Elsevier Editorial System(tm) for ISA Transactions

Manuscript Draft

Manuscript Number: ISATRANS-D-19-01651R2

Title: Reliability Enhancement with Dependable Model Predictive Control

Article Type: Research article

Section/Category: Design

Keywords: Operational Technology system
Time-Sensitive Networking
Dependable Model Predictive Control
Replacement Controller

Corresponding Author: Dr. Anthony Tri Tran,

Corresponding Author's Institution: Defence Science and Technology Group

First Author: Anthony Tri Tran

Order of Authors: Anthony Tri Tran; Quang Ha; Robert Hunjet

Abstract: Operational Technology (OT) systems are merging towards a conjoint architecture with the advances in communication networks and emerging standards such as IEC/IEEE 60802 for industrial automation, automotive, power and energy and other areas. In this paper, we present a Dependable Control System (DepCS) with Model Predictive Control (MPC) algorithm that works in such architectures using multiple MPC controllers (of a feedback control loop) to enhance the operational reliability. We termed this as Dependable Model Predictive Control (DepMPC) system. The reliability enhancement of a DepMPC system is achievable thanks to the fault-tolerance of multiple MPCs and the tractable information flows with Time-Sensitive Networking (TSN). Here, our discussion was focused on the logical connectivity and not the hardware architecture. Numerical simulations are studied with three multi-variable plants that have control constraints. In this study, we introduced a Replacement Controller (RC) to improve the control performance of a DepMPC system. The combination of both the Replacement Controller and Dependable Model Predictive Control (RC-DepMPC) system proves a promising solution for actual implementation.

Suggested Reviewers:

Anthony Tri Tran, PhD
Research Scientist and Senior Lecturer

Defence Science and Technology Group,
Edinburgh, South Australia 5111
Phone: +61 8 7389 4836
Email: anthony.tran@dst.defence.gov.au

To: The Editor-in-Chief
Professor Ahmad B. Rad,
The ISA Transactions

Date: 29 June 2020,

Dear Professor Ahmad B. Rad:

SUB: Submission ISATRANS-D-19-01651R2

We firstly thank the Reviewers and the Associate Editor for recommending the revised version R1 to be accepted without any further comments. This is a very positive result for us as we have spent time and efforts to carefully revise the paper.

The similarities have now been eliminated or replaced with the new writing in Sections 1.5, 2.1 and 2.2. We would like to provide some detailed explanations as follows:

The idea in ICCAIS 2016 paper may sound similar to that in this paper, but it does not have any simulations and theoretical developments. In that conference paper, we associated the concept with the generic IoT and M2M network which was actually incorrect (according to some computer scientists). In this paper, the context of Time Sensitive Networking (TSN) is practical and correct for dependable control systems. The newly introduced standards for the Operational Technology (OT) systems with TSN that accommodate both PLC systems and DCS are becoming the new norm for the industry. The concept of dependable control systems with model predictive controllers in this paper perfectly suits the new standards. So we had completely changed the presentation in this paper using the TSN and OT systems defined in the new standards. Furthermore, this paper includes the theoretical development and comprehensive simulation studies while the conference paper does not. Sections 2.1 and 2.2 have been revised to eliminate the similarities in writing.

The literature review for Distributed MPC has some common sense statements as in the Springer Nature book, which have now been removed with the new writing in Section 1.5.

We believe that our paper is now in a much better position to be considered for publishing in ISA Transactions. Thanks and look forward to receiving your acceptance in due course.

With kind regards,

Anthony Tri Tran

*Title page showing Author Details

Reliability Enhancement with Dependable Model Predictive Control

Tri Tran[a], Q. P. Ha[b], Robert Hunjet[a]

Defence Science and Technology Group, Australia[a,], University of Technology Sydney.[b]*

[a]81 Labs, Third Ave, Edinburgh, SA 5111, Australia.
[b]15 Broadway, Ultimo, NSW 2007, Australia.

---

*Corresponding author.

[1]Tri Tran and Robert Hunjet are with Defence Science and Technology Group, Edinburgh, Australia. Email: {anthony.tran; robert.hunjet}@dst.defence.gov.au

[2]Q. P. Ha is with Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia. Email: quang.ha@uts.edu.au

Preprint Submitted to ISA Transactions

Reponses to Reviewers' comments:

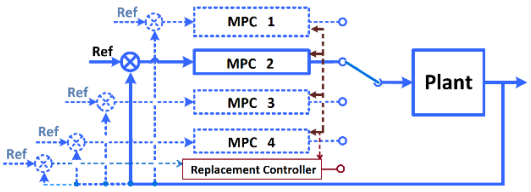
Submission ISATRANS-D-19-01651R2

Title: Reliability Enhancement with Dependable Model Predictive Control

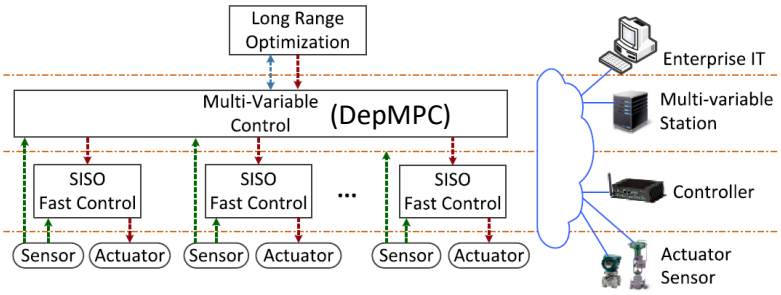
N/A as no further comments from Reviewers in R1 version.
The Associate Editor has recommended for accepting.

Reliability Enhancement with Dependable Model Predictive Control

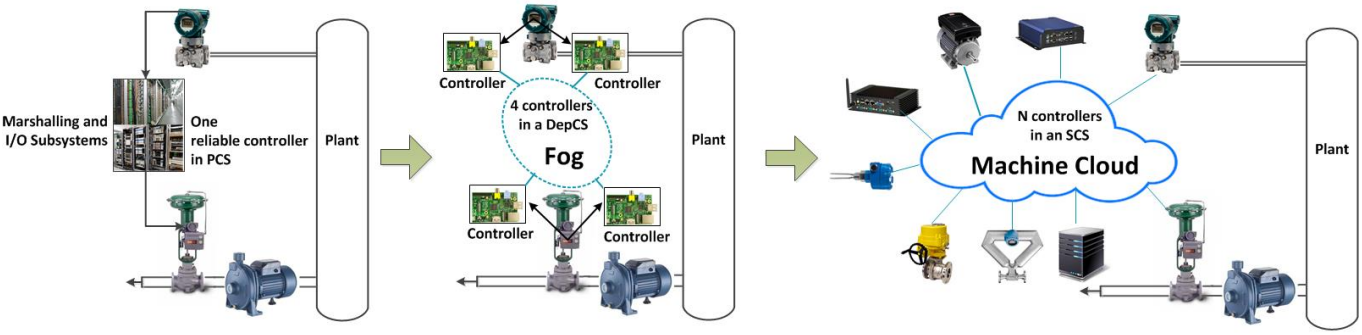
Graphical Abstract



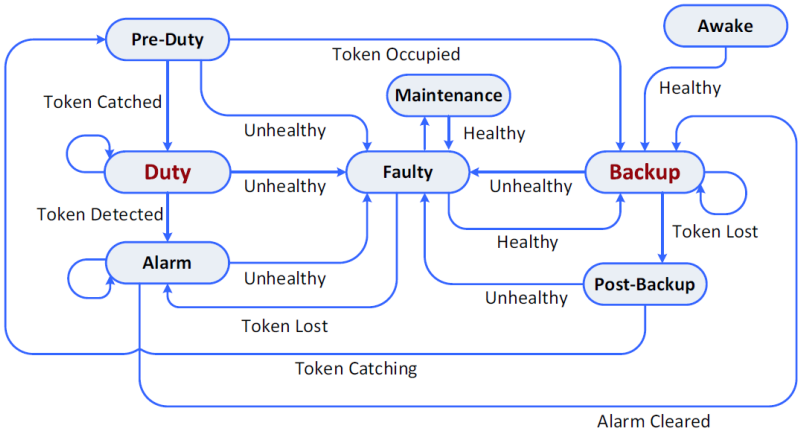
Dependable Model Predictive Control (DepMPC)



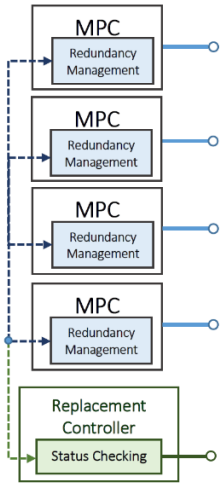
Hierarchical software-defined architecture



Dependable Control System (DepCS) with IoT versus DepMPC



State machine for redundancy management in DepMPC



Redundancy self-management

Reliability Enhancement with Dependable Model Predictive Control

Highlights

- Dependable Model Predictive Control (DepMPC) with a software-defined architecture for Operational Technology (OT) systems compatible with the emerging IEC/IEEE 60802 standards.
- Self-management of redundant controllers for smooth transferring between the standby and duty modes within a deterministic time thanks to time-sensitive networking.
- Replacement Controller (RC) in companion with the redundant controllers for improving the control performance of a DepMPC.
- Numerical simulations with three multi-variable plants and different choices of performance variables for illustration of the merits of the proposed framework.

Reliability Enhancement with Dependable Model Predictive Control

Abstract

Operational Technology (OT) systems are merging towards a conjoint architecture with the advances in communication networks and emerging standards such as IEC/IEEE 60802 for industrial automation, automotive, power and energy and other areas. In this paper, we present a Dependable Control System (DepCS) with Model Predictive Control (MPC) algorithm that works in such architectures using multiple MPC controllers (of a feedback control loop) to enhance the operational reliability. We termed this as *Dependable Model Predictive Control* (DepMPC) system. The reliability enhancement of a DepMPC system is achievable thanks to the fault-tolerance of multiple MPC controllers and the tractable information flows with Time-Sensitive Networking (TSN). Here, our discussion was focused only on the logical connectivity and not the hardware architecture. The numerical simulations are studied with three multi-variable plants that have control constraints. In this study, we introduced a *Replacement Controller* (RC) to improve the control performance of the DepMPC system. The combination of both the Replacement Controller and Dependable Model Predictive Control (RC-DepMPC) system proves a promising solution for actual implementations.

Keywords: Operational Technology system; Time-Sensitive Networking; Dependable Model Predictive Control; Replacement Controller

1. Introduction

Recent advances in telecommunication systems, cloud computing, Internet of Things (IoT) and data science are paving a new pathway for the next gener-

ation of systems and devices in all fields of human activities. Among many of the industrial applications, the Operational Technology (OT) systems in the manufacturing and processing industries seem to gain more benefits from these developments. The new OT systems that are facilitated by the emerging IEC/IEEE 60802 standards with Time-Sensitive Networking (TSN) [1] [1] will allow for innovative designs of the system architecture and interactions between the components and subsystems.

1.1. Operational Technology System

The description of the OT system is given in Appendix A. The term OT system is used to refer to the industrial computer system installed for machines or processing lines in manufacturing floors, and is different from the Information Technology (IT) systems installed in offices for business and enterprise applications. From the engineering and operational point of view, the term OT system infers the specialized computer systems that interface directly to the manufacturing or processing plants and take a critical role in (automated or semi-automated) the plant operations. The traditional system architecture (for a machine or processing line) usually consists of a single or a few redundant processors and input/output (I/O) interfaces integrated into a centralized platform (inside a control cabinet), which is sometimes extended to some remote I/Os via a proprietary communication network. Both digital and analog inputs and outputs are often accommodated in these systems; and the number of inputs and outputs usually ranges from a few to several thousand. For spatially larger-scale applications, a distributed architecture that is built up from the stand-alone systems is employed with some proprietary or standardized communication protocols and networks that are linked to a central control room (for example, an oil refinery). These configurations are prevalent in many systems, machinery, and manufacturing/processing plants worldwide. From the practi-

[1] The network with TSN protocols complying to IEC/IEEE 60802 standards is termed "TSN network" for conciseness, as in [1].

tioner's point of view, the industrial Programmable Logic Controller (PLC) and Distributed Control System (DCS) or Process Control System (PCS), and the likes, are all considered as OT systems. Interested readers may refer to the typical DCS/PCS hierarchical architecture given in [2] for further details. The term OT system was previously used in commercial documents. It is now formally defined and used in the IEC/IEEE 60802 standards which aim to cover both PLC systems and DCSs/PCSs (and the likes).

1.2. Time-Sensitive Networking

The Time-Sensitive Networking Task Group (TSN TG) is a part of the IEEE 802.1 Working Group. The following is the description from the task group [3]: "The charter of the TSN TG is to provide deterministic services through IEEE 802 networks, i.e., guaranteed packet transport with bounded low latency, low packet delay variation, and low packet loss. The TSN TG is a branch of the former IEEE 802.1 Audio Video Bridging TG." More specifically, the TSN network will be designed to allow for time-synchronized low latency streaming services and for building distributed and hard real-time systems with IEEE 802 networks. Thanks to the TSN network (and the associated Ethernet standards), an OT system can, as a result, use the same infrastructure to accommodate both the hard real-time control and the enterprise file exchange services. By defining queues based on time, a TSN network ensures a (bounded) maximum latency for traffic through switched networks. Thus, the TSN networks can be configured to provide services for deterministically fixed sampling-time and updating-time for both discrete and continuous control applications.

1.3. Dependable Control System

The Dependable Control System (DepCS), has been defined in a previous work [2], its details will be mentioned in Subsection 2.3. In short, a DepCS is a feedback control system having multiple controllers in a duty-standby structure, as depicted by Figure 3 in this paper (but with generic controllers, Figure 6 in

[2]). Here, we used the term "controller" as also used in feedback control literature such as PID controller or MPC controller, which can be implemented by software in a microprocessor or micro-controller [2]. It is worth to note that the term "dependable control" may have a different meaning: It can simply mean a generic reliable control action, or as formally used in the discrete control algorithms as indicative in some previous works at IFAC workshops in Dependable Control of Discrete Systems [4]. In this and other works, the DepCS is a feedback control loop for a continuous plant or a hybrid plant that have multiple controllers in duty and standby modes. Further discussions on DepCS and DepMPC are provided in the following sections.

1.4. Reliability Enhancement with Redundant Controllers

We assume in this work that the principles for designing fault-tolerant and reliable systems as given in literature, as well as practice (such as those in [5] and IEC 61508/61511 standards for functional protection systems) are used to enhance the operational reliability. In a DepMPC, multiple duplications of MPC controller is implemented to increase the integrity level (average probability of failure, typically from 10^{-6} to 10^{-1}) of the overall system. In this paper, we refer to them as redundant MPC controllers. Furthermore, we employed the duty-standby structure in which only a single controller will be on duty while the others are on standby. If the on-duty controller fails, one of the standby controllers will switch over to the duty role, to become the on-duty controller. The process of switching over from the standby to duty role and reversely is managed by the redundancy management function described in the following sections. In this work, we did not address the reliability of the physical hardware; we only consider the convex optimization in MPC, which is guaranteed recursively feasible at every time step (see Subsection 1.5). Hence, the optimizing feasibility of the MPC optimization problem is assured, and the solution

[2]The term "controller" may refer to a device such as PLC, but not the "controller" as in the feedback control literature in textbooks.

is unique at every time step. Therefore, the MPC algorithm can be certified failure proof; as a result, the failure of an MPC controller will come from the processor that runs the MPC algorithm.

The number of redundant MPC controllers will be dependent upon the front-end design and is specific to a particular project and customer requirements. For example, two controllers will be typically used (but not always), while one is on duty the other is on standby, to achieve the integrity level 2 (the average probability of failure ranges from 10^{-3} to 10^{-2}). Alternatively, three controllers will be typically used (but not always), one is on duty, and the other two are on standby, to achieve the integrity level 3 (the average probability of failure ranges from 10^{-4} to 10^{-3}), and so on. The integrity level of the overall system will be affected by all subsystems and components forming the system, including the data communication. There are different methods to calculate the desired integrity level of the overall system, it include the Reliability Block Diagram, details given in [5] and IEC 61508 functional safety standard, or others.

1.5. Distributed Model Predictive Control

The research in interconnected and network systems has become a specific focus within the control theory, see, e.g. [6, 7, 8, 9, 10, 11, 12]. Both dynamically-coupled and dynamically-decoupled systems have been considered in the past developments. The early work in [13] has presented different decomposition methods for large-scale systems. These methods, whilst interesting and insightful, are of little significance for the control problems using numerical optimization algorithms (for example, in the Model Predictive Control) in the environment of cyber-physical systems. The requirement for distributed approaches are evident; decentralized MPC for interconnected and network systems [14, 15, 16, 17, 18, 19, 10, 20, 21, 22, 23] has been employed in practical applications in various fields for diverse system architectures, such as plant-wide process systems, biological networks, transportation networks, and network robotics. Some new applications of decentralized MPCs include electric power systems, smart grids and heterogenous multi-agent systems, see for ex-

ample [24, 25, 26].

The use of MPC in the oil refining industry has been seen as a practical optimization-based approach to increase the profits since the 1980's [27, 28]. Some recent projects in autonomous vehicles have shown the potential of MPC for the robust control of fast real-time systems [29]. The comprehensive surveys in industrial MPCs are presented in [30, 31] with details of commercial software tools for different industries. From the academic research perspective a gap exists in that the industrial MPC algorithms of the past cannot guarantee the optimizing feasibility as time tends to infinity for hard-constrained systems – the recursive feasibility. Therefore, the closed-loop system stability is not assured unless additional constraints are employed in the MPC optimization. As a result of this analysis, the research in guaranteeing the stability of the closed-loop system with MPC controllers has been initiated and presented in [32, 33, 34, 35, 36, 37, 38, 14, 39, 40, 31, 41, 42, 43, 44, 45, 46], with the well-known paper [35] considered the authoritative resource in regards to stability and optimality within various MPC schemes.

Theoretically, the MPC optimization for systems having control and/or output constraints may become infeasible at some future time steps if an inappropriate predictive horizon is chosen [39]. In such cases, the MPC optimization may not be feasible, recursively, when the admissible set, as defined in [35], does not cover the initial states. Previous research has proved that if an adequate constraint is additionally imposed onto the MPC optimization at every time step, the closed-loop system stability will be obtained regardless of the length of the predictive horizon. The terminal state constraint developed from the maximal output admissible set [47, 48] has been used for this purpose and is now considered as the main stream approach. However, it has not been proved effective for distributed MPC, especially for dynamically-coupled systems with heterogenous subsystems [49, 21]. An alternative solution for decentralized MPCs with the quadratic dissipativity constraint has been presented in [50, 51, 52]. This approach, together with other two approaches, will be employed in the simulation

studies for DepMPC in this paper.

1.6. Summary of Contributions

The contribution of this work is threefold. First is the concept of DepMPC with multiple redundant MPC controllers integrated with duty-standby structure and presented together with the underlying state machine for self-managing the redundant MPC controllers. We did not address the reliability of the physical hardware in this paper. This work focuses on the MPC controller (application software), how multiple MPC controllers can be managed, and how the overall control performance is maintained in the presence of switching-over activities using a single performance variable. Furthermore, the hard real-time feature that is made available by TSN networks will be exploited for the duty-standby configuration. Second, three alternatives of the performance variable to be exchanged between the redundant MPC controllers are proposed and studied in simulation. The simulation results show that there is no single best performance variable among the three when applying to different plants. Therefore, the users will have the flexibility to choose the preferred performance variable, combining with simulation studies for a new application. Third, Replacement Controller (RC) is introduced to improve and recover the overall performance if the switching-over time is unusually long such that the control performance is significantly degraded. The three examples in the numerical simulation were conducted with both open-loop stable and unstable plants to study the control performance of the DepMPC in different scenarios. The results indicate that RC substantially improved the control performance in two out of three examples when the long switching-over time has caused the performance degradations. The TSN network also plays an important role in limiting the performance degradation and preventing unusual delays from the mode changes (e.g. standby to duty).

To the best of our knowledge, there is no previous study on the DepMPC with duty-standby structure and the companion RC. The DepMPC uses less exchanged data than the traditional Reliable Control System (RCS) designed

for single variable systems using output summation, see Subsection 2.1.2. In DepMPCs, the summation is not used, and only a single floating-point number is exchanged between the redundant MPC controllers to maintain the control performance. For example, the DepMPC uses a single floating-point number as the performance variable (see Subsection 2.1.2) while the RCS needs 100 floating-point numbers to be sent to the summation function for a 25-output controller if four controllers are installed. Furthermore, the redundant MPC controllers are also self-managed using the developed state machine to determine the duty and standby roles. The DepMPC can use this self-management ability and the TSN network to exploit the hardware resources for achieving a desirable integrity level.

1.7. Dependable Model Predictive Control System

The Dependable Model Predictive Control (DepMPC) system presented in this paper is a new implementation approach for dependable feedback-control loops in multi-variable systems with TSN networks; the control algorithm is MPC. The block diagram of a feedback control loop is shown in Figure 1 with a controller implemented in a processor.

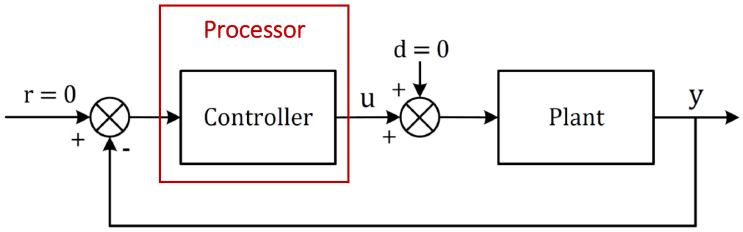


Figure 1: A feedback control loop - Block diagram.

In the presented system, we assumed that all instrumentations such as measuring transmitters and actuators can communicate and exchange real-time data (inputs and/or outputs and/or states) with all other processors within a logical domain. The processor is a computational platform for implementing the control laws or algorithms. The processor of a node can have numerous controllers

in operation (as explained in subsection 1.3). A controller can also exchange real-time data with other controllers within a logical domain. This means that a controller in a DepMPC will have both control calculation function for a feedback control loop and some data exchange functions, see Figure 6.

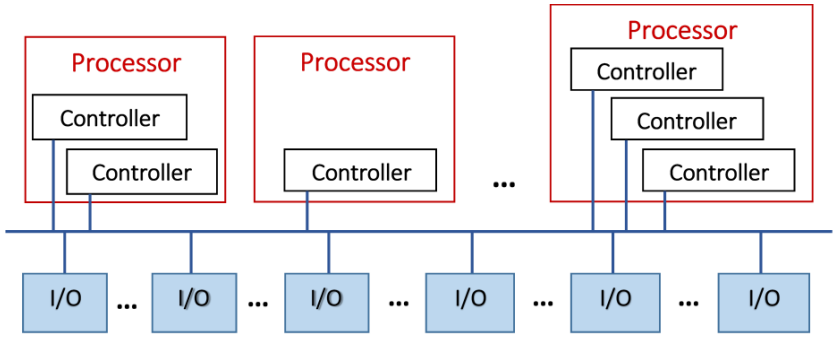


Figure 2: Logical communication and connectivity in an OT system.

Figure 2 depicts the processor, controllers, inputs/outputs, and logical connectivity in an OT system with TSN. A generic feedback-control loop usually has a single controller, an input, and an output for a single-input-single-output system. In the presented approach of dependable control system, a feedback control loop will have multiple controllers operating in a duty-standby architecture [2], as shown in Figure 3. We assume that there is only a single on-duty controller connecting to the plant at a given time. This concept is also applicable to multi-variable plants. As such, the proposed DepMPC in this paper is a dependable control system [2] using MPC algorithm to find the control vector for the controller (MPC controller). In MPC, a sequence of control vectors towards the predictive horizon is computed by a finite-horizon optimization-based algorithm [39]. MPC controllers are widely installed in oil refineries (for distillation columns and other processes). By providing the optimized set-points to the PID controllers, as illustrated in Figure 4, the MPC algorithm helped to increase the throughput and maximize the benefits.

The amount of data to be exchanged between the on-duty and standby MPC controllers in a DepMPC is kept to a minimal level compared to those

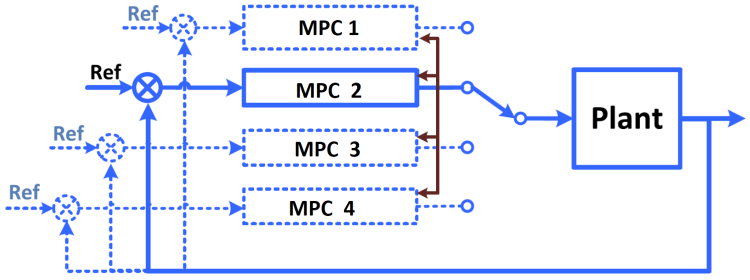


Figure 3: A Dependable Model Predictive Control system (DepMPC).

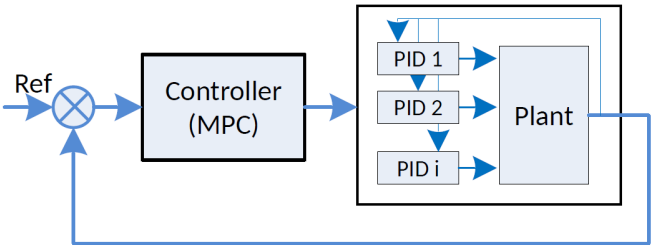
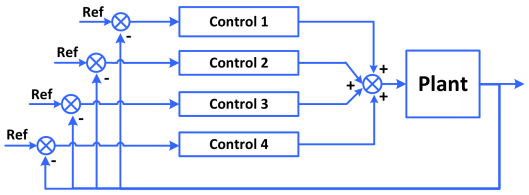


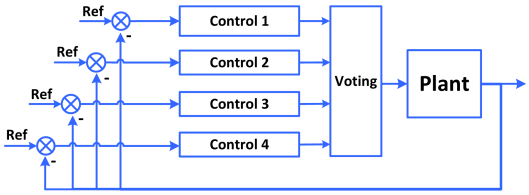
Figure 4: MPC provides set-points to PIDs.

in a Reliable Control System (RCS) as shown in Figure 5a. The RCS concept has been widely implemented in the industry in the past [53], as mentioned in a previous work [2]. In such an RCS, the outputs of all controllers sum up to a single vector. This is translated into a requirement to have several floating-point variables (e.g. four in Figure 5a) sent to the summation function for processing. Furthermore, a higher number of floating-point variables will be required for multiple-input multiple-output systems (e.g. four×the number of control inputs in Figure 5a). In a DepMPC, only a single floating-point variable will be exchanged between the MPC controllers, as shown in Figure 3, and perhaps a single digital variable for the token. This single exchanged-variable approach is particularly useful for multi-variable systems where the output is a stacking vector instead of a scalar. For example, if the controller has 25 outputs, 100 floating-point variables are transmitted to the summation block at every time step, when four controllers are implemented in the RCS.

In the first part of this paper, we will analyze the design requirements of a



(a) With control summation.



(b) With control voting.

Figure 5: Reliable Control Systems

DepMPC and propose the variables (performance variable and the token flag) to be exchanged among the redundant MPC controllers for managing the mode changes (duty and standby). Three new performance variables are presented in this work. They are different from that in the first paper [2], which introduced the concept of dependable control systems. In addition, the associated state machine for managing the redundant MPC controllers is presented here. In the simulation study for the DepMPC, the control performance degradation due to the long-duration switch-over time between the duty and standby MPC controllers will be addressed and discussed. Hence, we presented the Replacement Controller (RC) in the second part of this paper as a solution to improve the control performance of a DepMPC that has suffered long-duration transitions. Comprehensive numerical simulations for three applications including a paper machine, a helicopter, and a railway wheelset with bogie are studied in hundreds of different scenarios to illustrate effectiveness and control performance of the RC and DepMPC for both open-loop unstable and stable plants.

This paper is organized as follows. The design analysis for the dependable model predictive controller (DepMPC) is provided in Section 2, the model predictive control algorithm with the moving-horizon state estimation in Section

3, and the DepMPC problem description presented in this section. The performance variable for the DepMPC is addressed in Section 4 with three different alternatives. The results from simulation studies in Matlab with a paper machine, a helicopter and a railway wheelset with bogie are delivered in this section. We subsequently discuss the necessity of a Replacement Controller (RC) for the DepMPC which suffered from lengthy mode changes in a proposed control algorithm in Section 5. Simulation studies in Matlab with the RCs accompanying the DepMPC for the same three examples in Section 4 are also provided in this section. Section 6 concludes this paper.

2. Dependable Model Predictive Control - Design Analysis

The current approach for ensuring the continuous operation of a feedback control system implemented in a fault-tolerant computer system is to employ the technology of RCS from the control literature, see for example [53] and references therein. Figure 5a depicts a typical RCS block diagram. This technology may be difficult to apply in multi-variable systems owing to the requirement of summing up the manipulated variables of all redundant controllers. Furthermore, this is a legacy from the control systems using analog circuits and later expanded to digital circuits (and computerized systems) relying on hardware-defined architectures. Similarly, the architecture with output voting shown in Figure 5b also requires the manipulated variables from all redundant controllers to be sent to the voting function for processing (we will not address the voting architecture in this work). Those difficulties have already been described in a previous work [2]. In this work, we propose the DepMPC shown in Figure 3 for multi-variable systems as an alternative design using the TSN network defined in the emerging IEC/IEEE 60802 standards. In a DepMPC, only a single MPC controller is in the duty mode at any time instants, to manipulate the plant directly, while the other MPC controllers are in their standby mode. The summation is not required and only a floating-point variable and a token flag are exchanged between the member controllers for the redundancy self-management

as well as the control performance assurance. [3]

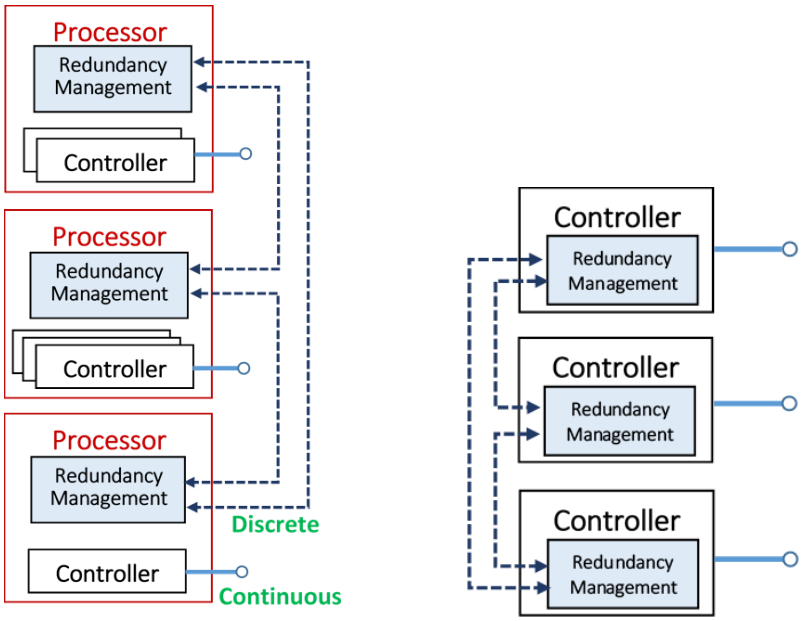
2.1. Dependable Model Predictive Control - Enhanced reliability with redundancy self-management

2.1.1. Merging redundancy management and controller

In order to enhance the reliability of a DepMPC system, the number of redundant MPC controllers have to be increased and the redundancy self-management implemented, as illustrated in Figure 6b. Figure 6a shows the approach of segregating the redundancy management to the control algorithm, which has been widely implemented in a DCS. The redundancy management is often treated as a part of the operating system of the DCS computer system. As a result, the quantitative reliability of the RCS depends on the number of redundant CPUs, which is usually two or three. The RCS architecture with variable summation is compelling with this segregated redundancy management because the control algorithm will not be dramatically changed.

The reliability of the DepMPC in this development will be enhanced by having a higher number of redundant MPC controllers with a software-defined architecture. The redundancy self-management is implemented for each MPC controller, but is not segregated as in the case of RCS. In other words, we merged the redundancy management and control algorithm into a single entity – the MPC controller itself. Therefore, the MPC controller is self-reliant in terms of operational activities. We will further discussed the self-reliant characteristics in Subsection 2.2. Thanks to the TSN network in an OT system (which makes the self-reliant features a reality with hard real-time capability), the number of redundant MPC controllers in a DepMPC system can be significantly higher than that in the RCS when the MPC controllers can manage the duty and standby modes by themselves. The TSN network helps to eliminate the unex-

[3] In the field of application software, the term "high-integrity" is often used to imply the characteristic of a software system that is fault-tolerant and achieve a higher integrity level, see for example [54]. So the DepMPC can be considered as a high-integrity software system.



(a) Separated redundancy management. (b) Redundancy self-management.

Figure 6: Merging redundancy management and controller

pected (or non-deterministic) delays in the data communication system that will subsequently prolong the switching-over time (and thus, significantly degrade the control performance of the DepMPC). The hardware resource of the OT system can be exploited, as a result, to enhance the reliability for the DepMPC using a software-defined architecture. This procedure is further discussed in Subsection 2.3.

2.1.2. Single performance-variable approach

The key for a successful implementation of a dependable system rests with the amount of data to be transferred between the redundant entities [5]. It is desirable to have as less as possible the exchanged data between the duty and standby MPC controllers – less communication will translate to fewer delays. The presented method requires only a single real number (floating point) exchange between the redundant MPC controllers for performance assurance, and a digital bit for the token flag, as shown in Figure 7. Initially, we have

introduced this single-variable concept with a particular example in the previous work [2]. Here, we generalized the approach with different alternatives and opened for new performance variables in the future.

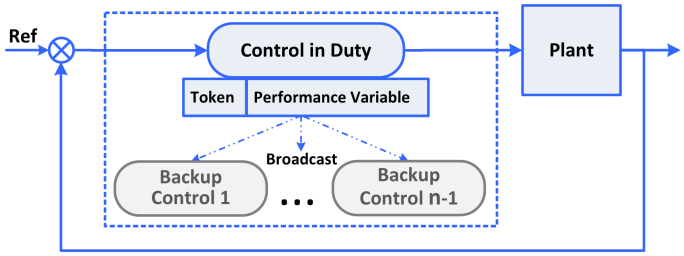


Figure 7: Dependable Model Predictive Control (DepMPC) with $n\times$redundancy. The on-duty controller broadcasts a single token and a single performance variable to the standby controllers. The on-duty controller keeps the token on a first-in-failure-out basis. The performance variable is broadcasted to the standby controllers whose responsibility is to use this performance variable to maintain the control performance of the DepMPC after a failure incident.

With this single-variable approach, the communication data usage in DepMPC is more efficient while achieving higher dependability thanks to the simplicity of broadcasting only a single floating-point number for both single-variable andseveral-variable plants. Furthermore, for some applications, it is sufficient to use the token flag only while the performance variable becomes negligible because the open-loop system is inherently stable (see numerical examples below). The DepMPC is a much better solution with TSN network for multi-variable systems than the RCS when counting the floating-point numbers that are required for exchange between the member (redundant) controllers. In other words, the DepMPC is more data efficient than the RCS.

In an RCS, the outputs of all controllers sums up to a single output vector. This translates into a requirement to have several floating-point numbers (e.g. four in Figure 5a) sent to the summation function for processing. A higher number of floating-point numbers will be required for multiple-variable systems. For example, if the controller has 25 outputs, 100 floating-point numbers will have to be transmitted to the summation block at every time step when four

controllers are implemented. On the contrary, only a single floating-point number is required together with a single digital bit in a DepMPC even when m MPC controllers are implemented, where m is much greater than four.

2.2. Dependable Model Predictive Control - Operational challenge and solution

The main challenge of the DepMPC lies at the redundancy self-management together with maintaining the control performance in the presence of mode changes (e.g. from standby to duty). The approach in this paper combines the control algorithm and the redundancy management into an integrated solution, rather than splitting their designs separately. In particular, the DepMPC shall be self-reliant: The on-duty MPC controller will not require any information from the other standby MPC controllers. On the other hand, the standby MPC controllers instantly become self-reliant using minimal information obtained from the on-duty MPC controller with a single performance variable. The proposed DepMPC has several redundant MPC controllers, as depicted by Figure 3. In a DepMPC, we only need two variables to be broadcasted from the on-duty MPC controller, one bit for the token flag, and one scalar variable (floating point number) for assuring the control performance, as illustrated in Figure 7. This figure shows the relay of information from the on-duty MPC controller to the standby MPC controllers with two variables, a digital token and a real-number performance variable.

We use a token-like procedure here. The on-duty MPC controller keeps the token on a first-in-failure-out basis and broadcasts the token status to all the standby MPC controllers. If the on-duty MPC controller stops broadcasting the token after the maximum transmission time (defined by the TSN protocol), the token will be considered free for the other standby MPC controllers to catch. The standby MPC controller that captures the token in the first place will become the on-duty MPC controller. The proposed state machine for this process is provided in Figure 8 for illustration and the detailed description in Section 2.4.1.

In addition to that, the performance variable (representing the control per-

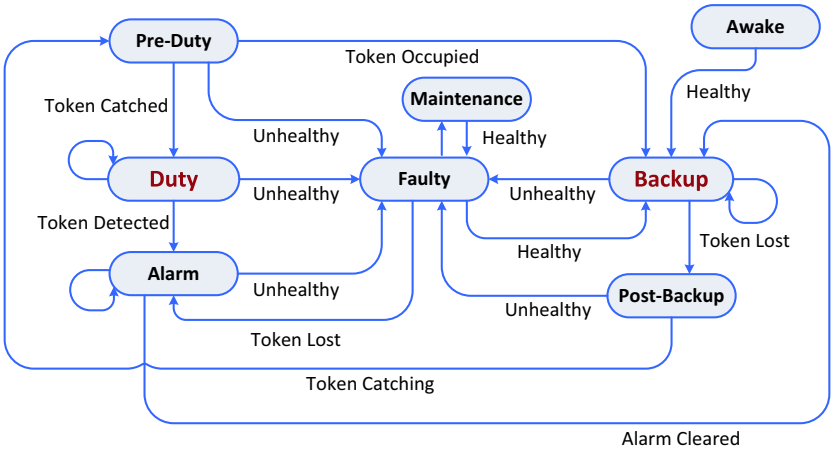


Figure 8: The proposed state machine for managing the redundant MPC controllers of a DepMPC. The block "Backup" refers to the "standby" mode.

formance) will also be broadcasted by the on-duty MPC controller to all the standby MPC controllers. The healthy MPC controllers (on standby) thus receive the performance variable from the on-duty MPC controller at every time step (within the maximum transmission time defined by the TSN protocol). With the use of this performance variable, the on-duty MPC controller (that had been activated from the standby mode) will ensure that the overall control performance of the DepMPC is satisfactory.

2.3. Dependable Model Predictive Control - New features

We have presented in a previous work [2] the so-called Dependable Control System (DepCS), but it is quite different to the presented DepMPC here. For clarity, the differences between DepCS and DepMPC are depicted by Figure 9. In a DepCS, the internal processors of the sensor and actuator are duplicated such that there are four (or more) redundant processors for a single feedback control loop. In this work, the software-defined architecture provides flexible choices for the DepMPC. The MPC controllers of a DepMPC can implemented on the same or on different processors positioned at various locations in an OT system; they are external to the actuators and sensors. Furthermore, the

DepMPC can act as a secondary control layer providing set-points to multiple DepCSs as single-input-single-output control loops, but not reversely. In terms of fault tolerance, the DepCS is $m\times$redundant (e.g. $4\times$redundant) while the DepMPC is $n\times$redundant with $n \geqslant m$, if there are more than m nodes available for use (which is usually the case).

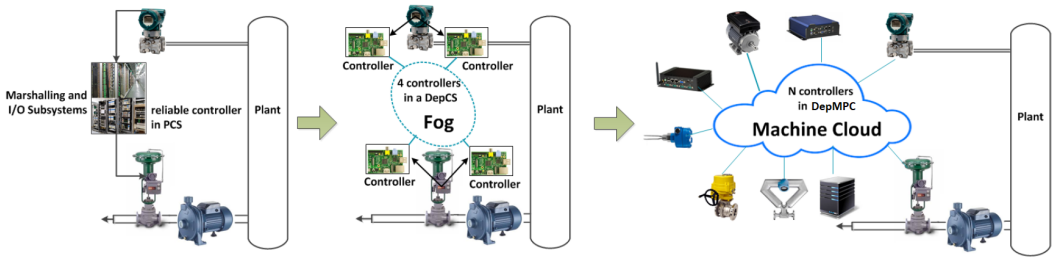


Figure 9: The controllers are implemented on the internal platforms of the transmitter and actuator in a Dependable Control System (DepCS) while the controllers are implemented on the processors of an OT system in a Dependable Model Predictive Control (DepMPC), external to the transmitters and actuators. In terms of fault tolerances, the DepCS is $4\times$redundant while the DepMPC can be $n\times$redundant, typically with $n \geqslant 4$.

The DepMPC uses MPC algorithm while the state-feedback control law is applied in DepCS with an incremental dissipativity constraint, as shown in previous study [2]. The numerical example therein was for an isolated power system while the MPC algorithm together with an RC (in Section 5) is employed in the DepMPC. The DepMPC with and without the RC are studied in simulation for three different examples of open-loop unstable and stable mechanical systems in Section 4.4. Moreover, three alternatives of the performance variable are presented in this work (Subsection 4.1), but is not limited to an incremental dissipativity constraint as in DepCS [2]. Therefore, it is more flexible for the user to chose the preferred performance variable for his/her applications. We also showed in the simulation studies that the performance variable might not be needed when the MPC algorithm is used as in the case of an open-loop stable system. Of course, the DepMPC for an open-loop unstable plant will still require the performance variable, as in the helicopter example.

2.4. Dependable Model Predictive Control - Operational description

In a DepMPC, only one of its $n\times$redundant controllers takes the duty role at any time instant. For the multi-variable control algorithm of a controller, we employ the MPC – a finite-horizon version of the Linear Quadratic Regulator (LQR) for systems having constraints [39], implemented on a rolling manner, as delineated in Section 3. Each redundant controller independently solves the MPC optimization problem at each time step (control calculation function), but only outputs of the on-duty controller are connected to the plant. A controller has both control calculation function and some data exchange functions for redundancy self-management and performance assurance.

2.4.1. Redundancy self-management with a single token

The activities of a self-managed DepMPC can be described as follows: The token status is consecutively broadcasted from the on-duty MPC controller to the standby MPC controllers within the population of the DepMPC. The rule for the token keeping is first-in failure-out. A standby MPC controller will be activated into the duty role upon a release of the token by the on-duty MPC controller due to failures. The state machine for this redundancy management is provided in Figure 8, and is described as follows:

Each controller have two status variables, "Healthy" and "Unhealthy", which are either from hardware or software failures, or both. There are eight states: "Awake", "Maintenance", "Alarm", "Pre-Duty", "Duty", "Backup" (i.e. standby mode), "Post-Backup", and "Faulty". The token has five statuses consisting of "Detected", "Catching", "Caught", "Lost", and "Occupied".

The initiation is at the "Awake" state where the transitions begin. Then, if the controller is "Healthy", the status becomes "Backup". Depending on the controller status which is either 'Healthy" or "Unhealthy", and the token status which is either "Lost" or "Occupied", the status will be either "Faulty" or "Post-Backup" (i.e. prior to standby mode). Similarly, the other states and transitions are delineated in the state machine in Figure 8. The stable states for each controller include "Duty" and "Backup" which represent the corresponding

on-duty and standby modes. Here, the first standby controller that successfully gets the token will become the on-duty controller. This state machine thus well serves the first-in failure-out principle for the DepMPC system.

With the relays in this state machines, a controller will spend several sampling times before requesting a duty-standby switch-over. This is well recognized in the area of fault-tolerant computer systems [5]. Therefore, the TSN protocol is crucial for achieving a successful switch-over within a deterministic time. In other words, the self-reliant DepMPC is realizable thanks to the TSN network.

2.4.2. Performance assurance with a single variable

In a DepMPC, the newly activated (on-duty) MPC controller maintains the control performance by using the passing-on performance variable. Here, we nominate three alternatives for the performance variable. They include (1) the real-time value of the MPC cost function, (2) the value of the Lyapunov function in the form $V_k = x_k^T P x_k,\ P = P^T \succ 0$, where x is the state vector, and (3) the quadratic dissipativity constraint presented in [50, 51]. The detailed development is given in Subsection 4.1. In the numerical examples in Section 4.4, we will show the resulting control performance of the DepMPC using these three performance variables.

If k_s is the time instant, at which the duty controller is faulty. We assume that the switching-over activity will take place in δ time steps, that is $\delta \geq 1$. The maximum value of δ is bounded in TSN networks. During the transition time, the last known value of the control vector $u(k_s)$ will be applied to manipulate the plant. This can be done by having a local buffer at the smart actuator, or simply using a mechanical latch to keep the actuator at the last position. Once the state transition is completed at the time instant $k_s + \delta$, the value of $u(k_s)$ will be retrieved to the newly assigned on-duty controller, by having $u(k_s) = u(k_s + \delta)$.

In some cases, the performance variable in DepMPC can be completely ignored (Subsections 3.2.2 and 4.4.1), this is dependent upon the plant property, especially for those having an open-loop stable model. From the control engi-

neering point of view, there are no universal answers to the performance assurance question for various applications. In general, the performance variable is still required, especially for the open-loop unstable plants. The performance variable will be chosen based on the control design and the control synthesis method. There are other performance variables from the knowledge of control theory for the user to choose than just the three in this paper.

2.5. Software-defined Architecture

The hierarchical structure of long-range optimization providing set-points to the multi-variable controllers, which in turn, provides set-points to the single feedback control loops will be needed for larger-scale plants similar to that in the current practice [55]. However, they are implemented by a software-defined architecture as shown in Figure 10 when the DepMPC is employed. The hierarchies only exist within the application software. The heterogeneous and multiple time-scale architecture can also be facilitated by the software-defined architecture, in which the data exchanged between the software components can be across the layers, as shown in Figure 11. This software-defined architecture fully supports the emerging standard IEC/IEEE 60802 with TSN. Each multi-variable control loop shown in Figures 10 and 11 can be designed as the DepMPC developed in this work. The problem formulation and MPC optimization are presented firstly in the next section.

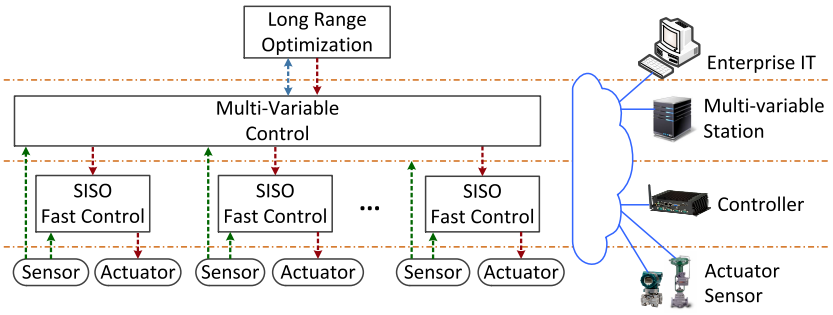


Figure 10: Hierarchical software-defined architecture. Each multi-variable control loop can be designed as a DepMPC.

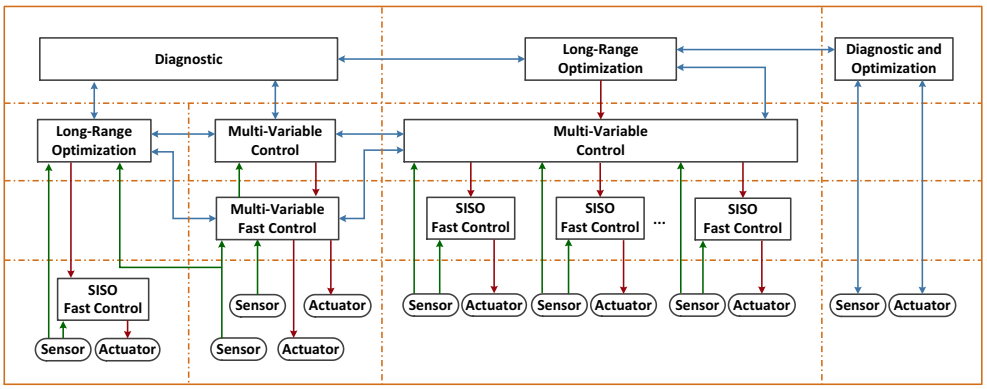


Figure 11: Multiple time-scale, heterogeneous, and hierarchical software-defined architecture. Each multi-variable control loop can be designed as a DepMPC.

3. Model Predictive Control Problem Formulation

3.1. Notation

In the equations in this section, capital letters denote matrices, while lower-case alphabets and Greek letters denote column vectors and scalars, respectively. The field of real numbers and the set of integers are denoted by $\mathbb{R}$, $\mathbb{Z}$, respectively. A function $\alpha : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ is of class $\mathcal{K}$ if it is continuous, strictly increasing and $\alpha(0) = 0$, and is of class $\mathcal{K}_\infty$ if, in addition, it is unbounded. The ℓ_2-norm (Euclidean) of vector u is denoted by $\|u\|$, while $\|x\|_{\mathcal{Q}}$ is the weighted ℓ_2-norm of x, $\mathcal{Q} \succ 0$. In the discrete-time domain, the time index is denoted by k, $k \in \mathbb{Z}$.

3.2. Problem Formulation

3.2.1. System Model and Model Predictive Control

Consider a plant Σ under control having a discrete-time state-space model of the form:

$$\Sigma : x(k+1) = Ax(k) + Bu(k), \tag{1}$$

where $x(k) \in \mathbb{R}^n$ and $u(k) \in \mathbb{R}^m$ are the state and control vector, respectively. The following control and state constraints are considered herein:

$$\mathbb{U} := \{u : \ \|u\|^2 \leq \eta, \ \eta > 0\}, \tag{2}$$

$$\mathbb{X} := \{x : \ \|x\|^2 \leq \rho, \ \rho > 0\}. \tag{3}$$

In this section, we consider state feedback problem with the state vector x as the output (controlled variable), and the control $u(k)$ (manipulated variable) is computed online by the model predictive control (MPC) algorithm that employs the model (1). The traditional objective function of MPC is considered [39], as follows:

$$\mathcal{J}(k) = \sum_{\ell=1}^{N} \|x(k+\ell)\|_{\mathcal{Q}}^2 + \|u(k+\ell-1)\|_{\mathcal{R}}^2,$$

where $\mathcal{Q}, \mathcal{R}$ are user chosen weighting matrices, and N is the predictive (and control) horizon. The current state vector $x(k)$ is known in the state feedback problem. For output feedback problem, the moving-horizon estimation will be used to derive the state vector $x(k)$ and disturbances.

The optimization problem of minimizing $\mathcal{J}(k)$ subject to the model (1), the control constraint $u \in \mathbb{U}$ (2), the sate constraint $x \in \mathbb{X}$ (3) in the following:

$$\begin{aligned} &\min_{\hat{\mathbf{u}}} \ \mathcal{J}(k) \\ &\text{subject to} \quad (1), \ (2), \ (3), \end{aligned} \tag{4}$$

is then solved for the predictive vector sequence $\hat{\mathbf{u}} := \{u(k), u(k+1), \ldots, u(k+N-1)\}$. The minimizing sequence $\hat{\mathbf{u}}^*$ which consists of N elements of $u^*(k+\ell), \ell = 0, 1, ..., N-1$, will be obtained as a result of this online computation. Only the first element $u^*(k)$ is output to control Σ (1). This rolling process is repeated at the next time step, and continues thereon.

Now, denote $\hat{\mathbf{u}}^*(x(k))$ as the optimal solution of (4) for the current state $x(k)$. The optimization problem (4) defines an implicit feedback law of the form

$$u = \kappa_N(x(k)) = u^*(0, x(k)), \tag{5}$$

where $u^*(k, x(k))$ is the optimal solution at the time step $k = 0, \ldots, N-1$ in the horizon, i.e. $u^*(0, x(k))$ is the first vector of the sequence $\hat{\mathbf{u}}^*(x(k))$. For

the convex optimization, we assume without loss of generality that $\hat{\boldsymbol{u}}^*(x(k))$ is uniquely defined herein.

3.2.2. Feasibility and Stability

According to the literature of MPC, the constrained closed-loop system Σ (1) and (5) is not guaranteed stable if the predictive horizon N is not sufficiently long [39]. And further, the optimization (4) may not also be recursively feasible if the initial state vector does not belong to an appropriate admissible set while an additional terminal constraint is also required [35]. In this work, we choose to have the predictive horizon sufficiently long to achieve the closed-loop system stability.

An advantage of deploying DepMPC is that the passing-on performance variable might not be needful. We found in simulation studies that the performance variable, which helped to improve the control performance in the static state feedback of an DepCS in [2] does not deliver the same result for DepMPC. The control performance may not degrade except by using the performance variable in a DepMPC. This is explained by the rolling principle in MPC: The control is re-computed (with a new feedback gain) at each time step using the updated state data. The numerical examples in the next section illustrate the advantage of having MPC as a control algorithm for multi-variable systems. The control performance improvement is less than 5% (only) when the passing-on performance variable is used. As a result, the DepMPC can be simpler without using the performance variable, which is transferred between the redundant controllers; for example, in some cases of open-loop stable plants.

We outlined the state estimation algorithm for constrained systems in the next subsection. The state estimation is required when the measurement output vector is different from the state vector, and the disturbances are present.

3.3. Moving Horizon State Estimation

For output feedback problems, consider the following state-space model with disturbance vector:

$$\Sigma : \begin{cases} x(k+1) = Ax(k) + Bu(k) + d(k), \\ y(k) = Cx(k) + v(k), \end{cases} \tag{6}$$

where $x \in \mathbb{X} \subset \mathbb{R}^n$ and $u \in \mathbb{U} \subset \mathbb{R}^m$ are the state and control vector, respectively, $d \in \mathbb{W} \subset \mathbb{R}^n$ is the unknown but bounded disturbance vector, $y \in \mathbb{R}^q$ is the measurement output vector, and $v \in \mathbb{V} \subset \mathbb{R}^q$ is the measurement error vector.

The state vector $x(k)$ is estimated from the measurement output $y(k)$ and the historical control u before being used in the MPC optimization (4). The predictive model used in (4) is assumed to be perfect without disturbances. When disturbances are present, the state $x(k)$ will be estimated but not measured directly. The Moving Horizon Estimation (MHE) algorithm developed in [56] is adopted for this state estimation problem for the constrained Σ. Denote the estimated state as $\hat{x}$ and the estimated disturbance sequence

$$\{\hat{d}(k)\} := \{\hat{d}(k-N+1), \hat{d}(k-N+2), \ldots, \hat{d}(k-1)\}.$$

At every time step, the state $x(k)$ is estimated by formulating and solving the MHE optimization having the following objective function:

$$\begin{aligned} &\Psi\big(\hat{x}(k-N+1), \{\hat{d}(k)\}\big) \\ &= \phi(k-N+1) + \sum_{\ell=k-N+1}^{k} \|y(\ell) - Cx(\ell)\|^2_{R^{-1}} + \sum_{\ell=k-N+1}^{k-1} \|w(\ell)\|^2_{Q^{-1}}, \end{aligned}$$

where the arrival cost $\phi(k-N+1)$ is given as

$$\phi(k-N+1) = \frac{1}{2}\|x(k-N+1) - \hat{x}(k-N+1|k-N)\|^2_{P^{-1}_{k-N+1|k-N}}, \tag{7}$$

in which

$$\begin{aligned} P_{k-N+1|k-N} \;=\; & Q + AP_{t-N|t-N-1}A^T - \\ & AP_{k-N|k-N-1}C^T\big(R + CP_{k-N|k-N-1}C^T\big)^{-1}CP_{k-N|k-N-1}A^T. \end{aligned} \tag{8}$$

The MHE optimization problem is summarized as follows:

$$\Theta(k)^* = \min_{x(k-N+1),\{d(k)\}} \Psi\big(\hat{x}(k-N+1), \{\hat{d}(k)\}\big) \tag{9}$$

subject to

$$\begin{aligned}
&x(\ell) = Ax(\ell-1) + Bu(\ell-1) + d(\ell-1), \text{ for } \ell = k-N+1, \ldots, k \\
&y(\ell-1) = Cx(\ell-1) + v(\ell-1), \\
&x(\ell) \in \mathbb{X},\ d(\ell) \in \mathbb{W},\ v(\ell) \in \mathbb{V}.
\end{aligned}$$

The updated state estimation is then obtained by the following prediction model:

$$\hat{x}(k) = A^{N-1}\hat{x}(k-N+1) + \sum_{j=k-N+1}^{k-1} A^{k-j-1}\big[Bu(j) + \hat{d}(j)\big]. \tag{10}$$

Once the state $x(k) = \hat{x}(k)$ is obtained, the MPC optimization (4) is then formulated and solved for the minimizing sequence $\hat{\mathbf{u}}^*$, where only the first element $u^*(k, \hat{x}(k))$ is outputted to control Σ (1).

3.4. DepMPC problem description

In a DepMPC there are multiple MPC controllers, that are also self-reliant in redundancy management using the two exchanged variables, the token and the performance variable among them. One MPC controller is on duty while the others are on standby. Each MPC controller will solve the MPC optimization problem at each time step, but only outputs of the on-duty MPC controller are connected to the plant (system under control). Whenever the on-duty MPC controller fails to operate, one of the standby MPC controllers takes over the duty role to become the on-duty controller. This switching-over process between the on-duty and standby MPC controllers can be managed using the token and the state-machine developed in this work.

During the regular failure-free operations, the performance variable is transferred from the on-duty MPC controller to the standby ones at every time step. The standby MPC controller that has switched over to the duty role will use

this obtained performance variable to maintain the control performance of the feedback loop, i.e., of the DepMPC. In this work, the performance variable is employed in an additional constraint that is to be imposed on the MPC optimization at each MPC controller. There are three proposed performance variables in this paper, presented in the next section. The user can also come up with another performance variable using the knowledge of control theory. If one of the performance variables above is to be chosen, simulation studies will be required to support the decision, as there is no universal best performance variable for all types of system models (of the plant under control).

If the switching-over time between the duty and standby controllers is excessively long, the system becomes an opened loop during that long duration, the Replacement Controller (RC) will then be activated by checking a performance index against the pre-defined threshold. The RC will control the plant (after being activated) until a standby MPC controller has successfully taken over the duty role.

4. Performance Variable for DepMPC and Control Procedure

4.1. Performance Variable for DepMPC

We propose three alternatives of the performance variable in this section. They are different from the variable employed f or the DepCS in [2]. Denote the time instant at which the on-duty controller is faulty as k_s, and k is the current time step.

Option 1: From the literature of MPC (e.g. [39]), it is well known that the value function $\mathcal{J}(k)$ is treated as a Lyapunov function for the closed-loop system. The closed-loop stability is achieved by the monotonicity of the optimizing $\mathcal{J}(k)$ over time. In this work, we apply this property to the DepMPC using the optimizing value of $\mathcal{J}(k)$ as the performance variable. The optimal $\mathcal{J}(k)$ is saved by the on-duty controller, which is then broadcasted to the standby controller. Whenever a standby controller takes the duty role, it will use this value of $\mathcal{J}(k_s)$

and formulates an additional constraint for the MPC. This constraint resembles the monotonically decreasing property of $\mathcal{J}(k)$, as follows:

$$\mathcal{J}(k) < \beta \times \sum_{\ell=k_s}^{k_s+N} \|x(k+\ell)\|_{\mathcal{Q}}^2 + \|u(k+\ell-1)\|_{\mathcal{R}}^2,\ \beta > 0,$$

which will be converted to an inequality constraint w.r.t. $\hat{\mathbf{u}}$. Here, a small coefficient $\beta < 1$ is used to enforce the decreasing for $\mathcal{J}(k)$ over time. We denote this constraint as $\hat{\mathbf{u}} \in \mathbb{U}_b$. The MPC optimization then becomes

$$\begin{aligned} &\min_{\hat{\mathbf{u}}}\ \mathcal{J}(k) \\ &\text{subject to}\quad (1),\ (2),\ (3),\ \hat{\mathbf{u}} \in \mathbb{U}_b. \end{aligned} \tag{11}$$

The minimizing sequence $\hat{\mathbf{u}}^*$ will be obtained as a result of this online computation. And only the first element $u^*(k)$ is outputted to control Σ (1).

Option 2: The second alternative is to use the non-increasing property of the Lyapunov function $V(k) = x(k)^T P x(k)$, where P is determined by Riccati equation for the LQR problem. The constraint of the form $V(k+1) \leq \gamma \times V(k)$, for $0 < \gamma < 1$, which is equivalent to

$$u(k^T)B^T PBu(k) + 2x(k)^T A^T PBu(k) + x(k)^T (A^T PA - \gamma P)x(k) < 0,\ 0 < \gamma < 1, \tag{12}$$

where $x(k) = x(k_s)$ is known, will be imposed additionally (replace $\hat{\mathbf{u}} \in \mathbb{U}_b$) on the MPC optimization (11). It is worth mentioning that $u(k)$ is the first vector in the sequence $\hat{\mathbf{u}}$. The value of $V(k)$ which is the performance variable will be broadcasted by the on-duty controller to the standby controllers. Once activated into the duty role, the MPC optimization in this controller uses the immediate value of $x(k_s)$ received from the on-duty controller (currently in a failure mode) in the above constraint (12). This is an ellipsoidal constraint w.r.t $u(k)$, denoted as $u(k) \in \mathbb{U}_V$.

Option 3: The third alternative is to use the dissipativity constraint developed in [50, 51] as an enforced stability constraint for the MPC problem. It

is also an ellipsoidal constraint w.r.t $u(k)$. In this approach, the MPC optimization (4) is additionally imposed with a dissipativity constraint of the form $\xi\big(u(k), x(k)\big) \leqslant \gamma\, \xi\big(u(k-1), x(k-1)\big)$, in which $\gamma < 1$ and $\xi(u, x)$ is a quadratic function w.r.t x and u. With $x(k), x(k-1), u(k-1)$ are known, this dissipativity constraint can be converted into a quadratic constraint w.r.t $u(k)$ which is the first vector element of the sequence $\{u(k), u(k+1), \ldots, u(k+N-1)\}$ in (4). For DepMPC, $\xi\big(u(k), x(k)\big)$ is the performance variable and a constraint of the form $\xi\big(u(k), x(k)\big) \leqslant \gamma\, \xi\big(u(k_s), x(k_s)\big)$ will also be imposed onto the MPC optimization (4) at the time step $k > k_s$. This is also an ellipsoidal constraint w.r.t $u(k)$, denoted as $u(k) \in \mathbb{U}_\xi$ here. Further details on the dissipativity constraint can be found in [51]; the convergence condition with the dissipativity constraint has been stated in [51], and restated in Proposition 1 below for clarity.

Proposition 1. *Consider Σ (1) and a non-negative real-value function $\xi\big(u(k), x(k)\big)$, $\xi : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}_0^+$, $\xi\big(u(0), x(0)\big)$ is finite. Let $\sigma \in \mathbb{R}^+$, $\sigma < 1$, and $\beta \in \mathbb{R}^+$, $\beta < 1$. Suppose there are two $\mathcal{K}_\infty$ functions $\underline{\alpha}(\|x\|)$, $\overline{\alpha}(\|x\|)$ and a real-value non-negative function $V\big(x(k)\big)$, $V : \mathbb{R}^n \to \mathbb{R}_0^+$, such that for each finite $x(0) \in \mathbb{R}^n$ the following conditions hold for all $k > 0$:*

1. $\underline{\alpha}\big(\|x(k)\|\big) \leq V\big(x(k)\big) \leq \overline{\alpha}\big(\|x(k)\|\big)$,
2. $V\big(x(k)\big) - \sigma\, V\big(x(k-1)\big) \leqslant \xi\big(u(k), x(k)\big)$,
3. $0 \leqslant \xi\big(u(k), x(k)\big) \leqslant \beta\, \xi\big(u(k-1), x(k-1)\big)$,

with some control sequences $\{u(k) \in \mathbb{R}^m\}$; Then $x(k)$ remains finite and $\|x(k)\| \to 0$ as $k \to \infty$.

Proof In [51]. ■

It is noted that the three presented performance variables are generic for both linear and non-linear systems, whereas the incremental constraint in [2] is only applicable for linear systems.

4.2. DepMPC Control Procedure

The control algorithm of DepMPC is outlined in a pseudo form, as follows: At each time instant k,

1. Obtain the measurement output $y(k)$ from the respective transmitters.
2. Estimate $\hat{x}(k)$ by solving the MHE optimization (10).
3. Compute $\hat{\mathbf{u}}^*$ by solving the MPC optimization (4).
4. Output the first element $u^*(k)$ of $\hat{\mathbf{u}}^*$ to control the plant Σ.
5. Save the values of $u(k)$, $\hat{d}(k)$, $\hat{v}(k)$ and $\mathcal{J}(k)$.
6. If a duty-standby transition request is received,
   (a) Verify the transition from the standby role to the duty role. If the transition has been successful:
      i. Estimate $\hat{x}(k = k_s + \delta + 1)$ by solving the MHE optimization (10).
      ii. Formulate the constraint $\hat{\mathbf{u}} \in \mathbb{U}_b$ with $\mathcal{J}(k_s)$ as in Option 1, or alternatively, the constraint $u(k) \in \mathbb{U}_v$ as in Option 2, or the constraint $u(k) \in \mathbb{U}_\xi$ as in Option 3.
      iii. Compute $\hat{\mathbf{u}}^*$ by solving the MPC optimization (11).
      iv. Output the first element $u^*(k)$ of $\hat{\mathbf{u}}^*$ to control the plant Σ.
      v. Calculate and broadcast the respective performance variable.
   (b) If the transition has not been successful:
      i. The control $u(k)$ retains the last value.
      ii. Return to Step 6.a).
7. Return to Step 1.

This control procedure will be applied to the following numerical examples. The MPC and MHE optimizations have been programmed in Matlab using Yalmip toolbox.

4.3. Comments on the three performance variables

In the numerical examples below, we will study the three proposed performance variables described in Subsection 4.1 above. The result is concisely

summarized here. In the first example, a paper processing machine, all three performance variables did not improve the control performance, defined as the accumulative value of $\mathcal{J}(k)$ and $V(k)$ over time, i.e. the control performance remains unchanged with and without the constraints $\hat{\mathbf{u}} \in \mathbb{U}_b$, $u(k) \in \mathbb{U}_{V}$, or $u(k) \in \mathbb{U}_{\xi}$. In the second example with a helicopter model, the second option of performance variables ($u(k) \in \mathbb{U}_{V}$) improved the control performance. However, when the switching-over time is relatively long, only the third option with the dissipativity constraint ($u(k) \in \mathbb{U}_{\xi}$) is able to stabilize the system in this second example while the other two constraints $\hat{\mathbf{u}} \in \mathbb{U}_b$ and $u(k) \in \mathbb{U}_{V}$ in the first and second options of performance variables have not stabilized the system. The result in the third example of the wheel set with bogie is similar to those in the first example of the paper processing machine: The control performance did not improve with any of the performance variables in the DepMPC, i.e. none of the employed constraints $\hat{\mathbf{u}} \in \mathbb{U}_b$, $u(k) \in \mathbb{U}_{V}$, or $u(k) \in \mathbb{U}_{\xi}$ lead to a better control performance (defined as the accumulative value of $\mathcal{J}(k)$ and $V(k)$ over time). These results were among the main drivers for introducing the Replacement Controller in this paper.

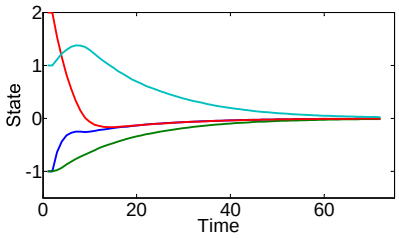
4.4. Numerical Examples

4.4.1. A Paper Processing Machine

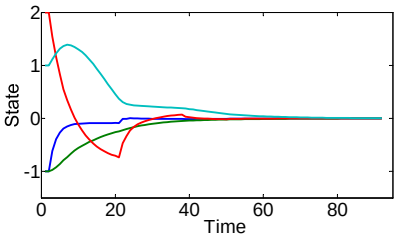
The continuous time state-space model of a paper machine [57], which is also the benchmark system in the MPC toolbox of Matlab, is

$$A = \begin{bmatrix} 1.93 & 0.27 & 0 & 0 \\ 0.94 & 0.43 & 0 & 0 \\ 0 & 0 & 0.63 & 0 \\ 0.82 & 0.78 & 0.41 & 0.42 \end{bmatrix}, \; B = \begin{bmatrix} 1.27 & 1.27 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.3 & 0.65 & 0.21 & 0.41 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

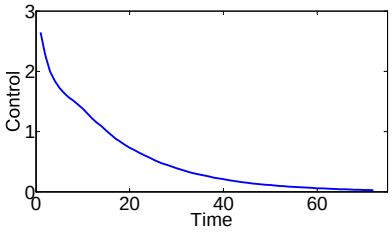
$$C = \begin{bmatrix} 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0.1 & 0 & 1.0 \end{bmatrix}.$$



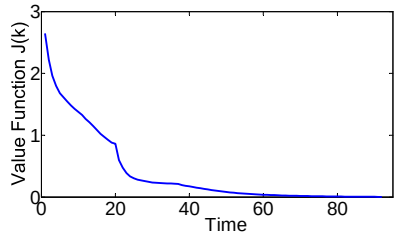
(a) State trajectories without duty-standby transitions (four states).



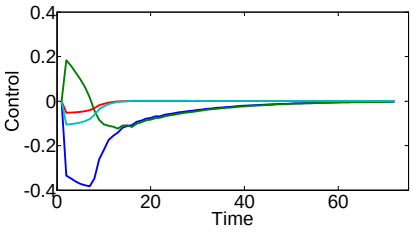
(b) State trajectories with duty-standby transitions - The transitions occur between $5-15,\ 25-35,\ 40-47,\ 50-55$ time steps.



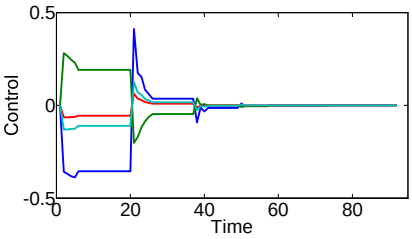
(c) Value function trajectory without duty-standby transitions.



(d) Value function trajectory with duty-standby transitions.



(e) Control trajectories without duty-standby transitions (four controls).



(f) Control trajectories with duty-standby transitions.

Figure 12: DepMPC running on the paper machine example with four duty-standby transitions in the 10 time step intervals. The MPC predictive horizon is $N = 3$. A persistent disturbance $d(k)$ has been added. Time intervals of the duty-standby transitions are between $5-15,\ 25-35,\ 40-47,\ 50-55$ time steps. The performance variables did not improve the control performance in this example of an open-loop stable plant.

The state realization matrices A, B from the model of Σ in (1) are obtained after discretizing with the sampling time of 0.27 steps. A very short predictive horizon of $N = 3$ was set for the simulation study. The state constraints are omitted to simulate the possible instability. The maximum of the control norm is $\eta = 0.4$. The optimizing value of $\mathcal{J}(k)$ is saved and used as the performance variable, to derive the constraint $\hat{\mathbf{u}} \in \mathbb{U}_b$ in the event of switching-over from the standby to the on-duty mode as explained above. Similarly, the other two constraints $u(k) \in \mathbb{U}_{_V}$ and $u(k) \in \mathbb{U}_\xi$ are also employed in this example. The time intervals for the standby-duty transitions are assumed as 10 time steps maximum in the simulation study. In particular, the transitions occur between $5-15,\ 25-35,\ 40-47$, and $50-55$ time steps. These are long durations compared to the predictive horizon of only 3 time steps. The resultant system stability is achievable with the control algorithm of DepMPC systems in this section, as illustrated by the state and control trajectories in Figure 12. The top two sub-figures (a) and (b) are the state trajectories of DepMPC without and with duty-standby transition events in the same order. The corresponding trajectories of $\mathcal{J}(k)$ are in sub-figures (c) and (d). The last two sub-figures (e) and (f) are the corresponding control trajectories of DepMPC without and with duty-standby transition events.

However, the result from this simulation study with over 100 cases of different standby-duty transitions led to a conclusion that the three presented performance variables did not improve the control performance of the DepMPC in this particular example. This can be explained by looking at the trajectories of $\mathcal{J}(k)$ in sub-figures 12(c) and (d) above, which are both monotonically decreasing, even during the standby-duty transition periods (when the last value of $u(k)$ has been used). This means, the performance variable is not needed for this particular kind of plant which is an open-loop stable system. Nevertheless, the proposed performance variables improved the control performance and helped to stabilize the system (open-loop unstable) in the second example below.

4.4.2. A Helicopter

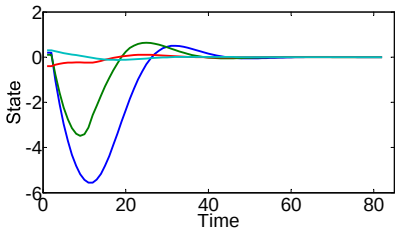
The continuous time state-space model of a helicopter [58] is

$$A = \begin{bmatrix} -0.02 & 0.005 & 2.4 & -32 \\ -0.14 & 0.44 & -1.3 & -30 \\ 0 & 0.018 & -1.6 & 1.2 \\ 0 & 0 & 1.0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0.14 & -0.12 \\ 0.36 & -8.6 \\ 0.35 & 0.009 \\ 0 & 0 \end{bmatrix}.$$

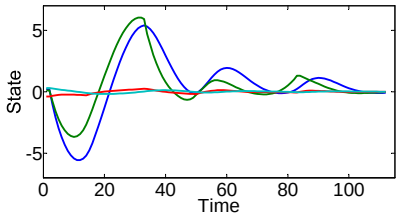
The state realization matrices A, B from the model of Σ in (1) are obtained after discretizing with the sampling time of 0.125 steps. A predictive horizon of $N = 12$ was set for the simulation study. The closed-loop system did not stabilize with $N < 8$ in this example. The state constraints are omitted to simulate the possible instability. The duty-standby transitions occur between $5-10,\ 25-30,\ 40-47,\ 70-77$ time steps. This is comparable with the minimum predictive horizon $N = 7$ for the stabilizing cases. Both the stability and control performance are achievable with the DepMPC using the control procedure in Subsection 4.2, except that the two steps (2) and (6a.i) are omitted, i.e. without MHE. The trajectories are reported in Figure 13. The detailed performance evaluations are given in Table 1.

The constraint $\hat{\mathbf{u}} \in \mathbb{U}_b$ (using the first option of performance variable) improved the control performance of $\sum_k(\mathcal{J}(k))$ for over 110 simulation steps by around 5% with $\beta = 1.6$, but did not improve the control performance of $\sum_k(V(k))$, as shown in the third column of Table 1. The second alternative of using $V(k)$ as the performance variable (with the constraint $u(k) \in \mathbb{U}_{_V}$) showed performance improvements of around 8% of $\sum_k(\mathcal{J}(k))$, and around 2% of $\sum_k(V(k))$ with $\gamma = 0.88$, as shown in the fourth column of Table 1. The third alternative with the dissipativity constraint(the constraint $u(k) \in \mathbb{U}_\xi$ is used) only slightly improved the control performance of $\sum_k(V(k))$, but not $\sum_k(\mathcal{J}(k))$, as shown in the fifth column of Table 1.

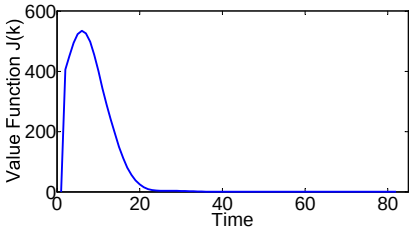
In this example, the system became unstable when the standby-duty transition intervals are greater than 7 time steps. The first two options using $\hat{\mathbf{u}} \in \mathbb{U}_b$ and $u(k) \in \mathbb{U}_{_V}$ did not stabilize the system when the standby-duty transitions



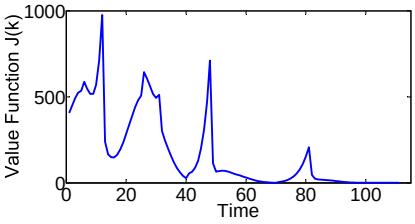
(a) State trajectories without duty-standby transitions (four states).



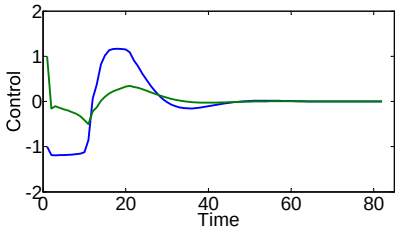
(b) State trajectories with duty-standby transitions - The transitions are between $5-10,\ 25-30,\ 40-47,\ 70-80$ time steps.



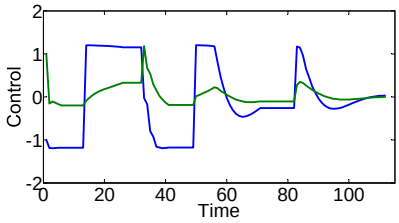
(c) Value function trajectory without duty-standby transitions.



(d) Value function trajectory with duty-standby transitions.



(e) Control trajectories without duty-standby transitions (two controls).



(f) Control trajectories with duty-standby transitions.

Figure 13: DepMPC running on the helicopter example with four duty-standby transitions in the 10 time step intervals. The MPC predictive horizon is $N = 12$. Time intervals of the duty-standby transitions are between $5-10,\ 25-30,\ 40-47,\ 70-80$ time steps. The constraint $\hat{\mathbf{u}} \in \mathbb{U}_b$ improved the performance of $\sum_k(\mathcal{J}(k))$ by around 5% with $\beta = 1.6$. The constraint of the form $V(k) < \gamma \times V(k_s)$ has improved the performance of $\sum_k(\mathcal{J}(k))$ by around 8% with $\gamma = 0.88$.

are longer than 7 time steps (however, these transitions can take a longer time when the states near zero, e.g. 10 time steps when $k > 50$ without affecting the system stability). The third option with dissipativity constraint (using $u(k) \in \mathbb{U}_\xi$) stabilized the system in the case of longer than 7 time-step standby-duty transitions, as studied with the intervals $5-12,\ 25-35,\ 40-50,\ 70-80$ here (the unstable trajectories are not printed out).

Performance Index	No Constraint	$\hat{\mathbf{u}} \in \mathbb{U}_b$	$u(k) \in \mathbb{U}_V$	$u(k) \in \mathbb{U}_\xi$
$\sum_k(\mathcal{J}(k))$	$4,9373.0$	$4,7707.0$	$4,5441.0$	$4,9378.0$
$\sum_k(V(k))$	$7,0311 \times 10^3$	$7,1057 \times 10^3$	$6,9914 \times 10^3$	$7,0309 \times 10^3$

Table 1: Performance evaluations: Two performance indices are used in the evaluation. The first performance index $\sum_k(\mathcal{J}(k))$ is the accumulation of weighted $\|x(k)\|^2$ and weighted $\|u(k)\|^2$. The second performance index $\sum_k(V(k))$ is the accumulation of weighted $\|x(k)\|^2$. The performance variable is not used in "No Constraint" cases. The constraints $\hat{\mathbf{u}} \in \mathbb{U}_b$, $u(k) \in \mathbb{U}_V$, and $u(k) \in \mathbb{U}_\xi$ correspond to the performance variables of $J(k)$ (option 1), $V(k)$ (option 2), and $\xi(k)$ (option 3), respectively.

4.4.3. A Railway Vehicle Wheel Set with Bogie

A typical model of the integrated tilting bolster and active lateral secondary suspension in railway vehicles, presented in [59], is simulated in this third example.

The continuous time state-space model is $A = \begin{bmatrix} A_u \\ 0 \mid I \end{bmatrix}$, $B = \begin{bmatrix} B_u \\ 0 \end{bmatrix}$, where

$$A_u = \begin{bmatrix} -3.423 & 0 & 0 & 0 & 3.423 & 4.193 & 0 & -.44 & 0 & 0 & 0 & .44 & .53 & 0 \\ 0 & -3.423 & 0 & 0 & 3.423 & -4.193 & 0 & 0 & -.44 & 0 & 0 & .44 & -.53 & 0 \\ 0 & 0 & -1.880 & 0 & 0 & 1.880 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1.880 & 0 & 1.880 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1.355 & 1.354 & 0 & 0 & -77.5 & 0 & 77.5 & .35 & .35 & 0 & 0 & -.58 & 0 & .58 \\ 1.803 & -1.803 & .45 & .45 & 0 & -3.181 & 0 & .46 & -.46 & .27 & .27 & 0 & -.11 & 0 \\ 0 & 0 & 0 & 0 & .37 & 0 & -.37 & 0 & 0 & 0 & 0 & .06 & 0 & -.06 \end{bmatrix},$$

$$B_u = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & .001880 & \\ 0 & 1 & 0 & .00188 \\ 0 & 0 & 0 & 0 \\ -1 & -1 & -.00045 & -.00045 \end{bmatrix}.$$

The state realization matrices A, B from the model of Σ in (1) are obtained after discretizing with the sampling time of 0.225 steps. A predictive horizon of $N = 5$ was set for the simulation study. The state constraints are omitted to simulate the possible instability. The maximum of the control norm is $\eta = 0.2$. The duty-standby transitions are between $5-12,\ 25-30,\ 40-47$ time steps. The trajectories with DepMPC are reported in Figure 14. The detailed performance evaluations are given in Table 2.

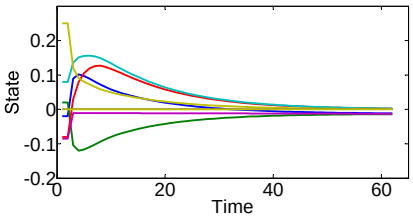
The first and third options of performance variables did not improve the control performance in this example. The second option with the constraint $u(k) \in \mathbb{U}_V$ slightly improved both control performances of $\sum_k(V(k))$ and $\sum_k(\mathcal{J}(k))$ by around 1%, as shown in the fourth column of Table 2.

Performance Index	No Constraint	$\hat{\mathbf{u}} \in \mathbb{U}_b$	$u(k) \in \mathbb{U}_V$	$u(k) \in \mathbb{U}_\xi$
$\sum_k(\mathcal{J}(k))$	0.2532	0.2532	0.2509	0.2532
$\sum_k(V(k))$	147.7547	147.7542	147.7412	147.7547

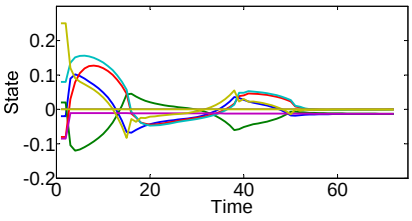
Table 2: Performance evaluations: Two performance indices are used in the evaluation. The first performance index $\sum_k(\mathcal{J}(k))$ is the accumulation of weighted $\|x(k)\|^2$ and weighted $\|u(k)\|^2$. The second performance index $\sum_k(V(k))$ is the accumulation of weighted $\|x(k)\|^2$. The performance variable is not used in "No Constraint" cases. The constraints $\hat{\mathbf{u}} \in \mathbb{U}_b$, $u(k) \in \mathbb{U}_V$, and $u(k) \in \mathbb{U}_\xi$ correspond to the performance variables of $J(k)$ (option 1), $V(k)$ (option 2), and $\xi(k)$ (option 3), respectively.

4.5. Summary of Simulation Studies without Replacement Controller

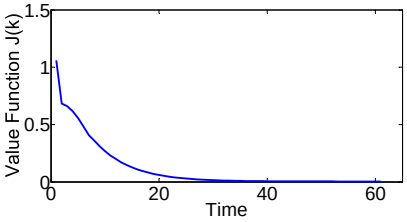
In summary, the three examples in this simulation study revealed maximum improvements of 10% of the control performance when the performance variables are used. The control performances are evaluated using two performance indices: $\sum_k(\mathcal{J}(k))$ and $\sum_k(V(k))$ over the simulation time. Moreover, each of



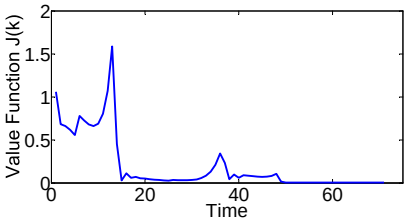
(a) State trajectories without duty-standby transitions (fourteen states).



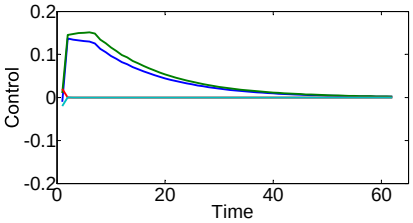
(b) State trajectories with duty-standby transitions- The transitions are between $5-12,\ 25-35,\ 40-47$ time steps.



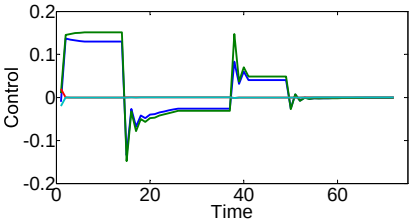
(c) Value function trajectory without duty-standby transitions.



(d) Value function trajectory with duty-standby transitions.



(e) Control trajectories without duty-standby transitions (four controls).



(f) Control trajectories with duty-standby transitions.

Figure 14: DepMPC running on the railway wheel set example with four duty-standby transitions in the 10 time step intervals. The MPC predictive horizon is $N = 5$. Time intervals of the duty-standby transitions are between $5-12,\ 25-35,\ 40-47$ time steps.

the three performance variables exhibits its pros and cons as can be seen in the case studies from the three examples above. Given the trade-off in using the performance variable for each of the three options, the users will have the flexibility in choosing a preferred performance variable that fits their application. It is worth noting here that the three performance variables in this work are not the only options. There are other choices from the knowledge of control theory that the users may develop and employ in new applications.

Since the control performance is among the key factors for the success of DepMPC in actual implementations, the Replacement Controller (RC) is introduced and added to the system architecture in the next section.

5. Replacement Controller

5.1. Replacement Controller - Analysis

From the above simulation study, we observed that an additional controller should be installed for the DepMPC to improve the control performance when the duty-standby transitions take a longer time to complete. It is noted that the DepMPC becomes an open-loop system, and with the control takes the last available value during this long transition periods.

In practice, the token-like protocol with the state machine in Figure 8 may assume several sampling times before requesting a duty-standby switch-over. This is well recognized in the area of fault-tolerant computer systems [5]. Therefore, long duration duty-standby transitions are likely the reality, even when the TSN networks are used. We may think that the solutions for networked-control systems [60, 61] can be employed for DepMPC. However, the networked-control system is different compared to the DepMPC; so the solutions are not suitable for the DepMPC, predominantly for two reasons: (1) The duty-standby transitions do not often occur at the frequency of the intermittent communication losses. And as a consequence, the solutions for a networked-control system may become too conservative for the DepMPC, or is simply not applicable. (2) The

duty-standby transitions usually need a longer time than the maximum allowable transmission interval (MATI) in a networked-control system [5]. Therefore, the closed-loop system stability may not be achievable because the solutions for the networked control systems are too conservative to be implementable.

The additional controller presented in this section to be installed for the DepMPC is termed "Replacement Controller" (RC). The algorithm of the RC here will use the monotonically deceasing property of the Lyapunov function $V_k = x_k^T P x_k$ to derive a state-feedback control law to be used during the duty-standby transition time (instead of the last available value). The RC will be used only when the Lyapunov function starts increasing until its value exceeds a predefined threshold that is relative to its saved value prior to the duty-standby transition. That means the RC will become active and control the plant if the Lyapunov function is larger than the predefined threshold. This is only applicable during a duty-standby transition. Thereafter, the RC is replaced by the MPC once a standby controller is successfully activated into the duty mode and connect to the plant. Since the RC is a static controller, the implementation is straight forward and implementable.

Of course, implementing a full MPC algorithm for the RC is a possible solution, but we believe that solution is impractical, for these two reasons: (1) The RC only controls the plant during the duty-standby transitions if the Lyapunov function displays large increments during the transitions. This means, the RC will work in short time intervals only. (2) The processor of this RC should be dedicated for each plant under control, and for monitoring the status of the other redundant controllers. So the resource for this RC is not abundant or readily available, but should be additionally installed (or configured) for each DepMPC. As a result, the RC control design should be as simple as possible such that the solution is affordable and implementable. A simple design for the RC is thus selected and presented in the next subsection. The positive result from this simple design resulted because the control performances in the numerical examples in the simulation study have all been improved with the (simple)

RC developed.

5.2. Replacement Controller - Design development

Consider the state feedback control law of the form $u_k = Kx_k$. Then the inequality of $V_{k+1} < V_k \Leftrightarrow (x_k^T A^T + x_k^T K^T B^T)P(Ax_k + BKx_k) < x_k^T P x_k$ holds for every x_k if the matrix inequality $(A^T + K^T B^T)P(A + BK) - P \prec 0$ is fulfilled. The following linear matrix inequality (LMI) will thus be used here:

$$\begin{bmatrix} -P + \lambda I & (A^T + K^T B^T)P \\ P(A + BK) & -I \end{bmatrix} \preccurlyeq 0, \ \lambda > 0, \tag{13}$$

in which the matrix P is pre-computed from the Riccati equation with the two weighting matrices $\mathcal{Q}$ and $\mathcal{R}$ of the MPC cost function. Then, the LMI condition below is applied to ensure that $u_k^T u_k \leq \eta$ for $u_k = Kx_k$:

$$\begin{bmatrix} -\eta & x_k^T K^T \\ Kx_k & -I \end{bmatrix} \prec 0. \tag{14}$$

And similarly for the iterative state constraint satisfaction:

$$\begin{bmatrix} -\rho & x_k^T (A + BK)^T \\ (A + BK)x_k & -I \end{bmatrix} \prec 0. \tag{15}$$

Combining these three LMIs, the state feedback gain K can be determined by solving the LMI optimization of

$$\begin{aligned} &\min_{K} \quad \lambda \\ &\text{s.t.} \quad (13), (14), (15). \end{aligned} \tag{16}$$

We will use the optimizing gain K from the optimization problem (16) in the state-feedback control law $u_k = Kx_k$ of the RC to manipulate the plant during the standby-duty transitions if $V_{k+1} - V_k > \theta$, with a pre-defined $\theta > 0$ (set by the user).

The stability condition for the DepMPC and RC is summarized in the Proposition 2 below.

Proposition 2. *Consider Σ (1) with an MPC (4) for each controller of the DepMPC, the state machine in Figure 8 for each controller, a replacement controller $u(k) = Kx(k)$, and a real-value non-negative function $V\big(x(k)\big) = x(k)^T Px(k)$, $P \succ 0$. Select the threshold $\theta > 0$. Assume that for each finite $x(0) \in \mathbb{R}^n$, the following conditions hold for all $k > 0$:*

1. *The state machine is executed by each redundant controller;*
2. *The value of $V\big(x(k)\big)$ is saved independently;*
3. *If the on-duty MPC is connected to the plant, the control $u(k) = \kappa_N(x(k)) = u^*(0, x(k))$ from the MPC (4) is feasible and applied to control Σ (1);*
4. *In the events of on-duty MPC disconnection to the plant or failure, and the token has not been caught by a standby controller, if $V\big(x(k)\big) - V\big(x(k-1)\big) > \theta$, the control $u(k) = Kx(k)$ is applied to control Σ (1), where K is a solution to (16);*

Then $x(k)$ remains finite and $\|x(k)\| \to 0$ as $k \to \infty$.

Proof

This is a direct result of the convergence of $x(k)$ with the feasible MPC (4) and $V\big(x(k)\big)$ from the two conditions (3) and (4) for both cases of the on-duty MPC connecting and disconnecting to Σ (1). ■

The next description is the modified control procedure with RC.

5.3. Replacement Controller - Modified control algorithm

The control algorithm in Subsection 4.2 is modified to implement the control law of the RC derived above. The modified algorithm in the pseudo form is as follows:

At each time instant k,

1. Obtain the measurement output $y(k)$ from the respective transmitters.
2. Estimate $\hat{x}(k)$ by solving the MHE optimization (10).
3. Compute $\hat{\mathbf{u}}^*$ by solving the MPC optimization (4).

4. Output the first element $u^*(k)$ of $\hat{\mathbf{u}}^*$ to control the plant Σ.
5. Save the values of $u(k)$, $\hat{d}(k)$, $\hat{v}(k)$ and $\mathcal{J}(k)$.
6. If a standby-duty transition request is received,
   (a) Verify the standby-duty transitions of redundant controllers. If a transition has been successful:
      i. Deactivate the RC if it is currently active (**modified**).
      ii. Estimate $\hat{x}(k = k_s + \delta + 1)$ by solving the MHE optimization (10).
      iii. Formulate the constraint $\hat{\mathbf{u}} \in \mathbb{U}_b$ with $\mathcal{J}(k_s)$ as in Option 1, or alternatively, the constraint w.r.t $u(k)$ in (12) as in Option 2, or the dissipativity constraint in Option 3 [51].
      iv. Compute $\hat{\mathbf{u}}^*$ by solving the MPC optimization (11).
      v. Output the first element $u^*(k)$ of $\hat{\mathbf{u}}^*$ to control the plant Σ.
      vi. Calculate and broadcast the respective performance variable.
   (b) If the transition has not been successful: (**modified with Replacement Controller**)
      i. Monitor the value of the Lyapunov function $V(k)$. If $V(k+1) - V(k) > \theta$ (θ is chosen from the simulation study - pre-defined),
      ii. Determine the feedback gain K using (16).
      iii. Activate the RC with the control law $u(k) = Kx(k)$.
      iv. Return to Step 6.a).
7. Return to Step 1.

5.4. Replacement Controller - Modified block diagram

The DepMPC block diagrams are now modified with an RC as shown in Figures 15 and 16. Assuming low latencies are offered by the TSN networks, the presented RC will be an implementable approach to improve the control performances. The RC resumes control during the overly long period of control transfer (i.e. when the switching-over time is too long); in this manner, the RC mitigates the control performance degradation during the transition periods.

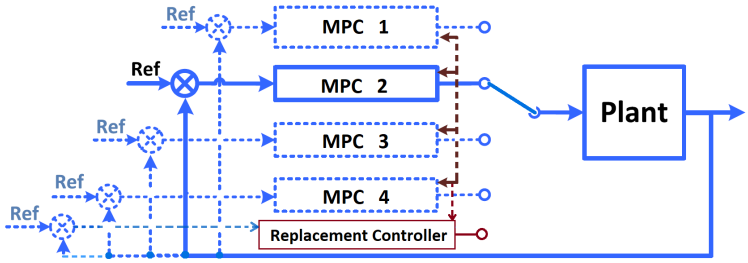


Figure 15: The DepMPC is modified with a Replacement Controller - Block diagram.

It is worth to note here that, during the switching-over event, the control system is an opened loop and the manipulation takes the last available value of the control from the MPC. According to the modified control procedure, the RC will have to check and verify the status of the token during the standby-duty transitions. The block diagrams in Figures 15 and 16 have been modified, the dotted lines (red color in Figure 15) represent the status signals to be used by the RC. The RC checks if an on-duty controller is connected to the plant and healthy, and if a standby controller has successfully caught the token and taken the duty role. If there are no active on-duty controller connecting to the plant, the value of the storage function $x(k)^T Px(k)$ will then be monitored and verified against a pre-defined threshold. The control law of the RC will only be computed and applied to the plant if the two conditions above do not hold.

5.5. Replacement Controller - Simulation studies

The modified control algorithm with the RC is used in this simulation study. The results from the simulation studies for the first and second examples (in the previous section) with the RCs are as follows: There are significant improvements in the control performances of the standby-duty transitions when RC is employed to the DepMPC. The trajectories of the DepMPC with RCs are shown in Figures 17 and 18 in the first and second examples, respectively.

The accumulative value of $V(k)$ over the simulation time (the performance index) was improved with over 35% in the helicopter example, as shown in Figure 17, and over 12% in the paper machine example, as shown in Figure

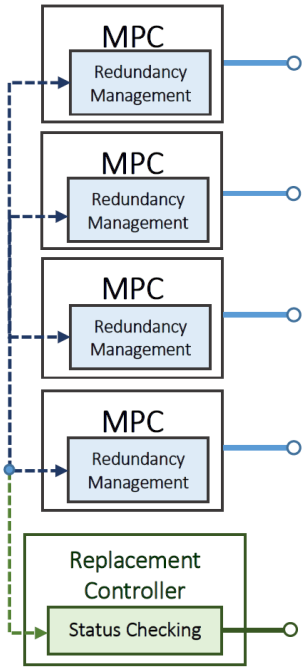
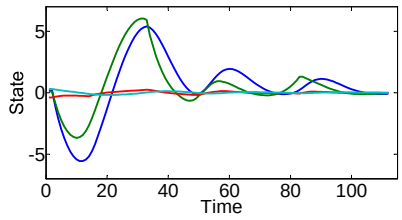


Figure 16: The status checking in a Replacement Controller.

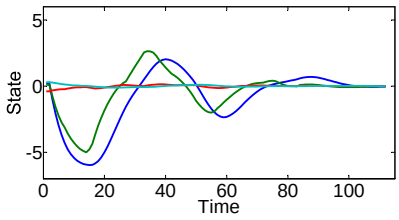
18. However, in the third example of a railway wheel set with bogie, the RC was not activated since the Lyapunov function is not increasing during the standby-duty transitions. This simulation study shows the RC capability for improving the control performance in long switching-over durations. Thus, the RC plays an important role in maintaining the desirable control performance of the DepMPC, and consequently, offers a feasible solution for the fault-tolerance of DepMPC.

5.6. Future Directions

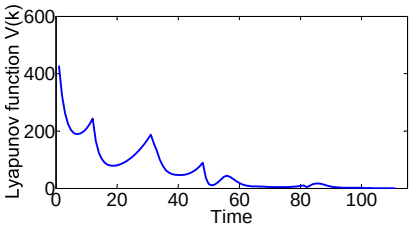
The potential future work includes hardware and software implementations of the presented DepMPC together with redundancy self-management using the developed state machine in test-beds. The applications shall be tested for continuous and hybrid plants to show the system operability. The test oracles design will allow the DepMPC to be verified with various open-loop stable



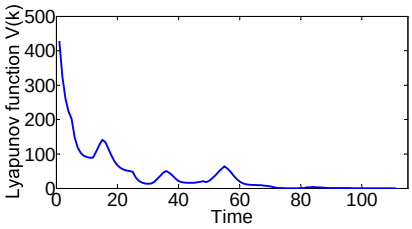
(a) State trajectories without replacement controller.



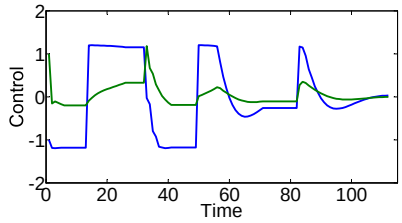
(b) State trajectories with replacement controller.



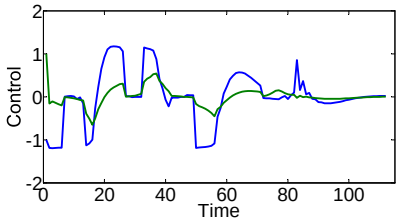
(c) Luapunov function trend without replacement controller.



(d) Luapunov function trend with replacement controller.

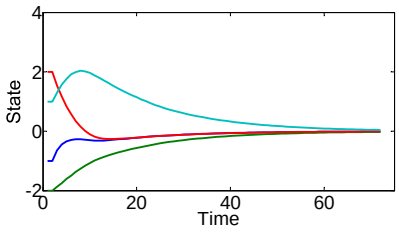


(e) Control trajectories without replacement controller.

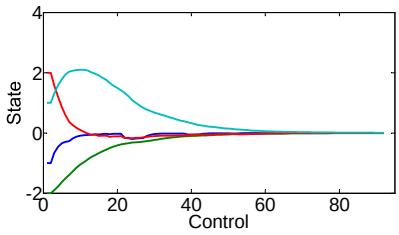


(f) Control trajectories with replacement controller.

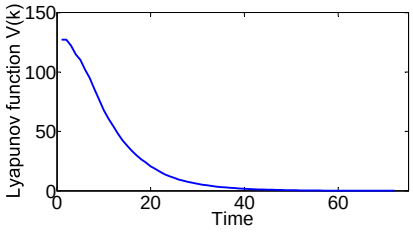
Figure 17: DepMPC and Replacement Controller for the helicopter model. The DepMPC has four duty-standby transitions between $5-10,\ 25-30,\ 40-47,\ 70-80$ time steps. The Replacement Controller improved the control performance of $\sum_k(V(k))$ by around 35%.



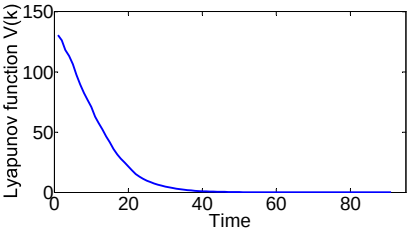
(a) State trajectories without duty-standby transitions.



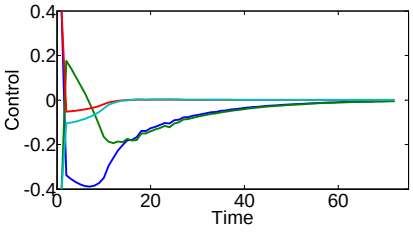
(b) State trajectories with replacement controller.



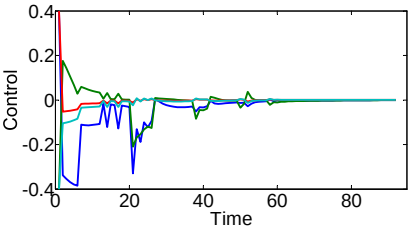
(c) Luapunov function trend without duty-standby transitions.



(d) Luapunov function trend with replacement controller.



(e) Control trajectories without duty-standby transitions.



(f) Control trajectories with replacement controller.

Figure 18: DepMPC and Replacement Controller for the paper machine. The DepMPC has four duty-standby transitions between $5-15,\ 25-35,\ 40-47,\ 50-55$ time steps. The Replacement Controller improved the control performance of $\sum_k (V(k))$ by around 12%.

and unstable plants to prove the advantages and operability of the self-reliant DepMPC and its fault tolerances.

6. Conclusion

A dependable model predictive control system (DepMPC) with replacement controller (RC), which enables the smooth switching of controller and redundancy self-management, has been presented in this paper. Numerical simulation studies with three linear system models (having control constraints) have illustrated the attainable operability and control performance of the DepMPC. The standout results from this work include the role of an RC and the DepMPC algorithm. The companion RC has improved the control performance of the DepMPC whose redundant MPC controllers are self-reliant in this approach.

The proposed approach of DepMPC and RC are applied to OT systems with the emerging standard IEC/IEEE 60802 with time-sensitive networking (TSN). They provide a promising solution for enhancing the reliability of a multi-variable control system by self-managing the redundant controllers while maintaining smooth transfers between the standby and duty modes. The hard real-time capability with TSN will ensure that the on-duty MPC controller is guaranteed to be available and connected to the plant under control within a deterministic period.

The limitations of the presented DepMPC with a software-defined architecture lie in the possible longer switching-over durations due to network traffic or communication breakdown compared to the RCS approach with a hardware-defined architecture. By the same reason, there are also chances that the system becomes an opened loop without an on-duty controller connecting to the plant, which may profoundly affect the control performance for fast dynamic systems. Thanks to the TSN network, the issue of a switching-over long duration may be improved or even eliminated, to avoid possible control degradations. We have not considered the voting requirement in this work, and the data integrity of a DepMPC will be lower than using voting.

References

[1] J. Farkas, L. L. Bello, C. Gunther, Time-sensitive networking standards, IEEE Communication Standards Magazine (2018) 20–21.

[2] T. Tran, Q. P. Ha, Dependable control systems with Internet of Things, ISA Transactions 59 (2015) 303–313.

[3] IEEE 802 Working Groups, IEC/IEEE 60802 TSN Profile for Industrial Automation - Use Cases V1.3, http://www.ieee802.org/1/files/public/docs2018/new-industrial-use-cases-0318-v04.pdf, [Online; accessed 11-Sep-2019] (2019).

[4] IFAC, IFAC workshops in Dependable Control of Discrete Systems, http://webserv.lurpa.ens-cachan.fr/dcds_series/, [Online; accessed 20-Jan-2020] (1st workshop in 2007).

[5] M. L. Schooman, Reliability of Computer Systems and Networks: Fault Tolerance, Analysis and Design, Wiley-Interscience, 2001.

[6] K. J. Åstrom, R. M. Murray, Feedback Systems: An Introduction for Scientists and Engineers, Princeton University Press, 2010.

[7] J. Lunze, Feedback Control of Large Scale Systems, Prentice Hall, 1992.

[8] D. D. Siljak, Decentralized Control of Complex Systems, Academic Press, 1991.

[9] K. J. Åstrom, M. Blanke, A. Isidori, W. Schaufelgerger, R. Sanz, Control of Complex Systems, Springer-Verlag, 2001.

[10] R. Scattolini, Architectures for distributed and hierarchical model predictive control - A Review, Journal of Process Control 19 (2009) 723–731.

[11] K. J. Åstrom, P. R. Kumar, Control: A perspective, Automatica 50 (1) (2014) 3–43.

[12] R. M. Murray, Control in an Information Rich World: Report of the Panel on Future Directions in Control, Dynamics, and Systems, SIAM, 2003.

[13] M. G. Singh, A. Titli, Systems: Decomposition, Optimisation and Control, Franklin Book Company, 1978.

[14] E. Camponogara, D. Jia, B. H. Krogh, S. Talukdar, Distributed model predictive control, IEEE Control System Magazine 1 (2002) 44–52.

[15] W. B. Dunbar, R. M. Murray, Distributed receding horizon control with application to multi-vehicle formation stabilization, Automatica 42 (4) (2006) 1549–558.

[16] T. Keviczky, F. Borrelli, G. J. Balas, Decentralized receding horizon control for large scale dynamically decoupled systems, Automatica 42 (12) (2006) 2105–2115.

[17] L. Magni, R. Scattolini, Stabilizing decentralized predictive control of nonlinear systems, Automatica 42 (7) (2006) 1231–1236.

[18] L. J., A. Momeni, A. Aghdam, A model predictive decentralized control scheme with reduced communication requirement for spacecraft formation, IEEE Transactionson Control System Technology 16 (2) (2008) 268–278.

[19] J. B. Rawlings, B. T. Stewart, Coordinating multiple optimization-based controllers: New opportunities and challenges, Journal of Process Control 18 (2008) 839–845.

[20] J. M. Maestre, R. R. Negenborn, Distributed Model Predictive Control – Made Easy, Springer Science & Business Media, 2013.

[21] S. V. Rakovic, R. H. Gielen, Positively invariant families of sets for interconnected and time-delay discrete-time systems, SIAM Journal on Control and Optimization 54 (4) (2014) 2261–2283.

[22] M. Zhao, B. Ding, Distributed model predictive control for constrained nonlinear systems with decoupled local dynamics, ISA Transactions 55 (2015) 1–12.

[23] G. Xiao, F. Liu, Distributed fault-tolerant model predictive control for intermittent fault: A cooperative way, ISA Transactions 89 (2019) 113–121.

[24] M.-R. Chen, G.-Q. Zeng, X.-Q. Xie, Population extremal optimization-based extended distributed model predictive load frequency control of multi-area interconnected power systems, Journal of the Franklin Institute 355 (17) (2018) 8266–8295.

[25] Y. Zheng, J. Zhou, Y. Xu, Y. Zhang, Z. Qian, A distributed model predictive control based load frequency control scheme for multi-area interconnected power system using discrete-time Laguerre functions, ISA Transactions 68 (2017) 127–140.

[26] Y. Liu, J. M. Montenbruck, D. Zelazo, M. Odelga, S. Rajappa, H. H. Bultho, F. Allgöwer, A. Zell, Distributed control approach to formation balancing and maneuvering of multiple multirotor UAVs, IEEE Transactions on Robotics 34 (4) (2018) 870–882.

[27] C. R. Cutler, B. L. Ramaker, Dynamic matrix control – a computer control algorithm, In Proceedings of the Joint Automatic Control Conference (1980).

[28] C. E. García, A. M. Morshedi, Quadratic programming solution of dynamic matrix control (QDMC), Chemical Engineering Communications 46 (1986) 73–87.

[29] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, E. A. Theodorou, Information theoretic model predictive control: Theory and applications to autonomous driving, IEEE Transactions on Robotics 34 (6) (2018) 1630–1622.

[30] S. J. Qin, T. A. Badgwell, An overview of industrial model predictive control technology, in: K. J. C., G. C. E., B. Carnahan (Eds.), AIChE Symposium Series: Fifth International Conference on Chemical Process Control, AIChE press Edition, Vol. 316, 1997, pp. 232–256.

[31] S. J. Qin, T. A. Badgwell, A survey of industrial model predictive control technology, Control Engineering Practice 11 (2003) 733–764.

[32] D. W. Clarke, C. Mohtadi, P. S. Tuffs, Generalized predictive control Part I - The basic algorithm, Automatica 23 (2) (1987) 137–148.

[33] D. M. Prett, C. E. Garcia, Fundamental Process Control, Butterworths, 1988.

[34] A. Bemporad, M. Morari, Control of systems integrating logic, dynamics, and constraints, Automatica 35 (1999) 407–427.

[35] D. Q. Mayne, J. B. Rawlings, C. V. Rao, P. O. M. Scokaert, Constraint predictive control: Stability and optimality, Automatica 36 (2000) 789–814.

[36] F. Allgöwer, A. Zheng, Nonlinear Model Predictive Control, Birkhauser, 2000.

[37] S. L. O. Kothare, M. Morari, Contractive model predictive control for constrained nonlinear systems, IEEE Transactions on Automatic Control 45 (6) (2000) 1053–1071.

[38] B. Kouvaritakis, M. Cannon, Nonlinear Predictive Control, IEE Control Engineering Series 61, Springer, 2001.

[39] J. M. Maciejowski, Predictive Control: With Constraints, Prentice Hall, 2002.

[40] J. A. Rossiter, Model-Based Predictive Control – A Practical Approach, CRC Press, 2003.

[41] E. F. Camacho, C. Bordons, Model Predictive Control, Springer-Verlag, Springer, 2004.

[42] J. R. Leigh, Control Theory, 2^{nd} Edition, IEE Control Engineering Series 64, 2004.

[43] F. Allgöwer, R. Findeisen, L. T. Biegler, Assessment and Future Directions of Nonlinear Model Predictive Control, LNCIS Vol. 358, Springer, 2007.

[44] L. Magni, D. M. Raimondo, F. Allgöwer, NMPC: Towards New Challenging Applications, LNCIS Vol. 384, Springer, 2009.

[45] J. Rawlings, D. Mayne, Model Predictive Control: Theory and Design, Nob Hill Publishing, Wisconsin, 2009.

[46] A. Boccia, L. Grüne, K. Worthmann, Stability and feasibility of state constrained MPC without stabilizing terminal constraints, Systems and Control Letters 74 (2014) 14–21.

[47] E. Gilbert, K. Tan, Linear systems with state and control constraints: The theory and application of maximal output admissible sets, IEEE Transactions on Automatic Control 36 (9) (1991) 1008–1020.

[48] I. Kolmanovsky, E. Gilbert, Theory and computation of disturbance invariant sets for discrete-time linear systems, Mathematical Problems in Engineering: Theory, Methods and Applications 4 (1998) 317–367.

[49] C. Conte, N. R. Voellmy, M. N. Zeilinger, M. Morari, C. N. Jones, Distributed synthesis and control of constrained linear systems, Proceedings of the American Control Conference (2012) 6017–622.

[50] T. Tran, Q. P. Ha, A Quadratic Constraint Approach to Model Predictive Control of Interconnected Systems, Springer Nature, Singapore, 2018.

[51] T. Tran, J. Maciejowski, K.-V. Ling, A general dissipativity constraint for feedback system design, with emphasis on MPC, International Journal of Robust and Nonlinear Control 29 (14) (2019) 4629–4984.

[52] T. Tran, Q. P. Ha, Semi-automatic control of networked systems with non-monotonic Lyaponuv function, International Journal of Control, Early Access (2019).

[53] G. K. Befekadu, V. Gupta, P. J. Antsaklis, On reliable stabilization via rectangular dilated LMIs and dissipativity-based certifications, IEEE Transactions on Automatic Control 58 (3) (2013) 792–796.

[54] A. D. Hutcheon, D. T. Jordan, J. A. McDermid, R. H. Pierce, I. C. Wand, B. J. Jepson, High integrity software development: process and tool issues, Microprocessors and Microsystems 19 (9) (1995) 517–524.

[55] T. Tran, K.-V. Ling, J. Maciejowski, Economic model predictive control – a review, in Proceedings of the 31st ISARC, Sydney, Australia, 2014, pp. 35–42. doi:10.22260/ISARC2014/0006.

[56] C. V. Rao, J. B. Rawlings, J. H. Lee, Constrained linear state estimation – A moving horizon approach, Automatica 37 (10) (2001) 1619–1628.

[57] Y. Ying, M. Rao, Y. Sun, Bilinear control strategy for paper making process, Chemical Engineering Communications 111 (1992) 13–28.

[58] L. Greco, A. Chaillet, A. Bicchi, Exploiting packet size in uncertain nonlinear networked control systems, Automatica 48 (11) (2012) 2801–2811.

[59] R. M. Goodell, R. Zhou, A. Zolotas, Integrated tilt with active lateral secondary suspension control for high speed railway vehicles, Mechatronics 21 (2011) 1108–1122.

[60] V. Gupta, A. F. Dana, J. P. Hespanha, R. M. Murray, B. Hassibi, Data transmission over networks for estimation and control, IEEE Transactions on Automatic Control 54 (8) (2009) 1807–1819.

[61] J. P. Hespanha, A. Mesquita, Networked control systems: Estimation and control over lossy networks, in T. Samad and J. Baillieul (Eds), Encyclopedia of Systems and Control. Springer, 2013.

Appendix A - Operational Technology System

The conceptual block diagram of an Operational Technology (OT) system is given below. This is extracted from the latest version of the developing IEC/IEEE 60802 standards, Time Sensitive Networking Profile for Industrial Automation - Use Cases document [3].

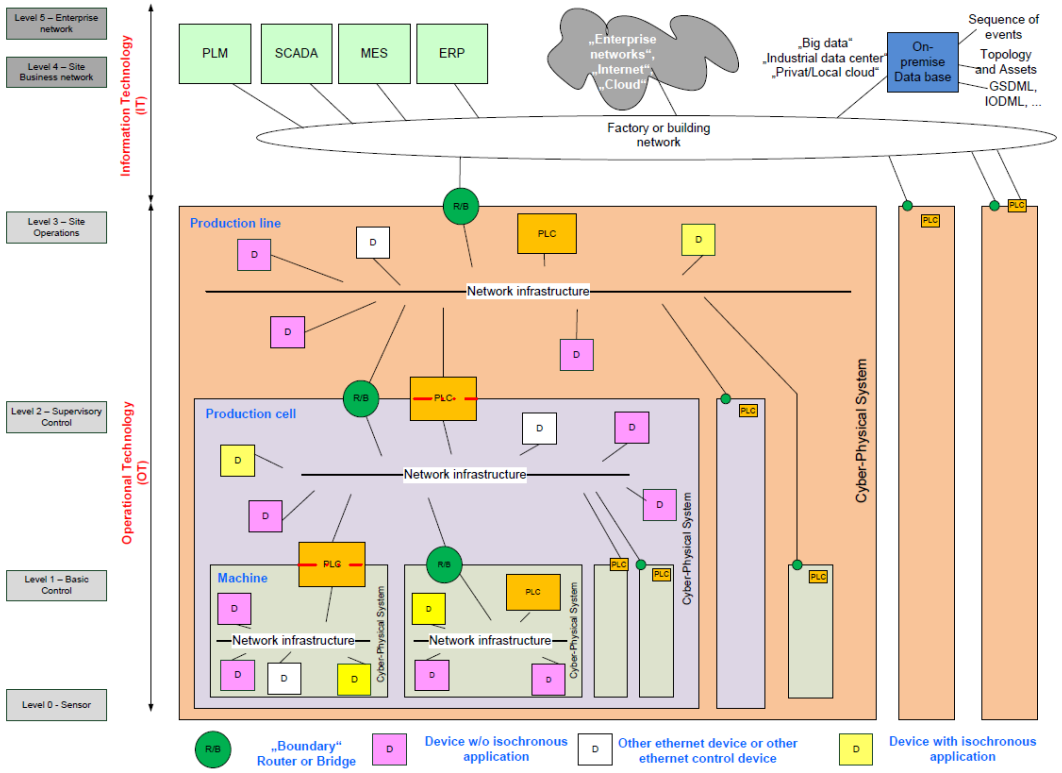


Figure 19: Hierarchical structure of industrial automation with Time Sensitive Networking – IEC/IEEE 60802 [3], fig. 1.

Figure

Reliability Enhancement with Dependable Model Predictive Control FIGURES

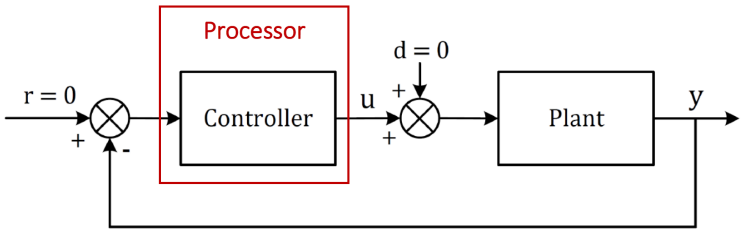


Figure 1: A feedback control loop - Block diagram.

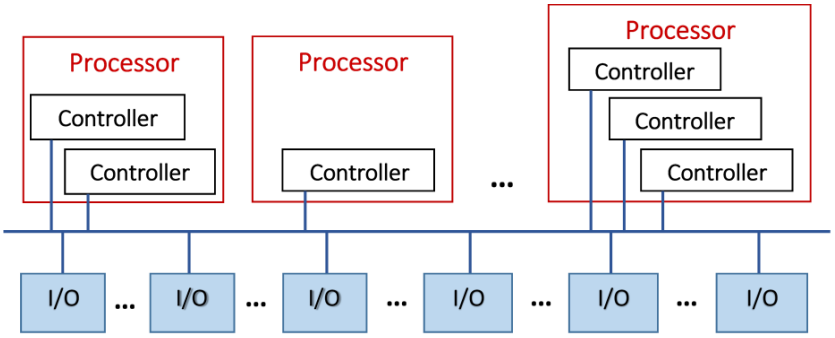


Figure 2: Logical communication and connectivity in an OT system.

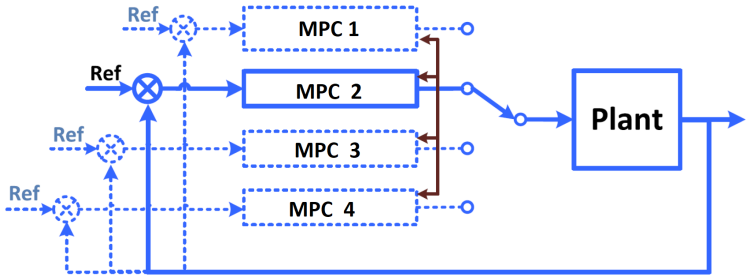


Figure 3: A Dependable Model Predictive Control system (DepMPC).

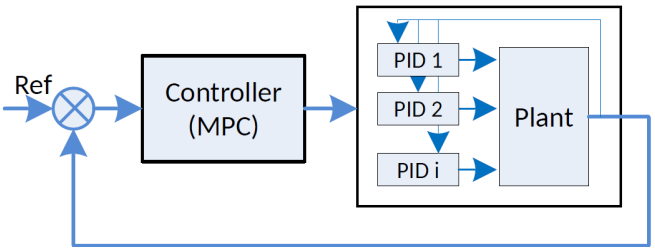
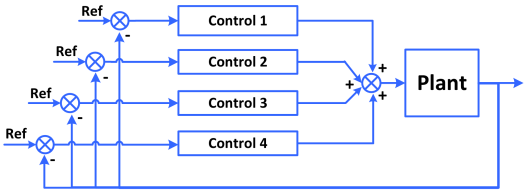
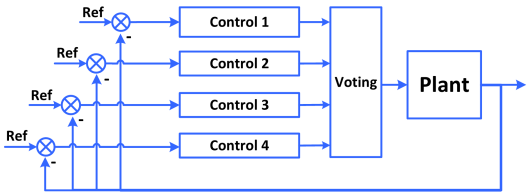


Figure 4: MPC provides set-points to PIDs.

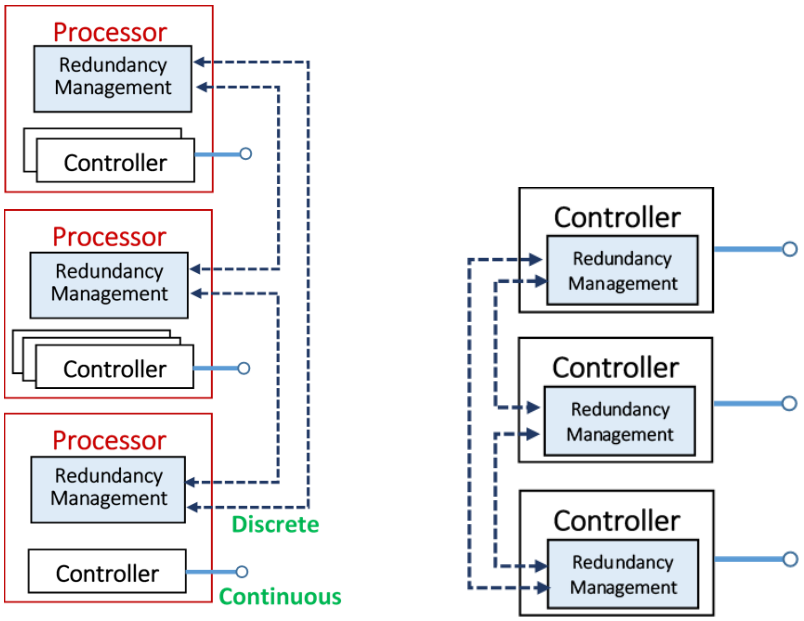


(a) With control summation.



(b) With control voting.

Figure 5: Reliable Control Systems



(a) Separated redundancy management. (b) Redundancy self-management.

Figure 6: Merging redundancy management and controller

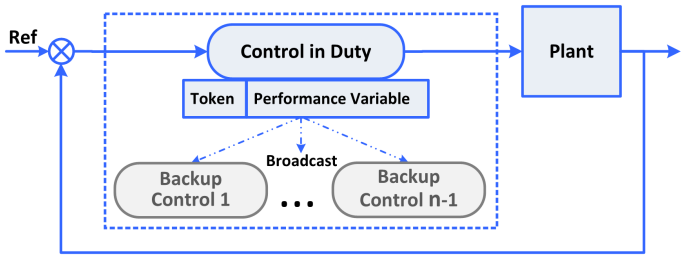


Figure 7: Dependable Model Predictive Control (DepMPC) with $n\times$redundancy. The on-duty controller broadcasts a single token and a single performance variable to the standby controllers. The on-duty controller keeps the token on a first-in-failure-out basis. The performance variable is broadcasted to the standby controllers whose responsibility is to use this performance variable to maintain the control performance of the DepMPC after a failure incident.

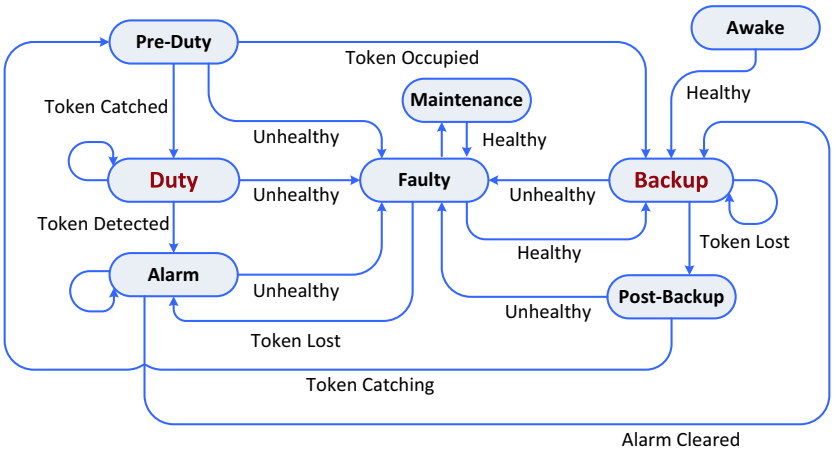


Figure 8: The proposed state machine for managing the redundant MPC controllers of a DepMPC. The block "Backup" refers to the "standby" mode.

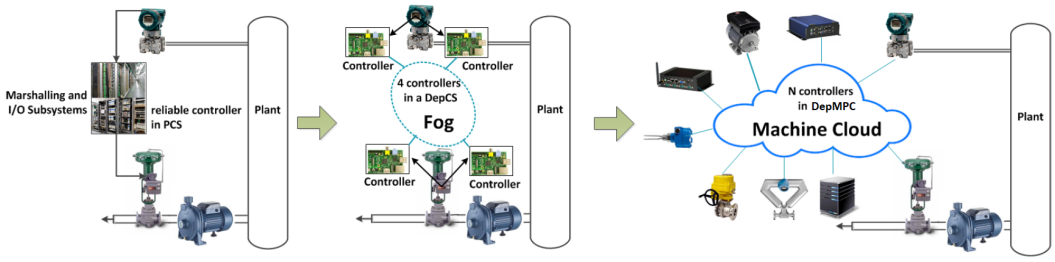


Figure 9: The controllers are implemented on the internal platforms of the transmitter and actuator in a Dependable Control System (DepCS) while the controllers are implemented on the processors of an OT system in a Dependable Model Predictive Control (DepMPC), external to the transmitters and actuators. In terms of fault tolerances, the DepCS is 4×redundant while the DepMPC can be n×redundant, typically with $n \geqslant 4$.

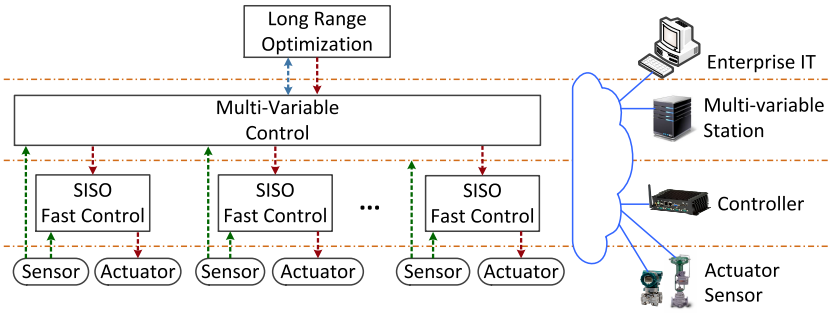


Figure 10: Hierarchical software-defined architecture. Each multi-variable control loop can be designed as a DepMPC.

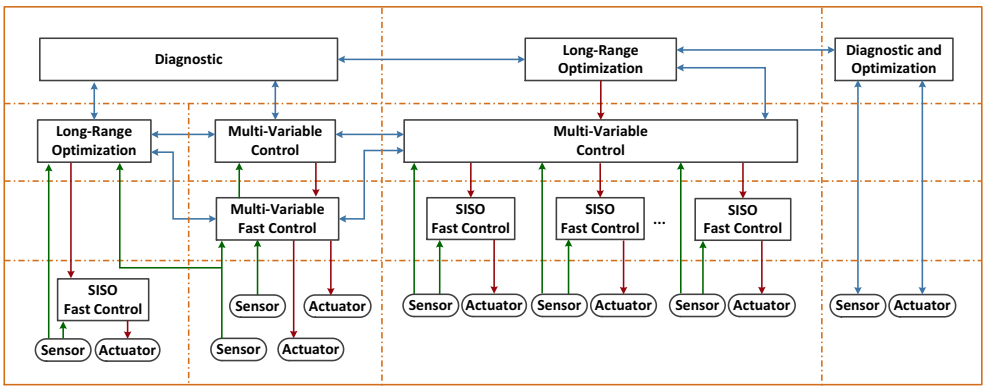
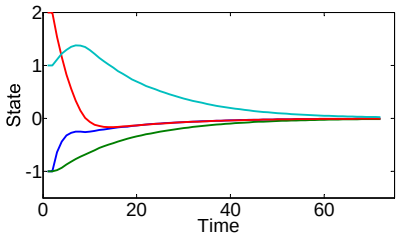
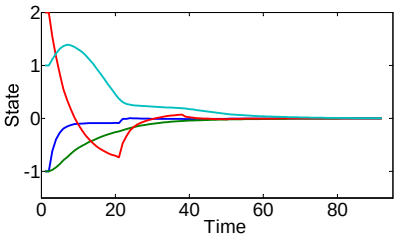


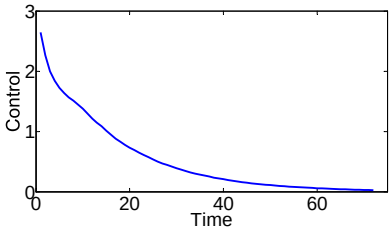
Figure 11: Multiple time-scale, heterogeneous, and hierarchical software-defined architecture. Each multi-variable control loop can be designed as a DepMPC.



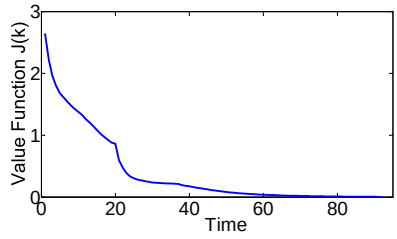
(a) State trajectories without duty-standby transitions (four states).



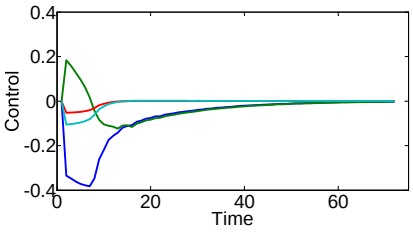
(b) State trajectories with duty-standby transitions - The transitions occur between $5-15,\ 25-35,\ 40-47,\ 50-55$ time steps.



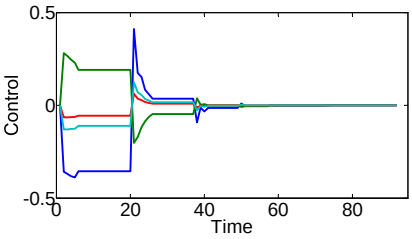
(c) Value function trajectory without duty-standby transitions.



(d) Value function trajectory with duty-standby transitions.

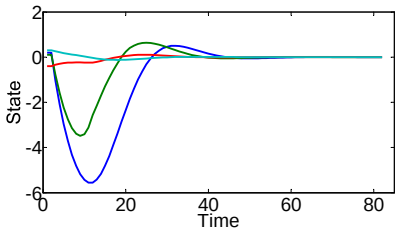


(e) Control trajectories without duty-standby transitions (four controls).

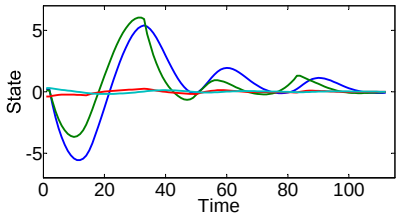


(f) Control trajectories with duty-standby transitions.

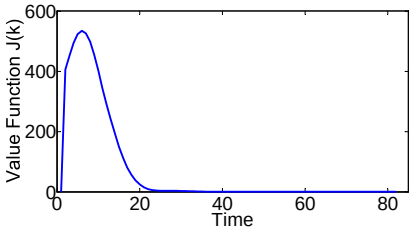
Figure 12: DepMPC running on the paper machine example with four duty-standby transitions in the 10 time step intervals. The MPC predictive horizon is $N = 3$. A persistent disturbance $d(k)$ has been added. Time intervals of the duty-standby transitions are between $5-15,\ 25-35,\ 40-47,\ 50-55$ time steps. The performance variables did not improve the control performance in this example of an open-loop stable plant.



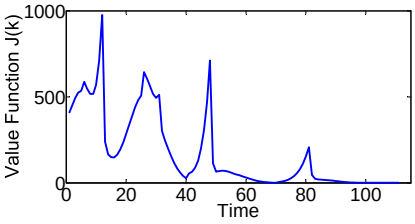
(a) State trajectories without duty-standby transitions (four states).



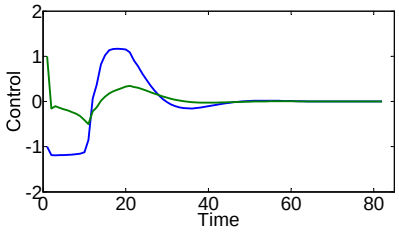
(b) State trajectories with duty-standby transitions - The transitions are between $5-10,\ 25-30,\ 40-47,\ 70-80$ time steps.



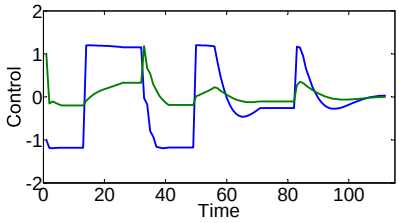
(c) Value function trajectory without duty-standby transitions.



(d) Value function trajectory with duty-standby transitions.

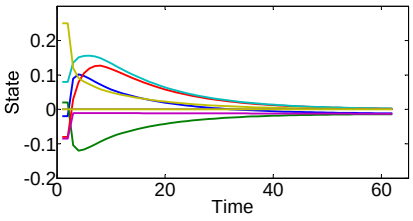


(e) Control trajectories without duty-standby transitions (two controls).

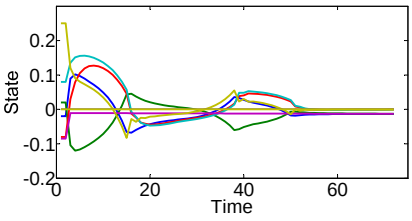


(f) Control trajectories with duty-standby transitions.

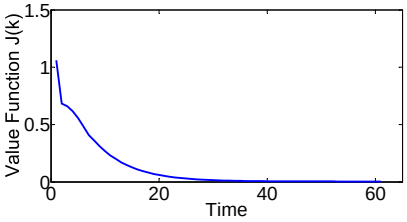
Figure 13: DepMPC running on the helicopter example with four duty-standby transitions in the 10 time step intervals. The MPC predictive horizon is $N = 12$. Time intervals of the duty-standby transitions are between $5-10,\ 25-30,\ 40-47,\ 70-80$ time steps. The constraint $\hat{\mathbf{u}} \in \mathbb{U}_b$ improved the performance of $\sum_k(\mathcal{J}(k))$ by around 5% with $\beta = 1.6$. The constraint of the form $V(k) < \gamma \times V(k_s)$ has improved the performance of $\sum_k(\mathcal{J}(k))$ by around 8% with $\gamma = 0.88$.



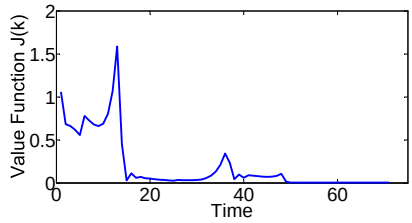
(a) State trajectories without duty-standby transitions (fourteen states).



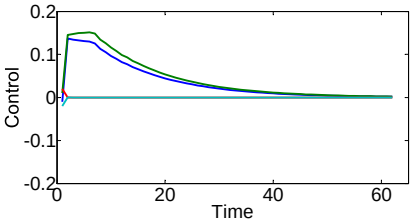
(b) State trajectories with duty-standby transitions- The transitions are between $5-12,\ 25-35,\ 40-47$ time steps.



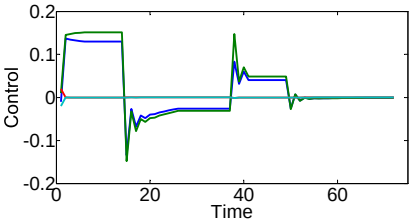
(c) Value function trajectory without duty-standby transitions.



(d) Value function trajectory with duty-standby transitions.



(e) Control trajectories without duty-standby transitions (four controls).



(f) Control trajectories with duty-standby transitions.

Figure 14: DepMPC running on the railway wheel set example with four duty-standby transitions in the 10 time step intervals. The MPC predictive horizon is $N = 5$. Time intervals of the duty-standby transitions are between $5-12,\ 25-35,\ 40-47$ time steps.

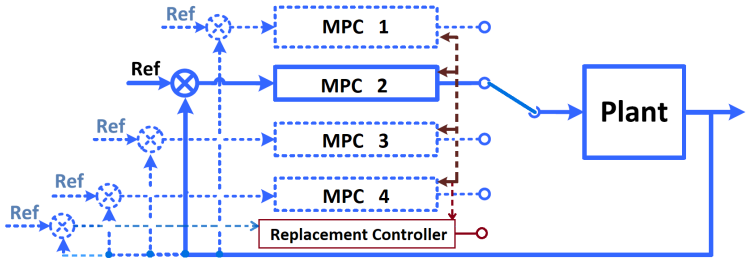


Figure 15: The DepMPC is modified with a Replacement Controller - Block diagram.

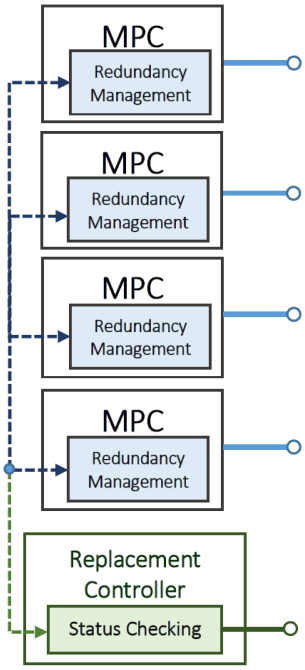
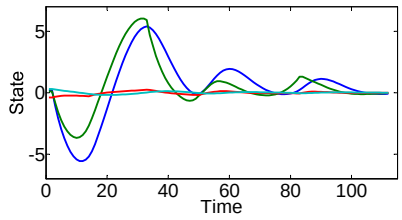
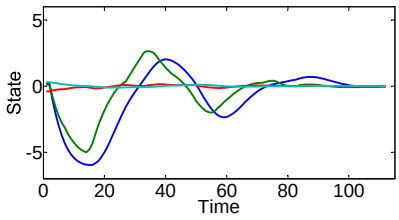


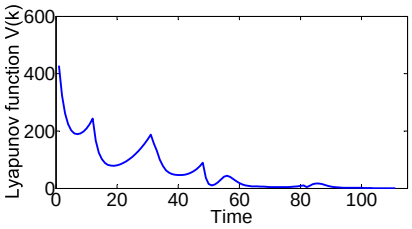
Figure 16: The status checking in a Replacement Controller.



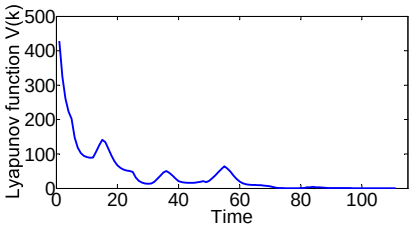
(a) State trajectories without replacement controller.



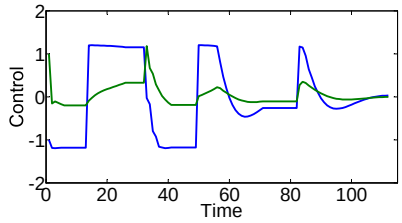
(b) State trajectories with replacement controller.



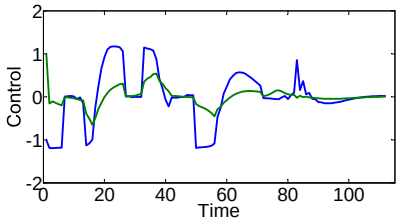
(c) Luapunov function trend without replacement controller.



(d) Luapunov function trend with replacement controller.

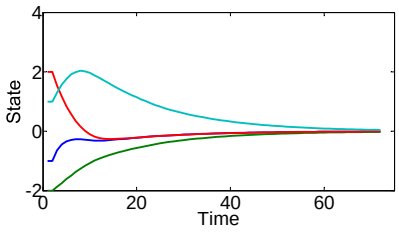


(e) Control trajectories without replacement controller.

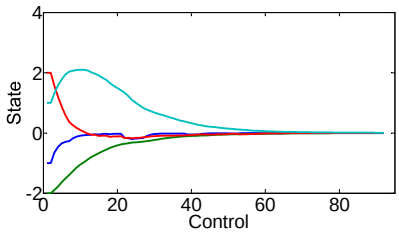


(f) Control trajectories with replacement controller.

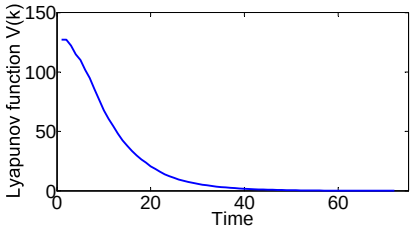
Figure 17: DepMPC and Replacement Controller for the helicopter model. The DepMPC has four duty-standby transitions between $5-10,\ 25-30,\ 40-47,\ 70-80$ time steps. The Replacement Controller improved the control performance of $\sum_k(V(k))$ by around 35%.



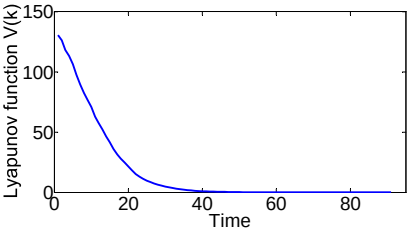
(a) State trajectories without duty-standby transitions.



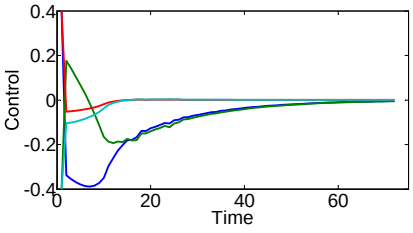
(b) State trajectories with replacement controller.



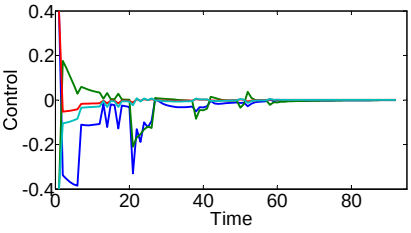
(c) Luapunov function trend without duty-standby transitions.



(d) Luapunov function trend with replacement controller.



(e) Control trajectories without duty-standby transitions.



(f) Control trajectories with replacement controller.

Figure 18: DepMPC and Replacement Controller for the paper machine. The DepMPC has four duty-standby transitions between $5-15,\ 25-35,\ 40-47,\ 50-55$ time steps. The Replacement Controller improved the control performance of $\sum_k (V(k))$ by around 12%.

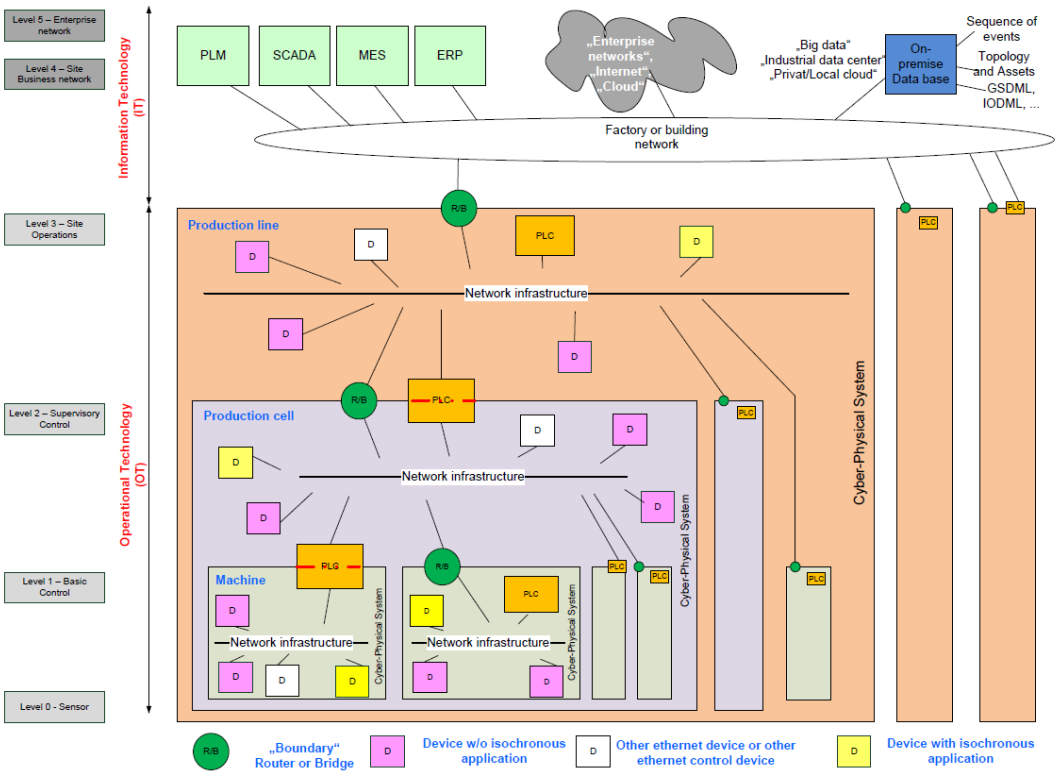


Figure 19: Hierarchical structure of industrial automation with Time Sensitive Networking – IEC/IEEE 60802, fig. 1.

***Conflict of Interest**

This piece of the submission is being sent via mail.