ECSEL Joint Undertaking
Electronic Components and Systems for European Leadership

iMOCO4.E

Intelligent Motion Control under Industry 4.E

D4.1 Requirements for advanced motion control (first iteration)

Due Date: M11 – 2022-07-31

Abstract:

The deliverable is dedicated to the description of the initial requirements for the control layer (Layer 2), which are partially also based on inputs from WP2 (in particular, D2.1 and D2.2), while working in parallel and considering D7.1 and D7.2. This report will summarize the control layer requirements specific for the relevant BBs (BB4, 5 and 10), pilots, demonstrators and use-cases.

Project Information

Grant Agreement Number	**101007311**
Project Acronym	IMOCO4.E
Project Full Title	Intelligent Motion Control under Industry 4.E
Starting Date	1st September 2021
Duration	36 months
Call Identifier	H2020-ECSEL-2020-2-RIA-two-stage
Topic	ECSEL-2020-2-RIA
Project Website	https://www.imoco4e.eu/
Project Coordinator	Arend-Jan Beltman
Organisation	SIOUX TECHNOLOGIES BV (SIOUX)
Email	Arend-Jan.Beltman@sioux.eu

Document Information

Work Package	WP4 Smart Control Layer design and development							
Lead Beneficiary	University of Granada (UGR)							
Deliverable Title	Requirements for advanced motion control (first iteration)							
Version	1.2							
Date of Submission	31/07/2022							
Author(s)	Navarro Cabrera, Diego (UGR), diegonavaca@ugr.es Luque Sola, Niceto (UGR), nluque@ugr.es Ros Vidal, Eduardo (UGR), eros@ugr.es							
Contributor(s)	See history table below							
Internal Reviewer(s)	Armendia, Mikel (TEK), mikel.armendia@tekniker.es Visioli Antonio (UNIBS), antonio.visioli@unibs.it							
Document Classification	**Draft**		**Final**	**X**				
Deliverable Type	**R**	X	**DEM**		**DEC**		**OTHER**	
Dissemination Lever	**PU**	X	**CO**		**CI**			

History					
Version	**Issue Date**	**Status**	**Distribution**	**Author**	**Comments**
1.0	10/05/2022	Draft	PU	Diego Navarro, Niceto Luque, Eduardo Ros, Mikel Armendia	Structure of the deliverable
1.1	11/07/2022	Draft	PU	Diego Navarro, Martin Čech, Martin Goubej, Jana Konigsmarková, Mikel Armendia, Carlos Yurre, Javier Arenas, Davide Colombo, Paolo Grande, Jorge Sánchez, Marco Fuentes, Eduardo Ros, Niceto	Partners contribution integration

				Luque, Pavel Balda, Roman Pišl, Tomáš Myslivec, Manuel Beschi, Antonio Visioli, Benjamin Rouxel, Marco Solieri, Claudio Scordino, Lorenzo Diana, Francesca Palumbo, Raquel Lazcano, Daniel Madroñal, Gert Witvoet, Tom Oomen, Max van Haren, Max van Meer, Dip Goswami, Kees Goossens, Martijn Koedam, Mart Coenen, Marcel van der Kraan, Jaap van der Voet, Henry Stoutjesdijk, Marco Alonso, Davide Plaia, Thorsten Hehn, Rahul Tomar, Biraj Harnal, Gregor Gebhardt,Matthias Schneider, Reinhard Kulke, Lars Meyer, Ansgar Bergmann, Jan Wytze van der Weit, Pekka Kilpeläinen, Cristi Irimia, Calin Husar, Frack Sellier, Minhao Yang, Jana Jost, Fabian Menebröker, Dennis Lünsch, Marc Hantzsch, Kalle Määttä, Pirkka Tukeva, Arttu Pulli, Marek Kysela, Gijs van der Veen, Sajid Mohamed	
1.2	31/07/2022	Final	PU	Antonio Visioli, Mikel Armendia, Diego Navarro	Internal revision

Type of Contribution	
Partner	**Description of Contribution to Contents**
TEK/UGR	Document structure definition
UGR	Contributions integration
SIEMENS	XiL methodology and Digital Twins
UNIBS/ UWB/ GDM	Vibration control
UNIBS/ GDM/ UWB/ TUE	Repetitive control
UGR	Compliant control
UGR/TEK	Impedance control
TEK	Friction compensation
TEK	MPC control
TUE/ FAG/TEK	LPTV and MIMO systems
TUE	Multi-rate systems

Table of Contents

1 Introduction.....13
1.1 Purpose of this Deliverable.....13
1.2 Structure of this Deliverable.....13
1.3 Intended readership.....13
2 IMOCO4.E framework overview.....14
3 XIL and Digital Twins tools for the development of motion control systems (Task 4.2).....15
3.1 Introduction.....15
3.2 State of the Art.....16
3.2.1 *xIL* methodology.....16
3.2.2 Reduced Order Model.....17
3.3 Progress beyond the SoA.....19
Definition of xIL methodology for the development of motion control algorithms.....19
Development of reduced models for control algorithm development and implementation....19
3.4 Requirements.....19
4 Smart control algorithms library (BB5/Task4.3/Task4.4).....20
4.1 Introduction.....20
4.2 Control functionalities.....20
4.2.1 Vibration control for mechatronic systems.....20
4.2.1.1 Introduction.....20
Active and passive vibration damping using input shaping and auxiliary sensors.....20
Vibration damping in time-varying elastic systems.....20
4.2.1.2 State of the art.....20
4.2.1.3 Progress beyond the SoA.....25
4.2.1.4 Requirements.....25
4.2.2 Repetitive control.....26
4.2.2.1 Introduction.....26
Torque ripple in switched reluctance motors.....26
Implementation issues in repetitive control schemes – robust, inferential and delay-varying methods.....26
Position-dependent repetitive disturbance estimation and compensation.....27

4.2.2.2 State of the art....27
Torque ripple in switched reluctance motors....29
Implementation issues in repetitive control schemes – robust, inferential and delay-varying methods....29
Position-dependent repetitive disturbance estimation and compensation....30
4.2.2.3 Progress beyond the SoA....30
Torque ripple in switched reluctance motors....30
Implementation issues in repetitive control schemes – robust, inferential and delay-varying methods....30
Position-dependent repetitive disturbance estimation and compensation....31
4.2.2.4 Requirements....31
4.2.3 Impedance or compliant control for robot manipulators....32
4.2.3.1 Introduction....32
4.2.3.2 State of the art....33
Classical algorithm control limitations based on analytical cobot model....33
Considerations when modelling an elastic robot....34
Force control, understanding the problem (Impedance control)....34
Indirect Active Control....35
Impedance Control....35
Mechanical Impedance....36
Design of an impedance controller....37
Coefficient design....38
Machine Learning (ML)-dynamic identification....38
ML dynamic modeling using recurrent neural networks (RNNs)....39
ML-based control. Controllers based on Reinforcement learning (RL)....39
Controllers based on Supervised learning....40
Iterative Learning Control....41
4.2.3.3 Progress beyond the SoA....41
Exploring different schemes for acquiring the dynamic cobot models....41
Building inverse and forward dynamic models....42
Accelerate learning....42
Integrating machine learning into the control loop....42

4.2.3.4 Requirements....42
4.2.4 Disturbance Observer for Friction Compensation....43
4.2.4.1 Introduction....43
4.2.4.2 State of the art....44
Static friction models....44
Dynamic friction models....45
Friction Compensation Methods....46
4.2.4.3 Progress beyond the SoA....47
4.2.4.4 Requirements....49
4.2.5 MPC Control....49
4.2.5.1 Introduction....49
4.2.5.2 State of the art....51
4.2.5.3 Progress beyond SoA....51
4.2.5.4 Requirements....53
4.2.6 Control for linear parameter-varying or time-varying systems....53
4.2.6.1 Introduction....53
4.2.6.2 State of the Art....53
LPV system representation....53
Data-driven parametric LPV identification....53
Inverse and feedforward for LPV systems....54
4.2.6.3 Progress beyond the SoA....54
Data-driven grey-box methods for LPV feedforward....54
Data-driven inverse LPV-FIR identification using machine learning techniques....55
4.2.6.4 Requirements....55
4.2.7 MIMO Control....55
4.2.7.1 Introduction....55
4.2.7.2 State of the art....56
LQR & DAC structures....57
MPC structure....57
Other strategies....58
4.2.7.3 Progress beyond SoA....58
4.2.7.4 Requirements....61

4.2.8 Control and identification of multi-rate systems....61
4.2.8.1 Introduction....61
4.2.8.2 State of the art....61
4.2.8.3 Progress beyond SoA....62
4.2.8.4 Requirements....63
4.3 Generic BB5 requirements....63
5 Path planning, trajectory generation, obstacle avoidance, navigation (BB10, Task 4.5)....64
5.1 Introduction....64
5.2 Control functionalities....65
5.2.1 Multi-sensory data input with sensor-fusion for extended environment detection....65
5.2.1.1 Introduction....65
5.2.1.2 State of the art....65
Radar processing....65
Camera based detection and visual servoing....66
5.2.1.3 Progress beyond the SoA....66
Radar processing....66
Camera based detection and visual servoing in boom control....66
5.2.1.4 Requirements....67
5.2.2 Real-time decision making in path planning....67
5.2.2.1 Introduction....67
5.2.2.2 State of the art....67
Global path planning....67
Local path planning....68
Collision avoidance....69
5.2.2.3 Progress beyond the SoA....70
Extend VDA5050 standard....70
Scale Global Path Planning....71
Integration of reserved subspaces into local planner....71
Advance control and planning methods for collision avoidance....71
Custom solution inverse kinematics of the charging boom....71
5.2.2.4 Requirements....72
5.3 Generic BB10 requirements....72

6 Real-Time Smart-Control Platform (BB4, Task 4.6)....73
6.1 Introduction....73
6.2 State of the Art....74
6.2.1 Hardware platforms....74
6.2.2 Software Virtualization....75
6.2.3 Real time communication....76
6.3 Progress beyond the SoA....77
6.4 Generic BB4 requirements....77
7 IMOCO4.E Building Block connections....78
8 Conclusion....80
9 References....81

List of Figures

Figure 1: Outline of different technologies and approaches related to Smart control and covered in D4.1. They are structured in groups with respect to their relation specific BBs....9

Figure 2: IMOCO4.E reference framework architecture viewpoint – initial version....11

Figure 3: AI viewpoint with BB interactions (from D2.3)....12

Figure 4: Evolution of product engineering....13

Figure 5: MiL, SiL and HiL phases in the V-diagram....14

Figure 6: ROM workflow....15

Figure 7: Creation Reduced Order Model: different methodologies....16

Figure 8: Mechanical damping using a) vibration isolator b) vibration absorber....18

Figure 9: Principle of Zero-vibration input shaping methods....19

Figure 10: Principle of active vibration control....20

Figure 11: Structure of industrial motion control system....21

Figure 12: Dynamic model of a lift [Roberts 1998]....21

Figure 13: Conventional structure of the repetitive controller R – plug-in type parallel control scheme complementing the standard feedback loop....25

Figure 14: Modified structure of the repetitive controller R – introduction of learning filter L providing phase compensation to improve disturbance rejection....26

Figure 15: Angle-based RC controller....27

Figure 16: Spring-damped-system....34

Figure 17: Two-mass system model....44

Figure 18: System structure with controller and friction estimator....46

Figure 19: Schematics of MPC....47

Figure 20: Schematics of control structures for DSDs....49

Figure 21: LQR&DAC controller structure....58

Figure 22: Sampled-data control structure with continuous-time plant, hold and sample operations....58

Figure 23: Approximation of sampled-data control by sampling the plant arbitrarily high....59

Figure 24: Multi-rate feedforward control: a high-rate and low-rate controller communicating in order to process high-rate signals with a low-rate control-loop....59

Figure 25: Radar sensor from RoKoRa project....62

Figure 26: Sensor unit with 77 GHz radar, camera and mini-PC from VIDETEC project....62

List of Tables

Table 1: Abbreviations table....11

Table 2: Vibration control requirements....26

Table 3: Repetitive control requirements....32

Table 4: Compliant control requirements....43

Table 5: Friction compensation requirements....49

Table 6: Model-based predictive control requirements....53

Table 7: Linear parameter-varying systems requirements....55

Table 8: BB5 requirements....66

Table 9: Multi-sensory data input and sensor-fusion requirements....68

Table 10: Path planning and real-time decision-making requirements....73

Table 11: BB10 requirements....75

Table 12: BB4 requirements....80

Table 13: Building block connection matrix....81

Abbreviations

Table 1: Abbreviations table

Abbreviation	**Explanation**
AGV	Automated Gated Vehicles
AI	Artificial Intelligence
ANN	Artificial Neural-Network
APD	Active Platform Damping
API	Application Programming Interface
ATD	Active Tower Damping
CAN	Controller Area Network
CARP	Context Aware Route Planning
CGR	Coarse Grain Reconfigurable
COTS	Commertial Of-The-Shelf
CNC	Control Numeric Machine
CNN	Convolutional Neural-Network
CPU	Central Processing Unit
d.o.f	Degrees of freedom
DSD	Digital Servo Drives
DT	Digital Twin
FIR	Finite-Impulse-Response
FMS	Fleet Management System
FPGA	Field Programmable Gate Array
HiL	Hardware-in-the-Loop
HW	hardware
HRI	Human-Robot Interaction
IIR	Infinite-Impulse Response
ILC	Iterative Learning Control
KF	Kalman Filter
LPV	Linear Parameter-Varying
LQR	Linear Quadratic Regulators
LTI	Linear Time Invariant
MBSE	Model-Based System Engineering
MiL	Model-in-the-Loop
MIMO	Multi-Input Multi-Output
ML	Machine learning
MPC	Model Predictive Control
PD	Proportional-Derivative (controller)
PID	Proportional-Integral-Derivative (controller)
PiL	Processor-in-the-Loop
PML	Product Lifecycle Management
RC	Repetitive Control
RL	Reinforcement Learning
ROM	Reduced Order Model
RNN	Recursive Neural-Network
RTOS	Real-time Operating System
SEA	Series Elastic Actuators

SiL	Software-in-the-Loop
SISO	Single-Input Single-Output
SL	Supervised Learning
SoA	State-of-theArt
SRM	Switched Reluctance Motors
SW	software
TSN	Time Sensitive Network
WPx (e.g., WP4)	Work Package x (e.g., Work Package 4)
WT	Wind Turbine
XiL	X-in-the-Loop

Executive Summary

The deliverable 4.1 addresses the requirements for the control layer (layer 2) on the IMOCO4.E project. It is the basis of WP4, which is dedicated to the development of smart software modules implementing advanced motion control algorithms. These control algorithms will target both single d.o.f. systems and multi-axis systems.

The algorithms discussed in this deliverable will focus on problems and situations where the classical control is not enough due to a higher complexity on the system itself (e.g., compliant robots) or due to the interaction between the robot and its environment (e.g., real-time obstacle avoidance).

WP4 will also be the basis for systematic control design methodology using XIL simulation and Digital Twins as development tool which will be discussed in this deliverable.

The block diagram in Figure 1 represents an outline of the different topics covered in the D4.1.

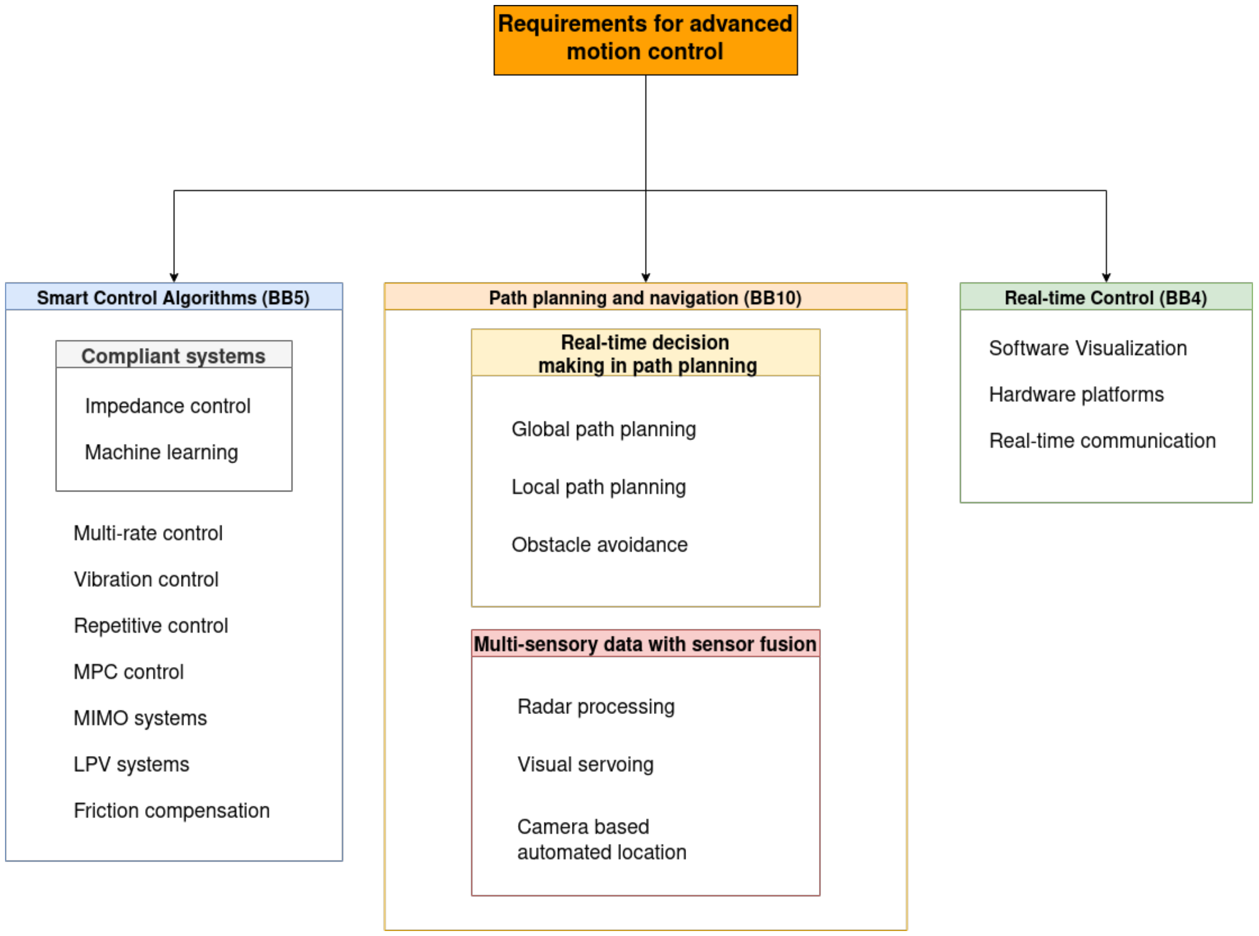


Figure 1: Outline of different technologies and approaches related to Smart control and covered in D4.1. They are structured in groups with respect to their relation specific BBs.

1 Introduction

1.1 Purpose of this Deliverable

The deliverable is dedicated to the description of the initial requirements for the control layer (mainly Layer 2), which are partially also based on inputs from WP2 (in particular, D2.1, D2.2 and D2.3), while working in parallel and considering D7.1 and D7.2. This report will summarize the control layer requirements specific for the relevant BBs (BB4, 5 and 10), pilots, demonstrators and use-cases.

1.2 Structure of this Deliverable

This IMOCO4.E deliverable (D4.1) contains a first iteration of the requirements of the IMOCO4.E framework control layer.

The deliverable provides a revision of different technologies and approaches for smart control that will be addressed in the project, including a state of the art, and the framework's architecture and relation to different BBs to give direction for IMOCO4.E tasks that will gather more detailed requirements about the implementation of functions and features.

Tasks 3.1, 4.1 and 5.1 focus on requirements for specific architecture layers of the IMOCO4.E platform, implementation requirements and methodology. Deliverable D4.1 will present approaches for smart control including a brief revision of the shortcomings from the state-of-the-art, needs for the future, and how this translates into specific requirements that outline the work to be done in WP4.

Section 2 of this deliverable will provide an overview of the IMOCO4.E framework and how it relates to Layer 2. Section 3 will review the xIL methodology and how Digital Twins can help in the development of motion control tools. Section 4 will cover the state of the art and planned future progress of several control algorithms and systems. Section 5 will study the relationship between path planning and real-time control/obstacle avoidance. Section 6 will review the different software and hardware components needed for real-time applications. Finally, section 7 will detail the relationships between BBs and the pilots/use cases/demos of the IMOCO4.E project and section 8 will close the document with a short conclusion.

1.3 Intended readership

This deliverable will be addressed to the partners involved in WP4, as well as any partner interested in the definition and development of Layer 2 technologies.

2 IMOCO4.E framework overview

In this chapter, the initial version of the IMOCO4.E reference framework is revisited after being defined in D2.3 to better frame the requirements related to smart control. This deliverable mainly considers the scientific and technological (ST) development objectives.

The IMOCO4.E reference framework is configurable from the lowest layer (Layer 1) to the human interfaces and supervisory layer (Layer 4). D4.1 is focused on smart control and thus mainly framed in Layer 2, although including also topics of Layers 1 (sensors and signals integration), 3 and 5 (such as models and Digital Twins): These topics being covered in D4.1 corresponding to other layers of IMOCO4.E architecture shall not be seen as unnecessary overlap, but rather as glue components that allow to integrate technologies and developments being done at different levels across IMOCO4.E architecture. An example is how in D4.1 we briefly explain how to extract and use models for smart control, while WP5 is more specifically dedicated to Digital Twins (DT), etc. This is part of how IMOCO4.E brings MBSE (Model-Based System Engineering) approach to all the architecture layers. Furthermore, these layers will demonstrate how AI supports the optimization of processes.

As stated in D2.3, the first version of the IMOCO4.E reference framework definition comprises the following viewpoints. Additionally, the BBs are abstracted as components.

- Architecture viewpoint
- AI viewpoint
- Digital twin viewpoint

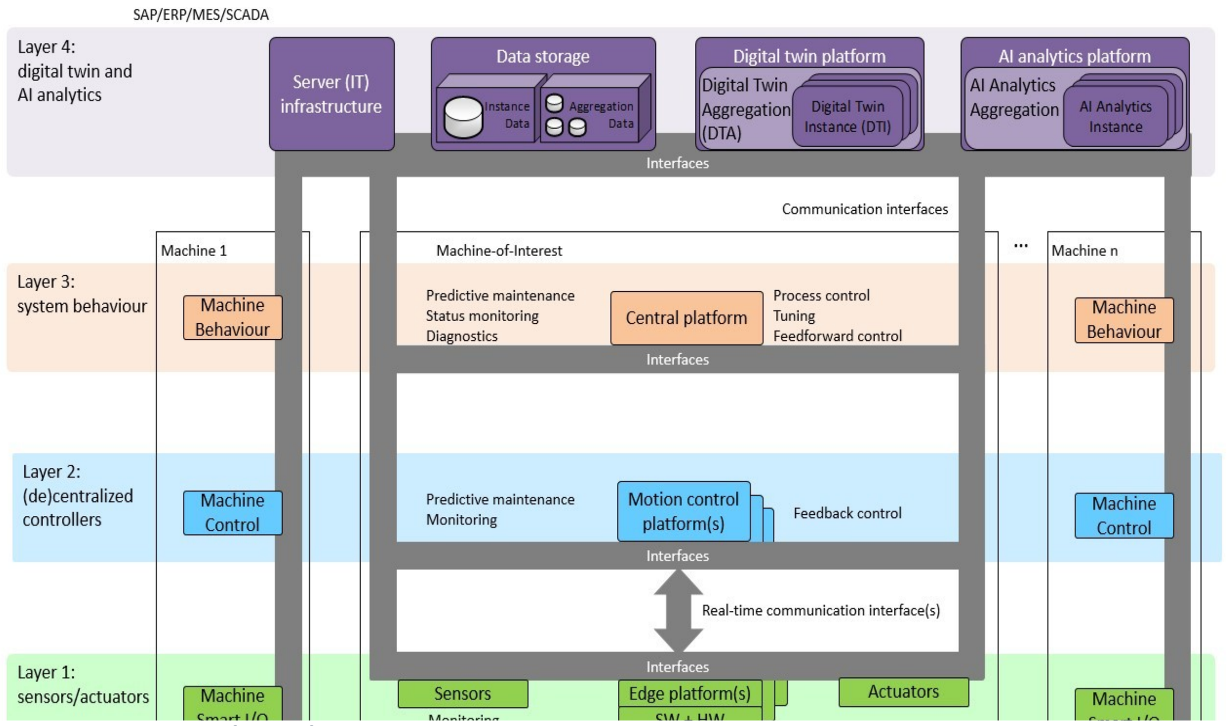


Figure 2: IMOCO4.E reference framework architecture viewpoint – initial version

The Architecture viewpoint is illustrated in Figure 2 (extracted from D2.3). Note that *Platforms* refer to the combination of software and hardware.

The AI viewpoint (illustrated in Fig. 3, extracted from D2.3) provides the data flow, interfaces and BB interactions required to achieve the project goal.

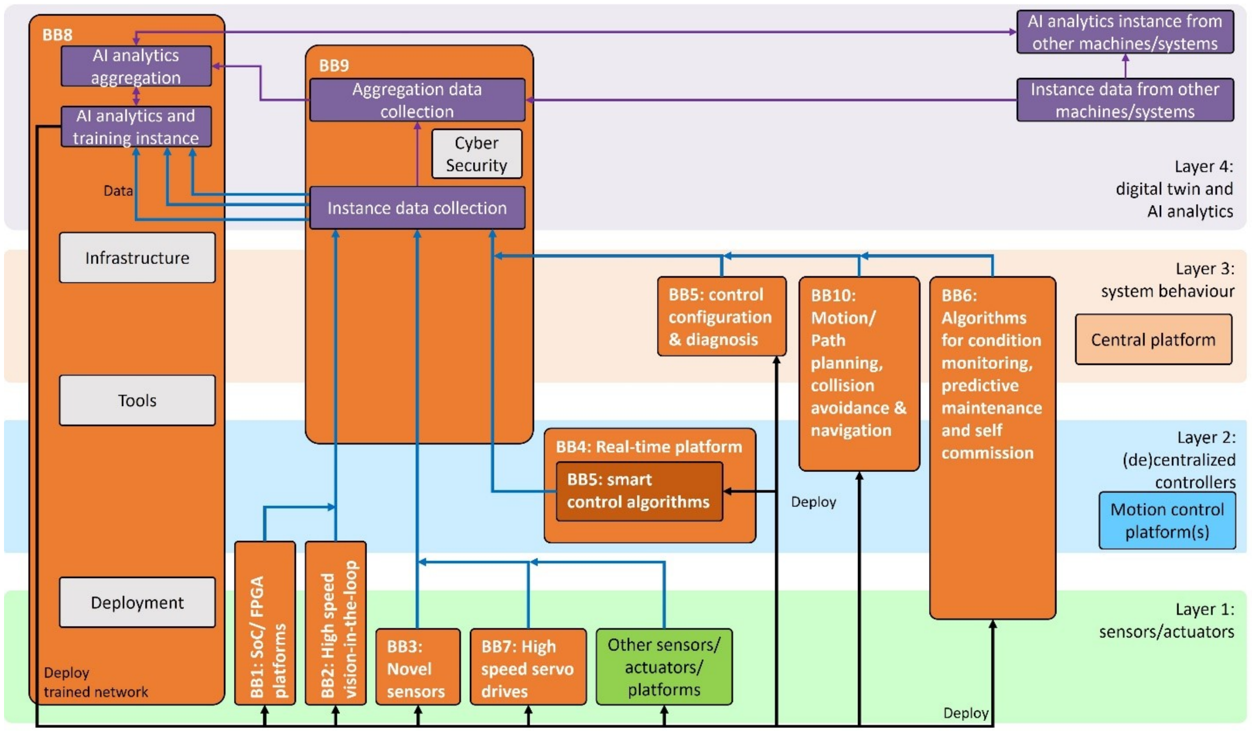

BB8
AI analytics aggregation
AI analytics and training instance
Data
Infrastructure
Tools
Deployment
Deploy trained network
BB9
Aggregation data collection
Cyber Security
Instance data collection
AI analytics instance from other machines/systems
Instance data from other machines/systems
Layer 4: digital twin and AI analytics
Layer 3: system behaviour
Central platform
BB5: control configuration & diagnosis
BB10: Motion/ Path planning, collision avoidance & navigation
BB6: Algorithms for condition monitoring, predictive maintenance and self commission
Layer 2: (de)centralized controllers
Motion control platform(s)
BB4: Real-time platform
BB5: smart control algorithms
Deploy
Layer 1: sensors/actuators
BB1: SoC/ FPGA platforms
BB2: High speed vision-in-the-loop
BB3: Novel sensors
BB7: High speed servo drives
Other sensors/ actuators/ platforms
Deploy

Figure 3: AI viewpoint with BB interactions (from D2.3)

The general principle is that data is collected from Layer 1 (sensors, edge platforms and actuators) and used by the AI framework for modelling, training, optimisation, analytics and/or services. Additionally, data can be collected from Layer 2. The data flows from layers Layer 1, Layer 2 and Layer 3 to the AI framework and back to the corresponding BBs are illustrated in Figure 3.

In this D4.1, there are different approaches that will create models (as part of the dataflow of the AI framework). Some of these models will be useful in upstream layers, but many will be also useful within Layer 2 (IA facilitating smart control).

As indicated in D2.3, the DT concept is rather diverse, being the core concept of a DT the coupling of physical entities with virtual models cross-enriching each other to benefit the entire system (Jones et al, 2020). Another widely accepted representation of a DT is the five-dimensional DT model illustrated in Figure 3 and derived from (Qi et al, 2021). In the IMOCO4.E framework, a DT comprises five dimensions – the physical entity, virtual model, data, service, and connection or interaction. The physical entity is the foundation of the DT. Virtual models model the physical entity and reproduce the physical geometries, properties, behaviours and rules (Qi et al, 2021). A virtual model is an abstraction of (a part of) the physical entity. Data is the core component and key driver of the DT. Data can be obtained from the physical entities, generated from the virtual models, obtained from services or provided by domain experts and users. A model is a part of the DT and can be independent of the DT. However, a DT has the models as one of the five dimensions and is one of the topics of D4.1.

This deliverable will also use I-MECH's D4.1 and D4.2 as a foundation of the more classical control algorithms and basic concepts of motion control. This foundation will allow us to focus on the more advanced control algorithms and the novel issues IMOCO4.E will tackle.

3 XIL and Digital Twins tools for the development of motion control systems (Task 4.2)

3.1 Introduction

Product representation has changed a lot over the last 30 years, from drafting to full digital mock-ups of the product assembly.

Performance verification has improved from a build-and-break approach to the current practice that includes significant simulation work as well as test. Therefore, product description evolved to full system mock-ups that cover not just mechanical but also electrical, software, and controls descriptions. Moreover, these must be fully integrated into an overall Product Lifecycle Management (PLM) system to ensure we can close the loop from requirements to as-designed behaviour and beyond to manufacturing and usage (Figure 4)

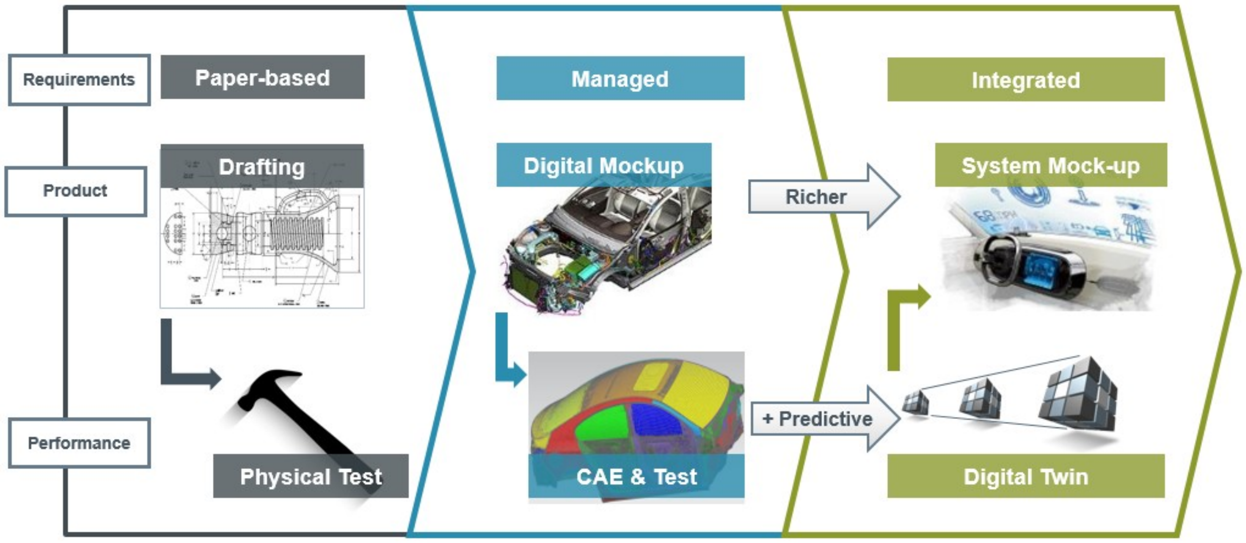


Figure 4: Evolution of product engineering

Many simulation packages have been developed in that philosophy to provide a large panel of coupled tools for engineers. The success of all these simulations packages is related to the possibility to be open and flexible, to interface with different 1D/3D software and to provide a consistent and continuous x in the Loop (where X can mean Software, Model, Processor, Hardware) capable framework. DT in manufacturing empowers intelligence to interconnected machines enabling them to orchestrate and communicate with simulation packages in xIL manner. One of the core technologies for the DT vision is Reduced Order Model since it allows compressing simulation models for real time simulation as well as model exchange by increasing the speed of model execution while maintaining required accuracies and predictability.

By involving xIL simulation and DT as important development tools IMOCO4.E project is proposing to provide a systematic control design methodology consisting of a joint modeling and (robust) model-based design approach.

3.2 State of the Art

The state of the art presented in this section is covering the two subtasks: Definition of xIL methodology for the development of motion control algorithms (subtask 4.2.1), Development of reduced models for control algorithm development and implementation (subtask 4.2.2) where the xIL and reduced model concepts are explained below.

3.2.1 *xIL* methodology

MiL/SiL/HiL/PiL are terms used in the context of control systems development which means that a control software is being built and it interacts with a mechatronic system. Typically, these are referred to as the "controller" and the "plant". When using a model-based workflow to create both the controller and the plant, then these terms represent how real the controller is.

- **Model in the Loop (MiL)**: a model of the control works with a model of the plant. The model of the control is in simulation and is connected directly to a model of the physical system within the same simulation. Extremely fast development occurs at this stage as you can make small changes to the control model and immediately test the system.
- **Software in the Loop (SiL)**: the control model is slightly more "real" in the sense that the model is no longer being executed but rather the model has been coded into C or C++ and then inserted back into the overall plant simulation. This is essentially a test of the coding system (whether auto-coded or human coded). Design iteration slows down slightly from MiL but coding failures start to become evident.
- **Hardware in the Loop (HiL)**: the control system is fully installed into the final control system and can only interact with the plant through the proper I/O of the controller. The plant is running on a real-time computer with I/O simulations to fool the controller into believing that it is installed on the real plant. In this case, the only difference between the final application and the HiL environment is the fidelity of the plant model and the test vectors that are being used. HiL is often used only for software validation rather than development as the design iteration is very slow at this point. However, this test is closest to the final application and therefore exposes most of the problems that will be seen.
- **Processor-in-the-Loop (PiL):** name given to a testing phase during which the software is executed typically on a prototypical hardware board equipped with the (planned) target processor, thus allowing the evaluation of the main possible scenarios. The objective is an early assessment of the concrete resource requirements of the software (timing domain, memory consumption, etc.). PiL is usually considered as a subset of the HiL phase.

The successive phases that have to be followed to perform real-time simulation of a controlled system known as Model-in-the-Loop, Software-in-the-Loop and Hardware-in-the-Loop, respectively, are presented in the figure below.

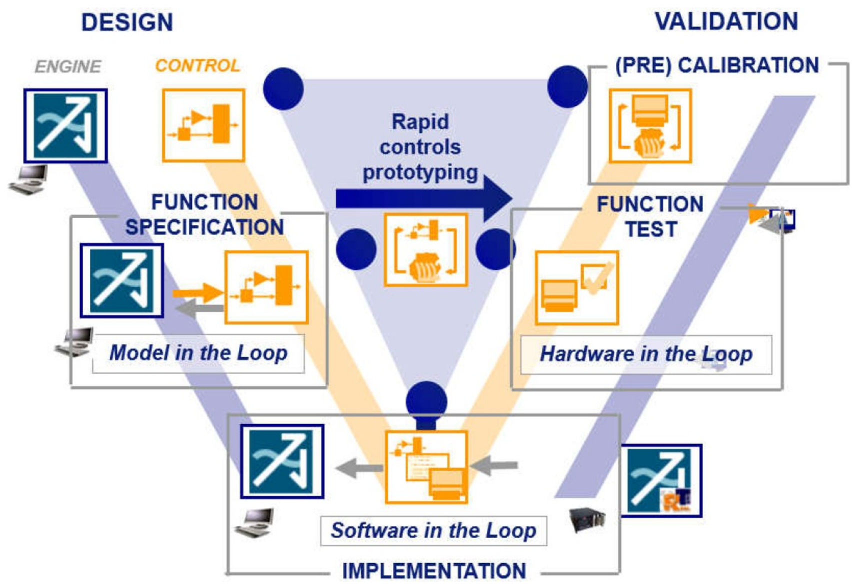


Figure 5: MiL, SiL and HiL phases in the V-diagram

3.2.2 Reduced Order Model

Reduced Order Model (ROM) is a simplification of a high-fidelity static or dynamical model that preserves essential behaviour and dominant effects, for the purpose of reducing solution time or storage capacity required for the more complex model.

ROM is probably the most important mathematical technique for realizing a DT during the life cycle of a product. Conform with (Dirk Hartmann et al, 2018), it enables new levels of interactivity, reliability, continuity, accessibility, and distributability of simulation models, where:

- Interactivity: Model order reduction directly reduces the number of degrees of freedom and therefore is able to generate arbitrary fast models. For real time applications a suitable compromise between accuracy and speed can be realized, e.g., by adaptive methods.

- Reliability: In most cases the speed-up of simulations tools lead to a lack of accuracy. However, this lack of accuracy should be quantified, because the user wants to know the expected accuracy of the reduced model.
- Continuity: Seamless workflow of simulation tools during the life cycle is still a major vision since many years. For the design, a lot of manual effort has to be spent in order to obtain accurate models. On system level simulation and for real time applications very fast models are required for exactly the same component, again requiring high manual efforts. The technique of model order reduction exactly bridges the different applications, i.e., model order reduction aims to automatically generate fast models from the comprehensive design models.
- Accessibility: Setting up complex models for design is still a task for experts. The generated fast models however have a standardized format and can be integrated into system simulation by non-experts. This point is a central aspect in the DT vision.
- Distributability: Solving the fast models generated by model order reduction requires only matrix-vector multiplications. Thus, they may run also on very small computers (and not only on specialized workstations).

Reduced order models can be created in 3 steps: collecting data from which model can be trained and validated, fitting reduced order models and exporting models for usage in various targets.

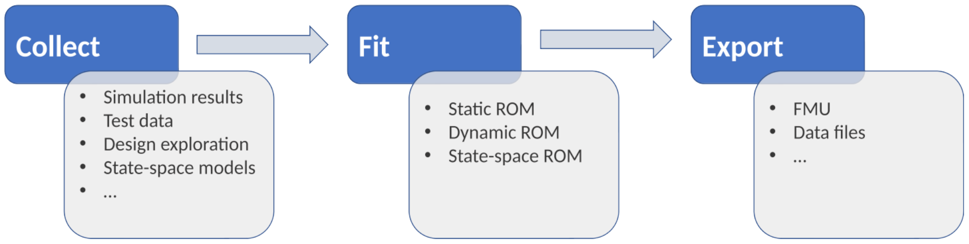
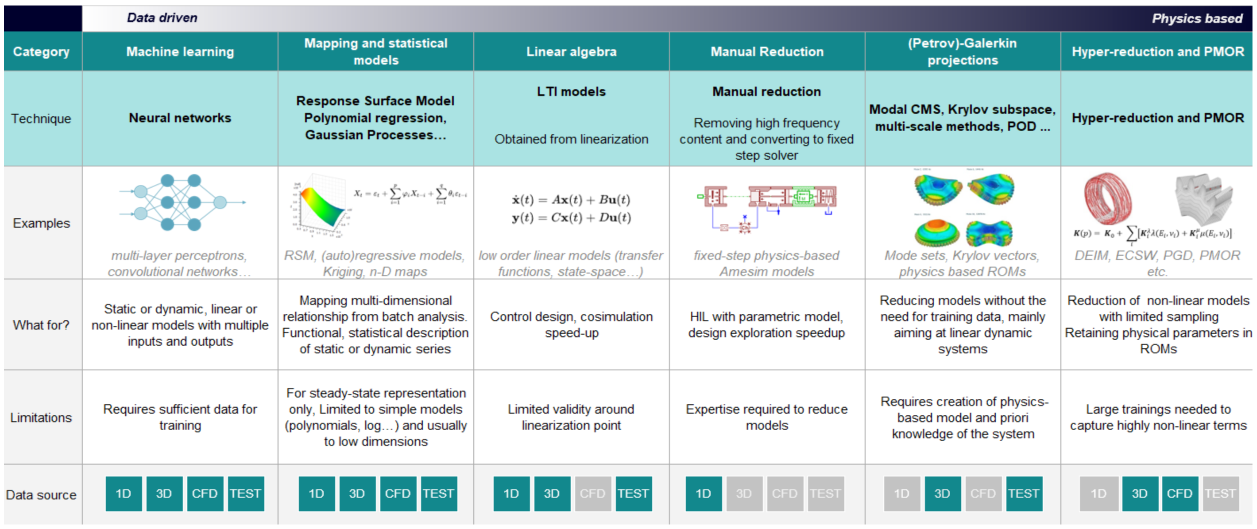


Figure 6: ROM workflow

To successfully use Reduced Order Model for simulation speed-up two techniques can be employed. Each technique has its own advantages and drawbacks:

- Physical-based order reduction: it requires a detailed understanding of the physics modelled, the tool, and the solver working principles to be used effectively. It mainly aims at reducing the number of state variables and the highest frequencies dynamics modelled to speed up the simulation.
- Artificial Neural Network (ANN): starting from a dataset of input/outputs of a given model (system), these algorithms can be trained to reproduce the correlations between them. They are very efficient from a computational point of view. On the negative side, the physics of the model they replace are not accessible. Their limitation adds up with those of the original model. Finally, their validity domain, determined by the input/output choice and the breadth of the dataset used for the algorithm training, must be respected with care.

The table in Figure 7 presents ROM techniques commonly applied to systems simulation, along with some guidelines and limitations.



	Data driven					*Physics based*
Category	**Machine learning**	**Mapping and statistical models**	**Linear algebra**	**Manual Reduction**	**(Petrov)-Galerkin projections**	**Hyper-reduction and PMOR**
Technique	**Neural networks**	**Response Surface Model Polynomial regression, Gaussian Processes…**	**LTI models** Obtained from linearization	**Manual reduction** Removing high frequency content and converting to fixed step solver	**Modal CMS, Krylov subspace, multi-scale methods, POD …**	**Hyper-reduction and PMOR**
Examples	*multi-layer perceptrons, convolutional networks…*	$X_t = \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$ *RSM, (auto)regressive models, Kriging, n-D maps*	$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t)$ $\mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t)$ *low order linear models (transfer functions, state-space…)*	*fixed-step physics-based Amesim models*	*Mode sets, Krylov vectors, physics based ROMs*	$K(p) = K_0 + \sum_i [K_i^{\lambda}\lambda(E_i, \nu_i) + K_i^{\mu}\mu(E_i, \nu_i)]$ *DEIM, ECSW, PGD, PMOR etc.*
What for?	Static or dynamic, linear or non-linear models with multiple inputs and outputs	Mapping multi-dimensional relationship from batch analysis. Functional, statistical description of static or dynamic series	Control design, cosimulation speed-up	HIL with parametric model, design exploration speedup	Reducing models without the need for training data, mainly aiming at linear dynamic systems	Reduction of non-linear models with limited sampling Retaining physical parameters in ROMs
Limitations	Requires sufficient data for training	For steady-state representation only, Limited to simple models (polynomials, log…) and usually to low dimensions	Limited validity around linearization point	Expertise required to reduce models	Requires creation of physics-based model and priori knowledge of the system	Large trainings needed to capture highly non-linear terms
Data source	1D ✓ 3D ✓ CFD ✓ TEST ✓	1D ✓ 3D ✓ CFD ✓ TEST ✓	1D ✓ 3D ✓ CFD TEST ✓	1D ✓ 3D CFD TEST	1D 3D ✓ CFD TEST ✓	1D 3D ✓ CFD ✓ TEST

Figure 7: Creation Reduced Order Model: different methodologies

HiL means Hardware in the Loop; LTI, Linear Time Invariant; POD, proper Orthogonal Decomposition; RSM, Response Surface Model; SVD, Singular Value Decomposition; CMS, Component Mode Synthesis; DEIM, Discrete Empirical Interpolation Method; ECSW, Energy Conserving Sampling and Weighting; PGD, Proper Generalized Decomposition; and PMOR, Parametric Model Order Reduction.

3.3 Progress beyond the SoA

The following results are expected as the project outcomes:

Definition of xIL methodology for the development of motion control algorithms

A methodology for control system development (e.g., with a help of Functional Mock-up Interfaces (FMI) or XIL standard) will be defined, to be used as a reference for tasks related to motion control algorithm development where the objectives are:

- To develop optimized algorithms in a shorter time;
- To directly implement model-based controllers.

Development of reduced models for control algorithm development and implementation

To improve dynamic performance of mechatronic systems complex models should be reduced by separating the relevant features that affect the controller. In this purpose the reduced order modeling will links a large variety of methods from different domains (equation-based, statistics, linear algebra, and machine learning) offering a selection of tailored methods for various data types to efficiently build up models without domain knowledge.

3.4 Requirements

ID	**Requirement**	**Comments**
R1-D4.1	XiL methodology and DTs must guarantee system safety by means of progressive testing of software and hardware components	XiL methodology allows the testing of different components on each stage, ensuring that the software is safe before it can be tested on real hardware
R2-D4.1	All DTs must provide a clear estimation of the sim-to-real gap between the DT and its physical counterpart	
R3-D4.1	DTs should be compatible with standard tools.	

4 Smart control algorithms library (BB5/Task4.3/Task4.4)

4.1 Introduction

There are many smart control algorithms that fit many different problems and situations. In some cases, a higher precision demands for algorithms that focus on a specific part of a problem, like friction or vibration damping. In other cases, the non-linearities of a system may force us to use non-conventional kinds of control like machine learning.

In D4.1 of I-MECH (I-MECH, 2022) basic concepts of control and several generic techniques where defined. This deliberable will use this foundation to explore new and more advanced control functionalities which tackle most of the unresolved problems in the field.

In this section we will review several smart control algorithms and the systems for which they can be used in order to obtain better results than traditional control can offer us

4.2 Control functionalities

4.2.1 Vibration control for mechatronic systems

4.2.1.1 Introduction

Modern motion control systems rely on properly adjusted velocity and position feedback loops. Their accurate tuning is essential when dealing with stringent performance requirements. One of the limiting factors which affect the achievable quality of control is the mechanical compliance of the driven load. Proper adjustment of the feedback control or smart feedforward utilization is necessary to cope with unwanted transient or residual oscillations.

Input shaping is a well-known technique to pre-compensate oscillatory modes of flexible systems. It offers some inherent advantages to conventional IIR notch filters used in industrial drives, such as finite-time monotonous response or inherent robustness to modelling errors that are part of the shaper design process. New possibilities for shaping filter design emerged with the evolution of numerical optimization methods. Novel design problem formulations are possible, allowing the embedding of various crucial design constraints in the shaping filter synthesis process. The development of optimization-based design algorithms for a class of Zero-vibration shaping filters is one of the planned outcomes of the project.

Active and passive vibration damping using input shaping and auxiliary sensors

A key issue of elastic servo systems is the absence of direct information about the behavior of the driven load. Only motor-side feedback is often available, making the stabilization and precise positioning of an oscillatory load difficult. A load attached accelerometer may be a suitable choice for a wide range of applications due to the low price, small dimensions and simple mounting. In other applications, a position encoder may be installed at the load side to measure the state of the driven working mechanism directly. This activity aims to develop novel control structures allowing integration of auxiliary sensory information from the point of interest to form fully closed-loop solutions. This can bring significant performance advantages compared to conventional collocated control schemes with the feedback closed solely from the actuator side variables.

Vibration damping in time-varying elastic systems

Mechanical systems often present significant variations in the dynamical model depending on the operating points. Thus, vibration modes could change during the execution of the task. On the one hand, the designer of the control strategy could cope with this parameter variation by considering the worst-case scenario, which leads to a robust but underperforming solution. On the other hand, the controller can embed variations of the model parameters resulting in higher performance and higher design cost.

4.2.1.2 State of the art

Unwanted mechanical oscillations encountered in motion control applications can lead to a deterioration of quality of control and may cause an increased wear of the mechanical components of the system. They should be avoided by a proper mechanical design of the machine or by appropriate adjustment of the control system during the commissioning phase. Three main approaches are used in practice to deal with vibrations.

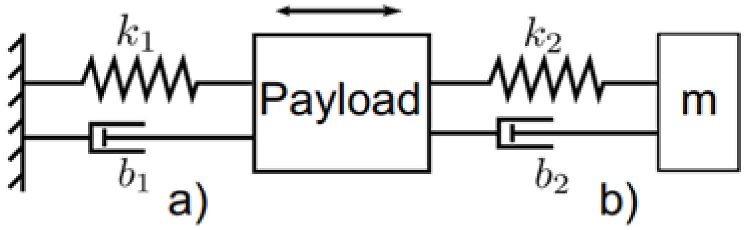


Figure 8: Mechanical damping using a) vibration isolator b) vibration absorber

Mechanical damping can be realized by suitable adjustment of machine construction. The system's resonance frequencies may be shifted to higher values outside of the expected excitation bandwidth by increasing the stiffness of the critical parts of the mechanics. However, an increase in the overall mass leads to higher construction costs, lower energy efficiency, slower dynamics of motion due to high inertias etc. Reinforcement of the machine construction can be problematic in some applications due to the inevitable use of flexible components such as ropes, gears with elastic parts or long manipulator arms. Additional mechanical elements improving the overall level of damping may be used. Passive isolation systems or vibration isolators contain mass, spring and damping elements inserted between a moving payload and a rigid frame or ground. The mass and spring cause energy dissipation and provide additional damping. They essentially behave as mechanical low-pass filters. Another approach uses vibration absorbers which consist of a secondary inertia-spring system coupled to the payload. They generally act as mechanical notch-filters. Proper tuning of the absorber parameters according to the resonance frequencies of the machine is needed for proper operation. Another issue is the proper placement of the machine with respect to stiffness of the foundation and sub-soil.

Vibration damping may also be provided by employing the motion control system. Passive vibration control (sometimes called gain stabilization) reduces the amplitude of the excitation signal (actuator force/torque) in the frequency range of the system resonances to prevent from excitation of vibrations. This is performed by implementing a low-pass or notch filter in the motion control loop. The low-pass filters are easy to use as only the cut-off frequency has to be set. However, the phase-lag introduced by the filter limits the achievable bandwidth of the closed loop. The filter may operate in the open-loop configuration to shape the setpoint command obtained from the trajectory generator or connected in series with the feedback compensator. Most of the commercial servo-drive units offer this capability. Usually, a two-pole two-zero IIR notch or biquad filter is implemented in the feedback loop. The filter zeros are commonly set to cancel the weakly damped poles of the system and the feedback compensator is designed for the resulting damped system to obtain a stable closed loop. This technique proved to be effective for high-frequency modes above the target closed-loop bandwidth. However, it is less appropriate when the resonance modes overlap with the desired bandwidth (Ellis & Gao, 2001).

An alternative way of the notch filter design is a so-called input shaping method or zero-vibration shaping. The goal is to design a Finite Impulse Response (FIR) shaping filter that modifies an input signal of an underdamped oscillatory system so that the level of excited vibrations is minimal. The idea of shaping filter consisting of time delay elements was presented in (Smith, 1958), followed by several contributions extending this concept (Singer & Seering 1990; Singhose, 2009).

A general n-pulse input shaping filter can be described in the continuous-time domain in the form of impulse function

$$h_s(t)=\sum_{i=1}^{n} A_i\cdot\delta(t-t_i),\ \ 0\leq t_1<t_{t+1},\ \ A_i\neq 0 \qquad \text{(Eq. 4.1)}$$

where A_i denotes amplitudes of the i-th pulse and d is Dirac function with time-shift t_i.

The response of the shaper in the time domain can be determined by performing the convolution operation

$$v(t)\int_{-\infty}^{\infty} h_s(\tau)u(t-\tau)\,d\tau=\int_{-\infty}^{\infty}\left(\sum_{i=1}^{n} A_i\delta(\tau-t_i)\right)u(t-\tau)\,d\tau=\sum_{i=1}^{n} A_i u(t-t_i) \qquad \text{(Eq. 4.2)}$$

$$v(t)=\int_{-\infty}^{\infty} h_s(\tau)\cdot u(t-\tau)\,d\tau=\sum_{i=1}^{n} A_i\cdot u(t-t_i) \qquad \text{(Eq. 4.3)}$$

It can be seen that the filter has the form of a sum of weighted time delayed values of the input. The goal of the filter design is to choose the values of amplitudes and time delays in such a way that after the last pulse has been led to the system, the amplitude of the excited residual vibrations is equal to zero (Fig. 9).

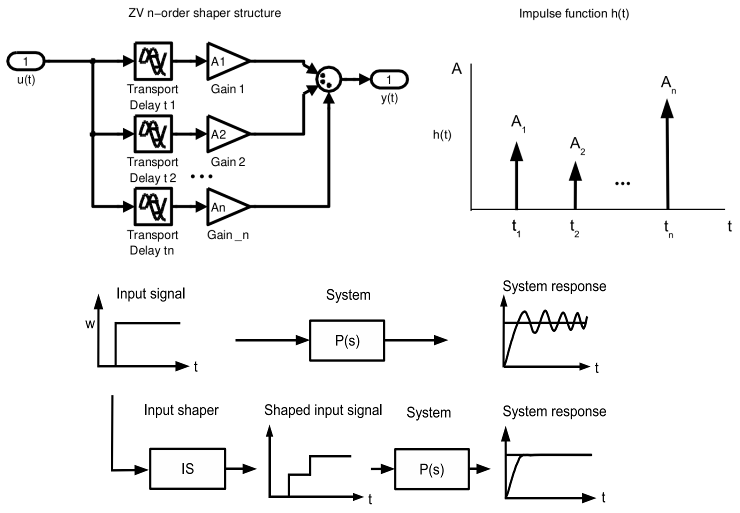


Figure 9: Principle of Zero-vibration input shaping methods

Various design methods to accomplish this objective have been proposed in the literature. A common design strategy of the input shaper is via the so-called residual vibration function

$$V(\omega,\xi)=e^{-\xi\omega t_n}\sqrt{C(\omega,\xi)^2+S(\omega,\xi)^2} \qquad \text{(Eq. 4.4)}$$

Where

$$C(\omega,\xi)=\sum_{i=1}^{n} A_i e^{\xi\omega t_i}\cos\left(\omega\sqrt{1-\xi^2 t_i}\right)\triangleq c(\omega,\xi)h \qquad \text{(Eq. 4.5)}$$

$$S(\omega,\xi)=\sum_{i=1}^{n} A_i e^{\xi\omega t_i}\sin\left(\omega\sqrt{1-\xi^2 t_i}\right)\triangleq s(\omega,\xi)h \qquad \text{(Eq. 4.6)}$$

determining the level of excited residual vibrations at the settling time of the shaper $t=t_n$. This led to the derivation of well-known shaper types called ZV (Zero Vibration), ZVD (Zero Vibration Derivative), EI

(Extra insensitive) and their variations, now being considered as state of the art algorithms in input shaping.

The complexity of the design problem increases considerably when more oscillatory modes of the controlled plant have to be considered. For such cases, several methods based on numerical optimization were proposed (Van den Broeck et al, 2008; Cole, 2012; Goubej et al, 2020). They can deal with the complexity of the problem and allow the definition of additional design constraints.

The design of the input shaper is usually defined as a constrained optimization problem

$$\min_{h} \; f(h) \text{ subject to } \left\{ \begin{array}{r} c_{eq} h = d_{eq} \\ c\, d \leq d \end{array} \right. \qquad \text{(Eq. 4.7)}$$

where f(h) is a properly-chosen objective function. Some choices of the objective function lead to convex optimization problems that can be readily solved with existing numerical methods.

The zero vibration filters have been used in various motion control applications such as overhead cranes, flexible robotic manipulators, CNC machine tools, voice-coil motors or spacecrafts. The main advantage compared to conventional notch filters is the finite impulse response which is advantageous for the setpoint command shaping (the overall duration of the motion is known in advance), the possibility of infinite damping of a certain frequency in case of discrete-time implementation and lower phase-lag introduced in the signal path. (Singh & Vyhlídal, 2020) reviews the latest research results in this field.

Active vibration control methods try to alter the oscillatory dynamics of a flexible mechanical system employing a proper feedback controller. Unlike the gain modulation of passive methods, the feedback provides *phase stabilization* to the oscillatory dynamics. Compared to the passive methods, the advantage is the ability to actively suppress the vibrations caused by external disturbances (load torque/force). The disadvantage is the risk of instability due to the introduction of the feedback. A typical problem related to flexible systems is so called spill-over effect. The unmodeled high-frequency dynamics (typically higher resonance modes) may be excited by the feedback controller causing the instability of the closed loop. Therefore, the goal is to suppress a certain number of resonances and achieve a sufficient high-frequency roll-off of the compensator in order to prevent from excitation of unmodeled dynamics.

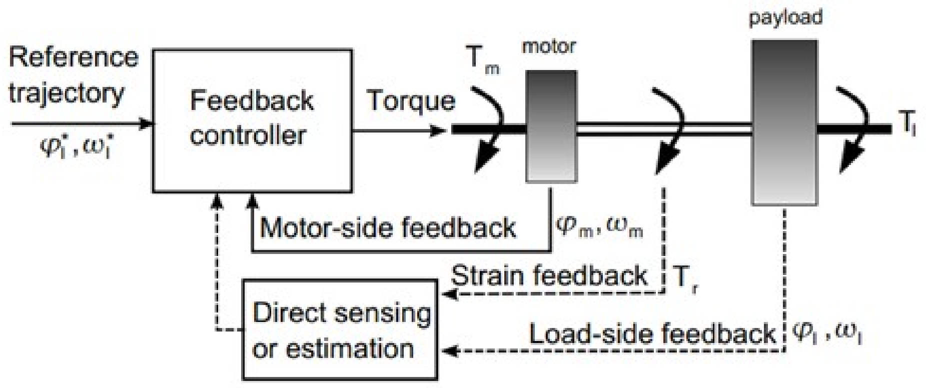


Figure 10: Principle of active vibration control

Numerous approaches to active vibration control have been proposed in the literature for the motion control of flexible mechanical systems ranging from PID control (Zhang & Furusho, 2000; Goubej, 2014]), state feedback (Ji & Sul, 1995), model-predictive control (Thomsen et al, 2011), over disturbance observers and resonance ratio control (Katsura & Ohnishi, 2007), H-infinity optimization (Lee et al, 2006), sliding mode control to soft-computing methods employing neural networks, fuzzy logic or expert systems (Orlowska-Kowalska & Szabat, 2007; Kaminski, 2010). Nevertheless, most of the industrial motion systems still use conventional decentralized cascade PID control scheme Fig. (10).

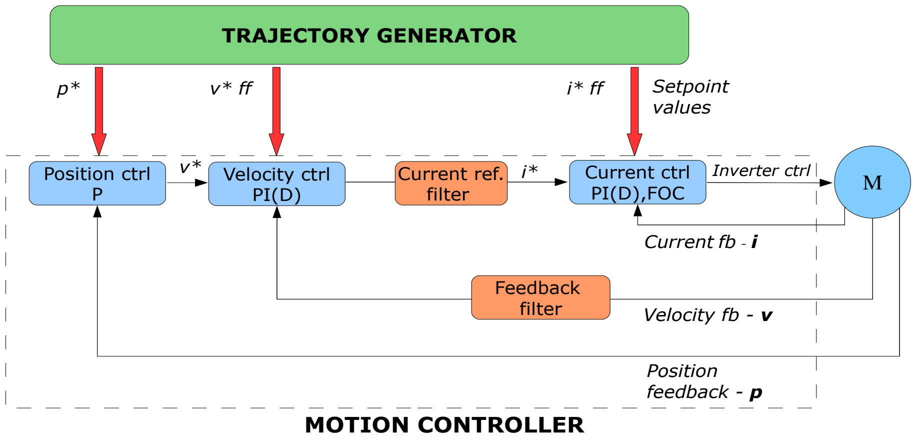


Figure 11: Structure of industrial motion control system

Its massive employment includes simple implementation with a reasonable number of parameters with a clear physical meaning that can be tuned by trial and error, the possibility of sequential tuning of the individual loops and long history and experience with PID controllers that practitioners understand. Although structurally simple, finding an optimal setting of the PID gains may be challenging when encountering a mechanically compliant system and pushing its performance to its limits (Goubej, 2016) There is still no generic method applicable to feedback control in motion systems that are widely accepted by the industry so far.

In many mechanical systems, the load mass can change during the operation; considering an elevator as an example, the mass of the suspended rope changes depending on the position of the cabin and the counterweight (Santo et al, 2016; Roberts, 1998). Load variations can significantly change the resonance/antiresonance frequencies of the system, requiring the adaptation of the control strategies to avoid vibrations (Mangare et al, 2022). This system can be described as a linear parameter-varying (LPV) system (Amato, 2006; Copot et al, 2018).

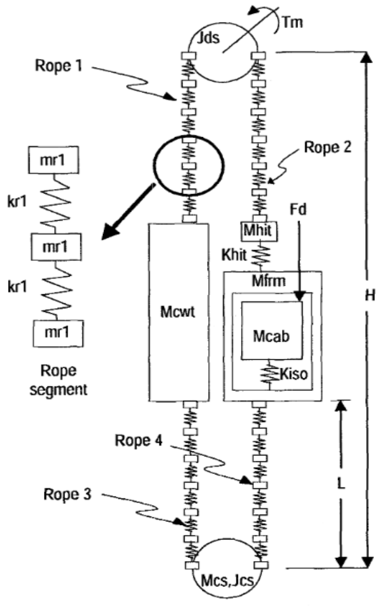


Figure 12: Dynamic model of a lift [Roberts 1998]

4.2.1.3 Progress beyond the SoA

The following results are expected as the project outcomes

1. Development of novel generic design methods for the optimal synthesis of robust multi-mode input shaping filters. A key idea is to embed prior knowledge about the uncertainty of the controlled system in the design process to derive robust shaping filters insensitive to modelling errors. Another key aspect to be considered is the *output* behavior of the controlled plant, unlike the relative excitation of the individual oscillatory modes commonly considered in existing design procedures. The importance of contribution of the individual bending modes to the dynamics of the chosen output is revealed via the modal transformation of the system, providing structural information applicable in shaper design procedure. New formulations leading to convex optimization problems solvable with state-of-the-art numerical methods are planned to be derived.

2. Development of algorithms for feedforward control of elastic systems with time or varying position dynamics. The problem of variability in controlled plant dynamics requires different feedforward and feedback control design approaches. Input shaping methods intended for stationary systems cannot be used directly due to the invalidity of the principle of superposition. Position varying dynamics is typical for mechatronic systems working in a wide range of operating points with different oscillatory behavior due to changes in the distributed elasticity of the driven load. An example is an elevator system or a gantry crane manipulator with varying rope lengths resulting in changes in resonance frequencies corresponding to the system eigenmodes. For the feedforward control, development of shaping filters in the domain of linear parameter-varying systems is envisaged.

3. Development of fully-closed loop control structures involving auxiliary sensors attached at the manipulated point of interest. Motion control performance of elastic drive systems can be significantly enhanced by introducing auxiliary feedback variables providing additional information about the load-side behavior at the point of interest. Auxiliary feedback can be provided, for example by a secondary position encoder or a load-attached accelerometer. Especially the accelerometric sensor may be a suitable choice for a wide range of applications due to the low price, small dimensions and simple mounting. Successful applications of the acceleration feedback were reported for robotic manipulators, machine tools feed drives or linear positioning stages. However, proper adjustment in the control algorithms is needed to embed the additional sensory information properly. Our goal is to focus on design methods for control loops involving auxiliary position or acceleration measurements to enhance performance achievable with conventional control structures. The possibility of enhancing model fidelity during data-driven system identification will also be explored.

4.2.1.4 Requirements

ID	Requirement	Comments
R4-D4.1	Gain or phase stabilization of dominant resonance modes must overlap with target closed-loop bandwidth	This can be achieved either by adequately adjusting the parameters of existing control structures or by employing more complex control laws involving the specific use of feedforward or feedback control actions, possibly with auxiliary sensors. A fundamental requirement is preserving the internal stability of the closed-loop and preventing excitation of higher bending modes and unmodelled dynamics
R5-D4.1	Systematic design procedures must allow automatic or semi-automatic synthesis and parameterization of the control structures without requiring a highly skilled operator	The goal is to minimize commissioning time and allow routine employment of the methods without the necessity for a deep background in advanced control engineering
R6-D4.1	Sufficient performance improvement needs to be had compared to conventional control schemes	
R7-D4.1	Load-side motion control requires the usage of additional sensors	i.e., accelerometers or encoders
R8-D4.1	Computation burdens should be compatible with available computation power on the drives	

Table 2: Vibration control requirements

4.2.2 Repetitive control

4.2.2.1 Introduction

From its first presentation for control of proton synchrotron magnet power supply (Inoue et al, 1980), Repetitive Control (RC) has seen wide success both in research and industrial environments because of the repetitive nature of the typical industrial task. RC configurations have been implemented in different fields, such as rejection of power supply disturbances (Nakano & Hara, 1986), control of disk-drive systems (Chew & Tomizuka, 1990) and optical disk-drive (Moon et al, 1998), vibration suppression (Hattori et al, 2001) and robotic manipulators performing repetitive tasks (Omata et al, 1987; Biagiotti et al, 2015).

However, many issues remain open and require further research and technological efforts to enlarge the field of application and the effectiveness of the methods. In particular, non-linear and varying dynamics, robustness, position-dependency and computation burdens limit RC application in many fields.

IMOCO4.E aims to overcome these limitations considering the technologies described in the following subsections.

Torque ripple in Switched Reluctance Motors (SRM)

Switched Reluctance Motors are attractive because of their high efficiency, mechanical simplicity and low cost (Miller, 1931). On the other hand, the non-linear relationship between torque, current and rotor position leads to a complicated control scheme. Typically, this involves the construction of a commutation function, describing currents per coil, required for some requested torque which is periodic in position, in order to linearize the system dynamics. When this commutation function is constructed imperfectly, either because the non-linear dynamics are unknown or because of sampling effects, torque ripple occurs, leading to a position error that is periodic in the spatial domain. While switched reluctance motors are a motivating example, theoretical advances in this field may carry over to other types of actuators that rely on commutation.

Implementation issues in repetitive control schemes – robust, inferential and delay-varying methods

Repetitive control methods aim at designing feedback loops capable of compensating generic periodic disturbances or tracking periodic reference signals with high accuracy. Numerous design methods were developed, followed by the successful application of this type of control in various application domains. Repetitive control technology has already been adopted by industry to some extent. This functionality became an integral part of many commercial off-the-shelf products for motion control, bringing potentially improved control performance in mechatronic applications. Nevertheless, there are still several issues to be addressed both at the level of generic design methods and algorithms, and in terms of implementation problems encountered when employing repetitive controllers in real-time motion systems. Several such issues have been identified to be solved in the IMOCO4E project. These are discussed in the state of the art and progress beyond SoA sections, outlining envisioned contributions in this field. The goal is to bridge the gap between the latest research and industrial applications of high-fidelity motion systems.

Position-dependent repetitive disturbance estimation and compensation

Several mechanical applications are affected by disturbances related to positional variables like torque ripples, pulleys, bearings, and rail guides (Tang et al, 2020; Oomen, 2018). Initial Repetitive control strategies focused on time-dependent disturbances providing an effective performance when the system works at constant speeds. However, a position-based repetitive control algorithm is necessary when the constant-speed assumption does not hold. Recent works focused on this problem showing the capability to compensate for the disturbance at the price of increased computational and memory requirements. This aspect makes the implementation of limited-power edge controllers, such as the industrial drives, challenging.

It is worth considering a class of applications where the disturbance, the target variable, and the sensor are not located in the same physical place. For example, a lift has a disturbance source due to the rail guides of the cabin and the counterweight, the sensor is typically an encoder mounted on the motor shaft, and the target variable is the position of the cabin. The goal is to move the cabin avoiding discomfort due to oscillations. A velocity controller achieves this by measuring the motor speed, coupled with on/off position sensors on the cabin's rail guide. In this kind of application, nullifying the motor-velocity error could lead to more oscillations and discomfort of the target variable. At the same time, the usage of disturbance estimation could mitigate these aspects by changing the system's behavior, for example, by reducing speed, changing motor law or suppressing frequency bands.

4.2.2.2 State of the art

The fundamental idea of repetitive control methods comes from the well-known Internal model principle (Francis & Wonham, 1976), stating that the model of an exogenous disturbance has to be inserted into the loop to asymptotically nullify tracking error. Considering the disturbance as a time-periodic function with period *T*, the model can be obtained from the Fourier series expansion

$$d\,t = \sum_{k=-\infty}^{\infty} c_k \cdot e^{j\omega k t} \qquad \text{(Eq. 4.8)}$$

where is the natural frequency.

Following (Yamamoto, 1993), a model that includes such dynamics is

$$M(s) = \frac{1}{s} \cdot \prod_{k=1}^{\infty} \frac{1}{s^2 + k^2 \omega^2} \qquad \text{(Eq. 4.9)}$$

This model can be written by recalling the identity

$$\sinh(x) = x \cdot \prod_{n=1}^{\infty} \left(1 + \frac{x^2}{n^2 \pi^2}\right) \qquad \text{(Eq. 4.10)}$$

For $x = \frac{T}{2} \cdot s$, the model can be reduced to

$$M(s) = \frac{e^{-sT}}{1 - e^{-sT}} \qquad \text{(Eq. 4.11)}$$

A pure time delay with positive feedback can serve as a minimal model for an arbitrary periodic function. Reference (Inoue et al, 1980) is considered to be one of the first documented practical applications of repetitive control. General analysis of closed-loop stability and performance of repetitive controllers followed later in (Hara et al, 1985; Hara et al, 1988; Tomizuka et al, 1989). Numerous design methods emerged in the last decades for linear, non-linear, continuous or sampled-data systems (Nagahara & Yamamoto 2010; Chen & Tomizuka 2014; Zhou & Jinhua 2018). Several successful applications were reported in various application domains ranging from optical storage systems and disk drives (Steinbuch et al, 2007), motion controls and robotics (Liuzzo & Tomei, 2008) to hydraulic manipulators (Luo et al, 2017) or power electronics (Weiss et al, 2004). A survey (Wang et al, 2009) maps important results and provides connections to the closely related topics of Iterative Learning Control (ILC) and run-to-run control.

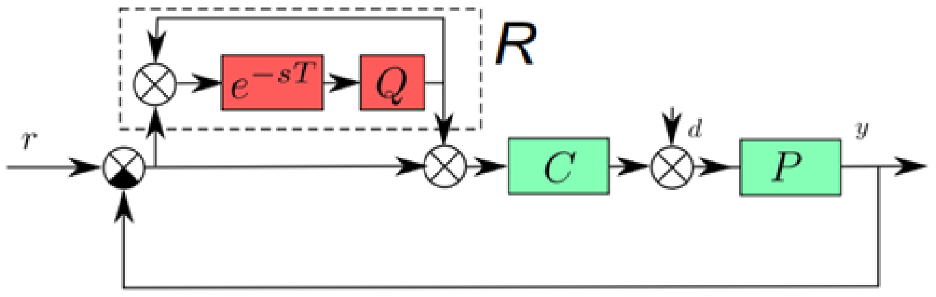


Figure 13: Conventional structure of the repetitive controller R – plug-in type parallel control scheme complementing the standard feedback loop

Among several versions of the repetitive control (RC) algorithm, a basic plug-in type parallel structure from Figure 13 is commonly applied in industrial motion systems. This scheme is suitable for practical implementation as it extends a conventional feedback loop with the repetitive control block, allowing it to be conveniently enabled or disabled when needed without disturbing the stability of the main loop. The repetitive controller in the parallel form introduces a transfer function into the loop in the form of

$$1 + R(s) = \frac{1}{1 - e^{-sT} Q(s)} \qquad \text{(Eq. 4.12)}$$

where R is the repetitive control block from Fig. 13 and Q is a suitable shaping filter which is commonly designed to ensure the stability of the closed loop. For the particular choice Q(s) = 1, the repetitive part generates an infinite number of poles pk on the imaginary axis at positions

$$p_k = j k \omega, \forall k \in (-\infty, \infty) \qquad \text{(Eq. 4.13)}$$

allowing asymptotic compensation arbitrary T-periodic exogenous signals r; d, provided that the closed-loop is internally stable. A sufficient condition of stability for an arbitrary period T can be derived from the Small-gain theorem as

$$\|Q(s) \cdot S(s)\|_\infty < 1, \quad S(s) = \frac{1}{1 + C P} \qquad \text{(Eq. 4.14)}$$

where *S* denotes the closed-loop sensitivity function. This leads to a viable choice of the Q filter that is commonly designed to attain low-pass amplitude-frequency characteristics. Further performance improvement can be achieved by employing a more complex structure shown in Figure 14. A second learning filter L is added to provide lead compensation of the repetitive controller, allowing extension of the effective applicable bandwidth of the disturbance rejection.

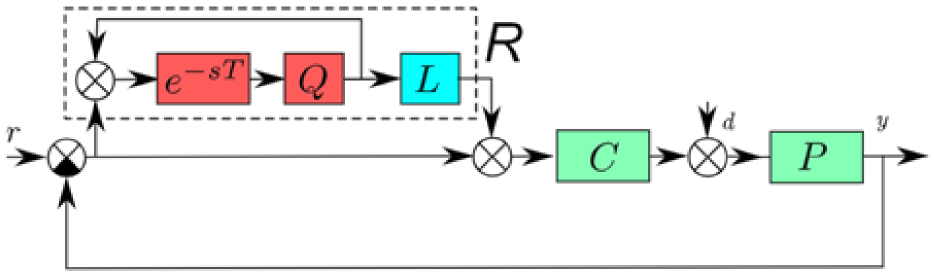


Figure 14: Modified structure of the repetitive controller R – introduction of learning filter L providing phase compensation to improve disturbance rejection

Torque ripple in switched reluctance motors

Existing commutation strategies that aim to linearize switched reluctance motor dynamics typically rely on inversion of the non-linear dynamics in a coil-per-coil fashion (Wang 2016; Vujicic, 2012), where an ad-hoc approach is taken to divide currents over different coils. Moreover, when the non-linear dynamics are not known, data-driven approaches use observer-based techniques to estimate the torque and apply ILC to achieve accurate tracking performance (Sahoo et al, 2007).

Implementation issues in repetitive control schemes – robust, inferential and delay-varying methods

Robustness of repetitive controllers in terms of sensitivity to modelling errors in both assumed plant dynamics and knowledge of fundamental disturbance period is a challenging problem that has to be considered. The introduction of the time delay in the exogenous signal generator affects the stability and performance of the whole motion control loop. Mismatch in the assumed disturbance period often causes severe degradation of disturbance rejection capability. It may even lead to motion tracking performance deterioration when compared to the conventional feedback controller without the added RC part. Sensitivity to disturbance period variations was studied in (Steinbuch, 2002), followed by later results in (Steinbuch et al, 2007; Pipeleers et al, 2008; Merry et al, 2011; Eielsen et al, 2015). The main idea is to modify the disturbance generator to introduce wider notch regions in the closed-loop sensitivity function, thereby achieving higher tolerance to period mismatch or adapt the length of the time-delay block continuously based on the actual state of the exogenous disturbance. Plant modelling errors were also addressed for example in (Goubej & Schlegel, 2019), dealing with a robust design method for fixed structure controllers. Position-domain repetitive control was proposed in (Mooren et al, 2020), allowing significant performance improvement in the case of position-dependent periodic disturbances.

Inferential control denotes a general class of motion control problems where a difference between feedback and controlled variables exists (Oomen et al, 2014). A typical example is a problem of collocated velocity or position control of a mechanically compliant motion system, aiming at achieving stringent tracking performance and vibration attenuation at the load side of the driven working mechanism, whereas only motor side feedback is available to close the feedback loop. Care must be taken when designing such motion systems since good actuator side performance does not generally guarantee acceptable behavior of the load side variables. The conventional solution is to attenuate unwanted transient or residual oscillations using notch or bi-quad filters inserted in the loop (Ellis & Gao, 2001) or by tuning the feedback controller to achieve active vibration rejection functionality (Goubej & Schlegel, 2014). Such solutions may not be sufficient in the scope of repetitive control. Commonly used structures need to be adopted in the inferential framework that inherently brings multivariable control challenges (Bolder et al, 2014).

Multi-rate implementation of motion controllers is commonly employed by utilizing several loops with different sampling rates. The discrepancy in the update rates typically comes from the inherent HW and SW limitations of the control system. The interface between adjacent loops is commonly realized by employing zero-order or first-order-hold blocks implementing necessary signal interpolation. This may be suboptimal in terms of distributed control structures with repetitive controllers working with different update rates than main motion loops.

Position-dependent repetitive disturbance estimation and compensation

Position-based disturbances and their compensation are an essential and actual research field (Oomen, 2018; Tang et al, 2016; Liu et al, 2017; Yuan et al, 2012). The most frequent approach is the angle-based approach (Chien and Ma, 2013; Tang et al, 2016), where the buffer obtained by the discretization of the time delay is fed by using the current position instead of the current time, as in Figure 15.

The angle-based method is practically implemented by converting the position to a motor angle between 0 and 2·π, and using a look-up table with the angle as the entry and the compensation as the output. The main challenges in this method are: the management of the non-monotonic variation of the position, as opposed to the monotonic increment of the time; the high required memory and the computational effort of the update of the look-up table.

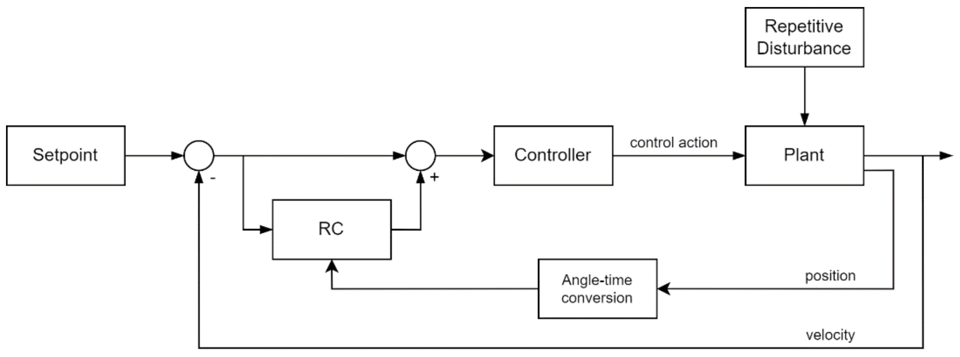


Figure 15: Angle-based RC controller

4.2.2.3 Progress beyond the SoA

Torque ripple in switched reluctance motors

The envisioned progress beyond the SoA in reducing torque ripple in switched reluctance motors is twofold. First, model-based commutation strategies may benefit from the realization that commutation problems are inherently over-parametrized and that the design freedom can be exploited to reduce torque ripple due to sampled-data effects. Second, data-driven commutation strategies currently rely on an accurate model-based torque observer, even if such a model may be unavailable in practice. Hence, the automatic generation of commutation functions in the absence of a perfect model is a possible direction for research beyond the SoA.

Implementation issues in repetitive control schemes – robust, inferential and delay-varying methods

The following results are expected as the project outcomes

- Development of generic design methods for automatic or semi-automatic synthesis of robust repetitive controllers:

  Robustness will be achieved in terms of inherent insensitivity to modelling errors both in the expected value of the fundamental disturbance period and in the assumed dynamic model of the controlled plant. As for the period sensitivity, new formulations of the design problem can be used to modify the structure of the exogenous signal generator. This can serve to shape the band-stop regions of the closed-loop sensitivity function according to prior information about the spectrum of the external disturbance. Insensitivity to plant modelling errors can be achieved by employing the latest robust control theory results, e.g., H-infinity loop-shaping methods for fixed structure controllers (Goubej and Schlegel, 2019).

- Development of new repetitive control structures in the inferential control setting:

  Applying repetitive controllers in motion systems with different feedback and performance variables requires careful modification of conventional structures to achieve optimal performance. The development of novel structures for repetitive controllers is envisioned for a class of single-input-two-outputs systems, where a maximum disturbance attenuation is required at a plant output that does not coincide with the one used for the stabilizing feedback control. Systematic design methods allow derivation and implementation of such unconventional control schemes to be developed, accounting for the stability and performance of the resulting closed-loop.

- Delay-varying repetitive controllers:

  Various control setups lead to variations in the fundamental period of the exogenous disturbance that may be periodic in another variable than time. Proper adjustment of the internal model in the repetitive controller is needed in such cases. Our goal is to propose algorithms capable of optimally adapting the control structure with respect to prior information about the disturbance properties. A specific application is the compensation of position-dependent disturbances in motion systems arising e.g., due to imbalanced load, periodic motion cycles or disturbances torques due to cogging and motor commutation.

Multi-rate realizations of the above-mentioned repetitive controllers will also be studied, extending possible employment of these control schemes when memory, SW or HW constraints may occur.

Position-dependent repetitive disturbance estimation and compensation

The main improvements expected beyond the SoA are:

- To obtain a reduction of the required memory and the computation effort allowing the implementation of the disturbance estimation and compensation in linear-parameter varying

(LPV) systems. The results will be achieved by employing a disturbance observer based on a low-harmonic model of the disturbance and exploring different representations of the disturbance, such as wavelet and linear regression.

- Design of non-collocated disturbance observer with low-computational requirements. This information will allow the controller to adapt its behavior to reduce the effect of the disturbance not only on the measured motor velocity but also on the load located after the elastic transmissions. The results will be achieved by employing state-space observers and adapting the setpoint accordingly.

4.2.2.4 Requirements

ID	**Requirement**	**Comments**
R10-D4.1	Preservation of (robust) internal stability of the closed-loop after the addition of the repetitive control structure	Careful design of both feedback compensator and repetitive control module has to be done as the inherent time-delay nature of the algorithm tends to destabilize the feedback loop. Robustness to plant and disturbance parameters variation is also an issue
R11-D4.1	Systematic design procedures allow automatic or semi-automatic synthesis and parameterization of the control structures without requiring a highly skilled operator	The goal is to minimize commissioning time and allow routine employment of the methods without the necessity for a deep background in advanced control engineering
R12-D4.1	Sufficient performance improvement compared to conventional feedback control to advocate more complex control structures requiring additional design, commissioning and maintenance costs	
R13-D4.1	The strategy should allow for flexibility towards different tasks	
R14-D4.1	Solutions to the over-parametrized commutation problem should penalize power consumption	
R15-D4.1	Control algorithms for switched reluctance motors should consider that reversing the direction of current does not reverse the direction of torque	

Table 3: Repetitive control requirements

4.2.3 Impedance or compliant control for robot manipulators

4.2.3.1 Introduction

In the last decades robotic technology has evolved rapidly for its application in industrial domains, with fast moving robotic platforms performing mainly stiff repetitive movements in very structured scenarios. This has represented a revolution in the industrial and automated manufacturing domain. Accuracy and fast movements are possible using high power actuators and stiff position-based control schemes. In this framework, a stiff robot end effector is designed to have predetermined positions or trajectories, despite any external forces applied to the robot. This is potentially dangerous for operators and thus their workspace is indicated with a security perimeter (hazard zone) that human operators shall not cross while the robot is working.

More recently, robotics has penetrated many other application domains that require Human-Robot Interaction (HRI) (Goodrich & Schultz, 2008). New generations of compliant robots intensively interact with humans (e.g., rehabilitation therapy (Volpe et al, 2009), social interaction (Wada & Shibata, 2007), and education (Baxter et al, 2017). Compliance motivates the use of new robotic elements, both hardware (flexible versus stiff materials, elastic actuators, low-power actuators, etc.) and software (torque versus position control, adaptive control systems, etc). Regarding hardware design, robots can be equipped with passive intrinsic compliance by means of different elastic components, muscle-like actuators, and/or soft materials. This approach offers a compliant alternative to classical stiff-bodied robots. But in this case (robots with elastic components), traditional position control methods are not of direct application thus demanding new control strategies (Diamond et al, 2012; Rus & Tolley, 2015).

Traditional position-based methods offer excellent accuracy for industrial rigid-bodied robots in well-structured environments (e.g., automated car factories), where HRI is explicitly avoided since safety cannot be guaranteed. On the other hand, when HRI is required, compliance demands torque control. But torque control strategies based on dynamics modeling cannot be efficiently applied since the nonlinearities of elastic components make detailed modeling or robot dynamics extremely complex (Chaoui et al, 2009). Artificial intelligence (AI) is now mature to address these challenges. In particular, widely used Artificial Neural Networks (ANNs) have been proposed and tested as a solution for the control of these compliant robots without requiring prior knowledge of the robot dynamics (Chaoui et al, 2009; Robinson et al, 2016).

The challenge: Traditional industrial robotic technology (position-based and powerful actuators) cannot be applied to robots interacting with humans because they are potentially dangerous in case of malfunctioning. Robots specifically designed for interacting with humans are called collaborative robots (Cobots) and need to be compliant (safe for human interaction). In robotics, compliant control can be defined as the allowance of deviations from its own equilibrium position, depending on the applied external force. In this framework, "passive compliance" can be achieved through elastic components at the effectors. This makes the robot inherently safer, since these elastic components will mitigate the impact in case of bumping into an object or operator.

Industrial robots mainly use stiff position driven control schemes with PD (proportional-derivative) controllers. The trajectories are defined in terms of target positions and the local controller implements torques depending on the distance to the target position. But in this case, the applied torque is not directly defined by a higher-level controller but rather at a lower level (local control) depending on the movement execution. This represents a threat for operators, since in case of external forces being applied to the robot, the local effector controller may drive compensation terms to correct these deviations from the target trajectories with torque commands not directly delivered by high control units.

In this framework, active robot compliance can be achieved by high frequency sampling of the applied torques and implementing safety driven mechanisms (such as breaking the robot in case of detecting high errors). A higher-level compliance control scheme can be implemented if accurate inverse dynamic models of the robotic plant are used. Inverse dynamics can be used to calculate the torques that a robot's actuator must deliver to make the robot's end-point move along a planned trajectory. But dynamic models of robots integrating elastic components at their joints are very complex.

4.2.3.2 State of the art

Classical algorithm control limitations based on analytical cobot model

Any robot dynamic model describes the relationship between the torque values applied on the robot joints and the resulting motion. This relationship is mathematically expressed using the analytic Lagrange formulation:

$$\tau = M(q'')q + C(q,q') + g(q) + \xi(q,q',q'') \qquad \text{(Eq. 4.15)}$$

where, (q,q',q''), are joint position, velocity, and acceleration, respectively. τ stands for the vector containing the joint torque values. $M(q)$ defines the robot inertia matrix. $C(q,q')$ matrix computes the inertia and Coriolis effects whereas $g(q)$ vector computes the gravitational effects on the robot. Finally, $\xi(q,q',q'')$ stands for the torque/force effects of those robot elements that were not considered elsewhere in the dynamic model, i.e., viscous friction, or the nonlinear effects of the series elastic actuators (SEAs) springs within the Baxter cobot (Fitzgerald, 2013).

Most rigid robots, equipped with high ratio gearboxes, assume $\xi=0$ in dynamic modeling since $M(q)$ and $C(q,q')$ torque contributions to the final τ are significantly larger than $\xi(q,q',q'')$. However, this is no longer the case for non-rigid robots (cobots) such as Baxter. Just a brief overlook on the rigid analytical dynamic model of Baxter (manufacturer's dynamic model) compared with the real Baxter cobot exemplified these divergences.

Modeling $\xi(q,q',q'')$ becomes key in accurately associating the applied cobot torque values and its subsequent motion. ξ related cobot parameters demand for accurate identification methods which are usually mathematically intractable (Wang et al, 2020). Mathematical intractability together with parameters that tend to degrade with age and/or usage, prevent using parametric methods for cobot dynamic modeling (Polydoros et al, 2015).

Considerations when modelling an elastic robot

Cobot joints are usually flexible with low ratio gearboxes. Gearboxes are to be modeled as if they were located ahead of the joint deformation. To obtain the dynamic model, the following three assumptions need to be considered.

Assumption 1: The joint flexibility is small so its effect shall be linear.

Assumption 2: Rotors are modeled as uniform bodies whose mass center shall be located on the rotation axis.

Assumption 3: The motors are located in the joints and specifically in a position ahead of the actuated link.

Cobots usually present elastic springs (as Series Elastic Actuators, SEAs) in between the rotor and the actuator. Thus, the elasticity in the joint i is modeled by a spring with stiffness $K_i > 0$, which is torsional for rotational joints. ***Assumption 2*** implies that the inertia matrix and the gravity term in the dynamic model of the robot are independent of the angular position of the motors. ***Assumption 3*** defines the most common configuration; rotors are solidly attached to the cobot structure, however, the displacement of the rotors along the structure shall dramatically influence on the movement dynamics.

A cobot dynamic model with flexible joints requires twice the number of state variables to fully characterize motors and links than a rigid bodied robot. These position variables present a similar dynamic which means that forward/inverse kinematics issues are almost identical to a completely rigid robot. We also need to define the state variable, the position-vector of the motor prior reduction. In this framework, the computation of the LaGrange equation becomes more complex, which considers the cobot as a black box system and computes the system kinetic energy 'K' and potential energy 'P'. The potential energy is caused by the gravity and the elasticity of the joint. Gravity is related to the position of the link and mass centers and rotor mass centers which are not coincidental due to ***Assumption 2.***

Defining a functional dynamic model would also demand for sensing the link rotation and the rotor rotation located prior to the elastic component, which is not available in most cases. Having access to those variables would imply dismantling the robot and locating new sensors.

We would also need the elastic component constants (K), which the manufacturer doesn’t usually provide. The behavior of elastic components (as SEAs) is not ensured to be similar when rotating clockwise or anticlockwise. The K vector is not constant either and varies depending on the operation conditions (heat, humidity, repetitions...) which implies a non-deterministic dynamic model.

Any analytic dynamic model will also demand for the inertia values for the rotors. Manufactures do not provide for those values with accuracy, nor the mass center of the rotor with respect to the robot chain. The presented scenario means that, for cobots, in which the elastic components matter for the dynamic modeling, half of the LaGrange equation elements cannot be calculated. The analytic dynamic model considering elastic components is therefore not available. This is what makes the approach challenging for cobots like this. And this is why approaches requiring these analytics cannot be directly applied.

Force control, understanding the problem (Impedance control)

The end effector force control is critical in practical tasks in which the cobot shall exert a certain force on a surface such as polishing, machining, assembling, etc. During the end effector contact with the environment, this environment may apply movement constraints to the end effector. Those movement constraints are referred to as kinematic. The end effector contact with the environment can be classified as inertial, dissipative, or elastic.

In the presence of an environment-robot end effector contact, considering only a position control would not be effective (and risky for the robot security). The motion control would indeed suffice only if the robot's task is always tracked as planned i.e., no perturbations or dynamic contacts are interfering. Keeping the cobot on track requires a detailed model of the robot (both kinematics and dynamics) and of the environment too (both geometrical and mechanical analytical models). Due to the unstructured nature of the environment and the elastic properties of cobots’ joints these analytic models are, on most occasions, unattainable

The control problem lies in the presence of non-modelled errors in the design of the robot's path. A non-modelled error can lead to the appearance of a contact force that is applied to the end effector which eventually forces it to move. When the cobot faces a deviation of the final effector with respect to the desired path, a classical PD controller does try to reduce that error generated by the contact force by applying torque to the motors to the point of being able to saturate them or to even break them down. The more rigid the environment, the more likely the motor saturation or breaking down problems.

The cobot-environment interaction control can be done passively or actively:

Passive control (passive compliance) is based on the fact that the interaction with the environment modifies the trajectory of the end effector. That is to say, that the robot arm itself, the joints or the links are elastic in some way. This is a way to protect both cobot and environment during end-effector environment interactions. The reaction of the passive control is instantaneous but it has to be designed for each specific application i.e., lack of flexibility. Note that since forces are not measured, passive control cannot guarantee that the contact forces are accurate enough for the task at hand.

Active control (active compliance). The end effector force guidance is ensured by a control system. The control system needs feedback access to torque and force values at stake. Note that active compliance control is slower than passive compliance control and the force/torque action requires minimal passive control to ensure that the torque and forces at stake are within a certain threshold.

Cobots such as Sawyer (Sawyer, 2022) or KUKA iiwa (KUKA iiwa, 2022) allow for both interactions at once combining the passive control instantaneousness (cobots are usually equipped with elastic passive actuators) guaranteeing that the controller contact forces are up to the task. Nevertheless, designing controlling schemes that take advantage of these features is challenging, since it requires accurate robot model building.

Indirect Active Control

Active control can be done through two approaches: indirect and direct active control. Direct active control mechanism requires feedback regarding force and torque value measurements, whereas indirect active control does not necessarily require a force feedback loop. Whilst the well-known hybrid force/motion control belongs to the direct active control family, impedance control belongs to the indirect active control family.

Impedance Control

Impedance control uses the deviation of the cobot end effector from the desired point as a result of an interaction with the environment according to an impedance-mechanics paradigm, that is, an impedance model of a mass-spring-damper system. Therefore, impedance control reacts to environment interactions (external forces) by generating a compensatory force towards the desired movement, whereas admittance control, on the contrary, reacts to the same environment interaction (external force) by imposing a deviation onto the desired movements.

In order to implement an impedance control scheme, we need to revisit the robot dynamics since the cobot dynamic behavior needs to be shaped. More specifically, we need to provide a given response or mechanical impedance in terms of force exerted to a certain robot-environment contact. We will also revisit the mechanical impedance significance using a spring-damped-system i.e., a mass that undergoes an acceleration being the robot-environment contacts a spring that is damped. The robot dynamic model and the spring-damped- impedance model will serve as the basis for the control law to be ultimately applied

Dynamic: Let us consider a robot that generates torque for its motors and an external force is applied to its end effector of the robot, the dynamics of the system follows:

$$M(q)q''+C(q,q')q'+g(q)=\tau+J_a^T(q)F_a=\tau+\tau_a \qquad \text{(Eq. 4.16)}$$

Where

- *J*$J_aT(q)$ is the Analytic Jacobian
- τ_a is the torque value applied to the robot motors given the external force F_a applied to end effector
- *M(q)* is the inertia matrix containing the masses of the robotic system.
- *C(q,q')q'* contains the Coriolis and centrifugal forces (assuming a non-inertial reference frame)
- *g(q)* gravity compensation term

Calculating F_a, (observer located at the end effector) via inverting the transposed Jacobian matrix:

$$M_x(q)x''+C_x(q,q')x'+g_x(q)=J_a^{-T}(q)\tau+F_a \qquad \text{(Eq. 4.17)}$$

Where

- $M_x(q)=J_a^{-T}M(q)J_a^{-1}(q)$ is the inertia matrix containing the masses of the robotic system.
- $C_x(q,q')=J_a^{-T}(q)C(q,q')J_a^{-1}(q)-M_x(q)J_a(q)'J_a^{-1}(q)$ contains the Coriolis and centrifugal forces (assuming a non-inertial reference frame)
- $g_x(q)=J_a^{-T}(q)g(q)$ gravity compensation term

Being the τ values at each joint:

$$\tau=J_a^T(q)\left(M_x(q)x''+C_x(q,q')x'+g_x(q)-F_a\right) \qquad \text{(Eq. 4.18)}$$

Mechanical Impedance

Mechanical impedance can be regarded as the ratio between force and speed. This concept is useful in order to define a control law to be used in impedance control. In a spring-damped-system as in Fig 16, the impedance will follow (in Laplacian space):

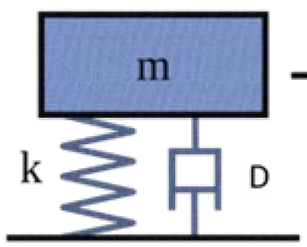


Figure 16: Spring-damped-system

$$Z_{impedance}(s)=\frac{Force(s)}{velocity(s)}=Ms+D+\frac{K}{s} \quad \text{(Eq. 4.19)}$$

Eq. 4.19 will therefore define the robotic response to an external force applied at the end-effector.

Design of an impedance controller

The first step in designing an impedance controller requires imposing a control law to be followed such as:

$$M_m(x''-x_d'')+D_m(x'-x_d')+K_m(x-x_d)=F_a \quad \text{(Eq. 4.20)}$$

Where x_d *is* the desired position, usually designed to allow going a little deeper into the robot-environment contact thus applying a certain Force on the environmental surface contact. The control law is established at the end effector, that is, (x,x',x'') that needs to be translated into joint torque command values. The torque values to be generated at the robot motor side are:

$$\tau=M(q)J_a^{-1}(q)\left(x_d''-J_a'(q)q'+M_m^{-1}\left(D_m(x_d'-x')+K_m(x_d-x)\right)\right)$$
$$+C(q,q')q'+g(q)+J_a^T(q)\left(M_x(q)M_m^{-1}-I\right)F_a \quad \text{(Eq. 4.21)}$$

From Eq. 4.21, we observe a dependence from the external force applied to the end effector F_a that needs to be measured in order to deploy impedance control. However, this expression can be simplified assuming the apparent inertia equals Cartesian inertia;

$$\tau=M(q)J_a^{-1}(q)\left(x_d'-J_a'(q)q'\right)+C(q,q')q'+g(q)+J_a^T(q)\left(D_m(x_d'-x')+K_m(x_d-x)\right) \quad \text{(Eq. 4.22)}$$

Eq. 4.22 shows no dependency from external force sensing, we do not need an end-effector sensor to measure the external force applied to the end effector F_a.

Note that it is possible to express the control law directly in terms of joint torque values:

$$\tau=M(q)q_d''+X(q,q')q'+g(q)+D(q_d'-q')+K(q_d-q) \quad \text{(Eq. 4.23)}$$

Where *K and D* are K_m and D_m (end effector reference frame) from the joint reference frame. We also assumed M=M(q) as aforementioned. The external force applied to the end effector from the joint reference frame perspective would be:

$$M(q)(q''-q_d'')+D(q'-q_d')+K(q-q_d)=\tau_a \tag{Eq. 4.24}$$

Coefficient design

Previously the design of M_m was assumed (the apparent inertia) equal to the Cartesian inertia; $M_m=M_x(q)$ to simplify the control scheme, however the value of K_m and D_m needs to be determined. Their values are usually chosen taking into account the following:

• In Cartesian *directions (i)* where contact is foreseen, a low (K_m, *i)* value shall be chosen.

• In the collision-free directions *(j)*, high (K_m, *j)* is advisable to make the position tracking on that *j* direction good.

• The (D_m, *i)* coefficients are used to experimentally model the transients.

As aforementioned, the analytic Jacobean needs to be available to accurately deploy impedance control (see control law). Cobot SEAs, whose inner parameters are to be represented in the analytic Jacobian, are commonly difficult to model due to their elastic components. These elastic components make the analytic Jacobian mathematically intractable, thus requiring discrete Jacobian approximation when possible. Impedance control accuracy and stability will mainly be applicable to rigid-body robots.

Note that impedance control only takes into account the contact forces at the robot end effector. Any other contact made with any robotic link is not accounted for, which may drive the robot to instability, not to mention the safety risks. A controller able to learn the robot's dynamic response to external forces over the entire robotic kinematic chain is required.

Machine Learning (ML)-dynamic identification

The classical methods for the analysis of a dynamic model of a robot are based either on the Lagrange (De Luca & Siciliano, 1991) or Newton-Euler equations (Buondonno & De Luca, 2015). These analytical methods involve the characterization of the physical parameters of the robot for its correct implementation. By simplifying the model, while remaining complex, it is possible to model the robot as a rigid body, omitting frictions and elastic constants whose effect may be minor depending on the mechanical configuration of the robot. This approximation usually works well for robot models with rigid joints; however, for robots with elastic joints (as cobots), the approximation is not accurate, and it is necessary to incorporate more complex equations that allow describing the dynamics of these elastic components with their corresponding characterization (Buondonno & De Luca, 2015).

Therefore, a good performance of these analytical models requires the identification and accurate characterization of the parameters of the robot components, including those phenomena that are nonlinear or stochastic. These phenomena are usually excluded from any mathematical description of the dynamics of a robot, even with highly complex analytical models. This limitation has motivated several proposals in recent years to obtain a dynamic model closer to the real one using Machine Learning (ML).

Different strategies have been proposed to generate dynamic robot models with artificial neural networks (ANNs). In (Chen et al, 2011), unidirectional and bidirectional recurrent neural networks (RNN) are used to learn the robot model being driven through position control; however, they do not obtain a dynamic model of the robot since they do not relate robot displacements to the applied torques. In contrast, in (Akyuz et al, 2011) they developed an inverse dynamic model with unidirectional RNNs capable of estimating torques from a given displacement for a robot with six degrees of freedom. Similarly, in (De Luca & Siciliano, 1991) they compare different RNN models studying their advantages in learning direct and inverse dynamics of robots with rigid joints. A hybrid approach to this problem has also been attempted, which consists of combining an analytical rigid robot dynamics model with a deep learning (DL) model (semiparametric models) (Fitzgerald, 2013). Note that in all these cases rigid robots, of lower dynamic complexity, have been used, and in no case have these techniques been applied to robots with elastic joints (cobots) with a priori more complex dynamics. This is important because cobots integrating elastic components are inherently safer but challenging for accurate torque-based control due to their complex dynamics.

ML dynamic modeling using recurrent neural networks (RNNs)

ML in cobotic control is required for dynamic modeling. Data gathering allows for dynamic cobot modeling, either forward or inverse dynamics:

Forward dynamics receives a set given by (τ, x_t, x_t', x_t'') (Among any other useful information) where τ is the set of torques applied on each joint and (x_t, x_t', x_t'') are the current position, velocity and acceleration of the end-effector. With this information, the goal of dynamic modeling is to predict $(x_{t+1}, x_{t+1}', x_{t+1}'')$, that is, the next position of the effector.

Inverse dynamics receives both the current state of the end-effector and its desired future state and then it tries to predict the set of torques τ that are to be applied to reach that desired state.

Usually, these dynamic models are built using a specific type of artificial neural network (ANN) called recurrent neural networks (RNNs). RNNs efficiently work on sequential data, also known as time series, and they use information of past states to calculate the current and future outputs. This makes them ideal to work on motion/torque control since past applied torque values are known to affect inertia on the robot limbs, and therefore, to the current and future state of the cobot movement.

RNN can have a single input, i.e., using the current state to predict the torque values of the next *t* future time steps, or a single output, i.e., using the past *t* states to predict the next position of the end effector.

Since real world movements are usually a continuous motion, they need to be discretized. This discretization is done by defining a time-step δ and using x_{t-1} to calculate the speed (x'), and x_{t-2} to calculate the acceleration (x'') of the end effector. As many steps as desired could be included within the model but three (x_t, x_{t-1}, x_{t-2}) is the minimum number recommended in order to be able to estimate the speed and acceleration of the effector or joint (Centurelli et al, 2022).

RNNs are usually built using “cells” which act as layers on an ANN. Some of the more popular RNN cells used for time series modeling are: a) Long Short-Term Memory (LTSM) (Hochreiter & Schmidhuber, 1997) and b) Gated Recurrent Unit (GRU) (Cho et al, 2014).

There exist programming “commodities” that facilitate the design and development of RNNs, such as TensorFlow (Abadi et al, 2015) or Pytorch (Paszke et al, 2019). These libraries allow for a modular design of deep ANNs and are extensively used in many fields. Although they usually run-on high-level programming languages like python and they are designed to work on the cloud, they can be adapted to edge-computing platforms. In order to bring these complex models from cloud computing to edge computing many tools are available, like ONNX or Torch-Script.

ML-based control. Controllers based on Reinforcement learning (RL)

Reinforcement learning (RL) is an ANN subfield of machine learning (ML), in which an optimal control strategy is defined through trial and error learning whose performance is given by a set of rewards and punishments, i.e., the policy, to be assigned based on the control behavior (accuracy, velocity, compliance, etc). RL tries to replicate the human and animal motor learning process when moving and it is well suited on exploratory motor tasks that require good generalization to achieve high performance in terms of precision, velocity or compliance. This generalization and exploration capacities make RL useful when working on finding an optimal torque control solution for driving a cobot.

The RL working principles in setting a torque cobot controller are based on torque-operating a cobot repeating a set of motor tasks (try and error process) and subsequently on giving the RL controller rewards based on the policy followed, i.e., performance in terms of accuracy, velocity, compliance.

The RL framework is based on the Markov Decision Process (MDP) which describes interaction between the agent and its environment. This framework consists of a 5-tuple (S,A,P_a,R_a,γ) where:

- S is the *state space*. Each state *s* is usually represented by a vector containing the relevant information for the controller (e.g., effector and joints positions, velocities and torques).
- A is the *action space*. Each action *a* corresponds to the output of the controller, usually in the form of the torques to be applied by each joint.
- $P_a(s,s')$ is the probability that action *a* takes the agent from state *s* to state *s'*.
- $R_a(s,s')$ is the reward obtained when successfully transitioning from state *s* to state *s'*. As mentioned above, this function represents how well the agent is performing the task. In most cases this function is based on the cartesian distance to the objective path, but many other alternatives exist. Building a good reward function which gives the agent constant or regular feedback in order to improve its policy is key to creating a successful RL model.
- γ is the discount factor that weighs how important future rewards are. In some cases, this discount factor is disregarded in favor of models with long term goals. If this factor is high the agent will act in a greedy manner and pursue only short-term rewards.

Once the architecture of the model is defined, an optimization algorithm must be established. An example can be seen in (Centurelli et al, 2022) where Trust Region Policy Optimization is used to train a RNN to work as an inverse dynamic model of the system. They also use a forward dynamic model to simulate the response of the cobot instead of using real data.

Adapting RL to cobot control poses several challenges. An RL controller, like any other ANN approach, will be as good as the training data set being used. Gathering the RL training data becomes pivotal to tackle control challenges such as the accountability for the wear and tear of the cobot joints or their inner elastic component. Offline RL ANN training is advisable due to real-time constraints and, more importantly, preservation of the cobots security. RL trial and error training is therefore adequate for simulation modeling prior to its deployment. Simulations mostly make use of rigid-body dynamic models which makes the sim- to-real transfer critical in the RL performance in a real environment.

There are infinite ways of covering the working space by the cobot operation, finding the subset of data derived from the robot operations that achieve a reward, and subsequently cause the model to start learning, may require high-capacity, compute-intensive requirements. This computational overload has motivated using imitation learning (Fang et al, 2019) (what a human being shall do) to accelerate training. Once the imitated subset of data training is provided, RL can optimize control over that subset in terms of the policy established beforehand. Note that the main disadvantage of this speeding up of the RL learning process lies in the local RL learning optimization to a particular cobot movement as consequence of the reduction of the RL search space.

Controllers based on Supervised learning

Supervised learning (SL) is a subfield of ML, in which labeled data (data for which we know the expected output) is used to train a model by teaching it with the appropriate outputs for each input in the dataset. SL is widely used in all fields due to its efficiency and learning capabilities. Training is usually limited by the dataset size, due to the high cost of labeling big amounts of data.

In the control field this limitation can be eased by using the sensors of the cobot and simulations to easily gather large quantities of data. Controllers of this kind typically use the desired path as labels and measure their error as a function of the distance between the desired trajectory and the obtained trajectory.

Imitation learning can be regarded as a form of supervised learning. In this case, a target trajectory is defined by a human demonstration, then a SL controller tries to replicate that same path as close as possible to the original, and it uses the distance between the followed and the target paths as an estimation of the error in the model.

Note that by only using SL, the controller shall be as good as the data provided (target trajectory), and, unlike RL, it lacks the ability to find new optimal paths or to further optimize the given one. However, this limitation is assumed provided that faster and easier learning can be achieved.

As with RL, most controllers use RNNs, although other ANN types can also be used for complementary tasks. The error function of a RNN is usually implemented using the Euclidean distance of the end-effector to the desired position; however, this function can be implemented using other distance metrics i.e, Manhattan distance, joint distance, etc. Actually, it is also common to measure the error on each joint instead of the effector's error, however, the Euclidean distance usually offers a closer estimation of the global performance of the model.

Note that, a controller architecture deployment for a cobot may demand a combination of both, RL and SL controllers in order to cope with the possible dynamic perturbation, sudden interaction forces or any unpredicted event that may occur (SL) or may need to be learned (RL) in a non-structured scenario.

Iterative Learning Control

Iterative Learning Control (ILC) is a learning control method different from machine learning, usually simpler and easier to tune but only applicable in scenarios of repetitive tasks. ILC can be generally divided in two approaches: Frequency-domain ILC and Norm-optimal ILC (Bristow et al, 2006). Both methods rely on performing a number of trials, using (feedback and) feedforward, in which the error is observed. This error is passed through a learning filter in order to obtain a feedforward signal for the next trial.

- In Frequency-domain ILC, the learning filter is given by the approximate inverse of the process sensitivity (for closed-loop) or the approximate inverse of the plant (for open-loop). Even if this model is imperfect, excellent performance can be achieved after a small number of iterations. However, if the model quality is too low, the ILC scheme might become unstable in the trial-domain. This can be overcome by introducing a Q-filter (chosen by the user, e.g, a lowpass-filter) to turn off learning at certain frequencies (Bristow et al, 2006). This facilitates convergence at the cost of a larger final error. The user should be aware that the inverse of a strictly proper, potential non-minimum phase system, can lead to an unstable and non-causal learning filter. Stable inversion techniques exist to deal with this issue (Blanken et al, 2017).
- In norm-optimal ILC, a cost function is posed in the time-domain for a finite-length task. Instead of filters expressed as discrete-time transfer functions (or difference equations), convolution matrices are used to express models. The cost function is defined to penalize the expected error at the next trial, given the data of the previous trial, and it is quadratic in the next feedforward signal (the design variables). It can be solved analytically (Bristow et al, 2006) for the unconstrained case or using a convex solver for the constrained case (Mishra et al, 2011).

4.2.3.3 Progress beyond the SoA

The expected outcomes of the project are the following:

Exploring different schemes for acquiring the dynamic cobot models

a) Off-line model building using explorative movements with the goal data-gathering. This data-set is stored relating torque-movement, i.e., cartesian and or multijoin movement. ML techniques can be applied in order to acquire the robot dynamic model. Eventually the model can be applied for cobotic torque-control tasks in later operation stages.

b) On-line model building in which the cobot models are refined during operation so that the torque-control scheme can adapt automatically to changes in the operation conditions or the environment.

Building inverse and forward dynamic models

The planned work will explore how to efficiently capture accurate inverse dynamic models of cobots. We will explore how to acquire the models within a specific "calibration stage" (robot data acquisition stage in off-line model building) and then apply torque-control algorithms able to efficiently use of this accurate dynamic model.

We will also explore how to acquire forward dynamic models of cobots to be used within control loops to compensate for delays in the control loop and also to detect deviations between the robot model and the actual robotic platform (early failure detection).

Accelerate learning

We will explore how to accelerate the learning phases by better focusing the learning process. For instance, we will use adversarial neural networks (Lee et al, 2021) as well as parameter optimization techniques (such as evolutionary algorithms (Ayala & dos Santos 2012)) to allow optimal control actions and data to be learned during the training phases in simulation that ultimately accelerate the actual deployment of the physical torque control solution.

We will also explore how to take advantage of analytical rough models (such as rigid body dynamic models) to accelerate model building processes using robot simulators and how to overcome the sim-to-real gap with incremental learning with the robot on the loop (hardware on the loop) on a subsequent refining stage. This is key, since simulation allows massive exploration of the operation space without degrading the robot performance nor exposing the robot to scenarios (such as operations close to singularities) that may lead to breakdown or failures. We will explore how the models can be partially captured using simplified simulations and then refined using the actual robotic platform with incremental learning techniques. In this second stage with hardware-in-the-loop (HIL) the security of the robot will be guaranteed thanks to the massive exploration done with the simulation tools.

Integrating machine learning into the control loop

The project will explore how to use different ML techniques (such as supervised learning, reinforcement learning) for capturing the models and applying them on smart control algorithms. This requires research also on how to efficiently integrate the captured models in the control loop. Different machine learning techniques (for instance supervised learning and reinforcement learning) can be integrated complementary within the control loop.

4.2.3.4 Requirements

ID	Requirement	Comments
R16-D4.1	Compliant applications must use torque-based control schemes	Safety motivates the torque to be applied to the effectors to be constrained. Thus, torque-based control schemes that provide the actual torque to be applied are key for safe human-robot interactions. Other control schemes (position based) require specific safety mechanisms (e.g., fast sampling and stop reaction mechanisms)
R17-D4.1	Accurate dynamic models must be automatically captured	Data-driven model building is key. Analytic dynamic models of cobots are not accurate. Furthermore, the robot's dynamics may change along operation since these elastic components may behave differently along operation cycles or the robot life cycle
R18-D4.1	Smart control algorithms must integrate models and adaptation mechanisms within the control loop.	
R19-D4.1	Control algorithms must be robust to modelling errors	
R20-D4.1	Iterative techniques must be able to achieve good performance even in the presence of trial-variant disturbances	
R21-D4.1	Stability of the system in both training and tests must be demonstrable, i.e, it must be safe	
R22-D4.1	Techniques in machine learning, applied to control, must have interpretable hyper-parameters such that the user knows what to expect when changing the values	
R23-D4.1	Real-time algorithms must be sufficiently resource-efficient to be applicable to hardware that is standard in industry	

Table 4: Compliant control requirements

4.2.4 Disturbance Observer for Friction Compensation

4.2.4.1 Introduction

Recent advances in Computer Numerical Control (CNC) machining processes have brought fundamental improvements such as the complete automation of the machining process and the programming procedure, allowing increased accuracy and surface quality to be achieved. However, the use of conventional CNC machines poses important limitations such as a restricted working area, the shapes that can be obtained and the economic cost of the equipment.

For certain applications, robots could replace the functionalities of CNC machines and become a real alternative to them. The advantages offered by robotic technology are a larger workspace, flexibility to apply them in different manufacturing processes or even auxiliary operations, and at a considerably lower cost than a CNC machine. However, the limitations of robots in terms of absolute precision and rigidity (Pan et al, 2006) are restricting their use in industrial applications. In recent years, work is being done to improve the accuracy of robots to overcome this problem, both by using real-time process control systems (Denkena & Deker, 2015) and by developing simulation models to generate compensated trajectories (Karim et al, 2017).

One of the requirements for robots to be used in manufacturing processes such as machining is the need for a control system that can be fed from trajectories generated by CAM software, that can resolve kinematics and interpolate trajectories at high frequency, adapt motion dynamics and that can synchronise movements with additional axes (Altintas, 2012).

The last few years have seen a change in trend and some industrial robot manufacturers, such as KUKA, are now offering CNC modules in their controllers for machining applications (KUKA.CNC, 2022). Another strategy not offered by all robot manufacturers but which is starting to be seen more and more is the possibility of using conventional controllers to control robots, both directly by attacking the controllers and indirectly by attacking the position/speed control loops of the robot controller (COMAU: Power System & Products, 2022).

In CNC machine tools, this variable is the position signal that generates a trajectory that must be followed during cutting. Large manufacturers of control systems continue to provide a cascade P-PI solution for their machine tools, due to its robustness, low cost, and relatively simple tuning rules (Sun et al, 2018). Many model-based control strategies have been explored, such as model predictive control (Serkies & Szabat, 2013), and robust control (Ponce et al, 2015).

Although there are solutions on the market for controlling industrial robots with the CNC, there is no generic approach available. In order to find a standard solution, the necessary functionalities such as kinematics, trajectory generation and control loops will be integrated into the CNC itself, and therefore will be common to all robots. To mitigate the dynamic effects of the robots, MIMO control strategies will be implemented, which will have to take into account friction and backlash compensations.

Friction is a well know nonlinear phenomenon in contact physics. Research carried out in this area shows that there are two states in the friction process: a sticking (or pre-sliding) state and the sliding state. In the sticking regime, the roughness (asperity) of the contact surfaces creates a friction force that is dependent on the micro-displacement between both contact surfaces. The roughness elements of the surfaces are deformed like nonlinear elastic elements, creating a sticking force on the surfaces. When this force exceeds a threshold, the surfaces start sliding, and the friction force behaves dependant on the sliding velocity.

The friction causes significant problems in mechatronics systems: static errors, stick-slip, etc. These problems are no compensated by the conventional (PID based) nested current, velocity, and position control loops. Precision machining needs the design of additional control strategies to compensate the friction phenomenon.

4.2.4.2 State of the art

The friction is an important phenomenon that appears in motion control systems, which can be defined as the tangential reaction force between two surfaces in contact that opposes the relative motion between them. Many physical effects are at the origin of this phenomenon: elastic and plastic deformations, material properties, etc.

It is necessary to understand the process of friction, to reduce its adverse effect in motion control. For this, several friction models have been proposed over the last years (detailed information about friction models is available in Olsson et al, 1998; Åström, 1996; van Geffen, 2009). These models can be divided in static and dynamic models.

Static friction models

They are the simplest models and describe the friction as a function of the relative velocity of the sliding surfaces:

$$F_C = \mu F_N \, sgn(v) \qquad \text{(Eq. 4.25)}$$

Where μ is the friction coefficient, F_N the normal load and *v* is the relative velocity. This model is called Coulomb friction model.

One problem of this model is that it does not capture the friction process when the relative velocity is zero. A simple way to capture this stiction behaviour is augmenting the Coulomb friction model specifying the friction force at zero velocity.

$$F=F_e \quad \text{if } v=0 \text{ and } F_e<F_s \qquad \text{(Eq. 4.26)}$$

$$F=F_s \, sgn(F_e) \quad \text{if } v=0 \text{ and } F_e \geq F_s$$

Where *Fe* is the applied force and *Fs* the stiction force.

More sophisticated models are also used to describe the dependency of the friction force with the sliding velocity. Among them, the most popular is the Stribeck curve:

$$F_v = F_C + (F_S - F_C) \cdot e^{-\left(\frac{v}{v \cdot s}\right)^{\delta s}} \qquad \text{(Eq. 4.27)}$$

Where *vs* is the Stribeck velocity and δs the Stribeck shape factor.

Some additional examples of static friction models are the Karnopp model (Karnopp, 1985) and the Armstrong’s model (Armstrong-Hélouvry et al, 1994).

Dynamic friction models

The dynamic friction models are obtained considering that the the friction phenomenon is indeed a dynamic process. The most popular models are the Dahl’s model (Dahl, 1968), the LuGre model (Canudas et al, 1995) and the GMS model (Al-Bender et al, 2004).

The Dahl’s friction model is an extension of the Coulomb friction model, but it produces a smoother transition around zero velocity. The frictional hysteresis at pre-sliding state is approximated by a 1st order differential equation of the position depending only on the sign of the relative velocity. The friction force is given by:

$$\frac{dF_f}{dt} = \sigma \left(v - \frac{F_f}{F_C} |v| \right) \qquad \text{(Eq. 4.28)}$$

Where σ is the contact stiffness.

The Dahl’s friction model has been widely used, but it exhibits some drawbacks: it does not describe stick slip motion and it does not capture the Stribeck effect.

A cooperation between the Lund and Grenoble universities has developed the LuGre friction model. This model is the result of combining the pre-sliding behaviour described in the Dahl’s model with the sliding regime described by the Stribeck curve.

The friction force is the sum of three components, given as a function of a state variable *z* and the relative velocity *v*:

$$F_f = \sigma_0 \cdot z + \sigma_1 \frac{dz}{dt} + \sigma_2 \cdot v \qquad \text{(Eq. 4.29)}$$

Where σ_0 is the contact stiffness, σ_1 is the micro-viscous friction coefficient and σ_2 is the viscous friction coefficient. The state variable *z* represents the deflection of the surface asperities (elastic springs) and is related to the bristle interpretation of friction:

$$\frac{dz}{dt} = v - \sigma_0 \frac{v}{s(v)} z \qquad \text{(Eq. 4.30)}$$

Where *s(v)* is the Stribeck curve described previously.

In the GMS friction model, the friction force is calculated as the sum of all the forces of the elementary elasto-slip elements and the viscous friction force:

$$F_f = \sum_{i=1}^{n} F_i + \sigma_2 \cdot v \qquad \text{(Eq. 4.31)}$$

Where F_i is the force in each elasto-slip element and $\sigma_2 \cdot v$ is the viscous component. The force in each elasto-slip element depends on its movement state:

- If the elementary model is sticking, the force is calculated as:

$$\frac{dF_i}{dt} = k_i \cdot v \qquad \text{(Eq. 4.32)}$$

Where k_i is the stiffness coefficient of the elasto-slip element.

- If the elementary model is slipping, the force is calculated according to:

$$\frac{dF_i}{dt} = sgn(v) \cdot C \left(\alpha_i - \frac{F_i}{s(v)} \right) \qquad \text{(Eq. 4.33)}$$

Where C is the attraction parameter that determines the attraction of the total friction force towards the Stribeck curve in sliding, and αi is the fractional parameter corresponding to the i-th elasto-slip element.

The elementary model is sticking until $F_i > \alpha_i \cdot s(v)$. The elementary model is slipping until the velocity crosses zero.

Other friction models can also be found in the literature (Olsson et al, 1998). Some examples are the Bristle Model, the Reset Integrator Model, models by Bliman and Sorine, model for lubricated contacts, the Leuven Model, etc.

Friction Compensation Methods

Several friction compensation methods can be found in the literature (Olsson et al, 1998; Åström, 1996; van Geffen, 2009), which can be divided in the non-model-based and model-based techniques. Among the first, may be mentioned the impulsive control, based on applying small impacts, dithering, which consist of adding high frequency signals in the control, Iterative Learning Control (ILC) for repetitive trajectories and strategies based on friction estimators.

On the other hand, regarding the model-based techniques, can be further divided into feedforward and feedback solutions. For this it is necessary to have a good friction model. This model is used to estimate the friction force acting on the system, and the control signal is added with an additional component that compensates this estimated friction value.

For the model-based friction compensation techniques, the problem does not lie in the friction model, but in the characterization of the different parameters involved in this friction model. Purpose-built testing must be performed for this, which can become labour-intensive and time-consuming task. Examples of the estimation of the parameters of a friction model, specifically the parameters of a GMS friction model, are available in (Grami & Gharbia, 2017; Grami & Fareh, 2018).

4.2.4.3 Progress beyond the SoA

For the reasons presented above, the selected option in this work will be to design a friction compensation strategy relying on a non-model-based technique, specifically a solution based on a model-free friction estimator (Ray et al, 2001; Lampaert et al, 2004).

The main idea to overcome the problems caused by the friction phenomenon is to add an additional term to the conventional system control command signal. This additional term is precisely the estimate of the friction force:

$$u_{command} = u_{control} + \alpha \cdot F_{friction} + \beta \frac{dF_{friction}}{dt} \quad \text{(Eq. 4.34)}$$

where $u_{command}$ is the total action on the system, $u_{control}$ is the command from the closed control loop, α and β are gains and $F_{friction}$ is the estimated friction force component.

The solutions proposed in the technical papers usually rely on the compensation of the estimated friction force. In this new approach, the derivative of this estimation will also be added, to guarantee a faster action on the friction compensation.

A model-free friction estimator (model-free in the sense that it does not depend on any specific friction model: Dahl, LuGre, GMS, etc.) will be developed to estimate the friction force.

In the literature, the friction observers are usually based on 1 d.o.f. models. In the present work, a 2 d.o.f. will be used. This will allow to obtain a better representation of the real system, and consequently, to obtain a better estimate of the friction force. Remember that, from the control design point of view, a good approximation of a real system can be obtained using a 2-mass system, if the dynamic characteristics of the simplified model matches adequately the dynamic characteristics of the lowest resonance of the system.

The first step in the development of the friction force observer is to approximate real system by a simplified two-mass model, as in the figure:

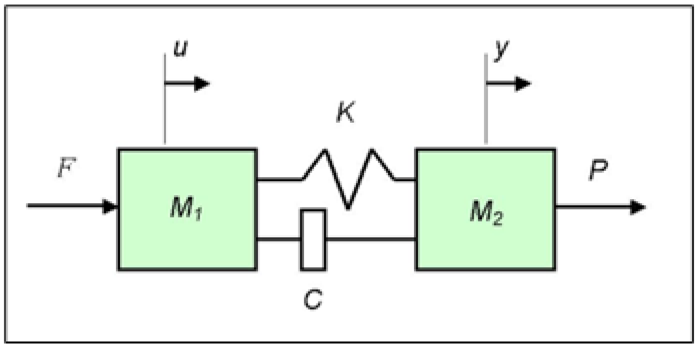


Figure 17: Two-mass system model.

Where M_1 is the drive-side mass, M_2 is the load-side mass, K, C, are the stiffness and damping of the most flexible element between drive and load, and u, y are the displacements of the drive and load respectively. Drive-side and load-side masses also include the masses of all the transmission elements placed at the drive and load sides respectively of the main flexibility in the transmission chain.

The usual second-order dynamic force equilibrium equation can be described in state-space form using modal variables:

$$\{q'\}=[A]\{q\}+[B]\begin{Bmatrix}F\\P\end{Bmatrix} \qquad \text{(Eq. 4.35)}$$

where *{q}* is the state vector (modal variables), *A* is the state matrix, *B* is the input matrix, and *F*, *P* are the force exerted by the drive and the friction force on the load respectively.

Although the force exerted by the drive is known, the friction force value remains unknown. The arrangement shown in the previous equation is not actually applicable. For this, a non-model-based friction force estimator will be built. The system state (the modal variables and their first order derivatives) will be augmented with the friction force component and its derivatives. This augmented state will be estimated using the Kalman Filter (KF) technique.

The friction force can be modelled as a random walk of any order. The simplest case is to use a first order model, where the friction is considered as a random constant. Using higher order models enables a better response of the friction estimate in face of transients. In this case, a second-order random walk model will be used. It is still a simple model and provides acceptable results. The structure is as follows:

$$\begin{Bmatrix}P'\\P''\end{Bmatrix}=\begin{bmatrix}0 & 1\\0 & 0\end{bmatrix}\begin{Bmatrix}P\\P'\end{Bmatrix} \qquad \text{(Eq. 4.36)}$$

The state-space representation of the second-order dynamic shown above is augmented with this second-order random walk model. In this way, the only input to the system state-space is the force exerted by the drive, *F*, which is known.

A state observer will be designed to estimate this augmented state, and hence the friction force perturbation. For this, the usually available signals will be used: the velocity of the drive, *u.*, and the position of the load, y. Additionally, and to respond better to transients, the acceleration signal on the load, y'', (easily measured with an accelerometer) will also be used.

The observer gain, *K*, will be calculated using the KF technique. To obtain a compromise between robustness and good sensitivity to transients, process noise covariance, *R*, and measurement noise covariance *Q*, will be carefully selected. Once *Q* and *R* are defined, the observer gain *K* can be computed offline, and only the state propagation and update equations are implemented in real time. Next figure shows the structure of the system with the controller and the friction estimator:

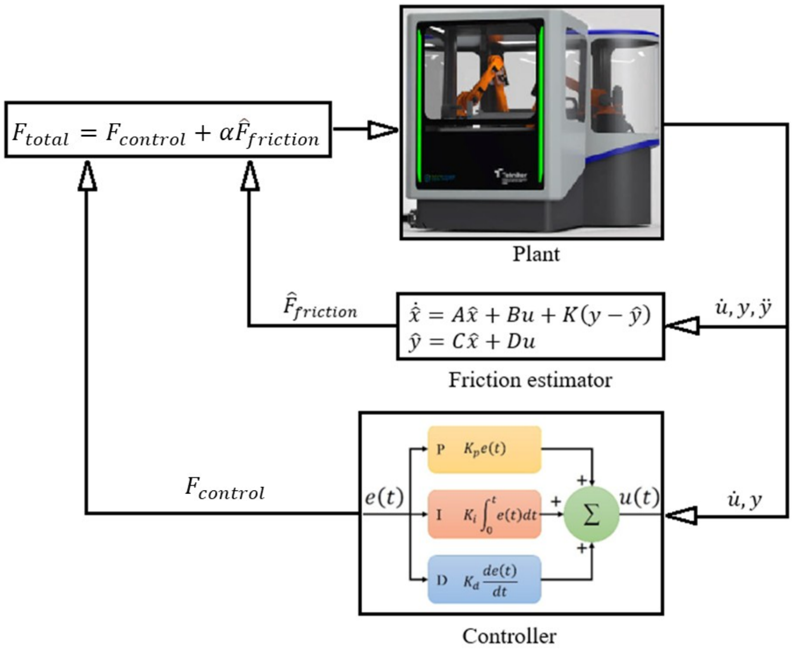


Figure 18: System structure with controller and friction estimator.

4.2.4.4 Requirements

ID	Requirement	Comments
R24-D4.1	Friction compensation must be added to the control action of the applied controller (current control loop input)	
R25-D4.1	Friction compensation must be executed at a high sampling frequency	Minimum sampling of 1 kHz
R26-D4.1	Although characterisation of friction is not required, a two-mass model of the controlled system is required	

Table 5: Friction compensation requirements

4.2.5 MPC Control

4.2.5.1 Introduction

The Model-based Predictive Control (MPC) methodology is also referred to as the moving horizon control or the receding horizon control (Mayne, 2014). The MPC is constructed using control and optimization tools. In the MPC approach, the current control action is computed on-line rather than using a pre-computed off-line control law. A model predictive controller uses, at each sampling instant, the plant's current input and output measurements, the plant's current state, and the plant's model. Some performance index defined by the user is optimized (restricted to constraints) over a sequence of future known inputs. The result of this optimization is used to:

- Calculate, over a finite horizon, a future control sequence that optimizes a given performance index and satisfies constraints on the control action.
- Use the first control in the sequence as the plant's input.

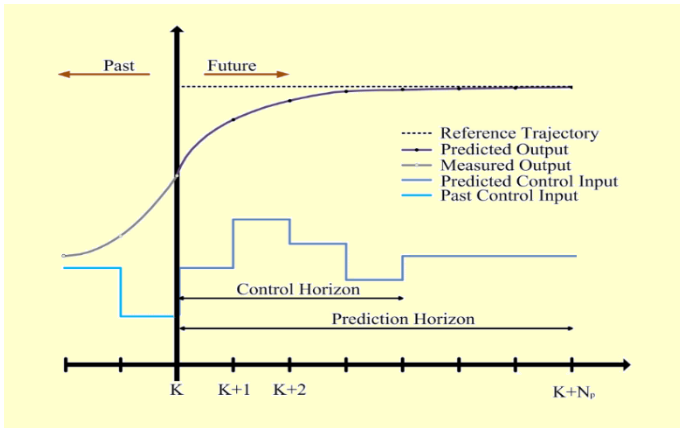


Figure 19: Schematics of MPC.

Mathematically, given a usual discrete-time state-space representation of a time-invariant linear system:

$$X_{k+1} = A X_k + B U_k$$
$$Y_k = C X_k \quad \text{(Eq. 4.37)}$$

Where *X* is the state vector, *U* the input vector, and *Y* the measured variables, the constrained linear MPC consists of solving the next optimal control problem (quadratic performance index):

$$\min \left(Y_N{}' \cdot P \cdot Y_N + \sum_{k=0}^{N-1} \left(Y_k{}' \cdot Q \cdot Y_k + U_k{}' \cdot R \cdot U_k \right) \right)$$

$$\textit{with constraints}: \quad u_{min} \le u_k \le u_{max} \quad \text{(Eq. 4.38)}$$
$$y_{min} \le y_k \le y_{max}$$

Where *P*, *Q*, *R*, are weighting matrices. The state-space of the system is used as prediction model:

$$x_k = A^k x_0 + \sum_{j=0}^{k-1} A^j B u_{k-1-j} \quad \text{(Eq. 4.39)}$$

N represents the optimization horizon, and only the first value of the calculated control sequence is used each time.

The main difference with the usual PID control is that the PID control builds the control action based on current and past data, whilst the MPC control additionally uses future reference and known disturbances to the system. According to (Schwenzer et al, 2021), the classical control can suit perfectly 90% of all the control problems. Only for the remaining 10% fraction advanced control solutions like the MPC are necessary. The main drawback of the MPC strategy is that a real-time optimization problem must be solved repeatedly, resulting in a higher computational cost compared with the classical PID control. But the use of model-based optimization, the inclusion of future information and the flexibility of considering restrictions makes the MPC an attractive approach for industrial applications.

4.2.5.2 State of the art

Regarding the industrial applications, MPC has been notably used in process industry, see (Schwenzer et al, 2021; Abdulrahman et al, 2020). MPC has become a standard approach to manage process industry, characterized by complex multi-variable systems with time delays.

Linear MPC strategies are used in processes with weak nonlinearity, such as refining and petrochemical process. On the other hand, Nonlinear MPC is used in processes with high nonlinearity, such as paper manufacturing and chemicals. But the high optimization time consumptions caused by the complexity of the nonlinear process, makes the Nonlinear MPC more suitable for processes with slow dynamics. Additionally, the presence of hybrid systems, represents an additional problem.

An approach to face nonlinear systems is the utilization of linear models. For this, linear models are obtained in the plant operation points. The great benefit of using linear models is that when using a quadratic objective function, the resulting problem is a convex problem.

Considering the industrial process control, MPC is usually integrated as a high-level supervisory control of classic PID loops in a cascaded control structure. The nonlinear models are usually linearized, and the optimization calculation times varies from seconds up to few hours depending on the process.

In the last decade, the use of MPC has raised in building climate. The aim is to reduce the energy consumption keeping thermal comfort (Heating, Ventilation and Air Conditioning, HVAC). MPC is a powerful instrument that benefits the available weather forecasts. The main problem for HVAC applications is the high modelling effort needed to correctly model a building and the energy generation and distribution systems.

Another field of application are the renewable energies. Usual objectives in wind turbines are the power regulation and load reduction. To apply an MPC structure, the knowledge of incoming wind is necessary: wind speed observers or measurement devices (Lidar) are used for this.

Other field of application of MPC is power electronics. As a difference of process industry, power electronics is characterized by very fast dynamics, in the range of ms. Hence, relatively simple models and short horizons are needed. In some cases, online optimization is removed and explicit MPC is used (optimization problem is solved in advance for a variety of cases). Examples of application are the control of electrical AC drives, the control of power converters and rectifiers, direct torque control of electrical drives, etc.

There are many other areas where MPC is also emerging: robotics, cranes, and manufacturing. Regarding manufacturing, path tracking is one of the objectives of the MPC (Stephens et al, 2013). Other objectives include, for example, the reduction of the friction effect (Rodriguez-Ayerbe et al, 2014) or for reduction of power consumption peaks in machine tools with periodic behaviour (Ubach-Pallas, 2017).

Cascaded PID control structures are the most usual approach to control mechatronic systems. These control structures rely only in past and current data. As an alternative solution, MPC structure can be beneficial in exploiting the prior knowledge of the trajectory. Some attempts have been performed to implement an MPC for trajectory tracking in DSDs, see for example (Stephens et al, 2013), or model predictive contouring (Khalick & Uchyiama, 2010; Lam et al, 2011). The main limitation is the computational effort needed to solve online the optimization problem and the fast update rates used in DSDs.

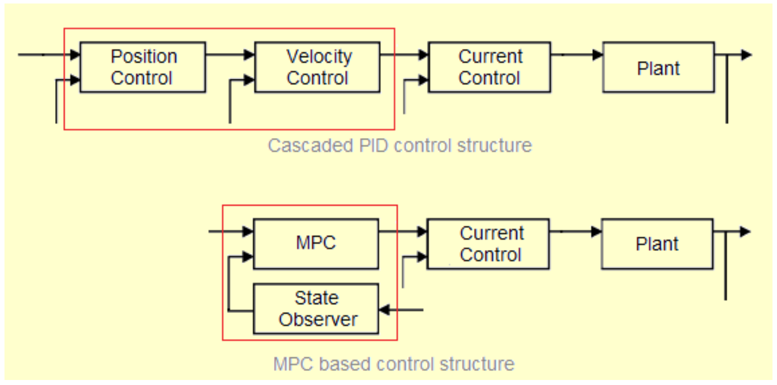


Figure 20: Schematics of control structures for DSDs.

In the last few years, research has been directed to develop fast MPC algorithms intended for embedded implementations. With the aim of reducing the time taken in the optimization problem, several solutions have been analysed in the literature: exploit the MPC structure tailoring the optimization process, multiplex the optimization procedure, development of the Economic MPC (EMPC) (Angeli & Müller, 2018), etc.

Another solution proposed in the literature is to perform the optimization offline (explicit solution approach) (Bemporad et al, 2002). The resulting gain-scheduled solution needs to interpolate the control action depending on the state vector and desired future output at each time step. This can be cumbersome depending on the number of states variables. Binary storage and search approaches have also proposed to reduce this storage and time consumption. Anyway, the memory requirements of this approach can become prohibitive for medium to large-sized systems and/or problems involving many constraints.

Other option is to use efficient algorithms capable of solving online the optimization problem specifically tailored to embedded systems. In this way, Krupa et al. (2021) propose a sparse, low-memory footprint optimization algorithm for the implementation of the model predictive control (MPC) for tracking formulation in embedded systems. This solution is dedicated for the implementation of these controllers in devices with very limited computational and memory resources.

4.2.5.3 Progress beyond SoA

Implementation of the MPC in real time for set-point tracking and specially for trajectory tracking is difficult. Hence, in this work an MPC for trajectory tracking will be designed, with the main objective of its industrial applicability. A very simple and robust solution is desired. Next aspects will be analysed in the proposed solution:

- MPC requires toolchains with some parts that are available off the shelf, but a complete and robust MPC solution requires a highly-integrated toolchain. Such toolchain will be developed, making a MPC controller building block that can substitute PID building blocks with minimal effort. The toolchain will consist of tools for modelling, solving, coordinating, designing, simulating, deploying, analysing, monitoring the (components) of the MPC controller.
- A simple 2 d.o.f. model will be used to represent the dynamics. This simple model must match adequately the dynamic characteristics of the lowest resonance. Drive-side and load-side masses will also include the masses of all the transmission elements placed at the drive and load sides respectively of the main flexibility.
- To reduce the online computation burden, different options will be analysed: relaxation of optimization convergence requirements, minimization of constraints applied to the optimization problem, reduction of the prediction horizon, avoiding hybrid and nonlinear models, etc.

- Modelling errors will also be considered. To avoid performance deterioration, modelling errors and unmodeled dynamics (friction, higher order dynamics…) must be compensated. Disturbance modelling will be included in the MPC formulation.

4.2.5.4 Requirements

ID	Requirement	Comments
R27-D4.1	Model Predictive Controller should be executed in COTS controllers	
R28-D4.1	Model Predictive Controller should be executed at a high sampling rate	
R29-D4.1	Model Predictive Control will need feedback of the main d.o.f of the mechatronic system	

Table 6: Model-based predictive control requirements

4.2.6 Control for linear parameter-varying or time-varying systems

4.2.6.1 Introduction

Performance requirements for motion control are getting more stringent, hence, effects of motion systems which were previously not taken into account for control design, cannot be neglected anymore. Two types of systems which are getting more relevant are linear parameter-varying (LPV) systems and time-varying systems.

4.2.6.2 State of the Art

LPV system representation

LPV systems can both be represented in continuous-time and discrete-time (Tóth, 2010). First, consider the continuous-time state-space representation

$$G(\rho):\ \{\begin{matrix} x' = A(\rho)x + B(\rho)u \\ y = C(\rho)x + D(\rho)u \end{matrix} \qquad \text{(Eq. 4.40)}$$

Which can be written in the input-output sense using a transfer function

$$G(s,\rho) = C(\rho)(s\cdot I - A(\rho))^{-1}\cdot B(\rho) + D(\rho) = \frac{N(s,\rho)}{d(s,\rho)} = \frac{\sum_{i=0}^{n} n_i(s)\cdot\theta_{n,i}}{\sum_{i=0}^{n} d_i(s)\cdot\theta_{d,i}} \qquad \text{(Eq. 4.41)}$$

Additionally, the transfer function can be rewritten by expanding into partial fractions as

$$\frac{N(s,\rho)}{d(s,\rho)} = D + \sum_i \frac{\alpha_i}{s+\lambda_i(\rho)} + \sum_j \frac{\beta_j s + \gamma_j}{s^2 + 2\cdot\sigma_j(\rho)s + \sigma_j^2(\rho) + \omega_j^2(\rho)} \qquad \text{(Eq. 4.42)}$$

In discrete-time, the most commonly used structures are state-space or ARX models, where the former is defined as

$$x(k+1) = A(\rho(k))\cdot x(k) + B(\rho(k))\cdot u(k) \qquad \text{(Eq. 4.43)}$$

$$y(k) = C(\rho(k))\cdot x(k) + D(\rho(k))\cdot u(k) \qquad \text{(Eq. 4.44)}$$

And the latter as

$$y(k) + \sum_{i=1}^{n_a} a_i(\rho(k))\cdot y(k-i) = \sum_{j=0}^{n_b} b_j(\rho(k))\cdot u(k-j) + e(k) \qquad \text{(Eq. 4.45)}$$

Data-driven parametric LPV identification

LPV identification has been investigated previously. This roughly started in (Bamieh & Giarre, 2002), where identification of SISO discrete-time rational transfer functions of the form

$$G(\rho[k],q) = \frac{B(\rho[k],q)}{A(\rho[k],q)} = \frac{\sum_{i=k}^{n_b} b_i(\rho[k])q^{-i}}{\sum_{i=0}^{n_a} a_i(\rho[k])q^{-i}} \qquad \text{(Eq. 4.46)}$$

are identified. This can be viewed as an ARX or Output-Error (OE) structure. In (Bamieh & Giarre, 2002), the input-output data is lifted over the identification period and a least squares solution for the parameters a_i and b_i is found. Extensions of this include adding orthonormal basis functions (Tóth et al, 2007) and regularization in the terms of kernels (Golabi et al, 2011; Tóth et al, 2011; Darwish et al, 2018). An overview of data-driven LPV identification techniques can be found in (Bachnas et al, 2014).

Inverse and feedforward for LPV systems

Inversion of LPV systems is used for feedforward control of LPV systems. A general view on inverse system design for LPV systems can be found in (Szabó et al, 2003; Sato, 2008). Historically, LPV (feedforward) control was done using gain-scheduling (Rugh & Shamma, 2000), and more recently a frozen feedforward strategy utilizing Gaussian processes for LPV systems has been investigated in (van Haren et al, 2022). The extension towards true LPV feedforward, i.e., not using a gain-scheduled or frozen feedforward is done in (Szabó et al, 2003; Sato, 2008; Rugh & Shamma, 2000; van Haren et al, 2022; Butcher & Karimi 2010), and in the iterative sense in (Butcher & Karimi 2010, de Rozario et al, 2018; de Rozario et al, 2017).

4.2.6.3 Progress beyond the SoA

The major contribution to the progress beyond the SoA is considered to be the following:

Data-driven grey-box methods for LPV feedforward

The proposed approach is to pursue a grey-box feedforward strategy for LPV systems. The considered set of (quasi-)LPV systems are

$$G(\rho): \begin{aligned} x' &= A(\rho)\cdot x + B(\rho)\cdot u \\ y &= C(\rho)\cdot x + D(\rho)\cdot u \end{aligned} \qquad \text{(Eq. 4.47)}$$

With scheduling variable $\rho(t)$, which can contain both outputs and derivatives of the output of the system and exogenous known scheduling signals. The system is rewritten to transfer function representation as

$$G(s,\rho) = C(\rho)(s\cdot I - A(\rho))^{-1} B(\rho) + D(\rho) = \frac{N(s,\rho)}{d(s,\rho)} = \frac{\sum_{¿}^{n} n_i(s)\cdot\theta_{n,i}}{\sum_{¿}^{n} d_i(s)\cdot\theta_{d,i}} \qquad \text{(Eq. 4.48)}$$

For these kinds of systems, in the LTI sense, typically polynomial feedforward is applied. For LPV systems, here the parameters are also chosen LPV as

$$u = \sum_{i=1}^{n_\theta} \psi_i \theta_i(p) \qquad \text{(Eq. 4.49)}$$

Where $p(t)$ is the scheduling variable, which for feedforward can contain the reference, it's derivatives or exogenous variable, e.g., $p=(r, r', \varphi)$. This means that for feedforward it is assumed that the output tracks the reference arbitrarily well. The basis functions ψ_i are determined based on first principle modelling of the LPV system. The parameters θ_i are learned using input-output data of a (stable quasi-) LPV system. The modelling of the parameters as a function of the scheduling signal will be done by utilizing a Gaussian process. This is envisioned since Gaussian processes are non-parametric, which is ideal for cases where the underlying structure is not known beforehand, which is generally the case for LPV feedforward parameters. The parameters are modelled using the covariance function (Rasmussen, 2003), i.e.,

$$cov(\theta_i(p), \theta_i(p')) = E(\theta_i(p)\theta_i(p')) = K(p, p') \tag{Eq. 4.50}$$

In combination with input-output data, the parameters can be jointly distributed as a Gaussian Process as

$$\begin{bmatrix} u(t) \\ \theta_i*(p(.)) \end{bmatrix} \sim N\left(0, \begin{bmatrix} k_u(t,t)+\sigma^2 \cdot I & K_i*(t,p(.)) \\ K_i*(t,p(.)) & K_i(p(.),p(.)) \end{bmatrix}\right) \tag{Eq. 4.51}$$

Where the kernels k_u and K_i^* are determined using the kernel function K_i and the basis functions ψ_i . The resulting algorithm can learn LPV basis function feedforward using input-output data in the Bayesian sense.

Data-driven inverse LPV-FIR identification using machine learning techniques.

In addition to applying grey-box methods for feedforward for LPV systems, LPV Finite-Impulse-Response (FIR) feedforward is envisaged. This is realized via the following parameterization

$$u(t) = \sum_{i=-\tau_{ac}}^{\tau_c} \theta_i(p(t)) \cdot r(t-i) \tag{Eq. 4.52}$$

With the amount of causal FIR parameters τ_c and amount of non-causal FIR parameters τ_{ac}. Again, input-output data of an LPV system is used to learn the parameters θ_i as a function of the scheduling sequence. Similarly, as the previous section, Gaussian processes are employed again.

4.2.6.4 Requirements

ID	Requirement	Comments
R30-D4.1	For LPV control to be applicable, the scheduling variable must be measured	
R31-D4.1	Given an LPV system, the structure of the LPV controller must be specified by the user	

Table 7: Linear parameter-varying systems requirements

4.2.7 MIMO Control

4.2.7.1 Introduction

Some mechanical systems need to fulfil more than one control objective using a single actuator. The problem of the classical control approach lies in that several objectives must be accomplished with independent Single-Input Single-Output (SISO) control structures, which can lead to opposite actuation commands, even destabilizing the system. Modern Multiple-Input Multiple-Output (MIMO) control methods are more appropriate to design control strategies that must meet several control objectives.

Pitch actuation of Wind Turbines (WT) in Floating Offshore (FO) are a clear example of this issue, as undesired cross effects appear when applying SISO control of the pitch control, Active Tower Damping (ATD) and Active Platform Damping (APD). Hence, this application is used as a reference to present the technology. In any case, the approach can be later on applied in other systems, like cranes.

A wind turbine control has usually several layers, being the two most important layers the supervisory control and the operational control. The supervisory control guarantees a safe operation of the machine, governing the turbine operation at different machine states and the transitions between these states. Software and hardware safety systems, shut down processes, detection and management of faulty conditions are supervisory control tasks.

The operational control is concerned with the power extraction and load minimization on the turbine components. Three actuations are available for this: blade pitch angle, generator torque and yaw angle. Due to the low dynamics, yawing actuation is usually performed with basic controllers, being thus not interesting for advanced control designs. Pitch and yaw actuations exhibit higher dynamics, and it is expected that higher benefits will be achieved using advanced MIMO controllers.

The classical approach for the pitch and yaw control is based on the conventional PID controllers. There are two main operation regions:

- Region II: in low-mid winds, the goal is to maximize the power extraction from the wind. For this, turbine rotation velocity is modified to be kept as close as possible to the Optimum operation point (maximum Cp).
- Region III. at mid-high winds, blade angles are regulated to maintain the nominal rotation and power production values.

The classical PID based control structures to achieve these objectives are:

1. A torque control, usually PI, that regulates the WT rotation speed in Region II. This loop is saturated in Region III.
2. A pitch control, usually PID, that regulates the WT rotation speed in Region III. In Region II, the pitch angle is kept constant at the optimum value.

Apart from these main two loops, the operational control also offers additional functionalities, usually with the objective of reduce load components that increase the fatigue of the machine components. Examples of these functionalities are:

1. DTD (Drive Train Damping), to damp the torsion mode of the drive train.
2. ATD (Active Tower Damping), to damp the tower 1st fore-aft bending resonance.
3. IPC (Individual Pitch Control), to reduce the 1P component in blade base loads.
4. In the case of FOWT (Floating Offshore), APD (Active Platform Damping), to damp the pitch oscillation of the floating platform.
5. Etc.

4.2.7.2 State of the art

The WT advanced controllers usually are based on Linear Time Invariant (LTI) linear state-space models. Linear state-space models are used to describe the dynamics of the WT system in a reduced order model. Next equations represent a state-space representation of a plant:

$$\begin{aligned} \delta x' &= A\cdot\delta x + B\cdot\delta u \\ \delta y &= C\cdot\delta x + D\cdot\delta u \end{aligned} \qquad \text{(Eq. 4.53)}$$

where δx is the state vector, δu the input vector and δy the available measurements. The term δ means small deviations around the linearized operation point.

One usual way to calculate the matrices involved in this state-space representation is by means of the dynamic equations of the WT components: 1 d.o.f. model of the drive train, simplified model of the tower capturing the 1st fore-aft bending resonance, etc.

Another way is using aeroelastic codes such FAST or Bladed. These codes allow the linearization of highly non-linear WT dynamics in the desired operation points. These codes also allow switching on the desired d.o.f. of the model in order to obtain a suitable reduced-order model for control design.

The most common MIMO control structures used in WT operation are briefly described below (detailed reviews are available in Kamran et al, 2021; Yuan & Jiong, 2017; Wright et al, 2019):

LQR & DAC structures

Linear Quadratic Regulators (LQR) supplemented with Disturbance Accommodation Control (DAC) is a very popular approach in WT operation control. This control structure allows to consider different objectives in a unified framework: rotor speed regulation and minimization of loads on the main WT components can be achieved at the same time.

The DAC strategy "accommodates" the wind disturbance inside the system plant model augmenting the plant state with the disturbance estimate. In this way, wind speed variations from the value used in the linearization point are properly considered in the controller.

For the controller action, the static gain of the plant state feedback can be obtained applying the LQR method. The LQR method is based on minimizing some cost function calculated weighting the plant estates and actuation variables.

In some cases, a Linear Quadratic Gaussian (LQG) controller is used instead of the LQR solution. The difference is that the LQG approach assumes that the states and the measurements are driven by zero-mean Gaussian white noise stochastic processes. In this case, the optimal gain is given by the Kalman filter.

As mentioned above, the LQR&DAC approach is widely used in WT operation control. Examples include regulation of rotor speed while damping the drive-train torsion resonance and the tower 1st bending moments, Individual Blade Pitch Control (IBPC) for reduction of blade base periodic 1P/2P loads, etc.

LQR&DAC control structures have been widely used in wind turbine control. As example, NREL has investigated the use of this control structure through the analysis of the LQR, DAC and combined LQR&DAC solutions. In (Stol & Fingersh, 2003; Wright et al, 2011), an LQR&DAC controller is designed for the two-bladed teetering hub upwind machine of 600-kW CART2 wind turbine. The design objective is to regulate turbine speed and enhance damping in several low-damped flexible modes of the turbine. In (Wang et al, 2016), two independent pitch controllers (IPCs) based on the disturbance accommodating control (DAC) algorithm are designed for the three-bladed CART3 wind turbine to mitigate blade root flap-wise bending loads in above-rated wind speed. Finally, in (Sinner & Pao, 2020), DAC is implemented for two cases: estimating the disturbance (upcoming wind) from feedback information and considering disturbance measurements produced by Lidar.

Robust Control, H2/H∞

Obtaining an exact WT model is a hazardous task caused by the highly nonlinear aero-elasto-dynamics, actuator saturations, higher order dynamics, etc. Robust Control allows to account for the model uncertainties as well as the disturbance uncertainties in the control design.

The Robust Control solution consists of stabilizing robustly a family of plants with uncertainty. H-infinity optimization methods based on minimizing H_2/H_∞ norms of the weighted closed-loop transfer functions of the system (sensitivity function, complementary sensitivity function, etc.) are applied for this.

Examples of a typical application of Robust Control in WTs are the damping of the drive-train torsion mode and the 1st fore-aft and side-to-side bending modes of the tower. One example of robust control application to wind turbine control can be seen in (Mirzaei et al, 2012).

MPC structure

Model Predictive Control (MPC) is an advanced control strategy that we already discussed in section 4.2.5. Based on the current system states vector and a prediction model of the system it estimates the input actions to the system by minimizing a cost function over a time horizon in the future. The control action for all the time horizon is calculated, but only the first control sequence is applied to the real system. The MPC formulation allows to apply constraints both in the inputs and outputs, as well as to deal with nonlinear and hybrid systems.

In the WT operation control, MPC is usually combined with LIDAR sensors. When designing an MPC solution and besides the prediction model, information of future disturbances (wind speed) is needed. In the past, wind disturbance used to be estimated using Kalman filters. Nowadays, a wind speed sensor LIDAR provides valuable information on the upcoming wind disturbance on the WT. Examples of application of MPC to wind turbine control are (Schlipf et al, 2014) and (Korber & King, 2010).

Other strategies

Other control solutions that have been tested in WT operation control are:

1. *Adaptive Control*. An adaptive control technique faces the uncertainties in the model and in the operating conditions. As one example, in the Model Reference Adaptive Control (DMRAC) the plant output tries to follow the output of a reference model (an example of application to wind turbine control can be seen in (Frost et al, 2009) and (Magar et al, 2016)).
2. *Linear Parameter Varying (LPV) control*. In (Eq. 4.39), a LTI system is described, i.e., the matrices of the model equations are constant, calculated for just one operation point. Control actions are designed for each operation point based on this LTI representation of the plant, and then scheduled to apply in all the operation points. A different approach is to consider the WT nonlinear behaviour in the equations. In this way, the nonlinear behaviour of the plant is accounted for in the model, and the controller is designed once for all the operation points. This is the LPV control (example of application to wind turbine control: Ossman et al, 2017).
3. Others: *Input/Output Feedback Linearization (IOFL) control*, *Sliding Mode Control (SMC)*…

4.2.7.3 Progress beyond SoA

There are many papers that deal with the use of LQR, DAC and combined LQR&DAC solutions to control wind turbines, but in most of the cases they are based on very simplified wind turbine controllers. The purpose of these developments is to show the validity of the control concept, in this case the validity of the LQR&DAC concept in its various variants. Unfortunately, the resulting control structures are far from their application in a real wind turbine controller.

The progress pursued with this technology is to reduce this gap from academy to industrial application. To this end:

1. A real problematic will be addressed: the stability of the control in the region III. It is well known the negative damping problem in FOWT turbines, caused by the interaction of the pitch loop (that controls the rotation speed of the rotor) and the platform pitching resonance. Some MIMO control strategy is needed to cope with this problem.
2. The MIMO solution will not be built from scratch. Taking in mid the industrial application, the baseline control will be a state-of-the-art industrial control. This industrial control, based on multiple SISO elements, will be kept as the baseline controller, and the LQR&DAC controller structure will be added only where it is necessary: speed regulation in region III, stabilizing the tower 1st fore-aft and platform pitching oscillations.

3. Taking in mind the industrial application of the final control solution, the sensing needed by the MIMO structure will be limited as closely as possible to the usually available measurements in a wind turbine.

In short, the final control solution will be ready to implement in a real turbine and easily implementable.

For the controller development and simulation environment, a Floating Offshore Wind Turbine (FOWT) model will be used. The WT model used will be the DTU 10MW Wind Turbine, first presented in the INNWIND.EU European Project. This turbine is designed for an IEC Class 1 A wind climate. This turbine is coupled to the Nautilus floating platform public model. The Baseline control of the plant is developed in the commercial software Matlab/Simulink. The new MIMO LQR & DAC solution will also be implemented in Matlab/Simulink.

The first step is to calculate the state-space model that represents the plant described in eq 4.54.

$$\begin{aligned}\delta x' &= A\cdot\delta x + B\cdot\delta u \\ \delta y &= C\cdot\delta x + D\cdot\delta u\end{aligned} \quad \text{(Eq. 4.54)}$$

where δx is the state vector, δu the input vector and δy the available measurements. The term δ means small deviations around the linearized operation point.

This model is a Linear Time Invariant (LTI) representation of the system obtained at one linearization point. The FAST dynamic code will be used to develop such linear models. Depending on the design structure and control objectives, some d.o.f. of the plant will be switched on and other switched off.

The idea is to design the MIMO controller for the operation Region III, acting only on the pitch loop (The Torque controller will be retained from the Baseline controller). The multiple objectives of the controller will be to regulate the rotor speed, and to add damping to the 1st tower fore-aft bending moment and to the platform pitch oscillation. For this, the roadmap is to proceed step by step in the development of the LQR & DAC controller: 1st regulate only the rotor speed, 2nd add damping to the tower bending resonance, and if possible, 3rd add damping to the platform pitch mode.

In the following lines, a deeper insight in the design of the LQR & DAC MIMO control is provided.

A typical control action performed in state-space representation is the feedback of the system state:

$$\delta u = G\cdot\delta x \quad \text{(Eq. 4.55)}$$

where *G* is the gain matrix.

This gain matrix can be calculated placing the closed-loop poles in the complex plane in the desired location. The closed-loop poles are determined by the eigenvalues of the closed-loop system: $eig(A+B\cdot G)$. *Matlab Control Design Toolbox* can be used for this.

Another option to calculate the gain matrix *G* is applying the LQR approach. In this approach, a quadratic cost function for the regulation problem at an operating point is defined:

$$J=\int_0^\infty \left(\delta x^T\cdot Q\cdot\delta x+\delta u^T\cdot R\cdot\delta u\right)dt \quad \text{(Eq. 4.56)}$$

Where *Q* and *R* are nonnegative and symmetric matrices of weights for the state and input vectors respectively. The optimal feedback gain matrix is calculated solving this LQR problem. The control law is just the one that optimizes this cost function. *Matlab Control Design Toolbox* can also be used to solve this LQR problem.

But usually, as in the WT case, the measurements of all the states are not available: the number of sensors is limited. A state estimator is needed. System states can be measured using next model:

$$\begin{aligned}\delta\dot{\hat{x}} &= A\cdot\delta\hat{x}+B\cdot\delta u+K\left(\delta y-\delta\hat{y}\right)\\ \delta\hat{y} &= C\cdot\delta\hat{x}+D\cdot\delta u\end{aligned} \qquad \text{(Eq. 4.57)}$$

where ^ indicates estimated value, and *K* is the weight applied to the estimation error.

The estimator dynamics can be tailored placing the eigenvalues of the closed-loop system: $eig\left(A-K\cdot C\right)$ in the desired location in the stability complex plane.

The control action is performed with the estimated state vector:

$$\delta u=G\cdot\delta\hat{x} \qquad \text{(Eq. 4.58)}$$

In the controller design described until now, the effect of the disturbances (wind and waves) is not accounted for. Adding these disturbances, the state-space representation of the system becomes:

$$\begin{aligned}\delta x' &= A\cdot\delta x+B\cdot\delta u+B_d\cdot\delta u_d\\ \delta y &= C\cdot\delta x+D\cdot\delta u+D_d\cdot\delta u_d\end{aligned} \qquad \text{(Eq. 4.59)}$$

where δx is the state vector, δu the input vector, δu_d the disturbance input (wind or wave) and δy the available measurements.

A state-space representation of the disturbance model is needed. The disturbance model can be expressed in the state-space form as:

$$\begin{aligned}\delta z_d' &= F\cdot\delta z_d\\ \delta u_d &= H\cdot\delta z_d\end{aligned} \qquad \text{(Eq. 4.60)}$$

where δz_d is the disturbance state vector and δu_d the disturbance input. Matrices *F*, *H* depend on the model used to represent the disturbance: step wind disturbance, shear disturbance, etc.

Now, this augmented state-space representation allows to include the effect of the disturbances in the control design, giving rise to the LQR & DAC control.

In the LQR & DAC approach, the control action is given by

$$\delta u=G\cdot\delta x+G_d\cdot\delta z_d \qquad \text{(Eq. 4.61)}$$

Where G_d is the gain affecting the disturbance state vector.

While the state vector gain *G* is calculated placing the closed-loop poles in the desired location in the complex plane (or solving the LQR problem), the disturbance state vector gain G_d is calculated to mitigate the effects of the disturbance.

Disturbances are also unmeasured: the disturbance state must also be estimated based on the available measurements on the WT. This is accomplished by augmenting the state-vector estimator described above. Now, the plant estimator is given by:

$$\begin{aligned}\delta\hat{x}' &= A\cdot\delta\hat{x}+B\cdot\delta u+B_d\cdot\delta\widehat{u_d}+K\left(\delta y-\delta\hat{y}\right)\\ \delta\hat{y} &= C\,\delta\hat{x}+D\,\delta u+D_d\,\delta\widehat{u_d}\end{aligned} \qquad \text{(Eq. 4.62)}$$

Where ^ indicates estimated value, and *K* is weight applied to the estimation error.

Similarly, the disturbance estimator is defined by the next model:

$$\delta \hat{z}_d' = F \cdot \delta \hat{z}_d + K_d (\delta y - \delta \hat{y}) \\ \delta \hat{u}_d = H \cdot \delta \hat{z}_d \qquad \text{(Eq. 4.63)}$$

where K_d is weight applied to the estimation error.

Finally, the control action is calculated from the estimated state and disturbance vectors:

$$\delta u = G \cdot \delta \hat{x} + G_d \cdot \delta \hat{z}_d \qquad \text{(Eq. 4.64)}$$

Figure 21 shows the structure of the LQR & DAC controller:

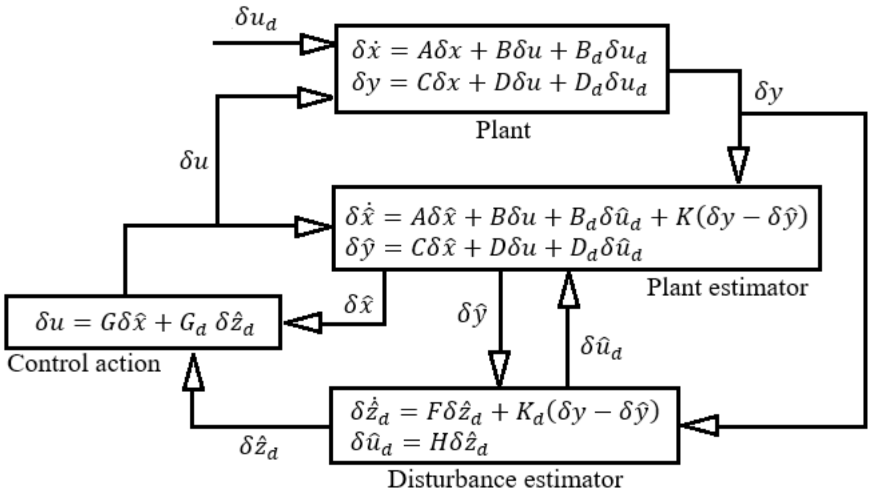

δu_d
$\delta \dot{x} = A\delta x + B\delta u + B_d \delta u_d$
$\delta y = C\delta x + D\delta u + D_d \delta u_d$
Plant
δy
δu
$\delta \dot{\hat{x}} = A\delta \hat{x} + B\delta u + B_d \delta \hat{u}_d + K(\delta y - \delta \hat{y})$
$\delta \hat{y} = C\delta \hat{x} + D\delta u + D_d \delta \hat{u}_d$
Plant estimator
$\delta u = G\delta \hat{x} + G_d\, \delta \hat{z}_d$
Control action
$\delta \hat{x}$
$\delta \hat{y}$
$\delta \hat{u}_d$
$\delta \dot{\hat{z}}_d = F\delta \hat{z}_d + K_d(\delta y - \delta \hat{y})$
$\delta \hat{u}_d = H\delta \hat{z}_d$
$\delta \hat{z}_d$
Disturbance estimator

Figure 21: LQR&DAC controller structure.

4.2.7.4 Requirements

No specific requirement is foreseen for this technology

4.2.8 Control and identification of multi-rate systems

4.2.8.1 Introduction

Many mechatronic systems contain multiple sampling rates in the same control loop. Consider for example vision-in-the-loop systems, where typically cameras are sampled at a much lower rate than the encoders. In that case, visual information from the cameras must be processed into a higher rate by interpolating the received data. Sometimes, sensor data must be decimated in order to reduce the rate at which information is received. In many cases, both interpolation and decimation must be combined in order to accurately adjust the rate of all incoming data. Multi-rate control and identification is the technique that tries to deal with these kinds of systems.

4.2.8.2 State of the art

- Multi-rate control for the use in sampled-data control

  Sample-data control is a control system where a continuous-time plant is managed by a digital controller, see Figure 22.

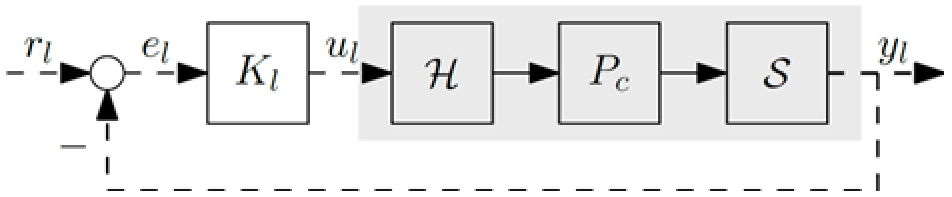


Figure 22: Sampled-data control structure with continuous-time plant, hold and sample operations.

An overview of sampled-data systems is given in (Chen & Francis, 2012). Typically, sampled-data control is approximated by sampling the continuous-time plant in a sampling rate much higher than the controller, which is referred to as multi-rate control (Glasson, 1983), see also Figure 23.

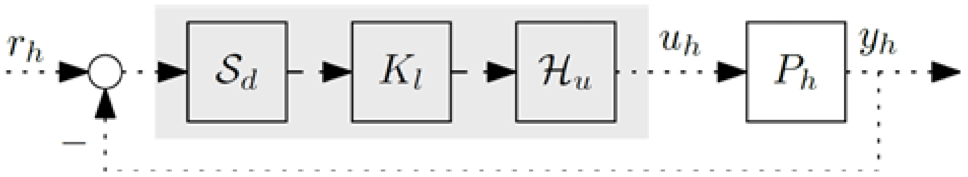


Figure 23: Approximation of sampled-data control by sampling the plant arbitrarily high.

In (Oomen et al, 2007), feedback control synthesis and performance evaluation in the frequency-domain for sampled-data systems is discussed. Additionally, inter-sample performance is of great interest in sampled-data control, which is investigated in (Bamieh, 2003; Oomen et al, 2009).

- Multi-rate feedforward control

Multi-rate control is also used to improve tracking performance by sampling a feedforward controller at a different rate, see (Fujimoto et al, 2001; Mae et al, 2020; Ohnishi et al, 2021). When using this approach, multi-rate techniques are used to overcome issues like sampling zeros when inverting systems, limited computation time of the controller and taking derivatives of the reference used in linear feedforward techniques.

Additionally, feedforward control for systems with different sampling rates is investigated in (van Zundert et al, 2018). This can generally be the case when systems operate at different sampling rates. Consider for example a dual stage system, or a system with additional sensors that cannot be sampled as fast as the encoders. In (Zundert et al, 2018), high-rate signals from a high-rate control loop are used in the feedforward controller of a low-rate control loop as shown in Figure 24.

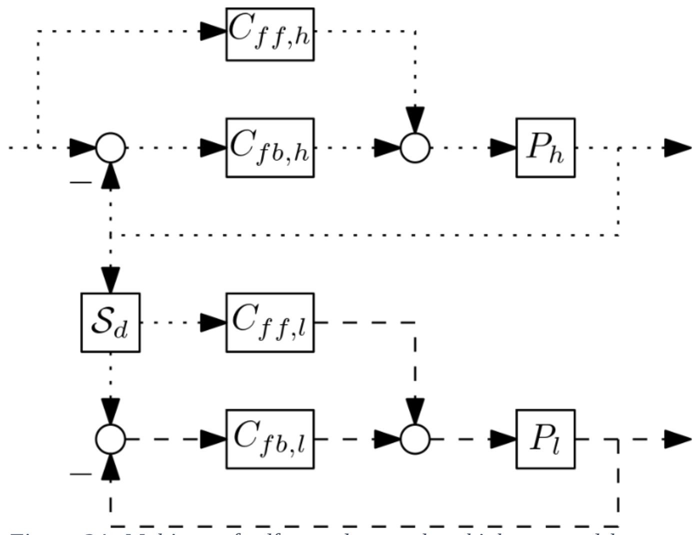


Figure 24: Multi-rate feedforward control: a high-rate and low-rate controller communicating in order to process high-rate signals with a low-rate control-loop.

4.2.8.3 Progress beyond SoA

Two major contributions are envisaged to progress beyond the SoA. These can be summarized as

- Multi-rate system identification.

  Identification of systems operating in multi-rate feedback in the frequency-domain, both in terms of transfer functions, but also in terms of multi-rate frequency-domain representations, such as the performance frequency gain.

- Multi-rate inferential identification and control.

  Since many systems have additional sensors that measure the inferential performance variable, which are sampled at a different rate (i.e., consider a camera (Vision-in-the-Loop) or accelerometer), inferential identification and control can be necessary.

4.2.8.4 Requirements

No specific requirement is foreseen for this technology. As mentioned, the proposed approach uses WT control as a clear application where MIMO control benefits are clear. The same approach can be adapte to equivalent applications, like crane systems.

4.3 BB5 requirements

The next table summarizes the requirements for Building Block 5, integrating the ones defined in D2.3 with the new ones defined for each tecnology in the different tables available insection 4 of this document.

ID	**Requirement**	**Priority**	**Verify**	**Comments**	**Tasks**
Interfaces and connectivity					
R124-D2.3	AI-based algorithms should be compatible with commercially available TPU's	C	I	Allows for application in embedded solutions	T4.1

R125-D2.3-L2	BB5 can be connected to Matlab/Simulink (Layer 2 to Layer 3) for configuration, etc.	C	A		T4.1
R1-D4.1	Gain or phase stabilization of dominant resonance modes must overlap with target closed-loop bandwidth				
R15-D4.1	Control algorithms for switched reluctance motors should consider that reversing the direction of current does not reverse the direction of torque				
R27-D4.1	Model Predictive Controller should be executed in COTS controllers				
Maintainability (modularity, analysability, testability)					
R126-D2.3-L2	All smart control algorithms shall have a clear documentation that explains input, outputs, description, and parameter settings	M	I		T4.1
R127-D2.3-L2	Smart control algorithms and models shall be tested in simulation	M	T	Validated in WP6	T4.1, T4.2
R128-D2.3-L2	Control functionalities should be able to be tested by automatic means and accordingly documented (requirement traceability)	C	T		T4.1
R11-D4.1	Systematic design procedures allow automatic or semi-automatic synthesis and parameterization of the control structures without requiring a highly skilled operator				
Performance					
R6-D4.1	Sufficient performance improvement on vibration control systems must be had compared to conventional control schemes				
R9-D4.1	Computation burdens should be compatible with available computation power on the drives				
R12-D4.1	Sufficient performance improvement on repetitive control systems must be had compared to conventional control schemes				
R20-D4.1	Iterative techniques must be able to achieve good performance even in the presence of trial-variant disturbances				
R24-D4.1	Friction compensation must be executed at a high sampling frequency			Minimum sampling of 1 kHz	
R28-D4.1	Model Predictive Controller should be executed at a high sampling rate			Minimum sampling of 1 kHz	

Power efficiency					
R14-D4.1	Solutions to the over-parametrized commutation problem should penalize power consumption				
Compatibility (interoperability, co-existence)					
R129-D2.3-L2-L3	The library in BB5 shall be compatible with BB4	M	D		T4.1, T4.3, T4.4, T4.6
R130-D2.3-L2-L3	All models should be compatible with Matlab/Simulink	S	D		T4.2, T4.3, T4.4
R131-D2.3-L2-L4	Modelling should be compatible with code generation tools	M	D		T4.1, T4.2, T4.3, T4.4
R132-D2.3	BB5 shall be compatible with smart control blocks developed in I-MECH	S	D		T4.2, T4.3, T4.4
R24-D4.1	Friction compensation must be added to the control action of the applied controller (current control loop input)				
Usability (operability)					
R133-D2.3	BB5 shall be able to be executed in real-time on provided execution platforms (e.g. via BB1, BB4)	M	T		T4.1
R5-D4.1	Systematic design procedures of vibration control systems must allow automatic or semi-automatic synthesis and parameterization of the control structures without requiring a highly skilled operator				
R13-D4.1	The strategy used for repetitive control should allow for flexibility towards different tasks				
R22-D4.1	Techniques in machine learning, applied to control, must have interpretable hyper-parameters such that the user knows what to expect when changing the values				
R31-D4.1	Given an LPV system, the structure of the LPV controller must be able to be specified by the user				
Reliability (fault tolerance, availability)					
R134-D2.3-L2-L4	Control algorithms will have self-diagnosis functions	S	I		T4.1
Portability (adaptability, replaceability)					

R135-D2.3	BB5 shall offer customizability such that non-standard tasks (i.e., tasks which are typically performed in research) can be performed. Examples include flexibility in allowed controller structures and reference/feedforward signals	S	T		T4.1
R23-D4.1	Real-time algorithms must be sufficiently resource-efficient to be applicable to hardware that is standard in industry				
Safety					
R136-D2.3-L2-L4	Smart control algorithms of collaborative robots (Cobots) need to be compliant with safety standards	M	A		T4.1, T4.2, T4.3, T4.4
R16-D4.1	Compliant applications must use torque-based control schemes				
R18-D4.1	Stability of the system in both training and tests must be demonstrable, i.e, it must be safe				
Digital twin					
R137-D2.3-L2-L4	Data-driven models shall be compared to analytical models and/or validated real robots	S	A		T4.1, T4.2, T4.3, T4.4
R8-D4.1	Load-side motion control requires the usage of additional sensors				
R17-D4.1	Accurate dynamic models must be automatically captured				
R18-D4.1	Smart control algorithms must integrate models and adaptation mechanisms within the control loop				
R19-D4.1	Control algorithms must be robust to modelling errors				
R26-D4.1	Although characterisation of friction is not required for friction compensation algorithms, a two-mass model of the controlled system is required				
R29-D4.1	Model Predictive Control will need feedback of the main d.o.f of the mechatronic system				
R30-D4.1	For LPV control to be applicable, the scheduling variable must be measured				

Table 8: BB5 requirements

5 Path planning, trajectory generation, obstacle avoidance, navigation (BB10, Task 4.5)

5.1 Introduction

Path planning, i.e., the search for sequence of points (states) that represent the feasible path that connect the start pose with the goal pose, is a task usually associated with Layer 3 (system behaviour), but for a machine to properly work on an unstructured environment it is essential to adapt the given path on real time, thus relying on the control layer (Layer 2).

This adaptation in real-time to the environment relies in 2 components: the analysis of the environment and the decision making. The first component tries to find unexpected elements not accounted for in the initial planning phase, such as obstacles. The second component works on how to adapt the initial navigation plan to the new information, and generate a new plan if necessary.

In this section of the deliverable, we will focus on these 2 components of the control Layer and the requirements related to them.

5.2 Control functionalities

5.2.1 Multi-sensory data input with sensor-fusion for extended environment detection

5.2.1.1 Introduction

Sometimes modelling the working environment is needed for enabling path planning that is safe for the operators, the environment and the machine itself. In the simplest approach, the environment model can be formed by measuring diverse distances on the environment, such as the diameter or length of a tunnel. In some cases, such as mining environments, this is not enough, and instead manually measuring points in the tunnel surfaces with the help of some pointer system is needed. This approach requires possibly too time-consuming manual operations after the machine is driven and settled to the work site and before the actual work can begin. Operation that is more automatic requires the use of some sensor system, which automatically measures a point cloud from the tunnel surfaces.

In other cases, even if the environment is well known, moving obstacles, such as a human or animal, can interfere on the trajectory of a machine. In these cases, in which the environment is not static, but may include some dynamic changes, sensors must be included into the vehicle so that these obstacles can be detected and avoided.

5.2.1.2 State of the art

Radar processing

Automotive radar is one of the main state-of-the-art technologies for collision avoidance. It is a sensor system based on frequency-modulated continuous-wave or FMCW radar. It is less expensive than pulse-Doppler radar and can be performed on low-cost FPGAs. This also influences and accelerates the development of radar chips. The trend is moving towards imaging radar processes in order to be able to guarantee the most precise possible detection of the surroundings in all weather and light conditions.

The basis for the IMOCO4.E project is a 77 GHz radar sensor for industrial robotics applications developed in the RoKoRa project (Abdelawwad et al, 2021). This sensor can be seen in figure 25. In the KLARA project (Hirsch et al, 2020), the radar backend was extended by a signal processor that can be used to execute AI algorithms.

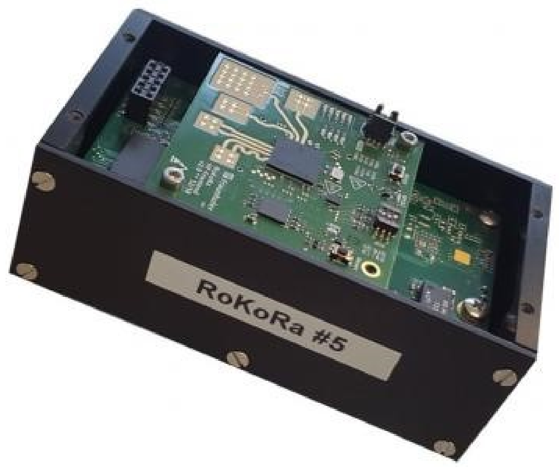


Figure 25: Radar sensor from RoKoRa project

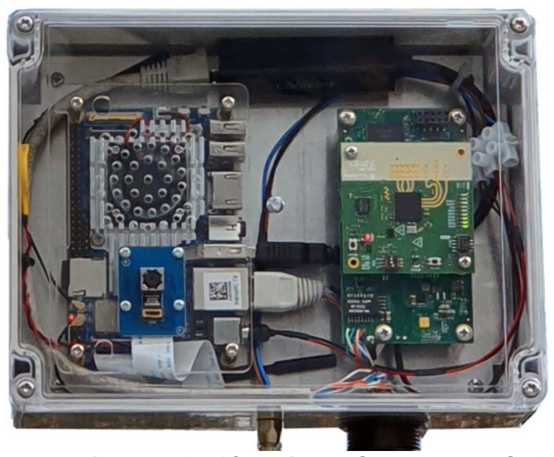


Figure 26: Sensor unit with 77 GHz radar, camera and mini-PC from VIDETEC project

For the work of sensor nodes, Mini-PC that can be individually adapted to the application by programming are commonly used. The VIDETEC project (Kulke et al, 2022) chosed the Nano Pc-T4 as a small high-performance Mini-PC that could work as a sensor unit. This unit, containing a miniature camera and a 77 GHz RoKoRa radar, can be seen in figure 26. Video and radar data are recorded by the Nano-PC and stored synchronously in a common data format. The data can be stored locally and read out via a data interface (Ethernet or USB).

Camera based detection and visual servoing

Visual servoing is a technique which uses feedback information extracted from a vision sensor. It is widely used in the field of boom mining as it guides the boom and helps react to changes in the environment. The visual sensing modality can be either 2D imaging or 3D with e.g., time-of-flight cameras, with the sensors mounted in the end-effector (hand-in-eye configuration) or fixed in the scene (eye-to-hand) and calibrated with the booms frame (Corke & Khatib, 2011). Key aspect of this approach is the selection of the used visual features, on which the process explicitly relies on. This is mainly a computer vision problem, in which accurate and robust feature and object detection and tracking algorithms are among the fundamental research topics. Convolutional neural network (CNN) based systems have surpassed more traditional computer vision algorithms and represent state of the art (Jiao et al, 2019; Liu et al, 2019). Furthermore, the visual control schemes can be divided into image, position and hybrid approaches. Image-based control schemes fully rely on features derived directly from 2D images, while position-based approach necessitates estimating the target object pose in full 3D space. Hybrid approaches represent state of the art and allow, for example, determination of a boom end-effector translation movements in the 2D image space and rotations in full 3D space (Chaumette & Malis, 2000; Malis & Chaumette, 1999).

5.2.1.3 Progress beyond the SoA

Radar processing

In IMOCO4.E, the frontend of the radar sensor is currently being redesigned. In particular, a new antenna is being developed that supports Multi-Input Multi-Output (MIMO) data processing. In MIMO, the different transmitting and receiving antennas are arranged in such a way that they function as a virtual antenna array, thus increasing the number of radar channels. A configuration of 3 transmitters and 4 receivers becomes a virtual system of 3x4=12 channels. This allows for a significantly improved detection of the environment. This radar should detect obstacles and the possible driving paths.

The sensor unit will also be further developed and programmed so that more radar modules can also be connected.

Camera based detection and visual servoing in boom control

Camera based detection of the objects is a challenging problem, as the environment and visual features of the mining environment can change dramatically. Previously mainly LIDAR 3D scanning-based approaches have been proposed in the literature (Bonchis et al, 2014). However, these suffer from poor detection power when the object axis is not aligned with the scanner as demonstrated in (Bonchis et al, 2014), and from use of relatively fragile sensors in harsh environments. For this reason, 2D-camera based approaches with state-of-the-art open-source object detection systems using CNN are utilized here. Such object detectors far surpass more traditional computer vision systems in both detection accuracy and generalization capabilities, which are crucial to robustly locate the objects in different environmental conditions. However, CNN approaches are hindered by their need of large training datasets and high computational needs. The latter factor has specially delayed the adaptation of these technologies in industrial environments and robot control, in which real-time processing with relatively modest computational hardware is needed. Nevertheless, modern optimization techniques (e.g., Augasta & Kathirvalavakumar, 2013; Zhuang et al, 2018) show promise to significantly reduce the processing needs especially when combined with hardware accelerated platforms.

The progress beyond state-of-the-art in visual servoing is based on the abovementioned use of latest CNN approach. This advancement will allow reliable use of visual servoing outside of laboratory conditions, with visual sensors that are robust in harsh conditions. In addition, sensor fusion with distant sensors will be used to complement the 2D image data and further aid the servoing task at a low cost. This also requires development of a new hybrid control law for the combination of 2D image features and 1D sensor data.

5.2.1.4 Requirements

ID	Requirement	Comments
R32-D4.1	Object detection must work in harsh environments with varying lightning conditions and dark surfaces	
R33-D4.1	The object detection algorithm must provide accurate position information for the boom control in real time	

Table 9: Multi-sensory data input and sensor-fusion requirements

5.2.2 Real-time decision making in path planning

5.2.2.1 Introduction

Based on a map of the environment, path planning solves the problem of calculating trajectory between a starting and a destination point. Several different path planning methods like “topology bound navigation”, “free navigation in maps” or “reactive navigation” exists. In the following a short overview over relevant path planning algorithms for IMOCO will be presented.

5.2.2.2 State of the art

Global path planning

A path or trajectory planned in the global planning stage incorporates the information and constraints known at planning time. This plan usually leaves out details that either are unknown beforehand or that would make the planning stage too complex. Some of the simpler cases can be solved using well known algorithms, such as Dijkstra’s algorithm (Dijkstra, 1959) or A* algorithm (Hart et al, 1968). In this document we will focus on more complex cases, such as the coordination and path planning of Automated Guided Vehicles (AGV).

Nowadays, companies strive to archive an increase in effectiveness and flexibility. In order to archive this, the intralogistics are organized with a heterogeneous fleet of AGVs. While AGVs increase the flexibility, they also introduce another problem: a fleet of AGVs needs to be coordinated in order to avoid deadlocks between them. This restricts the global path planning stage, as each AGV's path must be compatible with the rest. This is usually done through a fleet management system (FMS) which aims to coordinate the trajectories of a heterogeneous fleet of AGVs without deadlocks. For the coordination a time window-based algorithm is used to ensure collision-free path on a graph-based topology. Furthermore, for the heterogeneous AGV fleet a uniform communication standard will be used. Even in a controlled environment, as common in intralogistics, dynamic events like path blocking pallets on the ground can occur which require flexible handling. For this purpose, the FMS should enable dynamic re-planning of the coordinated trajectory.

Planned trajectories are communicated from the FMS to the individual robots which execute them under a limited local autonomy (local path planning). While the FMS has a global view on all the AGVs in a fleet and may therefore make informed decisions based on the set of all plans, its information is limited to what is available at time of planning. This usually excludes information about humans and human-operated vehicles as well as obstacles, which in an intralogistics setting may include objects like pallets or boxes. To facilitate safe motion of robots within such an environment, both appropriate sensors and the means to act on them need to be present. Since reacting to unforeseen obstacles leads to a deviation from the original plan, appropriate strategies for handling such deviations are required. We strive to implement such a strategy in the way of actively allowing bounded deviation within the FMS' plan as well as a re-planning routine to be triggered by an individual AGV if a planned trajectory may not be executed within that deviation.

One of the main global path planning algorithms for fleet management is the Context Aware Route Planning (CARP) approach of ter Mors et al, (2010) which presents a central graph-based approach for collision-free navigation of autonomous systems. The algorithm guaranties the shortest path from a start to a destination location, without colliding with any other robot. Time windows are used for the path calculation and coordination. Calculated routes are defined as $r_i, \tau_i, \dots, r_n \tau_n,\ \tau_i = \left(t_{i,} t_i^{'}\right)$ where r_i is an entity, vertex or edge of the resource graph and τ_n is a time window that indicates that resource r_i is available from time t_i to t_i'. The concept of CARP can be defined as follows. Each agent plans individual routes successively one after the other. Thereby each AGV is represented as an individual agent. If an agent a_n plans its route, the already planned routes are included in the current calculation. Every resource of the graph has its own time windows that defines the availability. For planning a conflict-free route, overlapping time windows between the individual resources of a route are required, in a way that $\tau \cap \tau ' \neq \varnothing$.

The Push and Rotate (P&R) algorithm was published by de Wilde, Ter Mors and Witteveen (2014) and describes a centralized approach for collision-free navigation in a multi robot scenario. The algorithm uses a graph-based topology, based on the environment, in order to calculate the path between a given start and destination for each autonomous robot in the system. P&R plans successively for each robot $r_i \in A$ a path. A path *p* is a sequence of steps s_i whereby every step represents a resource r_i which can be a vertex or edge of the graph $\left(p_i = s_1, s_2, \dots, s_n\right)$. First, the shortest path is calculated, with the help of path finding algorithms like A* or Dijkstra, while ignoring any obstacles. Afterwards, the algorithm checks for every step whether the corresponding resource is occupied by another robot *r*. When such a situation exists, two options are available to solve the conflict. The first option can be applied if the occupying robot r_0 has not planned a path. In this case it is allowed to push it to a neighbouring vertex without the need to place it back to its original vertex. If the robot r_0 has already planned its own path it is invalid to push the robot out of the way. For this case P&R defines exchange points $e \in E$. Exchange points are vertices which have at least three neighbours. With the help of a vertex e, r_i and r_j can switch their positions. After r_i passed the point of conflict r_j can return to the original position. In contrast to CARP and P&R, *Conflict Based Search* (CBS) *(*Sharon et al, 2015) does not employ an a-priori defined planning order or priority but instead determines the optimal priority on a per-conflict basis, where a conflict generally describes two agents a_i, a_j requiring a given resource r at the same time τ and is defined by the tuple (a_i, a_j, r, τ). A conflict may be resolved by preventing either agent from acquiring r at τ, prohibiting a_i or a_j from using at τ. Each conflict therefore offers a binary choice, resulting in a binary tree containing all possible combinations of constraint choices. By evaluating the tree in ascending order of cost associated with the set of constraints the optimal solution is guaranteed to be found.

Local path planning

While a map might accurately reflect the outline of an environment, obstacles, humans or other vehicles cannot always be accounted for. Furthermore, the complexity of a multi-robot path finding problem often necessitates an abstraction of the environment and a simplification of the robots' motion model such that resulting planned trajectories contain significant uncertainty or consist of only a sparse set of waypoints. For these reasons, a *local planner* must usually also be employed to execute a given global plan. In the following, a selected overview of existing approaches to local planners is given.

If reaction to unforeseen obstacles is not required due to operation in a controlled environment, a local planner that executes the global plan in accordance with the robot's kinematic constraints may be sufficient. Given a robot state $x=(x, y, \theta)$, such a planner describes a control law *u* that regulates the tracking error $x_e=(x_e, y_e, \theta_e)$ to 0, where the error term generally describes the deviation of the robot state from the path. In *Pure Pursuit* (Coulter, 1990) this error term is defined with respect to a *look ahead point* that is located on the path in some defined distance ahead of the robot. A similar approach is also taken in (Indiveri, 1999).

The dynamic window approach (Fox et al, 1997) describes a simple yet effective optimization-based model-predictive controller (MPC). It employs the simplification of reducing the search space to all trajectories $x=f(v,\omega)$ where *v* is a constant linear velocity and *w* is a constant angular velocity, limiting *x* to circular arcs. It is further demanded that $(v,\omega) \in V$ where *V* describes the dynamic window and only includes such pairs (v,ω) that do not result in a collision with obstacles while also being feasible given acceleration limits. The optimal trajectory $x^*(v,\omega)$ is found by minimizing a cost term $G(v,\omega)$. In the original publication by Fox et al, (1997), the cost term of:

$$G(v,\omega)=\sigma(\alpha \cdot heading(v,\omega)+\beta \cdot dist(v,\omega)+\gamma \cdot vel(v,\omega)) \quad \text{(Eq. 5.1)}$$

punishing deviation from global path in form of $heading(v,\omega)$, proximity to obstacles in $dist(v,\omega)$ as well as low velocities in $vel(v,\omega)$, is proposed. In the fashion of MPC, x^* is executed not for its full planned horizon, but only for a relatively small period of time before a new trajectory is optimized for again.

In the Time Elastic Band (TEB) Planner (Roesmann et al, 2012), an optimal discretized trajectory B^* is computed where B^* is a sequence of tuples of poses, augmented by time differences.

$$B^*=\arg\min(B)\sum_i k_i f_i(B) \quad \text{(Eq. 5.2)}$$

Here $f_i(B)$ are individual cost terms weighted by $k_i(B)$, which either penalize non-adherence to kinodynamic constraints or act as either attracting or repelling forces on the individual time-augmented poses within the trajectory. The latter forces make the trajectory behave in a fashion akin to an elastic band – stretching around obstacles and contracting in their absence – serving as namesake for this approach.

Collision avoidance

Planning motions of manipulators and robots so that collisions with workspace limits and obstacles do not occur has been an active research topic for decades. There are two approaches, which have been of the most interest in practice, heuristic potential field approach (Khatib, 1985; Yong & Narendra, 1992) and sampling based probabilistic roadmap approach (PRM) (Kavraki et al, 1996)

In potential field approach, each obstacle and manipulator are assigned a potential field function, which makes them to repel each other. The path planning the tries to navigate along potential valleys in workspace. Problems in this approach include high computational complexity and difficulty to find efficient potential functions. Several functions have been proposed and used mostly for low degree of freedom robots (Wang et al, 2000; Bounini et al, 2017).

In sampling based probabilistic map approach, the obstacles and the manipulator are modelled in some way that allows checking of collisions. Collision detection is a widely studied area and efficient real time algorithms have been developed and applied in e.g. 3D game engines. The collision detection systems usually contain two phases: broad phase and narrow phase. Broad phase contains checking whether various objects exist in the same region in 3D space. The most applied technique in broad phase is the use of octree data structure to divide space in eight subspaces and those again in eight subspaces and so on (Meagher, 1980). In narrow phase, pairs of objects or surface primitives are checked for collisions. This can be done using straight forward analytic geometry computations or by using more general algorithms developed for this purpose. One of the most used algorithms in 3D game engines is Gilbert-Johnson-Keerthi (GJK) algorithm (Gilbert et al, 1988) and its further variations.

When setting up a sampling based probabilistic map, a set of manipulator configurations, in which no collision occur, are chosen randomly. The number of configurations in this set depends on the complexity of the environment and manipulator. The connections between these configurations, i.e., manipulator movements from one configuration to another, are then checked using some simple local planner. In addition, some measure for the distance of the connection is recorded for valid connections. These configurations and distances between them represent nodes and connections in the graph representation of the map for collision free path planning. The construction of the map graph can be done once the machine is settled in the work area and it is allowed to take some computing time.

During the operation, the manipulator is typically commanded to move from current start configuration to some target configuration. These start and target configurations are added to the map graph by finding collision free connections to some nearest nodes in the graph using the local planner. This operation is the most time-consuming part of work-time operation. However, for point-to-point movements, the movement type is free; it does not have to be e.g., linear and the local planner can be very simple.

The basic form of probabilistic road map (PRM) planner is a multi-query planner, where pre-processing is used to build a map which can be used for planning trajectories between any start and goal configurations. Other approach is single-query planning with no pre-processing, where paths are planned using random sampling for each start and target configuration separately. Most popular is rapidly-exploring random trees (RRT) approach, where the path from start to target is searched by using RRT algorithm to grow a random tree from both ends toward each other (LaValle & Kuffner, 2015; Kuffner & LaValle, 2000). A review of sampling-based path planning algorithms is e.g., in (Karaman & Frazzoli, 2011).

This technology becomes quite relevant in the case of mining booms, which have to traverse narrow, winding tunnels full of obstacles. This machines usually also require complex inverse kinematic models to calculate the joint positions that correspond to given locations; and visual servoing which was already mentioned in section 5.2.1.2.

5.2.2.3 Progress beyond the SoA

The expected outcomes of the project are the following:

Extend VDA5050 standard

The current standardized interface for AGV communication between AGVs and master control systems is known as VDA5050. The current state of the vda5050 enables the control of a heterogeneous fleet of vehicles. This control is mostly directed from the fleet management to the vehicle. In the case of an obstacle, there is none communication in the other direction. This means that a vehicle cannot report the current situation to the fleet management system, in order to avoid further usage of the blocked passage in newly created vehicle trajectories. For this reason, the vda5050 needs to be expanded with a way to communicate a possible blockage in the topology to the fleet management system. When the reporting is done, the fleet management system can cooperate with the vehicle to solve the blockage. One solution could be to allow the vehicle to temporarily leave its designated path, for this the fleet management system needs to ensure the absence of other vehicles nearby, which could collide with the blocked vehicle, for the duration of leaving the designated path. The path consists of a sequence of entities in a graph-based abstraction of the surroundings. Each entity has a physical manifestation in the environment, which gets exclusively reserved for a specific vehicle for a calculated time window. If a vehicle has the permission from the fleet management system to temporarily leave its previous computed path, it needs to incorporate the new entities in its path while considering the physical manifestation of each entity. The current version of the vda5050 only supports the specification of a radius for the permitted deviation. This is not always enough since the radius is specified around the next goal coordinate and not on a variable length across a graph edge. In addition, the physical manifestation of the graph-based topology can support the specification of a capacity, in order to allow vehicle specific collision avoidance strategies within the defined area. This also enables autonomy of the vehicle, since the collision avoidance runs, as safety-critical application, on the vehicle itself. Another possible solution is a replanning request via an extra communications channel in the VDA5050. This case is useful, if the blockage is judged to be impassable by the vehicle. In this case the fleet management system needs to abort the current path and compute a new one, while avoiding the recently found blockage. Furthermore, this solution might include the replanning for serval vehicles, which are standing behind the blocked vehicle.

Scale Global Path Planning

While the reservations of the graph entities are based on time-windows, the coordination is done by a modified version of the context-aware route planning algorithm (Mors et al, 2010). In their paper, ter Mors et al. present the CARP algorithm in a single thread variant. To promote scalability, this algorithm should be parallelised. In order to estimate times-windows which are precise as possible, the vehicle's dynamics must be able to be reproduced as well as possible. However, the solution must not be too specific, as this would limit its use in heterogeneous fleets. One way of estimating the times is to use a neural network, which could be trained based on real recorded data.

Integration of reserved subspaces into local planner

The local planners outlined in 1.1.2 do not explicitly model multiple robots but treat them as obstacles from the perspective of the respective planned robot. Depending on use-case and environment, this may result in poor performance or even deadlocks. By reserving subsets of the configuration space over defined time windows in the global planning stage within the FMS, we account for the plans of multiple robots. This however must be synchronised with the local planner instances responsible for trajectory execution on the robots, which is not supported by the ones currently widely in use. We therefore intend to integrate the reserved subspaces into a local planner as well as the communication between FMS and AGV.

Advance control and planning methods for collision avoidance

Collision avoidance and trajectory planning algorithms must be able to determine workspace of the machine as well as generate safe and collision free paths. Obtained trajectories should be possible to validate before execution. Cartesian control method and visual servoing is needed for controlling machines to find objects accurately. DTs and HIL simulations are used to verify that this chain of technologies work before it is used in a real environment.

Custom solution inverse kinematics of the charging boom

Kinematic structure of charging booms differs from industrial robots and a custom method for solving inverse kinematics must be developed. In a usual charging boom, there are 6 rotational axes. It is useful to notice some features of the kinematic structure of a charging boom: a) Axis of the two last joints intersect b) Axis of the joints 2, 3 and 4 are parallel c) Axis 2, 3, 4 and 5 are in same plane. These features can be utilized in solving the inverse kinematics problem.

Inverse kinematics usually give several possible solutions (configurations), which allow robot to reach the goal position and orientation. Usually, the solution that gives joint values closest to the current joint values is selected. During the execution of the trajectory changes on the configuration are not allowed. In mining booms, the movements of the joints are much more limited than in industrial robots. It is very likely, that in some situations it is not possible to move from the current position to goal position without changing a configuration. A method for handling this situation must be developed. Since rotation of the tool around z-axis is not relevant during the charging operation, it is possible to select this freely. This gives us more freedom for path planning. This custom inverse kinematics solution could be used for validating the planned trajectory of a mining boom before execution of the automatic movement. It could also be used as an online cartesian set point calculation tool.

5.2.2.4 Requirements

ID	Requirement	Comments
R34-D4.1	Robots managed within the FMS must have access to appropriate on-board sensors to effectively detect obstacles and humans	
R35-D4.1	Robots must possess an accurate estimate on their own pose at all times	Posible solutions: • Graph-based representation of the environment for trajectory planning (FMS) • Automatic loading and unload mechanisms of the AGV • Simulation of the AGV

Table 10: Path planning and real-time decision-making requirements

5.3 BB10 requirements

The next table summarizes the requirements for Building Block 10, integrating the ones defined in D2.3 with the new ones defined for each tecnology in the different tables available insection 5 of this document.

ID	Requirement	Priority	Verify	Tasks
R219-D2.3-B10-sw	The control system algorithms can be integrated in a real HIL testing environment.	M	T	T4.5
R220-D2.3	The LIDAR sensor must be suitable for the usage of SLAM.	M	D	T4.5
R221-D2.3	Enough processing power is needed to work with real-time sensor data (for localisation and navigation calculation).	M	D	T4.5
R222-D2.3-B10-sw	Visual servoing for motion control is based on real-time camera systems integrated into the control system.	M	T	T4.5
R223-D2.3-B10-sw	Path planning algorithm must be possible to be done in near-real-time in the control system.	M	T	T4.5
R224-D2.3-B10-sw	Collision avoidance must be possible to execute in real time	M	T	T4.1, T4.5
R225-D2.3-B10-sw	The machine vision algorithms used in visual servoing comprise of ML open-source libraries, i.e. compatible with BB8.	S	T	T4.5
R226-D2.3-B10-sw	The libraries and algorithms used must be open source or industrial standard.	M	I	T4.5
R227-D2.3-B10-sw	Possible to port different programming languages and operating systems/embedded control systems	M	I	T4.1 T4.5
R228-D2.3-B10-sw	The calibration and parametrisation of the algorithms and sensors related must be able to be configured on-site.	M	T	T4.5
R229-D2.3-B10-sw	The motion control algorithms are intuitive to operate from a usability perspective.	M	T	T4.5
R230-D2.3-B10-sw	Path generation should work automatically with minimal input from the operator	M	T	T4.1, T4.5
R231-D2.3-B10-sw	User can intervene automatic path execution safely	M	T	T4.1, T4.5
R232-D2.3	The short-term future path of the robot should be predictable for human traffic participants.	S	D	T4.1 T5.1
R233-D2.3	Path planning should take into account the presence and movement of human traffic participants and generate cooperative movement behaviour.	C	D	T4.1 T5.1
R234-D2.3-B10-sw-SAF	The automatic movements are tolerant to failures of the control system (servo drives, sensors, actuators, software singularities)	M	T	T4.1 T4.5
R235-D2.3-B10-sw	Measurement outliers and incomplete data (LIDAR data) should not lead to dangerous or unexpected behaviour.	M	T	T4.1, T4.5
R236-D2.3	The optimised Neural Networks must be able to run on the existing hardware (I.MX), e.g. Nvidia Jetson Or the mini-pc	M	D	T3.3, T3.4
R237-D2.3	The FPGA hardware shall not cost more than 1500€ and the embedded hardware not more than 500€.	M	D	T3.3, T3.4

R238-D2.3-P5-hw	The target cost of the goods in the visual servoing application for the camera system (excluding servomotors and drives) < 1000€.	M	I	T4.1 T7.1
R239-D2.3-B10-sw	Visual servoing and motion control algorithms related are modular so that these can be ported to other manipulators	W	T	T4.1, T4.5
R240-D2.3-B10-sw	Algorithms can be adapted/extended to different sensor types and boom types.	W	T	T4.1, T4.5
R241-D2.3-B10-sw	The implemented motion control algorithms are directly interoperable with a HIL toolchain and dynamic DT counterpart of the boom.	M	I	T4.1, T4.5
R242-D2.3-B10-sw	Algorithms can be tested and verified in a simulation environment with DTs.	M	I	T4.1, T4.5
R243-D2.3-B10-sw	The motion control algorithms are fail-safe	M	T	T4.5
R244-D2.3-B10-sw-SAF	The algorithms must comply with mobile machinery directives	M	T	T4.1, T4.5
R32-D4.1	Object detection must work in harsh environments with varying lightning conditions and dark surfaces			
R33-D4.1	The object detection algorithm must provide accurate position information for the boom control in real time			
R34-D4.1	Robots managed within the FMS must have access to appropriate on-board sensors to effectively detect obstacles and humans			
R35-D4.1	Robots must possess an accurate estimate on their own pose at all times			

Table 11: BB10 requirements

6 Real-Time Smart-Control Platform (BB4, Task 4.6)

6.1 Introduction

Motion control systems require a fast reaction time in order to interact with the environment. These small reaction windows need the control system to work in real time at high frequencies, often requiring specific hardware platforms or modified software layers to ensure a regular update rate. In this section we will discuss some of these hardware and software solutions for real-time smart control and data processing.

6.2 State of the Art

6.2.1 Hardware platforms

In the last years, common single- or multi-core architectures have become uncapable of fulfilling the demand for high efficiency required by many industrial applications, such as the latest video coding standards, or by some execution contexts, like security, medical imaging and video processing. This is the reason why dedicated logic to process such applications is being used more and more alongside software cores, resulting thus in heterogeneous platforms.

Many control applications expect strictly periodic and deterministic execution of control which can be the most challenging part of their design, particularly in multi-application scenarios. This imposes several important requirements on the target implementation platform. Composability, predictable, and determinism are the three most notable ones.

Some of the most well-known predictable multi-core platforms for this kind of deterministic control are:

- CompSOC: this platform ensures deterministic and interference-free execution. (Goossens et al, 2017). It is a tile-based embedded platform of processor tiles, local and shared memories, and interconnections. The platform has three MicroBlaze processor tiles connected through shared memory.
- T-CREST (Schoeberl et al, 2015): it targets safety-critical applications by optimizing the Worst-Case Execution Time (WCET) of the application using predictable hardware architecture.
- FlexPRET (Zimmer et al, 2014): It is a multi-threaded processor which targets mixed-critical implementations. The applications running on the platform are either hard real-time threads (HRTT) or soft-real-time threads (SRTT).
- PTIDES (Derler et al, 2008): it's a programming model for cyber-physical systems and a special implementation of a discrete-event model of computation. In PTIDES, every hardware component (such as sensors and actuators) and applications (such as control) are actors which communicate through time-stamped events.

In some applications, adaptivity can be an essential necessity. This can be granted by leveraging on reconfiguration, but hardware reconfiguration is still a research niche, since execution efficiency and flexibility are contradicting requisites, hence complicating the design and management of heterogeneous substrates, which have to also comply with ever-shortening time to market.

To this regard, different studies (Palumbo et al, 2019; Palumbo et al, 2017; Yan et al, 2012) have demonstrated that adopting a Coarse-Grain Reconfigurable (CGR) hardware reconfiguration approach can aid in providing flexibility and adaptation to changeable functional and non-functional requirements; nonetheless, their main issue is still the complexity of their design, which has to cover aspects such as resource mapping, run-time management and optimization. As a result, automated methodologies for their design and management are becoming a must.

Within the related literature, the issue of developing automated methodologies for the design and management of CGR devices has been addressed through model-based automated and semi-automated strategies. On the one hand, a first attempt to exploit dataflow models to achieve reconfiguration has been made by Beaumin et al. (2010), whose reconfigurable coprocessor customizes a substrate of generic processing units (software cores executing actors) and hardware FIFOs. On the other hand, Ansaloni et al. proposed a novel scheduling strategy to efficiently map operations on CGRA architectures (Ansaloni et al, 2012). Finally, a more resolute step towards a complete dataflow based reconfigurable accelerators design support is the Multi-Dataflow Composer (MDC), a tool able to automatically generate a reconfigurable accelerator starting from the dataflow models of the desired functionalities by multiplexing in time common actors among them. Over the years, MDC has been coupled with High Level Synthesis (HLS) tools to derive the complete hardware specification in a totally automated way

6.2.2 Software Virtualization

Several open-source projects have aimed at creating a real-time version of the Linux operating system. These attempts can be classified in the following categories (Lipari & Scordino, 2006):

- Real-time Linux

  These projects have aimed at increasing the predictability of the standard mainline kernel. Robert Love introduced the Preemptible Kernel patch (Heursch et al, 2003). This mechanism, integrated into the 2.6 kernel series, has made the kernel fully preemptible. Then, the popular PREEMPT RT project has aimed at shortening the maximum latency experienced by a real-time task by increasing timers' accuracy and kernel concurrency and reducing non-preemptible sections. Very recently, De Oliveira et al. (2019) have proposed an automata-based model for describing and validating the behavior of threads in PREEMPT RT on single-core systems. In parallel, the Linux kernel community has implemented a real-time CPU scheduler, called SCHED DEADLINE (Lelli et al, 2016). It allows specifying the amount of CPU time reserved to a real-time task with a per-task timing granularity. Despite these projects having certainly contributed to improving the responsiveness of Linux, they have not been capable of reaching the performance needed for the fast control loops implemented nowadays in industrial automation.

- Dual kernel

  The "dual-kernel" approach consists of creating a Hardware Abstraction Layer (HAL) and modifying the interrupt handling routines for executing a tiny real-time operating system (RTOS) prior to Linux. The Linux kernel and its tasks are thus executed at a lower priority than the real-time tasks. This approach, originally introduced by RT-Linux (Yodaiken et al, 1999), has been then evolved by the RTAI (Mantegazza et al, 2000) and Xenomai projects, introducing the concept of "execution domains" for partially overcoming the original limitations of kernel-space programming. These projects have successfully proven the feasibility and the performance of the proposed approach (Barbalace et al, 2008).

- Partitioning hypervisor

  The most recent research direction in various application domains (e.g. automotive) consists in taking advantage of hardware-assisted virtualization of modern processors for allocating different physical cores to the execution of components with different timing characteristics. Xen is a type-1 open-source hypervisor, therefore running on top of the hardware, that can run using both plain hardware virtualization or exploiting paravirtualization offered by a Linux VM. Thanks to the device emulation provided by Qemu, it can also run unmodified versions of operating systems (e.g. Windows). RT-Xen (Xi et al, 2011), (Xi et al, 2014) was an attempt at creating a real-time version of Xen. More recently, the community developing the Xen hypervisor has implemented the “null” scheduler (Abeni & Faggioli, 2019), which statically assigns each virtual CPU to one and only one physical CPU, avoiding any scheduling decision — therefore, reducing the overhead. To simplify the design and avoid the need of device drivers in the hypervisor itself, Xen relies on a special privileged guest called “dom0”, usually consisting of a Linux OS, that boots before normal guests (called “domU”). This constraint represents an issue in the embedded domain, where a short boot time of the guests is often a mandatory requirement. A very recent attempt sponsored by Xilinx aims at removing the need of Xen’s “dom0” domain at least for ARM processors. KVM is the standard support for adding a hypervisor layer to the Linux kernel. A recent work has shown that KVM can obtain lower worst-case latencies than Xen (Abeni & Faggioli, 2019). Some kernel contributors have also worked on adapting KVM to real-time workloads (Zhang et al, 2011). In parallel, Siemens has developed a tiny open-source hypervisor, Jailhouse (Ramsauer et al, 2017), targeting the embedded safety critical domains. Very recently, the Bao hypervisor has been proposed (Martins et al, 2020) to deliver a virtually transparent, and highly secure partitioning layer for the most critical situations.

- Isolating hypervisor

  Contention on shared hardware resources, such as DRAM, bus or caches, undermines real-time performance in a mixed-criticality context. This has been studied by Cavicchioli et al. (2017) and Danielsson et al. (2019), and also discussed by Capodieci et al. (2020) in the context of automotive applications. Hardware solutions for cache management that exist are either rigid and obsoleted, or virtually not available among the industrial computing platform segment. This has been the motivation for elevating the hypervisor concept up to the hardware isolation role, which has been shown to be effective in restoring memory access determinism in high-performance embedded systems. Software cache partitioning has been demonstrated to be a key feature for real-time systems (Kloda et al, 2019) and military applications (Saudo et al, 2020), and it has been proposed on the Arm implementation of Jailhouse, Xen (Miccio and Solieri, 2019) and Bao (Martins et al, 2020).

- Industrial automation

  Most of the work in the literature for industrial automation has exploited virtualization only for cloud and fog computing (Givehchi et al, 2014). Among the notable exceptions, Mahmud et al. (2014) evaluated the feasibility of Xen-based virtualization in a multicore distributed system running RT-Linux. Azarmipour et al. (2018), instead, discussed design issues of a virtualized system based on PikeOS. Moreover, both these works did not measure the performance of field-buses like EtherCAT. In (Scordino et al, 2020) a couple IMOCO4.E partners designed a multi-OS platform for industrial automation. The platform was based on the Xen hypervisor and capable of executing a general-purpose OS (i.e. Windows or Linux) and an RTOS for real-time control through the EtherCAT fieldbus.

6.2.3 Real time communication

As pointed out above, systems designed with virtualization report great advantages related to security, cost, reliability, availability, adaptability (Obasuyi et al, 2015) making it a valid choice with remarkable performance. In recent years, paravirtualization exhibited higher performance compared to full virtualization (Motika & Weiss, 2012; Fayyad-Kazan et al, 2013) (where all the hardware is emulated like Qemu). In addition to partitioning computing resources, virtualization enables multiple applications to gain access to the hardware resources on the host machine (Obasuyi et al, 2015).

One of these hardware resources concerns connectivity, as the scarcity of the communication connectors raises new challenges for hypervisors, which need to also act as a transparent proxy between the network and the guest machines. Techniques have therefore been proposed in order to enable time-sharing mechanisms of shared hardware resources to all guest machines through different protocols. The original Ethernet has been extended to allow sharing a Network Interface Card (NIC) between multiple guest machines (Motika & Weiss, 2012), where, once again, it is shown that paravirtualization offers better performance compared to emulated contexts.

Moreover, the autonomous systems, targeted in the IMOCO4.E project, bring the requirement to use specific network protocols where temporal behavior control is a key. The most well-known protocol used in system automation is the Controller Area Network (CAN) (Gergeleit et al, 1994). It is a message-based protocol which does not need a central host computer. An extension for virtual environment vCAN has been proposed with great results (Herber et al, 2014; Breaban et al, 2016).

One of the most used protocols by IMOCO4.E partners is the field-bus protocol EtherCat (Jansen & Buttner, 2004). This protocol is suitable for both hard and soft real-time computing requirements in automation technology. It has been used in a real-time virtualized context for industrial automation (Scordino et al, 2020). EtherCat has proven to be a suitable choice for real-time control systems with guaranteed performance (Huang & Chien-Hao, 2014). However, EtherCat has strong requirements in terms of hardware support where a specific ethernet card needs to be present.

A more recent technology providing similar guarantees is named Time Sensitive Network (TSN) (Farkas et al, 2018). TSN is a set of standards that extend the Ethernet protocol by focusing on time (Finn, 2018). The desirable properties that a hypervisor should fulfill to be compliant with TSN protocols have been stated in (Leonardi et al, 2020), and in particular the limit of some scheduling policies and granularities to meet the timing requirements. An early framework has also been proposed (Caruso et al, 2021) in order to assess the viability of using TSN in a virtualized environment.

6.3 Progress beyond the SoA

Major Contributions are mentioned below to progress beyond the SoA. These can be summarized as

- Multi-sensor multi-rate control.

  In Vision-in-the-Loop control for achieving the high-precision high-throughput for motion control systems applications, a fusion of multi-sensors is required. The sensor fusion between encoder and vision to improve sensing accuracy with multi-rate sensing is necessary.

- Smart scheduling of Vision-in-the-Loop using parallel and pipelined processing.

  Once the sensor data is fused and the sensing accuracy of the motion control system is improved, the control performance of the system should be improved. Scheduling techniques e.g., parallelism, pipelining, and approximation can be employed to increase the control loop performance.

6.4 BB4 requirements

ID	Requirement	Priority	Verify	Comments	Tasks
Interfaces and connectivity					
R114-D2.3	BB4 shall offer industry standard interfaces to: 1. Encoders (e.g. Biss-C, EnDAT) 2. Drives (e.g. EtherCAT) 3. Layers 1 and 2 (e.g. EtherCAT)	M	I	COTS components are needed to interoperate with TSN	T3.1, T4.1, T5.1
R115-D2.3	BB4 must be able to run on an "ARM" based platform.	M	I		T4.1
R116-D2.3	BB4 can be connected to Matlab/Simulink	C	A		T4.1
Maintainability (modularity, analysability, testability)					
R117-D2.3	BB4 shall be ready for the vertical distribution of smart control algorithms	M	D	Interfacing with BB5	T4.1
Performance					
R118-D2.3	BB4 shall support control loop update rates of at least 20 kHz	M	D		T3.1 T4.1
Compatibility (interoperability, co-existence)					
R119-D2.3-B5	BB4 should be compatible with code generated from Simulink	M	D	Interfacing with BB5	T4.1
R120-D2.3	BB4 shall be compatible and portable with x86-based platforms	S	D		T4.1
Portability (adaptability, replaceability)					
R121-D2.3	BB4 shall offer customizability to run any combination of custom control loops in parallel, including MIMO control loops	M	D		T4.1
R122-D2.3	BB4 shall offer customizability such that non-standard tasks (i.e., tasks which are typically performed in research) can be performed. Examples include flexibility in allowed controller structures and reference/feedforward signals.	S	T		T4.1
Cost					
R123-D2.3	BB4 shall have a target cost of goods of €1000 for a basic version	M	I		T4.1

Table 12: BB4 requirements

7 IMOCO4.E Building Block connections

The IMOCO4.E project will feature several use cases/pilots and demonstrators. These will be related to some of the BBs contained in layer 2 and more related to smart control. Figure 7.1 details the relationships between use cases/pilots/demonstrators and BBs, as well as the partners responsible for this connection.

		BB4	**BB5**	**BB10**
Use Cases	**UC1**. Industrial drive for smart mechatronics applications		UNIBS & UWB	
	UC2. CNC for integrated machine tool and robot control		FAG & TEK	
	UC3. Tactile robot teleoperation			
	UC4. Advanced and intuitive robot control and programming		UWB	UWB
Pilots	**P1**. 3D printing	SIOUX	SIOUX	
	P2. Semiconductor manufacturing	EVI	TUE & ITEC	
	P3. High speed packaging	CRIT		
	P4. Healthcare robotics		PMS & TUE	PMS
	P5. Mining/tunneling robotic boom manipulator	UNIMORE		NORMET
Demonstrators	**D1**. High precision cold forming of complex 3D metal parts			
	D2. Smart sensoring on injected plastic parts	UNIMORE		
	D3. Autonomous intra-logistic transportation			STILL
	D4. Vision-based AI pick & place robotics for randomly arranged and differently shaped bottles			

Table 13: Building block connection matrix

8 Conclusion

D4.1 provides a revision of the shortcomings, state-of-the art and contributions of IMOCO4.E related to smart control. The focus of D4.1 is on smart control (mainly related to Layer 2). There are different technologies and approaches that are introduced discussed that will be the core of WP4. It is also illustrated how these different components are connected to different BBs (mainly BB4, 5 and 10).

D4.1 also integrates the generic requirements related to each BB (mainly BB5, BB10 and BB4, in this order) gathered from D2.3. Then more specific requirements are introduced related to identified shortcomings of current approach and state-of-the-art. D4.1 also outlines how IMOCO4.E will address these identified shortcomings and necessities for the future with specific contributions beyond the state of the art.

9 References

Abadi M. et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abdelawwad M. et al. (2021). Überwachung des Mensch-Roboter-Kollaborationsraums mittels Millimeterwellen-Radar. In: Zeitschrift für wirtschaftlichen Fabrikbetrieb, vol. 116, no. 6, pp. 377-381, 2021 https://doi.org/10.1515/zwf-2021-0108

Abdulrahman A, Emhemed A. & Mamat R. (2020). Model Predictive Control: A Summary of Industrial Challenges and Tuning Techniques. In: International Journal of Mechatronics, Electrical and Computer Technology (IJMEC).

Abeni L. & Faggioli D. (2019) An experimental analysis of the xen and kvm latencies. In: 2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC), pages 18–26. IEEE.

Akyuz, I.H. et al. (2011). PID and state feedback control of a single-link flexible joint robot manipulator. In: *IEEE International Conference on Mechatronics*: IEEE), 409-414.

Al-Bender F, Lampaert V. & Swevers J. (2004). A novel generic model at asperity level for dry friction force dynamics. In: Tribology Letters, vol. 16, pp. 81–93.

Altintas Y. (2012). Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design. In: 2nd edn. Cambridge U. Press.

Amato F. (2006). Robust control of linear systems subject to uncertain time-varying parameters. In: *Lecture Notes in Control and Information Sciences*, *325*, 1–194. https://doi.org/10.1007 /3-540-33276-6.

Angeli D. & Müller M.A. (2018). Economic Model Predictive Control: some design tools and analysis techniques. In: Handbook of Model Predictive Control pp 145–167.

Ansaloni G, Tanimura K, Pozzi L. & Dutt N. (2012). Integrated kernel partitioning and scheduling for coarse-grained reconfigurable arrays. In: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 31 (12) 1803–1816. doi:10.1109/TCAD.2012.2209886.

Armstrong-Hélouvry B, Dupont P. & Canudas de Wit, C. (1994). A survey of models, analysis tools and compensation methods for the control of machines with friction. In: Automatica, 30(7):1083–1138.

Åström K.J. (1996). Control of Systems with Friction. In: Lund Institute of Technology, Lund University, Sweden.

Augasta M & Kathirvalavakumar T. (2013). Pruning algorithms of neural networks—a comparative study. In: Open Computer Science, 3(3), 105-115.

Ayala H.V.H. & dos Santos Coelho, L. (2012). Tuning of PID controller based on a multiobjective genetic algorithm applied to a robotic manipulator. In: Expert Systems with Applications, 39(10), 8968-8974.

Azarmipour M. et al. (2018). Dynamic resource management for virtualization in industrial automation. In: IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society, pages 2878–2883. IEEE.

Bachnas A.A, Tóth R, Ludlage J.H.A, Mesbah A. (2014). A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study. In: Journal of Process Control, 24(4), 272-285.

Bamieh B. (2003). Intersample and finite wordlength effects in sampled-data problems. In: IEEE Transactions on Automatic Control, 48(4), 639-643.

Bamieh B. & Giarre L. (2002). Identification of linear parameter varying models. In: International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal, 12(9), 841-853.

Barbalace A. et al. (2008). Performance comparison of VxWorks, Linux, RTAI, and Xenomai in a hard real-time application. In: IEEE Transactions on Nuclear Science, 55(1):435–439, 2008.

Baxter P. et al. (2017). Robot education peers in a situated primary school study: Personalisation promotes child learning. In: PLoS ONE, 12(5), Art. no. E0178126.

Beaumin C, Sentieys O, Casseau E. & Carer A. (2010). A coarse-grain reconfigurable hardware architecture for RVC-CAL-based design. In: Conf. on Design and Architectures for Signal and Image Processing, pp. 152–159.

Bemporad A, Morari M, Dua V. & Pistikopoulos E.N. (2002). The explicit linear quadratic regulator for constrained systems. In: Automatica 38.

Biagiotti L, Moriello L. & Melchiorri C. (2015). A repetitive control scheme for industrial robots based on b-spline trajectories. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 5417–5422.

Blanken L, Boeren F, Bruijnen D. & Oomen T. (2017). Batch-To-Batch Rational Feedforward Control: From Iterative Learning to Identification Approaches, with Application to a Wafer Stage. In: *IEEE/ASME Transactions on Mechatronics*, *22*(2), 826–837. https://doi.org/10.1109 /TMECH.2016.2625309.

Bolder J, Oomen T. & Steinbuch M. (2014). On inferential Iterative Learning Control: With example to a printing system. In: American Control Conference.

Bonchis A, Duff E, Roberts J. & Bosse M. (2014). Robotic explosive charging in mining and construction applications. In: IEEE Transactions on Automation Science and Engineering 11, pp. 245–250.

Bounini F, Gingras D, Pollart H. & Gruyer D. (2017). Modified Artificial Potential Field Method for Online Path Planning Applications. In: 2017 IEEE Intelligent Vehicles Symposium (IV), Redondo Beach, CA, pp. 180-185.

Breaban G. et al. (2016). Virtualization and emulation of a CAN device on a multi-processor system on chip. In: 2016 5th Mediterranean Conference on Embedded Computing (MECO). IEEE.

Bristow D.A, Tharayil M, & Alleyne A.G. (2006). Survey Of Iterative Learning Control: A Learning-Based Method for High-Performance Tracking Control. In: *IEEE Control Systems*, *26*(3), 96–114. https://doi.org/10.1109/MCS.2006.1636313.

Buondonno G, & De Luca A. (2015). A recursive Newton-Euler algorithm for robots with elastic joints and its application to control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems *(IROS)*: IEEE), 5526-5532.

Butcher M. & Karimi A. (2010). Data-driven tuning of linear parameter-varying precompensators. In: International Journal of Adaptive control and Signal processing, 24(7), 592-609.

Canudas de Wit C, Olsson H, Åström K.J. & Lischinsky P. (1995). A new model for control of systems with friction. 40(3).

Capodieci N. et al. (2020). Real-time requirements for adas platforms featuring shared memory hierarchies. In: IEEE Design and Test, 2020. To appear.

Caruso B. et al. (2021). Experimental assessment of TSN support in heterogeneous platforms with virtualization for automotive applications. In: 2021 AEIT International Conference on Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE). IEEE.

Cavicchioli R, Capodieci N. & Bertogna M. (2017). Memory interference characterization between CPU cores and integrated GPUs in mixedcriticality platforms. In: 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pages 1–10. https://doi.org/10.1109/ETFA.2017.8247615.

Centurelli A. et al. (2022). Closed-Loop Dynamic Control of a Soft Manipulator Using Deep Reinforcement Learning. In: IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 4741-4748, doi: 10.1109/LRA.2022.3146903.

Chaoui H, Sicard P. & Gueaieb W. (2009). ANN-based adaptive control of robotic manipulators with friction and joint elasticity. In: IEEE Trans. Ind. Elect., 56(8), 3174–3187.

Chaumette F & Malis E. (2000). 2 1/2 D visual servoing: a possible solution to improve image-based and positionbased visual servoings. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 630-635.

Chen T. & Francis B.A. (2012). Optimal sampled-data control systems. In: Springer Science & Business Media.

Chen W. et al. (2011). Position control of a 2DOF underactuated planar flexible manipulator. In: IEEE International Conference on Mechatronics and Automation: IEEE, 464-469.

Chen X. & Tomizuka M. (2014). New repetitive control with improved steady-state performance and accelerated transient. IEEE Transactions on control systems technology, vol. 22.

Chien C.J. & Ma K.Y. (2013). Feedback control based sampled-data ILC for repetitive position tracking control of DC motors. CACS International Automatic Control Conference, CACS 2013 - Conference Digest. 377-382. 10.1109/CACS.2013.6734164.

Cho K. et al. (2014). On the properties of neural machine translation: Encoder–decoder approaches. In: *Proceedings of SSST 2014 - 8th Workshop on Syntax, Semantics and Structure in Statistical Translation.* Association for Computational Linguistics (ACL), pp. 103-111, 8th Workshop on Syntax, Semantics and Structure in Statistical Translation, SSST 2014, Doha, Qatar, 10/25/14.

Cole M.O. (2012). A class of low-pass FIR input shaping filters achieving exact residual vibration cancelation. In: Automatica, 48(9), 2377–2380.

Corke P. I. & Khatib O. (2011). Robotics, vision and control: fundamental algorithms in MATLAB (Vol. 73, p. 2). Berlin: Springer.

Copot C, Ionescu C, Vanlanduit S. & de Keyser R. (2018). Vibration suppression in multi-body systems by means of disturbance filter design methods. In: JVC/Journal of Vibration and Control, 24(14), 2957–2969. https://doi.org/10.1177/1077546317736190.

COMAU: Power System & Products, (2022) Available at: https://www.comau.com/Download/our-competences/robotics/Automation_Products/Folder%20Sinumerik%20A4%20last.pdf (Accessed at: 28 june 2022)

Coulter R. (1990) Implementation of the Pure Pursuit Path Tracking Algorithm. Carnegie Mellon University, Pittsburgh, Pennsylvania.

Dahl P. (1968) A solid friction model. In: Technical Report TOR-0158(3107–18)-1, The Aerospace Corporation, El Segundo, CA.

Danielsson J. et al. (2019). Testing performance-isolation in multi-core systems. In: 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), volume 1, pages 604–609. IEEE.

Darwish M.A.H, et al. (2018). Prediction-error identification of LPV systems: A nonparametric Gaussian regression approach. Automatica, 97, 92-103.

De Luca A, & Siciliano B. (1991). Closed-form dynamic model of planar multilink lightweight robots. In: *IEEE Transactions on Systems, Man, and Cybernetics* 21**,** 826-839.

De Oliveira D.B, De Oliveira R.S. & Cucinotta T. (2019). Untangling the Intricacies of Thread Synchronization in the PREEMPT RT Linux Kernel. In: 2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC), pages 1–9. IEEE.

De Rozario R, Pelzer R, Koekcbakker S, & Oomen T. (2018) Accommodating trial-varying tasks in iterative learning control for LPV systems, applied to printer sheet positioning. In: 2018 Annual American Control Conference (ACC) (pp. 5213-5218). IEEE.

De Wilde B, Mors A.W, & Witteveen C. (2014). Push and Rotate: a Complete Multi-agent Pathfinding Algorithm. In: Journal of Artificial Intelligence Research 51, Oktober, S. 443–492

Denkena B. & Lepper T. (2015). Enabling an Industrial Robot for Metal Cutting Operations.

Derler P, Feng T.H, Lee E.A, Matic S, Patel H.D, Zheo Y, & Zou, J. (2008) Ptides: A programming model for distributed real-time embedded systems. In: University of Berkley, Dept. of Electrical Engineering and Computer Science., Tech. Rep.

Diamond A, Knight R, Devereux D. & Holland O. (2012) Anthropomimetic robots: Concept, construction and modelling. In: Int. J. Adv. Robotic Syst., 9(5), 209.

Dijkstra E.W. (1959). A note on two problems in connexion with graphs. In: Numerische Mathematik. **1**: 269–271.

Eielsen A, Gravdahl T. & Leang K. (2015). Low-order continuous-time robust repetitive control: Application in nanopositioning. In: Mechatronics, Vol 30.

El Khalick A. & Uchyiama N. (2010). Model Predictive Approach to Precision Contouring Control for Feed Drive Systems. In: Journal of Computer Science.

Ellis G. & Gao Z. (2001) Cures for low-frequency mechanical resonance in industrial servo systems. In: Conference Record of the 2001 IEEE Industry Applications Conference. 36th IAS Annual Meeting (Cat. No.01CH37248), pp. 252-258 vol.1, doi: 10.1109/IAS.2001.955419.

Fang B. et al. (2019). Survey of imitation learning for robotic manipulation. In: Int J Intell Robot Appl 3, 362–369. https://doi.org/10.1007/s41315-019-00103-5.

Farkas J, Lo Bello L. & Gunther C. (2018). Time-sensitive networking standards. IEEE Communications Standards Magazine 2.2: 20-21.

Fayyad-Kazan H, Perneel L. & Timmerman M. (2013). Full and para-virtualization with Xen: a performance comparison. In: Journal of Emerging Trends in Computing and Information Sciences 4.9: 719-727.

Finn N. (2018). Introduction to time-sensitive networking. In: IEEE Communications Standards Magazine 2.2: 22-28.

Fitzgerald C. (*2013*). Developing baxter. In: IEEE Conference on Technologies for Practical Robot Applications *(TePRA)*: IEEE), 1-6.

Fox D, Burgard W. & Thrun S. (1997). The dynamic window approach to collision avoidance. In IEEE Robotics & Automation Magazine, vol. 4, no. 1, pp. 23-33, doi: 10.1109/100.580977.

Francis B.A. & Wonham W.M. (1976). The Internal Model Principle of Control Theory. In: Automatica, 12, 457-465.

Frost S, Balas M. & Wright A. (2009). Direct Adaptive Control of a Utility-Scale Wind Turbine for Speed Regulation. In: International Journal of Robust and Nonlinear Control.

Fujimoto H, Hori Y. & Kawamura A. (2001). Perfect tracking control based on multirate feedforward control with generalized sampling periods. In: IEEE Transactions on Industrial Electronics, 48(3), 636-644.

Gilbert E.G, Johnson D.W. & Keerthi S.S. (1988). A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space. In: IEEE Transactions of Robotics and Automation, vol. 4, no. 2, pp. 193-203.

Givehchi O, Imtiaz J, Trsek H. & Jasperneite J. (2014). Control-as-aservice from the cloud: A case study for using virtualized plcs. In: 2014 10th IEEE Workshop on Factory Communication Systems (WFCS 2014), pages 1–4. IEEE.

Glasson D. (1983). Development and applications of multirate digital control. In: IEEE Control Systems Magazine, 3(4), 2-8.

Golabi A, Meskin N, Tóth R. & Mohammadpour J. (2017). A Bayesian approach for LPV model identification and its application to complex processes. In: IEEE Transactions on Control Systems Technology, 25(6), 2160-2167.

Goodrich M.A. & Schultz A.C. (2008) Human–robot interaction: A survey. In: Found. Trends Human Comput. Interact., 1(3), 203–275.

Goossens K. et al. (2017) NOC-based multiprocessor architecture for mixed-time-criticality applications. In: Springer, Handbook of Hardware/Software Codesign, pp. 491–530.

Goubej M. (2016) Fundamental performance limitations in PID controlled elastic drive systems. In: Advanced Intelligence Mechatronics.

Goubej M. & Schlegel M. (2014) Robust PID Control of Electrical Drive with Compliant Load. In: IFAC Proceedings Volumes,47, Issue 3.

Goubej M. & Schlegel M. (2019). PID Plus Repetitive Control Design: H-infinity Regions Approach. In: Process Control.

Goubej M, Vyhlídal T. & Schlegel M. (2020). Frequency weighted H2 optimisation of multi-mode input shaper. In: Auomatica, 121.

Grami S. & Fareh R. (2018). Friction Compensation in a 2DOF Robot Manipulator. In: https://www.researchgate.net/publication/323995728.

Grami S. & Gharbia Y. (2017). Identification of GMS friction model using a new switching function: experimental investigation. In: Int. J. Modelling, Identification and Control.

Hara S, Omata T. & Nakano M. (1985). Synthesis of repetitive control systems and its application. In: 24th IEEE Conference on Decision and Control.

Hara S, Yamamoto Y, Omata T. & Nakano M. (1988). Repetitive control system: a new type servo system for periodic exogenous signals. In: IEEE Transactions on Automatic Control, vol. 33(7).

Hart P, Nilsson N. & Raphael B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In: IEEE Transactions Systems Science and Cybernetics, 4, 100-107. http://dx.doi.org /10.1109/TSSC.1968.300136.

Hartmann D, Herz M. & Wever U. (2018). Model Order Reduction a Key Technology for Digital Twins.

Hattori S, Ishida M, & Hori T. (2001). Vibration suppression control method for PMSM utilizing repetitive control with auto-tuning function and Fourier transform. In: IECON’01. 27th Annual Conference of the IEEE Industrial Electronics So ciety (Cat. No.37243). Vol. 3, 1673–1679.

Herber C. et al. (2014). A network virtualization approach for performance isolation in controller area network (CAN). In: 2014 IEEE 19th Real-Time and Embedded Technology and Applications Symposium (RTAS). IEEE.

Heursch A.C, Grambow D, Horstkotte, A, & Rzehak, H. (2003). Steps towards a fully preemptable linux kernel. In: memory, 48:31.

Hirsch H.G, Stähler J, Hägelen M. & Kulke R. (2020). Analyzing the classification capability of Micro-Doppler spectra. In: IEEE Radar Conference, Florence, Italy, 21.-25.

Hochreiter S. & Schmidhuber J. (1997). Long Short-term Memory. Neural computation.

Huang Y.-L, & Chien-Hao L. (2014). A VM-based approach for real-time EtherCAT control. In: 2014 CACS International Automatic Control Conference (CACS 2014). IEEE.

I-MECH. Deliverables (2022) Available at: https://www.i-mech.eu/publications/deliverables/ (Accessed at: 7 July 2022).

Indiveri G. (1999). Kinematic time-invariant control of a 2D nonholonomic vehicle. In: Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No.99CH36304), pp. 2112-2117 vol.3

Inoue T. et al. (1980). High accuracy control for magnet power supply of proton synchroton in recurrent operation. In: Transactions of the Institute of Electrical Engineers of Japan, 100(7).

Jansen D. & Buttner H. (2004). Real-time Ethernet: the EtherCAT solution. In: Computing and Control Engineering 15.1: 16-21.

Ji J.K. & Sul S.K. (1995). Kalman filter and LQ based speed controller for torsional vibration suppression in a 2-mass motor drive systém. In IEEE Transactions on Industrial Electronics, 42:564–571.

Jiao L. et al. (2019). A survey of deep learning-based object detection. In: IEEE Access 7, pp. 128837–128868.

Jones D. et al. (2020). Characterising the Digital Twin: A systematic literature review. In: CIRP Journal of Manufacturing Science and Technology 29: 36-52.

Kaminski M. & Orlowska-Kowalska T. (2010). Comparison of Bayesian regularisation and Optimal Brain Damage methods in optimisation of neural estimators for two-mass drive system. In: 2010 IEEE International Symposium on Industrial Electronics (ISIE),102–107.

Karaman S. & Frazzoli E. (2011). Sampling-based Algorithms for Optimal Motion Planning. In: The International Journal of Robotics Research, vol. 30, no. 7, pp. 846-894.

Karim A, Schmid S. & Verl A. (2017). Pose and Feed-Direction Dependency Analysis for Milling Tasks With Industrial Robots. In: 24th International Conference on Production Research (ICPR).

Karnopp D. (1985). Computer simulation of slip-stick friction in mechanical dynamic systems. In: Journal of Dynamic Systems, Measurement, and Control, 107(1):100–103.

Katsura S. & Ohnishi K. (2007). Force Servoing by Flexible Manipulator Based on Resonance Ratio Control. In: IEEE Transactions on Industrial Electronics, 54(1):539–547.

Kavraki L.E, Švestka P, Latombe J.C. & Overmars M.H. (1996). Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. In: IEEE Transactions on Robotics and Automation, vol. 12, no. 4, pp. 566-580.9

Khatib O. (1985). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. In: Proceedings of 1985 IEEE International Conference on Robotics and Automation, 25-28 March 1985, St. Louis,MO, pp. 500-505.

Kloda T, Solieri M, Mancuso R, Capodieci N, Valente P. & Bertogna M. (2019) Deterministic memory hierarchy and virtualization for modern multi-core embedded systems. In: 2019 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), pages 1–14. IEEE.

Korber A. and King R. (2010). Model predictive control for wind turbines. In: Proceedings of the European Wind Energy Conference.

Krupa P, Alvarado I, Limon D. & Alamo T. (2021). Implementation of model predictive control for tracking in embedded systems using a sparse extended ADMM algorithm. In: IEEE Transactions on Control Systems Technology.

Kuffner J.J. & LaValle S.M. (2000). RRT-Connect: An Efficient Approach to Single-Query Path Planning. In: Proceedings of the 2000 IEEE International Conference on Robotics & Automation, San Francisco, CA, pp. 995-1001.

KUKA.CNC (2022) Available at: https://www.kuka.com/es-es/productos-servicios/sistemas-de-robot/software/software-de-aplicaci%C3%B3n/kuka_cnc (Accessed at: 28 june 2022)

KUKA iiwa (2022) Available at: https://www.kuka.com/en-us/products/robotics-systems/industrial-robots/lbr-iiwa (Accessed at: 22 july 2022)

Kulke R. et al. (2022). Increased traffic safety by means of intelligent detection and localization technologies. In: European Microwave Week 2021, EuMW, EuRAD, London, 6. April 2022.

Lam D, Manzie C. & Good M. (2011) Application of Model Predictive Contouring Control to an X-Y Table. In: Proceedings of the 18th World Congress the International Federation of Automatic Control Milano (Italy).

Lampaert V, Swevers J. & Al-Bender F. (2004). Comparison of model and non-model-based friction compensation techniques in the neighbourhood of pre-sliding friction. In: Proceeding of the 2004 American Control Conference Boston, Massachusetts.

LaValle S.M. & Kuffner J.J. (1999). Randomized Kinodynamic Planning. In: Proceedings of the 1999 IEEE International Conference on Robotics & Automation, Detroit, MI, pp. 473-479.

Lee S.G, Hur J.O.S, Cho H.C. & Park J. (2006). A PID-Type Robust Controller Design for Industrial Robots with Flexible Joints. In: SICE-ICASE International Joint Conference, pages 5905–5910.

Lee Y, Lim J.J, Anandkumar A. & Zhu Y. (2021). Adversarial Skill Chaining for Long-Horizon Robot Manipulation via Terminal State Regularization. In: *arXiv preprint arXiv:2111.07999*.

Lelli J, Scordino C, Abeni L. & Faggioli D. (2016). Deadline scheduling in the Linux kernel. In: Software: Practice and Experience, 46(6):821–839.

Leonardi L, Lo Bello L. & Patti G. (2020). Towards time-sensitive networking in heterogeneous platforms with virtualization. 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). Vol. 1. IEEE.

Lipari G. & Scordino C. (2006). Linux and real-time: Current approaches and future opportunities. In: IEEE International Congress ANIPLA ’06, Rome, Italy.

Liu J, Li H. & Deng Y. (2017). Torque Ripple Minimization of PMSM based on Robust ILC via Adaptive Sliding Mode Control. In: IEEE Transactions on Power Electronics. PP. 1-1. 10.1109/TPEL.2017.2711098.

Liu L et al. (2019). Deep learning for generic object detection: A survey. In: International Journal of Computer Vision 128, pp. 261–318.

Liuzzo S. & Tomei P. (2008). A global adaptive learning control for robotic manipulators. In: Automatica, vol.44(5).

Luo C. et al. (2017). Adaptive repetitive control of hydraulic load simulator with RISE feedback. In: IEEE Access, vol. 5.

Mae M, Ohnishi W. & Fujimoto H. (2020). MIMO multirate feedforward controller design with selection of input multiplicities and intersample behavior analysis. In: Mechatronics, 71, 102442.

Magar K, Balas M. & Frost S. (2016). Direct Adaptive Torque Control for Maximizing the Power Captured by a Wind Turbine in Partial Loading Condition. In: Wind Energy.

Malis E, Chaumette F & Boudet S. (1999) 2D 1/2 visual servoing. In: IEEE Transactions on Robotics and Automation, 15, pp. 234-246.

Mahmud N, Sandström K. & Vulgarakis A. (2014). Evaluating industrial applicability of virtualization on a distributed multicore platform. In: Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA), pages 1–8. IEEE.

Mangera M, Pedro J.O. & Panday A. (2022). Direct adaptive neural network-based sliding mode control of a high-speed, ultratall building elevator using genetic algorithm. In: SN Applied Sciences, *4*(4), 1–19. https://doi.org/10.1007/S42452-022-04949-6/FIGURES/42.

Mantegazza P, Dozio L. & Papacharalambous, S. (2000). RTAI: Real time application interface. In: Linux Journal. 2000. 10.

Martins J. et al. (2020) Bao: A Lightweight Static Partitioning Hypervisor for Modern Multi-Core Embedded Systems. In: M. Bertogna and F. Terraneo, editors, Workshop on Next Generation Real-Time Embedded Systems (NG-RES 2020), volume 77 of OpenAccess Series in Informatics (OASIcs), pages 3:1–3:14, Dagstuhl, Germany. Schloss DagstuhlLeibniz-Zentrum fuer Informatik. https://doi.org/10.4230/OASIcs.NG-RES.2020.3.

Mayne D.Q. (2014). Model predictive control: Recent developments and future promise. In: Automatica. 50. 10.1016/j.automatica.2014.10.128.

Meagher D.J.R. (1980). Octree Encoding: A New Technique For The Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer. In: Technical Report IPL-TR-80-111, Rensselaer Polytechnic Institute, Troy, New York 12181, 121 p.

Merry R.J.E. et al. (2011) “Delay-varying repetitive control with application to a walking piezo actuator”. In: Automatica 47 1737–1743.

Miccio L. & Solieri M. (2019). Cache colouring for Jailhouse on Armv8. In: https://git.hipert.unimore.it /rtes/jailhouse.

Miccio L. & Solieri M. (2019) Cache colouring for Xen on Xilinx, In: https://github.com/Xilinx/xen/tree /xilinx/release-2019.2-coloring, https://git.hipert.unimore.it/rtes/xen.

Miller T. J. E. (1993). Switched Reluctance Motors and Their Control. In: Monographs in Electrical and Electronic Engineering (Vol. 31). Oxford University Press.

Mirzaei M, Poulsen N. & Niemann H. (2012) Wind Turbine Control: Robust Model Based Approach. In: Technical University of Denmark.

Mishra S, Topcu U. & Tomizuka M. (2011). Optimization-based constrained iterative learning control. In: IEEE Transactions on Control Systems Technology, *19*(6),1613–1621. https://doi.org/10.1109 /TCST.2010.2083663.

Moon J.-H et al. (1998) Repetitive control for the track-following servo system of an optical disk drive. In: IEEE Transactions on Control Systems Technology 6.5, 663–670.

Mooren N, Witvoet G. & Oomen Tom. (2020). Gaussian Process Repetitive Control for Suppressing Spatial Disturbances. In: IFAC-PapersOnLine,Volume 53, Issue 2.

Mors A.W, et al. (2010). Context-Aware Route Planning. In: Dix, J., Witteveen, C. (eds) Multiagent System Technologies. MATES 2010. Lecture Notes in Computer Science(), vol 6251. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-16178-0_14.

Motika G, & Weiss S. (2012) Virtio network paravirtualization driver: Implementation and performance of a de-facto standard. In: Computer Standards & Interfaces 34.1: 36-47.

Nagahara M. & Yamamoto Y. (2010). Robust repetitive control by sampleddata H-infinity filters. In: IEEE Conference on Decision and Control.

Nakano M. & Hara S. (1986). Microprocessor-Based Repetitive Control. In: Microprocessor-Based Control Systems. 279-296.

Obasuyi-Cephas G. & Sari A. (2015). Security challenges of virtualization hypervisors in virtualized hardware environment. In: International Journal of Communications, Network and System Sciences 8.07: 260.

Olsson H. et al. (1998). Friction Models and Friction Compensation, European Journal of Control. In: Volume 4, Issue 3, Pages 176-195.

Ohnishi W, Strijbosch N. & Oomen T. (2021). Multirate state tracking for improving intersample behavior in iterative learning control. In: 2021 IEEE International Conference on Mechatronics (ICM) (pp. 01-06). IEEE.

Omata T, Hara S. & Nakano M. (1987) Nonlinear repetitive control with application to trajectory control of manipulators. In: Journal of Field Robotics 4.5, 631–652.

Oomen T. (2018). Advanced Motion Control for Precision Mechatronics: Control, Identification, and Learning of Complex Systems. In: IEEJ Journal of Industry Applications. 7. 127-140.

Oomen T, Grassens E. & Hendriks F. (2014). Inferential Motion Control: Identification and Robust Control Framework for Positioning an Unmeasurable Point of Interest. In: IEEE Transactions on Control Systems Technology, vol. 23, no. 4.

Oomen T, van de Wal M. & Bosgra O. (2007). Design framework for high-performance optimal sampled-data control with application to a wafer stage. In: International Journal of Control, 80(6), 919-934.

Oomen T, Van De Wijdeven J, & Bosgra O. (2009). Suppressing intersample behavior in iterative learning control. In: Automatica, 45(4), 981-988.

Orlowska-Kowalska T. & Szabat K. (2007). Neural-Network Application for Mechanical Variables Estimation of a Two-Mass Drive System. In: IEEE Transactions on Industrial Electronics, 54(3):1352–1364.

Palumbo F. et al. (2019). Hardware/software self-adaptation in CPS: the CERBERO project approach. In: International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), pp. 416–428. doi:10.1007/978-3-030-27562-4_30.

Palumbo F., Sau C., Fanni T. & Raffo L. (2017). Challenging CPS trade-off adaptivity with coarse-grained reconfiguration. In: Applications in Electronics Pervading Industry, Environment and Society (APPLEPIES), pp. 57–63. doi:10.1007/978-3-319-93082-4_8.

Pan Z, Zhang H, Zhu Z & Wang J (2006). Chatter Analysis of Robotic Machining Process. In: Journal of Materials Processing Technology 173:301–309.

Pannocchia G, Gabiccini M. & Artoni, A. (2015) Offset-free MPC explained: novelties, subtleties, and applications. In: IFAC-PapersOnLine 48-23.

Paszke A. et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035.

Pipeleers G, Demeulenaere B, De Schutter J. & Swevers J. (2008). Robust high-order repetitive control. 2008 American Control Conference.

Ponce I.U, Orlov Y, Aguilar L.T. & Alvarez J. (2015). Robust tracking control of servo systems with backlash: Nonsmooth Hinf control vs. linear Hinf control. In: Proc. Amer. Control Conf., pp. 20512056.

Qi, Q. et al. (2021). Enabling technologies and tools for digital twin. In: Journal of Manufacturing Systems 58: 3-21.

Ramsauer R, Kiszka J, Lohmann D, & Mauerer W. (2017). Look mum, no VM exits!(almost). In: arXiv preprint arXiv:1705.06932.

Rasmussen C. E. (2003). Gaussian processes in machine learning. In Summer school on machine learning. (pp. 63-71). Springer, Berlin, Heidelberg.

Ray L.R, Ramasubramanian A. & Townsend J. (2001). Adaptive friction compensation using extended Kalman-Bucy Filter friction estimation. In: Control Engineering Practice.

Roberts R. (1998). Control of high-rise/high-speed elevators. In: Proceedings of the American Control Conference, 6, 3440–3444. https://doi.org/10.1109/ACC.1998.703233.

Robinson R.M. et al. (2016). Nonlinear control of robotic manipulators driven by pneumatic artificial muscles. In: IEEE/ASME Trans. Mechatron., 21(1), 55–68.

Rodriguez-Ayerbe P, Dumur D. & Lavernhe S. (2014). Axis control using model predictive control: identification and friction effect reduction. 3rd International Conference on Virtual Machining Process Technology.

Roesmann C, et al. (2012). Trajectory modification considering dynamic constraints of autonomous robots. In: ROBOTIK 2012; 7th German Conference on Robotics, pp. 1-6.

Rugh W.J. & Shamma J.S. (2000). Research on gain scheduling. In: Automatica, 36(10), 1401-1425

Rus D. & Tolley M.T. (2015) Design, fabrication and control of soft robots. In: Nature, 521(7553), 467–475.

Sahoo S.K, Panda S.K, & Xu J.X. (2007). Application of spatial iterative learning control for direct torque control of switched reluctance motor drive. In: 2007 IEEE Power Engineering Society General Meeting, PES.

Santo D. R. et al. (2016). On nonlinear horizontal dynamics and vibrations control for high-speed elevators: In: *Https://Doi.Org/10.1177/1077546316667324*, *24*(5), 825–838. https://doi.org /10.1177/1077546316667324.

Sato M. (2008) Inverse system design for LPV systems using parameter-dependent Lyapunov functions. In: Automatica, 44(4), 1072-1077.

Saudo I. et al. (2020). The Key Role of Memory in Next-Generation Embedded Systems for Military Applications. In: Proceedings of 6th International Conference in Software Engineering for Defence Applications, Advances in Intelligent Systems and Computing, pages 275–287, Cham. Springer International Publishing.

Sawyer (2022) Available at: https://www.rethinkrobotics.com/sawyer/ (Accessed at: 22 july 2022)

Schlipf D. et al. (2014). Comparison of linear and nonlinear model predictive control of wind turbines using LIDAR. In: Proceedings of the 2014 American Control Conference.

Schoeberl M. et al. (2015) T-CREST: Time-predictable multi-core architecture for embedded systems. Journal of Systems Architecture, vol. 61, no. 9, pp. 449–471, 2015.

Scordino C. et al. (2020). Real-time virtualization for industrial automation. In: 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA).

Schwenzer M, Ay M, Bergs T. & Abel D. (2021) Review on model predictive control: an engineering perspective. In: The International Journal of Advanced Manufacturing Technology.

Serkies P.J. & Szabat K. (2013) Application of the MPC to the position control of the two-mass drive system. In: IEEE Trans. Ind. Electron., vol. 60, no. 9, pp. 3679-3688.

Sharon G, Stern R, Felner A, Sturtevant N.R. (2015) Conflict-based search for optimal multi-agent pathfinding. In: Artificial Intelligence, Volume 219, pp. 40-66.

Smith O.J.M. (1958). Feedback Control Systems. In: McGraw-Hill, New York

Singer N.C, & Seering W.P. (1990). Preshaping command inputs to reduce system vibration. In: Journal of Dynamic Systems, Measurement, and Control, 123, 112(1), 76–82

Singh T. & Vyhlídal T. (2020). Recent Results in Reference Prefiltering for Precision Motion Control. IFAC-PapersOnLine 53

Singhose W. (2009). Command shaping for flexible systems: A review of the first 50 years. In: International Journal of Precision Engineering and Manufacturing, 128, 10(4), 153–168.

Sinner M. & Pao L.Y. (2020) Revisiting disturbance accommodating control for wind turbines. In: The Science of Making Torque from Wind (TORQUE 2020).

Steinbuch M. (2002) Repetitive control for systems with uncertain period-time. In: Automatica , Vol 38.

Steinbuch M, Weiland S. & Singh T. (2007). Design of noise and period time robust high-order repetitive control, with application to optical storage. In: Automatica 43(12).

Stephens M.A, Manzie C. & Good M.C (2013). Model Predictive Control for Reference Tracking on an Industrial Machine Tool Servo Drive. In: IEEE Transactions on Industrial Informatics.

Stol K.A. & Fingersh L.J. (2003) Wind Turbine Field Testing of State-Space Control Designs.

Sun Z, Pritschow G, Zahn P. & Lechler A. (2018). A novel cascade control principle for feed drives of machine tools. In: CIRP Ann., vol. 67, no. 1, pp. 389-392.

Szabó Z, Bokor J. & Balas G. (2003) Inversion of LPV systems. In: Mediterranean Control Conference (Vol. 400, pp. T7-083).

Tang M, et al. (2016) Enhanced DBCC for high-speed permanent magnet synchronous motor drives. In: IET Power Electronics. 9.

Tang M, Zanchetta P, Gaeta A. and Formentini A. (2016). A Variable Frequency Angle-Based Repetitive Control for Torque Ripple Reduction in PMSMs.

Tang M, Formentini A, Odhano S.A. & Zanchetta P. (2020). Torque Ripple Reduction of PMSMs Using a Novel Angle-Based Repetitive Observer. In: IEEE Transactions on Industrial Electronics, vol. 67, no. 4, pp. 2689-2699.

Thomsen S, Hoffmann N. & Fuchs F.W. (2011). PI Control, PIBased State Space Control, and Model-Based Predictive Control for Drive Systems With Elastically Coupled Loads; A Comparative Study. In: IEEE Transactions on Industrial Electronics, 58(8):3647–3657.

Tomizuka M, Tsao T.-C. & Chew K.-K. (1989). Analysis and synthesis of discrete-time repetitive controllers. In: Journal of Dynamic Systems, Measurement, and Control, vol. 111(3).

Tóth R. (2010). Modeling and Identification of Linear Parameter-Varying Systems. In: Lecture Notes in Control and Information Sciences (Vol. 403). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-13812-6.

Tóth R, Heuberger P.C. & Van den Hof P.J. (2007). LPV system identification with globally fixed orthonormal basis functions. In: 2007 46th IEEE Conference on Decision and Control (pp. 3646-3653). IEEE.

Tóth R, Laurain V, Zheng W. X, & Poolla K. (2011). Model structure learning: A support vector machine approach for LPV linear-regression models. In: 2011 50th IEEE Conference on Decision and Control and European Control Conference (pp. 3192-3197). IEEE.

Ubach-Pallas S.X. (2017). Modelling and optimization-based control design for power consuming reduction in industrial processes. In: Master's Degree in Automatic Control and Robotics, Escola Técnica Superior de Enginyeria Industrial de Barcelona.

Van den Broeck L. et al. (2008) A linear programming approach to design robust input shaping. In: Proceedings of the 10th international workshop on advanced motion control.

Van Geffen V. (2009). A study of friction models and friction compensation.

Van Haren M. et al. (2022) Gaussian Process Position-Dependent Feedforward: With Application to a Wire Bonder. In: 2022 IEEE 17th International Conference on Advanced Motion Control (AMC), pp. 268-273, doi: 10.1109/AMC51637.2022.9729327.

Van Zundert J. et al. (2018). Beyond performance/cost tradeoffs in motion control: A multirate feedforward design with application to a dual-stage wafer system. In: IEEE Transactions on Control Systems Technology, 28(2), 448-461.

Volpe B.T. et al. (2009). Robotic devices as therapeutic and diagnostic tools for stroke recovery. In: Archives Neurol., 66(9), 1086–1090.

Vujicic V. P. (2012). Minimization of Torque Ripple and Copper Losses in Switched Reluctance Drive. In: IEEE Transactions on Power Electronics, *27*(1), 388–399.

Wada K. & Shibata T. (2007). Living with seal robots—Its socio-psychological and physiological influences on the elderly at a care house. In: IEEE T. Robot., 23(5), 972–980.

Wang N, Wright A.D. & Johnson K.E. (2016). Independent Blade Pitch Controller Design for a Three-Bladed Turbine Using Disturbance Accommodating Control.

Wang J. J. (2016). A common sharing method for current and flux-linkage control of switched reluctance motor. In: Electric Power Systems Research, *131*, 19–30.

Wang S. et al. (2020). Deep learning aided dynamic parameter identification of 6-DOF robot manipulators. In: IEEE Access 8**,** 138102-138116.

Wang, Y. & Chirikjian G.S. (2000) A New Potential Field Method for Robot Path Planning. In: Proceedings of the 2000 IEEE International Conference on Robotics & Automation, San Francisco, CA, pp. 977-982.

Wang Y, Gao F & Doyle F. J. (2009). Survey on iterative learning control, repetitive control, and run-to-run control. In: Journal of Process Control, 19.

Weiss G, Zhong Q.-C, Green T.C. & Liang J. (2004) H1 repetitive control of DC-AC converters in microgrids. In: IEEE Transactions on Power Electronics, vol. 19(1).

Wright A.D, Fleming P. & van Wingerden J.W. (2011). Refinements and Tests of an Advanced Controller to Mitigate Fatigue Loads in the Controls Advanced Research Turbine.

Xi S, Wilson J, Lu C. & Gill C. (2011) RT-Xen: Towards real-time hypervisor scheduling in Xen. In: Proceedings of the ninth ACM international conference on Embedded software, pages 39–48. ACM.

Xi S, et al. (2014) Realtime multi-core virtual machine scheduling in xen. In 2014 International Conference on Embedded Software (EMSOFT), pages 1–10. IEEE.

Yamamoto Y. (1993). Learning Control and Related Problems in Infinite-Dimensional Systems. In: W. J. HL, Essays on Control. Perspectives in the Theory and its Applications (p. 191–222).

Yan M, Yang Z, Liu L, Li S. (2012) Prodfa: Accelerating domain applications with a coarse-grained runtime reconfigurable architecture. In: 2012 IEEE 18th International Conference on Parallel and Distributed Systems (ICPADS), pp. 834–839. doi:10.1109/ICPADS.2012.136.

Yodaiken V. et al. (1999). The rtlinux manifesto. In Proc. of the 5th Linux Expo.

Yong K.H. & Narendra A. (1992). A Potential Field Approach to Path Planning. In: IEEE Transactions on Robotics and Automation, vol. 8, no. 1, pp. 23-31.

Yuan Y, Auger F, Loron L, Moisy S. & Hubert M. (2012). Torque Ripple Reduction in Permanent Magnet In: Synchronous Machines Using Angle-Based Iterative Learning Control. 10.1109/IECON.2012.6388853.

Zhang G. & Furusho J. (2000). Speed control of two-inertia system by PI/PID control. In: IEEE Transactions on Industrial Electronics, 47(3):603–609.

Zhang J. et al. (2011) Performance analysis towards a kvm-based embedded real-time virtualization architecture. In: 5th International Conference on Computer Sciences and Convergence Information Technology, pages 421–426. IEEE.

Zhou L. & Jinhua S. (2018). Generalized-extended-state-observer-based repetitive control for MIMO systems with mismatched disturbances. In: IEEE Access, vol. 6.

Zhuang Z. et al. (2018). Discrimination-aware channel pruning for deep neural networks. In: Advances in neural information processing systems, 31.

Zimmer M, Broman D, Shaver C, & Lee E.A. (2014) Flexpret: A processor platform for mixed-criticality systems. In: IEEE 19th RTAS. IEEE, pp. 101–110.