A Jump-Markov Regularized Particle Filter for the estimation of ambiguous sensor faults


* ONERA DTIS Université Paris Saclay (e-mail: firstname.lastname@onera.fr).
** Coventry University (e-mail: (E.I) iglesise@uni.coventry.ac.uk; (N.H; J.B) firstname.lastname@coventry.ac.uk)

Abstract: Sensor or actuator faults occurring on a Unmanned Aerial Vehicle (UAV) can compromise the system integrity. Fault diagnosis methods is then becoming a required feature for those systems. In this paper, the focus is on fault estimation for a fixed-wing UAVs in the presence of simultaneous sensor faults. The altitude measurements of a UAV are commonly obtained from the combination of two different types of sensors: a Global Navigation Satellite System (GNSS) receiver and a barometer. Both sensors are subject to additive abrupt faults. To deal with the multimodal nature of the faulty modes, a Jump-Markov Regularized Particle Filter (JMRPF) is proposed in this paper to estimate the barometric altitude and GNSS altitude measurement faults, including the case when both faults occur simultaneously. This method is based on a regularization step that improves the robustness thanks to the approximation of the conditional density by a kernel mixture. In addition, the new jump strategy estimate the correct failure mode in 100% of the 100 simulations performed in this paper. This approach is compared with an Interacting Multiple Model Kalman Filter (IMM-KF) and the results show that the JMRPF outperforms the IMM-KF approach, particularly in the ambiguous case when both sensors are simultaneously subject to additive abrupt faults.

Keywords: fixed-wing UAVs, fault estimation, Jump-Markov Regularized Particle Filter, IMM-KF.

1. INTRODUCTION

Sensor faults have a direct impact on the Guidance Navigation and Control (GNC) loop of a Unmanned Aerial Vehicles (UAVs). If faulty information are used by the UAV GNC loop this could compromise the mission of the UAV. Thus Fault Detection, Isolation and Recovery (FDIR) is essential in large UAVs because of their high reliability requirements and increasingly essential to small UAV because of the lack of direct sensor redundancies to reduce costs and the trend towards higher levels of autonomy.

Typical faults are described by Varga (2017) and can be categorized into: time and shape related fault types. Time-related faults can be persistent or intermittent. Shape-related faults can be abrupt (sudden) and incipient (gradual build-up).

Various fault detection approaches were presented in the literature. Data-driven approaches (see Ding (2014)) require the use of a large dataset for statistical analysis. However, when the amount of available data is small, model-based approaches (see Ding (2008)) are preferred. Model-based approaches use the knowledge of plant models to provide analytical redundancy. In the case of a fixed-wing UAV, the laws governing the flight dynamics are known (see Beard and McLain (2012); Cook (2012)) allowing the use of model-based approaches.

Several ways to reach the model-based framework were developed. The general idea in this framework, for fault detection, is to compute the error between the actual and predicted measurements, called the residual. Predicted measurements are frequently computed by a state estimator, which may be a Kalman filter. Fault detection is usually performed through statistical tests on residuals. These tests aim to detect a significant change of the residual value. The simplest one consists of a direct comparison of the residual value to a given threshold, but others are based on an assumption about the residual distribution (see Basseville et al. (1993)), such as the Student t-test, the CUMulative SUM (CUSUM), a likelihood ratio test such as the Generalize Likelihood Ratio Test (GLRT), or the Sequential Probability Ratio Test (SPRT). The combined fault detection and isolation is the basis of fault diagnosis.

Fault estimation represents the next level of complexity in fault diagnosis and addresses the reconstruction of fault signals from the available measurements (see Varga (2017)). For fault estimation, a state estimator such as the Kalman filter could be used, or a more sophisticated method may consist of using a bank of Kalman type filters and enclosing it in an Interacting Multiple Model (IMM) architecture (see Zhang and Li (1998); Rago et al. (1998)), which is a widely used state-of-the-art approach. Each filter is associated with nominal or faulty dynamics. These filters interact to provide a global likelihood. However, the
IMM with Kalman Filters (IMM-KF) can diverge in a multimodal case and has the limitations associated with the use of Kalman filters.

The measurement equation considered in this paper used by the fault estimator algorithm has a redundant term that is distinguishable by the sensor noise only. When an additive fault occurs on one of this terms, a multimodality appears on the associated estimated state. Monte-Carlo and particle-based methods were introduced to tackle non-linear and multimodal problems (see Arulampalam et al. (2002); Tafazoli and Sun (2006)). Moreover, a faulty and a fault-free situation can be considered as a discrete state of the system, like a jump Markov system. Doucet et al. (2001) have coupled the Jump Markov Linear System (JMLS) with the particle filter. This approach is used in this paper with a regularized particle filter (see Musso et al. (2001)) to handle the multimodal fault estimation problem better than linearized-Gaussian approaches. It is based on two improvements:

- A regularization step: the particles are drawn from a continuous density rather than a sampled distribution. This yields a better robustness and accuracy.
- A new jump strategy: the a priori distribution of the fault accounts for the filter innovation, which makes it possible to move the particles towards the correct faulty mode.

Compared to conventional fault estimation methods, the Jump-Markov Regularized Particle Filter (JMRPF) introduced in this paper for fault estimation allows to distinguish between the standard deviation of the process noise of the fault estimate and the standard deviation of the jump, which is linked to the magnitude of the fault. In conventional estimation approaches such as Kalman filter or particle filter, the process noise of the fault estimate must be artificially inflated to detect additive faults with a large magnitude without any guarantee of stability. The JMRPF approach allows the estimator to fit the process noise with the knowledge of the state dynamics, and fit the standard deviation of the jump with the maximum magnitude expectation of the fault. In this paper, the JMRPF approach is applied to a fixed-wing UAV with an ambiguous measurement equation (i.e., a non injective measurement equation).

The paper is organized as follows. Section 2 describes the problem. One of the issues addressed by this paper is to detect when a fault occurs, which sensors are responsible; and perform a fault estimation by reconstructing the faults as an additive input signal. In Section 3, the Jump-Markov Regularized Particle Filter approach is introduced for fault estimation. Section 4 details the model used for the simulation. In Section 5, the fault detection and estimation algorithms are evaluated using a Monte-Carlo numerical simulation analysis. Section 6 concludes the paper.

2. PROBLEM STATEMENT

In this paper, a model-based approach will be used to detect and estimate UAV sensor faults and allow for a probabilistic transitioning between a normal operation mode and faulty sensor modes. Fast and accurate fault estimation together with an accurate state estimation algorithm will be necessary to avoid compromising the UAV mission integrity. The case where different types of sensors are used to observe the same state variable is considered here. Here, the altitude of the aircraft is observed using barometer and GNSS altitude measurements. The use of multiple sensors to measure the same variable is what makes the measurement equation ambiguous in a faulty situation. Indeed, only the sensor noise parametrisation makes it possible to differentiate between the observations. Fig. 1 shows the situation where two sensors with different noise standard deviations are used to estimate a state $x$. The measurements $y_1$ produced by the first sensor are faulty whereas the measurement $y_2$ from the second sensor are fault-free. The state estimation of $x$ is based on the conditional density of $p(x|Y_{1:k})$ after updating the predicted conditional density $p(x|Y_{1:k-1})$, where $Y_{1:k}$ is all the measurements of both sensor from time step 1 to $k$. In this situation, basing our decision on the conditional density $p(x|Y_{1:k})$ highlights the fact that there is an ambiguous choice to decide which mode of the conditional density corresponds to the state estimate. In this paper, this is called an ambiguous sensor fault.

The discrete state space model of the Jump-Markov Linear System (JMLS) based on Svensson et al. (2014) is given by:

\[
\begin{align*}
    m_{k+1} &\sim p(m_{k+1}|m_k) \quad \text{(1a)} \\
    x_{k+1} &= A_{m_k} x_k + B_{m_k} u_k + w_k \quad \text{(1b)} \\
    y_k &= C_{m_k} x_k + D_{m_k} u_k + v_k \quad \text{(1c)}
\end{align*}
\]

Where $m_k$ is a discrete mode of the system at time step $k$. The probability to switch from a mode $m^{(i)}$ to $m^{(j)}$ is denoted by $\pi_{ji} = P(m_{k+1} = m^{(j)} | m_{k} = m^{(i)})$. The $\pi_{ji}$ is an entry of the II matrix, where $i$ is the row index and $j$ the column index. The vector $x \in \mathbb{R}^{n_x \times 1}$ is the state vector of the system. Each state of the state vector is associated with a II matrix. The $A_{m_k}, B_{m_k}, C_{m_k}$ and $D_{m_k}$ denotes the usual discrete state space matrices terms, for the mode $m_k$. In addition to this $y \in \mathbb{R}^{n_y \times 1}$ is the output of the system and $u \in \mathbb{R}^{n_u \times 1}$ is the input. The terms $w_k \in \mathbb{R}^{n_w \times 1}$ and $v_k \in \mathbb{R}^{n_v \times 1}$ denote the process and sensors noise. Because those noise terms are zero mean, the covariance matrices...
can be defined as $E[ww_k^T] = Q_k$ and $E[v_kv_k^T] = R_k$ and they are independent $E[ww_k^T] = 0$.

2.2 Jump-Markov linear system for sensor faults estimation

There are two JMLS modes for the fault estimation application. A nominal mode $m^{(0)}$ and faulty mode $m^{(1)}$. The transition probability between each mode is:

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

(2)

The probability $\pi_{10}$ is the probability to switch from nominal to faulty mode while the probability $\pi_{01}$ is the probability to switch from faulty to be nominal. The other $\pi_{ji}$ probabilities are the probability remain in the same state. Therefore, $\pi_{00} = 1 - \pi_{10}$ and $\pi_{10} = 1 - \pi_{11}$. To apply this approach to fault estimation, an augmented state space must be considered. Hence state vector is given by:

$$
x = [z^T f^T]^T
$$

(3)

Where $z \in \mathbb{R}^{n_z \times 1}$ is the original state vector and $f \in \mathbb{R}^{n_f \times 1}$ is the fault estimates. Only the $f$ vector is associated to a JMLS. The Markov chain diagram of the JMLS applied to fault estimation is shown in Fig. 2.

![Markov chain diagram](image)

Fig. 2. Markov chain of the JMLS applied to fault estimation for the $s^{th}$ state of state vector $f$

The state-space matrices in nominal mode are:

$$
A_{m^{(0)}} = \begin{bmatrix} A_Z & A_{Zf} \\ 0_{n_f \times n_Z} & A_{f,m^{(0)}} \end{bmatrix}, \quad B_{m^{(0)}} = \begin{bmatrix} B_Z \\ 0_{n_f \times n_Z} \end{bmatrix},
$$

$$
C_{m^{(0)}} = [C_Z 0_{n_f \times n_f}], \quad D_{m^{(0)}} = \begin{bmatrix} D_Z \\ 0_{n_f \times n_f} \end{bmatrix}
$$

(4a)

Where $A$, $B$, $C$, and $D$ are the regular matrices associated to the $z$ state vector. The matrix $A_{Zf}$ represents the dynamics of the coupling between the fault and the state vector $z$. Finally $A_{f,m^{(0)}}$ represents the dynamics of the fault of the mode $m^{(0)}$. The state space matrices for the mode $m^{(1)}$ are:

$$
A_{m^{(1)}} = \begin{bmatrix} A_Z & A_{Zf} \\ 0_{n_f \times n_Z} & A_{f,m^{(1)}} \end{bmatrix}, \quad B_{m^{(1)}} = \begin{bmatrix} B_Z \\ 0_{n_f \times n_Z} \end{bmatrix},
$$

$$
C_{m^{(1)}} = [C_Z C_f], \quad D_{m^{(1)}} = \begin{bmatrix} D_Z \\ 0_{n_f \times n_f} \end{bmatrix}
$$

(5a)

Where $A_{f,m^{(1)}}$ represents the dynamics of the fault in mode $m^{(1)}$. The matrix $A^f$ can however be the same in mode $m^{(0)}$ and $m^{(1)}$. The unavoidable change is in the matrix $C_{m^{(1)}}$ given the effect of $f$ on the mode $m^{(1)}$ measurement equation with $C_f \in \mathbb{R}^{n_y \times n_f}$ denoting the output matrix of state $f$:

$$
y_{m^{(1)}} = y_{m^{(0)}} + C_f f
$$

(6)

where $y_{m^{(0)}}$ is the measurement equation for the mode $m^{(0)}$ and $y_{m^{(1)}}$ for the mode $m^{(1)}$. Given that the fault considered here occur on the sensors. The matrices $B$ and $D$ remain unchanged between the mode $m^{(0)}$ and $m^{(1)}$. If fault is considered on a sensor, the entry of $C_f$ associated to the fault estimate of the sensors on the state of $f$ must be set to 1 in order to get an equation measurement of the sensor equal to the nominal equation plus the fault estimate. However it is possible that only one sensor of the $f$ state vector is in mode $m^{(1)}$ and the others are in mode $m^{(0)}$. So in this configuration applying the state space of the mode $m^{(0)}$ or $m^{(1)}$ is not corresponding to the situation, a combination of mode must be considered. But being for the $s^{th}$ state of $f$ in mode $m^{(0)} \implies f_s = 0$. Therefore, always applying the state space of the mode $m^{(1)}$ has no effect on fault estimate in mode $m^{(0)}$ and it is the corresponding equations for the fault estimate in mode $m^{(1)}$. Indeed if $f = 0_{n_f}$ then:

$$
\begin{align*}
x_{k+1,m^{(1)}} &= x_{k+1,m^{(0)}} \\
y_{m^{(1)}} &= y_{m^{(0)}}
\end{align*}
$$

(7a)

(7b)

Then on the following sections only the discrete state space model $m^{(1)}$ is used.

2.3 Optimal filtering equation for state estimation

An extended state estimator is needed for fault and state estimation. The state estimator aims to approach the state $x_k$ conditional density at each time-step, given the past measurements $Y_k = [y_1 y_2 \cdots y_k]$. However, an additional issue to the classic state estimation problem is tackled here, since the measurement model is potentially faulty and is therefore unknown a priori. To tackle this issue, the dimension of the estimated parameters is enlarged to $E_k = [x_k^T m_k^T]^T$ in order to estimate both the state $x_k$ and the mode $m_k$, at each time-step. The modes $m_k$ is a vector associated to the state vector, the value of the mode define if a state is faulty ($m^{(1)}$) or not ($m^{(0)}$). The state conditional density for fault estimation is described by:

$$
p(x_k|Y_k) = \sum_{j=0}^{M^{n_x} - 1} p(x_k, m_k = J|Y_k)
$$

(8)

where $M$ is the number of modes, and $J$ is the $j^{th}$ subset-uplet $[m^{(0)}, \ldots , m^{(j)}]^T$, $\forall i, j \in N \setminus [0, M - 1]$. The optimal filtering for state estimation consists of two steps: the prediction and the update.

The prediction step aims to predict the state vector $E_{k|k-1}$. Based on the Chapman-Kolmogorov equation the density is propagated as:

$$
p(E_k|Y_{k-1}) = \int p(E_k|E_{k-1}) p(E_{k-1}|Y_{k-1}) \, dE_{k-1}
$$

(9)

Since $m_{k-1}$ is independent from $x_{k-1}$ given $m_{k-1}$, one has:

$$
p(E_k|E_{k-1}) = p(x_k|m_k, x_{k-1}) p(m_k|m_{k-1})
$$

(10)

The jump aims to propagate the fault parameters using the following transition density:

$$
x_k \sim p(x_k|m_k, x_{k-1})
$$

(11)

The update step uses the measurements to update the likelihood from the Bayes rule:

$$
p(E_k|Y_k) = \frac{p(y_k|E_k) p(E_k|Y_{k-1})}{p(y_k)}
$$

(12)
3. JUMP-MARKOV REGULARIZED PARTICLE FILTER

To perform the fault estimation, the estimator used in this paper is the regularized particle filter. The particle filter is based on the approximation of this state density by a cloud of particles. The regularization in a particle filter allows a better fit between the theoretical state density and its discrete approximation (see Musso et al. (2001)). Each particle represents a potential state of the system and can be associated with a probability Kernel. The association of a regularized particle filter with a JMLS method is in this paper called a Jump-Markov Regularized Particle Filter (JMRPF). The total number of particles is denoted $N_p$.

The JMRPF consist of 4 steps: the prediction, the update, the estimation, and the regularization-resampling.

The global algorithm is introduced in Algorithm 1. This algorithm is composed of Algorithm 2, 3, 4 and 5.

Algorithm 1 Jump-Markov Regularized Particle Filter

\[
\begin{align*}
\text{Initialization} & : \quad k \leftarrow 0 \\
\text{loop} & \quad \text{do} \\
& \quad \text{end loop} \\
& \quad \text{end function}
\end{align*}
\]

Prediction step: In the particle filter, the $i^{th}$ state variable is then propagated using the following transition density:

\[
x_{k|k-1}^{i} \sim p(x_{k|k-1}^{i}|m_{k}^{i}, x_{k-1}^{i})
\]  

Then, one obtains a predicted cloud of particles $(E_{k|k-1}^{1}, E_{k|k-1}^{2}, \cdots, E_{k|k-1}^{N_p})$ that makes it possible to approximate $p(E_{k|k-1})$.

The jump is part of the prediction step. The particles are drawn to jump. The draw is made according to a uniform law and it is compared to the transition density $\pi_{ji}$. If a particle is drawn for a jump its new mode is updated. The jump occurs only on the $f$ state vector and its equation is:

\[
\tilde{f}_{k|k-1}^{i} = \begin{cases} 
\beta_{k}^{i,s} & \text{if } U \leq \pi_{10} \text{ and } m_{k}^{i,s} = m^{(0)} \\
\beta_{k}^{i,s} & \text{if } U < \pi_{11} \text{ and } m_{k}^{i,s} = m^{(1)} \\
0 & \text{if } U \leq \pi_{01} \text{ and } m_{k}^{i,s} = m^{(1)} \\
0 & \text{if } U < \pi_{00} \text{ and } m_{k}^{i,s} = m^{(0)}
\end{cases}
\]  

(14)

Where $U \sim \mathcal{U}(0, 1)$ and, $\beta_{k}^{i,s}$ is an additive fault drawn according to a Gaussian distribution:

\[
\beta_{k}^{i} \sim \mathcal{N}(C_f^{\top} \tilde{y}_k, C_f^{\top} R_k C_f)
\]

(15)

The upper-scripts notation $i,s$ means $i^{th}$ particle and $s^{th}$ state of state vector $f$ with $s \in [1, n_f]$. The prediction step is described in Algorithm 2.

Algorithm 2 Detail of the function PREDICT from Algorithm 1

\[
\begin{align*}
\eta_k^{i} & \sim \mathcal{N}(0, Q_k) \\
x_{k|k-1}^{i} & \sim \text{multinomial}(u_k, y_k)
\end{align*}
\]

for each $s \in [1, n_f]$ do

if $m_{k}^{i,s} = m^{(0)}$ then

$\gamma_k^{i} \sim \mathcal{N}(0, 1)$

else if $m_{k}^{i,s} = m^{(1)}$ then

$\gamma_k^{i} \sim \mathcal{N}(0, 1)$

end if

\[
\begin{align*}
\eta_k^{i} & \sim \mathcal{N}(0, Q_k) \\
x_{k|k-1}^{i} & \sim \text{multinomial}(u_k, y_k)
\end{align*}
\]

end for

Update step: In the particle filter, each particle $E_{k|k-1}^{i}$ is assigned to a weight $w_{k}^{i}$ that is proportional to its likelihood:

\[
\tilde{w}_k^{i} \propto p(y_k|m_k^{i}, x_{k|k-1}^{i})
\]

(17a)

(17b)

However after updating the likelihood of each particle, its weight must be normalized. The Algorithm 3 describes the update step. In this algorithm it is assumed that the likelihood is a Gaussian distribution.
Regularization step: For the regularization of the previous estimate, the weighted innovation is computed. This step is described in Algorithm 3.

Algorithm 3 Detail of the function UPDATE from Algorithm 1

\begin{verbatim}
function UPDATE(\(\hat{x}^{t}_k, x^{t-1}_k, u_k, y_k\))
\(\hat{y}^t_k \leftarrow y_k - (C_{m(i)}x^{t-1}_{k-1} + D_{m(i)}u_k)\) \(\triangleright\) Innovation
\(\hat{w}^t_k \leftarrow w^{t-1}_k \Lambda^t_k, \quad \Lambda^t_k = \mathcal{N}[\hat{y}^t_k; 0; R_k] \)
end function
\end{verbatim}

Estimation step: This step aims to compute a global estimate \(\hat{x}_{k|k}\) with its associated covariance matrix \(P_{k|k}\) based on the likelihood of the particle. This step is described in Algorithm 4.

Algorithm 4 Detail of the function ESTIMATE from Algorithm 1

\begin{verbatim}
function ESTIMATE(\(\hat{x}_{k|k}, P_{k|k}, x_{k|k-1}\))
\(\hat{x}_{k|k} \leftarrow \sum_{i=1}^{N_k} w^t_i x^{t}_k \) \(\triangleright\) Innovation
\(P_{k|k} \leftarrow \sum_{i=1}^{N_k} w^t_i (x^{t}_k - \hat{x}_{k|k}) (x^{t}_k - \hat{x}_{k|k})^\top \)
end function
\end{verbatim}

Resampling step: The selection of the particles is performed according to a multinomial law with \(w_k\) as parameter. Then the probability to choose a particle is:

\[ P(\hat{x}_{k|k-1} = x^{t}_{k|k-1}) = w^t_i \]  

(18)

This corresponds to the function MULTINOMIAL\((w_k)\) in Algorithm 1.

Regularization-resampling step: This is the last step. It is 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.

Resampling step: The selection of the particles is performed according to a multinomial law with \(w_k\) as parameter. Then the probability to choose a particle is:

\[ P(\hat{x}_{k|k-1} = x^{t}_{k|k-1}) = w^t_i \]  

(18)

This corresponds to the function MULTINOMIAL\((w_k)\) in Algorithm 1.

Resampling step: The selection of the particles is performed according to a multinomial law with \(w_k\) as parameter. Then the probability to choose a particle is:

\[ P(\hat{x}_{k|k-1} = x^{t}_{k|k-1}) = w^t_i \]  

(18)

This corresponds to the function MULTINOMIAL\((w_k)\) in Algorithm 1.

Algorithm 5 Detail of the function REGULARIZE from Algorithm 1

\begin{verbatim}
function REGULARIZE(\(x^{t}_{k|k-1}, \hat{x}^{t}_{k|k-1}\))
\(\epsilon \sim K(\hat{x}^{t}_{k|k-1}) \) \(\triangleright\) see (22)
\(x^{t}_{k|k-1} \leftarrow \hat{x}^{t}_{k|k-1} + h D \epsilon \)  
end function
\end{verbatim}

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 \(N_{\text{eff}}\).

\[ N_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_k} w^t_i} \]  

(23)

If \(N_{\text{eff}}\) 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:

\[ p(x_k, m_k | Y_k) \approx \sum_{i=1}^{N_k} w^t_i K_h(x_k - x^{t}_i) \delta_{m_k}(m_k) \]  

(24)

where:

\[ K_h(x) = \frac{1}{h^{n_x}} K \left( \frac{1}{h} x \right) \]  

(25)

4. LINEAR LONITUDINAL UAV WITH NAVIGATION AND CONTROL

The system considered in this paper is a linearized longitudinal model of a fixed wing UAV. The state vector representing deviation with respect to the trim point is \(z = [p_u u w \theta q]^\top\) and \(z \in \mathbb{R}^{5 \times 1}\). The state \(p_d\) is a position in the direction of the vector \(k^b\) shown in Fig. 3 representing altitude loss, \(u\) is the velocity in direction of the vector \(k^b\) shown in Fig. 3, \(w\) in direction of the vector \(k^b\) axis shown in Fig. 3, \(\theta\) is the pitch and \(q\) to the pitch rate.

The state is partly measured with 5 sensors. The state \(p_d\) is directly measured with a GNSS receiver and a barometer, the state \(u\) with a pitot tube and the state \(\theta\) and \(q\) with an IMU. The sensors noise is defined by the \(R_k\) matrix introduced in the JMLS.

If the fault estimation is performed on the GNSS and barometer measurements only\(^1\), then state vector \(f = [f_g f_b]^\top\) and \(f \in \mathbb{R}^{2 \times 1}\), where \(f_g\) refers to the GNSS

\(^1\) This paper is focused on altitude measurements from two different sensors, however similar problem such as speed measurements from pitot tube (airspeed) and GPS (ground speed) could be addressed.
The UAV is controlled by a full-state feedback with integrator effect. The block diagram of the control of the UAV is shown in Fig. 4. The $C_i$ gain matrix allows to select the states used for the altitude regulation, here $p_d$. The gain $K_z$ is the gain of the full-state feedback and $K_i$ is the integral gain. The reference output is $-p_d^z$, $v^c$ and $\theta^c$ where $\theta_c$ is the desired pitch command defined by $-p_d^z$ the desired altitude and $u_c$ is the speed command. The navigation filter return the state estimate $\hat{z}$ of the $z$ state vector.

Because the model is linearized, an input of 0 refers to the trim condition. The UAV control input vector is $\delta = [\delta_e \; \delta_t]$ where $\delta_e$ is the elevator deflection and $\delta_t$ is the throttle.

5. SIMULATION RESULTS

To characterize the performances of the JMRPF, numerical simulations were performed with a JMRPF implemented on MATLAB®. The simulation results are compared with an IMM-KF. Even though the system under consideration is linear, there are multi-modalities due to multiple sensor faults modelled in a probabilistic way.

5.1 Numerical parameters of the UAV

A linearized longitudinal UAV model is used. The trim conditions of the linearization are: flight path angle set to 0 rad; straight flight; Altitude set to 500 m; Velocity set to 60 m s$^{-1}$. The sampling time period used for the simulation analysis is 0.01 s. The simulation parameters used for the UAV dynamics are the following (rounded up to $10^{-3}$):

$$A_z = \begin{bmatrix} 1 & 0 & 0.010 & -0.600 & 0 \\ 0 & 0.988 & -0.001 & -0.097 & 0.028 \\ 0 & -0.006 & 0.948 & 0.005 & 0.581 \\ 0 & 0 & 0 & 1.000 & 0.010 \\ 0 & -0.001 & -0.012 & 0 & 0.985 \end{bmatrix}$$

$$B_z = \begin{bmatrix} 0.002 & 0 \\ 0.001 & 0.885 \\ 0.016 & -0.003 \\ -0.005 & 0 \\ -0.963 & 0 \end{bmatrix}$$

The sensors considered for the simulation have an additive white Gaussian noise, the discretized output measurements and sensor error covariance noise matrices are given by:

$$C_z = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad D_z = 0_{n_v,n_w}$$

$$R_k = \begin{bmatrix} \sigma^2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0.02^2 & 0 \\ 0 & 0 & 0 & 0 & 0.002^2 \end{bmatrix}$$

The fault estimate is performed on the ambiguous measurements equation on the GNSS and the barometer. Then the corresponding output matrix of the fault estimate is:

$$C_f = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

and the fault estimate state vector is $f = [f_g \; f_b]$.

The numerical parameters of the control are the following:

$$C_z = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0.0998 & 0 & 0 & 0 \end{bmatrix}, \quad K_i = \begin{bmatrix} 0 & 22.361 \\ -1 & 0 \end{bmatrix}$$

$$K = \begin{bmatrix} 0 & 0 & 0.012 & -4.846 & -0.281 \\ 0.0998 & 0 & 0 & 0 \end{bmatrix}$$

$$C_{h} = [1 \; 0 \; 0 \; 0 \; 0]$$

5.2 Simulation parameters of the JMRPF

The JMRPF parameters are:

$$A_{f_f} = 0_{n_z,n_f}, \quad A_{f,m^{(0)}} = A_{f,m^{(1)}} = 0_{n_z,n_f}$$

The standard deviation vector used to compute the covariance matrix $P_0 = \text{diag}(\sigma_0^2)$ is equal to:

$$\sigma_0 = [10 \; 2 \; 2 \; 0.1 \; 0.01 \; 25 \; 5]^T$$

The standard deviation vector used to compute the covariance matrix $Q_k = \text{diag}(\sigma_Q^2)$ is equal to:

$$\sigma_Q = [1 \; 1 \; 0.02 \; 0.002 \; 0.08 \; 0.08]^T$$

For all the parameters of the $f$ state vector the $\Pi$ matrix used is the same and defined as:

$$\Pi = \begin{bmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{bmatrix}$$
For the resampling step:
\[ \Gamma = 0.015, \quad h_{opt} = 0.46 \]  
(34)
The matrix \( R_k' \) is the one used in the filter. Here \( R_k = \alpha R_k \), with \( \alpha > 1 \).

5.3 Simulation parameters of the IMM-KF

The IMM-KF (see Zhang and Li (1998)) is based on two Kalman filters. The first Kalman filter is the fault-free model. The second Kalman filter performs the state and fault estimation using the same measurements equation and fault model of the JMRPF. The matrices \( P_k \) and \( R_k' \) are the same as those used by the JMRPF. The matrix \( Q_k \) has been change on the IMM-KF for the fault estimates states to get a better result. The standard deviation \( \sigma_Q \) used for the IMM-KF is:
\[ \sigma_Q = [1 \ 1 \ 1 \ 0.02 \ 0.002 \ 5 \ 1]^\top \]  
(35)
The transition probability between the IMM-KF models is equal to the matrix \( \Pi \) defined in (33)

5.4 Flight scenario and results

The desired output is set to 0 (relative to the trim point) for desired altitude and velocity. For the flight scenario, two additive intermittent abrupt faults are considered. The first fault occurs on the GNSS altitude at 30 s with a magnitude of 50 m, the fault is deactivated at 50 s. The second fault occurs on the barometer altitude at 40 s with a magnitude of 30 m, the fault is deactivated at 60 s. No fault occurs on the other sensors. The IMM-KF and JMRPF are both applied to this scenario. The JMRPF is initialized in faulty mode \( m^{(1)} \) while the IMM-KF is initialized with a more favourable condition which consist of a weight of 0.5 on both of its models.

The navigation filter of the Fig. 4 is performed by the JMRPF or by the IMM.

The RMSE results of this scenario is shown in Fig. 5. Those results are based on 100 simulations with new measurements and initial states at each simulation. The same measurements were used for the IMM-KF and the JMRPF simulations.

On Fig. 5, a peak occurs at each time a fault is activated or deactivated. This can be explained by the fact that when the fault occurred the fault estimate has not predicted its occurrence and at this time step the gap is near the fault magnitude between the fault estimate and the real fault value.

Fault-free situation: The results of Fig. 5 show that in a fault-free situation after converging or after recovering from faulty situation both RMSE are near 0.

GNSS faulty & barometer fault-free: In Fig. 5 the IMM-KF is good at estimating the fault of the GNSS. But the activation of the fault has affected the fault estimate of the fault-free sensor for the IMM-KF while the RMSE of the JMRPF for the fault-free sensor remains unchanged. In this situation the RMSE of the JMRPF for the barometer is approximately 44 times lower than the RMSE of the IMM-KF and around 1.6 times lower for the GNSS.

Based on this RMSE simulation, a mean RMSE for fault estimate state has been computed on each simulation results and sorted. The median simulation results is shown in Fig. 6. The results of Fig. 6 show that the JMRPF seems to not have difficulties to estimate the fault when both faults are active. On the other hand, the IMM-KF fault estimate of the fault-free sensor is impacted when a fault is active. Moreover, the fault estimate of the barometer estimate only approximately half of the fault.

The impact of the fault estimate on the altitude state \( p_d \) of this simulation is shown in Fig. 7.

Fig. 7 shows that poor quality of the barometer fault estimate of the IMM-KF. With the JMRPF, the error on the barometric altitude estimate is not negligible but significantly lower. This significant change of the UAV
altitude can lead to a critical situation for the UAV and mission integrity. The impact of the GNSS fault estimation error is less significant, because the GNSS measurements are noisier than the barometer measurements and, as a consequence, the filter has less confidence in the GNSS measurements compared to the barometer measurements for the estimation of \( p_d \).

6. CONCLUSION

Even though the IMM-KF is well known to be a highly efficient fault estimator, it shows its limitations in the ambiguous case of simultaneous fault measurements on the same output. This was illustrated in this paper with two altitude measurements from barometric altitude sensor and a GNSS receiver. This limitation is due to the multimodal nature of such ambiguous simultaneous faults and may lead to some critical behaviour of the flight control. To address this problem, the JMRPF introduced in this paper can efficiently estimate simultaneous additive abrupt faults assuming knowledge of the likely faults’ dynamics.

The application of JMRPF algorithm to a UAV shows that even with intermittent faulty measurements from both altitude sensors, accurate estimation of UAV states is possible.

REFERENCES


