UNIVERSITY OF CALIFORNIA, MERCED

SELF-OPTIMIZING SMART CONTROL ENGINEERING ENABLED BY DIGITAL TWINS

by

Jairo Viola

A thesis submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Mechanical Engineering

Committee in charge:
Dr. YangQuan Chen, Advisor
Dr. Jian-Qiao Sun
Dr. Ricardo Pintro de Castro
Dr. Reza Ehsani

The thesis of Jairo Viola is approved:

Dr. YangQuan Chen Date

Dr. Jian-Qiao Sun Date

Dr. Ricardo Pinto de Castro Date

Dr. Reza Ehsani Date

University of California, Merced

Dedicated to the loving memory of my father Jairo.
To my mother Carmenza and my sister Diana.

ACKNOWLEDGEMENTS

First, I want to express my most sincere thanks to my Ph.D. supervisor, Prof. YangQuan Chen, for his continuous support, patience, discussions, guidance, and encouragement during these years of changes and uncertainty. Thank you, Dr. Chen, for letting me be part of MESALab and teaching me how to do world-class research and have practical experience with industry collaborations.

Likewise, I am grateful to Dr. Jian-Quiao Sun, Dr. Ricardo de Castro, and Dr Reza Ehsani for your valuable domain knowledge, comments, and precious time dedicated to serving as my dissertation committee members.

Thanks to my MESALab colleagues worldwide, with whom I had the honor of working during these years, Dr. Sina Dehghan, Dr. Derek Hollenbeck, Dr. Haoyu Niu, Dr. Piotr Oziablo, and Dr. Carlos Rodriguez, Alberto Radici, Mauricio Calderon, Furkan Guc, Di An, Justus Nwoke and many others.

Thanks go to my master and bachelor advisor Prof. Luis Angel Silva in Pontifical Bolivarian University in Colombia for his help and friendship on my early studies.

I am grateful to my mother Carmenza, and my sister Diana for their unconditional support since I decided to apply to UC Merced. Your encouragement has been crucial in the most challenging moments of this process.

Last but not least, thank goes to the Navarro-Hartford family for their support and sincere friendship during my time in the USA.

CURRICULUM VITAE

Education

Ph.D. in Mechanical Engineering, University of California, Merced, 2022.
M.S. in Electronic Engineering, Pontifical Bolivarian University, 2017.
B.S. in Electronic Engineering, Pontifical Bolivarian University, 2013.

Work Experience

- Control Engineer at Lam Research Corporation, Embedded systems Group, Fremont, CA. Main Responsibility: Providing Solutions and support for Etching Systems Related Control Problems. 2021-current.

- Teaching Assistant for ME21: Engineering Computing (Fall 2018); ME142: Mechatronics (Spring 2019 to 2021). University of California, Merced, 2018 - 2021.

- Lecturer of Electrical, Electronic Engineering and Computer Science departments, Pontifical Bolivarian University, Bucaramanga, Colombia (2013-2018). Taught the courses of Engineering Computing (Java, C/C++, Python), Computer Architecture, Operating Systems, Logic and Algorithms.

Publications

- Jairo Viola and YangQuan Chen, Digital Twin Enabled Smart Control Engineering: A Framework and Case Studies. Springer. 2022 (to be published)

- J. Viola, Y. Chen, and J. Wang, "FaultFace: Deep convolutional generative adversarial network (DCGAN) based ball-bearing failure detection method," Information Sciences, vol. 542, pp. 195–211, 2021.

- J. Viola and Y. Chen, "A Self Optimizing Control Framework and A Benchmark for Smart Process Control," in 2021 3rd International Conference on Industrial Artificial Intelligence (IAI), 2021, pp. 1–6.

- J. Viola and Y. Chen, "An Accelerated Self Optimizing Control Framework for Smart Process Control Using Fractional Order Stochasticity," in 2021 9th International Conference on Control, Mechatronics and Automation (ICCMA), 2021, pp. 104–109.

- J. Viola, C. Rodriguez, D. Hollenbeck. and Y. Chen. A radio frequency impedance matching control benchmark and optimal fractional-order stochastic extremum seeking method. Outliers in Control Engineering: Fractional Calculus Perspective (pp. 237-258). Berlin, Boston: De Gruyter. https://doi.org/10.1515/9783110729122-013

- C. A. Rodriguez, J. Viola, and Y. Chen, "An Radio Frequency Impedance Matching Control Benchmark System for Advanced Control Strategies Evaluation," in International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2021, vol. 85437, p.8.

- C. Rodriguez, J. Viola, and Y. Chen, "Data-Driven Modelling for a High Order Multivariable Thermal System and Control," IFAC-PapersOnLine, vol. 54, no. 20, pp. 753–758, 2021.

- J. Viola, F. Guc, M. Calderon, and Y. . Chen, "Digital Twin Based Interactive Mechatronics Lab Development for Remote Lab Offering and Evaluation," in Proceedings of the ASME 2021 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference IDETC/CIE, 2021, p. 9.

- M. R. C. Carrion, J. Viola, Y. Chen, and A. Visioli, "Digital twin technology for modeling, simulation and control of a mechatronic system," in 17th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, MESA 2021, Held as Part of the ASME 2021 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, IDETC-CIE 2021, 2021, vol. 7.

- F. Guc, J. Viola, and Y. Chen, "Digital twins enabled remote laboratory learning experience for mechatronics education," in 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI), 2021, pp. 242–245.

- J. Viola, D. Hollenbeck, C. Rodriguez, and Y. Chen, "Fractional-Order Stochastic Extremum Seeking Control with Dithering Noise for Plasma Impedance Matching," in 2021 IEEE Conference on Control Technology and Applications (CCTA), 2021, pp. 247–252.

- J. Viola and Y. Chen, "Parallel Self Optimizing Control Framework for Digital Twin Enabled Smart Control Engineering," in proceedings of the IEEE International Conference on Digital Twins and Parallel intelligence (DTPI), 2021, p. 6.

- S. Dehghan, J. Viola, and Y. Chen, "Preview Control Based on RIOTS MPC and H for A Thermal Hardware in the Loop System," IFAC-PapersOnLine, vol. 54, no. 20, pp. 729–734, 2021.

- Z. Wu, J. Viola, Y. Luo, Y. Chen, and D. Li, "Robust fractional-order [proportional integral derivative] controller design with specification constraints: more flat phase idea," Int. J. Control, vol. 0, no. 0, pp. 1–19, 2021, doi: 10.1080/00207179.2021.1992498.

- J. Viola, P. Oziablo, and Y. Chen, "A Portable and Affordable Networked Temperature Distribution Control Platform for Education and Research," IFAC-PapersOnLine, vol. 53, no. 2, pp. 17530–17535, 2020.

- J. Viola, P. Oziablo, and Y. Chen, "A Study of the Influence of Stochastic Fractional-Order Delay Dynamics in a Networked Control System," IFAC-PapersOnLine, vol. 53, no. 2, pp. 5789–5794, 2020.

- J. Viola and Y. Q. Chen, "Digital Twin Enabled Smart Control Engineering as an Industrial AI: A New Framework and Case Study," 2nd Int. Conf. Ind. Artif. Intell. IAI 2020, 2020, doi: 10.1109/IAI50351.2020.9262203.

- Z. Wu, Y. Chen, J. Viola, Y. Luo, Y. Q. Chen, and D. Li, "Fractional order [proportional integral derivative] controller design with specification constraints: More flat phase idea," IFAC-PapersOnLine, vol. 53, no. 2, pp. 3650–3656, 2020, doi: 10.1016/j.ifacol.2020.12.2047.

- J. Viola, Y. Chen, and J. Wang, "Information-Based Model Discrimination for Digital Twin Behavioral Matching," in 2nd International Conference on Industrial Artificial Intelligence (IAI), 2020.

- J. Viola, C. Rodriguez, and Y. Chen, "PHELP: Pixel Heating Experiment Learning Platform for Education and Research on IAI-based Smart Control Engineering," in 2nd International Conference on Industrial Artificial Intelligence (IAI), 2020, pp. 1–6.

- J. Viola, P. Oziablo, and Y. Chen, "An Experimental Networked Control System with Fractional Order Delay Dynamics," in 2019 7th International Conference on Control, Mechatronics and Automation (ICCMA), 2019, pp. 226–231.

- J. Viola and L. Angel, "Delta Parallel Robotic Manipulator Tracking Control using Fractional Order Controllers," IEEE Lat. Am. Trans., vol. 17, no. 03, pp. 393–400, 2019.

- J. Viola, S. Dehghan, and Y. Chen, “Embedded riots - Model predictive control towards edge,” in Proceedings of the ASME Design Engineering Technical Conference, 2019, vol. 9, doi: 10.1115/DETC2019-97046.

- L. Angel, J. Viola, and M. Paez, “Evaluation of the windup effect in a practical PID controller for the speed control of a DC-motor system,” in 2019 IEEE 4th Colombian Conference on Automatic Control (CCAC), 2019, pp. 1–6.

- L. Angel, J. Viola, and M. Vega, “Hardware in the loop experimental validation of PID controllers tuned by genetic algorithms,” in 2019 IEEE 4th Colombian Conference on Automatic Control (CCAC), 2019, pp. 1–6.

- J. Viola, A. Radici, S. Dehghan, and Y. Chen, “Low-cost real-time vision platform for spatial temperature control research education developments,” in Proceedings of the ASME Design Engineering Technical Conference, 2019, vol. 9, doi: 10.1115/DETC2019-97664.

- L. Angel, J. Viola, and M. Vega, “Metaheuristic Tuning and Practical Implementation of a PID Controller Employing Genetic Algorithms,” in ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2019.

- L. Angel and J. Viola, “Fractional order PID for tracking control of a parallel robotic manipulator type delta,” ISA Trans., vol. 79, pp. 172–188, 2018, doi: 10.1016/j.isatra.2018.04.010.

- M. Paez, J. Viola, and L. Angel, “Practical PID controller implementation for the speed control of a motor generator system,” MSE, vol. 437, no. 1, p. 12013, 2018.

- J. Viola and L. Angel, “Tracking control for robotic manipulators using fractional order controllers with computed torque control,” IEEE Lat. Am. Trans., vol. 16, no. 7, pp. 1884–1891, 2018.

- J. Viola, L. Angel, and J. M. Sebastian, “Design and Robust Performance Evaluation of a Fractional Order PID Controller Applied to a DC Motor,” IEEE/CAA J. Autom. Sin., vol. 4, no. 2, pp. 304–314, 2017.

- L. Angel and J. Viola, “Identification and control using fractional operators for the speed control of a motor-generator system,” in 2017 IEEE 3rd Colombian Conference on Automatic Control (CCAC), 2017, pp. 1–6.

- L. Angel, J. Viola, and M. Paez, “Speed control of a motor-generator system using internal model control techniques,” in 2017 IEEE 3rd Colombian Conference on Automatic Control (CCAC), 2017, pp. 1–6.

- C. Hernández, L. Angel, and J. Viola, "Modelling of uncertainness for a flow and level system," IOP Conf. Ser. Mater. Sci. Eng., vol. 138, no. 1, p. 12009, 2016.

- L. Angel and J. Viola, "Parametric identification of a delta type parallel robot," in 2016 IEEE Colombian Conference on Robotics and Automation (CCRA), 2016, pp. 1–6.

- L. Angel, J. Viola, and C. Hernández, "Parametric uncertain identification of a robotic system," in IOP Conference Series. Materials Science and Engineering (Online), 2016, vol. 138, no. 1.

- L. Angel and J. Viola, "Payload estimation for a robotic system using unsupervised classification," in 2016 XXI Symposium on Signal Processing, Images and Artificial Vision (STSIVA), 2016, pp. 1–5.

- J. Viola and C. Aceros, "Smart Grids and their Applicability for the Development of the Electricity Sector for Colombia in the Year 2050," in IOP Conference Series: Materials Science and Engineering, 2016, vol. 138, no. 1, p. 12010.

- L. Angel, J. Viola, M. Vega, and R. Restrepo, "Sterilization process stages estimation for an autoclave using logistic regression models," in 2016 XXI Symposium on Signal Processing, Images and Artificial Vision (STSIVA), 2016, pp. 1–5.

- L. Angel, S. Lizcano, and J. Viola, "Assessing the state of maturation of the pineapple in its perolera variety using computer vision techniques," in Signal Processing, Images and Computer Vision (STSIVA), 2015 20th Symposium on, 2015, pp. 1–6.

- L. Angel and J. Viola, "Design and statistical robustness analysis of FOPID, IOPID and SIMC PID controllers applied to a motor-generator system," IEEE Lat. Am. Trans., vol. 13, no. 12, pp. 3724–3734, 2015, doi: 10.1109/TLA.2015.7404900.

- J. Viola and L. Angel, "Design and statistical robustness analysis of FOPID, IOPID and SIMC PID controllers for the control of an input-output linearized plant model," in Automatic Control (CCAC), 2015 IEEE 2nd Colombian Conference on, Oct. 2015, pp. 1–6, doi: 10.1109/CCAC.2015.7345191.

- J. Viola and L. Angel, "Factorial design for robustness evaluation of fractional PID controllers," Lat. Am. Trans. IEEE (Revista IEEE Am. Lat., vol. 13, no. 5, pp. 1286–1293, May 2015, doi: 10.1109/TLA.2015.7111981.

- J. Viola and L. Angel, "Fractional control and robustness analysis of an inverted pendulum system," in 2015 IEEE 2nd Colombian Conference on Automatic Control (CCAC), Oct. 2015, pp. 1–6, doi: 10.1109/C-CAC.2015.7345223.

- J. Viola and L. Angel, "Identification, control and robustness analysis of a robotic system using fractional control," Lat. Am. Trans. IEEE (Revista IEEE Am. Lat., vol. 13, no. 5, pp. 1294–1302, May 2015, doi: 10.1109/TLA.2015.7111982.

- L. Angel and J. Viola, "Performance Comparison between a PID SIMC and a PD Fractional Controller," Robot Intelligence Technology and Applications 2, Springer, Cham, 2014, pp. 415–424.

- J. Viola and L. Angel, "Comparison between fractional order PI controller and SIMC PI controller," In proceeding of the 32nd Chinese Control Conference (CCC), 2013 , 2013, pp. 2786–2791.

- J. Viola and L. Angel, "Design of a fractional PI controller to control a flow and level system," in proceeding of the Control and Decision Conference (CCDC), 2013 25th Chinese, May 2013, pp. 1211–1216, doi: 10.1109/C-CDC.2013.6561109.

Workshops and invited talks

- Chen Y.Q, Viola, J, Domansky, P. and Wang, Y. and Smart Control Engineering (SCE) via Digital Twins (DT) and Industrial AI (IAI) . IEEE 2022 American Control Conference, Atlanta, GA, June 8-10, 2022.

- Chen Y.Q, Domansky, P. Wang, Y, and Viola, J. Smart Control Engineering (SCE), Digital Twins, and Industrial AI (IAI) – A New Research Frontier. IEEE 2021 60th IEEE conference on Decision and Control. December 12, 2021. Virtual.

- Chen Y.Q, Domansky, P. Wang, Y, and Viola, J. Smart Control Engineering (SCE), Digital Twins, and Industrial AI (IAI) – A New Research Frontier. 1st IEEE 2021 Conference on Control Technology and Applications (CCTA), August 9-11, 2021, Virtual.

- Domansky, P. Chen Y.Q and Viola, J. Workshop W3: Advanced big data analytics for control performance assessment in Industry 4.0 era. Preformed at 2021 American Control Conference. May 24, 2021, Virtual.

- J. Viola. An overview of Industry 4.0 towards Smart Control Engineering. Invited talk for the Master on Electronic Engineering, Pontifical Bolivarian University, Colombia, June 17 2020.

- J. Viola. Edge Computing and Machine Learning Techniques Applied to Failure Detection on Industrial Processes. Invited talk for the International Conference on innovation and appropriation of the information and communication technologies. Ocaña, Colombia, September 27 2019.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

There are two critical questions in control engineering: how optimal, and how robust the control system is? However, digital transformation, Industry 4.0, and the advent of breaking technologies like artificial intelligence, deep learning, big data analytics, edge computing, and etc., contribute to increased system health knowledge, sensing capabilities, and automation on performance assessment metrics. For this reason, two new questions emerge: how smart and how developmental a control system could be? Therefore, a new frontier in control engineering emerges, and this dissertation defines it as smart control engineering (SCE), supported by three groundbreaking technologies digital twin (DT), industrial artificial intelligence (IAI) and self optimizing control (SOC). Thus, smart control engineering transforms classic control systems into smart control systems. It means that systems are aware of their capabilities and limitations (cognizant), able to learn from past experience to improve its future performance (reflective), supported by a substantial body of knowledge (knowledge-rich), handling high-level instructions based on human vague commands (taskable), and always adhering to social and legal norms (ethical). This thesis tries to establish the foundations of the smart control engineering framework and its combination with digital twin, industrial artificial intelligence, and self optimizing control for the development of smart control systems. A set of smart and developmental controllers supported by digital twin are developed using real-time zeroth-order optimization algorithms to enable smartness on real systems. Likewise, a set of enabling capabilities resulting from breaking technologies like smart controller design, control performance assessment, or parallel intelligence and controls are integrated into the SCE framework, powered by real-time data analytics provided by IAI methods. The embedded implementation of smart controllers with enabling capabilities is performed and demonstrated for single-input and multi-input control systems using edge computing devices. Obtained results show that smart control engineering is a new and effective framework that can systematically improve the performance, reliability, and robustness under varying internal and external conditions.

ABBREVIATIONS

ACP	Artificial, Computational, and Parallel Execution
AIC	Akaike Information Criteria
BIC	Bayesian Information Criteria
BJ	Box-Jenkins Model
CPA	Control Performance Assessment
CPC	Cognitive Process Control
CPS	Cyber-Physical Systems
DT	Digital Twin
FOPDT	First Order Plus Dead Time system
GCNM	Globalized Constraint Nelder Mead optimization
IA	Artificial Intelligence
IAE	Integral Absolute Error
IAI	Industrial Artificial Intelligence
ICT	Information and Communication Technologies
IID	Independent and Identically Distributed Random Variables
ISE	Integral Square Error
ITAE	Integral Time Absolute Error
IoT	Internet of Things
IIoT	Industrial Internet of Things
LTI	Linear Time Invariant System

LRD	Long Range Dependence
MAD	Modeling, Analysis, and Design
MBD	Model-Based Design
MDL	Minimum Description Length
MIMO	Multiple Input Multiple Output System
MPC	Model Predictive Control
MNM	Modified Nelder-Mead Optimization Algorithm
MWAS	Matlab Web App Server
nAIC	Normalized Akaike Information Criteria
NM	Nelder-Mead Optimization Algorithm
NRMS	Normalized Root Mean Square Value
OV	Overshoot
PCA	Principal Component Analysis
PI	Proportional, Integral Controller
PID	Proportional, Integral, Derivative Controller
PWM	Pulse Width Modulation
RTO	Real-Time Optimization
RMS	Root Mean Square Value
RUL	Remaining Useful Life Estimation
SCE	Smart Control Engineering
SDE	Stochastic Differential Equation
SIMO	Single Input Multiple Output System
SISO	Single Input Single Output System
SLDO	Simulink Design Optimization Toolbox
SOC	Self Optimizing Control

SPSA	Simultaneous Perturbation Stochastic Approximation
TIR	Thermal Infrared Camera
ZN	Ziegler-Nichols Tuning Method

Chapter 1

INTRODUCTION

1.1 Motivation

Classic control system design is supported by three fundamental questions: 1) “What do I know about the system?”, 2) “What do I have for controlling the system?”, and 3) “What do I want of the system?”. This systematic view can be summarized by the modelling, analysis, and design methodology (MAD) proposed by [1]. It employs different approaches of control theory like system identification, linear, non-linear, predictive, optimal, adaptive, model-based, state-space, robust, intelligent, fuzzy, or fractional-order controls to analyze the system behavior, obtain a system model, and design a controller that satisfies the proposed closed-loop specifications. Thus, the resulting control system performance can be evaluated under two questions:

- 1) How robust is the closed-loop system in the presence of external disturbances and uncertainties?
- 2) How optimal is the controller performance to reach the desired closed-loop specifications?

Addressing these questions is crucial to improve the performance of any closed-loop control system. For this reason, several methodologies for its analysis and improvement has been proposed in the literature for Single-Input Single-Output (SISO) and Multiple-Input Multiple-Output (MIMO) systems [2–8].

However, the transition into Industry 4.0 is transforming classic control systems into Cyber-Physical Systems (CPS). It means that the traditional factory transforms into a Smart Factory where multiple individual systems interact in a complex and changing environment, sharing global performance objectives to improve the overall system quality and productivity. Therefore, new Modelling, Analysis, and Design techniques are required for control systems that are aware of the each system changes and the overall system as well as be able to adapt based on its interactions with the environment and other systems.

On the other hand, the Digital Transformation of companies accelerates the adaptation of breaking technologies like Artificial Intelligence (AI), Big Data, Data

Analytics, Deep Learning, or Internet of Things (IoT) in all the production processes [9–12]. Thus, the use of breaking technologies results in a renewed control engineering supported by increased knowledge of the different systems via enhanced sensing and actuation capabilities, with automated analytics of the system able to recognize the system's current condition and adapt the closed-loop behavior to satisfy the global operation objectives of the system. Based on this new features, two new questions need to be introduced in the control system design:

- 1) How smart is a control system to be aware not only of its current condition but also of its environment to reach the desired performance objectives?

- 2) How developmental is a smart control system to adapt its control law according to the system current and past conditions?

So that, Industry 4.0 manufacturing processes requires a new control framework that combines cognition features, automatic control, and advance modelling techniques that enable Smartness and developmental capabilities, which will be defined in this dissertation as Smart Control Engineering.

1.1.1 What Is A Smart System?

Defining what is a smart system is the first step towards Smart Control Engineering. It can be defined as a system with the capabilities of sensing, actuation, and control, able to make decisions based on real-time and historic knowledge in order to satisfy a set of desired performance specifications in the presence of uncertainties and external disturbances. A graphical representation of a smart system is shown in Fig. 1.1. As can be observed, each smart system incorporates the sensing, control and actuation features as well as the interaction with other smart systems. Besides, according to the Smart and Autonomous Systems (S&AS) program of NSF (Natural Sciences Foundation) [13], the characteristics of a smart system are:

- **Cognizant:** the system is aware of capabilities and limitations to face the dynamic changes and variability.

- **Taskable:** The system is capable of handle high-level, often vague instructions based on the automated stimulus or human commands.

- **Reflective:** A smart system is able to learn from previous experience to improve its performance.

- **Knowledge Rich:** All the reasoning processes make by a smart control system are performed over a diverse body of knowledge from a rich environment and sensor-based information.

- **Ethical:** The smart behavior of a system adheres to a system of societal and legal norms.

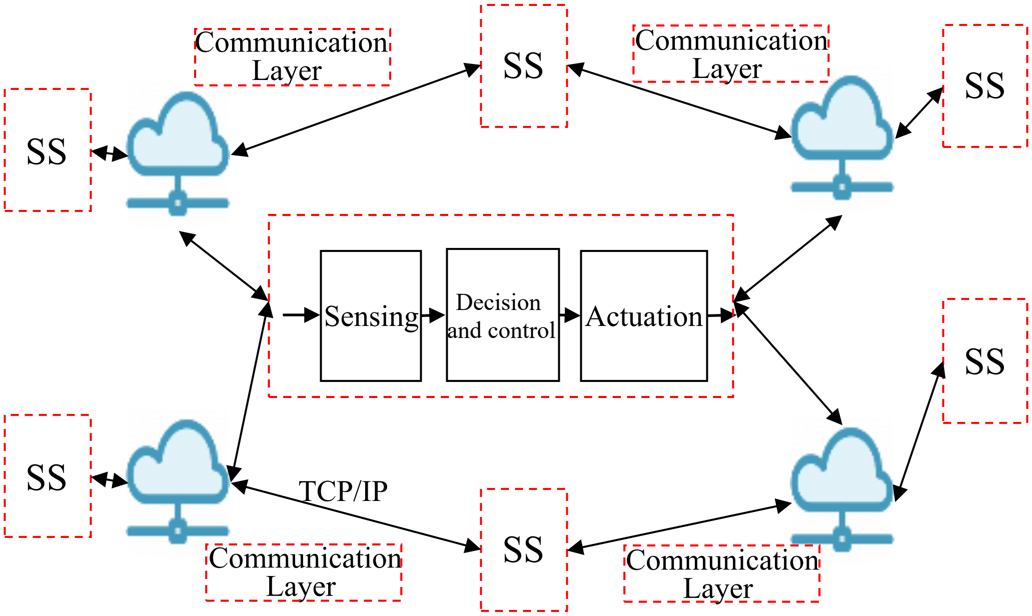


Figure 1.1: Structure of a Smart System

In recent years, smart systems widespread use has been increasing thanks to the groundbreaking technologies like Big Data, Machine Learning or Edge Computing, in addition to cheaper embedded wireless radio communication, low-cost large memory and storage, bigger microprocessor capabilities or rich information driven models that allows fast and reliable monitoring and control.

Some applications of smart systems includes smart transportation [14–16], smart grid [17–19], unmmaned autonomous vehicles and robotic systems [20, 21]. However many of these smart systems use to operate inside controlled environments with low variability, making the smart systems unable to handle unanticipated changes or dynamics given not only by a variable environment but also for the system non-modeled behaviors. In the context of Industry 4.0, the industrial processes are just a part of a more complex CPS, equipped with multiple sensors that produces big data as well as the interaction among its agents.

In this scenario, the Smart Control Engineering looks for leveraging all these groundbreaking technologies to make the system resilient to face any unanticipated situation as well as dynamic changes of the smart systems without human intervention, it means, producing automated responses to these events.

1.1.2 Smart Control Engineering: A New Frontier

The concept of Smart Control Engineering begins with the Cognitive Process Control (CPC) proposed by Chen in [22] which is divided in three generations. The first generation consist of the application of data-driven analysis techniques to improve the classic process control framework provided by the MAD methodology and supported by the standard closed-loop variables like output, control action or error, and reference signals. Thus, Cognitive Process Control is aware of processing vital signs for healthy runs of the process from the component level, supporting decision making and health issues alerts using multiple information sources, learning from the past behavior and self induced errors with control. Also, CPC allows obtaining insights from the known and unknown outlier failure behaviors at multiple time-scales following the steps of cognition, decision, and control.

In the second generation of CPC, the signal-based control is transformed into information-driven control given by sensor, actuator, and memory rich systems thanks to the use of breaking technologies like IoT, AI, Data Analytics, and cheaper sensing, obtaining new knowledge from the system based on the new sensing and actuation capabilities. In this CPC generation, many of the analysis are performed offline, specially those that requires optimzation and model estimation.

For this reason, the third generation of CPC brings to the source (real process) real-time data analytics, embedded artificial intelligence, health assessment, as well as other features powered by edge computing devices, introducing smartness directly into real systems.

Thus, the combination of the Cognitive Process Control generations with breaking technologies, and its application in real systems introducing smartness into closed-loop control can be considered as Smart Control Engineering. It can be defined as:

> *“A branch of control engineering that leverage breaking technologies like Edge Computing, IoT, AI, Big Data, Data Analytics to enhance the smartness and performance of a control system“* MESALab 2021.

Likewise, the Smart Control Engineering introduces a set of new characteristics like:

- Each system incorporates smartness capabilities given by breaking technologies and supported by edge computing devices, performing real-time health awareness status of the system.

- Involves one or multiple real-time optimization and artificial intelligence decision based stages that seeks the optimal performance of the system, based on economic, environmental, health, or fault conditions, operating as a single system or into an interconnected environment.

- Requires the use of virtual representation of each single element composing the system in order to perform analysis of the system behavior under different conditions that may not be feasible of evaluating during real operation due to the cost or risk associated.

- Introduces resilient behavior on the smart system being able of adapting to unknown changes on the system based on evolving controllers and its control actions according to the system current health status.

These features of Smart Control Engineering are supported by three core technologies corresponding to the industrial artificial intelligence (IAI), digital twin (DT), and self optimizing control (SOC).

The Industrial Artificial Intelligence (IAI) is defined as any application of AI related to the physical operations or systems of an enterprise. It is focused on helping an enterprise monitor, optimize or control the behavior of these operations and systems to improve their efficiency and performance [23]. Likewise, IAI is taking more relevance in the last years due to its multiple application in Industry 4.0 as shown in [24–28] for fault detection, cyber attack detection or federated learning. Thus, IAI becames a core technology for Smart Control Engineering due to it provides the analytic tools and algorithms that can be applicable for real-time analysis of a system in order to determine its current health and status, providing the right information to the Real-Time Optimization and Artificial Intelligence decision based layers that make decisions supported by the analytics provided by IA tools.

In the case of digital twin, it can be interpreted as a virtual, digital equivalent of a physical product, usually across multiple stages of its lifecycle [29, 30]. It uses real-world data, simulation or machine learning models, combined with data analysis, to enable understanding, learning, and reasoning based on the current and historic status of the system. Like IAI, digital twins have been under an increasing interest not only in academia but specially in industry, where the possibility of building virtual replicas of complex systems like shop floor or smart warehouses for operational optimization. Under the scope of Smart Control Engineering, digital twin became a relevant tool to leverage all its features via real-time updated virtual representations of the system, considering the fact that it is difficult to represent fault behaviors and undesired situations in real life due to the risk associated with high cost of running these tests.

Besides, Self Optimizing Control (SOC) is a control strategy designed initially for controlling chemical plants with tens or hundreds of control variables (CV). It consist of choosing the best CV that satisfy a cost function in terms of economical performance with constant setpoints based in a model of the system using one or multiple optimization stages at different timescales (minutes, days, months) [31]. However, SOC may have different scopes according to the technique employed to reach an acceptable performance of the system. One point of view for SOC can be the use of seeking algorithms like Extremum Seeking [32] or Maximum Power Point Tracking (MPPT) [33] when a desired behavior has to be reached based on a cost function with little or no knowledge of the system dynamics. Another scope is the use of a high Real-Time Optimization algorithms (gradient based or derivative free) for low-level control loops parameters or setpoint adjustment based on some desired specifications under some periodic task [34]. Likewise, other SOC schemes can be reached using repetitive control strategies like run to run, or iterative learning control.

In the case of Smart Control Engineering, the role of SOC is acting as that upper layer of control and optimization that takes the real-time analytics provided by IAI and the exhaustive analysis performed via real-time updated digital twin to generate the smart response of the system, satisfying the five attributes of a smart system: cognizant, reflective, knowledge rich, taskable and ethical.

Therefore, this dissertation looks for a methodological framework that allows a systematic design and implementation of Smart Control Engineering applications that can be applied to any system without considering its complexity or scale, supported by the core technologies of digital twin, Industrial Artificial Intelligence, and Self Optimizing Control.

1.1.3 Literature Review of Smart Systems and Controls Approaches

The idea of incorporating smartness into closed-loop systems has becoming relevant in the last years due to the increasing adoption of the breaking technologies. Some common applications of smart systems and controls are associated to smart grids, building energy efficiency, batteries and energy storage devices, transportation or autonomous vehicles.

From the point of view of batteries and energy storage smart controls, [35] presents a smart control management algorithms to aggregate different renewable power supplies according to consumer load and electricity prices in a community grid. In [36], a monitoring architecture is proposed for the management of the power grid at conceptual level based on real-time data from the system. Also, [37] combines unsupervised learning with closed loop control to improve the stability of Buck converters at simulation level. In [38], a smart control is proposed to improve parallel loads efficiency with an offline upper optimization layer using Particle Swarm Optimization (PSO) on top of low-level controllers. Besides, a smart control scheme is proposed by [39] for optimizing State of Charge of electric vehicle battery conected as energy storage device on a microgrid employing a two stages optimization algorithm. In [40], a smart based control is designed for fuel cells using MPPT algorithm. Some applications for microgrid power control and batteries charging are presented in [41–43] using fuzzy logic as decision policy only in combination with classic controls like PI. In these cases, the smart concept is limited to the Fuzzyfication of the system without considering variations on the system behavior.

Notice that in the cases presented above, some optimization or upper level law are proposed to improve the system operation. However, the algorithms proposed lacks of real-time support and analytics based on IAI and DT.

In the case of building energy management, there are several approaches of a smart control using different techniques. For example, [44] presents a simulation study for the energy management of a building with self-healing capabilities based on a decision policy-based algorithm without involving optimization. Similar applications using policy-based algorithms are [45, 46]. In [47], a reinforced-learning based approach for smart control of a building is proposed, showing some experimental results with long convergence time (+20 days). Likewise, [48–50] presents smart building control using deep reinforced learning or Q learning.

Another application of smart controls is in the automation of factory shop floors. In [51], the design of a smart shop floor is presented which is based on two two optimization layers, one for self organization of the agents, and another of self-adaptation of the events on the floor. Besides, [52] proposed a conceptual architectural design of a smart shop floor using digital twin. Another approach of smart systems for shop floors is presented in [53], where a linear optimization problem is proposed for the scheduling of human-robot devices tasks in a factory.

On the other hand, there are applications that combines digital twin and Industrial Artificial Intelligence towards a smart system approach. Some of these applications includes power systems like DC-DC converters, electric drive trains, electric vehicles, photo voltaic systems, and batteries [54–59], robotic manipulators [60, 61] plasma detection on industrial environments [62] and health monitoring [63, 64]. For all the cases, any of the following capabilities like fault detection, prognosis, or component degradation is included corresponding to direct uses of IAI methods.

Notice that the applications mentioned before that combines DT and IAI does not perform real-time execution of the analytics as well as real-time optimization method. However, there are few applications that performs real-time inference and use it for smart decision process. For example, [65] presents a DT application that performs real-time inference of Remain Useful Life of a drilling machine to correct the imbalance in the drilling axis. Besides, [66], performs real-time inference of the current state of a set of robotic manipulators configured via cloud service to perform real-time adjustment of the robotic systems tasks. Likewise, [67] combines big data with digital twin for the real-time monitoring and real-time decision for wind farms. Also, [68–70] presents the DT for CNC machines with self awareness interaction between physical and virtual parts of DT considering machinery tool condition assessment.

In some cases the concept of Smart System or Control is related to its simplest form to producing an automatic response based on a single rule, some threshold, or by the application of some basic stimulus to the system input. In [71–73], the authors claimed a smart control of the variation of an specific material properties like thickness using electrical signals and piezoelectric actuators. However, this notion of smartness is very limited considering that this smart approaches lacks of even a proportional closed-loop control and is limited only to actuation. Another particular case of smart control is presented in [74], where a neuro-fuzzy controller is employed as smart control system for concrete structures control.

1.1.4 What Is Not A Smart System?

The definition of smart system followed by the smart control engineering relies on the five attributes of a smart system: cognizant, taskable, reflective, knowledge rich and ethical. Likewise, in the previous section, the notions of smartness are diverse in the literature going from efficiency, automatic responses, or the use of artificial intelligence for particular applications. Thus, is natural that the following question surges: "what is not a smart system?".

This can be an extensive discussion depending of the point of view to be discussed. For this reason, the discussion will be limited to the five attributes of a smart system in order to provide answer this question. Thus, a system is not smart if it does not posses any of the five attributes of a smart system. Likewise, a system can be considered smart if it posses at least one of the smart system characteristics. Through the following examples, the notions of a smart and not smart system can be established:

1. **Not a smart system**: an automation process just based on the sequential execution of steps that activates or deactivate a set of actuators to perform an specific task. The result of the process is deterministic, any failure is easy traceable. In this case any of the five smart system characteristics is present on the system.

2. **Smart system**: The same automation process with a SCADA system integration that measure the process throughput, execution speed, quality metrics of the finished products, and integrates a set of machine learning algorithms that can infer if there is any bottleneck or process element that is affecting the product quality. In this case the system is smart, because the same automation process involves real-time analytics based on rich sensing that make the process aware of its current performance based on prior and new knowledge acquired.

3. **Not a smart system**: In an mature industrial process, a closed-loop controller like a PID, PI, state space, lead/lag compensator that just regulate the process variable without any learning mechanism aware of the closed-loop system performance. In this case the system produce an automated response

to a feedback variable but it does not integrate additional information in the decision process. Therefore, any of the smart system characteristics is present.

4. **Smart system**: The same industrial closed closed-loop system presented in the previous item that integrates a supervisory layer with a real-time optimization algorithm that is aware of the system performance and adjust the control gains and setpoints of the system to keep the optimal performance on the system through the calculation of real-time control performance assessment metrics. In this case the system is smart because it has the following properties: self aware (supervisory layer), cognizant (optimization algorithm), and knowledge rich (control performance assessment metrics.

1.2 Dissertation Objectives and Methods

Considering that Smart Control Engineering is an emerging field with high potential to enable smartness control systems, the main objective of this dissertation is developing a methodological framework for the implementation of the Smart Control Engineering supported by digital twin, Self Optimizing Control and Industrial Artificial Intelligence for its application in control systems. In order to accomplish the main objectives, the development of the SCE framework will be divided in the following specific parts:

1. Developing advanced virtual representation models for process control and mechatronic systems using digital twin.

2. Incorporating sensing, analytics, awareness, and inference capabilities to the system through IAI and DT.

3. Designing a real-time Self Optimizing Control strategy that use the knowledge obtained from IAI and DT to ensure the operation of the system with an acceptable performance including developmental features of the control structure.

4. Validating the SCE framework by simulation and experimental results on thermal systems.

In the first part, a DT development framework has been proposed and tested in this dissertation to build DT applications from real systems. This framework for DT is presented in chapter 3 and has been tested on thermal systems.

On the second part regarding to IAI methods, a deep learning method termed FaultFace was developed for fault detection using deep learning techniques, which has been tested for classifying different types of faults on ball-bearing joints. This method is treated in detail at chapter 2. Likewise, another fault detection methods has been developed and evaluated in Hardware in the Loop (HIL) configuration relying on Artificial Neural Networks for process control.

For the third part related to the Self Optimizing Control Strategy, Model Predictive Control (MPC) and Extremum Seeking Control (ESC) have been evaluated as Self Optimizing strategies. In particular, MPC has been successfully implemented for Single-Input Single-Output (SISO) and Multiple-Input Multiple-Output (MIMO) thermal systems running not only in HIL configuration but also as edge computing based implementation. However, a new type of SOC control based on real-time Optimization is being developed in order to act as outer optimization layer for controller parameter tuning. This algorithm is based on constrained gradient free optimization algorithms. The details on the MPC edge implementation as well as the current status of the new real-time SOC optimization algorithms are provided in chapter 4.

On the other hand, the assessment criteria to evaluating the level of smartness and developmental level of the SCE framework will be addresed using multimodel assessment and information theory methods as well as other indices based on the success rate of the SCE to ensure system performance under different conditions.

Finally, the proposed methods as well as the SCE framework will be tested for thermal systems built using peltier thermoelectric heating cooling elements in SISO and MIMO configurations.

1.3 Dissertation Contributions

The main contribution of this work is in defining the foundations of the Smart Control Engineering and creating a framework to enable smartness for different control systems in accordance to the increasing interest of smart control systems for Industry 4.0. Likewise, the following contributions can be obtained from this dissertation:

- A developing framework for DT applications adaptable to any system.
- A set of Industrial Artificial intelligence methods for fault detection, data analytics and prognosis applicable to closed-loop systems.
- A Self Optimizing Control method based on real-time Optimization executable by any embedded or edge computing device that can be incorporated next to the physical system and controller in order to satisfy the desired performance specifications even in the absence of a good model of the system.
- The development of the Globalized Constrained Nelder Mead optimization algorithm for self optimizing control and its convergence analysis.

1.4 Dissertation Outline

This dissertation is structured as follows. Chapter 1 introduces the research motivations towards smart control engineering and digital twin. Chapter 2 presents a conceptual background about industrial artificial intelligence and the breaking technologies required for it.

Chapter 3 presents the FaultFace methodology for fault detection on ball bearing systems as example of IAI and its comparison against several state of the art fault detection methods.

In Chapter 4, the DT is introduced, describing its main features, components, applications and challenges. Chapter 5 presents a systematic design framework for the development of Digital Twin applications for control systems and its application for the development of a virtual representation of thermoelectric systems. Chapter 6 present a set of digital twin enabled capabilities including prognosis and fault detection applied to thermal systems.

Chapter 7 introduces the self optimizing control framework employed for the smart control engineering, which incorporates a real-time optimization algorithm for the closed-loop performance monitoring and improvement of the system performance. Likewise, a convergence analysis of the algorithm is presented employing theoretical and numeric simulation results.

Chapter 8 presents a set of extensions of the self optimizing control framework including acceleration methods for the real-time optimization method algorithm employing fractional-order randomness and parallel computing with the simultaneous perturbation stochastic approximation (SPSA) algorithm.

Finally, the last chapter presents the conclusions and future research directions for the smart control engineering, digital twin and industrial artificial intelligence.

1.5 Results Reproducibility

The code for the methods presented in this dissertation can be found in the website `https://www.theedgeai.com/dtandscebook` where the users can donwload the codes for its use and evaluation.

Chapter 2

INDUSTRIAL ARTIFICIAL INTELLIGENCE BACKGROUND

2.1 Digital Transformation and IAI

Digital transformation can be defined as the process of changing society, economy, and industry by integrating information technologies that add value to the existing systems. It is often confused with Digitalization, but these are different phenomena. Digitization refers only to utilizing new technologies to organize, innovate, and increase the level of automation inside an industry. Thus, digital transformation creates new capabilities and processes, reduces capital and operating costs, improves decision-making, and creates new and better products and services for customers. The digital transformation is occurring worldwide and is powered by several breaking technologies like Artificial Intelligence (AI), Industrial Internet of Things (IIoT), Machine Learning (ML), Augmented Reality (AR), Virtual Reality (VR), 3D printing, advanced robotics, Cloud and Edge Computing, Digital thread and last but not less Digital Twin (DT) among others [75].

In the Industry 4.0 context, these breaking technologies play a fundamental role in enabling a knowledge-rich environment that can be used to develop real-time analytics for implementing smart systems with cognizant and reflective capabilities.

In that sense, the industrial use of artificial intelligence (AI) combined with cheap sensing and edge computing capabilities can act as an upper supervision layer to transform classic closed-loop controls into smart systems. This integration and its direct application into operating manufacturing processes are called Industrial Artificial Intelligence (IAI). In this chapter, a review of the Industrial Artificial Intelligence is presented to establish its relevance in towards the development of the smart control engineering.

2.2 What Is Industrial Artificial Intelligence?

According to [76], the concept of Industrial Artificial Intelligence was first introduced by Prof. Jay Lee on his course Intelligent Maintenance Systems (IMS) at the University of Cincinnati, with focus on increasing efficiency, reliability, reducing costs, and ensuring quality and special attention on predictive maintenance and productivity management.

In a general way, the Industrial Artificial Intelligence can be defined as the use and application of Artificial Intelligence techniques to the solution of industrial problems by creating value using unknown information extracted from the operation of an industrial process. Other definitions of Industrial AI are provided in the literature. In [76], IAI is defined a systematic discipline, which focuses on developing, validating and deploying various machine learning algorithms for industrial applications with sustainable performance. On the other hand, [77] describes Industrial AI as any application of AI relating to the physical operations or systems of an enterprise. Industrial AI is focused on helping an enterprise monitor, optimize or control the behavior of these operations and systems to improve their efficiency and performance.

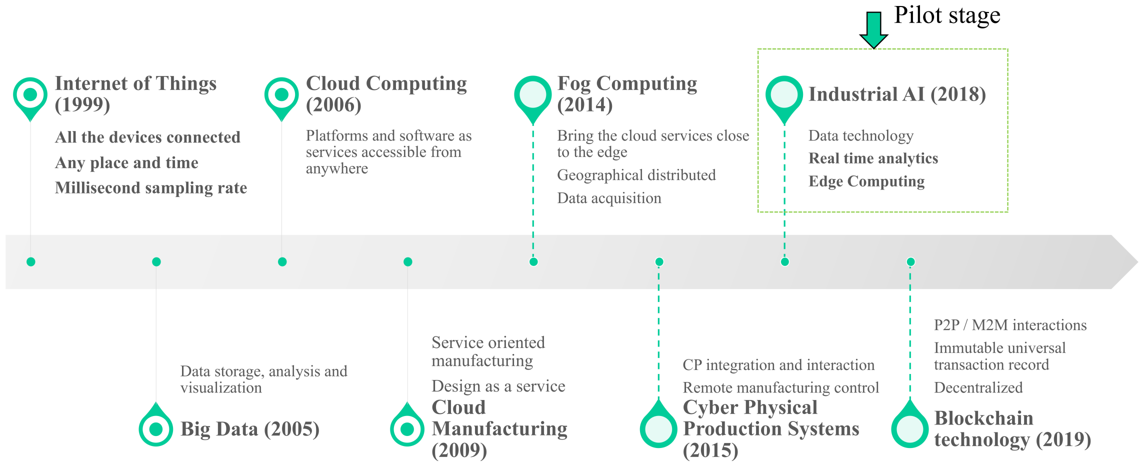
Based on these definitions, the purpose of IAI goes beyond than just apply artificial intelligence algorithms to an industrial processes. It means that the purpose of the IAI can be divided into two main goals:

1. Make visible problems hidden in the data domain to the process managers and engineers in order to avoid those problems before manifest on the system.

2. Generate new cumulative knowledge from a changing industrial process and apply its knowledge on a large scale analysis.

Thus, IAI not only provides visualization and solution for uncovered problems on a current industrial setup but also enables cognizant and knowledge rich characteristics of a smart system, which are important features for the smart control engineering.

Besides, IAI is supported by the evolution and integration of several breaking technologies as shown in Fig. 2.1. The initial milestone corresponds to Internet of Things, which refers to a series of connected devices that can communicate without human interaction by using a combination of sensors and connectivity technologies such as Bluetooth, WiFi, cellular networks and others, information regarding people and processes is shared and updated [78]. It is important to notice that IoT has been extended into Industrial IoT and has been powered thanks to cheaper sensing and actuation capabilities. Then, thanks to the wide internet based services adoption, new milestones appeared including big data, cloud and fog computing, until the consolidation of cyber physical systems. Thus, from 2018 the concept of Industrial AI begins with the concept of real time analytics supported of edge computing appears in an initial stage that is continuously powered by other breaking technologies like blockchain to reach the smartness requirements of IAI. It is important to notice that IAI is taking more interest among industry and academy, evidenced on the realization of the IEEE conference on Industrial Artificial Intelligence (IAI) which first edition was in 2019 and its four edition will be in 2022 as well as the new Springer journal Industrial Artificial Intelligence `https://www.springer.com/journal/44244` just released in April 2022 which is open for submissions for its first edition.


Pilot stage
Internet of Things (1999)
All the devices connected
Any place and time
Millisecond sampling rate
Cloud Computing (2006)
Platforms and software as services accessible from anywhere
Fog Computing (2014)
Bring the cloud services close to the edge
Geographical distributed
Data acquisition
Industrial AI (2018)
Data technology
Real time analytics
Edge Computing
Data storage, analysis and visualization
Big Data (2005)
Service oriented manufacturing
Design as a service
Cloud Manufacturing (2009)
CP integration and interaction
Remote manufacturing control
Cyber Physical Production Systems (2015)
P2P / M2M interactions
Immutable universal transaction record
Decentralized
Blockchain technology (2019)

J. Lee, J. Singh, and M. Azamfar, "Industrial Artificial Intelligence," 2019.

Figure 2.1: Evolution of IAI

2.3 Differences between AI and IAI

AI and IAI share several common characteristics like analysis methods, classification and detection algorithms, which are based on statistics, machine learning, and optimization theory. However, AI and IAI has important differences in the functional requirements, application fields, and algorithm scale (offline/real-time).

On the one hand, Artificial intelligence (AI) is a wide research field with with applications along all the human activities, including but no limited to image processing, natural language, robotics, recommendation systems, or self driving vehicles. Although AI provides solutions to challenges on the mentioned applications, aspects as its repeatability, consistency, or robustness makes industry users to consider some AI methods for its reliable use on real-time industrial processes. Besides, IAI is a systematic discipline focused on developing, validating, and deploying various machine learning algorithms for industrial applications with sustainable performance under established application conditions.

The main differences between AI and IAI are shown in Fig 2.2. The comparison is divided in four features, primary domain of application, user cases, data sources, and delivery mode.

In the case of AI, in a general use setup, the primary domain of application is the digital space, it means that is not related a physical asset. Likewise, the user cases cover activities like marketing, customer service or HR, which information come from several sources. In this scenario, AI is expected to order, process and perform inference from the data to know if there is any correlation or hidden information among the available dataset with commercial or business purpose. Likewise, its processing time is not real-time, requires huge amounts of computing power and employs classic delivery modes like web or mobile applications to delivery the analysis results to the user.

On the other hand, IAI primary domain is the physical, it means, that all the applications are related to an asset operating in the real world, which is constrained by an specific task performed in controlled environments. For this reason, the principal use of IAI is performing machine learning, optimization or real time analytics of the physical system, with precise goals that include predictive maintenance, quality control or HVAC automation. The data sources required by the IAI are clear and related to the process coming from rich sensing, SCADA systems or enterprise metrics. Likewise, one of the key differences with AI is that IAI requires producing an automatic response based on real-time inference performed on the system data streams, and its actions has a direct impact over the final product or users. For example the control of a HVAC system in a building needs to consider variables like real-time occupancy, kilowatt/hour cost, users comfort, as well as multiple temperature metrics. Thus any violation of physical and safety constraints for the system.

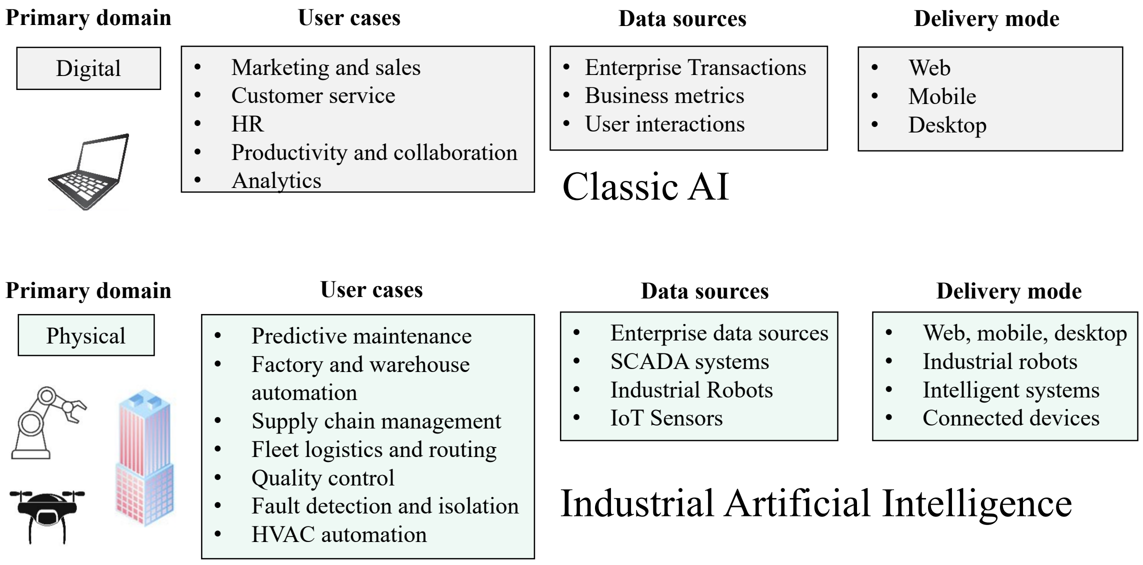


Figure 2.2: Differences Between AI and IAI

Based the differences between AI and IAI shown on Fig. 2.2, it can be say that IAI starting point is the AI and any IAI application should follow these characteristics know also as the 5th's [79].

1. Systematic: IAI levels needs to be defined from component, equipment, business in order to allocate the hardware/ software/ human resources to reach the added value levels expected.

2. Standards: it is concerned to the communication protocols, algorithms, data quality, real time analytics in order to have a standardized decision process.

3. Streamline: the workflow consist of a systematic methodology that stars from data acquisition, processing, decision making that enable fast development of the IAI applications.

4. Speed: determining IAI applications response (real-time, offline, long timescale) is critical to fragment tasks to customize system rapid response.

5. Sustainable: it means that the result of applying the same set of IAI algorithms and methods to similar datasets returns the same result each time is executed.

2.4 Why Using IAI?

AI has shown its capabilities for problem solving in several disciplines, including e-commerce, social modelling, financial technology or electronic payments. However, the main problem for classic AI solutions is that its behavior is not deterministic, does not follow uniform problem solving structures, varying from case to case, becoming a challenge for its application industry. Thus, using AI directly can be considered as an art with good but unique and not reproducible results.

In that sense, IAI as a systematic discipline based on the 5th's, providing a structured design procedure that is bounded into a physical application with an established domain knowledge that can add value to the current system by uncovering hidden problems, following a reproducible and sequential design and execution, with reproducible results from process to process, which is fundamental in industry to provide value not just for a particular implementation of a physical system, but into the global product design support and costumer experience [80].

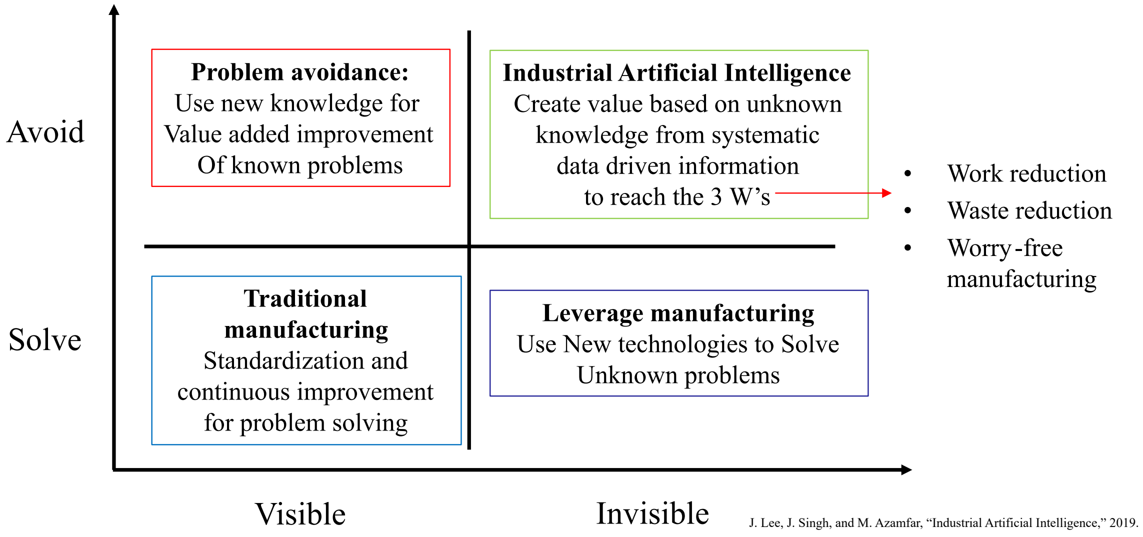
Likewise, using IAI has some additional benefits in addition to the structured and systematic approach for problem solving and added value in industry:

1. Enhanced, and predictive, situational awareness through modelling complex industrial processes to increase quality and reduce manufacturing time.

2. Better planning and decision-making business policies assessment to optimize management of complex systems

3. Improved automation efficiency and productivity by increasing production, lower labor cost, and reducing manufacturing errors

On the other hand, According to [81] a synthesis of the problem solving capabilities provided by IAI is shown in Fig. 2.3. As can be observed, the value of IAI is focused on deal with the avoidance of invisible problems by creating knowledge from a systematic data-driven approach in order to reach the 3W's in a manufacturing context: work reduction, waste reduction, and worry free manufacturing.



J. Lee, J. Singh, and M. Azamfar, "Industrial Artificial Intelligence," 2019.

Figure 2.3: IAI Impacts on Industry 4.0 Problem Solving

2.5 Enabling Technologies for IAI

Integrating IAI into an industrial environment requires the synergy of several technologies that enabling AI, sensing, processing, and decision capabilities. Thus, IAI requires a redefinition of the Computer Integrated Manufacturing model (CIM) to integrate and leverage IAI capabilities. A suitable model is proposed by [76] called the 5C- cyber physical system architecture which is shown in Fig. 2.4. The 5C model is composed of five layers connection, conversion, cyber, cognition and configuration, which are grouped into three core technology layers. The first layer corresponds to the data technology that contains all the required sensing technologies to enabling knowledge richness for IAI applications. Among these technologies can be found smart sensing provided by Industrial Internet of Things (IIoT) supported by embedded and edge computing devices to monitor and execute real time algorithms.

Besides, the analytics technology layer is composed by all the machine learning and data analytics algorithms that enables cognizant capabilities in an IAI system. Finally, the operation technology layer is composed by all the self optimizing, adjustments and configuration algorithms that supervise the IAI application performance and execute automatic actions over the lower layers sensors, actuators, and controls.

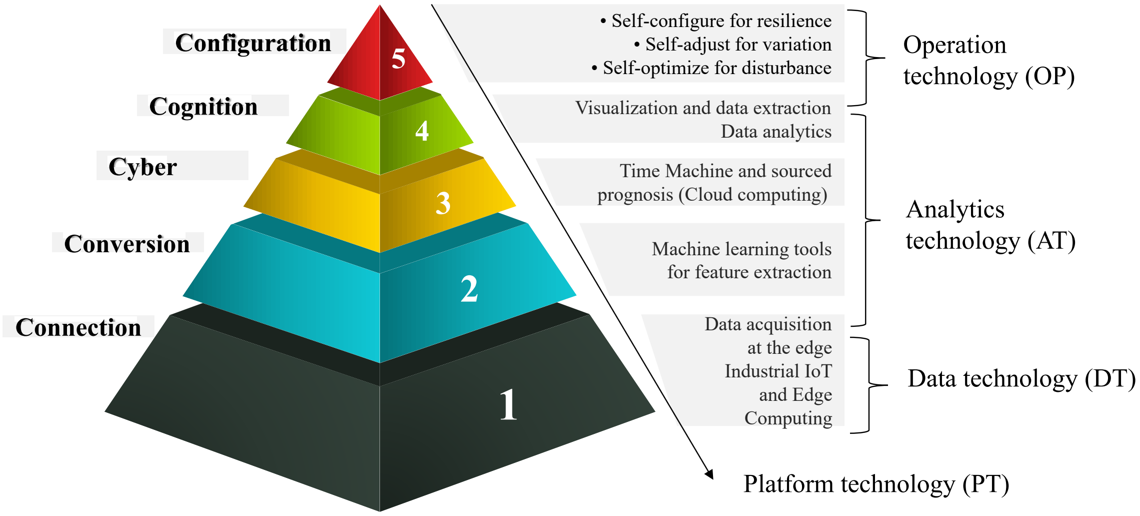


Figure 2.4: Computing Architecture for Full Scale IAI Implementation [76]

A block diagram representation of the 5C cyber physical system automation model in terms of the computing architecture is shown in Fig. 2.5. As can be observed, for the 5C model, the data technology layer relies on embedded and edge computing devices which interact directly with the physical assets of the system including IoT sensing and actuators. For the operation technology and analytics technology layers, a distribution between fog and cloud computing is established to perform analytics and self regulation considering an scenario with hundred or thousands instances of IAI enabled systems like simultaneous oil rigs or semiconductor manufacturing tools operating worldwide. It is important to notice that for this integration scale high computing capabilities for cloud and fog computing are required.

In this dissertation, the focus for the IAI applications are contained into the data technology layer, relying on edge and embedded computing devices. It means not only the sensing and actuation features for the physical system but also all the analytics and self optimization technologies are configured to be executed into the edge computing stage, so it can be integrated to an operating industrial process leveraging its current architecture and information available, enabling smartness on the system. A description and discussion over some alternatives of embedded and edge devices will be presented on Chapter 6.

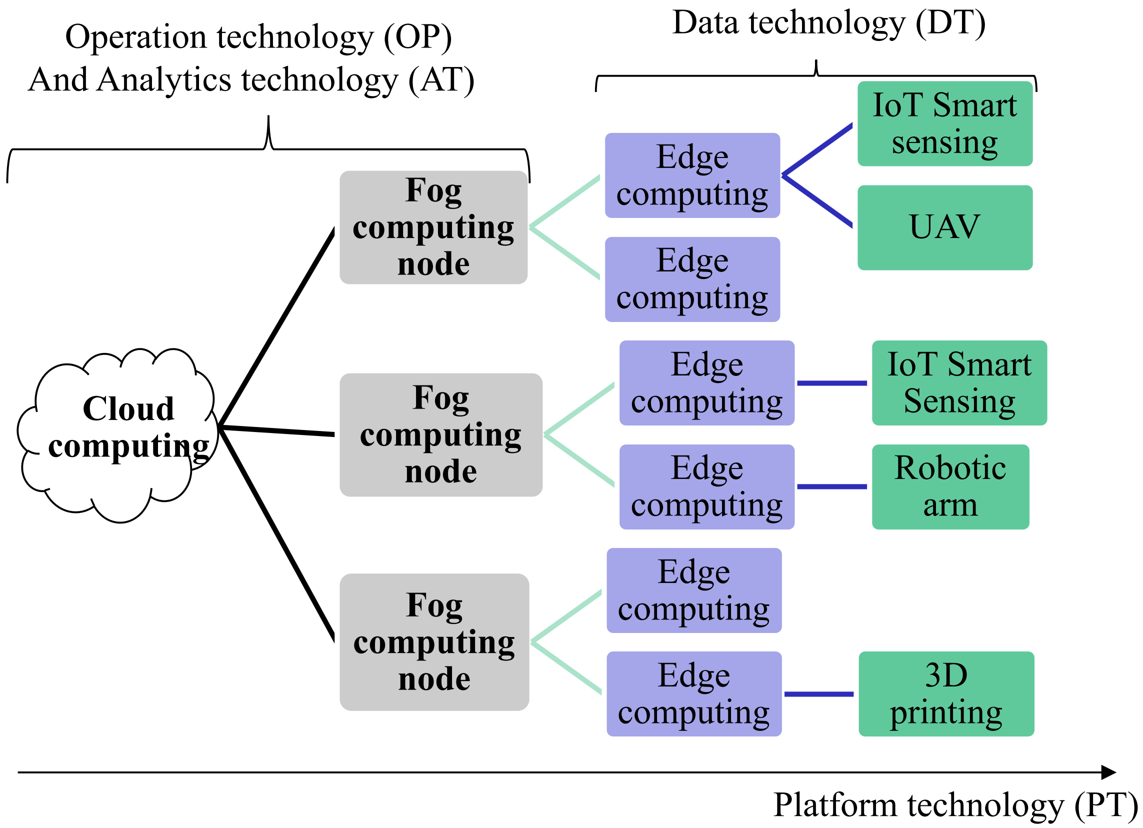


Figure 2.5: 5C Computing Architecture for Full Scale IAI Implementation

2.6 Chapter Summary

This chapter presented the concept of industrial AI, its definition, characteristics, goals, challenges, and breaking technologies for its implementation. Likewise, the difference between AI and IAI was established, reaffirming that the goal of the IAI goes beyond the use of AI methods but uncover hidden knowledge from an industrial process that provide added value that is useful to understand the current status of the system as well as taking actions to improve it satisfying a set of desired operation conditions. Thus, it is possible to say that IAI is an breaking technology towards the implementation of the smart control engineering, enabling two of the features of a smart system on an industrial process: cognizant and knowledge rich. In the next chapter, a case study will be presented of the use of IAI for fault detection on mechanical systems.

Chapter 3

AN IAI CASE STUDY: FAULTFACE, DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORK (DCGAN) BASED BALL-BEARING FAILURE DETECTION METHOD

3.1 Introduction

In control engineering, failure detection is the branch concerned on monitoring a system, identify the possible failures, and notifying its kind and location using only the available input and output data streams of the system. So, it makes possible detecting not only the system failures but also discovering hidden behavior patterns, which are reflected in plant stops that generate productivity and money losses for the companies. Also, failure detection is a challenging task for different reasons like the system complexity, the required prediction speed response, the size, and consistency of the dataset, or the number of performance indices evaluated. In the literature, there are several applications of machine learning and deep learning techniques for failure detection of industrial processes. In [82], a support vector machine (SMV) is employed to detect failures inside a wireless sensors network due to damages in the devices or faults in the communication. On the other hand, [83] shows the use of unsupervised K-means algorithm to detect failures on 3D stacked integrated circuits. In [84], a distributed machine learning classification algorithm to detect attacks into the power grid is shown, which use the K-means algorithm, SVM, decision tree, among other methods. Another application on semiconductors failure detection is given by [85], where an assessment of different Machine learning models is performed to detect several types of failures during the wafer manufacturing process. Also, [86] presents a failure detection algorithm that employs logistic regression models to detect failures due to mechanical component fatigue. On the other hand, [87] shows a prognosis method for shackles employing logistic regression to determine the decision boundaries for each failure. There is also an application of machine learning techniques for failure detection on directional drilling of oil wells [88], where the training process was performed using significant historical data from more than 80 oil wells for training a boosted gradient algorithm. Besides, machine learning can also be employed for Cyber-Physical Systems. In [89], the random forest method is used to perform disturbance detection on a smart grid system. Also, [90] present a survey of various machine learning algorithms like SVM, Logistic Regression, and random forest for failure detection on the Internet of Things (IoT) sensor networks. As can be observed, the applications presented on [82] - [90] employs time series analysis, machine learning, or deep learning methods for training the classifiers and perform the failure detection. Notice that these applications have a good quality dataset, allowing a correct training of the failure detection algorithms.

However, on industrial processes, there is not always available a balanced, complete, or consistent dataset related with the failure behavior due to the longer time required to run a complete cycle of the process. Likewise, the cost and risk of running a process to get data from a failure behavior may produce more significant damages in the physical system. For this reason, the training of classifiers for industrial process sometimes relies totally on simulated data. For example, [91] shows an application where a machine learning algorithm is employed for early failure detection on CNC machines, which is trained using an identified state-space model of the system to generate the failure and nominal data of the machine. Also, [92], employs a simulation model of an electric car power drivers to train a machine learning model for failure detection based on an artificial neural network. The main challenge for this approach is that a representative model of the system is not always available for training a machine learning model accurately.

So that, fault detection in industrial processes with unbalanced datasets is an active research topic, which combines machine and deep learning techniques for fault classification and additional feature mining over scare fault data. For example [93], presents the use of bilayer Convolutional Neural Networks (CNN) for fault detection in chemical processes with unbalanced datasets, which is based on a exhaustive feature mining of the available data using wavelet packet decomposition. In [94], a CNN network combined with an initial normalization kernel is employed for fault detection in bearing mechanisms, mining additional data with the CNN convolution layers. Also, [95] presents the use of fusion autoencoders for skewed, incomplete, unbalanced datasets, with several denoising and resampling stages for feature extraction applied to fault detection on bearing elements. Notice that these works relays on deep feature extraction to compensate the unbalanced and incomplete dataset in order to improve the fault detection accuracy.

On the other hand, Generative Adversarial Networks (GAN) [96], proposed by Goodfellow in 2014, expand the reaches of Artificial Intelligence (AI) allowing the creation of new datasets based on small amounts of available data. These generated data is not only closer to the original but also can produce images combining different features extracted from the original dataset. For this reason, there are many applications of the GAN networks for classification problems. For example, [97] shows the use of GAN networks for the artificial generation of synthetic data for training a detection model of Jellyfish swarms. In [98], a multi-class spectral GAN network is employed for the classification of multispectral images. Also, in [99], a Multiview GAN network is proposed for pearls classification, increasing the accuracy regarding classical methods. Likewise, [100] shows the application of GAN for medical images generation and classification for different body diseases. For failure detection on industrial processes, some reported works are using different GAN networks for dataset generation. In [101], the fault diagnosis is performed for a planetary gearbox system using GAN networks and Stacked Denoising Autoencoders. Besides, [102] and [103] present unsupervised classification algorithms for rolling bearings in combination with GAN networks, which contains an unbalanced dataset. For all the GAN networks applications presented above, the feature extraction process is performed using algorithms like Autoencoders, external to the GAN network.

Nonetheless, there is a particular implementation of the GAN network known as Deep Convolutional Generative Adversarial Networks (DCGAN), which incorporate the automatic feature extraction layers for the images with the GAN network. Thus, all the feature extraction and training process is performed using only this network. There are some applications that use DCGAN networks for medical image generation [104] [105], or image augmentation [106]. However, for failure detection, there are few applications of the DCGAN reported like [107] where is employed for intrusion detection.

This chapter presents a fault detection methodology called FaultFace, which is employed for the failure detection on ball-bearing joints for rotational shafts using DCGAN networks for dataset balancing. The system to be analyzed is the Case Western Reserve university benchmark [108], which is employed to evaluate different ball-bearing joints faults on a rotational shaft axis.

A face portrait of the vibration signals is obtained for the nominal and failure behaviors, which correspond to a time-frequency representation of each signal. Six different FacePortraits are obtained from the vibration data, using Continuous Wavelet Transformation (CWT) with Morse Wavelet [109], Wavelet transformation with HAAR Wavelet [110], Circular Matrix Reading (CMR) [111], Toepliz matrix [112], Hankel matrix, and Gramian matrix [113].

Considering that the ball-bearing dataset is unbalanced and contains few samples of nominal and failure cases, the DCGAN network is employed to generate new face portraits for the nominal and failure cases. Then, the balanced dataset generated by the DCGAN is used to train a Convolutional Neural Network (CNN) that perform the failure detection task. The structural similarity index (SSIM) is employed to measure the quality of the new dataset generated using the DCGAN network. Also, another balanced dataset is produced using a GAN network to compare not only the performance of the DCGAN network but also the overall performance of the faultFace methodology. The obtained results of the faultFace methodology are evaluated using the confusion matrix for the DCGAN and GAN datasets. The faultFace methodology is compared with a support vector machine (SVM) with Autoencoder and a Long Short Term Memory network (LSTM). Likewise, it is compared with other reported classification methods employed for the CWRU ball-bearing dataset.

This Chapter is structured as follows. Section II presents the DCGAN and CNN networks employed for fault detection. Section III presents the ball-bearing benchmark system and the description of the nominal and failure behaviors of the system. Section IV introduces the faultFace methodology which involves the procedures used for facePortraits generation, the training of the DCGAN network for dataset balancing, the CNN training based on the new face portraits produced by DCGAN as well as the performance assessment of the methodology using the confusion matrix as well as a quality evaluation of the generated balanced dataset using the DCGAN network. Section V shows a variant of the faultFace methodology using the GAN network for dataset balancing instead of the DCGAN network as well as the performance comparison between both approaches. Section VI presents a comparison of the faultFace methodology with other proposed methodologies for failure detection of this system including LSTM and SVM with Autoencoder. Finally, conclusions and future works are presented.

3.2 Generative Adversarial Networks (GAN)

According to [96], a Generative Adversarial Network (GAN) is a deep learning model based on two independent neural networks called generator (G) and discriminator (D), which are involved in a competition. The generator (G) network creates a new probability distribution $P_G(x)$ based on a prior defined probability distribution $P(x)$, which can be considered as a black box. On the other hand, the discriminator (D) network determines the difference between the $P_G(x)$ and $P(x)$. Once the discriminator cannot distinguish between $P_G(x)$ and $P(x)$, it means that the generator learns the black-box behavior of $P(x)$. Notice that G and D are trained simultaneously in order to improve the estimation of $P_G(x)$ as well as the differentiation of $P(x)$ against $P_G(x)$. So that, the GAN network can be defined as a minimax optimization problem as given by (1), where $x \sim P_{data}(x)$ is the data from the original distribution $P_(x)$ and $z \sim P_z$ is the data from the distribution generated by G.

$$\min_G \max_D V(D,G) = E_{x \sim P_{data}(x)}[LogD(x)] + E_{z \sim P_z}[log(1 - D(G(z)))]. \tag{3.1}$$

From (1), the GAN network tries to maximize the probability $log(D(G(z)))$ of an accurate classification by D, while simultaneously trying to minimize the error on G by $log(1 - D(G(z)))$. A block representation of the GAN network is presented in Fig.1. As can be observed, the generator network is feed with a random noise distribution to generate $P_G(x)$, which feed the discriminator network to determine whether the synthetic data produced by the generator is real or fake, and based on that result perform the training of the generator and the discriminator again. The minibatch stochastic gradient descent is employed as a training algorithm for the GAN network [96,111]. For the GAN network, the optimal training point is reached when $P(x) = P_G(x)$. Besides, the training process of G and D is performed simultaneously, reducing K times the gradient for training D and once for G, considering that the time for training D is higher than G.

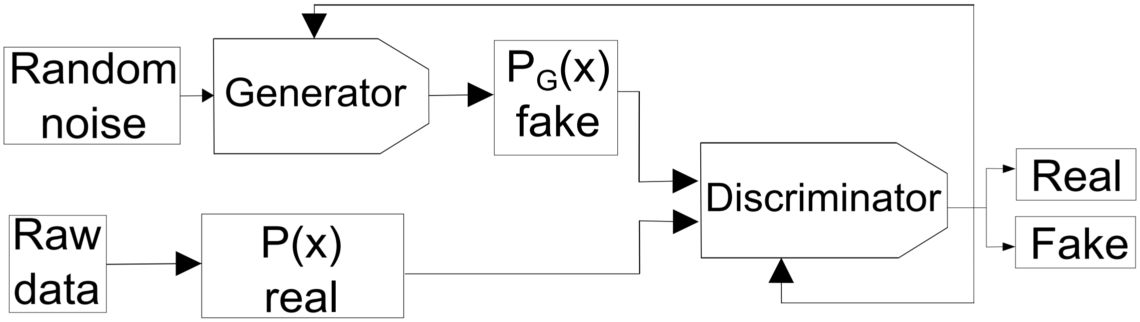


Figure 3.1: Block diagram of GAN Network

3.3 DCGAN Network

The deep convolutional GAN network (DCGAN) is a variation of the GAN network, where the generator and discriminator multilayer perceptron neural networks are replaced by a convolutional neural network to exploit its image processing capabilities. According to [114], the CNN networks employed on the DCGAN network architecture should have some specific features to ensure a stable training process of the generator and discriminator. The first one is replacing the pooling layers with strided convolutions for the discriminator, and fractional-strided convolutions for the generator. The second one is eliminating full layers connections in the hidden layers of the generator and discriminator, just leaving the output layer fully-connected. The third one is to apply batch normalization to all the hidden layers expect by the input and output layer on the generator and discriminator, ensuring zero mean and unit variance. The fourth one is to use the ReLU activation function for the input and hidden layers, and Tanh activation function for the output of the generator to accelerate the training process. Finally, the LeakyReLU activation function is recommended for all the layers on the discriminator.

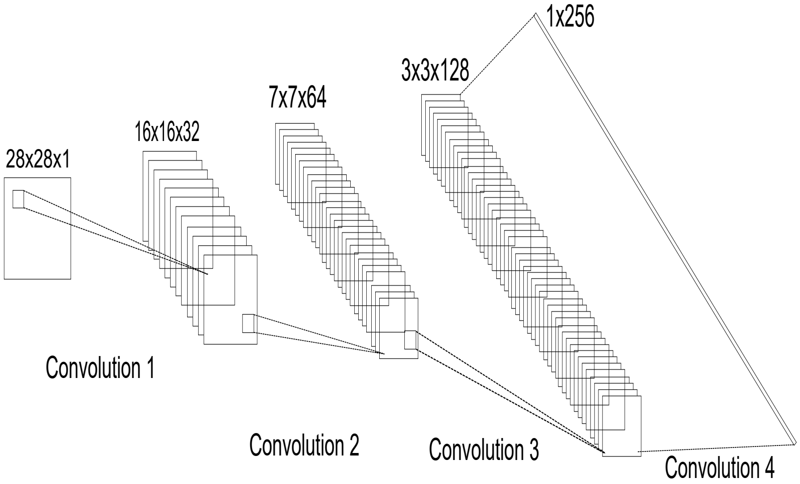


Figure 3.2: Discriminator CNN [90]

The discriminator network employs the standard structure of a CNN presented in Fig.2. As can be observed, the discriminator CNN has an input layer of 28x28. Also, three hidden layers are employed with LeakReLU as the activation function. Finally, the output layer has a dimension of 256x1, which is fully connected with a Sigmoid activation function for the real and fake data classification. The kernel size for the CNN is 3x3 in all its layers with striding of 2 for all the hidden layers except by the output with striding of 1.

Besides, the generator network structure differs from the discriminator CNN, as shown in Fig.3. As can be observed, the generator CNN works perform the inverse CNN process. Initially, the sample random noise goes from a minibatch of Gaussian Random noise samples projected into a bigger feature space. After that, a 3x3 convolutional filter is applied, and the result is upsampled using a striding factor of 2, resulting in a higher-dimensional space. Thus, after some convolution layers, the generator returns a 2D image representation of the data. In this case, the minibatch has an initial size of 100 samples, which is projected into a 128 feature dimensional space representation to apply three hidden convolutional layers with an upsampling factor of 2 that generate a 28x28 pixels 2D grayscale image in the output layer. That will be compared with the discriminator to perform the DCGAN network training.

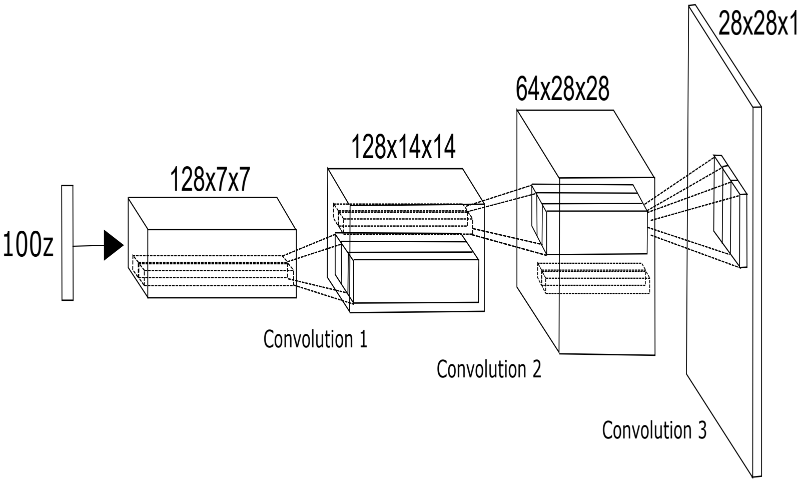


Figure 3.3: Generator Inverse CNN [111]

3.4 Case Study: Ball-Bearing Benchmark System

The ball-bearing benchmark system from Case Western Reserve University [108] and Rockwell automation were selected for testing the FaultFace method. The benchmark system is presented in Fig.4. It is composed of two DC motors of $2Hp$ running at 1700 RPM which rotational shafts are joined using a ball-bearing coupling. This reference system is designed for testing different ball-bearing couplings diameters as well as inducing failures on the couplings using electrical pulses. For this system, the diagnosis signal is the axis vibration measured with accelerometers for different nominal and failure operating conditions.

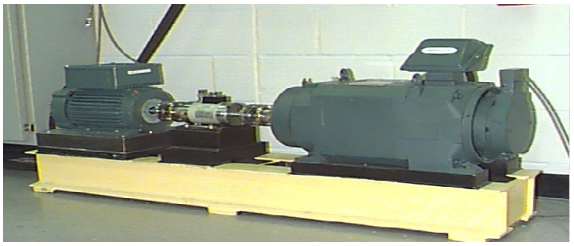


Figure 3.4: Ball-Bearing Benchmark system [108]

Ball-Bearing Coupling Failures

A ball-bearing coupling is presented in Fig.5. As can be observed, it is composed by an outer race (A), an inner race(B), the balls between the inner and outer race (C) to reduce the friction over the rotational shaft (D). According to [108], different failures can be induced into the ball-bearing benchmark system. The first failure corresponds to damage on the inner race of the ball-bearing, the second one is related to failures on the outer race due to the load position in the shaft (centered, opposite, orthogonal), and the third case is related to damages on the bearing balls. Table 1 summarize a set of possible failures for the benchmark system. As can be observed, the plant supports two different types of ball-bearing couplings denominated fan-end and drive-end with the possibility of generating different failure diameters.

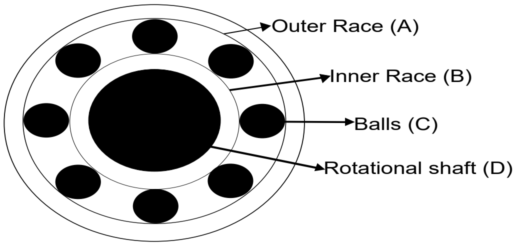


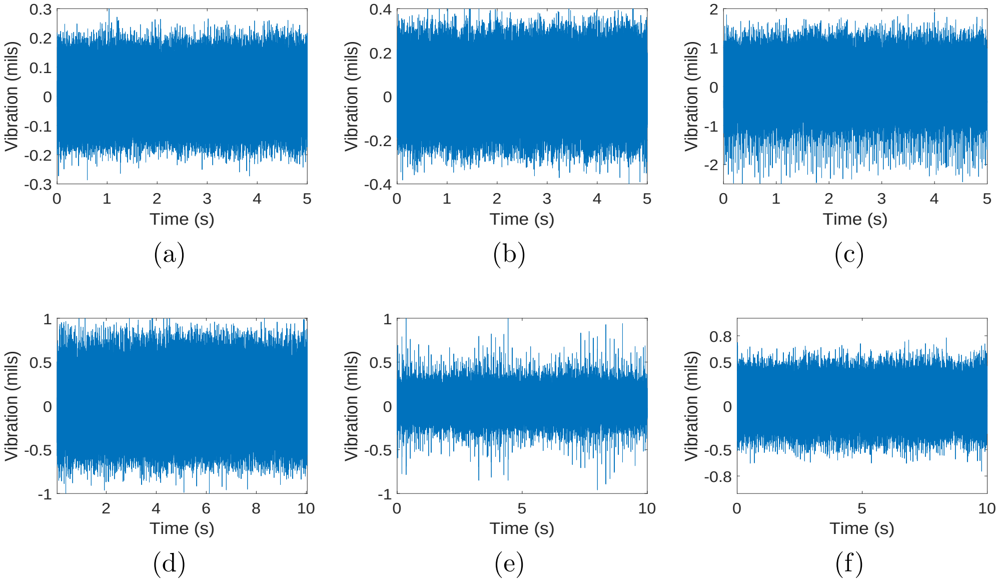
Figure 3.5: Ball-Bearing Coupling

Table 3.1: Ball-Bearing Benchmark System Failures

Bearing type	**Fault location**	**Fault diameters**
Fan end Drive end	Ball, Inner race, and Outer race load (Center, Opposite, Orthogonal)	0.004, 0.014, 0.028

Ball-Bearing Dataset

The Ball-Bearing benchmark system is composed of 114 datasets of the rotational shaft vibration signal. Four datasets correspond to the nominal operation of the ball-bearing coupling for fan end and drive end couplings. The remaining datasets are for the different failure behaviors of the system presented in Table.1. The data format is given as time series with sample rates of 12 kHz for the fan-end and 48 kHz for the drive-end Ball-Bearings. From the features presented above could be inferred that the Ball-Bearing Benchmark system is unbalanced with different sample rates. An example of nominal and failure behaviors time series are presented in Fig.6. It can be observed that the nominal and failure datasets were sampled by different times, and the failure vibration signals have a bigger amplitude than the nominal data for all the five failure cases. Therefore, the dataset should be balanced to obtain good performance from the failure classification technique. In this application, the benchmark dataset is divided into six categories for classification and training purposes. The first one is denominated nominal data considering all the nominal datasets for different ball-bearing types and sampling times. The other categories, corresponding to the failure cases are divided into the ball failure case, inner race case, outer race with centered load case, outer race with opposite load case, and outer race with orthogonal load case.



(a) (b) (c)

(d) (e) (f)

Figure 3.6: Vibration signals a) Nominal b) Inner Race c) Ball d) Outer race: Load Center e) Outer Race: Load Opposite and f) Outer Race: load Orthogonal

3.5 FaultFace Methodology

The block diagram representation of the FaultFace methodology is presented in Fig.7. Initially, the original unbalanced dataset of the ball-bearing nominal and failure behaviors is acquired. Then, the FacePortrait of the signals is determined. After that, the nominal and failure FacePortraits are introduced into the DCGAN network to generate new face portraits in order to balance each dataset. Next, using the new balanced datasets for nominal and failure behaviors generated from DCGAN, a Convolutional Neural Network is trained for failure detection. Finally, the obtained results are evaluated using the confusion matrix.

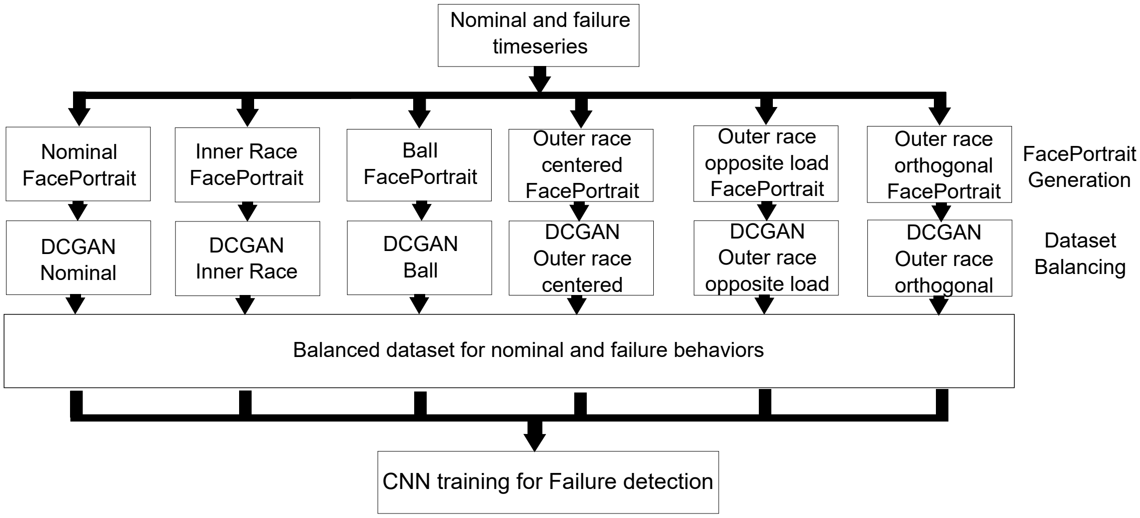

Nominal and failure timeseries
Nominal FacePortrait
Inner Race FacePortrait
Ball FacePortrait
Outer race centered FacePortrait
Outer race opposite load FacePortrait
Outer race orthogonal FacePortrait
FacePortrait Generation
DCGAN Nominal
DCGAN Inner Race
DCGAN Ball
DCGAN Outer race centered
DCGAN Outer race opposite load
DCGAN Outer race orthogonal
Dataset Balancing
Balanced dataset for nominal and failure behaviors
CNN training for Failure detection

Figure 3.7: FaultFace Methodology

3.5.1 FacePortraits Generation

The face portrait is a 2D image representation of a time series, which can be obtained employing time-frequency techniques. Six different face portraits representation for each signal ball-bearing vibration signal are obtained. The first one employs Continuous Wavelet Transformation (CWT) using the Morse wavelet [109]. The second one employs the Haar wavelet (HAAR) [110] instead of Morse wavelet. The third method employed is called Circular Matrix Reading (CMR) [111]. It consists of reading the time series, normalize regarding its maximum value and multiply each value of the time series by 255 to obtain a grayscale image of the time series, where each pixel represents a single value of the vibration signal. The fourth faceportrait uses a Toeplitz matrix transformation [112]. It produces a symetric Toepliz matrix from the normalized vibration timeseries, where the elements along a diagonal have the same value. Likewise, the fifth faceportrait employs a Hankel transformation matrix [113]. Unlike Toeplitz matrix, this transformation produce a symetric matrix where the antidiagonals elements are equal. The sixth faceportrait is generated using the Gram matrix G [113], that is defined as all the possible inner products of m vectors that conforms the set V. It is defined by $G = A^T A$, where A is a matrix with all the m vectors of V distributed as columns. In this chapter, the m columns for the matrix A were generated splitting the normalized vibration timeseries into equal length vectors. An example of the obtained face portraits for nominal and failure datasets is shown in Fig.8. As can be observed, all the vibration FacePotraits were transformed into a 28x28 pixels grayscale image that can be employed for training the DCGAN network for dataset balancing. Notice that each FacePortrait contains particular features that allow differentiating between nominal and failure behaviors. These features will be considered during CNN training in order to perform failure detection.

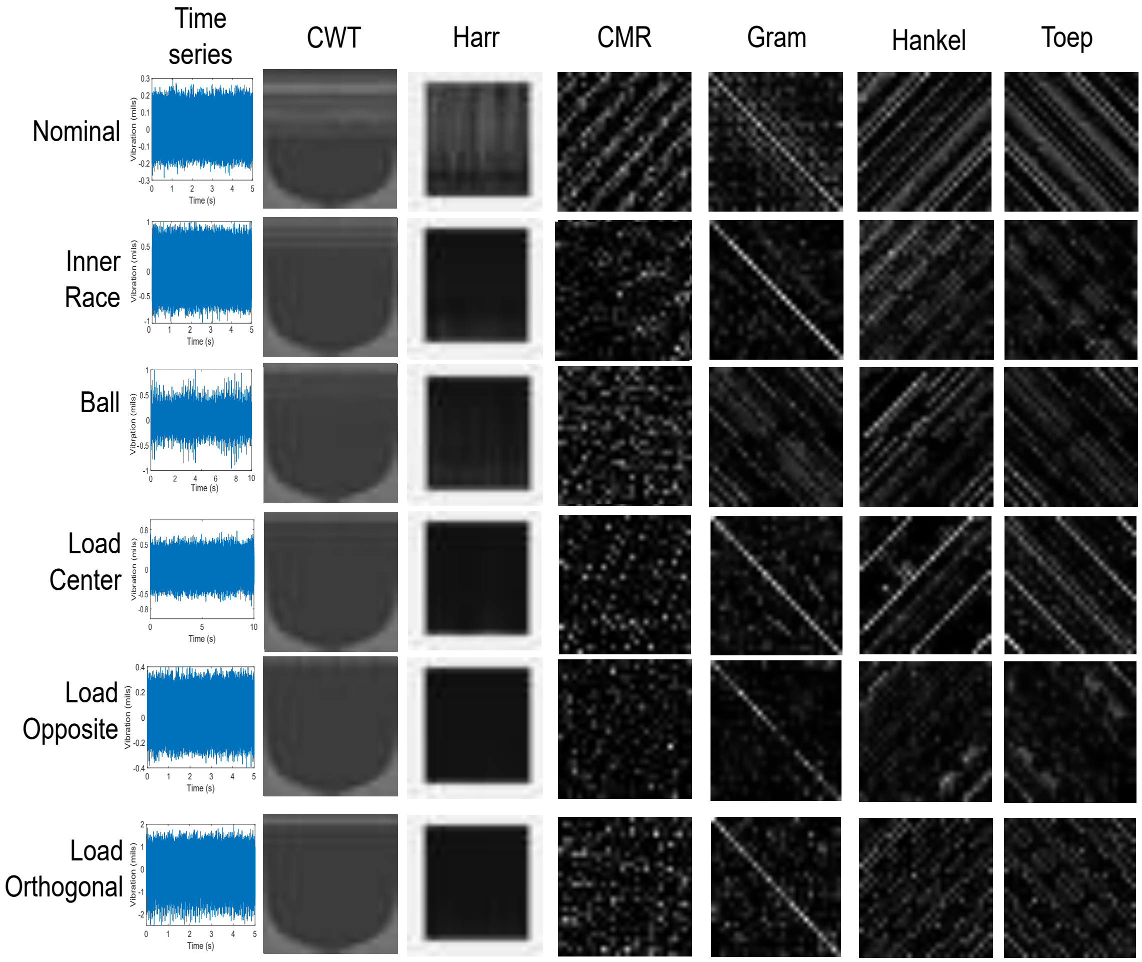


Figure 3.8: Vibration signal and Obtained FacePortraits for Nominal Behavior, Inner Race, Ball, Load Center, Load Opposite, and Orthogonal Load Failures

3.5.2 Dataset Balancing Using DCGAN Network

The face portraits for nominal and failure behaviors shown in Fig.3.8 are introduced into a DCGAN network to produce a balanced dataset. In this case, a individual DCGAN network was trained for the nominal behavior as well as for each fault case. For all the cases, each DCGAN networks were implemented in Tensorflow using the Keras framework and were trained with the minibatch stochastic gradient descent algorithm, using the Adam optimizer with a learning rate of 0.0001 for 40000 epochs. The results of the DCGAN network training for the CWT, Haar, and CMR faceportraits are shown in Fig.3.9. Likewise, the Gram, Hankel, and Toeplitz faceportraits are shown in Fig.3.10. As can be observed, the first epoch of the DCGAN generates an image that does not represent the face portrait and looks like random noise for all the cases. However, after 10000 epoch of training, the DCGAN networks begin to produce consistent face portraits, and after 40000 epochs, the result is similar to the original FacePortraits. Once the training process finishes, a balanced dataset is produced, which is composed of 1000 images of nominal behavior, and 1000 images for each failure behavior, it means a total of 6000 images. Notice that the original dataset only contains 114 time-series data, which only four represent the nominal behavior of the system.

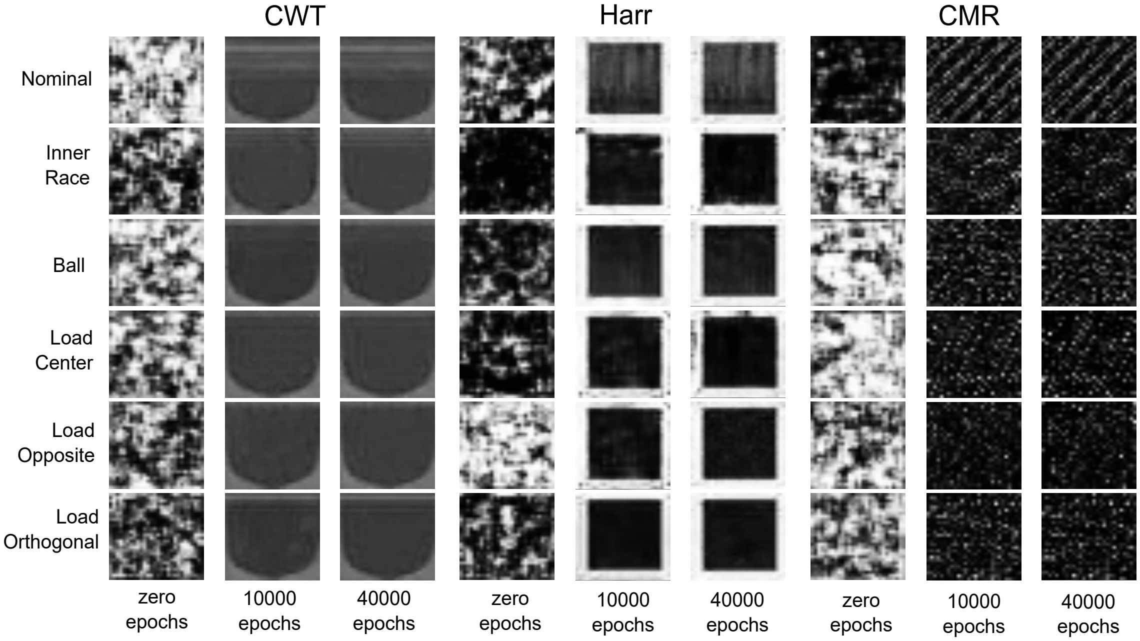


Figure 3.9: DCGAN generated CWT, Harr, and CMR Face portraits for Nominal, Inner Race, Ball, Load Center, Load Opposite, and Load Orthogonal at Zero, 10000, and 40000 Epochs

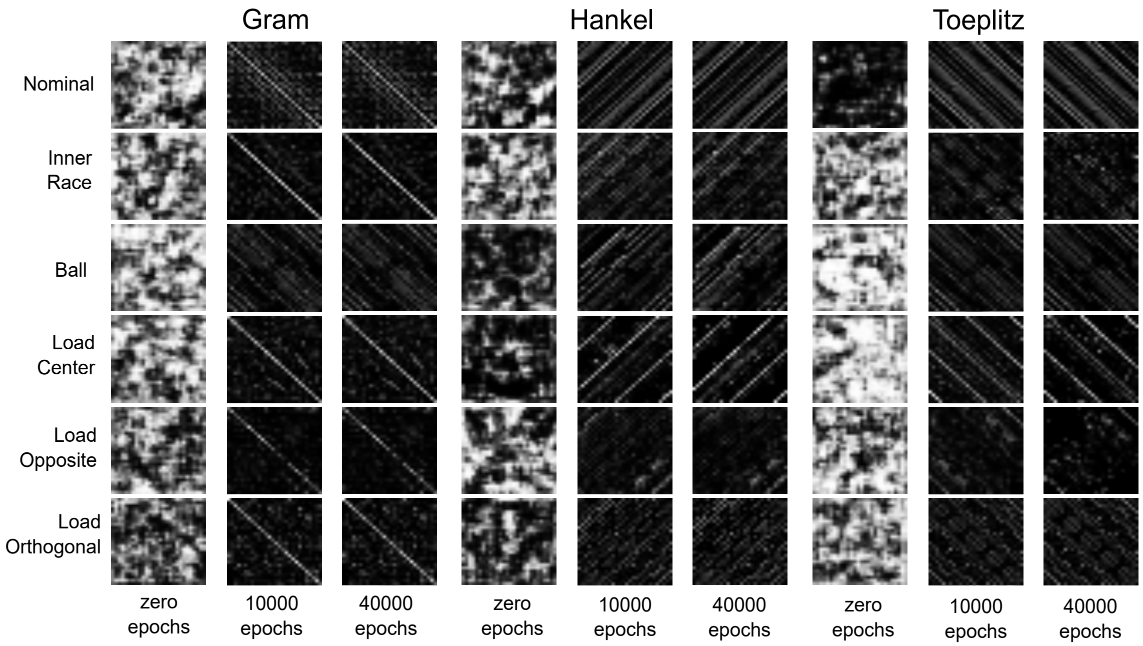


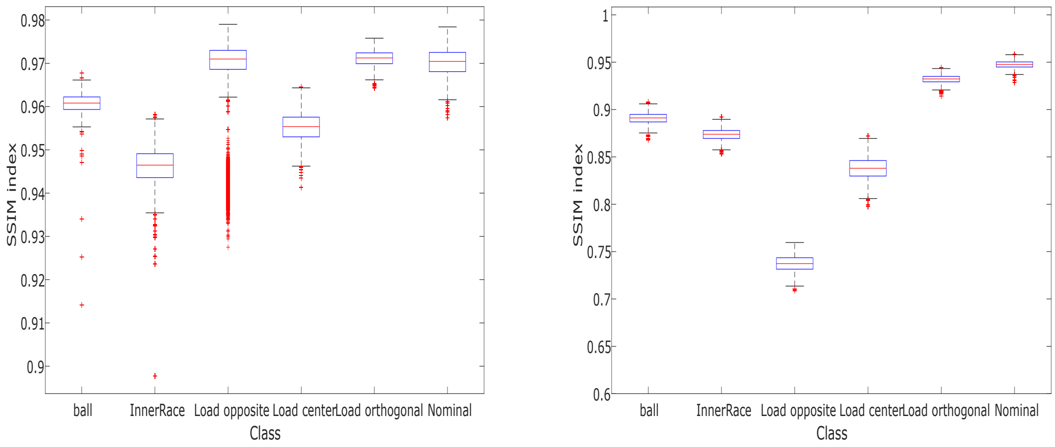
Figure 3.10: DCGAN Generated Gram, Hankel, and Toeplitz Faceportraits for Nominal, Inner Race, Ball, Load Center, Load Opposite, and Load Orthogonal at Zero, 10000, and 40000 Epochs

3.5.3 DCGAN Faceportrait Quality Assessment

A quantitative quality assessment of the balanced dataset produced by the DCGAN networks is performed to evaluate its accuracy for recreating the data distribution of the faceportraits. Thus, the structural similarity index (SSIM) is employed to measure the similarity of the generated faceportraits regarding to the original dataset. According to [115], the SSIM is given by (3.2) for two images x, and y, where μ_x, μ_y σ_x^2, σ_y^2, $sigma_{xy}$ correspond to the means, standard deviations and cross-covariance of x and y. Likewise, C_1, C_2, C_3 are the regularization constants given by $C_1 = (0.01L)^2$, $C_2 = (0.03L)^2$, and $C_3 = C_2/2$ with $L = 255$ as the dynamic range for grayscale images. The SSIM index (3.2) returns a normalized value between [-1,1] where 1 represents a perfect matching between images x and y.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.2}$$

In this chapter, the SSIM index is calculated for each single image of the original faceportrait dataset with respect to each single image generated for the DCGAN network for each case and faceportrait in order to see the distribution of the generated faceportraits. As example, Fig.3.11 shows a boxplot of the SSIM index calculated for the nominal CWT and Hankel faceportraits for the nominal and fault behaviors. As can be observed, the mean value for the SSIM index for the CWT faceportrait is above of 94% indicating a high similarity between the generated and the original dataset. Also, the deviation of the data is ±3%, which also indicates that the balanced dataset can improve the detection range of the faultFace methodology. In the case of Hankel faceportrait, the average SSIM index variates between 74% to 95%. In this case, the balanced dataset using Hankel faceportrait still performs a good representation of the system. In addition, the data distribution is symmetric and follows a normal distribution, considering that the DCGAN network uses a normalized Gaussian random seed to generate the initial distribution in the generator to produce the new faceportraits. Table 3.2 summarize the mean and standard deviation for all the faceportraits, which behavior is similar for all the generated faceportraits.



(a)

(b)

Figure 3.11: SSIM Index for Quality Assesment of the Balanced Dataset Produced by the DCGAN Network for the a) CWT and b) Hankel Faceportraits

Table 3.2: SSIM Index Normal Distribution for the DCGAN Generated Faceportraits

Faceportrait	Statistic	Ball	Inner Race	Load Center	Load Opposite	Load Orthogonal	Nominal
CWT	mean	0.961	0.946	0.969	0.955	0.971	0.97
	std	0.003	0.005	0.008	0.003	0.002	0.003
	Range	0.054	0.06	0.052	0.023	0.011	0.021
CMR	mean	0.945	0.91	0.76	0.864	0.959	0.981
	std	0.003	0.006	0.168	0.010	0.003	0.001
	Range	0.023	0.038	0.752	0.071	0.016	0.008
Gram	mean	0.027	0.892	0.721	0.865	0.939	0.936
	std	0.004	0.006	0.099	0.006	0.004	0.005
	Range	0.024	0.04	0.464	0.038	0.024	0.041
Hankel	mean	0.891	0.874	0.69	0.838	0.932	0.947
	std	0.006	0.006	0.186	0.012	0.004	0.004
	Range	0.04	0.039	0.798	0.074	0.03	0.03
Toep	mean	0.887	0.734	0.747	0.833	0.955	0.97
	std	0.008	0.014	0.199	0.012	0.003	0.003
	Range	0.064	0.082	0.875	0.083	0.021	0.017
Harr	mean	0.948	0.829	0.932	0.848	0.969	0.968
	std	0.006	0.015	0.007	0.021	0.003	0.003
	Range	0.056	0.094	0.0490	0.149	0.018	0.027

3.5.4 CNN Training for Fault Classification

A CNN network is trained to perform the failure detection between nominal and failure behaviors using the faceportraits balanced dataset generated by the DCGAN. The CNN is implemented in Matlab using the deep learning toolbox and is composed by three convolutional layers, two pooling stages with a ReLu activation function for the hidden layers and a sigmoid function in the output layer for the failure classification. One hundred epochs train the CNN with a learning rate of 0.001. The last layer has six outputs corresponding to the nominal case and the five failure behaviors inner race, ball, and outer race with center, opposite and orthogonal load. From the 12000 synthetic datasets, 3600 images were employed for the training process, using 300 images for each nominal and failure cases. The validation process employs 8400 images or 700 for each case. After that, a second validation process is performed using the original dataset confirmed by 114 FacePortraits to verify the effectiveness of the CNN network after being trained with the balanced dataset.

3.5.5 Faultface Obtained Results

The results of the FaultFace methodology are summarized using the confusion matrix. It allows identifying the amount of true and false classifications considering if the classifier is confusing classes in the process. It is defined in terms of the true positives (TP), false positives (FP), false negatives (FN), and true negatives

(TN) resulting from the fault detection algorithm. Table 3.3 and Table 3.4 present the confusion matrices obtained after applying the FaultFace methodology for each faceportrait. As can be observed, the CWT, CMR, Gram, Hankel, and Toeplitz FacePortraits gives a 100% matching for the validation data, indicating an excellent failure detection performance of the FaultFace methodology. However, in the case of the Haar FacePortrait, the obtained result shows that only the nominal, ball and load orthogonal behaviors have been detected correctly, while the inner race, load center, and load opposite failures are not well detected.

Table 3.3: Confusion Matrix for the FaultFace Methodology with CWT, CMR, Gram, Hankel, and Toeplitz FacePortraits

	Target class						
Output class		Ball	Inner Race	Load Center	Load Opposite	Load Orthogonal	Nominal
	Ball	28	0	0	0	0	0
	Inner Race	0	28	0	0	0	0
	Load Center	0	0	23	0	0	0
	Load Opposite	0	0	0	15	0	0
	Load Orthogonal	0	0	0	0	16	0
	Nominal	0	0	0	0	0	4

Table 3.4: Confusion Matrix for the FaultFace Methodology with Haar FacePortraits

	Target class						
Output class		Ball	Inner Race	Load Center	Load Opposite	Load Orthogonal	Nominal
	Ball	28	0	0	0	0	0
	Inner Race	0	0	0	0	28	0
	Load Center	0	0	0	0	23	0
	Load Opposite	0	0	0	0	15	0
	Load Orthogonal	0	0	0	0	16	0
	Nominal	0	0	0	0	0	4

3.5.6 Results Analysis of the FaultFace Methodology Using DCGAN Networks

The performance of the faultFace Methodology is quantified using the confusion matrix. Three indices given by (3.3) are calculated, the accuracy A, which establishes the fault rate of the method, the coverage C, which indicates the overall effectiveness of the classifier, and the harmonic mean F, which defines the deviation of the data from the mean.

$$A = \frac{TP}{TP+FP} \quad C = \frac{TP}{TP+FN} \quad F = \frac{2AC}{A+C}. \tag{3.3}$$

The proposed performance indices are summarized in Table.3.5. As can be observed, for the CWT and CMR FacePortraits, the FaultFace methodology gives an accuracy, coverage, and harmonic mean of 1. It means that the synthetic dataset created using the DCGAN has excellent performance for training the CNN for failure classification combined with a good generalization from the CNN. On the other hand, the performance indices show that the accuracy and consistency of the FaultFace method change when the Haar FacePortrait is employed. It can be noticed in the fact that only the nominal and ball failure has been correctly classified, but in the case of Inner race, and outer race with the center, opposite and orthogonal load the algorithm cannot differentiate between the failures.

Table 3.5: Performance Metrics for the FaultFace Methodology for Each Face Portraits

Face Portrait	**Failure**	**Index**		
		Accuracy	**Coverage**	**Harmonic mean**
CWT **CMR** **Gram** **Hankel** **Toep**	Nominal	1	1	1
	Ball	1	1	1
	Inner Race	1	1	1
	Load Center	1	1	1
	Load Opposite	1	1	1
	Load Orthogonal	1	1	1
Haar	Nominal	1	1	1
	Ball	1	1	1
	Inner Race	0	0	0
	Load Center	0	0	0
	Load Opposite	0	0	0
	Load Orthogonal	1	0.238	0.379

For the orthogonal load, the accuracy is one because the algorithm can recognize all the samples related to this behavior; however, the consistency is close to 0.238 because the classification algorithm confuses these with the orthogonal case. Likewise, the harmonic mean of 0.379 indicates a high data dispersion of the fault detector using this face portrait. A possible cause for this behavior is that the Haar wavelet does not represent adequately in the time-frequency domain the different features of that failure behaviors. For this reason, it is possible to say that the choice of the face portrait is not a trivial task and has a significant effect over the fault detection final performance.

3.6 FaultFace Methodology Using GAN Network

The FaultFace methodology is performed using a GAN network in order to compare with obtained results using the DCGAN for dataset balancing tasks. In this case, the GAN network employs multilayer perceptron networks for the discriminator and the generator. The structure of the generator uses three full connected layers of 256, 512, and 1024 neurons respectively and an output layer of 784 neurons to fit with the 28x28 generated faceportrait dimensions. The initial minibatch input size is 100 samples generated using Gaussian distribution. The activation function for the first two layers uses LeakyRelu as activation function, and hyperbolic tangent for the output layer. A batch normalization operator is included at the output of each activation function. The discriminator network is conformed by two fully connected layers with 512 and 256 neurons with leakyRelu activation function, and an output layer with sigmoid activation function to decide between a fake and correct image. A GAN network is trained for the nominal an failure behaviors of the ball-bearing system, for 40000 iterations using adam optimizer with a learning rate of 0.0002 with decay rate of 0.5. The obtained faceportraits obtained using GAN networks are shown in Fig.3.12 and Fig.3.13.

The quality of the new faceportraits generated with the GAN network is measured with the SSIM index presented in section 4.4. Fig.3.11 shows a boxplot of the SSIM index calculated for the nominal CWT and Hankel faceportraits for the nominal and fault behaviors and Table.3.6 summaries all the results obtained for the GAN network. It can be observed that the GAN network SSIM index has a big dispersion on the balanced dataset for all the cases, indicating that the generated data from the GAN diverges considerably from the original data, which will have an effect on the fault detection task.

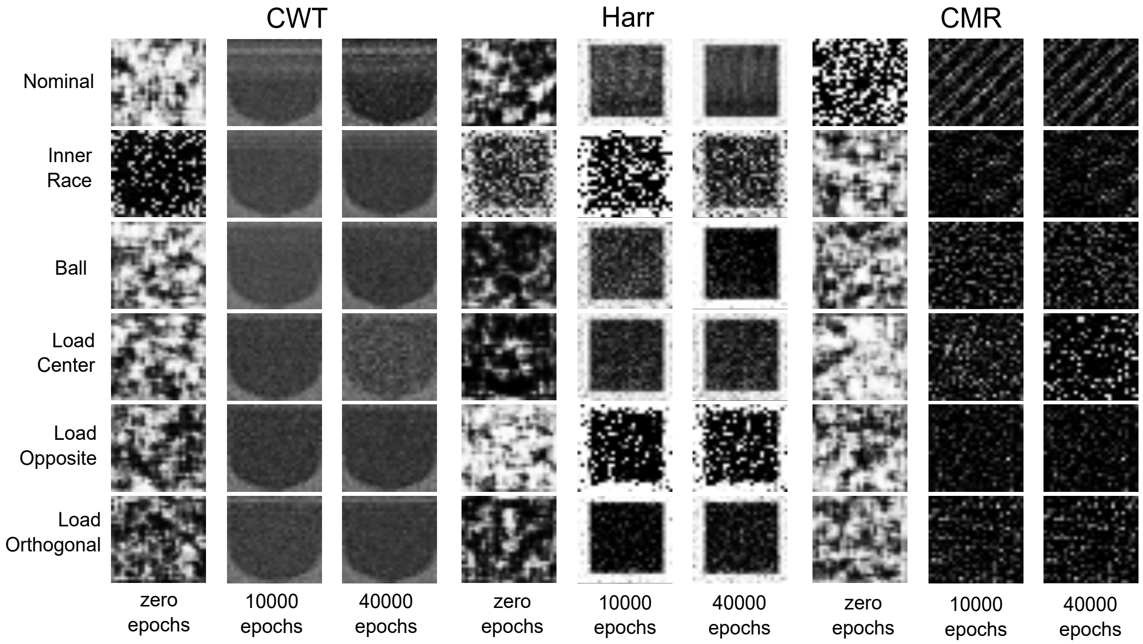


Figure 3.12: GAN Generated CWT, Harr, and CMR Faceportraits at Zero, 10000, and 40000 Epochs

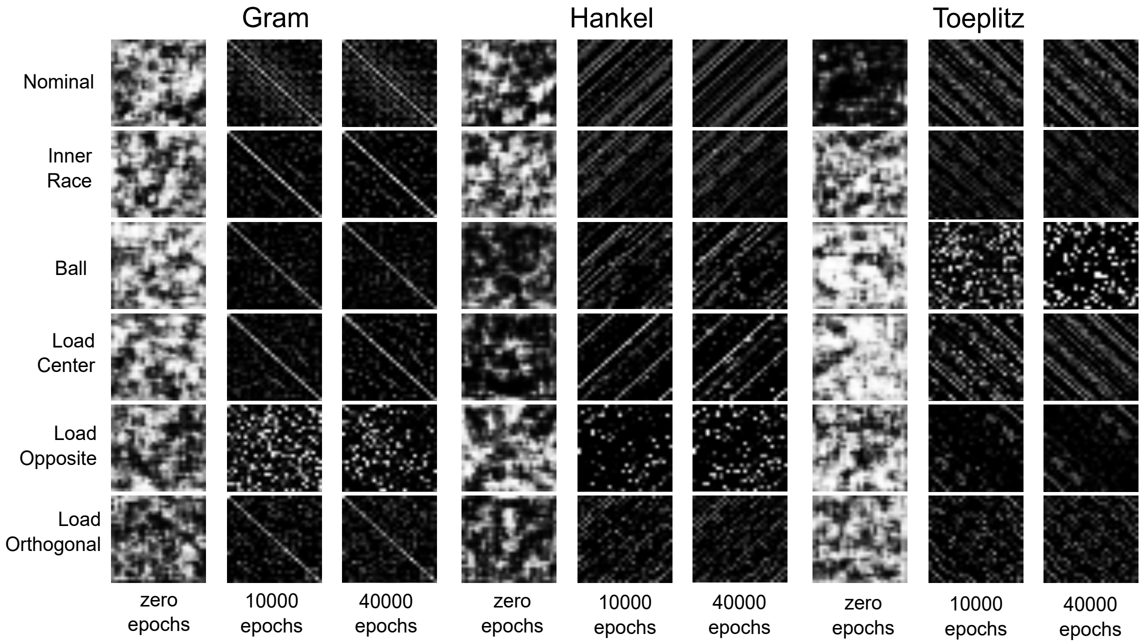
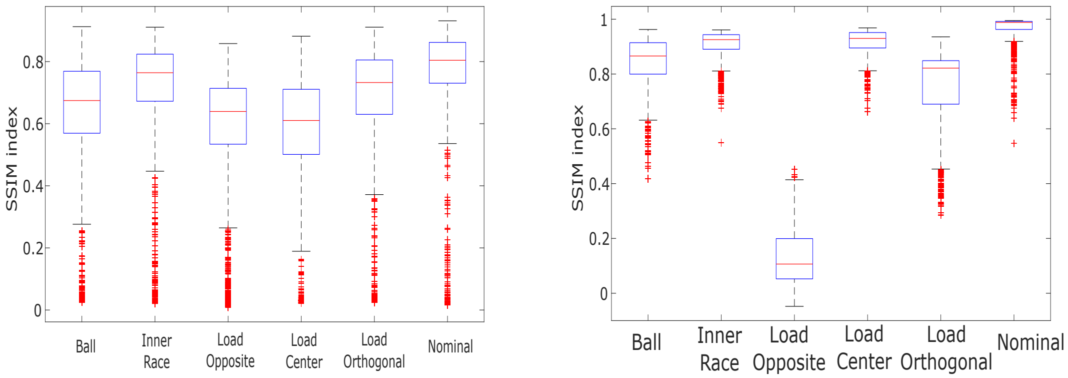


Figure 3.13: GAN generated Gram, Hankel, and Toeplitz face portraits at zero, 10000, and 40000 epochs



(a) (b)

Figure 3.14: SSIM Index for quality Assessment of the Balanced Dataset Produced by the DCGAN Network for the a) CWT and b) Hankel Faceportraits

Table 3.6: SSIM Index Normal Distribution for the GAN Generated Faceportraits

Faceportrait	Statistic	Ball	Inner Race	Load Center	Load Opposite	Load Orthogonal	Nominal
CWT	mean	0.625	0.663	0.561	0.582	0.673	0.715
	std	0.216	0.261	0.236	0.188	0.211	0.257
	Range	0.888	0.89	0.85	0.86	0.887	0.915
CMR	mean	0.85	0.914	0.793	0.512	0.942	0.887
	std	0.108	0.06	0.206	0.221	0.040	0.228
	Range	0.492	0.44	0.924	0.832	0.256	0.968
Gram	mean	0.859	0.77	0.090	0.867	0.855	0.91
	std	0.072	0.119	0.042	0.07	0.086	0.069
	Range	0.461	0.644	0.208	0.389	0.511	0.449
Hankel	mean	0.844	0.907	0.132	0.917	0.752	0.963
	std	0.094	0.052	0.102	0.047	0.145	0.057
	Range	0.545	0.412	0.50	0.306	0.65	0.448
Toep	mean	0.07	0.7	0.671	0.162	0.719	0.887
	std	0.037	0.128	0.225	0.021	0.126	0.079
	Range	0.29	0.642	0.905	0.108	0.6	0.496
Harr	mean	0.759	0.417	0.328	0.745	0.733	0.781
	std	0.055	0.116	0.075	0.037	0.036	0.207
	Range	0.533	0.536	0.432	0.293	0.245	0.883

In addition, a comparison between the SSIM of GAN and DCGAN generated balanced dataset is presented on Fig.3.15 for the CWT faceportrait. It can be observed that DCGAN network produces more accurate new data from the original dataset compared with the GAN network, with higher mean SSIM value and less dispersion of the data distribution.

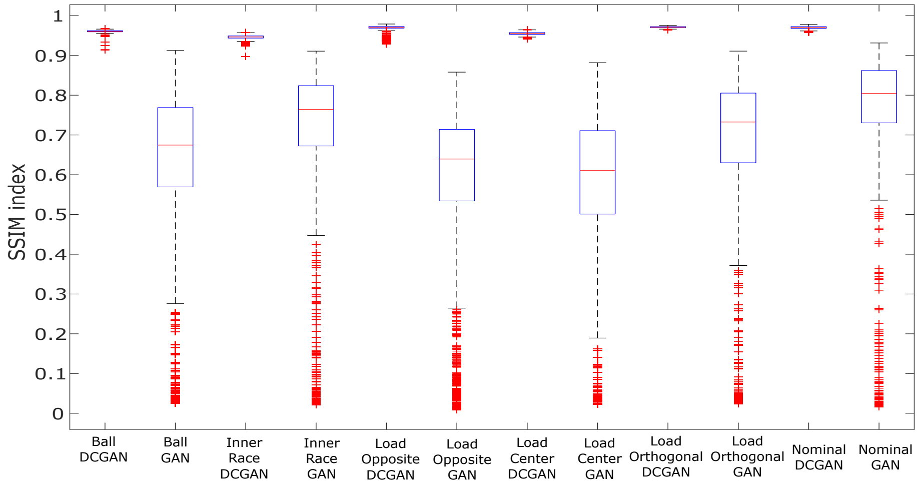


Figure 3.15: SSIM Index for Quality Assessment of the Balanced Datasets Produced by the DCGAN and GAN Networks for the CWT Faceportrait

Table.3.7 shows the accuracy, precision, and harmonic mean from the confusion matrices obtained each faceportrait using the FaultFace methodology with the GAN network balanced dataset. As can be observed, only the CMR faceportrait returns a 100% accuracy on the failure classification task. For the CWT and Toep faceportraits, the CNN makes an incorrect differentiation of the load center failure, confusing it with load opposite and inner race faults respectively. In the case of Gram and Hankel faceportraits, the classifier does not recognize properly the load opposite fault. Finally, the Harr faceportrait has similar classification problems as result with the DCGAN network due to the absence of features offered by this faceportrait for the fault detection task. Thus, it is possible to say that the DCGAN network is a good option for dataset balancing compared with GAN network for fault detection applications.

Table 3.7: Performance Metrics for the FaultFace Methodology for Each Faceportrait Generated Using GAN Network

Face Portrait	**Failure**	**Index**		
		Accuracy	**Coverage**	**Harmonic mean**
CMR	Nominal	1	1	1
	Ball	1	1	1
	Inner Race	1	1	1
	Load Center	1	1	1
	Load Opposite	1	1	1
	Load Orthogonal	1	1	1
CWT	Nominal	1	1	1
	Ball	1	1	1
	Inner Race	1	1	0
	Load Center	0	0	0
	Load Opposite	1	0.395	0.566
	Load Orthogonal	1	1	1
Gram	Nominal	1	1	1
	Ball	0.6585	1	0.7904
	Inner Race	1	1	1
	Load Center	1	1	1
	Load Opposite	0	0	0
	Load Orthogonal	1	1	1
Hankel	Nominal	1	1	1
	Ball	1	1	1
	Inner Race	0.6511	1	0.7878
	Load Center	1	1	1
	Load Opposite	0	0	0
	Load Orthogonal	1	1	1
Harr	Nominal	1	1	1
	Ball	0.4757	1	0.6447
	Inner Race	0	0	0
	Load Center	1	1	1
	Load Opposite	0	0	0
	Load Orthogonal	0	0	0
Toep	Nominal	1	1	1
	Ball	0	0	0
	Inner Race	0.5490	0.5	0.5233
	Load Center	0	0	0
	Load Opposite	1	1	1
	Load Orthogonal	1	1	1

3.7 FaultFace Comparison With Other Methodologies

An LSTM and a SVM with autoencoder networks are designed to perform the fault detection task for the ball-bearing system and compare its performance with the faultFace methodology. Likewise, the FaultFace methodology is also compared with results reported on the literature for the same ball-bearing benchmark system [108].

3.7.1 LSTM Network

A Long Short-Term Memory network (LSTM) [116] is used for the vibration time-series classification. The LSTM architecture is composed by a unidirectional LSTM layer of 100 hidden units, with a input size of 1000 samples, combined a fully connected layer with softmax activation function. The output layer has six outputs for the nominal and the five fault behaviors. The LSTM is trained for 50 epochs, with a minibatch size of 100 samples. The training and validation datasets are composed by 12 and 102 timseries respectively divided in minibatches with variable length between 80 and 100 values. The confusion matrix metrics for the LSTM network are presented in Table3.8. It can be observed that using LSTM for the ball-bearing fault detection problem, an overall accuracy of 69% is reached. Also, the LSTM network exhibit some challenges classifying between the load disturbances cases (center, orthogonal, opposite).

Table 3.8: Performance Metrics for the FaultFace Methodology for Each Faceportrait

Technique	**Failure**	**Index**		
		Accuracy	**Coverage**	**Harmonic mean**
LSTM	Nominal	0.5	1	0.6667
	Ball	1	1	1
	Inner Race	1	1	1
	Load Center	0.65	1	0.7878
	Load Opposite	0.5	1	0.6667
	Load Orthogonal	0	0	0

3.7.2 SVM with Autoencoder

An autoencoder with a support vector machine (SVM) is implemented for the fault detection of the ball-bearing system. The Autoencoder reduces the faceportraits dimensionality using a hidden layer of 100 neurons and an output layer of 10 output features. It is trained for 1000 epochs with L2 weight regularization of 0.004. After that, the SVM is trained using the output of the Autoencoder to perform the fault detection task. Table.3.9 shows the accuracy, coverage, and harmonic mean F metrics calculated for the ball-bearing system using the unbalanced dataset, and the balanced datasets using the GAN and DCGAN networks. The model trained with the unbalanced dataset employs 57 faceportraits for training and 57 for validation, from a total of 114 faceportraits. Thus, an accuracy of about 70% is reached. In the case of the balanced datasets generated with GAN and DCGAN networks, each dataset has 6000 faceportraits, which 3000 were used for training and 300 for validation. For the balanced dataset with the DCGAN network, the accuracy reached is almost 100% for all the cases. In the case of the GAN network, the balanced dataset is about 85%, improving the result obtained with the unbalanced dataset. So, the dataset balancing operation performed by the DCGAN and GAN network is essential to improve the fault detection task accuracy.

Table 3.9: Performance Metrics for the SVM with Autoencoder

FacePortrait	SVM with Autoencoder								
	Unbalanced dataset train			Balanced GAN dataset train			Balanced DCGAN dataset train		
	Accuracy	Coverage	Harmonic mean	Accuracy	Coverage	Harmonic mean	Accuracy	Coverage	Harmonic mean
CMR	0.772	1	0.8713	0.927	0.9411	0.9340	1	1	1
CWT	0.7189	1	0.8364	0.87	0.834	0.85	1	1	1
Gram	0.807	1	0.8931	0.873	0.9360	0.9034	1	1	1
Hankel	0.684	1	0.8123	0.97	0.997	0.7179	0.985	0.9879	0.983
Harr	0.684	1	0.8123	0.794	0.96	0.8691	0.99	1	1
Toep	0.86	1	0.9247	0.856	0.95	0.9	1	1	1

3.7.3 Fault Detection Techniques for Ball-Bearing Vibrations in the Literature

A review about another methodologies for the ball-bearing fault detection on the benchmark system [108] was performed to made a comparison with the FaultFace method [102], [103], [117]- [118]. In Table 3.10, a summary of the different reviewed papers is presented, which employ supervised learning in many cases, some unsupervised and another one use traditional vibration methods like fast Fourier transform. In [119] is presented a supervised machine learning approach using SVM for failure detection with the best fitness of 99%. Besides, [117] and [120] present the use of fractal theory for feature extraction and classification of failure with an accuracy of 98.4% and 96.59% respectively. On the other hand, [118] employs traditional Fourier analysis to detect the different failure behaviors based on the kurtosis of the frequency spectrum of the vibration signal. In the particular case of [102] and [103], both techniques employ unsupervised learning combined with deep learning techniques for failure classification of the ball-bearing system. On [102], the Kmeans algorithm is combined with a GAN network and an autoencoder to create a dimensional reduction of the dataset to detect failures reaching a peak accuracy of 94.69%. In [103], a Deep Neural Network is employed for the fault detection, beginning with a feature extraction from the frequency spectrum of the signals and the use of Principal Component Analysis (PCA) to reduce the data dimension. After that, the network is trained based on the 3D PCA map of each signal. The accuracy achieved is 100% for seven clusters. In addition, the LSTM and the SVM with autoencoder techniques proposed in this chapter are included in the table with accuracy of 90% and 69% respectively.

Table 3.10: Comparison Between Different Failure Detection Methods for Ball-Bearing Elements

Paper	**Type**	**Classification techniques**	**University**	**Best Accuracy**
FaultFace	Supervised	DCGAN with CNN network	U. of California Merced	100%
[20]	Unsupervised	Deep neural network	Tianjin Polytechincal U	100%
[34]	Supervised	Minimum entropy deconvolution with SVM	U. of Pardubice	99.30%
[31]	Supervised	Fractal box counting dimension	Harbin Engineering U.	98.40%
[32]	Supervised	Multifractal and gray relation	Shanghai Dianji U.	96.59%
[19]	Unsupervised	Kmeans, with Generative adversarial autoencoder	Huazhong U. of Technology	94.69%
FaultFace	Supervised	SVM with autoencoder	U.of California Merced	90%
FaultFace	Supervised	LSTM	U. of California Merced	69%
[33]	Traditional	Fast Fourier Transform envelop	U.of New South Wales	Kurtosis

3.7.4 Results Discussion

Comparing the FaultFace methodology proposed with the methods in Table.3.10, an accuracy of 100% can be reached using the proper FacePortrait as well as the DCGAN network for dataset balancing. Notice that most of the methods listed on table 3.10 requires a previous stage of feature extraction using different techniques, in order to create a rich training feature dataset to improve the detection accuracy. In the case of FaultFace method, automatic feature extraction is performed due to the use of trained CNN networks for the fault detection tasks. However, the quality of the balanced dataset is relevant for the success of the methodology. It can be observed when the DCGAN is replaced with a GAN network for dataset balancing, the accuracy of the detection is reduced as shown in Table 3.5 and Table 3.7. A possible cause for this condition is because DCGAN incorporate convolutional layers that can be trained for specific feature extraction and generation. But, in the case of GAN networks, classic multilayer perceptron layers are employed, which requires more training time and number of hidden elements to produce the desired data.

In the case of LSTM network, considering that the sampling frequency of the vibration signals in [108] is too high, more cell may be required to improve the method detection as well as different minibatch size to reduce the need of padding operators that affect the detection quality. For the SVM with autoencoder fault detection algorithm, the balanced dataset generated either with GAN or DCGAN networks improve significantly the overall performance of the detection over the unbalanced dataset.

So that, the combination between automated feature extraction layers, the dataset balancing methods (DCGAN), and deep learning classification algorithms as CNN makes a significative difference performing fault detection for ball-bearing elements regarding to other methodologies. For these reasons, we can conclude that FaultFace is a suitable methodology for failure detection for unbalanced datasets that could be employed not only for ball-bearing joints but also for different industrial processes.

3.8 Chapter Summary

This chapter presented the FaultFace method for failure detection on Ball-Bearing joints based on DCGAN and CNN networks. The proposed method uses a FacePortrait, a 2D representation of a signal that can be obtained using time-frequency representations. For this system, six different FacePortraits were employed using CWT, CMR, Haar, Hankel, Gram, and Toeplitz transformations for six operating conditions composed of the nominal operation and five failure behaviors. A DCGAN network was trained to generate new FacePortraits based on the available data of nominal and failure behaviors to produce a balanced dataset that improves failure detection performance. The balanced dataset of face portraits is employed to train a CNN network that classifies between nominal and failure behaviors. The CNN validation is performed employing the original dataset of the ball-bearing system. The FaultFace methodology is also performed using a GAN instead of the DCGAN network. Besides, an LSTM and SVM with autoencoder networks were trained to be compared with the Faultface methodology. Obtained results show that using the CWT, CWT, Hankel, Gram, and Toep face portraits of the vibration signals, the FaultFace methodology performs an accurate detection of nominal and failure behavior. However, the Haar FacePortrait has a reduced accuracy due to the absence of recognizable features in this representation. Also, when GAN is employed with the FaultFace methodology, the quality of the balanced dataset is different, reducing the FaultFace method accuracy. Likewise, using the balanced dataset produced by GAN and DCGAN networks shows an important improvement for the SVM with autoencoder detection algorithm. Also, a comparison between the FaultFace with other fault detection methods for the ball-bearing system shows that the FaultFace offer excellent accuracy without the need to perform additional feature extraction and dimensional data reduction. Thus, it is possible to say that the FaultFace method can be considered as an alternative for failure detection not only for the ball-bearing problem but also for different industrial processes with unbalanced datasets and complex dynamics. As future works, the real-time implementation of the FaultFace methodology using edge computing devices is proposed as well as the extension of this methodology to other industrial processes than ball-bearing elements. Moreover, the development of compressive deep learning algorithms is proposed to perform deep neural stable control techniques that introduce cognitive capabilities on the edge to smart industrial processes monitoring, prognosis, and control.

Chapter 4

DIGITAL TWIN BACKGROUND

4.1 Introduction

The use of groundbreaking technologies in human history set milestones in the manufacturing processes, known as industrial revolutions. As shown in Fig. 4.1, during the 18th century, the first industrial revolution was powered by the steam powered machine, increasing the speed and quality of manufacturing processes. The second one occurred during the beginning of the 20th century by the introduction of the electric machine, reducing the factories power requirements, improving the manufacturing quality, and defining new concepts like the assembly line and pipeline production. The third industrial revolution carried out during the 1970s driven by computer-assisted manufacturing and robotic systems in the production lines defined the modern industry standards.

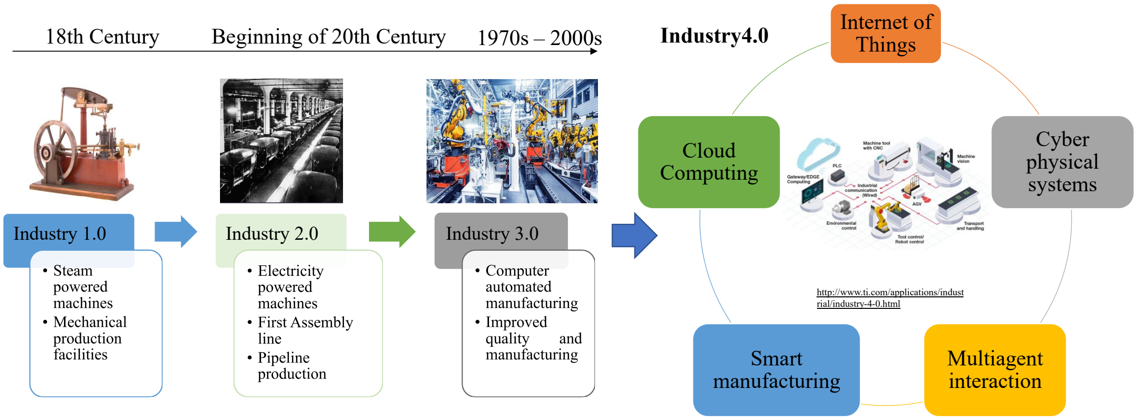


Figure 4.1: Industrial Revolutions from 1.0 to 4.0

Since the 2000s, the fourth industrial revolution or Industry 4.0 is transforming the manufacturing processes into Cyber-Physical Systems (CPS) driven by Information and Communication Technologies. The system features are defined into physical and virtual spaces, with multiple individual agents interacting simultaneously to perform a complicated task. This integration pursues the generation of intelligent responses for each element based on the dynamic of the whole system supported by new groundbreaking technologies like artificial intelligence , internet of things (IoT), cloud computing, big data, deep learning, among others. Thus, new capabilities can be enabled and integrated into manufacturing processes like real-time data analytics, fault detection, prognosis, and life cycle analysis.

The convergence of these capabilities into the industry is called industrial artificial intelligence (IAI), which is an application of AI to the physical operations or systems of an enterprise environments. Industrial AI is focused on helping an enterprise monitor, optimize or control the behavior of these operations and systems to improve their efficiency and performance [23]. The IAI looks for creating value based on unknown knowledge from systematic data-driven information to reach the 3 W's: Work reduction, Waste, and Worry-free manufacturing. It means that for reaching all the new smartness requirements in Industry 4.0 driven by IAI, a novel control systems framework is required, which integrates all these technologies to introduce and increase smartness in real systems, which can be denominated as smart control engineering (SCE). It is a novel control structure that make a system aware in real-time of its current health or variability. SCE is supported by multiple sources of information enabling the processes of cognition, decision, and control leveraging the enabling technologies like edge computing and edge AI producing smart and autonomous responses to reach the desired operation conditions with minimum human intervention. However, as shown in Fig. 4.2, the IAI in combination with disruptive technologies needs to converge into a unified, detailed, and realistic representation of the real system to take advantage of these new capabilities altogether with smart control engineering, corresponding to the Digital Twin of the system. Therefore, the following sections are focused on understanding Digital Twin, its concept, structure, and literature review. Likewise, an introduction to the concept of smart control engineering and the enabling capabilities resulting from Digital Twin are both presented.

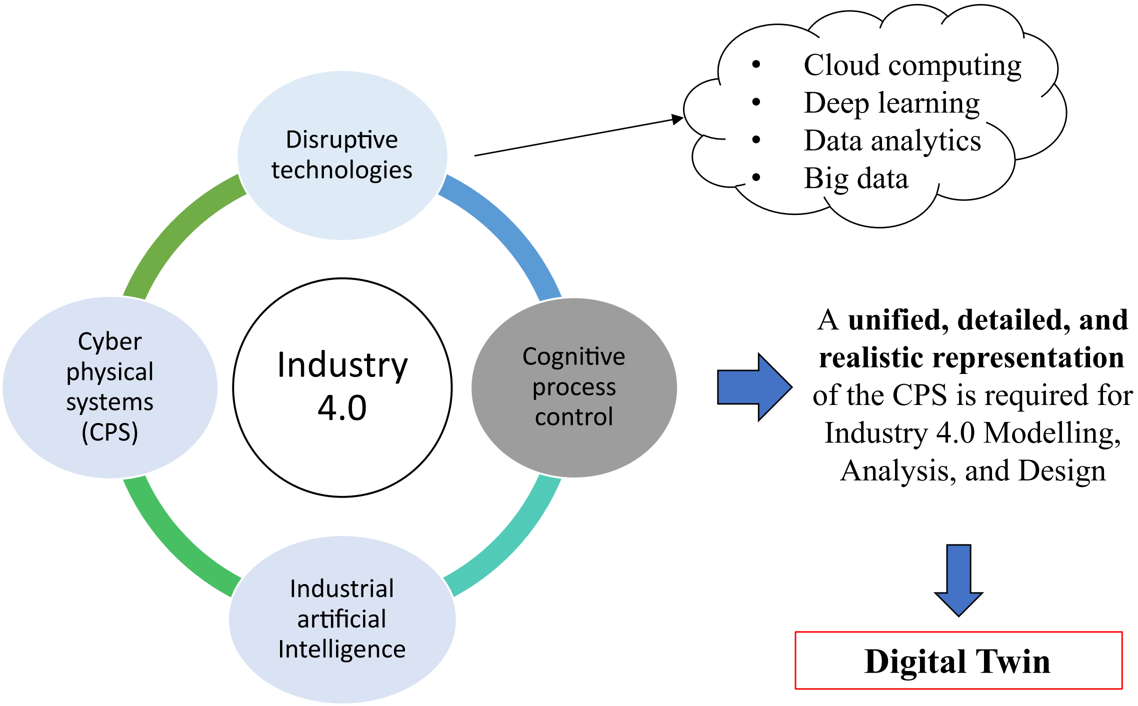

Disruptive technologies
Cloud computing
Deep learning
Data analytics
Big data
Cyber physical systems (CPS)
Industry 4.0
Cognitive process control
A unified, detailed, and realistic representation of the CPS is required for Industry 4.0 Modelling, Analysis, and Design
Industrial artificial Intelligence
Digital Twin

Figure 4.2: Industrial Artificial Intelligence (IAI) Towards Digital Twin (DT)

4.2 What Is A Digital Twin?

A Digital Twin can be defined as a virtual mirroring representation or a copy of a physical system. It is fed with real-time data coming from the system and is able to reflect any change present on the system. The virtual model is composed by multiple physics based models of the system that is complemented with data driven models. According to Garnet trends of emerging technologies [121], Digital Twin is in process to became a productive technology in the next five years, with an expected market of US$35.8 billions for 2025 [122, 123].

The first time that Digital Twin concept appeared was in 2003 given by Grieves [29] in his Product Lifecycle Management course at Michigan University, where the Digital Twin was initially defined as the virtual representation of physical products. At that time, DT concept was constrained by the lack of cheap sensing technology and lack of computing power as well as many of the groundbreaking technologies. However, due to the increasing interest and adoption of these technologies in industry, the concept of DT has a considerable evolution, resulting in several points of view of this technology. Thus, according to [124], the DT concept evolved as shown in Fig. 4.3.

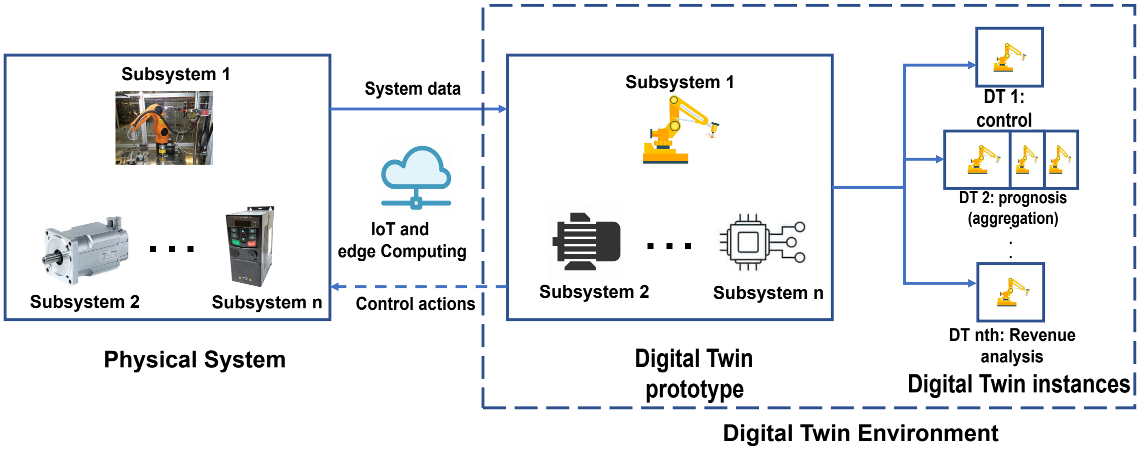


Figure 4.3: Digital Twin Conceptual Diagram

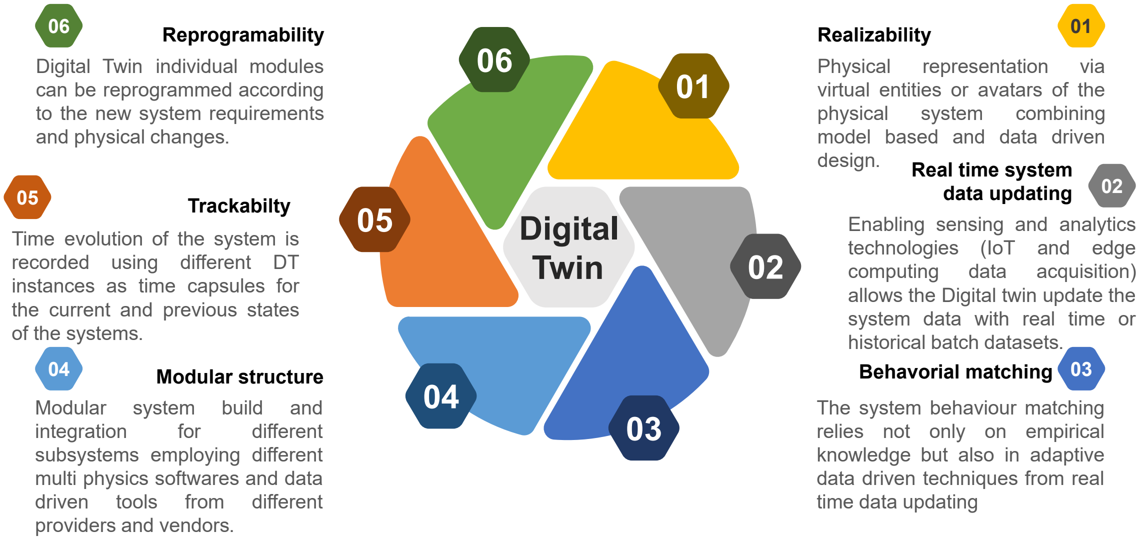
As can be observed a Digital Twin begins with a physical system composed of many different subsystems or elements that performs a specific task. The datastreams from the physical system are sent in real-time to the Digital Twin environment using IoT and edge computing devices. Inside the Digital Twin environment, a Digital Twin prototype describes the physical behavior of each component of the physical system and its interaction using multi-domain physics tools and data driven models incorporated in the Digital Twin environment. Thus, many Digital Twin instances can be created in the DT environment based on the DT prototype. The main purpose of DT instances is creating an individual representation of physical system for and specific task like controller design, system component health and prognosis, revenue analysis among others. Each instance has its own attributes defined at the time when the instance is created, working also as a time capsule to represent previous behaviors of the systems. A comprehensive review of DT definitions is presented in Table 4.1.

Table 4.1: Definitions of Digital Twin

Definitions	Authors	Year
DT is a virtual, digital equivalent to a physical product	Grieves [29]	2003
Up-to-date representation of an actual physical asset in operation	Mathworks [125]	2019
Dynamic virtual representation of a physical object or system, usually across multiple stages of its lifecycle. It uses real-world data, simulation or machine learning models, combined with data analysis, to enable understanding, learning, and reasoning.	IBM [30]	2020
Software representations of assets and processes that are used to understand, predict, and optimize performance to achieve improved business outcomes	GE Digital [126]	2020
DT is a perfect digital copy of the physical world: a Digital Twin. This twin would enable you to collaborate virtually, intake sensor data and simulate conditions quickly, understand what-if scenarios clearly, predict results more accurately, and output instructions to manipulate the physical world.	Deloitte [122]	2020
An integrated Multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates,fleet history, etc., to mirror the life of its corresponding flying twin.	NASA [127]	2020
A digital twin is a multi-faceted dynamic set of smart digital models of a system or a subsystem along with all its constituents, which accurately represent the design of a product, production process or the performance of a product or production system in operation.	Dufour [128]	2018
DT are precise, virtual copies of machines or systems driven by data collected from sensors in real-time, these sophisticated computer models mirror almost every facet of a product, process or service.	Tao et al. [129]	2020

Among these definitions, [125] defines DT as an up-to-date representation of an actual physical asset in operation. It means, the DT can represent in real-time the current state of any process based on the data driven from the system as well as its own constitutive physical laws. On the other hand, DT in the aeronautics context is defined as an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin [127]. Also, [128] states that a Digital Twin is a multi-faceted dynamic set of smart digital models of a system or a subsystem along with all its constituents, which accurately represent the design of a product, production process or the performance of a product or production system in operation.

Based on Table 4.1, some of the most relevant features of a Digital Twin application are presented in Fig. 4.4. It can be appreciated that in a Digital Twin application, a virtual realization of the system is desirable, enabled by sensing technologies and the capacity of reprogramming and modularity for its design and application in multiple scenarios.



06 **Reprogramability**

Digital Twin individual modules can be reprogrammed according to the new system requirements and physical changes.

01 **Realizability**

Physical representation via virtual entities or avatars of the physical system combining model based and data driven design.

05 **Trackabilty**

Time evolution of the system is recorded using different DT instances as time capsules for the current and previous states of the systems.

02 **Real time system data updating**

Enabling sensing and analytics technologies (IoT and edge computing data acquisition) allows the Digital twin update the system data with real time or historical batch datasets.

04 **Modular structure**

Modular system build and integration for different subsystems employing different multi physics softwares and data driven tools from different providers and vendors.

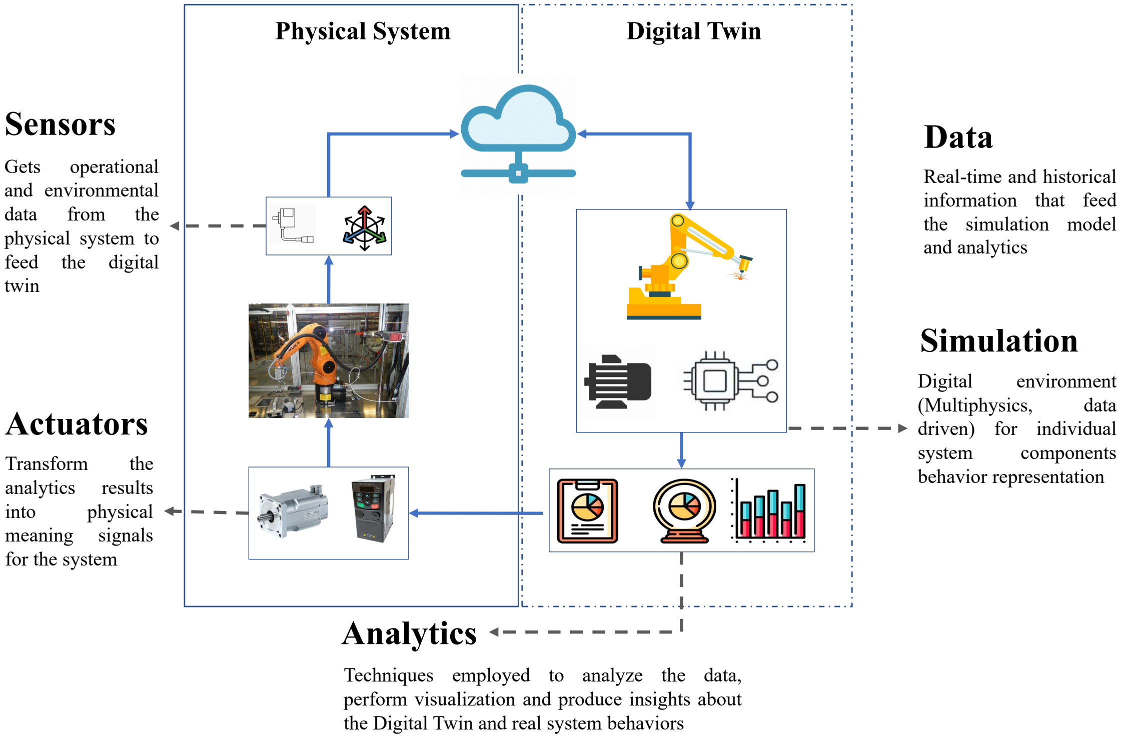
03 **Behavorial matching**

The system behaviour matching relies not only on empirical knowledge but also in adaptive data driven techniques from real time data updating

Figure 4.4: Digital Twin Features

4.3 Digital Twin Requirements and Structure

Digital Twin is composed by the fundamental blocks shown in Fig. 4.5 corresponding to the Sensors, Actuators, Data, Simulation, and Analytics. The sensors are distributed along the physical system capturing the system and its environment data. The actuators apply the control action calculated on the DT into the system. On the other hand, the real-time big data produced by the process feeds the Digital Twin and provide important information about the operating conditions of the system. The simulation corresponds to the mathematical and computational tools that makes representation of the system behavior, considering not only the physical constitutive effects but also the interaction between the individual elements of the system updated by the data acquired from the real system. Likewise, the analytic techniques enable data visualization and insights generations about the system to achieve a smart and optimal control.



Sensors: Gets operational and environmental data from the physical system to feed the digital twin

Actuators: Transform the analytics results into physical meaning signals for the system

Data: Real-time and historical information that feed the simulation model and analytics

Simulation: Digital environment (Multiphysics, data driven) for individual system components behavior representation

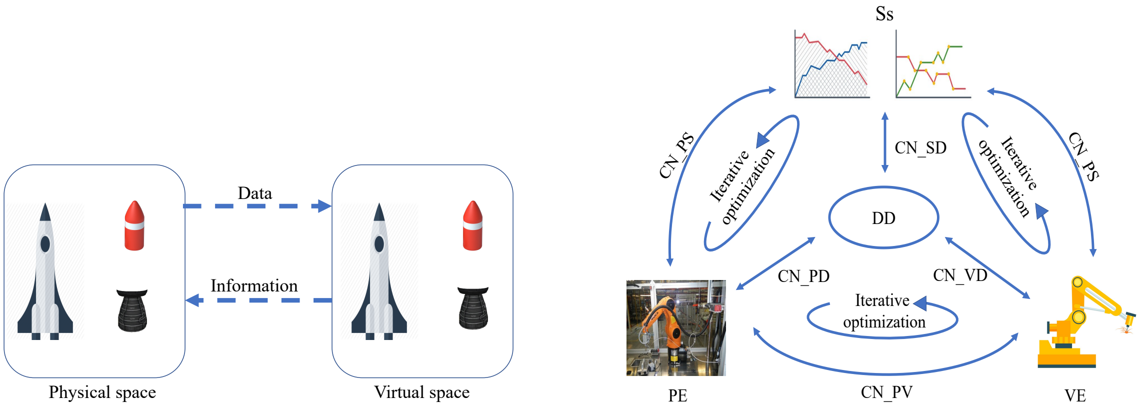
Analytics: Techniques employed to analyze the data, perform visualization and produce insights about the Digital Twin and real system behaviors

Figure 4.5: Digital Twin Components

Based on these fundamental blocks, there are two main Digital Twin architectures employed to build an DT application. One is known as three dimensional model proposed by Grieves [29], composed of a physical, virtual, and connection parts as described in Fig. 4.6a. In this architecture, Digital Twin is created to mirror the behavior of the system based on the information received through some interconnection mechanisms. The main limitation of this architecture is that it only provides a general overview of the Digital Twin without considering further implementation details.

The other architecture shown in Fig. 4.6b is known as five dimensional model [63]. It is composed of the physical entity (PE), virtual entity (VE), service system (SS), data model (DD), and connection model (CM). This architecture is more specific compared with the three dimension model, considering the interaction between physical and virtual dimensions, as well as the services required to perform the information exchange, representing the system behavior and performing data analytics of the system.

The Physical Entity (PE) corresponds to the real system on which the Digital Twin is based. The virtual entity (VE) is the one that replicates the Physical Entity behavior representing geometry, shape, size, and physical dynamics. Besides, Digital Twin data (DD) contains all the heterogeneous and different timescale information of the system that represents its behavior. The service systems (SS) provide to the DT user with applications like simulation, monitoring, fault diagnosis among others, which is supported by the DD and PE information and infrastructure. Finally, the connection model (CM) considers all the information exchanges methods between real and virtual systems and representations.



(a) (b)

Figure 4.6: Digital Twin Architectures a) Three Dimensional and b) Five Dimensional [63].

Likewise, according to [130], Digital Twin can be also classified in four levels according to the development status of the application as shown in Table 4.2. The level I is called pre-Digital Twin, where only the virtual environment is defined for a Digital Twin based only on the conceptual idea of the system, which means, there is no physical prototype of the system available. This stage is useful for initial design and develop of the real application.

In level II, a physical system and the DT environment are available. However, there is no real-time interaction between these two parts. At this level, the DT can be calibrated according to the real system behavior in order to use the DT environment as an accurate simulation tool for controller design for the system. However, this stage may not consider elements as prognosis, fault detection, and life cycle analysis of the system.

The level III involves the real-time interaction and monitoring of the physical system directly on the Digital Twin environment, as well as the real-time updating of the system status. Therefore, at this point a real-time mirroring representation of the system can be performed, which enables the capability to perform real-time fault detection on the system based on the DT environment.

Finally, level IV involves machine learning, artificial intelligence, and data analytics on the DT environment to produce automated responses that acts over the physical system in fault scenarios or generating recommendations alarms for components replacement or machinery service making the system smart.

Table 4.2: Levels of Digital Twin [130]

Level	System modeling	Physical system	System interaction	Smart capabilities	Features
I	Digital Twin Environment	not built	Not apply	Not applicable	Preliminary design
II	DT environment based on the real system	Operating standalone	Non real-time data acquisition	No	Performance analysis and system status
III	DT environment with monitoring interface and fault detection capabilities	Operating standalone	Real-time data acquisition	Limited	Data analytics, fault detection, prognosis
IV	DT environment with monitoring interface, and adaptive behavioral learning	Operating in the loop with DT environment	Real-time data acquisition and control	Total	Data analytics, Fault detection, prognosis, Automated recommendations, Automated actions over the physical system

4.4 Challenges on the Digital Twin Implementation

Considering the amount of elements involved in a DT application, there are some challenges that should be overcome in order to reach a successful implementation of DT:

- **Modeling:** DT models involve multiple individual mathematical and virtual representations of system components consisting of 3D simulations, finite element models, rigid body multiphysics among others. These models interact together in the DT virtual space and are built by different third party providers, usually without unified data standards, making it difficult to verify the accuracy of the model [129].

- **Sensor integration and data fusion:** one of the strengths of DT is the ability of incorporating real-time data coming from the physical system to reflect its behavior to get insights about the system components, performing that real-time interaction requires high computing power to run complex models of the system as well as reliable sensors, communication protocols and networks to feed the DT [131].

- **Big data driven behavioral matching:** from the acquired Big Data of the system, most of the current Digital Twin applications are tuned not only using physics and big data based knowledge about the system but also relaying on practical experience coming from the engineers and operators of the systems, lacking of a systematic methodology for optimal design and implementation of DT applications.

4.5 What Is Not A Digital Twin?

The term "Digital Twin" is employed in different ways with substantial conceptual variations among researchers and engineers. For example, in engineering, Digital Twin is considered as a tool with benefits on system design, optimization, process control, virtual testing, predictive maintenance, or lifetime estimation. However, the difference between a model and a Digital Twin is not made clear among other contexts. The risk associated with the multiple and vague definitions of Digital Twin may lead industry and academia users to consider it just another technique. Thus, once the DT hype is over, the final level of interest and use (the "plateau of productivity") may fall well below the the maximum potential of the technology [132, 133].

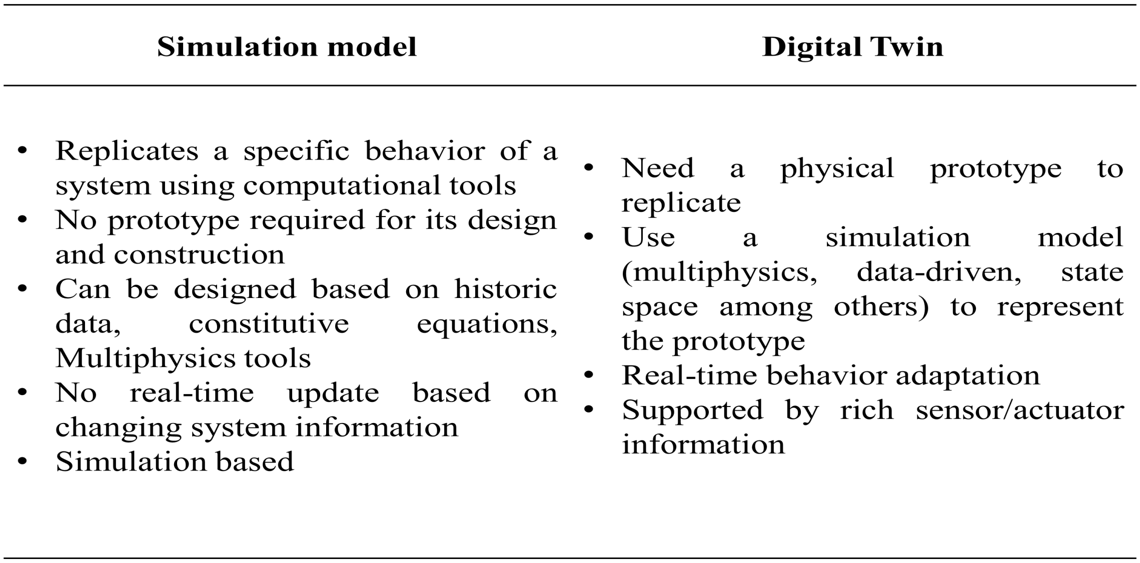
From all the definitions presented in Table 4.1, the most important components of a Digital Twin are:

- A model of the physical system
- A real-time varying dataset of the system
- A real-time update or adjustment of the system model based on the varying dataset.

Thus, one way to differentiate between a Digital Twin and simulation model is using the fact provided by [132] "A Digital Twin without a Physical Twin is just a model". It means that a Digital Twin has to be associated with an object that physically exists. So, the virtual representation can enhance new analysis further than real-life experimentation and assessment, based on an updatable model supported by a real-time changing dataset.

Therefore, it is essential to differentiate between a simulation model and a Digital Twin. A traditional model-based design (MBD) involves the verification and validation of models, which can be used to optimize the design or operation of a device or process. The model is usually validated by comparing experimental results with the model results and performing parameter estimation, but this operation is not performed in real-time since MBD can exist even without the presence of a physical counterpart; therefore, simulation is not a Digital Twin but it is an essential part of the Digital Twin.

Likewise, simulations are typically used for design, conceptualization, or offline optimization. However, Digital Twins are used for the entire designing, execution, change, decommission lifecycle, which can be performed in real-time, which traditional simulations cannot provide insights into the interactions of the physical system. Also, Digital Twins are a source for understanding the conflicts when physical components interact. Simulation only can help to understand "what may happen" in the real world. On the other hand, Digital Twin helps to understand what may happen but crucially "what is happening" and how the design is behaving in the real world. So, Digital Twin becomes an essential part of a smart system, which dynamics may be too complex to be described by a differential equation or equivalent representations. Figure 4.7 summarizes the differences between a Digital Twin and a simulation model.



Simulation model	**Digital Twin**
• Replicates a specific behavior of a system using computational tools • No prototype required for its design and construction • Can be designed based on historic data, constitutive equations, Multiphysics tools • No real-time update based on changing system information • Simulation based	• Need a physical prototype to replicate • Use a simulation model (multiphysics, data-driven, state space among others) to represent the prototype • Real-time behavior adaptation • Supported by rich sensor/actuator information

Figure 4.7: Digital Twin vs Traditional Simulation.

4.6 Digital Twin Applications

Thanks to the properties of the Digital Twin, it can be used on different applications, including advanced modelling, manufacturing, healthcare, supply chain, or retail sales.

In manufacturing applications, several industrial types of equipment produce big data that can be used to create Digital Twins that enable real-time data analytics and advanced analysis capabilities. For example, for product development and design customization DT allows the engineers to test the expected performance of upcoming products before release. Also, with Digital Twins , businesses can perform design iterations of a product to offer personalized products and services to their customers [134–136]. Likewise, the manufacturing shop floor performance can be improved using DT, enabling a real-time simulation model of the facility. So, possible bottlenecks can be predicted, leading to delayed production of the final product [137, 138]. Also, using DT contributes to introducing and improving predictive maintenance practices that predict critical faults that produce potential downtimes of the manufacturing processes. Thus, companies can improve the overall efficiency of processes thanks to an enhanced knowledge of the process that allows establishing an effective maintenance schedule of the process units [139, 140].

Besides, in the aerospace industry, physical twins were used to analyze the performance of the ships, including systems robustness against critical design faults [141–143]. On the other hand, the development of conventional and self-driving cars can be accelerated by employing DT by creating the virtual models of the vehicles to simulate and analyze the production phase. Thus, predictive analytics can be calculated to foresee any future problem involving the airframes, engine, or other components to ensure the safety of the people on board [144–146].

For healthcare applications, Digital Twins can help providers to provide a virtualized healthcare experience to optimize patient care, cost, and performance. For healthcare, use cases can be categorized into two groups. The first one is improving the operational efficiency of healthcare operations by creating a Digital Twin of a hospital; operational strategies, capacities, staffing, and care models help healthcare providers examine the operational performance of the organization. The second one is improving personalized care which healthcare providers and pharmaceutical companies can use digital Twin to model the genome code, physiological characteristics, and lifestyle of patients to provide personalized care guidelines [147–149].

In supply chain, Digital Twin has several applications. These include predicting the performance of packaging materials by virtualized DT testing running simultaneously with the physical packaging process looking for errors before a product is packed. Also, using DT is possible to enhance shipment protection logistics analyzing how different packaging conditions can affect product delivery or creating a logistics network DT with the road information about the traffic situation, road layout, and construction to design the distribution routes and inventory storage locations [150–153].

Besides, in construction processes, a Digital Twin can help construction firms understand how a building is performing in real-time, tweaking the performance to optimize efficiency. Data collected from Digital Twin can be used for planning and designing future buildings [154–157].

Finally, retailers can create Digital Twins of customer personas to improve the customer experience they deliver. For example, retailers can provide ideal fashion clothing products to customers based on their Digital Twin models.

4.7 A Literature Review of Digital Twin

Digital Twin is a novel and active research topic among both academic scholars and industry practitioners. So, a comprehensive literature review about Digital Twin applications has been performed to provide a big picture about the current status of the topic. As shown in Fig. 4.8, DT publications on ScienceDirect and IEEEXplore databases shows that DT has an increasing interest since 2018. Among these publications, there are some books, review, application and case study papers.

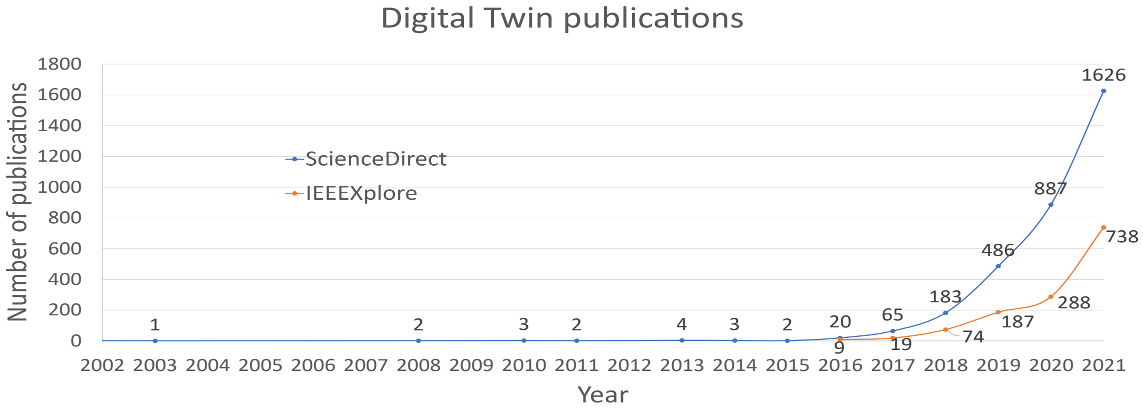


Figure 4.8: DT Publications Records in IEEEXplore.org and ScienceDirect.com by December 31 2021.

Thus, about 90 Digital Twin papers have been selected between the mentioned databases. These papers are classified into conceptual and review papers, and according to the four levels of DT presented in Table 4.2.

In the conceptual and review papers, [158,159] presents a survey of definitions, characteristics, applications, and design implications for Digital Twin assets in the manufacturing, healthcare, aviation among other industries. Also, [160] presents the combination of big data with Digital Twin, highlighting the last one as the bridge between information and physical assets modeling. Likewise, a list of opportunities and future works for Digital Twin are given in [160–168], including, multi time-scale DT, Bayesian based DT models, life cycle management, repeatability and representation of physical assets, supply chain, agent semantic DT, among others.

On the DT level I, there are multiple applications of DT for initial system modeling, design, and assessment. For example [169] proposes the design of a complete DT for a machine shop floor using the five dimensional architecture for DT applications, which considers not only the machinery but also the human-machine interaction as well as the multiple datastreams produced by the system, combined with a set of data, model, algorithm, function, and visualization services to manage the system interaction. In [170], a DT model is created to perform structural lifecycle analysis of an aircraft through stress, temperature, and vibration prediction multi physics models integrated in the DT of the system based on multiple mission trajectory data. Also [171] presents the design of a DT level I for a Computer Numeric Control (CNC) machine using Unified Modeling Language (UML). In [172], a three layer architecture for Cyber-Physical production systems is presented which involves several DT models for each real asset on the production system. Likewise, there are other level I DT applications for other systems like supply chains, power systems, automotive manufacturing, or internal combustion engines [51,79,130,140,173–189].

For DT level II applications (DT based on an existent physical asset), [190] presents a DT framework and its application regarding fault detection and prognosis of car assembly line using Deep Transfer Learning. In robotic systems there are some related works. For example [191] presents the design and implementation of the DT for an industrial robotic manipulator using Virtual Reality to recreate the manipulator interaction with the environment as well as the multiphysics of the system. [192] shows a robotic system DT employed for training a real manipulator for obstacle avoiding using Gazebo and Python as multiphysics simulation tools. More DT applications on robotics are presented in [193–195]. In aeronautics, [196] presents the design and implementation of a DT for an existing helicopter rotor dynamics to analyze the behavior of its mechanical components. There are also other DT Level II applications for manufacturing processes [197–203], power systems [204–206], smart farming [207,208], geologic analysis [209], process control [210,211], and smart buildings [212].

In the DT Level III (DT with real-time interaction with the physical asset), the number of applications is less than in the previous categories considering the novelty of DT . Among the applications, the DT development of power systems like DC-DC converters, electric drive trains, electric vehicles, photovoltaic systems, and batteries [54–59], robotic manipulators [60, 61], plasma detection on industrial environments [62] and health monitoring [63, 64]. DT level III applications should include any of the following enabling capabilities like fault detection, prognosis, or component degradation alert.

Finally, for the DT level IV (DT with real-time interaction, enabling capabilities and adaptive behavioral learning). In this case, few works satisfy the DT level IV conditions defined in Table 4.2. For example, [65] presents a DT application that performs real-time inference of remaining useful life of a drilling machine to correct the imbalance in the drilling axis. Besides, [66], performs real-time inference of the current state of a set of robotic manipulators configured via cloud service to perform real-time adjustment of the robotic systems tasks. Likewise, [67] combines big data with Digital Twin for the real-time monitoring and real-time decision for wind farms. Also, [68–70] present the DT for CNC machines with self awareness interaction between physical and virtual parts of DT considering machinery tool condition assessment.

4.8 Chapter Summary

This chapter presented an introduction about Digital Twin, its concept, architectures, and applications. Also, a clear definition of what is and not a Digital Twin is provided in order to clarify the real purpose of Digital Twin operating as a real-time updated representation of a physical system supported by breaking technologies as IoT and AI that enhance the system knowledge and enable smartness of the system. Also, applications and literature reviews confirm that DT has a significant relevance on industry and academia based on the increasing diversity and number of publications regarding to DT applications in sectors from construction, healthcare, to logistics. In the case of control engineering, there is also a considerable effort towards the integration of Digital Twin to control systems, however several cases are related to conceptual requirements with few real-life applications. For this reason, in this monograph, a methodological framework focused on DT for control applications will be introduced including a set of real-life cases study that show the Digital Twin applicability in control engineering.

Chapter 5

A DIGITAL TWIN DEVELOPMENT FRAMEWORK

5.1 Development Framework for Digital Twins Applications

Considering that the goal of Digital Twin is to replicate the real behavior of a physical system with the highest possible level of detail, a methodological framework is required in order to obtain a successful implementation of a DT as shown in Fig. 5.1. The framework is composed by five steps corresponding to the target system definition, system documentation, multidomain simulation, DT assembly and behavioral matching, and DT evaluation and deployment.

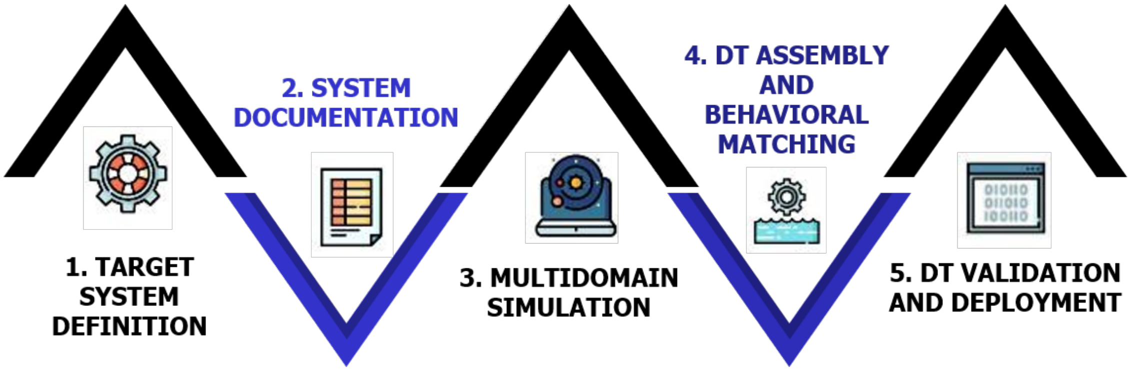

1. TARGET SYSTEM DEFINITION
2. SYSTEM DOCUMENTATION
3. MULTIDOMAIN SIMULATION
4. DT ASSEMBLY AND BEHAVIORAL MATCHING
5. DT VALIDATION AND DEPLOYMENT

Figure 5.1: A Development Framework of Digital Twin

Step 1: Target System Definition

This step is focused on recognizing the current status of the physical system to be replicated via Digital Twin with two possible scenarios. The first one is when the physical system is in a conceptual design stage. In this scenario, building a Digital Twin of the system is a preliminary step for physical sizing as well as emulating its real behavior as closer to reality as possible. In this case, using CAD/CAM tools and design is the best choice to create a real system representation. Notice that after the real system is assembled and operative, the Digital Twin is updatable with the real system information to be aware of the real behavior of the system.

On the other hand, the second scenario is when the physical system is already operative performing the desired tasks. In this case, even if a previous CAD/CAM model is not available, the Digital Twin can also be built based on the current system configuration using the system big data, information, and experimental knowledge provided by designers, engineers and operators. In both scenarios, the proposed framework can be employed to create the Digital Twin of a given system. The second scenario is applied as a case study in Sec. 5.3.

Step 2: System Documentation

Once the target and operating scenario of the system to be replicated by the Digital Twin is defined, the following step is to collecting all the available information of the system to create the most accurate representation. This relevant information includes the control algorithms employed (PID, MPC, State-Space) and its digital implementation, sensors and actuators data sheets, troubleshooting and problems records, cumulative experience of the system engineers and operators, and the system data streams. Notice that most of this data can be acquired via sensors, virtual metrology, indirect measurements or state estimators. In addition, the information about the environment where the system performs its tasks is crucial for the correct operation of the Digital Twin like wind speed, direction, environmental temperature, humidity among other variables relevant for each application. Once all the available information about the system is collected, the application of data analytics like principal component analysis (PCA), signal denoising and detrending, average removals, descriptive and inferential statistics, and machine learning techniques are required to perform cleaning, filtering, and organization of the collected information, specially in big data scenarios with multiple agents interaction with unknown behaviors, where some data may be corrupted, missing, or is irrelevant to the system dynamics.

Step 3: Multidomain Simulation

After gathering all the data related to the system, and performing the data processing operations described above, the next step is defining and configure the simulation models that will be employed to represent the system real behavior. In this case, the first task is define the simulation domains related to the system. It means, define the physical and constitutive laws that govern the system and select the computational tools to represent it. Usually, these domains include thermal, mechanical, electrical, fluids and digital components that can be simulated using multiphysics based simulators like COMSOL, ANSYS, MSC-ADAMS, Matlab Simscape, among other multiphysics software packages. In some cases, the system model also incorporates discrete and algorithmic elements like task scheduling or event based situations that can be managed using coding. Sometimes, when there is not enough information about some physical model of the system or that behavior cannot be characterized adequately, data driven models of the system should be employed as a blackbox to represent that unknown behavior.

Once the simulation domains are defined and each subsystem interaction is built using the corresponding computational simulation tools, the next step consists of integrating each single simulation model to reproduce the system behavior. In some cases, all the subsystem models can be integrated using a single software package, but it often results into a co-simulation model, combining the capabilities of each individual software. It is important to highlight that if the Digital Twin application is designed to run in parallel with the real system, the computational cost can be high according to the multi-physics simulation packages employed. Therefore, a trade off between model level of details and computational performance should be considered.

The initial runs of the multidomain simulation may run based on some ideal conditions of the represented system in order to verify the convergence and flow of the simulation environment. However, further behavioral matching is required to make the Digital Twin mimic the real system behavior.

Step 4: DT Assembly and Behavioral Matching

In this step, a stable and operative multidomain simulation model of the system is available for the Digital Twin realization. However, this DT model is operating under ideal conditions as stated before. So, a process called behavioral matching (BM) should be performed. It can be defined as a procedure to find the parameters of each subsystem composing the Digital Twin in order to fit its complete system dynamics, coinciding with the real state of the physical system.

A description of the behavioral matching is presented in Fig. 5.2. As can be observed, the complete Digital Twin model is fed with the input and output data from the real system, including the reference signals for control loops. Based on these data, the Digital Twin is set into an optimization loop to determine via optimal searching the unknown parameters that cannot be determined *a priori* in each subsystem that compose the Digital Twin. This optimization loop keeps working until the coincidence between the input and output data streams of the system is reached with a certain tolerance limit or after a fixed number of iterations.

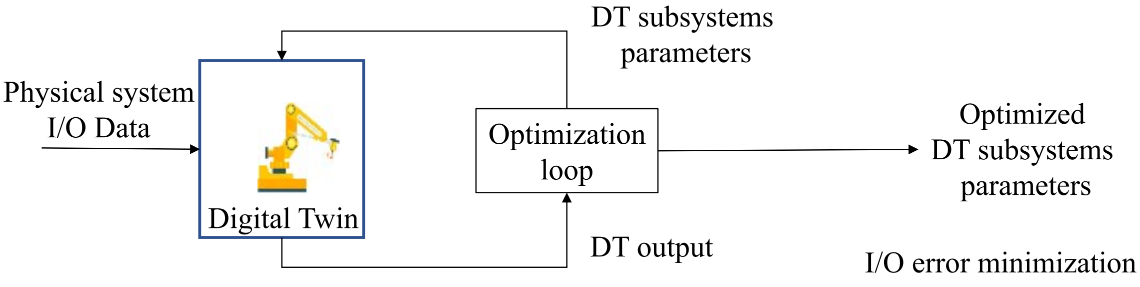


Figure 5.2: DT Behavioral Matching

There are some considerations to perform a correct behavioral matching. Initially, the required level of detail of the Digital Twin has to be evaluated about the complexity of the optimization process considering that Multidomain Simulation models that are complex and may take significant time to run a single simulation event. Also, BM nature requires that not only the system output but also the input matches the system real operation. In addition, the BM is performed considering the complete Multidomain Simulation model, so, choosing the most suitable optimization algorithm as well as setting a cost function may be challenging. In that sense, metaheuristic methods like genetic algorithms and Cuckoo search can be used as an alternative when we employ classic performance indices like ISE, ITAE or IAE as cost function with weighted input and output signals as part of the function.

Step 5: DT Validation and Deployment

This is the final step on the Digital Twin implementation, performing the validation and simultaneous deployment with the real system. Initially, starting from the behavioral matching of the system, the Digital Twin response is calculated for different input/output data sets collected for the system. With this purpose, a supervisory interface has to be designed in order to perform the Digital Twin offline and simultaneous execution of the system. Likewise, through the interface include the behavioral matching capabilities, combined with the analytics and fault detection modules, corresponding to the new enabling capabilities for the Digital Twin application.

5.2 Digital Twin Frameworks in the Literature

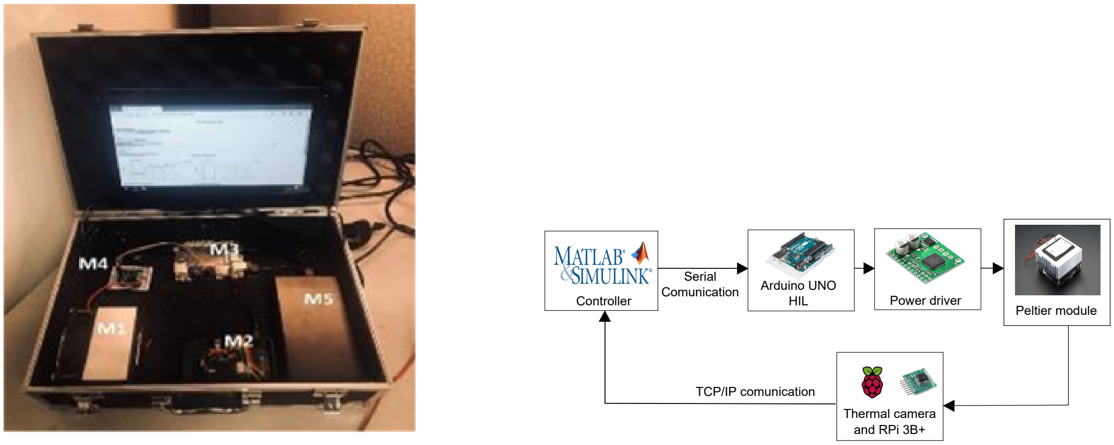
Digital Twin as an emergent technology requires to define a set of systematic steps that leads to the implementation of a functional DT application. In the literature, several Digital Twin frameworks has been proposed. One of the first frameworks for DT development is proposed in [159], which established a six steps method to build and develop a DT focused on manufacturing processes, which begins by building the DT of the system, its validation and simulation, to finish in a direct command of manufacturing goals over the physical system, which is proposed to be employed in a smart transportation system. Besides, [213] presented a DT development framework for hydraulic canal using steps like canal modelling, system identification and calibration with the capability of scenario based simulation to predict drought periods. In [214], a DT frameworks supported by IoT and Data-Driven techniques is proposed for monitoring a construction site and forecast the completion time and detect bottlenecks on the construction process. A distributed DT framework is proposed in [215] focused on fault detection using local Digital Twins communicated with a general DT. In this case the framework is focused on the workload distribution instead on the developing of the DT from the beginning. An interesting framework is proposed by [216], which presents a complete DT architecture and its implementation for the monitoring and representation of a petrochemical process. This framework in particular analyzes in real-time the performance of multiple variables (more than 30) to determine a data-driven model using machine learning algorithms to represent the system behavior and optimize the parameters at each step of the process, increasing the system productivity. Likewise, there are some frameworks proposals, which show only a conceptual architecture for a future Digital Twin implementation, lacking of practical applications in control engineering [217–219].

Unlike the related DT frameworks, the one proposed in this book is designed and focused on control systems design toward the design of smart systems. It means that the proposed framework starts from building the DT from scratch, builds a consistent virtual system representation and its validation. Then, this new framework can be used to introduce smartness into a physical system towards edge and embedded computing devices and supported by breaking technologies. Likewise, the previous methods are focused only on one specific task like fault detection or quality monitoring. This proposed DT framework can be extended further from control systems and integrate all the individual capabilities of the other frameworks. In addition, the proposed DT framework is not only based on machine learning models but also strongly supported by multiphysics simulations tools that brings an additional physics based principle for the system virtual representation.

5.3 A Step-by-Step DT Construction Showcase: Temperature Control with A TIR Camera

Step 1: Target System Definition

A temperature uniformity control system using a real-time thermal infrared (TIR) vision feedback presented in Fig. 5.3 is employed as a case study for developing its Digital Twin based on the proposed methodology summarized in Section 5.1.



(a) (b)

Figure 5.3: DT Showcase System: Real-Time Vision Feedback Infrared Temperature Uniformity Control

As shown in Fig. 5.3, the system is composed of a Peltier cell (M1) that works as a heating or cooling element, a thermal infrared camera (M2) acting as a temperature feedback sensor running on a Raspberry Pi and communicated using TCP/IP communication protocol, that allows performing temperature distribution measurement and control. An additional component of the system is LattePanda board (M3), which runs Windows 10 64-bits and executes Matlab for local operation as well as the remote laboratory application for the NCS (networked control system) control system. To manage the power applied to the Peltier cell an Arduino Leonardo board is used (M4) which via PWM controls the power driver. The platform is equipped with a battery (M5) that provides the power for all the components in the box with four hours of autonomy. This case study can be fit into the second scenario corresponding to the system with a stable, controlled physical prototype. In this case, the uniform temperature control system is open and closed-loop stable, employing a PID controller with anti-windup.

Step 2: System Documentation

From Step 1, the system is composed by four critical elements, the infrared thermal camera, the Peltier thermoelectric module, the control unit (Lattepanda board) and the power driver, whose operating specs are defined in [220–222]. Table 5.1 and Table 5.2 present a brief summary of the critical properties for the power driver, the Peltier module and the infrared thermal camera, which are required for the steps of Multidomain Simulation and behavioral matching. In the case of the Peltier module, its properties differs among different manufacturers. Therefore, a behavioral matching is required to determine the correct system parameters. More details about the system implementation and real test performed on the system can be found in [223, 224].

Table 5.1: Brief Thermal System Documentation

Component	Features
FLIR Lepton Thread Infrared Thermal Camera	Wavelength: 8 to 14 μm Resolution: 80x60 pixels Accuracy: ± 0.5°C
TEC1-12706 Peltier Module	$Q_{max} = 50W$ $\Delta_{Tmax} = 75°C$ $I_{Max} = 6.4A$ $V_{max} = 16.4V$
MC33926 DC Power Driver	Input: 0-5 V Output: 0-12V Peak Current: 5A
Lattepanda board	5 inch Windows 10 64 bits PC Intel Atom μp 4GB of RAM Built-in Arduino Leonardo board

Step 3: Multidomain Simulation

The case study is divided in four simulation domains presented in Fig. 5.4. The first domain is the electrical, composed by the power driver, the battery and the semiconductor joint on the Peltier module. The second one corresponds to the thermal domain defined by the heat transfer produced between the Peltier hot and cold sides, the system surface and the surroundings, as well as the thermal properties of the heat sink. The third domain corresponds to the fluids part, given by the air flow pumped into the heat sink to keep its temperature constant. Finally, the fourth domain corresponds to the digital domain, composed by the PID control algorithm and the analog to digital interfaces to communicate the control side with the thermal system. Also, this simulation domain includes the behavior of the infrared thermal camera.

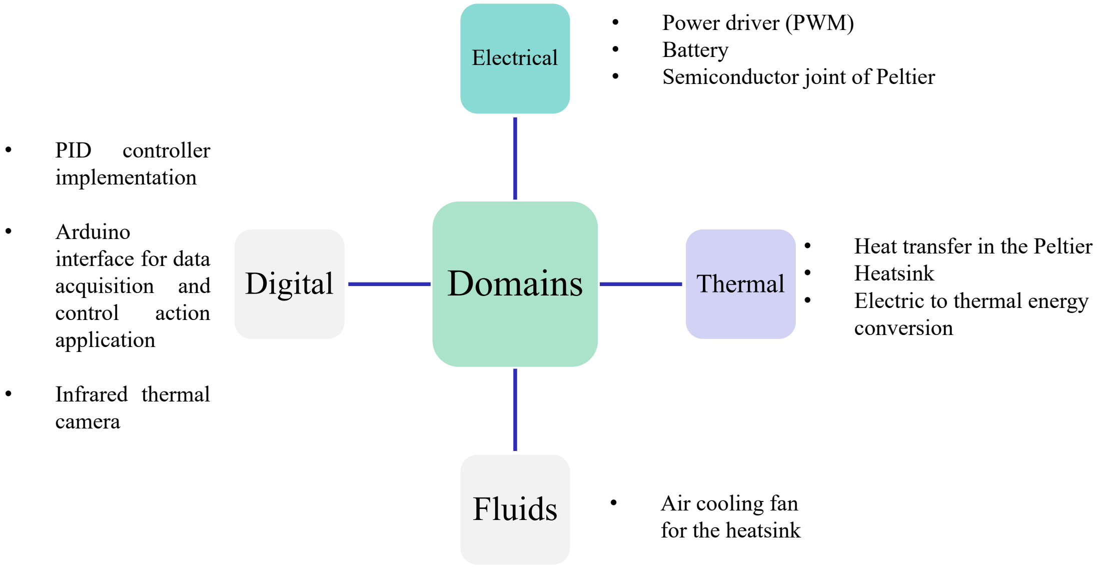

Electrical
Power driver (PWM)
Battery
Semiconductor joint of Peltier
PID controller implementation
Arduino interface for data acquisition and control action application
Infrared thermal camera
Digital
Domains
Thermal
Heat transfer in the Peltier
Heatsink
Electric to thermal energy conversion
Fluids
Air cooling fan for the heatsink

Figure 5.4: DT Case Study: Simulation Domains

In this case, the electric, thermal, and digital domains will be replicated in the Digital Twin application. Matlab-Simulink and Simscape electrical and thermal are employed as multidomain simulation packages to replicate the physical laws of the system as well as the PID control law employed. The complete multiphysics simulation model is presented in Fig. 5.5. A brief explanation of each domain is presented below. The Matlab-Simulink files used for the Digital Twin implementation are available at `https://www.theedgeai.com/dtandscebook`.

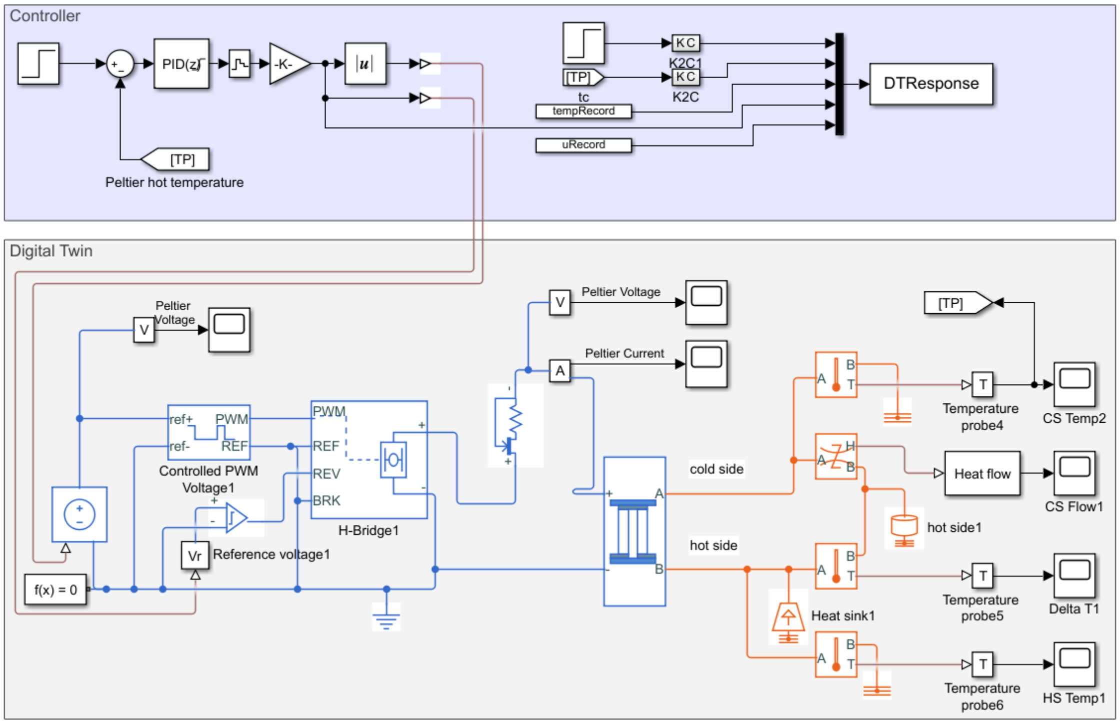


Figure 5.5: Assembled DT Multidomain Simulation

Electrical Domain

The electrical domain section of the DT model is shown in Fig. 5.6. This domain simulation is composed by the H-bridge to control the power flowing in the Peltier module via PWM using a controlled PWM voltage source. Also, a current sense comparator is included to set define the current flowing sense in the Peltier to alternate between heating and cooling behaviors. The PWM frequency is 500Hz given by the Arduino board used as PWM control module in the real system, with current sense threshold of 0.1(V). The properties of the H-bridge are the same defined in Table 5.1.

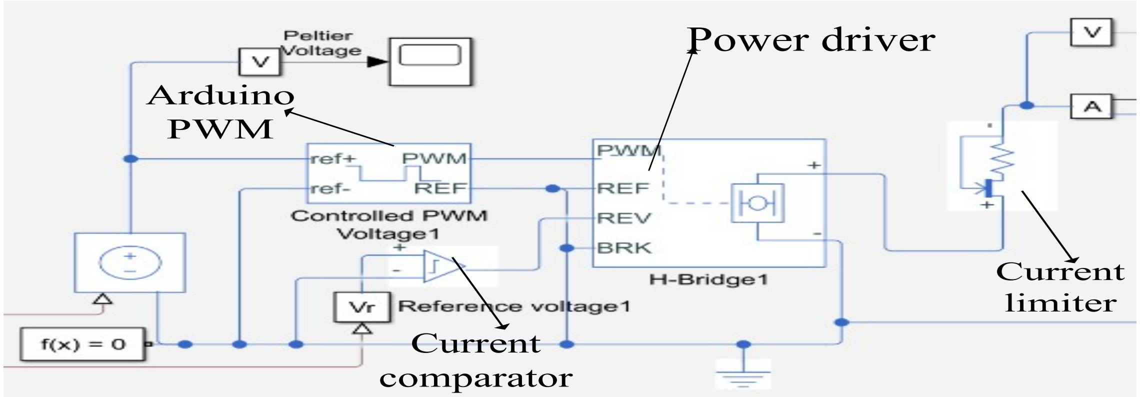


Figure 5.6: DT Case Study: Electrical Domain

Thermal Domain

The thermal domain simulation is presented in Fig. 5.7. As can be observed, the main element on this domain is the Peltier thermoelectric module, which is combined with a thermal mass and an ideal heat source to reproduce the real behavior of the Peltier. A description of the Simscape model of the Peltier thermoelectric module is presented in Fig. 5.8. It is composed of two faces A (hot side) and B (cold side) with an NP semiconductor junction between A and B and through an applied electric current flow, a temperature difference will be generated. This thermal interaction can be modeled using (5.1)-(5.3), where α is the Seebeck coefficient, R the electrical resistance, K the thermal conductance, T_A, T_B are the hot/cold side temperatures, Q_A, Q_B the hot/cold side thermal flows, and I, V represents the Peltier voltage and current respectively.

On the other hand, the thermal mass block represents the dynamic change of the heat flow Q in the hot side of the Peltier whose behavior is given by (5.4), where C is the specific heat of the Peltier device and m is the specific mass of the module. Finally the heat sink is modeled by a constant temperature source at the environment temperature, considering that its function is to keep the temperature constant in the opposite face of the Peltier to produce the difference between A and B.

Notice that α,R,K, and C required to run the Digital Twin model, initially can be derived from the Peltier datasheet. However, these values should be determined for the real system application data.

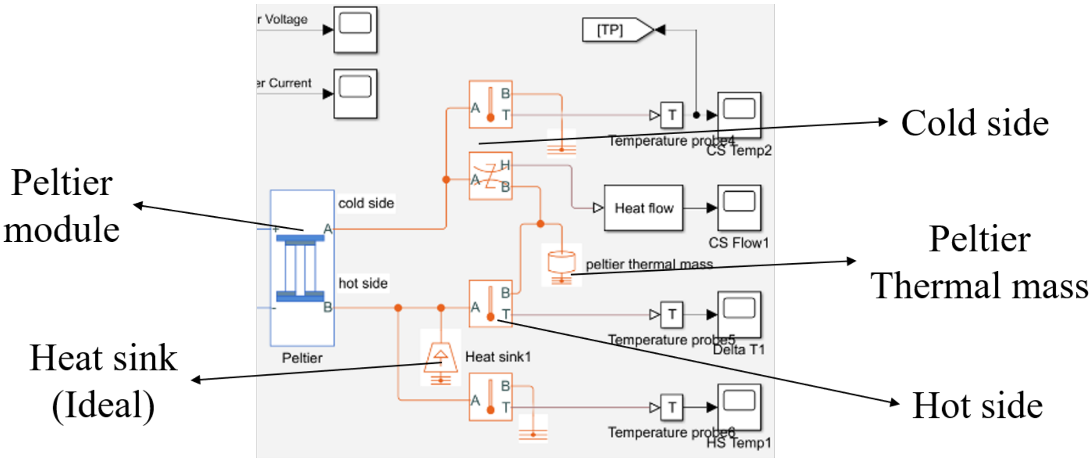


Figure 5.7: DT Case Study: Thermal Domain

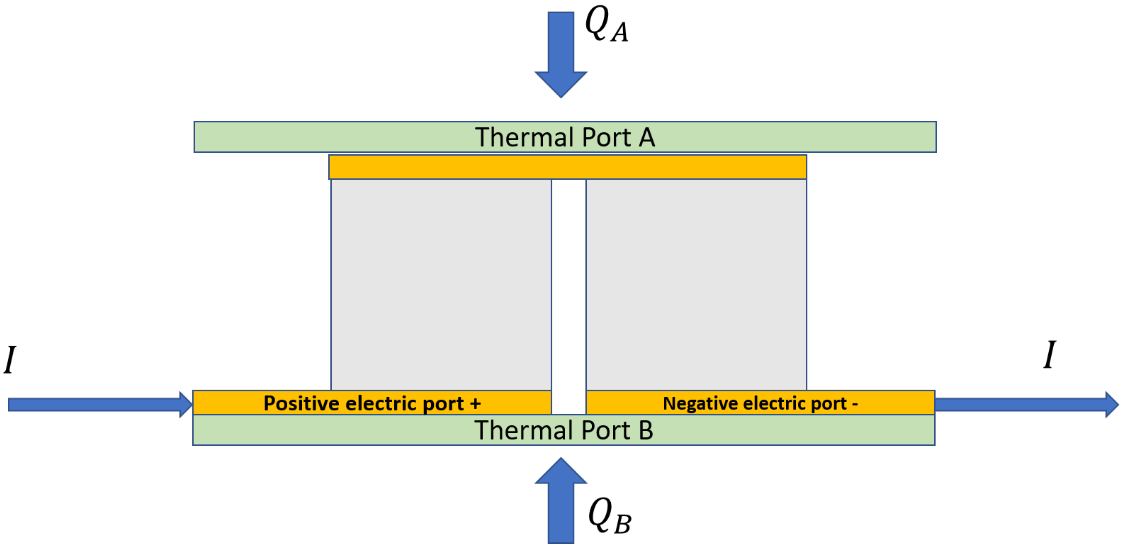


Figure 5.8: Peltier Thermoelectric Module Description

$$Q_A = \alpha T_A I - \frac{1}{2} I^2 R + K(T_A - T_B) \tag{5.1}$$

$$Q_B = \alpha T_B I - \frac{1}{2} I^2 R + K(T_B - T_A) \tag{5.2}$$

$$V = \alpha(T_B - T_A) + IR \tag{5.3}$$

$$Q = Cm\frac{dT}{dt}. \tag{5.4}$$

Digital Domain

The digital domain of the Digital Twin application is shown in Fig. 5.9. As can be observed, this domain includes the PID controller with anti wind-up, the reference signal, and the control action to be applied as PWM signal. For this system in particular, the physical implementation of this domain is performed using Hardware-in-the-Loop simulation with Matlab-Simulink. So, the behavior of these domain can be replicated with the best possible level of details. The gains and configuration of the PID controller are described in detail in [223, 224].

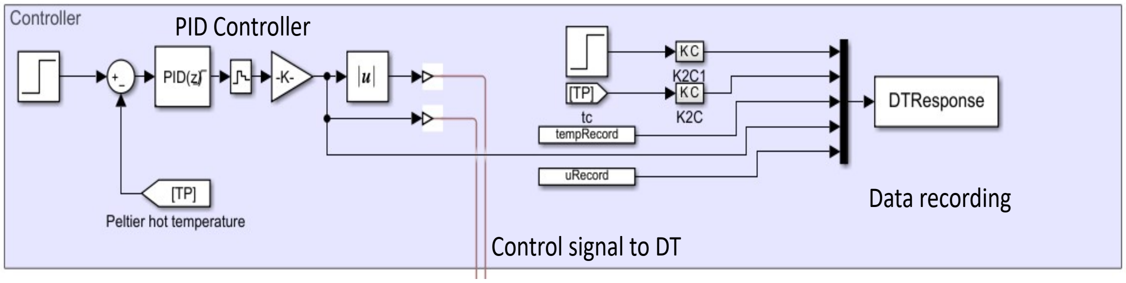


Figure 5.9: DT Case Study: Digital Domain

Step 4: Behavioral Matching

Due to the nonlinear behavior of the Peltier module, as well as the challenge for measuring heat flow and other thermal parameters, the behavioral matching is required to determine the values of α, R, K, and C. Based on the Peltier datasheet, some literature reported experimental measurements [220, 225], and previous experience manipulating the system; there is possible to know the initial guess for the behavioral matching process, which are presented in Table 5.2.

Table 5.2: Peltier Module Thermal Parameters

Parameter	Datasheet	Measurement [225]	Experience
α	53 mv	40 mv	75 mv
R	1.8 Ω	6 Ω	3.3 Ω
K	0.5555 K/W	0.3333 K/W	0.3808 K/W
C	15 J/K	15 J/K	31.4173 J/K

A set of real tests is performed to acquire real data from the system, consisting of applying different step reference signals, as shown in Fig. 5.10 to the system in order to evaluate its dynamic behavior for four different setpoints 30°C, 50°C, 70°C and 90°C. The control signal u, the system output temperature y and the reference signal r are registered for each setpoint to determine α, R, K, and C.

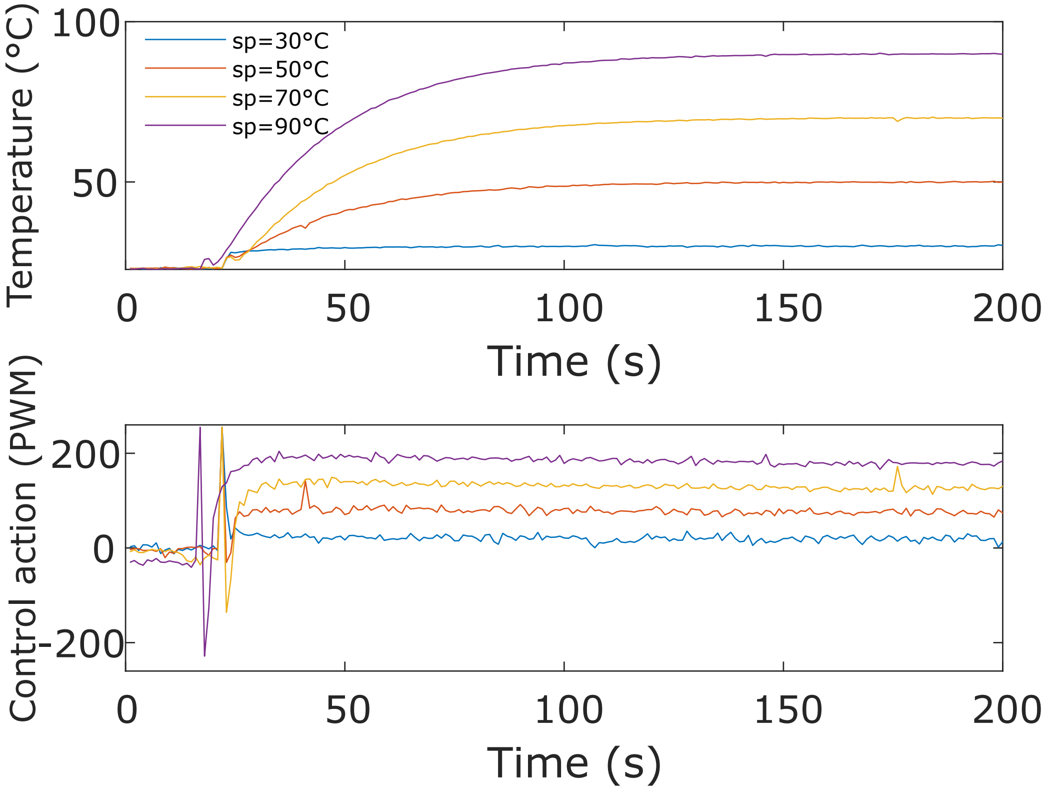


Figure 5.10: Peltier System Responses for Different Step Inputs

The nonlinear recursive least squares algorithm combined with the Simulink design optimization toolbox (SLDO) [226] is employed at each case to find the values of α, R, K, and C through matching the output and control action curves of the physical system with the Digital Twin. The sum of squared error is employed as the cost function for the parameter fitting problem defined by (5.5), where $e(k)$ are the system residuals and N the number of data samples. It is important to notice that $R = 3.3\Omega$, which was physically measured. The obtained parameters α, K, and C for each setpoint are presented in Table 5.3. It can be observed that the Peltier thermal parameters vary among the setpoints, indicating parametric uncertainty on the system as well as a significant difference with the parameters reported in Table 5.2. For example, Fig. 5.11 shows the Digital Twin response for 50°C setpoint with the parameters set obtained from behavioral matching registered in Table 5.3, confirming the presence of uncertainty also in the Digital Twin .

For this reason, applying model discrimination techniques is required in order to determine the most suitable and accurate set of parameters for the system's

Digital Twin.

$$F(x) = \sum_{k=0}^{N} e^2(k). \tag{5.5}$$

Table 5.3: Behavioral Matching Results for Different Setpoints

	Setpoint			
Parameter	30°C	50°C	70°C	90°C
α	$96.3mV$	$82.5mV$	$21.1mV$	$29.5mV$
R	3.3Ω	3.3Ω	3.3Ω	3.3Ω
K	$0.3K/W$	$0.35K/W$	$0.286K/W$	$0.38K/W$
C	$34.9J/K$	$31.93J/K$	$11.1J/K$	$13.7J/K$

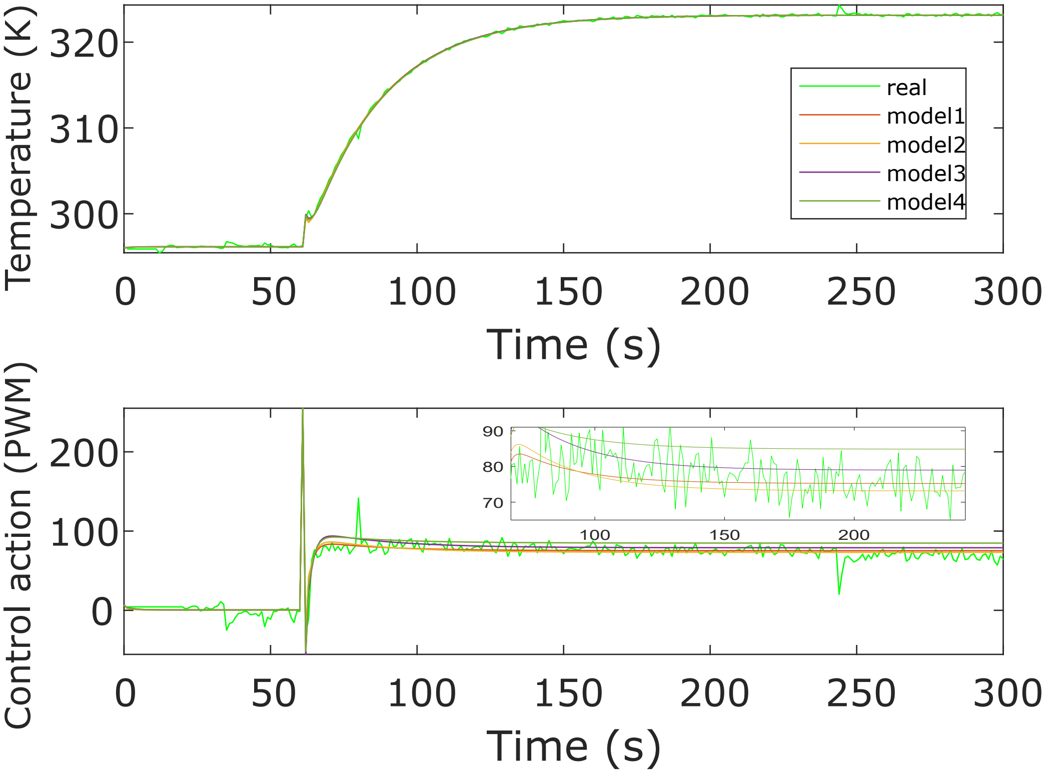


Figure 5.11: Digital Twin Uncertainty for a Setpoint of 50°C

Information Metrics

A set of information-based metrics are employed to perform the model discrimination assessment for the Digital Twin behavioral matching, which requires a model of the Digital Twin to determine the nominal set of parameters of the system.

Information Gain

Information gain proposed by [227] is based on the Kolmogorov complexity K. It is defined by (5.6) for a finite sequence of letters x, drawn using a finite alphabet A, where A is the output alphabet of a computer F, with p as a finite sequence of letters drawn using the input alphabet B for F, with $l(p)$ as the length of p. So that the Kolmogorov complexity is the length of the shortest program required to compute x.

$$K_f(x) = \begin{cases} \min_p & l(p), \quad \text{s.t } F(p) = x \\ \infty & \text{if no such } p \text{ exist}_. \end{cases} \tag{5.6}$$

Thus, $K_f(x)$ is a suitable measure of the smallest amount of information to obtain x. Considering that $K(x)$ is hard to compute, it can be related to the Shannon information in a random variable (5.7), with $H(x)$ as the entropy of x and $H(x|y)$ the conditional entropy of x with respect to another random variable y.

$$J(y : x) = H(x) - H(x|y). \tag{5.7}$$

So, if x and y are random sequences from an alphabet A, the algorithmic information of the sequence y regarding sequence x can be given in terms of the Kolmogorov complexity (5.8), where $I(y : x)$ is a measure of how much x relies on y for its calculation.

$$I(y : x) = K(x) - K(x|y). \tag{5.8}$$

This idea can be applied for model assessment, considering that the system observations can be divided in two datasets, one explained by the model (x) and another one that supports and helps to explain the first dataset (y). Thus, the quality of the model can be judged using a program to compute x from y and measure its length bounded by $K(x|y)$. As these value are lower, it indicates that the model better represents the system dynamics.

Assume a system S defined as a set of N input/output observations $S = (u, y)$ where $u = (u_1^m, u_2^m, .., u_N^m)$, $Y = (y_1^n, y_2^n, .., y_N^n)$ for some $m, n \geq 0$, $N > 0$. Each pair of observations u_i, y_i can be coded by a small integer r representing a numeration system ($r = 2$ and $r = 10$ for binary and decimal).

Besides, a model F for the system $S(u, y)$ can be defined as a computer program p that calculates the system output y based on its input u. So that, F can be defined as (5.9) where C_i is a subset $C_i = (A_i, B_i)$ with $A_i = u_i$ and $B_i = y_i$.

$$F(p, i, C_i) = y_i^n,\ i = 0, 1, ..., N. \tag{5.9}$$

From (5.9), the shortest model F of S is the one that uses the information $I((1, C_1),$ $(2, C_2), \cdots, (N, C_N) : y)$ more efficiently. However, calculating I from (5.8) is not possible due to the unknown $K_F(y)$ and $K_F(y|C)$, so only known models can be compared. For any system S, a trivial model t can be defined from the beginning, by reading the output y from a look-up table. So, for any model p of S, the information gain $I()$ is defined by (5.10)

$$I(p) = l(t) - l(p), \tag{5.10}$$

where $l(t)$ and $l(p)$ correspond to the lengths of the trivial and proposed models for the system. For any model, the length is given by (5.11), with $L_{program}()$ as the length of the computer program that describes the model, and $L_{table}()$, as the length of the lookup table.

$$l() = L_{program}() + L_{table}(). \tag{5.11}$$

In the case of t, the look-up table corresponds to the system output observations. For the model p, the look-up table records the difference between the system output y and the estimated output $\hat{y}$ given by the model p, quantifying the error or missing behavior captured by the model.

To calculate the length of the look-up tables $L_{table}()$ for t and p, these should be codified, assuming that each element in the table corresponds to a rational number that will be scaled and represented using a numeration system r. So, the code-length function $l()$ for each n element in the table is defined by (5.12), were [ ] represents the floor operation.

$$L(n) = [\log_r |n|] + 1. \tag{5.12}$$

We consider a decimal numeration ($r = 10$) system that is used for look-up table codification, treating each table element n as a high order integer, removing decimal period and adding the corresponding sign to n. For example, if $n = 10.34$, it is codified as "$+1034$" returning a length of 5, or if $n = -0.45$ its codification is "-45" returning a length of 3 always removing the leading zeros. Thus, the look-up table length is given by (5.13).

$$L_{table}() = \sum_{i=1}^{n} L(i). \tag{5.13}$$

Likewise, to calculate the program length $l_{program}$, a similar codification rule is applied, based on the number of code lines and commands required by a programming language to implement the model of the system. According to 5.10, some rules can be set to quantify the program length. Initially, an extended alphabet of 26 characters plus ten digits (0-9) and special symbols (#,%,+,-,.) are considered. Each character or digit in the code increases the length of the program by 1. However, the variable's names, as well as reserved words of the programming language, only increase the program length by 1. In [227], the models were implemented using ALGOL68, but in this monograph, the models will be implemented using Matlab.

Notice that as the information gain of the system $I(p)$ increases, it indicates that the model p offers a better explanation of the system behavior. Dividing $I(p)$ by $l(t)$ return the explanation degree of the model p, bounded between 0-1, where a value of 1 indicates the best level of explanation the system behavior by the model.

Normalized Akaike Information Criterion (nAIC)

According to [228], the Akaike Information Criterion (AIC) returns a measure of the model quality produced by simulating a situation where the model is tested in the presence of different datasets. This criterion compares the family of models information entropy via the Kullback-Leibler divergence. Thus, the most accurate model among a family of models is the one with the smallest AIC value. This criterion penalize the complexity of the system, it means, it will increase for systems with bigger structures and number of parameters. There are different AIC criterion forms. The normalized AIC is calculated, which is given by (5.14), where N is the number of samples, $\epsilon(t)$ is a vector of the prediction errors, θ_n is the vector of estimated parameters, n_y the number of model outputs and n_p the number of estimated parameters.

$$nAIC = N\log(\det(\frac{1}{N}\sum_{1}^{N})\epsilon(t,\hat{\theta})(\epsilon(t,\hat{\theta}))^T) + \frac{2n_p}{N}. \tag{5.14}$$

On the other hand, the Bayesian Information Criteria (BIC) [229] can be calculated from AIC, which is given by (5.16)

$$\begin{aligned} BIC = N\log(\det(\frac{1}{N}\sum_{1}^{N})\epsilon(t,\hat{\theta})(\epsilon(t,\hat{\theta}))^T) \\ +N(n_y\log(2\pi)+1)+n_p\log(N). \end{aligned} \tag{5.15}$$

Minimum Description Length (MDL)

The minimum description length (MDL) is an information theory based index for evaluating model complexity, penalizing the number of parameter required to represent the system behavior [230]. MDL can be calculated using (5.16), where V_{ml} is the loss function of the model for the estimated model parameters $\hat{\theta}$, d is the number of parameters in the system, and N the length of the output observations vector.

$$MDL = V_{ml}(\theta(\hat{z}), z)(1 + \frac{d}{N}) \ln(N). \tag{5.16}$$

ν-Gap Metric

The Vinnicombe ν-gap metric [231] is a measurement of distance between two linear time invariant (LTI) dynamic systems P_1 and P_2, with right coprime factorization $P_1 = N_1 M_1^{-1}$ and $P_2 = N_2 M_2^{-1}$ given by (5.17). It can be used as a stability indicator for robust control design. The ν-gap metric is always bounded between 0 and 1. As the value is close to zero the P_1 and P_2 are more similar with a stability margin degradation less than the ν-gap metric value.

$$\delta_v(P_1, P_2) = \max_{w} ||(I + P_2 P_2^*)^{-\frac{1}{2}}(P_1 - P_2)(I + P_1 P_1^*)^{-\frac{1}{2}}||_\infty. \tag{5.17}$$

The ν-gap metric is employed to measure the similarity between the family of models resulting from the behavioral matching. For this reason, the ν-gap metric is calculated between each model from the behavioral matching, creating a triangular ν-gap metric, which smaller column cumulative summation will indicate a better model.

Digital Twin Model Discrimination

Based on the information metrics presented above and considering that during the behavioral matching, the temperature y and control u action of the system were employed to determine the missing coefficients for a specific reference signal r, a single-input multiple-output (SIMO) system for the Digital Twin is proposed in Fig. 5.12. As can be observed, it is composed of two transfer functions one between $y(k)/r(k)$ and other for $u(k)/r(k)$. The goal of this SIMO model is to consider y and u in the model assessment regarding the same reference signal.

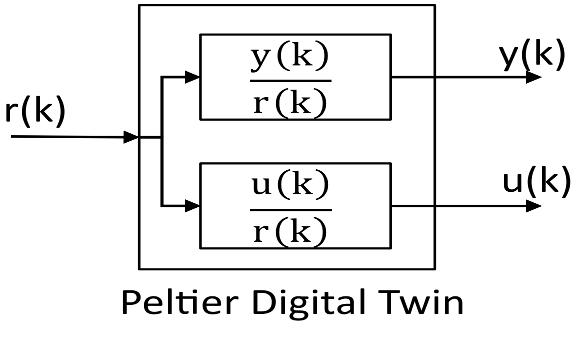


Figure 5.12: SIMO Model for the DT

On the other hand, the order of the SIMO model should be in the lowest order possible in order to satisfy the Occam's razor condition [232], it means reducing the model complexity to avoid overfitting. For this reason, four Box-Jenkins models (BJ) given by (5.18) are identified for $y(k)/r(k)$ and $u(k)/r(k)$ with second to fifth order polynomials for $B(z), C(z), F(z), D(z)$ for each set of parameters in Table 5.3, forming a 2x1 transfer function matrix. As an example, Table 5.4 shows the polynomial coefficients for the BJ models obtained for $y(k)/r(k)$ and $u(k)/r(k)$ using the second set of parameters for a setpoint of 50°C.

$$y(z) = \frac{B(z)}{F(z)}u(z) + \frac{C(z)}{D(z)}e(z). \tag{5.18}$$

Table 5.4: Box-Jenkins (BJ) Family Models for the Behavioral Matching Results at 50°C

Polynomial		Order 22221	Order 33331	Order 44441	Order 55551
$\frac{y(k)}{r(k)}$	**B**	0 0.03 -0.028	0 -0.001 0.002 0	0 -0.011 0.01 0 0	0 -0.045 -0.603 1.245 -0.601 -0.01
	C	1 -0.817 0.002	1 1.235 0.609 -0.016	1 -0.048 -0.007 -0.695 -0.001	1 2.098 1.155 -0.075 -0.085 0.047
	D	1 -1.772 0.786	1 0.247 -0.613 -0.634	1 -1.035 0.041 -0.689 0.683	1 1.094 -0.966 -1.31 -0.033 0.214
	F	1 -1.997 0.999	1 -2.949 2.899 -0.95	1 -1.806 -0.158 1.735 -0.77	1 -1.776 1.612 -1.067 0.392 -0.125
$\frac{u(k)}{r(k)}$	**B**	0 0 0	0 0 0 0	0 -0.006 0.011 -0.006 0.002	0 0.089 -0.217 0.144 0.013 -0.029
	C	1 0.003 0	1 0.127 -0.077 -0.001	1 0.142 -0.372 -0.297 -0.02	1 -0.161 0.017 0.607 -0.233 -0.057
	D	1 -0.995 -0.006	1 -0.87 -0.204 0.074	1 -0.84 -0.563 0.121 0.282	1 -1.156 0.971 -1.34 0.716 -0.186
	F	1 -1.978 0.978	1 -2.219 1.455 -0.237	1 -2.264 1.117 0.557 -0.41	1 -2.365 2.259 -2.21 2.148 -0.832

Now, the model discrimination criteria are calculated for the identified SIMO system for each set of parameters presented in Table 5.3. In the case of information gain, each BJ model is evaluated as a difference equation employing only the transfer function part of (5.18). From (5.10), the information gain is given by the difference between the trivial $l(t)$ model and the BJ model $l(BJ)$. Likewise, the length of each program is calculated as the sum of the lengths of the computer program plus the look-up table (5.11). In the case of the trivial program, its length $l(t)$ is calculated using the coding rules given by (5.12), which are implemented in Matlab with a length of 15, being the same for all the trivial models. Regarding the look-up table for the trivial model t, it is coded using the rules defined by (5.13), and its length depends on each real setpoint response.

The implementation of BJ models is also performed in Matlab with a length of $l(BJ) = 176$. Considering that the same code works for any of the proposed BJ models, the code length $l(BJ)$ keeps constant at each calculation. Regarding the look-up table, it is calculated as $y - \hat{y}$, where y is the physical system response, and $\hat{y}$ is the response obtained from each BJ model evaluated. Again, its length depends on $y - \hat{y}$, calculated using (5.13)-(5.11).

Finally, the total information gain of the SIMO model is calculated as the sum of the individual information gains from $y(k)/r(k)$ and $u(k)/r(k)$. In this case, the most suitable model is the one with the highest information gain, it means, the one that provides more information about the system. The trivial and BJ models codes can be found in `https://www.theedgeai.com/dtandscebook`.

Considering that only one criterion may not be enough to choose the most suitable model for the system, the nAIC, BIC, and MDL information gain criteria are calculated for the SIMO system, using the expressions (5.14)-(5.17). Table 5.5 shows the calculation of the information criteria for each MISO BJ model regarding its corresponding dataset. As can be observed, the information gain shows that for setpoints 50°C and 90°C, a second-order BJ model is enough to represent the system dynamics, while for setpoints 30°C and 70°C, models of third and fourth-order are more representative for that specific datasets. It is important to say that the information gain method is sensitive to the decimal precision of the measurements as well as the look-up table.

On the other hand, it can be noticed that using the nAIC, BIC, and MDL criteria, the second-order BJ model is the best model to represent the system dynamics. So, we can say that based on the multiple assessment metrics employed, a second-order BJ model represents the Digital Twin dynamic with the best trade-off between complexity and overfitting.

Once the best type of SIMO model for the Digital Twin is selected, the next step consists of determining the nominal set of parameters of the Digital Twin , that works for multiple operating points. In that sense, the νGap metric is calculated for the second-order BJ models obtained for each operating point. Thus, the set of parameters with the less cumulative ν-gap metric determines the nominal set of parameters, considering that νgap metric measures the distance between the models based on the H_∞ norm seeking presented in (5.17). The obtained result of the ν-gap metric for the second-order BJ models are 0.93, 0.74, 0.22, and 0.28 for the 30°C, 50°C, 70°C, and 90°C setpoints respectively. It can be observed that the smallest value of ν-gap metric is given for the third set of parameters corresponding to a setpoint of 70°C. So that, we can say that these values of α, R, K, C correspond to the nominal operation parameters for the Digital Twin.

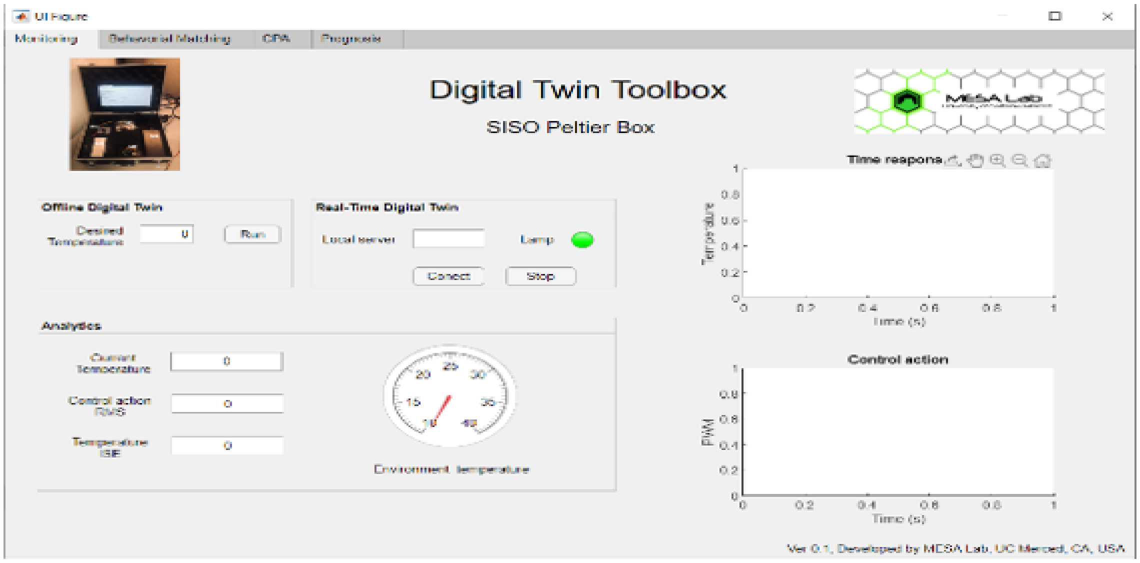
Table 5.5: Information Criterion Calculation for Digital Twin Model Assessment

SP	Model	$y(k)/r(k)$			$u(k)/r(k)$			IGT	nAIC	nAIC	nAIC
	order	l(t)	l(BJ)	IG(y)	l(t)	l(BJ)	IG(y)	(u,y)	$y(k)/r(k)$	$u(k)/r(k)$	
30	22221	1242	681	561	1019	1014	5	566	310.24	330.39	640.63
	33331	1242	741	601	1019	1055	-36	465	472.21	383.82	856.06
	44441	1242	637	605	1019	998	21	626	819.77	464.58	1284.35
	55551	1242	23	1019	1019	1078	-59	-36	429.28	467.93	897.21
50	22221	2048	979	1069	1622	1493	129	1198	0.78	0.80	1.58
	33331	2048	1082	966	1622	1506	116	1082	0.84	0.84	1.68
	44441	2048	1067	981	1622	1549	73	1054	0.86	0.84	1.70
	55551	2048	1836	212	1622	1653	-31	181	0.93	0.93	1.87
70	22221	2898	1286	1612	2890	2042	848	2460	1.82	2.24	4.06
	33331	2898	1258	1640	2890	1712	1178	2818	1.91	2.26	4.17
	44441	2898	1429	1469	2890	2107	783	2252	1.94	2.17	4.11
	55551	2898	2591	307	2890	2013	877	1184	1.97	2.29	4.26
90	22221	2721	1219	1502	2712	1930	782	2284	1.88	2.30	4018
	33331	2721	1519	1202	2712	1904	808	2010	2.28	2.29	4.57
	44441	2721	1386	1335	2712	2712	2014	698	2033	1.90	2.28
	55551	2721	1170	1551	2712	2143	569	2120	4.17	1.91	6.08

SP	Model	BIC	BIC	BIC	MDL	MDL	MDL
	order	$y(k)/r(k)$	$u(k)/r(k)$		$y(k)/u(k)$	$r(k)/r(k)$	Total
30	22221	310.21	330.39	640.63	0.27	0.27	0.54
	33331	472.21	383.82	856.03	0.34	0.34	0.68
	44441	819.77	464.58	1284.35	0.49	0.49	0.97
	55551	429.228	467.93	897.21	0.48	0.48	0.96
50	22221	1283.65	1290.03	2573.68	2.52	2.52	5.03
	33331	1327.9	1325.32	2653.19	2.75	2.75	5.50
	44441	1358.3	1349.62	2707.92	2.88	2.88	5.77
	55551	1404.68	1404.46	2809.15	3.33	3.33	6.67
70	22221	2337.72	2543.94	4881.6	10.41	10.41	20.80
	33331	2406.90	2579.63	4986.53	11.13	11.13	22.27
	44441	2446.20	2561.95	5008.16	10.61	10.61	21.21
	55551	2485.83	2642.01	5127.84	15.18	15.18	30.37
90	22221	2226.30	2417.66	4643.96	11.06	11.06	22.11
	33331	2434.50	2437.63	4871.86	11.43	11.43	22.86
	44441	2274.10	2457.17	4741.27	11.78	11.78	23.57
	55551	3354.17	2312.17	5666.34	12.38	12.38	24.77

Step 5: DT Validation and Deployment

Once the Digital Twin model passed through the behavioral matching process, it is ready for its deployment, running simultaneously with the physical system. For this reason, a supervisory interface as well as a communication architecture has to be defined to connect and monitor the real system with its Digital Twin . The monitoring interface is implemented in Matlab using the appdesigner tool, which is shown in Fig. 5.13. As can be observed, the interface offers the possibility of interacting with an offline version of the Digital Twin to verify its proper operations for different setpoints. Likewise, there is a panel for the real-time connection of the system with the Digital Twin .



Desired Temperature: 0

Current Temperature: 0

Control action RMS: 0

Temperature ISE: 0

Figure 5.13: Digital Twin Supervisory Interface

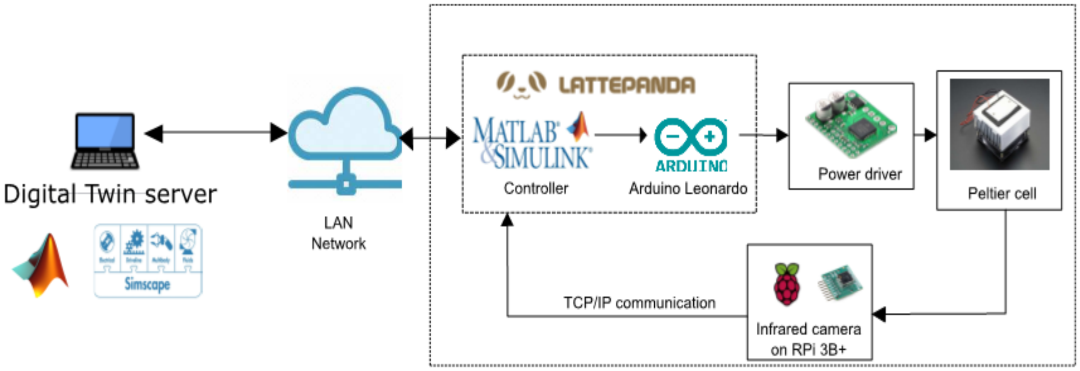


Figure 5.14: DT Parallel Deployment Architecture

On the other hand, the communication interface among the DT and the system is presented in Fig. 5.14. This interface connects the external computer where the Digital Twin is executed using a client/server configuration inside a local network with the real thermal system via TCP/IP protocol with a communication frequency of one second. Thus, the Digital Twin is fed in real-time with the same control action, output temperature, environmental temperature among other variables of the physical system to simulate its physical behavior via proper interface, one can see how different it is for the Digital Twin running in real-time with the system. It is important to notice that, even with multiphysics simulation software like Simscape, this Digital Twin implementation is able to handle the real-time simultaneous running. However, as the model increases its complexity, more computational power is required to run the Digital Twin in this way.

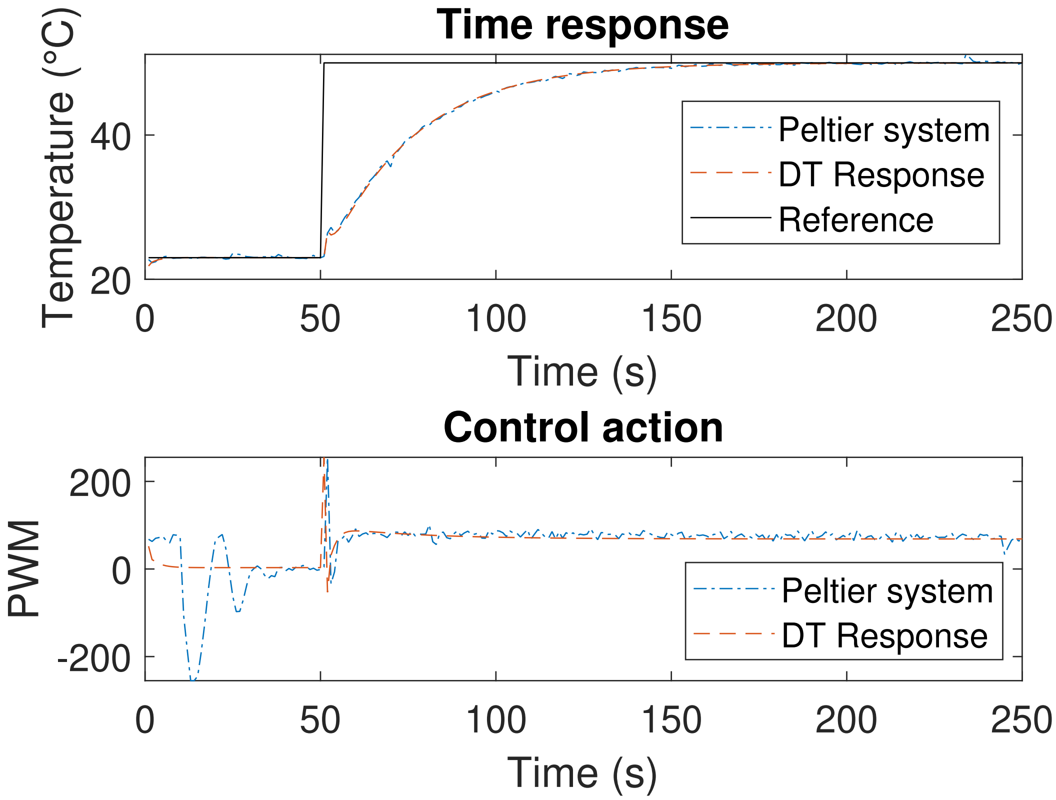


Figure 5.15: DT Parallel Architecture Deployment Response

Figure 5.15 shows the results of the real-time Digital Twin operation with the real system, for a setpoint of 50°C. As can be observed the Digital Twin replicates the real system behavior under the real operation conditions. Also, it is important to notice that the environmental temperature has a significant influence over the Digital Twin performance considering that it change the starting point of the physical system. For this reason, this variable is measured and incorporated in the Digital Twin modeling and execution. The Digital Twin interface operation can be watched at `https://youtu.be/acXTNmcCIYs`.

5.4 Chapter Summary

In this chapter, a methodological framework has been proposed to build a Digital Twin for a closed-loop system real-time infrared vision feedback temperature uniformity control towards the implementation of smart control engineering. The proposed framework is compared with other proposals in the literature, showing that our proposed framework is focused for control systems design. The framework employs five steps, which go from a detailed review of each component and subsystem, recreating the system behavior using multidomain simulation, adjusting to fit the real system using the behavioral matching technique to finish with the real-time interface between the Digital Twin and the physical system. A multimodel assessment methodology has been proposed to determine the best set of parameters that represent the physical system behavior based on information criteria. Obtained results show that the Digital Twin represents correctly the system behavior operating in parallel and offline modes. At this point, the Digital Twin of the real-time infrared vision feedback temperature uniformity control system is ready to incorporate all the enabling technologies required for the implementation of smart control engineering described in detail in Chapter 6.

Chapter 6

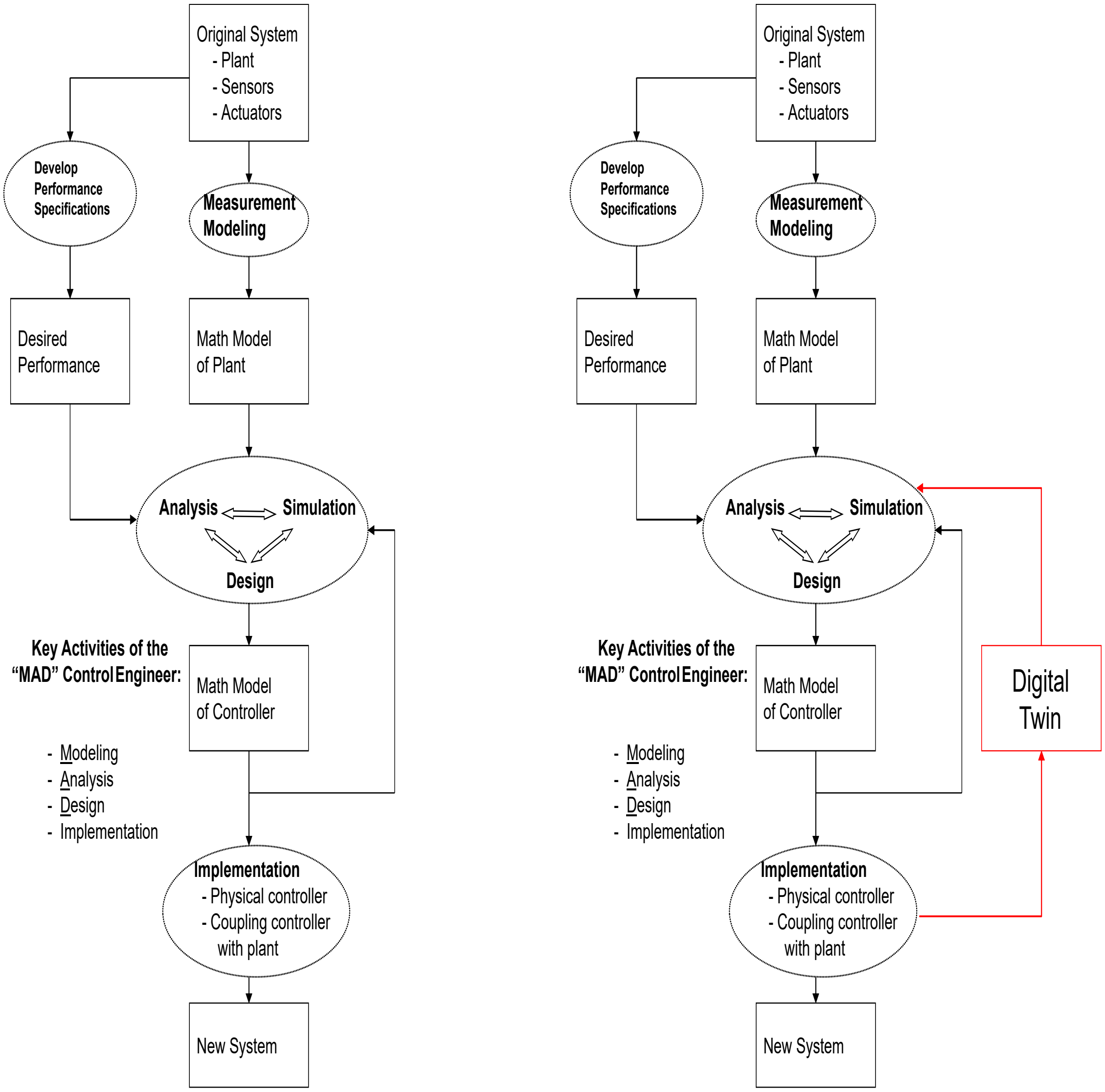
DIGITAL TWIN ENABLING CAPABILITIES

6.1 Introduction

Considering that digital twin offers a highly detailed virtual environment that represents the behavior of a physical system by component level as well as the interaction of the multiple agents or subsystems, a set of enabling capabilities and emerging applications for control engineering are open for exploration with potential uses on industrial processes improvement and smartness integration. In this chapter some of those enabling capabilities like smart control systems design, fault detection and prognosis, control performance assessment (CPA), or self optimizing control (SOC) are presented as Digital Twin applications on control systems. Likewise, some of these capabilities are tested using the Peltier thermoelectric system as a case study introduced in Chapter 5.

6.2 MAD Methodology for Control Engineering Practice and Education

Classic control engineering workflow can be described by the modeling, analysis, and design methodology (MAD) proposed in [1]. This framework is shown in Fig. 6.1(a) which is based on two questions: "What do you want?" and "What do you have?" The first one is related to the desired performance specifications for the system to be controlled. The second one, is what information of the system is available for modeling and controller designs. Once these questions are well defined, the first step, modeling is performed to obtain a mathematical model of the system to be controlled. These models are based on the available system data and could be expressed through a transfer function, a state space, or nonlinear representations obtained using different system identification techniques.



(a)

(b)

Figure 6.1: Modeling, Analysis, and Design (MAD) Methodology for Control Engineering (a) Classic Approach and (b) MAD Enabled with Digital Twin

In the second step, analysis, stability and robustness studies of the system are performed in order to determine the most suitable control strategy that satisfy the desired operating conditions like PID, model predictive, robust, adaptive, non-linear among other techniques.

In the third step, design, the mathematical model of the control strategy is defined and its parameters are calculated. After that, the system with controller is set up into a simulation model to validate the response of the system to verify that the desired operating conditions are reached. Finally, once the controller works on simulation environment, the next step is its implementation in the physical device and its validation.

The MAD framework has been followed by control engineers for many years with successful results. However in the Industry 4.0, with cyber-physical systems and multi agent interaction introducing environmental variability, static models of the system to be controlled are not capable of accommodating the uncertain dynamic changes. Also, these models do not consider detailed component modeling and agents interaction, reducing the cognizant capabilities of the system.

In this scenario, Digital Twin offers a possibility of having a real-time updated model of any system that can be combined into a MAD cycle to make the system aware of its current status and take the appropriate control actions to compensate the environment variability as presented in Fig. 6.1(b). It can be observed that the MAD methodology incorporates a new feedback loop to modify the design of the control system based on the updated model of the system and the multiple insights derived from the Digital Twin analysis. Thus, MAD is augmented with self-awareness of the current status of the system, incorporating the characteristics of an intelligent physical system (IPS) system.

6.3 Control Performance Assessment

Control performance assessment (CPA) is for measuring the control quality of a closed-loop system against different control methodologies [2]. In control engineering, it is a mandatory task for control system design in order to reach the desired and optimal response of the system to be controlled. The implementation of control performance assessment requires a closed-loop system operating in a stable condition, a set of important measurements from the system collected by sensing devices, a processing methodology that includes a set of key performance indices (KPI), and software tools to performing control performance assessment analysis. As an example, a standard single-input, single-output (SISO) control system block diagram is presented in Fig. 6.2. It is composed by the plant or the physical system to be controlled, the controller or control rule for the system, the actuators, and feedback sensors. Likewise, the SISO system incorporates the output signal $y(t)$, desired reference $r(t)$, error signal $e(t)$, control action $u(t)$, and the disturbance signals $d(t)$ and $z(t)$ for the control action and system output respectively.

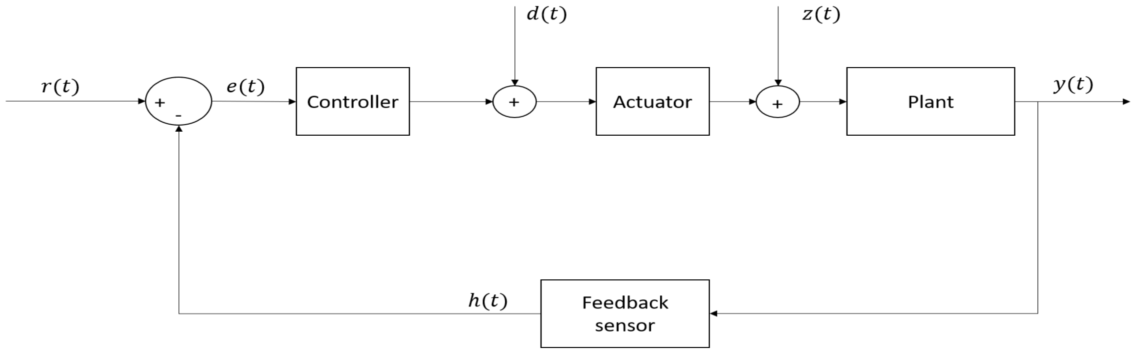
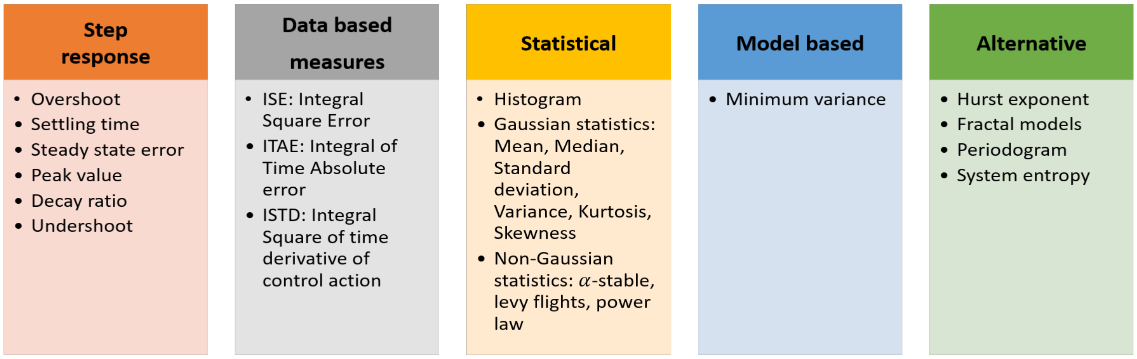


Figure 6.2: Closed-Loop System Block Diagram

Based in the signals in Fig. 6.2, the control performance assessment can be performed, calculating some of the key performance indices [2]. A list of the performance indices employed in control engineering can be found in Fig 6.3. It can be observed that these indices are segmented into step response, data based, statistical, model based, and alternative indices. Some of these indices like overshoot, settling time, mean or standard deviation, integral square error (ISE), or minimum variance are commonly employed for control performance assessment, assuming a Gaussian behavior of the sampled signals. Thus, the effect of aggressive or soft control actions, feedback sensor failure, or the presence of disturbances in the control loop can be quantified to determine how much it deviates from its desired behavior. In the context of Industry 4.0, not only the disturbance behavior but also the signals of the closed-loop system may behave following non-Gaussian probability distributions due to the environment variability and the system interconnected features. In this case, CPA requires alternative performance indices like the α-stable distributions, fractals, or system entropy [2].



Step response

- Overshoot
- Settling time
- Steady state error
- Peak value
- Decay ratio
- Undershoot

Data based measures

- ISE: Integral Square Error
- ITAE: Integral of Time Absolute error
- ISTD: Integral Square of time derivative of control action

Statistical

- Histogram
- Gaussian statistics: Mean, Median, Standard deviation, Variance, Kurtosis, Skewness
- Non-Gaussian statistics: α-stable, levy flights, power law

Model based

- Minimum variance

Alternative

- Hurst exponent
- Fractal models
- Periodogram
- System entropy

Figure 6.3: CPA Key Performance Indices (KPI)

However, CPA must also deal with information quality issues like wrong sampled data, missing, or unavailable data that cannot be produced in the system for multiple reasons like broken sensors, failures in the data recording, or lack of direct measurements of the system signals.

For closed-loop system analysis, Digital Twin offers the possibility of performing CPA based on a current model of the physical system, which can be adjusted based on the available real data. Indeed, this enables KPI calculation from subsystems without direct measurement datastreams. Also, DT enables testing multiple benchmark tests for controller performance employed in CPA like PID, MPC, or linear quadratic Gaussian regulator LQG [233] in order to have a quantitative measurement of the most suitable control strategy for the system evaluated via Digital Twin for its real implementation.

6.4 Parallel Control Under ACP Approach

The ACP approach (artificial systems + computational experiments + parallel execution) proposed in [234], is a framework to perform human and social studies recreating individual behaviors and elements of an actual society model into an artificial society model. It leverages artificial intelligence and computational tools to perform extended experimentation and evaluation tasks of a society at the same time that the artificial society model is into a constant training and learning process from real-time data collected from the actual society to perform management and control actions on the real world.

ACP approach relies on computational representations of each agent, environment and rule composing the complex social model, expanding the possible social analysis, which are hard to implement in real life. Likewise, the ACP virtual models run in parallel execution and it means simultaneously with the real social system to enable the functions of management, control, experimentation, evaluation as well as real-time training and learning for the virtual model based on the real system. Several applications of parallel control with ACP approach can be found on intelligent transportation [235, 236], agriculture [237], simultaneous measurement [238] or parallel dynamic programming [239].

Notice that the ACP approach has a strong relation with control theory, considering that the artificial models can be defined as an extension of an automatic control regulator, which based on a set of states and a defined model of the real system can reach a desired operation point. In the sense of smart control engineering, this artificial model can be understood as a virtual representation of a physical asset, which corresponds to the Digital Twin definition.

Therefore, parallel control and ACP approach requires a Digital Twin in order to perform regulation and analysis tasks over a complex physical system, transforming Digital Twin into the key component for a successful implementation of ACP approach.

6.5 Fault Diagnosis, Prognosis, and Health Management

Fault detection is a branch of control engineering focusing on monitoring a system, identifying its failures, and notifying its kind and location using only the input and output data streams of the system. Thus, it is possible to detect not only the system failures but also discover hidden behavior patterns. The benefits include reduced plant stops, increased productivity and minimizing money losses. However, failure detection could be a challenging task depending on the system complexity, the prediction speed (offline/real-time), the amount of data, the number of performance indices evaluated, or the previous failure classification. Indeed, sometimes the failure detection can be performed manually but it requires experimented and well-trained personnel. An alternative is to perform it automatically using machine and deep learning (ML/DL) models . In the literature there are several applications of machine and deep learning techniques employed in fault detection [240–245]. In some of the proposed methods, the dataset for the health and failure cases is enough for tuning a machine learning based detector. However, there are cases where the lack of information regarding to the fault behavior in particular is not enough to train the detector. This is common when the fault event represents some critical behavior on the system that is not common and is risky for the process and its operator. In those scenarios, Digital Twin, as a virtual representation of the system allows to analyze the system behavior under these dangerous conditions and produce reliable synthetic datasets to train the fault detection model. There are several examples regarding to the use of Digital Twin for fault detection [58, 246–249], where the DT replicates hazardous conditions, allowing the fault detection training and validation before its physical implementation.

On the other hand, Digital Twin can be employed to perform prognosis and health monitoring over the components of a complex system. The prognosis can be understood as the ability to know the condition of a specific component to perform maintenance before a critical fault occurs [244]. One of the most useful prognosis techniques is the estimation of the remaining useful life (RUL) of a component, which estimates the lifespan of a component before it reaches a critical fault behavior. An example can be observed in Fig. 6.4. It may be based on one or multiple health indicators corresponding to signal features obtained from the system available information including statistics like mean, median, standard deviation, kurtosis, skewness or other metrics like root mean square value (RMS). Thus, a safety margin on the parameter can be defined to indicate that the components need to be replaced before it reaches the failure threshold.

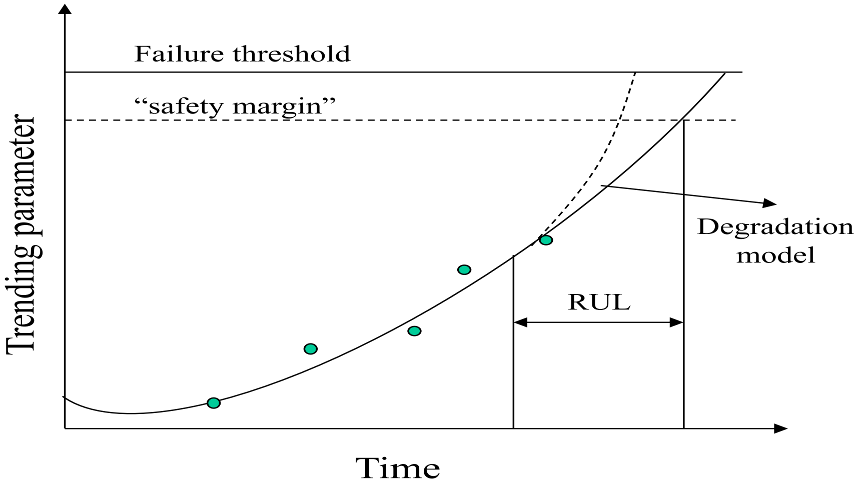


Figure 6.4: Remaining Useful Life (RUL) of a component

RUL estimation is based on the time series of the health indicator. In order to know the most suitable health indicator for a component three condition indicators are evaluated corresponding to monotonicity, prognosability and trendability [244, 250]. For a variable x with length N which is evaluated in M systems, the monotonicity given by (6.1) characterizes the trend of a feature as the system evolves toward failure, with positive or negative slope. Likewise, the prognosability (6.2) is a measure of the variability of a feature at failure relative to the range between its initial and final values. Finally the trendability (6.3) provides a measure of similarity between the trajectories of a feature measured in multiple run-to-failure experiments.

$$\text{monotonicity} = \frac{1}{M}\sum_{j=1}^{M}\left|\sum_{k=1}^{N_j-1}\frac{\mathrm{sgn}(x_j(k+1))-x_j(k)}{N_j-1}\right| \tag{6.1}$$

$$\text{prognosability} = \exp\left(-\frac{\mathrm{std}(x_j(N_j))}{\mathrm{mean}_j|x_j(1)-x_j(N_j)|}\right), \quad j=1,\cdots,M \tag{6.2}$$

$$\text{trendability} = \min_{j,k}|\mathrm{corr}(x_j,x_k)|, \quad j,k=1,\cdots,M. \tag{6.3}$$

Once the index is selected, the remaining useful life can be estimated using three types of prognosis models: similarity, degradation, and survival. The similarity models require a historic register of the run to failure history, the degradation models requires some knowledge regarding to the failure thresholds and the survival models employ lifetime data of the system with or without covariates like environmental variables. In this book, the degradation models will be applied to estimate the RUL in combination with the Digital Twin. Among these models, the linear and exponential degradation given by (6.4) and (6.5) are calculated, where ϕ is the model intercept that is positive indicating the lower bound or negative indicating the upper bound; $\theta(t)$ is the model slope as a random variable with a normal distribution with mean θ and variance θ_{var}; $\epsilon(t)$ is the model additive noise and is modeled as a normal distribution with zero mean and variance σ^2, and $\beta(t)$ is a random variable modeled as a Gaussian distribution with mean β and variance β_{var}.

$$S(t) = \phi + \theta(t)t + \epsilon(t), \tag{6.4}$$

$$S(t) = \phi + \theta(t)e^{\beta(t)t+\epsilon(t)-\frac{\sigma^2}{2}}. \tag{6.5}$$

It is important to notice that RUL estimation requires large information amount regarding to the system performance before it reaches its fault state. However, the information may be scarce specially when it is related to fault behaviors due to events whose occurrence rate is very low and its fault condition may not be reproducible due to cost and safety reasons. In this case, as with fault detection, using Digital Twins allows the evaluation of the physical system under different conditions, specially reproducing the degradation patterns in certain components and how the change affects the overall performance of the system, not only new consequences of the component degradation can be analyzed but also the correlation among the RUL of the system components. Thus, DT can be used to construct an indicator of a strong fault behavior or derive in new knowledge for smart predictive maintenance. In the literature, there are some related works that use Digital Twins to perform prognosis and RUL estimation. For example, [159] proposes a framework to perform prognosis and health management using rule-based methodologies performing sensor information fusion applied to wind power generators. In [251,252], a prognosis based on degradation models is applied to the gearbox bearings monitoring of a wind turbine.

Other health management applications of Digital Twin include aging estimation and outlier detection. In the case of aging estimation, there is a special focus on battery state of charge. For example, [56] shows the state of charge and RUL estimation of a spacecraft battery. In [25], a cloud computing DT architecture is presented for a building energy storage calculating the aging level based on state of charge and RUL. In [253,254], Digital Twin is employed to create a virtual representation of a battery testbed to estimate its RUL and state of charge. In the case of outlier detection, the topic is extensively treated in literature under different approaches [255]. In the case of Digital Twin, some applications can be found [155,247,256] which are related to the use of DT in an offline stage for tuning the detection algorithm by modeling the stochastic behaviors of random signals based on the Digital Twin.

In summary, Digital Twin has a wide range of applications towards fault detection, health management, outlier detection and aging considering that DT as a virtual representation provides a reliable environment for training and adjust each of the techniques by testing different scenarios not available in real operation due to the scarce of data or the associated risk to reproduction. In this chapter, a case study will be presented to show the potential of DT as well as the resultant enabling capabilities.

6.6 Self Optimizing Control

Self optimizing control (SOC) is a control strategy designed initially for controlling chemical plants with tens or hundreds of control variables (CV). It consists of choosing the best CV that satisfies a cost function in terms of economic performance with constant setpoints based on a system model using one or multiple optimization stages at different timescales (minutes, days, months) [31]. However, SOC may have different scopes according to the technique employed to reach an acceptable system performance as shown in Fig. 6.5. One point of view for SOC can be the use of seeking algorithms like extremum seeking [32], or maximum power point tracking (MPPT) [33], when the desired behavior has to be reached based on a cost function with little or no knowledge of the system dynamics. Another scope of SOC is using high Real-Time Optimization algorithms (gradient based or derivative free) for low-level control loop parameters, or setpoint adjustment based on some desired specifications under some periodic task [34]. Likewise, other SOC schemes can be reached using repetitive control strategies like run to run or iterative learning control [257, 258].

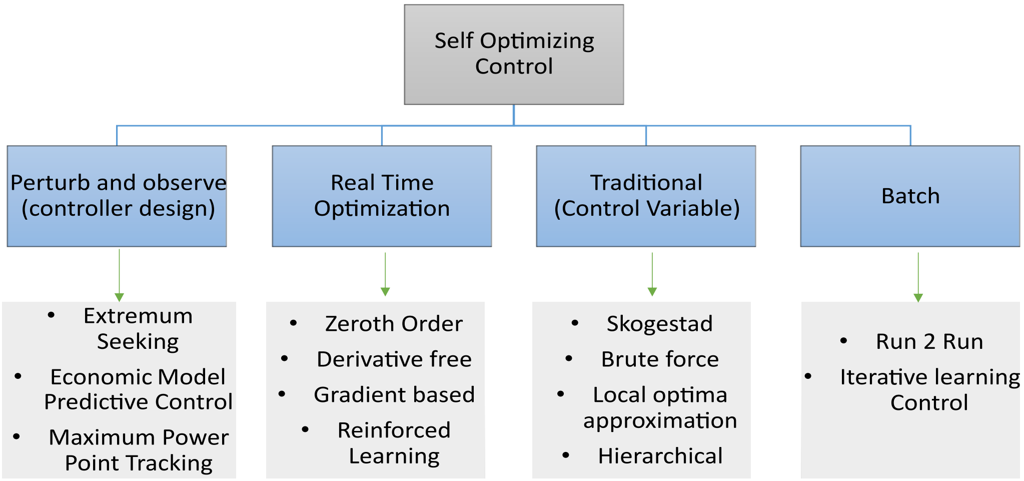

Self Optimizing Control
Perturb and observe (controller design)
Real Time Optimization
Traditional (Control Variable)
Batch
• Extremum Seeking
• Economic Model Predictive Control
• Maximum Power Point Tracking
• Zeroth Order
• Derivative free
• Gradient based
• Reinforced Learning
• Skogestad
• Brute force
• Local optima approximation
• Hierarchical
• Run 2 Run
• Iterative learning Control

Figure 6.5: Classification Self-Optimizing Control Methods

It is important to notice that for any of the proposed SOC strategies, a long time is required to find an optimal response that ensures the system performance and maintains the stability of the closed-loop system. This is specially critical in processes with low or slow response and big safety and security constraints like thermal or petrochemical. Therefore, it is desirable that any set of adjustments performed over the system control variables or tuning parameters can be simulated under various scenarios, verifying that the system keeps its performance and stability. Thus, Digital Twin as a reliable virtual representation of the physical system can be used as a complementary technology for testing and validate Self-Optimizing Control strategies in real-time. It means that a set of parallel Digital Twins can be executed with the physical system, which evaluates the feasibility of the current proposed solution as well as evaluate other candidate solutions proposed by the SOC strategy if any search methods with multiple solutions like metaheuristic algorithms are employed. This enabling capability is crucial for the smart control engineering and its implementation is covered in detail in Chapter 7.

6.7 Edge Computing Devices for Digital Twin

Considering the enabling capabilities presented above and the virtual architecture and software components of the Digital Twin applications, this section presents an overview of hardware alternatives for the implementation of Digital Twin applications. It is important to notice that the the Digital Twin applications presented in this book for control systems are designed for its implementation using edge computing devices that can be attached directly to the control systems to allow the implementation of the enabling capabilities.

In a formal sense, edge computing is a computational paradigm that decentralizes information management assigning more computing power close to the source of information to perform complex operations at the source. In the edge computing paradigm, the computing devices are not only data consumers but also data producers. At the edge, the computing devices can not only request service and content from the cloud but also perform the computationally expensive task directly on the source of the information, including executing deep learning algorithms for face recognition, fault detection, or prognosis as well as acts as supervisory layers for the closed-loop system. Edge can perform computing offloading, data storage, caching, and processing, as well as distribute request and delivery service from cloud to user.

Thus, an edge computing device can be any general purpose computer which have the following characteristics:

- Run an operating system (Windows or Linux) that allow the execution of original or light versions of complex tools like Tensorflow, Keras, OpenCV or OpenGL, which can be updated according to the application needs.
- Standard input-output interfaces like USB, and GPIO for direct interaction with the system sensors and actuators directly if required.
- Enough computing power to perform real-time execution of the enabling capabilities including image processing, machine learning, control or optimization

algorithms. The hardware requirements varies from application to application but an entry level specifications includes a multi-core ARM microprocessor with 1GB of Ram and of 16GB for storage.

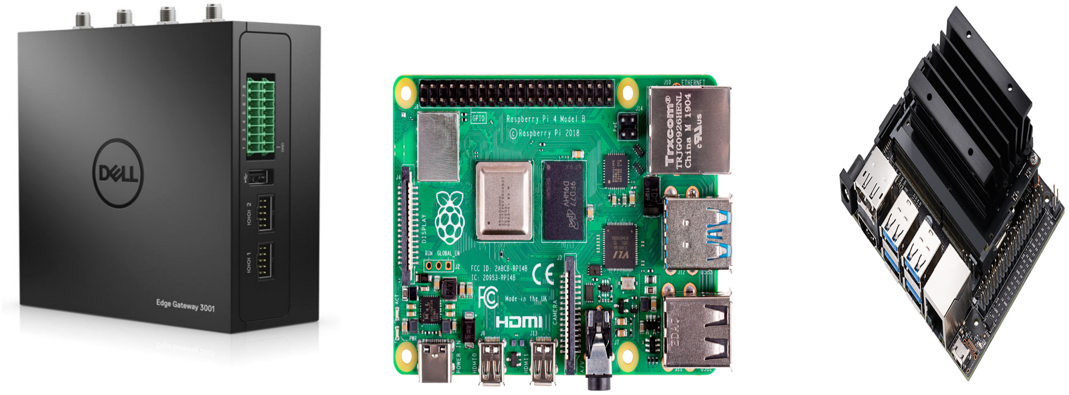
- The edge devices should have internet connection and interfaces compatible with the industrial communication standards like EtherCAT, Ethernet, ModBus, CANBus, TPC/IP, FTP or UDP communication protocols as well as wireless connections via bluetooth or Wifi (IEEE 802.11).

According to the features described above, the choices of edge computing devices are diverse and depend on the needs of each user. In the case of an industrial setup the Dell Edge Gateway series computers can be considered [259]. It contains a 9th generation Intel Atom microprocessor with 2GB RAM and up to 128 GB of storage executing Ubuntu server or Windows 10 IoT. It also includes RJ45 Ethernet communication, GPIO, Zigbee and 4G communication among other capabilities. The cost of this platform is around US $600 . Another option is the HPE Edgeline EL300 Mini-ITX [260], whose specifications are similar to the Dell Edge platform including a Core-i5 processor at a price of US $1700. Besides, another edge computer with the same processing and communication specs is the SuperServer E50-9AP or the UP Xtreme i11-0001 [261], in a price range from US $200 to $400.

Likewise, other less expensive Windows based alternatives for edge computing includes the Lattepanda board [222], which has specifications similar to the Dell Edge computer. It is a Intel atom microprocessor with 4GB of RAM and 64GB of storage running windows 10 with standard connectivity. Its price is around US $200 and supports all Windows based programs for web developing or modelling including Matlab. It was the solution used by the authors for the thermal system case study presented in the book.

Among the Linux edge computing devices several more options can be considered. The most popular one is the Raspberry Pi (RPI) in all its versions (1GHz ARM microprocessor and 1GB-4GB RAM) [262], which has several independent hardware that can be integrated to extend the RPI capabilities. Due to its low cost (US $30 to US $50) is a great option for academia and industry non-critical real-time applications. Likewise, there are other options like the Odroid [263], Beagle Bone or the Dev Board Ultra96. Another board that can be considered is the Nvidia Jetson in its full and Nano versions, which incorporates dedicated hardware resources for artificial intelligence and image processing [264]. A more detailed discussion about edge devices devices can be found at [265].

Some of the Edge computing devices listed are shown in Fig. 6.6. A common characteristic among these devices is the reduced size factor, designed for its integration into connection cabinets as well as a low power consumption. In the book support website `https://www.theedgeai.com/dtandscebook` a more detailed list of edge devices can be found with several tutorials for the implemented applications in the manuscript.



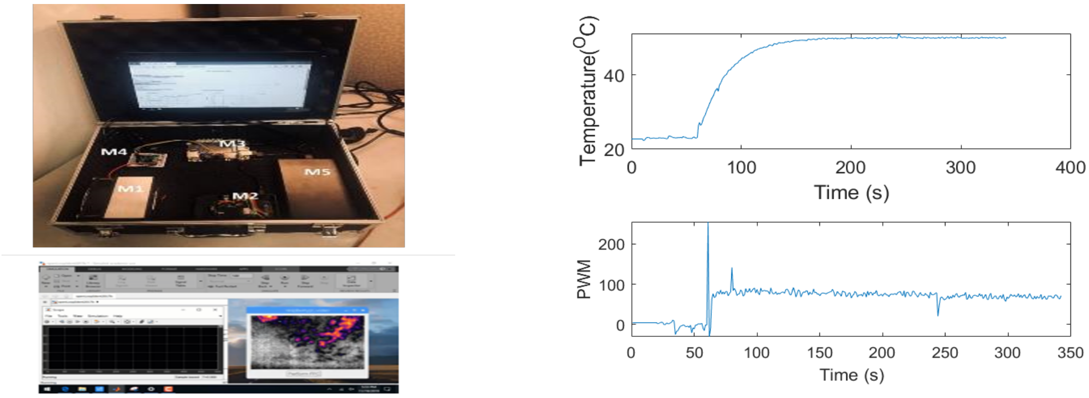
(a) (b) (c)

Figure 6.6: Edge Computing Devices a) Dell Edge Gateway 3001 b) Raspberry Pi 4 and c) Nvidia Jetson Nano

If the edge systems incorporates cloud computing capabilities, one alternative is using the lambda functions of Amazon web services (AWS), which are dedicated routines located in Amazon cloud that interacts with the edge device to perform long range analytics through the mosquito network communication protocol. In the book ”Digital Twin Development and Deployment on the Cloud” by Nassim Khaled [266], a detailed step by step tutorial on web applications using lambda functions for Digital Twins can be found for monitoring and analytics.

6.8 A Case Study: Fault Detection and Remaining Useful Life Analysis for Thermal Systems

The case study employed to show the application of enabled capabilities via Digital Twin is a Peltier thermoelectric system presented on Chapter 5, whose configuration and physical response are shown in Fig. 6.7. As can be observed, the output signal and control action of the system are contaminated with random noise due to the thermal infrared camera precision and accuracy. So, modeling the noise is relevant for the implementation of a fault detection strategy and increase the accuracy of the Digital Twin model.



(a)

(b)

Figure 6.7: Case Study: a) Thermal System, b) Temperature Sensor and Actuator Real Responses with Noise

Thus, the random noise of the feedback sensor is measured during the steady state operation of the system to fit its behavior according to a probability distribution. Notice that the thermal infrared camera precision and accuracy may change according to the measured temperature. So that, datastreams of random noise are acquired for setpoints from 30°C to 90°C. As an example, Fig. 6.8 shows the steady state temperature and control action of the system and the histogram of the estimated noise probability distributions for 50°C. In this case, a normal and α-stable distributions are employed to estimate the noise behavior. The normal distribution is defined by the mean μ and its standard deviation σ. Likewise, a random variable X can be considered α-stable if its characteristic function is given by (6.6), which is modeled by four parameters α, β, γ, δ [267] and [268]. The exponent α determines the thickness of the probability density function (PDF) tail, which relates to the spiky behavior of the random samples. It means that for larger values of α, the spiky behavior is reduced and is closer to a normal Gaussian distribution. The skewness factor β indicates if the distribution is skewed to the right or left tail if its value is positive or negative, respectively. The factor γ is related to the dispersion of the distribution. Finally, factor δ is the scale parameter and represents the mean or median of the entire distribution.

$$\varphi(v) = \exp\big\{j\delta\varphi - \gamma|v|^{\alpha}[1 + j\beta\text{sign}(v)\omega(v,\alpha)]\big\} \tag{6.6}$$

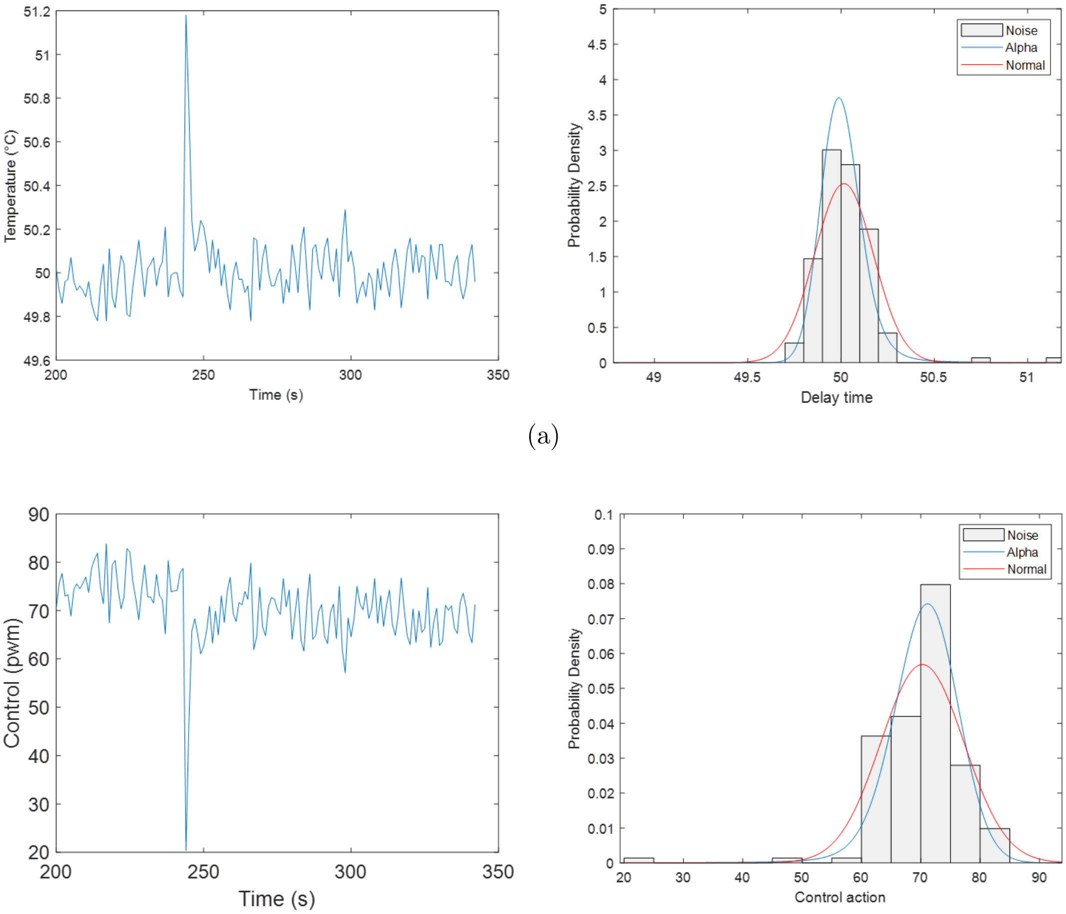
where:

$$\omega(v,\alpha) = \begin{cases} \tan\frac{\alpha\pi}{2} & \text{for } \ \alpha \neq 1 \\ \frac{2}{\pi}\log|v| & \text{for } \ \alpha = 1 \end{cases} \text{sign}(v) = \begin{cases} 1 & \text{for } \ \text{v} > 0 \\ 0 & \text{for } \ \text{v} = 0 \\ -1 & \text{for } \ \text{v} < 0 \end{cases}$$

and

$$0 < \alpha \leq 2, -1 \leq \beta \leq 1, \gamma > 0, -\infty < \delta < \infty.$$

After performing the noise histogram fitting, it can be appreciated that an alpha stable distribution can explain better the spiky noisy behavior of the system. Likewise, a Kolmogorov likelihood test is performed, which P value calculated for the system temperature and control action which are shown in Tables 6.1 and 6.2, confirming that the noise follows an α-stable distribution.



(a)

(b)

Figure 6.8: Steady State Noise Histogram for a) Temperature Signal and b) Control Action.

Table 6.1: Thermal Infrared Camera Noise Distribution

Setpoint	**Normal**			**α-stable**			P **value**
°C	μ	σ	α	β	γ	δ	
30	29.96	0.1549	2	0.15	0.105	29.96	0.9749
45	45.01	0.2615	1.7851	0.2297	0.0744	44.98	0.59
50	50	0.1576	1.83	1	0.0752	49.99	0.61
60	59.99	0.12	1.5102	0.827	0.0612	59.96	0.7014
90	90.01	0.0918	1.8780	0.1591	0.0549	90.01	0.2682

Table 6.2: Control Action Noise Modeling

Setpoint	**Normal**			**α-stable**			P **value**
°C	μ	σ	α	β	γ	δ	
30	29.96	0.1549	2	0.15	0.105	29.96	0.9749
45	45.01	0.2615	1.7851	0.2297	0.0744	44.98	0.59
50	50	0.1576	1.83	1	0.0752	49.99	0.61
60	59.99	0.12	1.5102	0.827	0.0612	59.96	0.7014
90	90.01	0.0918	1.8780	0.1591	0.0549	90.01	0.2682

Fault Generation on the Thermal System

Considering an improved knowledge of the random noise on the temperature and control action of the system, two faults are defined for the closed-loop configuration of the physical system as well as its Digital Twin. Those faults are shown in Fig. 6.9, which are induced on the actuator (Fault 1) and the sensor (Fault 2). For the fault 1, it is assumed as a critical failure considering that the power driver that manage the power on the Peltier system can fail due to a malfunction of any of its solid-state devices, producing a complete power shutdown on the Peltier. On the case of fault 2, it can be considered as a lost of communication between the thermal infrared camera and the main computer board where the control is performed due to a broken wireless communication or a byte communication error that fix a random value on the communication channel.

The response of the physical system and the Digital Twin to the induced faults is shown in Fig. 6.10. As can be observed, the actuator fault makes the temperature signal going down as well as the control action is saturated to its maximum, it indicates that there is not energy applied to the Peltier and the power driver is not able to provide it. In the case of the sensor malfunction, once an incorrect sensor value keeps registered, the control action also goes into saturation. The main difference with the actuator fault is the fact that sensor malfunction change is instantaneous compared with the actuator fault, which takes longer time to be detected.

Another important issue to consider is that as can be observed in Fig. 6.10 (a), there is a difference in the amplitude between the physical system and the Digital Twin fault response. For this reason, a fault matching criteria is established to determine the similarity between the Digital Twin and the physical system response using the normalized root mean square error (NRMSE) index given by (6.7) where x_{ref} is the real system response and x is the one for the Digital Twin. For the actuator fault the NRMSE index is 60% and for the sensor fault is 100%, indicating a reasonable behavior approximation to perform Fault detection .

$$NRMSE = \frac{||x_{ref} - x||}{||x_{ref} - mean(x_{ref})||}. \tag{6.7}$$

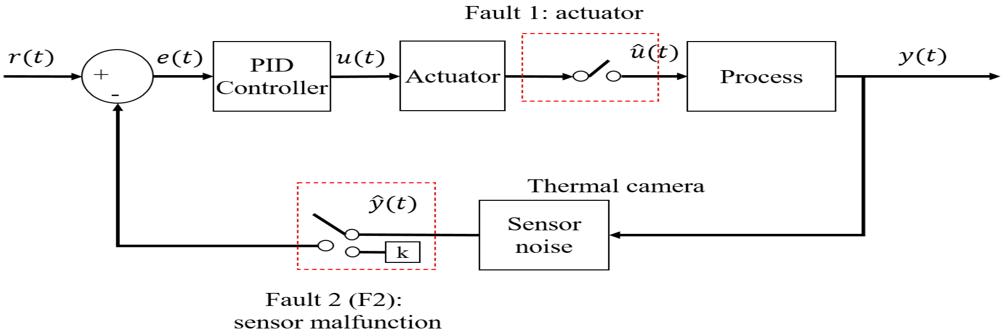


Figure 6.9: Sensor and Actuator Faults Applied in the Peltier System Closed-Loop Configuration

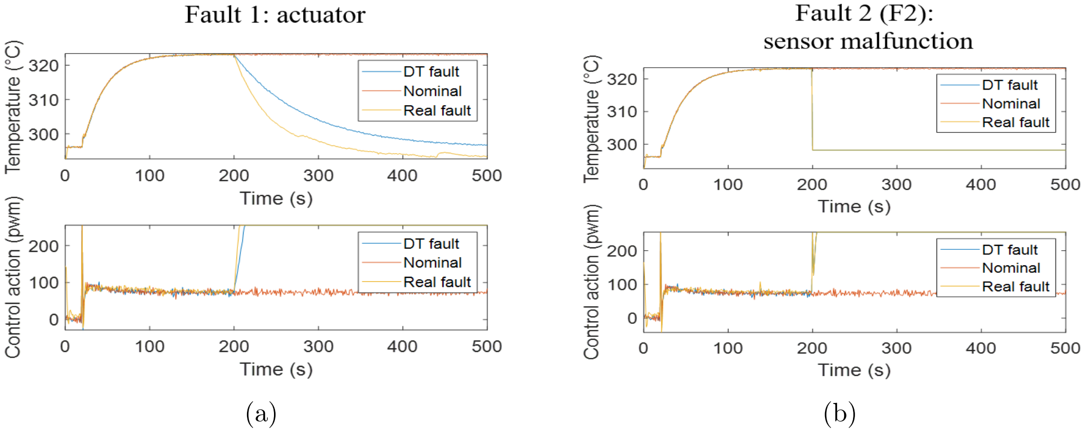
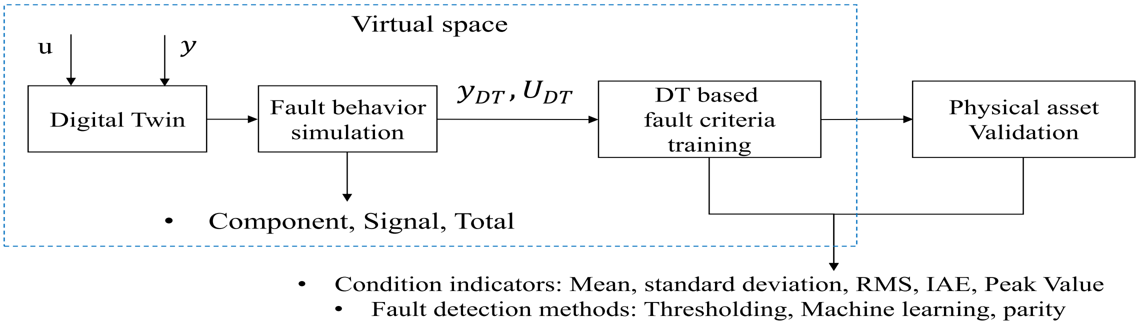


Figure 6.10: Induced Faults Response a) Actuator Failure and b) Sensor Malfunction

Fault Detection Using Digital Twin

Two Fault detection methodologies are implemented for the Peltier system. The first one corresponds to a statistical threshold method based on the statistics of the system and the second one employs artificial neural networks. The design and implementation of the fault detection methods is shown in Fig 6.11. As can be observed, the methodologies are tested first using the Digital Twin and then validated and implemented on the real system. In particular, for the neural network based fault detection, the neural networks training is performed using Digital Twin synthetic data and then implemented for real-time detection on the physical system. The proposed fault detection methods are designed to operate during the closed-loop steady state operation of the system, considering that many real systems produce faults during that operation stage.



Virtual space
u
y
Digital Twin
Fault behavior simulation
y_{DT}, U_{DT}
DT based fault criteria training
Physical asset Validation
• Component, Signal, Total
• Condition indicators: Mean, standard deviation, RMS, IAE, Peak Value
• Fault detection methods: Thresholding, Machine learning, parity

Figure 6.11: Fault Detection Methods Training and Calibration Using Digital Twin

Fault Detection Using Statistic Thresholding

The fault detection method using statistical thresholding is explained in Fig. 6.12. The method is designed to be executed in real-time with the system execution. Initially the values of temperature T, control action U and reference R of the system are acquired and stored in a sliding window of k samples length. Then according to the reference signal, the fault thresholds are defined for the temperature T_{up}, T_{low} and control action U_up, U_{low} as three times the sliding window standard deviation $\pm 3\sigma$ according to modelled noise level. Thus, the temperature signal is evaluated first. If the standard deviation of the temperature is outside the boundaries, the control action is evaluated to determine if the fault is due to the sensor or the actuator.

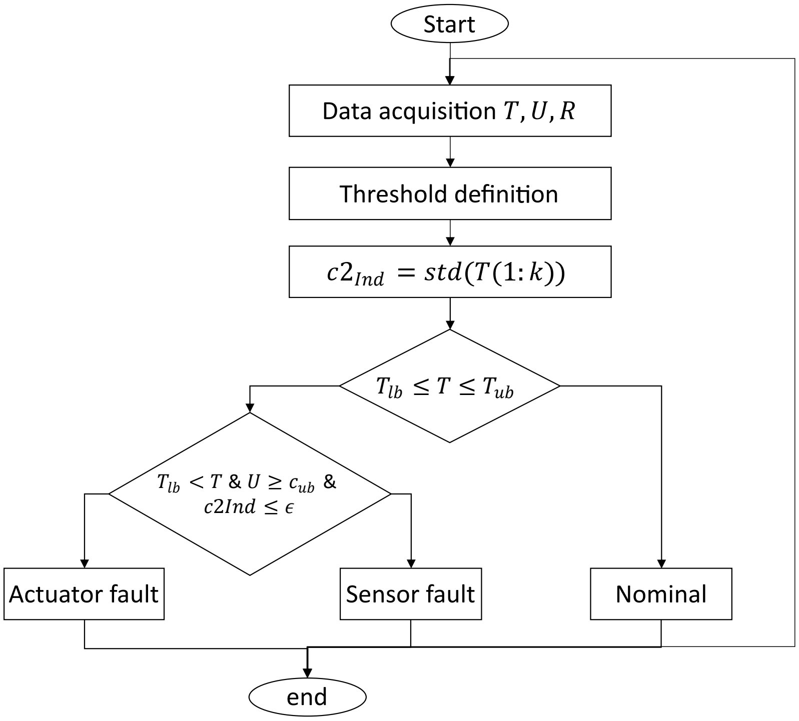
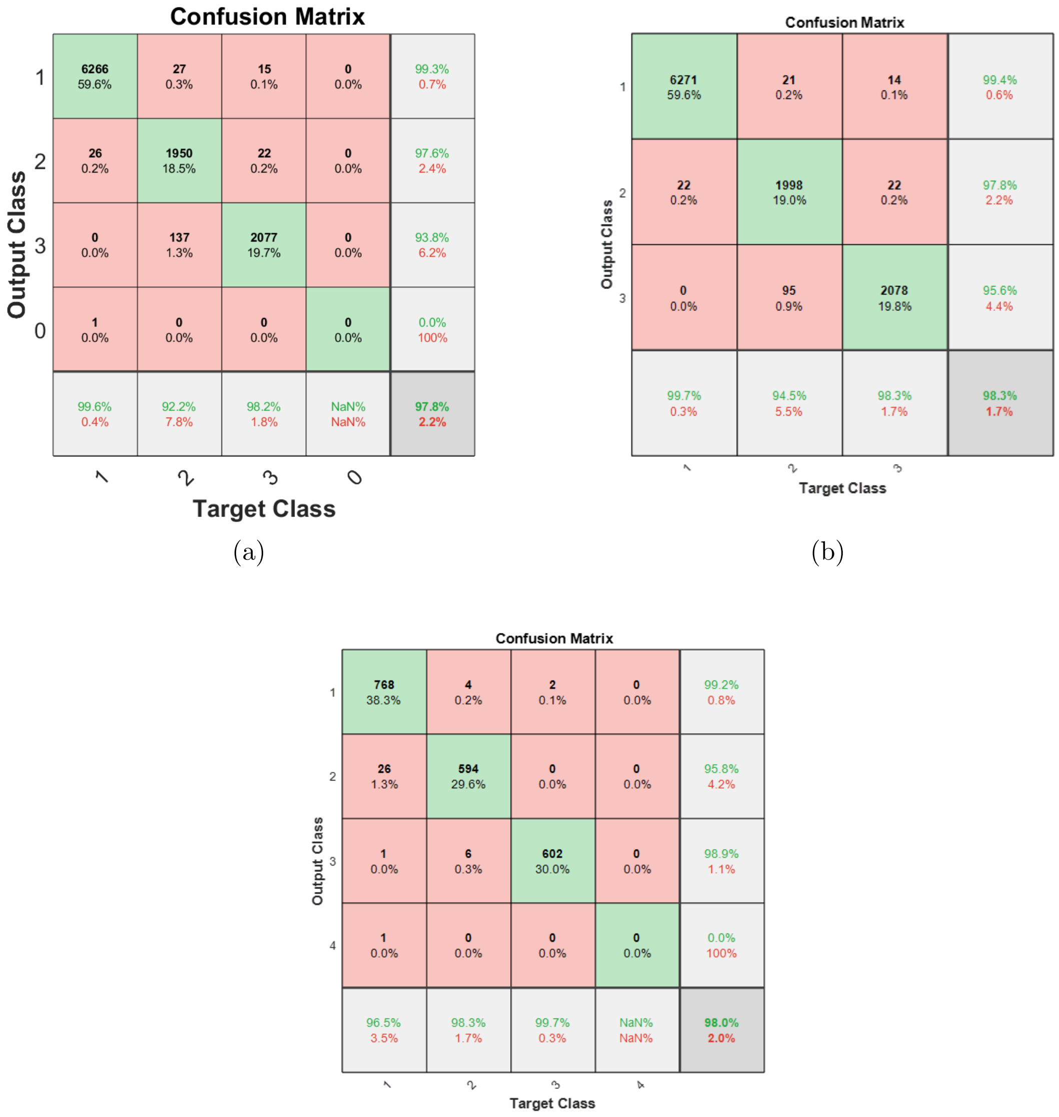

Start
Data acquisition T, U, R
Threshold definition
$c2_{Ind} = std(T(1{:}k))$
$T_{lb} \leq T \leq T_{ub}$
$T_{lb} < T$ & $U \geq c_{ub}$ & $c2Ind \leq \epsilon$
Actuator fault
Sensor fault
Nominal
end

Figure 6.12: Statistic Thresholding Fault Detection Method.

Fault Detection using Real-Time Multilayer Neural Networks

This fault detection method uses an artificial neural network to detect the current status of the system. It means, if the system is operating in nominal condition 1, or under sensor 2 or actuator fault 3. The neural network has ten hidden layers which inputs are the error, control action, output temperature and reference signal. The neural network is trained using synthetic data produced with the Digital Twin for the three possible scenarios, using 14 datasets for fault and seven for nominal behavior under different setpoint values from 30°C to 90°C. Likewise, 21 validation datasets are employed, and five experimental datasets from the fault behavior of the physical asset are used for cross-validation. The neural network is created and trained using Matlab Machine Learning Toolbox. The C code of the network is generated and runs simultaneously in real-time with the system and the Digital Twin as described in Chapter 5. The confusion matrix for the training and validation stages using synthetic dataset generated by the Digital Twin and the real fault data are shown in Fig. 6.13 where it can be appreciated that the algorithm has a 98% of accuracy in the three stages, indicating that using Digital Twin it is possible to obtain a high training accuracy and generalization of the fault behavior detection.



Confusion Matrix

	1	2	3	0	
1	6266 59.6%	27 0.3%	15 0.1%	0 0.0%	99.3% 0.7%
2	26 0.2%	1950 18.5%	22 0.2%	0 0.0%	97.6% 2.4%
3	0 0.0%	137 1.3%	2077 19.7%	0 0.0%	93.8% 6.2%
0	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0.0% 100%
	99.6% 0.4%	92.2% 7.8%	98.2% 1.8%	NaN% NaN%	97.8% 2.2%

Output Class

Target Class

(a)

Confusion Matrix

	1	2	3	
1	6271 59.6%	21 0.2%	14 0.1%	99.4% 0.6%
2	22 0.2%	1998 19.0%	22 0.2%	97.8% 2.2%
3	0 0.0%	95 0.9%	2078 19.8%	95.6% 4.4%
	99.7% 0.3%	94.5% 5.5%	98.3% 1.7%	98.3% 1.7%

Output Class

Target Class

(b)

Confusion Matrix

	1	2	3	4	
1	768 38.3%	4 0.2%	2 0.1%	0 0.0%	99.2% 0.8%
2	26 1.3%	594 29.6%	0 0.0%	0 0.0%	95.8% 4.2%
3	1 0.0%	6 0.3%	602 30.0%	0 0.0%	98.9% 1.1%
4	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0.0% 100%
	96.5% 3.5%	98.3% 1.7%	99.7% 0.3%	NaN% NaN%	98.0% 2.0%

Output Class

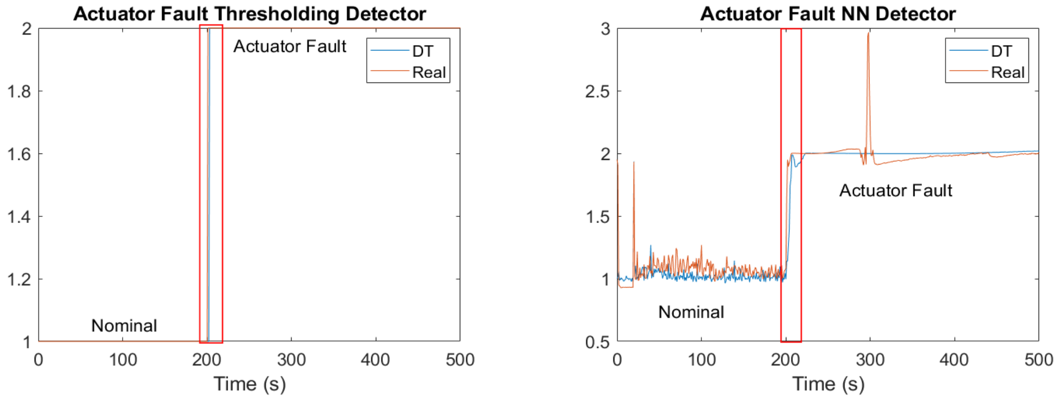
Target Class

(c)

Figure 6.13: Artificial Neural Network Fault Detector Confusion Matrices a) Training b) Validation (Synthetic Data) c) Cross Validation (Real Fault Dataset)

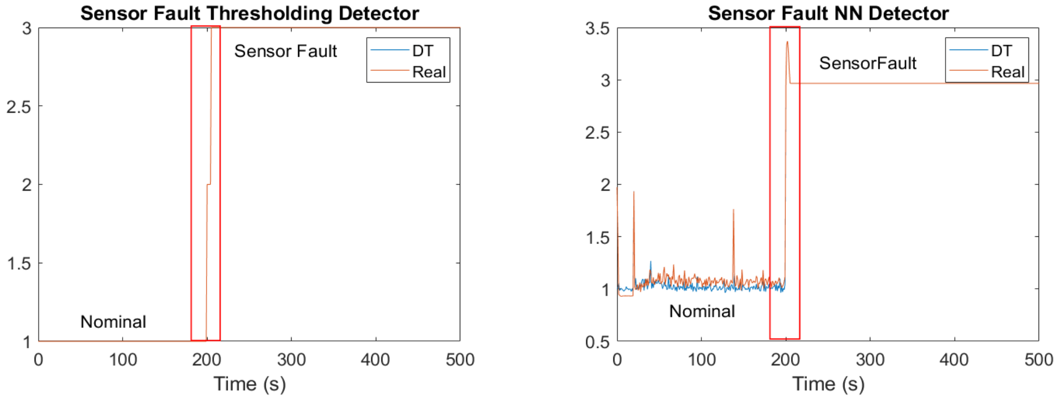
Fault Detection Obtained Results Under Real-Time Execution

The proposed fault detection methods are tested initially on the Digital Twin and then validated on the real system. Figures 6.14 and 6.15 show the performance of the statistical thresholding and neural network based fault detection techniques for the desired setpoint of 50°C. In this case, a discrete signal represents the current operation state of the system corresponding to nominal 1, actuator fault 2, or sensor fault 3. For each fault state, the fault is induced during the steady state operation at $t = 200s$. Figure 6.14(a) shows the actuator Fault detection using the thresholding method on the real system and the Digital Twin, showing an accurate detection of the fault near at the moment of the event occurrence. Likewise, Fig. 6.14(b) shows the actuator fault detection using the neural network approach. It can be observed that unlike the thresholding method, the state estimation of the system shows some variability due to the complex random noise of the signals in the DT and the real system. On the other hand the sensor fault of the system represented by Fig. 6.15 shows that the statistical thresholding and the neural network approaches perform a good real-time detection of the system nominal and fault states. So, it is possible to say that Digital Twin can enhance the accuracy and sensitivity of the fault detection methods serving as a platform for its training and validation before its implementation on the physical asset.



(a) (b)

Figure 6.14: Real-Time Detection of Actuator Fault using a) Statistical Thresholding and b) Neural Network methods



(a)

(b)

Figure 6.15: Real-Time Detection of Sensor Fault using a) Statistical Thresholding and b) Neural Network Methods

Remaining Useful Life Estimation Based on Digital Twin

Calculating the remaining useful life of a product or component may be a challenging task, considering that in many cases it requires a large dataset that register several time periods of the component operation, indicating how the system behavior is progressively approaches to the failure point. In the case of a Peltier thermoelectric system, its lifespan is bigger than other thermal devices due to its structure formed by solid state elements. As an example, Fig. 6.16 shows the useful life of a Peltier in terms of its thermal cycles [269], where one thermal cycle is considered when the Peltier is heated to 80°C and then cooled to 20°C. In the best scenario, the Peltier can reach up to 31000 thermal cycles and each thermal cycle with the proper closed-loop control may take about 10 minutes. Therefore, a complete real experiment can take around eight months to get the required information to perform the experiment.

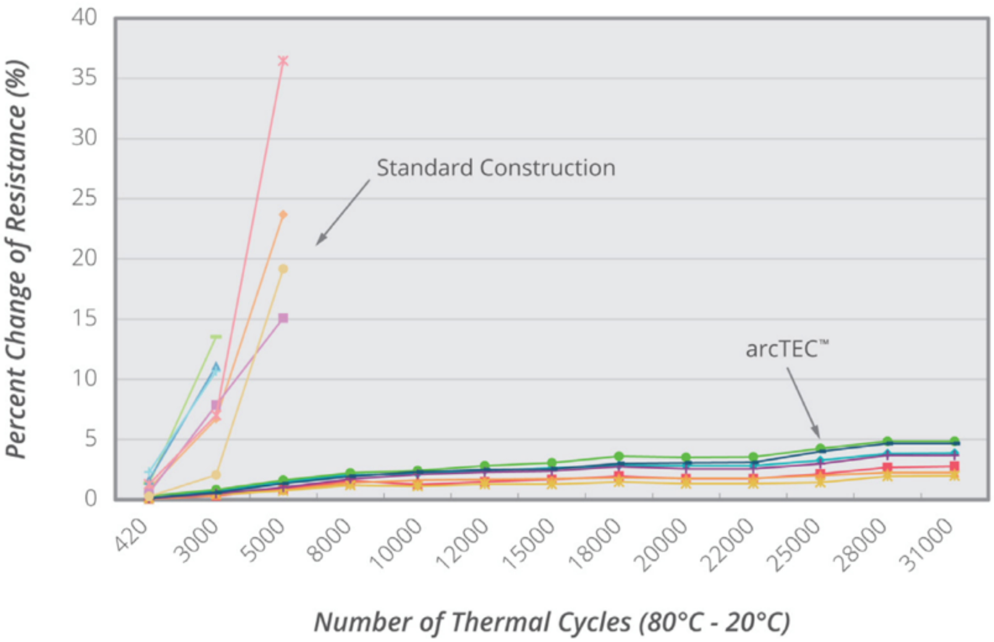


Figure 6.16: Electrical Resistance Variation Over Heating Cycle Batches of a Thermoelectric Module [269]

In the case of the product manufacturer, these long testing times are required to ensure its product quality. However, with the increasing product demand this time can represent a delay for its product release. Also, if the component is located inside a more complex system, there is not any chance to remove the component and evaluate its useful life individually without considering the other system components. In that sense, Digital Twin can help to obtain an estimation of the remaining useful life of a component inside a complex system using the available information and also generating synthetic data for some extreme conditions to predict the system remaining useful life.

Thus, using the thermal system Digital Twin, multiple scenarios of electrical resistance degradation are simulated as shown in Fig. 6.17. In this case the degradation level increases at an exponential rate, adding an specific amount of resistance on the simulation per heating cycle batch, corresponding to 500 Peltier heating cycles for each setpoint (30°C to 90°C). According to the amount of resistance variation, the closed-loop response of the thermal system is degraded.

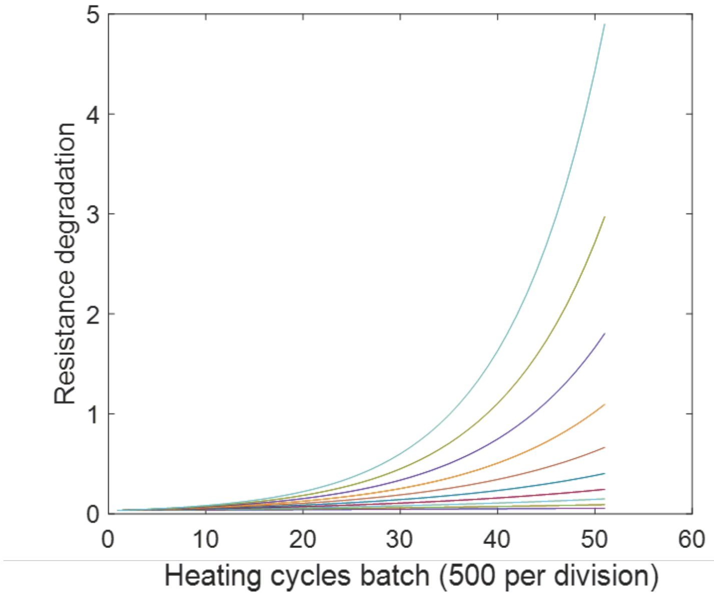


Figure 6.17: Electrical Resistance Variation Regarding to the Nominal Value $R = 3.3\Omega$

Based on the system Digital Twin and the available information of the physical system, six health indicator candidates are selected to determine the Remaining Useful Life of the Peltier, which are the temperature mean and standard deviation, the control action mean and standard deviation, and the temperature and control signals RMS values for these six health indicator candidates are shown in Fig. 6.18. For each health indicator, the monotonicity, prognosability, and trendability are calculated to determine which is the most relevant variable for the remaining useful life estimation. Figure 6.19 shows the calculation of these three indices for each health indicator using the synthetic data produced by the Digital Twin. As can be observed, all the health indicators have a good monotonicity and prognosability metric. However, the control action standard deviation and RMS value exhibit the best trendability among the health indicators. Therefore, the RUL calculation can be performed using these health indices.

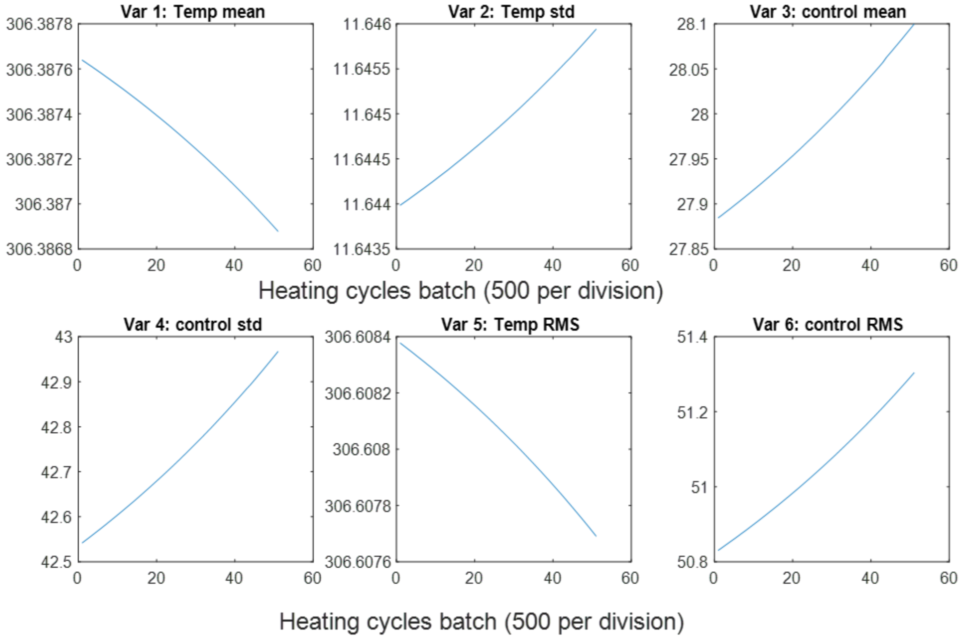


Figure 6.18: Health Indicators Calculated Using the Digital Twin

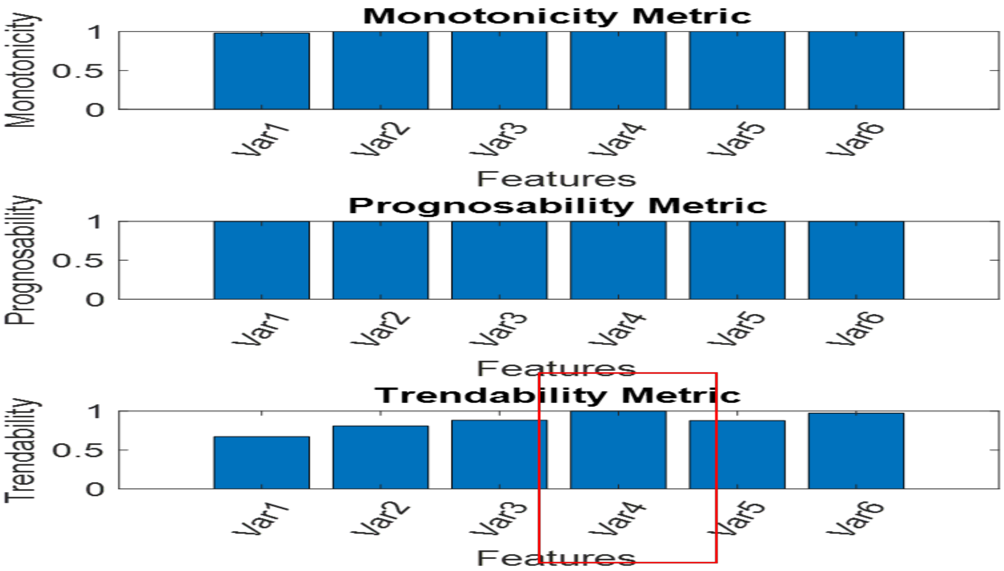


Figure 6.19: Monotonicity, Prognosability, and Trendability of the Proposed Health Indicators

Finally, the remaining useful life of the Peltier system is calculated using a linear and exponential models given by (6.8) and (6.9) respectively using the Matlab Predictive Maintenance Toolbox, where $\epsilon(t)$ corresponds to a stochastic term given by a normalized Gaussian distribution. The resulting RUL of the Peltier system estimated using the Digital Twin is given by Fig. 6.20. As can be observed, the exponential model predicts a 34 heating batch cycles of RUL for the system while the linear model is 33 cycles.

$$S(t) = 42.5173 + 0.0085t + \epsilon(t), \tag{6.8}$$

$$S(t) = 41.8794 + 0.6561e^{0.0099t+\epsilon(t)-2*10^{-32}}. \tag{6.9}$$

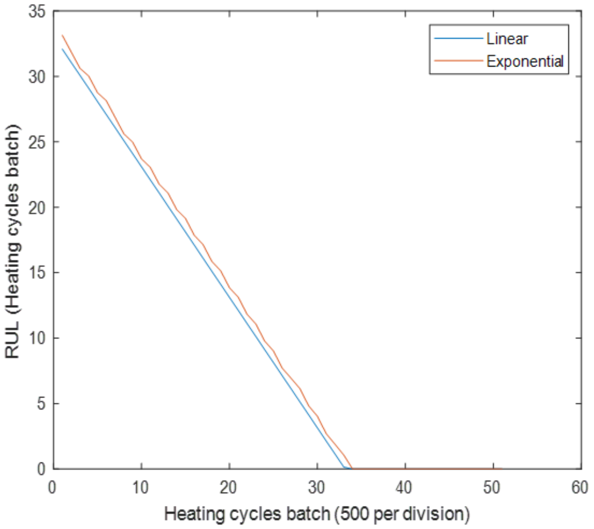


Figure 6.20: Faults Applied in The Closed-Loop Configuration

6.9 Chapter Summary

This chapter presents a review of the enabling capabilities obtained by the use of Digital Twin, including the smart control system design, self optimizing control, fault detection, prognosis, outlier detection, control performance assessment among others. A case study for the application of the enabling capabilities is presented by performing real-time Fault detection and remaining useful life analysis of the Peltier system supported by the Digital Twin. Obtained results show that Digital Twin can leverage these capabilities on the real system to obtain new analytics and insights of the system. Thus, Digital Twin can be considered as an option to accelerate product development and increase its reliability under multiple failure scenarios.

Chapter 7

SMART CONTROL ENGINEERING ENABLED BY DIGITAL TWIN AND SELF OPTIMIZING CONTROL

7.1 Introduction

According to the definition of a smart system, two of the main characteristics that defines any system as smart are the capability of being cognizant and reflective. It means, a smart system should be aware of the process current performance and being able to learn from past experiences to take the appropriated actions to keep the system operating on the desired conditions in the presence of system changes or external disturbances. Thus, Digital Twin and self optimizing control can be leveraged as enablers for the cognizant and reflective characteristics of a smart system.

This chapter presents the development and evaluation of a Self Optimizing Control framework based on derivative-free optimization algorithms for the performance improvement of a stable closed-loop system by adjusting the parameter of the closed-loop controller according to a performance cost function. The Globalized Constrained Nelder-Mead (GCNM) algorithm is proposed as an optimization method for the SOC Controller, a gradient-free technique. A simulation benchmark is designed for the performance assessment of the SOC controller, which employs a First Order Plus Dead Time (FOPDT) system with a closed-loop stable PI controller. Likewise, a convergence analysis of the SOC controller with the GCNM algorithm is performed using probabilistic theory and ”frame based method”.

Likewise, the integration of SOC with Digital Twin is presented for two cases study: the Peltier thermoelectric system presented in Chapter 6 and a mechatronic system. In this cases, the SOC performance is evaluated in the presence of external disturbances and parametric uncertainty on the Digital Twin.

7.2 SOC Control Framework Architecture

The proposed real-time self optimizing control architecture is shown in Fig. 7.1. As can be observed the self optimizing control acts as an upper optimization layer which takes the reference r, error signal e, and output y of the system to find the optimal values of the controller $c(s)$, corresponding to a PID controller (7.1) for the system $p(s)$, modelled as a first order plus dead time system (FOPDT) given by (7.2), where K,τ, and L correspond to the system gain, time constant and delay respectively.

$$c(s) = k_p + \frac{k_i}{s} + k_d s, \tag{7.1}$$

$$p(s) = \frac{K}{\tau s + 1} e^{-Ls}. \tag{7.2}$$

The SOC control objective is stated by (7.3) and (7.4), where T_s is the system sampling time, OV is the overshoot percentage, $\mu = k_p, k_i, k_d$ are the proportional, integral and derivative gains of the PID controller, A and B correspond to the maximum overshoot and settling time, $k_{p_{min,max}}, k_{i_{min,max}}, K_{d_{min,max}}$ are the minimum and maximum values for the PID gains, and W_1, W_2 and W_3 are the weights for the Overshoot, Settling time and and the integral square error index index respectively. In this case, the SOC controller executes an optimization step after one cycle of the periodic reference signal r, computing a performance cost function for that period.

$$\min_{\mu \in \mathbb{R}} J = W_1\ OV(\mu) + W_2\ T_s(\mu) + W_3 \int_0^t e^2(t,\mu)\,dt, \tag{7.3}$$

subject to the constraints

$$\begin{aligned} OV(\mu) < A;\ \ T_s(\mu) < B, \\ k_{p_{min}} \le k_p \le k_{p_{max}}, \\ k_{i_{min}} \le k_i \le k_{i_{max}}, \\ k_{d_{min}} \le k_d \le k_{d_{max}}. \end{aligned} \tag{7.4}$$

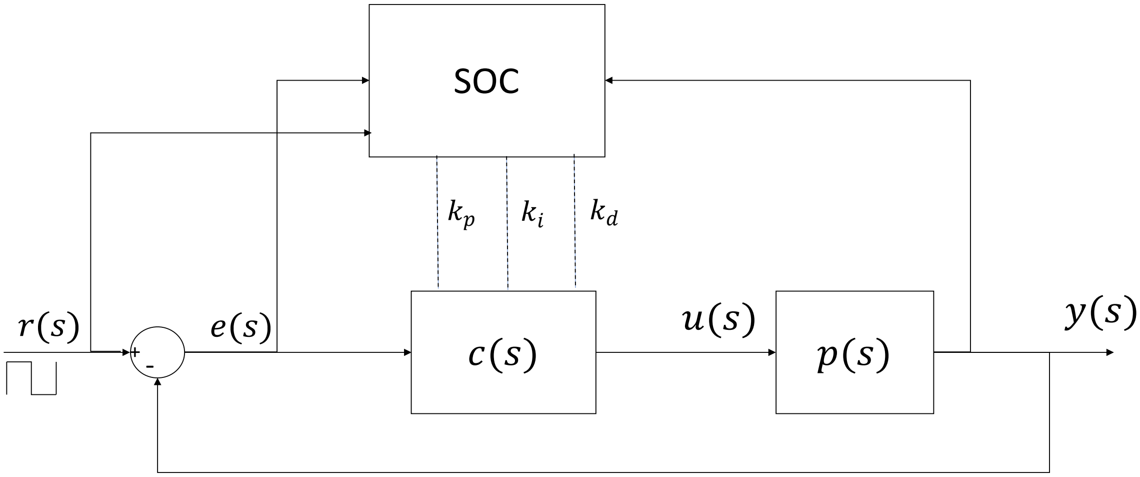


Figure 7.1: Proposed Self Optimizing Control Architecture

7.3 Globalized Constraint Nelder-Mead Algorithm

Nelder-Mead (NM) algorithm [270] is a gradient free optimization method widely employed for solving unconstrained convex optimization problems. It becomes a popular option for optimization problems when the model of the system is difficult to be used to calculate a reliable gradient. Also, its simple structure allows a straightforward implementation. Different applications of NM can be found in the literature [271–276], where many of these applications perform offline optimization only.

However, the unconstrained nature of the NM algorithm makes the technique susceptible to falling in local minimum specially for non-convex optimization problems. For this reason, this chapter presents a modified Nelder-Mead Algorithm that searches for the global optimum based on a set of constrains called Globalized Constrained Nelder-Mead (GCNM) searching algorithm. Its flowchart is shown in Fig. 7.2. As can be observed, the GCNM method employs the same operations performed by the original NM algorithm: evaluation, reflection, contraction, expansion, and shirking, to create the simplex shape consisting of $n+1$ vertices where n is the number of parameters of the optimization. The operations of reflection, contraction, expansion in the GCNM algorithm are associated to a set of constants $\alpha, \beta, \gamma, \delta$ respectively, which are selected as $\alpha = 1$, $\beta = 1 + 2n_r\gamma = 0.75 - 0.5n_r, \delta = 1 - n_r$ with $n_r = 1/n$ as the reciprocal of the number of dimensions following the adaptive rule for NM algorithm proposed by [277].

The main difference between the classic Nelder-Mead and the GCNM algorithm is that the GCNM introduces a probabilistic restart mechanism that reinitializes the search from a different random initial condition if the final solution of the NM search does not satisfy the constraints, preventing to fall into a local minimum. Initially, the probabilistic restart takes action if the cost function (7.3) reaches a steady value. For this, the algorithm evaluates if the standard deviation of the last m values of the simplex centroid is below a threshold ϵ. If this is true, then the optimization is in the steady state and the constraints are evaluated. If at least one of the constraints are not satisfied, the GCNM restarts the searching on a new random point assigning a new set of initial conditions in the parameter space defined by the problem.

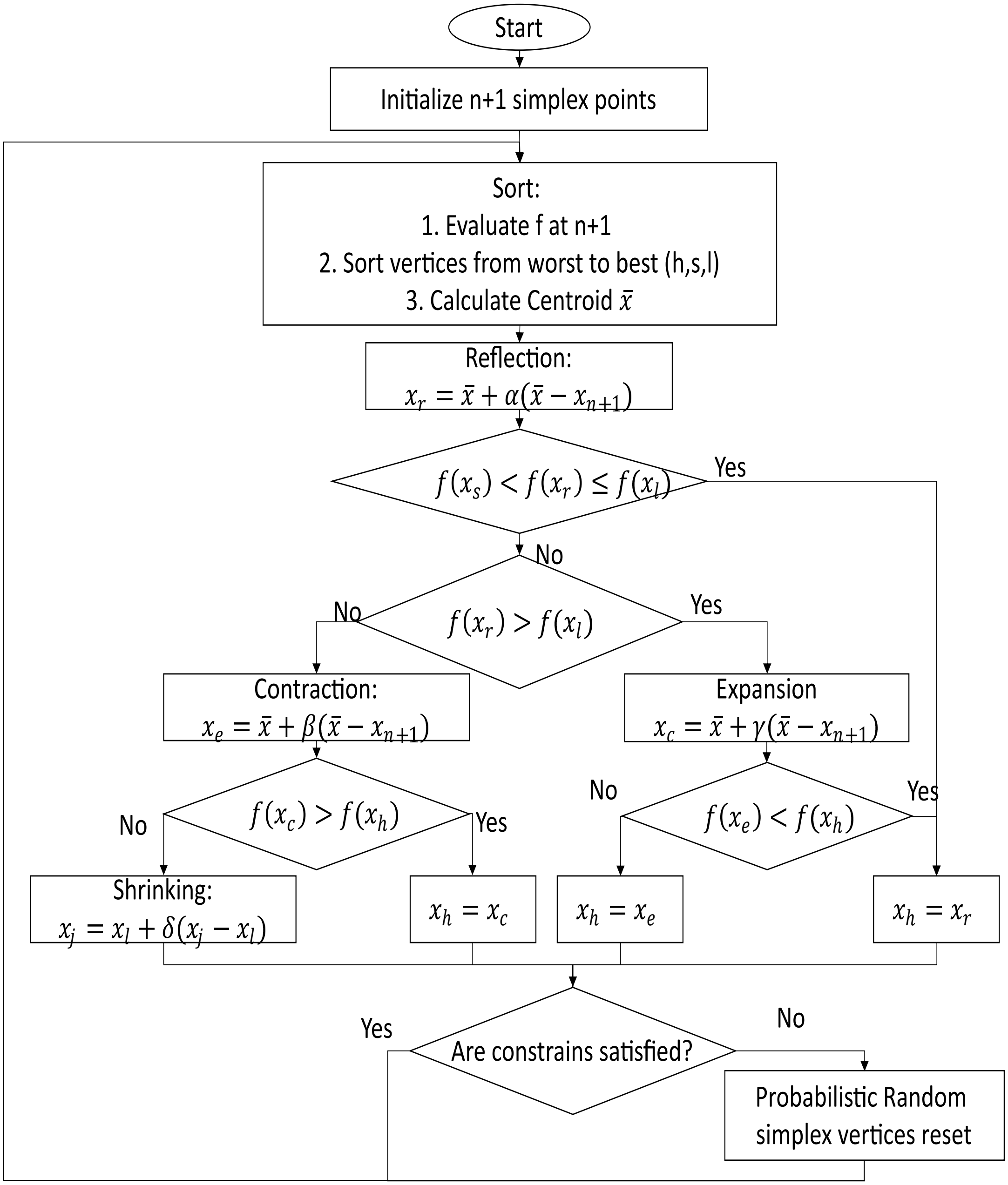

Start
Initialize n+1 simplex points
Sort:
1. Evaluate f at n+1
2. Sort vertices from worst to best (h,s,l)
3. Calculate Centroid $\bar{x}$
Reflection:
$x_r = \bar{x} + \alpha(\bar{x} - x_{n+1})$
$f(x_s) < f(x_r) \leq f(x_l)$
Yes
No
$f(x_r) > f(x_l)$
No
Yes
Contraction:
$x_e = \bar{x} + \beta(\bar{x} - x_{n+1})$
Expansion
$x_c = \bar{x} + \gamma(\bar{x} - x_{n+1})$
No
$f(x_c) > f(x_h)$
Yes
No
$f(x_e) < f(x_h)$
Yes
Shrinking:
$x_j = x_l + \delta(x_j - x_l)$
$x_h = x_c$
$x_h = x_e$
$x_h = x_r$
Yes
Are constrains satisfied?
No
Probabilistic Random simplex vertices reset

Figure 7.2: Globalized Constrained Nelder-Mead Algorithm Flowchart

One of the main advantages of the GCNM algorithm is that as a derivative free algorithm, it can be used for any system without prior knowledge of its dynamical model. Likewise, considering the sequential structure of the GCNM algorithm and the low computational complexity of the algorithm, it can be adapted for real-time execution. An example of the real-time execution of the GCNM algorithm running in an embedded hardware solving the sphere function optimization problem in real-time can be found in `https://youtu.be/xA4nO9M4Qqg`.

7.4 Self Optimizing Control Based on Globalized Constrained Nelder-Mead Test Benchmark

This case study presents details on how to develop and evaluate of a self optimizing control strategy based on derivative free optimization algorithms for the performance improvement of a stable closed-loop system by adjusting the parameter of the closed-loop controller according to a performance cost function. The Globalized Constrained Nelder-Mead (GCNM) algorithm is proposed as the optimization method for the SOC Controller, which is a gradient free technique. A simulation benchmark is proposed for the performance assessment of the SOC controller, which employs a first order plus dead time (FOPDT) system with a PI controller which is closed-loop stable. The initial conditions for the PI Controller parameters are obtained by Ziegler Nichols method [278]. The SOC controller is evaluated on three scenarios for the FOPDT: a time constant dominated, balanced, and a delay dominated system, with a periodic reference signal.

A Matlab/Simulink benchmark was built to evaluate the performance of the GCNM SOC controller which is shown in Fig. 7.3. The benchmark is composed of a GCNM controller on top, the FOPDT system on the bottom and the classic NM algorithm in the middle that can be enabled by the user for testing. It follows the SOC controller configuration presented on Fig. 7.1 and (7.3)-(7.4). In this case, a PI controller is employed to ensure a stable closed-loop behavior for an FOPDT system.

The Matlab/Simulink implementation of the GCNM algorithms is shown in Fig. 7.4. It consists of two blocks, one for the cost function computation, and the other for the GCNM algorithm. The inputs of the block are the reference signal r, error signal e, system output y, the constrains values for the overshoot, settling time, the PI controller gains as stated in (7.3)-(7.4), and the initial conditions for the search. The outputs are the PI controller gains adjusted by the GCNM algorithm after a period of the reference signal r.

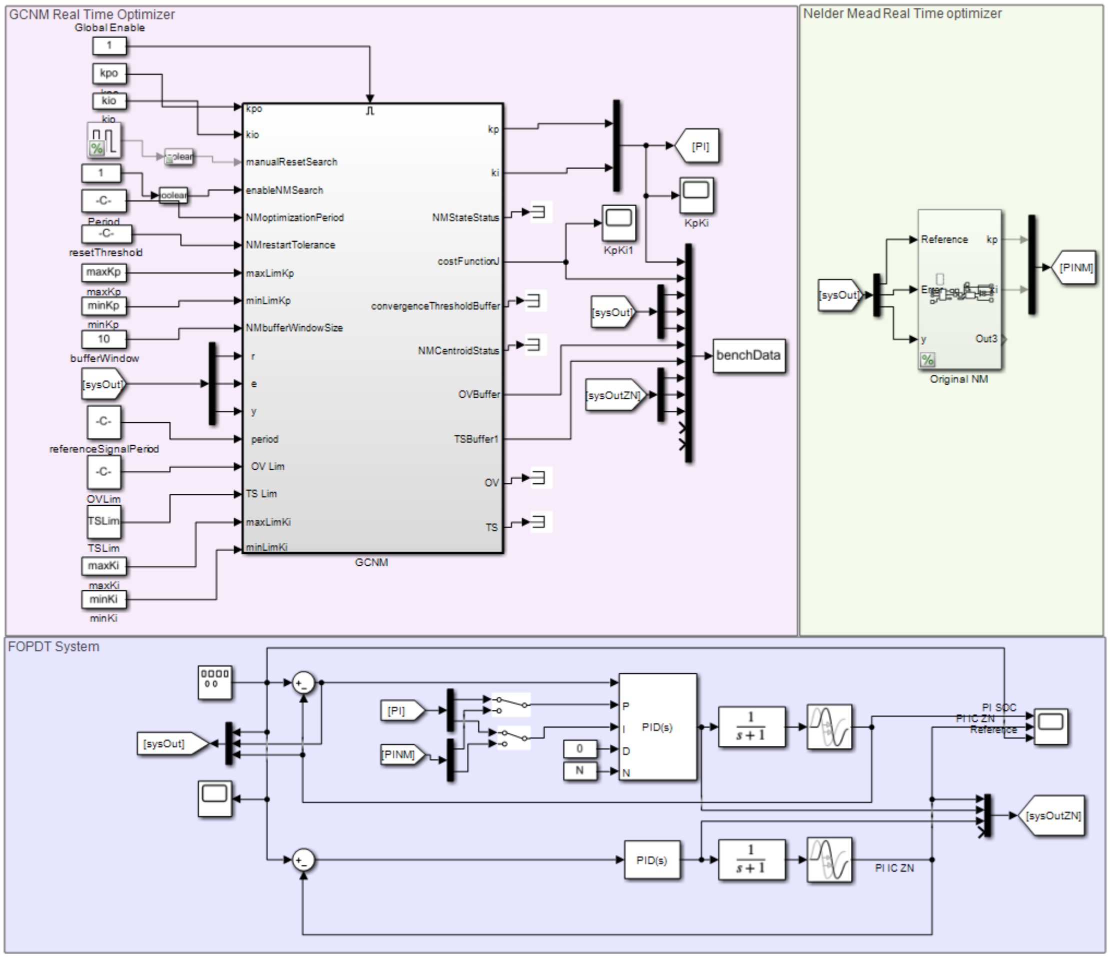


Figure 7.3: SOC Benchmark in Matlab/Simulink for FOPDT System

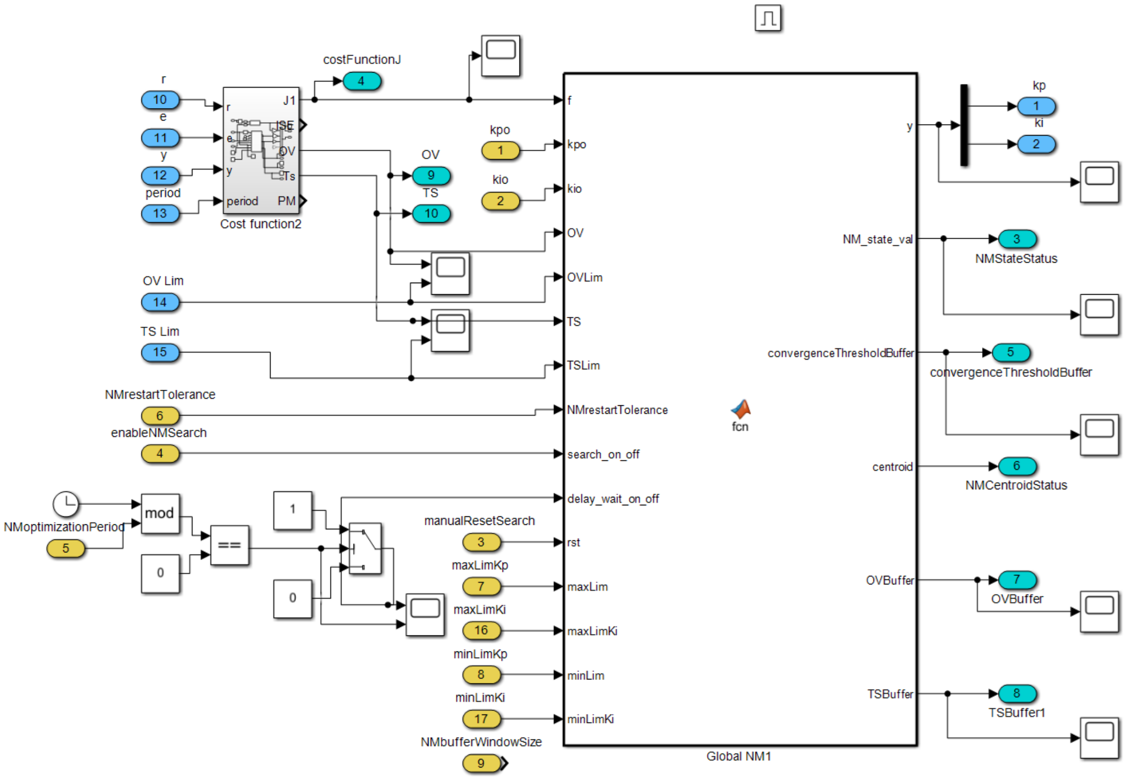


Figure 7.4: Globalized Constrained Nelder-Mead Algorithm Implementation in Simulink

7.4.1 FOPDT Model Generalization

Considering that the behavior of an FOPDT system (7.2) can be classified as time constant dominated ($\tau > L$), balanced time ($\tau = L$) and delay time dominated ($\tau < L$), the transfer function of the system can be normalized according to the Remark 11.3.1 presented in [279] by using (7.6).

Remark 11.3.1 1 *[279] The FOPDT system* (7.2) *can be normalized as follows:*

$$P_n(s) = \frac{1}{\tau s + 1} e^{-L/\tau \tau s} = \frac{1}{s' + 1} e^{-L' s'}, \tag{7.5}$$

where $s' = \tau s$ and $L' = L/T$. The parameter K in (7.2) *can be normalized to 1, since the steady-state gain of the plant can always be used as part of the gain of the PID controller. So, the normalized standard form of the control system plant can be written as* (7.6)*, where L is equal to L' in* (7.5)*.*

$$P_o(s) = \frac{1}{s + 1} e^{-Ls}. \tag{7.6}$$

Thus, based on the normalized FOPDT model (7.6), three tests will be performed using the benchmark for $L = 0.1,\ 1,\ 10$, which represent the time constant dominated, balanced time, and delay time dominated behaviors of the FOPDT system. For each scenario, two PID controllers are calculated using the Ziegler Nichols (ZN) and modified Ziegler Nichols (ZNM) techniques [278], We use the result of the ZN method as the initial condition for the GCNM algorithm to evaluate how much the SOC is able to improve the closed-loop system performance.

A square wave signal with equal high and low state time is employed as the reference with a period of 300s for all the tests. The constraint values and other configuration parameters for the benchmark are shown in Table 7.1. Notice that the maximum and minimum constraint values of K_p and K_i change from each test to ensure the stability of the closed-loop system. For all the benchmark tests, the cost function weights are $W_1 = 5,\ W_2 = 0.1,\ W_3 = 0.1$.

Table 7.1: SOC Benchmark Configuration Parameters

	Test 1 ($L = 0.1$)		Test 2 ($L = 1$)		Test 3 ($L = 10$)	
Parameter	min	max	min	max	min	max
k_p	0.01	10	0.01	1	0.01	0.1
k_i	0.01	5	0.01	0.5	0.01	0.1
Reset buffer samples			10			
Settling time	30 s		30 s		70 s	
OV (%)	1%		1%		5%	
Reset threshold ϵ			1e-3			
Reference signal period (s)			300			

7.4.2 Results and Discussions

The SOC controller is tested for the three scenarios defined in the previous section and the obtained PI Controller parameters using the ZN, ZNM, and SOC controllers are compared in Table 7.2. Notice that in the case of SOC, the initial condition for the PI controller is picked as the values obtained by the ZN method in order to evaluate how the SOC controller improve the baseline control performance provided by ZN. Thus, the values registered on the SOC row are the final value obtained after the SOC process is complete. A deeper analysis of the SOC benchmark results is performed in [280].

Table 7.2: Obtained PI Controller Gains Using ZN, ZNM, and SOC

Controller	Test 1 $(L = 0.1)$		Test 2 $(L = 1)$		Test 3 $(L = 10)$	
	K_p	K_i	K_p	K_i	K_p	K_i
ZN	9	27	0.9	0.27	0.09	0.0027
ZNM	12	60	1.2	0.6	0.12	0.006
SOC	2.68	2.63	0.44	0.42	0.087	0.046

7.4.2.1 Time Constant (lag) Dominated System $L = 0.1$

The time response of the SOC controller for the FOPDT system with $L = 0.1$ is shown in Fig. 7.5. As can be observed, the SOC controller solves the optimal control problem (7.3) satisfying the overshoot and settling time specifications with a convergence time closer to 20000s. Notice that for each 300s periodic cycle of the reference signal one candidate solution is evaluated, which is provided by the GCNM algorithm, explaining the longer convergence time. Likewise, the plots in the second row shows how the SOC controller evolves online the system response until the desired performance is reached We can see a high OV at the first stages but almost no OV at the end. Also, the evolution of the cost function (7.3), the overshoot, settling time, PI controller gains for the closed-loop system are showing in Fig. 7.6. It can be observed that the GCNM algorithm perform multiple probabilistic restart, starting the search from a new initial conditions for K_p, K_i to ensure the closed-loop performance specifications of the system.

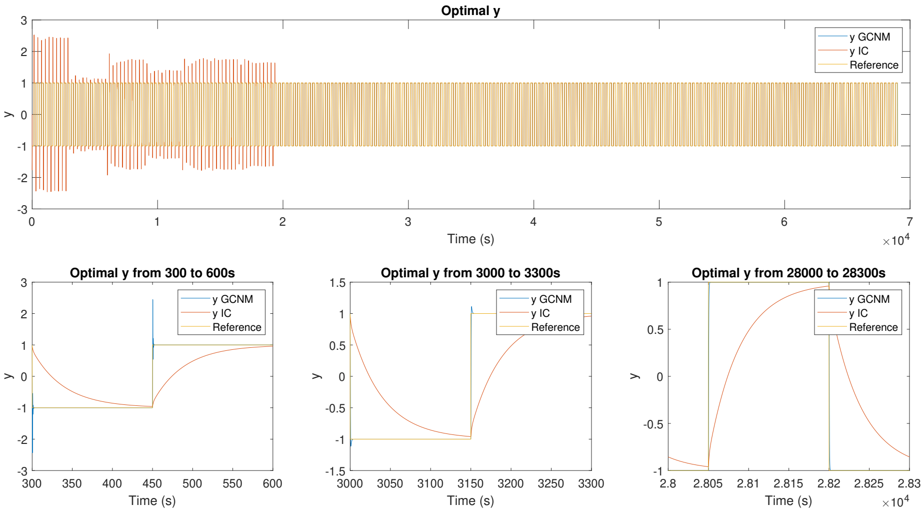
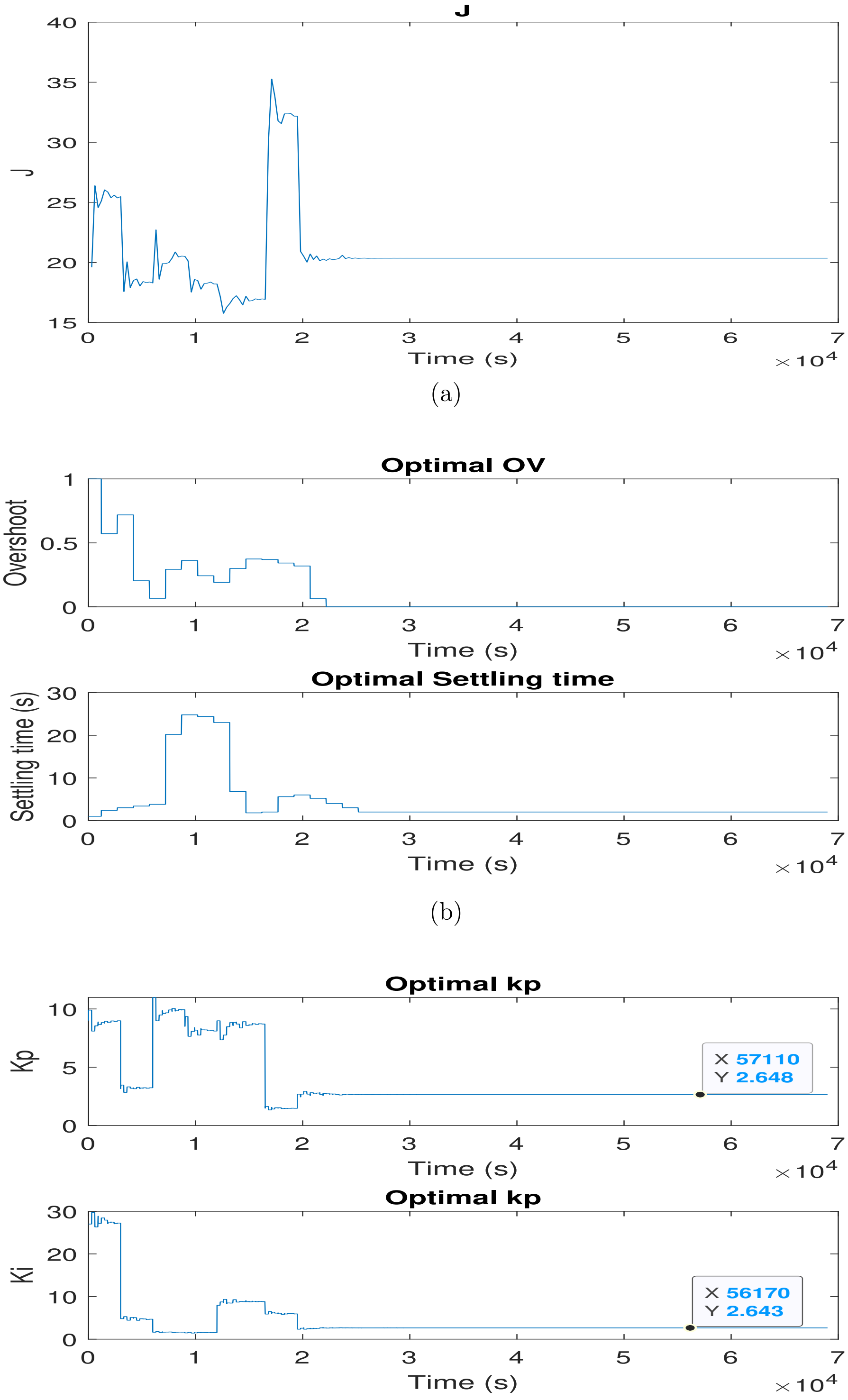


Figure 7.5: Closed-Loop System Output Evolution Using SOC Controller $L = 0.1$



(a)

(b)

(c)

Figure 7.6: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution for $L = 0.1$

7.4.2.2 Balanced System $L = 1$

The results of the benchmark for the balanced system are shown in Figs. 7.7 and 7.8. It can be appreciated that the performance of the SOC search ensures the desired operating conditions of 1% overshoot and settling time $T_s < 30s$. In this case, the convergence time is close to 10000 seconds and a multiple reset events can be appreciated. Due to the random reset feature, the convergence time may change from execution to execution. Also, the gains of the PI Controllers evolve such that the system keeps its stable operation. Notice that for this test, the searching range of K_p, K_i is reduced in order to ensure the closed-loop system stability.

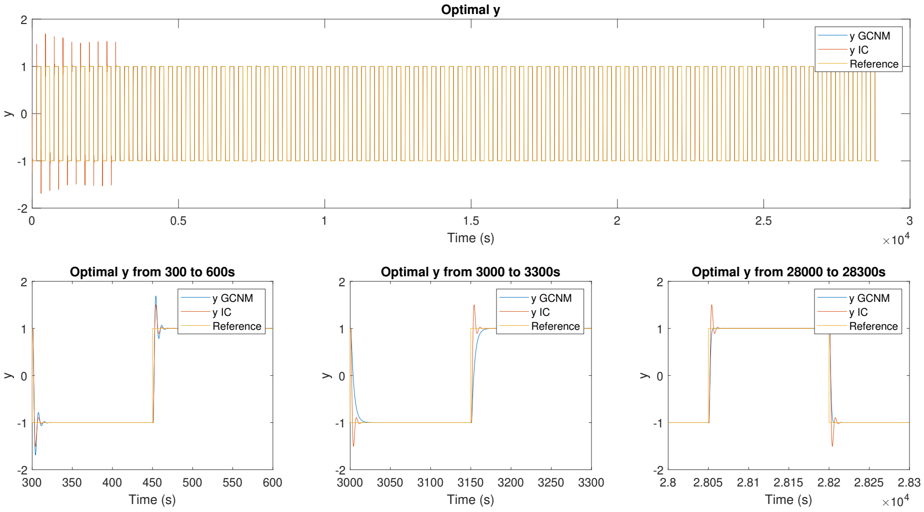
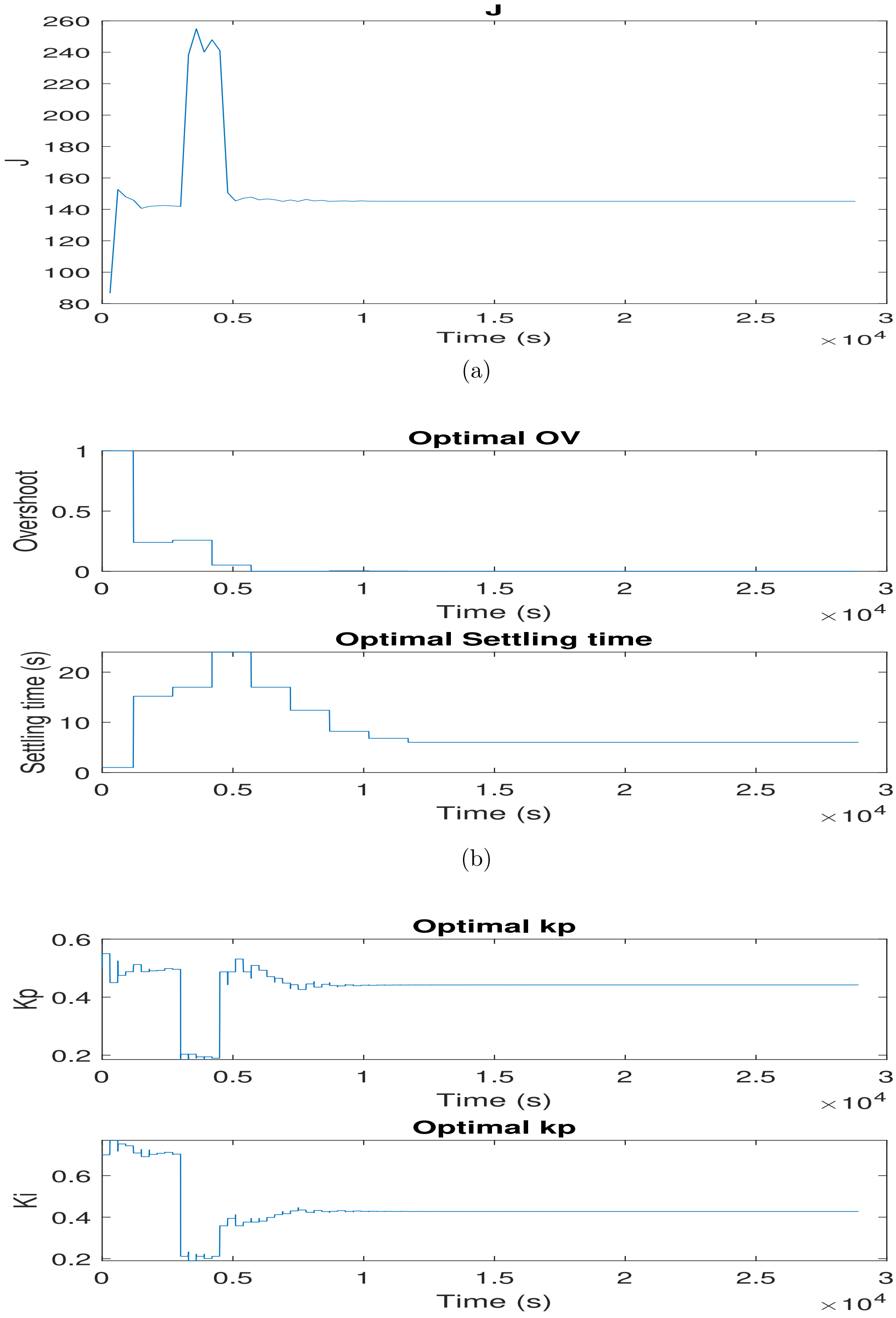


Figure 7.7: Closed-Loop System Output Evolution Using SOC Controller $L = 1$



(a)

(b)

(c)

Figure 7.8: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution for $L = 1$

7.4.2.3 Lag Dominated System $L = 10$

The results of the benchmark for the lag dominated system are shown in Figs. 7.9 and 7.10. In this case, the performance of the SOC search is ensured for a 1% overshoot and settling time $T_s < 70s$. The last one is seen bigger than in previous test due to the increased delay of the system. It is important to remark that the delay dominated system exhibits more tuning challenges for the SOC due to the bigger delay, requiring a tight K_p, K_i searching range compared with the previous test. Besides, even with a lower convergence time for this test, it requires further analysis for all the SOC schemes by multiple repetitive simulations due to the random nature of the restart feature.

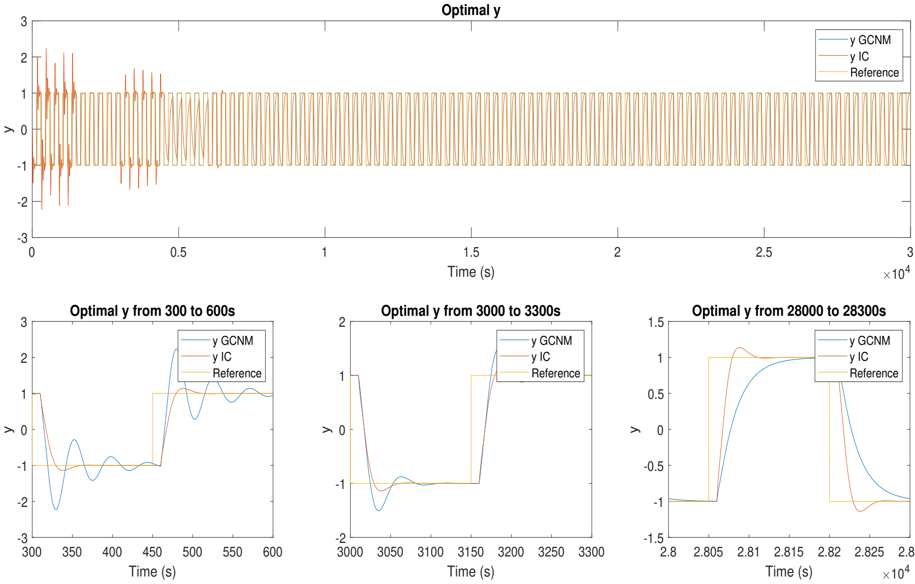
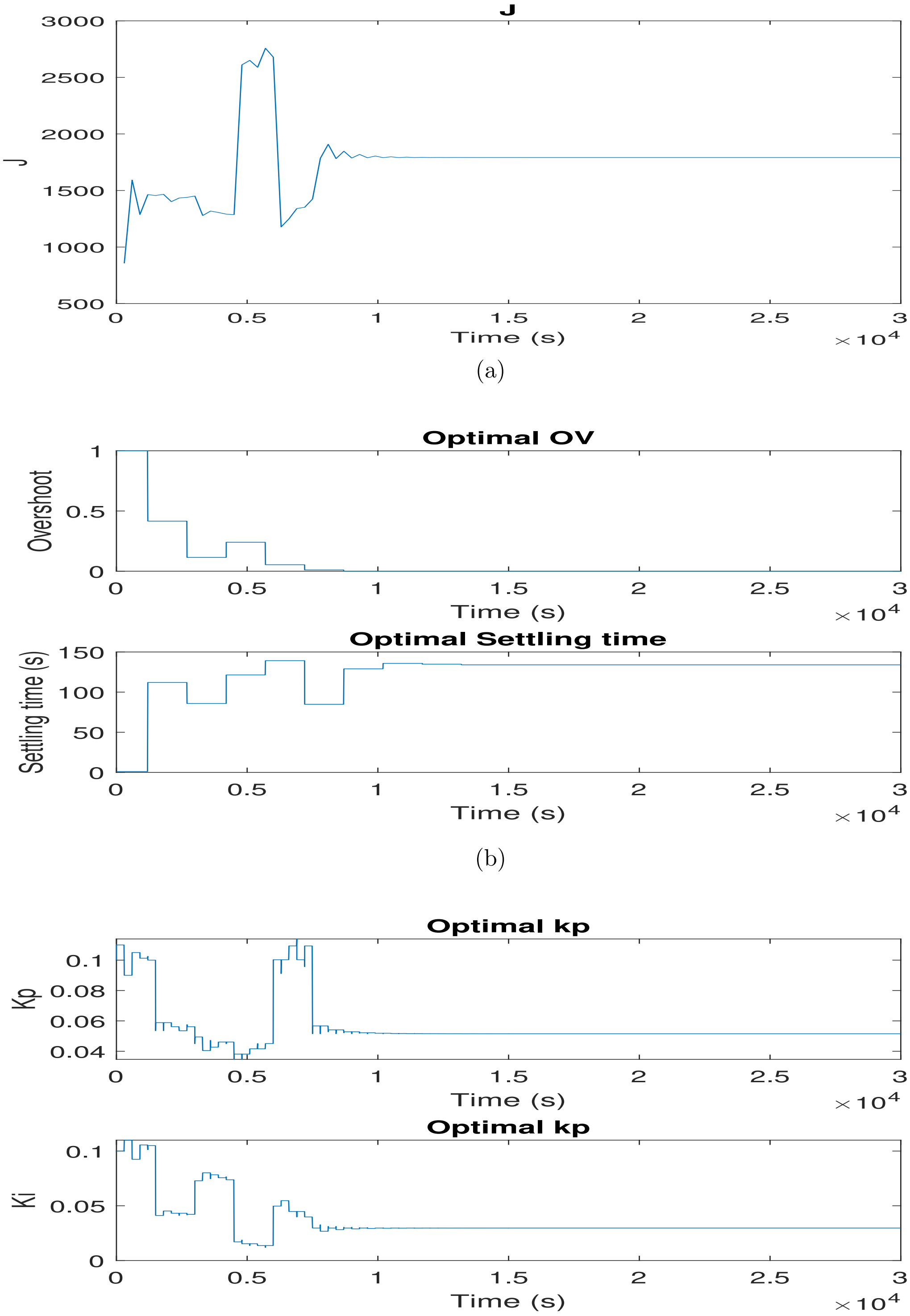


Figure 7.9: Closed-loop System Output Evolution Using SOC Controller $L = 10$



(a)

(b)

(c)

Figure 7.10: SOC Controller a) Cost Function b)Overshoot and Settling Time, and c) PI Gains Evolution for $L = 10$

7.4.3 Numeric Assessment of GCNM Algorithm for the Solution of Optimization Benchmark Problems

A set of classic optimization benchmark functions are used to evaluate the performance of the GCNM against the classic Nelder-Mead optimization algorithm. Table 7.3 shows the performance of the online implementations of the GCNM, the regular NM and the fminsearch optimization algorithms for the solution of the 2D Sphere, Rosenbrock, Akley, Griewank, and Rastrigin functions, which implementation is based on the optimization benchmark proposed by [281] with initial condition for the algorithm $x = [2, 100]$ with a stopping tolerance $\epsilon = 1e - 6$. As can be observed, all the algorithms can solve the sphere and rosenbrock functions in a reasonable amount of iterations. However, in the case of the Rastrigin, Griewank and Akley functions, which contains several local minima, just the GCNM optimization algorithm was able to solve the optimization problem. It indicates that the GCNM algorithm can be extended to different optimization problems.

Table 7.3: GCNM Numeric Benchmark Test With Standards Optimization Functions against classic Nelder-Mead and Matlab `fminsearch`

Algorithm	GCNM				
Function	$f(x)$	x_1	x_2	Iterations	Restarts
Sphere	8.60E-90	-2.86E-67	-6.57E-68	77	0
Rosenbrock	0	1	1	427	0
Akley	8.88E-16	-3.34E-17	-3.19E-16	220	3
Griewank	0	-8.51E-09	3.77E-09	150	1
Rastrigin	0	2.30E-09	1.61E-09	342	4

Algorithm	Nelder -Mead			
Function	$f(x)$	x_1	x_2	Iterations
Sphere	3.30E-96	-1.80E-48	-2.21E-49	77
Rosenbrock	0	1	1	369
Akley	20	10	200	1000
Griewank	0.118337	12.56009	-17.7537771	1000
Rastrigin	143.2651	11.93796	-7.75E-09	1000

Algorithm	`fminsearch`			
Function	$f(x)$	x_1	x_2	Iterations
Sphere	8.23E-10	5.93E-06	2.81E-05	62
Rosenbrock	0	10	200	18
Akley	0.1183	12.56008	-17.753738	52
Griewank	20	10	200	61
Rastrigin	143.5651	11.93799	3.49E-05	95

7.5 Theoretical Convergence Analysis of the SOC Control Framework

In the previous section, the numeric benchmark results shown that SOC controller is able to optimize the closed-loop system performance employing the GCNM algorithm employing the GCNM optimization algorithm. Thus, this section presents a theoretical convergence analysis for the SOC controller with the GCNM controller as an online optimization method. It can be considered a multistart global optimization method composed of global and local optimization stages. In the global stage, a pure random search (PRS) is performed to define the initial conditions for the local optimization stage, which perform the derivative-free Nelder-Mead (NM) optimization algorithm. For this reason, the convergence analysis of the algorithm is established in the global and local stages, using a probabilistic framework for the pure random search stage [282] and the frame-based theory [283] for the Nelder-Mead local search stage.

7.5.1 Pure Random Search Probabilistic Approach

According to [282], a pure random search (PRS) algorithm that solves the optimization problem (7.7) is given by (7.8)

$$\min_{x} F(x), \text{ s.t. } x \in D \tag{7.7}$$

$$X_t = \begin{cases} z_{t+1}, & z_{t+1} \in D \text{ and } F(z_{t+1}) < F(X_t) \\ X_t, & \text{if } z_{t+1} \ \notin \ D \ F(z_{t+1}) \geq F(X_t), \end{cases} \tag{7.8}$$

where X corresponds to a sequence of random vectors $X_0, X_1, \cdots, X_t$ in the domain D with initial condition $X_0 = x_0$ given by the distribution $P_{x_0} = \pi_{start}$ concentrated in the domain $D_{start} \subset D$ with the initial value $\pi_{start} = \epsilon_{x_0}$. If $Z_1 = z_1, \cdots, z_t$ are realizations of the random variable Z, so that, the conditional probability of Z_{t+1} belongs to the level set B is given by (7.9)

$$\begin{aligned} &P(z_{t+1} \in B | X_0 X_1, \cdots, X_t, Z_1, \cdots, Z_t) \\ &= P(z_{t+1} \in B | X_1, \cdots, X_t) = \pi_t(x^t, B). \end{aligned} \tag{7.9}$$

If the random search step solution Z_{t+1} is generated by following $\pi_t(x^t)$ falls into the area of success (7.10), the algorithm optimal solution $X_{t+1} = Z_{t+1}$, otherwise $X_{t+1} = X_t$ returning back into the random generation method.

$$G_F(x) := y \ \in D : \ F(y) < F(x) \tag{7.10}$$

if $X_{t+1} \in G_F(x_t)$, then also $X_s \in G_F(x_t)$ for all $s > t$. In addition, if $F^* = \inf F(x) : x \in D$ and for the given levels $\epsilon > 0, \ M < 0$, the set $B_{\epsilon,M}$ optimal solutions of (7.8) is given by

$$\begin{aligned} B_{\epsilon,M} := \{&y \in D : \ F^*(y) \leq F^* + \epsilon, \\ &\text{if } F^* \in \mathbb{R}, \ F(y) \leq \ M, \ if \ F^* = -\infty\} \end{aligned} \tag{7.11}$$

$$X_s \in B_{\epsilon,M} \Rightarrow X_{s+1} \in B_{\epsilon,M}, s = 0, 1, \cdots, \infty \tag{7.12}$$

hence:

$$P(X_s \in B_{\epsilon,M}) \leq \; P(X_{s+1} \in B_{\epsilon,M}), s = 0, 1, \cdots, \infty \tag{7.13}$$

For the PRS algorithm convergence, the cost function F on (7.8) is measurable and bounded $D \in \mathbb{R}$. In order to analyze the convergence of the PRS method, the probability (7.14) where the t-th iteration of X_t is an $B_{\epsilon,M}$ optimal solution for a given ϵ, M.

$$P(X_t \in B_{\epsilon,M}), \text{for } t = 0, 1, \cdots, \infty. \tag{7.14}$$

Thus, the convergence analysis goal is to determine that the probabilities on (7.14) form a non-decreasing sequence. According to (7.12),

$$X_t \notin B_{\epsilon,M} \Longleftrightarrow X_0 \notin B_{\epsilon,M}, \cdots, X_t \notin B_{\epsilon,M}, \tag{7.15}$$

hence

$$P(X_t \in B_{\epsilon,M}) = 1 - P(X_0 \notin B_{\epsilon,M}, \cdots, X_t \notin B_{\epsilon,M})$$

$$\begin{aligned} P(X_t \in B_{\epsilon,M}) = 1 - \int_{x_0 \notin B_{\epsilon,M}} & P(X_1 \notin B_{\epsilon,M}, \cdots, \\ & X_t \notin B_{\epsilon,M} | X_0 = x_0) \pi_t(d_{x_0}). \end{aligned} \tag{7.16}$$

Denoting by $K_t(x^t, \cdots)$ the conditional distribution of X_{t+1} given $X_0 = x_0$, $X_1, \cdots, X_t = x_t$, we have

$$\begin{aligned} K_t(x^t, B) = \pi_t(x^t, B \cap G_F(x_t)) + \\ (1 - \pi_t(x^t, B \cap G_F(x_t)))\epsilon_{x_t}(B), \end{aligned} \tag{7.17}$$

where ϵ_x is the one point measurement at x. Thus, with $\bar{B}_{\epsilon,M} := D_0 \; B_{\epsilon,M}$, we get

$$\begin{aligned} P(X_1 \notin B_{\epsilon,M}, \cdots, X_t \notin B_{\epsilon,M} | X_0 = x_0) = \\ \int_{x_1 \in \bar{B}_{\epsilon,M}} K_0(x_0, dx_1) \int_{x_{t-1} \in \bar{B}_{\epsilon,M}} K_{t-2}(x^{t-2}, dx_{t-1}) \\ \int_{x_t \in \bar{B}_{\epsilon,M}} K_{t-1}(x^{t-1}, dx_t). \end{aligned} \tag{7.18}$$

Considering that the t-th integral in (7.18)

$$\begin{aligned} \int_{x_t \in \bar{B}_{\epsilon,M}} K_{t-1}(x^{t-1}, dx_t) &= K_{t-1}(x^{t-1}, D_0 \backslash B_{\epsilon,M}) \qquad (7.19) \\ &= K_{t-1}(x^{t-1}, D_0) - K_{t-1}(x^{t-1}, B_{\epsilon,M}) \\ &= 1 - K_{t-1}(x^{t-1}, B_{\epsilon,M}) \end{aligned}$$

Suppose now that $D_0 = D$. Having $x_{t-1} \notin B_{\epsilon,M}$, $\epsilon_{x_{t-1}}(B_{\epsilon,M}) = 0$ and $B_{\epsilon,M} \subset G_F x_{t-1}$, (7.20) yields to (7.21) for all $x_s \in D_0 \backslash B_{\epsilon,M},\ s = 0, 1, \cdots, t-1$.

$$\begin{aligned} \int_{x_t \in \bar{B}_{\epsilon,M}} K_{t-1}(x^{t-1}, dx_t) &\leq 1 - \pi_{t-1}(x^{t-1}, B_{\epsilon,M}) \\ &\leq 1 - \inf(\pi_{t-1}(x^{t-1}, B_{\epsilon,M}) : x_s \in D_0 \backslash B_{\epsilon,M}, \qquad (7.20) \\ &\qquad 0 \leq s \leq t-1). \end{aligned}$$

Define $\alpha_t,\ t = 0, 1, \cdots$, by

$$\alpha_t := \alpha_t(B_{\epsilon,M}) = \inf(\pi_t(x^t, B_{\epsilon,M}) : \qquad (7.21)$$

$$x_s \in D_0 \backslash B_{\epsilon,M},\ 0 \leq s \leq t-1). \qquad (7.22)$$

From (7.17) and (7.18),

$$\begin{aligned} &P(X_1 \notin B_{\epsilon,M}, \cdots, X_t \notin B_{\epsilon,M} | X_0 = x_0) \qquad (7.23) \\ &\qquad \leq \prod_{s=0}^{t-1} (1 - \alpha_s(B_{\epsilon,M})) \end{aligned}$$

hence, by (7.16) and (7.24),

$$\begin{aligned} P(X_t \in B_{\epsilon,M}) &= 1 - \int_{x_0 \notin B_{\epsilon,M}} P(X_1 \notin B_{\epsilon,M}, \cdots, \\ &\qquad X_t \notin B_{\epsilon,M} | X_0 = x_0) \pi_{start}(dx_0) \qquad (7.24) \\ &\geq 1 - (1 - \pi_{start}(B_{\epsilon,M})) \prod_{s=0}^{t-1} (1 - \alpha_s(B_{\epsilon,M})) \end{aligned}$$

since log u $\leq u - 1$, for $u > 0$ we have

$$(1 - \pi_{start} B_{\epsilon,M})) \prod_{s=0}^{t-1} (1 - \alpha_s(B_{\epsilon,M}) \qquad (7.25)$$

$$\leq \exp(-\pi_{start} B_{\epsilon,M} - \sum_{s=0}^{t-1} \alpha_s) \qquad (7.26)$$

and therefore also,

$$P(X \in B_{\epsilon,M}) \geq 1 - exp(-\pi_{start} B_{\epsilon,M} - \sum_{s=0}^{t-1} \alpha_s). \tag{7.27}$$

So, Theorem 1 proposed by [282] can be used for the convergence evaluation of PRS algorithm. The proof for Theorem 1 can be found in [282, 284].

Theorem 1 *Suppose now that $D_0 = D$. The search process* (7.8) *has the following convergence properties:*

1. *If for an $\epsilon > 0$,$M < 0$, resp,*

   $$\sum_{s=0}^{\infty} \alpha_s(B_{\epsilon,M}) = +\infty \tag{7.28}$$

   then $\lim_{n\to\infty} P(X_t \in B_{\epsilon,M}) = 1$*;*

2. *Suppose that $F^* \in \mathbb{R}$ and*

   $$\lim_{n\to\infty} P(X_n \in B_\epsilon = 1) \text{ for every} \epsilon > 0 \tag{7.29}$$

   then $\lim_{n\to\infty} F(X_n) = F^*$ *with probability one (W.P.1)*

3. *Assume that $F^* \in \mathbb{R}$ and F is continuous and that the level sets D_ϵ are nonempty and compact for each $\epsilon > 0$. Then* $\lim_{t\to\infty} F(X_n) = F^*$ *implies that* $\lim_{n\to\infty} dist(X_t, D^*) = 0$*, where $dist(X_t, D^*)$ denotes the distance between X_t and the set D of global minimum points of the cost function.*

7.5.2 Positive Bases and Frame Theory

The positive bases and frame theory is a methodology to analyze the convergence and performance of pattern search type unconstrained optimization algorithms [283, 285]. A frame $\Phi^{(k)}$ consists of a set of at most $2n$ points given by (7.30) where $x^{(k)}$ is the frame central point, $\nu_+^{(k)}$ is a positive base and $h^{(k)}$ is the frame size

$$\Phi^{(k)} = \Phi(x^{(k)}, \nu^{(k)}, h^{(k)}). \tag{7.30}$$

Likewise, a positive base ν_+ satisfies the following properties:

1. Every vector in $\mathbb{R}^n$ is a non-negative combination of the members of ν_+.

2. Any proper subset of $\nu_+^{(k)}$ is not a positive basis.

Herein, each positive basis $\nu_+^{(k)}$ is defined in terms of a basis $\nu^{(k)}$, where

$$\nu^{(k)} = \nu_i^{(k)} \in \mathbb{R}^n : i = 1, \cdots, n \tag{7.31}$$

and k is the frame number. Each such basis $\nu^{(k)}$ is required to satisfy the conditions:

$$|\det([\nu_1^{(k)}, \nu_2^{(k)}, \cdots, \nu_n^{(k)}])| > \tau \tag{7.32}$$

$$||\nu_i^{(k)}|| \leq K_0, \ \forall i \in 1, \cdots, n \tag{7.33}$$

where τ and K_0 are positive constants independent of k. The positive bases used herein are of a particular form: the first n members of $\nu_+^{(k)}$ are those of $\nu^{(k)}$ in the same order and the final member of $\nu_+^{(k)}$ is given by:

$$\nu_{n+1}^{(k)} = -\left(\frac{\gamma - \alpha}{\alpha n}\right) \sum_{i=1}^{n} v_i^{(k)} \tag{7.34}$$

which, for $\gamma > \alpha$, yields ordered positive bases of the form:

$$\nu_+^{(k)} = \left[v_1^{(k)}, v_2^{(k)}, \cdots, v_n^{(k)}, \ \ -\left(\frac{\gamma - \alpha}{\alpha n}\right) \sum_{i=1}^{n} v_i^{(k)} \right]. \tag{7.35}$$

Equation (7.35) imposes a specific order on the members of $\nu_+^{(k)}$, and so these positive frames will be referred to as ordered positive bases from now on. The bound on each $||\nu_i^{(k)}||$ in conditions (7.32) and (7.33) can be extended to all members of each $\nu_+^{(k)}$, by setting K equal to the larger of K_0 and $(\gamma - \alpha)K_0/\alpha$ yielding to,

$$||\nu^{(k)}|| \leq \ K \ \forall k \ \text{ and } \forall i = 1, \cdots, n+1. \tag{7.36}$$

Herein a frame Φ consists of $n+1$ points arranged around the central point called the frame center $x^{(k)}$. The frame $\Phi(x, \nu_+^{(k)}, h^{(k)})$ is specified in terms of a frame center x, a positive basis $\nu_+^{(k)}$ and a frame size $h^{(k)}$ as follows:

$$\Phi = x + h^{(k)}\nu \ : \ \nu \in \nu_+^{(k)}. \tag{7.37}$$

The frame size $h^{(k)}$ is adjusted from time to time to guarantee the convergence under appropriate conditions. From the property of positive basis (7.32), the following stopping rules can be developed according to [283, 285]:

Theorem 2 *If the set of vectors ν_+ corresponds to a positive base, then*

$$g^T\nu \geq 0 \ \forall \nu \ \in \nu_+ \longrightarrow g = 0. \tag{7.38}$$

The proof for Theorem 2 can be found in [285]. In order to determine the convergence of the frame based algorithm, the following definitions about the minimal and quasi-minimal frames are established.

Definition 1 ***Minimal frame:*** *a frame $\Phi^{(k)} = \Phi(x^{(k)}, \nu^k, h^{(k)})$ is defined as minimal iff*

$$f(x + h\nu) \geq f(x) \ \forall \nu \in \nu_+. \tag{7.39}$$

Definition 2 ***Quasi-Minimal frame:*** *a frame $\Phi^{(k)} = \Phi(x^{(k)}, \nu^k, h^{(k)})$ is defined as quasi-minimal if it satisfies the weaker condition*

$$f(x + h\nu) + \epsilon \geq f(x) \forall \nu \in \nu_+ \tag{7.40}$$

is called ϵ-quasi-minimal, and the corresponding point x will be referred to as an ϵ-quasi minimal point.

According to [285], the value of ϵ can be chosen as (7.41), where $\beta > 1$, and N is a positive constant. In this case, ϵ can be used as a measure of sufficient descent of the algorithm.

$$\epsilon = Nh^{\beta}. \tag{7.41}$$

Based on Definitions 1 and 2, [285] defines an algorithm framework, which can be adapted to analyze pattern search type optimization algorithms, which is given by Algorithm 1. Here $f^{(k)} = f(x^{(k)})$ is used, where k counts the number of frames. In contrast the $x_i^{(j)}$ are points on the j^{th} simplex. The variables m and $z^{(m)}$ count the number of quasi-minimal frames, and denote the quasi-minimal points. The algorithmic framework is composed by two loops. The outer loop (steps 2 and 4) generates a sequence of quasi-minimal frames that under mild conditions must converge into a stationary point. The inner loop (step 3) generates a sequence of points until a quasi minimal is found, where a new frame $\Phi^{(k)} = \Phi(x^{(k+1)}, \nu^k, h^{(k)})$ is centered on $x^{(k+1)}$. Thus, if a search algorithm follows the structure given by Algorithm 1, according to the frame theory [285], the convergence of that algorithm into a stationary point is given by the Theorem 3 where the monotonicity of $f^{(k)}$ means that the sequence $x^{(k)}$ converges to a region where $f(k)$ as $k \to \infty$ is constant. In the usual case when $f^{(k)}$ converges to a unique point, that limit point is a stationary point of f. The proof for Theorem 3 can be found in [285, 286].

Algorithm 1: Frame based algorithm [285]

```
1 Initialize $m = k = 0$, $x^{(1)}, \beta > 1$ and $N > 0$
2 Choose $h$, set $\epsilon = Nh^{\beta}$.
3 Perform the following process repeatedly until a quasi-minimal frame is
    found: execute any finite process which first increments $k$ and then
    either generates a quasi minimal frame $\Phi^{(k)} = \Phi(x^{(k+1)}, \nu^k, h^{(k)})$ that
    satisfies $f^{(k+1)} \leq f^{(k)}$ or locates a point $x^{(k+1)}$ satisfying
        $$f^{(k+1)} \leq f^{(k)} - \epsilon$$
4 Increment $m$. let $z^{(m)} = x^{(k+1)}$ ($z^{(m)}$ is quasi minimal). If Stopping
    conditions are not satisfied, go back to step 2.
```

Theorem 3 *Assume the following:*

1. *The sequence of iterates $x^{(k)}$ is bounded;*
2. *f is continuously differentiable and its gradient ∇f is Lipschitz in any bounded region of $\mathbb{R}$.*
3. *$\exists K, \tau > 0$ such that $|\det(|v_1^{(k)}, \cdots, v_n^{(k)}|)| > \tau \ \forall \ k$, and $||v_i^{(k)}|| \leq K \ \forall \ K$ and $\forall \ i = 1, \cdots, n+1$ and*
4. *$h^{(k)} \rightarrow 0$ as $k \rightarrow \infty$.*

Then each cluster point $z^{(\infty)}$ of the sequence of quasi-minimal iterates $z^{(m)}$ (see Algorithm 1) is a stationary point of $f(x)$.

7.5.3 Redefinition of the GCNM Optimization Algorithm for Convergence Analysis

It is important to notice that for classic Nelder-Mead and multistart optimization algorithms, most of the convergence studies are based on numeric simulation results under the assumption of certain execution conditions of the NM steps for low dimensions optimization problems [287–291]. For this reason, the GCNM algorithm is modified as shown in Fig. 7.11 in order to apply theoretical tools for its convergence analysis.

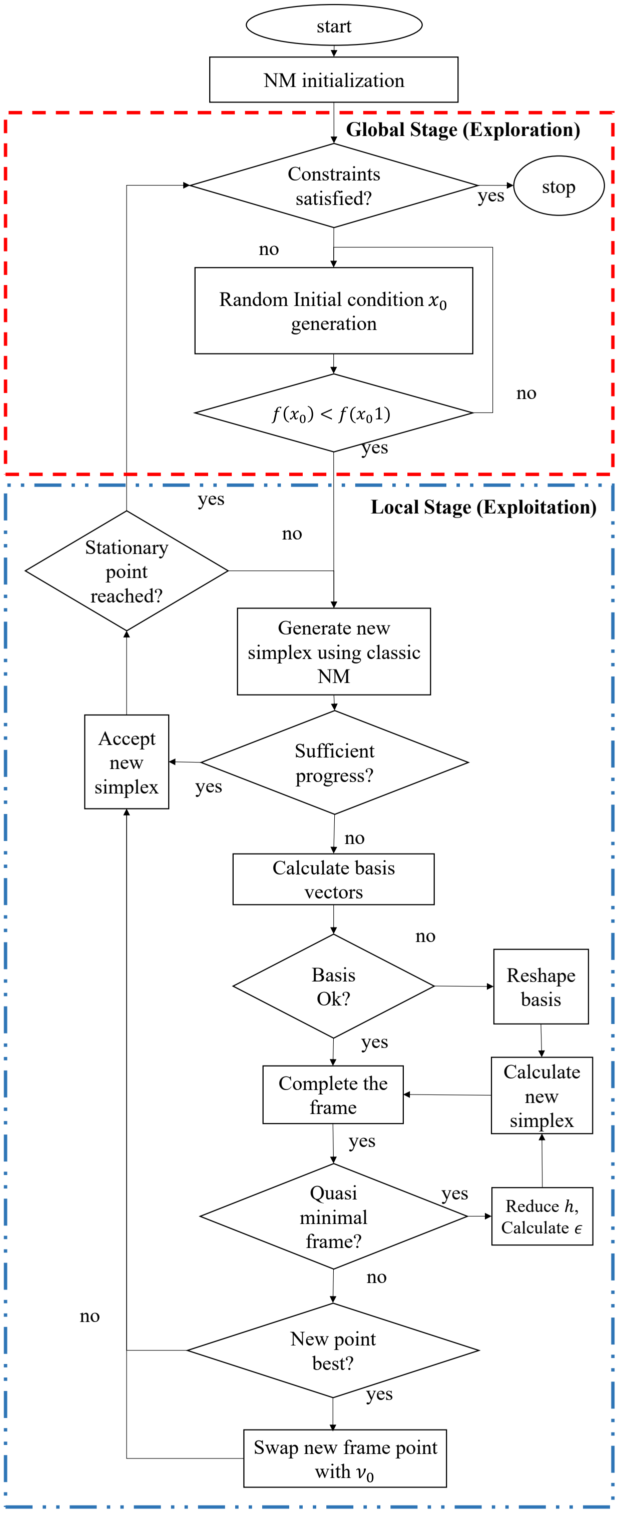


Figure 7.11: GCNM Algorithm with Simplex Degeneration Aware NM

As can be observed, the new GCNM initial stage performs a PRS to define the initial conditions for the second stage that runs a modified Nelder-Mead (MNM) optimization algorithm with simplex deformation awareness proposed by [286]. For the MNM algorithm, it starts performing a set of classic NM iterations to generate a new simplex. If the current simplex minimizes the cost function J, then the simplex is accepted. Otherwise, the positive basis ν_+ is calculated for the current simplex. If ν_+ does not satisfy (7.32) and (7.33) the basis is reshaped. In this case, the reshape method performs a QR matrix decomposition of the matrix $|\det(\nu_n)|$, where each member of Q is multiplied by the scaling factor (7.42) proposed by [283]. Thus, the new simplex satisfies (7.32) and (7.33) avoiding its possible degeneration.

$$D_i = \mathrm{sign}(R_{ii}) \min\left(K_0, \max\left(|R_{ii}|, \frac{\bar{R}}{10}\right)\right), \tag{7.42}$$

where,

$$\bar{R} = \frac{1}{n}\sum |R_{ii}|.$$

Once the basis is bounded, the frame is completed. To perform this action, consider an ascendant sorted simplex $x_0, \cdots, x_n$, the side vectors ν_i forming the positive basis ν_+ are defined by (7.43)

$$\nu_i = \frac{x_i - x_0}{h}, \quad \forall i = 1, \cdots, n. \tag{7.43}$$

In this case, ν_i needs one more point to complete the frame, which can be defined as x_p or pseudo-expand point given by

$$x_p = x_0 + h\nu_{n+1} = x_0 - h\left(\frac{\gamma - \alpha}{\alpha n}\right)\sum_{i=1}^{n} v_i. \tag{7.44}$$

Notice that x_p can be defined also as the “ghost” simplex H that can be rewritten in terms of the centroid $\bar{g}$ as $H = \bar{g} - \frac{x_0 - \bar{g}}{\alpha}$. The main property of H is that it is reflected over x_0 once the MNM calculates the simplex. After that, the current frame is evaluated if it is a Quasi-Minimal Frame (QMF) following Definition 2. If yes, h and ϵ are recalculated and basis directions Q are reversed, defining a new simplex. Otherwise, if any of the frame point is better than the original frame v_0, that point is swapped in the original frame and the new simplex is accepted. Otherwise, the frame is considered as non-QMF and the simplex is accepted.

The convergence analysis of the SOC controller requires to redefine the GCNM algorithm as a two-stage optimization process. Considering that the GCNM fits as a multistart optimization algorithm, it can be divided into a global (exploration) stage corresponding to the PRS search and local (exploitation) stage using the MNM algorithm. In this configuration, the global optimization stage provides the initial conditions for the NM local search that minimize $f(x_0)$ compared with the IC selected in the previous stage. Likewise, the NM search stage performs a local search until the algorithm reaches a stationary point. For the convergence analysis the following stability assumption is made for the SOC control problem.

Assumption 1 *The optimization parameters $x = [k_p,\ k_i]$ for the SOC controller using GCNM algorithm corresponds to the domain D, which is bounded as $D = \{k_{p_{min}} \leq k_p \leq, k_{p_{max}},\ k_{i_{min}} \leq k_i \leq k_{i_{max}}\}$ ensuring that the roots for the closed loop control system are always located in the OLHP.*

Based on Fig. 7.11, the convergence analysis can be performed in two independent stages. For the global stage the probabilistic theory is employed to analyze the convergence of the PRS algorithm. Likewise, the frame and positive basis method is used for the local optimization stage convergence analysis.

7.5.4 Global Stage Convergence Analysis

In the global optimization stage, the goal is to provide a different starting point x_0 that minimizes $f(x_0)$ with respect to the condition defined on the last restart. As can be observed in Fig. 7.11, the structure of the global optimization structure matches with that of the PRS algorithm given by (7.8). For this reason, the convergence of the global stage using PRS can be established by applying Theorem 1.

For Condition 1 in Theorem 1, the probability distribution π_{start} is given by a normal random distribution $N(\mu, \sigma)$ that belongs to D, which is a bounded set according to Assumption 1, with real and positive values, bounding the level set $f \in B_{\epsilon,M}$ and $D_0 \in x_0$, satisfying Condition 1 in Theorem 1. For condition 2, in the SOC control framework $J(x) \in \mathbb{R}$ s.t. $f^* \in \mathbb{R}$, this condition is satisfied. Finally, for condition 3, based on the considerations in Assumption 1, the function is continuous and the level sets D_ϵ are not empty, so $\text{dist}(X_t, D^*)$ will be zero as $t \to \infty$. So that, for the proposed SOC control architecture, the random search will converge to f^* as $t \to \infty$.

7.5.5 Local Stage Convergence Analysis

In the local stage, the MNM algorithm fits into the "frame based method" given by Algorithm 1; therefore, its convergence is given by Theorem 3. In the case of the SOC controller, for condition 1, the sequence of the inner loop iterates $x^{(k)}$ is bounded due to the constraints values for k_p, k_i in Assumption1 ensuring the algorithm monotonicity and satisfying Theorem 2. Likewise, in condition 2, f is a continuous function in D following the stability conditions in Assumption1. Condition 3 is guaranteed thanks to the simplex deformation metric based on (7.32) and (7.33), that reshape the simplex in order to satisfy the bounds τ, K respectively. For condition 4 is satisfied because each time the base $\Phi^{(k+1)} = \Phi(x^{(k+1)}, \nu^{(k+1)}, h^{(k+1)})$

is generated, it corresponds to a quasi-non minimal frame, which implies that the frame size h has been readjusted by κ as h/κ. Thus, it is possible to conclude that local optimization stage converge into a stationary point as $k \to \infty$.

Finally, based on the convergence analysis for global and local stages of GCNM algorithm, we can conclude that the SOC controller is able to converge to an optimal solution x^* as $k \to \infty$ if the conditions on Theorems 1 to 3 as well as Assumption 1 are satisfied in a bounded stable domain D.

7.5.6 Numerical Example

The numeric convergence evaluation of the GCNM algorithm is performed using the Matlab/Simulink benchmark presented in [280]. In this case, the benchmark system uses a generalized FOPDT system based on the generalized FOPDT system as defined in Appendix, with dead times $L = \{0.1,\ 1,\ 10\}$ corresponding to a lag dominated, balanced, and delay dominated system with a PI controller as (7.1) and (7.2). Likewise, the cost function (7.3) has the optimization weights $W_1 = 1,\ W_2 = 5,\ W_3 = 0.1$. The constraints for each dead time on the benchmark are shown in Table 7.4.

The configuration parameters for the GCNM on the global stage are for the global stage a standard Gaussian distribution $N(0, 1)$ for the random initial condition generation x_0 bounded according to the $k_p,\ k_i$ limits shown before. On the other hand, for the MNM algorithm, its implementation is based on [283]. The benchmark SOC control problem has $n = 2$ dimensions, where the reflection, contraction, expansion gains are $\alpha = 1,\ \beta = 1 + 2n_r\gamma = 0.75 - 0.5n_r, \delta = 1 - n_r$ with $n_r = 1/n$ as suggested in [277]. Likewise, the initial simplex size value is $h = 1$, with $N = 100, \kappa = 100, \beta = 4.5$. Considering the stochastic nature of the random search stage, the benchmark is executed 30 times to evaluate the average performance of the GCNM algorithm to find $[k_p^*, k_i^*]$. The benchmark version for this test can be downloaded from `https://github.com/tartanus/SOCBenchmark GCNMAware`. As example, Fig. 7.12 shows the convergence results for the SOC controller with GCNM controller for $L = 1s$ after 30 iterations. As can be observed, the GCNM algorithm converge into the region of attraction for the global minimum in all the cases, which can be noticed for the value of the cost function J and the $x^* = [k_p^*, k_i^*]$. Likewise, the cost function evolution has a monotonic decreasing behavior, which is desirable for SOC control system.

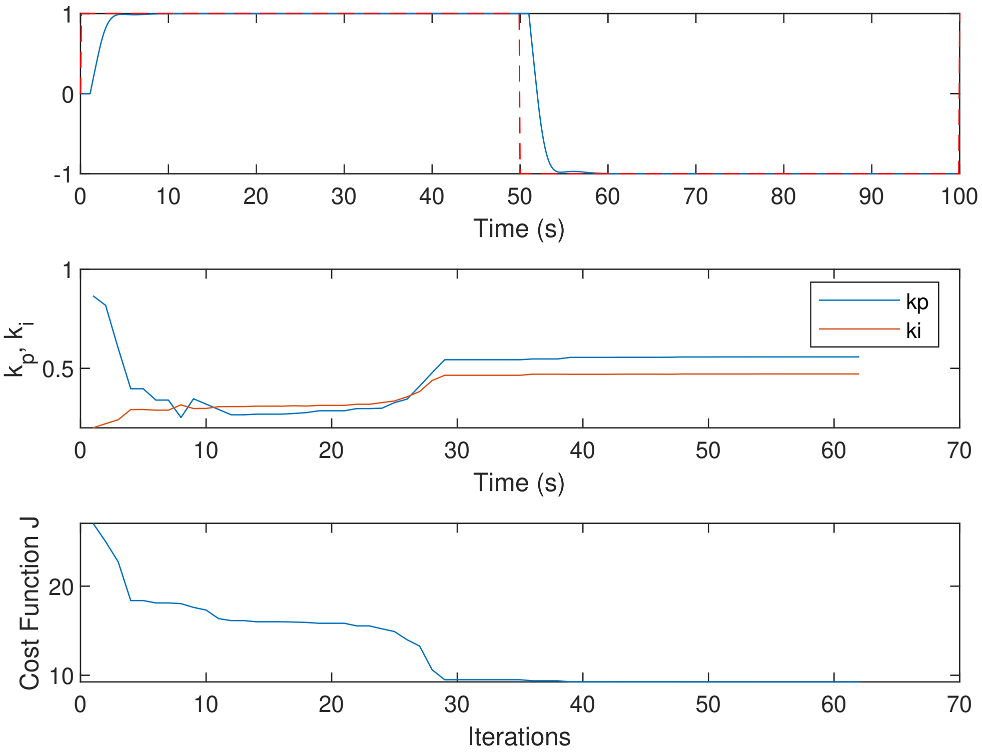


Figure 7.12: GCNM SOC Controller a) Time Response b) $k_p,\ k_i$ Gains and c) Cost function J for $L = 1s$

On the other hand, the average performance of the SOC controller with the benchmark problem is presented in Table 7.5. As can be observed, the SOC controller with the GCNM optimization algorithm satisfies the constraints defined for each case with an approximated convergence time of 21000, 145000, and 41000 seconds for $L = \{0.1,\ 1,\ 10\}$ respectively and a restart ratio about two restarts per execution. Likewise, the performance indices show a consistent behavior among the three benchmark cases. Thus, we can say that numeric simulation results confirm the theoretical convergence analysis for the SOC controller with GCNM optimization algorithm.

Table 7.4: SOC Benchmark Configuration Parameters

	Test 1		Test 2		Test 3	
	$L = 0.1$s		$L = 1$s		$L = 10$s	
Parameter	min	max	min	max	min	max
k_p	0.01	1	0.01	1	0.01	1
k_i	0.01	1	0.01	1	0.01	1
Settling time	2 s		5 s		30 s	
OV (%)	2%		2%		2%	
Reference signal period (s)	100					

Table 7.5: 30 Runs Average Performance Indices for the SOC Controller Benchmark for $L = \{0.1s,\ 1s,\ 10s\}$

Performance index	Dead time (s)		
	$L = 0.1$	$L = 1$	$L = 10$
Convergence time (s)	21496.66667	14523.33333	41453.33
Overshoot (%)	1.52E-13	1.67E-12	0.031098
Settling time (s)	1.243333333	4.8	27.89333
RMS	0.996460621	0.976644968	0.839253
ISE	0.015906878	0.08896999	0.648812
IAE	0.994283411	0.964255413	0.763283
RMSE	0.031586317	0.62302478	4.881796
Restarts	2.1667	1.333	2.4667
k_p	4.3231	0.5577	0.2788
k_i	4.0607	0.4715	0.0638

7.6 Self Optimizing Control Benchmark Extended Application Using Digital Twin

The proposed SOC benchmark capabilities can be extended by replacing the FOPDT system proposed in the benchmark problem for the Digital Twin of a real system. Thus, the performance of the SOC controller with GCNM optimization algorithm can be evaluated for a real system through its Digital Twin. For this reason, two cases study are analyzed, corresponding to the Peltier thermoelectric system employed in Chapters 5 and 6 as well as a position mechatronic system.

7.6.1 Case Study 1: SOC for A Peltier Thermoelectric System

In this case the Digital Twin for the Peltier Thermal system used in Chapters 5 and 6 is employed to extend the SOC capabilities into a real system. Figure 7.13 shows the GCNM SOC controller combined with the Peltier thermal system Digital Twin. In this case, an FOPDT model of the thermal system system is identified to determine if the system has a lag dominant, balanced, or delay dominant dynamics and set the right optimization parameters (k_p, k_i, OV, T_s) for the SOC controller. The identified FOPDT model for the Digital Twin is given by (7.45) corresponding to a lag dominated system. Thus, the design specifications for the SOC PI controller are the same proposed for Test 1 in Table 7.4. The only difference is the initial conditions for k_p and k_i, which is calculated using the ZN method proposed in [278] based on (7.45). The obtained values are $k_p = 102$ and $k_i = 85$. For the SOC controller application, the thermal parameters of the Peltier thermoelectric module selected for the Digital Twin are C= 38.15 J/K, $\alpha = 0.097989$ V/K, $R = 3.3\Omega$, and $K = 0.2207$ K/W. The remaining parameters of the Digital Twin are the same employed on Chapter 5.

$$g(s) = \frac{0.63}{32.53s + 1} e^{-0.6s}. \tag{7.45}$$

The performance of the SOC controller with the Digital Twin of the Peltier system to improve its closed-loop response is shown in Fig. 7.13 and Fig. 7.14. As can be observed, the SOC controller is able to improve the closed-loop response of the system about after 10000 seconds, corresponding to 34 heating cooling cycles of the thermal system. Likewise, it is important to notice that the real-time optimization is not affected by the presence of sensor noise. Likewise, it can be observed that the settling time and overshoot performance specifications are satisfied when the SOC PI terms reach the optimal value of $k_p = 162$ and $k_i = 82$. In the case of the cost function evolution, it can be observed that its value has small variations due to the presence of random noise which has no significant impact on the final values of k_p and k_i. Thus, we can say that using the Digital Twin in combination with the proposed SOC architecture is the first step towards the Smart Control Engineering implementation on real systems by allowing the performance evaluation of the control strategy as well as the repeatability of the system behavior under repetitive tasks. In particular, this SOC benchmark setup defines a framework to continue studying smart control strategies, extensible via Digital Twin not only for thermal systems but also for any other systems. Likewise, the SOC controller can be extended to more complex configurations with more degrees of freedom like integer or fractional order PID controller, state space, lead-lag compensation, among other controllers.

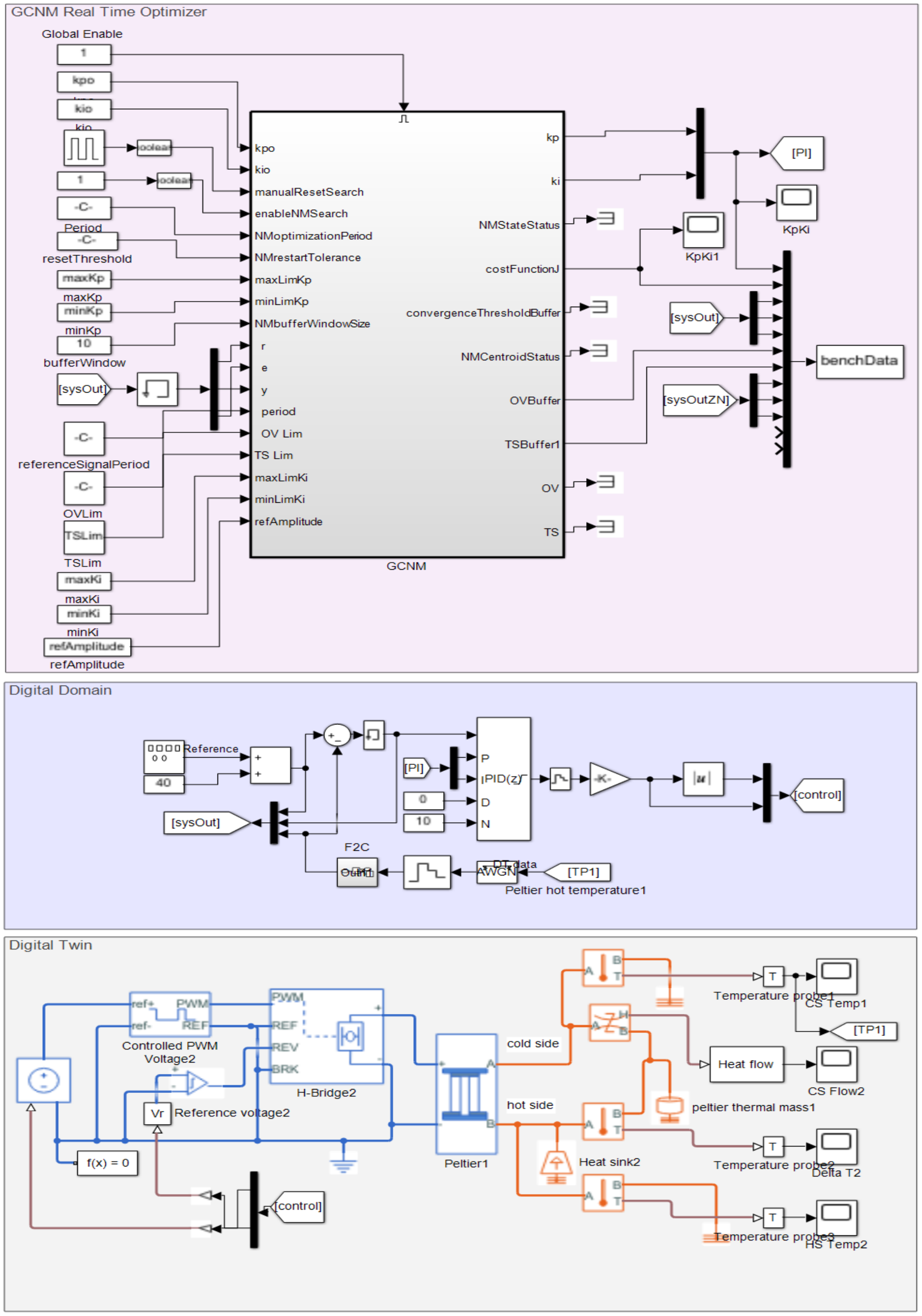


Figure 7.13: Self Optimizing PI Controller with Peltier Thermal System Digital Twin Benchmark Configuration.

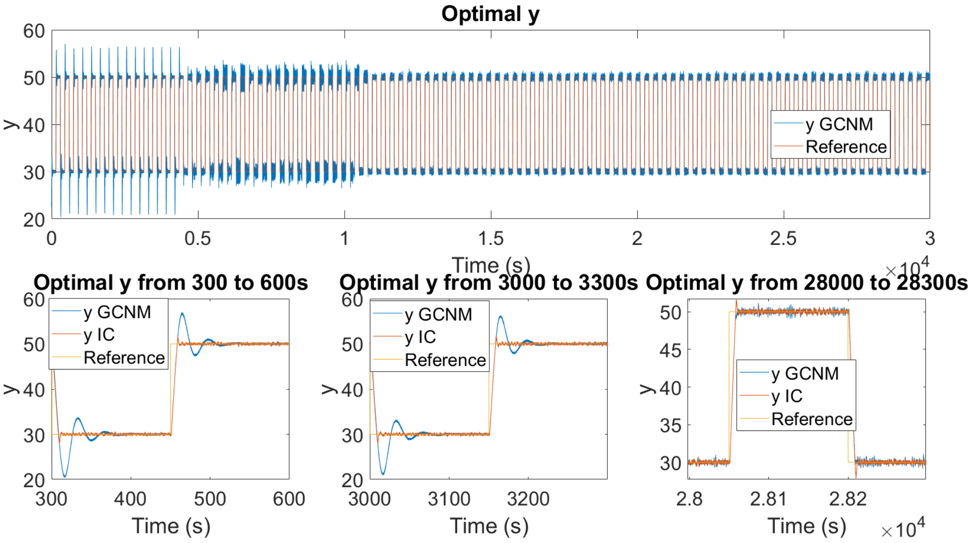
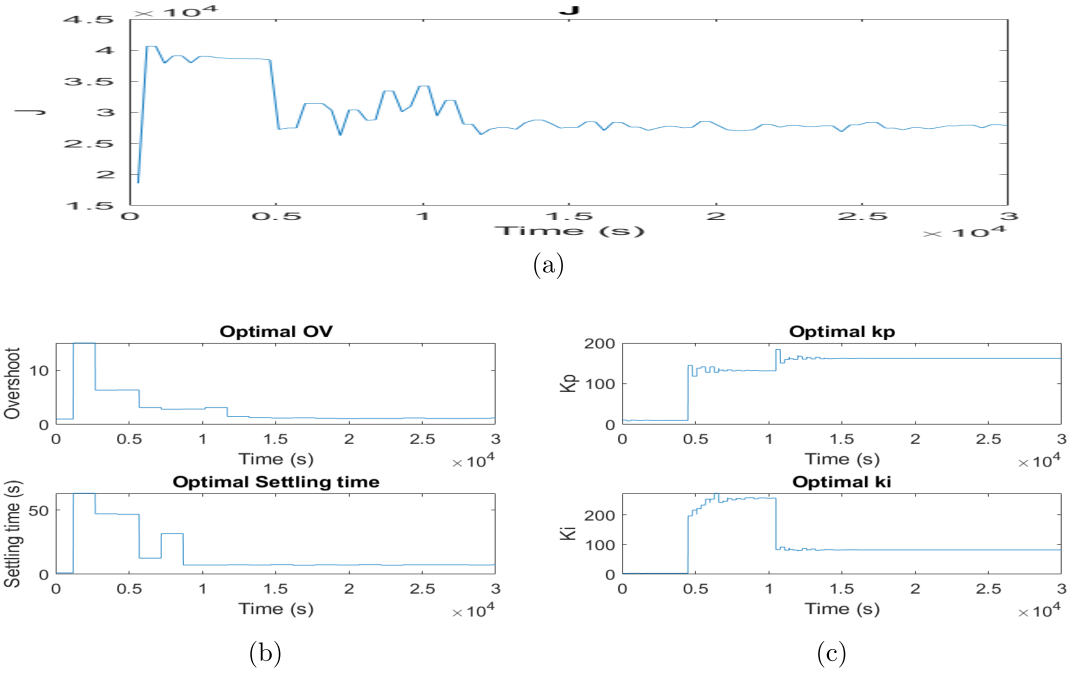


Figure 7.14: Closed-Loop System Output Evolution Using SOC Controller with the Peltier Thermal System Digital Twin.



(a)

(b)

(c)

Figure 7.15: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution for the Peltier Thermal System Digital Twin.

7.6.1.1 Analysis of SOC and DT for Disturbance and Parametric Uncertainty on the Peltier System

In this section, the performance of the SOC and DT in the presence of external disturbances and parametric uncertainties is evaluated. With this purpose, three test are developed. The first test apply an external disturbance to the temperature feedback signal at $t = 15000s$. The second test reduces the heat capacity of the Peltier to its 80% at $t = 7000s$. Finally, the third test modifies the Peltier electrical resistance of the Peltier by 33% at $t = 7000s$. These tests were selected considering that it can be possible scenarios during the closed-loop execution of the system. The results for the first test are shown in Fig. 7.16 and Fig. 7.17. As can be observed, the DT and SOC controller keep the optimal performance of the system. Once the perturbation appears, the SOC control layer detect the changes and restart the optimization process, resulting in a set of PI gains that compensate for the system performance. For test two, the results can be observed in Fig. 7.18 and Fig. 7.19. As can be observed, changing the heat capacity of the Peltier heating element alters the system dynamics. Again, once the SOC controller detects that the performance is not satisfied, restarts the optimal search minimizing the effects of the dynamic change. In test three, which result is shown in Fig. 7.20 and Fig. 7.21, the response of the SOC control coincides with the previous tests, being aware of the system performance degradation. In this case, the parameter change induces noisy behavior to the real system, which is handled by the SOC. A performance comparison of the SOC controller with Digital Twin is shown in Table 7.6, containing the same performance indices from the SOC test benchmark are calculated for this system. As can be observed, in the three cases the SOC controller with Digital Twin is able to satisfy the desired performance conditions. Thus, it is possible to say that combining DT and SOC provides support against parametric changes on the system that could be caused by degradation or failures on the system physical components.

Table 7.6: Performance of SOC Controller with Digital Twin for Peltier Thermoelectric System Agains External Disturbances and Parametric Uncertainties

Test	Convergence time (s)	OV	Settling Time	RMS	ISE	IAE	RMSE
Disturbance	33297.75	1.590153	17.1	41.1711	9.686518	39.9701	22.13833
Heat capacity Cp	11399.23	0.146174	17.79	41.04088	13.54457	39.87402	236.9956
Peltier resistance R	26998.17	1.60E-06	55.446	40.60866	20.77063	39.48017	900.3915

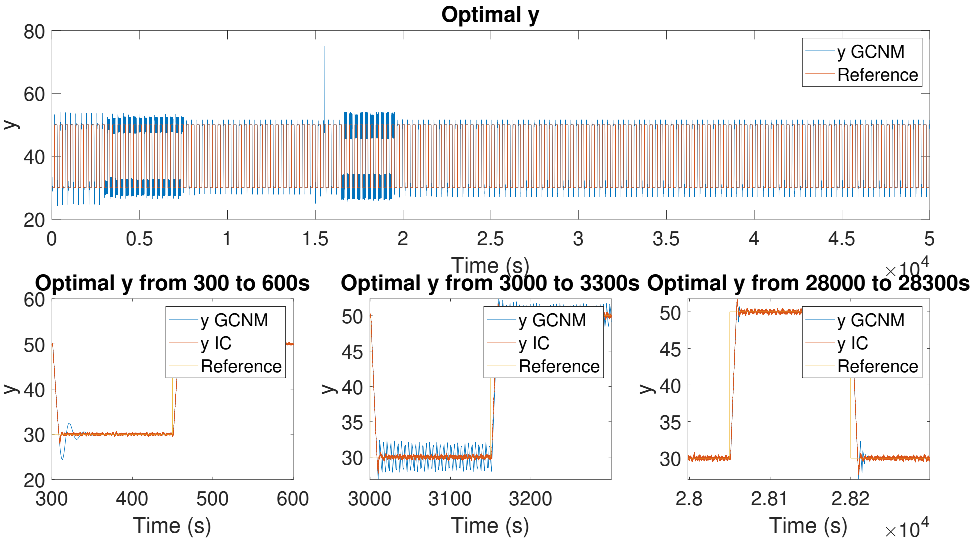


Figure 7.16: SOC controller Response with Digital Twin for Peltier Thermal System Against External Disturbance in Feedback Signal

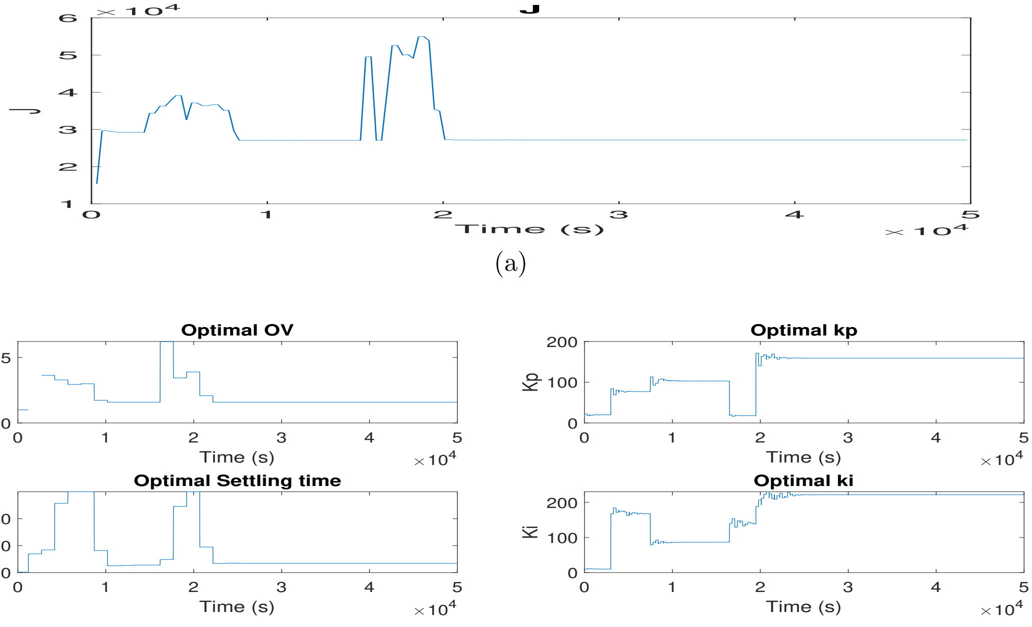


Figure 7.17: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution Peltier Thermal Digital Twin System Against External Disturbance in Feedback Signal.

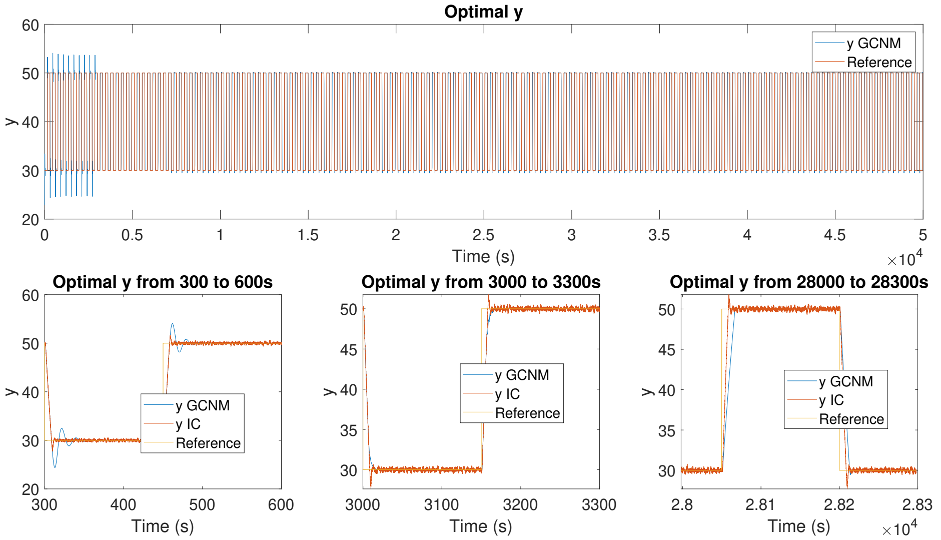
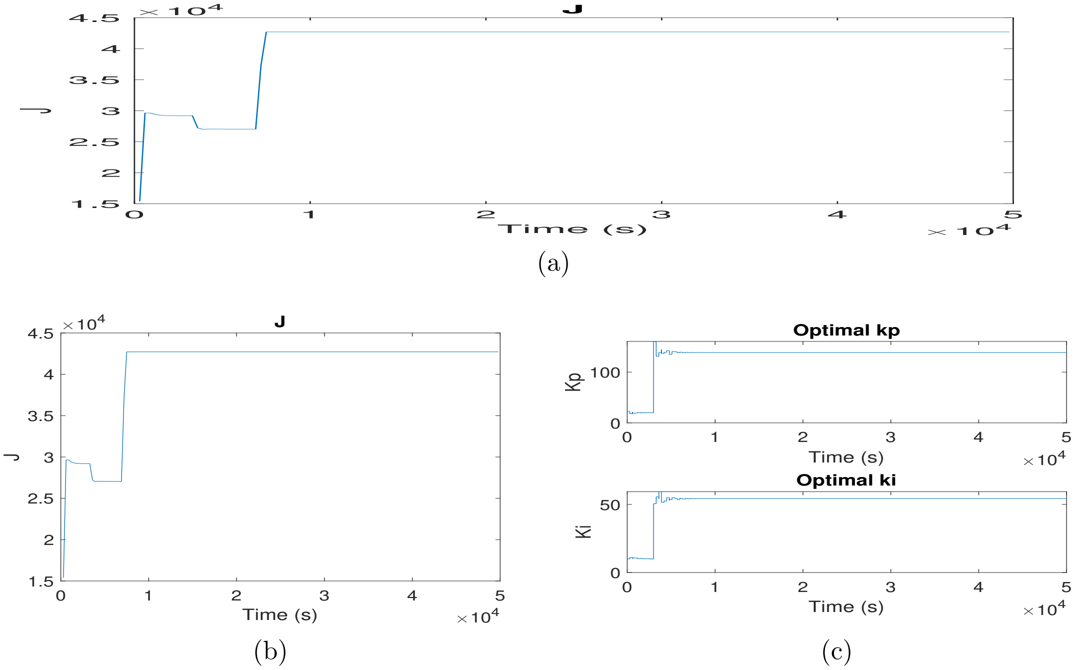


Figure 7.18: SOC Controller Response with Digital Twin for Peltier Thermal System Against Parametric Change on Peltier Heat Capacity



(a)

(b)

(c)

Figure 7.19: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution of the Peltier Thermal Digital Twin System Against Parametric Change on Peltier Heat Capacity

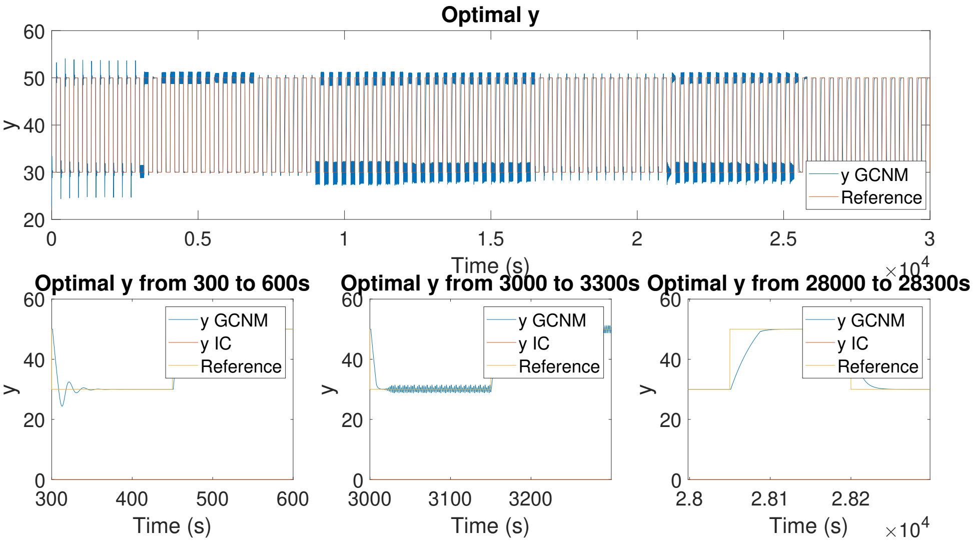


Figure 7.20: SOC Controller Response with Digital Twin for Peltier Thermal System Against Parametric Change on Peltier Electrical Resistance

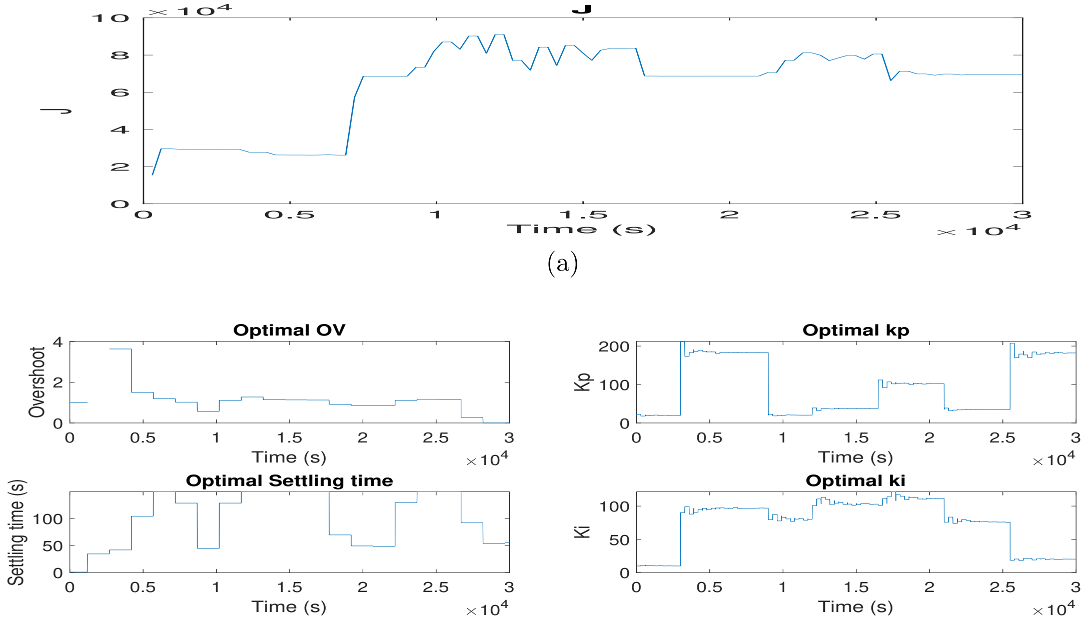


Figure 7.21: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution of the Peltier Thermal Digital Twin System Against Parametric Change on Peltier Electrical Resistance

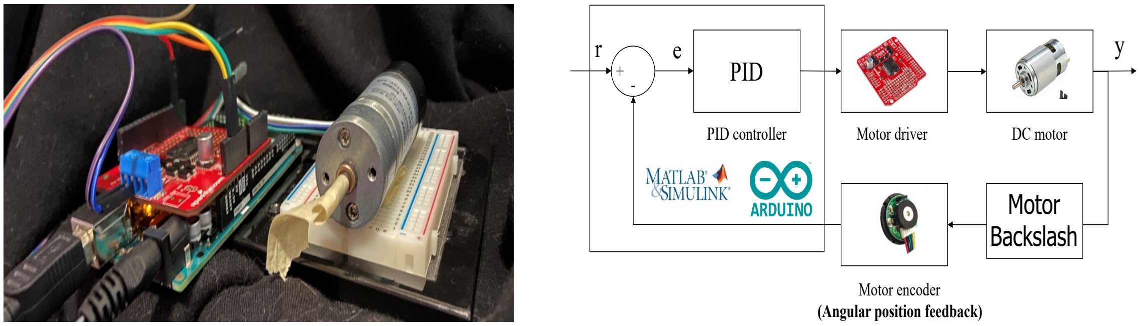
7.6.2 Case Study 2: Velocity and Position Control of A Smart Mechatronic System

In this section, the Digital Twin of a mechatronic system employed for position control is developed to introduce the self optimizing control as enabling capability to improve its closed-loop performance following the DT systematic development framework and the GCNM algorithm.

Digital Twin Modelling Using A Systematic Design Framework

Step 1: Target System Definition

The real-time velocity and position feedback motion control presented in Fig. 7.22 is employed as a case study. The system is composed of a DC motor, 12V, 600 RPM equipped with a Hall effect encoder, a dual motor controller for Arduino based on the L298 H-bridge, an Arduino Mega 2560 microcontroller that provides the control commands to the motor controller and the communications between the Digital Twin and the real system. A set of Simulink S-function have been developed to read encoder position and velocity real-time data. The position and velocity is controlled using a PID controller.



(a) (b)

Figure 7.22: DT Case Study: Smart Mechatronic System a) Physical System b) Closed-Loop Block Diagram

Step 2: System Documentation

From the target system definition, the system is composed of four elements, the DC motor, the Hall effect encoder, the dual motor controller and the Arduino Mega 2560. Table 7.7 shows the main characteristic of each component.

Table 7.7: Summary of DC Motion System Documentation

Component	Features
TSINY(TS-25GA37OH-10) DC motor	12V, 600RPM, 2.5W Quadrature encoder 460 CPR
Ardumoto Shield	dual motor controller up to 2A based on L298 H-bridge
Arduino Mega 2560	54 digital IØ,16 analog inputs SRAM 8KB, EEPROM 4KB
Hall effect encoder	460 CPR

Step 3: Multidomain Simulation

The system is composed of three simulation domains as shown in Fig. 7.23 corresponding to the digital, electrical, and mechanical domains. Matlab/Simulink and Simscape electrical and Mechanics are employed as multidomain simulation tools to replicate the physical laws of the system and the PID control law. The complete multiphysics simulation model is presented in Fig. 7.24. The Digital domain is composed of the PID control algorithm, the reference signal, and the control action to be applied as PWM. The practical implementation of this domain is performed using Hardware in the Loop simulation (HIL) with Matlab/Simulink. Besides, the electrical domain is composed of a controlled voltage source that provides the system power, a controlled PWM voltage that create the Pulse-Width Modulation (PWM) for position and velocity control of the motor. The H-Bridge operating in averaged mode delivers the power to the motors according to the PWM signal block. The simulation mode parameter value must be the same for the Controlled PWM Voltage and H-bridge blocks. If the REV port voltage is greater than the Reverse threshold voltage, then the output voltage polarity is reversed. The comparator block determines the turning sense of the motors. The DC Motor block contains the electrical and mechanical characteristics of the motor. Finally, the mechanical domain is composed of the ideal rotation motion sensor that converts an across variable measured between two mechanical rotational nodes into a control signal proportional to angular velocity or angle. The most relevant DC motor parameters based on the specs provided by the manufacturer [266, 292] are shown in the first column of Table 7.8. These parameters correspond to the armature inductance, armature resistance, back EMF constant, rotor damping coefficient, and rotor moment of inertia, which will be relevant to the behavioral matching.

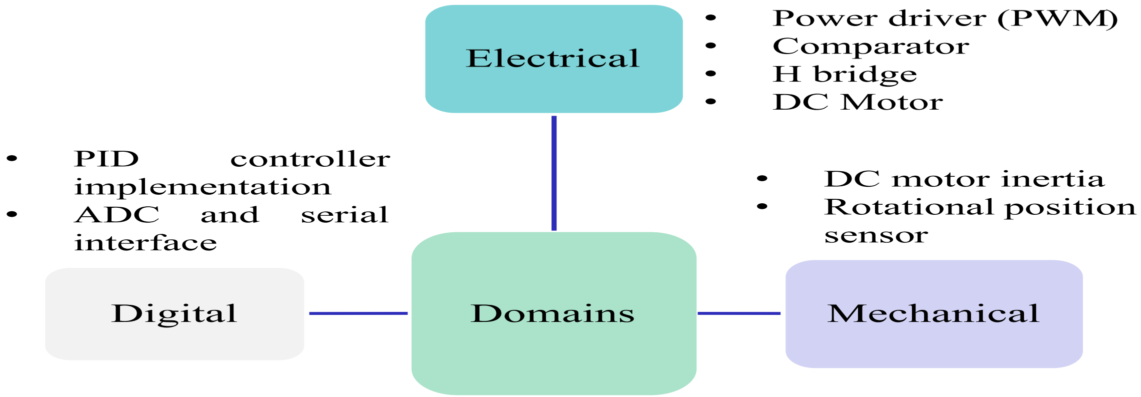


Figure 7.23: DT Case Study Simulation Domains: Digital, Electrical, and Mechanical

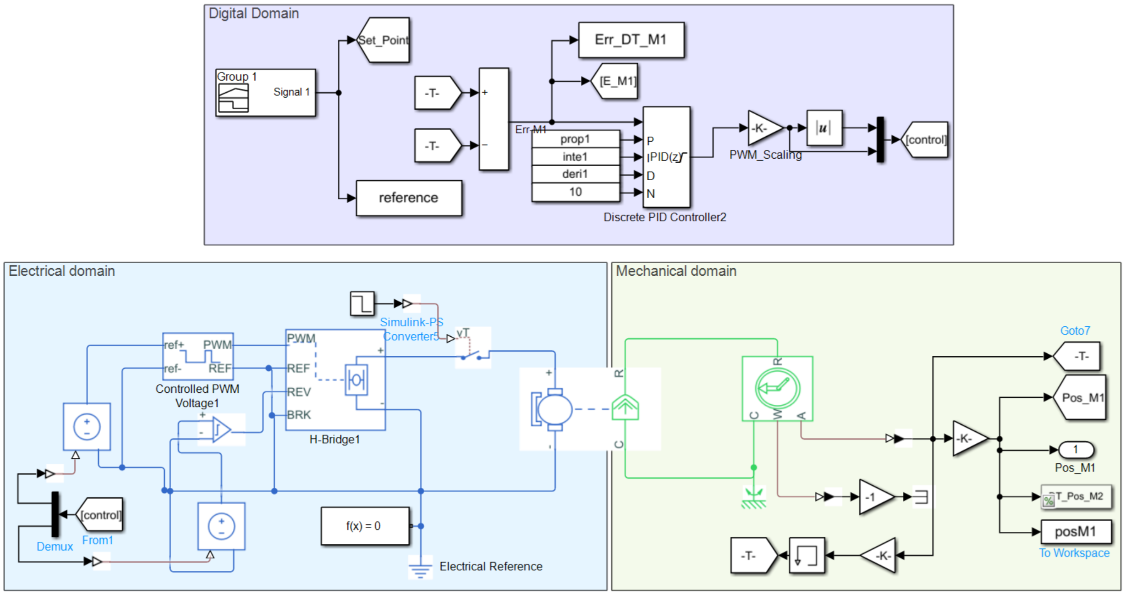


Figure 7.24: Smart Mechatronic System Multidomain Simulation Model

Step 4: Behavioral Matching

For the smart mechatronic system, the electrical and the mechanical domain parameters are the most relevant domains, which parameters like armature inductance, armature resistance, back EMF constant, rotor damping, and rotor moment of inertia were taken from [266, 292] with their values shown in Table 7.8.

For this system, the Simulink design optimization (SLDO) toolbox is used to determine the real parameters of the multiphysics model [293]. The SLDO contains several functions and tools that can be used to analyze and tune model parameters like sensitivity analysis, model fitness to real data employing algorithms as nonlinear least squares, gradient free optimization minimizing the error L_2 norm. Also, SLDO includes Monte Carlo Simulation and Design of Experiments to explore the influence of the estimated parameters in the model behavior and optimize the model response to satisfy time-domain or frequency-domain requirements like overshoot, phase margin or settling time.

Figure 7.25 shows the behavioral matching results from an open-loop experiment performed to update the multiphysics model with a new set of parameters. The results of the behavioral matching are presented in Table 7.8. As can be observed, the parameters obtained from the behavioral matching represent the system behavior properly close to the real data.

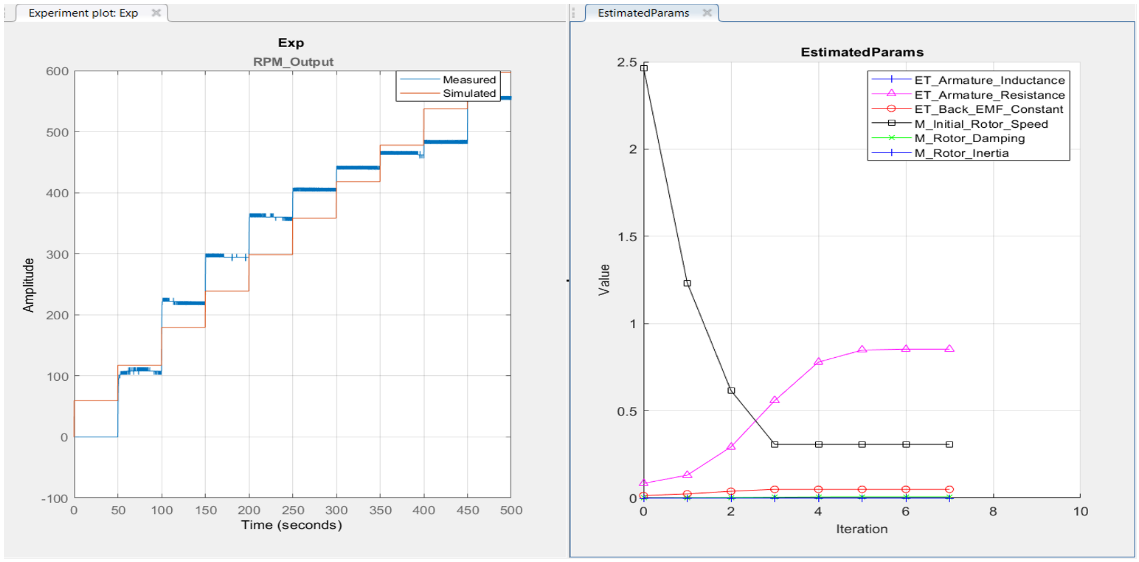


Figure 7.25: Behavioral Matching Using SLDO Optimization Toolbox

Table 7.8: DC Motor Parameters

Parameter	Original	Value from SLDO
Armature Inductance [H]	1.2577e-11	1.2304e-11
Armature Resistance [Ohm]	0.075922	0.091449
Back EMF constant [V/rpm]	0.0012462	0.041662
Rotor Damping [Nm/(rad/s)]	0.0004682	0.000405
Rotor Inertia [g cm^2]	1.8747e-13	1.8747e-13

Step 5: DT Validation and Deployment

A web application is created in Matlab using the Matlab App Designer Tool for the remote deployment of the DT [294]. Then it will be package using Matlab Compiler and then this entity will be hosted on a server using Matlab Web App Server (MWAS). It provides the infrastructure to host and share custom Matlab and Simulink simulation as interactive web apps. Thus, end-users can access and run the web app using a browser without installing additional software. It allows the user set the motor and PID controller as well as user monitoring if the system is following the established trajectory or if a fault has occurred due to a lack of power in the system. The details for the web server implementation as well as the Matlab/Simulink codes can be found at `https://www.theedgeai.com/dtandscebook`.

Digital Twin Performance Analysis

The Digital Twin performance is evaluated for position and velocity control tasks using a stepped and sinusoidal signals as velocity and position reference signals. Both tasks are controlled with a PID controller tuned by heuristic iterations based on the Digital Twin performance, which parameters are $k_p = 4$, $k_i = 0.5$, $k_d = 0.01$ and for the $k_p = 0.1$, $k_i = 1$, $k_d = 0.01$ for the position and velocity controllers. Figure 7.26 and Fig. 7.27 show the time responses and control actions of the Digital Twin of a DC motor system for velocity and position control tasks using PID controllers tuned by heuristic method. It can be observed that the Digital Twin represents properly the behaviors of the real asset in terms of the time responses of the position and velocity tasks. In the case of the control action, there is a difference for the position control on the Digital Twin due to the average mode execution required for the simulation of the electrical domain. However, the results shown by the Digital Twin are reliable to describe the physical asset behavior.

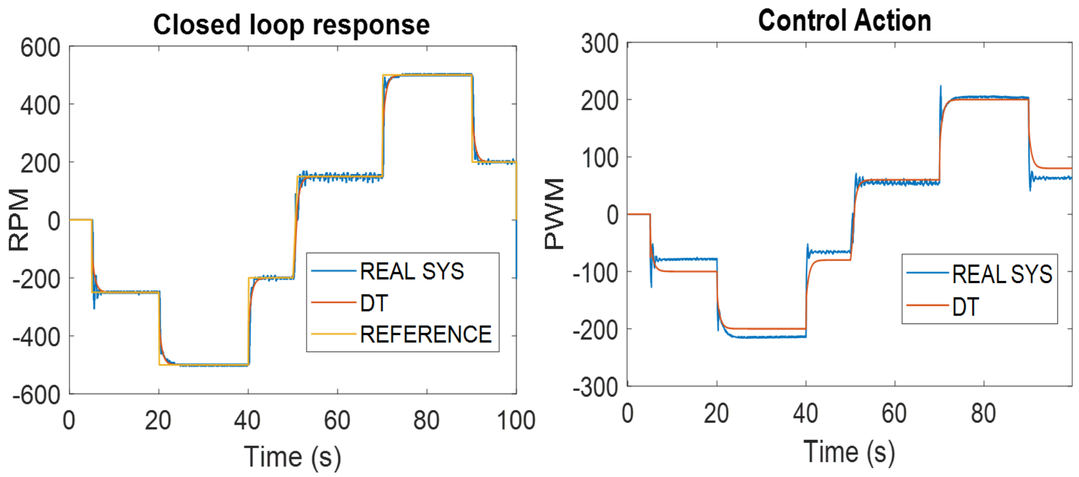


Figure 7.26: Control Performance Comparison of Smart Mechatronic System DT and Physical Asset

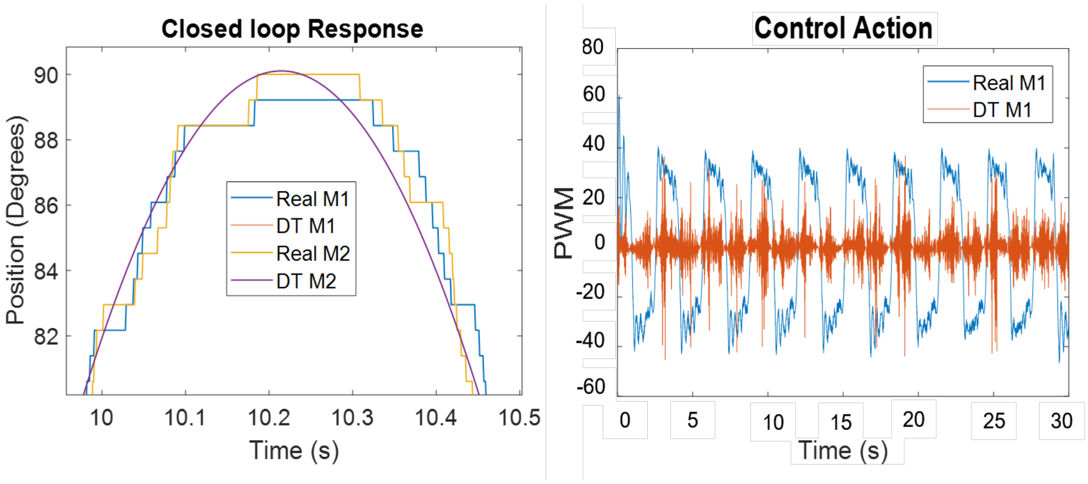


Figure 7.27: Position Control Performance Comparison of Smart Mechatronic system DT vs Physical Asset

Self Optimizing Control for Smart Mechatronic System Enabled by Digital Twin

The Digital Twin of the smart mechatronic system presented in the previous section is combined with the SOC architecture presented in Fig. 7.1 to improve the closed-loop control of the system on position control task. Figure 7.28 shows the

GCNM SOC controller combined with the Smart Mechatronic system and its Digital Twin.

Unlike the Peltier thermoelectric system, in this case, the performance of the Smart Mechatronic system DT is evaluated via simulation to determine reasonable limits for the optimization parameters (k_p, k_i, OV, T_s) of the SOC controller. Thus, the design specifications for the SOC PI controller are $0.01 \leq k_p \leq 1$, $0.01 \leq k_i \leq 0.3$, 10 reset buffer samples, $OV \leq 5\%$, $T_s \leq 1s$, a square reference signal with $2s$ period, and reset threshold $\epsilon = 0.001$. In this case, the initial conditions for $k_p,\ k_i$ are $k_{p0} = k_{i0} = 0.2$. The remaining parameters of the Digital Twin are the ones resulting from the behavioral matching as shown in Table7.8.

The performance of the SOC controller with the Digital Twin of the Smart Mechatronic system is shown in Fig. 7.29 and Fig. 7.30. As can be observed, the SOC controller is able to improve the closed-loop response performance of the system after 50 seconds. Likewise, it is important to notice that the real-time optimization is not affected by the presence of random noise due to the sensor noise. Also, it can be observed that the settling time and overshoot performance specifications are satisfied when the SOC PI terms reach the optimal value of $k_p = 0.6945$ and $k_i = 0.0636$. In the case of the cost function evolution, it can be observed that its value has small variations due to the fact of probabilistic restarts in the GCNM algorithm. Thus, we can say for the Smart Mechatronic system that using the Digital Twin in combination with the proposed SOC architecture allows the performance evaluation of the control strategy. It is effective when we have the system behavior under repetitive tasks, reaffirming the results in the previous sections obtained for the SOC of the Peltier thermoelectric system and the SOC benchmark test.

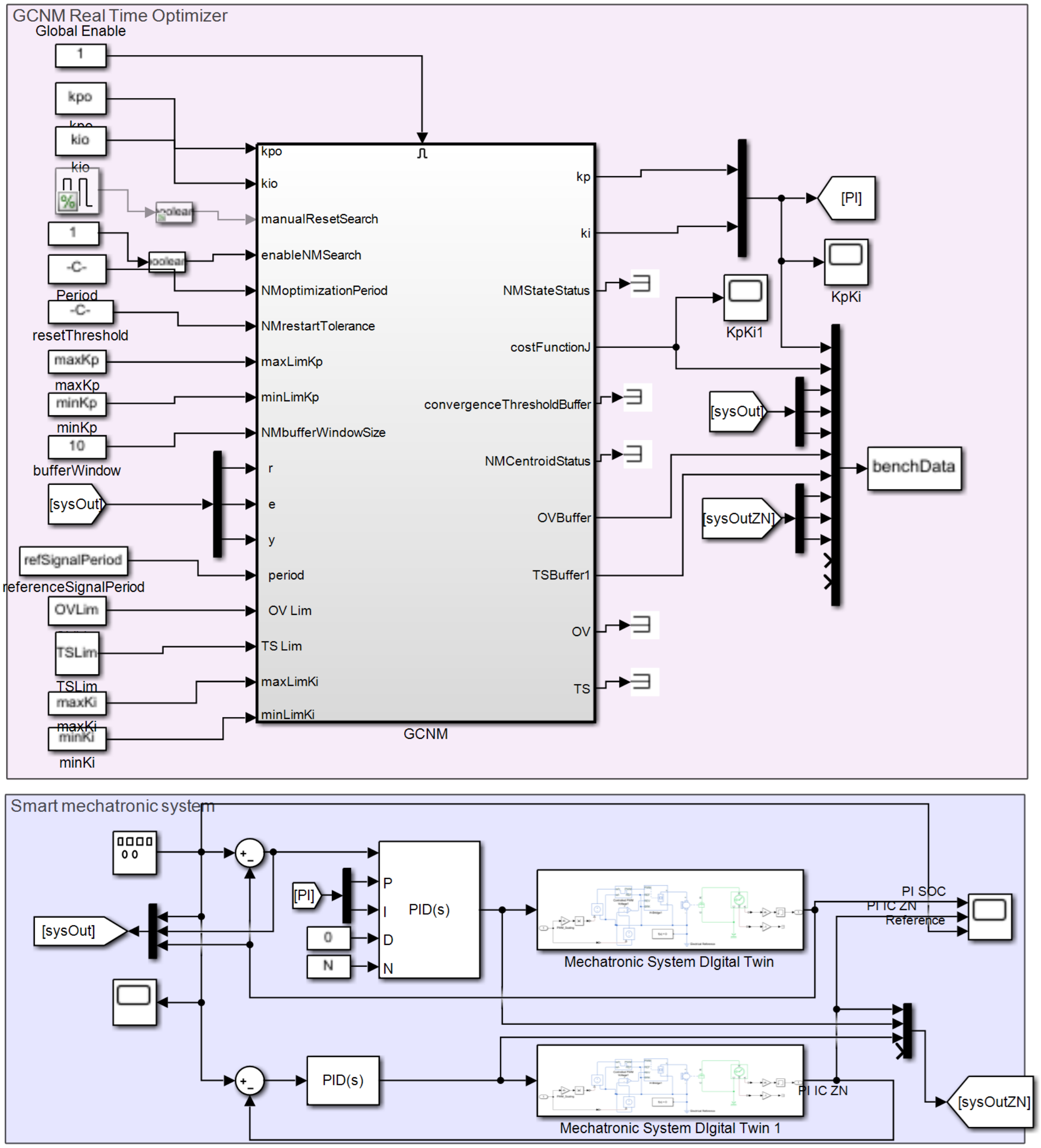


Figure 7.28: Self Optimizing PI Controller with the Smart Mechatronic System Digital Twin.

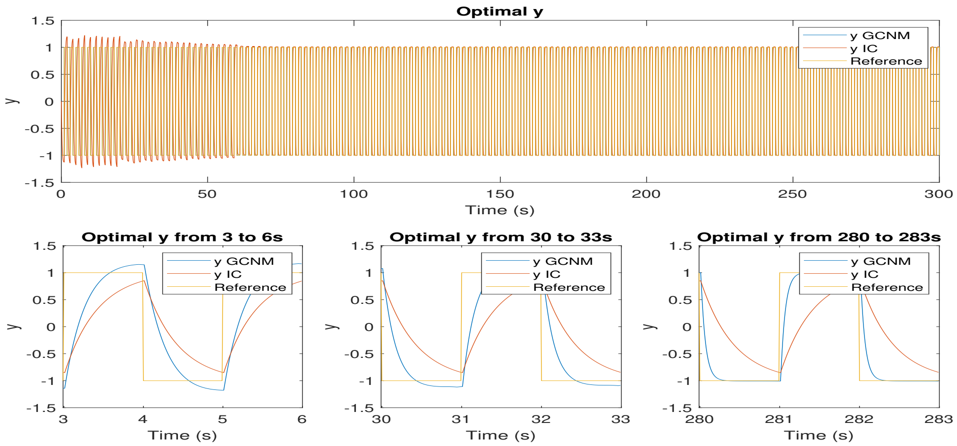
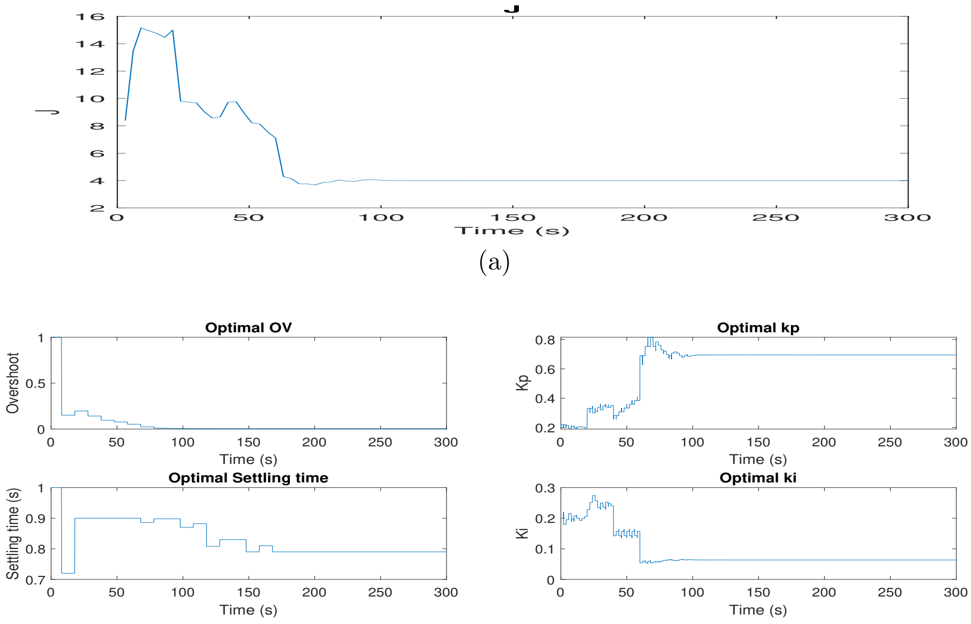


Figure 7.29: Closed-Loop System Output Evolution Using SOC Controller with the Smart Mechatronic System Digital Twin.



(a)

(b)

(c)

Figure 7.30: SOC Controller a) Cost Function b) Overshoot and Settling Time, and c) PI Gains Evolution for the Smart Mechatronic System Digital Twin.

7.7 Chapter Summary

This chapter introduced the notion of a smart system, smart control engineering and its relationship with the Digital Twin. Likewise, a self optimizing control architecture has been proposed to improve the closed-loop performance via online optimization for repetitive batch executions using the Globalized Constrained Nelder-Mead algorithm. A benchmark is built in Matlab/Simulink to evaluate the SOC controller performance for performance improvement of FOPDT systems under a normalized model with time-constant dominant, balanced, and delay-dominant configurations. Obtained results show that the SOC controller can improve the performance of the closed-loop control of the system under the testing conditions. Also, the SOC control benchmark was combined with the Digital Twin of the Peltier thermoelectric system employed in Chapters 5 and 6, showing that the SOC controller combined with Digital Twin can be very useful towards implementing the smart control engineering on real systems. Also, the SOC framework is applied for a Smart Mechatronic system for position and velocity control tasks extending the capabilities of SOC and DT for SCE . However, further analysis is required based on intensive simulation studies to analyze the global behavior and convergence rate of the SOC strategy due to the random probabilistic restart feature of the GCNM algorithm for different systems. Likewise, a study for other initial conditions for $k_p,\ k_i$ is required to evaluate the best possible performance of the SOC, which has shown a significant influence of these at each performed test. As future works, the implementation of the SOC algorithm in hardware-in-the-loop (HIL) configuration on the Peltier and the smart mechatronics systems are proposed as well as its extension to a high degree of freedom controllers like integer-order and fractional-order PID controllers as shown in [280, 295].

Chapter 8

IMPROVING SELF OPTIMIZING CONTROL APPLICATIONS USING DIGITAL TWIN, FRACTIONAL-ORDER STOCHASTICITY, AND PARALLEL COMPUTING

Industry 4.0 requires a new generation of control systems that ensure the closed-loop stability of the system and incorporate smartness in the control loop to keep the system operating under the desired performance given by an economical cost function by optimizing the control system parameters. As shown in Chapter 7, the Self Optimizing Control (SOC) can be adapted to this purpose by using derivative free online optimization algorithms for closed-control loop parameters or setpoint adjustment based on some desired specifications under some periodic task execution enabling cognizant and reflective capabilities on a closed-loop system.

On the other hand, it is important to notice that increasing the convergence speed of the SOC is crucial for real applications, especially for process with high dimensional optimization problems to be solved. For this reason, this Chapter introduces a set of novel characteristics to accelerate the convergence of the SOC framework presented in Chapter 7 employing fractional-order stochasticity and parallel computing techniques following the ACP approach combined with stochastic approximation algorithms.

8.1 An Accelerated Self Optimizing Control Framework for Smart Process Control Using Fractional Order Stochasticity

Fractional-order calculus has shown the capability of improving optimization processes by using more optimal randomness like levy flights, fractional Gaussian noise, or alpha-stable distributions, which are heavy tail distributions with Long-Range Dependence (LRD) properties. Applications of more optimal optimization using fractional order randomness includes improved Extremum Seeking control and Maximum Power Point Tracking [296, 297], metaheuristic optimization [281, 298], accelerated gradient descend [299] stochastic configuration networks [300], or more optimal consensus [301].

This chapter proposes a more optimal Self Optimizing Control (SOC) strategy employing fractional-order randomness to enhance the performance of derivative-free optimization algorithms to improve the response of closed-loop control systems according to a performance cost function. The Globalized Constrained Nelder Mead (GCNM) algorithm is employed as the gradient-free optimization method for the SOC Controller with a probabilistic restart feature to ensure the algorithm convergence when the system falls into a local minimum. In this case, the probabilistic restart is performed using fractional-order Gaussian noise with LRD behavior instead a normal distribution. The more optimal SOC is tested in the simulation benchmark presented in Chapter 7. The SOC controller is evaluated on three scenarios for the FOPDT, a time constant dominated, balanced, and delay dominated system, with a periodic reference signal. Likewise, the performance indices are calculated for the benchmark to obtain a quantitative performance comparison against the standard SOC. The benchmark test is performed for integer-order Gaussian noise and fractional order Gaussian noise with Hurst exponent $0.1 < H < 0.9$.

8.1.1 Fractional-Order Gaussian Noise for Probabilistic Restart

As stated before, the GCNM algorithm uses a normal distribution to pick up the new initial conditions after each probabilistic restart. However, this process can be performed by using a fractional-order Gaussian distribution to increase the convergence speed of the algorithm. The fractional-order Gaussian noise can be represented as the change in Brownian motion step defined by the Riemann-Liouville fractional integral (8.1), where $dB(s)$ is the general definition of white noise, $\Gamma(\cdot)$ is the gamma function, and H is the Hurst exponent which indicates the LRD property of the random disturbance signal [302, 303]. According to the value of H, the fractional-order randomness can represent a Brownian motion if $H = 0.5$, positively correlated if $0.5 < H < 1$ and negatively correlated if $0 < H < 0.5$.

$$B_H(t) = \frac{1}{\Gamma(H+1/2)} \int_0^t (t-s)^{H-0.5} dB(s). \tag{8.1}$$

8.1.2 Proposed Optimal Randomness Testing

The optimal randomness evaluation is performed using the normalized FOPDT system (7.6) for three different delay values $L = 0.1,\ 1,\ 10$ corresponding to the time constant dominated, balanced time, and delay time dominated behaviors of the system. For each delay value, the Hurst exponent of the fractional-order Gaussian noise is evaluated as $H = 0.1 : 0.1 : 0.9$, where $H = 0.5$ corresponds to the Normal distribution and the remaining H values to the fractional-order noise. A total of 50 evaluations are performed for each value of H for the different delay values. Thus, the average value for each performance indicator will be considered as the reference performance index for the system.

For each scenario, a Modified Ziegler Nichols (ZNM) controller is designed for the FOPDT system [278], which are the initial condition for the GCNM algorithm to evaluate how much the SOC can improve the closed-loop system performance, corresponding to $k_p = 12, k_i = 6$ for $L = 0.1$, $k_p = 1.2, k_i = 0.6$ for $L = 1$ and $k_p = 0.12, k_i = 0.06$ for $L = 10$. A square signal with equally high and low state time is employed as a reference with a period of 300s for all the tests. The constraints values and other configuration parameters for the benchmark are shown in Table 8.1. Notice that the maximum and minimum constraint values of K_p and K_i change from each test to ensure the stability of the closed-loop system. For all the benchmark tests cost function weights are $W_1 = 1$, $W_2 = 0.1$, $W_3 = 0.1$. As example, the histogram of a fractional-order Gaussian randomness series for $H = 0.3,\ 0.5$ for $L = 0.1$ is shown in Fig. 8.1, which are positive to coincide with PI controller gains.

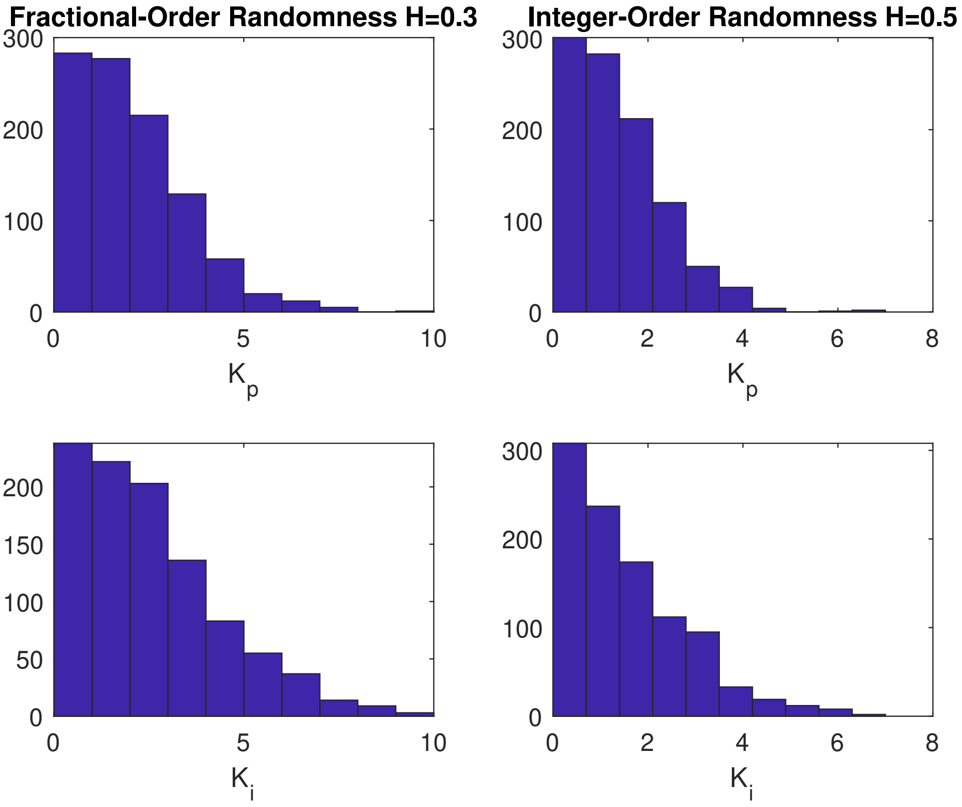


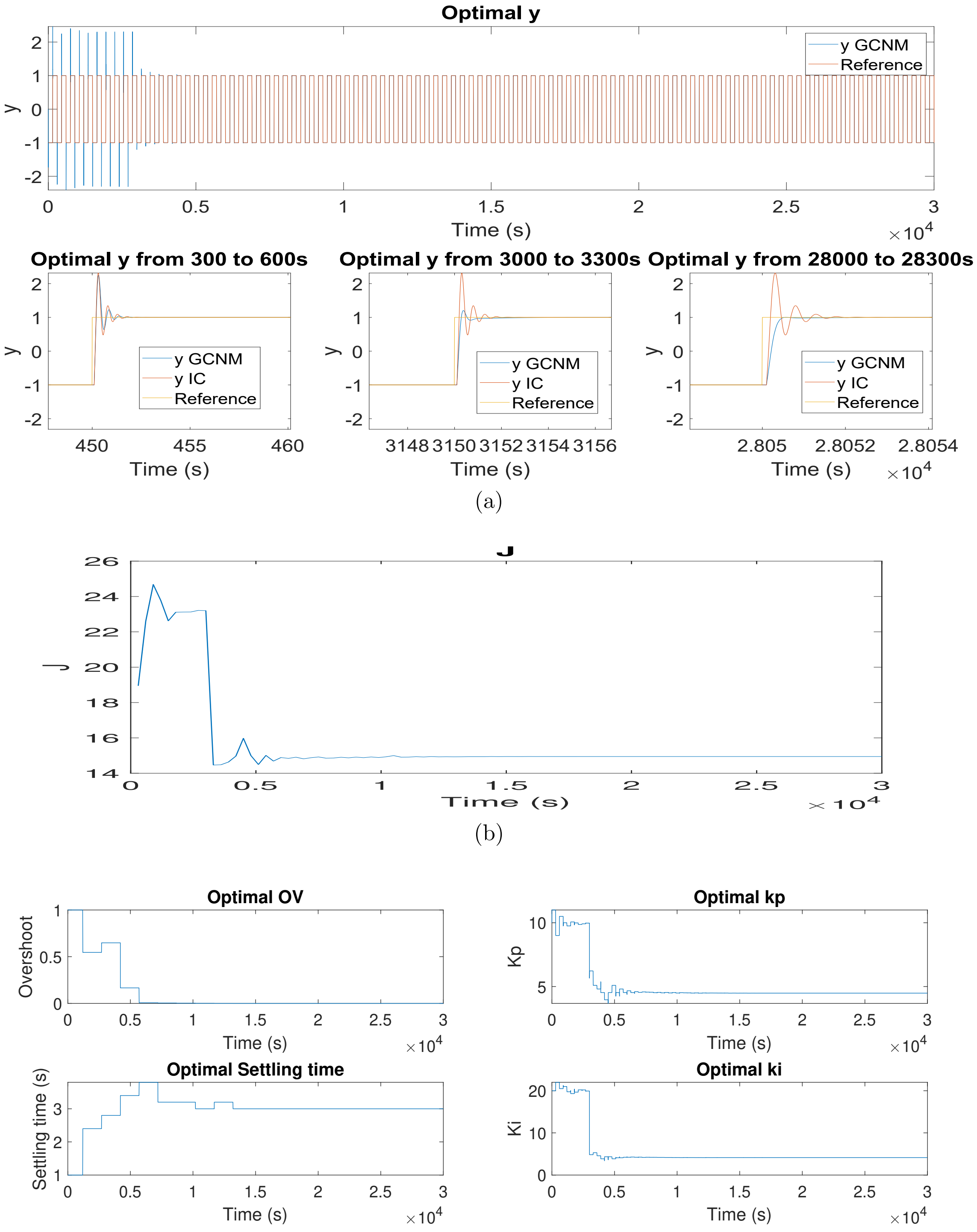
Figure 8.1: Fractional and Integer Order Randomness Sample Employed for the SOC Controller with $H = 0.3,\ 0.5$

Table 8.1: SOC Benchmark Configuration Parameters

	Test 1 (L=0.1)		Test 2 (L=1)		Test 3 (L=10)	
Parameter	min	max	min	max	min	max
k_p	0.01	10	0.01	1	0.045	0.1
k_i	0.01	5	0.01	0.5	0.045	0.1
Reset buffer samples	10					
Settling time	30 s		30 s		70 s	
OV (%)	5%		5%		5%	
Reset threshold ϵ	1e-3					
Reference signal period (s)	300					

8.1.3 Obtained Results for SOC Benchmark

The SOC controller is tested for the three scenarios defined in the previous section with normal and fractional-order Gaussian noise. As example, Fig. 8.2 shows the performance of the SOC controller for $L = 0.1$ with a Hurst exponent $H = 0.5$ that is equivalent to the normal distribution. As can be observed, the PI controller begins with the values provided by the ZN tuning, and then after each execution cycle the GCNM perform a real-time optimization that search for the optimal values of the PI controller gains until the desired performance specifications are satisfied according to the cost function. Next, a detailed analysis of the iterative benchmark is done for each of the proposed scenarios based on the overall convergence time, the closed-loop settling time and overshoot.



(a)

(b)

(c)

(d)

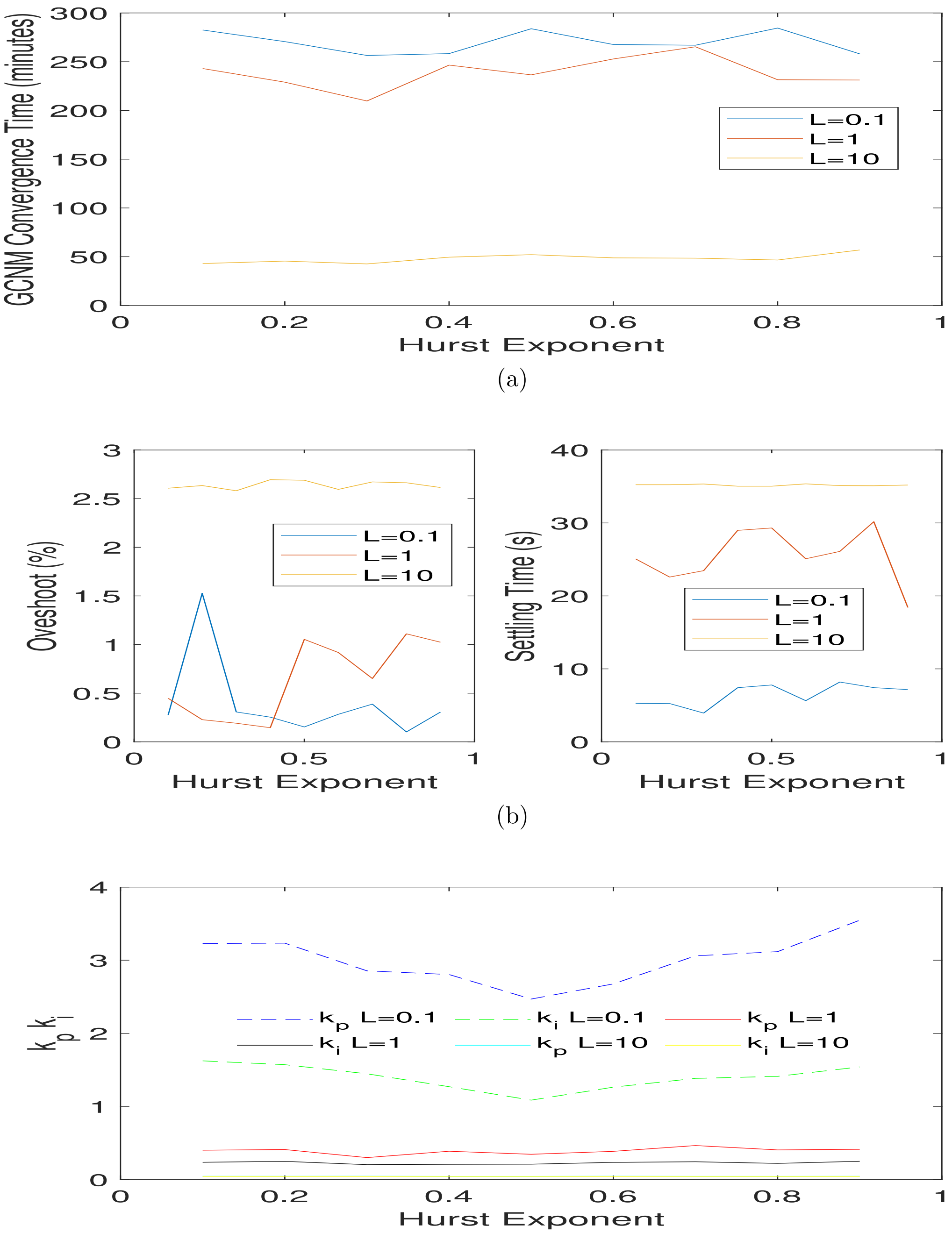
Figure 8.2: SOC Controller a) Time Response b) Cost Function C) Overshoot, Settling Time, and d) PI Gains Evolution for $L = 0.1$ with $H = 0.5$ (Gaussian Randomness).

The average performance of the SOC controller with GCNM algorithm for $L = 0.1, 1, 10$ regarding to the mean overall convergence time, closed-loop settling time and overshoot of the GCNM algorithm after 50 iterations is shown in Fig. 8.3 for the different Hurst exponents H. As can be observed the GCNM convergence time is reduced in a 10% when the fractional-order randomness is employed, reaching its minimum value at $H = 0.3$, indicating that a negative LRD in this case improve the algorithm convergence time. Likewise, Fig. 8.3 shows the closed-loop settling time and Overshoot of the system, which indicates that for all H the optimization conditions are satisfied, and at $H = 0.3$ has the lowest values for settling time.

In the case of the balanced system $L = 1$, the mean overall convergence of the GCNM convergence time is reduced in a 23% when the fractional-order randomness is employed, reaching its minimum value at $H = 0.3$. Regarding to the Overshoot and Settling time, the minimum values are reached at $= 0.4$ and $H = 0.9$ respectively. Although, in all the cases for the fractional order randomness the performance specifications are satisfied.

Besides, for the delay dominated system $L = 10$, the mean Overall convergence of the GCNM convergence time is reduced in a 19% with fractional-order randomness reaching its minimum value at $H = 0.3$. In the case of the Overshoot and Settling time, the minimum values are reached at $= 0.3$ and $H = 0.4$ respectively, again satisfying the performance specifications for all the Hurst exponents. For this particular case, the overall convergence time of the algorithm is reduced due to a strong lower boundary on the integral gain required to keep the system stability caused by the bigger delay of the system.

Finally, the average RMS, RMSE, ISE, and IAE values for the three tests $L = 0.1,\ 1,\ 10$ after 50 repetitions are similar with integer or fractional order randomness, indicating that the optimization process is able to achieve a similar performance for the different cases according to the economic cost function (7.3). Therefore, and based on the obtained results, we can say that the fractional-order randomness is able to reduce the overall convergence time of the SOC controller in the range of 10% to 23% compared with the integer-order Gaussian noise. It means that the fractional-order randomness with negative LRD can improve the Self Optimizing Control performance by accelerating the convergence of the optimization process without modifying the search algorithm or the SOC controller structure. So that, the fractional-order randomness can be considered as a suitable alternative to improve the performance not only of real time SOC controllers but also any searching process.



(a)

(b)

(c)

Figure 8.3: SOC Controller a) Overall Convergence Time b) Overshoot, settling Time, and c) Final PI Gains After 50 Iterations for $L = 0.1, 1, 10$.

8.2 Parallel Self Optimizing Control Framework for Digital Twin Enabled Smart Control Engineering

On the previous section, the fractional-order randomness with negative LRD shows promising results for improving the SOC controller convergence speed employing the GCNM optimization algorithm.

However, a SOC problem may take a long time before reaching an acceptable solution based on an performance cost function due to the nature of the optimization problem and the system dynamics like in thermal processes. In that case, parallel intelligence and control as a novel paradigm that looks for the integration of complex systems under the ACP approach (Analysis, Control, Parallel Execution) to improve the system performance can be integrated into the SOC to accelerate the system learning, and optimization [235, 304, 305].

In order to introduce parallel capabilities to SOC, a different optimization algorithm is required to handle the presence of multiple simultaneous executions to enhance the system performance and optimization speed. In that sense, the Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm can be used [306], which is a derivative-free optimization technique that performs a stochastic approximation of the function gradient using only two measurements of the cost function. It has applications in controls for offline controllers tuning [307, 308] and feature extraction in machine learning [309, 310]. However, the most important property of SPSA is that it can be implemented for parallel execution [311, 312]. Thus, SPSA can be used as bridge between virtual and real systems to enhance SOC performance by simultaneous evaluation of multiple models of a system represented by instances of Digital Twins.

In this section, a parallel Self Optimizing Control framework enabled by Digital Twins and the SPSA algorithm for the control of a stable closed-loop system based on an economic cost function is presented. The framework uses the parallel implementation of the SPSA algorithm supported by a pool of Digital Twins of the real system to increase the optimization speed. Likewise, the SPSA handle the interaction between a physical system and a parallel Digital Twin, monitoring the closed-loop system behavior and updating the controller parameters according to an economic performance cost function. The Digital Twin for the Peltier system in Chapter 7 is employed as a case study to evaluate the parallel SPSA framework. Two tests are performed for the system, one using only the real and virtual Digital Twin, and another with the support of the Digital Twin pool to leverage parallel capabilities of the SPSA.

8.2.1 Parallel Self Optimizing Control Framework

The parallel SOC framework is shown in Fig. 8.4. As can be observed, a Parallel control architecture is employed, differentiating the real domain and virtual domain where the Digital Twin is located. In the real domain, a closed-loop system operates using a controller $C(\theta)$ which tuning parameters θ ensure the system stability. Likewise, In the virtual domain, the Parallel Digital Twin 1 replicates the configuration of the physical system using a multiphysics model of the process.

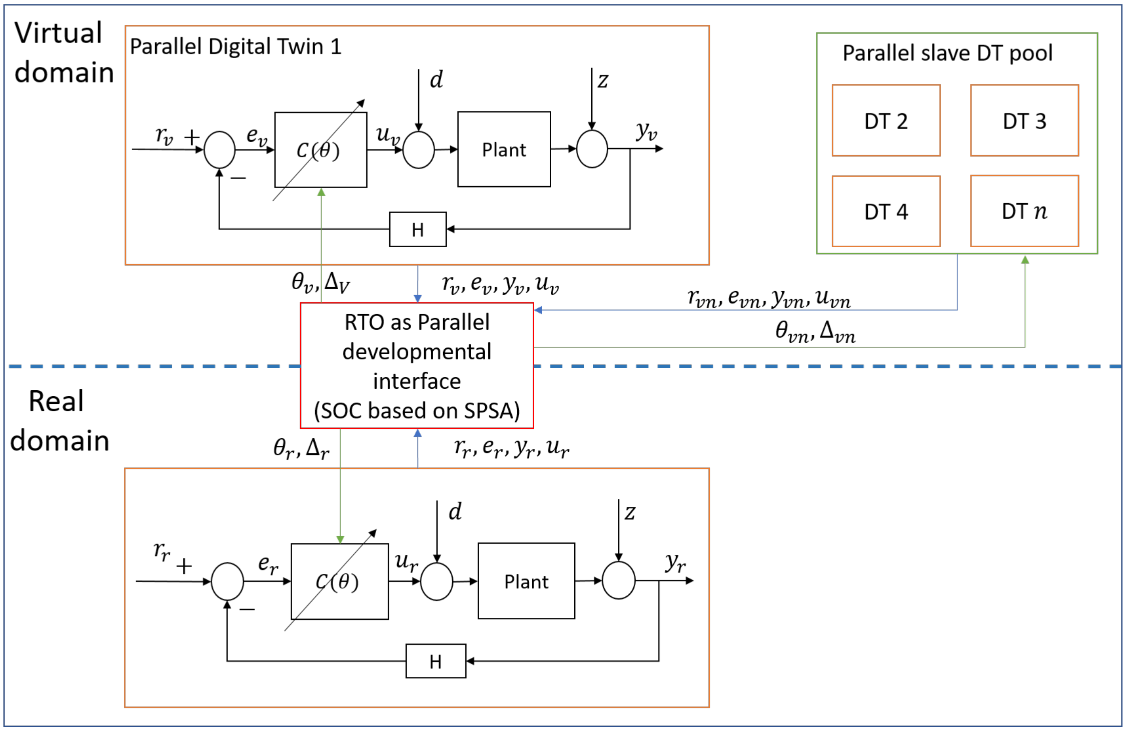


Figure 8.4: Parallel Self Optimizing Control Framework

The goal of the parallel SOC framework is to continuously update the controller parameters θ based on an economic cost function, which considers the current performance of the closed-loop system. In this paper, the economic cost function employed is given by (8.2), where T_s is the system settling time, OV is the overshoot percentage, θ is a vector with the controller parameters, $W_{1,2,3}$ are the weights for the Overshoot, Settling time and the Integral Square Error index respectively.

$$\min_{\theta \in \mathbb{R}} J = W_1 \; OV(\theta) + W_2 \; T_s(\theta) + W_3 \int_0^t e(t,\theta)^2 \, \mathrm{d}t. \tag{8.2}$$

The Simultaneous Perturbation Stochastic Approximation optimization algorithm is employed to find the optimal values of the controller tuning parameters θ. After each optimization step, executed in a different and higher timescale than the process closed-loop control, the result of the SPSA algorithm is updated to the Virtual and Real domains enabling a simultaneous interaction between domains.

Considering that the SPSA algorithm can be executed in parallel, a pool of Digital Twins can be enabled in the virtual domain as shown in Fig. 8.4 to improve the convergence of the SOC. These Digital Twins act as slaves of the parallel system (Real system and Digital Twin in the virtual system DT#1), evaluating one of the multiple simultaneous perturbations required for the parallel SPSA optimization at each iteration, increasing the convergence speed of the algorithm. Thus, the SOC control using SPSA acts as an integrating rule for the parallel system.

8.2.2 Parallel SPSA Algorithm

SPSA is a stochastic optimization algorithm proposed by Spall [306], which considers the following optimization problem

$$\arg\min_{x\in\mathbb{R}^n} \ f(x), \tag{8.3}$$

that uses the recursive form of a general Stochastic Approximation algorithm:

$$x_{k+1} = x_k - a_k \bar{g}_k(x_k), \tag{8.4}$$

where x_k represents the estimate of x at the k-th iteration, where a_k is a sequence of positive scalar coefficients. So the approximation of the gradient at x_k is

$$\bar{g}_k(x_k) = \begin{bmatrix} \frac{f(x_k+c_k\Delta_k)-f(x_k-c_k\Delta_k)}{2c_k\Delta_{k1}} \\ \frac{f(x_k+c_k\Delta_k)-f(x_k-c_k\Delta_k)}{2c_k\Delta_{k2}} \\ \vdots \\ \frac{f(x_k+c_k\Delta_k)-f(x_k-c_k\Delta_k)}{2c_k\Delta_{kn}} \end{bmatrix}, \tag{8.5}$$

where n is the size of the input x, $\Delta_k = [\Delta_{k1}, \Delta_{k2}, \ldots, \Delta_{kn}]$ are the elements of the random perturbation vector Δ_k generated using a sub Bernoulli distribution, which are assumed to be independent and symmetrically distributed around zero, c_k is a positive scalar that change its value per each iteration, $f(x_k + c_k\Delta_k)$, $f(x_k - c_k\Delta_k)$ are the cost function values evaluated with a different sign of the perturbation. According to [306], the values of a_k and c_k are given by (8.7), where $A,\ a,\ c > 0$, using $\alpha = 0.602$, $\gamma = 0.101$ as suggested by [306].

$$a_k = \frac{a}{(k+1+A)^\alpha} \tag{8.6}$$

$$c_k = \frac{c}{(k+1)^\gamma}. \tag{8.7}$$

The traditional SPSA [306] calculates for each iteration k the simultaneous perturbation vector $\Delta_k n$ and performs two evaluations $f(x_k + c_k\Delta_k)$, $f(x_k - c_k\Delta_k)$ of the cost function (8.5) to estimate the gradient and update the optimization parameters. However, the parallel implementation proposed by [312] uses i slave processes, each one with its own perturbation vector Δ_k to increase the number of cost function evaluations per iteration. Thus, for each i process, the values of $f(x_k + c_k\Delta_k)$ and $f(x_k - c_k\Delta_k)$ are calculated. These gradients are combined with the one obtained by the master process to find the new direction of the gradient using (8.8), where $u(k-1,i)$ is a subscript that means the i-th process in the $k-1$ iteration. Thus, the next system input parameters x_{k+1} are calculated. Algorithm 2 summarize the process of the parallel SPSA algorithm.

$$d_k = \bar{g}_k + \frac{\bar{g}_k^T \bar{g}_{u_{(k-1,i)}}}{||\bar{g}_{u(k-1,i)}||^2} \bar{g}_{u_{(k-1,i)}} \tag{8.8}$$

$$x_{k+1} = x_k - a_k d_k. \tag{8.9}$$

Algorithm 2: Parallel SPSA Algorithm [312]

```
1 Input $a, c, A, \alpha, \gamma$
2 Output x
3 Initialization : Initialize the SOC parallel framework and choose
  $a, c, A, \alpha, \gamma$
4 Generate the Simultaneous perturbation vector $\Delta_k$ for each process $i$.
5 Calculate the approximated gradient using (8.5) $\Delta_k$ for each process $i$.
6 Choice of the combined gradient direction based on the norm
  $||\bar{g}_k(k-1,i)||^2$ using (8.8).
7 Update $x_k$ applying (8.9)
8 Repeat 1 to 5 during $m$ iterations
9 return $x$
```

8.2.3 Parallel SOC Framework Evaluation

The parallel SOC framework is employed to optimize the proportional and integral gains of the PI controller (7.1) based on the cost function (8.2). For this purpose, two tests are proposed. The first test uses only the parallel interaction between the real system and the parallel Digital Twin 1 to optimize the values of the closed-loop PI controller employed in the system. Besides, the second test uses a pool of five Digital Twins of the system to accelerate the SPSA algorithm simultaneous perturbation with an independent perturbation vector Δ_k for each DT. Both tests are evaluated for a total of 200 iterations with the parameters $a = 60.17$, $\alpha = 0.602$, $\gamma = 0.101$, $c = 1.9$, and $W_1 = 1$, $W_2 = 0.1$, $W_3 = 1e-3$ for the SOC cost function (8.2) weights. Likewise, the initial conditions for the SPSA algorithm are given as a set of K_p, K_i that make the system stable obtained with the Ziegler-Nichols method [278]. For this reason, a First Order Plus Dead Time model of the system is identified using stepped inputs, which is given by (8.10), resulting in the initial values for K_p and K_i of $K_p = 10.3$,$K_i = 3.32$.

$$P(s) = \frac{2.7}{31.42s + 1}e^{-1.004s} . \tag{8.10}$$

The cost function evaluation, the controller parameters evolution, and the time response of the optimized controllers for tests 1 and 2 are shown in Fig. 8.5 and Fig. 8.6. As can be observed, the parallel SOC with SPSA converges in both scenarios, with a convergence time of 180 iterations in test 1 (with no DT pool), and 60 iterations when the DT pool is used. Likewise, the time response shows that the PI controller shows an improved performance after the self-optimization compared with the initial condition.

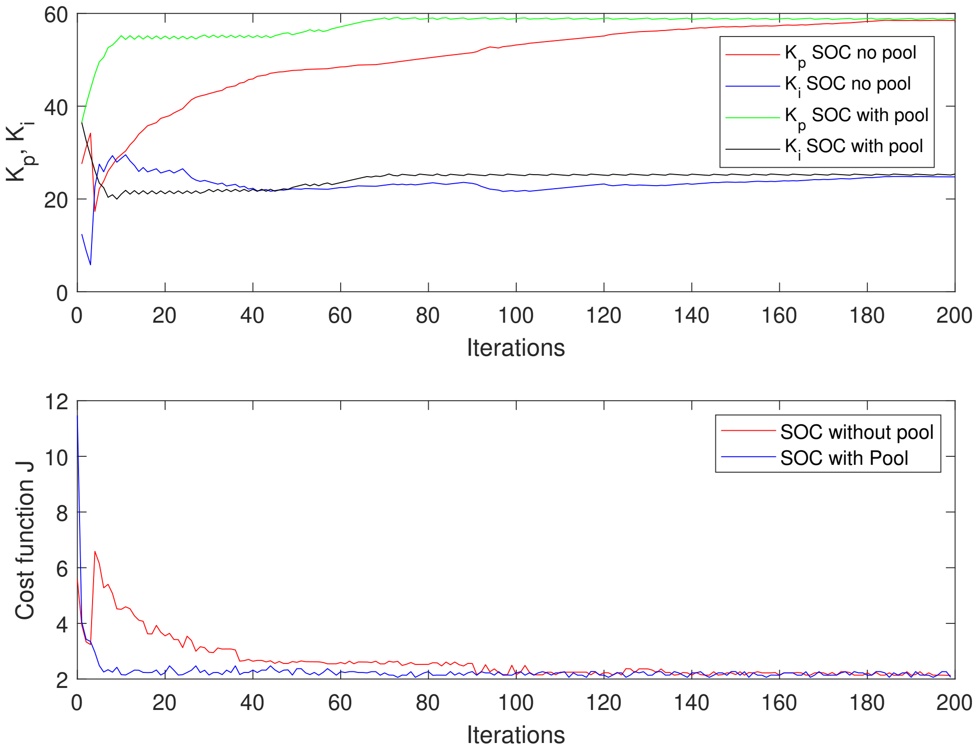


Figure 8.5: Parallel SOC test with real system and one mirror DT

Notice that the SOC control has been performed using the repetitive square reference signal shown in Fig. 8.6. Based on the obtained results, we can say that the parallel SOC framework with a Digital Twin pool can improve the closed-loop system response using an economical cost function based on the real-time updated system performance for each period.

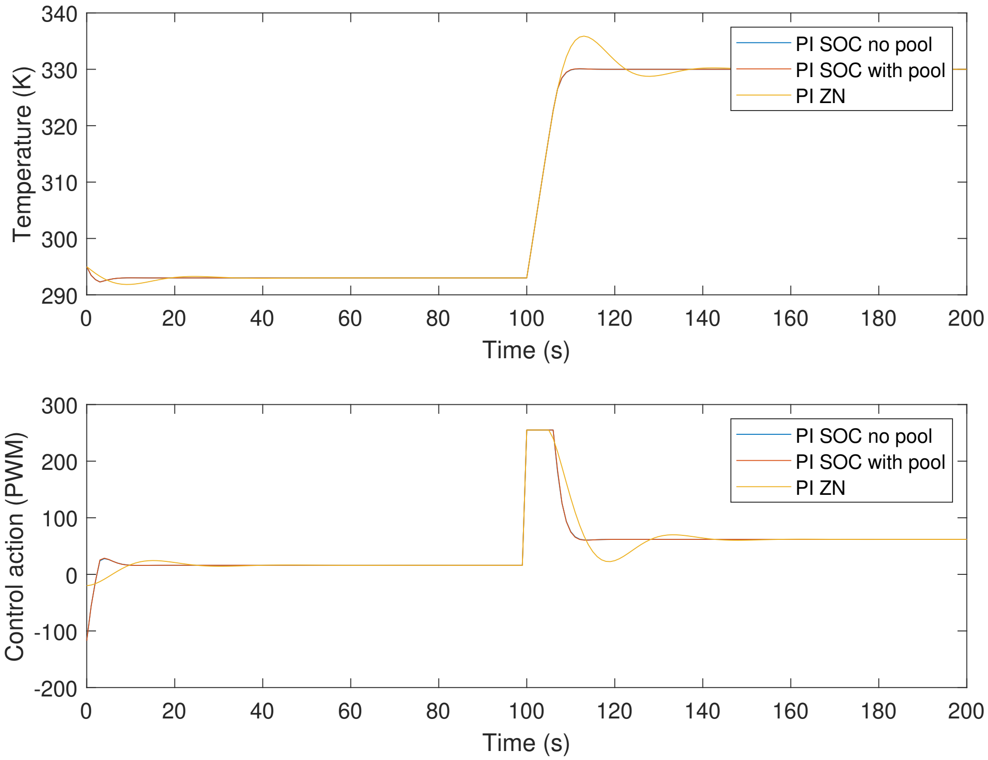


Figure 8.6: Parallel SOC test with real system, mirror DT and 10 DT slaves on pool

8.3 Chapter Summary

This chapter introduced two acceleration methods for SOC control employing fractional-order randomness for the GCNM optimization algorithm and a parallel computing architecture supported by Digital Twin and the parallel SPSA algorithm. The Digital Twin of a Peltier thermoelectric system has been employed as a case study for the framework in both cases. For the fractional-order randomness, obtained results shown that SOC convergence time is reduced by using fractional-order randomness over the classic Gaussian noise at the same time that the optimization constrains are satisfied by using negative LRD to reach a better convergence time. Likewise, The obtained results for the parallel SOC control show a significantly improvement the system closed-loop performance, reducing the convergence time using parallel SPSA optimization supported by multiple instances of Digital Twin. Thus, the fractional-order randomness as well as the parallel SOC framework are suitable acceleration mechanisms for the implementation of smart control systems. As future works, the convergence and stability analysis under accelerated configurations, its practical implementation using Hardware in the loop configuration, and its application for control problems with tens or hundreds of control variables to be tuned is proposed.

Chapter 9

CONCLUSIONS AND FUTURE WORKS

This dissertation presented the smart control engineering enabled by digital twins to make control systems smarter than ever following the characteristics of cognizant, reflective, taskable, knowledge rich and ethical. The smart control engineering is supported by three core technologies, industrial artificial intelligence (IAI), digital twin (DT), and self optimizing control (SOC), which are discussed along the dissertation with practical examples. In the case of IAI, the faultface methodology was developed leveraging face recognition technology for fault detection on mechanical components. Also, a methodological framework for using digital twin on control applications has been developed and tested with thermal systems. Several enabling capabilities derives from the use digital twin including fault detection and prognosis for thermal systems. A SOC control framework was developed based on real time optimization algorithms, monitoring the closed-loop performance of a control system in a supervisory layer that adjust the control system parameters. A new optimization method called globalized constrained Nelder-Mead has been designed and validated for the SOC control. Also, a theoretical convergence analysis of the SOC control framework was performed using proabilistic theory and "frame based method". A test benchmark and two case studies with digital twin have been designed to validate the performance of the SOC control method. Likewise, two acceleration alternatives for the SOC controller have been proposed, employing fractional-order randomness and parallel computing. The obtained results shows that using these acceleration methods improve the convergence speed of the SOC controller. The results presented in this thesis are reproducible and can be accessed online at `https://www.theedgeai.com/dtandscebook`.

Based on the obtained results, the smart control engineering powered by IAI, DT, and SOC provides a path towards introducing smartness not only on classic closed-control systems but also it can be extended into more complex configurations, with more constraints and complex dynamics.

This work serves as a starting point in the road of smart control engineering and there are several potential research topics to be explored. In the following sections, we outline briefly some of the exciting new ideas for future research exploration on DT and SCE.

9.1 Digital Twin Multi-Model Assessment

In the process of behavioral matching for Digital Twin applications, the data amount, quality, as well as its preprocessing employed to match the DT behavior with the real system will derive in an estimation uncertainty of the DT relevant parameters. In this book, an approach was presented on chapter5, where the multimodel assessment employs classic selection criteria like the Akaike Information Criteria or Minimum Description Length. These metrics rely on the maximum likelihood estimation based on non-parametric models of the Digital Twin analyzed like ARMA, ARX, or Box Jenkins. Likewise, the selection of the best suitable model relies on the analysis of several information metrics with the ν-gap metric to evaluate the system closed loop stability.

However, for a more complex system with higher dimensions Digital Twins, novel data-driven metrics are required to analyze the Digital Twin behavior and quantify the uncertainty on the model parameters based on numeric estimations of the Hessian matrices without relies on a known model.

In that sense, optimization techniques like the Simultaneous Perturbation Stochastic Approximation (SPSA) [306] can provide support for second order derivative estimation that leads to data-driven Hessian of the DT in terms of the critical parameters of the system. Thus, if the residuals during the optimization process are Independent and Identically Distributed (IDD) and follows a normal distribution $N(0, \Sigma)$, information theory tools like the Fisher information matrices can be computed to determine the information quality employing a Data Driven approach for the system.

9.2 Digital Twin Standardization and Interoperability

One of the main advantages of Digital Twin is leveraging the existing multiphysic and modelling tools to create accurate representations of complex physical assets. Usually, inside an industrial environment, a physical asset (machinery, vehicles, planes, chemical reactors) is composed of several components provided by different vendors, which are modelled employing multiple software tools that cannot easily perform inter-operation, integration, and communication. Examples of this situation can be found on aeronautic, semiconductor, oil and gas or communications industries [313].

Thus, the develop of Digital Twin applications requires an abstraction layer where the different models developed for each component and different vendors can interact to replicate the system behavior within a reasonable time. Indeed, it also requires to set standardization rules and norms to develop Digital Twins able to intercommunicate each other.

For this reason, the International Organization for Standarization (ISO) released in 2021 the ISO 23247-2:2021 Automation systems and integration — Digital twin framework for manufacturing for the integration of Digital Twin into manufacturing processes [314, 315].

Likewise, other organizations like the National Institute of Standards and Technology (NIST) released the Considerations for Digital Twin Technology and Emerging Standards defining rules for instrumentation, sensors, Digital Twin interfaces, and cybersecurity [316].

In the case of digital twin and smart control engineering, providing standard conditions for the sensing, actuation, simulation, and controls defines research direction into plug and play smart controllers that can be attached on edge devices on top of any industrial process, perform data acquisition tasks to infer data-driven models as Digital Twin that can be combined with the SCE framework, improving the closed-loop control execution of the system.

9.3 Convergence, Stability, Monotonicity, and Globalness Analysis of Self Optimizing Control Algorithms

The SOC framework presented in this dissertation allows the improvement of the closed-loop behavior performance of a system. In the framework, the Digital Twin is employed to adjust and validate the SOC strategy proposed based on a batch execution of the process. In this case, the optimization strategy relies on a derivative free algorithm like the Globalized Constrained Nelder Mead which convergence and stability is validated via numerical simulation using the Digital Twin under the assumption that the cost function evaluated is smooth, convex and will have a defined minimum.

However, these assumptions are harder to satisfy as the system complexity and its dimensions increase. For this reason, the analysis of convergence, and stability of the Self Optimizing algorithms is crucial to ensure the optimal performance of the controlled system on difficult conditions like non-convex or non-smooth cost functions, which analytic gradients cannot be calculated and should be approximated numerically. Likewise, the globalness of the SOC algorithms should be addressed to ensure the optimization does not fall into a local minimal.

In this work, we address the convergence analysis using probabilistic theory and “frame based method” for the Nelder-Mead optimization stage with stability assumptions for the dynamic behavior of the system. However, the convergence, stability, and convergence analysis for the SOC framework can be potentially extended by using convex analysis theory, Stochastic Differential Equations (SDE), or stochastic processes. So, it can be analyzed not only the convergence of the random time series corresponding to the approximated gradient optimization error, its direction, residuals or bias towards the convergence to a minimum following the ODE convergence analysis proposed by [306, 317, 318].

Another important aspect of the SOC control framework is to ensure the monotonicity of the solution evaluated on the real system. Thus, the optimization algorithm always performs an improvement for each run period of the system. In that case, the GCNM algorithm proposed for the theoretical analysis shows its potential for a monotonic behavior. However, if other searching methods like stochastic gradient or pattern search are used, additional considerations are required to ensure its monotonicity.

9.4 Accelerated Learning Using Faster Convergence Optimization Algorithms for Self Optimizing Control and Behavioral Matching

Considering that the Self Optimizing Control performs real-time optimization to improve the system performance, accelerate the convergence and learning for the optimization algorithm is desirable to get the best performance of the system in a reasonable number of iterations, specially when a disturbance or external event like a sensor malfunction alters the normal operation of the system.

In that sense, accelerated optimization algorithms can be considered. If stochastic approximation gradient descend methods are employed, schemes like heavy ball, Nesterov, momentum, or ADAM can be implemented to increase the SOC convergence speed. However, considering the random probabilistic restart, considerations of random search needs to be included in the convergence analysis for these algorithms.

Likewise, in the case of derivative free optimization methods, using heavy tail distributions like the symmetric α-stable or fractional-order Gaussian noise with LRD represented by the Hurst exponent can accelerate the convergence of the random search component of the SOC algorithms. in [319], the authors have proved that accelerating the SOC is possible using fractional-order Gaussian noise in the case of the GCNM algorithm. However, a deep theoretical analysis and use with other derivative free optimization methods is required to determine the best randomness that can enhance the SOC performance.

Notice that these acceleration techniques can be applied not only to the SOC but also to the behavioral matching as well as during the real-time analytics and feature analysis on the Digital Twin smart capabilities to train new fault detection algorithms or perform faster inference from real time data coming into the DT.

9.5 Parallel Computing and Digital Twin

The solution of the SOC optimization problem may take a long time before reaching an acceptable solution based on an economical cost function due to the nature of the optimization problem and the system dynamics, for example, thermal processes even using accelerated optimization algorithms as discussed on the previous subsection.

In that case, parallel intelligence and control as a novel paradigm that looks for the integration of complex systems under the ACP approach (Analysis, Control, Parallel Execution) to improve the system performance can be integrated into the

SOC to accelerate the system learning, and optimization [235, 304, 305]. In order to introduce parallel capabilities to SOC, a different optimization algorithm is required to handle the presence of multiple simultaneous executions to enhance the system performance and optimization speed like the Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm [306], which structure allows enabling parallel execution [311, 312]. Thus, SPSA can be used as bridge between virtual and real systems to enhance SOC performance by simultaneous evaluation of multiple models of a system represented by instances of Digital Twins. In [295], the Digital Twin is employed to accelerate the speed of SPSA using Digital Twins. However, ACP can be extended to other searching algorithms by enabling DT for simultaneous cost functions evaluations, reducing the optimization time and incorporating more intelligence into the physical assets.

9.6 Digital Twin for Control Education

Laboratory experiences have a considerable role in the engineering education and learning experience. For the undergraduate level courses like Mechatronics and Control Engineering, hands-on experience with a hardware creates huge impact on the learning process as defined in MAD methodology [1]. However, the COVID-19 pandemic had a huge influence on the educational processes. Universities all over the world switched to remote learning experiences, resulting a challenging adaptation problem for the students especially for the laboratory intensive courses. In this concept, there are different applications of remote applications for Mechatronics and control engineering [320–322], remote laboratory [323–325], robotics [326] as well as high-end commercial products that enable remote laboratory applications such as Roboholics Maniacs [327] and Quanser [328]. These remote applications enable user remote connection. However, almost all of them require time scheduling for users to dedicate hardware, server or connection to limited number of users, being a problem for a course with big number of students.

Therefore, in this scenario Digital Twin became a feasible candidate for the remote laboratory applications considering the situation with a need of remote access laboratories and undesirable hardware issues. Thus, having virtual representations of the system will give the students a more immersive experience on the control design and modelling of the physical systems from remote perspective. in [329], a Digital Twin based laboratory system is proposed, leveraging the virtual features and remote access given the Digital Twin for the students interaction with 10 different types of control systems, enhancing their knowledge on MAD methodology and being closer as possible to real hardware amid remote learning constraints.

9.7 Performance Assessment of Different Real-Time Optimization Algorithms

In this thesis, the Nelder-Mead had been used as principal optimization algorithm, deriving a variant called globalized constrained Nelder-Mead with online execution properties. However, there are several alternatives to Nelder-Mead that can be explored for its online implementation, which fall into the derivative free optimization like particle swarm, implicit filtering, simultaneous perturbation stochastic approximation etc. For this reason, the framework provided by the GCNM optimization algorithm can be extended for the algorithms mentioned below and be tested not only for closed loop control but also in applications like gas sensing, drug dosage for chemotherapy among others.

9.8 Smartness Metric

Once a system is smart, by following the five characteristics defined on chapter 1, one question to be answered is how smart is my system?. In a qualitative sense, we can say that the system is smart if meets at least one of the five attributes. However, quantification of smartness goes beyond the qualitative scope. Traditionally, the reasoning and analysis capabilities from computers are measured through the Turing test to determine if the computer can imitate human behavior. In the case of smart control engineering, the smartness is focused on the ability of keep the system performance against changes on the system dynamic response, due to the change on setup conditions. In that sense, a smartness test should be focused on evaluate or assess the capacity of a smart control system (SOC+DT+IAI) track the performance goals as it holds the smart system capabilities. For this reason, the smartness metric can be proposed as a set of real life tests, which evaluate the response of the system in the five characteristics of the smart system and provide a quantitative indicator in terms of classic analytics for control systems. For example, in the case of cognizant capabilities, one possible metric could be the response time on parameter adaptation once the change is detected by the algorithm. On the reflective feature, a possible indicator can be the time required to stabilize the system after the disturbance is detected. In the knowledge rich sense, one metric could be the minimum amount of information in storage units (MB) that the SOC requires before reacts to the system changes. In the case of taskable, it may be challenging, because it means the system has the freedom to take the best course of action to execute one task. From the ethical feature, it can be a checklist to determine if the system is close to some prestablished danger thresholds like critical heating conditions in a chemical reactor or how many times the system is set to its safety limits to perform one task. Notice that the proposed methods are based on the five attributes of smart system, however, a wide and open discussion is required to determine a correct set of metrics to measure the smartness of a system.

BIBLIOGRAPHY

[1] D. Xue, Y. Chen, Modeling, Analysis and Design of Control Systems in MATLAB and Simulink, World Scientific, 2014. doi:10.1142/9260.
URL https://www.worldscientific.com/doi/abs/10.1142/9260

[2] P. D. Domański, Control Performance Assesment: Theoretical Analyses and Industrial Practice, Springer London, 2020.

[3] L. C. Linyuan Shang, Xuemin Tian, A Multi-index Control Performance Assessment Method Based on Historical Prediction Error Covariance, IFAC-PapersOnLine 50 (1) (2017) 13892–13897. doi:10.1016/j.ifacol.2017.08.2207.

[4] J. A. Primbs, V. Nevistic, A New Approach to Stability Analysis for Constrained Finite Receding Horizon Control Without End Constraints, IEEE Transactions on Automatic Control 45 (8) (2000) 287–291.

[5] J. Wang, C. Zhao, A probabilistic framework with concurrent analytics of Gaussian process regression and classification for multivariate control performance assessment, Journal of Process Control 101 (2021) 78–92. doi:https://doi.org/10.1016/j.jprocont.2021.03.007.

[6] Y. Wang, H. Zhang, S. Wei, D. Zhou, B. Huang, Control Performance Assessment for ILC-Controlled Batch Processes in a 2-D System Framework, IEEE Transactions on Systems, Man, and Cybernetics: Systems 48 (9) (2018) 1493–1504. doi:10.1109/TSMC.2017.2672563.

[7] Q. W. Chunyu Liu, Biao Huang, Control performance assessment subject to multi-objective user-specified performance characteristics, IEEE Transactions on Control Systems Technology 19 (3) (2011) 682–691. doi:10.1109/TCST.2010.2051669.

[8] X. Wang, B. Huang, T. Chen, Multirate minimum variance control design and control performance assessment: A data-driven subspace approach, IEEE Transactions on Control Systems Technology 15 (1) (2007) 65–74. doi:10.1109/TCST.2006.883240.

[9] F. Abolhassan, The Drivers of Digital Transformation Why There is No Way Around the Cloud, Springer, 2017. doi:10 1007 978-3-319-31824-0 4.

[10] A. Ustundag, E. Cevikcan, Industry 4.0: Managing The Digital Transformation, Springer London, 2018. doi:10.1007/978-3-319-57870-57.

[11] R. Neugebauer, Digital Transformation, Springer, 2018. doi:10.1109/MS.2018.2801537.

[12] A. Bounfour, Digital Futures, Digital Transformation From Lean Production to Acceluction, Springer, 2016. doi:10.1007/978-3-319-23279-9_4.

[13] NSF, Smart and Autonomous Systems (S&AS).
URL https://www.nsf.gov/pubs/2018/nsf18557/nsf18557.htm?WT.mc_id=USNSF_25&WT.mc_ev=click#pgm_intr_txt

[14] F. Zhu, Z. Li, S. Chen, G. Xiong, Parallel Transportation Management and Control System and Its Applications in Building Smart Cities, IEEE Transactions on Intelligent Transportation Systems 17 (6) (2016) 1576–1585. doi:10.1109/TITS.2015.2506156.

[15] P. Vicuna, S. Mudigonda, C. Kamga, K. Mouskos, C. Ukegbu, A generic and flexible geospatial data warehousing and analysis framework for transportation performance measurement in smart connected cities, Procedia Computer Science 155 (2018) (2019) 226–233. doi:10.1016/j.procs.2019.08.033.

[16] J. Schmalzel, F. Figueroa, J. Morris, S. Mandayam, R. Polikar, An architecture for intelligent systems based on smart sensors, in: proccedings of the IEEE Instrumentation and Measurement Technology Conference, 2004, pp. 71–75. doi:10.1109/imtc.2004.1350998.

[17] A. H. Bagdadee, Z. Hoque, L. Zhang, IoT Based Wireless Sensor Network for Power Quality Control in IoT Based Wireless Sensor Smart Network for Power Quality Control in Grid Smart Grid, Procedia Computer Science 167 (2019) (2020) 1148–1160.

[18] P. Zhang, F. Li, N. Bhatt, Next-generation monitoring, analysis, and control for the future smart control center, IEEE Transactions on Smart Grid 1 (2) (2010) 186–192. doi:10.1109/TSG.2010.2053855.

[19] M. Rahimi, M. Songhorabadi, M. H. Kashani, Fog-based smart homes: A systematic review, Journal of Network and Computer Applications 153 (2020) 102531. doi:10.1016/j.jnca.2020.102531.

[20] B. J. Chang, J. M. Chiou, Cloud Computing-Based Analyses to Predict Vehicle Driving Shockwave for Active Safe Driving in Intelligent Transportation System, IEEE Transactions on Intelligent Transportation Systems 21 (2) (2020) 852–866. doi:10.1109/TITS.2019.2902529.

[21] L. Chen, Y. He, J. Chen, Q. Li, Q. Zou, Transforming a 3-D LiDAR Point Cloud into a 2-D Dense Depth Map Through a Parameter Self-Adaptive Framework, IEEE Transactions on Intelligent Transportation Systems 18 (1) (2017) 165–176. doi:10.1109/TITS.2016.2564640.

[22] YangQuan Chen, Cognitive Process Control (2012).
URL http://mechatronics.ucmerced.edu/digital-twin

[23] C. Strategies, Artificial Intelligence for Industrial Applications, Tech. rep., CloudPulse Strategies (2017).
URL https://www.cloudpulsestrat.com/go/industrialai

[24] Y. Li, Y. Ma, Z. Yin, A. Gu, S. Sun, A Time-Sensitive Streams Management Method based on IEEE 802.1Qat SRP for Industrial Internet, in: proceedings of the 1st International Conference on Industrial Artificial Intelligence, IAI 2019, Vol. 2019-January, 2019, p. 6. doi:10.1109/IAI47267.2019.9085308.

[25] G. Li, K. Ota, M. Dong, J. Wu, J. Li, DeSVig: Decentralized Swift Vigilance against Adversarial Attacks in Industrial Artificial Intelligence Systems, IEEE Transactions on Industrial Informatics 16 (5) (2020) 3267–3277. doi:10.1109/TII.2019.2951766.

[26] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, S. Liu, Efficient and Privacy-Enhanced Federated Learning for Industrial Artificial Intelligence, IEEE Transactions on Industrial Informatics 16 (10) (2020) 6532–6542. doi:10.1109/TII.2019.2945367.

[27] R. S. Peres, X. Jia, J. Lee, K. Sun, A. W. Colombo, J. Barata, Industrial Artificial Intelligence in Industry 4.0 -Systematic Review, Challenges and Outlook, IEEE Access (2020) 220121–220139doi:10.1109/ACCESS.2020.3042874.

[28] J. Zhang, J. Cao, Fault diagnosis for closed loop nonlinear system using generalized frequency response functions and least square support vector machine, in: proceedings of the 1st International Conference on Industrial Artificial Intelligence, IAI 2019, IEEE, 2019, pp. 2–5. doi:10.1109/ICIAI.2019.8850734.

[29] M. Grieves, Digital Twin : Manufacturing Excellence through Virtual Factory Replication This paper introduces the concept of a A Whitepaper by Dr . Michael Grieves, White Paper.

[30] IBM, Digital twin: Helping machines tell their story (2020).
URL https://www.ibm.com/internet-of-things/trending/digital-twin

[31] S. Skogestad, Plantwide control: the search for the self-optimizing control structure, Journal of Process Control 10 (5) (2000) 487–507. doi:https://doi.org/10.1016/S0959-1524(00)00023-8.

[32] K. Bariyur, M. Krstic, Real-time optimization by extremum-seeking control, Wiley-Interscience, 2003. doi:10.5860/choice.41-4064.

[33] D. Verma, S. Nema, A. M. Shandilya, S. K. Dash, Maximum power point tracking (MPPT) techniques: Recapitulation in solar photovoltaic systems, Renewable and Sustainable Energy Reviews 54 (2016) 1018–1034. doi:10.1016/j.rser.2015.10.068.

[34] H. S. Ahn, Y. Q. Chen, K. L. Moore, Iterative learning control: Brief survey and categorization, IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews 37 (6) (2007) 1099–1121. doi:10.1109/TSMCC.2007.905759.

[35] K. Jia, B. Liu, M. Iyogun, T. Bi, Smart control for battery energy storage system in a community grid, in: proceedings of the POWERCON 2014 - 2014 International Conference on Power System Technology: Towards Green, Efficient and Smart Power System, Proceedings, IEEE, 2014, pp. 3243–3248. doi:10.1109/POWERCON.2014.6993790.

[36] P. Zhang, F. Li, N. Bhatt, Next-generation monitoring, analysis, and control for the future smart control center, IEEE Transactions on Smart Grid 1 (2) (2010) 186–192. doi:10.1109/TSG.2010.2053855.

[37] B. W. Abegaz, M. Cmiel, Smart control of buck converters using a switching-based clustering algorithm, 2019 14th Annual Conference System of Systems Engineering, SoSE 2019 (2019) 334–339doi:10.1109/SYSOSE.2019.8753864.

[38] J. H. Teng, S. H. Liao, W. H. Huang, C. C. Chiang, Smart Control Strategy for Conversion Efficiency Enhancement of Parallel Inverters at Light Loads, IEEE Transactions on Industrial Electronics 63 (12) (2016) 7586–7596. doi:10.1109/TIE.2016.2594791.

[39] Y. Yu, O. S. Nduka, B. C. Pal, Smart Control of an Electric Vehicle for Ancillary Service in DC Microgrid, IEEE Access 8 (2020) 197222–197235. doi:10.1109/ACCESS.2020.3034496.

[40] J. S. Artal-Sevil, I. Sanz-Gorrachategui, P. Pastor, C. Bernal-Ruiz, F. J. Perez-Cebolla, A. Bono-Nuez, New Smart Control based on MPPT/MEPT Algorithm for Hybrid Fuel Cell Power System, in: proceedings of the 15th International Conference on Ecological Vehicles and Renewable Energies, EVER 2020, 2020, pp. 1–6. doi:10.1109/EVER48776.2020.9243139.

[41] J. Lopez, C. Alcaraz, R. Roman, Smart control of operational threats in control substations, Computers and Security 38 (2013) 14–27. doi:10.1016/j.cose.2013.03.013.

[42] C. Benachaiba, A. M. Haidar, M. Habab, O. Abdelkhalek, Smart control of UPCQ within microgrid energy system, Energy Procedia 6 (2011) 503–512. doi:10.1016/j.egypro.2011.05.058.

[43] U. Datta, A. Kalam, J. Shi, Smart control of BESS in PV integrated EV charging station for reducing transformer overloading and providing battery-to-grid service, Journal of Energy Storage 28 (December 2019). doi:10.1016/j.est.2020.101224.

[44] M. S. Jonban, L. Romeral, A. Akbarimajd, Z. Ali, S. S. Ghazimirsaeid, M. Marzband, G. Putrus, Autonomous energy management system with self-healing capabilities for green buildings (microgrids), Journal of Building Engineering 34 (April 2020) (2021) 101604. doi:10.1016/j.jobe.2020.101604.

[45] J. Wang, J. Liu, C. Li, Y. Zhou, J. Wu, Optimal scheduling of gas and electricity consumption in a smart home with a hybrid gas boiler and electric heating system, Energy 204. doi:10.1016/j.energy.2020.117951.

[46] S. Rhys, S. R. Jones, J. Beardmore, M. Gillott, R. Boukhanouf, G. Walker, T. Built, A. Pina, P. Ferrão, J. Fournier, B. Lacarrière, O. L. Corre, A Control Methodology for Building Energy Management Systems, in: proceedings of the 5th International Conference on Energy and Environment Research, ICEER 2018, Vol. 00, 2018, p. 6.

[47] G. T. Costanzo, S. Iacovella, F. Ruelens, T. Leurs, B. J. Claessens, Experimental analysis of data-driven control for a building heating system, Sustainable Energy, Grids and Networks 6 (2016) 81–90. arXiv:1507.03638, doi:10.1016/j.segan.2016.02.002.

[48] X. Huang, D. Zhang, X. S. Zhang, Energy management of intelligent building based on deep reinforced learning, Alexandria Engineering Journal 60 (1) (2021) 1509–1517. doi:10.1016/j.aej.2020.11.005.

[49] S. Kim, H. Lim, Reinforcement learning based energy management algorithm for smart energy buildings, Energies 11 (8). doi:10.3390/en11082010.

[50] K. U. Ahn, C. S. Park, Application of deep Q-networks for model-free optimal control balancing between different HVAC systems, Science and Technology for the Built Environment 26 (1) (2020) 61–74. doi:10.1080/23744731.2019.1680234.

[51] Y. Zhang, C. Qian, J. Lv, Y. Liu, Agent and cyber-physical system based self-organizing and self-adaptive intelligent shopfloor, IEEE Transactions on Industrial Informatics 13 (2) (2017) 737–747. doi:10.1109/TII.2016.2618892.

[52] H. Zhang, G. Zhang, Q. Yan, Dynamic resource allocation optimization for digital twin-driven smart shopfloor, in: proceedings of the 2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC), IEEE, 2018, pp. 0–4.

[53] I. Chatzikonstantinou, D. Giakoumis, D. Tzovaras, A new shopfloor orchestration approach for collaborative human-robot device disassembly, Proceedings - 2019 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Internet of People and Smart City Innovation, SmartWorld/UIC/ATC/SCALCOM/IOP/SCI 2019 (2019) 225–230doi:10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00081.

[54] H. Brandtstaedter, C. Ludwig, L. Hübner, E. Tsouchnika, A. Jungiewicz, U. Wever, Digital twins for large electric drive trains, in: proceedings of the 2018 Petroleum and Chemical Industry Conference Europe (PCIC Europe), 2018, pp. 1–5. doi:10.23919/PCICEurope.2018.8491413.

[55] M. Milton, C. O. De La, H. L. Ginn, A. Benigni, Controller-Embeddable Probabilistic Real-Time Digital Twins for Power Electronic Converter Diagnostics, IEEE Transactions on Power Electronics 35 (9) (2020) 9852–9866. doi:10.1109/TPEL.2020.2971775.

[56] Y. Peng, H. Wang, A. B. C. Circuit, Application of Digital Twin Concept in Condition Monitoring for DC-DC Converters, 2019 IEEE Energy Conversion Congress and Exposition (ECCE) (2019) 2199–2204.

[57] Y. Peng, X. Zhang, Y. Song, D. Liu, A low cost flexible digital twin platform for spacecraft lithium-ion battery pack degradation assessment, in: proceedings of the I2MTC 2019 - 2019 IEEE International Instrumentation and Measurement Technology Conference, Proceedings, IEEE, 2019, p. 6. doi:10.1109/I2MTC.2019.8827160.

[58] P. Jain, J. Poon, J. P. Singh, C. Spanos, S. R. Sanders, S. K. Panda, A digital twin approach for fault diagnosis in distributed photovoltaic systems, IEEE Transactions on Power Electronics 35 (1) (2020) 940–956. doi:10.1109/TPEL.2019.2911594.

[59] M. Ruba, R. O. Nemes, S. M. Ciornei, C. Martis, A. Bouscayrol, H. Hedesiu, Digital twin real-time fpga implementation for light electric vehicle propulsion system using EMR organization, in: proceedings of the 2019 IEEE Vehicle Power and Propulsion Conference, VPPC 2019, 2019, pp. 0–5. doi:10.1109/VPPC46532.2019.8952428.

[60] J. Eyre, C. Freeman, Immersive Applications of Industrial Digital Twins, EuroVR 2018 1 (November) (2018) 9.

[61] T. Cichon, J. Robmann, Digital Twins: Assisting and Supporting Cooperation in Human-Robot Teams, in: proceedings of the 2018 15th International Conference on Control, Automation, Robotics and Vision, ICARCV 2018, 2018, pp. 486–491. doi:10.1109/ICARCV.2018.8580634.

[62] A. Piros, L. Trautmann, E. Baka, Error handling method for digital twin-based plasma radiation detection, Fusion Engineering and Design 156 (February). doi:10.1016/j.fusengdes.2020.111592.

[63] F. Tao, M. Zhang, Y. Liu, A. Y. Nee, Digital twin driven prognostics and health management for complex equipment, CIRP Annals 67 (1) (2018) 169–172. doi:10.1016/j.cirp.2018.04.055.

[64] R. Martinez-Velazquez, R. Gamez, A. E. Saddik, Cardio Twin: A Digital Twin of the human heart running on the edge, Medical Measurements and Applications, MeMeA 2019 - Symposium Proceedings (2019) 1–6doi:10.1109/MeMeA.2019.8802162.

[65] L. Cattaneo, M. MacChi, A Digital Twin Proof of Concept to Support Machine Prognostics with Low Availability of Run-To-Failure Data, IFAC-PapersOnLine 52 (10) (2019) 37–42. doi:10.1016/j.ifacol.2019.10.016.

[66] L. Li, W. Xu, Z. Liu, B. Yao, Z. Zhou, D. T. Pham, Digital twin-based control approach for industrial cloud robotics, in: proceedings of the ASME 2019 14th International Manufacturing Science and Engineering Conference, MSEC 2019, Vol. 1, 2019, pp. 1–7. doi:10.1115/MSEC2019-2920.

[67] H. Pargmann, D. Euhausen, R. Faber, Intelligent big data processing for wind farm monitoring and analysis based on cloud-Technologies and digital twins: A quantitative approach, in: proceedings of the 2018 3rd IEEE International

Conference on Cloud Computing and Big Data Analysis, ICCCBDA 2018, 2018, pp. 233–237. doi:10.1109/ICCCBDA.2018.8386518.

[68] N. Stojanovic, D. Milenovic, Data-driven Digital Twin approach for process optimization: An industry use case, in: Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018, IEEE, 2019, pp. 4202–4211. doi:10.1109/BigData.2018.8622412.

[69] Q. Qiao, J. Wang, R. X. Gao, P. Stief, J.-y. Dantan, A. Etienne, A. Siadat, Digital Twin for Machining Tool Condition Prediction Twin for Machining Tool Condition Prediction, physical analyze the functional and architecture of assembly oriented product, Procedia CIRP 81 (2019) 1388–1393. doi:10.1016/j.procir.2019.04.049.

[70] M. R. Shahriar, S. M. A. Sunny, X. Liu, M. C. Leu, L. Hu, N. T. Nguyen, MTComm based virtualization and integration of physical machine operations with digital-twins in cyber-physical manufacturing cloud, in: Proceedings - 5th IEEE International Conference on Cyber Security and Cloud Computing and 4th IEEE International Conference on Edge Computing and Scalable Cloud, CSCloud/EdgeCom 2018, IEEE, 2018, pp. 46–51. doi:10.1109/CSCloud/EdgeCom.2018.00018.

[71] N. M. Khan, R. Suresh Kumar, Smart control of cylindrical shells incorporating Murakami Zig-Zag function, Composite Structures 257 (September 2020) (2021) 113044. doi:10.1016/j.compstruct.2020.113044.

[72] A. Farajpour, A. Rastgoo, M. Mohammadi, Vibration, buckling and smart control of microtubules using piezoelectric nanoshells under electric voltage in thermal environment, Physica B: Condensed Matter 509 (December 2016) (2017) 100–114. doi:10.1016/j.physb.2017.01.006.

[73] M. H. Hajmohammad, A. Farrokhian, R. Kolahchi, Smart control and vibration of viscoelastic actuator-multiphase nanocomposite conical shells-sensor considering hygrothermal load based on layerwise theory, Aerospace Science and Technology 78 (2018) 260–270. doi:10.1016/j.ast.2018.04.030.

[74] K. S. Arsava, Y. Kim, K. H. Kim, B. S. Shin, Smart fuzzy control of reinforced concrete structures excited by collision-type forces, Expert Systems with Applications 42 (21) (2015) 7929–7941. doi:10.1016/j.eswa.2015.05.024.

[75] S. Nadkarni, R. Prügl, Digital transformation: a review, synthesis and opportunities for future research, Management Review Quarterly 71 (2) (2021) 233–341. doi:10.1007/s11301-020-00185-7.

[76] J. Lee, Industrial AI: Applications with Sustainable Performance, 1st Edition, Springer, Singapore, 2020. doi:https://doi.org/10.1007/978-981-15-2144-7.

[77] J. Clemons, Digital transformation: time for industry to kick into high gear (2020).
URL https://www.controleng.com/articles/digital-transformation-time-for-industry-to-kick-into-high-gear/

[78] Protivity, What is Internet of Things (IoT)? (2021).
URL https://www.protiviti.com/US-en/technology-consulting/iot-services

[79] J. Lee, M. Azamfar, J. Singh, S. Siahpour, Integration of digital twin and deep learning in cyber-physical systems: towards smart manufacturing, IET Collaborative Intelligent Manufacturing 2 (1) (2020) 34–36. doi:10.1049/iet-cim.2020.0009.

[80] J. Lee, J. Singh, M. Azamfar, Industrial artificial intelligence, CoRR abs/1908.02150. arXiv:1908.02150.

[81] B. B. Jay Lee, Hossein Davari Ardakani, Shanhu Yang, Industrial Big Data Analytics and Cyber-physical Systems for Future maintenance and service innovation, in: Proceedings of the Fourth International Conference on Through-life Engineering Services, Elsevier B.V., 2015, pp. 3–7. doi:10.1016/j.procir.2015.08.026.

[82] S. Zidi, T. Moulahi, B. Alaya, Fault detection in wireless sensor networks through SVM classifier, IEEE Sensors Journal 18 (1) (2018) 340–347. doi:10.1109/JSEN.2017.2771226.

[83] Y. J. Huang, C. L. Pan, S. C. Lin, M. H. Guo, Machine-Learning Approach in Detection and Classification for Defects in TSV-Based 3-D IC, IEEE Transactions on Components, Packaging and Manufacturing Technology 8 (4) (2018) 699–706. doi:10.1109/TCPMT.2017.2788896.

[84] L. J. Defu Wang, Xiaojuan Wang, Yong Zhang, Detection of power grid disturbances and cyber-attacks based on machine learning, Journal of Information Security and Applications 46 (2019) 42–52.

[85] U. Lee, G. Tello, Y. Al-Hammadi, S. Muhaidat, P. D. Yoo, O. Y. Al-Jarrah, Deep-structured Machine Learning Model for the Recognition of Mixed-defect Patterns in Semiconductor Fabrication Processes, IEEE Transactions on Semiconductor Manufacturing 31 (2) (2018) 1–1. doi:10.1109/tsm.2018.2825482.

[86] D.-w. Jang, S. Lee, J.-w. Park, D.-c. Baek, Failure detection technique under random fatigue loading by machine learning and dual sensing on symmetric structure, International Journal of Fatigue 114 (December 2017) (2018) 57–64. doi:10.1016/j.ijfatigue.2018.05.004.

[87] M. Azevedo, B. Wullt, M. Norrlöf, S. Gunnarsson, Failure detection in robotic arms using statistical modeling, machine learning and hybrid gradient boosting, Measurement 146 (2019) 425–436. doi:10.1016/j.measurement.2019.06.039.

[88] E. Gurina, N. Klyuchnikov, A. Zaytsev, E. Romanenkova, K. Antipova, I. Simon, V. Makarov, D. Koroteev, Application of machine learning to accidents detection at directional drilling, Journal of Petroleum Science and Engineering 184 (2020) 106519. doi:https://doi.org/10.1016/j.petrol.2019.106519.

[89] M. Hasan, M. Islam, I. I. Zarif, M. M. A. Hashem, Internet of Things Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches, Internet of Things 7 (2019) 100059. doi:10.1016/j.iot.2019.100059.

[90] D. Wang, X. Wang, Y. Zhang, L. Jin, Detection of power grid disturbances and cyber-attacks based on machine learning, Journal of Information Security and Applications 46 (2019) 42–52. doi:10.1016/j.jisa.2019.02.008.

[91] F. Peng, B. Li, B. Luo, H. Wang, H. Liu, Early Fault Detection of Machine Tools Based on Deep Learning and Dynamic Identification, IEEE Transactions on Industrial Electronics 66 (1) (2018) 509–518. doi:10.1109/tie.2018.2807414.

[92] Yi Lu Murphey, M. A. Masrur, ZhiHang Chen, Baifang Zhang, Model-based fault diagnosis in electric drives using machine learning, IEEE/ASME Transactions on Mechatronics 11 (3) (2006) 290–303.

[93] J. Wang, W. Zhang, J. Zhou, Fault Detection with Data Imbalance Conditions Based on the Improved Bilayer Convolutional Neural Network, ACS journal on Industrial and Engineering Chemistry Research 13 (59) (2020) 5891–5904. doi:10.1021/acs.iecr.9b06298.

[94] W. Zhang, C. Li, G. Peng, Y. Chen, Z. Zhang, A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load, Mechanical Systems and Signal Processing 100 (2018) 439–453. doi:10.1016/j.ymssp.2017.06.022.

[95] J. Yang, G. Xie, Y. Yang, Control Engineering Practice An improved ensemble fusion autoencoder model for fault diagnosis from imbalanced and incomplete data, Control Engineering Practice 98 (February) (2020) 104358. doi:10.1016/j.conengprac.2020.104358.

[96] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Networks (2014). arXiv:1406.2661.

[97] J. Feng, H. Yu, L. Wang, X. Cao, Classification of Hyperspectral Images Based on Multiclass Spatial – Spectral Generative Adversarial Networks, IEEE Transactions on Geoscience and Remote Sensing 57 (8) (2019) 5329–5343. doi:10.1109/TGRS.2019.2899057.

[98] K. Kim, S. Member, H. Myung, S. Member, Autoencoder-Combined Generative Adversarial Networks for Synthetic Image Data Generation and Detection of Jellyfish Swarm, IEEE Access 6 (2018) 54207–54214. doi:10.1109/ACCESS.2018.2872025.

[99] Q. Xuan, Z. Chen, Y. Liu, H. Huang, G. Bao, D. Zhang, Multiview Generative Adversarial Network and Its Application in Pearl Classification, IEEE Transactions on Industrial Electronics 66 (10) (2019) 8244–8252. doi:10.1109/TIE.2018.2885684.

[100] X. Yi, E. Walia, P. Babyn, Generative adversarial network in medical imaging : A review, Medical Image Analysis 58.

[101] Z. Wang, J. Wang, Y. Wang, An intelligent diagnosis scheme based on generative adversarial learning deep neural networks and its application to planetary gearbox fault pattern recognition, Neurocomputing 310 (2018) 213–222.

[102] H. Liu, J. Zhou, Y. Xu, Y. Zheng, X. Peng, W. Jiang, Unsupervised fault diagnosis of rolling bearings using a deep neural network based on generative adversarial networks, Neurocomputing 315 (2018) 412–424. doi:10.1016/j.neucom.2018.07.034.

[103] Y. Yang, P. Fu, Y. He, Bearing fault automatic classification based on deep learning, IEEE Access 6 (2018) 71540–71554. doi:10.1109/ACCESS.2018.2880990.

[104] H. Salehinejad, E. Colak, T. Dowdell, J. Barfett, S. Valaee, Synthesizing Chest X-Ray Pathology for Training Deep Convolutional Neural Networks, IEEE Transactions on Medical Imaging 38 (5) (2019) 1197–1206. doi:10.1109/TMI.2018.2881415.

[105] A. Diaz-pinto, A. Colomer, V. Naranjo, S. Morales, Y. Xu, A. F. Frangi, Retinal Image Synthesis and Semi-Supervised Learning for Glaucoma Assessment, IEEE Transactions on Medical Imaging 38 (9) (2019) 2211–2218. doi:10.1109/TMI.2019.2903434.

[106] H. Heo, Y. Hwang, Automatic Sketch Colorization using DCGAN Hwan Heo and Youngbae Hwang, in: proceedings of the 2018 18th International Conference on Control, Automation and Systems (ICCAS), Institute of Control, Robotics and Systems - ICROS, 2018, pp. 1316–1318.

[107] T. Li, G. Liang, W. He, Y. U. E. Zhao, A Simple Recurrent Unit Model Based Intrusion Detection System With DCGAN, IEEE Access 7 (3).

[108] C. W. R. University, Ball Bearing data center (2010).
URL http://csegroups.case.edu/bearingdatacenter/home

[109] S. C. Olhede, A. T. Walden, A. Member, Generalized Morse Wavelets, IEEE Transactions on Signal Processing 50 (11) (2002) 2661–2670. doi:10.1109/TSP.2002.804066.

[110] P. Porwik, A. Lisowska, The Haar – Wavelet Transform in Digital Image Processing : Its Status and Achievements, Machine GRAPHICS & VISION 13 (1) (2004) 79–97.

[111] Y. Du, W. Zhang, J. Wang, H. Wu, Dcgan based data generation for process monitoring, in: proceedings 2019 IEEE 8th Data Driven Control and Learning Systems Conference (DDCLS), 2019, pp. 410–415.

[112] G. Robert M., Toeplitz and circulant matrices: A review, Foundations and Trends in Communications and Information Theory 2 (3) (2006) 155–239. doi:10.1561/0100000006.

[113] H. Schwerdtfeger, Introduction to Linear Algebra and the Theory of Matrices, P. Noordhoff, 1961.

[114] A. Radford, L. Metz, S. Chintala, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, in: Proceeding of 4th International Conference on Learning Representations, ICLR, 2016, pp. 1–16. arXiv:1511.06434, doi:10.1051/0004-6361/201527329.

[115] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, Image quality assessment: From error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–612. doi:10.1109/TIP.2003.819861.

[116] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, Neural Computation 9 (8) (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[117] C. Yang, H. Wang, Z. Gao, Improving rolling bearing online fault diagnostic performance based on characteristics, Royal Society of Open Science 5 (5).

[118] W. A. Smith, R. B. Randall, Rolling element bearing diagnostics using the Case Western Reserve University data : A benchmark study, Mechanical Systems and Signal Processing 64-65 (2015) 100–131. doi:10.1016/j.ymssp.2015.04.021.

[119] O. Kılınç, J. Vágner, Fault severity detection of ball bearings and efficiency of one-period analysis in early fault diagnosis of rotating machinery, Vibroengineering Procedia 7 (2016) 76–81.

[120] C. Yang, H. Wang, Z. Gao, Improving rolling bearing online fault diagnostic performance based on characteristics, Royal Society of Open Science 5 (5).

[121] Gartnet Inc., 5 Trends Emerge in the Gartner Hype Cycle for Emerging Technologies, 2018 (2018).
URL https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/

[122] D. consulting, Digital Twins (2020).
URL https://www2.deloitte.com/us/en/insights/focus/tech-trends/2020/digital-twin-applications-bridging-the-physical-and-digital.html{\#}endnote-10

[123] Markets and Markets Research Private Ltd, Digital Twin Market by Technology, Type (Product, Process, and System), Industry (Aerospace & Defense, Automotive & Transportation, Home & Commercial, Healthcare, Energy & Utilities, Oil & Gas), and Geography - Global Forecast to 2025 (2020).
URL https://www.marketsandmarkets.com/Market-Reports/digital-twin-market-225269522.html

[124] M. Grieves, J. Vickers, Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems, Springer International Publishing, Cham, 2017, Ch. 1, pp. 85–113. doi:10.1007/978-3-319-38756-74.

[125] Mathworks Inc, What is a digital twin? (2019).
URL https://www.mathworks.com/discovery/digital-twin.html

[126] GE Digital, What is a digital twin? (2019).
URL https://www.ge.com/digital/applications/digital-twin

[127] D.S. Stargel, E. H. Glaessgen, The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles, in: proceedings of the 53rd Structures, Structural Dynamics, and Materials Conference: Special Session on the Digital Twin, 2012, pp. 1–14.

[128] C. Dufour, Z. Soghomonian, W. Li, Hardware-in-the-Loop Testing of Modern On-Board Power Systems Using Digital Twins, SPEEDAM 2018 - Proceedings: International Symposium on Power Electronics, Electrical Drives, Automation and Motion (2018) 118–123doi:10.1109/SPEEDAM.2018.8445302.

[129] F. Tao, Q. Qi, Make more digital twins, Nature 573 (7775) (2019) 490–491. doi:10.1038/d41586-019-02849-1.

[130] A. M. Madni, C. C. Madni, S. D. Lucero, Leveraging Digital Twin Technology in Model-Based Systems Engineering, Systems (2019) 1–13doi:10.3390/systems7010007.

[131] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, H. Jahankhani, Internet of Things Digital Twin Technologies and Smart Cities, Springer, 2020.
URL http://www.springer.com/series/11636

[132] L. Wright, S. Davidson, How to tell the difference between a model and a digital twin, Advanced Modeling and Simulation in Engineering Sciences 7 (1). doi:10.1186/s40323-020-00147-4.

[133] Gartner, Gartner Top 10 Strategic Technology Trends for 2020 (2019).
URL https://www.gartner.com/smarterwithgartner/gartner-top-10-strategic-technology-trends-for-2020.

[134] Y. Yi, Y. Yan, X. Liu, Z. Ni, J. Feng, J. Liu, Digital twin-based smart assembly process design and application framework for complex products and its case study, Journal of Manufacturing Systems 58 (PB) (2021) 94–107. doi:10.1016/j.jmsy.2020.04.013.

[135] J. Leng, D. Wang, W. Shen, X. Li, Q. Liu, X. Chen, Digital twins-based smart manufacturing system design in Industry 4.0: A review, Journal of Manufacturing Systems 60 (2021) 119–137. doi:10.1016/j.jmsy.2021.05.011.

[136] Y. Fan, J. Yang, J. Chen, P. Hu, X. Wang, J. Xu, B. Zhou, A digital-twin visualized architecture for Flexible Manufacturing System, Journal of Manufacturing Systems 60 (May) (2021) 176–201. doi:10.1016/j.jmsy.2021.05.010.

[137] S. Dittmann, P. Zhang, A. Glodde, F. Dietrich, Towards a scalable implementation of digital twins - A generic method to acquire shopfloor data, Procedia

CIRP 96 (2020) 157–162. doi:10.1016/j.procir.2021.01.069.
URL https://doi.org/10.1016/j.procir.2021.01.069

[138] B. Brenner, V. Hummel, Digital Twin as Enabler for an Innovative Digital Shopfloor Management System in the ESB Logistics Learning Factory at Reutlingen - University, Procedia Manufacturing 9 (2017) 198–205. doi:10.1016/j.promfg.2017.04.039.
URL http://dx.doi.org/10.1016/j.promfg.2017.04.039

[139] W. Luo, T. Hu, Y. Ye, C. Zhang, Y. Wei, A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin, Robotics and Computer-Integrated Manufacturing 65 (March) (2020) 101974. doi:10.1016/j.rcim.2020.101974.

[140] P. Aivaliotis, K. Georgoulias, Z. Arkouli, S. Makris, Methodology for enabling digital twin using advanced physics-based modelling in predictive maintenance, Procedia CIRP 81 (2019) 417–422. doi:10.1016/j.procir.2019.03.072.

[141] S. Liu, J. Bao, Y. Lu, J. Li, S. Lu, X. Sun, Digital twin modeling method based on biomimicry for machining aerospace components, Journal of Manufacturing Systems 58 (PB) (2021) 180–195. doi:10.1016/j.jmsy.2020.04.014.

[142] H. Yin Z, L. Wang, Application and Development Prospect of Digital Twin Technology in Aerospace, IFAC-PapersOnLine 53 (5) (2020) 732–737. doi:10.1016/j.ifacol.2021.04.165.

[143] A. Hänel, T. Schnellhardt, E. Wenkler, A. Nestler, A. Brosius, C. Corinth, A. Fay, S. Ihlenfeldt, The development of a digital twin for machining processes for the application in aerospace industry, Procedia CIRP 93 (2020) 1399–1404. doi:10.1016/j.procir.2020.04.017.

[144] A. Prisacaru, E. O. Guerrero, B. Chimmineni, P. J. Gromala, Y.-H. Yang, B. Han, G. Q. Zhang, Towards virtual twin for electronic packages in automotive applications, Microelectronics Reliability 122 (May) (2021) 114134. doi:10.1016/j.microrel.2021.114134.

[145] P. K. Rajesh, N. Manikandan, C. S. Ramshankar, T. Vishwanathan, C. Sathishkumar, Digital Twin of an Automotive Brake Pad for Predictive Maintenance, Procedia Computer Science 165 (2019) (2019) 18–24. doi:10.1016/j.procs.2020.01.061.

[146] G. Bhatti, H. Mohan, R. Raja Singh, Towards the future of ssmart electric vehicles: Digital twin technology, Renewable and Sustainable Energy Reviews 141 (February) (2021) 110801. doi:10.1016/j.rser.2021.110801.

[147] C. Verdouw, B. Tekinerdogan, A. Beulens, S. Wolfert, Digital twins in smart farming, Agricultural Systems 189 (2021) 103046. doi:10.1016/j.agsy.2020.103046.

[148] F. Lareyre, C. Adam, M. Carrier, J. Raffort, Using Digital Twins for Precision Medicine in Vascular Surgery, Annals of Vascular Surgery 67 (August) (2020) e577–e578. doi:10.1016/j.avsg.2020.04.042.

[149] K. Gillette, M. A. Gsell, A. J. Prassl, E. Karabelas, U. Reiter, G. Reiter, T. Grandits, C. Payer, D. Štern, M. Urschler, J. D. Bayer, C. M. Augustin, A. Neic, T. Pock, E. J. Vigmond, G. Plank, A Framework for the generation of digital twins of cardiac electrophysiology from clinical 12-leads ECGs, Medical Image Analysis 71 (2021) 102080. doi:10.1016/j.media.2021.102080.

[150] T. Defraeye, G. Tagliavini, W. Wu, K. Prawiranto, S. Schudel, M. Assefa Kerisima, P. Verboven, A. Bühlmann, Digital twins probe into food cooling and biochemical quality changes for reducing losses in refrigerated supply chains, Resources, Conservation and Recycling 149 (June) (2019) 778–794. doi:10.1016/j.resconrec.2019.06.002.

[151] T. Defraeye, C. Shrivastava, T. Berry, P. Verboven, D. Onwude, S. Schudel, A. Bühlmann, P. Cronje, R. M. Rossi, Digital twins are coming: Will we need them in supply chains of fresh horticultural produce?, Trends in Food Science and Technology 109 (March 2020) (2021) 245–258. doi:10.1016/j.tifs.2021.01.025.

[152] J. A. Santos, M. R. Lopes, J. L. Viegas, S. M. Vieira, J. M. Sousa, Internal Supply Chain Digital Twin of a Pharmaceutical Company, IFAC-PapersOnLine 53 (2) (2020) 10797–10802. doi:10.1016/j.ifacol.2020.12.2864.

[153] Y. Wang, X. Wang, A. Liu, Digital twin-driven supply chain planning, Procedia CIRP 93 (2020) 198–203. doi:10.1016/j.procir.2020.04.154.

[154] X. Xie, Q. Lu, A. K. Parlikad, J. M. Schooling, Digital twin enabled asset anomaly detection for building facility management, IFAC-PapersOnLine 53 (3) (2020) 380–385. doi:10.1016/j.ifacol.2020.11.061.

[155] Q. Lu, L. Chen, S. Li, M. Pitt, Semi-automatic geometric digital twinning for existing buildings based on images and CAD drawings, Automation in Construction 115 (December 2019) (2020) 103183. doi:10.1016/j.autcon.2020.103183.
URL https://doi.org/10.1016/j.autcon.2020.103183

[156] G. Angjeliu, D. Coronelli, G. Cardani, Development of the simulation model for Digital Twin applications in historical masonry buildings: The integration between numerical and experimental reality, Computers and Structures 238 (2020) 106282. doi:10.1016/j.compstruc.2020.106282.

[157] A. Agouzoul, M. Tabaa, B. Chegari, E. Simeu, A. Dandache, K. Alami, Towards a Digital Twin model for Building Energy Management: Case of Morocco, Procedia Computer Science 184 (2021) 404–410. doi:10.1016/j.procs.2021.03.051.

[158] B. R. Barricelli, E. Casiraghi, D. Fogli, A survey on digital twin: Definitions, characteristics, applications, and design implications, IEEE Access 7 (Ml) (2019) 167653–167671. doi:10.1109/ACCESS.2019.2953499.

[159] F. Tao, H. Zhang, A. Liu, A. Y. Nee, Digital Twin in Industry: State-of-the-Art, IEEE Transactions on Industrial Informatics 15 (4) (2019) 2405–2415. doi:10.1109/TII.2018.2873186.

[160] Q. Qi, F. Tao, Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison, IEEE Access 6 (2018) 3585–3593. doi:10.1109/ACCESS.2018.2793265.

[161] R. Ganguli, S. Adhikari, The digital twin of discrete dynamic systems: Initial approaches and future challenges, Applied Mathematical Modelling 77 (2020) 1110–1128. doi:10.1016/j.apm.2019.09.036.

[162] D. A. T. G. F. Malykhina, Digital Twin Technology as a Basis of the Industry in Future, in: proceedings of the 18th PCSF Professional Culture of the Specialist of the Future, 2018, pp. 416–428.

[163] A. Fuller, Z. Fan, C. Day, C. Barlow, Digital twin: Enabling technologies, challenges and open research, IEEE Access 8 (2020) 108952–108971. doi:10.1109/ACCESS.2020.2998358.

[164] M. Borth, J. Verriet, G. Muller, Digital Twin Strategies for SoS, in: proceedings of the 2019 14th Annual Conference System of Systems Engineering (SoSE), IEEE, 2019, pp. 164–169. doi:978-1-7281-0457-7.
URL https://ieeexplore-ieee-org.ukzn.idm.oclc.org/stamp/stamp.jsp?tp={\&}arnumber=8753860

[165] R. Saracco, Digital Twins: Bridging Physical Space and Cyberspace, Computer 52 (12) (2019) 58–64. doi:10.1109/MC.2019.2942803.

[166] C. Constantinescu, S. Giosan, R. Matei, D. Wohlfeld, A holistic methodology for development of Real-Time Digital Twins, Procedia CIRP 88 (i) (2020) 163–166. doi:10.1016/j.procir.2020.05.029.

[167] A. Barni, A. Fontana, A. Menato, M. Sorlini, L. Canetta, Exploiting the Digital Twin in the Assessemt and Optimization of Sustainability Performances, in: Proceedings of the 2018 International Conference on Intelligent Systems, 2018, pp. 706–713.

[168] D. Jones, C. Snider, A. Nassehi, J. Yon, B. Hicks, Characterising the Digital Twin: A systematic literature review, CIRP Journal of Manufacturing Science and Technology 1 (2019). doi:10.1016/j.cirpj.2020.02.002.

[169] F. Tao, M. Zhang, Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing, IEEE Access 5 (2017) 20418–20427. doi:10.1109/ACCESS.2017.2756069.

[170] E. J. Tuegel, A. R. Ingraffea, T. G. Eason, S. M. Spottswood, Reengineering aircraft structural life prediction using a digital twin, International Journal of Aerospace Engineering 2011. doi:10.1155/2011/154798.

[171] W. Luo, T. Hu, W. Zhu, F. Tao, Digital twin modeling method for CNC machine tool, in: proceedings of the ICNSC 2018 - 15th IEEE International Conference on Networking, Sensing and Control, IEEE, 2018, pp. 1–4. doi:10.1109/ICNSC.2018.8361285.

[172] H. Park, A. Easwaran, S. Andalam, TiLA: Twin-in-the-loop architecture for cyber-physical production systems, in: Proceedings - 2019 IEEE International Conference on Computer Design, ICCD 2019, 2019, pp. 82–90. arXiv:2003.09370, doi:10.1109/ICCD46524.2019.00019.

[173] A. Ait-Alla, M. Kreutz, D. Rippel, M. Lütjen, M. Freitag, Simulation-based Analysis of the Interaction of a Physical and a Digital Twin in a Cyber-Physical Production System, IFAC-PapersOnLine 52 (13) (2019) 1331–1336. doi:10.1016/j.ifacol.2019.11.383.

[174] T. DebRoy, W. Zhang, J. Turner, S. S. Babu, Building digital twins of 3D printing machines, Scripta Materialia 135 (2017) 119–124. doi:10.1016/j.scriptamat.2016.12.005.
URL http://dx.doi.org/10.1016/j.scriptamat.2016.12.005

[175] M. Schluse, J. Rossmann, From Simulation to Experimentable Digital Twins, IEEE International Symposium on Systems Engineering (2016) 1–6.

[176] G. Sugumar, A. Mathur, Assessment of a method for detecting process anomalies using digital-twinning, in: Proceedings - 2019 15th European Dependable Computing Conference, EDCC 2019, IEEE, 2019, pp. 119–126. doi:10.1109/EDCC.2019.00031.

[177] A. Rasheed, O. San, T. Kvamsdal, Digital twin: Values, challenges and enablers from a modeling perspective, IEEE Access 8 (2020) 21980–22012. doi:10.1109/ACCESS.2020.2970143.

[178] C. Brosinsky, X. Song, D. Westermann, Digital Twin – Concept of a Continuously Adaptive Power System Mirror, 2019 International ETG-Congress (2019) 147–152.

[179] V. Damjanovic-Behrendt, A Digital Twin-Based Privacy Enhancement Mechanism for the Automotive Industry, in: proceedings of the 2018 International Conference on Intelligent Systems, 2018, pp. 272–279.

[180] M. Eckhart, A. Ekelhart, E. Weippl, Enhancing Cyber Situational Awareness for Cyber-Physical Systems through Digital Twins, in: proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, IEEE, 2019, pp. 1222–1225. doi:10.1109/ETFA.2019.8869197.

[181] L. Atorf, J. Rossmann, Interactive Analysis and Visualization of Digital Twins in High-Dimensional State Spaces, in: proceedings of the 2018 15th International Conference on Control, Automation, Robotics and Vision, ICARCV 2018, 2018, pp. 241–246. doi:10.1109/ICARCV.2018.8581126.

[182] A. A. Malozemov, V. N. Bondar, V. V. Egorov, G. A. Malozemov, Digital Twins Technology for Internal Combustion Engines Development, in: Proceedings - 2018 Global Smart Industry Conference, GloSIC 2018, IEEE, 2018, pp. 1–6. doi:10.1109/GloSIC.2018.8570162.

[183] C. Gehrmann, M. Gunnarsson, A Digital Twin Based Industrial Automation and Control System Security Architecture, IEEE Transactions on Industrial Informatics 16 (1) (2020) 669–680.

[184] H. Zipper, C. Diedrich, Synchronization of Industrial Plant and Digital Twin, in: proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, IEEE, 2019, pp. 1678–1681. doi:10.1109/ETFA.2019.8868994.

[185] Y. He, J. Guo, X. Zheng, From Surveillance to Digital Twin: Challenges and Recent Advances of Signal Processing for Industrial Internet of Things, IEEE Signal Processing Magazine 35 (5) (2018) 120–129. doi:10.1109/MSP.2018.2842228.

[186] M. Lermer, C. Reich, Creation of Digital Twins by Combining Fuzzy Rules with Artificial Neural Networks, in: IECON Proceedings (Industrial Electronics Conference), IEEE, 2019, pp. 5849–5854. doi:10.1109/IECON.2019.8926914.

[187] F. Jaensch, A. Csiszar, C. Scheifele, A. Verl, Digital Twins of Manufacturing Systems as a Base for Machine Learning, in: Proceedings of the 2018 25th International Conference on Mechatronics and Machine Vision in Practice, M2VIP 2018, IEEE, 2019, p. 6. doi:10.1109/M2VIP.2018.8600844.

[188] A. Khan, M. Dahl, P. Falkman, M. Fabian, Digital twin for legacy systems: Simulation model testing and validation, in: proceedings of the 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE), 2018, pp. 421–426. doi:10.1109/COASE.2018.8560338.

[189] Z. Xu, A. Easwaran, A game-theoretic approach to secure estimation and control for cyber-physical systems with a digital twin, in: Proceedings - 2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems, ICCPS 2020, 2020, pp. 20–29. doi:10.1109/ICCPS48487.2020.00010.

[190] Y. Xu, Y. Sun, X. Liu, Y. Zheng, A Digital-Twin-Assisted Fault Diagnosis Using Deep Transfer Learning, IEEE Access 7 (2019) 19990–19999. doi:10.1109/ACCESS.2018.2890566.

[191] X. Li, B. He, Y. Zhou, G. Li, Multisource Model-Driven Digital Twin System of Robotic Assembly, IEEE Systems Journal (2020) 1–10doi:10.1109/jsyst.2019.2958874.

[192] R. Bansal, M. A. Khanesar, D. Branson, Ant colony optimization algorithm for industrial robot programming in a digital twin, in: proceedings of the 2019 25th International Conference on Automation and Computing (ICAC), 2019, pp. 1–5. doi:10.23919/IConAC.2019.8895095.

[193] A. Vassiliev, V. Samarin, D. Raskin, E. Evseev, V. Veris, I. Peschinski, D. Cabezas, Y. Kurniawan, Designing the Built-In Microcontroller Control Systems of Executive Robotic Devices Using the Digital Twins Technology, in: Proceedings of 2019 International Conference on Information Management and Technology, ICIMTech 2019, IEEE, 2019, pp. 256–260. doi:10.1109/ICIMTech.2019.8843814.

[194] B. H. Huynh, H. Akhtar, M. K. Sett, A universal methodology to create digital twins for serial and parallel manipulators, in: Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, IEEE, 2019, pp. 3104–3109. doi:10.1109/SMC.2019.8914195.

[195] T. Moi, A. Cibicik, T. Rølvåg, Digital Twin Based Condition Monitoring of a Knuckle Boom Crane: an Experimental Study (2020). doi:10.1016/j.engfailanal.2020.104517.

[196] D. Guivarch, E. Mermoz, Y. Marino, M. Sartor, Creation of helicopter dynamic systems digital twin using multibody simulations, CIRP Annals 68 (1) (2019) 133–136. doi:10.1016/j.cirp.2019.04.041.

[197] Y. Fang, C. Peng, P. Lou, Z. Zhou, J. Hu, J. Yan, Digital-Twin-Based Job Shop Scheduling Toward Smart Manufacturing, IEEE Transactions on Industrial Informatics 15 (12) (2019) 6425–6435.

[198] R. H. Guerra, R. Quiza, A. Villalonga, J. Arenas, F. Castaño, Digital Twin-Based Optimization for Ultraprecision Motion Systems With Backlash and Friction, IEEE Access 7. doi:10.1109/ACCESS.2019.2928141.

[199] Y. Cai, B. Starly, P. Cohen, Y. S. Lee, Sensor Data and Information Fusion to Construct Digital-twins Virtual Machine Tools for Cyber-physical Manufacturing, Procedia Manufacturing 10 (2017) 1031–1042. doi:10.1016/j.promfg.2017.07.094.

[200] V. Souza, R. Cruz, W. Silva, S. Lins, V. Lucena, A Digital Twin Architecture Based on the Industrial Internet of Things Technologies, in: proceedings of the 2019 IEEE International Conference on Consumer Electronics, ICCE 2019, IEEE, 2019, p. 6. doi:10.1109/ICCE.2019.8662081.

[201] Y. Gao, H. Lv, Y. Hou, J. Liu, W. Xu, Real-time modeling and simulation method of digital twin production line, in: Proceedings of the 2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference, ITAIC 2019, 2019, pp. 1639–1642. doi:10.1109/ITAIC.2019.8785703.

[202] R. Zhao, D. Yan, Q. Liu, J. Leng, J. Wan, X. Chen, X. Zhang, Digital twin-driven cyber-physical system for autonomously controlling of micro punching system, IEEE Access 7 (2019) 9459–9469. doi:10.1109/ACCESS.2019.2891060.

[203] J. Liu, X. Du, H. Zhou, X. Liu, L. Li, F. Feng, A digital twin-based approach for dynamic clamping and positioning of the flexible tooling system, Procedia CIRP 80 (2019) 746–749. doi:10.1016/j.procir.2019.01.063.

[204] C. Moussa, K. Al-Haddad, B. Kedjar, A. Merkhouf, Insights into digital twin based on finite element simulation of a large hydro generator, in: Proceedings: IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society, IEEE, 2018, pp. 553–558. doi:10.1109/IECON.2018.8591653.

[205] S. Gandzha, D. Aminov, I. Kiessh, B. Kosimov, Application of Digital Twins Technology for Analysis of Brushless Electric Machines with Axial Magnetic Flux, in: Proceedings of the 2018 Global Smart Industry Conference, GloSIC 2018, IEEE, 2018, p. 6. doi:10.1109/GloSIC.2018.8570132.

[206] R. Brannvall, J. Sarkinen, J. Svartholm, J. Gustafsson, J. Summers, Digital twin for tuning of server fan controllers, in: proceedings IEEE International Conference on Industrial Informatics (INDIN), 2019, pp. 1425–1428. doi:10.1109/INDIN41052.2019.8972291.

[207] R. G. Alves, G. Souza, R. F. Maia, A. L. H. Tran, C. Kamienski, J.-P. Soininen, P. T. Aquino, F. Lima, A digital twin for smart farming, in: proceedins of the 2019 IEEE Global Humanitarian Technology Conference (GHTC), IEEE, 2020, pp. 1–4. doi:10.1109/ghtc46095.2019.9033075.

[208] J. Monteiro, J. Barata, M. Veloso, L. Veloso, J. Nunes, Towards sustainable digital twins for vertical farming, in: proceedings of the 2018 13th International Conference on Digital Information Management, ICDIM 2018, IEEE, 2018, pp. 234–239. doi:10.1109/ICDIM.2018.8847169.

[209] H. Zhang, R. Wang, C. Wang, Monitoring and Warning for Digital Twin-driven Mountain Geological Disaster, in: Proceedings of 2019 IEEE International Conference on Mechatronics and Automation, ICMA 2019, IEEE, 2019, pp. 502–507. doi:10.1109/ICMA.2019.8816292.

[210] G. S. Martínez, S. Sierla, T. Karhela, V. Vyatkin, Automatic generation of a simulation-based digital twin of an industrial process plant, in: proceedings: IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society, 2018, pp. 3084–3089. doi:10.1109/IECON.2018.8591464.

[211] P. Taranenko, D. Telegin, A. Yaushev, Development of a Digital Twin of the Mechanical Part of Coriolis Flowmeters Based on Frequency Response Functions, in: Proceedings - 2018 Global Smart Industry Conference, GloSIC 2018, IEEE, 2018, p. 6. doi:10.1109/GloSIC.2018.8570072.

[212] S. H. Khajavi, N. H. Motlagh, A. Jaribion, L. C. Werner, J. Holmstrom, Digital Twin: Vision, benefits, boundaries, and creation for buildings, IEEE Access 7 (2019) 147406–147419. doi:10.1109/ACCESS.2019.2946515.

[213] R. Ranjbar, E. Duviella, L. Etienne, J. M. Maestre, Framework for a digital twin of the Canal of Calais, Procedia Computer Science 178 (2019) (2020) 27–37. doi:10.1016/j.procs.2020.11.004.

[214] Y. Pan, L. Zhang, A BIM-data mining integrated digital twin framework for advanced project management, Automation in Construction 124 (July 2020) (2021) 103564. doi:10.1016/j.autcon.2021.103564.

[215] A. Villalonga, E. Negri, L. Fumagalli, M. Macchi, F. Castaño, R. Haber, Local Decision Making based on Distributed Digital Twin Framework, IFAC-PapersOnLine 53 (2) (2020) 10568–10573. doi:10.1016/j.ifacol.2020.12.2806.
URL https://doi.org/10.1016/j.ifacol.2020.12.2806

[216] Q. Min, Y. Lu, Z. Liu, C. Su, B. Wang, Machine Learning based Digital Twin Framework for Production Optimization in Petrochemical Industry, International Journal of Information Management 49 (October 2018) (2019) 502–519. doi:10.1016/j.ijinfomgt.2019.05.020.

[217] J. O'Sullivan, D. O'Sullivan, K. Bruton, A case-study in the introduction of a digital twin in a large-scale smart manufacturing facility, Procedia Manufacturing 51 (2019) (2020) 1523–1530. doi:10.1016/j.promfg.2020.10.212.
URL https://doi.org/10.1016/j.promfg.2020.10.212

[218] D. D'Amico, J. Ekoyuncu, S. Addepalli, C. Smith, E. Keedwell, J. Sibson, S. Penver, Conceptual framework of a digital twin to evaluate the degradation status of complex engineering systems, Procedia CIRP 86 (2020) 61–67. doi:10.1016/j.procir.2020.01.043.

[219] P. Stief, J.-Y. Dantan, A. Etienne, A. Siadat, Advancing digital twin implementation: a toolbox for modelling and Simulation, Elsevier 93 (2018) 7.
URL www.sciencedirect.comAvailableonlineatwww.sciencedirect.comwww.elsevier.com/locate/procedia2212-8271

[220] L. Hebei I . T . (Shanghai) Co ., Thermoelectric Cooler TEC1-12706 (2010).

[221] F. S. Inc., FLIR LEPTON ® Engineering Datasheet (2018).

[222] DFROBOT, Lattepanda Documentation (2018).
URL http://docs.lattepanda.com/

[223] J. Viola, P. Oziablo, Y. Q. Chen, An Experimental Networked Control System with Fractional Order Delay Dynamics, in: proceedings of the 2019 IEEE 7th

International Conference on Control, Mechatronics and Automation, ICCMA 2019, 2019, pp. 226–231. doi:10.1109/ICCMA46720.2019.8988779.

[224] J. Viola, A. Radici, S. Dehghan, Y. Chen, Low-cost real-time vision platform for spatial temperature control research education developments, in: Proceedings of the ASME Design Engineering Technical Conference, Vol. 9, 2019, p. 6. doi:10.1115/DETC2019-97664.

[225] V. I. Kubov, Y. Y. Dymytrov, R. M. Kubova, Ltspice-model of thermoelectric peltier-seebeck element, in: proceedings of the 2016 IEEE 36th International Conference on Electronics and Nanotechnology (ELNANO), 2016, pp. 47–51. doi:10.1109/ELNANO.2016.7493007.

[226] Mathworks Inc, Simulink Design Optimization (2020).
URL https://www.mathworks.com/products/sl-design-optimization.html

[227] J. M. Maciejowski, Model Discrimination Using an Algorithmic Information Criterion, Automatica 15 (1977).

[228] H. Akaike, A New Look at the Statistical Model Identification, IEEE Transactions on Automatic Control 19 (6) (1974) 716–723. doi:10.1109/TAC.1974.1100705.

[229] L. Ljung, System Identification: Theory for the User, Prentice Hall information and system sciences series, Prentice Hall PTR, 1999.
URL https://books.google.com.co/books?id=nHFoQgAACAAJ

[230] R. Pintelon, J. Schoukens, System Identification: A Frequency Domain Approach, Wiley, 2004.
URL https://books.google.com/books?id=KhonXGwETWsC

[231] K. Zhou, J. C. Doyle, Essentials of Robust Control, Prentice Hall Modular Series for Eng, Prentice Hall, 1998.
URL https://books.google.com/books?id=QviHQgAACAAJ

[232] W. M. Thorburn, Occam's razor, Mind 24 (2) (1915) 287–288.

[233] D. Gómez, E. J. Moya, E. Baeyens, Control performance assessment: A general survey, in: A. P. de Leon F. de Carvalho, S. Rodríguez-González, J. F. De Paz Santana, J. M. C. Rodríguez (Eds.), Distributed Computing and Artificial Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 621–628.

[234] F.-Y. Wang, Toward a paradigm shift in social computing: The acp approach, IEEE Intelligent Systems 22 (5) (2007) 65–67. doi:10.1109/MIS.2007.4338496.

[235] F. Y. Wang, Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications, IEEE Transactions on Intelligent Transportation Systems 11 (3) (2010) 630–638. doi:10.1109/TITS.2010.2060218.

[236] F.-y. Wang, N.-n. Zheng, D. Cao, C. M. Martinez, L. Li, T. Liu, Parallel Driving in CPSS : A Unified Approach for Transport Automation and Vehicle Intelligence, IEEE/CAA Journal of Automatica Sinica 4 (4) (2017) 577–587.

[237] M. Kang, F. Y. Wang, From parallel plants to smart plants: Intelligent control and management for plant growth, IEEE/CAA Journal of Automatica Sinica 4 (2) (2017) 161–166. doi:10.1109/JAS.2017.7510487.

[238] P. Yin, J.-H. Wang, Y. Chen, W. Fei-Yue, Parallel Measurements: A New Theory and Framework for Complex Measurement System and A Case Study, Acta Automatica Sinica 44 (3) (2018) 8.

[239] F. Y. Wang, J. Zhang, Q. Wei, X. Zheng, L. Li, PDP: Parallel dynamic programming, IEEE/CAA Journal of Automatica Sinica 4 (1) (2017) 1–5. doi:10.1109/JAS.2017.7510310.

[240] S. Shen, H. Lu, M. Sadoughi, C. Hu, V. Nemani, A. Thelen, K. Webster, M. Darr, J. Sidon, S. Kenny, A physics-informed deep learning approach for bearing fault detection, Engineering Applications of Artificial Intelligence 103 (November 2020).

[241] J. Yu, X. Liu, A Fault Detection Method based on Convolutional Gated Recurrent Unit Auto-encoder for Tennessee Eastman Process, Proceedings - 2020 Chinese Automation Congress, CAC 2020 (2020) 1234–1238doi:10.1109/CAC51589.2020.9326895.

[242] D. Neupane, J. Seok, Bearing fault detection and diagnosis using case western reserve university dataset with deep learning approaches: A review, IEEE Access 8 (2020) 93155–93178. doi:10.1109/ACCESS.2020.2990528.

[243] T. Pan, J. Chen, Z. Zhou, C. Wang, S. He, A Novel Deep Learning Network via Multiscale Inner Product With Locally Connected Feature Extraction for Intelligent Fault Detection, IEEE Transactions on Industrial Informatics 15 (9) (2019) 5119–5128. doi:10.1109/tii.2019.2896665.

[244] Y. Lei, Intelligent Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery, Xi'An Jiaotong University Press, 2017. doi:https://doi.org/10.1016/C2016-0-00367-4.

[245] I. Lomov, M. Lyubimov, I. Makarov, L. E. Zhukov, Fault detection in Tennessee Eastman process with temporal deep learning models, Journal of Industrial Information Integration 23 (March). doi:10.1016/j.jii.2021.100216.

[246] V. Rjabtsikov, A. Rassolkin, B. Asad, T. Vaimann, A. Kallaste, V. Kuts, S. Jegorov, M. Stepien, M. Krawczyk, Digital Twin Service Unit for AC Motor Stator Inter-Turn Short Circuit Fault Detection, 2021 28th International Workshop on Electric Drives: Improving Reliability of Electric Drives, IWED 2021 - Proceedingsdoi:10.1109/IWED52055.2021.9376328.

[247] H. Huang, L. Yang, Y. Wang, X. Xu, Y. Lu, Digital Twin-driven online anomaly detection for an automation system based on edge intelligence, Journal of Manufacturing Systems 59 (February) (2021) 138–150. doi:10.1016/j.jmsy.2021.02.010.
URL https://doi.org/10.1016/j.jmsy.2021.02.010

[248] K. Pichler, R. Haas, C. Kastl, A. Plockinger, P. Foschum, Comparison of fault detection methods for a hydraulic accumulator loading circuit, in: Proceedings - 2020 IEEE Conference on Industrial Cyberphysical Systems, ICPS 2020, 2020, pp. 117–122. doi:10.1109/ICPS48405.2020.9274787.

[249] H. Darvishi, D. Ciuonzo, E. R. Eide, P. S. Rossi, Sensor-Fault Detection, Isolation and Accommodation for Digital Twins via Modular Data-Driven Architecture, IEEE Sensors Journal 21 (4) (2021) 4827–4838. doi:10.1109/JSEN.2020.3029459.

[250] J. Coble, J. Wesley Hines, Identifying optimal prognostic parameters from data: A genetic algorithms approach, in: proceedings of the Annual Conference of the Prognostics and Health Management Society, PHM 2009, 2009, pp. 1–11.

[251] F. Elasha, S. Shanbr, X. Li, D. Mba, Prognosis of a wind turbine gearbox bearing using supervised machine learning, Sensors (Switzerland) 19 (14) (2019) 1–17. doi:10.3390/s19143092.

[252] F. K. Moghadam, A. R. Nejad, Online condition monitoring of floating wind turbines drivetrain by means of digital twin, Mechanical Systems and Signal Processing 162 (2022) 108087. doi:10.1016/j.ymssp.2021.108087.

[253] W. Wang, J. Wang, J. Tian, J. Lu, R. Xiong, Application of Digital Twin in Smart Battery Management Systems, Chinese Journal of Mechanical Engineeringdoi:10.1186/s10033-021-00577-0.
URL https://doi.org/10.1186/s10033-021-00577-0

[254] X. Qu, Y. Song, D. Liu, X. Cui, Y. Peng, Lithium-ion battery performance degradation evaluation in dynamic operating conditions based on a digital twin model, Microelectronics Reliability 114 (July). doi:10.1016/j.microrel.2020.113857.

[255] P. D. Domański, Y. Chen, M. Ławryńczuk (Eds.), Outliers in Control Engineering: Fractional Calculus Perspective, De Gruyter, 2022. doi:doi:10.1515/9783110729122.
URL https://doi.org/10.1515/9783110729122

[256] C. Bigoni, J. S. Hesthaven, Simulation-based Anomaly Detection and Damage Localization: an application to Structural Health Monitoring, Computer Methods in Applied Mechanics and Engineering 363. doi:10.1016/j.cma.2020.112896.

[257] Y. Sun, J. Reichelt, T. Bormann, A. Gondorf, A multi-step wafer-level run-to-run controller with sampled measurements for furnace deposition and CMP process flows: APC: Advanced process control, in: proceedings of the 2016 27th Annual SEMI Advanced Semiconductor Manufacturing Conference, ASMC 2016, IEEE, 2016, pp. 399–402. doi:10.1109/ASMC.2016.7491084.

[258] F. Tan, T. Pan, Z. Li, S. Chen, Survey on run-to-run control algorithms in high-mix semiconductor manufacturing processes, IEEE Transactions on Industrial Informatics 11 (6) (2015) 1435–1444. doi:10.1109/TII.2015.2490039.

[259] D. EMC, Dell Gateway E3001 Datasheet (2022).

[260] HP, HP Edgeline EL300 (2022).
URL https://buy.hpe.com/us/en/servers/edgeline-systems/edgeline-systems/edgeline-converged-edge-systems/hpe-edgeline-el300-converged-edge-system/p/1011127891

[261] UP, UP Xtrem i11 Board (2022).
URL https://up-shop.org/up-xtreme-i11-boards-0001-series.html

[262] Raspberry Pi Foundation, Raspberry Pi 4 (2022).
URL https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

[263] HardKernel, Odroid board (2022).
URL https://www.hardkernel.com/

[264] Nvidia Corporation, Nvidia Jetson (2022).
URL https://www.nvidia.com/en-us/autonomous-machines/jetson-store/

[265] M. Calderon, Digital Twin Technology for modeling , simulation and control of smart mechatronic system, Ph.D. thesis, University of Brescia (2020).

[266] P. B. Khaled Nassim, S. Affan, Digital Twin Development and Deployment on the Cloud, Academic Press, 2020. doi:10.1016/c2019-0-03782-x.

[267] Y. Q. Chen, Fractional calculus , delay dynamics and networked control systems, Proceedings of the 3rd International Symposium on Resilient Control Systems.

[268] C. L. Nikias, M. Shao, Signal Processing with Alpha-stable Distributions and Applications, Wiley-Interscience, New York, NY, USA, 1995.

[269] J. Smoot, arcTEC™ Structure - Improving Performance and Life Span in Peltier Modules (2019).
URL https://www.cuidevices.com/catalog/resource/arctec-structure-improved-performance-and-life-span-in-peltier-modules.pdf

[270] J. A. Nelder, R. Mead, A Simplex Method for Function Minimization, The Computer Journal 7 (4) (1965) 308–313. doi:10.1093/comjnl/7.4.308.

[271] S. Berezina, O. Solonets, M. Bortsova, Referencing of U A v Images Using the Nelder-Mead Method, in: proceedings of the International Conference on Mathematical Methods in Electromagnetic Theory, MMET, Vol. 2018-July, IEEE, 2018, pp. 103–106. doi:10.1109/MMET.2018.8460283.

[272] V. Sinlapakun, W. Assawinchaichote, Optimized PID controller design for electric furnace temperature systems with Nelder Mead Algorithm, in: proceedings of the ECTI-CON 2015 - 2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, IEEE, 2015, pp. 39–42. doi:10.1109/ECTICon.2015.7206925.

[273] J. Ehrenberger, J. Svec, Nelder Mead method used in protection setting optimization, in: Proceedings of the 2016 17th International Scientific Conference on Electric Power Engineering, EPE 2016, IEEE, 2016, p. 6. doi:10.1109/EPE.2016.7521806.

[274] A. B. Iztok Fajfar, Janez Puhan, Evolving a Nelder–Mead algorithm for optimization with genetic programming, Evolutionary Computation 25 (3) (2017) 351–373. doi:10.1162/EVCO_a_00174.

[275] H. Xia, An improved wolf search algorithm introduced by nelder-mead operator, in: proceedings of the 2018 3rd IEEE International Conference on Cloud Computing and Big Data Analysis, ICCCBDA 2018, IEEE, 2018, pp. 11–14. doi:10.1109/ICCCBDA.2018.8386459.

[276] E. J. Wyers, M. B. Steer, C. T. Kelley, P. D. Franzon, A bounded and discretized Nelder-Mead algorithm suitable for RFIC calibration, IEEE Transactions on Circuits and Systems I: Regular Papers 60 (7) (2013) 1787–1799. doi:10.1109/TCSI.2012.2230496.

[277] F. Gao, L. Han, Implementing the Nelder-Mead simplex algorithm with adaptive parameters, Computational Optimization and Applications 1 (1). doi:10.1007/s10589-010-9329-3.

[278] D. Xue, Y. Chen, D. Atherton, Linear Feedback Control, Analysis and design with MATLAB - Advances in Design and Control, SIAM, 2007.

[279] Y. Luo, Y. Q. Chen, Fractional Order Motion Controls, Wiley, 2012.
URL https://books.google.com.co/books?id=AYnucAN0tQQC

[280] J. Viola, Y. Chen, A Self Optimizing Control Framework and A Benchmark for Smart Process Control, in: proceedings of the IEEE 3rd International Conference on Industrial Artificial Intelligence, 2021, p. 6.

[281] J. Wei, Y. Q. Chen, Y. Yu, Y. Chen, Optimal randomness in swarm-based search, Mathematics 7 (9) (2019) 1–19. arXiv:1905.02776, doi:10.3390/math7090828.

[282] K. Marti, Optimization Under Stochastic Uncertainty: Methods, Control and Random Search Methods, Vol. 12118 LNCS, Springer, 2020. doi:10.1007/978303061792915.

[283] C. J. Price, I. D. Coope, D. Byatt, A convergent variant of the Nelder-Mead algorithm, Journal of Optimization Theory and Applications 113 (1) (2002) 5–19. doi:10.1023/A:1014849028575.

[284] G. Rappl, Konvergenzraten von Random-Search-Verfahren zur Globalen Optimierung. Dissertation, Phd dissertation, Universität der Bundeswehr München (1984).

[285] I. D. Coope, C. J. Price, Frame based methods for unconstrained optimization, Journal of Optimization Theory and Applications 107 (2) (2000) 261–274, copyright - Plenum Publishing Corporation 2000; Last updated - 2021-09-11.
URL `https://www.proquest.com/scholarly-journals/frame-based-methods-unconstrained-optimization/docview/196608832/se-2?accountid=14515`

[286] David Byatt, Convergent Variants of the Nelder Mead Algorithm, Ph.D. thesis, University of Canterbury (2000).

[287] J. C. Lagarias, B. Poonen, M. H. Wright, Convergence of the restricted nelder–mead algorithm in two dimensions, SIAM Journal on Optimization 22 (2) (2012) 501–532. doi:`https://doi.org/10.1137/110830150`.

[288] J. C. Lagarias, J. A. Reeds, M. H. Wright, P. E. Wright, Convergence properties of the nelder–mead simplex method in low dimensions, SIAM Journal on Optimization 9 (1) (1998) 112–147. doi:`10.1137/S1052623496303470`.

[289] K. I. M. McKinnon, Convergence of the nelder–mead simplex method to a nonstationary point, SIAM Journal on Optimization 9 (1) (1998) 148–158. doi:`10.1137/S1052623496303482`.

[290] C. Boender, Bayesian Stopping Rules For Multistart Global Optimization Methods, Mathematical Programming 37 (1987) 59–80.

[291] D. C. Karnopp, Random Search Techniques for Optimization Problems, Automatica I (1961) 111–121.

[292] S. Saab, R. Kaed-Bey, Parameter identification of a dc motor: an experimental approach, in: proceedings of the ICECS 2001. 8th IEEE International Conference on Electronics, Circuits and Systems (Cat. No.01EX483), Vol. 2, 2001, pp. 981–984 vol.2. doi:10.1109/ICECS.2001.957638.

[293] Mathworks Inc, Simulink Design Optimization ™ Getting Started Guide R 2015 b (2015).

[294] MATLAB, Matlab web app server (`https://www.mathworks.com/products/matlab-web-app-server.html`).

[295] J. Viola, Y. Chen, Parallel Self Optimizing Control Framework for Digital Twin Enabled Smart Control Engineering, in: The First IEEE International Conference on Digital Twins and Parallel Intelligence (DTPI 2021), 2021, p. 4.

[296] D. Hollenbeck, Y. Chen, A more optimal stochastic extremum seeking control using fractional dithering for a class of smooth convex functions, in: 2020 IFAC World Congress, 2020, pp. 1–6.

[297] C. Yin, Y. Chen, S. M. Zhong, Fractional-order sliding mode based extremum seeking control of a class of nonlinear systems, Automatica 50 (12) (2014) 3173–3181. doi:http://dx.doi.org/10.1016/j.automatica.2014.10.027.

[298] W. A. E. M. Ahmed, H. M. A. Mageed, S. A. Mohamed, A. A. Saleh, Fractional order darwinian particle swarm optimization for parameters identification of solar pv cells and modules, Alexandria Engineering Journal 61 (2) (2022) 1249–1263. doi:https://doi.org/10.1016/j.aej.2021.06.019.

[299] Y. Chen, Q. Gao, Y. Wei, Y. Wang, Study on fractional order gradient methods, Applied Mathematics and Computation 314 (2017) 310–321. doi:10.1016/j.amc.2017.07.023.

[300] H. Niu, J. Wei, Y. Chen, Optimal randomness for stochastic configuration network (SCN) with heavy-tailed distributions, Entropy 23 (1) (2021) 1–13. doi:10.3390/e23010056.

[301] Z. Zhang, J. Lu, L. Xia, S. Wang, H. Zhang, R. Zhao, Digital twin system design for dual-manipulator cooperation unit, in: proceedings of 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2020, 2020, pp. 1431–1434. doi:10.1109/ITNEC48623.2020.9084652.

[302] R. L. Magin, Fractional calculus in bioengineering., Critical reviews in biomedical engineering 32 (1) (2004) 1–104. doi:10.1615/critrevbiomedeng.v32.i1.10.

[303] H. Sheng, Y. Chen, T. Qiu, Fractional Processes and Fractional-Order Signal Processing Techniques and Applications, Springer, 2012.

[304] F. Y. Wang, L. Q. Yang, J. Yang, Y. Zhang, S. Han, K. Zhao, Urban intelligent parking system based on the parallel theory, in: proceedings of the 2016 International Conference on Computing, Networking and Communications, ICNC 2016, IEEE, 2016, p. 6. doi:10.1109/ICCNC.2016.7440708.

[305] S. Wang, J. Wang, X. Wang, T. Qiu, Y. Yuan, L. Ouyang, Y. Guo, F. Y. Wang, Blockchain-Powered Parallel Healthcare Systems Based on the ACP Approach, IEEE Transactions on Computational Social Systems 5 (4) (2018) 942–950. doi:10.1109/TCSS.2018.2865526.

[306] J. Spall, Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control, 1st Edition, John Wiley & Sons, Incorporated, 2003.

[307] W. Ai, X. Li, S. Tian, A novel SPSA-based IMC-PID Data-driven Control Method, in: proceedings of the Chinese Control and Decision Conference (CCDC), IEEE, 2016, pp. 4981–4986.

[308] Priyatmadi, A. P. Sandiwan, H. Wijaya, A. Cahyadi, Application of SPSA LQR tuning on quadrotor, Proceedings - 2016 6th International Annual Engineering Seminar, InAES 2016 (2017) 32–36doi:10.1109/INAES.2016.7821902.

[309] Y. F. Ning, W. S. Tang, S. U. Lei, Comparison between hybrid genetic-SPSA algorithm and GA for solving random fuzzy dependent-chance programming, in: proceedings of the 2005 International Conference on Machine Learning and Cybernetics, ICMLC 2005, 2005, pp. 2742–2746. doi:10.1109/icmlc.2005.1527409.

[310] Z. Huajun, Z. Jin, G. Tao, Convergence accelerated by the improvements of stepsize and gradient in SPSA, in: Proceedings of the 2011 Chinese Control and Decision Conference, CCDC 2011, IEEE, 2011, pp. 1–6. doi:10.1109/CCDC.2011.5968131.

[311] Y. Fan, T. Liu, Parallel implementation of simultaneous perturbation stochastic approximation with adaptive step sizes, in: ACM International Conference Proceeding Series, 2018, pp. 3–6. doi:10.1145/3242840.3242845.

[312] H. Zhao, T. Liu, A parallelized combined direction simultaneous perturbation stochastic approximation algorithm, in: proceedings of the 2017 2nd IEEE International Conference on Computational Intelligence and Applications, ICCIA 2017, 2017, pp. 141–144. doi:10.1109/CIAPP.2017.8167196.

[313] V. Piroumian, Digital Twins: Universal Interoperability for the Digital Age, Computer 54 (1) (2021) 61–69. doi:10.1109/MC.2020.3032148.

[314] International Organization for Standardization, ISO/IEEE 11073-10101:2020 Health informatics — Device interoperability — Part 10101 (2020).

[315] International Organization for Standardization, ISO 23247-2:2021 Automation systems and integration — Digital twin framework for manufacturing (2021). URL https://www.iso.org/standard/78743.html

[316] (ANSI) American National Standards Institute, Automation Systems And Integration−Digital Twin Framework For Manufacturing (2021).

[317] L. Ljung, Analysis of Recursive Stochastic Algorithms, IEEE Transactions on Automatic Control 22 (4) (1977) 551–575. doi:10.1109/TAC.1977.1101561.

[318] H. J. Kushner, G. G. Yin, Stochastic Approximation Algorithms and Applications, Springer, 1997. doi:10.1007/978-1-4899-2696-8.

[319] J. Viola, Y. Chen, An Accelerated Self Optimizing Control Framework for Smart Process Control Using Fractional Order Stochasticity, in: proceedings of the 9th International Conference on Control, Mechatronics and Automation ICCMA 2021, IEEE, 2021, pp. 104–109.

[320] M. B. Naumovic, D. Zivanovic, Remote experiments in control engineering education laboratory., Int. J. Online Eng. 4 (2) (2008) 48–53.

[321] M. Popescu, N. Paraschiv, Remote laboratory for learning in control engineering, IFAC Proceedings Volumes 46 (17) (2013) 333–338.

[322] K.-M. Chao, A. E. James, A. G. Nanos, J.-H. Chen, S.-D. Stan, I. Muntean, G. Figliolini, P. Rea, C. B. Bouzgarrou, P. Vitliemov, et al., Cloud e-learning for mechatronics: Clem, Future Generation Computer Systems 48 (2015) 46–59.

[323] A. Rojko, D. Hercog, K. Jezernik, Mechatronics e-course for regular students and adults: Realization and comparison of efficiency, in: proceedigns of the IEEE EDUCON 2010 Conference, IEEE, 2010, pp. 959–966.

[324] A. Rojko, D. Hercog, K. Jezernik, E-training in mechatronics using innovative remote laboratory, Mathematics and Computers in Simulation 82 (3) (2011) 508–516.

[325] K. Geevarghese, K. Gangadharan, et al., Design and implementation of remote mechatronics laboratory for e-learning using labview and smartphone and cross-platform communication toolkit (scct), Procedia Technology 14 (2014) 108–115.

[326] H. Temeltas, M. Gokasan, S. Bogosyan, A. Kilic, Hardware in the loop simulation of robot manipulators through internet in mechatronics education, in: proceedings of the IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02, Vol. 4, IEEE, 2002, pp. 2617–2622.

[327] RoboholicManiacs, Roboholic maniacs: Remote lab (https://www.roboholicmaniacs.com/remotelab, 2020).

[328] P. Martin, Quanser remote labs: Tools and techniques (https://www.quanser.com/blog/remote-labs-tools-and-techniques/, 2020).

[329] F. Guc, J. Viola, Y. Chen, Digital twins enabled remote laboratory learning experience for mechatronics education, in: Proceedings of the 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI), IEEE, 2021, pp. 242–245.