The International Journal of Advanced Manufacturing Technology (2019) 104:4803–4812
https://doi.org/10.1007/s00170-019-04335-4

ORIGINAL ARTICLE

Fault-tolerant model predictive control of a de-manufacturing plant

A. Cataldo[1] · M. Morescalchi[2] · R. Scattolini[2]

Received: 20 December 2018 / Accepted: 23 August 2019 / Published online: 5 September 2019

Abstract
This paper describes the efficient implementation of a model predictive control (MPC) algorithm for the management of the pallets loaded on the transportation line of a de-manufacturing plant. In order to reduce the computational burden required for the solution of the online optimization problem, and make it compatible with industrial applications, different control and prediction horizons are used. In this way, the complexity of the optimization problem is reduced without significantly affecting the performance of the plant. In addition, a detailed inspection of the transportation line configuration, and the parallelization of the optimization and implementation tasks, allows one to obtain computational times fully comparable to those of simple heuristic rules but with significant improvements in terms of the plant throughput. In the second part of the paper, a fault detection procedure is developed for the identification and isolation of sensors' and actuators' faults. Then, the basic MPC algorithm is modified to obtain a control scheme tolerant to instrumentation faults. Both simulation and experimental results are reported and discussed to show the control performance and the practical applicability of the proposed approach.

Keywords Manufacturing systems · Model-based control · Fault-tolerant control · Optimal control · Mixed integer linear programming

1 Introduction

The tumultuous evolution of information technology (IT) provides growing opportunities in terms of sensing, computing power, transmission of information, and networking capabilities. In the manufacturing industry, this revolution, going under the name Industry 4.0, requires new paradigms for the design and implementation of automation systems and for the development of "smart machines" and "smart factories," characterized by distributed sensing, computing, control, self-diagnosis, and self-organizing properties to enhance production flexibility, interoperability, and autonomy. Indeed, according to recent studies on the future of manufacturing production systems [10], the enhancement of productivity will be mandatory in the near future to maintain high levels of competitiveness. However, the possibilities offered by the new technological scenarios in terms of flexibility have not been fully exploited yet, and innovative fault detection and control algorithms are required to improve efficiency and fault tolerance of manufacturing systems.

In this context, model predictive control, or MPC (see [4, 19, 25]), represents one of the most promising and effective control design methods in view of its ability to cope with multivariable problems and multiple objectives. In MPC, at any time instant, the sequence of future control variables over a given prediction horizon is computed by minimizing a cost function, typically weighting the control variables and the discrepancy between the future evolution of the system with respect to its desired profile. In addition, constraints on the plant and control variables can be considered to cope with safety requirements and actuators' limitations. Once the optimal future control sequence has been computed, and according to the so-called receding horizon (RH) strategy,

✉ A. Cataldo
andrea.cataldo@stiima.cnr.it

M. Morescalchi
morescalchi.marco@gmail.com

R. Scattolini
riccardo.scattolini@polimi.it

[1] Institute of Intelligent Industrial Technology and Systems for Advanced Manufacturing, National Research Council, Milan, Italy

[2] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133, Milan, Italy

only its first value is applied and the overall optimization problem is solved again at the new time instant. Although MPC is nowadays a standard in many process industries, such as chemical and petrochemical, its diffusion in manufacturing is still limited (see the contributions [1, 7, 9, 15, 23, 29, 31, 32]). This is due to the intrinsic discrete-event nature of manufacturing plants, usually described by integer (or even Boolean) variables, so that large mixed integer linear programming (MILP) or quadratic programming (MIQP) optimization problems must be solved online. For this reason, the common approach adopted to design the control systems of manufacturing plants relies on the definition of heuristic rules, which do not require long computational times for their implementation.

The potentialities and limitations of MPC applied to a manufacturing plant have been recently analyzed in [5, 8, 9], where MPC has been used to control the movement of the pallets along the transportation line of the de-manufacturing plant shown in Fig. 1. The results reported in these papers, referred both to simulation experiments and to real-time operations, clearly show that the control system designed with MPC provides better results, in terms of throughput, with respect to a set of heuristic rules specifically tailored on the topological characteristics of the system. This is due to the ability of MPC to exploit the inherent flexibility of the plant provided by the multiple paths available to the pallets on the transportation line. On the other hand, the computational issues tend to grow dramatically with the number of pallets on the transport line and with the length of the adopted prediction horizon. For the practical possibility to consider MPC as a viable design approach for control of manufacturing plants, these limitations must be overcome and the potentialities of MPC must be fully exploited.

Among the strengths of MPC, there is also the possibility to easily account for structural changes due to sensors' and actuators' faults, and to design active fault-tolerant (FT) control strategies, as discussed in [11, 18]. This property has been widely explored in the process industry (see for instance the application examples reported in [20, 26, 35]). More in general, many results are nowadays available for the design of FT control algorithms for continuous and discrete time systems (see the books [3, 34]), while only few methods have been proposed for discrete-event systems, such as those typically considered in manufacturing applications. In this context, an interesting survey is reported in [12], an approach for systems modeled as automata has been described in [22], while the system representation in terms of Petri nets has been considered in [28]. It is then interesting to exploit the possibility of using MPC for the design of FT control algorithms for discrete-event systems. To this regard, it is worth recalling the method described in [30] and based on a description of the system with max-plus algebra.

In view of the above reasons, and with reference to the de-manufacturing plant of Fig. 1, the aim of this paper is twofold:

(1) The first goal is to describe an efficient implementation of MPC useful to reduce the computational load. The main idea is to resort to the so-called *control horizon*, widely used in classical MPC applications. Specifically, the MPC algorithm is implemented by optimizing the future control variables over a control horizon shorter than the prediction horizon and assuming that heuristic rules are used from the end of the control horizon onwards. This strategy, previously partially described in [6], allows to significantly reduce the number of optimization variables to be computed through the solution of a MILP problem, while still considering long prediction horizons, often required to achieve satisfactory performance. Notably, since the RH approach is adopted, the heuristic rules are never applied in practice, and at any time instant, the control action is given by the solution of the stated optimization problem.

(2) The second goal is to fully take advantage of the flexibility provided by MPC to design a control system tolerant to the maximum possible extent and to actuators' and sensors' faults. A simple fault detection procedure is developed, and the reconfiguration of the control system is obtained by redefining, in case of faults, the constraints on the control variables in the MPC problem.

The paper is organized as follows. The considered plant and the overall control architecture are described in Section 2. In Section 3, the model of the plant is briefly summarized (the reader is referred to [5] for a detailed description of the model), and the proposed MPC algorithm with control

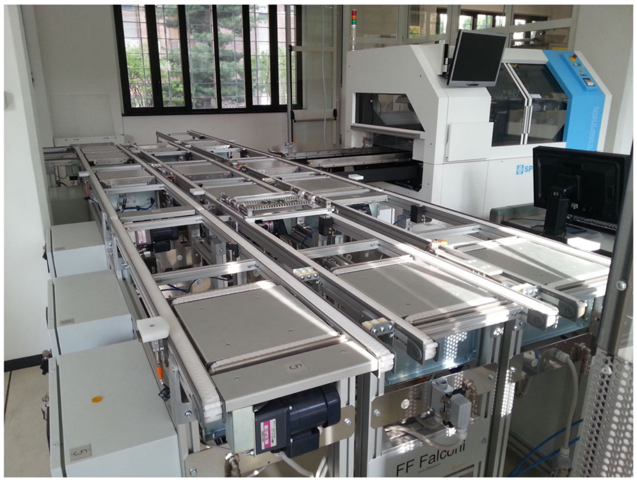
Fig. 1 The de-manufacturing plant

horizon is presented. The results of simulation experiments are discussed to compare, in terms of computational times and machined pallets, the performance of the proposed MPC algorithm to those of the standard MPC developed in [9]. Section 4 first describes the algorithm developed for the identification and isolation of faults of actuators and sensors. Then, in case of faults, it is described how to modify the MPC algorithm to guarantee fault tolerance, and some experimental results are discussed. Finally, Section 5 closes the paper with some conclusions and possible future developments.

2 The de-manufacturing plant and the hierarchical control structure

2.1 The plant

The de-manufacturing plant has been designed for testing, repairing, or discharging electronic boards. Its main components are:

- A transportation line, made by fifteen transport modules T_i $(i = 1, \ldots, 15)$ connected together according to a specific meshed configuration. On a transport module, up to three pallets can lay in three adjacent positions, called buffer zones BZ_j $(j = 1, 2, 3)$;
- Machine M_1: a load/unload robot cell that either loads the electronic board on a pallet, which is then placed on the adjacent transport module, or unloads it from the pallet;
- Machine M_2: a testing machine that tests the board and identifies its failure mode;
- Machine M_3: a reworking machine that machines the board to be repaired;
- Machine M_4: a discharge machine that discharges the nonrepairable boards to be destroyed from the pallet.

A sketch of the plant is shown in Fig. 2, while a detailed description of its devices is reported in [5, 8, 9]. The sequence of operations to be performed on each board is as follows:

1. The board is loaded on the pallet by M_1;
2. The transport line moves the pallet to M_2 where the board is tested and its failure mode is identified;
3. The pallet with the board is moved to M_3 where the failure is repaired, if possible;
4. The pallet is moved back to M_2 and the test is repeated. If the board is properly working, it is sent back to M_1 where it is unloaded from the pallet and stored in the warehouse; otherwise, it is sent to M_4 where it is discharged from the pallet and destroyed. Then, the pallet is ready to load a new board to be tested.

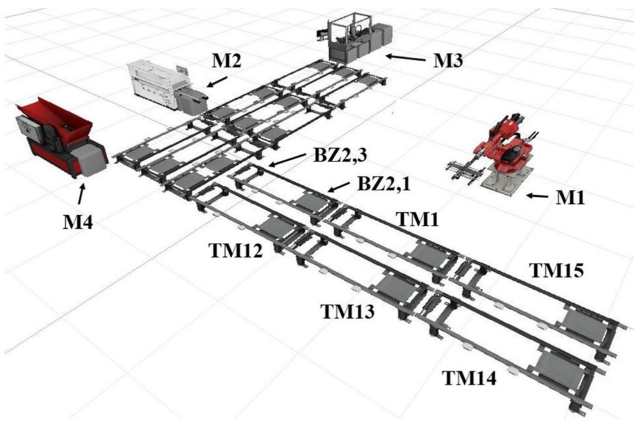

Fig. 2 Structure of the de-manufacturing plant

2.2 The control structure

The fault-tolerant control system has been designed according to the hierarchical architecture shown in Fig. 3. The higher layer receives from the intermediate one the state of the system, i.e., the position of the pallets on the transport line, the target machines for the electronic boards loaded on the pallets, and the state of actuators and sensors (safe or fault). Then, based on the model of the system, it computes with MPC the optimal control sequence over the prediction horizon and sends it to the lower layers. If a fault of the instrumentation has been detected by the intermediate layer, the MPC algorithm is reconfigured to compute, if possible, a feasible solution; otherwise, the system is stopped.

The intermediate layer transforms the commands provided by the higher layer into sequences to be sent to the programmable logic controllers (PLCs) at the lower layer. In addition, based on the measurements and information provided by the PLCs, it runs the fault detection algorithm to monitor the status of actuators and sensors.

Finally, the PLCs at the lower layer produce the commands to the motors of the transport modules, acquire the sensors' signals, and transmit these information upwards.

The software environments used at the higher layer are MATLAB, YALMIP, and CPLEX (see [13, 16]), while the intermediate layer has been implemented with ISaGRAPH (see [14]).

3 MPC of the de-manufacturing plant

An abstract model has first been developed by describing the transport line and the machines as a directed graph (see Fig. 4), where the nodes (filled with the gray color) are associated with the buffer zones (also identified with the gray color in the transport line structure showed in Fig. 2)

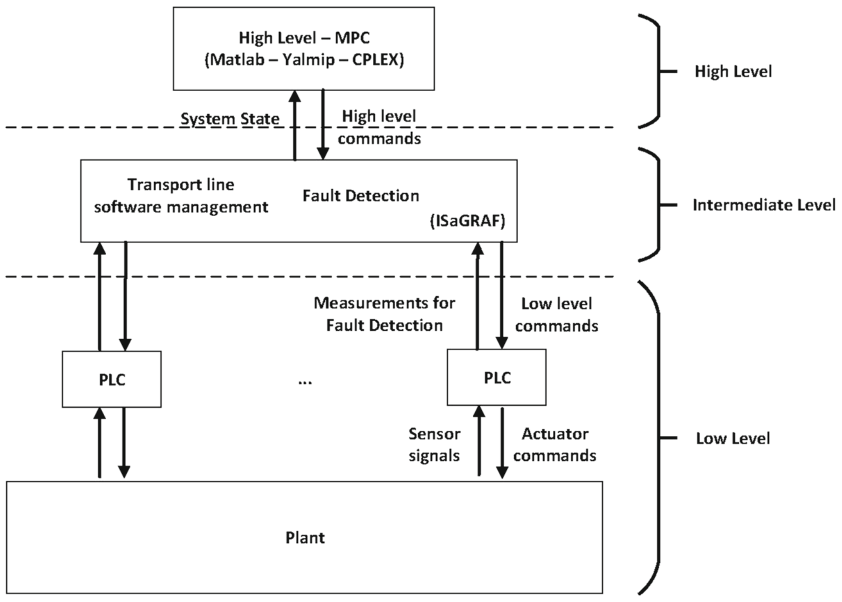

Fig. 3 Hierarchical control architecture

or with the machines, while the oriented arcs represent the admissible paths for the pallets. The commands $u_{i,j}$, $i, j = 1, \ldots, 35$, associated with the arcs, allow moving the pallet from node i to node j. Starting from the graph representation of Fig. 4, in [9], a set of heuristic rules for the movement of the pallets along the transport line have been defined. These rules are mainly based on the predefinition both of specific paths connecting the machines and traffic lights to avoid collisions. However, their use prevents one from fully taking advantage of the available degrees of freedom of the plant. For this reason, an optimization algorithm based on MPC has been developed.

3.1 MPC algorithm

For the design of the MPC law, an analytic model of the pallet movements, machines, and logical constraints expressed with the rules of propositional calculus, has been derived in [9]. For the sake of conciseness, this model is not reported here; it suffices to say that it takes the form of a discrete-event MLD (mixed logical dynamical) model (see [2, 17, 24, 33]).

In this model, the state $x \in \mathbb{I}^{82}$ is a vector of integer states, including the position and the target of the pallets on the transport line, the distance from their own target, and

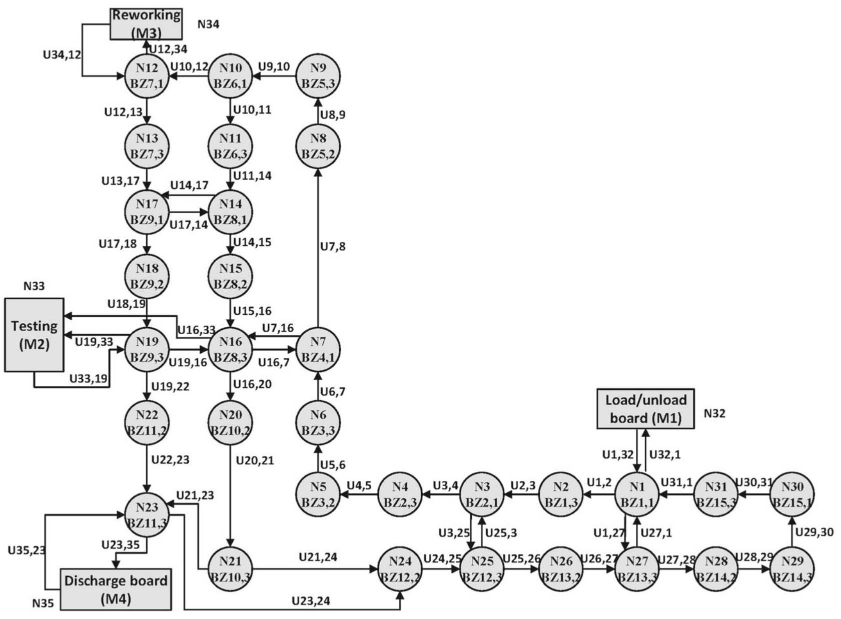

Fig. 4 Directed graph representation of the plant

the working state of the machines; the input $u \in \mathbb{I}^{51}$ is a vector of Boolean variables $u_{i,j}$; and the output $y \in \mathbb{I}^{35}$ is a vector of integer elements, coinciding with the subset of states describing the distance of the pallet from its target machine. In addition, two vectors z and δ of continuous and integer auxiliary variables are required to implement the MLD system. Note that the dimension of the model is quite large and represents a very significant benchmark for the implementation of MPC.

Now let the positive integers N_{RH} and N_{c}, $0 < N_{\mathrm{c}} \leq N_{\mathrm{RH}}$, be the prediction and control horizons, respectively, define by $U(k) = [u(k)' \dots u(k + N_{\mathrm{c}} - 1)']'$, the vector of future control moves to be optimized, and assume that from the end of the control horizon onwards, the heuristic rules are applied. Note that the use of a control horizon $N_{\mathrm{c}} \ll N_{\mathrm{RH}}$ allows to significantly reduce the number of the optimization variables so that also large MILP optimization problems can be considered and solved with reduced computational times (see the results reported in Section 3.2).

In view of the above considerations, the optimization problem to be solved at any k is (see [9])

$$\min_{U(k)} \sum_{h=1}^{N_{\mathrm{RH}}} \bar{J}(k+h) \tag{1}$$

under specific constraints and with

$$\begin{aligned} &\bar{J}(k+h) = \\ &= \underbrace{\sum_{i=1}^{35} \gamma_i(\Gamma_i(k+h))}_{(a)} + \underbrace{\sum_{i=32}^{35} q_{xi} x_{i3}(k+h)}_{(b)} + \\ &+ \underbrace{\sum_{i=1}^{31} q_{\eta_i} \eta_i(k+h)}_{(c)} + \underbrace{\sum_{(i,j)\in I_u} q_{ui,j} u_{i,j}(k+h-1)}_{(d)} + \\ &+ \underbrace{\sum_{(m,r,i,j)\in\Psi} \lambda_{m,r}\sigma_m(k+h-1)u_{i,j}(k+h-1)}_{(e)} \end{aligned} \tag{2}$$

The terms in (2) have the following meaning:

(a) Penalizes the distance of the pallets from their target machines. Γ_i is the status of node i (with/without a pallet to be moved to a given target). $\gamma_i(\cdot)$ is a function representing the minimal distance from the node i to the target machine of the loaded pallet;
(b) Penalizes the permanence of manufactured pallets inside the machines. x_{i3} is the internal status of machine i (equal to 1 if the machine has completed its operations, and 0 otherwise);
(c) Penalizes the permanence of the pallets on the transport line. η_i is a counter of the permanence of a pallet in node i;
(d) Penalizes the control actions, strictly related to the energy spent by the actuators to move the pallets;
(e) Penalizes the permanence of the pallets in nodes adjacent to the machines, which could easily lead to deadlocks. Ψ is the set of the nodes involved; σ_m are Boolean variables.

The coefficients q_{xi}, $q_{\eta i}$, q_{ui}, are $\lambda_{m,r}$ are weights to be properly tuned. The prediction horizon N_{RH} must be selected large enough to avoid possible deadlocks due to conflicting paths of the pallets; moreover, it must be greater than the minimum number of steps required by the machines M_i to work the pallets.
In addition to using a control horizon shorter than the prediction one, the optimization problem (1), (2) can be further simplified in view of the following considerations. First, in some nodes, the feasible paths are uniquely defined (see for instance the ones defined by nodes N4–N7, N8–N10, and N28–N1 in Fig. 4). In these cases, the corresponding commands $u_{i,j}$ are not variables to be optimized and the pallets are moved forward until they reach a position where possible conflicts can arise.
Second, the computations at the lower (actuators' activation) and higher (MPC) layers of the control structure can be largely parallelized by noting that MPC can start computing a new optimal control sequence assuming that the previous one is used and the pallets' configuration is the one predicted at the previous time. Then, a check of consistency is performed and, if the predicted and measured configurations are equal, the new computed commands are sent to the actuators.

3.2 Simulation results

Two experiments have been carried out to compare the performance of the MPC with control horizon against those of the standard MPC, with $N_{\mathrm{c}} = N_{\mathrm{RH}}$, in terms of (average) computational times and number of machined pallets. In the cost function (2), the following weights have been used: $q_{ui,j} = 0.02$; $q_{\eta i} = 1$; $q_{xi} = 10,000$; $\lambda_{32,1} = \lambda_{32,2} = \lambda_{33,2} = \lambda_{33,3} = \lambda_{34,1} = \lambda_{35,1} = 10$; $\lambda_{33,1} = 4$ (see [9]).

Experiment 1 The number of pallets on the transport line at $k = 0$ and the prediction horizon N_{RH} have been varied, while the initial states and the targets of the pallets (whenever present) are:

Pallet 1	in N28	with target M_4;
Pallet 2	in N29	with target M_2;
Pallet 3	in N30	with target M_3;
Pallet 4	in N31	with no target;

Table 1 Average computational time (s) per optimization step with standard MPC (a) and with MPC and control horizon (b)

	Num. pallets	$N_{\mathrm{RH}} = 5$	$N_{\mathrm{RH}} = 6$	$N_{\mathrm{RH}} = 7$
(a)				
	3	0.32	0.87	2.62
	4	0.47	1.22	3.41
	5	0.59	1.29	6.70
	6	1.34	6.30	56.35
	7	4.17	> 100.00	> 100.00
(b)				
	3	0.28	0.63	1.45
	4	0.39	0.91	2.12
	5	0.43	1.12	4.46
	6	1.12	3.66	19.13
	7	2.54	> 100.00	> 100.00

Pallet 5	in N16	with target M_2;
Pallet 6	in N19	with target M_3;
Pallet 7	in N23	with no target.

The control horizon N_c has been set equal to 2, and 100 simulation steps have been considered.

The mean computational time of the optimization step has been computed for the two methods. Simulations have been stopped when the computational time exceeds 100 s[1]. Table 1 summarizes the results obtained.

As expected, the optimization time depends on N_{RH}, on the number and the position of the pallets on the transport line. Moreover, note that the paths followed by the pallets with the two control algorithms are different and, in the view of the previous considerations, the comparison between the two control techniques can only be made in terms of average computational time. To quantify the reduction of computational time due to the use of a short control horizon, the following index has been defined:

$$\epsilon_{\%} = \frac{\tau_M - \tau_H}{\tau_M} \cdot 100 \qquad (3)$$

where τ_M and τ_H are the average computational time of standard MPC and of MPC with control horizon, respectively. By evaluating $\epsilon_{\%}$ with the data reported in Table 1, the values summarized in Table 2 show that the reduction is in the range 13–66%.

Experiment 2 In this experiment, it has been set $N_{\mathrm{RH}} = 6$, $N_c = 2$, five pallets have been loaded on the transport line at $k = 0$ (in the same positions previously defined),

[1]Simulations run on a computer with Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10 GHz, 16.0 GB of installed RAM, system type 64-bit operating system, x64-based processor, Windows 8.1 Pro., MATLAB R17a, CPLEX R12.6 (settings: Parallelmode = 0, Threads = 0).

Table 2 $\epsilon_{\%}$ values

Num. pallets	$N_{\mathrm{RH}} = 5$	$N_{\mathrm{RH}} = 6$	$N_{\mathrm{RH}} = 7$
3	13.18	27.92	44.69
4	17.26	25.69	37.78
5	26.80	13.08	33.42
6	16.03	41.80	66.05
7	38.98	–	–

and 1000 simulation steps have been considered. The number of pallets worked by machines M_1–M_4 with the two approaches (standard MPC and MPC with control horizon) has been computed. The obtained results in terms of the number of pallets visiting the machines are reported in Fig. 5a, b, which clearly shows that the performance of the two algorithms are very similar. Thus, we may conclude that the suboptimality inherent to MPC with control horizon is negligible.

For completeness, also the number of pallets machined by the system controlled only with the adopted heuristic rules has been computed. The comparison with the number of pallets machined by MPC with control horizon is reported in Fig. 6a, b. It is evident that MPC with control horizon outperforms the heuristic rules, i.e., it allows to significantly increase the throughput of the system even with a short control horizon.

4 Fault-tolerant control

The design of a fault-tolerant control scheme is divided into two main steps: fault detection and control reconfiguration.

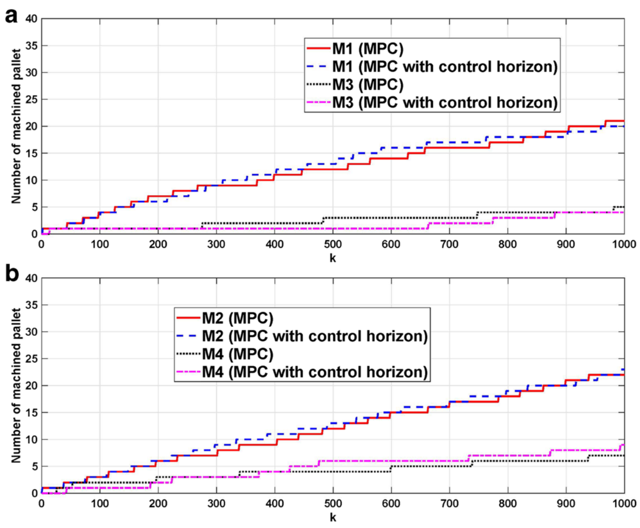

Fig. 5 Machined pallets with the standard MPC and the MPC with control horizon. M_1 and M_3 (**a**). M_2 and M_4 (**b**)

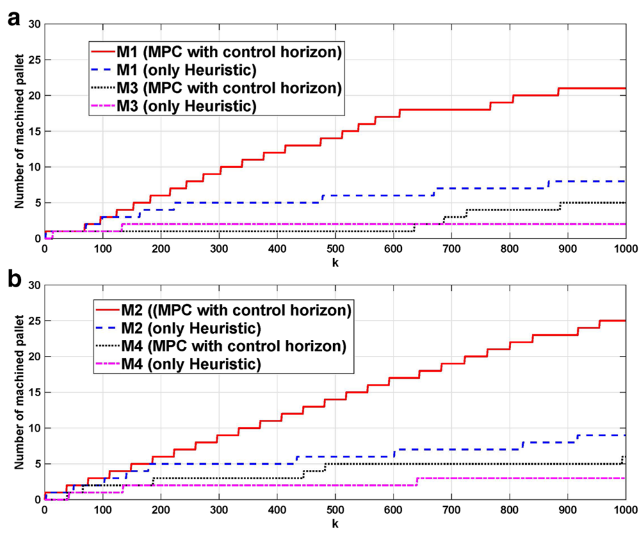

Fig. 6 Machined pallets with heuristic rules and MPC with control horizon. M_1 and M_3 (**a**). M_2 and M_4 (**b**)

In this section, these two phases are developed for instrumentation faults, i.e., faults of the actuators and sensors of the transportation line. For simplicity of exposition, the description of the corresponding algorithms have been developed under the following fundamental assumption:

Only one permanent fault can be present in the system at any time.

However, both for sensors' and actuators' faults, such assumption can be partially removed, as extensively discussed in [21].

4.1 Fault detection

The line supervisor placed at the intermediate layer of the control hierarchy and implementing the fault detection algorithm is based on the graph representation of Fig. 4. The status of this graph model, in terms of position of the pallets on the transport line, is used by MPC to compute the future control action.

The command u_{ij} to move a pallet from buffer zone i (node N_i) to the adjacent buffer zone j (node N_j) updates the status to the graph model and activates a motor governed by a PLC. When the movement is completed, a proximity sensor positioned in node j is used to acknowledge the pallet movement and to provide a command to the PLC to switch off the motor. The basic idea for the development of a simple and effective fault detection method is to implement in the PLC software a timer and to define a maximum actuation time within which the pallet movement must be completed. If, for any reason, the requested operation has not been completed in the maximum actuation time, a diagnostic flag is switched on, i.e., a Boolean variable w_{ij}, $i, j = 1, \ldots, 35$, is set to one, and the fault detection algorithm is run.

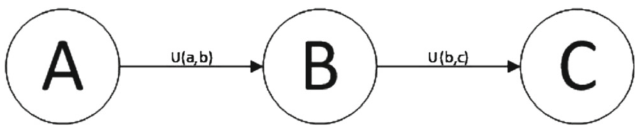

Fig. 7 Fault detection of a simple configuration

In order to describe the adopted fault detection logic, a simple example is now discussed. Consider the three adjacent nodes shown in Fig. 7, assume that at k the pallet is in node A and the activation command is switched on ($u_{\text{ab}} = 1$). Now assume that, due to a fault of the motor or of the proximity sensor, the maximum actuation time is exceeded and the motor is switched off by the PLC. This can be due to:

– A fault of the motor so that the pallet remains in A;
– A fault of the proximity sensor in B so that the pallet has reached B, but the proximity sensor has not detected its movement.

As a consequence, the Boolean variable w_{ab} is set to one. In any case, the line supervisor updates its status as if the operation was terminated correctly and transmits it to the MPC algorithm which, at $k+1$, activates the command to move the pallet from B to C ($u_{\text{bc}} = 1$). Two cases are in order:

– If the pallet was not in B at $k+1$, the maximum actuation time is again exceeded, the system diagnoses a fault of the motor, and sets $w_{\text{bc}} = 1$. In this case, it can be concluded that the motor associated to the command $u_{\text{ab}} = 1$ is in fault;
– If the pallet was in B at $k+1$, but it was not detected, the new command is completed within the maximum actuation time, the pallet reaches C, its presence is acknowledged, and the boolean variable w_{bc} remains null. In this case, it can be concluded that the proximity sensor in B is in fault.

In summary, the corresponding matrix of residuals is reported in Table 3, and since its rows, i.e., the signatures of the faults, are different, the fault detection procedure can be completed.

Table 3 Matrix of residuals

	w_{ab}	w_{bc}
Actuator fault	1	1
Sensor fault	1	0

This simple logic has been extended to all the nodes of the graph of Fig. 4 so that all the motors and proximity sensors have been monitored, and once a fault has been isolated, the information on its characteristics, together with the real status of the pallets on the transport line, is sent to the MPC algorithm for its reconfiguration.

4.2 Control reconfiguration

If a sensor fault has been identified by the fault detection logic, no action is required to recover the proper behavior of the MPC algorithm. In fact, MPC relies on the status of the transport line provided by the line supervisor running at the intermediate layer. As discussed in the previous Section 4.1, this status is updated by assuming that the control commands computed by MPC are always activated so that the estimated position of the pallets is correct and the line supervisor acts as a virtual sensor. In this case, the information provided by the fault detection algorithm is used only as a warning for the plant operators.
On the contrary, some actuators' faults are critical, since they prevent the movement of the pallets from a machine to another along the paths with the actuator in fault. As an example, consider the actuators needed to move the pallets from the load machine M_1 to the other machines and, among them, the actuators associated with the control commands u_{45}, u_{56}, and u_{67}: it is apparent that the fault of just one of them is such that the pallets loaded by M_1 cannot reach their target machines. Therefore, once one of these faults occurs, the plant must be stopped until the motor in fault is repaired.

In view of these considerations, a preliminary analysis has been conducted to identify the critical actuators. Specifically, the following algorithm has been implemented:

1. The fundamental links among the machines, which must be guaranteed to be available for the plant operation, have been identified. These six links are shown in Fig. 8. For each one of them, all the possible paths connecting the source machine to destination one have been computed with a best-first search approach (see [27]).
2. The actuator associated with the command u_{ij} is critical if it belongs to all the paths of at least one of the fundamental links.

In run time, if the fault of a critical actuator is detected, the plant must be stopped and the functionality of the actuator must be restored immediately. In fact, in this case, the proper functioning of the actuator is mandatory to guarantee that at least one save path exists in each one of the fundamental links depicted in Fig. 8. On the contrary, if the actuator in fault is a noncritical one, in view of the previous analysis, it is known that at least

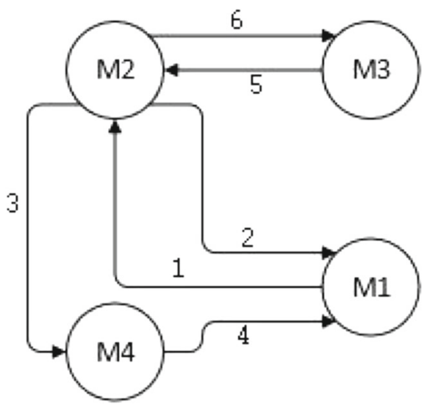

Fig. 8 Fundamental links among the machines

one safe path still exists for each one of the fundamental links. Therefore, the MPC algorithm can be used to find the optimal control sequence simply by adding to the optimization problem the constraint that the faulty actuator is unavailable. Assuming that the actuator in fault is the one moving the pallet from node i to node j, this corresponds to include the constraint $u_{ij} = 0$ in the problem formulation (this must be implemented by including the constraint $u_{ij} \leq 0$ in the MLD system). No other modifications are required and the algorithm previously described can be used as it is.

4.3 Simulation and experimental results

Many simulations and experiments on the real system have been performed to assess the performance of the control algorithm tolerant to faults. The results of one simulated test case are summarized in Fig. 9. Specifically, in the experiment, a pallet is initially placed in machine M_3 and must reach machine M_1. In the absence of faults, the path N_{12}, N_{13}, N_{17}, N_{18}, N_{19}, N_{22}, N_{23}, N_{21}, N_{24}, N_{25}, N_{26}, N_{27}, N_1 (yellow line in Fig. 9) is computed with MPC. When a fault of the noncritical actuator associated with the control action $u_{18,19}$ occurs, the modified path N_{18}, N_{15}, N_{16}, N_{20}, N_{21}, N_{24}, N_{25}, N_{26}, N_{27}, N_1 (red line) is recomputed by formulating the MPC problem with the additional constraint $u_{18,19} = 0$. Then, a sensor fault has been forced in node N_{21}. Since the pallet is detected in node N_{24} even if it is not detected in node N_{21}, then the low-level controller is able to manage this failure without any effect on the MPC control.

Concerning the experiments on the plant, both sensor and actuator faults have been forced on the real system, a video (Online Resource ESM_1) showing the behavior of the MPC algorithm and its fault tolerance properties is available at the Springer website.

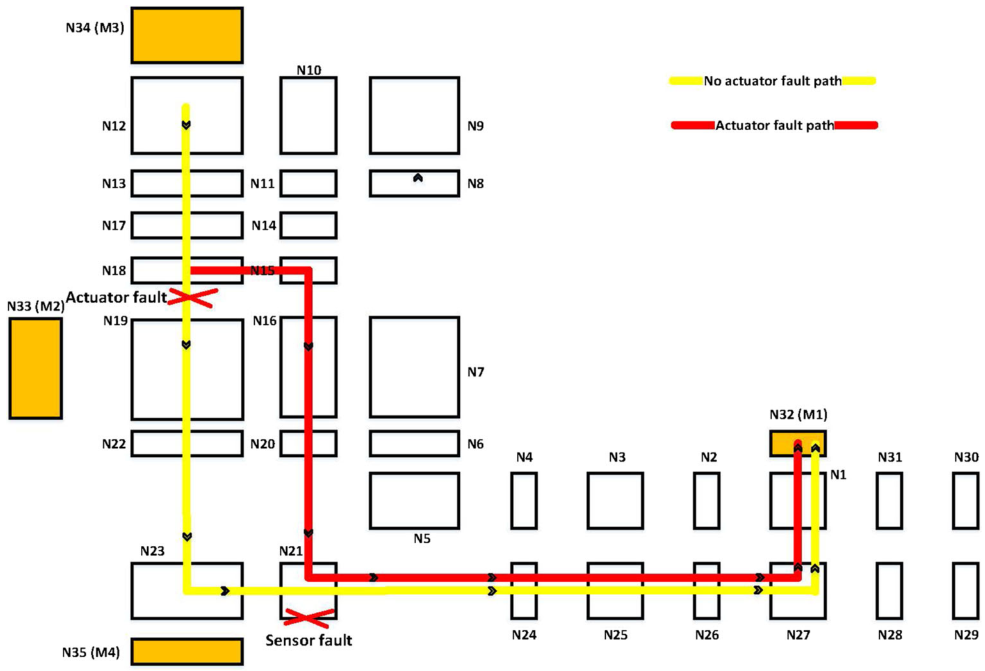

Fig. 9 Paths with fault-tolerant control

5 Conclusions

The paper has described the design of a fault-tolerant control scheme based on MPC for a de-manufacturing plant. The goal of this activity was to prove that advanced control methods can be used also in manufacturing systems, where their solution can be made difficult due to the mixed integer nature of the optimization problem to be solved online. In this regard, in the considered test case, not only the potentialities of MPC in terms of plant efficiency have been proven, but it has also been possible to use its flexibility for the design of a fault-tolerant control algorithm. This has been possible due to the fault detection method specifically developed for the plant. As a side effect of the work reported in the paper, the study provides clear indications on how to modify the plant layout to reduce the number of critical actuators and to enhance the fault tolerance properties of the system.

References

1. Alessandri A, Gaggero M, Tonelli F (2011) Min-max and predictive control for the management of distribution in supply chains. IEEE Trans Control Syst Technol 19(5):1075–1089
2. Bemporad A, Morari M (1999) Control of systems integrating logic, dynamics, and constraints. Automatica 35(3):407–427
3. Blanke M, Kinnaert M, Lunze J, Staroswiecki M, Schröder J. (2006) Diagnosis and fault-tolerant control, vol 2. Springer, Berlin
4. Camacho EF, Bordons C (2007) Model predictive control, 2nd edn. Springer, Berlin
5. Cataldo A (2017) Model predictive control in manufacturing plants. Ph.D. thesis, Politecnico di Milano
6. Cataldo A, Lanzarone E, Morescalchi M, Scattolini R (2018) Complexity reductionof model predictive control for a de-manufacturing plant. In: 16Th IFAC symposium on information control problems in manufacturing, Bergamo, pp 964–972
7. Cataldo A, Perizzato A, Scattolini R (2015) Production scheduling of parallel machines with model predictive control. Control Eng Pract 42:28–40
8. Cataldo A, Scattolini R (2014) Logic control design and discrete event simulation model implementation for a de-manufacturing plant. Automazione-plus. www.automazione-plus.it
9. Cataldo A, Scattolini R (2016) Dynamic pallet routing in a manufacturing transport line with model predictive control. IEEE Trans Control Syst Technol 24:1812–1819
10. European Commission (2013) Factory of the future. Multi-annual roadmap for the contractual PPP under Horizon 2020
11. Fernandez-Camacho E, Bordons-Alba C (1995) Model predictive control in the process industry. Springer
12. Fritz R, Zhang P (2018) Overview of fault-tolerant control methods for discrete event systems. IFAC-PapersOnLine 51(24):88–95
13. IBM (2016) IBM ILOG CPLEX Optimization Studio CPLEX Users Manual. www.ibm.com
14. Inc ITI (2009) ISAGRAF Getting Started
15. Lao L, Ellis M, Christofides PD (2014) Smart manufacturing: handling preventive actuator maintenance and economics using model predictive control. AIChE J 60:2179–2196
16. Löfberg Y. (2004) YALMIP : a toolbox for modeling and optimization in MATLAB. In: Proceedings of the CACSD Conference. Taipei, Taiwan. http://users.isy.liu.se/johanl/yalmip
17. Lucas C, Mitra G, Moody S (1992) Tools for reformulating logical forms into zero-one mixed integer programs (MIPS). Maths Technical Papers (Brunel University) (TR/03/92), 1–27
18. Maciejowski J (2000) Fault-tolerant aspects of MPC. In: IEE Seminar on Practical Experiences with Predictive Control
19. Maciejowski JM (2002) Predictive control with constraints, 1st edn. Pearson Prentice Hall, Upper Saddle River

20. Mhaskar P (2006) Robust model predictive control design for fault-tolerant control of process systems. Ind Eng Chem Res 45(25):8565–8574
21. Morescalchi M Fault tolerant model predictive control of a de-manufacturing plant. Msc thesis, Politecnico di Milano (2018). Supervisor R. Scattolini, co-supervisor A. Cataldo
22. Paoli A, Sartini M, Lafortune S (2011) Active fault tolerant control of discrete event systems using online diagnostics. Automatica 47(4):639–649
23. Perea-López E, Ydstie BE, Grossmann IE (2003) A model predictive control strategy for supply chain optimization. IEEE Computers and Chemical Engineering 27:1201–1218
24. Raman R, Grossmann IE (1991) Relation between MILP modelling and logical inference for chemical process synthesis. Comput Chem Eng 15(2):73–84
25. Rawlings JB, Mayne DQ (2009) Model predictive control: theory and design, 1st edn. Nob Hill Publishing, Madison
26. Robles D, Puig V, Ocampo-Martinez C, Garza-Castañón LE (2016) Reliable fault-tolerant model predictive control of drinking water transport networks. Control Eng Pract 55:197–211
27. Russell SJ, Norvig P (2016) Artificial intelligence: a modern approach Malaysia; Pearson Education Limited
28. Santos CCR, De Souza CHF, da Silva RM (2013) Modeling an active fault-tolerant control for manufacturing system
29. Schmid V, Doerner KF, Laporte G (2013) Rich routing problems arising in supply chain management. European Journal of Operational Research 224:435–448
30. Seybold L, Witczak M, Majdzik P, Stetter R (2015) Towards robust predictive fault–tolerant control for a battery assembly system. Int J Appl Math Comput Sci 25(4):849–862
31. Vargas-Villamir FD, Rivera DE (2001) A model predictive control approach for real-time optimization of reentrant manufacturing lines. Comput Ind 45(1):45–57
32. Wang W, Rivera D (2008) Model predictive control for tactical decision-making in semiconductor manufacturing supply chain management. IEEE Trans Control Syst Technol 16(5):841–855
33. Williams H (2013) Model building in mathematical programming, 5th edn. Wiley, Hoboken
34. Witczak M (2014) Fault diagnosis and fault-tolerant control strategies for non-linear systems, vol 266. Springer, Berlin
35. Yang X, Maciejowski JM (2012) Fault-tolerant model predictive control of a wind turbine benchmark. IFAC Proceedings 45(20):337–342