SBAI)*.

Silva, P. R. A. and Bruno, M. G. S. (2006). Mobile Robot Localization using Improved SIR Filters and Parametric Models of the Environment, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2006*.