Actuator fault tolerance evaluation approach of nonlinear model predictive control systems using viability theory

Majid Ghaniee Zarch [a], Vicenç Puig [b,*], Javad Poshtan [a], Mahdi Aliyari Shoorehdeli [c]

[a] Electrical Engineering Department, Iran University of Science and Technology (IUST), Narmak, 16846-13114 Tehran, Iran
[b] Institut de Robtica i Informtica Industrial (CSIC-UPC), Carrer Llorens Artigas, 4-6, 08028 Barcelona, Spain
[c] Electrical Engineering Department, K.N. Toosi University of Technology, Seyed Khandan, 16317 Tehran, Iran

ARTICLE INFO

Article history:
Received 25 February 2018
Received in revised form 20 August 2018
Accepted 22 August 2018
Available online 15 October 2018

Keywords:
Fault tolerance evaluation
Viability theory
Set invariance theory
Model predictive control
Linear parameter varying system

ABSTRACT

In this paper, an approach to evaluate the actuator fault tolerance of a nonlinear model predictive control (NMPC) system using viability theory is proposed. Viability theory provides several concepts formulated in a set form (viability kernel and capture basin) that are very useful to assess if after the fault the NMPC controller will be able to achieve their goal either using a reconfiguration or an accommodation strategy. By representing the nonlinear model of the system in a linear parameter varying (LPV) form and using zonotopes to evaluate viability sets, an algorithm is developed and implemented that is able to assess the tolerance of the NMPC controller. To illustrate the proposed approach two application examples based on well-known control problems (a pasteurization plant and a mobile robot) are used.

1. Introduction

Modern control systems are developed taking into account the demand for reliability, safety and fault tolerance. Consequently, it is necessary to design control systems that are capable of tolerating faults. A closed-loop control system which can tolerate component malfunctions, while maintaining desirable performance and stability properties is said to be a fault tolerant control system (FTCS) [1]. The goal of a fault tolerant system is to increase safety and reliability. Safety is the ability of a system to prevent any danger to human life, equipment or environment, while reliability is the ability of a system to perform required functions correctly over a given period of time under a given set of conditions. Control reconfigurability assesses the ability of system to allow performance restoration in the presence of faults [2]. Some works provide quantitative measures for reliability and reconfigurability of FTCS [1]. Automated and real-time analysis for the reliability and reconfigurability of FTCS are also important topics that have been researched [3].

After a fault has occurred, the control loop should be reconfigured to cope with the fault effects. Before starting to apply the fault tolerant control strategy, it should be evaluated whether the controller will be able to continue operating after the fault appearance. In this paper, this is done by means of an analysis based on the viability theory [4]. Considering whether is worth to attempt to recover the original system performance after occurrence of a fault or to accept some degree of performance degradation is very important issue before any fault-tolerant strategy is applied to the control loop. In practice, if a sensor fault occurs, the original system performance could be recovered as long as the correct information can be obtained either from a redundant sensor or from a virtual sensor based on the use of observers. However, in case of actuator faults, to maintain the original performance, the remaining actuators could be forced to work beyond their normal range of operation. This is undesirable in practice because this may lead to actuator saturation or increase deterioration. Therefore, a trade-off between achievable performance and available actuator capabilities must be considered. This situation is often referred to as graceful degradation in performance [1].

Viability theory develops mathematical and algorithmic methods for investigating the adaptation to viability constraints of evolutions governed by complex systems under uncertainty [4]. Viability is a theory that until now has mostly used in safety verification in control systems [5]. Viability theory has also been found useful in areas different from automatic control as e.g. economics [6] or biology [7]. This theory provides some concepts that are actually more general than what is used in set-invariance theory [8]. However, both theories share a common concept: the invariant set or invariant kernel that can be used for proving the stability of control systems. Viability kernel is an accepted tool for safety verification. However, the problem with this theory is how to compute the different sets involved. Nowadays, several algorithms have

* Corresponding author.
E-mail address: vicenc.puig@upc.edu (V. Puig).

been proposed that can approximate these sets effectively. Some of these algorithms are surveyed [5].

The use of viability theory to fault diagnosis has already been explored by the authors [9] following similar ideas [10] but using set-invariance theory. Finding the viability theory concepts that can be used in fault tolerance evaluation is a major contribution of this paper. This paper will also try to relate these concepts with set-based concepts introduced to address the admissibility evaluation defined in [11] in the context of Model Predictive Control (MPC).

Faults will cause changes in the set of feasible solutions. This causes that the set of admissible solutions for the control objective could be empty. Therefore, the admissibility of the control law facing faults can be determined knowing the feasible solution set. One of the aims of this paper is to provide methods to compute this set and to evaluate the admissibility of the control law.

This paper will focus on the fault tolerant evaluation approach of a given actuator fault configuration, considering a nonlinear model predictive control (NMPC) law with constraints. However, the proposed approach could be easily applied to any other type of controller. The method proposed in this paper is not of analytical but of computational nature. It follows the idea proposed by [11] for linear model predictive control (MPC): Faults can cause changes in the constraints related to the control signals (inputs), which modifies the set of feasible solutions of the MPC controller. This can cause the set of feasible solutions for a given control objective to be empty.

In this paper, an approach to evaluate the actuator fault tolerance of NMPC system using viability theory is proposed. Viability theory provides several concepts formulated in a set form (viability kernel and capture basin) that are very useful to assess if after the fault the NMPC controller will be able to achieve their goal either using a reconfiguration or an accommodation strategy. By representing the nonlinear model of the system in a linear parameter varying (LPV) form and using zonotopes to evaluate viability sets, an algorithm is developed and implemented that is able to assess the tolerance of the NMPC controller. To illustrate the proposed approach two application examples based on well-known control problems (a pasteurization plant and a mobile robot) are used.

This paper is organized as follows: In Section 2, some definitions and preliminary concepts are provided in context of viability theory as well as algorithms for computing relevant sets in viability theory used in the paper are proposed using zonotope representation and arithmetic. Problem formulation is presented in Section 3. How viability theory can be used in fault tolerance evaluation is a task that will be addressed in Section 4. An algorithm for admissibility evaluation of a given fault configuration will be the main outcome of this section. Water heater section of pasteurization process and a wheeled mobile robot are considered as case studies in Section 5 in order to illustrate the proposed approach. Finally, in Section 6 concluding remarks are drawn. In the Appendix, necessary background on zonotopes is provided.

2. Review of viability theory concepts

2.1. Viability theory concepts

In this paper, we assume that the system to be controlled can be represented with a nonlinear model in a LPV (or quasi-LPV) representation

$$x(t+1) = A(\rho(t))x(t) + B(\rho(t))u(t) + B_d(\rho(t))d(t) \tag{1}$$

where $x(t) \in X$ is the state, $u(t) \in U$ is the control input and $d(t)$. The bounding sets X and U are defined as

$$X = \left\{x(t) \in \mathbb{R}^n : \left|x(t) - x_t^c\right| \leqslant \bar{x}_t, x_t^c \in \mathbb{R}^n, \bar{x}_t \in \mathbb{R}^n\right\}$$
$$U = \left\{u(t) \in \mathbb{R}^m : \left|u(t) - u_t^c\right| \leqslant \bar{u}_t, u_t^c \in \mathbb{R}^m, \bar{u}_t \in \mathbb{R}^m\right\}$$

where $x_t^c, u_t^c, \bar{x}_t$ and $\bar{u}_t$ are constant vectors. The sets X and U can be rewritten as zonotopes

$$X = x_t^c \oplus H^{\bar{x}_t}\beta^n$$
$$U = u_t^c \oplus H^{\bar{u}_t}\beta^m$$

where $H^{\bar{x}_t} \in \mathbb{R}^{n\times n}$ and $H^{\bar{u}_t} \in \mathbb{R}^{m\times m}$ are diagonal matrices with their diagonal entries composed of $\bar{x}_t$ and $\bar{u}_t$, respectively. The parameter ρ is a time-varying parameter whose measurement is available and used to embed nonlinear terms of the system model. For more details about how to obtain LPV models for nonlinear systems, see [12,13].

Remark. The disturbance $d(t)$ can be bounded and managed similarly to the controlled input $u(t)$ (see [4] for more details). For this reason and for simplicity in the explanations, in the following the viability theory definitions and algorithms will be presented only considering the controlled input $u(t)$. Moreover, the consideration of uncertainty in the disturbance in the MPC controller would lead to use robust MPC schemes that is out of the scope of the paper [14].

It is assumed that the system (1) is defined in a proper open set $\Phi \subseteq \mathbb{R}^n$ and that there exist a globally defined solution for every initial condition $x(0) \in \Phi$. We assume for each $u(t) \in U$ and $x(t) \in X$, Eq. (1) has a unique solution

$$S(t, x, u), \quad t \in \mathbb{T}$$

where $S(0, x, u) = x(0)$.

Viability theory is concerned with ensuring that a system state remains within a viability constraint set K. Any trajectory of system (1) that leaves the set K at some point in time is considered to be no longer viable.

Definition 2.1 *(Viability kernel).* The viability kernel of $K \subseteq \Phi$ under the evolutionary system $S(t, x, u)$ is the set $Viab_S(K)$ of initial states $x(0) \in K$ from which starts at least one evolution $x(t) \in S(t, x, u)$ viable in K for all times $t \geqslant 0$

$$Viab_S(K) = \left\{x(0) \in K | \exists u \in U, \forall t \geqslant 0; x(t) \in K\right\} \tag{2}$$

This means that from a point $x(0)$ in the viability kernel of the environment K starts at least one evolution viable in K forever. This is equivalent to say that all evolutions starting from a state belonging to the complement of the viability kernel K leave the environment in finite time.

Viability kernel and weak positive invariance are equivalent definitions in viability and set invariance theories, respectively [8]. Capture basin is another concept that has a wide range of applications, for example, in process control [15] and economics [16].

Definition 2.2 *(Capture basin).* The capture basin of T (viable in K) under the evolutionary system $S(t, x, u)$ is the set $Capt_S(K, T)$ of initial states $x(0) \in K$ from which starts at least one evolution $x(t) \in S(t, x, u)$ viable in K on $[0, N)$ until the finite time N when the evolution reaches the target at $x(N) \in T$

$$Capt_S(K, T) = \left\{x(0) \in K | \exists u \in U, \forall t \in [0, N); x(t) \in K, x(N) \in T\right\}$$

From a state $x(0)$ in the capture basin of the target T viable in the environment K starts at least one evolution viable in K until it reaches T in finite time. This is equivalent to say that, starting from a state belonging to the complement of K, all evolutions remain outside the target T until they leave the environment K. Development of the methods for obtaining these two sets is still an important area and not an easy task [5].

2.2. Viability kernel computations

Reachability analysis identifies the set of states backward (forward) reachable by a constrained dynamical system from a given target (initial) set of states. The notions of maximal and minimal reachability analysis were introduced in [17]. Their corresponding constructs differ in how the time variable and the bounded input are quantified. In the formation of the maximal reachability construct, the inputs tries to steer as many states as possible to the target set. On the other hand, in the formation of the minimal reachability construct, the trajectories reach the target set regardless of the input applied. Based on these differences, the maximal and minimal reachable sets and tubes (the set of states traversed by the trajectories over the time horizon [17]) are formed.

In [4,17], it is shown that the minimal reachable tube and the viability kernel are the only constructs that can be used to prove safety of the system and to synthesize inputs (controllers) that preserves this safety. Since the viability kernel and the minimal reachable tube are dual concepts, they do not need to be treated separately.

Definition 2.3 *(Backward maximal reachable set).* The backward maximal reachable set at time instant t is the set of initial states for which there exists an input such that the trajectories emanating from those states reach T exactly at time instant t:

$$\text{Reach}_t^B(T) \triangleq \left\{x(0) \in \mathbb{R}^n | \exists u(.) \in U_{[0,t]}, x(t) \in T\right\} \tag{3}$$

There exists several families of sets which can be used to compute the viability sets with varying degrees of accuracy. An important limiting factor is the numerical reliability of their representation. That is, a particular family may be able to represent a great number of shapes but due to computationally expensive manipulations will be useless in practice. Usually there exists an inverse relation between flexibility of a family and the numerical cost of the representation.

Here we will use zonotopes. See Appendix for more details. Using zonotopic representation and operations in the Appendix, the backward reachable set for system (1) over a single time step is computed as

$$\text{Reach}_1^B(X) = A(\rho(t))^{-1}\left\{X \oplus (-B(\rho(t)))U\right\} \tag{4}$$

Here $A^{-1}(.)$ denotes the preimage of a set under the map $A: \mathbb{R}^n \to \mathbb{R}^n$. Note that we consider that A is invertible. This is a fair assumption because we are mainly concerned with discrete-time systems that arise from the discretization of continuous-time systems. Such systems have a dynamics matrix of the exponential form which is always invertible [5]. Following computation algorithm in [18], we can find this reachable set using zonotopes by

$$\text{Reach}_1^B(X) = x_{t+1}^c \oplus H^{\bar{x}_{t+1}}\beta^n \tag{5}$$

where

$$x_{t+1}^c = mid\left(A(\rho(t))^{-1}\right)x_t^c + mid\left(-A(\rho(t))^{-1}B(\rho(t))\right)u_t^c$$

$$H^{\tilde{x}_{t+1}} = \begin{bmatrix} J_1 & J_2 & J_3 & J_4 \end{bmatrix}$$

$$J_1 = seg\left(\Diamond A(\rho(t))^{-1}H^{\tilde{x}_t}\right)$$

$$J_2 = \frac{diam\left(A(\rho(t))^{-1}\right)}{2}x_t^c$$

$$J_3 = seg\left(-\Diamond A(\rho(t))^{-1}B(\rho(t))H^{\tilde{u}_t}\right)$$

$$J_4 = \frac{diam\left(-A(\rho(t))^{-1}B(\rho(t))\right)}{2}u_t^c$$

'*mid*' denotes the center, '*diam*' the diameter of the interval, $seg(Q) = H$ considering that Q is a zonotope and $\Diamond$ is zonotope inclusion. For computation of viability kernel, this backward reachable set is calculated step by step. The reachable tube will finally converge toward the viability kernel.

Algorithm 1. Viability kernel estimation.

$K_0 \longleftarrow X$
$t \longleftarrow 1$
while $t \leqslant N$ **do**
  if $K_t = \emptyset$ **then**
$K_N \longleftarrow \emptyset$
break
  end if
  if $K_t = K_{t-1}$ **then**
$K_N \longleftarrow K_t$
break
  end if
  $L \longleftarrow \text{Reach}_1^B(K_t)$ (see Eq. (5))
  $K_{t+1} \longleftarrow K_0 \cap L$ (see Eq. (27))
  $t \longleftarrow t+1$
end while
return (K_N) $K_N = Viab_S(X)$

It is important to notice that the set of estimated states is approximated by a zonotope that has an increasing number of segments $\text{Reach}_1^B(K_t)$ using this method. In order to control the domain complexity, a reduction step is thus implemented. Here, we use the method proposed in [19] to reduce the zonotope complexity.

Finally, for computation of capture basin, we can find backward reachable tube using (5) for desired time steps. Final reachable set is the capture basin.

Algorithm 2. Capture basin estimation.

$K_0 \longleftarrow C$
$t \longleftarrow 1$
while $t \leqslant T$ **do**
  if $K_t = \emptyset$ **then**
$K_T \longleftarrow \emptyset$
break
  end if
  $K_{t+1} \longleftarrow \text{Reach}_1^B(K_t)$ (see Eq. (5))
  $t \longleftarrow t+1$
end while
$K_N = K_T \cap X$ (see Eq. (27))
return (K_N) $K_N = Capt_S(X, C)$

3. Problem formulation

3.1. MPC control problem

The solution of a control problem consists in finding a control law in a given set of control laws U such that the controlled system achieves the control objectives O while its behavior satisfies a set of constraints C. Thus, the solution of the problem is completely defined by the triple $\langle O, C, U\rangle$ [20]. In the following, let particularize this general definition to the case that we use a MPC controller.

Let us denote the sequence variables over the time horizon N

$$\tilde{x} = (x(t))_0^N = (x(0), x(1), \ldots, x(N)),$$

$$\tilde{u} = (u(t))_0^{N-1} = (u(0), u(1), \ldots, u(N-1)).$$

of the dynamic system (1) expressed here for simplicity as follows

$$x(t+1) = F(x(t), u(t)) \tag{6}$$

where F is the function that describes the dynamics of the system.

Thus, in the case of a model predictive control law, the triple $\langle O, C, U\rangle$ is defined by

$$O: \min_{\tilde{u}} J(\tilde{x}, \tilde{u}) \tag{7}$$

subject to

$$C:\begin{cases} x(t+1)=F(x(t),u(t)) \\ u(t)\in U \qquad t\in[0,N-1]\subset\mathbb{N} \\ x(t)\in X \qquad t\in[0,N]\subset\mathbb{N} \end{cases} \tag{8}$$

where

$$U\triangleq\left\{u(t)\in\mathbb{R}^m \,|\, u_{\min}\le u(t)\le u_{\max}\right\} \tag{9}$$

$$X\triangleq\left\{x(t)\in\mathbb{R}^n \,|\, x_{\min}\le x(t)\le x_{\max}\right\}, \tag{10}$$

The control law belongs to the set U and it is obtained using the *receding horizon philosophy* [14]. This technique consists on taking only the first value from the sequence $\tilde{u}$ computed at each time instant by solving the previous optimization problem. The initial states $x(0)$ are updated from measurements or state estimation. The objective function J is defined, in general form, as

$$J(\tilde{x},\tilde{u})=\phi(x(N))+\sum_{i=0}^{N-1}\Phi(x(i),u(i)) \tag{11}$$

where ϕ is a function that constrains the final state value over N and Φ is a function of states and inputs. Depending on the applications, the objective function forces the system to follow a reference $(u_{ref}(t),x_{ref}(t))$ or to optimize some economic performance index.

3.2. Fault tolerant control problem

Fault tolerant control is concerned with the control of the faulty system. This can be done by changing the control law without changing the plant (adaptation, accommodation), or by changing both the control and the system (reconfiguration). As the result of the fault, the control problem is transformed from $\left\langle O, C\left(\theta\right), U\right\rangle$ into $\left\langle O, C_f\left(\theta_f\right), U_f\right\rangle$.

$$O:\min_{\tilde{u}} J(\tilde{x},\tilde{u}) \tag{12}$$

$$C_f:\begin{cases} x(t+1)=A(\rho(t))x(t)+B(\rho(t))u(t) \\ u(t)\in U_f \qquad t\in[0,N-1]\subset\mathbb{N} \\ x(t)\in X_f \qquad t\in[0,N]\subset\mathbb{N} \end{cases} \tag{13}$$

$$U_f\triangleq\left\{u(t)\in\mathbb{R}^m \,|\, u_{f,\min}\le u(t)\le u_{f,\max}\right\} \tag{14}$$

$$X_f\triangleq\left\{x(t)\in\mathbb{R}^n \,|\, x_{f,\min}\le x(t)\le x_{f,\max}\right\}, \tag{15}$$

Suppose that both $C_f\left(\theta_f\right)$ and U_f are perfectly known, then the fault tolerant control law is obtained by solving $\left\langle O, C_f\left(\theta_f\right), U_f\right\rangle$. If such a solution exists, the system is fault tolerant with respect to the objective O and the faulty situation $C_f\left(\theta_f\right), U_f$. If the problem $\left\langle O, C_f\left(\theta_f\right), U_f\right\rangle$ has no solution, then the system is not fault tolerant and the relaxation of the control objective has to be considered.

Fault tolerant approaches. Active fault tolerant control approaches requires that the knowledge of $C_f\left(\theta_f\right)$ and U_f is available to solve the fault tolerant control problem $\left\langle O, C_f\left(\theta_f\right), U_f\right\rangle$. Provide such fault information is the role of fault detection and isolation (FDI) algorithm. In case that such algorithm is not available, a passive fault tolerant control approach based on robust control should be used where set of faulty parameters should be considered $\hat{\Theta}_f$ instead of estimation of the faulty parameters $\hat{\theta}_f$.

Remark. In this paper, it will be assumed that FDI algorithm exists and has been implemented using standard model-based approaches as the ones presented in as e.g. [21]. As discussed in the introduction, the FDI could also be implemented using viability theory as presented in [9].

Fault tolerant strategies.

If an active FTCS is considered, two main strategies to adapt the MPC law in order to introduce fault tolerance can be applied [20]:

1. *System reconfiguration*: Consists in finding a new set of constraints $C_f(\theta_f)$, where θ_f is the set parameters changed by the faults, such that the control problem $\langle O, C_f(\theta_f), U_f\rangle$ can be solved. This strategy could be applied in case the fault detection and isolation (FDI) module does not provide the fault estimation. Therefore, the faulty components are unplugged by the supervisory system and the control objectives should be reached using the non-faulty components.

   In case of actuators, this implies that the model (1) used by the MPC controller is modified as follows:

   $$x(t+1)=A(\rho(t))x(t)+\sum_{i\in I_N}B_i(\rho(t))u_i(t)+B_d(\rho(t))d(t), \tag{16}$$

   where I_N is the subset of still-healthy actuators.
2. *Fault accommodation*: Consists in solving the control problem $\langle O,\hat{C}_f(\hat{\theta}_f), \hat{U}_f\rangle$, being $\hat{C}_f(\hat{\theta}_f)$ an estimation of current system constraints and parameters provided by the FDI module. This strategy can be applied when a change either in the system structure or parameters occurs. In this strategy, the control law is modified while the rest of the elements within the control loop are kept unchanged.

   In case of actuators, this implies that the system model (1) used by the MPC controller should be modified as follows:

   $$x(t+1)=A(\rho(t))x(t)+\sum_{i\in I_N}B_i(\rho(t))u_i(t) +\sum_{i\in I_F}\beta_i(u_i(k),\theta_i)+B_d(\rho(t))d(t), \tag{17}$$

   where the function β_i as well as the parameters θ_i should be estimated by the FDI module in case of actuators belonging to the faulty actuator subset I_F.

4. Fault tolerance evaluation using viability theory

4.1. Introduction

In this section, an algorithm is provided to evaluate admissibility of a FTCS using viability theory concepts presented in Section 2 when fault-tolerant strategy (accommodation/reconfiguration) has been applied as described in previous section.

Based on the viability concepts, it can be readily deduced that there are some conditions that allow us to use viability theory in fault tolerance evaluation.

Definition 4.1 *(Feasible solution set).* The feasible solution set of the MPC controller (7) and (8) is given by

$$\Omega=\left\{(\tilde{x},\tilde{u})\in(X,U)\,|\,\forall t\in[0,N], x(t)\in Viab_S(X)\right\}$$

Definition 4.2 *(Feasible objective set).* The feasible objective set of the MPC controller (7) is given by

$$\Gamma_\Omega=\left\{J(\tilde{x},\tilde{u})\,|\,(\tilde{x},\tilde{u})\in\Omega\right\}$$

4.2. Fault tolerance evaluation

Consider the system with a fault as in (13). Feasible solution and objective sets of the faulty system, named Ω_f and Γ_{Ω_f}, can be computed using (13) and previous definitions.

Assume that the desired control objective set is given as to achieve $\Gamma_A \subseteq \Gamma_\Omega$, therefore the target set can be defined as

$$T = \left\{ \tilde{x} \in Viab_S\left(X_f\right) | \exists \tilde{u} \in U_f, J(\tilde{x}, \tilde{u}) \in \Gamma_A \right\} \tag{18}$$

The target T is the set of desired states for the system with acceptable performance. Then, the admissible solution set is the set of solutions that is achievable in the faulty case. This concept can be redefined in the viability theory terms.

Remark. In the case of a MPC controller, the target T can be defined by establishing the degradation of the control objective in a faulty situation. This can be quantified by means of a maximal loss of efficiency α with respect to the objective function in non-faulty situation J_0. This allows to establish if the control objective degradation after an actuator fault J_f is acceptable. Thus, an actuator fault configuration is admissible regarding performance if the following condition is satisfied $J_f \leq (1+\alpha)J_0$. This will allow a performance analysis considering the faulty actuator either with an accommodation or a reconfiguration strategy.

Definition 4.3 *(Admissible solution set).* Given the subset Ω_f as the feasible solution set of faulty system (13), the admissible solution set is defined as

$$\Xi = \left\{ (\tilde{x}, \tilde{u}) \in \Omega_f | x(t) \in Capt_S\left(Viab_S\left(X_f\right), T\right) \right\}$$

4.3. Algorithm

Now, a fault tolerance evaluation algorithm can be proposed using previous definitions and viability theory sets. The algorithm starts obtaining the viability kernel $Viab_S(K)$ given a set of initial states and the system dynamics in faulty condition (13). After finding viability kernel based on constraints of states and inputs, the capture basin can be obtained.

Considering that there exists a fault in the system, the admissible solution set can be obtained using the model for the faulty system dynamics (13) and constraints. In this manner, the new viability kernel can define the set of admissible states of the system after fault occurs allowing to check if the reference is achievable or not with acceptable performance.

On the other hand, finding capture basin with faulty dynamics (13) allows to verify in at least how many steps the system can reach the target. The target can be considered as small set near steady state inside viability kernel or a small set around a predefined trajectory. Algorithm 3 summarizes the procedure for fault tolerance evaluation using viability theory concepts.

Algorithm 3. Fault tolerance evaluation using viability theory.

for $t = 1, 2, \ldots$ **do**
  apply the fault tolerant strategy to the MPC controller (accommodation/reconfiguration)
  find $Viab_S(K)$ using Algorithm 1
  find $Capt_S(Viab_S(K), T)$ using Algorithm 2
  find Ω_f using Definition 4.1
  find Ξ using Definition 4.3
  if the reference $x_{ref} \in \Omega_f$ **then**
the reference is achievable
if $\Xi = \emptyset$ **then**
the system is not fault tolerant
else
the system is fault tolerant
end if
  else
the reference is not achievable, therefore the system is not fault tolerant
  end if
end for

5. Application examples

5.1. Pasteurization plant

The proposed method for fault tolerance evaluation has been tested in simulation using the water heater part of a pasteurization plant. Fig. 1 presents the conceptual diagram. The reservoir is an electrically heated, which is covered in order to minimize heat losses [22]. The water is heated by means of the power P supplied to the resistance. A peristaltic pump with an upper limit of 700 ml/min moves the heated water. This flow, described as hot flow F_h, transfers heat to the pasteurization product in a heat exchanger, before returning to the heat water. If the flow is maintained at a constant value, the water heater behaves as a linear process. However, in the pasteurization process, F_h changes frequently to provide the adequate amount of energy to maintain the heat exchanger input temperature.

The water heater can be modeled as a LPV model (1) that considers the flow F_h as the scheduling variable. The state space model of the heater G_p has the following parameters [22]: $A_p = -1/z_p$, $B_p = k_p/z_p$, $C_p = 1$, $D_p = 0$ while the state space model of the holding tube G_d is $A_d = -1/z_d$, $B_d = k_d/z_d$, $C_d = 1$, $D_d = 0$ where

$$\begin{aligned}
k_d = {} & -1.3035 \times 10^{-11} \times {F_h}^4 + 2.1189 \times 10^{-8} \times {F_h}^3 \\
& -1.3487 \times 10^{-5} \times {F_h}^2 + 0.0044 \times F_h + 0.2127 \\
k_p = {} & 1.3426 \times 10^{-12} \times {F_h}^4 - 2.3729 \times 10^{-9} \times {F_h}^3 \\
& +1.6624 \times 10^{-6} \times {F_h}^2 - 0.0006 \times F_h + 0.1189 \\
z_d = {} & 3.8483 \times 10^{-8} \times {F_h}^4 - 6.7677 \times 10^{-5} \times {F_h}^3 \\
& +0.0471 \times {F_h}^2 - 16.8007 \times F_h + 3303.7905 \\
z_p = {} & 3.8483 \times 10^{-8} \times {F_h}^4 - 6.7677 \times 10^{-5} \times {F_h}^3 \\
& +0.0471 \times {F_h}^2 - 16.8007 \times F_h + 3303.7905
\end{aligned}$$

The flow F_h is considered to vary in [100, 600]. The model of the heater and holding tube are integrated in a single state space model as follows

$$\begin{aligned}
\dot{x} &= \begin{bmatrix} A_p & 0 \\ 0 & A_d \end{bmatrix} x + \begin{bmatrix} B_p & 0 \\ 0 & B_d \end{bmatrix} \begin{bmatrix} P \\ T_{iw} \end{bmatrix} \\
y &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x
\end{aligned} \tag{19}$$

where $x = \begin{bmatrix} x_p & x_d \end{bmatrix}^T$ is the state vector. The system (19) is discretized by Euler method considering a sampling time $T = 60$(s). In healthy state, the controlled input $P(w)$ is considered to be in [0, 2000] and disturbance $T_{iw} = 20$ (°C). It is clear that the desired state to be controlled is x_p. For viability kernel estimation, the initial set is considered to be

$$X_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \oplus \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \end{bmatrix} \beta^2$$

This zonotope covers all the possible values of the states. The viability kernel for the system in healthy mode is presented in Fig. 2. Some approximations in Algorithm 1 lead to deviation from initial zonotope. With good accuracy, it can be seen that the system in healthy mode could track nearly all possible references considering normal system condition and input.

Now, two actuator fault scenarios are considered affecting the range of the operation of the controlled input P:

- Fault scenario 1: $P \in [0, 500]$
- Fault scenario 2: $P \in [0, 1000]$

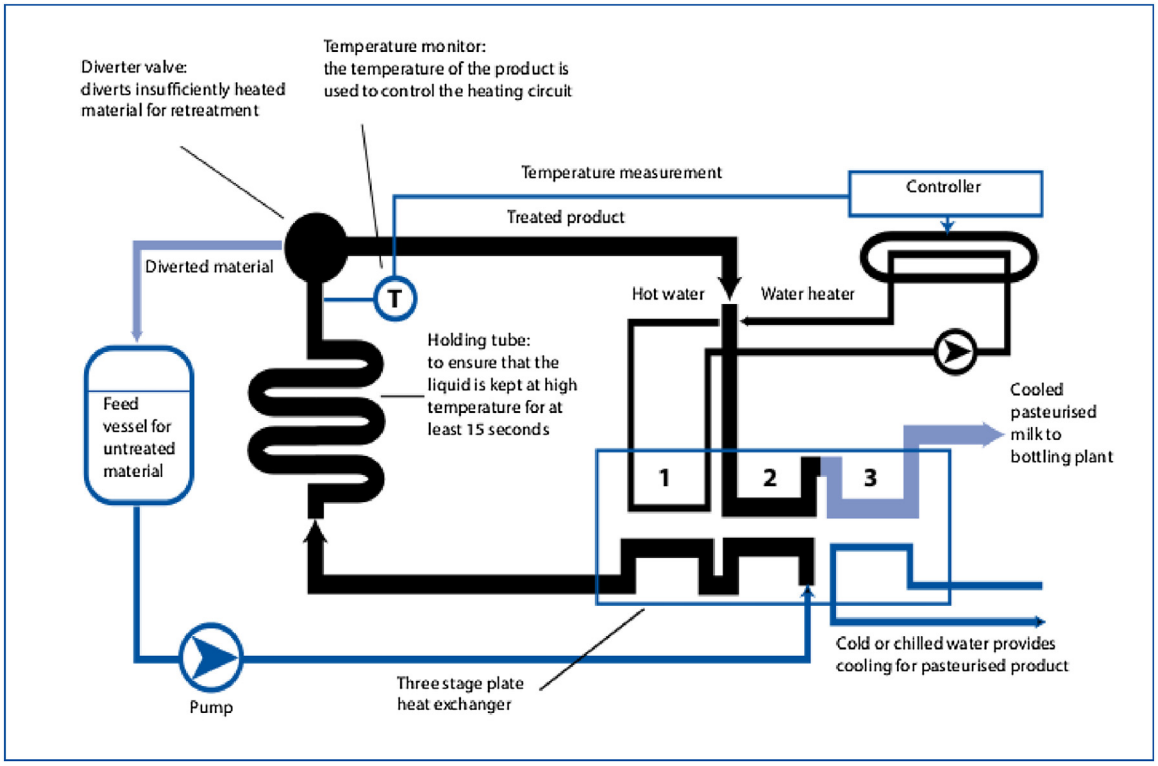

Fig. 1. Conceptual diagram of the pasteurization plant.

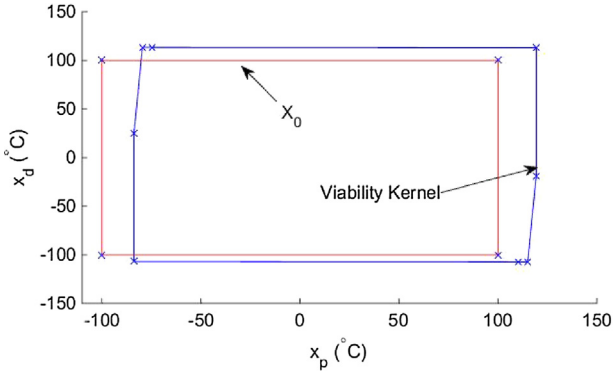

Fig. 2. Viability kernel for the system (19) in state space plane.

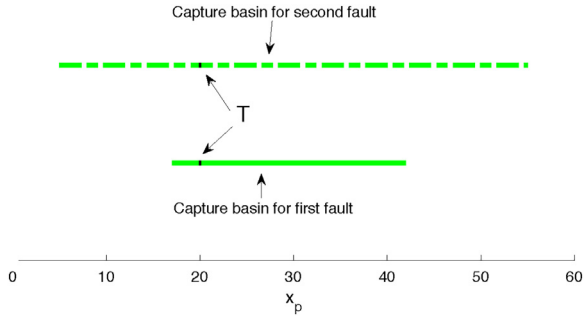

Fig. 3. Capture basin for the first and second fault scenarios in 10 sample times.

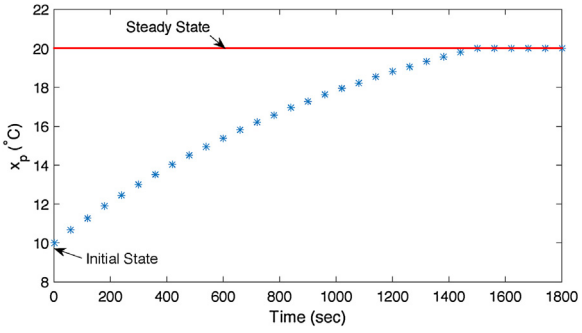

Fig. 4. Closed-loop simulation with first fault scenario.

Now consider that after fault occurrence, a fault accommodation strategy is applied in the controller by updating the control bounds, the system output (19) is in $y = \begin{bmatrix} 10 & 14 \end{bmatrix}^T$ and the goal is to move the system output to $y = \begin{bmatrix} 20 & 14 \end{bmatrix}^T$. Therefore, the target is considered to be:

$$T = \begin{bmatrix} 20 \\ 14 \end{bmatrix} \oplus \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \beta^2$$

Consider that the system has only 10 sample times to go to target. To check in which of the considered fault scenarios the goal is achievable, the capture basin considering 10 time steps is computed and presented in Fig. 3. The state x_d is not important in our analysis and therefore it has not been shown in the figure. It is clear that if the considered specification for the system is to achieve the target in 10 steps, only in second faulty case it is tolerant. This means that the NMPC controller will be able to find an admissible trajectory to reach the target in the desired time (600 s) in spite of the fault. Capture basin for second fault scenario is much bigger than capture basin in first fault scenario. This result can be expected as first fault is more severe than second one. However, if the goal is to achieve the target in 5 sample times, none of the above fault scenarios could be tolerated.

A closed loop system simulation using the NMPC controller has been conducted in Figs. 4 and 5 to show that the results obtained are

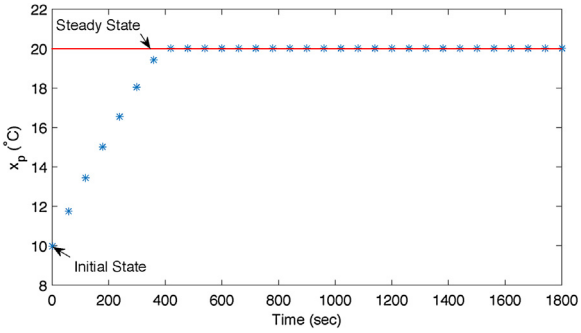

Fig. 5. Closed-loop simulation with second fault scenario.

in accordance with the ones obtained with the proposed viability based methodology. In Fig. 4, it takes more than twenty sample times to get the system to steady state, while in Fig. 5 less than ten sample times are required.

5.2. Wheeled mobile robot

A second example based on wheeled mobile robot (WRM) is used for further illustrating the proposed approach (see [23] for additional details). A kinematic model of WMR is used for control purposes considering the non-holonomic constraint of pure rolling and non-slipping. Based on the kinematic model, an error model and a controller for the tracking control problem is presented. An LPV representation of the error model is used to derive the viability sets required for the application of the proposed fault tolerance evaluation approach.

One of the major differences in the kinematic model with respect to the dynamic model is the null skidding assumption. There exist another important assumption which is the consideration of small lateral forces. Both assumptions share the idea of low speed, therefore it can be said a kinematic model presents acceptable results when the vehicle goes at low speed [24]. In [25], a complete development of such a kinematic model can be found. The set of kinematic equations of the Cartesian position (x, y) and orientation (θ) of the real vehicle are presented as follows:

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_l \\ \omega \end{bmatrix} \tag{20}$$

where x_c and y_c denotes the position of the center of mass of the WMR along the x and y coordinate frames and θ represents the orientation of the WMR (see Fig. 6). Linear and angular velocities are denoted by v_l and ω, respectively. The control objective is to force the actual Cartesian position and orientation to a constant reference (regulation) or to track a time-varying reference trajectory (tracking). To quantify the tracking control objective, $\tilde{x}, \tilde{y}, \tilde{\theta} \in \mathbb{R}^1$ are defined as follows

$$\tilde{x} = x_c - x_{cr}, \quad \tilde{y} = y_c - y_{cr}, \quad \tilde{\theta} = \theta_c - \theta_r$$

where x_{cr}, y_{cr}, θ_r denotes the reference position and orientation. The kinematic equations for the reference system can be defined as

$$\begin{bmatrix} \dot{x}_{cr} \\ \dot{y}_{cr} \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \cos\theta_r & 0 \\ \sin\theta_r & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{lr} \\ \omega_r \end{bmatrix} \tag{21}$$

An error model between the virtual and real vehicle is typically used in robotics for solving the trajectory tracking problem [25]. The error vector has been defined as the difference between real measurements and desired values multiplied by the rotation matrix as follows

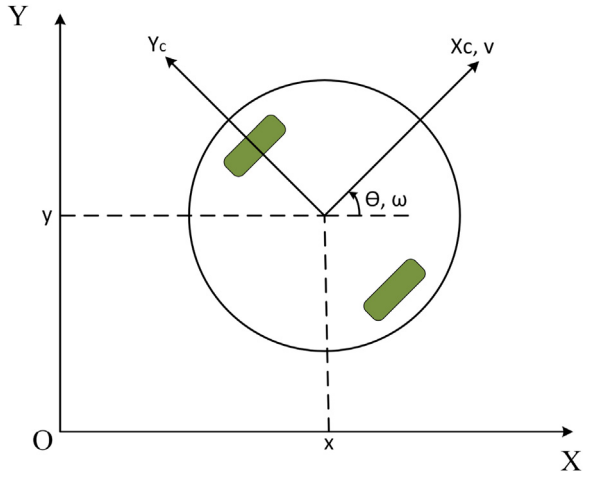

Fig. 6. Wheeled mobile robot (WMR).

$$\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{\theta} \end{bmatrix} \tag{22}$$

To develop the open-loop tracking error system, the time-derivative of Eq. (22) is taken to obtain the following expression

$$\begin{bmatrix} \dot{e}_x \\ \dot{e}_y \\ \dot{e}_\theta \end{bmatrix} = \begin{bmatrix} \omega e_y + v_{lr}\cos e_\theta - v_l \\ -\omega e_x + v_{lr}\sin e_\theta \\ \omega - \omega_r \end{bmatrix} \tag{23}$$

The trajectory tracking problem is solved with a NMPC strategy. To this aim, the kinematic error model (23) is discretized using forward Euler method by considering sample time $T_s = 0.1$ (s)

$$\begin{bmatrix} e_x(t+1) \\ e_y(t+1) \\ e_\theta(t+1) \end{bmatrix} = \begin{bmatrix} e_x(t) + T_s\left\{\omega(t)e_y(t) + v_{lr}(t)\cos(e_\theta(t)) - v_l(t)\right\} \\ e_y(t) + T_s\left\{-\omega(t)e_x(t) + v_{lr}(t)\sin(e_\theta(k))\right\} \\ e_\theta(t) + T_s\left\{\omega(t) - \omega_r(t)\right\} \end{bmatrix}$$

The above equation can be written in LPV form (1) as follows

$$\begin{aligned} \begin{bmatrix} e_x(t+1) \\ e_y(t+1) \\ e_\theta(t+1) \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_x(t) \\ e_y(t) \\ e_\theta(t) \end{bmatrix} + \begin{bmatrix} -T_s & T_s e_y(t) \\ 0 & -T_s e_x(t) \\ 0 & T_s \end{bmatrix} \begin{bmatrix} v_l(t) \\ \omega(t) \end{bmatrix} \\ &+ \begin{bmatrix} T_s\cos(e_\theta(t)) & 0 \\ T_s\sin(e_\theta(t)) & 0 \\ 0 & -T_s \end{bmatrix} \begin{bmatrix} v_{lr}(t) \\ \omega_r(t) \end{bmatrix} \end{aligned} \tag{24}$$

where $\begin{bmatrix} v_{lr}(t) & \omega_r(t) \end{bmatrix}^T$ is considered as a disturbance vector.

Defining $u(t) = \begin{bmatrix} v_l(t) & \omega(t) \end{bmatrix}^T$ as the control vector and $e(t) = \begin{bmatrix} e_x(t) & e_y(t) & e_\theta(t) \end{bmatrix}^T$ as the error vector, the following objective function for the NMPC controller can be formulated

$$J = \sum_{t=1}^{N} e^T(t)Qe(t) + u^T(t)Ru(t) \tag{25}$$

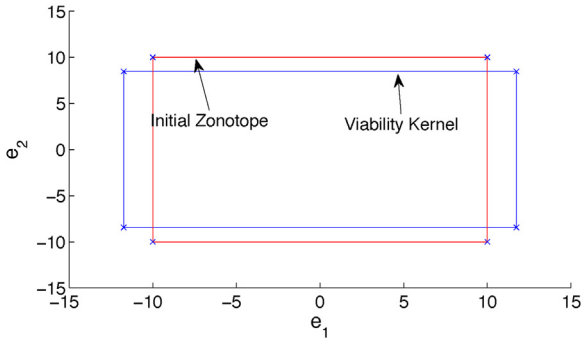

Fig. 7. Viability kernel for error model (24).

where N is the prediction horizon and $Q \geqslant 0$, $R > 0$ are weighting matrices for the error in the state and control variables, respectively.

$$Q = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$
$$R = 0.1 I_{2\times 2}$$

Bounds on the amplitude of the control variables are assumed as

$$\begin{bmatrix} 0 \\ -2\pi \end{bmatrix} \leqslant u(t) \leqslant \begin{bmatrix} 3 \\ 2\pi \end{bmatrix} \tag{26}$$

Hence, the NMPC problem (24) can be formulated considering the objective function (25), the model (Fig. 6) and the bounds (26).

In this case the viability kernel and capture basin are determined using the error model (23) instead of the plant model (20) since the control objective is to bring the error to zero. Therefore, the analysis can be done around origin instead of desired values. To derive kernels, initial zonotope for inputs and disturbances are

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 20 & 0 \\ 0 & 20 \end{bmatrix} \beta^2$$

Initial zonotope and viability kernel for the system (24) using the algorithm described in Section 2.1 is depicted in Fig. 7. It is clear that viability kernel is almost all the initial set (some approximations like zonotope order reduction has been done in the algorithm). It means that from where the system starts (inside initial zonotope), it finally can go to steady state (origin) by considered inputs.

To illustrate the fault tolerance evaluation approach proposed in this paper, two fault scenarios are also considered affecting the angular speed bounds:

Fault scenario 1: $-0.2 \leqslant \omega(t) \leqslant 0.2 \qquad 2 < t < 4$
Fault scenario 2: $-0.01 \leqslant \omega(t) \leqslant 0.01 \qquad 8 < t < 12$

Simulation results of the robot with the considered NMPC controller in these fault scenarios are shown in Figs. 8–10. It is assumed that after the fault the NMPC model constraints are adapted as proposed in Section 3.2, assuming the existence of a FDI module that provides such information.

It can be seen in Fig. 8 that after the first fault disappears (in $t = 4$ s), the controller is able to force the robot to follow the desired trajectory in about 0.4 s (4 sample times) and after the second fault (in $t = 12$ s), about 2 s (20 sample times). It is clear from these simulations that the second fault scenario is more severe than the first one as it needs more control effort (Fig. 9) and it goes far away from desired trajectory (Fig. 10). These results could be anticipated with the approach proposed in this paper. To show that, capture basin in each fault scenario is computed. For capture basin derivation, in first fault scenario, it is considered that the system starts in the zonotope:

$$E_0 = \begin{bmatrix} 0.52 \\ -0.19 \\ -0.59 \end{bmatrix} + \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix} \beta^3$$

The initial zonotope E_0 is chosen according to where the system states are when the fault disappears. To analyze fault tolerance property of the system in a given faulty case, this initial set can be chosen arbitrarily according to the system situation. Capture basin after three iterations is shown in Fig. 11. The iterations are recurred until the computed capture basin includes the target (origin). This means that the system (23) could converge to zero after three sampling times and there is no tracking error. It does not mean that the implemented controller forces the system to go to the origin in three sampling times, it means that the system can not reach the origin in less than three sampling times.

In the second fault scenario, according to the system states when the fault disappears, we consider initial zonotope as:

$$E_0 = \begin{bmatrix} 3.45 \\ 0.03 \\ -1.86 \end{bmatrix} + \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix} \beta^3$$

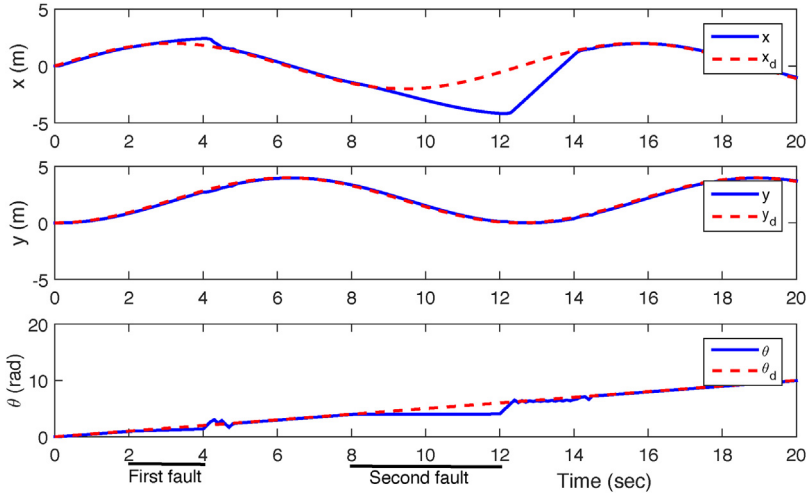

Fig. 8. A system simulation – system states.

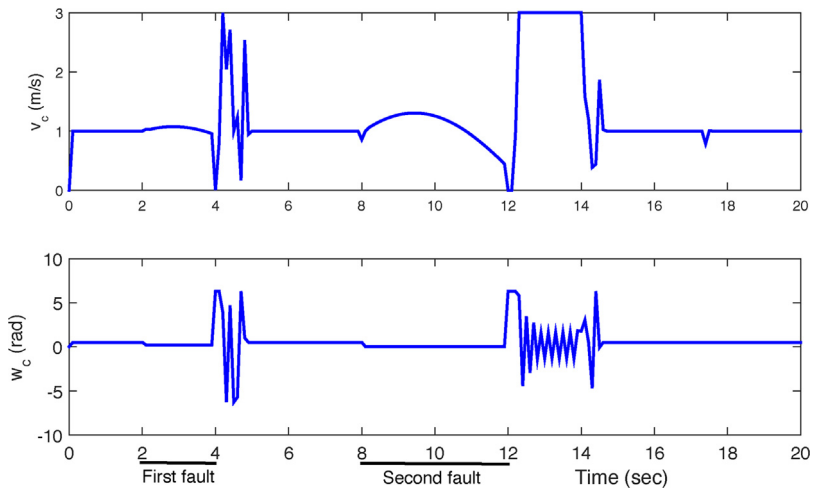

Fig. 9. A system simulation – control signals.

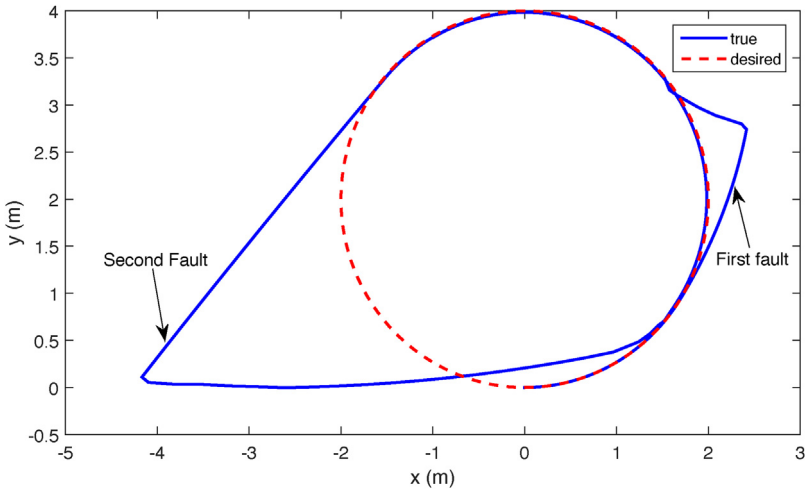

Fig. 10. A system simulation – in x–y plane.

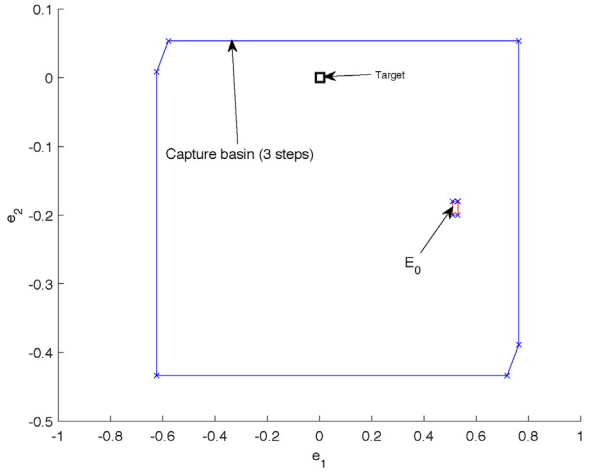

Fig. 11. Capture basin for the first fault scenario.

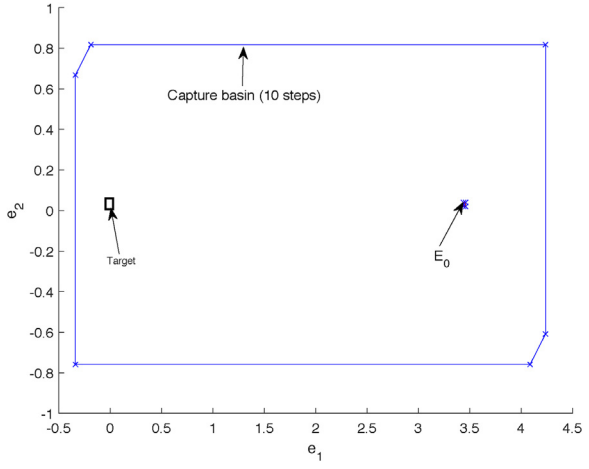

Fig. 12. Capture basin for the second fault scenario.

Computed capture basin using Algorithm 2 is shown in Fig. 12. Here, similar to the first fault scenario, the algorithm is iterated until final set includes the target (origin) requiring ten iterations. It means that the system (20) can not reach the desired trajectory with zero error after fault appears in less than ten sampling times (1 s). Comparing with Fig. 11, it is clear that because of severity of the second fault scenario, it needs more time to go to desired trajectory (zero tracking error).

In the simulation results shown in Fig. 8, the robot needs near twenty sampling times to go back to the desired trajectory. From

the capture basin analysis, it was already proved that the robot can not reach the reference with zero error in less that ten sampling times. Let us consider that the control specifications require that the robot must reach the desired trajectory in less than 5 sampling times (0.5 s). Then, from the capture basin analysis, it is clear that in the first fault scenario, the system is fault tolerant but this is not the second fault scenario. These results show that the proposed approach is able to anticipate the performance of the controller after the fault allowing to decide if the performance is acceptable or not, that is if it is not fault tolerant with respect to the considered faults and control objectives.

6. Conclusions

In this paper, a fault tolerance evaluation approach for nonlinear MPC using viability theory has been proposed and developed. First, the concepts of viability kernel and capture basin have been revisited, proposing algorithms based on zonotopic representation and arithmetic to compute them. The detailed analysis of the viability concepts applied to the fault tolerance evaluation of a NMPC strategy proved that assess the admissibility of any specification that can be expressed as a capture basin. Then, the fault strategies that can be applied to NMPC are presented. Using the viability sets and the considered fault tolerance strategies, a fault tolerance evaluation approach is proposed and developed formalising it in terms of computational algorithm. Finally, fault scenarios simulated in a part of a pasteurization process and a mobile robot have been used in order to show the effectiveness of the proposed approach. As future work, the research presented in this paper will be extended to consider stability issues by means of the invariant kernel concept as well as to consider uncertainty in the MPC using robust approaches.

Acknowledgments

This work has been funded by the Spanish Ministry of Economy and Competitiveness (MINECO) and FEDER through the projects ECOCIS (Ref. DPDEOCS (ref. DPI2016-76493) and SCAV (ref. DPI2017-88403-R), the FPI grant (ref. BES-2014-068319), the María de Maeztu Seal of Excellence to IRI (ref. MDM-2016-0656), and by AGAUR of Generalitat de Catalunya through the Advanced Control Systems (SAC) group grant (ref. 2017-SGR-482).

Appendix. Background on zonotopes

Zonotopes represent a particular class of polytopes which exhibit symmetry with respect to their center. In realistic situations, often the constraints that are given in polytopic form have enough symmetry to be described as zonotopic sets. Even when this is not the case, zonotopic approximations may be constructed. For polytopic sets, [26] gives the tightest approximations in fixed directions and [27] discusses an iterative algorithm. In [28], it is proven that any Euclidean ball can be approximated arbitrarily close, in the sense of the Hausdorff distance, by a zonotope.

Definition 6.1 *(Minkowski sum).* The Minkowski sum of two sets X and Y is defined by

$$X \oplus Y = \left\{ x + y : x \in X, y \in Y \right\}$$

Definition 6.2 *(Zonotope).* Given a vector $c \in \mathbb{R}^n$ (center) and a matrix $H \in \mathbb{R}^{n \times m}$ (segment matrix), the set represented as:

$$Z = c \oplus H\beta^m = \{c + Hz : z \in \beta^m\}$$

is called a zonotope of order m and corresponds to the Minkowski sum of the segments defined by the columns of matrix H. In this expression, β^m is a unitary box composed by m unitary intervals.

Definition 6.3 *(Interval hull).* The interval hull $\square X$ of a closed set X is the smallest interval box that contains X.

Given a zonotope $X = \pi \oplus H\beta^{m_z}$, its interval hull can be easily computed by

$$\square X = \left\{ x \quad \forall i = 1, \ldots, n : \left| x_i - \pi_i \right| \leqslant \left\| H_i \right\|_1 \right\}$$

where x_i and π_i are the ith components of x and π, respectively and H_i is the ith row of H.

Property 6.1. Zonotope linear image transformation [29]: *Consider a zonotope represented by $X = \pi \oplus H\beta^{m_z}$, where $\pi \in \mathbb{R}^{n_z}$ is a vector and $H \in \mathbb{R}^{n_z \times m_z}$ is a matrix. The image of the zonotope X through a linear transformation $T \in \mathbb{R}^{n_z \times n_z}$ is a zonotope Y defined by*

$$Y = q_z \oplus N_z \beta^{m_z}$$

where $q_z = T\pi$ and $N_z = TH$.

Property 6.2. Zonotope inclusion [26]: *Consider a family of zonotopes represented by $X = \pi \oplus M\beta^{m_z}$, where $\pi \in \mathbb{R}^{n_z}$ is a real vector and $M \in \mathbb{R}^{n_z \times m_z}$ is an interval matrix. A zonotope inclusion $\Diamond(X)$ is defined by*

$$\Diamond(X) = \pi \oplus \left[mid(M) \quad G \right] \begin{bmatrix} \beta^{m_z} \\ \beta^{n_z} \end{bmatrix} = \pi \oplus J\beta^{n_z + m_z}$$

where $G \in \mathbb{R}^{n_z \times m_z}$ is a diagonal matrix that satisfies:

$$G_{ii} = \sum_{j=1}^{m_z} \left(diam(M_{ij})/2 \right), \quad i = 1, 2, \ldots, n$$

where 'mid' denotes the center and 'diam' the diameter of the interval according to [30]. Under this definition $X \subseteq \Diamond(X)$.

Property 6.3 (Intersection). *Given two zonotopes $Z_1 = p_1 \oplus H_1\beta^{r_1}$ and $Z_2 = p_2 \oplus H_2\beta^{r_2}$ and matrix E, let us define*

$$\hat{p}(E) = Ep_1 + (I - E)p_2$$
$$\hat{H}(E) = \left[EH_1 \quad (I - E)H_2 \right]$$

then,

$$Z_1 \cap Z_2 \subseteq \hat{Z}(E)$$
$$\hat{Z}(E) = \hat{p}(E) \oplus \hat{H}(E)\beta^{r_1 + r_2} \tag{27}$$

Testing whether the intersection of two convex sets is empty or not can be done by collision detection algorithms. In particular, testing the emptiness of the intersection between two sets is equivalent to test the membership of the origin in the Minkowski difference of the two sets [31]. Some collision detection algorithms can thus be used to test whether a point belongs to a given set. The GJK (Gilbert–Johnson–Keerthi) algorithm is the robust and fast collision detection algorithm that is introduced in [32].

As previously mentioned, detecting the collision between Z_1 and Z_2 can be reformulated as testing the inclusion of origin in the Minkowski difference between Z_1 and Z_2:

$$0 \in Z_d = Z_1 \oplus (-Z_2) = c_d \oplus H_d\beta^{m_d}$$

The proposed solution is to find a separation vector w whose direction aims at proving the non-membership of 0 in the zonotope. Indeed,

$$\begin{aligned} 0 \notin Z_d &\Leftrightarrow \exists w, 0 \notin w^T Z_d \\ &\Leftrightarrow \exists w, \left|w^T c_d\right| > \left\|w^T H_d\right\|_1 \end{aligned}$$

Therefore, the collision detection can be reformulated as the maximization of the criterion J:

$$\begin{aligned} w^* &= \max_w J(w) \\ J(w) &= \frac{\left|w^T c_d\right|}{\left\|w^T H_d\right\|_1} \end{aligned} \tag{28}$$

If $\left|w^{*T} c_d\right| > \left\|w^{*T} H_d\right\|_1$ then $0 \notin Z_d$. The problem is directly addressed by solving (28) by means of an iterative algorithm. In [31], a sub-optimal solution based on the optimization of a criterion involving the Euclidean norm instead of the 1-norm is proposed:

$$J_{subopt}(w) = \frac{\left\|w^T c_d\right\|_2^2}{\left\|w^T H_d\right\|_2^2}$$

References

[1] Y. Zhang, J. Jiang, Bibliographical review on reconfigurable fault-tolerant control systems, Annu. Rev. Control 32 (2) (2008) 229–252.
[2] N.E. Wu, K. Zhou, G. Salomon, Control reconfigurability of linear time-invariant systems, Automatica 36 (11) (2000) 1767–1771.
[3] F. Guenab, D. Theilliol, P. Weber, Y. Zhang, D. Sauter, Fault tolerant control system design: a reconfiguration strategy based on reliability analysis under dynamic behavior constraints, IFAC Proc. Vol. 39 (13) (2006) 1312–1317.
[4] J.-P. Aubin, A.M. Bayen, P. Saint-Pierre, Viability Theory: New Directions, Springer Science & Business Media, 2011.
[5] J.N. Maidens, S. Kaynama, I.M. Mitchell, M.M. Oishi, G.A. Dumont, Lagrangian methods for approximating the viability kernel in high-dimensional systems, Automatica 49 (7) (2013) 2017–2029.
[6] H.-W. Lorenz, J.-P. Aubin, Dynamic Economic Theory: A Viability Approach, Springer-Verlag, Berlin Heidelberg, 1999.
[7] G. Deffuant, N. Gilbert, Viability and Resilience of Complex Systems: Concepts, Methods and Case Studies From Ecology and Society, Springer Science & Business Media, 2011.
[8] F. Blanchini, S. Miani, Set-Theoretic Methods in Control, Springer, 2008.
[9] M.G. Zarch, V. Puig, J. Poshtan, M.A. Shoorehdeli, Fault detection and isolation using viability theory and interval observers, Int. J. Syst. Sci. 49 (7) (2018) 1445–1462.
[10] M.M. Seron, J.A. De Doná, J.H. Richter, Invariant-set-based fault diagnosis in lure systems, Int. J. Robust Nonlinear Control 24 (16) (2014) 2405–2422.
[11] C. Ocampo-Martinez, P. Guerra, V. Puig, J. Quevedo, Actuator fault-tolerance evaluation of linear constrained model predictive control using zonotope-based set computations, Proc. Inst. Mech. Eng. I: J. Syst. Control Eng. 221 (6) (2007) 915–926.
[12] R. Tóth, Modeling and Identification of Linear Parameter-Varying Systems, vol. 403, Springer, 2010.
[13] P. Gáspár, Z. Szabo, J. Bokor, A grey-box identification of an LPV vehicle model for observer-based side slip angle estimation, in: American Control Conference, 2007, ACC'07, IEEE, 2007, pp. 2961–2966.
[14] J. Maciejowski, Predictive Control with Constraints, Prentice Hall, Great Britain, 2002.
[15] R.J. Spiteri, D.K. Pai, U.M. Ascher, Programming and control of robots by means of differential algebraic inequalities, IEEE Trans. Robot. Autom. 16 (2) (2000) 135–145.
[16] P. Saint-Pierre, Viable capture basin for studying differential and hybrid games: application to finance, Int. Game Theory Rev. 6 (01) (2004) 109–136.
[17] I.M. Mitchell, Comparing forward and backward reachability as tools for safety analysis, in: International Workshop on Hybrid Systems: Computation and Control, Springer, 2007, pp. 428–443.
[18] S. Montes de Oca, V. Puig, J. Blesa, Robust fault detection based on adaptive threshold generation using interval LPV observers, Int. J. Adapt. Control Signal Process. 26 (3) (2012) 258–283.
[19] C. Combastel, A state bounding observer based on zonotopes, European Control Conference (2003).
[20] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, Diagnosis and Fault-Tolerant Control, Springer, 2006.
[21] S. Ding, Model-Based Fault Diagnosis Techniques Design Schemes, Algorithms and Tools, Springer, 2013.
[22] J. Ibarrola, J. Sandoval, M. Garcıa-Sanz, M. Pinzolas, Predictive control of a high temperature-short time pasteurisation process, Control Eng. Pract. 10 (7) (2002) 713–725.
[23] E. Alcala, V. Puig, J. Quevedo, T. Escobet, R. Comasolivas, Autonomous vehicle control using a kinematic Lyapunov-based technique with LQR-LMI tuning, Control Eng. Pract. 73 (2018) 1–12.
[24] E. Alcalá Baselga, Modelling, planning and nonlinear control techniques for autonomous vehicles (Master's thesis), Universitat Politècnica de Catalunya, 2016.
[25] J.M. Snider, Automatic steering methods for autonomous automobile path tracking, Tech. Rep. CMU-RITR-09-08, Robotics Institute, Pittsburgh, PA, 2009.
[26] T. Alamo, J.M. Bravo, E.F. Camacho, Guaranteed state estimation by zonotopes, Automatica 41 (6) (2005) 1035–1043.
[27] T. Dang, Approximate reachability computation for polynomial systems, in: International Workshop on Hybrid Systems: Computation and Control, Springer, 2006, pp. 138–152.
[28] J. Linhart, Approximation of a ball by zonotopes using uniform distribution on the sphere, Arch. Math. 53 (1) (1989) 82–86.
[29] C. Combastel, A state bounding observer for uncertain non-linear continuous-time systems based on zonotopes, in: 44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference, CDC-ECC'05, IEEE, 2005, pp. 7228–7234.
[30] V.T.H. Le, C. Stoica, T. Alamo, E.F. Camacho, D. Dumur, Zonotopes: From Guaranteed State-Estimation to Control, John Wiley & Sons, 2013.
[31] A. Lalami, C. Combastel, Generation of set membership tests for fault diagnosis and evaluation of their worst case sensitivity, IFAC Proc. Vol. 39 (13) (2006) 569–574.
[32] G.V.d. Bergen, A fast and robust GJK implementation for collision detection of convex objects, J. Graph. tools 4 (2) (1999) 7–25.