

A continuous and discrete framework for reconfiguration of control of faulty systems

Adèle Boche, Jean-Loup Farges, Henry de Plinval

▶ **To cite this version:**

Adèle Boche, Jean-Loup Farges, Henry de Plinval. A continuous and discrete framework for reconfiguration of control of faulty systems. IEEE SSCI2016, Dec 2016, ATHENES, Greece. hal-01442305

HAL Id: hal-01442305
https://hal.science/hal-01442305

Submitted on 20 Jan 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A continuous and discrete framework for reconfiguration of control of faulty systems

Adèle Boche*, Jean-Loup Farges* and Henry De Plinval*
*Onera - The French Aerospace Lab
Toulouse, France
Email: name.surname@onera.fr

***Abstract*—A novel approach for reconfigurable control systems design against sensor and actuator faults is proposed. The scheme is based on an estimation of the state and of the fault parameters, and on a Partially Observable Markov Decision Process (POMDP), used for the decision task of the feedback controller. The following fault types are considered: locking and loss of effectiveness of the device. The modelling framework includes sensor and actuator faults, while the reconfiguration logic is developed only for actuator faults. The effectiveness of the proposed approach is demonstrated on an academic example.**

I. Introduction

Over the last three decades, the growing demand for safety and reliability in complex systems has drawn significant research in Fault-Tolerant Control (FTC) [1]. Actuator and sensor FTC is more and more discussed in the literature and many effective methodologies and algorithms for FTC have been developed. FTC is a control system that combines diagnosis with control methods to handle faults in an intelligent way. It permits to maintain overall system stability and acceptable performance in the event of component faults. Generally, methods for FTC are classified into two types. Passive FTC methods which are designed to be robust against faults [2], and active FTC methods which react to the system component faults actively by reconfiguring control actions. A direct adaptive approach to achieve good reconfiguration performance without system identification is possible and is demonstrated for actuator faults [3]. However, most of the time, the two steps of active FTC are considered separately. The first step, the Fault Detection and Diagnosis (FDD) has drawn significant work [4]. The second step, i.e. the reconfiguration unit assumes that an FDD is available. This paper addresses active FTC system against actuator and sensor faults. Indeed, actuators and sensors are essential for safety and reliability of complex systems. It has been shown that the FDD problem for sensor and actuator faults can be treated in two steps. First an approach for detecting and isolating sensor faults is proposed, and in a second time the sensor and the actuator faults are isolated and the loss of control effectiveness and the magnitude of stuck actuator faults are estimated [5]. Differents reconfigurable controller design methods exist. A method based on a two-stage adaptative Kalman filter for the detection and identification of fault parameters, and on eigenstructure assignment technique for computing on-line new feedback and feedforward gains is demonstrated in [6]. However, this scheme only covers loss of effectiveness actuator faults. Most of the time, the reconfiguration is performed either by considering a bank of controllers associated to a bank of models of the system in nominal and faulty conditions or by adapting indirectly a controller to fault parameters [7]. Finally, another approach proposed an FTC method using Linear Parameter Varying (LPV) virtual actuators and sensors for non-linear systems [8]. This method masks the faults adapting the virtual devices to the nominal LPV controller instead of adapting the LPV controller.

This paper presents a novel approach which builds on the advantages of discrete and continuous frameworks to enable efficient reconfiguration in the case of faults. The approach combines indirect adaptive methods, based on continuous framework, and decision making methods, based on discrete framework. It presents the following originalities with respect to the literature:

- Indirect adaptation for a bank of controllers.
- No abrupt detection of faults but an update of a probability distribution over nominal and faulty modes.
- Selection of a controller in the bank taking into account the expected performances given the probability distribution.

The paper is organized as follows. Section II describes the modelling framework, first the continuous part of the model, second the discrete part of it. Section III discusses the proposed method for the reconfiguration problem. Section IV describes the controllers used for the particular case of actuator faults reconfiguration. Section V shows simulation results for actuator faults obtained on an academic example.

II. Modelling

The systems considered have several modes, between which they may switch: a nominal mode, and degraded modes. A degraded mode corresponds to the locking or loss of efficiency of a given sensor or actuator. As a result, they exhibit two types of behaviours: A/ a continuous behaviour, as long as the mode and the controller do not change, which is defined by differential equations; B/ a discrete behaviour, which consists in the modes and controllers switchings.

A. Modelling of the continuous behaviour of the system in any given mode

Fig. 1 depicts the general structure of the model.

In the figure, $\mathbf{u}_k \in \mathbb{R}^l$ is the desired control input, $\mathbf{e}_k \in \mathbb{R}^l$ is the effective control, $\mathbf{x}_k \in \mathbb{R}^n$ is the system state, $\mathbf{y}_k \in \mathbb{R}^m$ corresponds to the system outputs, $\mathbf{z}_k \in \mathbb{R}^m$ corresponds to the measurement, $\mathbf{w}_k^x$ is a zero-mean white Gaussian noise sequence with covariance Q_k^x representing the modelling errors, $\mathbf{v}_k$ is a zero-mean white Gaussian measurement noise sequence with covariance R_k, A, B, and C are known matrices, and F_1 and F_2 are functions. The function F_1, respectively F_2, permits to model the actuator, respectively sensor, faults.

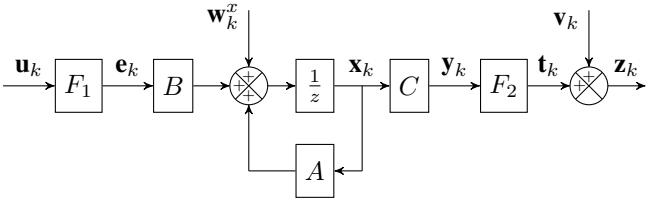


Fig. 1. System modelling

The objective is to control the system in such a way that the system state $\mathbf{x}_k$ follows a reference input $\mathbf{r}_k$.
A cost: $\sum_{k=0}^{k=K} ||\mathbf{r}_k - \mathbf{x}_k||$ indicates the lack of ability of the FTC to perform this tracking.

The modelling framework of F_1 and F_2 proposed in this article is inspired by [3] and is as follow.

1) Actuator faults: An actuator fault is modelled using a simple function of actuator command as shown in Fig. 2:

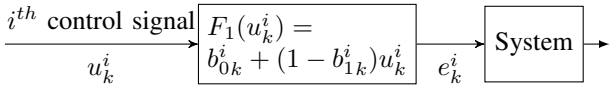


Fig. 2. Modelling of the i^{th} actuator faults

$$e_k^i = b_{0k}^i + (1 - b_{1k}^i)u_k^i \tag{1}$$

The output $\mathbf{e}_k$ corresponds to the physical motion performed by the actuator, that is limited in its amplitude. Thus, $\mathbf{e}_k$, and $\mathbf{u}_k$ are bounded in $[\underline{\mathbf{u}}, \overline{\mathbf{u}}]$.

Within this framework, several types of faults are modelled:

- If there is no fault, the nominal case corresponds to the following setting: $b_{0k}^i = 0$ and $b_{1k}^i = 0$.
- For a locking case in which the effective control is fixed at a certain value δ_0, $b_{0k}^i = \delta_0$ and $b_{1k}^i = 1$.
- Consider a bias of a certain value δ_0. This case can be modelled as $b_{0k}^i = \delta_0$ and $b_{1k}^i = 0$.
- With this modelling, we can also represent the decreased effectiveness case by setting $b_{0k}^i = 0$ and $b_{1k}^i \in]0, 1[$

2) Sensor faults: Similarly, a sensor fault is modelled using a simple function as shown in Fig. 3:

$$t_k^i = c_{0k}^i + (1 - c_{1k}^i)y_k^i \tag{2}$$

The outputs $\mathbf{z}_k$ and $\mathbf{y}_k$ are bounded in $[\underline{\mathbf{z}}, \overline{\mathbf{z}}]$. With this modelling, the same types of faults may be represented: locking, decreased effectiveness, and bias.

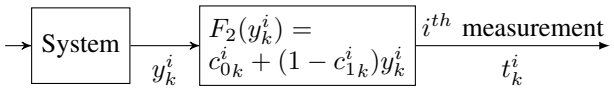


Fig. 3. Modelling of the i^{th} sensor faults

Starting from Fig. 1, the modelling of F_1 (1) and the modelling of F_2 (2), the discretised equations of the continuous system are modelled by the following equations:

$$\begin{cases} \mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k + B\mathbf{b_0}_k + D_k(\mathbf{u}_k)\mathbf{b_1}_k + \mathbf{w}_k^x \\ \mathbf{z}_k = C\mathbf{x}_k + \mathbf{c_0}_k + C_k(\mathbf{x}_k)\mathbf{c_1}_k + \mathbf{v}_k \end{cases} \tag{3}$$

$$\text{with } D_k(\mathbf{u}_k) = B \begin{bmatrix} -u_k^1 & 0 & \cdots & 0 \\ 0 & -u_k^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & ... & 0 & -u_k^l \end{bmatrix}$$

$$\text{and } C_k(\mathbf{x}_k) = \begin{bmatrix} -C(1,:)x_k^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & ... & -C(m,:)x_k^m \end{bmatrix}$$

The values of the parameters $\mathbf{b_0}$, $\mathbf{b_1}$, $\mathbf{c_0}$, and $\mathbf{c_1}$ change in the case of faults. From the estimation viewpoint, these values and their behaviour are unknown. In the continuous part of the model their behaviours are random walks:

$$\begin{cases} \mathbf{b_0}_{k+1} = \mathbf{b_0}_k + \mathbf{w}_k^{b_0} \\ \mathbf{b_1}_{k+1} = \mathbf{b_1}_k + \mathbf{w}_k^{b_1} \\ \mathbf{c_0}_{k+1} = \mathbf{c_0}_k + \mathbf{w}_k^{c_0} \\ \mathbf{c_1}_{k+1} = \mathbf{c_1}_k + \mathbf{w}_k^{c_1} \end{cases} \tag{4}$$

B. Discrete modelling

The reconfiguration problem can be modelled as a Partially Observable Markov Decision Process (POMDP) [9]. It is classically defined as a 6-tuple $\{S, A, T, R, \Omega, O\}$:

- $S = \{s_0, ...s_{|S|-1}\}$ is a finite set of states,
- $A = \{a_0, ...a_{|A|-1}\}$ is a finite set of actions,
- $T : S \times A \times S \to [0; 1]$ is a set of conditional transition probabilities between states,
- $R : S \times A \times S \to \mathbb{R}$ is the cost function,
- $\Omega = \{\omega_0 ... \omega_{|\Omega|-1}\}$ is a finite set of observations, and
- $O : S \times \Omega \to [0; 1]$ is a set of conditional observation probabilities

These elements are defined as follow for the framework.

1) States: The proposed framework considers only single faults. Let $m_{actuator}$ be the number of fault modes of an actuator and m_{sensor} be the number of fault modes of a sensor. Thus, we have at most a nominal mode, $m_{actuator} \times l$ actuator fault modes and $m_{sensor} \times m$ sensor fault modes.
Each mode m_i has an associated controller k_i which has been designed for this mode, so its performance is expected to be better than that obtained with the other controllers. However, at a given time step, it is possible that the system is in mode m_i and is controlled by another controller k_j. Thus, a set S of states $s_{i,j} = (m_i, k_j)$ is defined with a mode and a controller. S has at most $(1 + m_{actuator} \times l + m_{sensor} \times m)^2$ states.

2) Actions: The action a_i consists by definition in selecting the controller k_i for next time step. The system dynamics is as shown in Fig. 4.

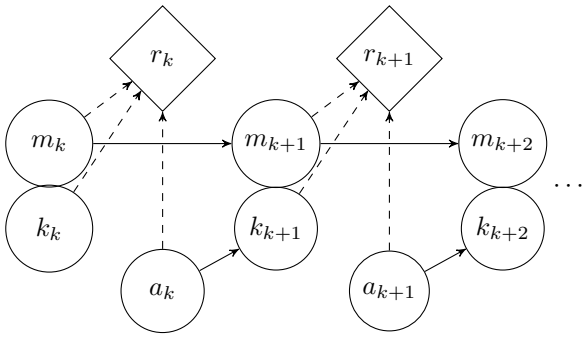


Fig. 4. State update with the action

3) Conditional transition probabilities: The general form of a transition probability is $p(s'|s,a)$, however the problem can be simplified. Indeed, on the one hand the dynamics of the active controller: $k_{k+1} = a_k$ is deterministic, and on the other hand the transitions between the modes are not dependant on the controller.

The transition probabilities $p(m'|m)$ between modes are stationary. To define a realistic model, these probabilities are considering the sensor/actuator documentation that may contain key information such as the failure rate, and previous studies addresing this topic [10].

Fig. 5 shows an example for a problem with three modes.

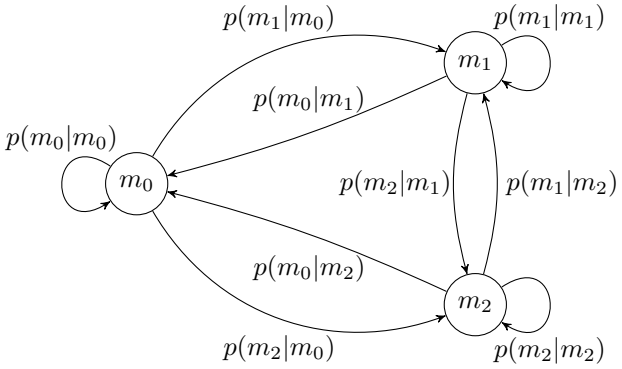


Fig. 5. Directed probabilistic graph

4) Cost function: To find the best controller to choose, different values of costs are imposed depending on the current state s_k, the next state s_{k+1}, and the action a_k. With this method, penalties can be added to the event of changing of controller, so that the potential performance loss triggered by this switch may be taken into account. The costs and the penalties can be calculated analytically or numerically.

In this article, the costs are calculated numerically with the error between the state and the reference, assuming that the future of the reference input $\mathbf{r}_k$ is known, and depend on the time. The penalties permit to avoid controller switching without reason and add a conservatism reward.

The cost function is as follow:

$$R(s_k, a_k, s_{k+1}) = E_{/s_{k+1}}||\mathbf{r}_{k+1} - \mathbf{x}_{k+1}|| + P(k_k, a_k) \quad (5)$$

where $E_{/s_{k+1}}||\mathbf{r}_{k+1} - \mathbf{x}_{k+1}||$ is an estimation of the elementary cost of the tracking error at time $k+1$ of the controller indicated in s_{k+1} for the mode of s_{k+1} and $P(k_k, a_k)$ equal to zero is $k_k = a_k$ and to a given value otherwise. Note that, unlike the classical POMDP framework, the cost function is not stationary.

5) Observations: The classical POMDP framework requires observations taking values in a discrete set; but, here, the observation is based on the information of the continuous processus provided by an estimation scheme: $\omega = \{\widehat{\mathbf{b_0}}_k, \widehat{\mathbf{b_1}}_k, \widehat{\mathbf{c_0}}_k, \widehat{\mathbf{c_1}}_k\}$.

6) Conditional observation probabilities: In this part, the principle is only explained for an actuator which can be faulty. Assuming that the estimation scheme offers a continuous output, namely: $\omega = \{\widehat{b_0}, \widehat{b_1}\}$ and a probability density function; that is even with respect to the errors between on the one hand b_0 and $\widehat{b_0}$ and on the other hand b_1 and $\widehat{b_1}$, the role of the parameters b_0, b_1 and the role of the estimates $\widehat{b_0}, \widehat{b_1}$, are exchanged.

Thus, assuming $p(b_0, b_1|\omega) = p(\omega|b_0, b_1)$, the observation probability is given by integrating this quantity over the domain corresponding to each mode, namely:

$$p(\omega|m) = \int\int f_m(b_0, b_1) p(\omega|b_0, b_1) \mathrm{d}b_0 \mathrm{d}b_1 \quad (6)$$

where $f_m(b_0, b_1)$ depend on the mode. In order to design an estimation scheme able to determine the most probable current mode, several assumptions are made on the distribution of the parameters b_0 and b_1, for each mode.

For example, in this paper, the hypothesis for the actuator fault modes are the following:

- Nominal case:

$$f_m(b_0, b_1) = \delta(b_0)\delta(b_1) \quad (7)$$

- Locking case:

$$f_m(b_0, b_1) = \frac{1}{\overline{u} - \underline{u}} H(b_0 - \underline{u}) H(\overline{u} - b_0)\delta(b_1) \quad (8)$$

- Loss of effectiveness case:

$$f_m(b_0, b_1) = H(b_1) H(1 - b_1)\delta(b_0) \quad (9)$$

III. Reconfiguration method

A. Estimation

1) Filtering state and fault model parameters: In order to estimate the current mode of the system, the proposed methods consists in estimating the fault parameters b_0^i, b_1^i, c_0^i and c_1^i, which, in turn, necessitates to estimate also the state $\mathbf{x}$ of the system. The fault parameters estimates will be exploited to deduce the most probable current mode, which will be used as the observation input for the POMDP.

To estimate the state and these parameters, several methods

exist. They are usually based on models, and this is the choice made in this paper in the hope that using an accurate model will lead to improved performances. For the estimation of the actuator parameters, the following methods can be used: observer method, Kalman filter or particle filter.
For the estimation of the sensor parameters, a non-linear method is required. For example, the following methods can be used: extended Kalman filter, unscented Kalman filter or particle filer.
However, the random walks of equation (4) are not fully consistent with an abrupt fault occurrence. Thus, the classical filtering methods are modified in order to test the likelihood of each $\mathbf{z}_k$ and to flatten the probability density function of b_0^i, b_1^i, c_0^i and c_1^i in case of an unlikely $\mathbf{z}_k$.

2) Estimating fault mode probabilities: At each time period k, the system is in some state $s_k \in S$. The FTC takes an action $a_k \in A$, which causes the system to transition to a new state $s_{k+1} \in S$ with probability $T(s_{k+1}|s_k, a_k)$. Then, an observation $\omega \in \Omega$ is computed from the probability density function managed by the continuous filter. The FTC updates a vector made of probabilities of the system being in each discrete state, $b(s)$. This vector, named belief state, is updated using the Bayes rule. In general, the update $b^{a,\omega}(s')$ depends on the action a, the observation ω, and is function of the previous belief state ($s' = s_{k+1}$ follow the state $s = s_k$)

$$b^{a,\omega}(s') = \frac{p(\omega|s')p(s'|b,a)}{p(\omega|b,a)} \tag{10}$$

$$\text{where : } \begin{cases} p(s'|b,a) = \sum_s p(s'|s,a)b(s) \\ p(\omega|b,a) = \sum_{s'} p(\omega|s')p(s'|b,a) \end{cases}$$

As explained previously, the transitions between the modes are not dependant on the controller. Thus, the equation (10) can be reduced as follows:

$$b^{\omega}(m') = \frac{p(\omega|m')p(m'|b)}{p(\omega|b)} \tag{11}$$

$$\text{where : } \begin{cases} p(m'|b) = \sum_m p(m'|m)b(m) \\ p(\omega|b) = \sum_{m'} p(\omega|m')p(m'|b) \end{cases}$$

and $p(\omega|m)$ is given by equation(6).

B. Switching Decision Optimization

1) Discretisation of the observation domain: In order to use classical POMDP framework, one needs to discretise the observation domain to recover a discrete observation.

This can be done mostly in two ways:

- by forcing an *a priori* discretisation on the observation space,
- by finding a partition such that each different subdomain leads to a change in the resulting policy [11].

Let $\{\Omega^1, ..., \Omega^{|\Omega|}\}$ a partition of the continuous space in which the ω parameters evolves. Then, the probability to have the parameters in a given set Ω^i is given by:

$$p(\Omega^i|m) = \int_{\omega \in \Omega^i} p(\omega|m)\mathrm{d}\omega \tag{12}$$

Note that this is a multiple integral. For instance, for a single actuator which can be faulty, it is a double integral with $\mathrm{d}\omega$ standing for $\mathrm{d}\widehat{b_0}\mathrm{d}\widehat{b_1}$. Finally, to this $p(\Omega^i|m)$ corresponds a $p(\Omega^i|s)$.

2) Dynamic Programming: In the previously defined framework, the goal is to construct a policy $\pi : \Delta \rightarrow A$ (Δ the set of belief state) which minimize a criteria. The objective is to minimize the cost R.

Equation 13 represents the problem to be solved:

$$V_k(b) = \min_{a \in A}[r(b,a) + \gamma \sum_{i=1}^{|\Omega|} p(\Omega^i|b,a)V_{k+1}(b^{a,\Omega^i})] \tag{13}$$

where $\gamma \in [0,1[$ is a discount factor, and

$$\begin{cases} r(b,a) = \sum_s \sum_{s'} b(s)R(s,a,s')p(s'|s,a) \\ p(\Omega^i|b,a) = \sum_{s'} p(\Omega^i|s')p(s'|b,a) \\ b^{a,\Omega^i}(s') = \frac{p(\Omega^i|m')p(m'|b)}{p(\Omega^i|b)} \end{cases}$$

This problem can be solved over a finite horizon from time k to time $k + H$. The policy π specifies the action a which maximizes the value function $V_k(b)$

Equation (13) is solved by:

1) expending from the current belief state the tree of possible b values in function of all possible future actions and discrete observations and,
2) computing backwards the value of V.

In order to compensate for a small value for H, the quantity: $\sum_{k'=k+H+1}^{K} R(s_k, k, s_{k+1})$ with a constant controller is used for initializing V_{k+H+1}.

IV. Controllers For Actuator Faults

As discussed previously, there are $(1 + m_{actuator} \times l + m_{sensor} \times m)$ controllers k, one for each mode m. Each controller sends different control signal, $\mathbf{u}_k$, in order the effective control $\mathbf{e}_k$ to be like:

$$\mathbf{e}_k = -K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \tag{14}$$

The information of $\mathbf{x}_k$ is not available, thus the state estimate $\widehat{\mathbf{x}}_k$ is used in the control law.

A. Adaptivity

The controllers for the faulty modes should be adaptive, based on the estimation of the fault parameters $\widehat{\mathbf{b_0}}_k$ and $\widehat{\mathbf{b_1}}_k$ as shown in Fig. 6.

With no fault, there is no need to have an adaptive controller because $\mathbf{b_0}$ and $\mathbf{b_1}$ are null. Moreover, $\widehat{\mathbf{b_0}}$ and $\widehat{\mathbf{b_1}}$ are noisy because they are estimates. So, in the nominal configuration, a controller without the estimated parameters seem to be more adapted than an adaptive controller which is noisy.

B. Nominal Controller k_0

In the nominal case, all the actuators are nominal, i.e:

$$\forall i, \begin{cases} b_{0k}^i = 0 \\ b_{1k}^i = 0 \end{cases}$$

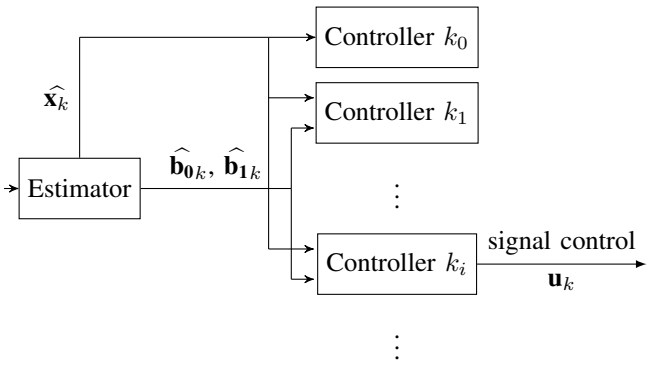


Fig. 6. Adaptative Controller

Thus, $\mathbf{e}_k = \mathbf{u}_k$ and so the nominal controller is as follows:

$$\mathbf{u}_k = -K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \tag{15}$$

C. Controllers for locking modes

By definition, in the locking case, one actuator, the j^{th} one, is locked at some value δ_0, while the others are nominal, i.e:

$$\forall i \neq j, \begin{cases} b^i_{0k} = 0 \\ b^i_{1k} = 0 \\ b^j_{0k} = \delta_0 \in [\underline{u}, \overline{u}] \\ b^j_{1k} = 1 \end{cases}$$

Thus, $\mathbf{e}_k = \begin{bmatrix} u^1_k & \cdots & u^{j-1}_k & \delta_0 & u^{j+1}_k & \cdots & u^l_k \end{bmatrix}^T$.

To be able to compensate for the locked actuator, it is assume that the action of this actuator is a linear combination of other actuators, i.e. the j^{th} column of B is a linear combination of the other columns, i.e. $B_{.j} = \sum_{i \neq j} \lambda_i B_{.i}$, with $B = [B_{.1} \cdots B_{.i} \cdots B_{.l}]$.

On the one hand, the control implies:

$$\begin{cases} B\mathbf{e}_k &= B_{.j}\delta_0 + \sum_{i \neq j} B_{.i} u^i_k \\ &= \sum_{i \neq j} B_{.i}(\lambda_i \delta_0 + u^i_k) \end{cases}$$

On the other hand, the requirement implies:

$$\begin{cases} B\mathbf{e}_k &= -BK(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \\ &= -\sum_{i \neq j} B_{.i}(\lambda_i K_{j.} + K_{i.})(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \end{cases}$$

with $K^T = [K^T_{1.} \cdots K^T_{i.} \cdots K^T_{l.}]$.

Identifying the terms in $B_{.i}$ gives:

$$\mathbf{u}_k = \begin{bmatrix} -(\lambda_1 K_{j.} + K_{1.})(\widehat{\mathbf{x}}_k - \mathbf{r}_k) - \lambda_1 \delta_0 \\ \vdots \\ \widehat{b_0} \\ \vdots \\ -(\lambda_l K_{j.} + K_{l.})(\widehat{\mathbf{x}}_k - \mathbf{r}_k) - \lambda_l \delta_0 \end{bmatrix} \tag{16}$$

However, with this controller, there is an observability problem. If the j^{th} actuator is no longer locked, the control signal still send the value $\widehat{b_0}$ and thus, the estimator do not detect the restoration of the actuator. To solve this problem, an input signal, such as a 3211 input signal [12], could be added to $\widehat{b_0}$ in order to make the system observable. This signal will have no effect if the j^{th} actuator is actually locked, and in the opposite case, it can solve the problem of observability.

D. Controllers for loss of effectiveness modes

In the loss of effectiveness case, the j^{th} actuator is reduced by a factor γ_1 and the others are nominal, i.e:

$$\forall i \neq j, \begin{cases} b^i_{0k} = 0 \\ b^i_{1k} = 0 \\ b^j_{0k} = 0 \\ b^j_{1k} = \gamma_1 \in]0, 1[\end{cases}$$

Thus,
$\mathbf{e}_k = \begin{bmatrix} u^1_k & \cdots & u^{j-1}_k & (1-\gamma_1)u^j_k & u^{j+1}_k & \cdots & u^l_k \end{bmatrix}^T$, and so the associated controller is chosen as follow:

$$\mathbf{u}_k = \begin{bmatrix} -K_{1.}(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \\ \vdots \\ \frac{-K_{j.}(\widehat{\mathbf{x}}_k - \mathbf{r}_k)}{1 - \widehat{b_{1k}}^i} \\ \vdots \\ -K_{l.}(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \end{bmatrix} \tag{17}$$

V. Illustrative example

A. Model

An academic model is used to illustrate the proposed reconfiguration framework. The state $\mathbf{x}$ is composed of the position and the velocity of a mobile. Two actuators are used, $\mathbf{u} = \begin{bmatrix} u^1 & u^2 \end{bmatrix}^T$, such as the acceleration is equal to the sum of effects of the actuators, $e^1 + e^2$. The control inputs u^1, u^2 and e^1, e^2 are bounded to $[-10, 10]$. The system has two sensors, one for the position and another for the speed. Using a one second sampling time, the discretised model is the following:

$$\begin{cases} \mathbf{x}_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} 0.5 & 0.5 \\ 1 & 1 \end{bmatrix} \mathbf{e}_k + \mathbf{w}^x_k \\ \mathbf{y}_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x}_k + \mathbf{v}_k \end{cases} \tag{18}$$

The reference signal to be followed by the mobile, $\mathbf{r}$, for an horizon of 100 seconds, is presented in the upper part of Fig. 8, for r^1 and in the upper part of Fig. 9 for r^2.

Only faults of the second actuator are considered:

$$\begin{cases} e^1_k = u^1_k \\ e^2_k = b_{0k} + (1 - b_{1k})u^2_k \end{cases} \tag{19}$$

Parameters used in the simulation are as follows. $Q^x_k = diag\{0.05^2, 0.05^2\}$, $Q^b_k = diag\{0.005^2, 0.005^2\}$, and $R_k = diag\{1^2, 0.1^2\}$.

In this example, there are three modes: nominal m_0, locking m_1, and loss of effectiveness m_2. Each mode will have an associated controller k_i. Thus, we have nine discrete states: $S = \{s_0, ...s_8\} = \{\{m_0, k_0\}, \{m_0, k_1\} ... \{m_2, k_2\}\}$. The mode transition graph is the one of Fig. 5 with:

- $p(m_0|m_0) = e^{-10^{-5}}$
- $p(m_1|m_0) = p(m_2|m_0) = (1 - p(m_0|m_0))/2$
- $p(m_0|m_1) = p(m_0|m_2) = 10^{-4}$
- $p(m_1|m_1) = p(m_2|m_2) = (1 - 10^{-4})^2$
- $p(m_2|m_1) = p(m_1|m_2) = 10^{-4}(1 - 10^{-4})$

B. Controllers

Based on the IV. part, the three controllers are:

- the nominal one k_0 associated to the nominal mode m_0:

$$\begin{bmatrix} u_k^1 \\ u_k^2 \end{bmatrix} = \begin{bmatrix} -K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \\ -K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \end{bmatrix} \tag{20}$$

- the controller k_1 associated to the locking mode m_1:

$$\begin{bmatrix} u_k^1 \\ u_k^2 \end{bmatrix} = \begin{bmatrix} -2K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) - \widehat{b_0} \\ \widehat{b_0} \end{bmatrix} \tag{21}$$

- the controller k_2 associated to the loss of effectiveness mode m_2:

$$\begin{bmatrix} u_k^1 \\ u_k^2 \end{bmatrix} = \begin{bmatrix} -K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) \\ \frac{-K(\widehat{\mathbf{x}}_k - \mathbf{r}_k) - \widehat{b_0}}{1 - \widehat{b_1}} \end{bmatrix} \tag{22}$$

with $K = \begin{bmatrix} 0.9 & 0.8 \end{bmatrix}$.

C. Cost function

For this example, the cost is calculated as follow:

$$E_{/s}||\mathbf{r}_k - \mathbf{x}_k|| = E_{/s} \sum_{k'=k}^{100} |x_{k'}^1 - r_{k'}^1| + |x_{k'}^2 - r_{k'}^2| \tag{23}$$

where $E_{/s}$ indicates the expectation with respect to the discrete state. For the nominal mode, the expectation is computed by performing 100 runs with different values for noise sequences and averaging the cost values. For non nominal modes, the 100 runs are repeated 19 and 9 times, for respectively the locking and the loss of effectiveness modes, varying the fault parameter. The average is then computed considering the 19×100 runs for variation of b_0 and 9×100 runs for variation of b_1. Fig. 7 shows the evolution of the cost.

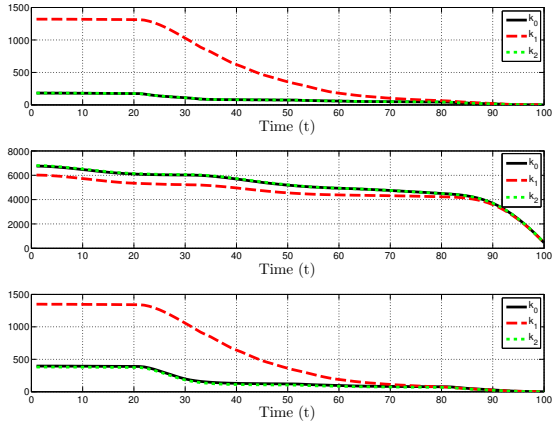


Fig. 7. Cost

For the nominal case shown in the upper part of Fig. 7:

- The controller k_0 and k_2 are very close. Indeed, b_0 and b_1 are null, thus k_2 controls a loss of effectiveness close to zero, which leads to a controller similar to k_0. The difference comes from the noise.
- The controller k_1 is not good before $70s$, but after the three controllers have similar performance. The bad performance of the controller k_1 at the start of the simulation comes from the saturation of the command.

For the locking case shown in the middle part of Fig. 7:

- The controllers k_0 and k_2 have the same performance. Indeed, the first control is the same in the two controllers and the second control is locked.
- The controller k_1 has better performance all along the horizon because the locked actuator is compensated by the other.

For the loss of effectiveness case shown in the lower part of Fig. 7:

- The controller k_2 has the best performance result at the start of the simulation.
- The controller k_1 is not good before $70s$ because of the saturation of the control. After the three controllers have similar performance.

Arbitrary penalties, $P(k_0, a_1) = P(k_0, a_2) = 1$, $P(k_1, a_0) = P(k_1, a_2) = 12.5$ and $P(k_2, a_0) = P(k_2, a_1) = 4$, are added in order to avoid sudden changes of controller at the end of the simulation. Indeed, in Fig. 7, it is shown that some costs are similars.

D. Estimation method

To estimate the state $\mathbf{x}$ and the parameters $\mathbf{b_0}$ and $\mathbf{b_1}$, an augmented system is considered, which includes the system "natural" state -position, velocity- with equations 18, together with the fault parameters. Then a Kalman filter is applied to the resulting augmented system, which may be described by:

$$\begin{cases} \mathbf{X}_{k+1} = A^{aug}\mathbf{X}_k + B^{aug}\mathbf{u}_k + \mathbf{W}_k^{aug} \\ \mathbf{y}_k = C^{aug}\mathbf{X}_k + \mathbf{v}_k \end{cases} \tag{24}$$

with $A^{aug} = \begin{bmatrix} A & B_{.,2} & D_{k_{.,2}}(\mathbf{u}_k) \\ 0_{1,2} & 1 & 0 \\ 0_{1,2} & 0 & 1 \end{bmatrix}$, $B^{aug} = \begin{bmatrix} B \\ 0_{1,2} \\ 0_{1,2} \end{bmatrix}$, $C^{aug} = \begin{bmatrix} C & 0_{2,1} & 0_{2,1} \end{bmatrix}$, and $\mathbf{W}_k^{aug} = \begin{bmatrix} \mathbf{w}_k^x \\ \mathbf{w}_k^{b_0} \\ \mathbf{w}_k^{b_1} \end{bmatrix}$

In the estimation algorithm, the covariance $P(k+1|k)$ is reopened when the following test is not passed:

$\forall i \in [1, m]$,

$$(C_{i.}P(k+1|k)C_{i.}^T + R_{ii})^{-1}(y_{k+1}^i - C_{i.}\mathbf{X}(k+1|k))^2 \leq 5^2 \tag{25}$$

The covariance is reopened as follows:

$$P(k+1|k)_{reset} = \begin{bmatrix} P(k+1|k)^x & 0_{2,1} & 0_{2,1} \\ 0_{1,2} & 10 & 0 \\ 0_{1,2} & 0 & 1/5^2 \end{bmatrix} \tag{26}$$

E. Conditional Observation probabilities

Applying Equations 6 to 9 to the example leads to:

$$p(\omega|m_0) = \frac{1}{2\pi\sqrt{|P|}} e^{-\frac{1}{2}\widehat{b}^T P^{-1}\widehat{b}} \tag{27}$$

$$P(\omega|m_1) = \frac{1}{40\pi\sqrt{|P|}} \int_{-10}^{10} e^{-\frac{1}{2}\left(\begin{bmatrix} b_0 \\ 1 \end{bmatrix} - \widehat{b}\right)^T P^{-1} \left(\begin{bmatrix} b_0 \\ 1 \end{bmatrix} - \widehat{b}\right)} \mathrm{d}b_0 \tag{28}$$

$$P(\omega|m_2) = \frac{1}{2\pi\sqrt{|P|}} \int_{0}^{1} e^{-\frac{1}{2}\left(\begin{bmatrix} 0 \\ b_1 \end{bmatrix} - \widehat{b}\right)^T P^{-1} \left(\begin{bmatrix} 0 \\ b_1 \end{bmatrix} - \widehat{b}\right)} \mathrm{d}b_1 \tag{29}$$

with P the 2×2 matrix at the lower right part of $P(k+1|k)$.

F. Partition of Ω and Dynamic Programming horizon

To avoid a long computational time, Ω is partitioned in four areas, and the horizon is $H = 5$. Ω is partitioned as follows:

- $b_0 \in [-0.1, 0.1]$ and $b_1 \in [0, 0.1]$
- $b_0 \in [-0.1, 0.1]$ and $b_1 \in [0.1, 0.9]$
- $b_0 \in [-0.1, 0.1]$ and $b_1 \in [0.9, 1]$
- $b_0 \in [-10, -0.1[\cup]0.1, 10]$ and $b_1 \in [0.9, 1]$

G. Fault scenario

To test the design, the scenario including a loss of effectiveness case and a locking case, as shown in Table I.

Event time	Event Type	b_0	b_1
$20s$	Loss of effectiveness	0	0.5
$50s$	Restoration	0	0
$70s$	Locking	5	1

TABLE I
FAULT SCENARIO

H. Results

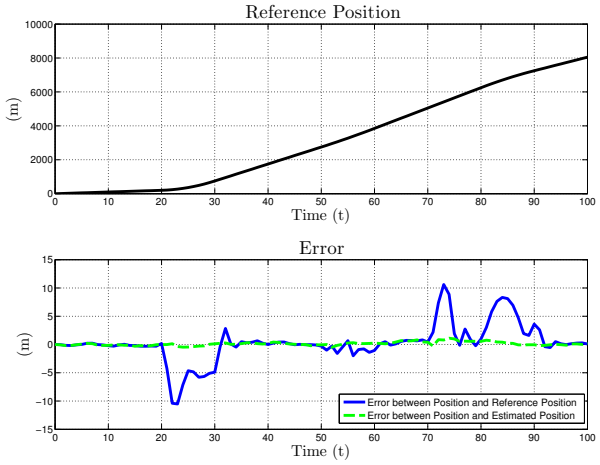


Fig. 8. Position

1) Estimation of state and fault parameters: The dashed curves in lower parts of Fig. 8 and Fig. 9 presents the state estimation error. These errors are small, with the exception of a peak at time $70s$. The locking event induces a temporary error of about $3m/s$ for velocity and $1m$ for position.

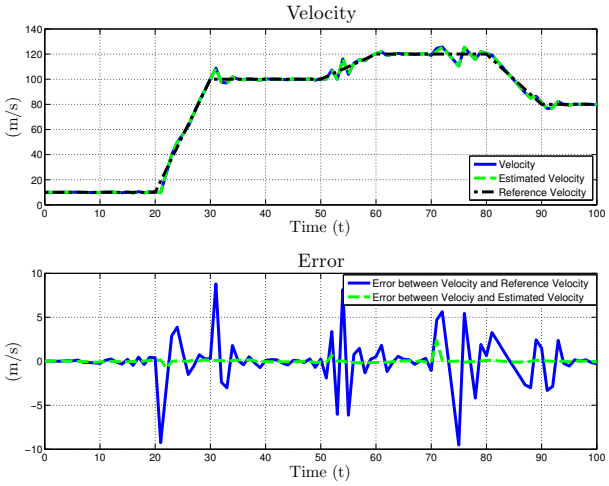


Fig. 9. Velocity

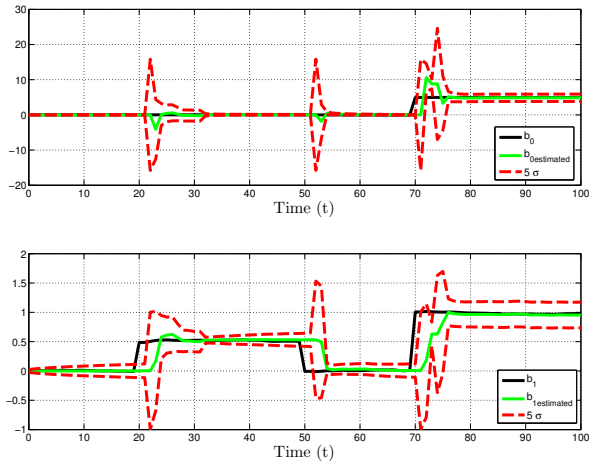


Fig. 10. Fault parameters: upper part b_0 and lower part b_1

The values b_0, b_1 and their estimates are shown in Fig. 10. The estimator detects the fault in less than $10s$. For each event, the covariance is reopened to detect the fault. For the locking event, it is reopened twice, indicating an adaptation. This is more difficult than for the other events.

2) Probabilities: The probabilities of the modes are shown in Fig. 11. The probability of the most likely mode is close to one, except for a short time during the transition.

3) Action: As shown in Fig. 12, the controller is changed as follows:

- When the loss of effectiveness is detected, at $23s$, the controller is changed at the nest step, at $24s$. Indeed the effectiveness of the nominal controller k_0 in the case of loss of effectiveness is lower than the one of the associated controller k_2.
- When the effectiveness of the actuator is back, the controller is not changed from k_2 to k_0 because the reward is not large enough compared to the penalty associated to any change of controller.

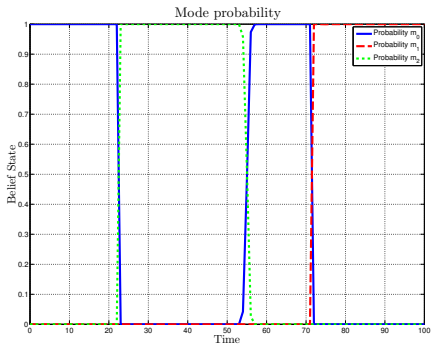


Fig. 11. Probabilities of modes

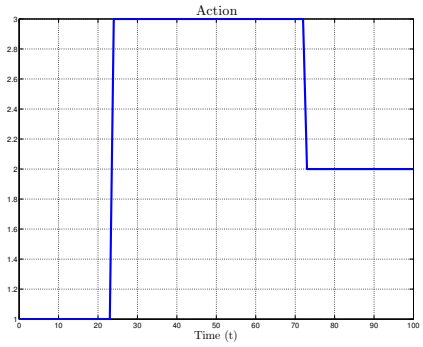


Fig. 12. Action

- When the actuator is locked, at $72s$, the controller is changed at the next step, at $73s$.

4) Control: Fig. 13 presents the control for the two actuators. Between $20s$ and $50s$, the actuator is reduced of 50%. After $70s$, it is locked to $5m/s^2$. The effective control e^2 is over estimated for a short time after the locking event.

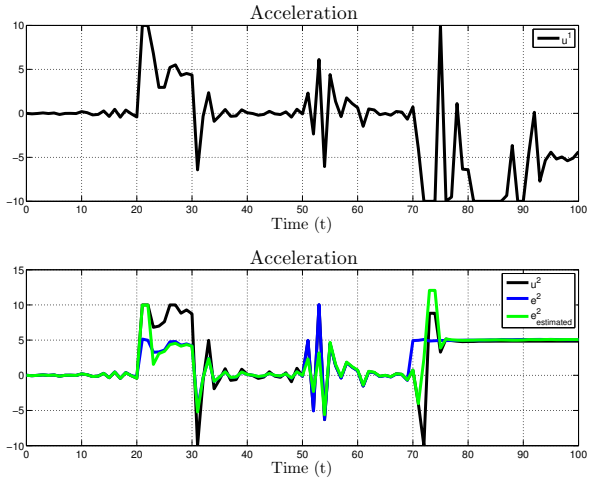


Fig. 13. Desired and effective control inputs

5) Tracking the reference: The plain curves in the lower part of Fig. 8 and Fig. 9 present the tracking error for respectively position and velocity. The errors are between -12 and $+12$ and are most of the time close to zero, indicating that despite the presence of faults the system state follows the reference leading to an acceptable performance.

Finally, with this FTC method, a gain of 25% on the sum of tracking errors is obtained with respect to the use of only the nominal controller.

VI. Conclusion

In this article, a modelling framework is defined for actuator and sensor faults. This modelling combines discrete and continuous frameworks, allowing to take profit of the advantages of discrete and continuous methods. A reconfiguration method is developed for the actuator faults and is tested on an academic example. This method has shown good results; in the event of component faults, this method improves significantly the performance with respect to the use of a single controller. The framework proposed here can be the basis for additional researches about:

- Addition of input signals to the controller for locking modes in order to make the fault parameters observable.
- Reconfiguration for sensor faults.
- Computation time reduction for the choice of the action by solving a simplified but still relevant discrete optimisation problem.
- Empiric assessment on a large number of scenarios with several devices, possibly faulty.

References

[1] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," *Annual reviews in control*, vol. 32, no. 2, pp. 229–252, 2008.

[2] F. Liao, J. L. Wang, and G.-H. Yang, "Reliable robust flight tracking control: an LMI approach," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 1, pp. 76–89, 2002.

[3] K.-S. Kim, K.-J. Lee, and Y. Kim, "Reconfigurable flight control system design using direct adaptive method," *Journal of guidance, control, and dynamics*, vol. 26, no. 4, pp. 543–550, 2003.

[4] R. Isermann, "Model-based fault-detection and diagnosis–status and applications," *Annual Reviews in control*, vol. 29, no. 1, pp. 71–85, 2005.

[5] F. Caliskan and C. Hajiyev, "Active fault-tolerant control of UAV dynamics against sensor-actuator failures," *Journal of Aerospace Engineering*, p. 04016012, 2016.

[6] Y. M. Zhang and J. Jiang, "Active fault-tolerant control system against partial actuator failures," *IEE proceedings-Control Theory and applications*, vol. 149, no. 1, pp. 95–104, 2002.

[7] I. Hwang, S. Kim, Y. Kim, and C. E. Seah, "A survey of fault detection, isolation, and reconfiguration methods," *IEEE Transactions on Control Systems Technology*, vol. 18, no. 3, pp. 636–653, 2010.

[8] D. Rotondo, F. Nejjari, and V. Puig, "A virtual actuator and sensor approach for fault tolerant control of LPV systems," *Journal of Process Control*, vol. 24, no. 3, pp. 203–222, 2014.

[9] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1, pp. 99–134, 1998.

[10] P. M. Freeman, "Reliability assessment for low-cost unmanned aerial vehicles," Ph.D. dissertation, UNIVERSITY OF MINNESOTA, 2014.

[11] J. Hoey and P. Poupart, "Solving pomdps with continuous or large discrete observation spaces," in *IJCAI*, 2005, pp. 1332–1338.

[12] G. Heredia, B. Remu, A. Ollero, R. Mahtani, and M. Musal, "Actuator fault detection in autonomous helicopters," in *Proceedings of the 5th IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon, Portugal*, 2004, pp. 569–574.