2021 International Conference on Unmanned Aircraft Systems (ICUAS)
Athens, Greece. June 15-18, 2021

Nonlinear Estimation of Sensor Faults With Unknown Dynamics for a Fixed Wing Unmanned Aerial Vehicle

Enzo Iglesis[1] Nadjim Horri[2] Karim Dahia[3] James Brusey[4] and Helene Piet-Lahanier[5]

***Abstract*— In this paper, the estimation of additive inertial navigation sensor faults with unknown dynamics is considered with application to the longitudinal navigation and control of a fixed wing unmanned aerial vehicle. The faulty measurement is on the pitch angle. A jump Markov regularized particle filter is proposed for fault and state estimation of the nonlinear aircraft dynamics, with a Markovian jump strategy to manage the probabilistic transitions between the fault free and faulty modes. The jump strategy uses a small number of sentinel particles to continue testing the alternate hypothesis under both fault free and faulty modes. The proposed filter is shown to outperform the regularized particle filter for this application in terms of fault estimation accuracy and convergence time for scenarios involving both abrupt and incipient faults, without prior knowledge of the fault models. The state estimation is also more accurate and robust to faults using the proposed approach. The root-mean-square error for the altitude is reduced by 77 % using the jump Markov regularized particle filter under a pitch sensor fault amplitude of up to 10 degrees. Performance enhancement compared to the regularized particle filter was found to be more pronounced when fault amplitudes increase.**

I. Introduction

Sensor fault detection and diagnosis systems are increasingly important to aircraft mission integrity. Inertial navigation malfunctions in particular have often been the cause of flight incidents and crashes as in the case of the Qantas F72 and Croatia Boeing 737–200 flights [1]. Small unmanned aerial vehicles are also becoming increasingly autonomous, with a need to further develop their ability to detect, estimate and recover from sensor faults. The certification of autonomous unmanned aerial vehicle (UAV) systems is also subject to the development of systems to monitor sensor data and fault flags.

Fault detection often consists of statistical tests used to detect a change on residuals. This is either performed by comparison against a threshold or by assuming a residual distribution, such as the Student's t-test, the cumulative sum (CUSUM) or a likelihood ratio test such as the generalized likelihood ratio test (GLRT) [2].

A higher level of fault diagnosis is fault estimation. Indeed, fault detection does not provide quantitative information on the impact of fault on the system. Hence, fault estimation is required to obtain valuable information that can be exploited e. g. to compensate the fault. For linear Gaussian models, the Kalman filter (KF) is known to be the optimal estimator and the approach has been extended to fault estimation. For nonlinear models, the extended Kalman filter (EKF) has been used for both fault detection [3] and fault estimation [4]. When a variety of faults is considered, interacting multiple model (IMM) approaches are often used, and they rely on the use of several filters, all of which are associated to a specific fault mode. IMMs usually consist of interacting KFs or EKFs depending on the linearity of the system. They have been proven efficient for various applications as in [5] but present the major drawback of requiring detailed knowledge of the models of the various faults.

Particle filters have proved their efficiency in estimation problems for non-linear dynamics and non-Gaussian noises [6]. In the context of fault estimation it has been combined with a jump Markov transition model that enables to switch from a nominal to faulty model using the transition probability matrix that accounts for abrupt changes of modes [7], [8]. This approach is combined in this paper with a regularized particle filter (RPF) [9] and a Kalman correction step to handle unexpected fault dynamics. It is called a jump Markov regularized particle filter (JMRPF).

The dynamical model of fixed wing UAVs is well known [10], a model-based approach [11]–[14] is therefore used in this paper. With no prior knowledge of the fault dynamics, a zero-order constant fault model is used by the JMRPF, which is initialized by default in a fault free mode. The actual incipient sensor faults applied to the system do not match this zero-order model and the process and regularization noises are not set to handle abrupt fault amplitudes, but the JMRPF is shown in Section VI to accurately estimate sensor faults and robustly estimate the UAV states despite this model mismatch. This is due to the fact that piecewise constant approximations of the faults can be rapidly tracked using the JMRPF that switches more swiftly when a mode change is detected.

The proposed approach is applied to a nonlinear model of longitudinal dynamics for a fixed wing UAV.

The main contributions of the paper are as follows:

- A jump strategy between the fault free and faulty sensor modes is proposed, where the a priori distribution of

This work was developed as part of a cotutelle agreement between Coventry University, ONERA and Paris-Saclay University

[1]Enzo Iglesis is a PhD student at Coventry University, Coventry, UK. iglesise@uni.coventry.ac.uk

[2]Nadjim Horri is a senior lecturer in aerospace engineering at the School of Mechanical, Aerospace and Automotive Engineering, Coventry University, Coventry, UK. nadjim.horri@coventry.ac.uk

[3]Karim Dahia is a research scientist at DTIS, ONERA, Palaiseau, France. karim.dahia@onera.fr

[4]James Brusey is a professor at the faculty research Centre for Data Science, Coventry University, Coventry, UK. james.brusey@coventry.ac.uk

[5]Helene Piet-Lahanier is scientific deputy at DTIS, ONERA, Palaiseau, France. helene.piet-lahanier@onera.fr

2021 International Conference on Unmanned Aircraft Systems (ICUAS) | 978-1-6654-1535-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICUAS51884.2021.9476852

the fault is computed using sensor innovation terms. Under both fault free and faulty modes, a small set of sentinel particles is allowed to test the alternate mode, leading to fast mode transitions and higher fault and state estimation accuracy.

- A Kalman correction is added to the JMRPF to place the particles in the most likely areas of the state space. This correction further improves state and fault estimation accuracy and was not used in reference [15].

The paper is organized as follows. In Section II, a nonlinear model of fixed wing UAV flight dynamics is presented, including the longitudinal UAV autopilot. Section III is focused on the sensor fault types and modes that the nonlinear filters have to estimate. In Section IV, a more general stochastic nonlinear Markovian jump model is presented including the effect of the fault modes Section V details the JMRPF approach for state and sensor fault estimation in a stochastic Markovian jump model framework. In Section VI, the fault and state estimation performance of the JMRPF is evaluated and compared against the RPF for scenarios including abrupt and incipient faults with no prior knowledge of the fault dynamics and amplitudes. Section VII concludes the paper.

II. Longitudinal Flight Dynamics and Control System

A. Longitudinal UAV Dynamics

A nonlinear model of longitudinal fixed wing UAV dynamics is used in this paper. The state vector representing UAV longitudinal dynamics is denoted $\mathbf{z} = \begin{bmatrix} \mathrm{p_d} & \mathrm{u} & \mathrm{w} & \theta & \mathrm{q} \end{bmatrix}^\top$. The state $\mathrm{p_d}$ denotes the position component in the downward direction along k^v (see Fig. 1), u represents the variation with respect to the trim condition on longitudinal velocity along the i^b axis, w represents the vertical velocity along the k^b axis (see Fig. 1) and the states θ and q respectively represent the pitch angle and pitch rate variations. The nonlinear longitudinal model of the Aerosonde UAV is obtained from [10] and is given by:

$$\begin{cases} \dot{\mathrm{p}}_\mathrm{d} = -\sin(\theta)\mathrm{u} + \cos(\theta)\mathrm{w} & \text{(1a)} \\ \dot{\mathrm{u}} = -\mathrm{qw} + \dfrac{F_x}{\mathfrak{m}} & \text{(1b)} \\ \dot{\mathrm{w}} = \mathrm{qu} + \dfrac{F_z}{\mathfrak{m}} & \text{(1c)} \\ \dot{\theta} = \mathrm{q} & \text{(1d)} \\ \dot{\mathrm{q}} = -\dfrac{J_{xz}}{J_y}\mathrm{q}^2 + \dfrac{1}{J_y}M & \text{(1e)} \end{cases}$$

where F_x and F_z respectively represent the external forces along the i^b and k^b axes (see Fig. 1), $\mathfrak{m}$ is the mass of the UAV, J_{xz} is a product of inertia, J_y is the moment of inertia about the pitch axis and M is the pitching moment. All these parameters are defined in [10].

The discrete time model derived from (1) can be expressed as:

$$\mathbf{z}_{k+1} = \mathcal{F}_k(\mathbf{z}_k, \mathbf{u}_k) \tag{2}$$

where $\mathbf{u} = \begin{bmatrix} \delta_e & \delta_t \end{bmatrix}^T$ is the control input vector and δ_e and δ_t respectively represent the elevator deflection and throttle input. The input δ_e is bounded in the interval $[-25°, 25°]$ and the input δ_t is bounded in the interval $[0, 1]$. The full state is observed using an inertial navigation system (INS), which is hybridized with a global navigation satellite system (GNSS) receiver, a magnetometer and a barometer.

The measurement equation is given in Section IV where a stochastic model including the effects of sensor faults is presented. The observation function is denoted $\mathcal{H}_k$ and is given by:

$$\mathcal{H}_k(\mathbf{z}_k) = \begin{bmatrix} -\mathrm{p_d}_k & \mathrm{u}_k & \mathrm{w}_k & \theta_k & q_k \end{bmatrix}^\top \tag{3}$$

B. Longitudinal Autopilot

In the following, we assume that the desired trajectory to be followed by the UAV consists of a desired flight path angle γ^c and a desired velocity norm V^c that are used as reference inputs to the longitudinal control loop. The actuator inputs δ_e and δ_t are given by:

$$\begin{cases} \bar{\delta}_{e\,k+1} = -L_\theta \hat{\bar{\mathbf{z}}}_k - L_{\theta_i}\bar{\theta}_{i\,k+1} & \text{(4a)} \\ \bar{\delta}_{t\,k+1} = -L_\mathrm{u} \hat{\bar{\mathbf{z}}}_k - L_{\mathrm{u}_i}\bar{\mathrm{u}}_{i\,k+1} & \text{(4b)} \end{cases}$$

where the bar notation represents a variation around the trim condition[1]. The gains L_θ, L_u, L_{u_i}, L_{θ_i} are obtained after solving the Riccatti equation of the linear quadratic regulator (LQR) problem for the desired steady trim condition with integral correction and weighting matrices $\mathbf{Q} = \mathrm{diag}\left(\begin{bmatrix} 1 & 0 & 4 & 0 & 0 \end{bmatrix}\right)$, $\mathbf{R} = I_{2\times 2}$. The LQR approach is suboptimal for the nonlinear system, but the computed gains will be shown to achieve robust stability despite the nonlinearity of UAV dynamics, particularly when the initial condition does not lead to a severe coupling between rotational and translational dynamics. The integral gains are $L_{\theta_i} = 1.00$

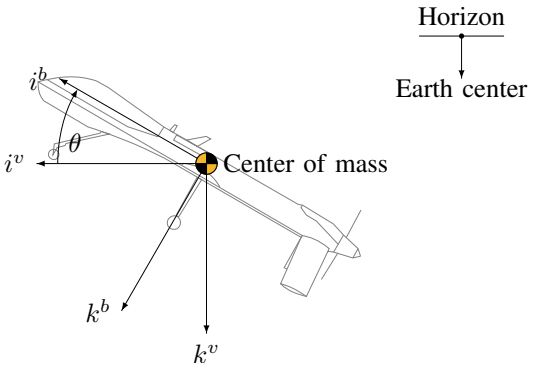


Fig. 1. Side view of an UAV with references axis and angles

[1] In our case a straight flight at an altitude of $500\,\mathrm{m}$ with an air speed of $40\,\mathrm{m\,s^{-1}}$

and $L_{\mathrm{u}_i} = -1.00$. Integrated state deviations $\bar{\theta}_i$ and $\bar{\mathrm{u}}_i$ are given by:

$$\begin{cases} \bar{\theta}_{ik+1} = (\bar{\gamma}_k^c + A_{\mathrm{u}}\hat{\tilde{\mathrm{u}}}_k + A_{\mathrm{w}}\hat{\tilde{\mathrm{w}}}_k - \hat{\bar{\theta}})dt + \bar{\theta}_{ik} & \text{(5a)} \\ \bar{\mathrm{u}}_{ik+1} = ((\bar{V}_k^c - V_{\mathrm{w}}\hat{\tilde{\mathrm{w}}}_k)\dfrac{1}{V_{\mathrm{u}}} - \hat{\tilde{\mathrm{u}}}_k)dt + \bar{\mathrm{u}}_{ik} & \text{(5b)} \end{cases}$$

where the parameters $V_{\mathrm{u}} = 1.00$, $V_{\mathrm{w}} = 0.05$, $A_{\mathrm{u}} = 0$ and $A_{\mathrm{w}} = 0.03$ relate pitch and speed reference guidance commands to γ^c and V^c.

III. Sensor Fault Types and Modes

As in [16], a fault is defined here as an "*unpermitted deviation of at least one characteristic property of the system*". In this paper, sensor outputs can be faulty and the fault states have two possible modes, a fault-free mode denoted $m^{(0)}$ and a faulty mode denoted $m^{(1)}$.

For the UAV model under consideration, the number of sensors denoted n_y is set to 5 without loss of generality but in the next section, the fault and state estimation algorithm presented here is written for a more general case where n_y depends on the application. Fault estimation is implemented for n_Θ sensors where $n_\Theta \leq n_y$ is user-defined depending on the sensors deemed likely to be faulty. Each sensor has an index $j \in [1,\, n_y]$, a fault variable $\Theta^j(m_k^j)$ and an associated fault mode m_k^j at time step k.

A fault and state estimation algorithm is developed here for the purpose of fault tolerant navigation, with no prior knowledge of fault dynamics or amplitudes.

The fault types applied to the system are additive and of unknown amplitude. Their dynamics are also unknown as the approach tackles abrupt or incipient faults. Exponential fault models are for example considered in the numerical simulation section, but the estimation algorithms use a zero-order fault model, which is initialized by default in mode $m^{(0)}$. Abrupt faults occur suddenly (stepwise), while incipient faults occur gradually and driftwise [12], [17].

The estimation algorithms also have to determine when the fault is no longer active. Both abrupt and incipient faults scenarios under consideration in this paper are therefore intermittent, such that the fault is deactivated after a certain time [12].

In the next section, a stochastic Markovian jump system model is used to represent the dependency of the nonlinear aircraft model on sensor faults.

IV. Stochastic Jump Markov Nonlinear System Model

A discrete-time stochastic Markovian jump system model is used to represent the dynamics of the system, including the transitions between fault modes. This generic system model is given by:

$$\begin{cases} \mathbf{m}_{k+1} \sim p\left(\mathbf{m}_{k+1}|\mathbf{m}_k\right) & \text{(6a)} \\ \mathbf{z}_{k+1} = \mathcal{F}_k\left(\mathbf{z}_k,\, \mathbf{u}_k\right) + \boldsymbol{\eta}_k & \text{(6b)} \\ \mathbf{y}_k = \mathcal{H}_k\left(\mathbf{z}_k\right) + \mathcal{G}_k\left(\boldsymbol{\Theta}_k\left(\mathbf{m}_k\right)\right) + \boldsymbol{\nu}_k & \text{(6c)} \end{cases}$$

where $\mathbf{z}_k \in \mathbb{R}^{n_z}$ represents the system state $\mathbf{u}_k \in \mathbb{R}^{n_u}$ is the control input, $\mathbf{y}_k \in \mathbb{R}^{n_y}$ is the vector of measurements. $\mathcal{F}_k\left(\cdot\right)$ represents the dynamics of the system (see (2)), $\mathcal{H}_k\left(\cdot\right)$ represents the measurement function (see (3)) and $\mathcal{G}_k\left(\cdot\right)$ represents the impacts of the sensor faults on the measurements. The process and sensors noises are $\boldsymbol{\eta}_k \in \mathbb{R}^{n_z}$ and $\boldsymbol{\nu}_k \in \mathbb{R}^{n_y}$. They are assumed to be of zero mean and the covariance matrices are respectively defined as $\mathbb{E}\left[\boldsymbol{\eta}_k\boldsymbol{\eta}_k^\top\right] = \mathbf{Q}_k$ and $\mathbb{E}\left[\boldsymbol{\nu}_k\boldsymbol{\nu}_k^\top\right] = \mathbf{R}_k$. They are assumed to be independent $\mathbb{E}\left[\boldsymbol{\eta}_k\boldsymbol{\nu}_k^\top\right] = 0$.

A sensor fault vector $\boldsymbol{\Theta}_k \in \mathbb{R}^{n_\Theta}$ at time step k is defined. It is a function of the fault mode vector $\mathbf{m}_k$, for which each element m_k^j where j is a sensor index can either be faulty $m^{(1)}$ or fault free $m^{(0)}$, at time step k. The number of sensors $n_\Theta \leq n_y$ for which fault diagnosis is applied is user-defined, but taken to be equal to one in Section VI, where the method is applied to a single sensor fault on the pitch measurement. To simplify notations, $\boldsymbol{\Theta}_k = \boldsymbol{\Theta}_k(\mathbf{m}_k)$ is used for the remainder of this article.

In this Markovian jump system, the mode switching between fault free and faulty modes is managed by a jump strategy, which is presented in Subsection IV-A

A. The Transition Probability Based Jump Strategy

At every time step k, the probability $\mathbb{P}\left(m_{k+1}^{(j)}|m_k^{(i)}\right)$ to switch from mode $m^{(i)}$ to $m^{(j)}$ is constant π_{ji}. Hence, π_{10} is the probability to switch from nominal mode $m^{(0)}$ to a faulty mode $m^{(1)}$ while the probability π_{01} is the probability to switch from a faulty mode $m^{(1)}$ to a nominal mode $m^{(0)}$. The transition probability matrix $\boldsymbol{\Pi}$ represents the probability of switching from one mode to another. It is given by:

$$\boldsymbol{\Pi} = \begin{bmatrix} \pi_{00} & \pi_{10} \\ \pi_{01} & \pi_{11} \end{bmatrix} \tag{7}$$

Each state of the fault vector $\boldsymbol{\Theta}$ is associated with a $\boldsymbol{\Pi}$ matrix. Note that for the numerical analysis in Section VI, a single scalar fault is considered without loss of generality and to focus the analysis on estimation performance for abrupt and incipient fault types, but the proposed approach applies to multiple observable sensor faults. The diagonal elements of the π_{jj} matrices represent probabilities to remain in the same mode for the given sensor.

The Markov chains can be represented by the transitions diagram shown in Fig. 2:

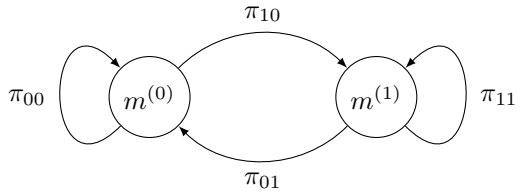


Fig. 2. Markov chain of the JMRPF applied to fault estimation

The objective of the method presented here is to simultaneously detect the occurrence of a fault and to estimate its amplitude. An extended state vector $\mathbf{x}_k \in \mathbb{R}^{n_x}$ is introduced and given by:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{z}_k^\top & \boldsymbol{\Theta}_k^\top \end{bmatrix}^\top \tag{8}$$

which extends the original system state $\mathbf{z}_k$ to account for the fault values. The system model (6) can then be written as an extended state model:

$$\begin{cases} \mathbf{x}_{k+1} = f_k\left(\mathbf{x}_k,\, \mathbf{u}_k\right) + \boldsymbol{\eta}_k & \text{(9a)} \\ \mathbf{y}_k = h_k\left(\mathbf{x}_k\right) + \boldsymbol{\nu}_k & \text{(9b)} \end{cases}$$

where $\boldsymbol{\eta}_k \in \mathbb{R}^{n_x}$ now represents the process noise of the extended state space model. The dynamics of the system and measurements functions f_k and h_k are given by:

$$f_k\left(\mathbf{x}_k,\, \mathbf{u}_k\right) = \begin{bmatrix} \mathcal{F}_k\left(\mathbf{z}_k,\, \mathbf{u}_k\right) \\ \mathcal{T}_k\left(\boldsymbol{\Theta}_k\right) \end{bmatrix} \quad \text{(10a)}$$

$$h_k\left(\mathbf{x_k}\right) = \mathcal{H}_k\left(\mathbf{z}_k\right) + \mathcal{G}_k\left(\boldsymbol{\Theta}_k\right) \quad \text{(10b)}$$

where $\mathcal{T}_k\left(\cdot\right)$ represents the dynamics of the sensor fault.

For this representation of the problem under consideration, estimation of $\mathbf{x_k}$ requires nonlinear filtering techniques. As introduced previously, jump Markov based filters such as the JMRPF are well-suited to handle such systems.

V. Jump Markov regularized particle filter

The proposed JMRPF combines the RPF [9] with a new jump Markov strategy to manage transitions between faulty and fault free modes using a transition probability matrix. As in a RPF, particles are resampled from a continuous approximation to the posterior density to reduce degeneracy without compromising the diversity of solutions. A first version of a JMRPF was introduced in [15] to recover from ambiguous abrupt sensor fault scenarios. In this paper, a Kalman correction is introduced to correct particles states by placing them in more likely state space regions. The Kalman innovation covariance matrix $\mathbf{S}_k$ is taken into consideration in the calculation of the particles weights. This reduces the variance of the weights. The estimated state vector $\mathbf{x}_k$ and the estimated covariance matrix $\hat{\mathbf{P}}_k$ are obtained by the JMRPF algorithm using estimated state feedback, previous estimated state and the measurement as inputs to the filter. The total number of particles is denoted N_p.

The proposed JMRPF algorithm is introduced in Algorithm 1. The algorithm has prediction, update, estimation and regularization-resampling steps. The PREDICT and UPDATE functions of Algorithm 1 are improved compared to [15] to deal with sensor fault estimation and a JUMP function is introduced within the prediction step.

Algorithm 1 jump Markov regularized particle filter

$k \leftarrow 0$
$\vdots$ ▷ Initialization
loop
  $k \leftarrow k + 1$
  for each $i \in [1,\, N_p]$ **do**
    PREDICT($\mathbf{x}^i_{k|k-1}$, $\mathbf{x}^i_{k-1}$, $\mathbf{m}^i_k$, $\mathbf{u}_k$, $\mathbf{y}_k$)
  end for
  ESTIMATE($\hat{\mathbf{x}}_{k|k-1}$, $\hat{\mathbf{P}}_{k|k-1}$, $w^{1:N_p}_{k-1}$, $\mathbf{x}^{1:N_p}_{k|k-1}$)
  $\hat{\mathbf{y}}_{k|k-1} \leftarrow \sum_{i=1}^{N_p} w^i_{k-1} h_k\left(\mathbf{x}^i_{k|k-1}\right)$
  $\mathbf{S}_k \leftarrow \sum_{i=1}^{N_p} w^i_{k-1}\left(\mathbf{y}^i_{k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right)\left(\mathbf{y}^i_{k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right)^\top + \mathbf{R}_k$
  $\hat{\mathbf{P}}_{XY} \leftarrow \sum_{i=1}^{N_p} w^i_{k-1}\left(\mathbf{x}^i_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right)\left(\mathbf{y}^i_{k|k-1} - \hat{\mathbf{y}}_{k|k-1}\right)^\top$
  $\mathbf{K}_k \leftarrow \hat{\mathbf{P}}_{XY}\mathbf{S}_k^{-1}$ ▷ Kalman gain
  for each $i \in [1,\, N_p]$ **do**
    UPDATE($\mathbf{x}^i_k$, w^i_k, w^i_{k-1}, $\mathbf{x}^i_{k|k-1}$, $\mathbf{K}_k$, $\mathbf{S}_k$, $\mathbf{y}_k$)
  end for
  ESTIMATE($\hat{\mathbf{x}}_k$, $\hat{\mathbf{P}}_k$, $w^{1:N_p}_k$, $\mathbf{x}^{1:N_p}_k$)
  $N_{eff} \leftarrow \frac{1}{\sum_{i=1}^{N_p} {w^i_k}^2}$
  if $N_{eff} \leq N_p\Gamma$ **then** ▷ if true then resample
    MULTINOMIAL($\acute{\mathbf{x}}^{1:N_p}_k$, $\mathbf{x}^{1:N_p}_k$, $w^{1:N_p}_k$)
    for each $i \in [1,\, N_p]$ **do**
      $w^i_k \leftarrow \frac{1}{N_p}$ ▷ Reset the weights
      REGULARIZE($\mathbf{x}^i_k$, $\acute{\mathbf{x}}^i_k$)
    end for
  end if
end loop

A. Prediction Step

The prediction step is described in Algorithm 2. The i^{th} state variable is propagated using the following probability transition density for the state $\mathbf{x}_k$:

$$\mathbf{x}^i_{k|k-1} \sim p\left(\mathbf{x}_{k|k-1}|\mathbf{x}^i_{k-1}, \mathbf{m}^i_k\right) \quad \text{(11)}$$

Then, one obtains a predicted cloud of particles ($\mathbf{x}^1_{k|k-1}$, $\mathbf{x}^2_{k|k-1}$, $\cdots$, $\mathbf{x}^{N_p}_{k|k-1}$). The mode of each state and particles of $\boldsymbol{\Theta}_k$ are updated in the prediction step. This update is performed here using a uniform law and compared to user-defined probabilities π_{ji} to switch from one mode to another as described in Fig. 2.

To simplify notations in this section, it is assumed that the maximum number of possible sensor faults n_Θ is equal to the number of sensors n_y, although this number can easily be reduced in practice to restrict fault management to specific sensors. The same index j is also used to denote the j^{th} state of vector $\boldsymbol{\Theta}_k$, representing fault estimates of the j^{th} sensor of $\mathbf{y_k}$. The jump amplitude of the i^{th} particle of the j^{th} state of $\boldsymbol{\Theta}$ at time step k is computed as follows:

$$\Theta^{i,j}_{k|k-1} = \begin{cases} \beta^{i,j}_k & \text{if } U \leq \pi^j_{10} \text{ and } \mathrm{m}^{i,j}_k = m^{(0)} \\ \Theta^{i,j}_{k|k-1} & \text{if } U < \pi^j_{11} \text{ and } \mathrm{m}^{i,j}_k = m^{(1)} \\ 0 & \text{else} \end{cases} \quad \text{(12)}$$

where $U \sim \mathcal{U}\left(0,\, 1\right)$ and $\boldsymbol{\beta}^i_k$ is given by:

$$\boldsymbol{\beta}^i_k = \mathbf{y}_k - h_k\left(\mathbf{x}^i_{k|k-1}\right) \quad \text{(13)}$$

A new jump strategy for sensors fault modes is proposed. It uses a priori distribution of the fault, which is computed using sensor innovation terms from (13). Irrespective of the current estimated mode, the alternate mode will continue to be tested using a small subset of fault state particles that will be called sentinel particles. Those particles are selected from a uniform distribution and their number are smaller

Algorithm 2 Detail of the function PREDICT from Algorithm 1

function PREDICT($\mathbf{x}^i_{k|k-1}$, $\mathbf{x}^i_{k-1}$, $\mathbf{m}^i_k$, $\mathbf{u}_k$, $\mathbf{y}_k$)
- $\boldsymbol{\eta}^i_k \sim \mathcal{N}(0, \mathbf{Q}_k)$
- $\mathbf{x}^i_{k|k-1} \leftarrow f_k\left(\mathbf{x}^i_{k-1}, \mathbf{u}_k\right) + \boldsymbol{\eta}^i_k$
- $\boldsymbol{\beta}^i_k \leftarrow \mathbf{y}_k - h_k\left(\mathbf{x}^i_{k|k-1}\right)$ ▷ See (13)
- **for each** $j \in [1, n_\Theta]$ **do** ▷ Jump step of $\boldsymbol{\Theta}$
  - JUMP($\Theta^{i,j}_{k|k-1}$, $\mathrm{m}^{i,j}_k$, $\beta^{i,j}_k$) ▷ See Algorithm 3
- **end for**

end function

when the off-diagonal terms π_{01} and π_{10} of the transition probability matrix are smaller. Those terms are set based on empirical sensor false alarm and missed detection rates. This small number of sentinel particles will continuously test the probability of transition to the alternate fault modes. The use of a small number of sentinel particles to test the alternate mode enhances real time operation prospects compared to previously published particle filter jump strategies, where the total or an arbitrary number of particles is used to evaluate the probabilities of both modes before any jump between them [8].

The JUMP function used in Algorithm 2 is described in Algorithm 3.

Algorithm 3 Detail of the function JUMP from Algorithm 2

function JUMP($\Theta^{i,j}_{k|k-1}$, $\mathrm{m}^{i,j}_k$, $\beta^{i,j}_k$)
- $U \sim \mathcal{U}(0, 1)$
- **if** $\mathrm{m}^{i,j}_k = m^{(0)}$ **then** ▷ $\Theta^{i,j}_{k|k-1}$ in mode $m^{(0)}$
  - **if** $U \leq \pi_{10}$ **then** ▷ Transition $m^{(0)} \rightarrow m^{(1)}$
    - $\Theta^{i,j}_{k|k-1} \leftarrow \beta^{i,j}_k$
    - $\mathrm{m}^{i,j}_k \leftarrow m^{(1)}$
  - **else** ▷ Transition $m^{(0)} \rightarrow m^{(0)}$
    - $\Theta^{i,j}_{k|k-1} \leftarrow 0$
  - **end if**
- **else if** $\mathrm{m}^{i,j}_k = m^{(1)}$ **then** ▷ $\Theta^{i,j}_{k|k-1}$ in mode $m^{(1)}$
  - **if** $U \leq \pi_{01}$ **then** ▷ Transition $m^{(1)} \rightarrow m^{(0)}$
    - $\Theta^{i,j}_{k|k-1} \leftarrow 0$
    - $\mathrm{m}^{i,j}_k \leftarrow m^{(0)}$
  - **end if**
- **end if**

end function

B. Update Step

The update step is described in Algorithm 4. The i^{th} particle $\mathbf{x}^i_k$ is assigned to a weight w^i_k that is proportional to its likelihood:

$$\tilde{w}^i_k = w^i_{k-1} p\left(\mathbf{y}_k | \mathbf{x}^i_{k|k-1}, \mathbf{m}^i_k\right) \tag{14a}$$

$$w^i_k = \frac{\tilde{w}^i_k}{\sum_{i=1}^{N_p} \tilde{w}^i_k} \tag{14b}$$

In (14b) a normalization is applied to ensure that $\sum_{i=1}^{N_p} w^i_k = 1$.

Compared to the update step described in [15], an additional feature was introduced. Indeed, a Kalman update on the particles $\mathbf{x}^i_{k|k-1}$ is applied to place the particles in more likely regions of the state space, which is shown in numerical simulations to enhance estimation performance accuracy. The Kalman update is given by:

$$\mathbf{x}^i_k = \mathbf{x}^i_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}^i_k \tag{15}$$

At this step, the likelihood is assumed to follow a Gaussian distribution.

Algorithm 4 Detail of the function UPDATE from Algorithm 1

function UPDATE($\mathbf{x}^i_k$, w^i_k, w^i_{k-1}, $\mathbf{x}^i_{k|k-1}$, $\mathbf{K}_k$, $\mathbf{S}_k$, $\mathbf{y}_k$)
- $\tilde{\mathbf{y}}^i_k \leftarrow \mathbf{y}_k - h_k\left(\mathbf{x}^i_{k|k-1}\right)$ ▷ Innovation
- $\tilde{w}^i_k \leftarrow w^i_{k-1} \mathcal{N}(\tilde{\mathbf{y}}^i_k; 0, \mathbf{S}_k)$ ▷ See (14a)
- $w^i_k \leftarrow \frac{\tilde{w}^i_k}{\sum_{i=1}^{N_p} \tilde{w}^i_k}$
- $\mathbf{x}^i_k \leftarrow \mathbf{x}^i_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}^i_k$ ▷ See (15)

end function

C. Estimation

The estimation step aims to perform a global estimation of the state vectors $\hat{\mathbf{x}}_k$ and $\hat{\mathbf{x}}_{k|k-1}$, with its associated covariance matrices $\hat{\mathbf{P}}_k$ and $\hat{\mathbf{P}}_{k|k-1}$ respectively.

This step is described in Algorithm 5.

Algorithm 5 Detail of the function ESTIMATE from Algorithm 1

function ESTIMATE($\hat{\mathbf{x}}$, $\hat{\mathbf{P}}$, $\mathbf{x}^{1:N_p}$, $w^{1:N_p}$)
- $\hat{\mathbf{x}} \leftarrow \sum_{i=1}^{N_p} w^i \mathbf{x}^i$
- $\hat{\mathbf{P}} \leftarrow \sum_{i=1}^{N_p} w^i \left(\mathbf{x}^i - \hat{\mathbf{x}}\right)\left(\mathbf{x}^i - \hat{\mathbf{x}}\right)^\top$

end function

D. Regularization-Resampling Step

This step consists of two stages, the resampling and the regularization of the selected particles. Its purpose is to remove the particles with a low likelihood and to replace them by duplicating the particles with a high likelihood and regularizing the duplicated particles.

a) Resampling step: In the MULTINOMIAL function of algorithm 1, the particles are selected according to a multinomial law with w^i_k as parameter. Then the probability to choose a particle is:

$$\mathbb{P}\left(\acute{\mathbf{x}}^j_k = \mathbf{x}^i_k\right) = w^i_k \tag{16}$$

b) Regularization step: The regularization step is described in Algorithm 6. The particles are randomly moved according to a regularization kernel $\mathcal{K}(\mathbf{x})$. The regularization equation is:

$$\mathbf{x}_k^i = \acute{\mathbf{x}}_k^i + h\mathbf{D}_k\boldsymbol{\varepsilon}_k^i \tag{17}$$

where $h \in \mathbb{R}^{+*}$ is the bandwidth factor in the re-scaled kernel density $\mathcal{K}(\cdot)$ and with $\mathbf{P}_k = \mathbf{D}_k\mathbf{D}_k^\top$ and $\boldsymbol{\varepsilon} \sim \mathcal{K}(\mathbf{x})$. The kernel density is a symmetric probability density function such that:

$$\int \mathbf{x}\mathcal{K}(\mathbf{x})\, d\mathbf{x} = 0, \quad \int \|\mathbf{x}\|^2\mathcal{K}(\mathbf{x})\, d\mathbf{x} < \infty \tag{18}$$

The optimal kernel $\mathcal{K}(\cdot)$ and bandwidth factor h are chosen to minimize the mean integrated square error (MISE) between the theoretical and estimated posterior density, and is given by:

$$\text{MISE}(\hat{p}) = \mathbb{E}\left[\int \left(\hat{p}(\mathbf{x}_k|\mathbf{Y}_{1:k}) - p(\mathbf{x}_k|\mathbf{Y}_{1:k})\right)^2 d\mathbf{x}_k\right] \tag{19}$$

where $\hat{p}(\mathbf{x}_k|\mathbf{Y}_{1:k})$ is the JMRPF approximation of the conditional density. The Epanechnikov kernel is used as a regularization kernel [18].

$$\mathcal{K}(\mathbf{x}) = \begin{cases} \frac{n_x+2}{2c_{n_x}}\left(1 - \|\mathbf{x}\|^2\right) & \text{if} \quad \|\mathbf{x}\| < 1 \\ 0 & \text{else} \end{cases} \tag{20}$$

where c_{n_x} is the volume of the unit hypersphere in $\mathbb{R}^{n_x}$.

Algorithm 6 Detail of the function REGULARIZE from Algorithm 1

function REGULARIZE($\mathbf{x}_k^i$, $\acute{\mathbf{x}}_k^i$)
  $\boldsymbol{\varepsilon}_k^i \sim \mathcal{K}\left(\mathbf{x}_k^i\right)$ ▷ see (20)
  $\mathbf{x}_k^i \leftarrow \acute{\mathbf{x}}_k^i + h\mathbf{D}_k\boldsymbol{\varepsilon}_k^i$
end function

The resampling step is only performed if N_{eff}/N_p is lower than a user-defined threshold $\Gamma \in (0;\, 1)$, where the efficiency N_{eff} is given by:

$$N_{\mathit{eff}} = \frac{1}{\sum_{i=1}^{N_p} {w_k^i}^2} \tag{21}$$

The conditional density is then approached by:

$$p(\mathbf{x}_k, \mathbf{m}_k|\mathbf{Y}_{1:k}) \approx \sum_{i=1}^{N_p} w_k^i \mathcal{K}_h\left(\mathbf{x}_k - \mathbf{x}_k^i\right)\delta_{\mathbf{m}_k^i}(\mathbf{m}_k) \tag{22}$$

where:

$$\mathcal{K}_h(\mathbf{x}_k) = \frac{1}{h^{n_x}}\mathcal{K}\left(\frac{1}{h}\mathbf{x}_k\right) \tag{23}$$

VI. Simulation Results and Analysis

The application example is the fixed-wing UAV described in Section II. A pitch sensor fault in an inertial navigation system is assumed. The UAV is initialized in straight level flight with an initial velocity of $40\,\mathrm{m\,s^{-1}}$ and initial altitude of $500\,\mathrm{m}$. The simulation time step is $50\,\mathrm{ms}$.

During all the simulation, the desired flight path angle is set to $\gamma^c = 0°$ and the desired norm of the velocity vector is $V^c = 40\,\mathrm{m\,s^{-1}}$.

A. Filter Parameters

For the two filters compared, the JMRPF and the RPF the stochastic process model of the filter is given by:

$$\begin{cases} \begin{bmatrix}\mathbf{z}_{k+1} \\ \boldsymbol{\Theta}_{k+1}\end{bmatrix} = \begin{bmatrix}\mathcal{F}_k(\mathbf{z}_k,\, \mathbf{u}_k) \\ \boldsymbol{\Theta}_k\end{bmatrix} + \boldsymbol{\eta}_k & \text{(24a)} \\ \mathbf{y}_k = \mathcal{H}_k(\mathbf{z}_k) + \mathcal{G}_k(\boldsymbol{\Theta}_k) + \boldsymbol{\nu}_k & \text{(24b)} \end{cases}$$

where $\mathcal{F}_k(\cdot)$ is the longitudinal dynamic of the UAV given by (2), and $\mathcal{H}_k(\cdot)$ is the measurement function given by (3). As previously explained, a zero-order fault model is used by the filter, but the actual dynamical model of the fault does not always match the fault model of the filter. The process noise $\boldsymbol{\eta}_k$ and measurement noise $\boldsymbol{\nu}_k$ are both Gaussian and independent with a zero mean and their covariance matrices are respectively $\mathbf{Q}_k$ and $\mathbf{R}_k$. The impact of the sensor fault on the state and measurements is given by:

$$\mathcal{G}_k(\boldsymbol{\Theta}_k) = \begin{bmatrix}0 & 0 & 0 & \boldsymbol{\Theta}_k & 0\end{bmatrix}^\top \tag{25}$$

The JMRPF and RPF parameters are[2]:

- The standard deviation vector used to compute the covariance matrix $\mathbf{P}_0 = \mathrm{diag}\left(\boldsymbol{\sigma}_0^2\right)$ for the extended state vector $\mathbf{x}_k$ is given by:

$$\boldsymbol{\sigma}_0 = \begin{bmatrix}1 & 1 & 1 & 0.3 & 0.1 & 0.3\end{bmatrix} \tag{26}$$

- The standard deviation vectors used to compute the covariance matrices $\mathbf{Q}_k = \mathrm{diag}\left(\boldsymbol{\sigma}_Q^2\right)$ and $\mathbf{R}_k = \mathrm{diag}\left(\boldsymbol{\sigma}_R^2\right)$ are respectively given by:

$$\boldsymbol{\sigma}_Q = \begin{bmatrix}0.1 & 0.1 & 0.1 & 0.03 & 0.01 & 0.1\end{bmatrix} \tag{27}$$

$$\boldsymbol{\sigma}_R = \begin{bmatrix}1 & 1 & 1 & 0.3 & 0.1\end{bmatrix} \tag{28}$$

- The resampling threshold Γ and regularization bandwidth h are:

$$\Gamma = 0.5 \tag{29}$$

$$h = 0.2817 \tag{30}$$

- The number of particles N_p is set to 1000.
- The transition probability matrix for sensor faults (JMRPF only) is equal to:

$$\boldsymbol{\Pi} = \begin{bmatrix}0.99 & 0.01 \\ 0.01 & 0.99\end{bmatrix} \tag{31}$$

  Selection of the transition probability matrix $\boldsymbol{\Pi}$ was driven by the expected device false alarm probability.

B. Fault Scenario

The simulation scenario was chosen to illustrate the efficiency of the JMRPF and RPF algorithms to estimate intermittent pitch sensor faults with unknown amplitude and unknown dynamics. To assess those two aspects of estimation performance, the fault scenario has two phases. On the first phase, an abrupt bias type sensor fault is injected with unexpected amplitude starting at $t = 10\,\mathrm{s}$ and ending at $t = 20\,\mathrm{s}$. On the second phase, the fault also has unexpected dynamics, starting from $t = 30\,\mathrm{s}$ until $t = 40\,\mathrm{s}$.

[2]Note: The angle unit used is degree

1) Unexpected Amplitude: Estimation performance is assessed for faults amplitudes much larger than process and regularization noises. To ensure a fair comparison, the process and regularization noises of the fault state are taken to be the same for both filter and are given by the 6^{th} element of (27) and (30). The abrupt amplitude fault follows zero-order dynamics, as expected by both filters where the assumed fault dynamics are given by (24a). The fault estimates are initialized at zero. The amplitude of the pitch fault is taken to be $5°$ which is significant compared to the process and regularization noises of both filters. The fault equation for this phase is given by:

$$\boldsymbol{\Theta}(t) = 5 \tag{32}$$

2) Unexpected Fault Dynamics: When the fault has unexpected dynamics, both filters continue to assume zero-order filter dynamics. To compare the effect of unexpected fault dynamics on the two filters, an incipient fault with an exponential dynamic and a maximum amplitude of $10°$ is considered for this phase. The actual sensor fault equation for this phase is given by:

$$\boldsymbol{\Theta}(t) = 10\exp(t-40) \tag{33}$$

The full sensor fault sequence with both phases is illustrated in Fig. 3.

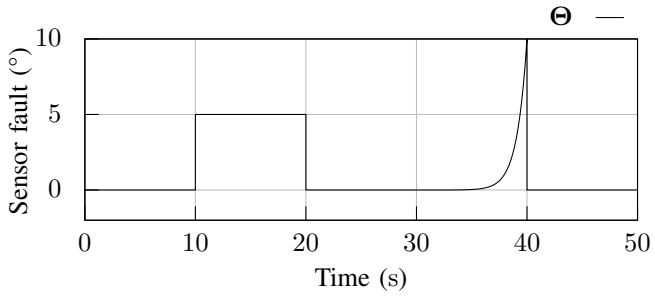


Fig. 3. Faults scenario used for the simulation

C. Estimation Performance Comparison

In this section, the JMRPF and RPF are compared in terms of root-mean-square error (RMSE) and of median fault and state estimation for $N_{\text{MC}} = 100$ Monte Carlo runs. The RMSE is defined for any state variable x as:

$$\text{RMSE}_k^x = \sqrt{\frac{\sum_{i=1}^{N_{\text{MC}}} (\hat{x}_k^i - x_k^i)^2}{N_{\text{MC}}}} \tag{34}$$

where x_k^i and $\hat{x}_k^i$ respectively represent the true and estimated state variable at time step k for the i^{th} Monte Carlo run and x is a generic notation that could for example represent pitch θ or altitude p_d. True states depend on the Monte Carlo run because the autopilot uses estimated state feedback. The average RMSE over the total simulation time is also used as an estimation performance metric and is given by:

$$\overline{\text{RMSE}}_x = \frac{\sum_{k=1}^{N_s} \text{RMSE}_k^x}{N_s} \tag{35}$$

where N_s is the total number of time steps for the simulation.

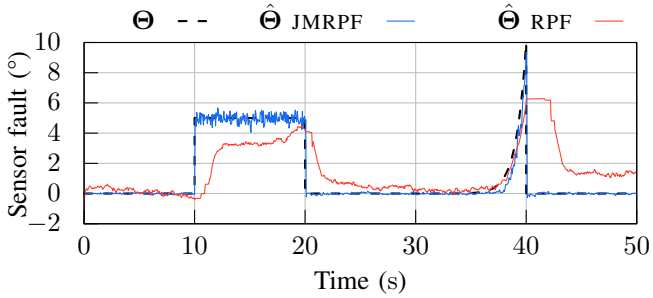


Fig. 4. Median result of the estimated pitch sensor fault

In Fig. 4, both filters estimate the abrupt fault between $10\,\text{s}$ and $20\,\text{s}$ and return to the fault free mode between $20\,\text{s}$ and $30\,\text{s}$ when the fault is no longer active. The JMRPF has higher accuracy and significantly lower convergence time compared to the RPF during both fault free and faulty phases When the transient fault phase starts at $t = 30\,\text{s}$, the JMRPF estimates the unexpected exponential fault more and more accurately than the RPF as time approaches $40\,\text{s}$. At that time, the fault is no longer active and the JMRPF returns to the fault free mode almost instantaneously and with very high estimation accuracy compared to the RPF that converges much more slowly to the fault free mode. The jump strategy of the JMRPF was indeed designed to return more rapidly to the correct mode. Moreover, between $40\,\text{s}$ and $42.55\,\text{s}$, the RPF does not converge. This non convergence is due to the fact that the estimated fault amplitude of approximately $6.12°$ at those times is high compared to the process and regularization noises. The JMRPF handles this abrupt change more efficiently.

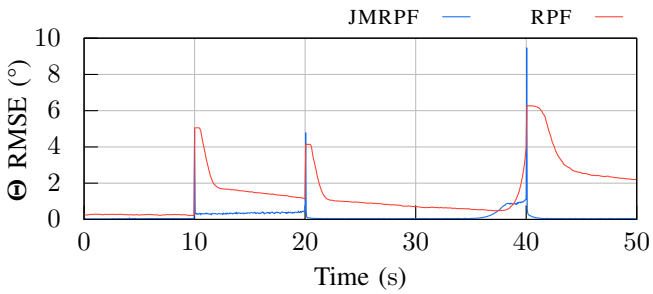
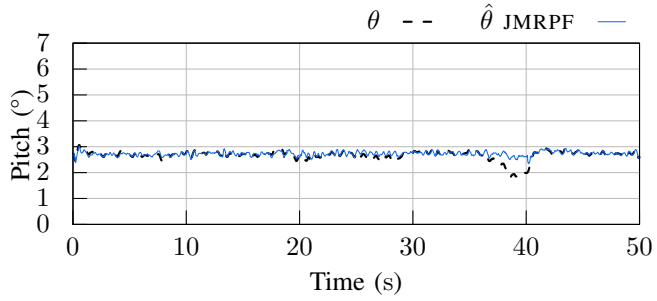


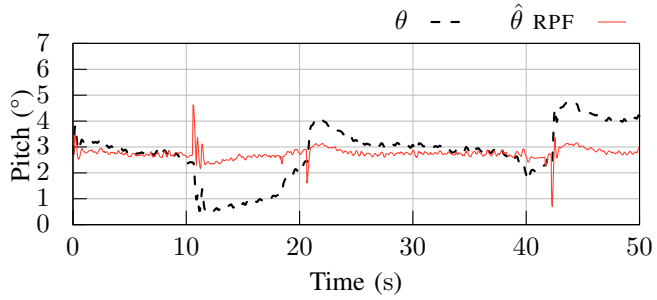
Fig. 5. RMSE of the pitch sensor fault

In Fig. 5, the RMSE is indeed shown to be significantly lower with the JMRPF at all times except for a brief $1.35\,\text{s}$

interval of time at $37.5\,\mathrm{s}$ when the exponential fault has not yet compensated the slow convergence dynamics of the RPF. The RMSE is only very briefly higher with the JMRPF at $t = 20\,\mathrm{s}$ and $t = 40\,\mathrm{s}$ before a sharp decrease to substantially smaller errors. This is due to the fact that the fault is suddenly activated and deactivated and during one time step, there is an error between the current fault estimate and the new fault.



(a) Estimated pitch with the JMRPF



(b) Estimated pitch with the RPF

Fig. 6. Comparison of the median results for estimated pitch

In Fig. 6, the pitch estimation error is also shown to be significantly smaller with the JMRPF from $t = 10\,\mathrm{s}$ until the end of the simulation. Maximum error is below $0.74°$ with the JMRPF at $38.65\,\mathrm{s}$ and exceeds $2.23°$ with the RPF at $10.60\,\mathrm{s}$.

Note that the true pitch signal differs between the two simulations because estimated pitch is used by the LQR feedback controller with integral action, which was presented in the control system Subsection II-B.

The JMRPF also provides significantly higher altitude estimation accuracy compared to the RPF, as shown in Fig. 7. This is particularly the case for the duration of the faults and when the faults are deactivated. True altitudes are different between the two filters because the feedback control scheme uses state estimation feedback.

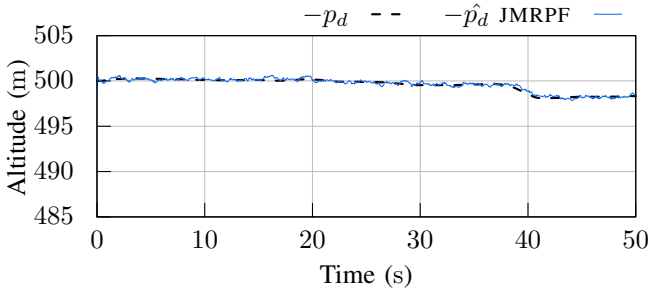
The controller is designed to maintain a zero flight path angle. Fault estimation errors have a visible impact on both filters as also shown in Fig. 7 when the altitude tracking accuracy temporarily deteriorates. The same figure also shows that the altitude fluctuations are less significant with the JMRPF compared to the RPF. This is due to the fact that the JMRPF has higher estimation accuracy. The non-convergence of the RPF is also visible between $40\,\mathrm{s}$ and $42.55\,\mathrm{s}$ when the estimated altitude does not follow the true altitude. For this scenario, the average altitude RMSE was

TABLE I

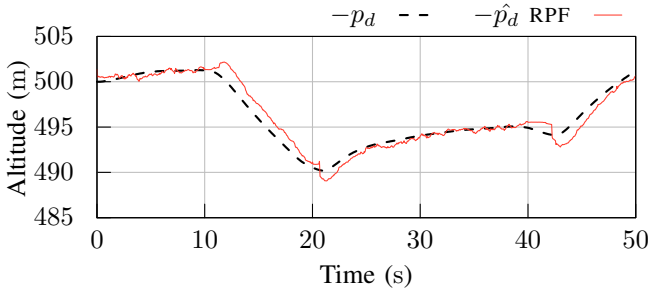
RMSE RESULTS FOR THE SCENARIO WITH $10°$ FAULT AMPLITUDE

Times (s)	10	21	30	41	$\overline{\text{RMSE}}$
Altitude					
RMSE JMRPF (m)	0.201	0.215	0.203	0.239	0.215
RMSE RPF (m)	0.223	1.278	0.527	1.246	0.932
Pitch					
RMSE JMRPF (°)	0.061	0.082	0.044	0.123	0.114
RMSE RPF (°)	0.208	0.884	0.704	0.899	1.014

found to be $0.215\,\mathrm{m}$ with the JMRPF and $0.932\,\mathrm{m}$ with the RPF.This average RMSE is therefore reduced by $77\,\%$ by the JMRPF and the pitch RMSE is also reduced by $89\,\%$, as shown in as shown in Table I.



(a) Estimated altitude with the JMRPF



(b) Estimated altitude with the RPF

Fig. 7. Comparison of the median results for estimated altitude

The amplitude used for this simulation scenario has been bound by the constraints on RPF parameters tuning. The difference between the convergence characteristics of the JMRPF and the RPF is more pronounced when larger amplitudes are considered. To illustrate this, the same simulation has been run with fault amplitudes 10 times larger than the one previously introduced. The filters parameters remains unchanged.

In Fig. 8, the fault estimation performance of the JMRPF is not significantly affected by a tenfold increase in fault amplitude, without changing the JMRPF parameters. On the other hand, the RPF was unable to handle this fault amplitude and does not converge to the abrupt fault, and at $38\,\mathrm{s}$ the incipient fault is also too steep for it to converge.

In Fig. 9, the RMSE of the pitch sensor fault of the RPF is also shown not to converge when the amplitude of the

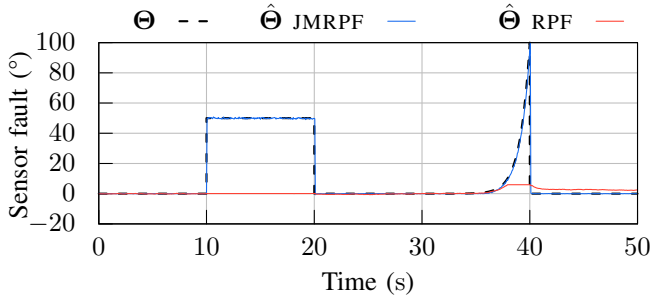


Fig. 8. Median result of the estimated pitch sensor fault with 10 times larger fault amplitudes

fault is too large compared to its process and regularization noises. Indeed, with no a priori knowledge of the fault amplitude, process noise cannot be adjusted to increase when the unknown fault amplitude increases. An arbitrarily large process noise would also degrade performance in the fault free case or when the fault is small. That is why the same moderate value of process noise was adopted for both RPF and JMRPF. Using the JMRPF, the jump Markov and Kalman correction compensate for the lack of process and regularization noises when fault amplitudes are high and drive the particles towards the high-likelihood regions. This promotes the particles' diversity and reduces particle degeneracy and sample impoverishment issues, and thus enhances the ability to track high amplitude abrupt and incipient faults. For the JMRPF, this figure illustrates its capacity to handle a variety of fault amplitudes and dynamics despite them being significantly different from the zero-order fault model used by the filter.

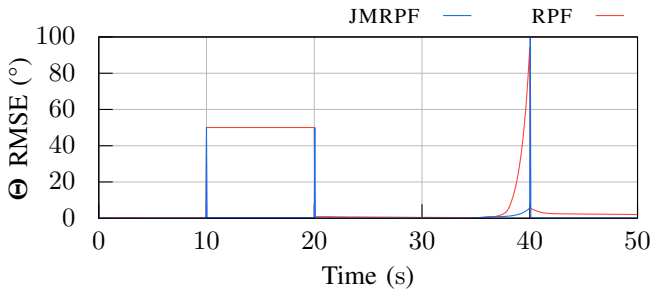


Fig. 9. RMSE of the pitch sensor fault with 10 times larger fault amplitudes

VII. Conclusion

A new nonlinear filter that combines regularized particle filtering with a jump Markov strategy and a Kalman correction was used in this paper to estimate abrupt and incipient sensor faults with unexpected dynamics and amplitudes. The approach was applied to state and fault estimation for a nonlinear longitudinal fixed wing unmanned aerial vehicle (UAV) model with a pitch sensor fault in the inertial navigation system. Numerical simulations have shown that the jump Markov regularized particle filter (JMRPF) significantly outperforms the regularized particle filter (RPF) with a higher accuracy and a faster convergence to both faulty and fault free modes. State estimation accuracy is also significantly enhanced with better robustness to faults using the JMRPF, particularly when the sensor fault is active and when the system switches back to a fault free mode. The simulation results also illustrate that the JMRPF has a much higher ability to estimate larger fault amplitudes compared to the RPF which is better suited to small fault amplitudes with slower dynamics. This presents a practical benefit for a fixed wing UAV with no sensor redundancy to maintain a steady flight in the presence of sensor faults with unexpected dynamics and amplitudes.

References

[1] S. Wang, X. Zhan, Y. Zhai, and B. Liu, "Fault detection and exclusion for tightly coupled gnss/ins system considering fault in state prediction," *Sensors*, vol. 20, no. 3, 2020.

[2] J. Marzat, H. Piet-Lahanier, F. Damongeot, and E. Walter, "Model-based fault diagnosis for aerospace systems: a survey," *IMechE Journal of Aerospace Engineering: Part G*, vol. 226, pp. 1329–1360, 2011.

[3] L. Eykeren, Q. Chu, and J. Mulde, "Sensor fault detection and isolation using adaptive extended kalman filter," *IFAC Proceedings Volumes*, vol. 45, no. 20, pp. 1155–1160, 2012.

[4] G. Alcalay, C. Seren, G. Hardier, M. Delporte, and P. Goupil, "An adaptive extended kalman filter for monitoring and estimating key aircraft flight parameters," *IFAC-PapersOnline*, vol. 51, no. 24, pp. 620–627, 2018.

[5] C. Rago, R. Prasanth, R. K. Mehra, and R. Fortenbaugh, "Failure detection and identification and fault tolerant control using the imm-kf with applications to the eagle-eye uav," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171)*, vol. 4. IEEE, 1998, pp. 4208–4213.

[6] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[7] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump markov linear systems," *IEEE Transactions on signal processing*, vol. 49, no. 3, pp. 613–624, 2001.

[8] S. Tafazoli and X. Sun, "Hybrid system state tracking and fault detection using particle filters," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 6, pp. 1078–1087, Nov 2006.

[9] C. Musso, N. Oudjane, and F. Le Gland, *Improving Regularised Particle Filters*. New York, NY: Springer New York, 2001, pp. 247–271.

[10] R. W. Beard and T. W. McLain, *Small unmanned aircraft: Theory and practice*. Princeton university press, 2012.

[11] S. X. Ding, *Model-based fault diagnosis techniques: design schemes, algorithms, and tools*. Springer Science & Business Media, 2008.

[12] A. Varga, *Solving fault diagnosis problems*. Springer, 2017.

[13] J. Chen and R. J. Patton, *Robust model-based fault diagnosis for dynamic systems*. Springer Science & Business Media, 2012, vol. 3.

[14] I. Castillo, T. F. Edgar, and B. R. Fernández, "Robust model-based fault detection and isolation for nonlinear processes using sliding modes," *International Journal of Robust and Nonlinear Control*, vol. 22, no. 1, pp. 89–104, 2012.

[15] E. Iglésis, K. Dahia, H. Piet-Lahanier, N. Merlinge, N. Horri, and J. Brusey, "A jump-markov regularized particle filter for the estimation of ambiguous sensor faults," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 756–763, 2020, 21th IFAC World Congress.

[16] R. Isermann, *Terminology in fault detection and diagnosis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 321–323.

[17] ——, *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media, 2005.

[18] B. W. Silverman, *Density estimation for statistics and data analysis*. Routledge, 2018.