Fault-tolerant model predictive control within the hybrid systems framework: Application to sewer networks

Carlos Ocampo-Martinez[1, *, †] and Vicenç Puig[2]

[1]*Institut de Robòtica i Informàtica Industrial (IRI), Spanish National Research Council (CSIC), Technical University of Catalonia (UPC), Llorens i Antigas, 4-6, 08028 Barcelona, Spain*
[2]*Automatic Control Department, Campus de Terrassa, Technical University of Catalonia, Rambla Sant Nebridi, 10, 08222 Terrassa, Spain*

SUMMARY

In this paper, model predictive control (MPC) problem with fault-tolerance capabilities is formulated within the hybrid systems framework. In particular, the mixed logical dynamic form to represent hybrid systems is considered. Using this approach, a hybrid model of the system to be controlled is obtained, which includes inherent hybrid phenomena and possible modes caused by faults occurrence. This allows to adapt the system model on-line by taking into account the fault information provided by a fault diagnosis and isolation module. In this way, the controller can cope with the considered faults. Additionally, different implementation schemes and fault-tolerance evaluation procedures for hybrid MPC (HMPC) considering fault-tolerance capabilities are proposed and discussed. Finally and in order to exemplify the implementation of the proposed approach, it is applied to include actuator fault tolerance in the design of a HMPC controller for a portion of the sewer network of Barcelona.

KEY WORDS: fault-tolerant control; MPC; hybrid systems; MLD; actuator faults; sewer networks

1. INTRODUCTION

Fault-tolerant control (FTC) is a relatively new idea recently introduced in the research literature [1], which allows to have a control loop that fulfills its objectives (may be with a possible degradation) when faults in system components (instrumentation, actuators and/or plant) appear.

*Correspondence to: Carlos Ocampo-Martinez, Institut de Robòtica i Informàtica Industrial (IRI), Spanish National Research Council (CSIC), Technical University of Catalonia (UPC), Llorens i Antigas, 4-6, 08028 Barcelona, Spain.
†E-mail: cocampo@iri.upc.edu

Contract/grant sponsor: Spanish Science and Technology Ministry; contract/grant number: CICYT Ref. DPI2006-11944
Contract/grant sponsor: *Juan de la Cierva* Research Program; contract/grant number: Ref. JCI-2008-2438
Contract/grant sponsor: DGR of Generalitat de Catalunya; contract/grant number: 2005/SGR/00537

A closed-loop control system can be considered as fault tolerant when it includes either adaptation strategies for its control law or redundancy mechanisms in its sensors and/or actuators. From the point of view of the control strategies, the literature considers two main groups. On the one hand, the *passive* techniques are based on the design of control laws that take into account the fault appearance as a system perturbation. In this case, the control algorithm is designed to achieve the control objectives either in healthy or in faulty situations using robust control techniques. Thus, within certain margins, the control law has inherent fault-tolerant capabilities, allowing the system to cope with the fault presence. In works such as [2–6], among many others, complete descriptions of passive FTC techniques can be found. On the other hand, the *active* FTC techniques, which are the ones used in this paper, consist in adapting the control loop using the information given by the fault detection and isolation (FDI) module within the fault-tolerant architecture (see [1]). Using this information, some automatic adjustments are done trying to reach the control objectives. See [7] for a complete review of active FTC.

A fault can be viewed as a discrete event affecting the system by changing some of its particular properties (either the structure or/and the parameters). Then, in turn, an active FTC should detect and isolate the fault and, if possible, estimate its magnitude (fault diagnosis) through the FDI module, and adapt the controller to the faulty situation such that the control objectives could be satisfied even in the fault presence (control re-design). In particular, the whole active FTC scheme can be expressed using the three-layer architecture for FTC systems proposed by Blanke *et al.* [1], where the first layer corresponds to the control loop, the second layer corresponds to the fault diagnosis and accommodation modules while the third layer is related to the supervisor system. Some important reasons for splitting FTC systems by layers are the clear development structure determination, the independent specification and development of each layer, and the capacity of testing the detection and supervision functions/modules. However, there does not exist any guarantee for the proper operation of the whole FTC system when the integration of its subsystems/layers is done. Because of the discrete event nature of the fault occurrence and the controller re-design actions, an FTC system can be considered as a *hybrid system* by nature, what makes not trivial its analysis and design. This has been already noticed in [8], where the three levels identified by Blanke *et al.* [1] are named as *continuous-state level* (which would correspond to the control level), *interface level* (associated with the fault diagnosis and accommodation level) and the *discrete-event level* (which is related to the supervision level). However so far, the hybrid nature has been traditionally neglected in order to facilitate a simple design, reliable implementation and systematic testing.

One of the main objectives of this paper consists in embedding the active fault-tolerant design of controllers based on model predictive control (MPC) within the hybrid system framework considering the FTC hybrid nature. There are several approaches in the literature to handle hybrid systems, see [9]. Here, the hybrid system representation considered is the *Mixed Logical Dynamical* (MLD) form, introduced in [10]. In particular, a hybrid model of the system to be controlled including faulty modes is proposed. Then, an active fault-tolerant *hybrid MPC* (AFTHMPC) controller is designed using hybrid systems modeling and control methodologies. This will allow the hybrid HMPC (HMPC) to take into account the fault information provided by an FDI module automatically doing on-line the controller re-design. The fault information will act as a discrete event (discrete input) that will switch the controller mode in case of fault enabling the controller re-design. In other words, the proposed FTC design methodology allows to design the three layer of an FTC system in an integrated manner and verify the global behavior of the whole FTC system using hybrid system theory.

Table I. List of acronyms used throughout the paper.

Acronym	Meaning
MPC	Model predictive control
MLD	Mixed logical dynamic
FTC	Fault-tolerant control
HMPC	Hybrid MPC
AFTHMPC	Active fault-tolerant HMPC
PFTHMPC	Passive fault-tolerant HMPC
FDI	Fault diagnosis and isolation
PWA	Piece-wise affine
HYSDEL	Hybrid systems description language
MIP	Mixed integer programming
MILP	Mixed integer linear programming
MIQP	Mixed integer quadratic programming
MIPC	Mixed integer predictive control
CSP	Constraint satisfaction problem
CSO	Combined sewer overflow

The paper is structured as follows. In Section 2, the parallelism existing between active FTC scheme and hybrid systems is presented, which allows to embed the active FTC design in the hybrid framework. Section 3 presents the MLD formalism to represent hybrid systems and the MPC control using such formalism (either using the implicit or explicit form). In Section 4, implicit and explicit inclusion of fault-tolerance in HMPC controllers is discussed. In Section 5, the possible ways of implementation for the HPMC controllers are discussed. In Section 6, fault-tolerance evaluation of HMPC schemes are proposed and discussed. In Section 7, the proposed design and implementation of fault-tolerant HMPC schemes are applied over a case study based on a piece of the Barcelona sewer network. Finally, conclusions are presented in Section 8.

Since throughout the paper an important number of acronyms is used, Table I is presented with the intention to offer a full understanding and comfort to the reader during the reading of this paper. Within this table are listed the most used acronyms.

2. HYBRID SYSTEMS APPROACH TO FTC

2.1. Hybrid systems

In recent years, hybrid systems have attracted attention of both academia and industry. The name hybrid comes from the fact that those systems combine continuous dynamics described by differential equations with discrete dynamics described by finite state machines (FSM), if-then-else rules, propositional and temporal logic [10]. Such heterogeneous models, denoted as *hybrid models*, switch among many operating modes, where each mode is associated with a different dynamic law, and mode transitions are triggered by events, like states crossing pre-specified thresholds. The practical relevance of hybrid models is twofold. First, profitability of logic controllers embedded in a continuous environment (for instance in the automotive industry) can be increased. Second, many physical phenomena admit a natural hybrid description like circuits integrating switching elements, biomolecular networks and TCP/IP networks.

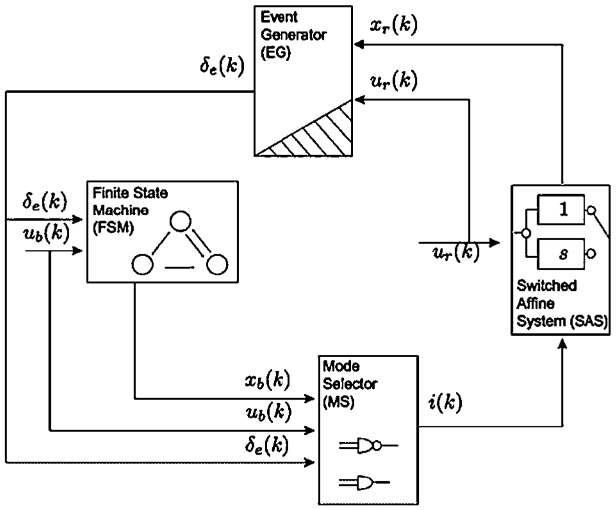


Figure 1. Discrete hybrid automata (taken from [12]).

Several classes of hybrid systems have been proposed in the literature, each class is usually tailored to solve a particular problem. Timed and hybrid automata have proved to be a successful modeling framework for formal verification, and have been widely used in the literature [11]. The starting point for both models is a FSM equipped with continuous dynamics.

The type of hybrid systems considered in this paper are those that can be described by a *discrete hybrid automata* (DHA); see [12]. DHA results from the connection of an FSM, which provides the discrete part of the hybrid system, with a *switched affine system* (SAS), which provides the continuous part of the hybrid dynamics (see Figure 1). The interaction between these two parts is based on two connecting elements: *the event generator* (EG) and the *mode selector* (MS). The EG extracts logic signals from the continuous part. Those logic events and other exogenous logic inputs trigger the switch of the FSM state. The MS combines all the logic variables (states, inputs and events) to choose the mode (continuous dynamics) of the SAS. Continuous dynamics are expressed as linear affine difference equations. DHA models are a mathematical abstraction of the features provided by many computational oriented and domain specific hybrid frameworks: MLD models [10], *piecewise affine* (PWA) systems [13], *linear complementarity* (LC) systems [14] and *max-min-plus-scaling* (MMPS) systems [15].

2.2. *FTC using a hybrid systems approach*

As discussed in the introduction, active FTC relies on including an on-line FDI module and a supervisor module within the fault-tolerant architecture. When a fault is detected and isolated by the FDI module, the supervisor reacts by activating some remedial actions such that the faulty control loop still satisfies the given specifications. As discussed in Section 1, an architecture for active fault tolerance, which is constituted by three layers, has been proposed in [1]:

- Layer 1 (*Control Loop*), which comprises a traditional control loop with sensor and actuator interfaces, signal conditioning and filtering and the controller. The solution of a control problem consists in finding a control law in a given set of *control laws* $\mathscr{U}$ such that the controlled system achieves the *control objectives* $\mathscr{O}$ while its behavior satisfies a set of *constraints* $\mathscr{C}$. Thus, the solution of the problem is completely defined by the triple $\langle \mathscr{O}, \mathscr{C}, \mathscr{U} \rangle$.

- Layer 2 (*Fault Diagnosis and Accommodation*), which comprises an FDI module and a closed-loop re-design module which, in turn, defines the remedial actions given by the supervisor. The functions of this module are: detection based on either physical or analytic redundancy using FDI methods, detection of faults in control algorithms and application of software and effector modules to execute the fault accommodation actions.
- Layer 3 (*Supervision*). The supervisor is a discrete-event dynamical system (DEDS) that allows to consider the state-event logic to describe the state (mode) of the controlled system and whose transitions between its states are induced by events (faults). When the system is in a faulty mode, a set of pre-established remedial actions should be activated in order to correct and recover the faulty system and then to achieve the control objective.

As already noticed in [8] and pin pointed in the introduction, comparing this structure with a conceptual scheme of a hybrid system (see Figure 2), there is a quite precise correspondence. In particular, the control loop level matches the continuous part of the hybrid system, the FDI and re-design level matches the interface between continuous and discrete system dynamics. Finally, the supervision level matches the discrete dynamics part of the hybrid system. Moreover, the events might be associated with faults within the FTC architecture and re-design actions are related to changes in the operating mode of the continuous part. This leads to the use of modeling and control methodologies that allow to consider both continuous and discrete dynamics. The continuous part is associated with each particular operating mode and typically can be modeled using physical first principles. On the other hand, the discrete part comes from logic conditions that establish

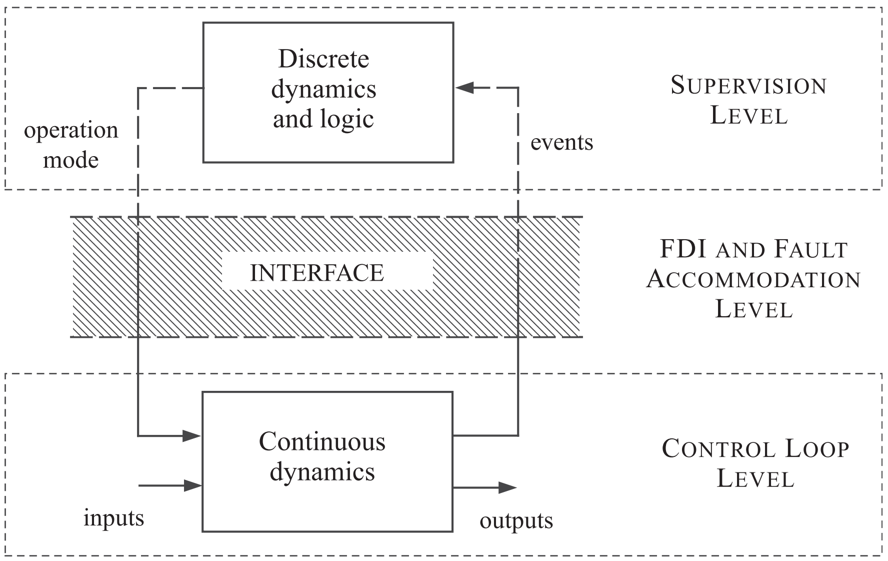


Figure 2. Parallelism between the basic scheme of a hybrid system and the three levels of an FTC architecture.

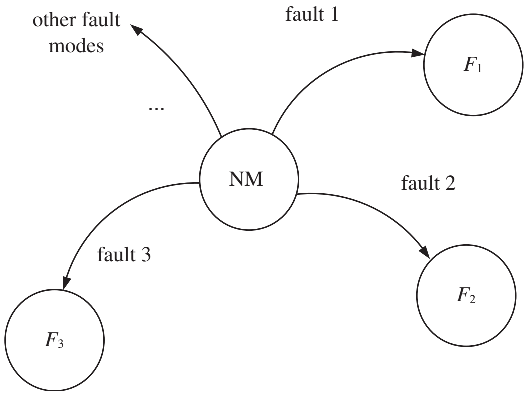


Figure 3. Conceptual scheme of system mode changing considering the effect of faults. NM denotes the nominal (non-faulty) system mode.

commutations of operational mode depending on internal system variables or external events (for example, faults).

Moreover, faults in control systems can affect any element of the control loop (sensors, actuators or the plant), changing either the structure or the parameters. According to Blanke *et al.* [1], dynamic models of the plant subject to the faults have to be considered when designing a FTC. The occurrence of the faults change the set of constraints of the control problem $\mathscr{C}(\theta)$, either by changing the parameters or the constraints themselves. These changes can be seen as new operating modes associated WITH the type of the fault occurred. In this way, when the system to be controlled is modeled, two kind of operating modes should be taken into account (see Figure 3):

- *Intrinsic operating modes*, related to changes in the system due to the evolution of its hybrid dynamics.
- *Faulty operating modes*, related to changes in the system due to the fault occurrence.

Once the hybrid model of the system considering the set of operating modes (intrinsic and faulty) has been stated, a control methodology that takes advantage of such information should be devised. Note that the total amount of intrinsic modes considered depends on the accuracy in the hybrid description of the system, i.e. it is possible to take into account not only all hybrid dynamics, but also increasing the complexity of the system representation. On the other hand, there will be as many faulty modes as faulty models considered (considering that only one fault can occur at each time instant). Hybrid control methodologies based on MPC that are able to handle naturally models including operating modes (intrinsic and faulty) are discussed in the following sections.

3. HMPC WITHIN THE MLD FRAMEWORK

3.1. MLD representation

The computational oriented hybrid systems approach considered in this paper is based on MLD form, introduced in [10]. MLD forms have been shown to be equivalent to other representations of hybrid systems such as LC systems, MMPS systems and PWA systems, among others, under mild conditions; see [9]. By considering hybrid dynamical systems in discrete-time, a number of mathematical problems (like Zeno behavior, see [16]) are avoided. Moreover, this allows to derive models for which tractable analysis and optimal/predictive control techniques exist [17–21].

In general form, a hybrid system in MLD form can be written as

$$x(k+1) = Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) \tag{1a}$$

$$y(k) = Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) \tag{1b}$$

$$E_2\delta(k) + E_3z(k) \leqslant E_1u(k) + E_4x(k) + E_5 \tag{1c}$$

where $x(k) \in \mathbb{R}^n$ is the state vector, $u(k) \in \mathbb{R}^m$ is the vector of manipulated variables (inputs) and $y(k) \in \mathbb{R}^p$ denotes the system outputs. The binary vector $\delta(k) = [\delta^1(k), \ldots, \delta^{r_l}(k)] \in \{0, 1\}^{r_l}$ of dimension r_l and the continuous-valued vector $z(k) \in \mathbb{R}^{r_c}$ are auxiliary variables associated with the MLD form. A specific value of the variable $\delta(k)$ is referred to as a *mode* of the hybrid system. A, B_i, C, D_i and E_i correspond to the system matrices of suitable dimensions. Equation (1c) collects the set of constraints on system variables as well as translations from logic propositions [10].

The transformation of certain hybrid system descriptions into the MLD form requires the application of a set of given rules. To avoid the tedious procedure of deriving the MLD form by hand, a compiler was developed in [12] to automatically generate the matrices A, B_i, C, D_i and E_i in (1) through the specification language hybrid system description language and associated compiler. This tool also allows to convert an MLD system (1) into an equivalent PWA system of the form [22]

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i \end{aligned} \quad \text{for} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \Omega_i \tag{2}$$

where

$$\Omega_i = \left\{ \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} : H_{ix}x(k) + H_{iu}u(k) \leqslant K_i \right\}, \quad i = 1, \ldots, s \tag{3}$$

are convex polyhedra in the input/state subspace. H_{ix}, H_{iu} and K_i are real matrices of appropriate dimensions and f_i and g_i are real vectors for all $i = 1, \ldots, s$, where s is the number of state space regions (modes). Equivalence means that for the same initial conditions and input sequences, the trajectories of the system are identical.

3.2. *Implicit HMPC control*

One of the most studied control techniques associated with hybrid systems involves optimal control and related variants such as MPC. The formulation of the optimization problem in MPC applied to hybrid systems (HMPC) follows the approach in standard linear MPC design; see [23]. The desired performance variables are expressed as functions of the control variables, initial states and known disturbances.

In general, the HMPC structure is defined by the following optimal control problem. Assume that the hybrid system output should track a reference signal y_r and x_r, u_r, z_r are desired references for the states, inputs and auxiliary variables, respectively. For a fixed prediction horizon H_p, the sequences

$$\begin{aligned}
\mathbf{x}_k &= \{x(1|k), x(2|k), \ldots, x(H_p|k)\} \\
\Delta_k &= \{\delta(0|k), \delta(1|k), \ldots, \delta(H_p-1|k)\} \\
\mathbf{z}_k &= \{z(0|k), z(1|k), \ldots, z(H_p-1|k)\}
\end{aligned} \tag{4}$$

are generated by applying the input sequence $\mathbf{u}_k = \{u(0), u(1), \ldots, u(H_p-1)\}$ to the system (1) from initial state $x(0|k) \triangleq x(k)$, where $x(k)$ is the measurement of the current state.

Hence, the HMPC optimal control problem is defined as

$$\begin{aligned}
\mathscr{O}: \min_{\mathbf{u}_k, \Delta_k, \mathbf{z}_k} \; & J(\mathbf{u}_k, \Delta_k, \mathbf{z}_k, x(k)) \triangleq \| Q_{xf}(x(H_p|k) - x_f)\|_p \\
& + \sum_{i=1}^{H_p-1} \|Q_x(x(i|k) - x_{\mathrm{r}})\|_p + \sum_{i=0}^{H_p-1} \|Q_u(u(i) - u_{\mathrm{r}})\|_p \\
& + \sum_{i=0}^{H_p-1} (\|Q_z(z(i|k) - z_{\mathrm{r}})\|_p + \|Q_y(y(i|k) - y_{\mathrm{r}})\|_p)
\end{aligned} \tag{5a}$$

$$\mathscr{C}: \text{s.t.} \begin{cases} x(0|k) = x(k) \\ x_f = x_{\mathrm{r}}(H_p|k) \\ x(i+1|k) = Ax(i|k) + B_1 u(i) + B_2 \delta(i|k) + B_3 z(i|k) \\ y(i|k) = Cx(i|k) + D_1 u(i) + D_2 \delta(i|k) + D_3 z(i|k) \\ E_2 \delta(i|k) + E_3 z(i|k) \leqslant E_1 u(i) + E_4 x(i|k) + E_5 \end{cases} \tag{5b}$$

for $i = 0, 1, \ldots, H_p - 1$, where $J(\cdot)$ is the cost function and x_f corresponds to the final desired value for the state variable over the prediction horizon H_p. p is related to the selected cost norm (i.e. 1-norm, Euclidean or infinity).

Assuming that the HMPC problem (5) is *feasible* for $x(k)$, there exists an optimal solution given by the sequence

$$\mathbf{u}_k^* = \{u^*(0), u^*(1), \ldots, u^*(H_{p-1})\}$$

Then, the *receding horizon control* philosophy sets [23, 24]

$$\mathscr{U}: u_{\mathrm{MPC}}(x(k)) \triangleq u^*(0) \tag{6}$$

and disregards the computed inputs $u^*(1)$ to $u^*(H_p - 1)$. The whole process is repeated at the following time step. Equation (6) is known in the MPC literature as *the MPC control law*. The equality constraint related to the final state within the HMPC problem (5) can be relaxed so that $x(H_p|k)$ is only required to belong to a terminal constraint set $\mathbb{X}_{\mathrm{T}}$ in order to ensure stability [25]. In particular, Lazar *et al.* [25] establish sufficient conditions to be satisfied by the terminal constraint set $\mathbb{X}_{\mathrm{T}}$ and cost function of the HMPC problem (5) to achieve asymptotic and exponential stability using Lyapunov methods and formulating the hybrid system in PWA form (see [9]). Moreover, this reference provides a computational procedure to determine the terminal set $\mathbb{X}_{\mathrm{T}}$ and cost function satisfying those conditions.

Thus, the control problem in case of HMPC (formulated as in (5)) is defined by a cost function J in (5a), which is given by the control objectives, and by a set of constraints in (5b). In [10], it is shown that due to the presence of logical variables, the resulting optimization problem associated with the one in (5) is a *mixed integer programming* (MIP) problem. Depending on the type of norm used in the objective function, the optimization associated with the design of the HMPC controller results in a *mixed integer quadratic* or *linear program* (MIQP or MILP, respectively) that can be efficiently solved to determine the optimal control sequences. The control law obtained in this way is referred to as *mixed integer predictive control* (MIPC).

For those cases where on-line optimization is not possible due to the high computation times with respect to real-time restrictions, an alternative approach is discussed in the following subsection.

3.3. *Explicit HMPC control*

The approach presented in [26] provides an explicit form of a linear MPC controller that performs off-line the computations for the implementation of MPC while preserves all its other characteristics. Such an explicit form of linear MPC is based on the solution of a quadratic programming (QP) problem, whose coefficients of the linear term in the cost function and the right-hand side of the constraints depend linearly on the current state. Then, the QP problem associated to a linear MPC can be viewed as a *multiparametric Quadratic Programming* (mp-QP) problem. The optimal solution is a PWA function of the vector of states (parameters). As a consequence, the solution is an explicit MPC law that is PWA with respect to the states. This result can be generalized to hybrid systems by transforming the MLD system given in (1) and in the HMPC optimization problem (5), into an equivalent PWA system [22]. In particular, in [27] it was shown a way to transform the HMPC control law (5) into the following explicit PWA form:

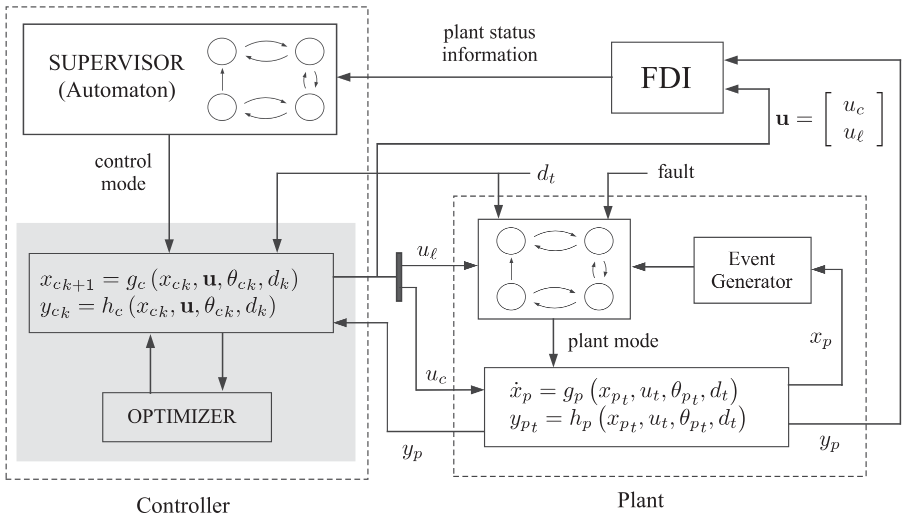
$$\mathscr{U}: u_{\text{MPC}}(x(k)) = \begin{cases} F_1 x(k) + g_1 & \text{if } x(k) \in \Omega_1 = \{x : H_1 x \leqslant S_1\} \\ & \vdots \\ F_s x(k) + g_s & \text{if } x(k) \in \Omega_s = \{x : H_s x \leqslant S_s\} \end{cases} \tag{7}$$

where $\bigcup_{i=1}^{s} \Omega_i$ is the set of states for which a feasible solution to (5) exists and s is the number of state space regions. There are several approaches to compute the explicit representation (7). For instance, this explicit solution can be computed using the Hybrid Toolbox for MATLAB® [28], which uses a combination of reachability analysis, multiparametric linear programming and computational geometry for comparison of convex PWA functions. The controller is returned as a collection of PWA maps of the form (7).

4. FAULT-TOLERANCE INCLUSION IN HMPC

4.1. *Implicit fault-tolerant HMPC*

According to Maciejowski [23] when using an MPC controller, if the knowledge of faults is available, either the internal model or system constraints can be modified accordingly. In this way, fault tolerance can be *implicitly* incorporated into an MPC controller in a natural way. Furthermore, due to the flexibility for expressing the control objectives within the MPC formalism, when faults



SUPERVISOR (Automaton)
plant status information
FDI
$\mathbf{u} = \begin{bmatrix} u_c \\ u_\ell \end{bmatrix}$
control mode
d_t
fault
u_ℓ
Event Generator
$x_{ck+1} = g_c(x_{ck}, \mathbf{u}, \theta_{ck}, d_k)$
$y_{ck} = h_c(x_{ck}, \mathbf{u}, \theta_{ck}, d_k)$
plant mode
x_p
u_c
$\dot{x}_p = g_p(x_{p_t}, u_t, \theta_{p_t}, d_t)$
$y_{p_t} = h_p(x_{p_t}, u_t, \theta_{p_t}, d_t)$
OPTIMIZER
y_p
y_p
Controller
Plant

Figure 4. Scheme of the AFTHMPC architecture.

cause control objectives to become unattainable, they can be dropped from the optimization problem or degraded in priority, for example, by changing hard constraints to soft ones [29].

However, looking at FTC in the hybrid framework and following the parallelism presented in Section 2, a new subset of 'faulty modes' could be added in the hybrid model used by the MPC controller additionally to the operating plant modes since each fault type induces different dynamics in the plant. These faulty modes will be reached by the plant after fault occurrence. The FDI module informs the HMPC controller about change of modes due to the fault. In this way, this change of mode in the hybrid model used by the HMPC controller will change either the internal model or system constraints taking into account the fault effect. Therefore, when faulty modes are included in the hybrid model used by the HMPC controller, the implementation of active fault tolerance is very straightforward. This is because in case of having knowledge about the presence of the fault due to the existence of an FDI module, a change of mode is induced in the associated hybrid model of the controller. Hence, and as a consequence, it allows the hybrid controller to adapt its operation mode in order to handle the faulty plant operating mode. This strategy is referred in the sequel as AFTHMPC. Figure 4 shows a conceptual scheme for this strategy.

In this paper, it is supposed that modules associated with fault-tolerant mechanisms will not be affected by faults since they are designed properly. However, in practice faults in these modules may occur, what adds more uncertainty to the whole fault-tolerant architecture. Moreover, the FDI module functionality is assumed to work correctly, in spite of the fact that it is a strong condition. That is, the fault is immediately detected and isolated after its apparition.[‡] This allows

[‡]This can be easily guaranteed in case of abrupt faults by the adequate design of the FDI module. In case of incipient faults, the fault detection and isolation can be delayed and even in some cases not detected. The case of undetected fault should be treated as uncertainty and the FTC problem should be addressed using a passive approach.

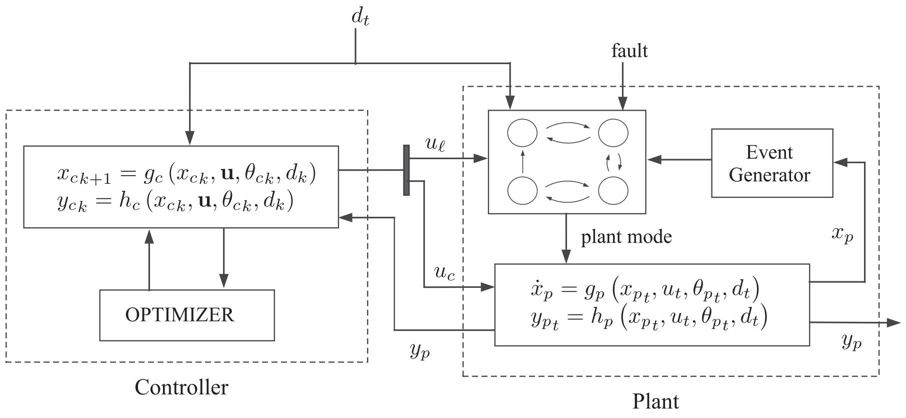

d_t
fault
u_ℓ
Event Generator
$x_{ck+1} = g_c(x_{ck}, \mathbf{u}, \theta_{ck}, d_k)$
$y_{ck} = h_c(x_{ck}, \mathbf{u}, \theta_{ck}, d_k)$
plant mode
x_p
u_c
OPTIMIZER
$\dot{x}_p = g_p(x_{p_t}, u_t, \theta_{p_t}, d_t)$
$y_{p_t} = h_p(x_{p_t}, u_t, \theta_{p_t}, d_t)$
y_p
y_p
Controller
Plant

Figure 5. Scheme of the PFTHMPC architecture.

the *reconfiguration* of the control loop.[§] If additionally the FDI module returns the fault magnitude, then *fault accommodation* would be possible.[¶] In the case of lack of knowledge about the presence of the fault, the hybrid controller should deal with a plant that has changed its operating mode due to the fault effect. In this case, fault tolerance relies on the inherent capabilities of the feedback control loop, what is referred as *Passive Fault-tolerant Hybrid Model Predictive Control* (PFTHMPC) (see Figure 5). The conceptual scheme for this case follows the scheme related to the AFTHMPC, but without considering the top modules regarding FDI and supervisor. In practice, it means that the HMPC controller should be designed to be robust against the uncertainty introduced by the fault.

In case that the hybrid plant model including the faults modes is represented using the MLD formalism, the MLD form introduced in (1) is modified as follows:

$$x(k+1) = Ax(k) + B_1u(k) + B_2\delta(k) + B_3z(k) + B_f f(k) \tag{8a}$$

$$y(k) = Cx(k) + D_1u(k) + D_2\delta(k) + D_3z(k) + D_f f(k) \tag{8b}$$

$$E_2\delta(k) + E_3z(k) \leqslant E_1u(k) + E_4x(k) + E_5 + E_f f(k) \tag{8c}$$

where f represents the faults being modeled as binary inputs in case that only information about existence of the fault is provided by the FDI module in real time. If additional knowledge about the fault is available as, for instance, its size, a more detailed fault model can be included in the MLD model given by (8). According to the case, matrices B_f, D_f, and E_f will contain on these positions values that reflect the fault effect. Therefore, the HMPC problem should be modified accordingly by replacing in (5) the MLD (1) by the MLD including faults in (8).

Trying to define all faulty modes taking into account the fault influence could be extremely difficult and sometimes an impossible mission. In practice, fault tolerance is only provided for a

[§] *System reconfiguration* after fault occurrence consists in finding a new set of constraints $\mathscr{C}_f(\theta_f)$ such that the control problem $\langle \mathscr{O}, \mathscr{C}_f(\theta_f), \mathscr{U}_f \rangle$ can be solved. Once found, the resulting control problem is solved and the reconfigured control law is applied [1].

[¶] *Fault accommodation* to the fault consists in solving the control problem $\langle \mathscr{O}, \hat{\mathscr{C}}_f(\hat{\theta}_f), \hat{\mathscr{U}}_f \rangle$, $\hat{\mathscr{C}}_f(\hat{\theta}_f)$ being an estimation of actual system constraints and parameters provided by the FDI module [1].

set of given faulty modes [1]. Then, this is the only set of faulty modes that should be included in the hybrid model of the plant. Moreover, this set of faults constitutes the specification for the FDI module, since this module should be able to detect and isolate each particular fault from this set.

For each of the faulty modes considered, a new MLD (or PWA) system results. This implies that the terminal constraint set $\mathbb{X}_{\mathrm{T}}$ and cost function to guarantee stability should be changed to those determined for instance by the computational method provided in [25]. The task of changing such conditions and the model constraints affected by the fault can be implemented in several ways as will be further discussed in Section 5.

4.2. *Explicit fault-tolerant HMPC*

Alternatively, fault tolerance of HMPC can be made explicit by using explicit HMPC methods presented in Section 3.3 by introducing faults as additional states

$$\tilde{x}(k)=\begin{bmatrix} x(k) \\ f(k) \end{bmatrix} \tag{9}$$

into the parametric programming algorithms. Then, the controller can be updated by means of the fault information given by the FDI module as follows:

$$u_{\mathrm{MPC}}(\tilde{x}(k))=\begin{cases} F_1\tilde{x}(k)+g_1 & \text{if } \tilde{x}(k)\in\Omega_1=\{\tilde{x}(k): H_1\tilde{x}(k)\leqslant S_1\} \\ & \vdots \\ F_{s+n_f}\tilde{x}(k)+g_{s+n_f} & \text{if } \tilde{x}(k)\in\Omega_{s+n_f}=\{\tilde{x}(k): H_{s+n_f}\tilde{x}(k)(k)\leqslant S_{s+n_f}\} \end{cases}$$

where $\bigcup_{i=1}^{s+n_f}\Omega_i$ is the set of states (s is the number of intrinsic modes) and faults (n_f is the number of faulty modes) for which a feasible solution of (5) exists.

In this way, the optimization problem (5) considering the MLD model including faults (8) does not need to be solved on-line, but instead it can be computed off-line.

Example 4.1

In order to show the fault-tolerant explicit MPC idea, faults affecting the operating range of the plant actuators will be considered. Let us consider discrete-time system given by

$$\begin{aligned} x_{k+1} &= 0.9512x_k+0.0975u_k \\ y_k &= 0.8x_k \end{aligned} \tag{10}$$

where $u_k\in[\underline{u},\overline{u}]=[-1,1]$, controlled using an MPC law with the following cost function:

$$J(x_k,u_k)=Px_{H_p}^2+\sum_{i=0}^{H_p-1}(Qx_i^2+Ru_i^2) \tag{11}$$

where $H_p=2$ for simplicity of presentation and the terminal weight matrix P is determined using the Ricatti equation with $Q=1$ and $R=0.1$.

In the case of faults that affect the actuator range, they would imply to change the control input constraints in the formulation of optimization problem (11) if the MPC law is computed implicitly. On the other hand, if the MPC law is computed explicitly, the range limits can be considered as parameters of the MPC control law by modifying the state equations in (10) as follows:

$$\begin{bmatrix} x(k+1) \\ f_1(k+1) \\ f_2(k+1) \end{bmatrix} = \begin{bmatrix} 0.9512 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ f_1(k) \\ f_2(k) \end{bmatrix} + \begin{bmatrix} 0.0975 \\ 0 \\ 0 \end{bmatrix} u(k) \tag{12}$$

$$\begin{bmatrix} y(k) \\ y_{f_1}(k) \\ y_{f_2}(k) \end{bmatrix} = \begin{bmatrix} 0.8 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ f_1(k) \\ f_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix} u(k) \tag{13}$$

where $f_1(k)$ *and* $f_2(k)$ correspond to the lower and upper bounds of the actuator range, respectively.

Introducing this fault parametrization and using results given in [26], the state-feedback explicit control law for the MPC controller, PWA with respect to the states and faults, can be derived using multiparametric quadratic programming (mpQP). Using this approach in the current example, the expression of u_{MPC} is given as a function of $\tilde{x}(k) = [x(k) \;\; f_1(k) \;\; f_2(k)]^{\mathrm{T}}$, which are the system state and the lower and upper limits of the actuator range, and corresponds to the explicit PWA control law given by

$$u_{\mathrm{MPC}}(\tilde{x}(k)) = \begin{cases} [-2.299 \;\; 0 \;\; 0]\tilde{x}(k) & \text{if } \begin{bmatrix} 1.6713 & 1 & 0 \\ -1.6713 & 0 & -1 \end{bmatrix} \tilde{x}(k) \leqslant \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ & \text{(Region\#1)} \\ [-2.656 \;\; 0 \;\; -0.2135]\tilde{x}(k) & \text{if } \begin{bmatrix} 0 & 1 & -1 \\ 1.272 & 0 & 0.7611 \end{bmatrix} \tilde{x}(k) \leqslant \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ & \text{(Region\#2)} \\ [-2.656 \;\; -0.2135 \;\; 0]\tilde{x}(k) & \text{if } \begin{bmatrix} 0 & 1 & -1 \\ -1.272 & -0.7611 & 0 \end{bmatrix} \tilde{x}(k) \leqslant \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ & \text{(Region\#3)} \end{cases} \tag{14}$$

which has been obtained using mp-programming tools included in the Hybrid Toolbox for MATLAB® [28] and represented graphically in Figure 6. This figure presents graphically how the optimal control gain varies with the system state and the considered faults (upper/lower limits of the actuation range). For a given actuation range, the gain only varies with the system state as can be seen from the figure and the corresponding control law can be determined from (14). Summarizing, if a fault in the actuator range appears, the explicit MPC controller (14) allows on-line adaptation using the fault estimation provided by the FDI module.

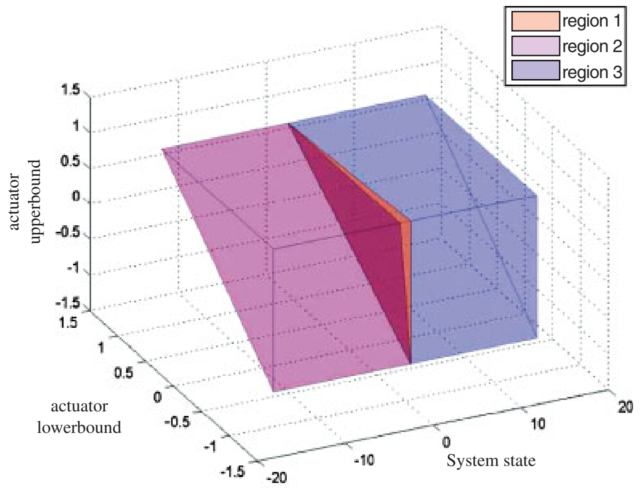


Figure 6. Polyhedra partitions of control law (14).

5. IMPLEMENTATION OF FTC USING HMPC UNDER THE MLD FRAMEWORK

In this section, once presented the implicit and explicit ways of including fault tolerance in HMPC, the different ways of implementing AFTHMPC using MLD formulation will be summarized and discussed. First of all, there exist two ways in which the controller can handle the faulty modes of the plant:

- Internally handled by including the faulty modes in the MLD model of the system in the same way as intrinsic modes. This generates additional binary variables corresponding to the faulty modes that should be initialized according to the fault information provided by the FDI module. In this case, the automata-based supervisor is embedded in the global MLD model of the system.
- Externally handled by generating a different MLD model corresponding to every possible faulty mode. Then, an external automata-based supervisor system will select which is the MLD model that should be used in each iteration.

Additionally, the HMPC controller for each MLD model can be computed (see Section 3):

- On-line (implicit) by solving the optimization problem associated with the HMPC controller at every iteration.
- Off-line (explicit) by transforming the MLD model into a PWA model and then computing a PWA controller.

Combining the different possible ways of faulty modes inclusion in the MLD with the possible ways of computing the HMPC controller for a given MLD model, four different possible implementations appear according to Figure 7. The option in Figure 7(a) is the most straightforward, since it just includes the faulty modes in the hybrid model of the system represented using the MLD formalism that will coexist with the inherent hybrid modes. The controller will commute on-line

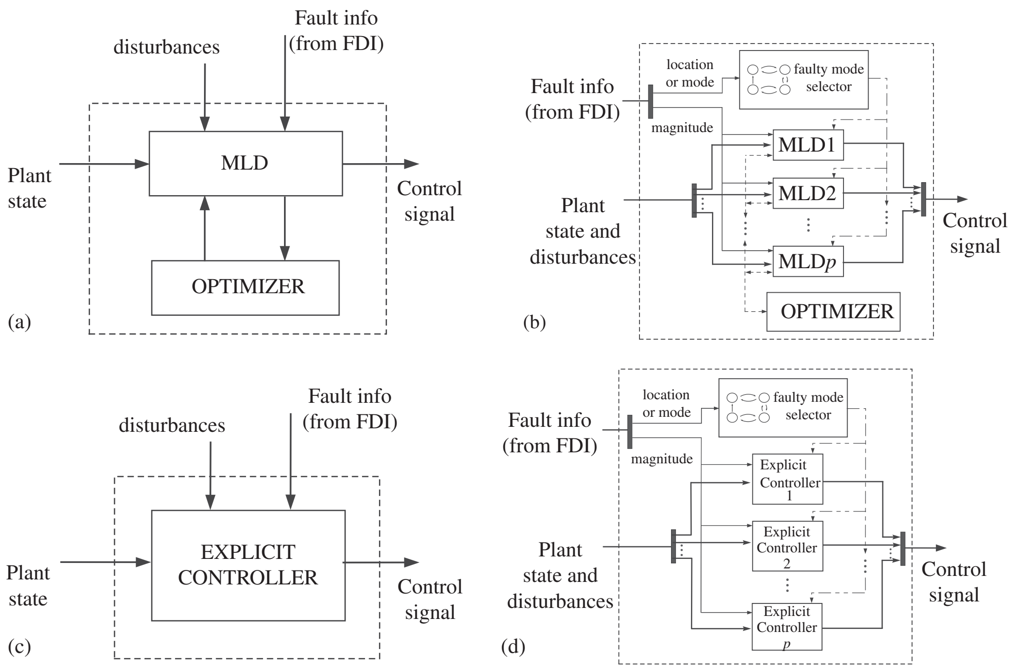


Figure 7. Implementation architectures for FTHMPC: (a) controller A; (b) controller B; (c) controller C; and (d) controller D.

among both modes depending on external events (faults) or internal event (changes of operating mode). The stability of the control loop can be guaranteed by setting properly some design conditions as in the traditional MPC design process (e.g. lengths of prediction horizon, terminal state constraints) [23]. In this architecture, the classical FTC supervisor module is embedded in the HMPC model. In Figure 7(b), only hybrid modes are included in the MLD used by the HMPC controller. This means that a bank of MLD models (one for each faulty mode) should be used and an external supervisor should change the MLD used by the controller depending on the external faulty event. Each MLD model is obtained considering the fault model within the nominal hybrid dynamics of the plant, resulting on an amount of MLD models equivalent to the number of fault models considered. Comparing both architectures, the first one provides a complete hybrid view of the FTC problem while the second one follows the classical three-level architecture. Both architectures use the implicit fault-tolerant capabilities of MPC described in Section 4. The benefits of the first option are that no external design of the supervisor should be done, but the price is the inclusion of additional binary variables to reflect the faulty modes.

On the other hand, considering the explicit computation of the HMPC controller, two additional architectures can be designed. In Figure 7(c), an explicit MPC law parameterized with respect to the system states and faults is proposed. The advantage of this architecture is that an off-line controller, which includes the reconfiguration logic, is used. The reconfiguration mechanism associated with the fault is derived in the same way that the mechanism related to the change of mode. The drawback of this approach is that the number of regions of the obtained PWA controller grows very fast with the amount of states and considered faults. A possible way to reduce such a complexity consists in generating an explicit MPC controller for each faulty mode (Figure 7(d)). This fact would imply the use of an external supervisor that switches between controllers, which are designed with the corresponding conditions in order to ensure the closed-loop stability [23, 25].

6. FAULT-TOLERANCE EVALUATION OF THE HMPC

Faults cause changes in the constraints of the HMPC control problem (5), which modify the set of feasible solutions. This fact, in turn, might cause the set of admissible solutions for a given control objective to be empty. Therefore, the admissibility evaluation of the control law facing faults implies checking if the feasible solutions set is empty or not for a given specified control objective. This problem has been already treated in the literature for the case of LQR problem without constraints [30], using the available analytical solution. However, constraints (on states and inputs) are always present in real industrial control problems and they could be easily handled using MPC [23]. In this case, an analytical solution for this type of control law does not exist, which makes difficult to do the analysis proposed in [30] for the LQR case. In [31], a method for admissibility evaluation for MPC using zonotope-based set computations has been proposed.

The aim of this section is to provide a method to evaluate the admissibility of HMPC control after a fault occurrence. This method can be used either off-line to analyze which fault configurations are or not fault tolerant with respect to a pre-established level of degradation in the control objectives, or on-line to analyze whether or not the control objectives can be achieved (may be, with a certain pre-established level of degradation) after the fault occurrence. In the case where the control objectives could not be achieved, they should be changed or adapted using some relaxation or prioritization mechanism [32].

The admissibility evaluation problem can be naturally handled as *Constraint Satisfaction Problem* (*CSP*) on sets, as discussed in [31].

6.1. Constraints satisfaction problem

A CSP on sets can be formulated as a 3-tuple $\mathscr{H}=(\mathscr{V},\mathscr{D},\mathscr{C})$ [33], where

- $\mathscr{V}=\{v_1,\dots,v_n\}$ is a finite set of variables;
- $\mathscr{D}=\{\mathscr{D}_1,\dots,\mathscr{D}_n\}$ is the set of their domains (where, for this paper, $\mathscr{D}_i\subseteq\mathbb{R}, i=1,\dots,n$);
- $\mathscr{C}=\{c_1,\dots,c_n\}$ is a finite set of constraints relating variables of $\mathscr{V}$.

A solution point of $\mathscr{H}$ is a n-tuple $(\tilde{v}_1,\dots,\tilde{v}_n)\in\mathscr{D}$ such that all constraints $\mathscr{C}$ are satisfied. The set of all solution points of $\mathscr{H}$ is denoted by $\mathscr{S}(\mathscr{H})$. This set is called the *global solution set*. The variable $v_i\in\mathscr{V}_i$ is *consistent* in $\mathscr{H}$ if and only if

$$\forall v_i\in\mathscr{V}_i \;\; \exists(\tilde{v}_1\in\mathscr{D}_1\dots,\tilde{v}_n\in\mathscr{D}_n)|(\tilde{v}_1,\dots,\tilde{v}_n)\in\mathscr{S}(\mathscr{H})$$

with $i=1,\dots,n$. The solution of a CSP is said to be *globally consistent* if and only if every variable is consistent considering the whole set of constraints. A variable is *locally consistent* if and only if it is consistent with respect to a group of constraints. Thus, the solution of a CSP is said to be locally consistent if all variables are locally consistent [34].

6.2. *Solving the CSP*

It is well known that the solution of CSP on sets has a high computational complexity since in order to represent the solution sets accurately they should be decomposed using subpavings, which implies exponential computation times [33].

However, in order to evaluate the fault tolerance of a HMPC controller after a fault occurrence, it is not required to find a point solution of the CSP problem, but whether the CSP problem has or not solution. In particular, the fact that a CSP does not have solution means that the HMPC controller is not fault tolerant with respect to the considered fault. Otherwise, it would be fault tolerant.

The existence of the solution of the CSP problem associated WITH the tolerance evaluation of HMPC controller for a given faulty situation can be proved by solving the following optimization problem:

$$\mathcal{O}:\ \min g(\cdot) \tag{15a}$$

$$\mathcal{C}:\ \text{s.t.}\begin{cases} J(\mathbf{u}_k,\Delta_k,\mathbf{z}_k,x(k))\leqslant J_f\\ x(0|k)=x(k)\\ x_f=x_{\mathrm{r}}(H_p|k)\\ x(i+1|k)=Ax(i|k)+B_1u(i)+B_2\delta(i|k)+B_3z(i|k)\\ y(i|k)=Cx(i|k)+D_1u(i)+D_2\delta(i|k)+D_3z(i|k)\\ E_2\delta(i|k)+E_3z(i|k)\leqslant E_1u(i)+E_4x(i|k)+E_5\\ \quad\text{for } i=0,1,\ldots,H_p-1\end{cases} \tag{15b}$$

with

$$\begin{aligned} J(\mathbf{u}_k,\Delta_k,\mathbf{z}_k,x(k)) = {} & \|Q_{xf}(x(H_p|k)-x_f)\|_p\\ & + \sum_{i=1}^{H_p-1}\|Q_x(x(i|k)-x_{\mathrm{r}})\|_p+\sum_{i=0}^{H_p-1}\|Q_u(u(i)-u_{\mathrm{r}})\|_p\\ & + \sum_{i=0}^{H_p-1}(\|Q_z(z(i|k)-z_{\mathrm{r}})\|_p+\|Q_y(y(i|k)-y_{\mathrm{r}})\|_p) \end{aligned} \tag{16}$$

and $g(\cdot)$ denotes the null function since it is only needed to know whether or not the problem constraints are violated. In the case that this problem was feasible, the CSP problem has solution and the HMPC controller is fault tolerant with respect to the control objective (16) with a control objective degradation less or equal to J_f, which establishes the admissibility threshold.

7. APPLICATION TO SEWER NETWORKS

7.1. *Application description*

To exemplify the usefulness of the FTC design methodology proposed in this paper, a portion of the sewer network of the city of Barcelona is used.

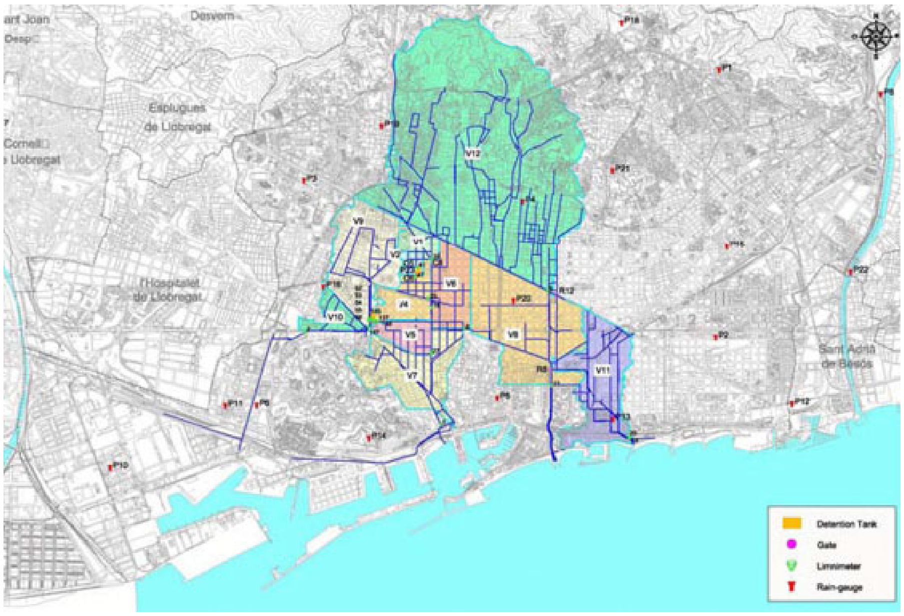


Figure 8. Test Catchment located over the Barcelona map. Courtesy of CLABSA.

7.1.1. The Barcelona sewer network. The city of Barcelona has a combined sewage system of approximately 1697 km length with a storage capacity of 3 038 622 m^3. It is a unitary system, that is, it combines waste and rainwater into the same sewers. It is worth to notice that Barcelona has a population which is around 1 593 000 inhabitants on a surface of 98 km^2 approximately. This results in a very high density of population. Additionally, the yearly rainfall is not very high (600 mm/year), but it includes heavy storms (arriving to 90 mm/h) typical of the Mediterranean climate that can cause a lot of flooding problems and combined sewer overflow (CSO) to the receiving environments.

There exists a remote control system operated by CLABSA[‖] since 1994, which includes sensors, regulators, remote stations, communications and a Control Center. Nowadays, for control purposes, the urban drainage system contains 21 pumping stations, 36 gates, 10 valves and 8 detention tanks which are regulated in order to prevent flooding and CSO to the environment. The remote control system is equipped with 56 remote stations including 23 rain-gauges and 136 water-level sensors which provide real-time information about rainfall and water levels into the sewer system. All this information is centralized at the Control Center using a global supervisory control system and data acquisition system. The regulated elements (pumps, gates and detention tanks) are currently controlled locally, i.e. they are handled from the remote control center according to the measurements of sensors connected only to the local station.

In this paper, a representative portion of the Barcelona sewer network is studied (see Figure 8) since it contains the representative components and the main characteristics of the whole sewer

[‖]*Clavegueram de Barcelona, S.A.* (CLABSA) is the company that manages the sewer network of Barcelona.

network. The catchment has a surface of $22.6\,\text{km}^2$ and is constituted by 11 sub-catchments (virtual tanks**), several level gauges (limnimeters), 5 flow links and 2 treatment plants. It has 1 retention gate associated with one real tank and three redirection gates. Also, there are five rain-gauges in the catchment but some virtual tanks share the same rain sensor.

The optimal global control of a sewer network in real time aims to minimize flooding and CSO to the environment, and to maximize the wastewater treatment plants utilization. This is an important challenge due to the large scale of the network. In addition, adverse meteorological conditions involve a high probability of errors in sensor measurements and/or malfunction in actuators. Owing to these restrictions, the control algorithm must implement an easy operational model in order to calculate its optimal configuration to meet optimization requirements and fault-tolerant strategies in order to accommodate faults and to avoid stopping the control application. In this paper, only the problem of including fault tolerance against fault in actuators is considered.

7.1.2. The sewer network control model. The modeling methodology used in this paper is based on the approach presented in [35]. There, the sewer system was divided into connected subgroups of catchments and treated as interconnected virtual tanks. The volume is calculated through the mass balance of the ith stored volume v_i, taking into account the area rainfall and flow exchanges between tanks. For each tank (catchment), the equation of the mass balance can be written as follows:

$$v_i(k+1) = v_i(k) + \varphi S P_i(k) + \Delta t (q_i^{\text{in}}(k) - q_i^{\text{out}}(k)) \tag{17}$$

where φ is the ground absorption coefficient of the ith catchment, S_i is the area of the ith catchment, P is the precipitation intensity in Δt of the ith catchment and Δt is the time interval between measurements. $q_i^{\text{in}}(k)$ and $q_i^{\text{out}}(k)$ are the sum of inflows and outflows, respectively. *Real detention tanks* are modeled in the same way but without the precipitation term.

The tanks are connected with flow paths or links that represent the main sewage pipes between the catchments. The manipulated variables of the system, denoted as q_{u_i}, are related to the outflows from the tanks. The outflows are assumed to be proportional to the tank level, that is,

$$q_i^{\text{out}}(k) = \beta_i v_i(k) \tag{18}$$

as suggested in [36], where additional procedures for the calibration of φ and β (proportionality factor) are given.

The presence of intense precipitation can cause some sewers to reach their limits of capacity. When this happens, excess of sewage that would normally have been collected in the sewer might flow to other parts of the network. In this way, flow paths appear that are not always present and depend on the system state and inputs. Using the virtual tanks model methodology within the hybrid systems framework, the portion of the sewer network under study is shown in Figure 9, where the dashed lines represent the overflow from virtual tanks and sewers. This behavior and other particular hybrid phenomena/elements of sewer systems that depend on system state can be conveniently modeled using the MLD form introduced in Section 3.1. A calibrated and validated MLD model of this system, including all intrinsic operating modes, is available as well as rain

*******Virtual tank* is a concept related to the volume of sewage that, at any given time, is stored inside the mains associated with a determined subcatchment of the sewer network [35].

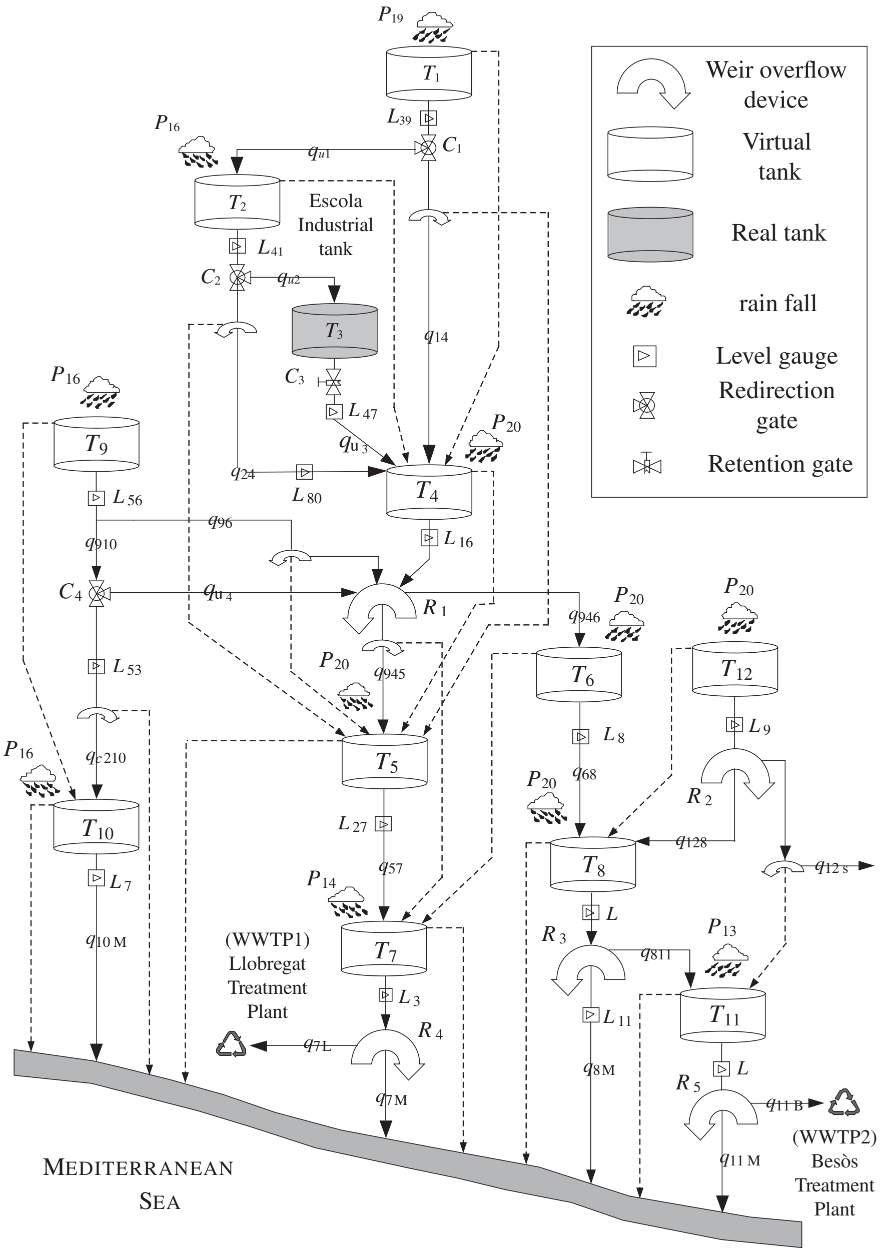


Figure 9. Barcelona test catchment scheme for hybrid design.

gauge data for an interval of several years; see [37]. The whole case study expressed in MLD form can be written as follows:

$$v(k+1) = Av(k) + B_1 q_u(k) + B_2\delta(k) + B_3 z(k) + B_4 d(k) \tag{19a}$$

$$y(k) = Cv(k) + D_1 q_u(k) + D_2\delta(k) + D_3 z(k) + D_4 d(k) \tag{19b}$$

$$E_2\delta(k) + E_3 z(k) \leqslant E_1 q_u(k) + E_4 v(k) + E_5 + E_6 d(k) \tag{19c}$$

where $v \in \mathbb{R}_+^n$ corresponds to the vector of tank volumes (states), $q_u \in \mathbb{R}_+^m$ is the vector of manipulated sewer flows (inputs), $d \in \mathbb{R}_+^{md}$ is the vector of rain measurements (disturbances), logic vector $\delta \in \{0, 1\}^{r\ell}$ collects the discontinuous overflow conditions and vector $z \in \mathbb{R}_+^{rc}$ is associated with variables that appear depending on system states and inputs.[††] Notice that this model is a more general MLD than was presented in [10] due to the addition of the measured disturbances.

If it is assumed that the disturbances are described with a disturbance model

$$d(k+1) = A_d d(k)$$

the MLD form (19) could be rewritten as

$$\begin{bmatrix} v(k+1) \\ d(k+1) \end{bmatrix} = \begin{bmatrix} A & B_4 \\ 0 & A_d \end{bmatrix} \begin{bmatrix} v(k) \\ d(k) \end{bmatrix} + \begin{bmatrix} B_1 & B_2 & B_3 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} q_u(k) \\ \delta(k) \\ z(k) \end{bmatrix} \tag{20a}$$

$$y(k) = [C \;\; 0] \begin{bmatrix} v(k) \\ d(k) \end{bmatrix} + [D_1 \;\; D_2 \;\; D_3] \begin{bmatrix} q_u(k) \\ \delta(k) \\ z(k) \end{bmatrix} \tag{20b}$$

$$[E_2 \;\; E_3] \begin{bmatrix} \delta(k) \\ z(k) \end{bmatrix} \leqslant [E_4 \;\; E_6] \begin{bmatrix} v(k) \\ d(k) \end{bmatrix} + [E_1 \;\; E_5] \begin{bmatrix} q_u(k) \\ 1 \end{bmatrix} \tag{20c}$$

The type of disturbance model to be used in sewage system control is, in general, an open research topic. Different types of rain prediction can be considered since this procedure can be developed either using theoretical tools (statistical, AR models, etc.) [38] or practical tools (radars, meteorological satellites, etc.) [39]. According to [40], different assumptions can be done for the rain prediction when an optimal control law is used in the RTC of sewer networks. Results show that the assumption of constant rain over a short prediction horizon gives results that can be compared with the assumption of known rain over the considered horizon, confirming similar results reported in [35, 41]. In the latter case, matrix A_d can be set as an identity matrix of suitable dimensions.

7.2. *HMPC problem formulation*

As discussed in previous sections, the proposed hybrid modeling methodology allows the straightforward treatment of hybrid phenomena such as overflow and flooding for the particular application.

[††]Notation $\mathbb{R}_+$ denotes the set of non-negative real numbers.

Moreover, HMPC has been successfully applied to a variety of control problems during the past few years using several approaches, see [17–21], among others. This section presents a description of the HMPC formulation applied to sewer networks. The different aspects discussed here are presented considering the particular case study but can be easily extrapolated to other sewage system topologies.

7.2.1. Control objectives. The sewer system control problem has multiple objectives with varying priority, see [42]. In general, the most common objectives are related to the manipulation of the sewage in order to avoid undesired sewage overflow on the streets of a city. Another type of objectives are, for instance, related to the control energy, i.e. the energy cost of the regulation of the gate movements [43]. According to the Barcelona sewer network managers, the main objectives for the case study of this paper are listed below in order of decreasing priority:

- *Objective 1*: minimize flooding in streets (virtual tank overflow).
- *Objective 2*: minimize flooding in links between virtual tanks.
- *Objective 3*: maximize sewage treatment avoiding CSO in the environment.
- *Objective 4*: minimize control action.

A secondary purpose of the third objective is to reduce the volume in the tanks to anticipate future rainstorms. This objective indirectly reduces pollution to the environment. This is because if the treatment plants are used optimally along with the storage capacity of the network, the amount of pollution released to the environment should be at a minimum. It should be noted that in practice the difference between the first two objectives is small.

7.2.2. Problem constraints. The modeling approach is based on mass conservation. Additionally, physical restrictions are included as constraints in the optimization problem. The sum of inflows into nodes that connect links have to be equal the outflow. The control variables are limited to a range given by the maximum capacities of the related links. The constraints associated with the HMPC problem are in general the constraints associated with the hybrid behavior as well as the system physical constraints for manipulated links and real tanks and the initial condition corresponding to the measurements of the tanks volumes at time instant $k \in \mathbb{Z}_+$.

All the constraints can be expressed in the form given by (19c). The physical constraints are considered as *hard constraints* into the control problem. On the other hand, the overflows in sewers and virtual tanks are considered as *soft constraints* and a constraint manager could be designed and implemented to solve the control problem with prioritization of constraints [29].

7.2.3. The cost function. Each control objective corresponds to one term in the cost function. Hence, the expression of that function depends on its constitutive variables (auxiliary or output type). In general form, the structure for the cost function in (5a) has the form

$$J(\mathbf{u}_k, v(k)) \triangleq \sum_{i=0}^{H_p-1} \|Q_z(z(k+i|k)-z_\mathrm{r})\|_p + \sum_{i=0}^{H_p-1} \|Q_y(y(k+i|k)-y_\mathrm{r})\|_p \tag{21}$$

where Q_z and Q_y correspond to weight matrices of suitable dimensions, H_p denotes the prediction horizon and z_r, y_r are reference trajectories related to auxiliary and output variables, respectively, as stated in (5). For the objectives 1–2, the references are zero flow. For the third objective, the references are the maximum capacity of the associated sewage treatment plants. Priorities are set by selecting matrices Q_z and Q_y. The norm p can be selected as $p=1,2$ or $p=\infty$. Notice that due to the fact that all performance variables are positive, the case when $p=1$ is actually a simple sum of the performance variables.

7.2.4. MIPC problem. Taking into account all issues mentioned so far, the predictive control problem for a sewer network considering its hybrid model is defined as follows:

$$\min_{\mathbf{q_u}(k),\Delta(k),\mathbf{z}(k)} J(\mathbf{q_u}(k), \Delta(k), \mathbf{z}(k), v(k)) \tag{22a}$$

$$\text{s.t.} \quad \begin{cases} v(i+1|k) = A\,v(i|k) + B_1\,q_u(i) + B_2\,\delta(i|k) + B_3\,z(i|k) + B_4\,d(i|k) \\ y(i|k) = C\,v(i|k) + D_1\,q_u(i|k) + D_2\,\delta(i|k) + D_3\,z(i|k) + D_4\,d(i|k) \\ E_2\,\delta(i|k) + E_3\,z(i|k) \leqslant E_1\,q_u(i|k) + E_4\,v(i|k) + E_5 + E_6\,d(i|k) \\ d(i+1|k) = A_d\,d(i|k) \end{cases} \tag{22b}$$

for $i=0,\ldots,H_p-1$. Assuming that the problem is feasible for $v\in\mathbb{R}^n$, the receding horizon philosophy is then used considering as the MPC law

$$q_{u\,\mathrm{MPC}}(v(k)) \triangleq q_u{}^*(0|k) \tag{23}$$

and the entire optimization process is repeated for time $k+1$.

7.2.5. Control results for non-faulty scenarios. To show the performance of HMPC, some real episodes of rain storms are used [44]. The performance of the control scheme is compared with the simulation of the sewer system without control (open loop) when the manipulated links have been used as passive elements, i.e. the amount of flows $q_{u1}(k), q_{u2}(k)$ and $q_{u4}(k)$ only depend on the inflow to the corresponding gate that are not manipulated, while $q_{u3}(k)$ is the natural outflow of the real tank.

Table II summarizes the results for ten of the more representative rain episodes in Barcelona between 1998 and 2002. These episodes were selected to represent the meteorological behavior of Barcelona, i.e. they contain representative meteorologic phenomena in the city. The duration of the simulations scenarios was selected to have 8 h approximately ($k\in[0,100]$) as the rain storm generally had peaks of duration around 10 samples or 50 min. The cost function in (21) has been used with quadratic norm and values for the weight matrices Q_z and Q_y such that the prioritization of the control objectives according to Section 7.2 can be achieved. The prediction horizon H_p and control horizon H_u were selected as six samples or 30 min (with a sampling time $\Delta t=300$ s), which corresponds to the *time of concentration*[‡‡] for the Barcelona sewer network. Another reason for the selection of these prediction and control horizon values is that prediction provided by the used sewer network model becomes less reliable for larger time horizons. Results were obtained using the Hybrid Toolbox for MATLAB® [28] and ILOG CPLEX 9.1 as the MIP solver [45].

According to this table, the main control objectives are achieved in the most of the cases even though the secondary objectives (lower priority lever within the considered objectives) are sometimes degraded. Owing to the topology of the network and its physical design, the performance

[‡‡]The time of concentration of a sewer network is determined as the time required for water to travel from the most remote catchment to its outlet to the environment.

Table II. Obtained results of closed-loop performance using 10 representative rain episodes.

		Open loop			Closed loop	
Rain episodes	Flooding $\times 10^3$ (m^3)	pollution $\times 10^3$ (m^3)	Treated W. $\times 10^3$ (m^3)	Flooding $\times 10^3$ (m^3)	pollution $\times 10^3$ (m^3)	Treated W. $\times 10^3$ (m^3)
14-09-1999	108	225.8	278.4	92.9 (14%)	223.5	280.7
09-10-2002	116.1	409.8	533.8	97.1 (16%)	398.8	544.9
03-09-1999	1	42.3	234.3	0 (100%)	44.3	232.3
31-07-2002	160.3	378	324.4	139.7 (13%)	374.6	327.8
17-10-1999	0	65.1	288.4	0	58.1 (11%)	295.3
28-09-2000	1	104.5	285.3	1	98 (6%)	291.9
25-09-1998	0	4.8	399.3	0	4.8	398.8
22-09-2001	0	25.5	192.3	0	25	192.4
01-08-2002	0	1.2	285.8	0	1.2	285.8
20-04-2001	0	35.4	239.5	0	32.3 (9%)	242.5

improvement due to the use of a control strategy is not always achieved and depends on the magnitude of the rain episodes. For instance, in the case of light episodes, the performance indexes related to the main control objectives of the network are barely modified. However, the system performance is in general improved when the hybrid control strategy proposed in this paper is applied (see percentages).

Remark 1
Simulations related to the results summarized in Table II showed the improvement of the system performance when a HMPC controller is used. However, for some rain episodes the obtained computation times were too high with respect to the sampling time of the system. This fact makes very difficult or even impossible the on-line implementation of the proposed approach.

This has motivated the development of a suboptimal control strategy presented in [46], which considerably reduces the computation time until respecting real-time restrictions. This HMPC strategy limits on-line the number of feasible nodes in the MIP problem. This is done by adding constraints to the MIP based on insight into the system dynamics. The idea consists in helping the MIP solver by adding cuts in the search space. In this way, the main source of complexity, namely the combinatorial explosion related to the binary search tree, is reduced at the expense of a suboptimal solution.

7.3. Hybrid modeling of the faulty modes

In this paper, faults in the sewer system actuators (control gates) are considered. There may exist many types of fault modes that can appear in the control gates within a sewer network. Here, the considered faulty modes consist in limiting the range of the gates in three ways:

1. Limit range from below (range is 50–100%), denoted as $f\underline{q}_{u_i}$.
2. Limit range from above (range is 0–50%), denoted as $f\overline{q}_{u_i}$.
3. Limit from below and above, simulating stuck gate (50–51%) and denoted as $f\underline{\overline{q}}_{u_i}$.

To model such faulty modes, the hybrid systems modeling methodology to take into account faulty modes proposed in this paper is used. It is assumed that only single actuator faults are present in the system. However, two or more simultaneous faults can easily be considered. For instance,

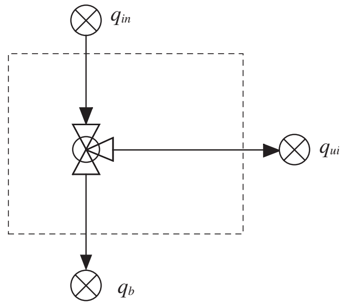


Figure 10. Control gate scheme used to explain the fault hybrid modeling.

let us consider the fault actuator mode $f\underline{q}_{u_i}$ for a redirection gate in Figure 10 (with 1 input q_{in} and 2 outputs q_{u_i} and q_b). Let us show how such mode can be expressed using the proposed hybrid modeling approach.[§§] This fault limits explicitly the range of the manipulated outflow q_{u_i}. Using the principle of mass conservation, this limitation can be expressed as follows:

$$q_{u_i} \leqslant \min(\overline{q}_{u_i}, q_{in}) \tag{24a}$$

$$q_{u_i} \geqslant \min(\underline{q}_{u_i}, q_{in}) \tag{24b}$$

It can be noticed that the new restrictions derive in non-convex constraints on the optimization problem, what implies that QP programming algorithms could not be used. This fact further motivates the use of the proposed hybrid modeling methodology. Inequality (24a) (related to the upper bound) can be expressed with two linear inequalities as $q_{u_i} \leqslant \overline{q}_{u_i}$ and $q_{u_i} \leqslant q_{in}$. Notice that in the case of fault scenarios $f\overline{q}_{u_i}$ and $f\underline{q}_{u_i}$, the fault affects the system when $q_{in} > f\overline{q}_{u_i}$. Otherwise, the fault does not have any influence over the behavior of the network. The non-convex constraint (24b) can be easily treated in the hybrid framework by introducing auxiliary variables

$$[\delta_i = 1] \rightarrow [q_{u_i} \leqslant q_{in}] \tag{25a}$$

$$z_{q_{u_i}} = \begin{cases} \underline{q}_{u_i} & \text{if } \delta_i = 1 \\ q_{in} & \text{otherwise} \end{cases} \tag{25b}$$

and replacing (24b) by $q_{u_i} \geqslant z_{q_{u_i}}$.

7.4. FTC results

This section presents the results when the AFTHMPC scheme presented in Section 5 is used. To see the benefits of using such FTC scheme, for comparison purposes results obtained without the FTC scheme are also presented (those results follow the PFTHMPC scheme). Results were obtained simulating the closed-loop system for all fault scenarios presented in Section 7.3 and for the set of rain episodes used in Section 7.2. To compare AFTHMPC and PFTHMPC cases, total flooding, pollution and treated water released over the whole scenario are used. The cost function

[§§]Similarly, the rest of faulty modes described in Section 7.3 can be modeled using this approach.

Table III. Results for FTHMPC with rain episode occurred on October 17, 1999.

Fault scenario		PFTHMPC			AFTHMPC		
Actuator	Type	Flooding $\times 10^3$ (m^3)	Pollution $\times 10^3$ (m^3)	Treated W. $\times 10^3$ (m^3)	Flooding $\times 10^3$ (m^3)	Pollution $\times 10^3$ (m^3)	Treated W. $\times 10^3$ (m^3)
q_{u_2}	$f\underline{q}_{u_2}$	0.0	61.9	291.4	0.0	61.9	291.4
	$f\overline{q}_{u_2}$	15.9	62.8	290.5	10.4	63.3	290.0
	$f\overline{\underline{q}}_{u_2}$	14.8	63.3	290.0	9.6	63.8	289.5
q_{u_3}	$f\underline{q}_{u_3}$	0.0	59.1	294.2	0.0	58.8	294.5
	$f\overline{q}_{u_3}$	3.5	57.7	295.6	0.2	58.7	294.7
	$f\overline{\underline{q}}_{u_3}$	0.8	58.8	294.6	0.0	58.9	294.4

structure, norm and control tuning used were the same as in the simulations performed for nominal HMPC presented in Section 7.2.

Generally, flooding in streets was reduced when AFTMPC was used compared with PFTMPC. The biggest improvements were obtained when precipitation was large enough so that actuators needed to operate close to the upper limit of their range, that is, when the precipitation brought the sewer network close to its maximum capacity. Even though results are shown for specific rain episodes, the conclusions presented were based on simulation of several representative scenarios.

Results are shown in Table III for a rain storm that occurred on 17 October, 1999. This rain episode has a 0.7-year return period with regard to total amount and 10-year return period with regard to maximum intensity. The particular feature of this episode lies on its behavior during the time window considered. As seen in Figure 11, this rain presents a double peak of intensity, what yields that the sewer network behavior is more complex and the nominal HMPC and the FTHMPC designs have a lot of work trying to control the system and avoiding the fault influence.

In this case, the most representative flooding reduction occurred in the fault scenario $f\overline{q}_{u_2}$, with about 35% of improvement caused by the use of the AFTHMPC strategy with respect to the PFTHMPC. This improvement is reached by means of a set of procedures caused by the computed control signals. Figure 12 shows this set of actions after the second rain peak for different parts of the test catchment for both active and passive approaches. In the AFTHMPC case and due to the manipulated flow q_{u_2} has lost capacity, the controller cannot take advantage of the real tank T_3 in a short/medium term (keep in mind that real tanks—reservoirs—are generally used as a buffer within the network; using this capability, they can store enough water to avoid flooding and/or CSO downstream).

This fact induces that sewage coming from T_1 is conveniently derived through sewer q_{14} (see Figure 12, top graph), what produces that sewers close to T_3 do not have as much overflow as in the case of applying the passive strategy (see Figure 12, medium graph). The slow filling of T_3 plus its convenient outflow manipulation (control signal related to q_{u_3}, see bottom graph in Figure 12) make that a bit of buffer capability benefits the overflow avoidance in T_5. All these actions produce the mentioned improvement of flooding reduction for this fault scenario in this rain episode.

Finally, let us consider a medium intensity rain episode that occurred on September 3, 1999. This episode is well supported by the network topology design, i.e. implementing an adequate

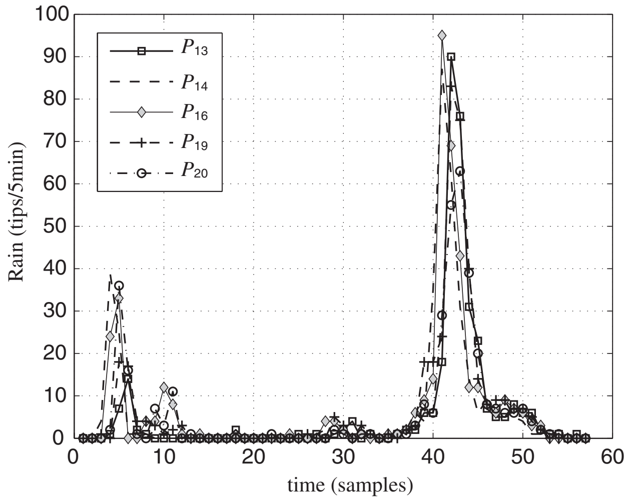


Figure 11. Rain episode occurred on 17 October, 1999 in Barcelona. Each curve represents a rain gauge P_i.

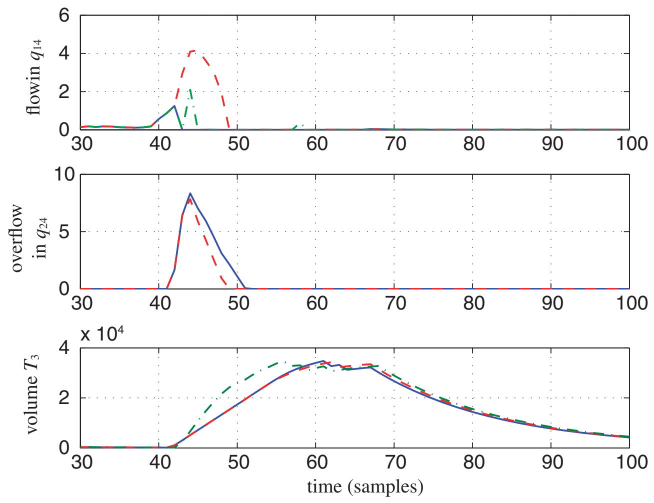


Figure 12. Set of actions in different parts of the test catchment for both active and passive FTHMPC approaches for fault scenario $f\overline{q}_{u_2}$. Solid curve (–): PFTHMPC; dashed curve (– –): AFTHMPC; dotted-dashed curve (– · –): no-fault HMPC (rain episode: 17-10-1999).

control law as the one proposed in this paper and depending on the fault scenario considered, the sewer network would not have flooding. Table IV collects the obtained results. The AFTHMPC yielded improvements with respect to the PFTHMPC strategy. In the scenarios $f\overline{q}_{u_3}$ and $f\underline{q}_{u_3}$, the FTC strategy achieved around 100% of flooding reduction. The reason of this improvement is

Table IV. Results for FTHMPC with rain episode occurred on 3 September, 1999.

Fault Scenario		PFTHMPC			AFTHMPC		
Actuator	Type	Flooding $\times 10^3$ (m^3)	Pollution $\times 10^3$ (m^3)	Treated W. $\times 10^3$ (m^3)	Flooding $\times 10^3$ (m^3)	Pollution $\times 10^3$ (m^3)	Treated W. $\times 10^3$ (m^3)
q_{u_2}	$f\underline{q}_{u_2}$	0.0	44.3	232.3	0.0	44.3	232.3
	$f\overline{q}_{u_2}$	15.2	44.5	232.1	12.2	44.7	231.9
	$f\overline{\underline{q}}_{u_2}$	14.7	44.3	232.3	11.8	44.4	232.2
q_{u_3}	$f\underline{q}_{u_3}$	0.0	45.2	231.4	0.0	45.2	231.4
	$f\overline{q}_{u_3}$	4.1	44.1	232.5	0.2	44.3	232.3
	$f\overline{\underline{q}}_{u_3}$	1.5	44.3	232.3	0.0	44.3	232.3

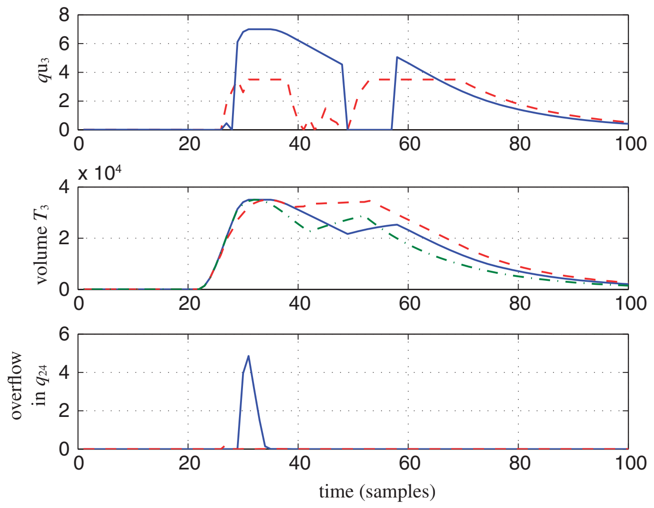


Figure 13. Network behavior for fault scenario $f\overline{q}_{u_3}$. PFTHMPC (—), AFTHMPC(− −) and non-faulty HMPC (−·−) (rain episode: 03-09-1999).

because the AFTHMPC takes advantage of the sewage accumulation in the real tank imposed by the emptying restriction due to the fault (see medium graph in Figure 13) and computes adequately the set of control signals in order to redirect the sewage avoiding big quantities around the faulty elements within the sewer network.

When the PFTHMPC strategy is used, the controller computes a control signal $q_{u_3}(k)$ without knowing the fault in the actuator, which implies that the computed control signal and the applied signal related to the control action will be different (notice that the applied signal in this case corresponds to the computed signal but saturated according to the faulty upper limit). Hence, since the physical constraints impose a limit on the real tank inflow (manipulated link q_{u_2}) in function of its current volume, sewage that enters in C_2 is derived through sewer q_{24} causing overflow in this element and then the increasing of the amount of flooding.

On the other hand, the AFTHMPC strategy computes the control signal $q_{u_1}(k)$ in such a way that the sewage from T_1 is redirected through q_{14} and then less water goes toward the real tank and its faulty output actuator. Thus, despite the slow emptying of T_3, the C_2 water inflow is conveniently distributed between sewers q_{u_2} and q_{24}, avoiding the overflow in this latter sewer and therefore preventing the flooding increase. Figure 13 shows the obtained signals related to the computed control signal $q_{u_3}(k)$ (top graph), the volume in T_3 (medium graph) and overflow in q_{24} (bottom graph) using both active and passive FTC strategies.

Summarizing, the fault-tolerant capabilities added to the HMPC closed-loop makes the system behavior to get better despite the fault occurrence.

8. CONCLUSIONS

In this paper, the FTC problem was formulated in the hybrid systems framework. In particular, the MLD form to represent hybrid systems was considered. Using this approach, an hybrid model of the system to be controlled was obtained, which includes inherent hybrid phenomena and possible modes caused by faults occurrence. This has allowed to adapt on-line the system model taking into account the fault information provided by the FDI module. In this way, the controller, based in this paper on MPC, could cope with faults. Additionally, different implementations of fault-tolerant HMPC were proposed and discussed. Despite the complexity of the problem from the point of view of the on-line implementation, the use of a suboptimal approach can be considered in order to apply the proposed strategy to complex large-scale systems. Finally, to show the usefulness of the proposed approach, it was applied to include fault tolerance in the design of a HMPC controller for a portion of the sewer network of Barcelona.

ACKNOWLEDGEMENTS

This work has been supported by the CICYT Ref. DPI2006-11944 of the Spanish Science and Technology Ministry, the *Juan de la Cierva* Research Program (Ref. JCI-2008-2438), and the DGR of Generalitat de Catalunya (SAC group Ref. 2005/SGR/00537). The authors thank the invaluable comments and suggestions received from the reviewers of this paper.

REFERENCES

1. Blanke M, Kinnaert M, Lunze J, Staroswiecki M. *Diagnosis and Fault-Tolerant Control* (2nd edn). Springer: Berlin, 2006.
2. Chen J, Patton R, Chen Z. An LMI approach to fault-tolerant control of uncertain systems. *Proceedings of the IEEE Conference on Decision and Control*, Tampa, FL, U.S.A., vol. 1, 1998; 175–180.
3. Liang Y, Liaw D, Lee T. Reliable control of nonlinear systems. *IEEE Transactions on Automatic Control* 2000; **45**:706–710.
4. Qu Z, Ihlefeld C, Yufang J, Saengdeejing A. Robust control of a class of nonlinear uncertain systems. Fault tolerance against sensor failures and subsequent self recovery. *Proceedings of the IEEE Conference on Decision and Control*, Orlando, FL, U.S.A., vol. 2, 2001; 1472–1478.
5. Liao F, Wang J, Yang G. Reliable robust flight tracking control: an LMI approach. *IEEE Transactions on Control Systems and Technology* 2002; **10**:76–89.
6. Qu Z, Ihlefeld C, Yufang J, Saengdeejing A. Robust fault-tolerant self-recovering control of nonlinear uncertain systems. *Automatica* 2003; **39**:1763–1771.
7. Zhang Y, Jiang J. Bibliographical review on reconfigurable fault-tolerant control systems. *Proceedings of IFAC SAFEPROCESS*, Washington, U.S.A., 2003; 265–276.

8. Parisini T, Sacone S. Fault diagnosis and controller re-configuration: an hybrid approach. *Proceedings of 1998 IEEE/ISIC/CIRA/ISAS Joint Conference*, Gaithersburg, MD, 1998; 163–168.
9. Heemels W, De Schutter B, Bemporad A. Equivalence of hybrid dynamical models. *Automatica* 2001; **37**: 1085–1091.
10. Bemporad A, Morari M. Control of systems integrating logic, dynamics, and constraints. *Automatica* 1999; **35**(3):407–427.
11. Silva B, Stursberg O, Krogh B, Engell S. An assessment of the current status of algorithmic approaches to the verification of hybrid systems. *Proceedings of IEEE Conference on Decision and Control*, Orlando, FL, 2001; 2867–2874.
12. Torrisi F, Bemporad A. Hysdel—a tool for generating computational hybrid models for analysis and synthesis problems. *IEEE Transactions on Control Systems and Technology* 2004; **12**(2):235–249.
13. Sontag E. Nonlinear regulation: the piecewise linear approach. *IEEE Transactions on Automatic Control* 1981; **26**:2346–2358.
14. Van Der Schaft A, Schumacher J. Complementarity modelling of hybrid systems. *IEEE Transactions on Automatic Control* 1998; **43**:483–490.
15. De Schutter B, Van den Boom T. On model predictive control for max-min-plus-scaling discrete event systems. *Automatica* 2001; **37**(7):1049–1056.
16. Ames A, Sastry S. Characterization of zeno behavior in hybrid systems using homological methods. *Proceedings of the IEEE American Control Conference*, Portland, OR, U.S.A, June 2005; 1160–1165.
17. Branicky MS, Borkar VS, Mitter SK. A unified framework for hybrid control: model and optimal control theory. *IEEE Transactions on Automatic Control* 1998; **43**(1):31–45.
18. Schutter BD. Optimal control of a class of linear hybrid systems with saturation. *Proceedings of 38th IEEE Conference on Decision and Control*, Phoenix, AZ, U.S.A., 1999; 3978–3983.
19. Tomlin C, Lygeros J, Sastry S. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE* 2000; **88**(7):949–970.
20. Lincoln B, Rantzer A. Optimizing linear system switching. *Proceedings of the 40th IEEE Conference on Decision and Control*, Orlando, FL, U.S.A., 2001; 2063–2068.
21. Bemporad A, Borrelli F, Morari M. On the optimal control law for linear discrete time hybrid systems. *Hybrid Systems*: *Computation and Control*. Lecture Notes of Computer Science, vol. 228. Springer: Berlin, 2002; 105–119.
22. Bemporad A. Efficient conversion of mixed logical dynamical systems into an equivalent piecewise affine form. *IEEE Transactions on Automatic Control* 2004; **49**(5):832–838.
23. Maciejowski J. *Predictive Control with Constraints*. Prentice-Hall: Great Britain, 2002.
24. Camacho E, Bordons C. *Model Predictive Control* (2nd edn). Springer: London, 2004.
25. Lazar M, Heemels W, Weiland S, Bemporad A. Stability of hybrid model predictive control. *IEEE Transactions on Automatic Control* 2006; **51**(11):1813–1818.
26. Bemporad A, Morari M, Dua V, Pistikopoulos E. The explicit linear quadratic regulator for constrained systems. *Automatica* 2002; **38**(1):3–20.
27. Borrelli F, Baotić M, Bemporad A, Morari M. Dynamic programming for constrained optimal control of discrete-time linear hybrid systems. *Automatica* 2005; **41**:1709–1721.
28. Bemporad A. *Hybrid Toolbox—User's Guide*, April 2006 [Online]. Available: http://www.dii.unisi.it/ hybrid/ toolbox.
29. Kerrigan E, Bemporad A, Mignone D, Morari M, Maciejowski J. Multi-objective prioritisation and reconfiguration for the control of constrained hybrid systems. *Proceedings of the American Control Conference*, Chicago, IL, 2000; 1694–1698.
30. Staroswiecki M. Actuator faults and the linear quadratic control problem. *Proceedings of the IEEE Conference on Decision and Control*, Hawaii, U.S.A., vol. 1, 2003; 959–965.
31. Ocampo-Martinez C, Guerra P, Puig V, Quevedo J. Actuator fault tolerance evaluation of linear constrained MPC using zonotope-based set computations. *Journal of Systems and Control Engineering* 2007; **22**(16):915–926.
32. Kerrigan E, Maciejowski J. Designing model predictive controllers with prioritised constraints and objectives. *Proceedings of IEEE International Symposium on Computer Aided Control System Design*, Glasgow, Scotland, vol. 1, 2002; 33–38.
33. Jaulin L, Kieffer M, Braems I, Walter E. Guaranteed nonlinear estimation using constraint propagation on sets. *International Journal of Control* 2001; **74**(18):1772–1782.
34. Jaulin L, Kieffer M, Didrit O, Walter E. *Applied Interval Analysis*, *with Examples in Parameter and State Estimation*, *Robust Control and Robotics*. Springer: London, 2001.

35. Gelormino M, Ricker N. Model-predictive control of a combined sewer system. *International Journal of Control* 1994; **59**:793–816.
36. Singh V. In *Hydrologic Systems*: *Rainfall-runoff Modeling*, Cliffs E (ed.), vol. I. Prentice-Hall: New Jersey, 1988.
37. Ocampo-Martinez C. Model predictive control of complex systems including fault tolerance capabilities: application to sewer networks. *Ph.D. Dissertation*, Technical University of Catalonia, April 2007.
38. Smith K, Austin G. Nowcasting precipitation—a proposal for a way forward. *Journal of Hydrology* 2000; **239**:34–45.
39. Yuan J, Tilford K, Jiang H, Cluckie I. Real-time urban drainage system modelling using weather radar rainfall data. *Physics and Chemistry of the Earth* (*B*) 1999; **24**:915–919.
40. Cembrano G, Quevedo J. In *Optimization in Water Networks*, Powell R, Hindi K (eds). Research Studies Press: U.K., 1999.
41. Ocampo-Martinez C, Puig V, Quevedo J, Ingimundarson A. Fault tolerant model predictive control applied on the Barcelona sewer network. *Proceedings of IEEE Conference on Decision and Control* (*CDC*) *and European Control Conference* (*ECC*), Seville, Spain, 2005.
42. Marinaki M, Papageorgiou M. *Optimal Real-time Control of Sewer Networks*. Springer: Berlin, 2005.
43. Ermolin Y. Mathematical modelling for optimized control of Moscow's sewer network. *Applied Mathematical Modelling* 1999; **23**:543–556.
44. Ocampo-Martinez C, Bemporad A, Ingimundarson A, Puig V. On hybrid model predictive control of sewer networks. In *Identification and Control*: *The Gap between Theory and Practice*, Sánchez-Peña R, Puig V, Quevedo J (eds). Springer: Berlin, 2007.
45. ILOG. *ILOG CPLEX 9.1 User's Manual*. 2003.
46. Ingimundarson A, Ocampo-Martinez C, Bemporad A, Puig V. Suboptimal hybrid model predictive control: application to sewer networks. *Proceedings of IFAC World Congress*, Seoul, Korea, July 2008.