Simultaneous Actuator and Sensor Faults Estimation for Aircraft Using a Jump-Markov Regularized Particle Filter

Iglésis, E., Horri, N., Brusey, J., Dahia, K. & Piet-Lahanier, H.

Original citation & hyperlink:
Iglésis, E, Horri, N, Brusey, J, Dahia, K & Piet-Lahanier, H 2021, Simultaneous Actuator and Sensor Faults Estimation for Aircraft Using a Jump-Markov Regularized Particle Filter. in 2021 IEEE International Conference on Prognostics and Health Management (ICPHM) (PHM2021). IEEE, Detroit, USA, 2021 IEEE International Conference on Prognostics and Health Management, Detroit, United States, 7/06/21.
https://dx.doi.org/10.1109/ICPHM51084.2021.9486593

DOI 10.1109/ICPHM51084.2021.9486593
ISBN 978-1-6654-1970-3

Publisher: IEEE

Simultaneous Actuator and Sensor Faults Estimation for Aircraft Using a Jump-Markov Regularized Particle Filter

Enzo Iglésis
Coventry University
Coventry, United Kingdom
Email: iglesise@coventry.ac.uk

Nadjim Horri
School of Mechanical
Aerospace and Automotive Engineering
Coventry University
Coventry, United Kingdom
Email: nadjim.horri@coventry.ac.uk

James Brusey
Faculty Research Centre for Data Science
Coventry University
Coventry, United Kingdom
Email: james.brusey@coventry.ac.uk

Karim Dahia
Information Processing and Systems Department
ONERA
Palaiseau, France
Email: karim.dahia@onera.fr

Hélène Piet-Lahanier
Information Processing and Systems Department
ONERA
Palaiseau, France
Email: helene.piet-lahanier@onera.fr

***Abstract*—The advances in aircraft autonomy have led to an increased demand for robust sensor and actuator fault detection and estimation methods in challenging situations including the onset of ambiguous faults. In this paper, we consider potential simultaneous fault on sensors and actuators of an Unmanned Aerial Vehicle. The faults are estimated using a Jump-Markov Regularized Particle Filter. The jump Markov decision process is used within a regularized particle filter structure to drive a small subset of particles to test the likelihood of the alternate hypothesis to the current fault mode. A prior distribution of the fault is updated using innovations based on predicted control and measurements. Fault scenarios were focused on cases when the impacts of the actuator and sensor faults are similar. Monte Carlo simulations illustrate the ability of the approach to discriminate between the two types of faults and to accurately and rapidly estimate them. The states are also accurately estimated.**

I. Introduction

A major issue in small Unmanned Aerial Vehicles (UAVs) is to maintain a safe flight in the event of faults in either actuators or embedded sensors. As these faults may deeply impact control system performance and cause catastrophic accidents, it is essential to detect and estimate them in order to limit their adverse effect on the flight. An efficient approach for state and fault estimation relies on the use of mode switching between faulty and non-faulty models, governed by a Markovian process. In recent years, many approaches have been reported to detect and estimate faults using a jump Markov representation [1]–[3].

It should be pointed out that most of the methods mentioned above are only tackling one type of fault, i. e., either sensor fault or actuator fault, when both should be considered simultaneously in a multimode fault tolerant system. Kalman Filters have for example been adapted to a variety of joint state and fault estimation problems. In [4], an adaptive Kalman filter was designed to estimate actuator faults by modelling the aircraft dynamics as a Linear Parameter Varying (LPV) system. In [5], sensor and actuator faults were estimated using an Interacting Multiple Model Kalman Filters (IMM-KF) architecture, where interacting filters are associated with nominal or faulty models. However, the IMM-KF has the limitations associated with Kalman filters and can diverge in highly nonlinear or multimodal cases [6].

In [7], [8] fault estimation and fault tolerant control for Markov Jump Systems (MJS) are considered in the presence of actuator faults and sensor faults. State and fault observers have been proposed in [9], [10], but it is difficult to guarantee that their speed of convergence to the fault state will be high enough to allow reconfiguration of the vehicle. Moreover, an issue that is seldom tackled in the presented approach is the need to discriminate between sensor or actuator faults when both

have the same impacts on the system behaviour, at least on a given time horizon, creating a variation of the same level of magnitude on the impacted state components. It is thus necessary to develop estimation methods that efficiently handle ambiguity and/or multimodality.

Particle filters were successfully used for estimation problems including non-linearities or ambiguous measurements [6], [11]. In multimode systems involving sensor and actuator faults, one possible approach is the extension of the multiple model concept to particle filters, as in [12] where the emphasis was on target tracking. However, the computational demand of such an approach would not be suitable to real time implementation in small and medium endurance UAVs. MJS models are therefore often used to transition between faulty and nominal operation modes in such systems. MJS models were combined with a sequential importance sampling based particle filters by Doucet et al. in [13]. Tafazoli and Sun [14] also developed a particle filter for a hybrid MJS model with improvements in fault detection compared to conventional particle filtering, but the mode selection was performed by testing all modes with the same number of particles and selecting the most likely one, which is computationally demanding for real time applications. In [6], a regularization step (see [15] from Musso et al.) was added to a jump-Markov particle filter approach to deal with ambiguities due to sensor redundancies, with Markov jumps between nominal and faulty sensing. Particle filters were also applied to incipient faults in [16].

In this paper, the focus is on the estimation of intermittent and abrupt sensor and actuator additive faults for ambiguous and multimodal fault scenarios. The approach is applied to a fixed-wing UAV. A Jump-Markov Regularized Particle Filter (JMRPF) approach is proposed to estimate sensor and actuator faults even in cases when both occur simultaneously.

The main contributions of the paper are as follows:

- A jump strategy for both sensors and actuators is presented, where the a priori distribution of the fault is computed using sensor and actuator innovation terms. Then it makes possible to keep testing the alternate mode using a small subset of sentinel particles, allowing them to transition to the correct fault mode. This enhances real time operation prospects compared to previous Jump-Markov Particle Filter approaches.
- A Kalman correction step is introduced in the JMRPF to place the particles in the most likely areas of the state space.

The paper is organized as follows. Section II describes the problem formulation and a jump Markov linear system model, adapted to fault estimation, is presented. Section III details the JMRPF approach for actuator and sensor fault estimation. In Section IV, the fault detection and estimation algorithms are evaluated on a scenario involving ambiguous faults in the elevator and in pitch rate. The JMRPF is compared to a Regularized Particle Filter (RPF) (see [15]). Section V concludes the paper.

II. Problem statement

A. Ambiguous fault

A fault from an actuator or sensor with an impact on the same measurement is hereby referred to as an ambiguous fault. It is common in feedback control systems when a sensor is used to measure a state variable and an actuator is used to control the same variable to a setpoint. If an actuator is faulty, the associated state variable and measurement will be affected. Likewise, a sensor fault will have a direct impact on the same output measurement. Therefore, if measurements are detected as being faulty, it is not trivial to determine if the fault originated from the sensor or the actuator[1].

This ambiguous fault case may lead to a multimodality in the likelihood and conditional density. Indeed, let us consider a discrete system with a state vector denoted $\mathbf{z}_k$ at time step k. This system has one actuator and one sensor that provides a measurement at each time step k denoted y_k. A fault on the actuator or on the sensor induces a similar effect on the measurements.

To jointly estimate the state, actuator fault and sensor fault, an extended state vector is defined and given by:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{z}_k^\top & \mathrm{f_a}_k^\top & \mathrm{f_s}_k^\top \end{bmatrix}^\top \tag{1}$$

where $\mathrm{f_a}$ and $\mathrm{f_s}$ respectively denote the actuator and sensor faults. When a faulty measurement occurs, the likelihood $p\left(\mathrm{y}_k|\mathbf{x}_k\right)$ has two peaks corresponding to two possible modes (solutions) that are: $\begin{bmatrix} \mathbf{z}_k{}^\top & \mathrm{f_a}_k{}^\top & 0 \end{bmatrix}^\top$ and $\begin{bmatrix} \mathbf{z}_k{}^\top & 0 & \mathrm{f_s}_k{}^\top \end{bmatrix}^\top$. In other words, several states $\mathbf{x}_k$ may be associated with the same measurement.

This results in the multimodality of conditional density $p\left(\mathbf{x}_k|\mathbf{Y}_{1:k}\right)$ as illustrated in Fig. 1. In this case, the Extended Kalman Filter (EKF) is not suitable.

B. Jump Markov linear system model

The problem considered in this paper is the joint state and fault estimation in the case of sensor and actuator

[1]Fault can also come from other sources, but they are not considered in this paper

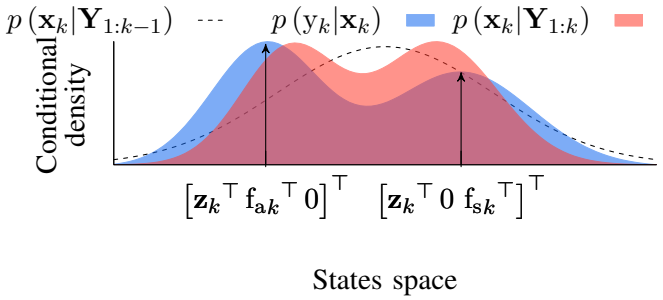


Fig. 1. A representation of the conditional density $p(\mathbf{x}_k|\mathbf{Y}_{1:k})$ in the case of sensor and actuator faults

faults that can occur simultaneously. The occurrences of faults can be modelled using a MJS, by associating a fault-free and faulty mode to the system. The discrete state space model system based on [6], [17] is represented as follows:

$$\begin{cases} \mathbf{m}_{k+1} \sim\ p(\mathbf{m}_{k+1}|\mathbf{m}_k) & \text{(2a)} \\ \mathbf{x}_{k+1} = \mathbf{A}_{\mathbf{m}_k}\mathbf{x}_k + \mathbf{B}_{\mathbf{m}_k}\mathbf{u}_k + \boldsymbol{\eta}_k & \text{(2b)} \\ \mathbf{y}_k = \mathbf{C}_{\mathbf{m}_k}\mathbf{x}_k + \mathbf{D}_{\mathbf{m}_k}\mathbf{u}_k + \boldsymbol{\nu}_k & \text{(2c)} \end{cases}$$

where $\mathbf{m}_k$ is a discrete mode vector of the system at time step k. A mode of this vector can either be faulty $m^{(1)}$ or fault-free $m^{(0)}$. The vector $\mathbf{x}_k \in \mathbb{R}^{n_x}$ 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. For jointly state, actuator fault and sensor fault estimation the state vector $\mathbf{x}_k$ is an extended state vector that contain the regular states of the system denoted here $\mathbf{z}_k \in \mathbb{R}^{n_z}$, the actuator and sensor fault states estimate denoted here respectively $\mathbf{f_a}_k \in \mathbb{R}^{n_a}$ and $\mathbf{f_s}_k \in \mathbb{R}^{n_s}$. Then, the state vector $\mathbf{x}_k$ is given by $\mathbf{x}_k = \begin{bmatrix} \mathbf{z}_k^\top & \mathbf{f_a}_k^\top & \mathbf{f_s}_k^\top \end{bmatrix}^\top$. The process and sensors noises are $\boldsymbol{\eta}_k \in \mathbb{R}^{n_x}$ 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 $\mathbb{E}\ \boldsymbol{\eta}_k\boldsymbol{\eta}_k^\top = \mathbf{Q}_k$ and $\mathbb{E}\ \boldsymbol{\nu}_k\boldsymbol{\nu}_k^\top = \mathbf{R}_k$. They are assumed to be independent $\mathbb{E}\ \boldsymbol{\eta}_k\boldsymbol{\nu}_k^\top = 0$. The matrices $\mathbf{A}_{\mathbf{m}_k}$, $\mathbf{B}_{\mathbf{m}_k}$, $\mathbf{C}_{\mathbf{m}_k}$ and $\mathbf{D}_{\mathbf{m}_k}$ denotes the usual discrete state space matrices terms and depend on the modes $\mathbf{m}_k$ at time step k. They are given by:

$$\mathbf{A}_{\mathbf{m}_k} = \begin{bmatrix} \mathbf{A_z} & \mathbf{G_{azm}}_{ak} & 0_{n_z\times\ n_s} \\ 0_{n_a\times\ n_z} & \mathbf{G_{am}}_{ak} & 0_{n_a\times\ n_s} \\ 0_{n_s\times\ n_z} & 0_{n_s\times\ n_a} & \mathbf{G_{sm}}_{sk} \end{bmatrix} \quad \text{(3a)}$$

$$\mathbf{B}_{\mathbf{m}_k} = \begin{bmatrix} \mathbf{B_z} \\ 0_{n_a+n_s\times\ n_u} \end{bmatrix} \quad \text{(3b)}$$

$$\mathbf{C}_{\mathbf{m}_k} = \begin{bmatrix} \mathbf{C_z} & 0_{n_y\times n_s} & \mathbf{G_{sy}}_{\mathbf{m}_{sk}} \end{bmatrix} \quad \text{(3c)}$$

$$\mathbf{D}_{\mathbf{m}_k} = \begin{bmatrix} \mathbf{D_z} \\ 0_{n_a+n_s\times\ n_u} \end{bmatrix} \quad \text{(3d)}$$

Where $\mathbf{A_z}$, $\mathbf{B_z}$, $\mathbf{C_z}$ and $\mathbf{D_z}$ denotes the usual discrete state space matrices terms applied to state vector $\mathbf{z}_k$. The matrices $\mathbf{G_{am}}_{ak}$ and $\mathbf{G_{sm}}_{sk}$ respectively represent the actuator and sensor fault dynamics, applied respectively to state vectors $\mathbf{f_a}_k$ and $\mathbf{f_s}_k$, and depend on the mode vectors $\mathbf{m_a}$ and $\mathbf{m_s}$ at time step k. The mode vectors $\mathbf{m_a}$ and $\mathbf{m_s}$ are respectively the mode vector associated to the actuator and sensor fault sate vector $\mathbf{f_a}$ and $\mathbf{f_s}$ that compose the mode vector $\mathbf{m}_k$ which is given by $\mathbf{m}_k = \begin{bmatrix} \mathbf{m_a}_k^\top & \mathbf{m_s}_k^\top \end{bmatrix}^\top$. The matrices $\mathbf{G_{az}}$ and $\mathbf{G_{sy}}$ respectively represent the coupling matrices of the actuator and sensor faults on the state and measurements at time step k, and they depend respectively on the mode vectors $\mathbf{m_a}$ and $\mathbf{m_s}$

According to (2), the system switches between as many dynamical models as there are elements of $\mathbf{m}_k$. However, if a dynamical model is associated to a mode $m^{(0)}$, then the fault state is in a fault-free mode and its value is equal to 0. In this case, irrespective of the state transition matrix, when it is multiplied by the estimated fault state in a fault free mode, then the new fault state is equal to 0. Only a state in a faulty mode $m^{(1)}$ has an influence on the new fault state. The dynamical model is then simplified and only associated to the state transition matrix for mode $m^{(1)}$. In the following sections, matrices $\mathbf{A}_{\mathbf{m}_k}$, $\mathbf{B}_{\mathbf{m}_k}$, $\mathbf{C}_{\mathbf{m}_k}$ and $\mathbf{D}_{\mathbf{m}_k}$ with all the modes of the vector $\mathbf{m}_k$ set to $m^{(1)}$ are denoted $\mathbf{A}_{\mathbf{m}^{(1)}}$, $\mathbf{B}_{\mathbf{m}^{(1)}}$, $\mathbf{C}_{\mathbf{m}^{(1)}}$ and $\mathbf{D}_{\mathbf{m}^{(1)}}$. The modes then only affect the $\mathbf{f_a}$ and $\mathbf{f_s}$ states. A jump strategy is then defined to switch between the fault-free and faulty states.

C. The transition probability based jump strategy

The probability to switch from a mode $m^{(i)}$ to $m^{(j)}$ is denoted by $\pi_{ji} = \mathbb{P}\left(m_{k+1}^{(j)}|m_k^{(i)}\right)$. 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 $m^{(0)}$. A matrix $\mathbf{\Pi}$ is defined as the transition probability matrix, which represents the probability of switching from one mode to another. It is given by:

$$\mathbf{\Pi} = \begin{bmatrix} \pi_{00} & \pi_{10} \\ \pi_{01} & \pi_{11} \end{bmatrix} \quad \text{(4)}$$

Each state of the state vector $\mathbf{f_a}$ and $\mathbf{f_s}$ is associated with a $\mathbf{\Pi}$ matrix. The diagonal elements of the π_{jj} matrices represent probabilities to remain in the same mode for the given sensor or actuator.

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

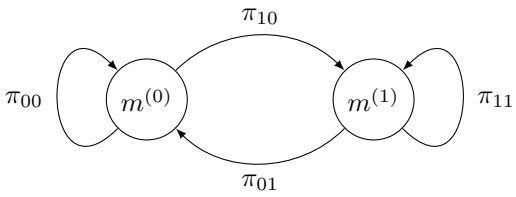


Fig. 2. Markov chain of the JMRPF applied to fault estimation

The objective of the method presented here is to simultaneously detect the occurrence of faults and to estimate their amplitude.

Evolution of $\mathbf{x}_k$ is represented by (2) and is conditioned by the probabilities of switching from one mode to another represented by $\mathbf{\Pi}$.

III. Jump-Markov Regularized Particle Filter

The transition matrix allows for abrupt changes between non-faulty and faulty modes. Jump Markov based filters have been designed to handle such transitions. Moreover, in the context of simultaneous sensor and actuator faults, the system may present multi-modalities, as in the case where the conditional density $p\left(\mathbf{x}_k|\mathbf{Y}_{1:k}\right)$ with $\mathbf{Y}_{1:k} = [\mathbf{y}_1, \ldots, \mathbf{y}_k]$ presents several peaks. This justifies the use of a Jump-Markov Particle Filter. In [6], a first version of a JMRPF was introduced to handle sensor fault estimation in presence of ambiguous measurements. In this paper, the particles $\mathbf{x}_k^i$ are corrected using a Kalman update to place the particles in the most likely areas of the state space. The particle weights are calculated by taking into account the Kalman update, which makes it possible to reduce the variance of the weights and improve the estimation accuracy of the JMRPF. The JMRPF is fed by the control input, the previous state estimate and the measurement to provide the estimated state vector $\hat{\mathbf{x}}_k$ and its associated estimated covariance matrix $\hat{\mathbf{P}}_k$. The total number of particles is denoted N_p.

The proposed global JMRPF algorithm [6] with the Kalman update and the actuator fault estimation is introduced hereafter in Algorithm 1. It is composed of prediction, update, estimation and regularization-resampling steps. The PREDICT and UPDATE functions were modified as described in Algorithms 2 and 4 for combined actuator and sensor fault estimation and using a jump function within the predict function as part of the hypothesis testing.

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

A. Prediction step

In the particle filter, the i^{th} state variable is propagated using the following probability transition density for the state $\mathbf{x}_{k|k-1}$:

$$\mathbf{x}_{k|k-1}^i \sim p\left(\mathbf{x}_{k|k-1}|\mathbf{x}_{k-1}^i,\ \mathbf{m}_k^i\right) \tag{5}$$

Then, one obtains a predicted cloud of particles $(\mathbf{x}_{k|k-1}^1,\ \mathbf{x}_{k|k-1}^2,\ \cdots,\ \mathbf{x}_{k|k-1}^{N_p})$. The mode of each state and particles of $\mathbf{f_a}$ and $\mathbf{f_s}$ are updated in the prediction step. This update is performed here using a uniform draw 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 number of possible actuator faults n_a is equal to the number of actuators n_u and the possible number of sensor faults n_s is equal to the number of sensors n_y. The same index j is also used to denote the j^{th} state of vectors $\mathbf{f_a}$ and $\mathbf{f_s}$, respectively representing fault estimates of the j^{th} actuator of $\mathbf{u}$ and of the j^{th} sensor of $\mathbf{y}$.

A new jump strategy for sensors and actuators fault modes is proposed. It uses the a priori distribution of the fault, which is computed using sensor and actuator innovation terms from (9) and (7). When a sensor or actuator is in a fault free or faulty mode, the alternate mode of the device will continue to be tested using a small number of fault particles that will be named sentinel particles. Those particles will be selected from a uniform distribution and their number will depend on the transition probability matrix. Indeed, the number of sentinel particles can be decreased by reducing the transition probabilities π_{01} and π_{10}. Those probabilities will be set based on expected device false alarm and missed detection rates. This small number of sentinel particles will continuously test the probability of transition of jump to the alternate fault modes. The fact that the number of sentinel particles is small reduces computational demand compared to previous particle filter jump strategies, such as the one of reference [14] where both modes are tested using the total number of particles at all times before selecting the mode with the higher probability.

The jump amplitudes of the i^{th} particle of the j^{th} state of $\mathbf{f_a}$ at time step k are computed as follows:

$$\mathrm{f_a}_{k|k-1}^{i,j} = \begin{cases} \lambda_k^{i,j} & \text{if } U \leq \pi_{10}^j \text{ and } \mathrm{m_a}_k^{i,j} = m^{(0)} \\ \mathrm{f_a}_{k|k-1}^{i,j} & \text{if } U < \pi_{11}^j \text{ and } \mathrm{m_a}_k^{i,j} = m^{(1)} \\ 0 & \text{else} \end{cases} \quad (6)$$

where $U \sim \mathcal{U}(0,\, 1)$ and $\boldsymbol{\lambda}_k^i$ is given by:

$$\boldsymbol{\lambda}_k^i = \mathbf{u}_k - r\left(\mathbf{z}_{k|k-1}^i\right) \quad (7)$$

where r is a stabilizing control law used to compute $\mathbf{u}_k = r\left(\hat{\mathbf{z}}_k\right)$.

The jump amplitudes of the i^{th} particle of the j^{th} state of $\mathbf{f_s}$ at time step k are computed as follows:

$$\mathrm{f_s}_{k|k-1}^{i,j} = \begin{cases} \beta_k^{i,j} & \text{if } U \leq \pi_{10}^j \text{ and } \mathrm{m_s}_k^{i,j} = m^{(0)} \\ \mathrm{f_s}_{k|k-1}^{i,j} & \text{if } U < \pi_{11}^j \text{ and } \mathrm{m_s}_k^{i,j} = m^{(1)} \\ 0 & \text{else} \end{cases} \quad (8)$$

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

$$\boldsymbol{\beta}_k^i = \mathbf{y}_k - \left(\mathbf{C}_{\mathbf{m}^{(1)}}\mathbf{x}_{k|k-1}^i + \mathbf{D}_{\mathbf{m}^{(1)}}\mathbf{u}_k\right) \quad (9)$$

The prediction step is described in Algorithm 2.

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

Algorithm 2 Detail of the function PREDICT from Algorithm 1

function PREDICT($\mathbf{x}_{k|k-1}^i$, $\mathbf{x}_{k-1}^i$, $\mathbf{m}_k^i$, $\mathbf{u}_k$, $\mathbf{y}_k$)
- $\boldsymbol{\eta}_k^i \sim \mathcal{N}(0,\, \mathbf{Q}_k)$
- $\mathbf{x}_{k|k-1}^i \leftarrow \mathbf{A}_{\mathbf{m}^{(1)}}\mathbf{x}_{k-1}^i + \mathbf{B}_{\mathbf{m}^{(1)}}\mathbf{u}_k + \boldsymbol{\eta}_k^i$
- $\boldsymbol{\lambda}_k^i \leftarrow \mathbf{u}_k - r\left(\mathbf{z}_{k|k-1}^i\right)$ ▷ See (7)
- **for each** $j \in [1,\, n_a]$ **do** ▷ Jump step of $\mathbf{f_a}$
  - JUMP($\mathrm{f_a}_{k|k-1}^{i,j}$, $\mathrm{m_a}_k^{i,j}$, $\lambda_k^{i,j}$) ▷ See Algorithm 3
- **end for**
- $\boldsymbol{\beta}_k^i \leftarrow \mathbf{y}_k - \left(\mathbf{C}_{\mathbf{m}^{(1)}}\mathbf{x}_{k|k-1}^i + \mathbf{D}_{\mathbf{m}^{(1)}}\mathbf{u}_k\right)$ ▷ See (9)
- **for each** $j \in [1,\, n_s]$ **do** ▷ Jump step of $\mathbf{f_s}$
  - JUMP($\mathrm{f_s}_{k|k-1}^{i,j}$, $\mathrm{m_s}_k^{i,j}$, $\beta_k^{i,j}$) ▷ See Algorithm 3
- **end for**

end function

Algorithm 3 Detail of the function JUMP from Algorithm 2

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

end function

B. Update step

In the particle filter, each i^{th} particle $\mathbf{x}_k^i$ are assigned to a weight w_k^i that is proportional to its likelihood:

$$\tilde{w}_k^i \propto w_{k-1}^i p\left(\mathbf{y}_k | \mathbf{x}_{k|k-1}^i,\, \mathbf{m}_k^i\right) \quad (10a)$$

$$w_k^i = \frac{\tilde{w}_k^i}{\sum_{i=1}^{N_p} \tilde{w}_k^i} \quad (10b)$$

In (10b) a normalization is applied to ensure that $\sum_{i=1}^{N_p} w_k^i = 1$.

Compared to the update step described in [6], an additional feature was introduced. Indeed, a Kalman update on the particles $\mathbf{x}_{k|k-1}^i$ is applied to better place the particles. The Kalman update is given by:

$$\mathbf{x}_k^i = \mathbf{x}_{k|k-1}^i + \mathbf{K}_k \tilde{\mathbf{y}}_k^i \tag{11}$$

This step is performed by the UPDATE function which is detailed in Algorithm 4 In this algorithm it is assumed that the likelihood is a Gaussian distribution.

Algorithm 4 Detail of the function UPDATE from Algorithm 1

function UPDATE($\mathbf{x}_k^i$, w_k^i, w_{k-1}^i, ${\mathbf{x}^i}_{k|k-1}$, $\mathbf{K}_k$, $\mathbf{S}_k$, $\mathbf{u}_k$, $\mathbf{y}_k$)
  $\tilde{\mathbf{y}}_k^i \leftarrow \mathbf{y}_k - \left(\mathbf{C}_{\mathbf{m}^{(1)}} \mathbf{x}_{k|k-1}^i + \mathbf{D}_{\mathbf{m}^{(1)}} \mathbf{u}_k\right)$ ▷ Innovation
  $\tilde{w}_k^i \leftarrow w_{k-1}^i \mathcal{N}(\tilde{\mathbf{y}}_k^i;\, 0,\, \mathbf{S}_k)$ ▷ See 10a
  $w_k^i \leftarrow \frac{\tilde{w}_k^i}{\sum_{i=1}^{N_p} \tilde{w}_k^i}$
  $\mathbf{x}_k^i \leftarrow \mathbf{x}_{k|k-1}^i + \mathbf{K}_k \tilde{\mathbf{y}}_k^i$ ▷ See (11)
end function

C. Estimation

The estimation step aims to perform a global estimate 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($\hat{\mathbf{x}}$, $\hat{\mathbf{P}}$, $\mathbf{x}^{1:N_p}$, $w^{1:N_p}$)
  $\hat{\mathbf{x}} \leftarrow \sum_{i=1}^{N_p} w^i \mathbf{x}^i$
  $\hat{\mathbf{P}} \leftarrow \sum_{i=1}^{N_p} w^i \quad \mathbf{x}^i - \hat{\mathbf{x}} \quad \mathbf{x}^i - \hat{\mathbf{x}} \quad ^\top$
end function

D. Regularization-Resampling step

The regularization-resampling 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 by duplicating the particles with a high likelihood and regularizing the duplicated particles.

a) Resampling step: The particles are selected according to a multinomial law with w_k^i as parameter. Then the probability to choose a particle is:

$$\mathbb{P}\left(\acute{\mathbf{x}}_k^j = \mathbf{x}_k^i\right) = w_k^i \tag{12}$$

This corresponds to the MULTINOMIAL function in Algorithm 1.

b) Regularization step: The particles are randomly moved according to a regularization kernel $\mathcal{K}(\mathbf{x})$. The regularization is given by:

$$\mathbf{x}_k^i = \acute{\mathbf{x}}_k^i + h\mathbf{D}_k \boldsymbol{\varepsilon}_k^i \tag{13}$$

where $h \in \mathbb{R}^{+*}$ is the bandwidth factor in the rescaled 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:

$$\int \mathbf{x}\mathcal{K}(\mathbf{x})\, d\mathbf{x} = 0, \quad \int \|\mathbf{x}\|^2 \mathcal{K}(\mathbf{x})\, d\mathbf{x} < \infty \tag{14}$$

The optimal kernel $\mathcal{K}(\cdot)$ and bandwidth factor h are those which minimize the Mean Integrated Square Error (MISE) between the theoretical and estimated posterior density, and is defined as:

$$\text{MISE}(\hat{p}) = \mathbb{E} \int \left(\hat{p}(\mathbf{x}_k|\mathbf{Y}_{1:k}) - p(\mathbf{x}_k|\mathbf{Y}_{1:k})\right)^2 d\mathbf{x}_k \tag{15}$$

where $\hat{p}(\mathbf{x}_k|\mathbf{Y}_{1:k})$ is the particle filter approximation of the state conditional density. In the case where all particles have the same weight, during the resampling step, a suitable choice of the kernel is the bounded Epanechnikov kernel [18].

$$\mathcal{K}(\mathbf{x}) = \begin{cases} \frac{n_x+2}{2c_{n_x}} \left(1 - \|\mathbf{x}\|^2\right) & \text{if} \quad \|\mathbf{x}\| < 1 \\ 0 & \text{else} \end{cases} \tag{16}$$

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

The algorithm of the regularization is described in Algorithm 6.

Algorithm 6 Detail of the function REGULARIZE from Algorithm 1

function REGULARIZE($\mathbf{x}_k^i$, $\acute{\mathbf{x}}_k^i$)
  $\boldsymbol{\varepsilon}_k^i \sim \mathcal{K} \quad \mathbf{x}_k^i$ ▷ see (16)
  $\mathbf{x}_k^i \leftarrow \acute{\mathbf{x}}_k^i + h\mathbf{D}_k \boldsymbol{\varepsilon}_k^i$
end function

However, this regularization-resampling step is not performed at each time step. A criterion is defined to

know if a resampling step is needed. The criterion that is used in this paper is the efficiency N_{eff}.

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_p} {w_k^i}^2} \tag{17}$$

If $\frac{N_{eff}}{N_p}$ is lower than user-defined threshold $\Gamma \in (0;\, 1)$ then the resampling step is performed.

Finally, after performing all above mentioned steps, the approached conditional density is given by:

$$p\left(\mathbf{x}_k, \mathbf{m}_k | \mathbf{Y}_{1:k}\right) \approx \sum_{i=1}^{N_p} w_k^i \mathcal{K}_h \left(\mathbf{x}_k - \mathbf{x}_k^i\right) \delta_{\mathbf{m}_k^i}\left(\mathbf{m}_k\right) \tag{18}$$

where:

$$\mathcal{K}_h\left(\mathbf{x}_k\right) = \frac{1}{h^{n_x}} \mathcal{K}\left(\frac{1}{h}\mathbf{x}_k\right) \tag{19}$$

IV. Simulation model and results

A. UAV Dynamical model

The application example is a fixed-wing UAV. We focus on the longitudinal model as ambiguities can occur when we consider faults on the pitch rate sensor or on the elevation actuator. The state vector representing UAV longitudinal dynamics is $\mathbf{z} = \begin{bmatrix} p_d & \mathrm{u} & \mathrm{w} & \theta & q \end{bmatrix}^\top$. The state p_d denotes altitude loss, u represents the deviation with respect to the longitudinal velocity trim condition along the i^b axis, w represents the vertical velocity along the k^b axis (see Fig. 3) and the states θ and q respectively denote the pitch and pitch rate. The control input vector is $\mathbf{u} = \begin{bmatrix} \delta_e & \delta_t \end{bmatrix}^T$, where δ_e and δ_t respectively represent the elevator deflection and throttle input. The full state is observed using an Inertial Navigation System (INS) hybridized with a Global Navigation Satellite System (GNSS) receiver, a magnetometer and a barometer.

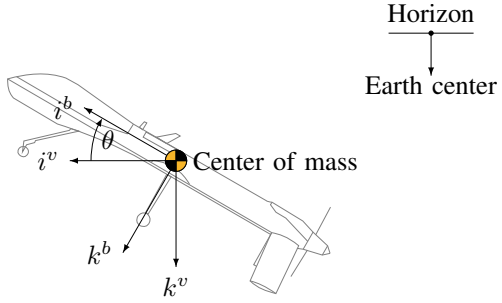


Fig. 3. Side view of an UAV with references axis and angles

The trim condition for the UAV model is straight level flight at a velocity of $40\,\mathrm{m\,s^{-1}}$ and an altitude of $500\,\mathrm{m}$. A linear model of the Aerosonde UAV given by [19] is used for the simulation analysis. The linear model is discretized at $40\,\mathrm{ms}$ and the state, control, observation and output matrices associated with the state vector $\mathbf{z}_k$ are given by:

$$\mathbf{A_z} = \begin{bmatrix} 1 & 0 & 0.04 & -1.6 & 0 \\ 0 & 0.98 & 0.01 & -0.39 & -0.07 \\ 0 & -0.01 & 0.91 & -0.01 & 1.51 \\ 0 & 0 & 0 & 1 & 0.04 \\ 0 & 0 & -0.03 & 0 & 0.95 \end{bmatrix} \tag{20a}$$

$$\mathbf{B_z} = \begin{bmatrix} 0.01 & 0.05 & -1.04 & -0.03 & -1.69 \\ 0 & 1.28 & -0.01 & 0 & 0 \end{bmatrix}^\top \tag{20b}$$

$$\mathbf{C_z} = I_5 \tag{20c}$$

$$\mathbf{D_z} = 0_{7\times 2} \tag{20d}$$

B. Control system

The longitudinal autopilot uses a desired flight path angle command γ^c and a velocity vector norm command V^c. The actuator inputs δ_e and δ_t are given by:

$$\begin{cases} \bar{\delta}_{ek+1} = -L_\theta \hat{\bar{\mathbf{z}}}_k - L_{\theta_i} \bar{\theta}_{ik+1} & \text{(21a)} \\ \bar{\delta}_{tk+1} = -L_{\mathrm{u}} \hat{\bar{\mathbf{z}}}_k - L_{\mathrm{u}_i} \bar{\mathrm{u}}_{ik+1} & \text{(21b)} \end{cases}$$

where the bar notation represents a variation around the trim condition. The gains L_θ, L_{u}, L_{u_i}, L_{θ_i} are obtained using a Linear Quadratic Regulator (LQR) with integral correction and weighting matrices $\mathbf{Q} = \mathrm{diag}\begin{bmatrix} 1 & 0 & 4 & 0 & 0 \end{bmatrix}$, $\mathbf{R} = I_{2\times 2}$. The integral gains are $L_{\theta_i} = 1.00$ and $L_{\mathrm{u}_i} = -1.00$. Integrated state deviations $\bar{\theta}_i$ and $\bar{\mathrm{u}}_i$ are given by:

$$\begin{cases} \bar{\theta}_{ik+1} = (\bar{\gamma}_k^c + A_{\mathrm{u}} \hat{\bar{\mathrm{u}}}_k + A_{\mathrm{w}} \hat{\bar{\mathrm{w}}}_k - \hat{\bar{\theta}}) dt + \bar{\theta}_{ik} & \text{(22a)} \\ \bar{\mathrm{u}}_{ik+1} = ((\bar{V}_k^c - V_{\mathrm{w}} \hat{\bar{\mathrm{w}}}_k) \dfrac{1}{V_{\mathrm{u}}} - \hat{\bar{\mathrm{u}}}_k) dt + \bar{\mathrm{u}}_{ik} & \text{(22b)} \end{cases}$$

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.

C. Filter parameters

For the simulations two filters are used: a JMRPF introduced in this paper and a RPF for comparison. The stochastic process model is given by:

$$\begin{cases} \mathbf{x}_{k+1} = \begin{bmatrix} \mathbf{A_z} & \mathbf{G_{az}} & 0_{5\times 1} \\ 0_{1\times 5} & \mathbf{G_a} & 0 \\ 0_{1\times 5} & 0 & \mathbf{G_s} \end{bmatrix} \mathbf{x}_k + \begin{matrix} \mathbf{B_z} \\ 0_{2\times 2} \end{matrix} \mathbf{u}_k + \boldsymbol{\nu}_k & \text{(23a)} \\ \mathbf{y}_k = \begin{bmatrix} \mathbf{C_z} & 0_{5\times 1} & \mathbf{G_{sy}} \end{bmatrix} \mathbf{x}_k + \begin{matrix} \mathbf{D_z} \\ 0_{2\times 2} \end{matrix} \mathbf{u}_k + \boldsymbol{\nu}_k & \text{(23b)} \end{cases}$$

where $\boldsymbol{\eta}_k$ and $\boldsymbol{\nu}_k$ are independent Gaussian noises with zero mean and covariance matrices denoted by $\mathbf{Q}_k$ and $\mathbf{R}_k$ respectively. $\mathbf{G_{az}}$, $\mathbf{G_a}$, $\mathbf{G_s}$ and $\mathbf{G_{sy}}$ are given by:

$$\mathbf{G_{az}} = \begin{bmatrix} 0.01 & 0.05 & -1.04 & -0.03 & -1.69 \end{bmatrix}^\top \quad (24a)$$

$$\mathbf{G_a} = 1 \quad (24b)$$

$$\mathbf{G_s} = 1 \quad (24c)$$

$$\mathbf{G_{sy}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}^\top \quad (24d)$$

The JMRPF and RPF parameters are[2]:

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

$$\boldsymbol{\sigma}_0 = \begin{bmatrix} 1 & 1 & 1 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix} \quad (25)$$

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

$$\begin{aligned} \boldsymbol{\sigma}_Q &= \begin{bmatrix} 0.01 & 0.02 & 0.02 & 0.3 & 0.1 & 0.3 & 0.3 \end{bmatrix} \\ \boldsymbol{\sigma}_R &= \begin{bmatrix} 1 & 1 & 1 & 0.3 & 0.1 \end{bmatrix} \end{aligned} \quad (26)$$

- The transition probability matrices for sensor and actuator faults are both equal to:

$$\boldsymbol{\Pi} = \begin{matrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{matrix} \quad (27)$$

  Selection of the transition $\boldsymbol{\Pi}$ matrix has been driven by reflecting the ambiguity which required to select identical matrices for sensor and actuator faults and being of the magnitude of potential false alarm probabilities indicated in such device.
- The resampling threshold Γ and regularization bandwidth h are:

$$\Gamma = 0.75, \quad h = 0.27 \quad (28)$$

- The number of particles N_p is set to 5000.

D. Fault scenario

From the expressions of the $\mathbf{B_z}$ matrix given by (20b), pitch rate is linked to elevator deflection δ_e. An actuator fault on δ_e is therefore difficult to discriminate from a sensor fault on q, which represents an ambiguity on the source of the fault. For the simulation analysis, the fault sequence is described in Fig. 4. High fault amplitudes of 10 degrees on the elevator and 10 degrees per second on the pitch rate are assumed to evaluate robustness to severe abrupt faults.

[2]Note: The angle unit used is degree

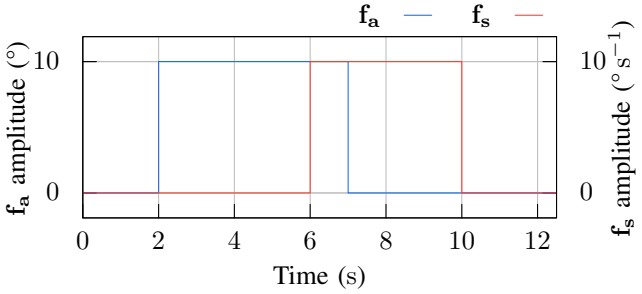


Fig. 4. Faults scenario used for the simulation

E. Simulation results and analysis

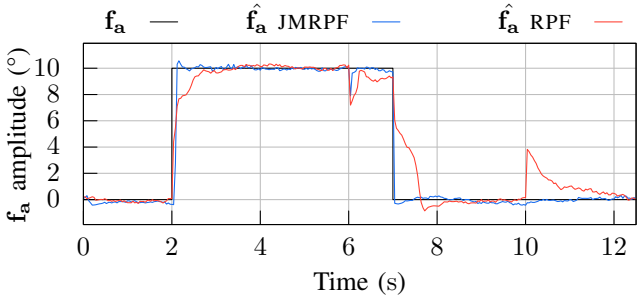
In Fig. 5, the JMRPF initially estimates the actuator fault faster when it occurs at 2 s, although the estimate of the sensor fault is briefly disrupted in the JMRPF. However, this deviation is shown in Fig. 7 to be so brief that altitude estimation is not affected, as hypothesis testing quickly resolves the ambiguity between sensor and actuator faults. Between 6 s and 7 s, both faults are simultaneously active and the JMRPF converges faster to the true fault modes and amplitudes compared to the RPF. Both estimators then accurately track the sensor fault from 7 s to 10 s. When the sensor fault is no longer active, the JMRPF quickly jumps to the fault free state but the RPF response to this change is approximately 2 s slower because the response of the RPF is more heavily restricted by the model dynamics, while the jump strategy of the JMRPF has a more instantaneous effect.

In Fig. 6, except for a brief jump at time 2 s, the JMRPF is shown to have significantly lowered Root-Mean-Square Error (RMSE) than the RPF, for both sensor and actuator faults. The evolution of RMSE on the faulty scenarios illustrates the difference of convergence speed between JMRPF and RPF.

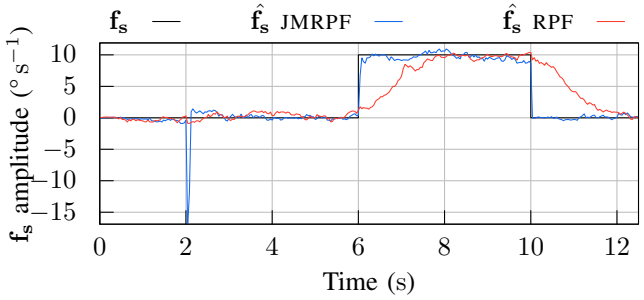
In Fig. 7, pitch rate and altitude deviations with respect to the setpoint are shown and the JMRPF clearly outperforms the RPF in terms of state estimation accuracy. The JMRPF accurately estimates both sensor and actuator faults and also estimates longitudinal states with better robustness to faults.

V. Conclusion

In this paper, a new JMRPF was proposed and applied to state and fault estimation in the presence of pitch rate sensor and elevator faults affecting the longitudinal states of a fixed-wing UAV. The proposed filter accurately estimates the states and faults even in the ambiguous case when both faults are active at the same time. The

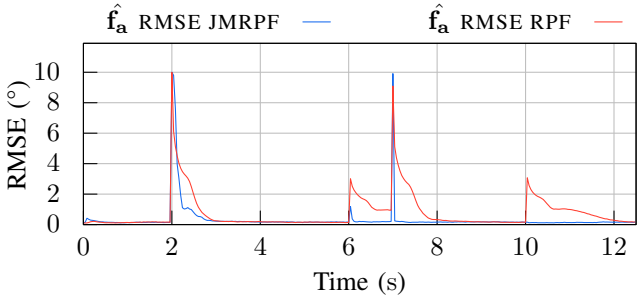


(a) Elevator deflection fault estimate.

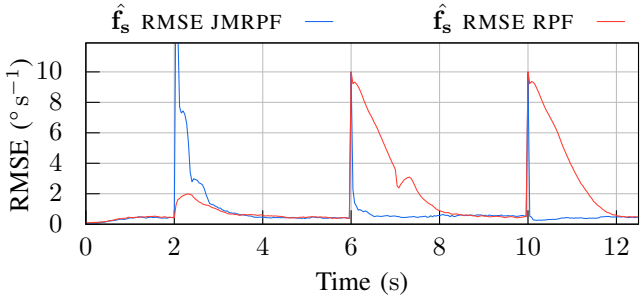


(b) Pitch rate sensor fault estimate.

Fig. 5. Actuator (a) and Sensor (b) additive fault estimates. Median results of the 100 simulations.



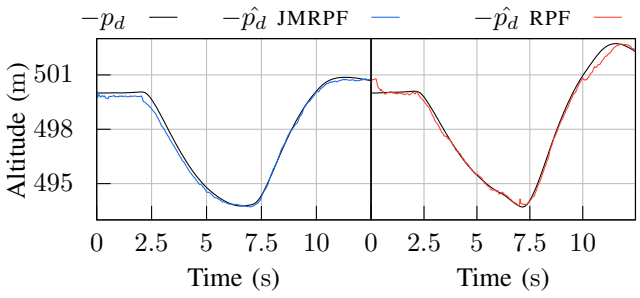
(a) Elevator deflection RMSE.



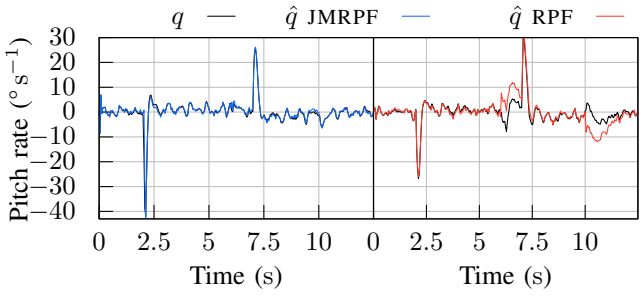
(b) Pitch rate RMSE.

Fig. 6. (a) RMSE of the actuator additive estimated fault. (b) RMSE of sensor additive estimated fault. RMSE are based on 100 simulations.

proposed Jump strategy allows a small subset of sentinel particles to explore the alternate mode and quickly detect



(a) Altitude.



(b) Pitch rate.

Fig. 7. Median result of the 100 simulations of selected state variables: (a) Altitude, (b) Pitch rate

mode changes between nominal and faulty operation. A Kalman correction places the particles in the most likely regions of the state space. Numerical simulations illustrate that the proposed JMRPF outperforms a RPF, in terms of convergence rate to the correct fault mode and fault estimation accuracy, even when sensor and actuator faults are simultaneously active. State estimation is also more accurate and robust to faults with the JMRPF.

References

[1] L. Blackmore, S. Rajamanoharan, and B. C. Williams, "Active estimation for jump markov linear systems," *IEEE Transactions on Automatic Control*, vol. 53, no. 10, pp. 2223–2236, 2008.

[2] J. Škach, I. Punčochář, and O. Straka, "Active fault diagnosis for jump markov nonlinear systems," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7308–7313, 2017.

[3] G. Wang, C. Yi, and M. Shen, "Fault estimation for continuous-time markovian jump systems by a mode-dependent intermediate estimator," *IET Control Theory & Applications*, vol. 12, no. 14, pp. 1924–1931, 2018.

[4] Q. Zhang, "Adaptive kalman filter for actuator fault diagnosis," *Automatica*, vol. 93, pp. 333–342, 2018.

[5] Y. Zhang and X. R. Li, "Detection and diagnosis of sensor and actuator failures using imm estimator," *IEEE Transactions on aerospace and electronic systems*, vol. 34, no. 4, pp. 1293–1313, 1998.

[6] 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," in *21st IFAC World Congress*. Elsevier, 2020.

[7] X. Li, H. R. Karimi, Y. Wang, D. Lu, and S. Guo, "Robust fault estimation and fault-tolerant control for markovian jump systems with general uncertain transition rates," *Journal of the Franklin Institute*, vol. 355, no. 8, pp. 3508–3540, 2018.

[8] X. Li, C. K. Ahn, D. Lu, and S. Guo, "Robust simultaneous fault estimation and nonfragile output feedback fault-tolerant control for markovian jump systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 9, pp. 1769–1776, 2019.

[9] H. Yang and S. Yin, "Descriptor observers design for markov jump systems with simultaneous sensor and actuator faults," *IEEE Transactions on Automatic Control*, vol. 64, no. 8, pp. 3370–3377, 2018.

[10] X. Song, J. Lam, X. Chen, and B. Zhu, "Descriptor state-bounding observer design for positive markov jump linear systems with sensor faults: Simultaneous state and faults estimation," *International Journal of Robust and Nonlinear Control*, vol. 30, no. 5, pp. 2113–2129, 2020.

[11] N. Merlinge, K. Dahia, H. Piet-Lahanier, J. Brusey, and N. Horri, "A box regularized particle filter for state estimation with severely ambiguous and non-linear measurements," *Automatica*, vol. 104, pp. 102–110, 2019.

[12] M. Yu, H. Oh, and W.-H. Chen, "An improved multiple model particle filtering approach for manoeuvring target tracking using airborne gmti with geographic information," *Aerospace Science and Technology*, vol. 52, pp. 62–69, 2016.

[13] 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.

[14] 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.

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

[16] M. Balchanos, D. Mavris, D. W. Brown, G. Georgoulas, and G. Vachtsevanos, "Incipient failure detection: A particle filtering approach with application to actuator systems," in *2017 13th IEEE International Conference on Control & Automation (ICCA)*. IEEE, 2017, pp. 64–69.

[17] A. Svensson, T. B. Schön, and F. Lindsten, "Identification of jump markov linear models using particle filters," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 6504–6509.

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

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