University of the Basque Country (UPV/EHU)

Universidad del País Vasco
Euskal Herriko Unibertsitatea

Systems Engineering and Automation Department
Intelligent Control Research Group

Iterative learning control in the commissioning of industrial presses

Ignacio Trojaola Bolinaga

1. Supervisor Iker Elorza Pinedo
Control and Monitoring
Ikerlan Technology Research Center

2. Supervisor Eloy Irigoyen Gordo
Intelligent Control Research Group
University of the Basque Country

December, 2021

Ignacio Trojaola Bolinaga
Iterative learning control in the commissioning of industrial presses
Supervisors: Iker Elorza Pinedo and Eloy Irigoyen Gordo

University of the Basque Country (UPV/EHU)
Intelligent Control Research Group
Systems Engineering and Automation Department
Plaza Ingeniero Torres Quevedo, 1
48013, Bilbao (Bizkaia)

Abstract

This thesis presents solutions to the control problems that exist nowadays in industrial presses, followed by a discussion of the most appropriate control schemes that may be used for their solution. Iterative Learning Control is subsequently analyzed, as the most promising control scheme for machine presses, due to its capability to improve the performance of a system that operates repeatedly.

A novel Iterative Learning Control design is presented, which makes use of the dynamic characteristics of the system to improve the current controller performance and stability. This, results in an adaptation of the presented Iterative Learning Control design to two use cases: the single-input-single-output force control of mechanical presses and the multiple-input-multiple-output position control of hydraulic presses. While existing Iterative Learning Control approaches are also described and applied to the previously mentioned use cases, the presented novel approach has been shown to outperform the existing algorithms in terms of control performance.

The proposed Iterative Learning Control algorithms are validated in an experimental hydraulic test rig, in which the performance, robustness and stability of the algorithm have been demonstrated.

Although the presented Iterative Learning Control algorithms have been proved in industrial presses, the design methods that are addressed in this thesis can be applied in many other industrial systems, which make the contributions proposed in this thesis be transferable to other actual problems.

Resumen

Esta tesis presenta soluciones a los problemas de control que existen hoy en día en las prensas industriales, seguida de una discusión de las técnicas de control más adecuadas que se pueden emplear para su solución. Posteriormente, se analiza el Control por Aprendizaje Iterativo, como la técnica de control más adecuada para las prensas industriales, debido a su capacidad para mejorar el rendimiento de un sistema que opera repetidamente.

Se presenta un nuevo diseño de Control por Aprendizaje Iterativo, que hace uso de las características dinámicas del sistema para mejorar el rendimiento y la estabilidad del controlador. Se presentan dos casos de uso para la validación del diseño de control propuesto: el control de fuerza de entrada única, salida única de prensas mecánicas y el control de posición de entrada múltiple, salida múltiple de prensas hidráulicas. Si bien también se explican los métodos de Control por Aprendizaje Iterativo existentes hoy en día, también se aplican dichos métodos a los dos casos de uso mencionados anteriormente. Se ha demostrado que el diseño propuesto en esta tesis supera a los algoritmos existentes en términos de rendimiento de control.

Los algoritmos de Control por Aprendizaje Iterativo propuestos se validan en un banco de pruebas hidráulico, en el que se ha demostrado el rendimiento, la robustez y la estabilidad de los algoritmos.

Si bien los algoritmos de Control por Aprendizaje Iterativo presentados han sido probados en prensas industriales, los métodos de diseño que se abordan en este trabajo se pueden aplicar en muchos otros sistemas industriales, lo que hace que los aportes propuestos en esta tesis sean transferibles a otros problemas reales.

Laburpena

Tesi honek gaur egun prentsa industrialetan topa ditzakegun kontrol-arazoei irtenbideak aurkezten dizkie. Horren harira, horiek konpontzeko erabil daitezkeen kontrol teknika egokienak eztabaidatzen dira lanean zehar. Ondoren, Ikaskuntza Iteratiboaren Kontrola aztertzen da etorkizun handiko kontrol teknika bezala, modu errepikakor batean funtzionatzen duen sistema baten errendimendua hobetzeko duen gaitasunagatik.

Ikaskuntza Iteratiboko Kontrolaren diseinu berri bat aurkezten da, sistemaren ezaugarri dinamikoak erabiltzen dituena egungo kontroladorearen errendimendua eta egonkortasuna hobetzeko. Honen ondorioz, aurkeztutako Ikaskuntza Kontrol Iteratiboaren diseinua bi erabilera-kasuetara egokitzen da: prentsa mekanikoen sarrera bakarreko-irteera bakarreko indar kontrolerako eta prentsa hidraulikoen sarrera anitzeko-irteera anitzeko posizio kontrolerako. Orain arte bibliografian proposatu diren Ikaskuntza Kontrol Iteratiboaren teknikak lehen aipatutako erabilera-kasuetara ere aplikatzen dira. Kontrol-errendimenduari dagokionez, tesi honetan aurkeztutako Ikaskuntza Kontrol Iteratiboaren diseinuak lehendik dauden algoritmoak gainditzen dituela frogatzen da. Proposatutako Ikaskuntzaren Kontrol Iteratiboaren algoritmoak banku hidrauliko esperimental batean balioztatu dira, non algoritmoen errendimendua, sendotasuna eta egonkortasuna frogatu diren.

Aurkeztutako Ikaskuntzaren Kontrol Iteratiboaren algoritmoak prentsa industrialetan frogatu badira ere, tesi honetan jorratzen diren diseinu-metodoak beste industria-sistema askotan aplika daitezke. Honek balioztatzen ditu tesi honetan proposatzen diren ekarpenak, beste arazo batzuetara transferitu daitezkeelako.

Acknowledgement

The author would like to thank the Department of Development and Infrastructures of the Government of the Basque Country for its support via Industrial Doctorate Program BIKAINTEK, under grant agreement 20-AF-W2-2018-00015 (Official Bulletin of the Basque Country nº67 on 09/04/18).

Additionally, the author would like to thank FAGOR ARRASATE for its financial support and for having provided indispensable industrial equipment, from which this thesis has been benefited to enhance the interaction between the industry and academia.

Contents

Abstract

AS	Asymptotic Stability
BDC	Bottom Dead Center
D-type	Derivative-type
FF	Feed Forward
FL	Feedback Linearization
FP	Force Percentage
ILC	Iterative Learning Control
IPC	Industrial PC
IRT	Isochronous Real Time
MIMO	Multiple Input Multiple Output
MI-ILC	Model Inverse Iterative Learning Control
MIC-ILC	Model Inverse Controller Iterative Learning Control
ML	Machine Learning
MS	Monotonic Stability
PI	Proportional Integral
PID	Proportional Integral Derivative
PLC	Programmable Logic Controller
PP	Pressure Percentage
P-type	Proportional type
PD-type	Proportional Derivative type
PILC	Proportional Iterative Learning Control
POILC	Parameter Optimal Iterative Learning Control
PSD	Power Spectral Density
RL	Reinforcement Learning
RMSE	Root Mean Square Error
RT	Real Time
SIMO	Single Input Multiple Output
SISO	Single Input Single Output
TDC	Top Dead Center
TRL	Technology Readiness Level
ZPF	Zero Phase Filtering

1 Introduction

> *Sometimes I'll start a sentence and I don't even know where it's going. I just hope I find it along the way.*
>
> — **Michael Scott**
> (The Office)

1.1 Motivation

In the context of metal forming, press machines have been used for decades to transform a piece of metal into any form or shape. Indeed, as industry automation has grown, most production lines have made use of press machines, as their quick processing and easy maintenance make them ideal for mass production.

Nowadays, modern press machines manufacturers face increasingly challenging specifications as, for the best economy in press operations, the press should have as high accuracy as possible during the working operation. The high accuracy must be preserved during the force application of the press to the workpiece, where a specific force has to be provided to guarantee the correct forming of the workpiece.

If a press controller cannot meet the workpiece design requirements, this inevitably will lead to higher costs and longer lead-times, as the workpiece would have to be discarded. However, providing a fully functional press controller that satisfies every design requirement, will often result in high productivity and growth for businesses due to the technology improvement.

At this point, is where the control challenge arises. The force applied to the workpiece varies extremely fast, due to the high speeds involved in the press operation, which requires fine-tuned controllers to be able of controlling the force correctly. In order to control the force applied to the workpiece, a feedback controller and the pressure feedback are commonly used in the press operation. Feedback controllers generally provide good response in

rejecting slow disturbances, however, when the disturbance is very fast, the controller is not effective as the disturbance is faster than the controller can respond, which could cause a considerable force overshoot, not satisfying the force requirements.

The feedback controller is well known in industry, and many engineers are trained in their use. While it is easy to design and tune for simple applications, for more challenging tasks, its tuning turns cumbersome and laborious. In press machines, the tuning of the feedback controllers is carried out in the commissioning. This process could take several weeks, as the press operator must ensure that the press specifications are fulfilled for the entire operating envelope.

Furthermore, if the workpiece design is modified during the press operation, the feedback controller tuning should be modified accordingly, starting the commissioning process from scratch, as the controller performance could no longer be suitable. One can imagine how delicate this process could turn, and would think of an alternative control that reduces the commissioning time of the press and that adapts automatically to every workpiece design scenario.

When proposing a new control method for the industry, however, one should be cautious, as there exist a number of requirements that need to be satisfied before it is implemented. As any controller oriented to industry, it should most likely be implemented in an industrial controller and it must fit into a programming environment of a suitable programmable logic controller (PLC). This means that the control scheme complexity and the real-time computation power will be limited. Furthermore, the industry is usually reluctant to new control methodologies, which could bring complexity to their daily routine. Removal of the well-known and easy-to-tune feedback controller may not be easy to accept.

Therefore, it is necessary to propose a controller that goes along with the feedback controller, easy to tune, with a light computation burden and, above all, capable of improving the existing control performance and fulfilling the control specifications.

Iterative Learning Control (ILC) techniques have been successfully implemented to solve a wide variety of industrial control-engineering problems, e.g. mechanical systems such as robot arm manipulators, chemical processes such as batch reactors, as well as rotatory systems such as synchronous motors [1]. When such systems are operated repeatedly, ILC can improve the systems' performance from one iteration to the next.

ILC is known for its structural simplicity, as its implementation is an enhancement of the feedback controller, not a replacement. It is attractive to industry as its commissioning can be carried out without requiring expert knowledge by the operator, always guaranteeing its stability and good performance.

Press machines are ideal for ILC, as they perform the same task over and over again during their production. The considerations above lead to the following statement:

What can the press control system learn and obtain from the repeated task to improve the performance of the control systems?

Through this thesis we will demonstrate how ILC can help us to improve the performance of press machines, and show that they can be more competitive with the help of machine learning.

1.2 Objectives

This thesis provides solutions to the control problems that can be found in the everyday operation of mechanical and hydraulic presses. Although both presses are aimed at the forming of workpieces, their operation is different and, therefore, their control challenges differ as well.

On the one hand, in mechanical presses, in order for the companies to provide the best solution for the production of high volumes of workpieces, the force control requirements must be fulfilled accurately. These specifications typically relate to the maximum force peak allowed in the hydraulic circuit and the settling time of the force signal. Through the thesis we show that when the customer sets too demanding specifications, the fulfillment of them with the press actual feedback controllers turns into a laborious task. Therefore, in order to satisfy every specification set by the customer, we propose to combine the actual feedback controllers with an extra feed-forward signal, based on ILC. Making use of the repetitive behavior of mechanical presses, the ILC algorithm will learn which input signal will be adequate to fulfill the desired specifications.

The proposed ILC algorithm should be simple, in the terms of ease of maintenance and reliability. Furthermore, the required time to learn the optimal input is of great importance. The learning should be as fast as possible, so the commissioning time of the controller is reduced, which translates into a reduction of the production costs.

On the other hand, in hydraulic presses, the control challenges arise in the position control. Typically the position control is carried out with predefined control signals to the actuators, which have been previously tested for a specific position trajectory. However, when the position trajectory is modified, due to variations in the workpiece design or hydraulic modifications, the predefined signals have to be calculated from scratch. This is a tedious task that takes a great amount of time. Furthermore, while the new signals are being designed the hydraulic press is inoperative, reducing the benefits.

We propose to automate the position control based on the ILC algorithm. However, this control approach differs from the force control in mechanical presses, as in order to provide accurate position control two variables will need to be controlled with two different actuators, yielding a multiple-input-multiple-output (MIMO) control. In the same way as in the force control problem, we need to guarantee algorithm simplicity while providing fast learning. Therefore, the challenge remains to design a stable and fast MIMO ILC algorithm for the hydraulic presses.

1.3 Research outcomes

Regarding the scientific contributions generated during the research, the following list displays the articles that have been published and include contributions of the author.

(a) Trojaola I., Elorza I., Irigoyen E., Pujana A., Calleja C. (2020) Iterative Learning Control for a Hydraulic Cushion. In 14th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2019), Sevilla. Advances in Intelligent Systems and Computing, vol 950. Springer, Cham. `https://doi.org/10.1007/978-3-030-20055-8_48`

(b) Trojaola I., Elorza I., Irigoyen E., Pujana A., Calleja C. (2019) Hydraulic Press Commissioning Cost Reductions via Machine Learning Solutions. 10th EUROSIM Congress on Modelling and Simulation Logroño, La Rioja, Spain.

(c) Trojaola, I., Elorza, I., Irigoyen, E., Pujana-Arrese, A., & Calleja, C. (2020). The Effect of Iterative Learning Control on the Force Control of a Hydraulic Cushion. Logic Journal of the IGPL. `https://doi.org/10.1093/jigpal/jzaa056`

(d) Trojaola, I., Elorza, I., Irigoyen, E., Pujana-Arrese, A., & Calleja, C. (2020). Iterative Learning Control and Gaussian Process Regression for Hydraulic Cushion Control. IFAC-PapersOnLine, 53(2), 1421-1426. https://doi.org/10.1016/j.ifacol.2020.12.1909

(e) I. Trojaola, I. Elorza, E. Irigoyen, A. Pujana-Arrese and G. Sorrosal, "An Innovative MIMO Iterative Learning Control Approach for the Position Control of a Hydraulic Press," in IEEE Access, vol. 9, pp. 146850-146867, 2021. https://doi.org/10.1109/ACCESS.2021.3123668

1.4 Organization of the thesis

The thesis is organized in five main chapters. Chapter 1 and 7 are dedicated to the introduction and conclusions, respectively.

Chapter 2 introduces the existing limitations in the force control of mechanical presses and discusses alternative control approaches in order to improve it. This lays to a machine learning control proposal based on ILC for the hydraulic cushion control problem.

Chapter 3 presents a formal definition of ILC applied to the hydraulic cushion control problem. First, a model-based design to improve the convergence rate of the ILC is presented, which is accompanied by a graphical analysis for the algorithm stability. The proposed ILC algorithm is validated under a nonlinear high-fidelity model of a mechanical press in Matlab/Simulink and the results are then compared to the most common ILC algorithm used in the literature.

Chapter 4 introduces the existing limitations in the position control of a hydraulic press, discussing the actual control approaches used in industry. As an alternative to the existing hydraulic press control, a new MIMO position control is proposed, so the hydraulic press operation can be automated and the position control improved. The proposed control approach is not straightforward, as the hydraulic press control loops are coupled, therefore ILC is used to automatically learn suitable control actions.

Chapter 5 introduces the proposed MIMO ILC design for the position control of hydraulic presses. The same design approach as in Chapter 3 is followed, however, it is extended to the MIMO case. A new graphical evaluation approach has been introduced to intuitively analyze the stability condition of

a MIMO ILC algorithm. The algorithm is validated under a nonlinear high-fidelity hydraulic press model in Matlab/Simulink, and it is subsequently compared to the most common MIMO ILC algorithms in the literature.

Chapter 6, shows the experimental implementation of the two ILC algorithms proposed in Chapters 3 and 5. Two scenarios are presented, a hydraulic test rig and a Digital Twin platform. In the hydraulic test rig, the ILC algorithms have been tested under realistic scenarios that can arise in the working operation of hydraulic and mechanical presses. In the Digital Twin platform, the ILC algorithm has been tested on a real press controller algorithm under different conditions.

2

Mechanical press force control system

2.1 Introduction

A mechanical press is a device that translates the rotational force of a motor into a translational force that performs the pressing operation. A mechanical press is ideal for fast and quickly repeatable applications of force over a limited distance, as the momentum generated by the motor allows high-production rates and accuracy.

Figure 2.1 shows the drawing process of a standard mechanical press. The major parts are pointed out in the figure: the mechanical slide, the hydraulic cushion and the die. The slide comprises one or more motors connected to cranks and connecting rods, or a variety of similar mechanisms, which move the slide in a typically sinusoidal up-and-down motion. During this motion, the slide impacts the cushion to aid in the forming of the workpiece, by regulating the force between some parts of the workpiece and the upper die, which forces the workpiece against the lower die to give it the desired shape.

The left picture shows the mechanical press before the working cycle has begun. The workpiece is placed on the lower die. Both cushion and slide are placed at the initial position, at the top dead center (TDC). The hydraulic cylinders of the cushion circuit are completely extended.

The right picture shows the mechanical press once the workpiece forming operation is finished. The workpiece shape is given by the upper and lower die parts, and both slide and cushion are at the bottom dead center (BDC). The hydraulic cylinders of the cushion circuit are completely retracted.

Figure 2.2 shows the typical working operation of a mechanical press. At the beginning, both slide and cushion start at the TDC. The slide accelerates and starts the falling motion. At the impact the *Drawing* phase begins, where the slide velocity is at its maximum. The main purpose of the cushion is to control the force between some parts of the workpiece and the upper die,

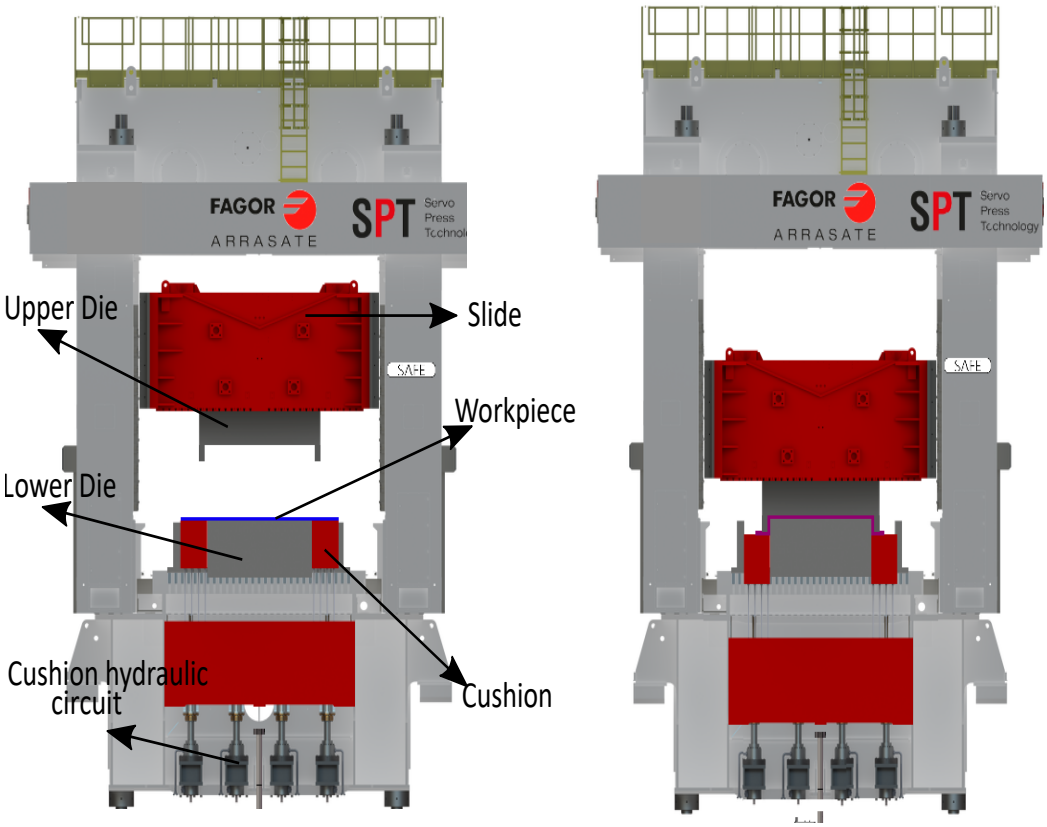


Fig. 2.1: Mechanical press drawing, with mechanical slide and hydraulic cushion.

typically to control the slipping of the sheet metal as it is sucked in by the drawing. When slide and cushion reach the BDC the *Drawing* phase finishes, and both slide and cushion are returned to their respective TDC, to start the next operation.

One of the major concerns over the last years has been to control the cushion force, necessary for guaranteeing the correct forming of the workpiece. For the best workpiece forming, the drawing force control should have as fast a response time and as high an accuracy as possible during the forming of the workpiece.

A typical hydraulic cushion force control system is shown by Fig. 2.3, which consists of a single-acting cylinder, a proportional valve and an axial piston pump. The external velocity into the cylinder rod represents the velocity that the slide induces on the cushion during the stroke. This velocity is regarded as an external disturbance, which should be counteracted in order to achieve precise force control.

The cylinder rod is connected to the cushion and, during the *Drawing* phase, as cushion and slide move down together, the piston retracts and the pressure

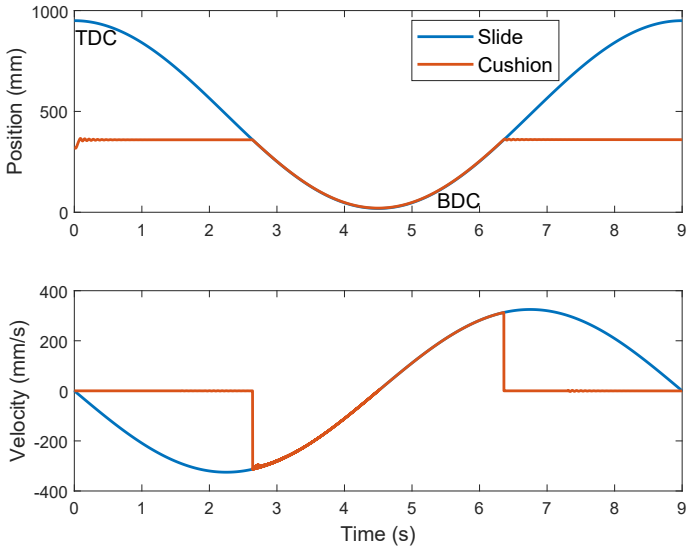


Fig. 2.2: Slide and cushion position and velocity for one press cycle.

inside the cylinder increases. During the retraction the fluid inside the cylinder chamber is channeled through the proportional valve into the tank, from port A to port T. By modifying the proportional valve opening ratio, the pressure inside the cylinder chamber can be regulated, so the desired pressure is achieved.

The cylinder pressure is the variable to control, as the force can be precisely calculated by knowing the cylinder chamber area and pressure. Therefore, although it will be referred to as force control, the pressure inside the cylinder will be controlled.

We can look at the dynamic equations of the cylinder and the proportional valve to analyze further the force control problem. The volumetric flow rate into the cylinder port and the piston motion given by [2] is shown in the following equation:

$$q_A(t) = A_A\dot{x}(t) + (V_{\text{Ad}} + A_A x(t))\beta\dot{P}_A(t), \tag{2.1}$$

where $q_A(t)$ is the volumetric flow rate into the cylinder chamber (m^3/s), A_A is the cylinder chamber area (m^2), $x(t)$ is the piston position (m), V_{Ad} is the cylinder chamber dead volume (m^3), β is the hydraulic fluid compressibility (1/bar) and $P_A(t)$ is the pressure in the cylinder chamber (bar).

The hydraulic compressibility is the measure of the amount of fluid volume reduction when pressure is applied to said fluid. The first term in (2.1) determines the flow rate change depending on the cylinder velocity. The

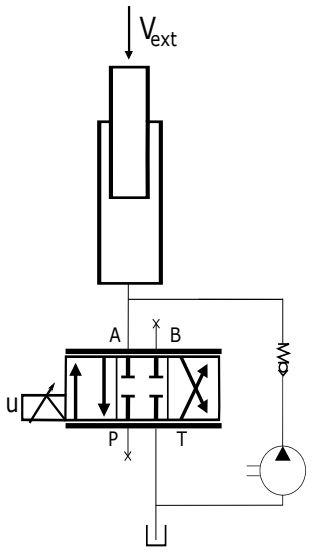


Fig. 2.3: Hydraulic cushion force control circuit.

second term determines the flow rate change depending on the total volume compression at a specific pressure change rate.

The position and velocity in (2.1) are given by the slide movement, and change continuously during the working operation of the press. Once a specific constant pressure, $P_A(t)$, is obtained in the cylinder chamber, $\dot{P}_A(t)$ will be zero and the volumetric flow rate will be proportional to the velocity.

In the cushion hydraulic circuit shown by Fig. 2.3, the cylinder chamber pressure can only be controlled by means of the proportional valve. Modifying the proportional valve spool position the desired $P_A(t)$ is obtained, so that the drawing of the workpiece is carried out correctly.

From Fig. 2.3, the pressure at valve port A will be equal to the cylinder chamber pressure. In the same way, the flow into valve port A will be equal to the flow out of the cylinder. Therefore, by controlling the pressure at valve port A we can control the cylinder pressure. The relationship between the pressure at the port A and the valve spool position is as follows [2]:

$$P_A(t) = \frac{q_A^2(t)\Delta P_{ref}}{q_{ref}^2 K_{\mathrm{v}}(y_{\mathrm{v}}(t))^2}, \tag{2.2}$$

where $y_{\mathrm{v}}(t)$ is the spool position, $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ is the hydraulic conductance ($\mathrm{m}^3/(\mathrm{bar}\cdot\mathrm{s})$), q_{ref} is the nominal flow rate of the valve (m^3/s) and ΔP_{ref} is the nominal pressure of the valve (bar).

ΔP_{ref} and q_{ref} are included to normalize $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ between the [-1, +1] interval. The hydraulic conductance, $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, is a function of the valve spool position, $y_{\mathrm{v}}(t)$, it is nonlinear and often obtained via empirical tests.

During the *Drawing* phase, the fluid in the cylinder chamber will be channeled through the valve to the tank. Assuming that $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ is linear, i.e. $y_{\mathrm{v}}(t) = K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, from (2.2), if the spool position is decreased the cylinder pressure will raise, conversely, if the spool position is increased the cylinder pressure will decrease.

The target cushion force varies depending on the desired workpiece design, and there exist specifications that need to be fulfilled to ensure a correct forming process, else the workpiece could get damaged. The common specifications regarding the force control are listed below and shown graphically by Fig. 2.4. These specifications will be used through the section as a performance criteria of the designed controllers:

- The force signal overshoot must to be less than 7.5% of the nominal force of the cylinder.
- The force signal undershoot must to be less than 5% of the nominal force of the cylinder.
- The force signal must settle within a $\pm$2.5% of the nominal force before the cushion has descended 40 mm.

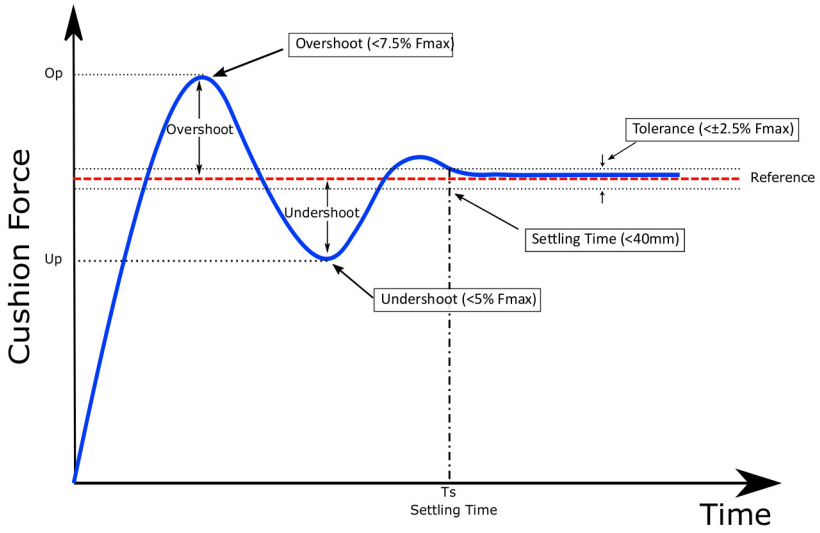


Fig. 2.4: Cushion force specifications during the *Drawing* phase.

The force specifications depicted above vary depending on the workpiece design, and are stipulated by the customer. Therefore, it is of great importance for the press manufacturer to provide a cushion control system that

fulfills the specifications set, else the arrangement with the customer could be lost.

2.2 Traditional hydraulic cushion control

Classical cushion force control relies on PI control, which is widely applied in industrial applications due to its simplicity and robust performance. However, the highly nonlinear behavior of the cushion hydraulic systems affects the performance of the PI controller.

We can analyze the system nonlinearities from the cylinder and valve equations, (2.1) and (2.2), respectively. Matching the cylinder and valve flow rates, we can obtain the relationship between the pressure in the cylinder circuit, the piston velocity and the proportional valve spool position:

$$\dot{P}_A(t) = \frac{K_{\mathrm{v}}(y_{\mathrm{v}}(t))q_{ref}\sqrt{\frac{P_A(t)}{\Delta P_{ref}}} - A_A\dot{x}(t)}{(A_A x(t) + V_{Ad})\beta}. \tag{2.3}$$

From (2.3), we can see that the nonlinearities in the system come from the variables relationship. The relationship between the pressure and its derivative is quadratic, and $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ is nonlinear. Furthermore, the position and velocity that the slide induces on the cushion are sinusoidal and change continuously, as shown by Fig. 2.2. This results in a continuous pressure variation and, thus, of the pressure error, which is the input variable into the PI controller to obtain the desired cylinder pressure.

Before the slide impacts into the cushion, the latter is at a standstill and, on the impact, the cushion quickly accelerates to the slide's falling velocity, as shown by Fig. 2.2. Feedback controllers generally provide good response in rejecting slow disturbances, however, when the disturbance is very fast, the controller is not effective as the disturbance is faster than the controller can respond, resulting in a considerable force overshoot in cushion force control [3]. Intuitively, in order to get a fast controller response, we would design a controller with high gains. However, when the controller gain is high, noise is amplified as well and fed into the system, which could result in system fluctuations.

We could make use of a PID controller to dampen the controller effort and stabilize the force signal at the reference sooner. The derivative term decreases the pressure overshoot but makes the system highly sensitive to

noise. Indeed, it is the noise sensitivity which makes PID controller not suitable for the pressure control in hydraulic circuits. On impact, there is an extremely fast increase in the cylinder chamber pressure with enough noise to render the derivative of that signal unusable.

The PI controller has to deal with the continuous pressure error variation, external disturbances, system nonlinearities and different force operating points, which can result in poor performance. Some authors have defined different PI controller gains for different operation points [4, 5], however, the idea of defining PI controller gains for every force, position and velocity operating point is inconceivable.

Ideally, we would like a fast response controller, so the force reference signal is reached as fast as possible. However, no matter how much we increase the controller gains, the velocity and acceleration of the system depend on the hydraulic compressibility of the circuit, β, which is the ratio at which a hydraulic fluid volume decreases as a result of a pressure increase. It is related to the system's natural frequency which is a function of the inverse of β, the area of the piston and the volume of the fluid.

Ideally, the natural frequency should be at least three to four times the required frequency of acceleration i.e. the frequency at which the cylinder's piston and load accelerate. The system's natural frequency is given by the following formula [2]:

$$\omega_n = \sqrt{\frac{4A_A^2}{VM\beta}}, \tag{2.4}$$

where M is the mass of the load (kg) and V is the total volume of oil trapped between the valve and the piston (m^3).

From (2.4), increasing the cylinder area increases the natural frequency of the cylinder. However, this results in more oil flow rate to pressurize the cylinder chamber, which increases the time required to reach the force reference. Furthermore, higher flow rate requires bigger hydraulic components which usually have worse dynamic behavior. Systems with lower natural frequency can use smaller components which tend to be cheaper, however it will be more challenging to control them. Therefore, one must account for a compromise between system response and circuit dimensionality.

Nowadays, the PI controller tuning process is carried out during the commissioning of the press. In this stage, the plant operators should validate the press control by ensuring that the specifications defined earlier in the project

are fulfilled. This is a tedious and time-consuming process that typically includes several weeks and can involve high costs.

The success of such a controller depends on the fine-tuning of the gains, which usually depends on the operator's ability and experience. Depending on the systems characteristics, it could be challenging to obtain good controller performance, and one may not always be able to find adequate gains. Furthermore, the performance may as well be poor far from the operating point.

The incentive to design a controller that fulfills the force control specifications is, therefore, enormous, since even an efficient control system can achieve astonishing system performance improvements and reduce the commissioning time considerably.

In the subsequent sections, we give an overview of alternative control approaches to traditional hydraulic control.

2.3 Feed-forward control of a hydraulic cushion

The velocity that the slide induces on the cushion on and after the impact is an external disturbance that the PI controller cannot counteract. A common technique for disturbance rejection is feed-forward (FF) control, which based on a system model tries to counteract the disturbance before it occurs [6].

Typically the force feedback has been combined with the velocity FF as a solution to the force control problem. The FF controller relies on comprehensive system knowledge and accurate modeling so, given an inverse model of the hydraulic cushion system, it calculates a proportional valve spool position that rejects the velocity disturbance.

The FF controller design is influenced by model uncertainties, which could reduce the accuracy of the velocity disturbance rejection. This means that the FF needs a feedback control loop in order to get accurate tracking [7]. The FF controller makes the proportional valve open faster to reject the velocity disturbance while the feedback control loop corrects the small deviations.

FF has already been used in hydraulic systems to improve the control performance. In [8], a velocity FF was included in the force control of a hydraulic system. The FF gain was based on the piston area and the servo-valve flow gain, from which a spool position proportional to the velocity was obtained. Incorporating the FF controller considerably improved the system performance.

A FF controller to counteract the servo valve pressure drop in the working operation of a shock actuator was designed in [9]. The FF controller signal was based on the nominal model of the mechanical system with the desired motion as input and the required servo valve command as output. The FF controller obtained better performance than the feedback controller.

Other works, such as [10, 11, 12, 13], have also investigated on the use of FF control in hydraulic systems. However, they are extensions of the works carried out by [8, 9], in which fuzzy models or robustness against model uncertainties are incorporated.

In this thesis, in order to reject the slide velocity disturbance into the cushion, as shown in [8], a velocity FF is designed. To that end, an inverse model of the hydraulic cushion system is derived to obtain the required relationship between the velocity and the valve spool position.

The model inverse for the FF controller has to represent the nonlinear relationship of all the known characteristics of the hydraulic cushion components: hydraulic cylinder and proportional valve. Using the cylinder and valve equations, (2.1) and (2.2), respectively, we can obtain the mathematical model relating the slide velocity to the required valve spool position. Matching the cylinder and valve flow rates, we get the following inverse model:

$$K_{\mathrm{v}}(y_{\mathrm{v}}(t)) = \left(\frac{\dot{P}_A(t)(V_{\mathrm{Ad}} + A_A x(t))\beta + A_A \dot{x}(t)}{\sqrt{P_A(t)}} \right) \frac{\sqrt{\Delta P_{ref}}}{q_{ref}}. \tag{2.5}$$

From (2.5) we do not know the value of the pressure derivative $\dot{P}_A(t)$. Here is where we make use of the feedback controller, which, as we have explained above, supports the FF controller to get accurate tracking. We can set $\dot{P}_A(t)$ to be the output of the PI controller so, if there exist any uncertainties in the system parameters, the deviation will be corrected by the PI controller by modifying $\dot{P}_A(t)$. Actually, the output of the PI controller will be the estimation of $\hat{\dot{P}}_A(t)$, which in general differs from $\dot{P}_A(t)$.

From (2.5) we must obtain the spool position, $y_{\mathrm{v}}(t)$, so it can be directly applied to the valve. However, we have the hydraulic conductance, $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, as output. Therefore, we need to take the inverse of $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, so $y_{\mathrm{v}}(t)$ is obtained as an output. Note that, as explained in Section 2.1, we do not know $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, the inverse will be just an estimation, $\hat{K}_{\mathrm{v}}^{-1}(t)$. The resulting FF controller with the $\hat{\dot{P}}_A(t)$ and $\hat{K}_{\mathrm{v}}^{-1}(t)$ considerations results as follows:

$$\hat{y}_{\mathrm{v}}(t) = \hat{K}_{\mathrm{v}}^{-1} \left(\underbrace{\frac{\dot{\hat{P}}_A(t)(V_{\mathrm{Ad}} + A_A x(t))\beta\sqrt{\Delta P_{ref}}}{\sqrt{P_A(t)}q_{ref}}}_{FL} + \underbrace{\frac{A_A \dot{x}(t)\sqrt{\Delta P_{ref}}}{\sqrt{P_A(t)}q_{ref}}}_{FF} \right). \quad (2.6)$$

The first term corresponds to feedback linearization (FL) and cancels the non-linearities in the system to get an equivalent linear system. It will contribute by correcting the minor errors remaining in the force reference tracking The second term, which corresponds to FF, eliminates the velocity disturbance in the system. When the slide velocity decreases, the FF term will become smaller and the pressure level will be maintained by the FL term, as it depends on the PI controller.

In the estimation of $\hat{y}_{\mathrm{v}}(t)$, the FL term will not be a significant contributor, as β is very small, see Appendix C.0.2. Therefore, if there exist any model uncertainties present in FL, the estimation of $\hat{y}_{\mathrm{v}}(t)$ will still be within an acceptable margin and the deviation will be corrected by the PI controller. However, the velocity disturbance FF term is large and, if it is not eliminated correctly, it will cause large overshoot and settling time.

The resulting block diagram with the FF, FL and the PI controllers is shown on Fig. 2.5.

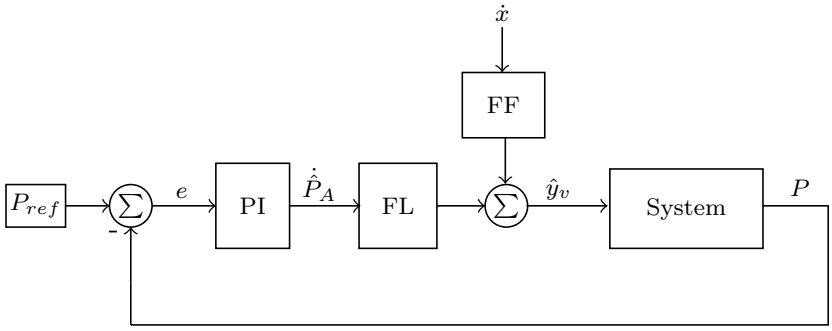


Fig. 2.5: Cushion force control block diagram with FF+FL+PI control.

Apart from the velocity disturbance rejection, the FF and FL controllers insertion in the cushion force control loop, gives an interpretation of how the PI controller gains should be designed. Consider (2.6), during the force control, we can define $P_A(t)$ to be equal to the desired pressure reference, P_{ref}, as the pressure signal will be close to the reference. During the *Drawing* phase, the piston position will change continuously. However, we can define

the piston position to be the mean of the total piston stroke length, $\bar{x}$. As the piston stroke is very small, by considering $x(t)$ as $\bar{x}$, there will not be significant differences.

With the above simplifications in the FF controller design, we can obtain a simplified model of the cushion force control. Assuming that every system parameter cancels out with the introduction of the FF and FL controllers, the cushion force control loop would result as shown by Fig. 2.6.

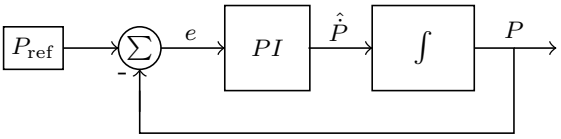


Fig. 2.6: Cushion force control block diagram simplified.

After the simplifications with the FF and FL controller, the resulting plant is an integrator, which relates the PI output with the feedback pressure. With the PI proportional and integral gains being, K_P and K_I, respectively, the resulting closed loop system response is as follows:

$$\frac{P(s)}{P_{ref}(s)} = \frac{K_P s + K_I}{s^2 + K_P s + K_I}. \tag{2.7}$$

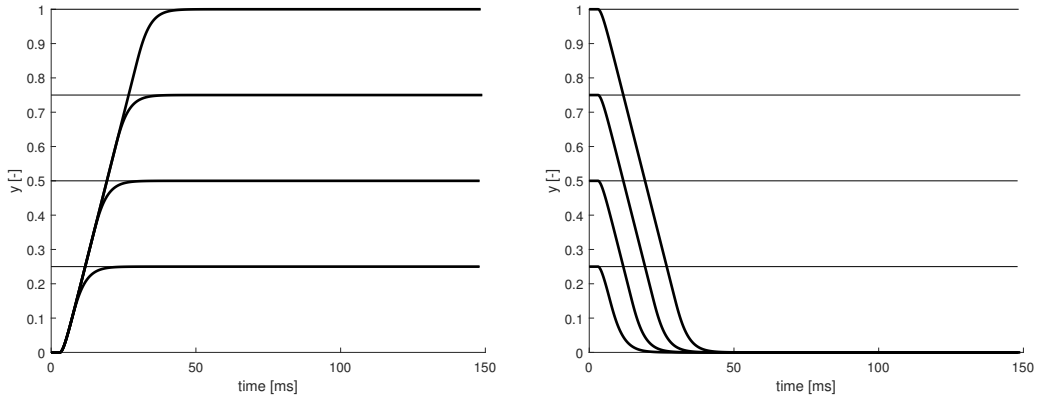
Equation (2.7), leads to an expression we are familiar with to choose appropriate PI gains. Indeed, (2.7) is a second order transfer function with natural frequency $\omega_0 = \sqrt{K_I}$ and with damping $K_P = 2\zeta\omega_0$.

However, in the system simplification we have not considered the system high-frequency dynamics, just the known dynamics. Therefore, this PI gains design is only true for frequencies low enough for the system high-frequency dynamics, such as valve dynamics, to remain negligible. This limitation effectively sets an upper limit for K_I and K_P and a lower limit for the settling time.

With the inclusion of the FF and FL terms, we have established a design procedure for the PI controller, which is based on the desired damping and natural frequency. In the FF and FL design we have not included the valve dynamics as they are unknown. Some authors neglect the effect of these dynamics [14, 15], however, most studies approximate these dynamics with a second-order transfer function, depending on the valve natural frequency and damping coefficient [16, 17, 18]. In the next section, we will analyze how these valve dynamics affect the performance of the FF and FL controllers.

2.3.1 Valve dynamics anticipation

From the FF and FL controllers output, we obtain the valve spool position input. If the design of the FF and FL controllers is correct, the calculated input will counteract the system's external velocity disturbance yielding a precise force control. As explained in the previous section, we have not included the valve dynamics in the FF and FL design, therefore the velocity disturbance will not be completely eliminated. The valve dynamics are usually given by the manufacturer, under different working conditions. In Fig. 2.7, the valve spool dynamics for a proportional servo valve Bosch Rexroth 4WRTE, at $\Delta P_{ref} = 10$ bar and $q_{ref} = 600$ lpm are shown.



(a) (b)

Fig. 2.7: Valve step responses.

From Fig. 2.7, if we want the valve to open, for instance, from 0 to 0.5, it will take around 30ms. These dynamics need to be considered for the FF and FL controllers input to the valve, as if the valve response time is too slow the disturbance will not be counteracted. Furthermore, if we are working with an industrial controller, there will exist communication delays between the controller and system. According to [19], there exist three communication delays: communication delay between the sensor and the controller, the computation delay, and the controller and actuator communication delay.

Ideally, if the valve was infinitely fast and no communication delays between the controller and the system existed, it would be enough to introduce the FF and FL signals just when the force signal reaches the reference, as the valve would open instantly to the spool position commanded by the controller.

In Fig. 2.8 the FF and FL response for a 240 bar step response is shown. The FF and FL input have been introduced just when the cylinder chamber pressure signal reaches the reference. The FF signal input opens the valve

spool instantaneously to attenuate the effect of the velocity disturbance into the cushion. As described above, the effect of the FL input is smaller than the FF input. Note that these simulations have been obtained from a mechanical press model simulation in Matlab/Simulink, which precision and accuracy will be detailed in Section 3.5.

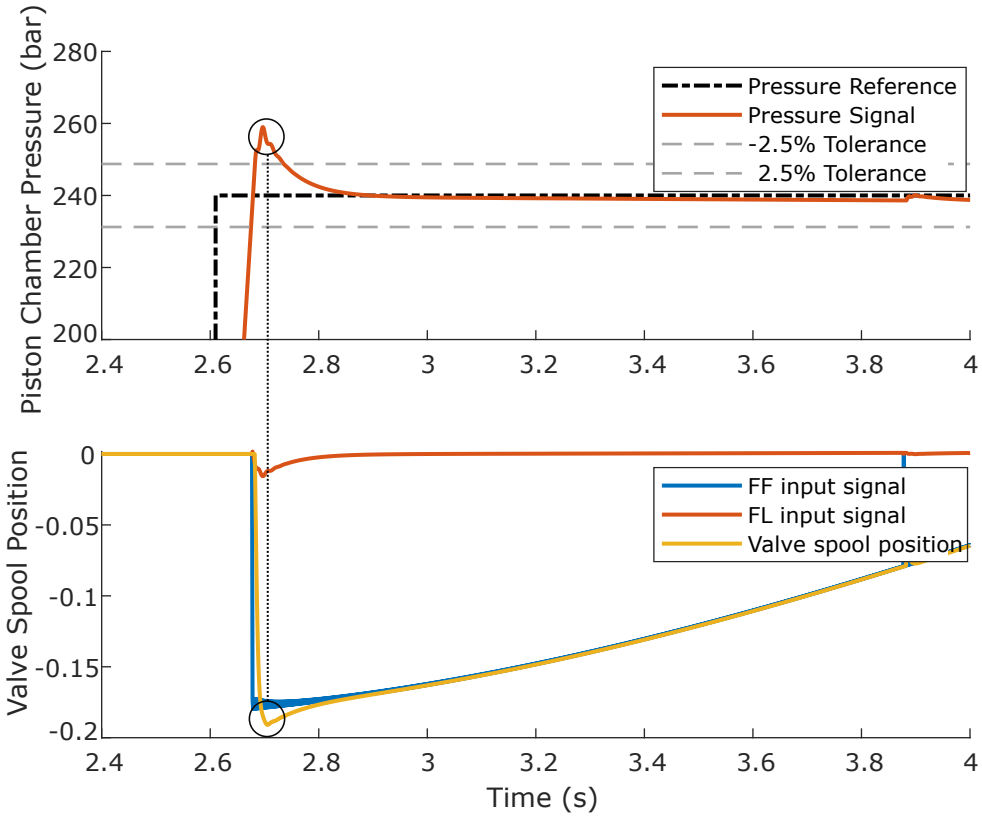


Fig. 2.8: FF control for a pressure step reference of 240 bar.

In Fig. 2.8 the overshoot is due to the neglected valve response time. The maximum opening spool position is reached after the pressure has reached the specified reference, as it has been pointed out with the circles. If the valve had been opened sooner, the overshoot would have been reduced. However, if the valve is opened too soon, we would see the opposite behavior and the pressure increase would be too slow.

To eliminate the overshoot caused by the velocity disturbance, it is necessary to consider the valve dynamics and the system communication delays to obtain the exact instant the FF and FL signals should be introduced. This instant will be referred to as the pressure percentage (PP).

The PP calculation is an approximate estimation method of the instant the FF and FL signals should be introduced, to avoid any overshoot. It is based on measured system delay communications and modeled valve dynamics.

As shown by Fig. 2.2, the trajectory the press must follow is known beforehand. Under this assumption, in (2.6) the slide velocity at the impact, $\dot{x}$, is known and with P_A defined as the desired pressure reference, an estimation of $\hat{y}_{\mathrm{v}}$ at the impact can be obtained.

Once the value of $\hat{y}_{\mathrm{v}}$ at the stroke is obtained, the valve response time, λ_{v}, to that spool position can be calculated based on the valve dynamics shown by Fig. 2.7. If any communication delays existed they would be added to λ_{v}.

The pressure increase that takes place during λ_v is the overshoot that exists in the force control. We can anticipate this pressure increase, ΔP, by introducing the FF signal at $P_{ref} - \Delta P$ so no overshoot exists in the pressure reference tracking. To that end, ΔP should be calculated beforehand.

ΔP can be computed by means of the isothermal compressibility equation, which relates the variation of the pressure at a constant temperature with the change in volume inside a body [20]:

$$\Delta P = \frac{\Delta x A_A}{V_T \beta}, \tag{2.8}$$

where Δx is the piston displacement during the valve opening delay that can be computed from the reference trajectory. V_T is the total oil volume between the piston and the proportional valve.

Subtracting ΔP from the pressure reference, P_{ref}, the PP at which the FF input signal should be introduced is obtained:

$$PP(\%) = \frac{P_{ref} - \Delta P}{P_{ref}} 100. \tag{2.9}$$

Consider, for example, the following system parameters: $\Delta x = 0.0021$ m, $A_A = 0.0284$ m^2, $\beta = 1.25 \cdot 10^{-4}$ 1/bar and $V_t = 0.0279$ m^3. Following (2.8):

$$\Delta P = \frac{0.0021 \cdot 0.0284}{1.25 \cdot 10^{-4} \cdot 0.0279} = 17.1 \text{ bar}. \tag{2.10}$$

For a 240 bar pressure reference, the following PP is obtained:

$$PP = \frac{240 - 17.1}{240} 100 = 92.88\%. \tag{2.11}$$

Introducing the computed $\hat{y}_v$ from the FF control at a 92.88% with respect to the pressure reference, the force control shown by Fig. 2.9 is obtained.

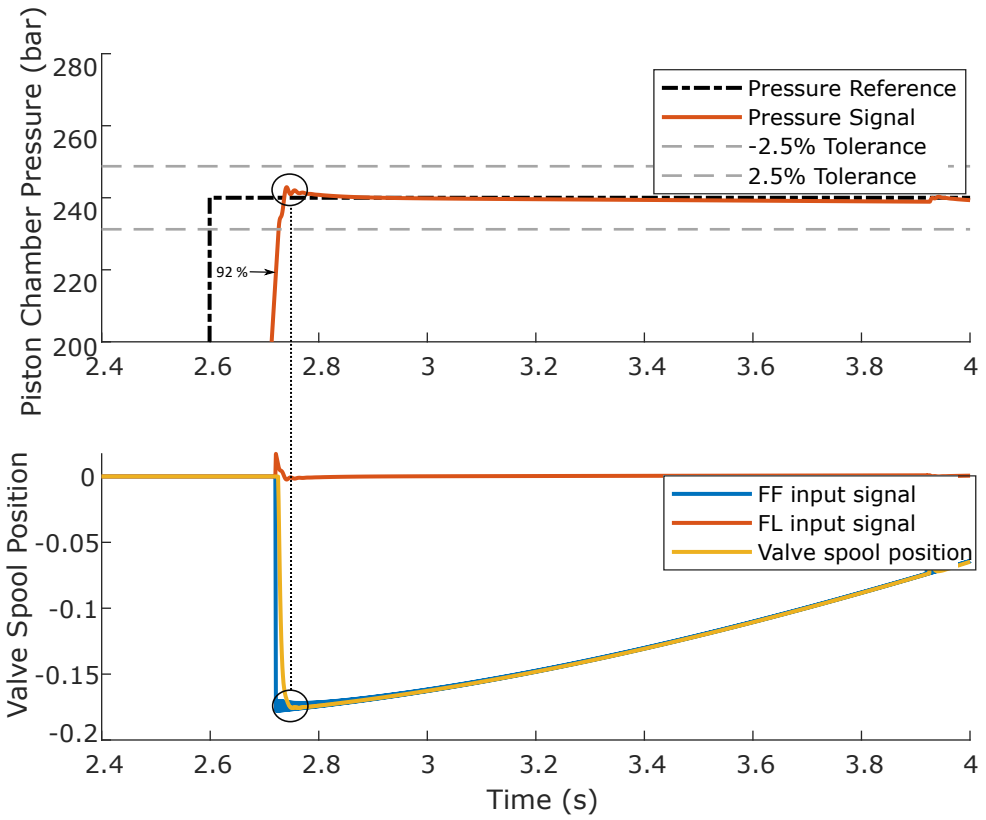


Fig. 2.9: FF control for a pressure step reference of 240 bar, with the PP calculation at 92%.

Now, due to the anticipation of the pressure increase, the valve spool is opened at the exact moment when the pressure signal reaches the reference. The result is an improved pressure reference tracking.

To analyze the FF and FL performance for the entire pressure operating range, a 60 bar pressure reference tracking is shown by Fig. 2.10. From equations (2.8) to (2.9), for 60 bar step reference the FF control must be introduced at a 71% of the reference. The resulting overshoot is minimum and every specification are fulfilled.

This calculation is an approximation as it is based on a simplified model of the valve dynamics and on system communications that can vary considerably. However, it illustrates the nuances of FF use in the context of hydraulic actuator force control.

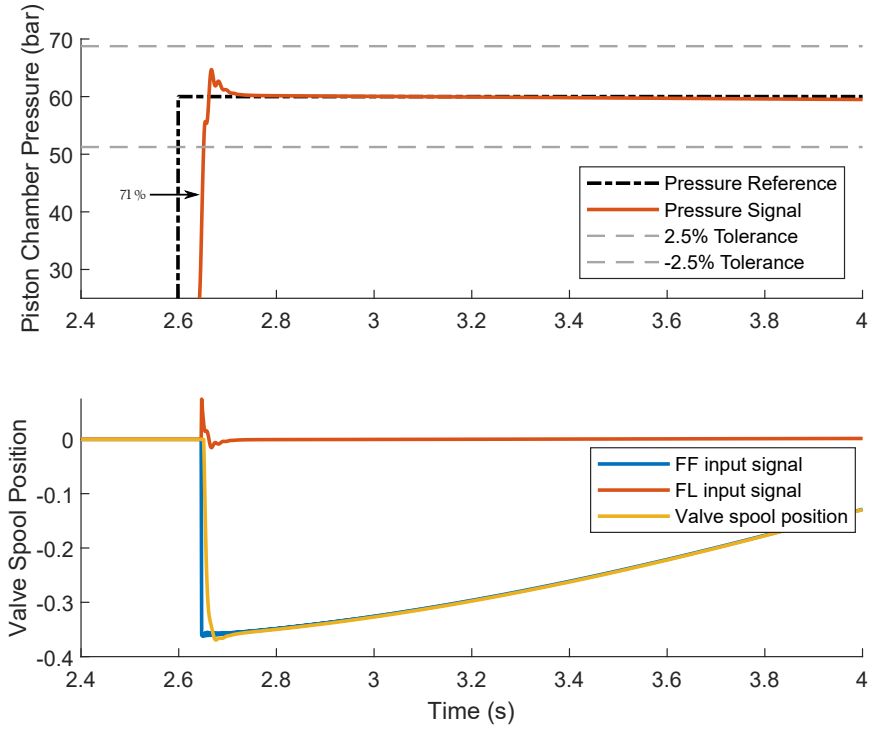


Fig. 2.10: FF control for a pressure step reference of 60 bar, with the PP calculation at 71%.

2.3.2 Feed-forward model mismatch

In the previous section, we have analyzed how much the FF performance is affected by the valve dynamics. In this section, an analysis of the dependency of the FF controller on the model parameters is carried out.

In the FF controller design, the estimation of $\hat{K}_{\mathrm{v}}^{-1}$ has been included in order to obtain $y_{\mathrm{v}}(t)$ as output. This function is unknown and nonlinear, and its estimation affects both the FF and FL terms. The slightest model mismatch in $\hat{K}_{\mathrm{v}}^{-1}$ influences the FF and FL performance considerably.

$K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ relates the calculated valve spool position with the valve command input. In the simulations carried out in Section 2.3.1, a linear function has been considered for $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, i.e. the valve spool position is equivalent to the valve command input, $K_{\mathrm{v}}(y_{\mathrm{v}}(t)) = y_{\mathrm{v}}(t)$.

With this consideration, if the valve real $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ was linear too, $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ and $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ would cancel each other, simplifying the system and obtaining a precise pressure tracking with the FF and FL controllers.

However, if the estimation $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ and the real valve $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ differ, it will affect both the FF and FL terms, resulting in an inaccurate estimation of $\hat{y}_{\mathrm{v}}(t)$. Consider the scenario shown by Fig. 2.11, where the real $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ has a slight nonlinearity, a -0.2 command input gives -0.16 spool position and, on the contrary, our estimation, $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$, is completely linear.

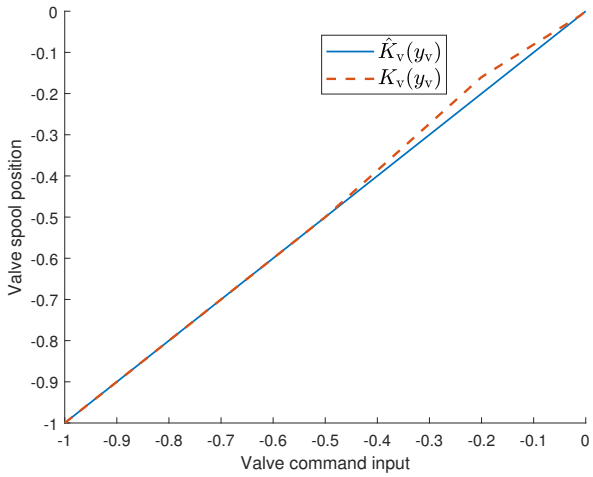


Fig. 2.11: Scenario where the real $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ differs from the estimation $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$.

A slight difference between $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ and $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ results in an inaccurate reference tracking, with high overshoot and large settling time, as shown by Fig. 2.12.

If we do not consider the valve nonlinearity in the FF and FL design, a poor estimation of $y_{\mathrm{v}}(t)$ is obtained and the control performance is penalized as shown by Fig. 2.12.

In order to improve the estimation of $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$, we can make use of regression techniques to estimate $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ and improve the performance of the FF control. There exist a great deal of machine learning (ML) regression techniques to estimate the value of nonlinear functions such as $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, e.g., Neural Networks [21], *k*-nearest neighbor [22], support-vector machine [23] or Gaussian processes [24]. These methods have a common procedure, they receive multiple pair of input and output values from which they optimize the function that generates such outputs.

With an good estimation of $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ we can correct the input obtained from the FF controller, and improve the force control. Figure 2.13 shows the resulting force controller diagram after incorporating the $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ estimation. After the FF and FL controllers, we include the estimation of the inverse of $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$, so the controller output is corrected before it is sent to the valve.

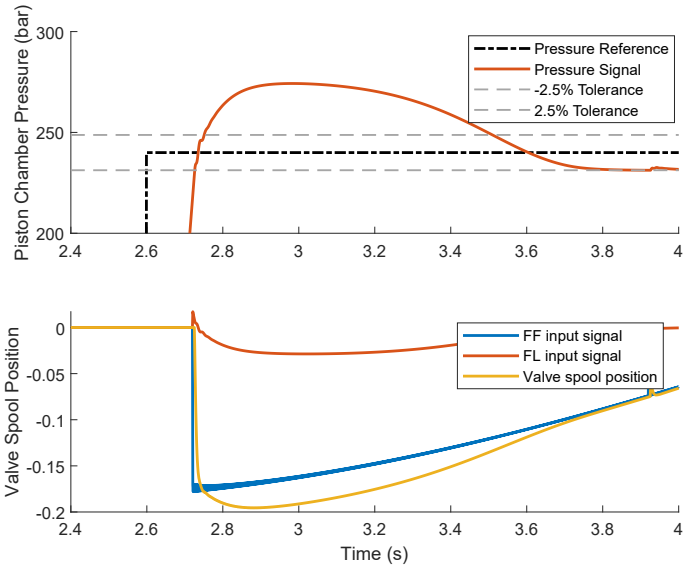


Fig. 2.12: 240 bar pressure tracking and valve spool position for nonlinear $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$.

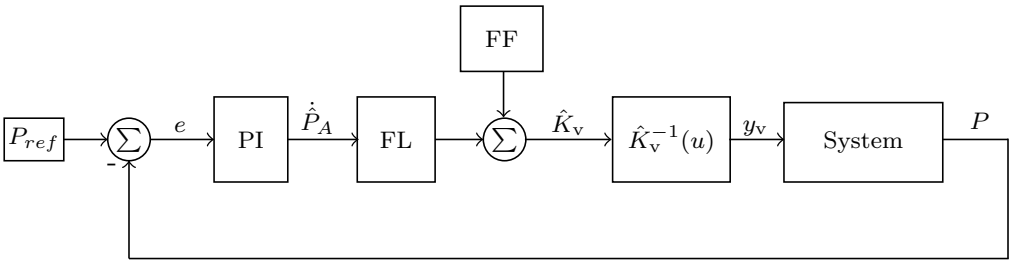


Fig. 2.13: Cushion force control block diagram with FF and $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ estimation.

Note that from the FF and FL controllers we obtain $\hat{K}_{\mathrm{v}}$, different to Fig. 2.5, as in Fig. 2.11 we have defined a $K_{\mathrm{v}}(y_{\mathrm{v}}(t)) \neq y_{\mathrm{v}}(t)$.

The estimation of $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ will be as good as the data available, therefore, the more data under different operating conditions available the better the prediction will be. To obtain training points for $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ we apply open-loop control to the cushion and let the slide be in standstill at the TDC during the entire cycle.

In open-loop control, when the cushion is at the TDC if we set a valve constant spool position, the fluid in the cylinder chamber (Fig. 2.3) will be channeled through the valve to the tank. This results in the cushion falling at a constant velocity and, in the cylinder chamber the pressure signal will settle at a constant value until the piston reaches the rod-end.

The above can be seen setting a -0.2 command input to the proportional valve, the cushion descends due to its weight at a constant velocity of $\dot{x} = -85.59$ mm/s and the chamber pressure settles at $P_A = 23.1$ bar, see Fig. 2.14.

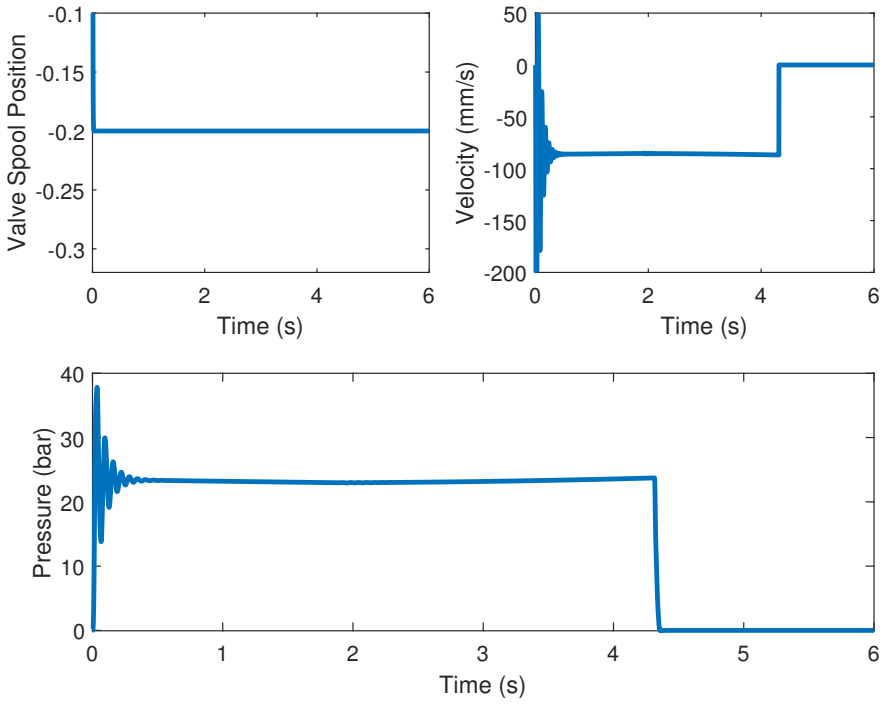


Fig. 2.14: Valve spool position, cushion velocity and cylinder chamber pressure, with decoupled slide.

In the same way as in Section 2.3, matching the valve and cylinder volumetric flow rates in equations (2.1) and (2.2), we can obtain a value for $K_{\mathrm{v}}(y_{\mathrm{v}})$ from the values obtained in Fig. 2.14. At steady-state conditions the pressure is constant, therefore, the pressure differential is zero. The expression (2.5) for $K_{\mathrm{v}}(y_{\mathrm{v}})$ simplifies to:

$$K_{\mathrm{v}}(y_{\mathrm{v}}) = \frac{A_A \dot{x} \sqrt{\Delta P_{ref}}}{\sqrt{P_A} q_{ref}}. \tag{2.12}$$

For $A_A = 0.0284$ m^2, $\Delta P_{ref} = 10$ bar and $q_{ref} = 0.01$ m^3/s, we obtain $K_{\mathrm{v}}(y_{\mathrm{v}}) = -0.16$, which is the biased value of the spool position in Fig. 2.11. This procedure is carried out for different valve spool positions, so training samples at different operating conditions are obtained, to perform the regression.

Note that any regression based technique of the ones depicted above could be used. We do not aim to investigate on the regression method, but to illustrate the performance benefits of estimating $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$. Figure 2.15, shows a regression-based estimation for $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$.

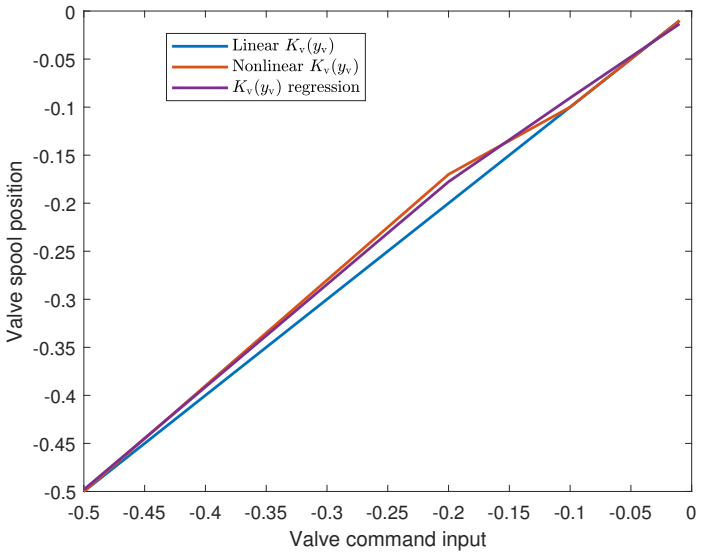


Fig. 2.15: $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ estimation for the nonlinear case, based on training samples.

We include the $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ estimation in the force control loop, as shown by Fig. 2.13, and perform a 240 bar reference tracking, which is shown by Fig. 2.16.

With the inclusion of $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ in the FF and FL design, the overshoot and settling time of the pressure signal are improved. Now the valve spool position is not the sum of the FF and FL controllers, as we apply their output to $\hat{K}_{\mathrm{v}}^{-1}(u(t))$ before sending it to the valve.

In Fig. 2.17, we show a comparison between the $y_{\mathrm{v}}(t)$ with the $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ estimation in the FF and FL controllers and without it. With the inclusion of $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$, the valve opens faster and more, so the overshoot is eliminated. Furthermore, from $t \approx 2.8$s on, the valve closes more than the linear estimation to keep the pressure around the reference.

As already concluded by [25], for the FF and FL to work, it is necessary to know every system parameter so that they can be canceled out to obtain accurate reference tracking. The hydraulic circuits model uncertainties, complicate the correct application of techniques such as FF and FL.

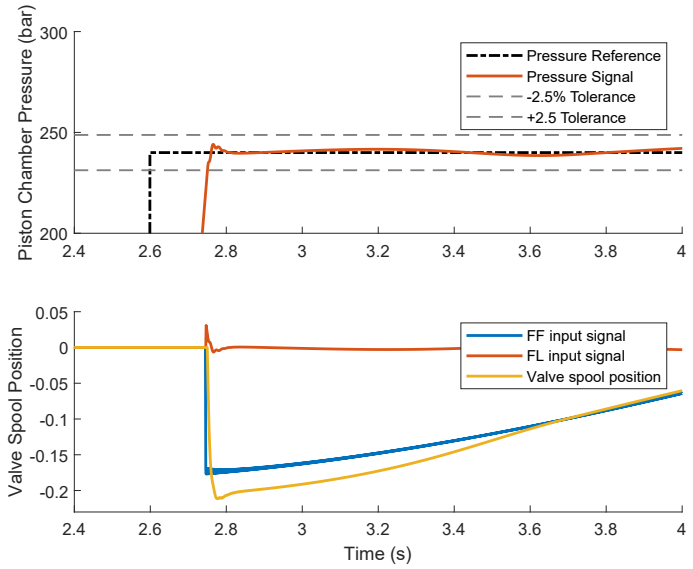


Fig. 2.16: 240 bar pressure tracking and valve spool position with $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}}(t))$ estimation included.

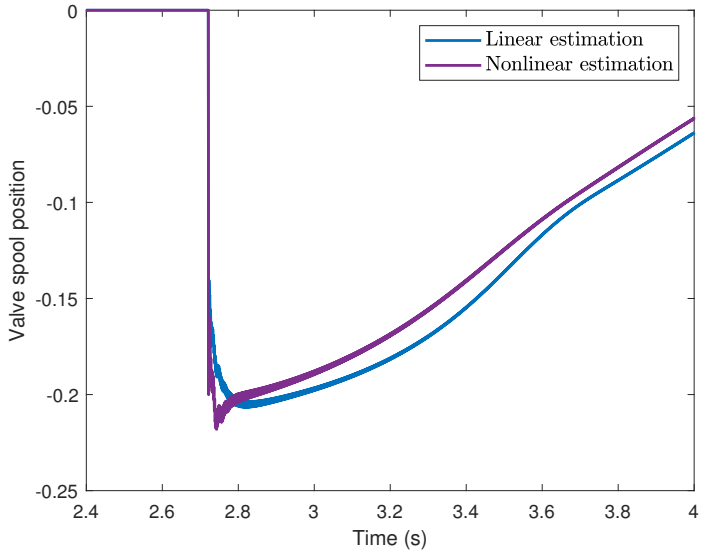


Fig. 2.17: Valve spool position without considering $K_{\mathrm{v}}(y_{\mathrm{v}}(t))$ nonlinearity, and considering it.

2.4 Conclusions to Chapter 2

During the working operation of a mechanical press it is of great importance to fulfill the force control specifications. These specifications ensure the correct forming of the workpiece and are set by the customer. Not fulfilling the

specifications can cause serious consequences, from damaging the workpiece to losing the sale arrangement with the customers.

Nowadays, the force control is carried out with a PI controller, however, the hydraulic circuit nonlinear behaviors result in poor control performance as the process deviates from the operating point. The need for advanced control algorithms that can obtain good force reference tracking is deemed necessary, so the force specifications can be fulfilled for the entire working range.

Control techniques such as FF control provide a good alternative for counteract external disturbance before it affects the system. This is ideal for the velocity that the slides induces on the cushion on and after the impact, which results in the pressure overshoot, as the slide velocity is a known disturbance.

We have seen that the FF controller is sensitive to model mismatches, and we need to make use of dynamics anticipation approaches or machine learning techniques to counteract the system uncertainties. However, in order to generate a valid data set for the machine learning algorithm training, several test have to be done in the press. This procedure should be repeated for every desired spool position until a valid dataset is obtained. Although it could be automated, it has been shown in the previous section that it requires a substantial effort to produce a large-scale set of data. Furthermore, during the service time of the valves, the dynamics might change, which would involve conducting the tests for the training points again.

Machine learning regression techniques are not deemed an appropriate solution to the cushion force control problem, as creating suitable data for the prediction is time-consuming and tedious. Techniques such as Reinforcement Learning (RL) could be of use, to obtain a policy that based on a specific cost function fulfills the cushion force control specifications.

RL is a machine learning technique for control problems, i.e. applications where actions need to be taken for the realization of a particular goal. The RL algorithm will choose the best action to take in a given situation, which is learned from experience by performing interactions with the environment. In general, it is best suited for applications where you can realize a lot of interactions with the environment quickly and cheaply.

In recent years the trend in RL has been to increase the efficiency of algorithms. When the RL algorithm does not have any information about the environment, the number of interactions that must be carried out to find an adequate control strategy can be problematic. To reduce the complexity of these algorithms, in systems in which a description of the model is available,

supervised learning has been incorporated into the design of RL algorithms [26, 27].

At this point is where we consider the suitability of RL techniques for our cushion force control problem. RL learns from experience, which is basically trial and error learning. However, we know how the hydraulic cushion must be controlled, we aim to improve the controller performance so the specifications are fulfilled. To this end, we seek a controller with the ability to learn from the existing control scheme, to update the control behavior and, consequently, improve the system performance.

Considering the limitations of the existing ML techniques and taking advantage of the repetitive process of a hydraulic press, we propose ILC as an alternative Machine Learning approach to the hydraulic cushion control problem. ILC can be regarded as an RL technique, as learning a task by repeating it many times (interactions with the model) can yield to perfecting it (optimize the task) by making minor adjustments progressively (modify the policy). An extensive analysis and design of ILC in a hydraulic actuator force control system is given in the next chapter.

3

Iterative Learning Control in hydraulic cushion force control

3.1 Introduction

ILC is founded on the concept that a system that performs the same process repeatedly, as is the case of mechanical presses, can improve its performance by learning from previous experience. Ideally, as iterations go on, the error between the desired force reference and the system response will vanish.

Figure 3.1 shows a typical ILC scheme in a feedback control loop. At the end of each j-th iteration the error, E_j, is filtered through the learning filter, L, added to the previous control, and filtered again through Q. The resulting signal is added, in the subsequent iteration, to the output of the feedback controller, C.

ILC was originally suggested by Uchiyama [28] (in Japanese) to improve the trajectory tracking of a mechanical arm. The idea of correcting the performance of a system by trial was further extended by Arimoto in [29], for a mechanical robot operation. The ILC learning law proposed by Arimoto is given by:

$$u_{j+1}(t) = u_j(t) + \gamma(t)\dot{e}_j(t), \tag{3.1}$$

where $u_j(t)$ is called the FF term of the ILC and $\gamma(t)\dot{e}_j(t)$ is the updating law for ILC. Equation (3.1) is a differential-type (D-type) ILC algorithm as it uses the error derivative, $\dot{e}_j(t)$. $\gamma(t)$ is the learning function and determines the contribution of the tracking error into the next iteration input $u_{j+1}(t)$ at the j-th iteration.

The principles that underlie the correct working of the ILC algorithm were summarized as a set of postulates by Arimoto in [30]. If these postulates are

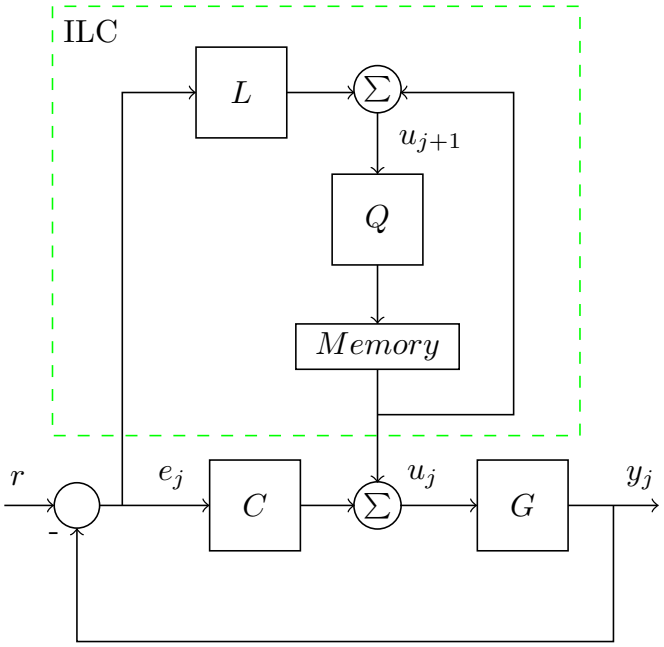


Fig. 3.1: ILC parallel arrangement with feedback controller.

fulfilled, the system output converges into the desired output trajectory as the number of iterations approaches infinity.

These postulates are reformulated to fit the framework of this project:

1. Every iteration ends in a fixed time duration t_f.
2. The desired reference $r(t)$ is known over the total time interval the ILC is working.
3. The initial state of the system is repeated at every iteration, i.e. the initial state is the same at every iteration.
4. Invariance of the system is ensured throughout the iterations.
5. Every output $y_j(t)$ can be measured and thereby the error tracking signal, $e_j(t)$, can be used in the calculation of the next input.
6. Given a desired output trajectory $r(t)$ with a piecewise continuous derivative, there is a unique input that excites the system and yields the output $y_j(t) = r(t)$.

Over the last decades, there have been significant contributions regarding the development of ILC design methods. To avoid the additional noise in the

error evaluation in (3.1), the proportional-type (P-type) ILC was proposed by Arimoto in [30]:

$$u_{j+1}(t) = u_j(t) + \gamma(t)e_j(t). \tag{3.2}$$

Unlike the D-type ILC algorithm, the P-type algorithm does not require for the error derivative. Both D-type and P-type algorithms multiply the error by the learning gain $\gamma(t)$, which maps the error to the corresponding control space. However, the design of such algorithms can be problematic, specially in the presence of measurement noise and disturbances [31].

A common technique to eliminate the noise and disturbance in the error signal is to design the learning of the ILC algorithm as a low-pass filter, so the undesired high-frequencies are attenuated [32, 33, 34]. A typical ILC learning law can be designed with $\gamma(t)$ being a low-pass filter to guarantee the algorithm stability under disturbances. However, one must be conservative when choosing the cut-off frequency of the learning filter, as it could cause over-correction of the error components, affecting the algorithm learning.

The use of a low-pass filter improves system stability, however, the phase lag introduced by the filter could affect the stability condition to be satisfied at a certain frequency range [35]. To improve the performance of the filter in a wider frequency range, zero-phase filtering (ZPF) is commonly used in the ILC design [36, 37], as it allows to filter the error signal forward and then, apply backward filtering to the filtered signal. Backward filtering generates phase lead to compensate for the phase loss of the forward filtering to achieve a zero-phase effect.

Furthermore, higher-order ILC algorithms were developed to further improve the convergence rate of the algorithm [38, 39], in which the error profile information from two or more previous iterations was used. However, this approach increases considerably the required memory usage of the algorithm. A comprehensive review of the existing ILC techniques and theory is given by Bristow in [40] and by Ahn in [1].

Regarding the application of ILC algorithms in hydraulic circuits, it was first applied by [41] for the position tracking control of a hydraulic cylinder. The D-type ILC algorithm was used with a delay time included in the learning law. Increasing the delay time, the convergence rate of the algorithm was improved, however, the stability got affected. With a small delay time, the error converged to a bigger value but the overall stability improved.

To improve the convergence rate, a PD-type ILC algorithm was designed for the position control of a six degrees of freedom hydraulic platform in [42].

The algorithm stability issues were accounted including a forgetting factor in the learning law, to reduce the initial control value influence.

The PD-type experiences the same drawbacks as the P-type and D-type algorithms in the presence of noise and disturbances [31]. To that end, a P-type ILC algorithm with a low-pass filter was proposed for the position tracking of a hydraulic cylinder in [43]. To improve the stability and robustness of the algorithm, a filter was included in the learning gain design, to discard the high frequencies.

This approach was also followed by [44], where a low-pass filter was included in the learning filter design to improve the pressure control of two hydraulic cylinders. With the low-pass filter, the high frequency uncertainties and noise were filtered. The phase loss of the Butterworth filter was counteracted by introducing a time-shift operator to provide system stability. The convergence rate of the algorithm was low compared to other studies as the ILC design was focused on the stability of the system towards external disturbances rather than improving the convergence rate.

Some works have studied the implementation of ILC in industrial environments. In [45], a fast norm-optimal ILC that simplifies the algorithm structure and increments its speed is designed. A combination of a PID controller and ILC is implemented by [46] in an industrial SIEMENS SIMOTION controller for a production machine. The authors focused on the applicability of the ILC in industrial applications rather than on stability and convergence conditions. No analysis about the learning gain design is given as it is chosen by trial an error.

These works, although they improved the control performance of a hydraulic system in comparison to the existing feedback controller, do not focus on the convergence rate analysis of the algorithm. The convergence rate is a fundamental aspect of an ILC algorithm and one should aim to obtain the fastest convergence rate possible to reduce the number of iterations needed. The fewer iterations required, the shorter the commissioning time of the hydraulic cushion control will be, which translates into a reduction of the production costs.

From the above, one can conclude that there exists a gap in the literature regarding the design of ILC algorithms in hydraulic systems. No analysis of the convergence rate improvement of the ILC algorithm to optimize the press working operation has been done. Furthermore, this improvement should be enhanced with a stability analysis of the algorithm, as stability is an essential property for industrial practice. The next section is dedicated to the analysis

of the ILC algorithm, which yields a design proposal to improve the stability and convergence rate of the existing ILC designs.

3.2 SISO ILC

The design and analysis of the ILC is done in frequency-domain, as it allows the analysis of the convergence and the stability making use of techniques such as Bode plots or Nyquist diagrams [36]. With the frequency-domain analysis, we can make use of frequency domain methods to filter those uncertain and undesired dynamics of the hydraulic circuit, as well as guarantee the stability of the system within a specific frequency range.

Equation (3.3) shows the error propagation equation for the frequency-domain, which derivation is described in Appendix A, and relates the error at the current iteration to the next. We can drop the matrix notation, as we are working in a single-input-single-output (SISO) system, which results in the following error propagation equation:

$$E_{j+1}(s) = (1 - G(s)S(s)L(s))E_j(s), \tag{3.3}$$

where $S(s) = 1/(1 + G(s)C(s))$ is the sensitivity transfer function of the system.

From (3.3), if the term $1 - G(s)S(s)L(s)$ is less than one for every frequency, then system stability is ensured and the error will decrease from one iteration to the next. Otherwise, if $1 - G(s)S(s)L(s) > 1$, the error decrease will not be guaranteed as iterations go on. From the above a sufficient condition for the stability of the algorithm and for convergence can be set, which was given by [40] in the frequency domain, where $s = jw$:

$$|1 - G(jw)S(jw)L(jw)| < \frac{1}{Q(jw)} \qquad \forall \omega \in [-\infty, \infty], \tag{3.4}$$

where $Q(jw)$ is normally a gain that enlarges the stability region. The $L(s)$ learning filter must fulfill (3.4) for the system to be stable and the error to converge towards zero.

There exist several design approaches for $L(s)$, some of which have been mentioned in the previous section. The inversion of the plant dynamics is a benchmark approach to achieve a quick convergence rate. Generally, it is not easy to obtain an exact model that represents the real plant, and this

approach is sensitive to modeling errors [47]. Therefore, one must accept that without an exact plant model, the convergence rate will get affected.

Several studies have addressed the incorporation of the known model dynamics in the $L(s)$ design to increase the convergence rate of the ILC algorithm. A parameter optimal ILC (POILC) was designed in [47]. The POILC design was based on an approximate inverse model of the plant and, to improve robustness, a learning gain was included that adaptively changed at each iteration. However, this design yielded a more complex condition for monotonic convergence and relied on the good tracking performance at the first iteration.

The optimization problem for model-based ILC controller is also developed in [48, 49, 50]. In these works, a quadratic cost function is defined, from which an optimal solution for the ILC problem is obtained. A quadratic cost function is also introduced to analyze the convergence on a model-based ILC in [51] for time-varying systems. These approaches rely on the tuning of the weighting matrix in order to obtain a global minimum in the minimization problem. Furthermore, it is hard to explicitly define the ILC multiple objectives in a cost function such as the convergence, robustness, stability and input constraint.

Finding an adequate cost function to optimize can turn hard and laborious. An alternative and more straightforward approach is to compute an approximate inverse of the nominal plant, rather than the exact model system. For systems with unstable zeros, pseudo-inverse and stable inversion methods have been proposed in [52, 53], respectively. In [54], an analysis of which inversion approach to use depending on the system zeros is carried out.

For the hydraulic cushion model inverse system, we need to map the pressure error with the valve spool position. To that end, we use the same dynamic model as the one used for the FF controller, which was defined in (2.6). Neither unstable zeros nor unstable poles exist in this model, therefore direct inversion can be applied.

In [55] direct inversion was applied in an electronic printer, where the learning filter was designed as $L(s) = (G(s)S(s))^{-1}$. Instead of including a low-pass filter in the $L(s)$ design, the $Q(s)$ filter was used to guarantee robustness to modeling errors. However, $Q(s)$ should be designed properly, as a $Q(s)$ with a module different from one penalizes the algorithm performance [40].

Direct model inversion was also employed in [56] for a wafer system. In [56], it is proposed to include the controller in the model inverse ILC design

based on the series and parallel ILC designs. $P_p(s)$ is the process sensitivity function for the parallel architecture, which is used as a parametric model of the system to design the learning filter, and is such as:

$$P_p(s) = \frac{G(s)}{1 + G(s)C(s)}. \tag{3.5}$$

In [56], unlike in [55], the low-pass filter was applied to the inverse of $P_p(s)$ to deal with the high frequency differences. This is the most common procedure that one can find in the literature when applying direct inversion [56, 57, 58, 59]. Although fast convergence tracking is achieved, the algorithm performance is penalized as the inclusion of the low-pass filter to counteract model mismatches affects both $G(s)$ and $C(s)$.

We claim it can be avoided that the low-pass filter included to achieve robustness to plant uncertainty affects both $G(s)$ and $C(s)$. One can design $L(s)$ so that the low-pass filter only affects the inverse of $G(s)$, excluding $C(s)$, so the model differences can be reduced.

For a system with a parallel structure comprising an ILC algorithm and a feedback controller, we propose the following model inverse $L(s)$ design:

$$L(s) = (S(s)G(s))^{-1} = \frac{1 + G(s)C(s)}{G(s)} = G^{-1}(s) + C(s). \tag{3.6}$$

In (3.6), only $G(s)$ is inverted and $C(s)$ is completely known, as it is the existing feedback controller in the system. The low-pass filters to deal with high frequency differences will only apply to $G^{-1}(s)$, not affecting $C(s)$. With this $L(s)$ design, the convergence rate is improved regarding [56, 57, 58, 59] as the model differences are reduced.

3.3 SISO ILC learning filter design

For the purpose of designing $L(s)$ based on (3.6), a linearized state space system of the hydraulic cushion circuit is derived, from which $G^{-1}(s)$ will be obtained.

In the same way as carried out in Section 2.3, in the cushion hydraulic circuit shown in Fig. 2.3, during the *Drawing* phase the flow out of the cylinder chamber will equal to the flow into the proportional valve. From the cylinder and valve model equations, see (2.1) and (2.2), a linearized state space with

the pressure of the cylinder, the valve spool position and velocity as state variables, and the valve command as input, is derived.

For illustrative purposes the dynamics of the proportional valve from u to y_v are modeled as a second-order transfer function, $G_{\mathrm{v}}(s)$. These dynamics will not be considered in the learning filter design. However, they will serve us to see how the unmodelled dynamics affect the ILC performance:

$$G_{\mathrm{v}}(s) = \frac{y_{\mathrm{v}}(s)}{u(s)} = \frac{\omega_v^2}{s^2 + 2\omega_v\zeta s + \omega_v^2}, \tag{3.7}$$

where ω_v and ζ are the valve natural frequency and damping, respectively.

The linearized hydraulic cushion system results in:

$$\begin{bmatrix} \delta\dot{P}_A \\ \delta\dot{y}_{\mathrm{v}} \\ \delta\ddot{y}_{\mathrm{v}} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & 0 \\ 0 & 0 & 1 \\ 0 & -\omega_{\mathrm{n}}^2 & -2\omega_{\mathrm{n}}\zeta \end{bmatrix} \begin{bmatrix} \delta P_a \\ \delta y_{\mathrm{v}} \\ \delta\dot{y}_{\mathrm{v}} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega_{\mathrm{n}}^2 \end{bmatrix} \delta u. \tag{3.8}$$

With A_{11} and A_{12} defined as:

$$A_{11} = -\frac{K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})\sqrt{\Delta P_{ref}}}{(V_{\mathrm{Ad}} + A_A x)\beta q_{ref}} \frac{1}{2\sqrt{\bar{P}_A}} \qquad A_{12} = -\frac{\sqrt{\bar{P}_A} q_{ref}}{(V_{\mathrm{Ad}} + A_A x)\beta\sqrt{\Delta P_{ref}}} K'_{\mathrm{v}}(\bar{y}_{\mathrm{v}}),$$

where $\delta P_A = P_A - \bar{P}_A$ and $\delta y_{\mathrm{v}} = y_{\mathrm{v}} - \bar{y}_{\mathrm{v}}$, are the small-signal deviation from an operating point obtained in steady-state conditions, $\bar{P}_A$ and $\bar{y}_{\mathrm{v}}$ respectively. $K'_{\mathrm{v}}(\bar{y}_{\mathrm{v}})$ is the hydraulic conductance derivative with respect to y_{v}.

The dynamics corresponding to the valve are not considered in the plant inverse design for $L(s)$, the simplified plant inverse is designed as follows:

$$\hat{G}^{-1}(s) = \frac{s - A_{11}}{A_{12}}. \tag{3.9}$$

The Bode diagram of the inverse of the system shown in (3.8), $G^{-1}(s)$, and the simplified system, $\hat{G}^{-1}(s)$, without the valve dynamics is shown in Fig. 3.2. At low frequencies, both systems have the same response and, as frequency increases, the responses deviate. This is due to the exclusion of the valve dynamics, which affect the system at high frequencies.

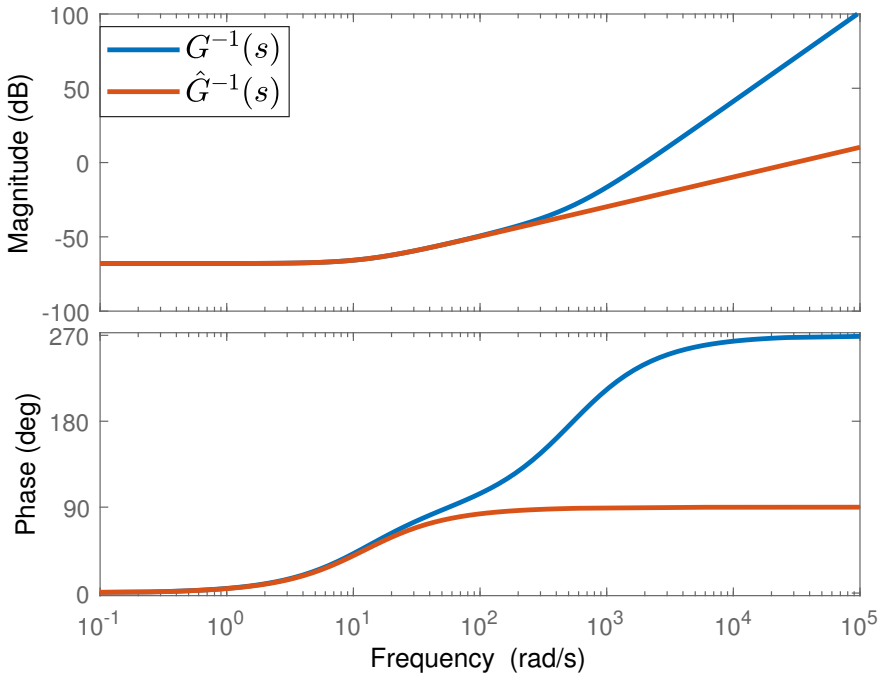


Fig. 3.2: Bode diagram of the plant inverse $G^{-1}(s)$, and simplified plant inverse $\hat{G}^{-1}(s)$.

The $L(s)$ filter design depends on the position, x, which has been considered constant although the cylinder retracts during the *Drawing* phase. The fluid compressibility varies because of pressure and temperature changes taking place in the cylinder chamber and the air mixed with the oil. Moreover, $K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})$ and its derivative $K'_{\mathrm{v}}(\bar{y}_{\mathrm{v}})$ are unknown. Clearly, as some system dynamics are not included in the $L(s)$ filter design, the plant will not be completely simplified in (3.3), which will affect the convergence rate of the algorithm.

As shown in Fig. 3.2, $\hat{G}^{-1}(s)$ acts as a high frequency amplifier, by which the unknown and undesired high frequency dynamics of the system will not be attenuated. Therefore, as already done in [56, 60], we include a low-pass filter in the $L(s)$ design, however, it will only affect $\hat{G}^{-1}(s)$, which results as:

$$\hat{G}_f^{-1}(s) = \frac{s - A_{11}}{A_{12}} \frac{\omega_r^4}{(s + \omega_r)^4}, \tag{3.10}$$

where ω_r is the filter cutoff frequency in rad/s.

With the low-pass filter we introduce a considerable phase lag in our system, see the response of $\hat{G}_f^{-1}(s)$ in Fig. 3.3. To avoid the additional phase lag, ZPF is widely used in ILC design as explained in Section 3.1.

ZPF is applied in the model inverse design to avoid any phase lag and to filter the undesired high frequency dynamics. From the ILC block diagram shown in Fig. 3.1, the signal out of the ILC scheme is added to the feedback controller signal. In our case, the controller signal is the proportional valve spool position input, which has a physical limitation, the spool position must not exceed the range [-1, 0]. Therefore, the signal obtained from the ZPF must not exceed said range. To that end, an anti-windup has been included in the filtering process. The ZPF algorithm with the anti-windup is explained in Appendix B.

The resulting plant inverse design with the ZPF is:

$$\hat{G}_{zpf}^{-1}(s) = \frac{s - A_{11}}{A_{12}} \frac{\omega_r^2}{(s+\omega_r)^2} \frac{\omega_r^2}{(s-\omega_r)^2}. \tag{3.11}$$

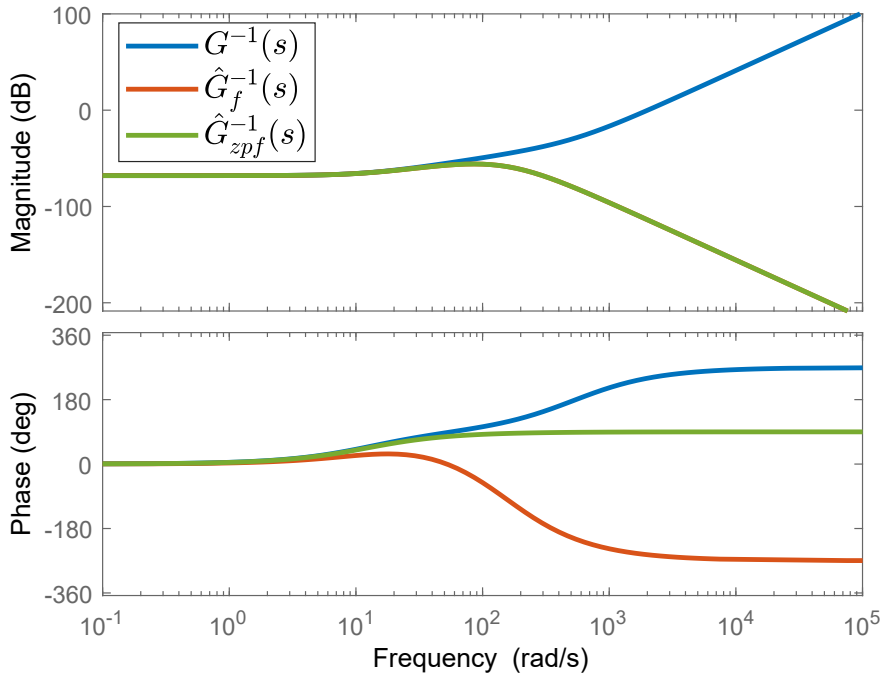


Fig. 3.3: Bode diagram of the plant inverse $G^{-1}(s)$, with low-pass filters $\hat{G}_f^{-1}(s)$ and with ZPF $\hat{G}_{zpf}^{-1}(s)$.

In Fig. 3.3, the Bode diagram of the resulting plant inverse after the ZPF is shown. The high frequencies are attenuated without losing phase. The resulting $L(s)$ design is thus:

$$L(s) = \underbrace{\frac{s - A_{11}}{A_{12}} \frac{\omega_r^2}{(s+\omega_r)^2} \frac{\omega_r^2}{(s-\omega_r)^2}}_{\hat{G}_{zpf}^{-1}} + C(s). \tag{3.12}$$

3.4 SISO ILC stability analysis

The stability of the ILC algorithm, given the proposed $L(s)$ design, must be proven. As explained in Section 3.2, the $L(s)$ filter must fulfill the monotonic convergence condition in (3.4).

In the literature there exist various approaches to analyze the algorithm stability. The Hurwitz stability is used in [61], the Schur stability is applied in [62, 63] and the vertex points of the interval Markov parameters are used in [64].

These methods require many computations and cannot be directly applied to an ILC algorithm. A more straightforward method is to analyze the stability graphically in the frequency domain, as it is done in [44, 47, 37, 65].

In these studies, the response for all the frequencies up to the Nyquist frequency is plotted to analyze the convergence condition given in (3.4). This convergence condition can be interpreted on a Nyquist plot; if the frequency response remains inside the unit circle, the ILC algorithm will converge to zero error. Else, if the response gets out of the unit circle, the ILC algorithm will diverge.

If an accurate model inverse is included in the $L(s)$ design, the resulting Nyquist plot would be zero for all frequencies, as the right-hand side in (3.3) would vanish. However, our $L(s)$ design is far from being ideal as some simplifications have been done and the uncertain high frequencies have been attenuated with ZPF.

Hence, we should aim to obtain the frequency response as close as possible to the origin. The closer to the origin, the faster those error frequencies will be corrected by the ILC algorithm. The frequency response of $L(s) = \hat{G}_{zpf}^{-1}(s) + C(s)$ is shown by Fig. 3.4, for the system parameters specified in Appendix C.0.2. As the frequency increases, the response diverges from the origin but does not get out of the stability circle, which guarantees the

stability of the algorithm at high frequencies. Note that the unit circle has been enlarged, with a $Q = 0.96$.

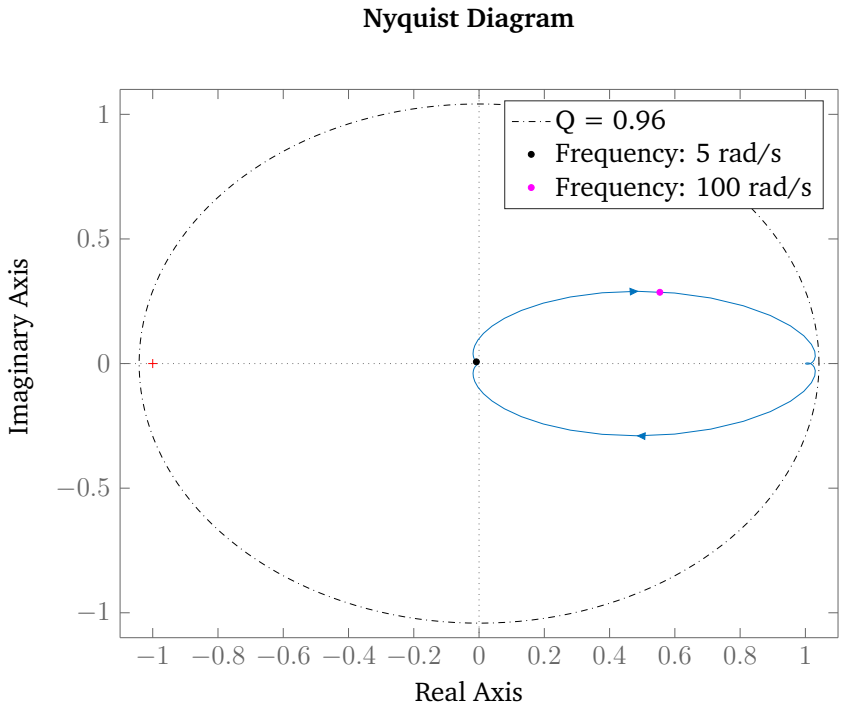


Fig. 3.4: $|1 - G(s)S(s)L(s)|$ frequency response with $L(s) = \hat{G}^{-1}_{zpf}(s) + C(s)$.

3.5 Simulation study: SISO ILC implementation in a hydraulic cushion circuit

The performance of the designed ILC algorithm is evaluated in a nonlinear model of a cushion hydraulic circuit, implemented in Matlab/Simulink. The system implementation is shown in Fig. 3.5, and consists of every element shown previously in Fig. 2.3. A translational sensor has been added to monitor the cushion displacement and velocity. With the pressure sensor we measure the cylinder chamber pressure to carry out the force control during the *Drawing* phase. From the valve sensor we obtain the actual spool position values. With the ideal translational source, we simulate the slide trajectory and adjust the impact velocity.

The press model elements have been developed in a novel library made by Ikerlan, the precision and accuracy of which have been validated in [66].

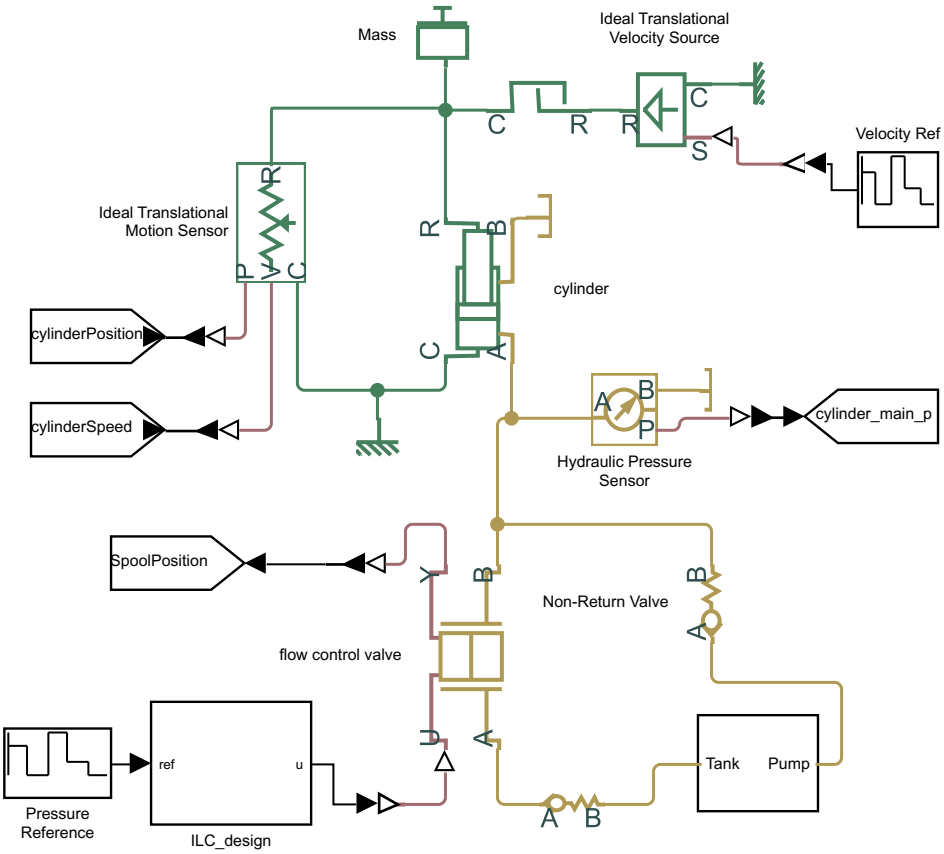


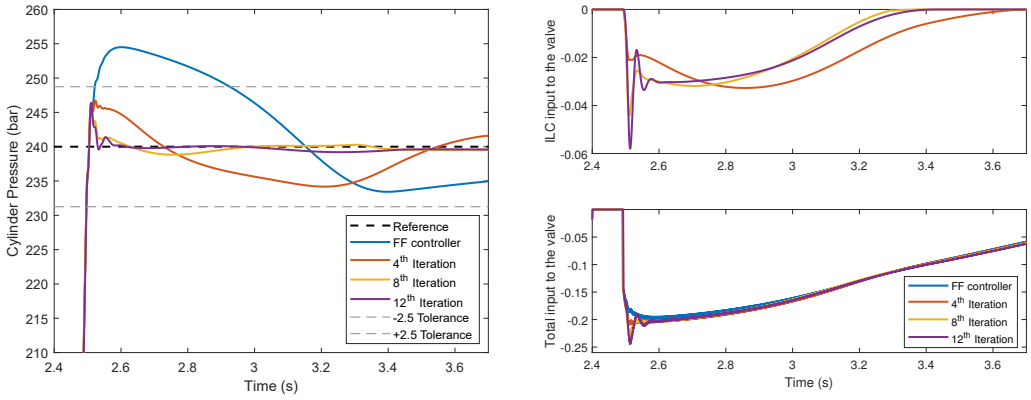
Fig. 3.5: Simscape implementation of the cushion hydraulic circuit.

With this library, components are easily parametrized with data-sheet information in contrast to other Simscape libraries, which requires acquiring constructive parameters obtainable only from laboratory experiments. This novel approach allows reproducing the physical behavior of industrial components with high precision without losing Real-Time capabilities.

We consider the scenario of Section 2.3.2, where the real $K_{\mathrm{v}}(y_{\mathrm{v}})$ differs from the estimation $\hat{K}_{\mathrm{v}}(y_{\mathrm{v}})$, as shown in Fig. 2.11. The pressure signal tracking for a 240 bar reference is shown in Fig. 3.6a. Under this scenario, at the first iteration where no ILC signal exists, the FF control is poor due to the model mismatches included. From iteration two on, the ILC algorithm is used in combination with the FF controller, to improve the FF controller performance under model uncertainty conditions.

The ILC algorithm impact on the control over iterations can be seen in Fig. 3.6b, where the total input and the ILC contribution in the input are shown. In order to reach as fast as possible the reference and avoid any overshoot, the ILC introduces an oscillation in the input, at $t \approx 2.5$s, that maintains the pressure around the reference. From $t \approx 2.5$s to $t \approx 2.8$s, the spool position

reaches -0.2 value, just where the mismatch had been introduced in Fig. 2.11. To counteract this mismatch, the ILC adds a constant input value of -0.03 as a correction to the controller. At the end of the step, from $t \approx 2.8$s on, the ILC input signal impact decreases and the FF controller maintains the pressure level around the reference, as shown in Fig. 3.6a.



(a) (b)

Fig. 3.6: Pressure and input signal for a 240 bar reference.

As a performance index for the proposed force control ILC algorithm, the root mean square error (RMSE) between the pressure reference and the cylinder pressure signal is shown in Fig. 3.7. Here, the steady state error is defined as the value the RMSE converges to. A high steady state error indicates that the error between the reference and the actual signal has not been corrected.

The RMSE is reduced by a factor of 8 in the pressure tracking, and a fast convergence rate is obtained as at the fourth iteration the error is considerably reduced, with respect to the first iteration with the FF controller. Note that it is physically impossible to reduce the RMSE below 0.8083, as we cannot obtain a faster pressure increase due to oil compressibility.

We evaluate the robustness of the ILC algorithm at a different operating point for a step reference of 60 bar, as shown in Fig. 3.8a. We can see that at a low pressure reference the specifications are fulfilled and the pressure signal does not exceed the settling time tolerance or overshoot limits. At the first iteration, with the FF controller, the steady state error is considerable and the signal does not recover. With the ILC signal, as iterations go on, the pressure signal follows the reference, with no steady state error. The ILC contribution to the FF controller can be seen in Fig. 3.8b.

The RMSE over iterations is shown in Fig. 3.9, although at the first iteration with the FF controller the error is low, the ILC algorithm reduces that error

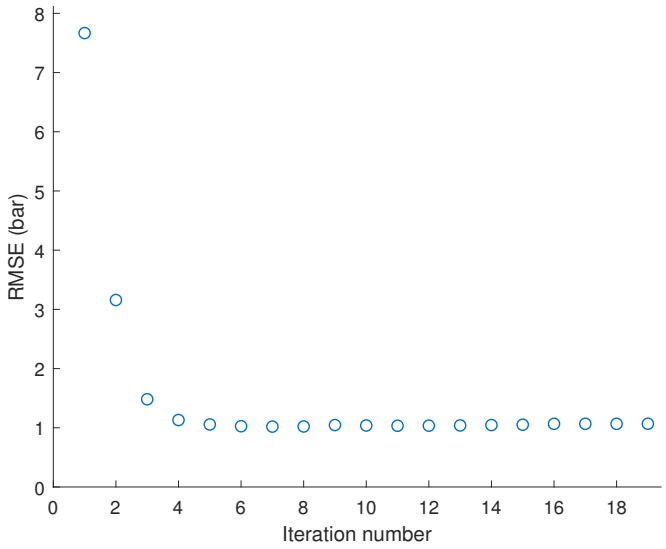


Fig. 3.7: RMSE between the cylinder pressure and the 240 bar reference over iterations.

by a factor of 3.5. Note that, for a 60 bar pressure reference, it is impossible to obtain a RMSE less than 0.2, due to oil compressibility.

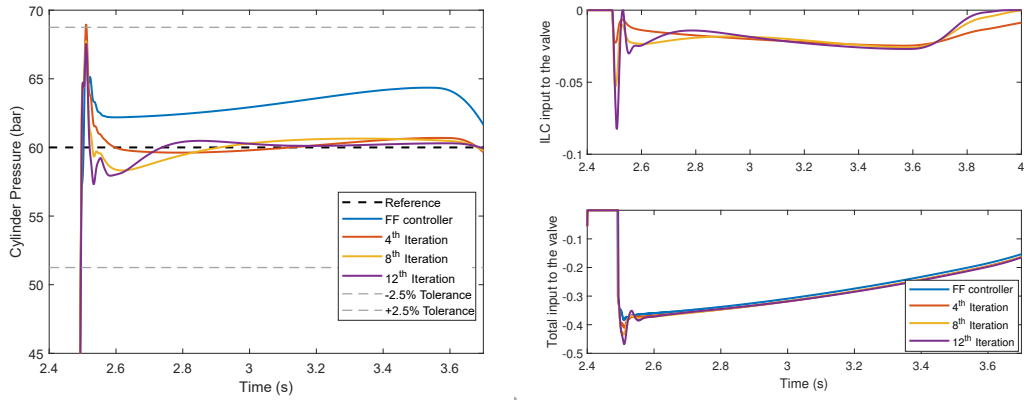
3.6 SISO ILC performance comparison

To evaluate the performance of the proposed ILC algorithm, two different ILC algorithms are designed through this section to carry out a comparison. The first algorithm is the proportional ILC (P-ILC) which is the most common algorithm used in industrial application due to its simplicity.

The second algorithm is the model inverse based ILC algorithm proposed in [56, 57, 58, 59], so a comparison can be made with other model inverse approaches. For ease of notation, the model inverse algorithm proposed in the previous sections will be referred to as MIC-ILC.

3.6.1 SISO P-ILC algorithm

The P-ILC is one of the most used ILC algorithms in the literature due to its simplicity, as it calculates an input signal proportional to the error. It was first proposed by Arimoto in [30], the learning law is as follows:



(a) (b)

Fig. 3.8: Pressure and input signal for a 60 bar reference.

$$U_{j+1}(s) = U_j(s) + K_P E_j(s), \tag{3.13}$$

where K_P is a proportional gain that multiplies the error to update the input for the next iteration.

From (3.13), the P-type algorithm is conservative in the sense that by choosing a sufficiently small K_P learning gain, the convergence rate property can be guaranteed. However, this can result in slow convergence.

Therefore, K_P must be picked conservatively to guarantee the stability of the algorithm and fulfill (3.3). However, its value can be tunned to maximize the convergence rate of the algorithm. Setting a $K_P = 0.0002$ 1/bar, the frequency response (3.4) of the P-ILC algorithm is shown in Fig. 3.10. Q is set to 0.98, so the frequency response lies inside the stability circle.

In Fig. 3.10, at zero frequency the response lies at (+1.02,0), whereas with the MIC-ILC algorithm at zero frequency the response lies at the origin (Fig. 3.4). This is due to the cancellation of the model dynamics at low frequencies, that optimizes the convergence rate of the ILC algorithm.

The closer to the origin the frequency response lies, the faster those frequencies will be corrected by the ILC algorithm. In Fig. 3.10 we have pointed out the 5 rad/s point, which lies far from the origin because of the K_P design. As frequency increases, the response gets away from the origin faster than the MIC-ILC, see the 100 rad/s point in Fig. 3.10.

In Fig. 3.11, the pressure signals and control inputs for a 240 bar reference tracking are shown. The P-ILC improves the FF controller behavior consid-

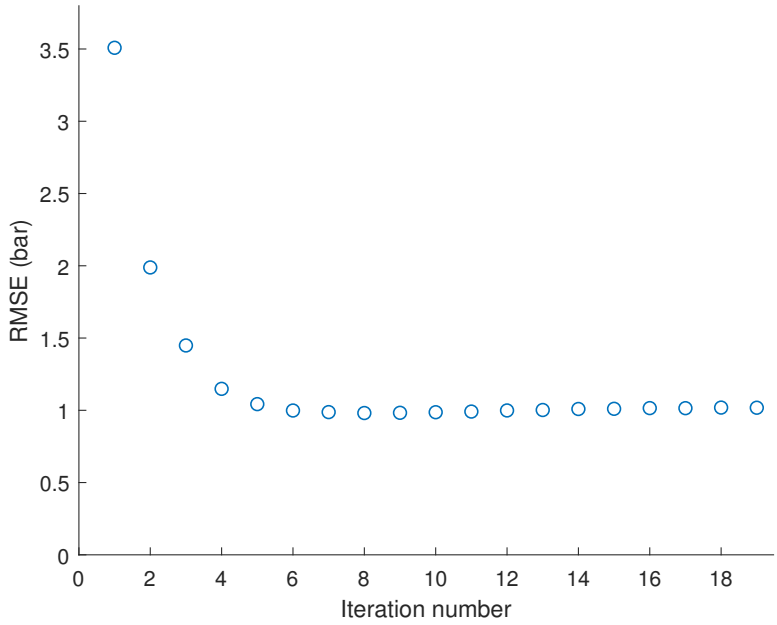


Fig. 3.9: RMSE between the cylinder pressure and the 60 bar reference over iterations.

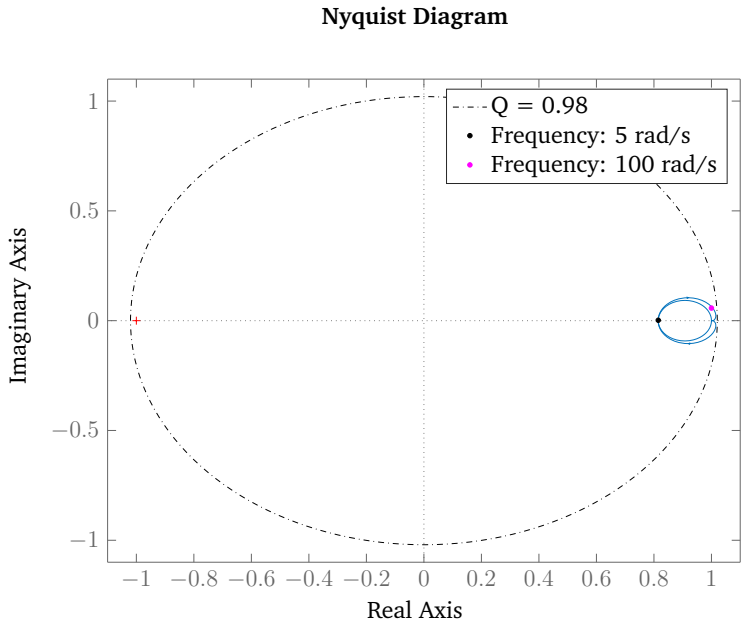
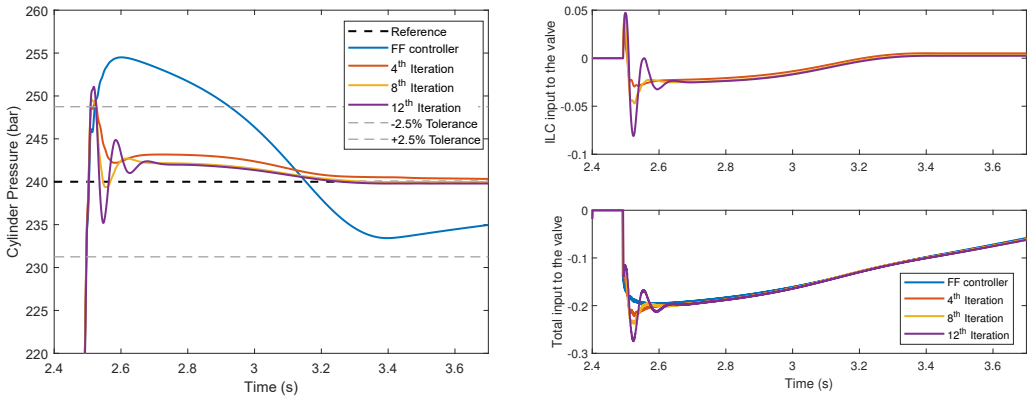


Fig. 3.10: $|1 - G(jw)S(jw)L(jw)|$ frequency response with $K_P = 0.0002$ 1/bar for the P-ILC algorithm.

erably, yet once the learning has finished, after 12 iterations, it takes from $t \approx 2.5$s to $t \approx 3.2$s to reach the reference.



(a) (b)

Fig. 3.11: Pressure and input signal for a 240 bar reference with the P-ILC algorithm.

A comparison of the pressure RMSE between the P-ILC and the MIC-ILC algorithms is shown in Fig. 3.12. A very low convergence rate is obtained with the P-ILC algorithm, as the learning between iterations is small. Furthermore, the steady-state error towards which the RMSE converges is high, with respect to the MIC-ILC. This is because the initial overshoot at the beginning of the step, that it is not eliminated.

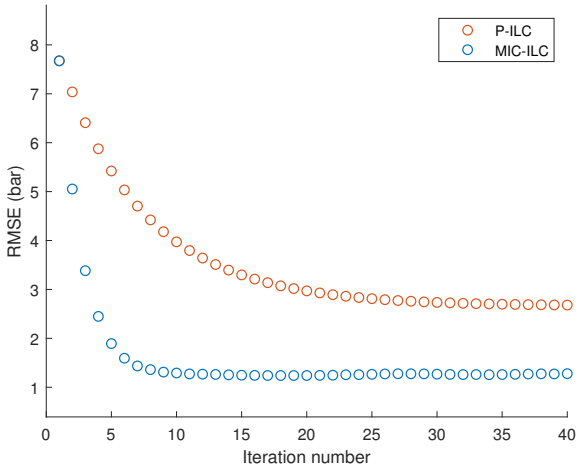


Fig. 3.12: RMSE between the cylinder pressure and the 240 bar reference over iterations with the P-ILC and the MIC-ILC algorithms.

We can analyze the pressure error signal in the frequency domain, with the Fast Fourier transform, to obtain the frequency components of the error and see their behavior as iterations go on. The single-sided power spectrum of the error signal is shown in Fig. 3.13, for the FF controller, the P-ILC and

MIC-ILC algorithms. The evolution of the iterations is shown, to see how the low frequencies are attenuated faster with the MIC-ILC than the P-ILC at the first iterations. From iteration 10 on, the MIC-ILC remains constant, while the P-ILC continues decreasing, though it does not decrease the error as much as the MIC-ILC.

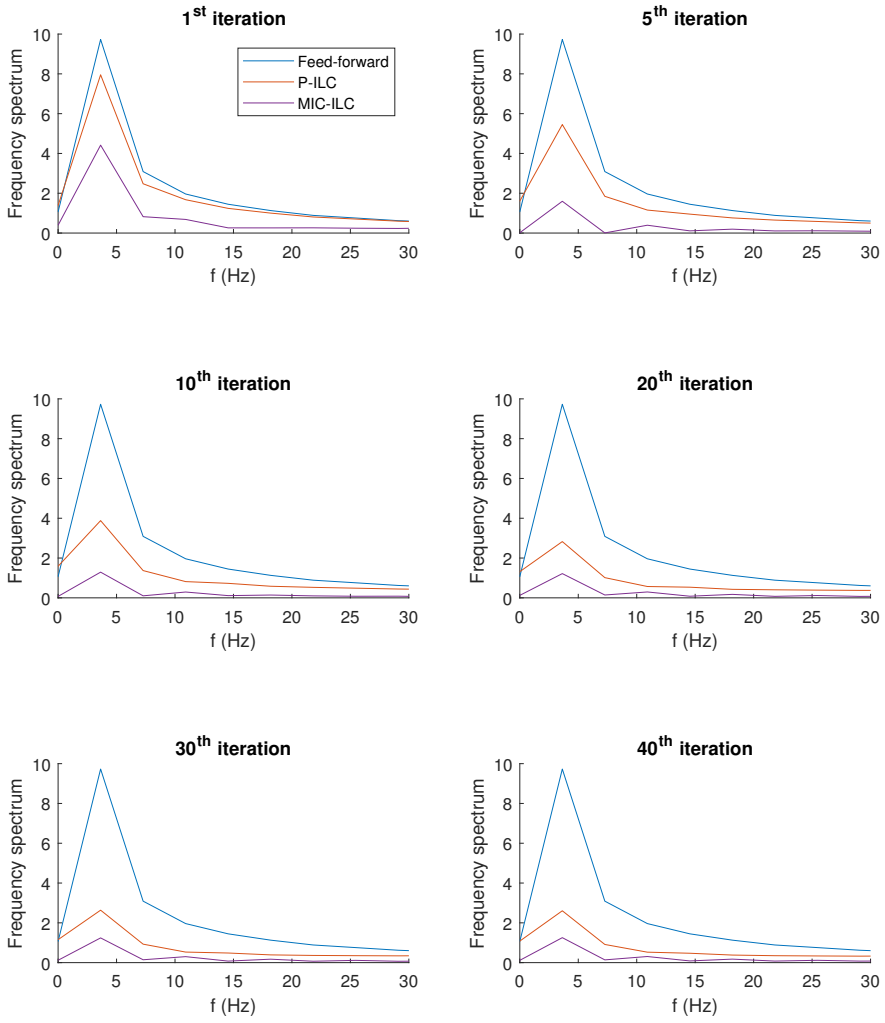


Fig. 3.13: Single-sided power spectrum of the pressure error signal, for the three different controllers.

We could try to improve the convergence rate of the P-ILC algorithm increasing the learning gain value in (3.13). Increasing K_P from 0.0002 to 0.0006, yields the frequency response shown in Fig. 3.14. As it has been pointed

out in the figure, at the frequency 83.4 rad/s the response gets out of the unit circle. Therefore, the frequencies bigger than 83.4 rad/s will not get attenuated and the system will get unstable.

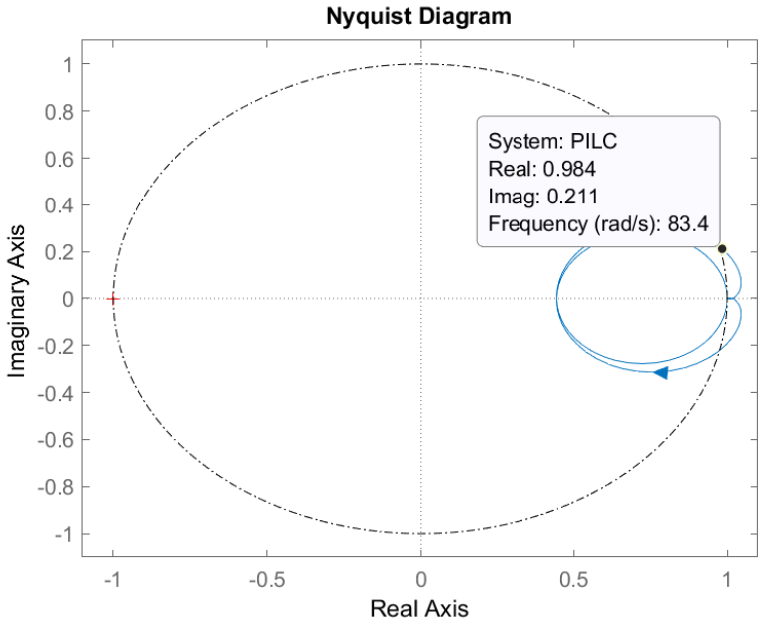


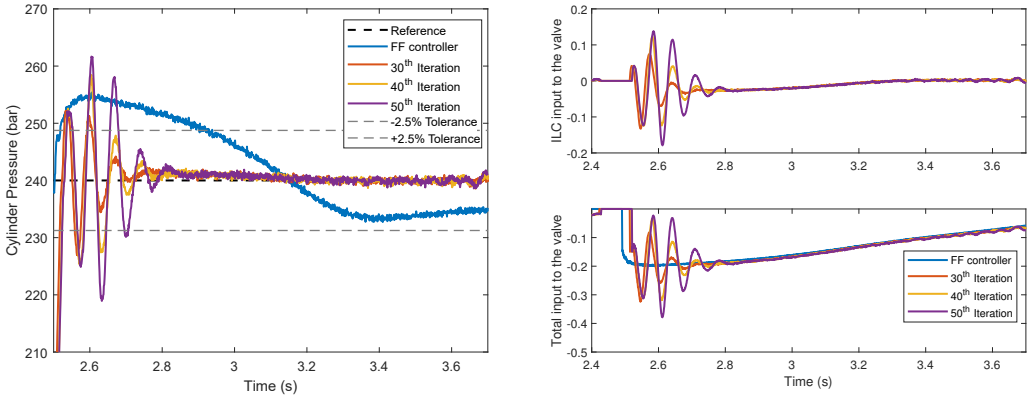
Fig. 3.14: $|1 - G(jw)S(jw)L(jw)|$ frequency response with $K_P = 0.0006$ for the P-ILC algorithm.

We could make use of Q to enlarge the stability circle so the frequency response of the P-ILC algorithm remains stable. By setting a $Q = 0.94$, the stability circle would be large enough so the algorithm response is stable at all frequencies. However, we want to see how the frequency response of the algorithm out of the unit circle affects the pressure tracking.

To that end, the same scenario than before is considered, a 240 bar reference tracking, but with a $K_P = 0.006$ 1/bar. The test outcomes can be seen in Fig. 3.15a, as iterations go on the amplifications at the beginning of the step increase, yielding an unstable response. This is a consequence of the ILC input to the valve, shown in Fig. 3.15b, that gets unstable and introduces oscillations into the valve total input.

If we analyze the pressure signal amplifications, the frequency of excitation of the oscillations is the same as the frequency at which the response of the P-ILC gets out of the unit circle. The oscillations cycle time is 0.0739s with a fundamental frequency of 13.54 Hz, which converted to rad/s results in 85 rad/s. This frequency coincides with the value at which the frequency

response of the P-ILC gets out of the unit circle, as it is pointed out in Fig. 3.14.



(a) (b)

Fig. 3.15: Pressure and input signal for a 240 bar reference with the P-ILC algorithm.

The RMSE between the P-ILC and the MIC-ILC for a 240 bar reference is shown in Fig. 3.16. At the beginning, the P-ILC converges faster than in Fig. 3.12, however it gets unstable from iteration 10 on, due to the bad learning gain design.

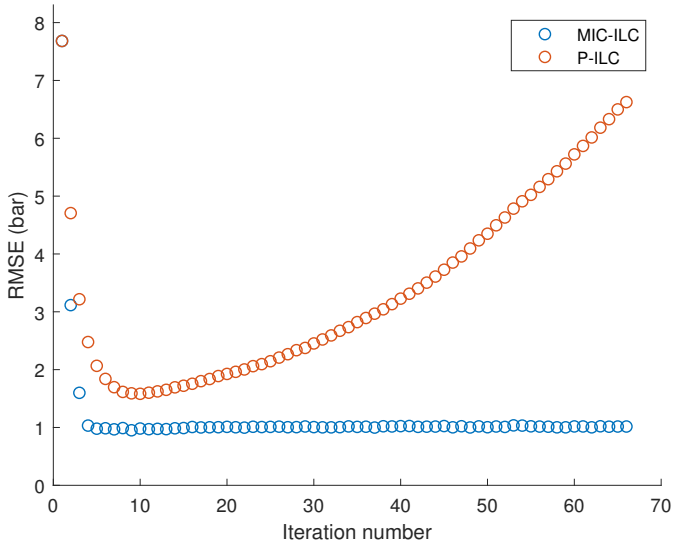


Fig. 3.16: RMSE between the cylinder pressure and the 240 bar reference over iterations with the P-ILC and the MIC-ILC algorithms.

From the above, we can see that the graphical frequency response analysis can help us in the design of the leaning filter, not only to increase the convergence rate but also to avoid algorithm instability.

3.6.2 Conventional SISO model inverse ILC

It has been shown that the MIC-ILC outperforms the P-ILC algorithm. Now, through this section we compare the MIC-ILC algorithm with the model inverse ILC (MI-ILC) approach proposed by [56, 57, 58, 59].

Following the design procedure in the above mentioned studies, the learning filter is thus:

$$L_{MI}(s) = (G_{zpf}^{-1}(s) + C(s))\frac{\omega_n^2}{(s-\omega_n)^2}\frac{\omega_n^2}{(s+\omega_n)^2}, \tag{3.14}$$

a fourth-order low-pass filter is added to $L_{MI}(s)$, and ZPF is applied to avoid phase-loss.

Note the difference with the MIC-ILC design in (3.12), where the low-pass filter is only applied to the plant inverse, and not to the sum of the plant and the controller. This variation affects directly the convergence rate of the algorithm, as the model differences are bigger and the frequency response at low frequencies lies further from the origin.

The above can be seen in Fig. 3.17, where the frequency responses of both algorithms are shown. The 50 rad/s frequency point has been pointed out in both frequency responses. In the MIC-ILC response, the frequency point lies at a radial distance from the origin of 0.2402, and in the MI-ILC response, the point lies at a radial distance of 0.3606. Although the MI-ILC design has a good frequency response at low frequencies, it will not achieve as good a convergence rate as that of the MIC-ILC.

In Fig. 3.18, the pressure signals and control inputs for a 240 bar reference tracking are shown. The MI-ILC achieves good convergence rate, as at iteration eight good pressure tracking is obtained.

A comparison of the RMSE between the MI-ILC and the MIC-ILC algorithm is shown in Fig. 3.19. The MI-ILC achives similar convergence rate than the MIC-ILC, however, the value towards which the MI-ILC converges is slightly bigger than with the MIC-ILC.

Although both algorithms look similar in terms of RMSE value and convergence rate, we may look at the settling time the pressure signal takes to stabilize within ± 1 bar difference with respect to the reference. Figure 3.20, shows the settling time value at each iteration for both algorithms. The settling time obtained with the MIC-ILC is 200ms whereas with the MI-ILC is of 250ms. It is improved by 20%.

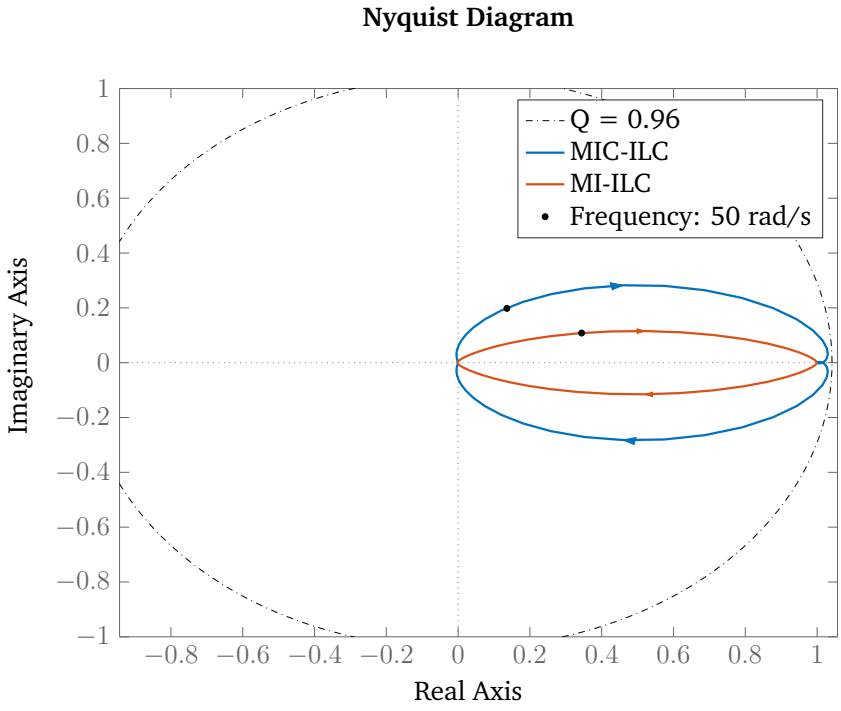
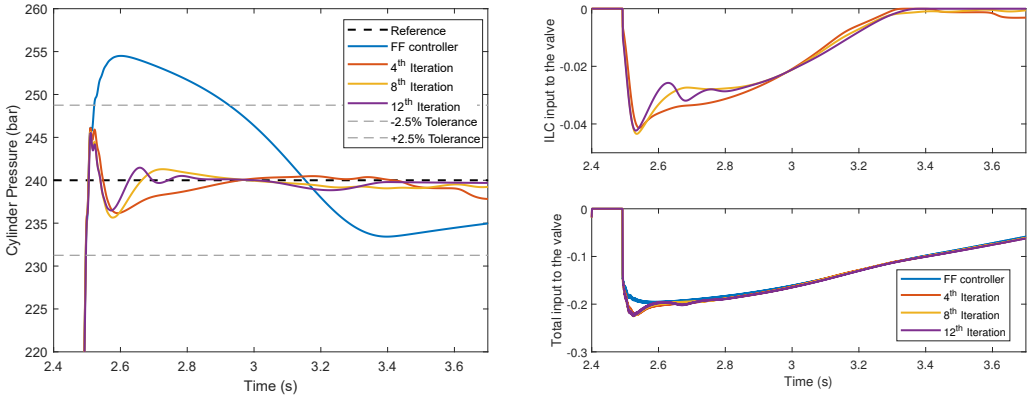


Fig. 3.17: $|1 - G(jw)S(jw)L(jw)|$ frequency response comparison for MI-ILC and MIC-ILC algorithms.



(a) **(b)**

Fig. 3.18: Pressure and input signal for a 240 bar reference with the MI-ILC algorithm.

The pressure error signal power spectrum is shown in Fig. 3.21 with the three ILC algorithms. In line with the results obtained throughout the section, the MI-ILC outperforms the P-ILC, due to its design at low frequencies, but the MIC-ILC reduces faster those frequencies.

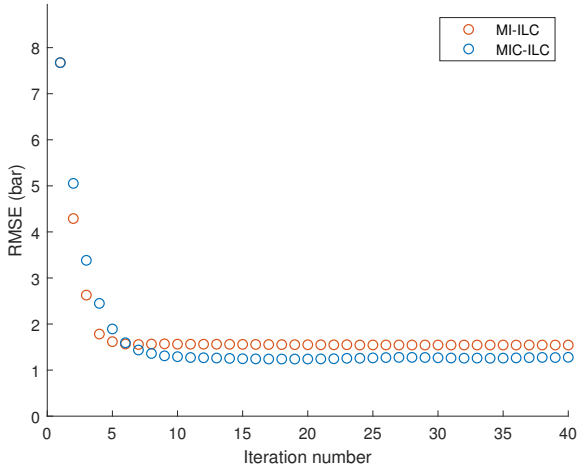


Fig. 3.19: RMSE between the cylinder pressure and the 240 bar reference over iterations with the MI-ILC and the MIC-ILC algorithms.

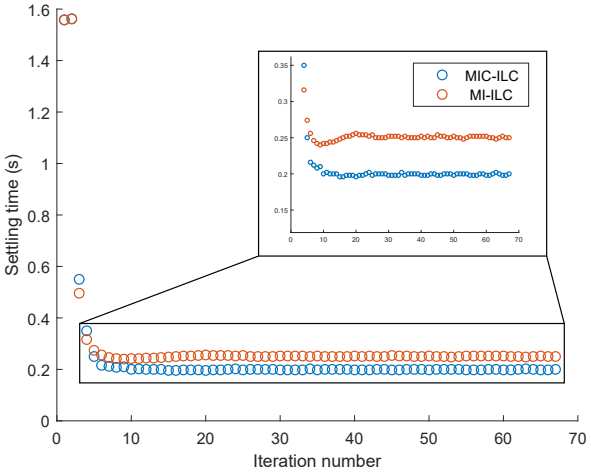


Fig. 3.20: Settling time to within 1 bar with respect to the reference for the MI-ILC and the MIC-ILC algorithms.

3.7 Conclusions to Chapter 3

To improve the force control of a hydraulic actuator and meet the specifications set by the costumers, an ILC algorithm is proposed. Regarding the application of ILC in the hydraulic system, as far as this author has been able to find, no work exists providing a convergence rate and stability analysis for ILC in hydraulic systems.

In this chapter, we have proposed a design methodology for a model-based ILC, to speed up the learning process. The design is based on the known

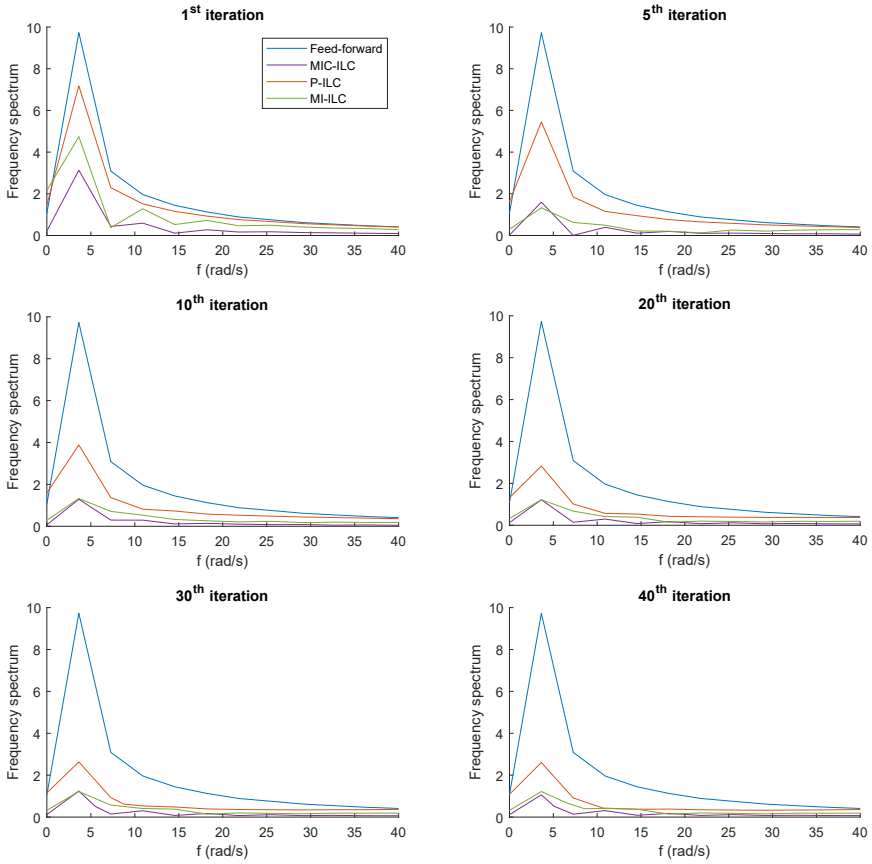


Fig. 3.21: Single-sided power spectrum of the pressure error signal, for the three ILC algorithms and FF controller.

plant dynamics, from which the unknown high frequency components have been discarded. A low-pass filter has been included to achieve robustness towards plant uncertainty, and ZPF has been applied to avoid phase loss.

The proposed graphical analysis for the algorithm performance has proved effective to achieve fast convergence rate and stability. The closer the algorithm frequency response to the origin, the faster those frequencies will be corrected, and a faster convergence rate will be achieved. Furthermore, in the case the algorithm frequency response gets out of the stability circle, one can detect which frequencies will be unstable, in order to attenuate them.

A performance and stability comparison has been provided with other ILC algorithms. A proportional-ILC has been used as it is the most common ILC algorithm used in industry. To compare the proposed design with other model-based approaches, the model-based ILC algorithm proposed by [56,

57, 58, 59] has been designed. The proposed MIC-ILC algorithm achieves better convergence rate and stability than other methods. The algorithm has been implemented in high-fidelity simulations of a mechanical press, in which precise force reference tracking has been obtained.

4

Hydraulic press MIMO position control

4.1 Introduction

A hydraulic press is a device that uses hydraulic cylinders to generate a compressive force. It has traditionally been used for high-force applications due to its easy operation and adaptability to suit a wide range of forming conditions. One of the advantages of hydraulic presses is that the force can be applied throughout the stroke, unlike mechanical presses where the force is maximum at the bottom.

Figure 4.1 shows the drawing of a hydraulic press comprising a hydraulic slide and a cushion. Unlike mechanical presses, the forming force that the slide applies to the workpiece is carried out by a hydraulic circuit that controls the slide position and force.

A typical hydraulic press operation is shown in Fig. 4.2, which consists of four main phases. During the *Free Fall* phase, the slide rapidly descends under its own weight. During the *Drawing* phase, the stroke of the slide with the workpiece takes place and the slide should maintain the specified position reference until it reaches the BDC. In the *Making force* phase, force control instead of position control is done. To guarantee the correct forming of the workpiece, a specific pressure is held during the entire *Making force* phase. After the forming is completed, the slide quickly returns to its initial position at TDC during the *Fast Rise* phase.

4.2 Hydraulic press control

Traditionally, the position and force control of a hydraulic slide has been carried out via valve-controlled or pump-controlled circuits. Valve-controlled hydraulic circuits provide fast and high frequency response behavior and accurate performance. Pump-controlled hydraulic systems provide efficient

Fig. 4.1: Hydraulic press drawing, with a hydraulic slide.

operation, as the delivered flow rate is defined by the adjustable displacement of the pump [67].

The working operation of valve-controlled and pump-controlled hydraulic circuits will be explained in the subsequents subsections. This will lead to an analysis of the current limitations of both controllers, from which a new position control approach will be proposed.

4.2.1 Valve-controlled hydraulic press

The explanation of a valve-controlled hydraulic slide is based on a typical hydraulic circuit, shown in Fig. 4.3. It consists of a double-acting cylinder with both chambers connected to pressure relief valves, to limit the pressure in the system. Two proportional valves are installed to control the pressure inside the cylinder chambers and the piston position. As the piston rod is connected to the slide, by controlling the piston position we directly control the slide. The variable displacement axial piston pump provides the necessary flow rate to control the cylinder pressure and position.

At the TDC position, the piston is completely retracted, and both proportional valves are kept closed and the pump delivers no flow. It is the rod-side

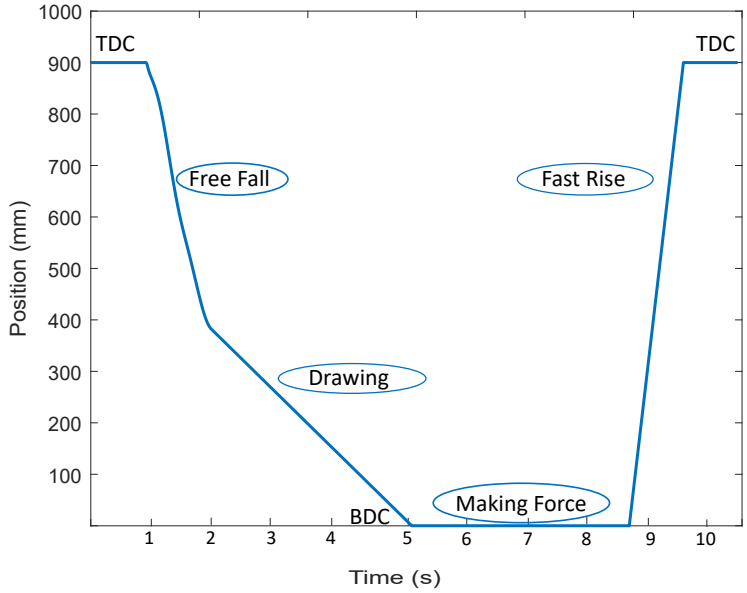


Fig. 4.2: Hydraulic slide press cycle.

cylinder chamber that holds the slide weight to keep it at the TDC. The rod-side relief valve cracking pressure is set to be slightly bigger than the maximum design weight of the slide. Else, the relief valve would open due to the pressure resulting from the slide weight, channeling the hydraulic fluid from the rod-side chamber to the tank, extending the cylinder piston.

During the *Free fall* phase, the rod-side proportional valve controls the slide velocity. A constant spool position is set, which depends on the desired falling velocity. The bigger the valve opening the faster the slide will fall and, on the contrary, the smaller the valve opening the slower the slide falling velocity. The valve spool position that determines the slide falling velocity is set manually.

The transition between the *Free fall* and *Drawing* phases is carried out by progressively closing the rod-side proportional valve. A closing ramp is specified to the valve that slows down the slide. Care has to be taken in the design of the closing ramp, as the slide slowing down could yield a rebound which, if increased excessively, could prolong the cycling time of the hydraulic press.

In the *Drawing* phase, after the progressive closing of the rod-side proportional valve, the piston-side proportional valve controls the slide position. As the rod-side proportional valve is closed and the slide continues falling, the pressure in the rod-side chamber increases until the pressure relief valve cracking pressure is reached. At this point, the oil is channeled through the

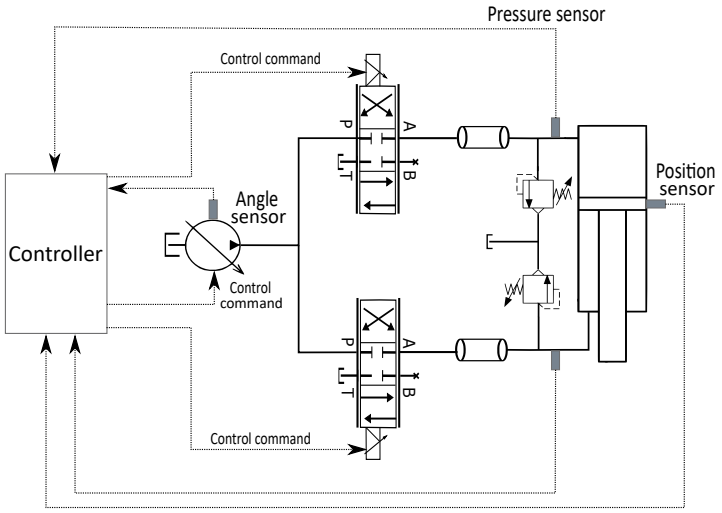


Fig. 4.3: Slide hydraulic circuit.

rod-side relief valve. The *Free fall* and *Drawing* phases of a valve-controlled hydraulic slide and every signal discussed above are shown in Fig. 4.4.

During the *Free fall* phase, the piston-side proportional valve is opened completely so the pump flow extends the piston. The desired falling velocity is obtained with a constant -50% position of the rod-side proportional valve. At this position, the fluid from the rod-side chamber is channeled through port A to port T.

The transition between the *Free fall* and *Drawing* phases is done with a predefined closing ramp of the rod-side proportional valve. This closing results in a deceleration of the slide velocity causing a small rebounding in the slide position signal at $t \approx 1.2$s.

During the *Drawing* phase, once the rod-side proportional valve is closed, the pressure in the rod-side chamber increases until it reaches the relief valve cracking pressure of 180 bar. The slide position is controlled with the piston-side proportional valve, which is opened by the controller at a +10% spool position to obtain a constant falling velocity of -56 mm/s.

Once the slide reaches the BDC, the *Making Force* phase starts. During this phase, as the cylinder remains completely extended, the force control is carried out in the piston-side chamber. The flow delivered by the pump is accumulated in the piston-side chamber and the chamber pressure increases. As soon as the desired pressure is reached, the piston-side proportional valve is closed, causing a severe pressure increase in the valve inflow, which can damage the actuator.

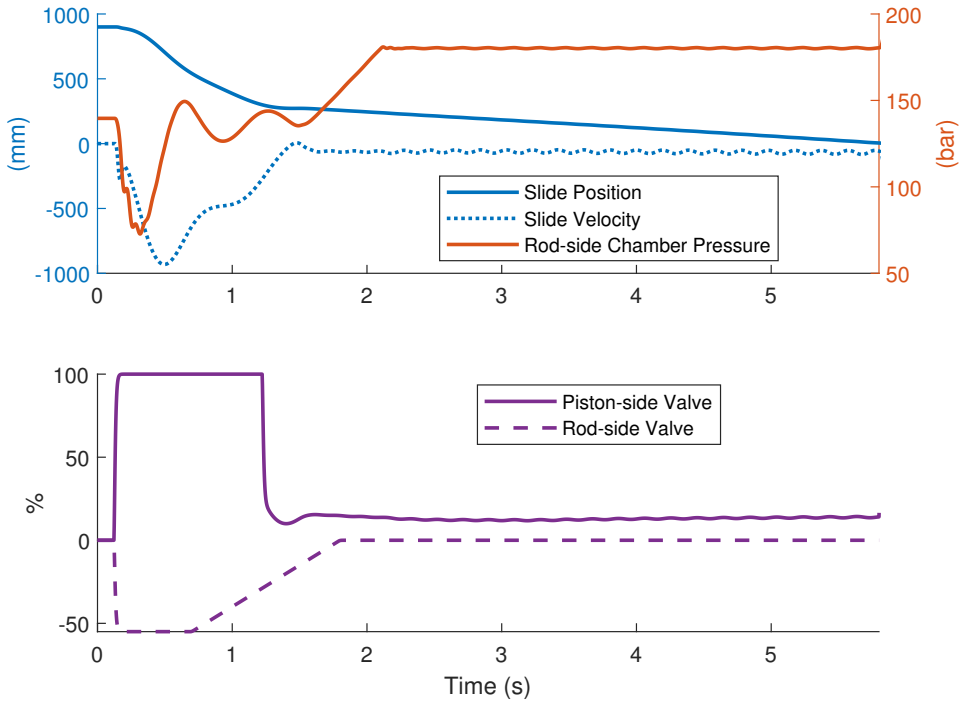


Fig. 4.4: *Free Fall* and *Drawing* phases of a valve-controlled hydraulic slide.

In the *Fast Rise* phase, the rod-side proportional valve is opened completely, moving the spool to the left position, connecting port P with A and port B with T. The fluid from the pump line retracts the cylinder piston back to the initial position, TDC. The *Making Force* and *Fast Rise* phases of a valve-controlled hydraulic slide and every signal discussed above are shown in Fig. 4.5.

During the *Making Force* phase, the piston-side proportional valve opens completely so the pump flow goes to the piston-side chamber and pressurizes it. When the pressure reference of 230 bar is reached, the valve closes completely, and the pressure is maintained until $t \approx 9.8$s, when the *Making Force* phase finishes.

Note that the abrupt closing of the proportional valve results in a pressure increase in the valve inflow that reaches 314 bar. This can cause damage or malfunctioning of the actuator.

In the *Fast Rise* phase, the rod-side valve opens completely, connecting port P with port A, to retract the piston to the initial position TDC.

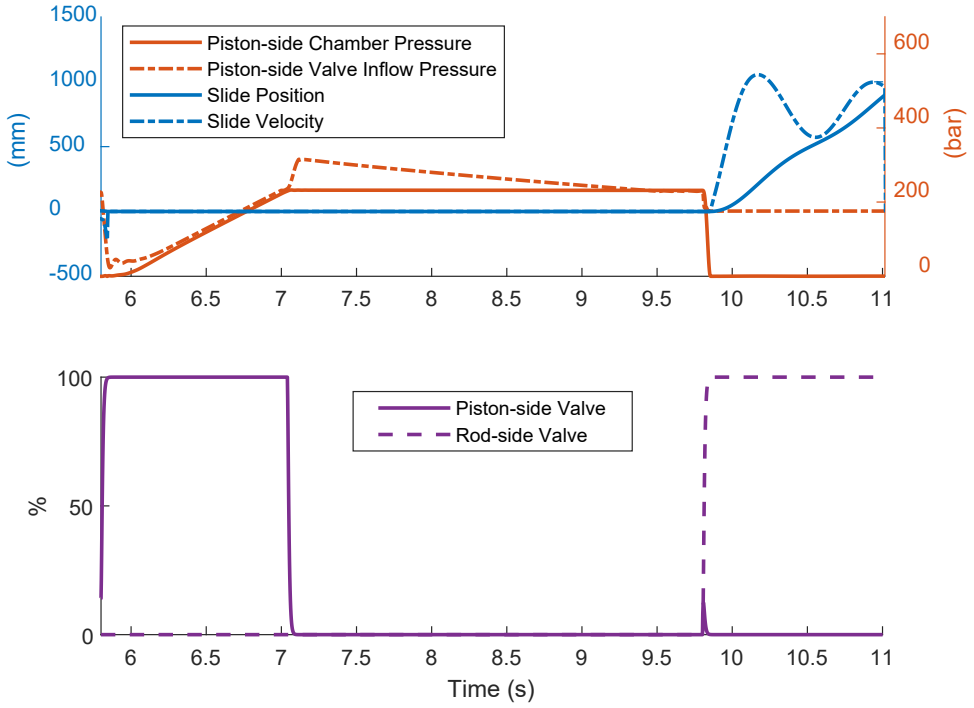


Fig. 4.5: *Making Force* and *Fast rise* phases of a valve-controlled hydraulic slide.

4.2.2 Pump-controlled hydraulic press

In the same way as in Section 4.2.1, the pump-controlled hydraulic slide explanation is based on the hydraulic circuit shown in Fig. 4.3. Pump and valve-controlled hydraulic slides follow the same procedure during the *Free Fall* phase and the transition between the *Free Fall* and *Drawing* phases.

In the *Drawing* phase, however, the piston-side proportional valve is opened completely and the slide position is controlled by the pump. A position reference is set to the pump controller, which modifies the pump swash angle according to the desired slide position.

The *Free fall* and *Drawing* phases of a pump-controlled hydraulic slide and the signal discussed above are shown in Fig. 4.6.

The rod-side proportional valve controls the *Free Fall* phase and the transition between the *Free Fall* and *Drawing* phases. A constant spool position is set that is closed progressively during the two phases transition.

In the *Drawing* phase, the piston-side proportional valve is opened completely, while the rod-side valve is closed. The pump controller opens the swash angle around 70%, to control the slide position. Similar to valve control, at

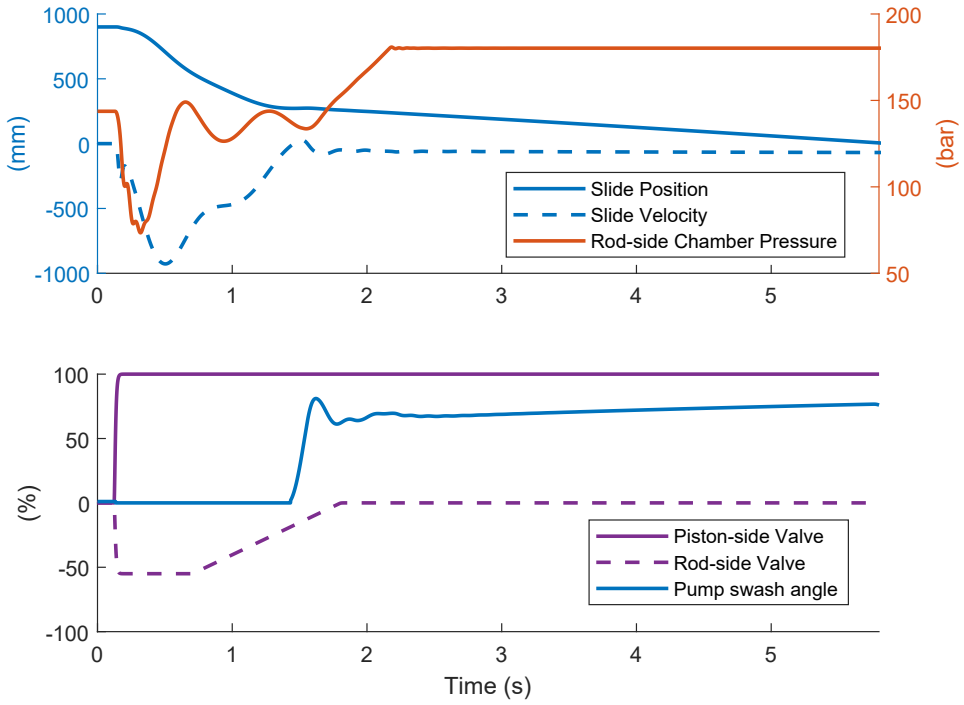


Fig. 4.6: *Free Fall* and *Drawing* phases of a pump-controlled hydraulic slide.

$t \approx 9.8$s the relief valve cracking pressure of 180 bar is reached and the fluid in the rod-side chamber is channeled to the tank.

At the BDC the *Making Force* phase starts, the pump controller opens the swash plate angle completely to reach the 230 bar reference as fast as possible. Once the desired pressure reference is reached the pump stops delivering flow. However, due to the slow dynamics of the pump, until the pump swash angle is completely closed it continues delivering oil resulting in an overshoot of the pressure reference tracking.

The *Making Force* and *Fast Rise* phases of a pump-controlled hydraulic slide and the above mentioned signals are shown in Fig. 4.7.

The pump controller switches from position control during the *Drawing* phase to force control in the *Making Force* phase. The pump swash angle is opened completely, until the piston-side chamber pressure reaches 230 bar, at $t \approx 6.7$s. The pump swash angle is closed, but as it has been pointed out in Fig. 4.7, due to the pump slow dynamics there exists a pressure overshoot. This overshoot should be controlled within a limit, else the workpiece could be damaged.

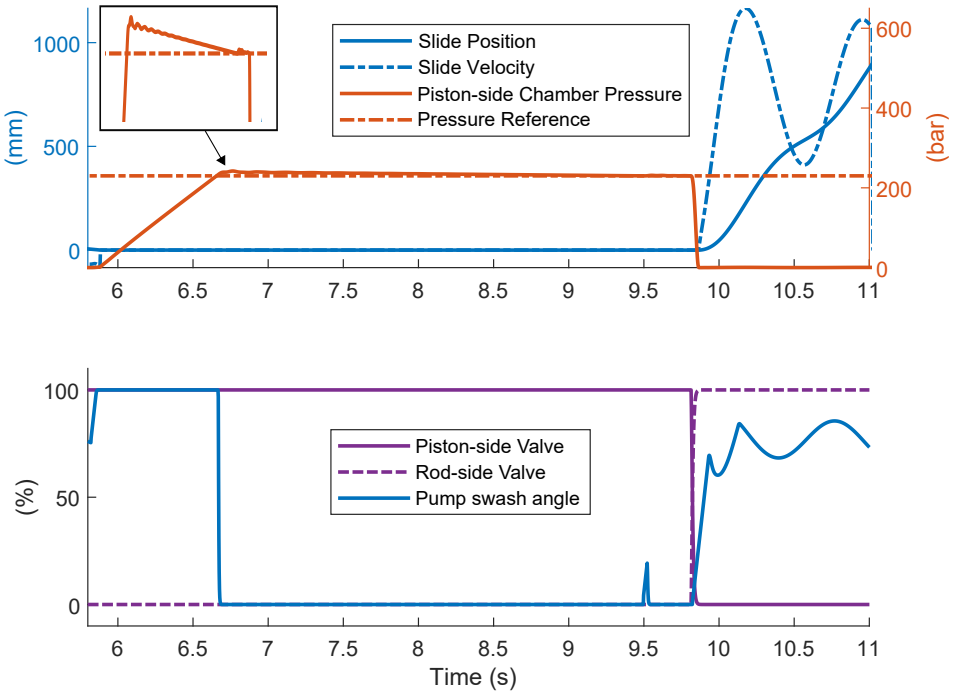


Fig. 4.7: *Making Force* and *Fast Rise* phases of a pump-controlled hydraulic slide.

Once the *Making Force* is finished, the piston-side valve closes and the rod-side valve opens. The pump delivers the necessary flow to return the slide to the initial TDC position.

4.2.3 Enhanced hydraulic press control

Two control approaches have been presented, which illustrate the design and performance of the control schemes used nowadays for the working operation of a hydraulic press. Both control approaches are subjected to manually define control signals to the rod-side proportional valve during the *Free Fall* and *Drawing* phases position control. First, a constant spool position has to be set for the *Free Fall* phase falling velocity. Then, a progressive closing has to be specified for the *Free Fall* and *Drawing* phases transition. This process could turn tedious, as every time the position reference is changed the control signals need to be modified accordingly.

The analysis of the two control approaches brings up a major challenge, the possibility to automate the position control process during the *Free Fall* and the *Drawing* phases. To automate the rather tedious hydraulic slide

position control, we propose to control the rod-side chamber pressure with the variable axial piston pump, while controlling the slide position with the rod-side proportional valve. The position controller will not differentiate between phases, as the control will be carried out continuously from the start of the *Free Fall* phase to the end of the *Drawing* phase, regardless of the position reference.

It is necessary to guarantee a certain pressure level in the auxiliary chamber during the *Drawing* phase, as the forming of the workpiece is carried out in this phase. In the event that more force is required for the workpiece forming, if the auxiliary chamber is pressurized at a certain level it will take less effort for the pump to supply said force. Else, if no pressure existed in the auxiliary chamber, the main chamber would have to be pressurized completely before more force could be applied.

This control proposal eliminates the need to manually define any control signal and adapts automatically to every position reference required. Such a controller needs a MIMO structure, as the cylinder position and rod-side chamber pressure need to be controlled at the same time.

As a first approach to the position MIMO control problem, two PI controllers are designed to control the slide position and the rod-side chamber pressure. However, the regulation of these controls is not straightforward as both control loops are coupled.

Figure 4.8 shows the proposed MIMO position control with two PI controllers during the *Free Fall* and the *Drawing* phases. The position control is poor, as the two control loops are highly coupled.

During the *Free Fall* phase, no pressure control is carried out as the rod-side chamber pressure is given by the slide weight, therefore, good position control is obtained. However, in the two phases transition, the controller closes the valve to reduce the slide velocity, which produces a pressure increase in the rod-side chamber. To counteract this pressure increase, the pump controller reduces the swash angle, which results in a slowing down of the slide at $t \approx 2.5$s and a pressure decrease.

To reach the pressure reference, the pump swash angle is opened completely to deliver more flow. This flow increase speeds up the slide falling velocity and the valve controller reacts, closing the valve to slow down the slide and reach the position reference, from $t = 2.5$s to $t = 4$s.

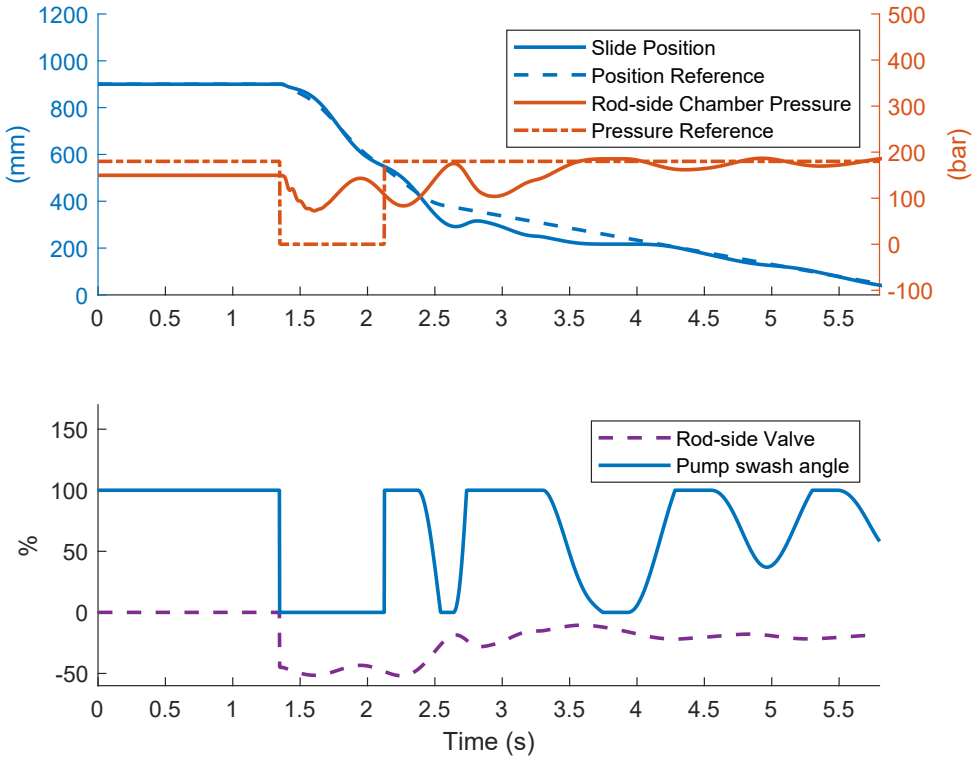


Fig. 4.8: Position control during the *Free Fall* and the *Drawing* phases with MIMO PI control.

4.3 Conclusions to Chapter 4

The working operation of hydraulic presses via valve-controlled and pump-controlled circuits has been analyzed. From the analysis, some limitations have been pointed out, such as the requirement to manually define control signals to the rod-side proportional valve, during the *Free Fall* and *Drawing* phases. The control signals are subjected to a specific position reference, and every time the position reference is modified the manually defined signals need to be redefined accordingly.

We have proposed an enhanced hydraulic press control, to automate position control and avoid defining signals manually. We propose to control the rod-side chamber pressure with the variable axial piston pump while controlling the slide position with the rod-side proportional valve. As a first approach we try to control the pump and the valve with two PI controllers. However, as we have seen in Section 4.2.3, the pump and valve control loops are highly

coupled and, together with the nonlinear behavior of the hydraulic system, yield poor position control of the hydraulic press.

Therefore, in order to improve the performance of the PI controllers in the position MIMO control problem, in Chapter 5, a revision of the existing MIMO controllers is carried out, which yields an innovative MIMO position controller scheme based on ILC.

5

Iterative Learning Control in hydraulic press position control

5.1 Introduction

Nowadays, the working operation of a hydraulic press requires the need of manually define constant spool positions and closing ramps to a proportional valve. In Section 4.2.3, we have proposed a new MIMO position control, so the hydraulic press operation can be automated, by controlling the pump and the valve simultaneously. However, as already concluded in Section 4.2.3, the control is not straightforward, as pump and valve control loops are coupled.

Literature has already investigated the problem of hydraulic coupling control. In [68], a pump and a pressure relief valve are used to implement the pressure control of a compression machine, to reduce the power consumption based on fuzzy logic. In [69], a MIMO fuzzy controller is proposed for the position and force control of an electro-hydraulic system. In these two studies, compared to a PID controller, the results obtained achieved better performance. However, we aim to automatize the hydraulic press operation so it can automatically adapt to every force and position scenario. This opposes the idea of heuristic strategies, typical of the fuzzy control.

In the same line of manually defining control parameters, in [70], four single-input-multiple-output (SIMO) PI controllers are designed for the force reference trajectory tracking of a stamping machine. The SIMO PI controller gains are designed based on the estimated perturbation model, however, as the authors point out, the fine-tuning of the controller could result in a time-consuming and expensive process.

To counteract the coupling effect, FF and feedback control have been widely studied in the literature for the hydraulic systems decoupling. In [71] the pressure and velocity of a hydraulic actuator are decoupled to eliminate

unstable valve behavior by velocity feedback. In [72], a FF control scheme is presented for decoupled pressure control in an actuator. In [73], a MIMO inversion-based FF controller designed by input-output linearization is implemented for a hydraulic test bed. However, these works focus on the energy efficiency of the hydraulic system rather than on the control performance. Furthermore, as concluded through Section 2.3, FF control performs correctly only to the extent the system is accurately known.

Considering the existing limitations of the MIMO control approaches in hydraulic circuits, we propose ILC as a solution to the MIMO position control problem. We design the ILC algorithm to decouple the pump and valve control loops and, obtain precise position and force tracking. Therefore, the challenge remains to guarantee stability and high convergence rate for multiple variables.

Several studies have been realized regarding the design of MIMO ILC algorithms. In [74], a conjugate-gradient ILC was proposed to guarantee fast monotonic convergence without the need to calculate the inverse of the system matrix. This method converged fast at the first iterations, but the algorithm performance suffered from the control loop couplings, penalizing the convergence rate later on. This was improved in [75], with a Quasi-Newton optimized ILC. However, to guarantee stability in nonlinear systems with modeling uncertainties, a small gain was included in the learning function, which required more iterations to converge.

In fact, several optimization-based MIMO ILC algorithms have been proposed recently, in which the learning function is based on an error minimization function [76, 77, 78]. These approaches rely on the tuning of the weighting matrix in order to obtain a global minimum in the minimization problem. It is hard to explicitly define the ILC multiple objectives in a cost function. In such a cost function we would have to include a condition for fast convergence, robustness towards uncertainties, long-term stability and satisfy the input constraints. This approach requires a too complex optimization problem.

To avoid the dependency on tuning parameters, model-free MIMO ILC approaches were proposed in [79, 80, 81], to reduce the modeling requirement. However, not including the system dynamics in the learning design decreases the convergence rate of the system, as in order to achieve good performance, knowledge of the plant must be assumed [82].

In [83], an overview of which MIMO ILC design approach to use, illustrated on an industrial printer, is given. It concludes that the centralized, model-based, approach achieves the better performance in terms on convergence,

however it requires more designing and computation effort, in comparison to the decentralized, model-free, approach.

In line with [83], we have shown in Chapter 3 that the model-based ILC outperforms other ILC algorithms, when the known dynamics and the feedback controller are included in the learning model design. We can manage the modeling design effort better than a slow convergence. Investing in an accurate plant model design can optimize the convergence of the algorithm, which results in commissioning cost savings. Else, the slower the convergence, the more iterations will be required to obtain an appropriate workpiece forming that fulfills all the design requisites.

In the following section, the MIC-ILC algorithm proposed in Chapter 3 will be extended to the MIMO case, to obtain precise hydraulic press position control.

5.2 MIMO ILC design

The hydraulic slide circuit can be regarded as a general time invariant MIMO linear system:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}_c\boldsymbol{x}(t) + \boldsymbol{B}_c\boldsymbol{u}(t) \qquad \boldsymbol{y}(t) = \boldsymbol{C}_c\boldsymbol{x}(t), \tag{5.1}$$

where $\boldsymbol{x}$ is the four-dimensional state vector, consisting of the piston position and velocity, and the pressure inside the two cylinder chambers. $\boldsymbol{u}$ is the two-dimensional control input vector, consisting of the pump swash angle and the valve spool position. $\boldsymbol{y}$ is the two-dimensional output vector, consisting of the piston position and the cylinder rod-side chamber pressure. $\boldsymbol{A}_c$, $\boldsymbol{B}_c$ and $\boldsymbol{C}_c$ are matrices with appropriate dimensions.

In this section, signals in the time domain denoted with bold style, denote vectors. In the same way, capital Laplace transforms in bold style, denote transfer matrices.

The Laplace transformed system transfer matrix is:

$$\boldsymbol{G}(s) = \boldsymbol{C}_c(s\boldsymbol{I} - \boldsymbol{A}_c)^{-1}\boldsymbol{B}_c. \tag{5.2}$$

Note that $\boldsymbol{G}(s)$ is a two-by-two transfer matrix.

The error propagation equation, that relates the current error vector with the next, is derived in Appendix A and shown in (5.3):

$$\boldsymbol{E}_{j+1}(s) = (\boldsymbol{I} - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{L}(s))\boldsymbol{E}_j(s), \tag{5.3}$$

where $\boldsymbol{S}(s)$ is the sensitivity transfer matrix $\boldsymbol{S}(s) = (\boldsymbol{I} + \boldsymbol{G}(s)\boldsymbol{C}(s))^{-1}$, and $\boldsymbol{L}(s)$ the learning transfer matrix.

As explained in Section 4.2, the hydraulic press circuit consists of two control loops, each of them containing a PI controller. $\boldsymbol{C}(s)$ is the controller transfer matrix containing the pump PI controller, $C_P(s)$, and the valve PI controller, $C_V(s)$, transfer functions:

$$\boldsymbol{C}(s) = \begin{bmatrix} C_P(s) & 0 \\ 0 & C_V(s) \end{bmatrix}. \tag{5.4}$$

From equation (5.3), if the term $\boldsymbol{\Lambda}(s) = \boldsymbol{I} - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{L}(s)$ is less than one, then the error vector at the next iteration will be smaller than the error at the current iteration. Otherwise, the error would diverge.

The analysis of the convergence properties of $\boldsymbol{\Lambda}(s)$ is a key focus to guarantee the stability of the algorithm. According to [1], there are two convergence conditions: monotonic stability (MS) and asymptotic stability (AS). The former refers to the error getting smaller from trial to trial. The latter, refers to whether the algorithm will converge as the number of iterations goes to infinity. AS is a necessary and sufficient condition, whereas MS is a sufficient condition to ensure the error gets smaller on each trial.

Clearly, MS is a stronger condition to achieve, and not easy to verify in real applications, as it was shown in [84]. The MS condition was given by [85], and implies:

$$\bar{\sigma}(\Lambda(s)) < 1, \tag{5.5}$$

where $\bar{\sigma}(\cdot)$ denotes the maximum singular value of a matrix.

A more relaxed condition is given for the AS in [85], and implies:

$$|\lambda_i(\boldsymbol{\Lambda}(s))| \leq 1. \tag{5.6}$$

The tracking error will converge to zero as iterations go on if $\boldsymbol{\Lambda}(s)$ has all its eigenvalues, λ_i, within the unit circle.

We design the learning transfer matrix $\boldsymbol{L}(s)$ so the eigenvalues of $\boldsymbol{\Lambda}(s)$ are less than one in module following the AS condition. For the hydraulic press working operation, we will control the rod-side chamber pressure and the

piston position, with the pump and valve, respectively. Therererefore, the $\boldsymbol{L}(s)$ will have the following two-by-two structure:

$$\boldsymbol{L}(s) = \begin{bmatrix} L_{11}(s) & L_{12}(s) \\ L_{21}(s) & L_{22}(s) \end{bmatrix}. \tag{5.7}$$

For the model inverse $\boldsymbol{L}(s)$ design, the same approach as in Section 3.2 will be followed. For MIMO systems with parallel structure with feedback controller and ILC algorithm, the following model inverse $\boldsymbol{L}(s)$ design is proposed:

$$\begin{aligned} \boldsymbol{L}(s) =& \boldsymbol{G}^{-1}(s)\boldsymbol{S}^{-1}(s) = \boldsymbol{G}^{-1}(s)(I + \boldsymbol{G}(s)\boldsymbol{C}(s)) \\ =& \boldsymbol{G}^{-1}(s) + \boldsymbol{C}(s), \end{aligned} \tag{5.8}$$

note that (5.8) is a MIMO extension of the design shown in (3.6).

5.3 MIMO ILC learning filter design

In order to carry out the model inverse design, first, the system modeling needs to be derived. The slide hydraulic circuit is shown in Fig. 4.3, from which the linearized state-space system will be obtained.

First, in order to derive the complete hydraulic circuit model, the cylinder and pump dynamic equations need to be defined. Unlike in Chapter 2, the cylinder in Fig. 4.3 is a double-acting cylinder, the operation of which is similar to that of the single-acting cylinder except for the retraction, which is done hydraulically. Therefore, the volumetric flow rate out of the rod-side chamber need to be defined, which is as follows:

$$q_B(t) = -A_B\dot{x}(t) + (V_{\mathrm{Bd}} + A_B(l - x(t)))\beta\dot{P}_B(t), \tag{5.9}$$

where $q_B(t)$ is the volumetric flow rate out of the rod-side chamber (m^3/s), A_B is the rod-side chamber area (m^2), V_{d} is the rod-side chamber dead volume (m^3) and $P_B(t)$ is the pressure in the rod-side chamber.

The force balance equation of the cylinder, that relates both cylinder chamber pressures is:

$$F(t) = m\ddot{x}(t) = P_A(t)A_A - P_B(t)A_B, \tag{5.10}$$

where m is the cylinder moving mass (kg) and $F(t)$ is the output force of the rod (N).

The relationship between the axial piston pump outflow rate and the swash plate angle is as follows:

$$q(t) = q_N \frac{\omega}{\omega_N} \alpha(t), \tag{5.11}$$

where $q(t)$ is the volumetric flow rate out of the pump (m^3/s), q_N is the nominal flow rate (m^3/s), $\alpha(t)$ is the swash plate angle (-), ω is the shaft rational speed (rad/s) and ω_N is the nominal shaft rational speed (rad/s)

We are interested in the position control, which takes place during the *Free Fall* and *Drawing* phases. During these phases, the flow rate out of the pump will be equal to the cylinder piston-side chamber flow rate. Likewise, the flow rate out of the rod-side chamber will be equal to the flow rate through the proportional valve.

We can rearrange equation (2.1) with (5.11), to match the cylinder and the pump flows. Rearranging equation (5.9) with (2.2) we can match the cylinder and valve flow rates. From these considerations, and using (5.10) to relate both cylinder chamber pressures, we can obtain the slide hydraulic circuit system equations, from which the linearized state-space system is derived in (5.12):

$$\begin{bmatrix} \delta\ddot{x} \\ \delta\dot{x} \\ \delta\dot{P}_{\mathrm{A}} \\ \delta\dot{P}_{\mathrm{B}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & A_{13} & A_{14} \\ 1 & 0 & 0 & 0 \\ A_{31} & 0 & 0 & 0 \\ A_{41} & 0 & 0 & A_{44} \end{bmatrix} \begin{bmatrix} \delta\dot{x} \\ \delta x \\ \delta P_{\mathrm{A}} \\ \delta P_{\mathrm{B}} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ B_{31} & 0 \\ 0 & B_{42} \end{bmatrix} \begin{bmatrix} \delta\alpha \\ \delta y_{\mathrm{v}} \end{bmatrix}$$

$$\begin{bmatrix} \delta x \\ \delta P_{\mathrm{B}} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta\dot{x} & \delta x & \delta P_{\mathrm{A}} & \delta P_{\mathrm{B}} \end{bmatrix}^T. \tag{5.12}$$

The cylinder piston position and velocity, and the pressure in both chambers are chosen as state variables. The swash plate angle and the valve spool position are the system inputs; the piston position and the rod-side chamber pressure are the system outputs.

$\delta P_x = P_x - \bar{P}_x$, $\delta x = x - \bar{x}$, $\delta y_{\mathrm{v}} = y_{\mathrm{v}} - \bar{y}_{\mathrm{v}}$ and $\delta\alpha = \alpha - \bar{\alpha}$, are the small-signal deviations from an operating point obtained in steady-state conditions, $\bar{P}_x$, $\bar{x}$,

$\bar{y}_{\mathrm{v}}$ and $\bar{\alpha}$ respectively. The definition of every term A_{xx}, based on the system parameters, is shown in Appendix C.

Laplace transforming the linearized state space system in (5.12), we can obtain $\boldsymbol{G}(s)$ to use it in the $\boldsymbol{L}(s) = \boldsymbol{G}^{-1}(s) + \boldsymbol{C}(s)$ design. In the $\boldsymbol{G}(s)$ inversion, we need to be cautious, as there could be unstable zeros in the system or some dynamics could become unstable. Therefore, we first analyze $\boldsymbol{G}(s)$ poles and zeros.

$\boldsymbol{G}(s)$ is a two-by-two transfer matrix with four transfer functions with a common denominator. The denominator is a fourth-order polynomial which coefficients are shown in Table 5.1, where $V_1 = V_A + A_A\bar{x}$ and $V_2 = V_B + A_B(l - \bar{x})$.

Tab. 5.1: $\boldsymbol{G}(s)$ system denominator coefficients.

Order	Coefficient
4	$4V_1^2V_2^2\beta^2 m\omega_N\sqrt{\bar{P}_B\Delta P_{ref}}$
3	$-V_1^2V_2\beta m\omega_N K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})q_{ref}$
2	$4A_AV_1\beta\omega_n\sqrt{\bar{P}_B\Delta P_{ref}}\,(A_AV_2^2 + A_B^2V_1V_B)$
1	$4A_A^2V_2^2\beta q_N\omega\sqrt{\bar{P}_B\Delta P_{ref}} - A_A^2V_1V_2\omega_N K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})q_{ref}$ $+2A_B^2V_1^2\beta\omega_N K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})\bar{P}_B\Delta P_{ref}$
0	0

From Table 5.1 we can see that the zero-order coefficient is zero, thus we will have a pole at zero. In order to see how the system behaves, we can put Table 5.1 into numbers, using the parameters shown in Appendix C.0.2, to obtain the poles of the hydraulic slide system. These are shown in Table 5.2.

Tab. 5.2: $\boldsymbol{G}(s)$ system poles.

Pole	Damping	Frequency (rad/s)
0.00	-1.00	0.00
-2.09e-01	1.00	2.09e-01
-4.26e-03 + 5.86e+01i	7.28e-05	5.86e+01
-4.26e-03 - 5.86e+01i	7.28e-05	5.86e+01

From Table 5.2, the system has a pole at zero, a low-frequency pole and a pair of high-frequency conjugates poles. In the case there exists a mismatch between the cylinder velocity and the flow rates the pressure will move, see (2.1) and (5.9), that is why the zero pole appears. Then, the low-frequency real pole exists as, in order to change the cylinder velocity, the pressure needs to be changed first. The high-frequency complex poles have small damping,

as no friction model has been included in the slide system. These frictions usually appear between the slide and the press frame rails, and between the cylinder and piston. Some authors have included these frictions in the force balance cylinder equation, see [86, 87]. We will not consider them as they are unknown and will only affect our system at high frequencies.

We are not interested in the high-frequency poles, as these frequencies are where the uncertainties of the slide hydraulic system abound, e.g. pump and valve dynamics. From Table 5.1, we can see that the high-frequency poles result from the oil compressibility. Indeed, the effects of the oil compressibility appear only at high frequencies. Therefore, discarding β we will get rid of those frequencies.

As β is small $P_B(t)$ is large, we can neglect those terms with a larger order in β than in $P_B(t)$. In the fourth-order coefficient, β appears squared, thus the contribution of the fourth-order coefficient will be minimal. In the third-order coefficient, β does not multiply any pressure signal, the resulting term will be very small and, therefore, negligible.

After these considerations, the resulting simplified polynomial with the low-frequency poles is shown in Table 5.3.

Tab. 5.3: Low-frequency $\boldsymbol{G}(s)$ system denominator coefficients.

Order	Coefficient
2	$4A_AV_1\beta\omega_n\sqrt{\bar{P}_B\Delta P_{ref}}\,(A_AV_2^2+A_B^2V_1V_B)$
1	$4A_A^2V_2^2\beta q_N\omega\sqrt{\bar{P}_B\Delta P_{ref}}-A_A^2V_1V_2\omega_N K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})q_{ref}$ $+2A_B^2V_1^2\beta\omega_N K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})\bar{P}_B\Delta P_{ref}$
0	0

Regarding the zeros of $\boldsymbol{G}(s)$ we carry out the same procedure, see if there exists any term that can be neglected due to its small value. The numerators, $b_{g_{xx}}$, corresponding to each transfer function of $\boldsymbol{G}(s)=[g_{11}(s)\ g_{12}(s);\ g_{21}(s)\ g_{22}(s)]$ are shown in (5.13):

$$
\begin{aligned}
b_{g_{11}}(s) &= -A_AV_1V_2q_N\omega(q_{ref}K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})-4V_2\beta\sqrt{\bar{P}_B\Delta P_{ref}}s)\\
b_{g_{12}}(s) &= -2A_B\bar{P}_BV_1^2V_2\beta\omega_Nq_{ref}\sqrt{\Delta P_{ref}}s\\
b_{g_{21}}(s) &= A_AA_Bq_N\omega\sqrt{\bar{P}_B\Delta P_{ref}}\left(2V_2s+K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})q_{ref}\sqrt{\bar{P}_B}\right)\\
b_{g_{22}}(s) &= 2\bar{P}_BV_2q_{ref}\sqrt{\Delta P_{ref}}(\omega_NA_A^2V_1s\\
&\quad + q_N\omega A_A^2+\beta m\omega_NV_1^2s^3)
\end{aligned}
\tag{5.13}
$$

In the $g_{22}(s)$ numerator, $b_{g_{22}}(s)$, a high-frequency zero appears from the term $\beta m \omega_N V_1^2 s^3$. In this term, β does not multiply any pressure variable, thus, it is discarded. Resulting in:

$$\hat{b}_{g_{22}}(s) = 2\bar{P}_B V_2 q_{ref} \sqrt{\Delta P_{ref}} \left(\omega_N A_A^2 V_1 s + q_N \omega A_A^2 \right). \tag{5.14}$$

In Fig. 5.1, the Bode diagram of $\boldsymbol{G}(s)$ and the simplified system without the high-frequency zeros and poles, $\hat{\boldsymbol{G}}(s)$, is shown. At low frequencies, both systems responses are identical and, as expected, the responses deviate at high frequencies.

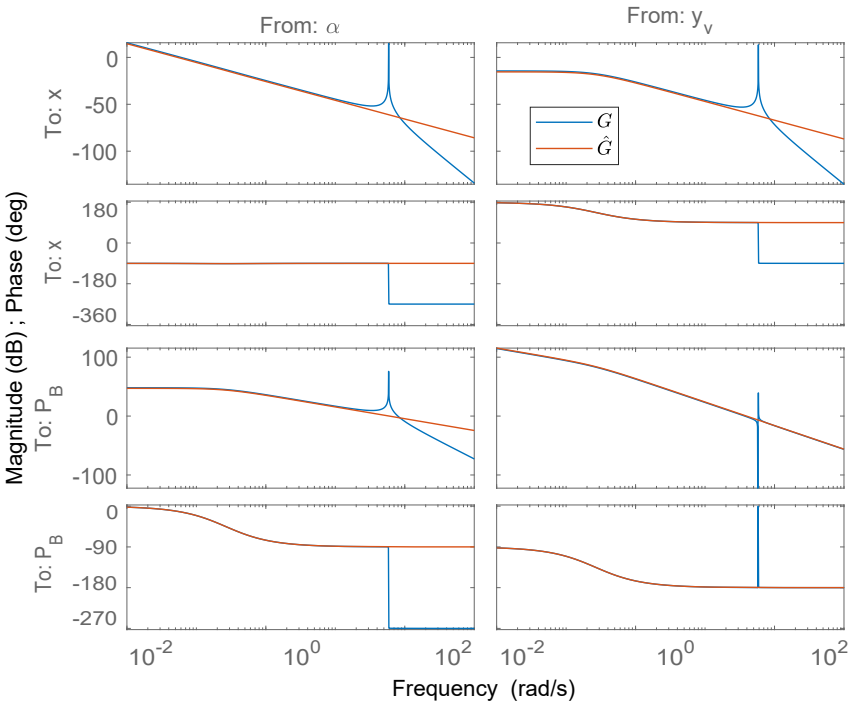


Fig. 5.1: Bode diagram of $\boldsymbol{G}(s)$ and the simplified system, $\hat{\boldsymbol{G}}(s)$, without the high frequency poles and zeros.

The new design of the learning gain matrix $\boldsymbol{L}(s)$ will be:

$$\boldsymbol{L}(s) = \hat{\boldsymbol{G}}^{-1}(s) + \boldsymbol{C}(s), \tag{5.15}$$

with the simplified system inverse.

Clearly, by including a simplified system in the learning matrix design, the convergence rate of the ILC algorithm will get affected, as some dynamics will not get simplified. To attenuate the uncertain high frequencies we include a

low-pass filter in $\hat{\boldsymbol{G}}^{-1}(s)$, and carry out ZPF, in the same way as in Section 3.3. In this way, we can filter the uncertain and undesired high frequencies without losing phase. The resulting learning gain matrix with the ZPF is:

$$\boldsymbol{L}(s) = \underbrace{\hat{\boldsymbol{G}}^{-1}(s)\frac{\omega_c^2}{(s+\omega_c)^2}\frac{\omega_c^2}{(s-\omega_c)^2}}_{\hat{\boldsymbol{G}}^{-1}_{zpf}} + \boldsymbol{C}(s), \tag{5.16}$$

where ω_c is the filter cutoff frequency in rad/s.

In Fig. 5.2, the Bode diagram of the system with ZPF is shown. The unknown high frequencies are attenuated without losing phase.

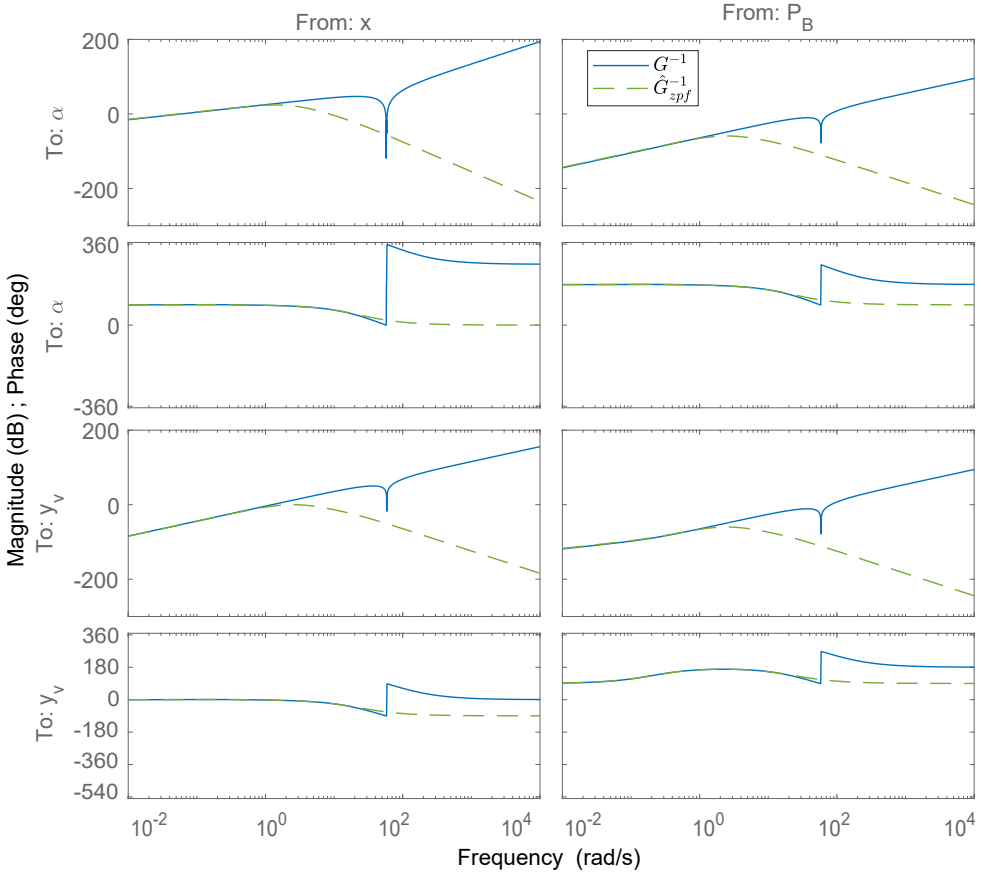


Fig. 5.2: Bode diagram of the plant inverse $\boldsymbol{G}^{-1}(s)$ and the simplified plant system with ZPF $\hat{\boldsymbol{G}}^{-1}_{zpf}(s)$.

In order to show the advantages of not affecting $\boldsymbol{C}(s)$ with the low-pass filter, we compare the plant inverse design proposed in (5.16), $\boldsymbol{L}_{MIC}(s)$, and the

traditional model inverse design used in literature [56, 57, 58, 59], $\boldsymbol{L}_{MI}(s)$, with the inverse design without system simplifications nor low-pass filter introduction $\boldsymbol{L}_D(s)$. The learning filters are defined as follows:

$$\begin{aligned}
\boldsymbol{L}_{MIC}(s) &= \hat{\boldsymbol{G}}^{-1}(s)\frac{\omega_c^2}{(s+\omega_c)^2}\frac{\omega_c^2}{(s-\omega_c)^2} + \boldsymbol{C}(s) \\
\boldsymbol{L}_{MI}(s) &= \left(\hat{\boldsymbol{G}}^{-1}(s) + \boldsymbol{C}(s)\right)\frac{\omega_c^2}{(s+\omega_c)^2}\frac{\omega_c^2}{(s-\omega_c)^2} \\
\boldsymbol{L}_D(s) &= \boldsymbol{G}^{-1}(s) + \boldsymbol{C}(s)
\end{aligned} \tag{5.17}$$

The Bode diagrams of the three different model inverse approaches are shown in Fig. 5.3. It can be seen that $\boldsymbol{L}_D(s)$ contains the high frequency dynamics of the slide hydraulic system, as no simplifications have been done and no low-pass filter has been applied.

In Fig. 5.3, the Bode diagrams (b) and (c) are not affected by the low-pass filter placement, as $\boldsymbol{C}(s)$ only consists of elements in the diagonal. However, in (a) and (d) both systems $\boldsymbol{L}_{MIC}(s)$ and $\boldsymbol{L}_{MI}(s)$ have different frequency responses from 0.2 rad/s on. In (d), it is more visible how $\boldsymbol{L}_{MIC}(s)$ is closer to the exact model than $\boldsymbol{L}_{MI}(s)$. Indeed, from frequency 0.2 rad/s to 100 rad/s, $\boldsymbol{L}_{MIC}(s)$ behaves in the same way as $\boldsymbol{L}_D(s)$.

This is a consequence of applying a really low cutoff frequency in the low-pass filter introduced, as we want to get rid of most of the high frequencies and just kept those very low frequencies that we are interested in. In case we applied a higher cutoff frequency, the differences between $\boldsymbol{L}_{MIC}(s)$ and $\boldsymbol{L}_{MI}(s)$ would reduce at low frequencies, there would only be differences at the high frequency regions.

5.4 MIMO ILC stability analysis

In Section 3.4, the stability analysis has been done by interpreting the frequency response of the SISO ILC algorithm in a Nyquist plot. However, in the MIMO ILC, the convergence condition is not given directly by the frequency response, but by the eigenvalues of the frequency response matrix.

We introduce the MIMO AS convergence condition with the new $\boldsymbol{L}(s)$ design:

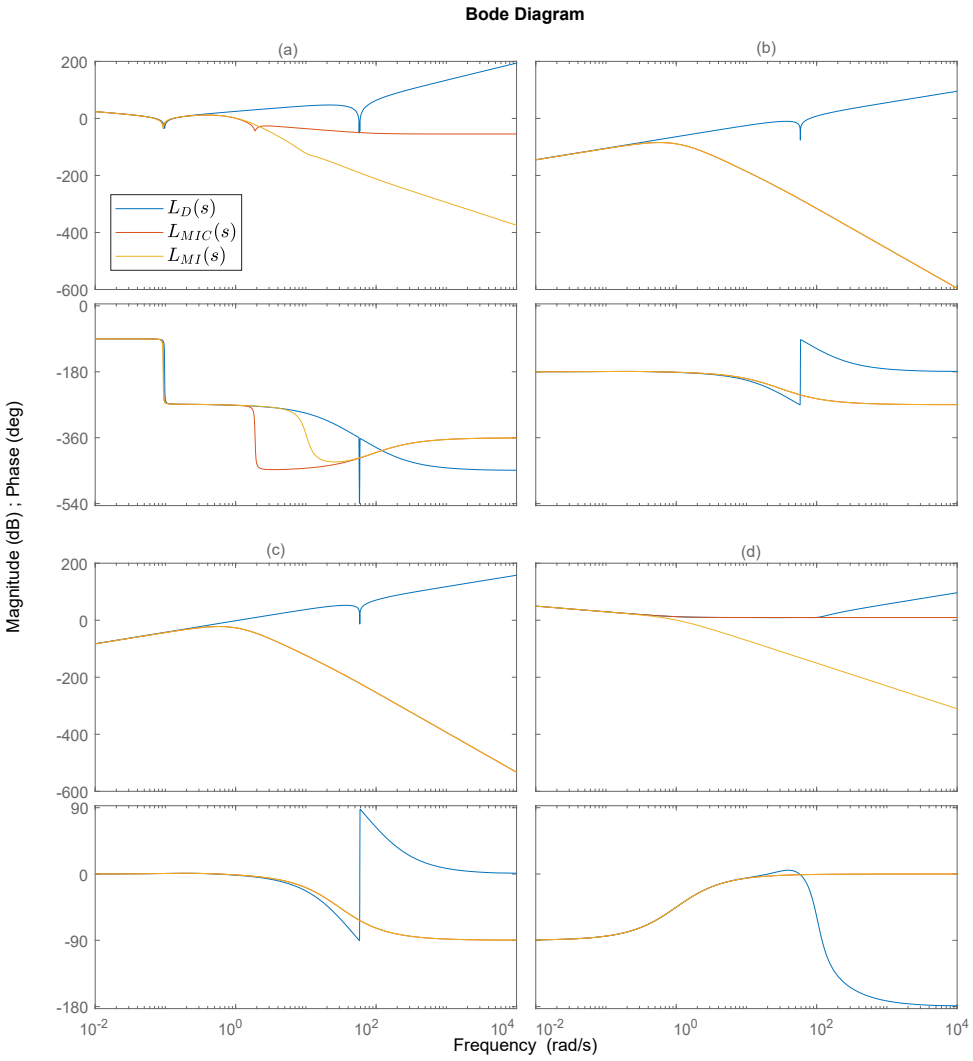


Fig. 5.3: Bode diagram of $\boldsymbol{L}_{MIC}(s)$, $\boldsymbol{L}_{MI}(s)$ and $\boldsymbol{L}(s)$.

$$|\lambda_i(\boldsymbol{I} - \boldsymbol{S}(s)\boldsymbol{G}(s)(\hat{\boldsymbol{G}}_{zpf}^{-1}(s) + \boldsymbol{C}(s)))| \leq |\boldsymbol{Q}^{-1}(s)|. \tag{5.18}$$

So far, no studies have been done regarding the graphical analysis of the convergence condition of a MIMO ILC algorithm. Therefore, in order to carry out an intuitive and a graphical approach we propose to apply the same procedure as in the SISO ILC, however, instead of looking to a Nyquist plot of the frequency response, we analyze the eigenvalues of (5.18) at all frequencies.

As it has been explained in Section 5.2, if every eigenvalue of the term $\boldsymbol{\Lambda}_2(s) = \boldsymbol{I} - \boldsymbol{S}(s)\boldsymbol{G}(s)(\hat{\boldsymbol{G}}_{zpf}^{-1}(s) + \boldsymbol{C}(s))$ is less than one in module, then AS of the MIMO ILC algorithm will be guaranteed.

Ideally, with an exact model inverse in $\boldsymbol{L}(s)$, $\boldsymbol{\Lambda}_2(s)$ would be a zero matrix with every eigenvalue at zero for all frequencies. However, due to the simplifications at high frequencies included in $\hat{\boldsymbol{G}}^{-1}_{zpf}(s)$ and the incorporation of low-pass filters, the eigenvalues will be close to the origin at low frequencies and deviate from the origin as frequency increases. In the same manner as in SISO ILC, the closer the eigenvalues to the origin, the faster those error frequencies will be corrected by the ILC algorithm.

$\boldsymbol{\Lambda}_2(s)$ is a two-by-two transfer matrix from which two eigenvalues can be obtained at each frequency point. Figure 5.4 shows each eigenvalue obtained from $\boldsymbol{\Lambda}_2(s)$ at each frequency.

At low frequencies, the eigenvalues are close to zero, as it has been pointed out in Fig. 5.4 for a frequency of $\omega = 2$ rad/s. At high frequencies the eigenvalues value increases, therefore, penalizing the convergence rate at those frequencies. Although the eigenvalues increase, none of them get out of the unit circle, which guarantees the stability of the algorithm at high frequencies.

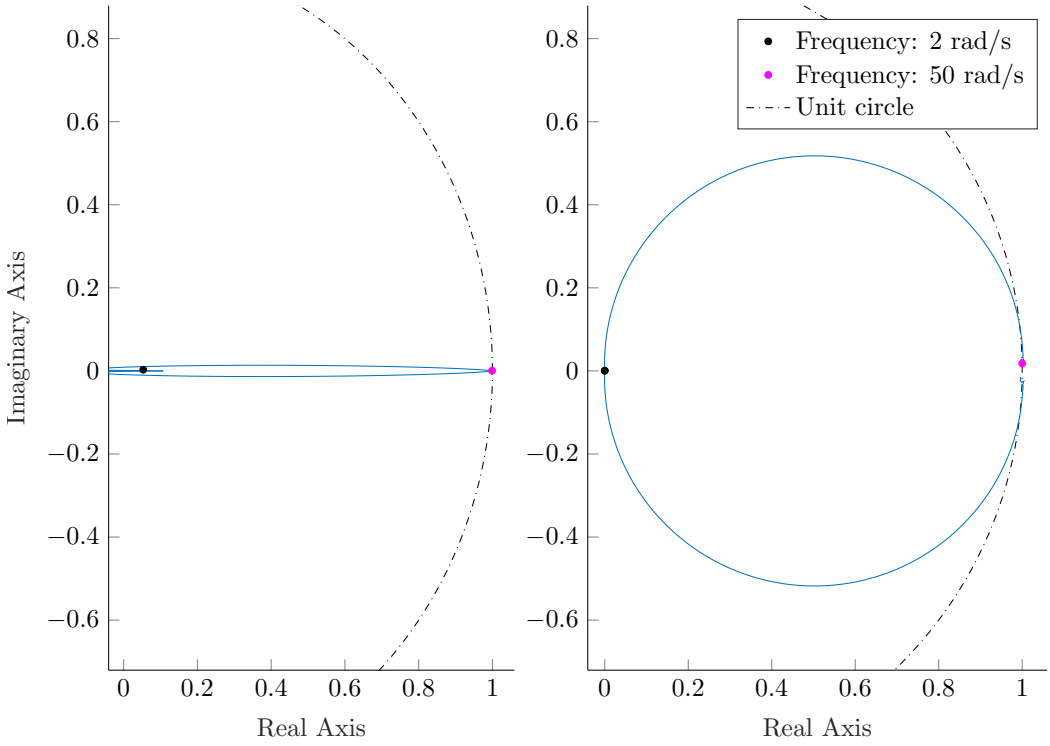


Fig. 5.4: $\boldsymbol{\Lambda}_2(s)$ eigenvalues plot at each frequency.

If the eigenvalues got out of the unit circle at a specific frequency, we could make use of $\boldsymbol{Q}(s)$ to enlarge the stability circle so the eigenvalues at all

frequencies remain inside the new stability circle. By setting a $|\boldsymbol{Q}(s)| < |\boldsymbol{I}|$ the stability circle increases accordingly, see in (5.18) that the smaller the value of $\boldsymbol{Q}(s)$ the bigger the value of the right-hand side. However, one should be cautious in the $\boldsymbol{Q}(s)$ design, as a $\boldsymbol{Q}(s)$ value different from one affects the convergence performance of the ILC algorithm, as it was shown in [40].

Some authors design $\boldsymbol{Q}(s)$ as a low-pass filter, to ensure the condition for stability is met [34, 40]. Using this $\boldsymbol{Q}(s)$ design approach, one can determine which frequencies are emphasized in the learning process. However, perfect tracking will not be achieved.

5.5 Simulation study: MIMO ILC implementation in a hydraulic press

The designed MIMO control ILC algorithm has been implemented in a non-linear model of a hydraulic press in Matlab/Simulink. In the same way as in the SISO case, the implementation has been done in Ikerlan's library, to guarantee the reproduction of industrial components with high precision. The system implementation can been seen in Fig. 5.5.

The position control MIMO ILC algorithm is compared to the performance of a PI controller, which is the most prevalent controller applied in hydraulic circuits. Two different PI controllers have been designed for the pump swash plate angle and the rod-side chamber proportional valve spool position. The proportional valve PI controller gains have been set to $K_{Pv} = 3$ and $K_{Iv} = 3$. The pump PI controller gains have been set to $K_{Pp} = 0.0569$ and $K_{Ip} = 0.3129$. Both PI controllers have been discretized by applying Tustin approximation as the hydraulic press models run in discrete time with a fixed step-size of 0.002s.

The control aim is to track the position reference during the *Free Fall* and *Drawing* phases accurately. The position control is shown in Fig. 5.6a. The PI controller tracks accurately the first part of the *Free Fall* phase, however, in the transition between phases the performance deteriorates.

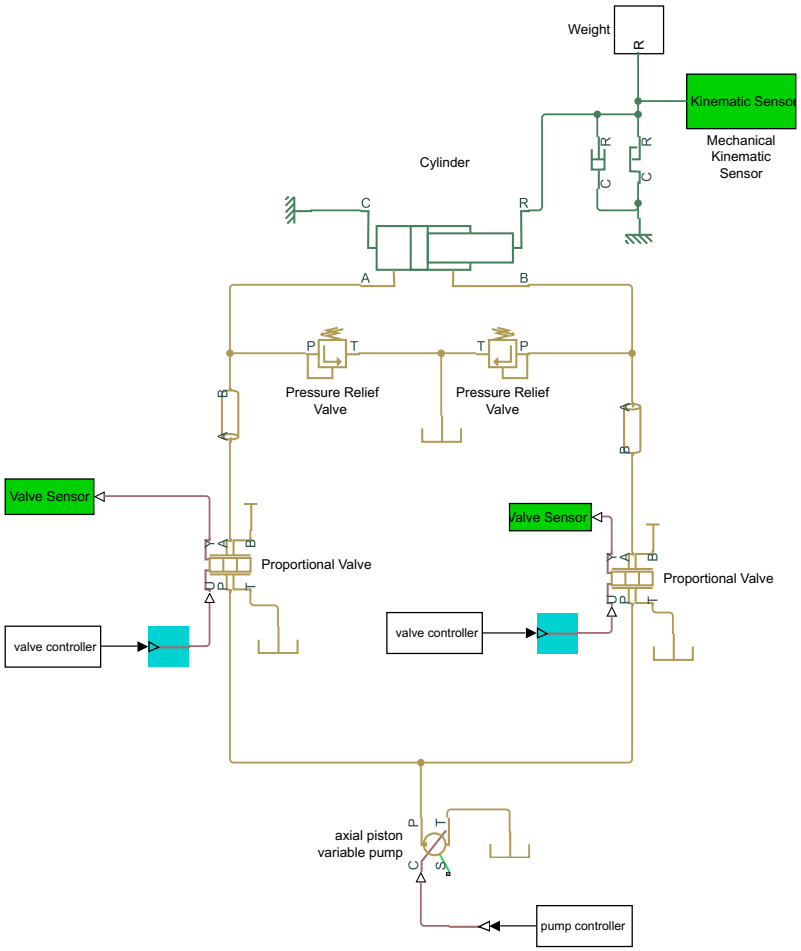
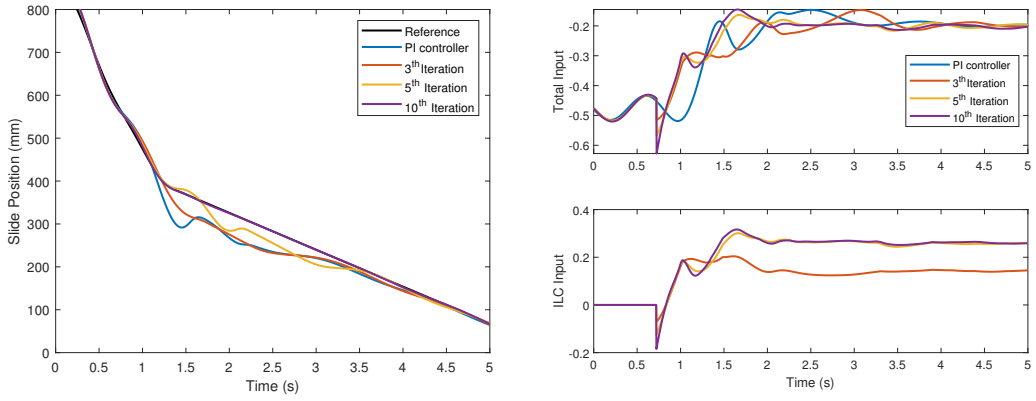


Fig. 5.5: Simscape implementation of the hydraulic press circuit.



(a) **(b)**

Fig. 5.6: Position and valve input signal during *Free Fall* and *Drawing* phases.

In Fig. 5.6a, at the first iteration with the PI controllers and no ILC input signal, there is an obvious rebounding at $t \approx 1.25$s. This rebounding results from the pump swash plate closing. At that moment, the PI controller opens the valve to speed up the slide and to track the reference trajectory. However, it is not until $t \approx 3.5$s that the reference is reached.

The ILC signal is introduced at $t \approx 0.5$s, before the transition between the two phases. With the introduction of the ILC algorithm, as iterations go on, both the valve spool position and the pump swash angle are corrected to get a precise position tracking. At iteration 10, the ILC algorithm closes the valve faster in order to prepare for the two phases transition. This can be seen in Fig. 5.6b at $t \approx 1.0$s, comparing the valve closing speed of the PI controller with the ILC algorithm.

Once in the *Drawing* phase, the ILC maintains the spool position around -0.2 value to reach a constant falling velocity. This effect can be seen in the slide velocity progress through iterations in Fig. 5.7. At the first iteration, there is a significant overshoot in the velocity signal which is attenuated as iterations go on. Introducing the ILC signal, once the slide reaches -100 mm/s speed, it is maintained at this speed.

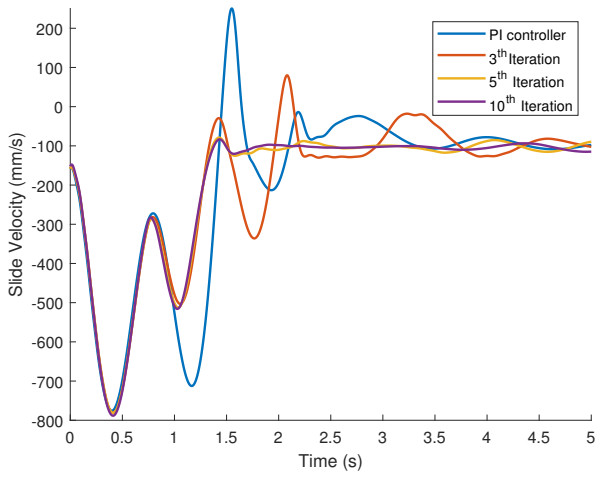
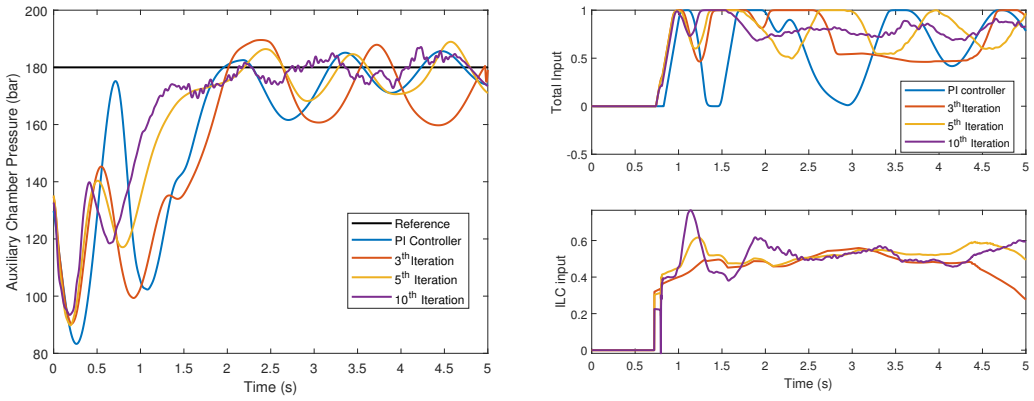


Fig. 5.7: Slide velocity during *Free Fall* and *Drawing* phases.

The pressure control for a 180 bar reference is shown in Fig. 5.8a. During the *Free Fall* phase, no pressure control is carried out as the rod-side chamber pressure is given by the slide weight. During the *Drawing* phase, from $t \approx 1.5$s on, with the PI controller the pressure tracking oscillates considerably around the reference. This results from the swash plate angle, see Fig. 5.8b, that saturates in one and then oscillates harshly.



(a) (b)

Fig. 5.8: Pressure tracking for a 180 bar reference and pump swash angle during *Free Fall* and *Drawing* phases.

Introducing the ILC algorithm, the swash plate oscillations are reduced until, at iteration 10, a constant swash plate is achieved to keep the pressure around the reference. With this improvement, the pressure reference signal is reached faster and the oscillations are considerably reduced.

It should be pointed out that perfect pressure tracking is not needed, it is sufficient to keep the auxiliary chamber pressure close to the reference so, when the slide strikes the workpiece during the *Drawing* phase and extra force is needed, the remaining force can supplied faster.

As a performance index for the proposed MIMO ILC algorithm, the RMSE between the position reference and the slide position signal is shown in Fig. 5.9. The RMSE is reduced by a factor of 6 in the position tracking, and a fast convergence rate is obtained as at the seventh iteration the error is considerably reduced, with respect to the first iteration with the two PI controllers.

The RMSE between the 180 bar pressure reference and the rod-side chamber pressure signal is shown in Fig. 5.10. Note that only the RMSE for the *Drawing* phase has been obtained, the *Free Fall* phase has not been considered. The error is reduced by a factor of seven regarding the first iteration with the two PI controllers.

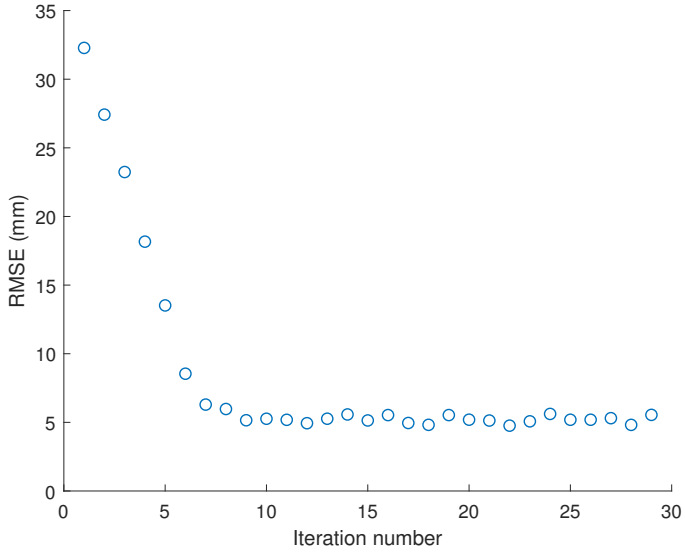


Fig. 5.9: RMSE between the position reference and slide position over iterations.

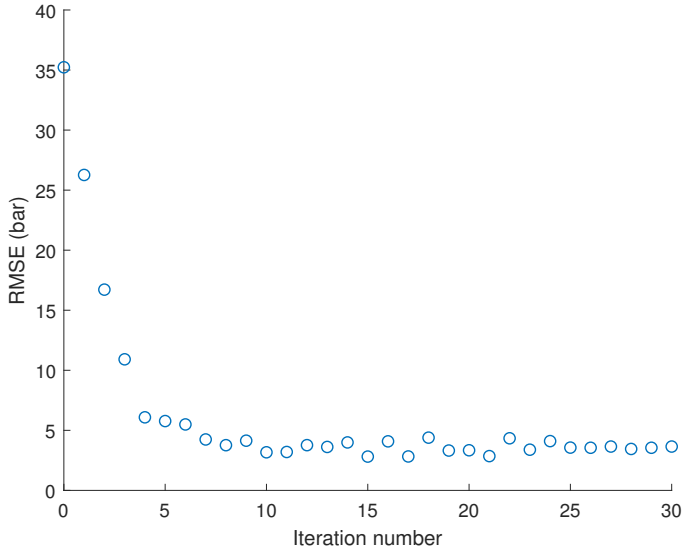


Fig. 5.10: RMSE between 180 bar pressure reference and rod-side chamber pressure over iterations.

5.6 MIMO ILC performance comparison

To validate the proposed MIMO ILC design, a performance comparison is carried out with a P-ILC and MI-ILC, as already done in Section 3.6 for the SISO case. We extend the P-ILC and MI-ILC to the MIMO case and the RMSE

between the position reference and the slide position will be used as the performance index.

5.6.1 MIMO P-ILC algorithm

The P-ILC algorithm proposed by Arimoto in [30] is extended in this section to the MIMO case. The MIMO P-ILC learning law is as follows:

$$\boldsymbol{U}_{j+1}(s) = \boldsymbol{U}_j(s) + \boldsymbol{K}_P \boldsymbol{E}_j(s), \tag{5.19}$$

where $\boldsymbol{K}_P$ is the proportional learning matrix, which is used to maximize the convergence rate of the algorithm. The following $\boldsymbol{K}_P$ design is proposed:

$$\boldsymbol{K}_P = \begin{bmatrix} 0.001 & 0.2 \\ 0.0001 & 2.5 \end{bmatrix}. \tag{5.20}$$

$\boldsymbol{K}_P$ is chosen to optimize the convergence rate of the P-ILC algorithm, so the frequency matrix eigenvalues lie as close as possible to the origin.

At low frequencies the eigenvalues lie far from the origin, but by means of $\boldsymbol{K}_P$ we can move closer the high frequencies to the origin, as it is pointed out for 50 rad/s in Fig. 5.11. However, there exists a stability limit for increasing the values in (5.20) as, by moving closer to the origin the high frequencies, the low frequencies get out of the stability circle, see Fig. 5.11. With the MIC-ILC, at zero frequency, the eigenvalues lie at the origin. This is because of the cancellation of the model dynamics at low frequencies, that optimizes the convergence rate of the ILC algorithm.

The RMSE between the position reference and the slide position with the P-ILC and the MIC-ILC algorithm is shown in Fig. 5.12. The P-ILC converges at the first 20 iterations, however, as iterations go on, the RMSE increases. This is due to the stability design in Fig. 5.11, that at low frequencies the eigenvalues of the P-ILC algorithm get out of the unit circle. The MIC-ILC not only achieves faster convergence rate, but also remains stable.

We know, from Section 5.4, that we can make use of the $\boldsymbol{Q}$ matrix to enlarge the stability circle so the algorithm remains stable for all frequencies. With $\boldsymbol{Q} = [0.93 \quad 0; 0 \quad 0.93]$, the resulting eigenvalues response with the new stability circle is shown in Fig. 5.13.

By decreasing $\boldsymbol{Q}$ the stability circle has been enlarged and all the eigenvalues lie inside it. However, as shown in [40], decreasing $\boldsymbol{Q}$ the convergence of

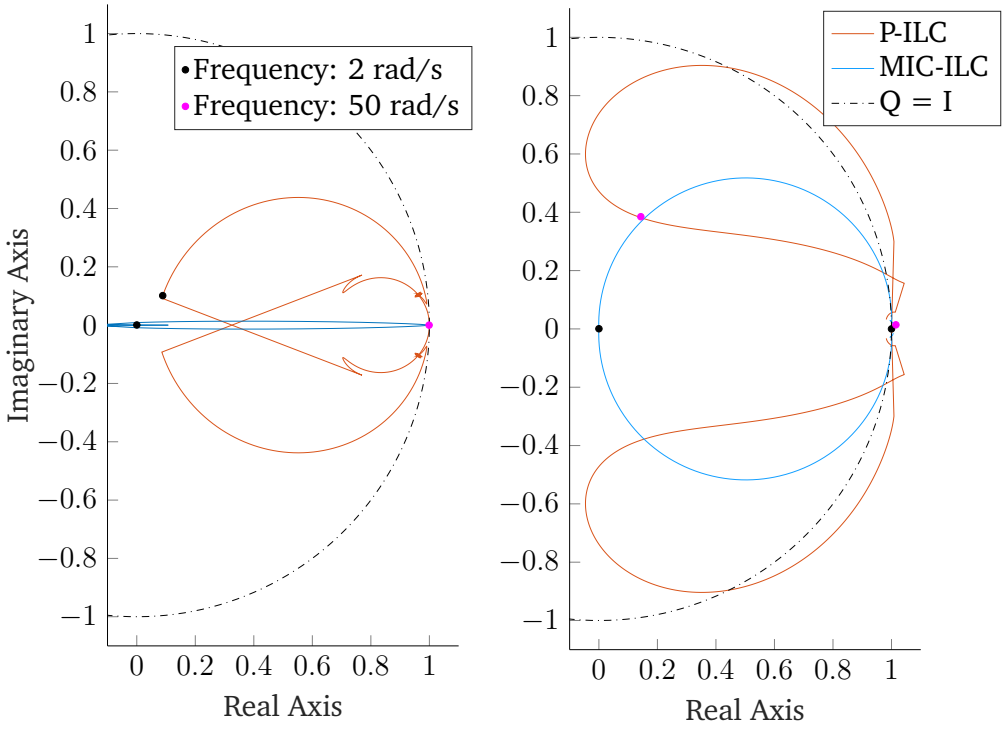


Fig. 5.11: Eigenvalues plot at each frequency for the P-ILC and MIC-ILC algorithms.

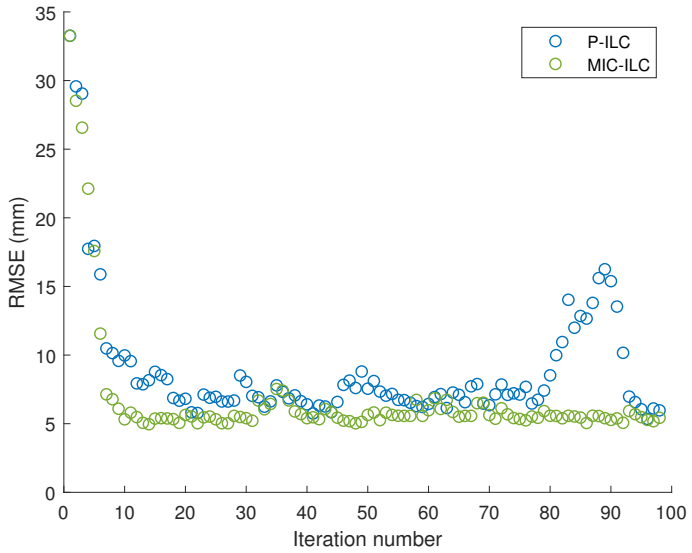


Fig. 5.12: Position control RMSE comparison with P-ILC and the MIC-ILC algorithm.

the algorithm is penalized and the error converges to a larger steady state

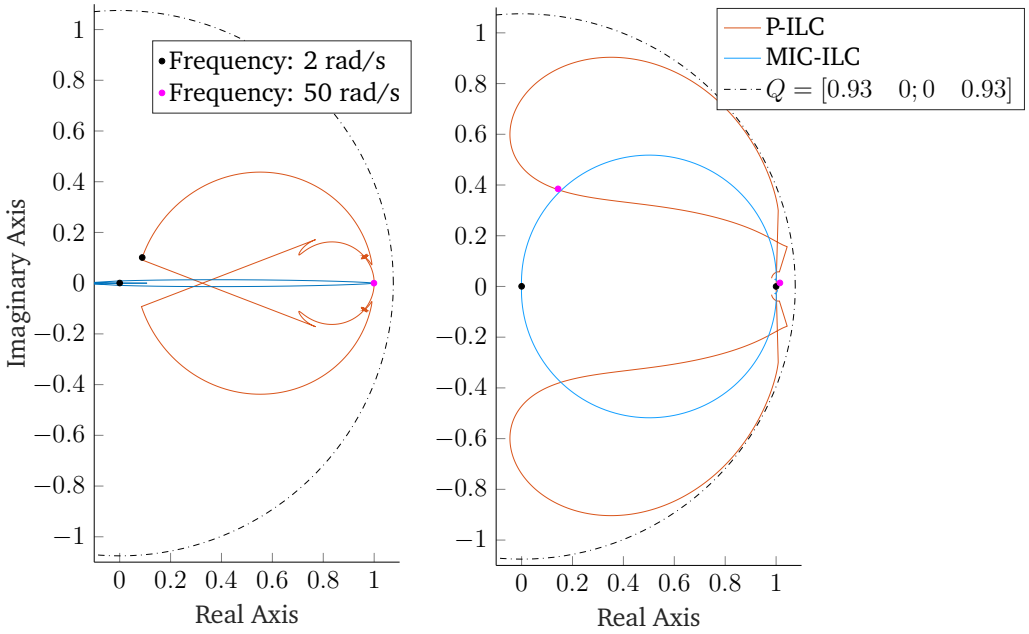


Fig. 5.13: Eigenvalues plot at each frequency for the P-ILC and MIC-ILC algorithms.

error. This can be seen in the RMSE analysis over iterations with the new value for $\boldsymbol{Q}$, in Fig. 5.14.

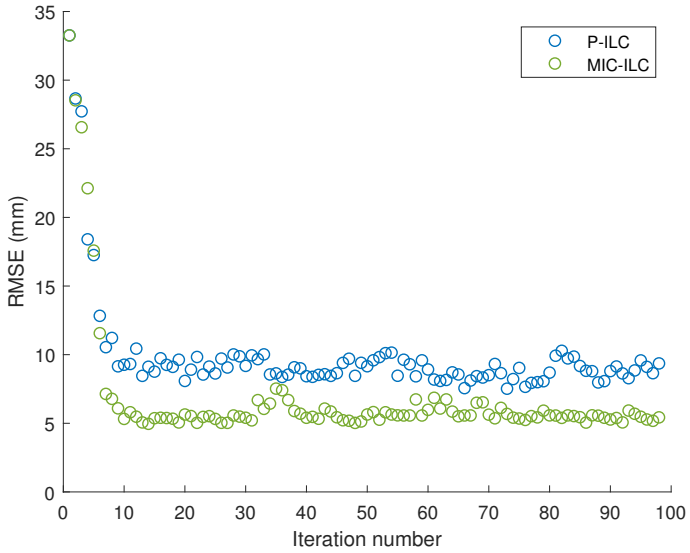


Fig. 5.14: Position control RMSE comparison with P-ILC and the MIC-ILC algorithm for $\boldsymbol{Q} = [0.93 \quad 0;0 \quad 0.93]$.

With the MIC-ILC a $\boldsymbol{Q} = \boldsymbol{I}$ is used, and the convergence is not penalized, the RMSE reaches a 5 mm value. However, with the P-ILC algorithm, the value towards the RMSE converges is 10 mm, double the value of the MIC-ILC.

5.6.2 Conventional MIMO model inverse ILC

The conventional model-inverse approach used in the literature will be designed in this section. It follows the same design as the one followed in Section 3.6, but extended to the MIMO application.

The model inverse learning filter design is as follows:

$$\boldsymbol{L}_{MI}(s) = \hat{\boldsymbol{G}}^{-1}(s) \underbrace{(I + \hat{\boldsymbol{G}}(s)\boldsymbol{C}(s))}_{\boldsymbol{S}^{-1}(s)}. \tag{5.21}$$

$\hat{\boldsymbol{G}}$ is the linearized hydraulic slide circuit derived in Section 5.2. In the same way as for the MIC-ILC algorithm, we need to include low-pass filters to attenuate the unknown high frequencies not included in model inverse modeling. A fourth-order low-pass filter is added to $L_{MI}(s)$ and ZPF is carried out to avoid the phase loss. Following the conventional model-inverse ILC design (MI-ILC), the filter is added to the entire $\boldsymbol{L}_{MI}(s)$ transfer matrix as following:

$$\boldsymbol{L}_{MI}(s) = \hat{\boldsymbol{G}}^{-1}(s)(I + \hat{\boldsymbol{G}}(s)\boldsymbol{C}(s))\frac{\omega_c^2}{(s+\omega_c)^2}\frac{\omega_c^2}{(s-\omega_c)^2}. \tag{5.22}$$

The placement of the low-pass filter varies with respect to the proposed MIC-ILC, where the low-pass filter is only applied to $\hat{\boldsymbol{G}}^{-1}(s)$, see (5.16). This variation directly affects the convergence rate of the MI-ILC algorithm, as the model differences are bigger and the eigenvalues lie further from the origin, see Fig. 5.15.

In Fig. 5.15, although in the MI-ILC algorithm at zero frequency the eigenvalues lie at the origin, they rapidly diverge from it. Comparing both algorithms response, in the MIC-ILC the eigenvalues deviate from the origin at a slower pace than in the MI-ILC. This can be seen comparing the distance from the origin of the eigenvalues at 2 rad/s frequency point.

In Fig. 5.16 a faster convergence rate is obtained with the MIC-ILC compared to the MI-ILC. The former converges in 15 iterations and the latter converges rapidly at the first 10 iterations. As iterations go on, the MI-ILC

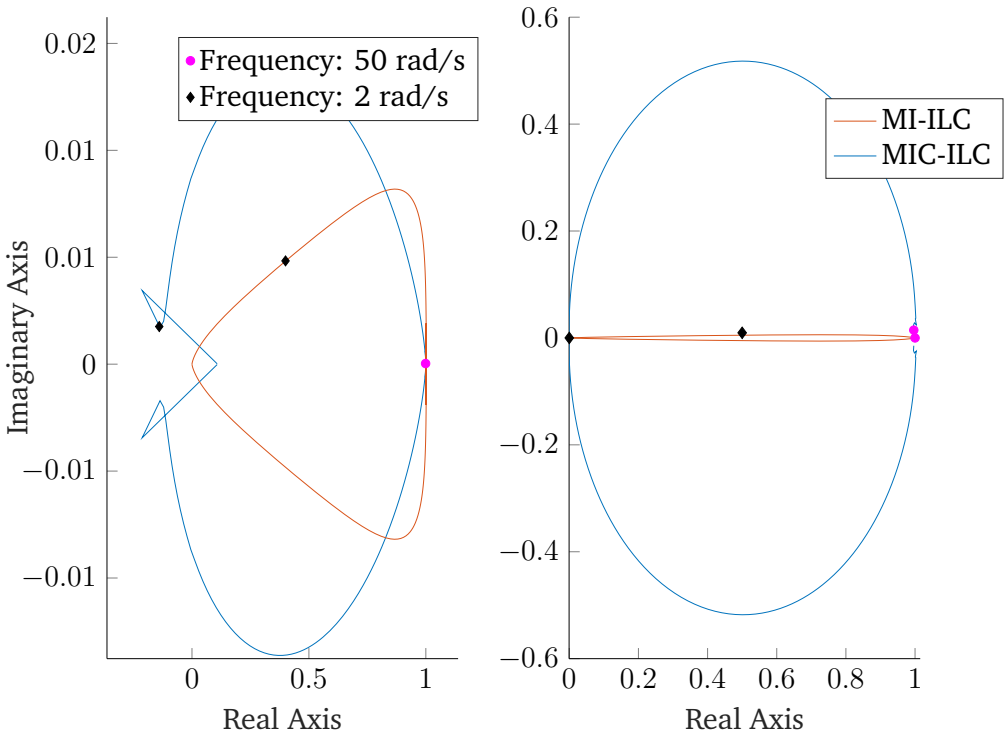


Fig. 5.15: Eigenvalues plot at each frequency for the MI-ILC and MIC-ILC algorithms.

gradually minimizes the RMSE at each iteration. This is a consequence of the eigenvalues' distance from the origin as the frequency increases, as the low frequencies are corrected faster than the high frequencies. With the MIC-ILC, the eigenvalues are closer to the origin, so the low and high frequency components of the error are corrected faster.

Apart from the RMSE as a performance indicator, the Power Spectral Density (PSD) also provides useful insights for the comparison between ILC algorithms. In Fig. 5.17, the PSD of the error signal over iterations is shown. It can be seen that the energy is concentrated on the frequency band $[0 \sim +30]$ Hz.

At the first 10 iterations, the MIC-ILC reduces faster the low frequency components of the error. Specially, the frequency band from 0 Hz to 5 Hz is decreased considerably, as those frequencies are where the eigenvalues lie close to the origin. At high frequencies, the error reduction is slower, however, the MIC-ILC still achieves faster correction rate. In line with the results obtained with the RMSE, at iteration 20, the three algorithms

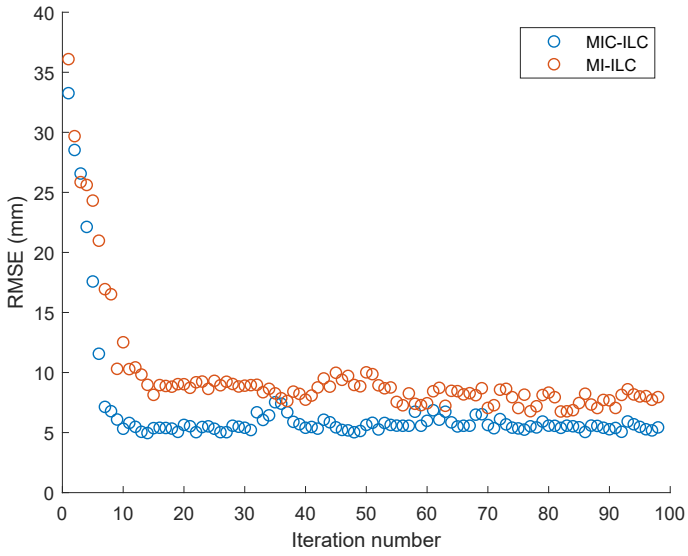


Fig. 5.16: RMSE comparison with MI-ILC and the MIC-ILC algorithm under noisy conditions.

learning process has already converged and the difference between iterations is minimal, the frequency component of the error remains similar.

5.7 Conclusions to Chapter 5

In order to accomplish the MIMO position control proposed in Chapter 4, we propose a MIMO ILC algorithm, to take advantage of the repetitive behavior of a hydraulic slide and overcome the coupling of the pump and valve control loops.

The proposed MIMO ILC follows the same design procedure as the one shown for the SISO ILC in Chapter 3. However, it has been extended to the MIMO case, in which a simplified multivariable model inverse of the hydraulic press, and the two existing feedback controllers have been considered.

For the MIMO ILC stability and convergence analysis, no graphical evaluation has been found in the literature, in order to intuitively design the learning matrix and seek for the algorithm stability. Therefore, we propose a graphical approach based on the eigenvalues of the frequency response matrix. In this way, the same procedure as in the SISO ILC can be followed, however, instead of looking at the frequency response we look at the eigenvalues.

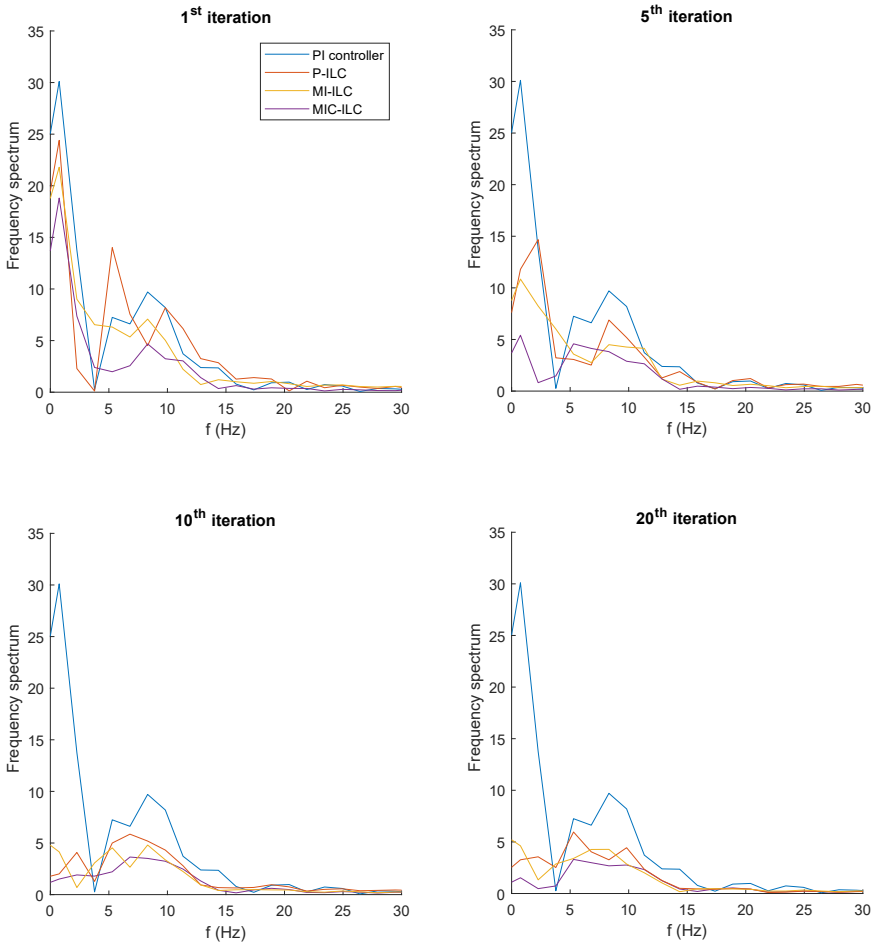


Fig. 5.17: Power spectrum of the pressure error signal, for the four different controllers.

With this method, the stability of the algorithm is fulfilled if the learning filter eigenvalues lie inside the unit circle. The closer the eigenvalues lie to the origin, the faster the MIMO ILC algorithm will converge.

The proposed MIC-ILC algorithm has been implemented in a nonlinear hydraulic press model in Matlab/Simulink, for which satisfactory results have been obtained. Furthermore, the MIC-ILC has been compared to other ILC designs and a performance analysis has been carried out.

Following the proposed graphical design approach, the MIC-ILC eigenvalues lie closer to the origin than the other ILC algorithms, which yields faster convergence rate, as well as stability.

This is one of the main contributions of the MIC-ILC, the ability of correcting the low frequency components faster than the existing ILC algorithms in the literature. The enhancement of the MIC-ILC has been verified in the frequency domain, with the Power Spectral Density, analyzing the error signal at each iteration for each ILC algorithm. The evolution of the error signal in the frequency domain also shows that the MIC-ILC outperforms other ILC methods in terms of stability and convergence rate.

The next section discusses the implementation of the ILC algorithms in two industrial scenarios: a hydraulic press test rig and a Digital Twin platform with real industrial hydraulic press controllers.

6 Experimental Results

6.1 Introduction

This thesis has analyzed two control problems, the force control of mechanical presses and the position control of hydraulic presses. Two novel schemes have been proposed, based on ILC, and their superiority over the available control schemes has been shown in a simulation environment.

At this point is where we ask: how ready are the proposed algorithms to be implemented in a real hydraulic or mechanical press? This can be answered by looking at the Technology Readiness Level (TRL) scale that was originated at NASA, to measure how far a technology is from being deployed in space [88]. This definition was afterwards adopted by the European Commission, and was extended to any technology. The summary of the 9 stages that form the TRL scale made by the European Commission were given in [89] and are shown in Table 6.1.

Tab. 6.1: Horizon 2020 Technological Readiness Levels [89].

TRL 9	Actual system proven in operational environment
TRL 8	System complete and qualified
TRL 7	System prototype demonstration in operational environment
TRL 6	Technology demonstrated in relevant environment
TRL 5	Technology validated in relevant environment
TRL 4	Technology validated in lab
TRL 3	Experimental proof of concept
TRL 2	Technology concept formulated
TRL 1	Basic principles observed

The TRL scale can be separated in four levels: TRL 1-3 is where the fundamental research takes place, the proof concept. TRL 4-5 is where the validation is done in lab and the results presented should be statistically relevant. TRL 6-7 is the demonstration that the concept actually works in a industrially relevant environment. Finally, TRL 8-9 is where the technology is put into production.

We can state that, up to this section, the proposed position and force controllers lie under TRL-5, as the controllers have been implemented in high-fidelity press simulations in MATLAB/SIMULINK. According to [89], high-fidelity addresses the concept of a laboratory environment that can simulate and validate all system specifications within a laboratory setting. This brings us to the novel hydraulic library made by Ikerlan, used to model the hydraulic and mechanic press models, which validation under laboratory conditions and reproduction of physical behaviors was proven in [66].

Through Chapters 3 and 5, we have validated under simulation conditions both force and position controllers. Now, once TRL-5 is complete, we may advance to TRL-6 and TRL-7. To that end, two experimental scenarios are presented. On the one hand, the force and position control are implemented in a hydraulic test rig, that allows experimental testing of the controllers under actual conditions. On the other hand, in a Digital Twin platform, the force controller will be implemented in a Simotion D445-2 industrial press controller. This enables Real-Time (RT) representation of the mechanical press operation under industrial control platforms.

With the implementation in the hydraulic test rig, we will jump into TRL-6. The hydraulic test rig can be regarded as a representative prototype of a hydraulic press, as it consists of every hydraulic element that we can find in a hydraulic press. Therefore, as the controllers will be tested in a relevant environment, it represents a major step up in a technology's demonstrated readiness.

With the implementation in the Digital Twin environment, we test the proposed force control algorithm in an operational environment, i.e. the Simotion controllers that are used in real industrial presses. In the Digital Twin platform, the press models have been embedded into a control system that supports RT communications. Therefore, this is a step up with respect to TRL-6, as the press model prototypes will be implemented in a real operational environment, with real industrial controllers, and not in a custom made controller.

6.2 Hydraulic test rig setup

The proposed force and position ILC control algorithms are implemented in the hydraulic test rig shown in Fig 6.1. It consists of two identical hydraulic circuits and each circuit comprises: a double-acting cylinder, a proportional valve, a hydraulic pump, a pressure relief valve, an accumulator, pressure

sensors at each cylinder chamber and a flow rate sensor. The hydraulic scheme of the test rig is shown in Appendix C.0.4 and the parameter values are depicted in Appendix C.0.3.

Fig. 6.1: The available hydraulic test rig at Ikerlan Technology Research Center.

Figure 6.2 shows the detailed view corresponding to a hydraulic circuit, note that the other hydraulic circuit is identical. The proportional valve is a Moog D633 4-way valve, which spool position is controlled electronically. At ports A and B the valve has two stopcock valves which, if closed, limit the flow rate to the cylinder. Two pressure sensors have been installed at ports A and B, so the pressure at each cylinder chamber can be measured.

The gas accumulator has been pre-charged with nitrogen to 100 bar. It has a stopcock valve so, if desired, it can be isolated from the circuit. For the experimental tests in this document, the accumulator will be isolated from the hydraulic circuit.

Several needle valves have been installed in the circuit to reproduce flow losses in the circuit, e.g. cylinder chamber losses, valve losses, pump losses. A flow rate sensor has been installed between the hydraulic pump and the proportional valve, which corresponds to element 12.2 in Appendix C.0.4. The maximum working pressure of the flow rate sensor is 160 bar, therefore, the pressure relief valve cracking pressure has been set to 160 bar for security reasons. A contact-less magnetostrictive linear position transducer has been included to measure the cylinder piston position.

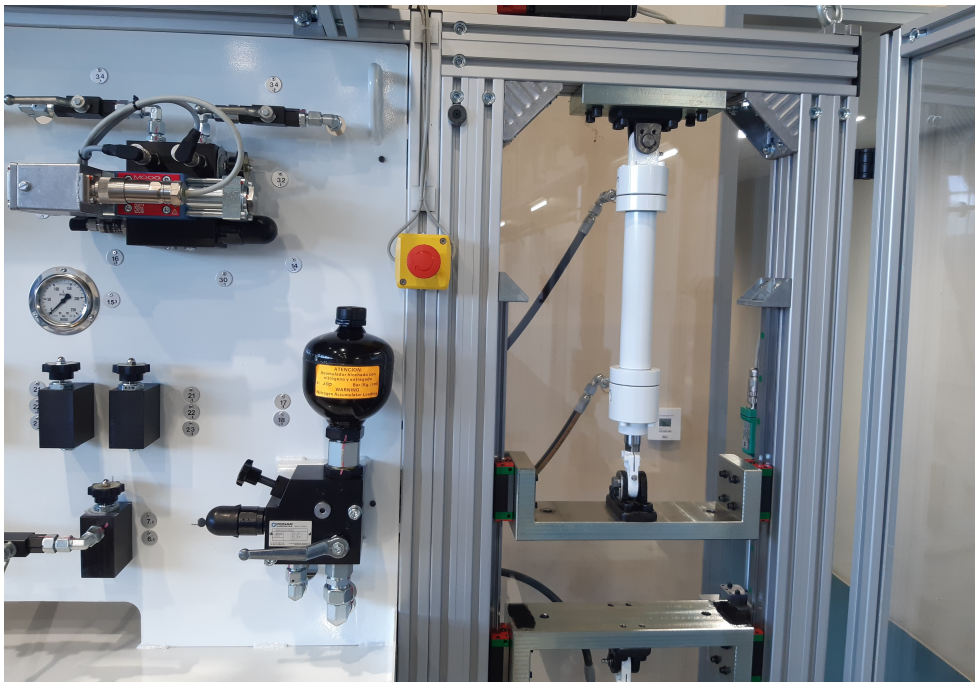
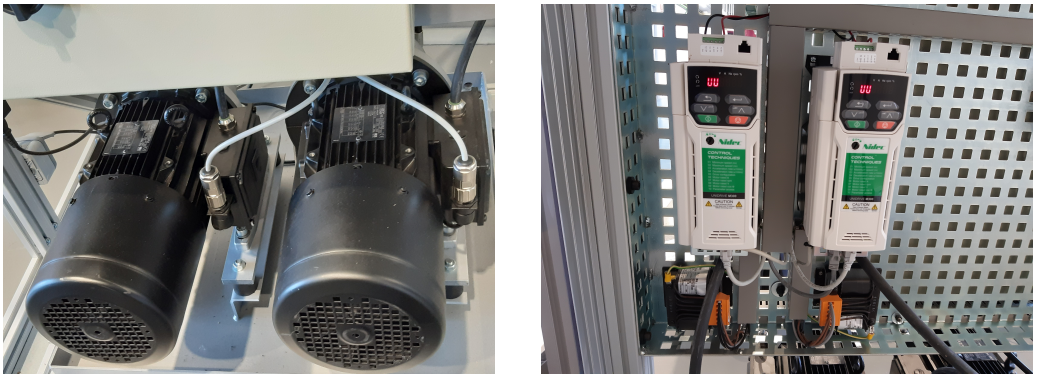


Fig. 6.2: Detailed view of one of the hydraulic circuits of the hydraulic test rig.

The detailed view of both hydraulic pumps and both Unidrive M300 is shown in Fig. 6.3. A helical gear rotor pump has been installed, which is ideal for

high pressures and low noise. The pump flow rate varies depending on the rotational speed of the pump.



(a) (b)

Fig. 6.3: Hydraulic pumps and Unidrive M300.

The hydraulic pump is driven by an induction motor. The motor is connected to an Unidrive M300, which is a variable speed AC drive for induction motors. The Unidrive M300 has an onboard PLC and SI-interface that allows the communication with the driver via EtherCAT protocols.

Each hydraulic circuit has two actuators and four sensors. The actuators are the proportional valve and the hydraulic pump. The sensors are: a pressure sensor for each cylinder chamber, a transducer sensor for the cylinder piston position and a flow rate sensor.

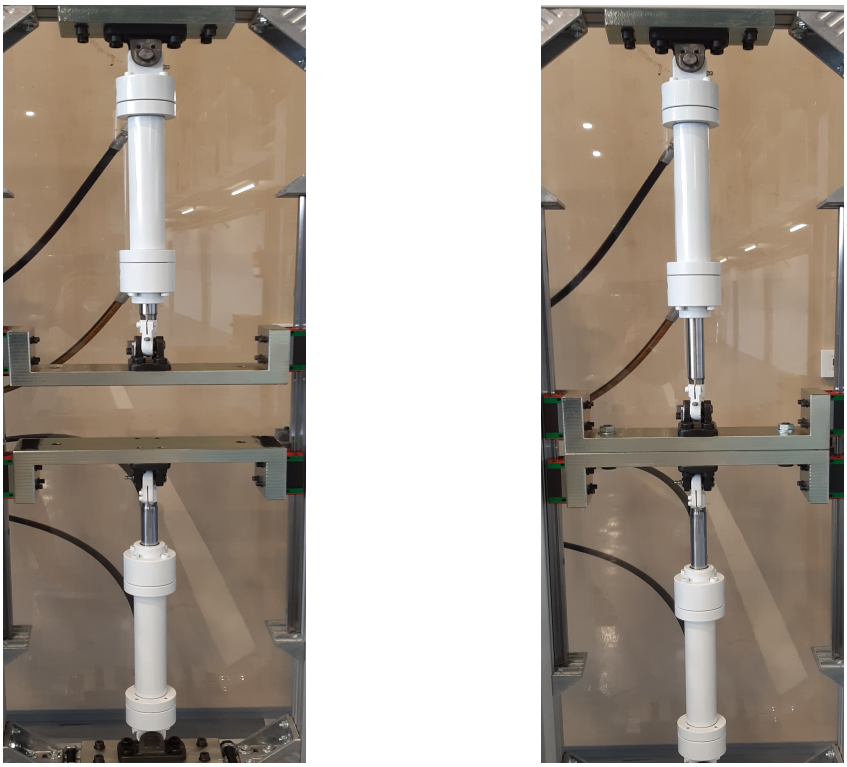
Every system actuator and sensor is connected via EtherCAT communication to a Beckhoff 6930-0050. The Beckhoff Industrial PC (IPC) receives and processes the sensor signals, and delivers back the corresponding control inputs to the actuators.

The position and control ILC algorithms are implemented in TwinCAT 3, which is a platform developed by Beckhoff, where PLC programming is carried out. The integration into Beckhoff's industrial environment allows RT communications through EtherCAT deterministic protocol.

To validate the proposed force and position control ILC algorithms, two different cylinder configurations are used. On the one hand, with the two cylinders separated, we can reproduce the working operation of a mechanical press, see Fig. 6.4a. The upper-cylinder acts as the press slide, with the up and down motion, whereas the lower-cylinder acts as the press cushion, which waits for the slide stroke.

On the other hand, connecting both cylinders by their piston rods by means of two bolts, we can reproduce the hydraulic press working operation, see

Fig. 6.4b. With this arrangement, both cylinders move together, as they are coupled, which allows us to control the position trajectory of the joint.



(a) (b)

Fig. 6.4: Cylinder decoupled and coupled in hydraulic test rig.

6.3 Force Control in hydraulic test rig

Here, the force control tests will be carried out, with the setup shown in Fig. 6.4a. The upper-cylinder acts as the press slide, for which a position trajectory is defined. The lower-cylinder acts as the press cushion, and waits in a stand-still position for the upper-cylinder's stroke. When the stroke takes place, the pressure in the piston-side chamber of the lower-cylinder must be controlled.

A 750 rpm rotational velocity is set to the upper-cylinder hydraulic pump, to obtain a constant falling velocity during the *Drawing* phase. The lower-cylinder hydraulic pump is only used to return the lower-cylinder to the TDC position once the *Drawing* phase is finished.

The trajectory that both cylinders follow is shown in Fig 6.5. The upper-cylinder starts at the TDC position at $x \approx 18$ cm and the stroke takes place at $x \approx 6$ cm, where the *Drawing* phase starts. After the *Drawing* phase, both cylinders are moved back to their respective TDC positions. Note that the lower-cylinder moves upward at a slower pace, as the piston-side chamber needs to be pressurized in advance, to move up the piston. In the upper-cylinder, however, the rod-side chamber needs to be pressurized, this process is faster as the rod-side area is smaller.

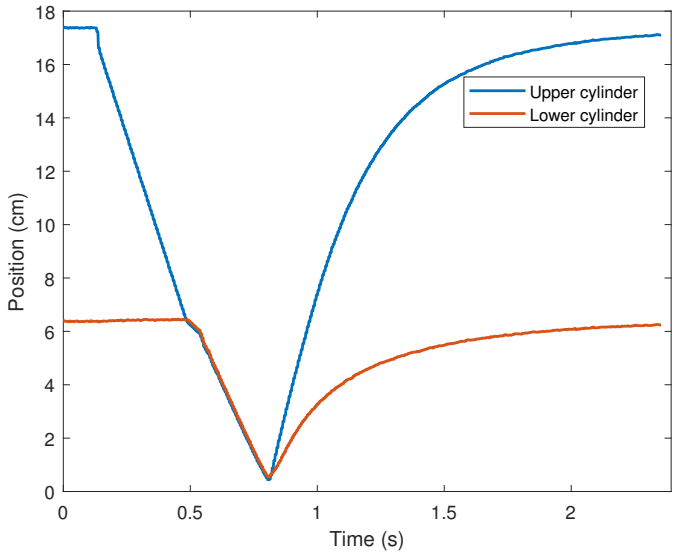


Fig. 6.5: Cylinder position trajectories during hydraulic rig test.

First, for the force control, a PI controller is designed to control the pressure during the stroke. The PI controller gain values are: $K_{Pc} = 25$ and $K_{Ic} = 5$. For a pressure reference of 110 bar, the PI controller gives the force reference tracking shown in Fig. 6.6.

From Fig. 6.6, it can be seen that there is a considerable overshoot in the pressure signal, that reaches 128 bar. In order to reduce that overshoot, we introduce the SISO MIC-ILC algorithm designed through Chapter 3, in the hydraulic rig.

The plant inverse design for the learning filter is as follows:

$$L(s) = \underbrace{\frac{s - A_{11}}{A_{12}} \frac{\omega_n^2}{(s+\omega_n)^2} \frac{\omega_n^2}{(s-\omega_n)^2}}_{\hat{G}_{zpf}^{-1}} + C(s). \tag{6.1}$$

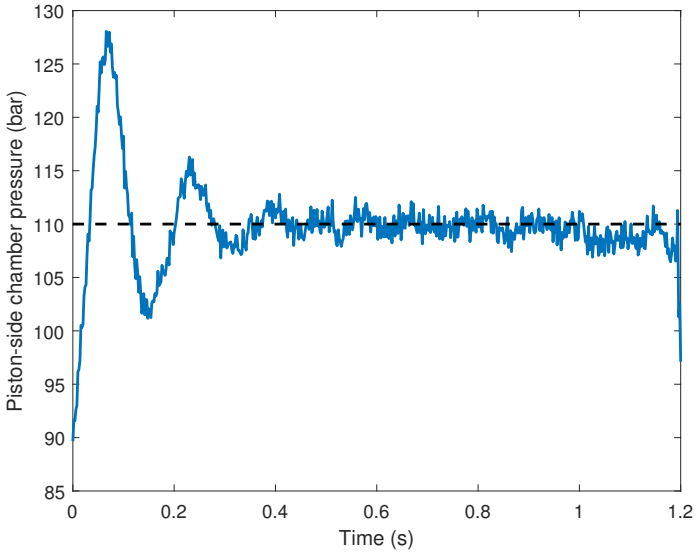


Fig. 6.6: 110 bar pressure reference tracking with PI controller.

Now, A_{11} and A_{12} are based on the hydraulic rig system parameters, which are shown in Appendix C.0.3. The controller, $C(s)$, consists of the PI controller shown above.

The frequency response of the system is shown in Fig. 6.7. A $Q = 0.92$ has been set, so the response of all the frequencies remains inside the stability circle, guaranteeing the algorithm stability.

As it has been pointed out in Fig. 6.7, at low frequencies the response of the ILC algorithm remains close to the origin, which ensures high convergence rate. As frequency increases, the response deviates at a slow pace from the origin towards the point (+1,0).

In the following sections, several tests are carried out, to analyze the SISO MIC-ILC performance towards different scenarios that can arise during the working operation of an actual hydraulic cushion circuit.

6.3.1 Robustness towards modeling mismatch

In the A_{11} and A_{12} terms used for the $L(s)$ design. In (6.1) β appears, which provides information about how much pressure must be applied to the oil to compress it. The definition of the terms can be seen in (6.2):

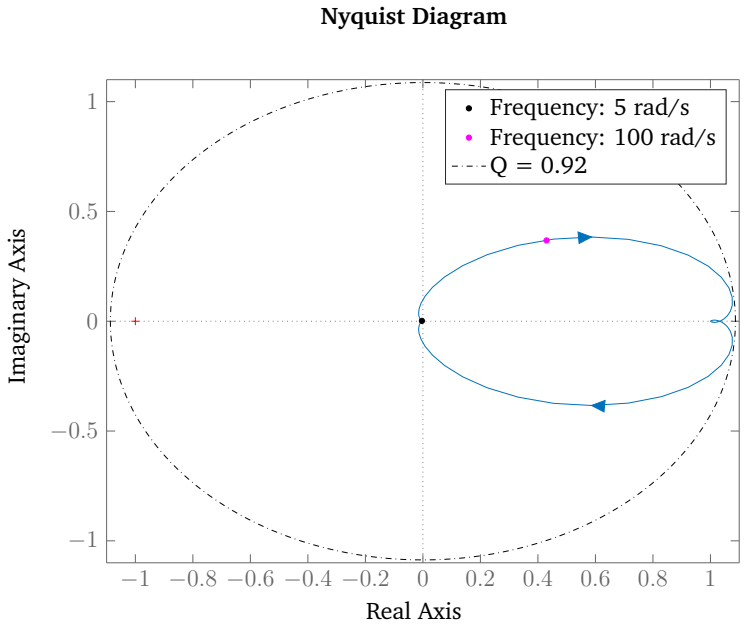


Fig. 6.7: $|1 - G(s)S(s)L(s)|$ frequency response with $L(s) = \hat{G}_{zpf}^{-1}(s) + C(s)$ for hydraulic rig.

$$
\begin{aligned}
A_{11} &= -\frac{K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})\sqrt{\Delta P_{ref}}}{V_T \beta q_{ref}} \frac{1}{2\sqrt{\bar{P}_A}} \\
A_{12} &= -\frac{\sqrt{\bar{P}_A} q_{ref}}{V_T \beta \sqrt{\Delta P_{ref}}} \dot{K}_{\mathrm{v}}(\bar{y}_{\mathrm{v}}).
\end{aligned}
\tag{6.2}
$$

As it has been shown in Appendix D, the value of the term $V_T\beta$ in the test rig is 22% smaller than the theoretical value. This means that in the hydraulic test rig, the volume change of the hydraulic oil as a response to a pressure change will be less.

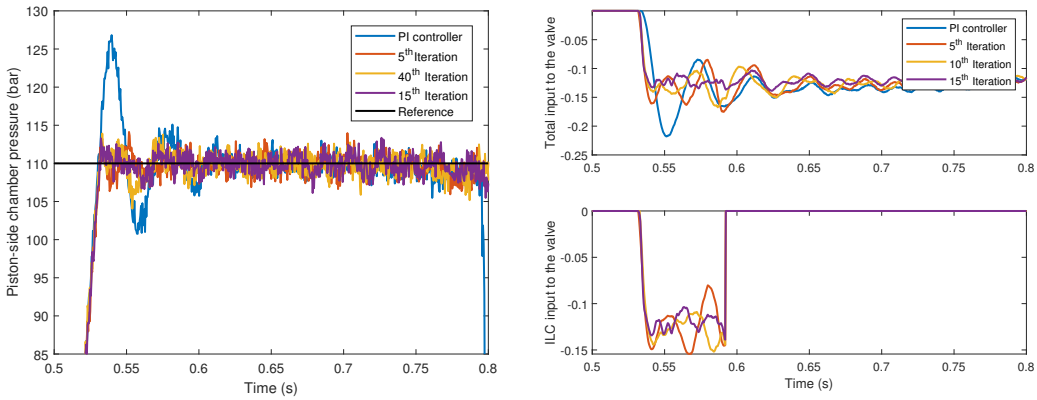
We have seen in Section 2.3, that in the FF controller any model mismatch affected considerably its performance, yielding a poor force control. In order to analyze the SISO MIC-ILC algorithm robustness towards model uncertainties, in the $L(s)$ design in (6.1), we use the theoretical value of $V_T\beta$ used through Chapter 4, instead of using the actual value corresponding to the hydraulic rig. In this way, we will see how it affects the convergence and stability of the algorithm.

The performance of the designed SISO MIC-ILC algorithm is demonstrated under a 110 bar pressure reference scenario. The pressure signal tracking for

a 110 bar step reference is shown in Fig. 6.8a. The ILC signal is activated for a limited range, from $t \approx 0.53$s to $t \approx 0.6$s, just where the overshoot signal exists. From $t \approx 0.6$s on, the PI controller maintains the pressure around the reference.

It should be pointed out that once the ILC is deactivated, the last input from the ILC is assigned to the PI controller. Else, a pressure bump would appear due to a large change in the PI input. To avoid it, the last ILC input signal is assigned to the PI controller integral action, to achieve a bumpless switching of controllers.

Instead of opening the valve spool position further to eliminate the pressure overshoot, the MIC-ILC opens the valve faster as it can be seen in Fig. 6.8b. With the PI controller, the maximum spool position is obtained at $t \approx 0.55$s, however, with the MIC-ILC the maximum spool position is at $t \approx 0.54$s. This anticipation is sufficient to eliminate the overshoot in the pressure signal.



(a) (b)

Fig. 6.8: Pressure and input signal for a 110 step bar reference.

As a performance index for the proposed MIC-ILC algorithm, the overshoot of the pressure signal at each iteration is shown in Fig. 6.9. Convergence of the pressure overshoot towards the reference is reached in 10 iterations, eliminating the overshoot. The overshoot at the first iterations is 17 bar and, at the tenth iteration, the overshoot is 3 bar. Therefore, the overshoot is reduced by a 82% with respect to a PI controller, which is a enormous improvement.

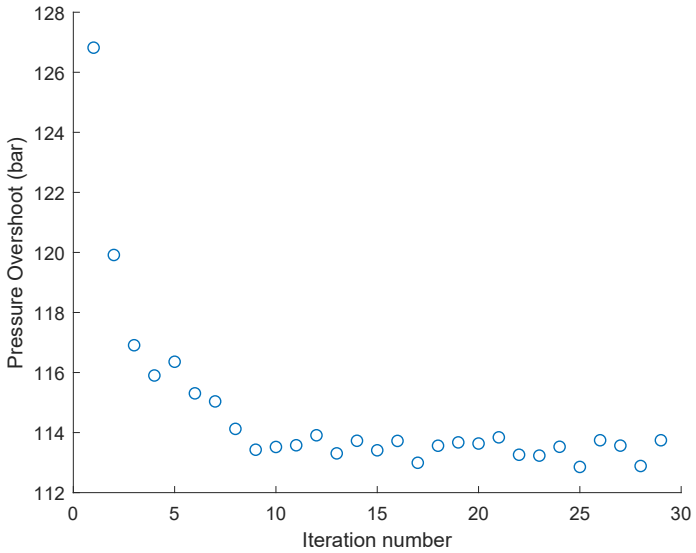


Fig. 6.9: Convergence to 110 bar pressure reference.

6.3.2 Actuator Fault

As we have explained in Section 6.2, in the hydraulic test rig there exist needle valves that simulate flow losses in the circuit. The adjustment of the needle valves is gradual and smooth for controlling the flow rate. The open position can be regulated manually to any opening degree.

If we look at Fig. C.1, by opening the needle valve 32.2 in the lower-cylinder, during the piston retraction a certain amount of the flow out of the cylinder piston-side will be delivered to the tank through said valve. This will result in a pressure decrease in the cylinder chamber, therefore the valve will have to close the spool more than in a normal scenario to obtain the desired pressure.

This flow loss can be regarded as a variation in the hydraulic conductivity, $K_{\mathrm{v}}(y_{\mathrm{v}})$, of the valve, as a specific spool position will result in a different pressure level than before. Therefore, with this flow loss, a similar scenario than the one carried out for the FF controller in Section 2.3.2, is tested. Where the $K_{\mathrm{v}}(y_{\mathrm{v}})$ nonlinearity introduced in Fig. 2.12, penalized the FF controller performance considerably.

For a pressure step reference of 110 bar, in Fig. 6.10 the difference in the PI controller response for a normal scenario and with the flow loss introduced can be seen. As the needle valve has been opened to a 10°, it yields a pressure level decrease in the cylinder piston-side chamber.

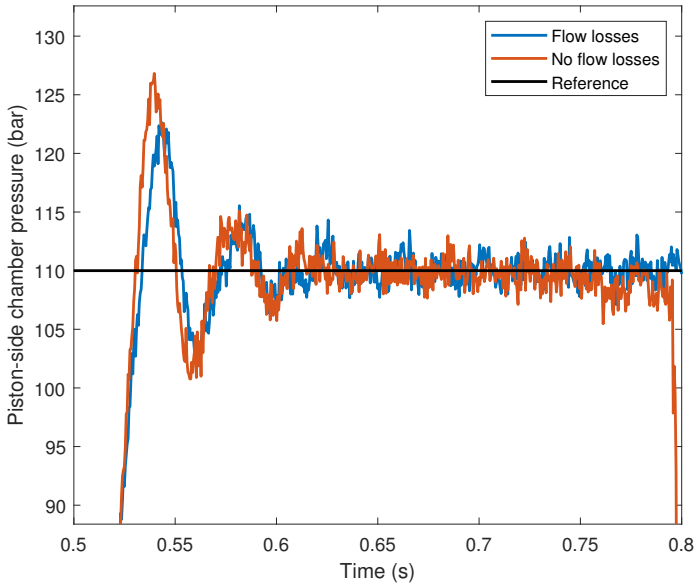
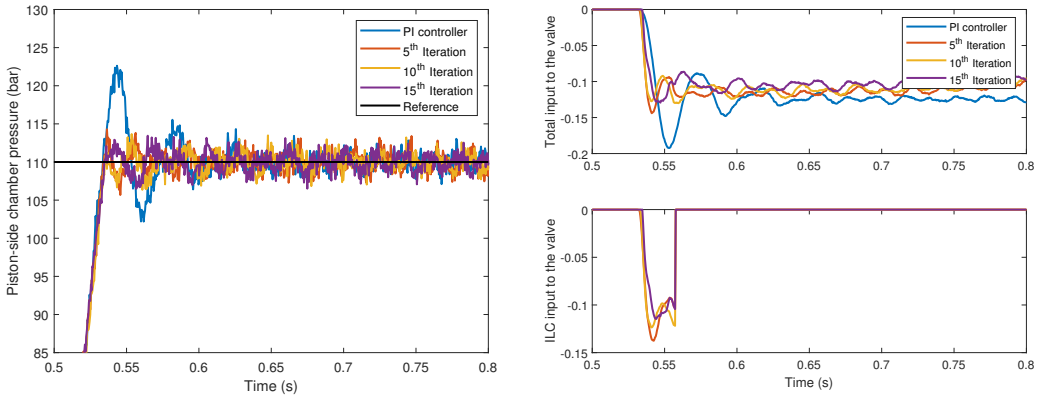


Fig. 6.10: Piston-side chamber pressure with and without flow losses, during *Free Fall* and *Drawing* phases.

We introduce the SISO MIC-ILC to see its robustness towards unmodeled losses. This can be seen in Fig. 6.11a, the MIC-ILC corrects the existing overshoot, and the oscillations are removed. See in Fig. 6.11b that now the total input to the valve is smaller than in a normal scenario in Fig. 6.8b, as due to the flow losses the valve has to be closed more to reach the same pressure level than before. The MIC-ILC automatically learns the new valve input.



(a) **(b)**

Fig. 6.11: Pressure and input signal for a 110 bar step reference with flow losses.

The overshoot of the pressure signal at each iteration is shown in Fig. 6.12. At the first iteration, with a PI controller, the overshoot is smaller than in Fig. 6.9. The MIC-ILC adjusts rapidly and at iteration 10 the overshoot is decreased considerably. At the first iteration, with the PI controller, the overshoot is 12.5 bar and, at the tenth iteration, the overshoot is 3.5 bar, a reduction of a 72% is achieved.

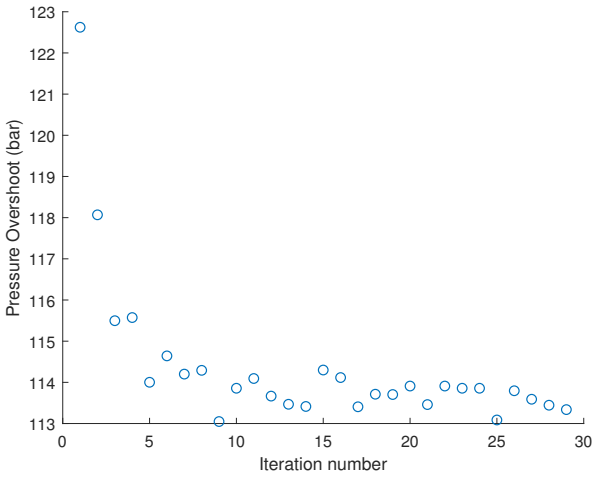


Fig. 6.12: Convergence to 110 bar pressure reference.

6.3.3 Actuator drifting

Sensors can be characterized by two fundamental attributes: precision and accuracy. Precision refers to how close are the values to each other, i.e. the ability to reproduce consistently a measurement. Accuracy refers to how close are the obtained measurements to the actual value.

Generally, sensor precision remains high. However, the sensor accuracy is often affected by drifting, which causes the measurement error to get worse over time. It is a natural phenomenon that affects all the sensors, regardless of the type.

Drift can also affect actuators and can turn into a tricky problem in the control of hydraulic circuits. For instance, if there exists a drift in the proportional valve, the control of the system will get affected. If the valve has a drift in the control signal, the valve at the centered position will be opened. This means the valve is not centered due to bad calibration, and the cylinder piston will creep, although no control signal exists.

This is a common issue in hydraulic actuators and it can be solved by turning a screw on the proportional valve solenoid to calibrate correctly the valve. However, the calibration should be done frequently and each actuator requires a different calibration.

To see if the MIC-ILC algorithm automatically adapts to an actuator drift, we introduce a manual drift to the lower-cylinder proportional valve input of a $\pm 5\%$. A negative drift will cause the valve to close the spool more, therefore less oil will be channeled through the valve to the tank, and the pressure inside the cylinder piston-side chamber will increase. On the contrary, a positive drift will cause the valve to open more the spool, and the pressure inside the piston-side chamber will decrease, benefiting the pressure control. Both scenarios can be seen in Fig. 6.13.

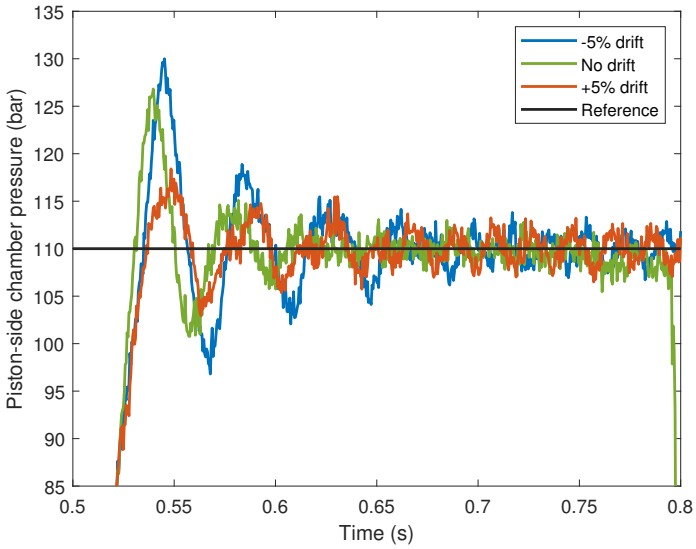
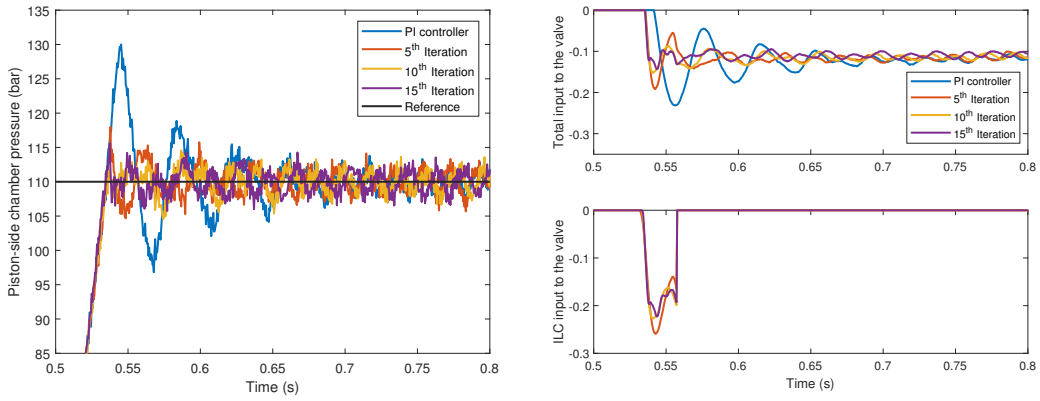


Fig. 6.13: PI controller 110 bar step for different drift scenarios.

As the positive drift results in a favorable scenario, we will test the MIC-ILC algorithm for the negative drift case. The pressure reference tracking over iterations is shown in Fig. 6.14a. The overshoot increase due to the drift introduced is counteract by the MIC-ILC by opening the valve spool position to a higher value, see Fig. 6.14b. In the normal scenario in Fig. 6.8b, the ILC input signal reached -0.13 spool position, and in this case it reaches -0.2 value, to counteracted the extra overshoot.

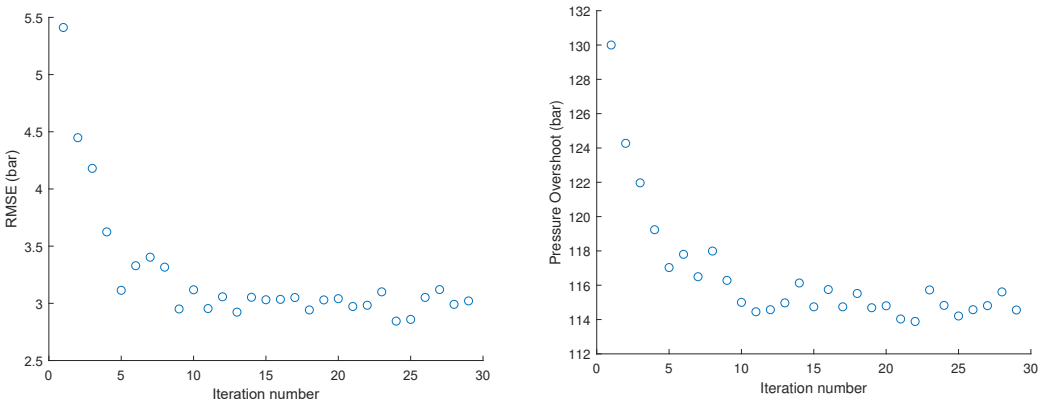
In order to see the error decrease evolution, we show the RMSE and the overshoot over iterations in Fig. 6.15. The overshoot is reduced a 86% with respect to the first iteration with a PI controller. The RMSE is reduced a 50%



(a) (b)

Fig. 6.14: Pressure and input signal for a 110 bar step reference with -5% drift.

with respect to the first iteration. Note that due to the compressibility of the oil, it is physically impossible to obtain a RMSE lower that 2.17.



(a) (b)

Fig. 6.15: RMSE and overshoot for a 110 bar step reference for -5% drift.

6.4 Position Control in hydraulic test rig

Here, the position control tests will be carried out, with the setup shown in Fig. 6.4b. We cannot test the same scenario as in Chapter 4, as we do not have two proportional valves to control the cylinder position and rod-side chamber pressure simultaneously. However, with the cylinders connected, we can still carry out a similar test to validate the MIMO MIC-ILC position control.

As both cylinders are coupled, both rod-side chambers are opened to the tank, they do not contribute to the operation. With the upper-cylinder's hydraulic pump, we control the lower-cylinder piston-side chamber pressure, while controlling both cylinders' joint position with the lower-cylinder proportional valve. This is a realistic scenario, as there exists the coupling of the pump and valve control loops.

Unlike the hydraulic pump used through Chapter 4, in the hydraulic test rig, the displacement of the pump is defined by the volume the pistons displace in one revolution. Therefore, instead of controlling the swash angle, the angular velocity of the pump is controlled.

The relationship between the fixed displacement axial piston pump outflow rate and the shaft speed is as follows:

$$q(t) = \frac{q_N}{\omega_N}\omega(t), \tag{6.3}$$

where $\omega(t)$ is the shaft rational speed (rad/s).

The position reference during the *Free Fall* and *Drawing* phases is shown in Fig. 6.16. This is the position the cylinders' joint position, from now on referred to as the slide, must follow. The transition between the two phases takes place at $t \approx 2$s, where the slide falling velocity is reduced.

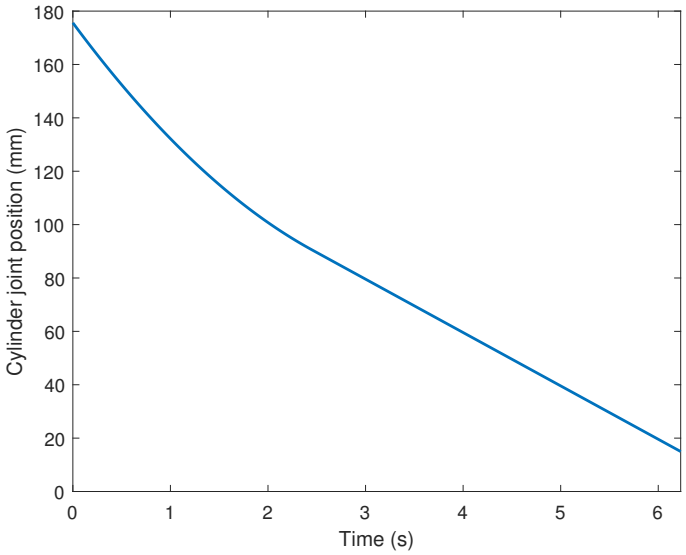


Fig. 6.16: Slide trajectory, *Free Fall* and *Drawing* phases.

We first design two PI controllers, to control the pump angular velocity and the valve spool position, and to follow the previously defined slide trajectory.

The pump PI controller gains values are: $K_{Pp} = 4$ and $K_{Ip} = 3.8$. The valve PI controller gains values are: $K_{Pv} = 1.5$ and $K_{Iv} = 0.25$. The resulting position and force control is shown in Fig. 6.17. When the pressure is higher that the reference is due to an excessive pump velocity, which yields faster falling velocity of the slide. On the contrary, when the pressure is lower than the reference, is due to insufficient pump velocity, which translates into a slide velocity slowing down.

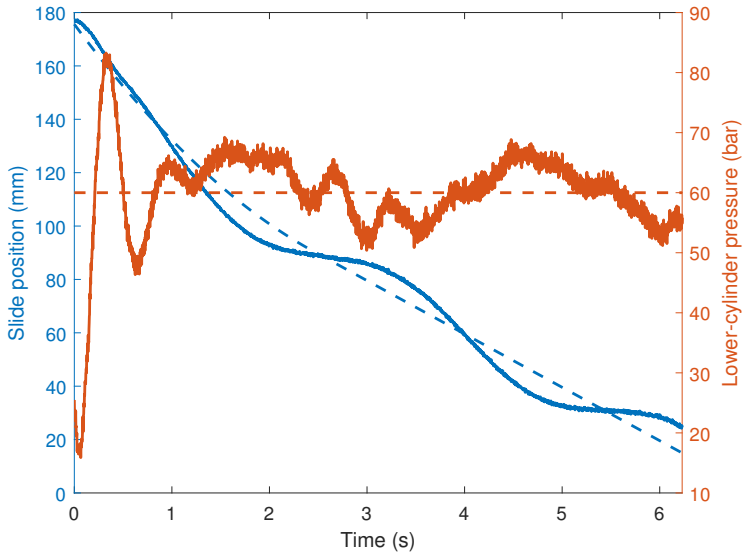


Fig. 6.17: Position and pressure signal during the press operation with PI controller.

In order to improve the position and force tracking, we implement the MIMO MIC-ILC algorithm designed in Chapter 5 in the hydraulic test rig. To that end, we introduce again the proposed plant inverse design for the learning matrix:

$$\boldsymbol{L}(s) = \underbrace{\hat{\boldsymbol{G}}^{-1}(s)\frac{\omega_c^2}{(s+\omega_c)^2}\frac{\omega_c^2}{(s-\omega_c)^2}}_{\hat{\boldsymbol{G}}_{zpf}^{-1}} + \boldsymbol{C}(s). \tag{6.4}$$

Now, $\boldsymbol{C}(s)$ consist of the pump and valve PI controllers introduced above, and $\hat{\boldsymbol{G}}^{-1}(s)$ is the simplified hydraulic test rig system with the parameters shown in Appendix C.0.3. To analyze the stability and convergence rate of the algorithm, we obtain the eigenvalues at each frequency, shown in Fig. 6.18.

At low frequencies, the eigenvalues are close to zero, as it has been pointed out in Fig. 6.18 for a frequency of $\omega = 2$ rad/s. At high frequencies the

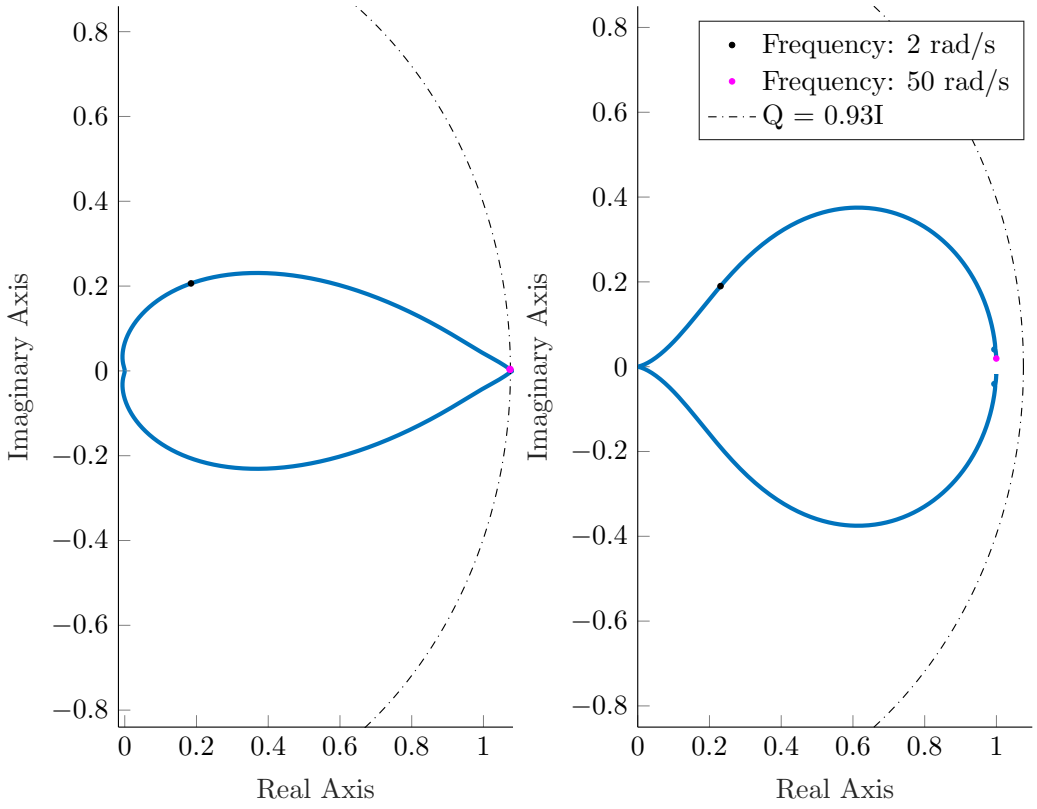


Fig. 6.18: Eigenvalues plot at each frequency for the hydraulic test rig model inverse design.

eigenvalues value increases, therefore, penalizing the convergence rate at those frequencies.

In the following section, several tests are carried out, to analyze the MIMO MIC-ILC performance towards different scenarios that can arise during the working operation of an actual hydraulic slide circuit.

6.4.1 Robustness towards modeling mismatch

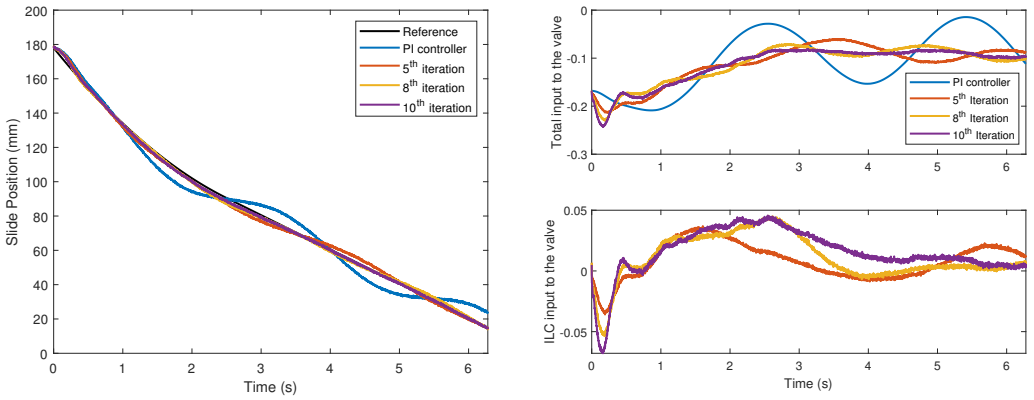
In the $\boldsymbol{L}(s)$ design, the plant model inverse with the low frequency poles and zeros is included, which are shown in Table 5.3 and in (5.13), respectively. As we have explained in Section 5.3, the terms containing β and P_B provide us with information on the rate the pressure increases. However, as shown in Appendix D, the β value from the hydraulic test rig differs from the one used in the theoretical analysis, which is less compressible.

The value of β has great importance in the modeling of the system dynamics, therefore, a model mismatch will considerably influence the control input

value obtained from $\boldsymbol{L}(s)$. In the same manner as for the SISO MIC-ILC implementation in Section 6.3.1, we use the theoretical β value in the $\boldsymbol{L}(s)$ design, to analyze the robustness of the MIMO MIC-ILC algorithm towards modeling differences.

From Fig. 6.17, we have seen that the PI controller performance is poor all along the two phases, therefore, we introduce the ILC signal during the *Free Fall* and *Drawing* phases. The oscillations in the position tracking are reduced considerably and a good position control is achieved, see Fig. 6.19a.

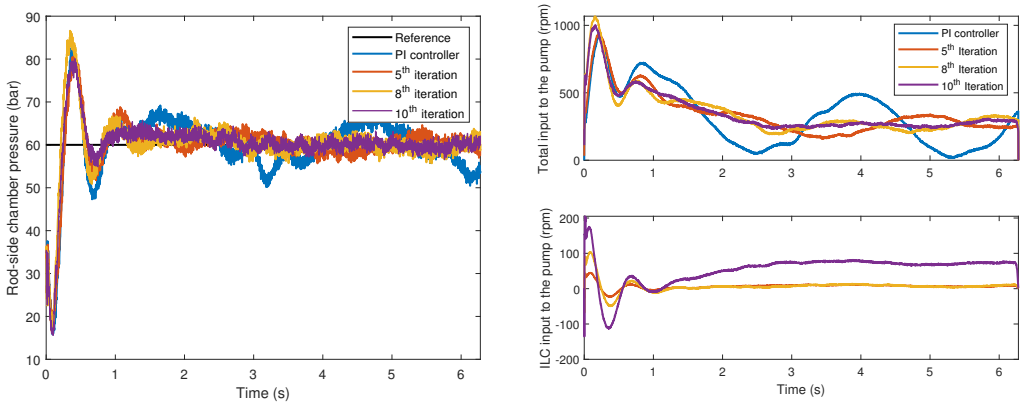
In Fig. 6.19b, how the MIC-ILC algorithm has adjusted the valve spool position to the slide velocity change can be seen. At iteration 10, from $t \approx 0$s to $t \approx 2.2$s, the valve spool position is reduced progressively from -0.2 to -0.1. This progressive valve closing is introduced to achieve fast velocity during the *Free Fall* phase and to prepare for the two phases transition where the velocity is reduced. Once in the *Drawing* phase, the valve spool position is already at a -0.1 value, which is maintained so constant falling velocity is achieved.



(a) (b)

Fig. 6.19: Position and valve input signal during *Free Fall* and *Drawing* phases in hydraulic test rig.

The pressure reference tracking is shown in Fig. 6.20a. The oscillations of the PI controller are reduced and the overshoot and undershoot are improved as well. The MIC-ILC first introduces a positive angular velocity to reach the pressure reference as fast as possible, see Fig. 6.20b. Once the pressure reaches the reference, the MIC-ILC introduces a negative angular velocity to reduce the signal overshoot. Once the signal is stable around the reference, the MIC-ILC introduces a positive angular velocity to keep the pressure constant.



(a) (b)

Fig. 6.20: Pressure tracking for a 60 bar reference and pump velocity during *Free Fall* and *Drawing* phases in hydraulic test rig.

As a performance index for the proposed MIMO MIC-ILC algorithm, the RMSE between the position reference and the slide position signal is shown in Fig. 6.21. The RMSE is reduced by a factor of 4 in the position tracking, and a fast convergence rate is obtained as at the seventh iteration the error is considerably reduced, with respect to the first iteration with the two PI controllers.

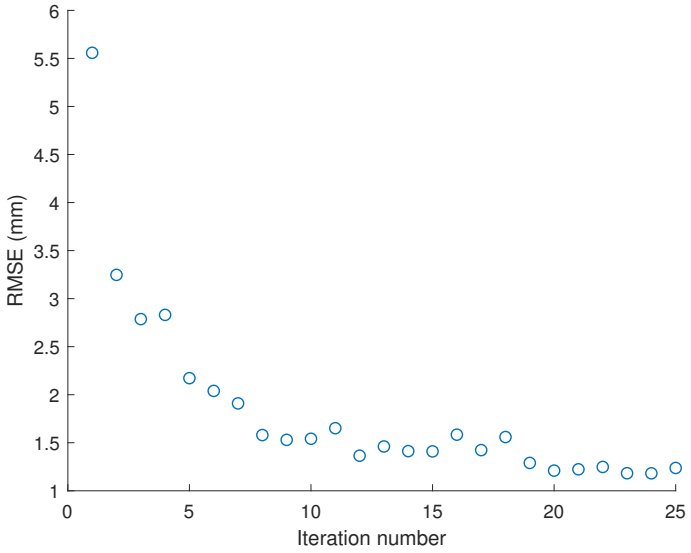


Fig. 6.21: RMSE between the position reference and the slide position over iterations.

The RMSE between the 60 bar pressure reference and the lower-cylinder piston-side chamber pressure signal is shown in Fig. 6.22. The error is

reduced by a factor of 2.6 regarding the first iteration with the two PI controllers.

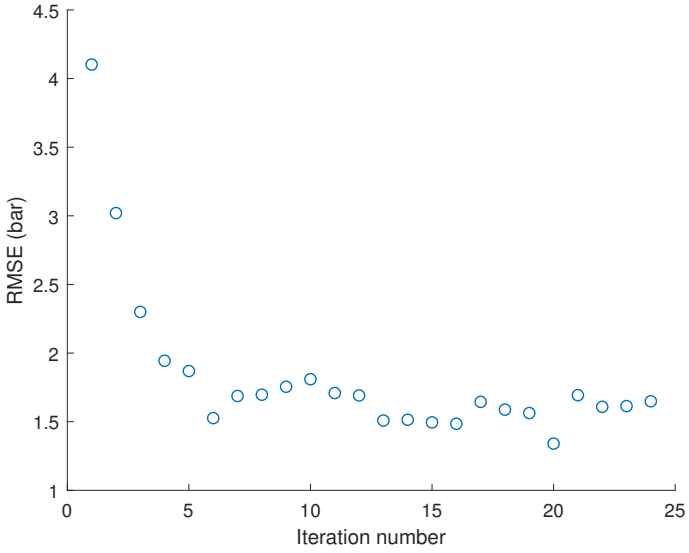


Fig. 6.22: RMSE between the pressure reference and the piston-side pressure over iterations.

6.4.2 Actuator Fault

In this section, we introduce a pump leakage into the pump with which the cylinder pressure is controlled. This scenario can be regarded as a faulty scenario, as the pump requires more oil flow rate to reach a specific pressure level than in a normal scenario. In this way, we can see how the MIC-ILC automatically adapts to a leakage in an actuator.

In order to simulate a pump leakage, we open 40° the 7.1 needle valve, see Fig. C.1. By opening this valve, a certain amount of the oil flow rate displaced by the pump will go through needle valve 7.1 to the tank. Therefore, more flow rate will be needed from the pump to reach a specific pressure level in the lower-cylinder piston-side chamber.

In Fig. 6.23, the comparison of the pressure control in a faulty scenario and a non-faulty scenario is shown. In the pump leakage scenario, the pressure overshoot is less and the pressure tracking is slightly better than in the leakage scenario. This is a consequence of the leakage introduced, most of the oil returns to the tank through the needle valve 7.1, and less oil flow rate reaches the cylinder chamber, resulting in a less pressure increase. However,

note that in order to reach the same pressure level of 60 bar, the pump has to run faster, thus increasing the pump's energy consumption.

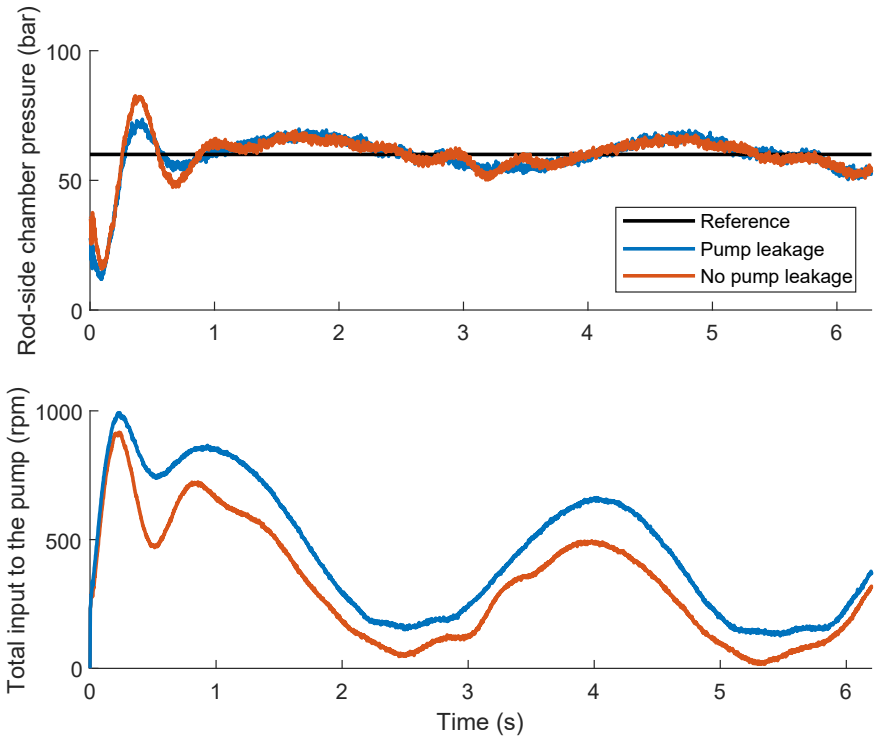


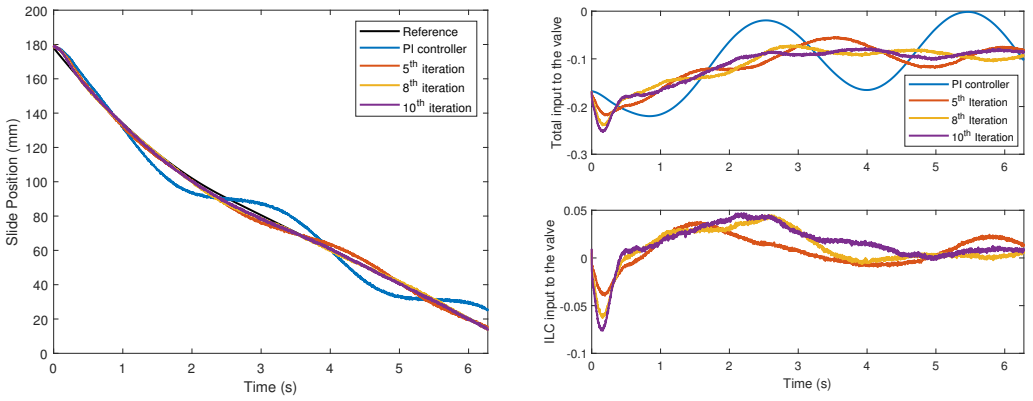
Fig. 6.23: Pressure control and pump velocity for a normal scenario and a pump leakage scenario.

We introduce the MIMO MIC-ILC to analyze its performance under a faulty pump scenario. The position tracking, see Fig. 6.24a, is similar to that shown in the previous section with the hydraulic compressibility mismatch, as a less aggressive and oscillating pressure control favors the position control.

The pressure reference tracking, however, gets considerably affected by the leakage introduced, as more pump power is required to reach the same pressure level. In Fig. 6.25a, the pressure reference tracking over iterations is shown. As iterations go on, the oscillations existing with the PI controller are removed, and a precise pressure reference tracking is achieved.

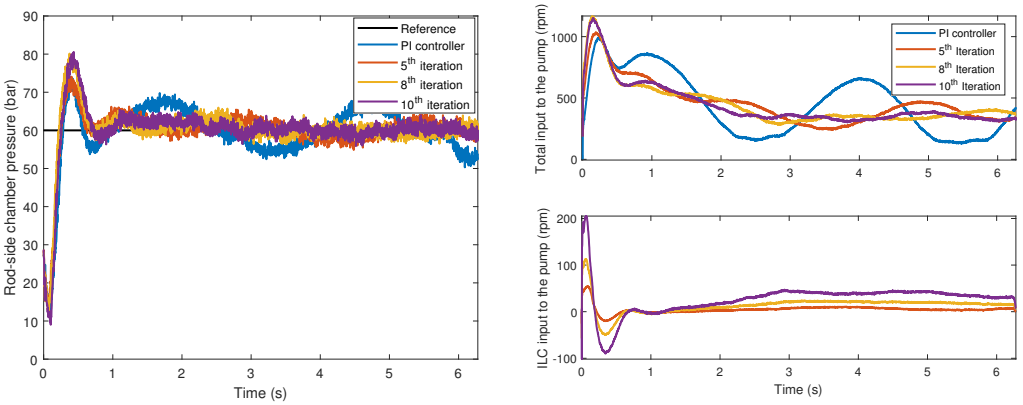
The MIC-ILC contribution to the pump velocity is shown in Fig. 6.25b. In comparison to the previous section with the hydraulic compressibility mismatch, see Fig. 6.20b, the ILC introduces a higher pump velocity, in order to counteract the pump leakage. See that the first oscillation of ILC input reaches 200 rpm, whereas in Fig 6.20b, it reached 170 rpm.

For the performance index, the RMSE between the 60 bar pressure reference and the lower-cylinder piston-side pressure signal is shown in Fig. 6.26. The



(a) (b)

Fig. 6.24: Position and valve input signal during *Free Fall* and *Drawing* phases in hydraulic test rig.



(a) (b)

Fig. 6.25: Pressure tracking for a 60 bar reference and pump velocity during *Free Fall* and *Drawing* phases in hydraulic test rig with pump leakage.

RMSE for the position control is not shown, as the result is similar to that obtained in Fig. 6.21.

As the pressure overshoot at the first iteration is smaller due to the leakage introduced, the RMSE value is smaller too. Still, the RMSE is improved by a factor of 2.5, with respect to the first iteration with the PI controller.

6.4.3 Actuator drifting

In the same way as in Section 6.3.3, we will test the MIMO MIC-ILC for an actuator drift scenario. For the MIMO control problem, as both control loops

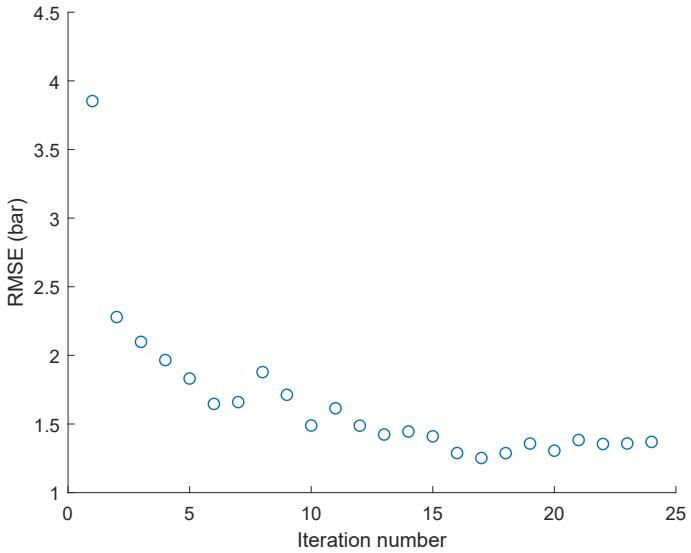


Fig. 6.26: RMSE between the pressure reference and the piston-side pressure over iterations.

are coupled, the actuator drift will affect both the position and the pressure control.

The slide position and the valve input signals are shown in Fig. 6.27. With the -5% drift, the valve spool signal reaches negative values as the centered position is not at 0 anymore. The -5% drift case has the higher amplitude in the oscillations of the valve input which results in a very poor position tracking.

Although the drift has been introduced in the valve and not in the pump, the pressure tracking gets affected by the control loops coupling, as it is shown in Fig. 6.28. The worts case, similar to the position signal, is with -5% drift, where the oscillations in the valve input affect the pressure control, which oscillates considerably around the reference as well.

We introduce the MIMO MIC-ILC to the -5% drift case, as it is the worst scenario, to analyze the algorithm performance. The position tracking over iterations is shown in Fig. 6.29a, although the initial iteration is worse due to the drift introduced, the MIC-ILC is able to correct it as iterations go on. The MIC-ILC input signals are shown in Fig. 6.29b.

In Fig. 6.30 we can see the position error evolution with the RMSE over iterations. The first iteration RMSE is higher than in a non-drift scenario, however, it does not affect the MIC-ILC performance and the same steady-state RMSE value of one is obtained after 10 iterations. The position error is reduced by a factor of 10.

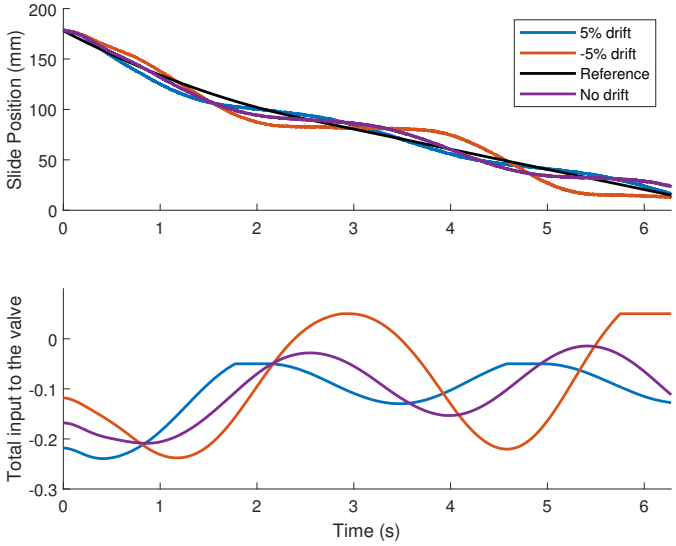


Fig. 6.27: Position and valve input signal with ±5% drift, during *Free Fall* and *Drawing* phases.

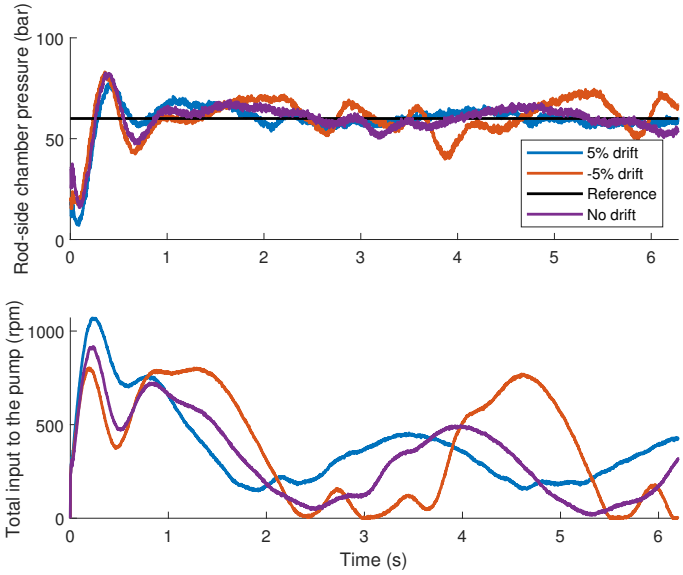
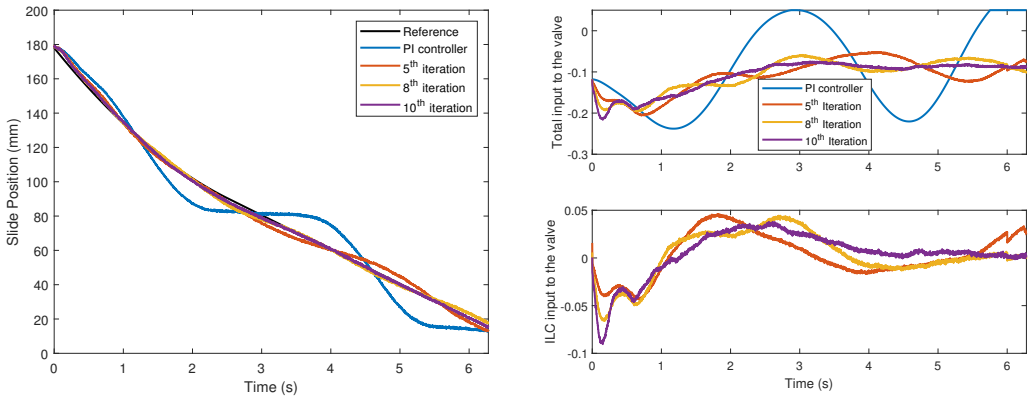


Fig. 6.28: Lower-cylinder piston-side pressure and pump velocity with ±5% drift, during *Free Fall* and *Drawing* phases.

The pressure reference tracking is shown in Fig. 6.31a, the oscillations and overshoot of the first iteration with the PI controller are reduced considerably over iterations. The MIC-ILC contribution to the pump velocity is shown in Fig. 6.31b.



(a) (b)

Fig. 6.29: Position and valve input signal during *Free Fall* and *Drawing* phases in hydraulic test rig with -5% drift.

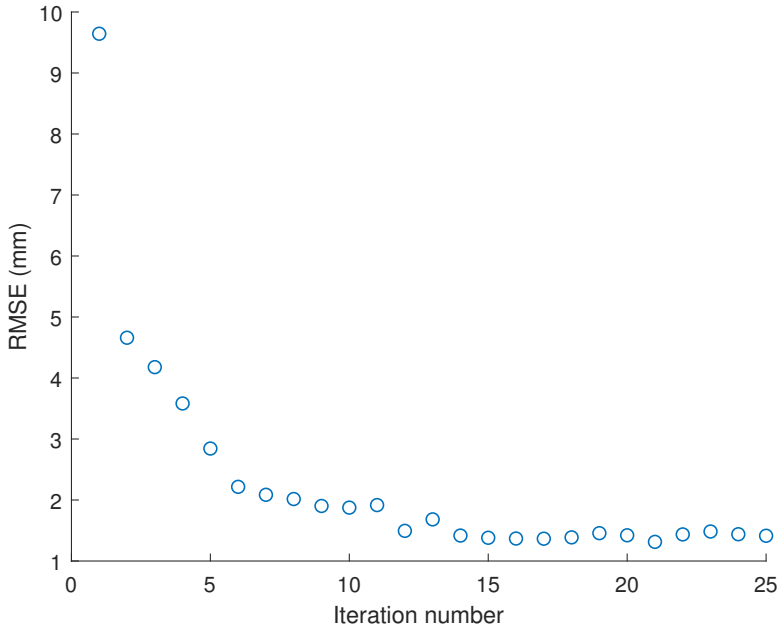
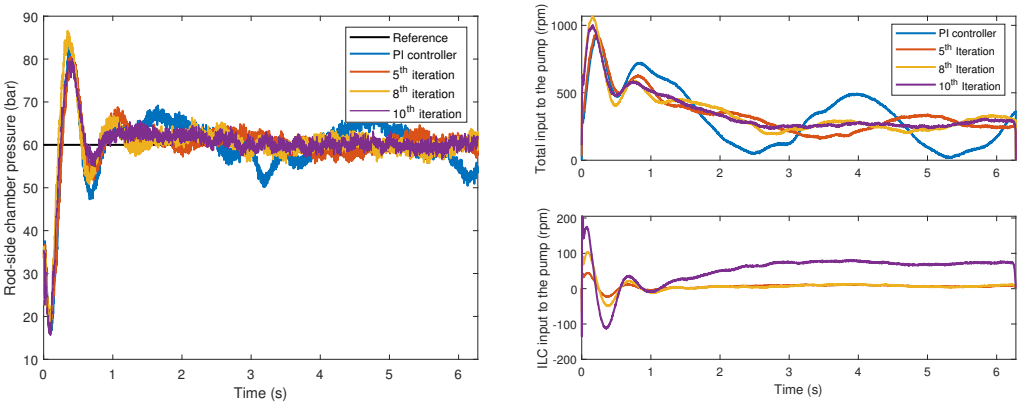


Fig. 6.30: RMSE between the position reference and slide position over iterations.

As a performance index of the pressure error reduction the RMSE over iterations is shown in Fig. 6.32. The initial RMSE pressure error is reduced a 80%. Despite the drift introduced in the valve, fast convergence rate is obtained, and at iteration 15 the RMSE already converges.



(a) (b)

Fig. 6.31: Pressure tracking for a 60 bar reference and pump velocity during *Free Fall* and *Drawing* phases in hydraulic test rig.

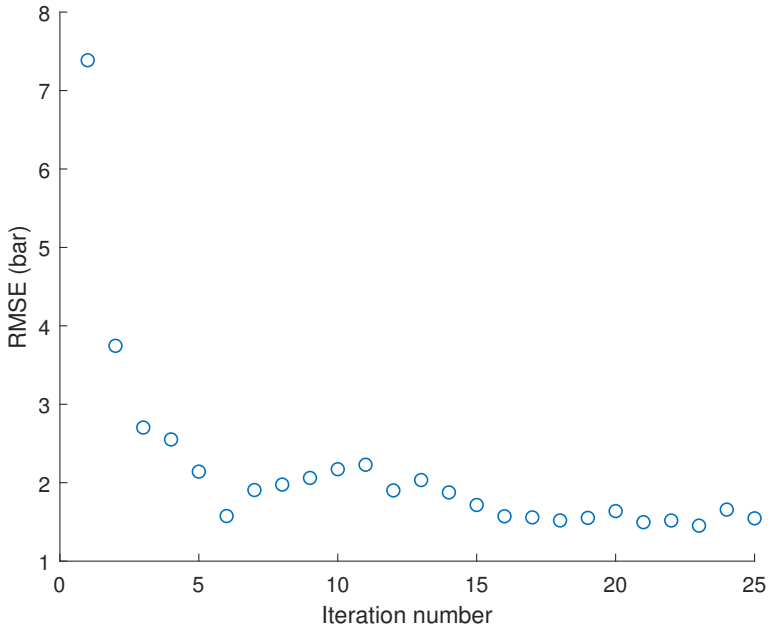


Fig. 6.32: RMSE between the pressure reference and the piston-side pressure over iterations.

6.5 ILC implementation in Digital Twin

One of the main technological trends highlighted in the recent years in the industry is the creation of Digital Twins. The Digital Twin is a virtual replica of a system that simulates the behavior of its real counterpart. This

technology allows the possibility to develop new and innovative control strategies without having a real machine.

Figure 6.33 shows the Digital Twin platform used for implementing the proposed ILC algorithms. It consists of two main parts: a digital model of a mechanical press embedded in a Beckhoff IPC and a mechanical press real control hardware embedded in Siemens Simotion Industrial Controller provided by FAGOR ARRASATE.

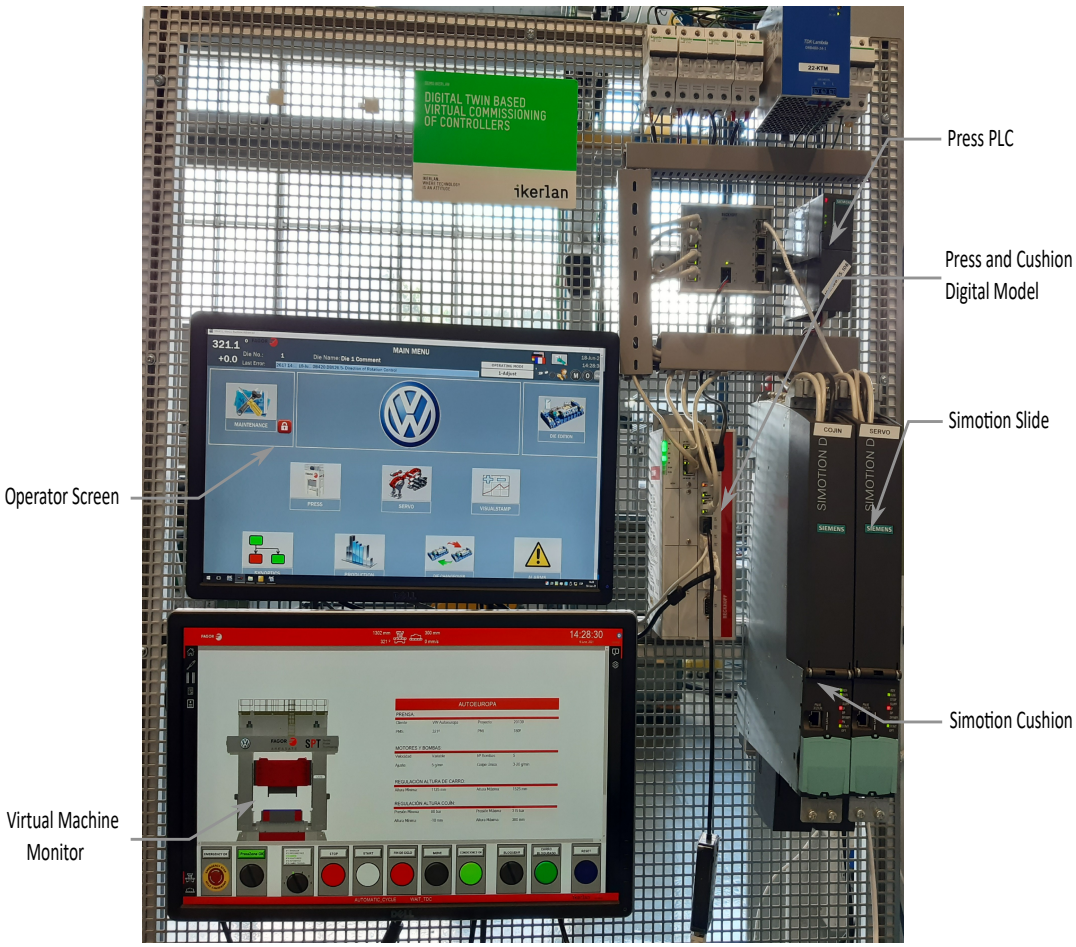


Fig. 6.33: Digital Twin platform at Ikerlan Technology Research Center.

In the digital model of the press, in order to reproduce a virtual representation of the working operation of a mechanical press, the press models designed with Ikerlan's library have been integrated on a Beckhoff Industrial Controller. This is carried out by compiling the Matlab/Simulink press models into TwinCAT3, which is a platform developed by Beckhoff. The integration into Beckhoff's industrial environment allows RT communications through deterministic protocols such as EtherCAT or ProfiNET. These deterministic protocols allow the communication between the Simotion control hardwares

with Beckhoff's virtual model in real execution time. We will not go further into details with the integration of the press models into TwinCAT3, for a thorough explanation refer to [66].

The real control hardware collects the dynamic working signals of the simulated mechanical press. These signals are processed by the controller and delivered back to the corresponding actuators of the simulated mechanical press.

Note that we only have available the real control hardware corresponding to a mechanical press, therefore, only the SISO ILC algorithm for the force control is tested in the Digital Twin platform.

The mechanical press control is carried out by two different Siemens Simotion D445-2, one of which is used to control the press slide position trajectories, the other one is used to control the cushion force and position. With the slide controller, we can generate new position trajectories based on the velocity, drawing distance, number of pumps, etc. With the cushion controller, we can control the desired cushion blank-holder force. The SISO MIC-ILC algorithm is integrated in the cushion's Simotion controller.

The entire process is governed by a PLC (S7-300), which carries out the communication between the two Simotions and the virtual model in Beckhoff. There have been installed two screens: an actual operator's screen which can be found in a real mechanical press, where the press cycle parametrization is done. The other one is the virtual machine monitor and gives data of every variable simulated in the virtual press model.

Every Digital Twin connection is depicted in Fig. 6.34. The communication between the real press controllers, i.e., slide controller, cushion controller, and PLC is carried out by Profinet Isochronous Real Time (IRT), which ensures fast cycle times and that frames are transmitted and received on schedule and in order [90]. The communications between the press controllers and the Beckhoff IPC is carried out with Profinet RT [91]. Profinet RT is commonly used for those applications where standard cyclic data acquisition is enough, as it is the case in the communication between the controller and Beckhoff IPC.

The integration of the SISO MIC-ILC into the Digital Twin platform represents a major step up from TRL 6. We implement the ILC force control algorithm into a real press controller environment in Simotion. This requires addressing the RT implementation in the actual control hardware, considering the computation and memory requirements.

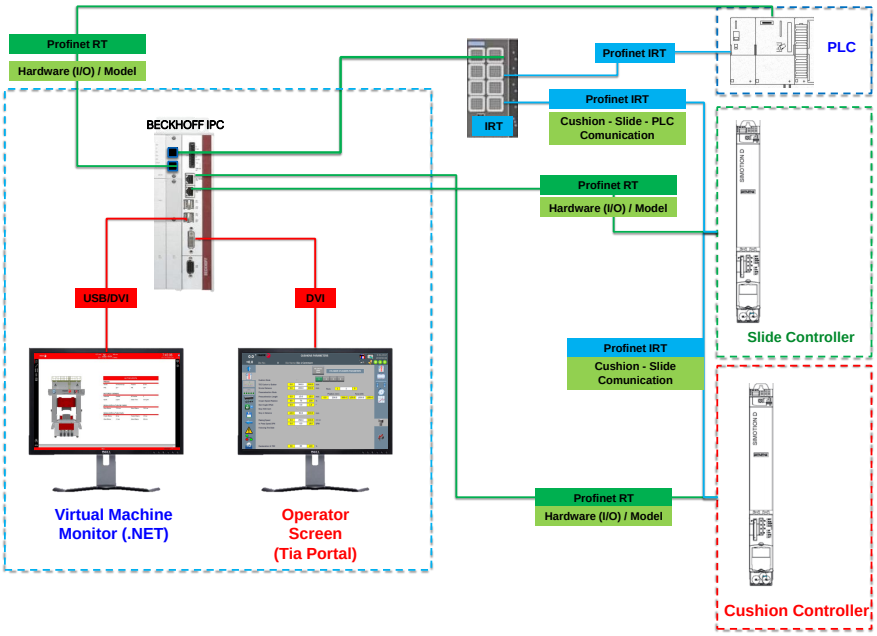


Fig. 6.34: Digital Twin platform elements and their connections.

In order to design the ILC algorithm in Simotion hardware, Simotion SCOUT has been used, which is an engineering software used to perform hardware and network configuration, programming and commissioning [92]. The algorithm is implemented in PLC programming and it is executed in a time-triggered task.

In Simotion, there exist five levels of execution tasks, see Fig. 6.35, giving priority to the system execution levels beyond the user's execution levels.

- System level: DP, Servo and IPO.
- Interruption: Interrupt.
- Users programs: Round Robin.

Each system level is time-triggered, i.e. a sampling time is specified for each one. During the force control, it is necessary to send an input to the valve continuously, therefore, the designed algorithm is implemented under the Servo task, to which a sampling time of 0.002s has been specified. This task is activated during the force control, and deactivated otherwise.

In the next section, the FF controller proposed in Section 2.3 and the ILC algorithm proposed in Chapter 3 will be validated in the Digital Twin.

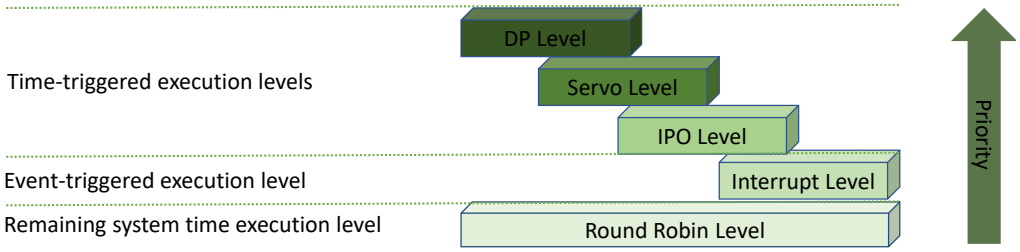


Fig. 6.35: Simotion execution levels.

6.5.1 Feed-forward control in Digital Twin

The Simotion controllers have been provided by FAGOR ARRASATE press manufacturer. Generally, the control in Simotion is carried out with hydraulic axis technology objects, which provide straightforward functionalities for a specific controlling element, such as proportional valves in the force control.

The Simotion axis technology can be regarded as the starting point of the force control problem, since it is the controller used nowadays in real presses. Therefore, we can compare in the Digital Twin the controllers developed through this document, i.e. FF controller and SISO ILC algorithm, with the actual press controller with the axis technology.

Figure 6.36 shows the typical configuration of a PID controller axis in Simotion. It consist of a PID controller, which gains must be defined manually. With the pre-control value, an FF input can be added to the PID controller output signal. Usually, the D-term of the PID controller is set to zero, to avoid noise problems. Furthermore, to avoid extra designing burden, the pre-control is not used as it requires the tuning of a weighting factor.

We can carry out a test with the actual PI controller used in real presses by FAGOR ARRASATE, to analyze its performance and compare it with the controllers proposed in this document. FAGOR ARRASATE carries out the force control in [KN] units, therefore in this section the force signal of the cylinder instead of the piston-chamber pressure signal will be used. The two chambers pressure and force relate to the following expression: $F = P_A A_A - P_B A_B$.

Figure 6.37 shows the FAGOR ARRASATE PI controller response to a 500 KN step reference. There is a force overshoot of 50 KN and at the end of the step the force signal decays slightly. This force loss results from the slowing down

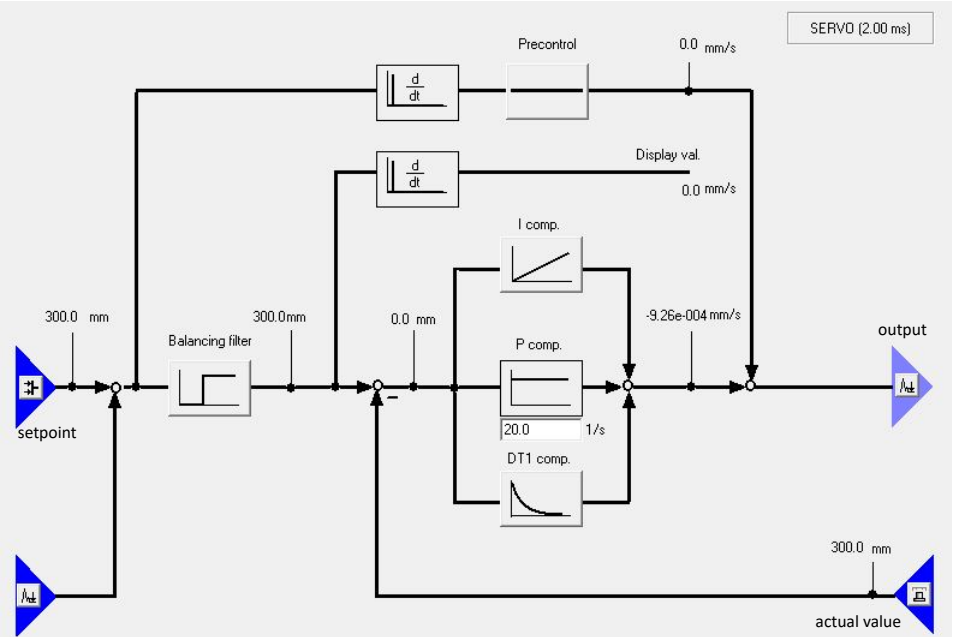


Fig. 6.36: Simotion PID controller axis.

of the cushion velocity that, at the end of the step, is close to zero. This decay must be prevented, as it could affect the workpiece design.

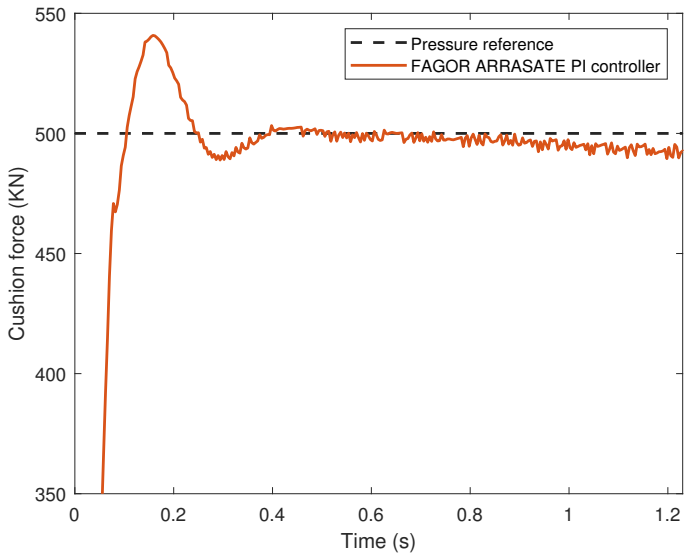


Fig. 6.37: FAGOR ARRASATE PI controller response to a 500 KN step reference.

We implement the FF controller into the cushion Simotion device to improve the FAGOR ARRASATE PI controller performance. As the FF controller consists of an external input proportional to the velocity and a gain that

multiplies the output of the PI controller, it can be easily implemented in Simotion. The resulting Simotion control loop is as shown in Fig. 6.38.

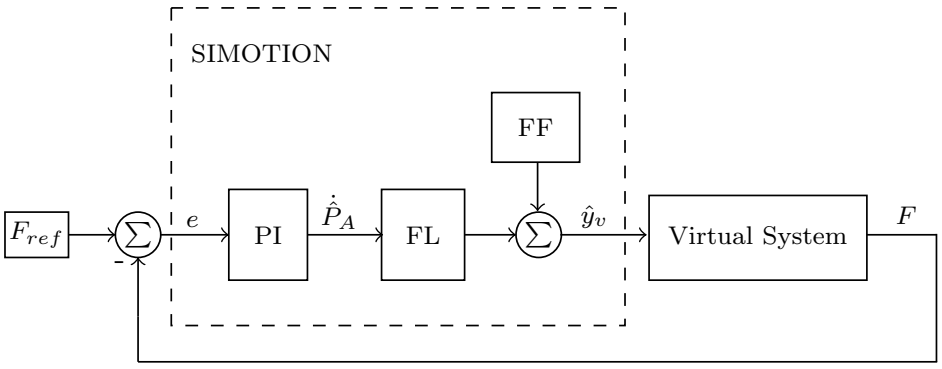


Fig. 6.38: Cushion force control block diagram with FF controller in Digital Twin.

We know, from Section 2.3.1, that in order to successfully implement the FF controller the force percentage (FP) at which the signal is introduced must be calculated. Using (2.9), we obtain a FP of 66% for a 500 KN reference, which yields the force signal response in the Digital Twin shown in Fig. 6.39.

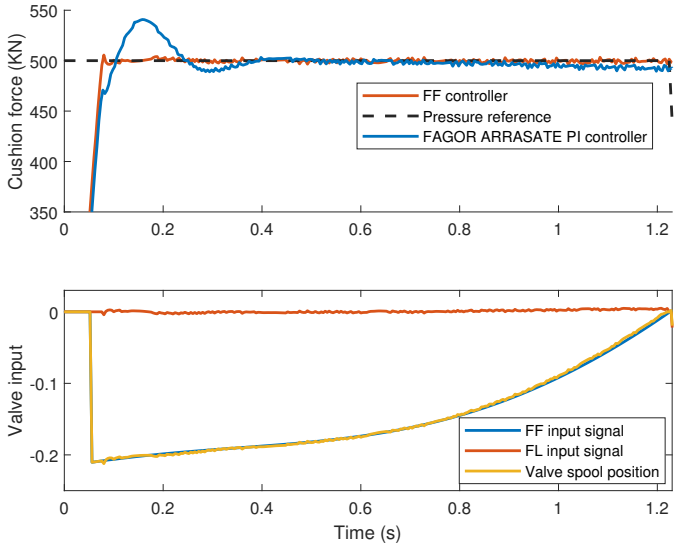
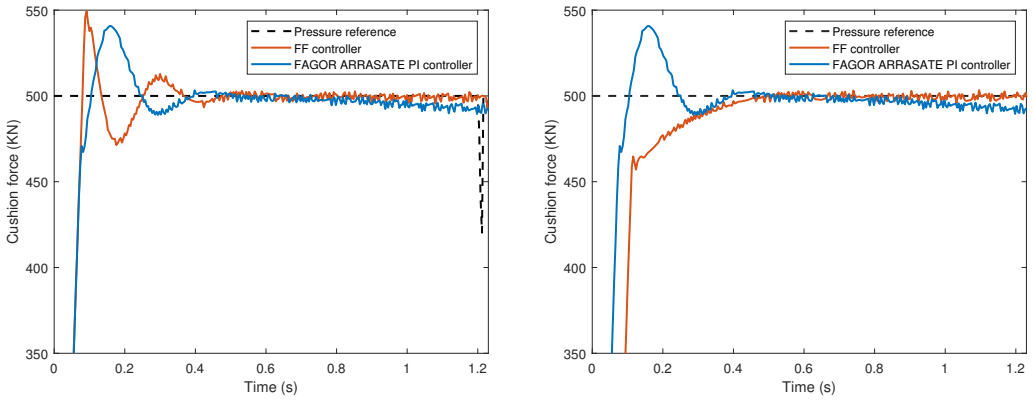


Fig. 6.39: FF controller response to a 500 KN step reference and valve input signals in Digital Twin.

The FL input signal contribution is small, however, it maintains the force signal around the reference so the slowing down of the cushion velocity does not affect the force tracking at the end of the step. With the FF input signal the force overshoot is reduced, as it opens the valve spool at a position proportional to the velocity of the cushion, anticipating it.

However, as we have shown in Section 2.3, if the FP is not calculated correctly, the performance of the FF controller deteriorates notably. In Fig. 6.40, two different scenarios are shown for a 500 KN force reference: in Fig 6.40a, the FF signal is introduced too late, at a 86% FP. In Fig 6.40b, the FF is introduced too soon, at a 50% FP. Recall that the correct FP is 66%.



(a) (b)

Fig. 6.40: 500 KN force step reference tracking with FF controller at two different FP in Digital Twin.

If the FF is introduced ahead of time, the valve opens too early and the force signal takes a large time to reach the pressure reference. If the FF is introduced too late, the velocity disturbance is not eliminated and an overshoot is obtained in the force tracking. Unlike with FAGOR ARRASATE controller in Fig. 6.37, no force loss exists at the end of the step, as the FL term corrects it.

Apart from the dependency of the FF controller to the instant at which it is introduced, as shown in Section 2.3.2, if there exists any model mismatch in the FF controller design, the performance of the force tracking gets affected as well.

We consider the same scenario as the one shown in Fig. 2.11, where a linear valve conductivity function is used in the FF design, but the real valve has a nonlinear conductivity function. This model discrepancy results in the force reference tracking shown in Fig. 6.41.

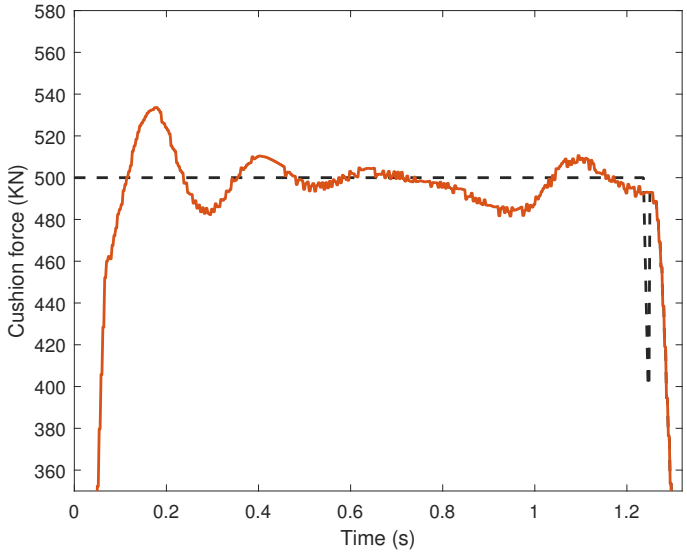


Fig. 6.41: FF controller force reference tracking under valve model discrepancies in the Digital Twin.

6.5.2 MIC-ILC in Digital Twin

In the same way as in Chapter 3, we use ILC to overcome the limitations of the FF controller. The resulting SIMOTION control loop with the ILC implementation is shown in Fig. 6.42.

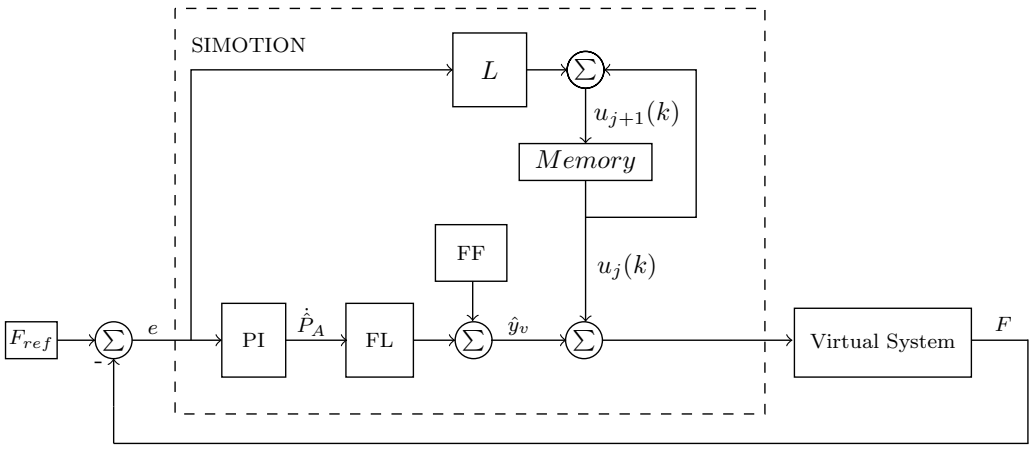
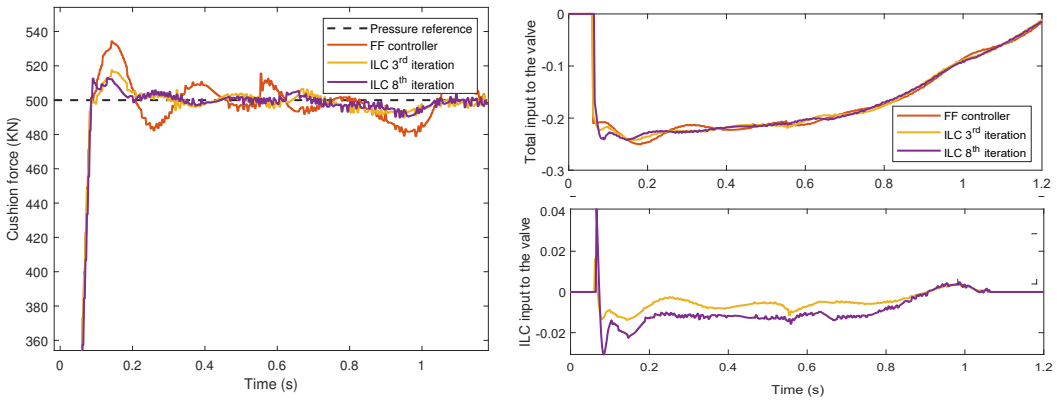


Fig. 6.42: Cushion force control block diagram with ILC controller in Digital Twin.

We analyze how the MIC-ILC behaves in the Digital Twin under two different scenarios: with the FF controller introduced at an incorrect FP and with the FF controller with valve model mismatch.

In Fig. 6.43, we introduce the MIC-ILC algorithm to improve the FF controller performance under valve model mismatches. At the first iteration, where no ILC signal exists, the FF performs poorly with an oscillating force reference tracking. As iterations go on, the ILC reduces those oscillations and the initial overshoot as well.

In Fig. 6.43b, at iteration eight, the valve spool is opened faster and to a larger value, so the initial force overshoot is reduced. The oscillations of the FF control in the total input to the valve are also reduced, obtaining a more accurate force reference tracking.



(a) (b)

Fig. 6.43: Pressure and input signal for a 240 step bar reference with the P-ILC algorithm.

The RMSE between the force signal and the 500 KN force reference is shown in Fig. 6.44. The error is decreased by a factor of three and although perfect tracking is not achieved, the FF performance is considerably improved.

Now, we evaluate the SISO MIC-ILC performance in the Digital Twin when a late FP is calculated for the FF controller. This will cause a force overshoot similar to the one shown in Fig. 6.40a, with a 86% FP. The MIC-ILC performance is shown in Fig. 6.45. At the first iteration, where no ILC signal exists, there exists a considerable overshoot in the force signal, that reaches 600 KN.

As iterations go on, the ILC algorithm corrects the total input to the valve by opening earlier the valve. See in Fig. 6.45b, that with the FF controller the valve maximum spool position takes place at $t \approx 0.15$s and at the 15$^{\text{th}}$ iteration the valve is already opened at $t \approx 0.1$s. This anticipation results in a considerable reduction of the force overshoot.

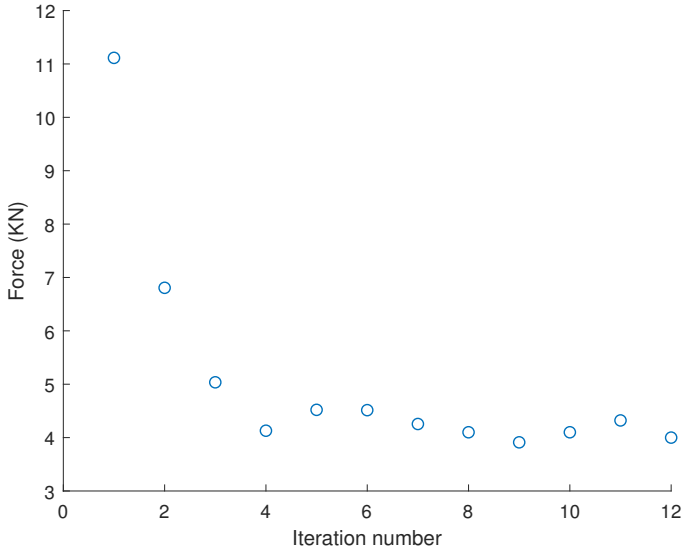


Fig. 6.44: RMSE between cushion force and 500 KN force step reference over iterations.

The overshoot over iterations is shown in Fig. 6.46. The force overshoot is reduced by 100 KN, with respect to the first iteration with the FF controller.

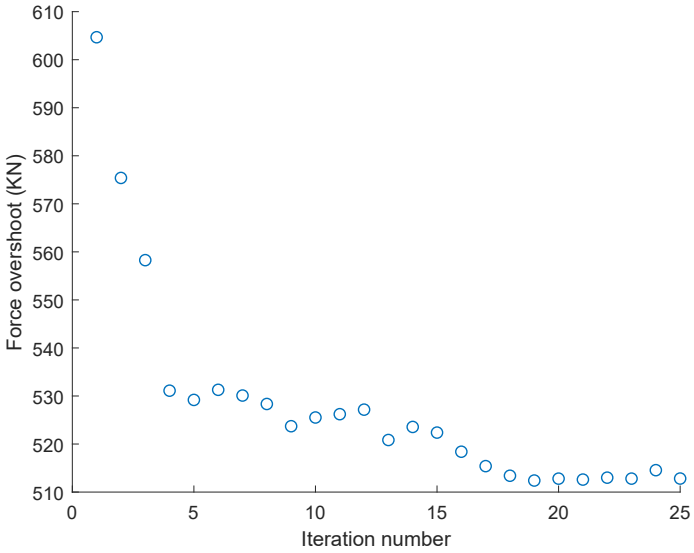
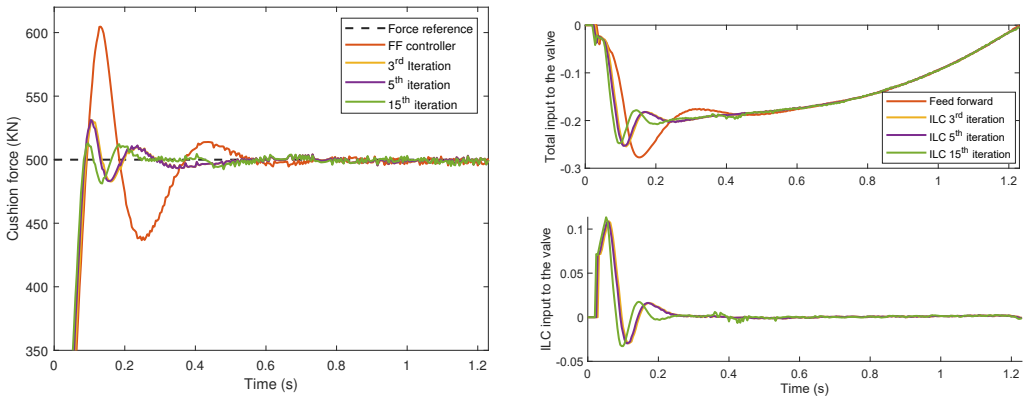


Fig. 6.46: Overshoot of the cushion force signal over iterations.



(a) (b)

Fig. 6.45: Cushion force control with FF controller and ILC algorithm, under late FP conditions.

6.6 Conclusions to Chapter 6

The SISO and MIMO ILC algorithms proposed in this thesis have been proven in high-fidelity nonlinear simulations, from which satisfactory results have been obtained. Based on the TRL scale originated at the NASA, the implementation of a technology in high-fidelity simulations validates the technology in a relevant scenario, which corresponds to TRL 5.

In order to show the maturity of the proposed controllers, in this chapter we have demonstrated that the controllers work in a relevant and operational environment and not only in simulation. To that end, we have presented two different scenarios from which we can achieve TRL 6 and 7, for the MIMO and SISO ILC control algorithms.

Both MIMO and SISO ILC algorithms have been implemented in the hydraulic test rig available at Ikerlan. The hydraulic test rig consists of every hydraulic element that we can find in a real press, therefore, a validation of the controllers in a relevant environment can be demonstrated, which corresponds to TRL 6.

The results obtained from both MIMO and SISO MIC-ILC show that the proposed controllers achieve satisfactory results, as good tracking and fast convergence rate is got. Different scenarios have been proposed, from a modeling mismatch introduction in the MIC-ILC design, to actuator faults. Under every scenario, the MIC-ILC has remained stable, and no instabilities have appeared.

The good performance of the MIC-ILC under faulty scenarios, can take one to think that ILC could possibly be used as a Fault Tolerant Control algorithm, as its adaptability to both valve and pump actuator faults has been proven through Sections 6.3 and 6.4.

Additionally, a Digital Twin platform has been built to validate the SISO MIC-ILC in an operational environment. We have in our possession the real control hardware of a mechanical press, provided by FAGOR ARRASATE, in which we have implemented the proposed SISO MIC-ILC. The MIC-ILC has been embedded in Siemens Industrial Controller platform, for which PLC programming has been done. The Digital Twin environment enables RT representation of the mechanical press under actual industrial control platform operation. This entails a jump into TRL 7.

In the Digital Twin platform we have implemented the FF controller proposed in Section 2.3 and the SISO MIC-ILC. The FF controller has shown satisfactory results in ideal conditions, where no model uncertainties exist. However, its model dependency makes it unsuitable for the hydraulic cushion system, where uncertainties abound.

The SISO MIC-ILC validation has been proven for two scenarios in the Digital Twin environment: under FF controller model uncertainties, and with a late FP in the FF controller. In the former, the oscillations arising due to the model mismatches are eliminated by the MIC-ILC. In the latter, the late FP results in a force overshoot which is reduced by the MIC-ILC as iterations go on.

7 Conclusions

7.1 General conclusions

This thesis analyses the control problems occurring in mechanical and hydraulic presses. The analysis is based on the identification of the actual control limitations in industrial presses, followed by the analytical development of the control schemes proposed to counteract those limitations.

On the one hand, regarding mechanical presses, the control of the cushion plays a fundamental role in deep drawing processes in which customer-specific force profiles need to be ensured. There usually exist performance specifications regarding the maximum force peak and the settling time of the force signal, which should be satisfied to guarantee the correct forming of the workpiece.

The fast dynamics, nonlinear behavior and external disturbances make it difficult to ensure said specifications with the actual press controllers. To that end, we have proposed a single-input-single-output (SISO) Iterative Learning Control (ILC) algorithm, that allows to improve the press performance by learning from previous experience.

The SISO ILC algorithm design is based on the known plant dynamics from which the unknown high frequency components have been discarded. With this design, fast convergence is obtained as well as the improvement of the algorithm stability.

In order to analyze the convergence rate and stability of the SISO ILC algorithm, a graphical approach based on Nyquist diagram has been proposed. By following this intuitive graphical approach, the closer the algorithm frequency response to the origin, the faster those frequencies will be corrected, improving the convergence rate. Furthermore, in the case the algorithm frequency response gets out of the stability circle, one can detect which frequencies will be unstable, in order to attenuate them.

The SISO ILC algorithm has been implemented in high-fidelity simulations of a mechanical press in Matlab/Simulink, in which precise force reference

tracking has been obtained. Furthermore, a performance and stability comparison with other ILC design methods has been provided. The results have shown that the proposed algorithm obtains faster convergence than other ILC algorithms while guaranteeing stability.

On the other hand, regarding hydraulic presses, a new control approach to improve and automate the position control has been proposed. The traditional hydraulic press position control approaches depend on hand-tuned controllers, which need to be modified every time the press operating conditions change. To that end, we have proposed a multiple-input-multiple-output (MIMO) ILC algorithm, so the hydraulic press operation can be automated and the position control improved.

The MIMO ILC design follows the same design procedure as the one proposed for mechanical presses, nonetheless, it has been extended to the MIMO case. In order to analyze the convergence rate and stability of a MIMO ILC algorithm, no graphical approaches have been found in the literature. Therefore, we have proposed a graphical approach based on the analysis of the eigenvalues at all frequencies. This procedure allows us to intuitively analyze the stability and convergence of the MIMO ILC algorithm.

The MIMO ILC algorithm has been tested in a nonlinear high-fidelity simulation of a hydraulic press model in Matlab/Simulink. In the same way as for the mechanical press, the MIMO ILC algorithm performance has been compared to other ILC designs. The comparison shows that the proposed MIMO ILC design yields a faster convergence rate than the other algorithms.

Indeed, one of the major contributions of the proposed model based ILC design, both for the SISO and MIMO cases, is the ability to correct the low frequency components faster than the existing ILC algorithms in the literature. This yields a faster convergence rate and, as a consequence, faster commissioning time of the press, as fewer iterations are needed to obtain the optimal input.

In order to show the technology readiness level, TRL scale, of the proposed SISO and MIMO ILC algorithms, we have presented two industrial scenarios to test the controllers. First, both algorithms have been implemented in a hydraulic test rig, in which real case scenarios occurring in the working operation of industrial presses have been tested. These scenarios include faulty actuators, modeling mismatches, and flow leakages. The proposed ILC has successfully performed under the faulty and unfavorable scenarios, obtaining a good position and force control.

Second, the SISO ILC algorithm has been implemented in an actual industrial press controller, a Siemens Simotion D445-2, and proved under a Digital Twin platform. This has allowed us to validate the SISO ILC in an operational environment, in which the results correspond to those obtained under simulation.

All in all, as the main conclusion, it can be stated that through this thesis we have encountered different control problems in mechanical and hydraulic presses, which have been solved via ILC. Furthermore, a new design method has been proposed for the ILC design, which outperforms the existing ILC algorithms in the literature. The proposed ILC designs have been validated under relevant operational environments.

7.2 Future Work

During this thesis the following new challenges have been identified that could be suitable for future works:

From a practical point of view, it would be worth researching the possibility of not starting the learning from scratch every time the reference changes. Although we have obtained a fast convergence rate, designing an ILC algorithm that uses past learning experience to adapt to the new reference would reduce considerably the iterations required. There have been several works studying adaptability of the ILC to non-repetitive references, see [93, 94, 95], however none of them make use of the already learned input to improve the learning, they just focus on providing stability towards non-repetitive uncertainties.

In line with the above, the current ILC algorithm design is limited to the press operator to wait until the algorithm has learned the optimal input with which the specifications are satisfied. A more attractive ILC implementation would be to be able to specify the number of iterations required to reach the optimal input. In this way, the press operator could impose a maximum number of iterations in which to learn. This approach would require to know and analyze the system learning dynamics beforehand. Perhaps the graphical frequency analysis proposed in this thesis could be of use.

A major step up in the technology readiness level of the proposed ILC algorithm would be to test it in an actual system. This step forward would not require a lot of effort, as the ILC algorithm is already implemented and validated in a real press control system hardware, as shown in Chapter 6.

This control hardware will only have to be installed in a real press, as a plug and play controller, to finish its validation.

A

ILC error propagation

In this appendix, from the ILC block diagram shown in Fig. 3.1, the error propagation equation which yields the stability condition of a MIMO ILC algorithm will be derived.

We will put the time notation aside, and the Laplace transform of the signals will be used. For instance, $\boldsymbol{R}(s)$ and $\boldsymbol{Y}(s)$, are the Laplace transforms of the $\boldsymbol{r}(t)$ and $\boldsymbol{y}(t)$ vector signals, respectively.

From Fig. 3.1, the input-output relationship is as follows:

$$\boldsymbol{Y}_j(s) = \boldsymbol{G}(s)(\boldsymbol{C}(s)\boldsymbol{E}_j(s) + \boldsymbol{U}_j(s)), \tag{A.1}$$

where $\boldsymbol{Y}_j(s)$, $\boldsymbol{U}_j(s)$, $\boldsymbol{G}(s)$, $\boldsymbol{C}(s)$ and $\boldsymbol{E}_j(s)$, are the system output, the system input, the controller and the error, respectively.

Substituting $\boldsymbol{Y}_j(s) = \boldsymbol{R}(s) - \boldsymbol{E}_j(s)$ and isolating the error, we obtain:

$$\boldsymbol{E}_j(s) = \boldsymbol{S}(s)\boldsymbol{R}(s) - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{U}_j(s), \tag{A.2}$$

where $\boldsymbol{S}(s) = (I + \boldsymbol{G}(s)\boldsymbol{C}(s))^{-1}$ is the system sensitivity function.

From Fig. 3.1, the input at the iteration $j+1$ is defined as:

$$\boldsymbol{U}_{j+1}(s) = \boldsymbol{U}_j(s) + \boldsymbol{L}(s)\boldsymbol{E}_j(s). \tag{A.3}$$

Evaluating (A.2) at $j+1$, and using (A.3), we obtain:

$$\begin{aligned}\boldsymbol{E}_{j+1}(s) &= \boldsymbol{S}(s)\boldsymbol{R}(s) - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{U}_{j+1}(s) = \boldsymbol{S}(s)\boldsymbol{R}(s) - \boldsymbol{S}(s)\boldsymbol{G}(s)(\boldsymbol{U}_j(s) + \boldsymbol{L}(s)\boldsymbol{E}_j(s)) \\ &= \underbrace{\boldsymbol{S}(s)\boldsymbol{R}(s) - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{U}_j(s)}_{\boldsymbol{E}_j(s)} - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{L}(s)\boldsymbol{E}_j(s)\end{aligned} . \tag{A.4}$$

Rearranging the terms in (A.4), we obtain the relationship of the error propagation from iteration to iteration, which has been already shown in [96, 55]:

$$\boldsymbol{E}_{j+1}(s) = (I - \boldsymbol{S}(s)\boldsymbol{G}(s)\boldsymbol{L}(s))\boldsymbol{E}_j(s). \tag{A.5}$$

B

Zero-phase filtering algorithm

Zero-phase filtering (ZPF) is applied in the model inverse design to avoid any phase lag and to filter the undesired high frequency dynamics. From the ILC block diagram shown in Fig. 3.1, the signal out of the ILC scheme is summed to the feedback controller signal. In our case, the controller signals are the proportional valve spool position input and the pump swash angle, which have a physical limitation, the spool position must not exceed the range [-1, 0] and the pump swash angle [0,+1]. Therefore, the signal obtained from the ZPF must not exceed said ranges, to that end, an anti-windup has been included in the filtering process.

The general recursive formulation for discrete-time filters in z-transform holds as follows:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_M z^{-M}}{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_J z^{-J}}. \tag{B.1}$$

In (3.10) a fourth-order filter has been included in the model inverse design, to carry out ZPF the filter is divided into two second-order low-pass filters. Therefore, we set $M = J = 2$ and the resulting linear differential equation is:

$$\begin{aligned} y(n) = &\frac{b_0}{a_0} x(n) + \frac{b_1}{a_0} x(n-1) + \frac{b_2}{a_0} x(n-2) \\ &- \frac{a_1}{a_0} y(n-1) - \frac{a_2}{a_0} y(n-2), \end{aligned} \tag{B.2}$$

where n is a nonnegative integer.

The static gain of the filters is given as:

$$K_s = \frac{\sum_{i=1}^{M} b_i}{\sum_{i=1}^{J} a_i}. \tag{B.3}$$

The filter output is given by (B.2). However, as explained above, the ZPF output must not exceed the valve and pump limitations, else the actuators inputs will saturate.

If the output of the filter exceeds the ranges of interest, the anti-windup is applied to the recursive filter for the last J outputs and M inputs. In this way, we avoid the integration of the error for the recursive outputs and inputs, increasing the performance of the filter.

The resulting filter algorithm including the anti-windup is shown in Algorithm 1.

Algorithm 1 Recursive Filtering with Anti-windup

input: **b** (numerator coefficients vector), **a** (denominator coefficients vector), **u** (input vector), y_{max} (upper saturation), y_{min} (lower saturation), K_s static gain

for $k = 1 \rightarrow length(\mathbf{u})$ **do**
  $y(k) = \frac{b_0}{a_0}x(k) + \frac{b_1}{a_0}x(k-1) + \frac{b_2}{a_0}x(k-2) - \frac{a_1}{a_0}y(k-1) - \frac{a_2}{a_0}y(k-2)$ ▷ Filter the inputs signal sample by sample using (B.2)
  if $y(k) \leq y_{min}$ **then**
    for $i = 1 \rightarrow length(a)$ **do**
      $y(i) = y_{min}$ ▷ Saturate previous outputs
    end for
    for $i = 1 \rightarrow length(\mathbf{b})$ **do**
      $u(i) = y_{min}(i) * \frac{1}{K_s}$ ▷ Saturate previous inputs by multiplying the minimum value by the inverse of the static gain
    end for
  end if
  if $y(k) \geq y_{max}$ **then**
    for $i = 1 \rightarrow length(\mathbf{a})$ **do**
      $y(i) = y_{max}$ ▷ Saturate previous outputs
    end for
    for $i = 1 \rightarrow length(\mathbf{b})$ **do**
      $u(i) = y_{max}(i) * \frac{1}{K_s}$ ▷ Saturate previous inputs by multiplying the maximum value by the inverse of the static gain
    end for
  end if
end for
return: **y** ▷ The entire filtered signal is returned

C

System parameters

C.0.1 Linearized System Parameters

The complete expressions of the terms introduced in the linearized state-space model, shown in (5.12), are shown:

$$
\begin{aligned}
A_{13} &= \frac{A_A}{m} \\
A_{14} &= -\frac{A_B}{m} \\
A_{31} &= -\frac{A_1}{(V_\mathrm{A} + A_A \bar{x})\beta} \\
A_{41} &= \frac{A_2}{(V_\mathrm{B} + A_B(l - \bar{x}))\beta} \\
A_{44} &= \frac{K_\mathrm{v}(\bar{y}_\mathrm{v}) q_{ref}}{(V_\mathrm{B} + A_B(l - \bar{x}))\beta} \frac{1}{2\sqrt{\bar{P}_B / \Delta P_{ref}}} \\
B_{31} &= \frac{\frac{q_N}{\omega_N}\omega}{(V_\mathrm{A} + A_A \bar{x})\beta} \\
B_{42} &= \frac{\sqrt{\bar{P}_B / \Delta P_{ref}}}{(V_\mathrm{B} + A_B(l - \bar{x}))\beta} \dot{K}_\mathrm{v}(\bar{y}_\mathrm{v}) q_{ref}
\end{aligned}
\tag{C.1}
$$

C.0.2 Nominal values for Simulink hydraulic circuit parameters

The system parameters appearing in the terms introduced in the linearized state-space model, shown in (5.12), are shown:

- Cylinder moving mass: $m = 26500$ kg.
- Cylinder piston-side chamber area: $A_A = 0.16$ m^2.
- Cylinder rod-side chamber area: $A_B = 0.02$ m^2.

- Piston-side chamber dead volume: $V_1 = 0.35$ m^3.
- Rod-side chamber dead volume: $V_2 = 0.14$ m^3.
- Cylinder stroke length: $l = 1.2$ m.
- Hydraulic compressibility: $\beta = 1.23 \cdot 10^{-4}$ 1/bar.
- Pump nominal flow rate: $q_N = 0.0088$ m^3/s.
- Shaft rotational speed: $\omega = 150$ rad/s.
- Nominal shaft rotational speed: $\omega_N = 138.23$ rad/s.
- Auxiliary chamber operating point: $\bar{P}_B = 180$ bar.
- Hydraulic conductance operating point: $K_{\mathrm{v}}(\bar{y}_{\mathrm{v}})$ m^3/s.

C.0.3 Hydraulic test rig parameters

The hydraulic rig system parameters are shown:

- Cylinder main chamber area: $A_A = 0.000804$ m^2.
- Cylinder auxiliary chamber area: $A_B = 0.000424$ m^2.
- Cylinder stroke length: $l = 0.2$ m.
- Valve nominal pressure: $\Delta P_{ref} = 35$ bar.
- Valve nominal flow rate $q_{ref} = 1.667 \cdot 10^{-4}$ m^3/s.
- Hydraulic compressibility: $\beta = 1.23 \cdot 10^{-4}$ 1/bar.
- Nominal shaft rotational speed: $\omega_N = 1000$ rad/s.

C.0.4 Test rig hydraulic representation

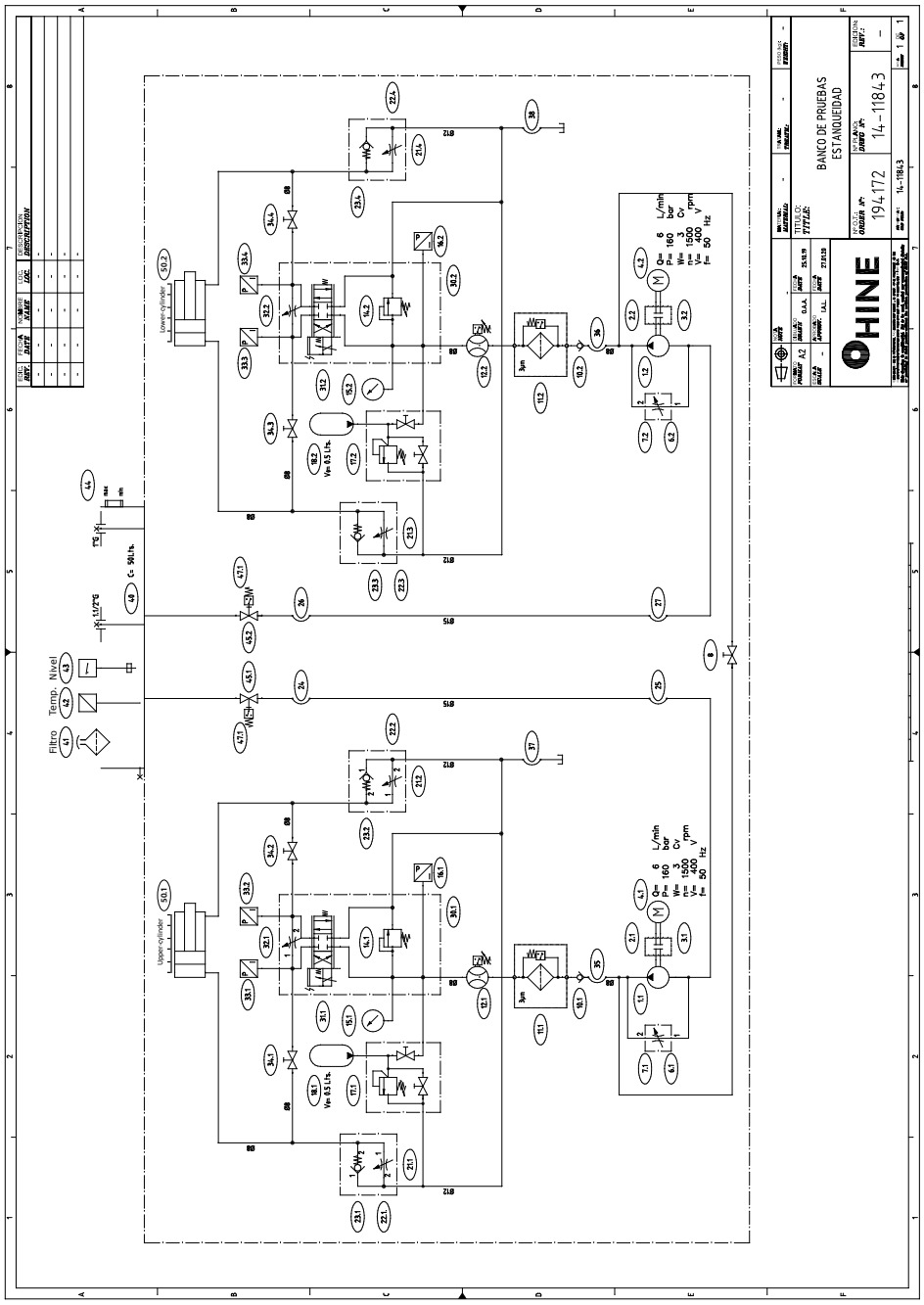


Fig. C.1: Test rig hydraulic circuit.

D

Hydraulic Compressibility Test

In the design of the learning filter for the SISO and MIMO ILC algorithms, the hydraulic compressibility, β, plays a fundamental role, as it gives us information of the rate at which the pressure is increasing. The default value used in the literature for the compressibility is $\beta = 1.23 \cdot 10^{-4}$ 1/bar, as it was shown in [97, 98]. This value has been used both in the Matlab/Simulink models parametrization and in the ILC learning gain design, therefore, both designs are equivalent and no model difference exists.

However, the default hydraulic oil compressibility value may differ from that in the hydraulic test rig. This deviation can affect the ILC implementation in relation to response time, stability, and performance. To that end, we conduct a test to measure the compressibility of the hydraulic fluid.

During the test, only the lower-cylinder is used. The stopcock 34.3, see Fig. C.1, is closed so the oil in the piston-side chamber is enclosed. The lower-cylinder proportional valve is opened completely, moving the spool to the right, connecting port P with port B, and port A with port T.

With this arrangement, we can supply pressure to the rod-side chamber with the pump and analyze how much the oil in the piston-side chamber compresses. We cannot use the pressure sensor 33.3 to see the pressure variation in the piston-side chamber as the stopcock 34.3 is closed. However, we can use the pressure sensor 33.4 and the relationship between the cylinder areas to obtain a good estimation of the piston-side pressure.

Known the pressure variation in the piston-side chamber and the piston displacement from the position sensor, we can make use of the isothermal compressibility equation shown in (2.8), to obtain a value of the compressibility. Isolating for β, we obtain:

$$\beta = \frac{\Delta x A_A}{V_T \Delta P}, \tag{D.1}$$

where A_A is the cushion piston-side chamber area, and V_T is the total oil volume from the cylinder piston-side chamber to the stopcock 34.3.

The pipes provide a flexibility to absorb the thermal expansion, which yields a volume increase inside the pipes during the test operation. This volume increase calculation is deemed too complex and out of the scope of this study. Therefore, both β and V_T are combined and instead of obtaining a value for β, we obtain a value for βV_T.

In Fig. D.1, the lower-cylinder rod-side chamber pressure and the piston position are shown. At $t \approx 2.5$s pressure is applied from the pump and the piston-side chamber pressure reaches 156 bar. At this moment, there exists a slight displacement in the cylinder piston from $x = 63.47$ mm to $x = 62.32$ mm, which yields $\Delta x = 1.149$ mm. These values are obtained with an average value of 200 data points, due to the existing noise in the position signal.

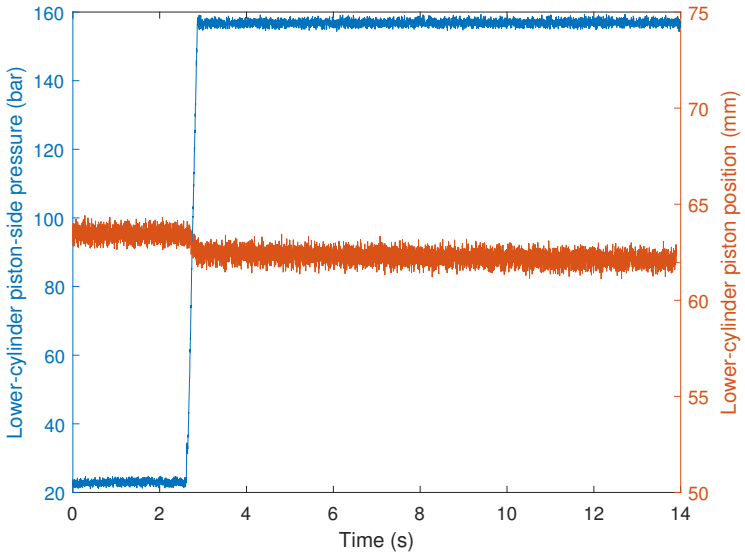


Fig. D.1: Lower-cylinder piston position and rod-side chamber pressure.

The pressure in the rod-side chamber suffers a variation of $\Delta P_B = 134.01$ bar. From the hydraulic test rig parameters shown in Appendix C.0.3, we can obtain the pressure variation in the piston-side chamber:

$$\Delta P_A = \frac{\Delta P_B A_B}{A_A} = \frac{134.01 \cdot 4.24 \cdot 10^{-4}}{8.04 \cdot 10^{-4}} = 70.67 \text{ bar.} \tag{D.2}$$

Solving (D.1) for βV_T, we obtain the following:

$$\beta V_t = \frac{1.15 \cdot 10^{-3} \cdot 8.04 \cdot 10^{-4}}{70.67} = 1.31 \cdot 10^{-8} \text{ m}^3/\text{bar}. \tag{D.3}$$

Therefore, $1.31 \cdot 10^{-8}$ m^3/bar is the real value of the compressibility and the total volume in the hydraulic test rig. We compare it to the theoretical value used for the design of the learning gain. In theory, we assume the pipes are rigid, therefore we can obtain an approximation of the total oil volume from the piston-side chamber to the stopcock 34.3. With V_P as the pipes volume, we obtain:

$$V_T = A_A x + V_p = 8.04 \cdot 10^{-4} \cdot 63.47 \cdot 10^{-3} + 8.54 \cdot 10^{-5} = 1.36 \cdot 10^{-4} \text{ m}^3. \tag{D.4}$$

Therefore, the theoretical value for βV_T is as follows:

$$\beta V_T = 1.23 \cdot 10^{-4} \cdot 1.36 \cdot 10^{-4} = 1.67 \cdot 10^{-8} \text{ m}^3/\text{bar}. \tag{D.5}$$

We can see that the real value for βV_T is a 22% smaller than the theoretical value. This means that in the hydraulic test rig, the volume change of the hydraulic oil as a response to a pressure change will be less.

Bibliography

[1] Hyo Sung Ahn, Yang Quan Chen, and Kevin L. Moore. "Iterative learning control: Brief survey and categorization". In: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 37.6 (2007), pp. 1099–1121 (cit. on pp. 2, 33, 72).

[2] Herbert Merritt. *Hydraulic control systems*. John Wiley & Sons, 1967 (cit. on pp. 9, 10, 13).

[3] Thomas E Marlin. "Process control". In: *Chemical Engineering Series, McGraw-Hill International Editions: New York* (1995) (cit. on p. 12).

[4] Yuzhong Wang, Qibing Jin, and Ridong Zhang. "Improved fuzzy PID controller design using predictive functional control structure". In: *ISA Transactions* 71 (2017), pp. 354–363 (cit. on p. 13).

[5] Khaled Eltag, Muhammad Shamrooz Aslamx, and Rizwan Ullah. "Dynamic Stability Enhancement Using Fuzzy PID Control Technology for Power System". In: *International Journal of Control, Automation and Systems* 17.1 (2019), pp. 234–242 (cit. on p. 13).

[6] Makoto Iwasaki and N Matusi. "Robust speed control of IM with torque feedforward control". In: *IEEE Transactions on Industrial Electronics* 40.6 (1993), pp. 553–560 (cit. on p. 14).

[7] Jos Havinga, Ton van den Boogaard, Franz Dallinger, and Pavel Hora. "Feedforward control of sheet bending based on force measurements". In: *Journal of Manufacturing Processes* 31 (2018), pp. 260–272 (cit. on p. 14).

[8] F. Conrad and C.J.D. Jensen. "Design of Hydraulic Force Control Systems with State Estimate Feedback". In: *IFAC Proceedings Volumes* 20.5 (1987), pp. 307–312 (cit. on pp. 14, 15).

[9] Thomas M. Hessburg and Donald G. Krantz. "Feedforward control (based on model inversion) and system performance prediction using high-fidelity nonlinear dynamic hydraulic system modeling". In: *IEEE Conference on Control Applications - Proceedings* (1997), pp. 57–62 (cit. on p. 15).

[10]Jinsong Zhao, Gang Shen, Weidong Zhu, Chifu Yang, and Jing Yao. “Robust force control with a feed-forward inverse model controller for electro-hydraulic control loading systems of flight simulators”. In: *Mechatronics* 38 (2016), pp. 42–53 (cit. on p. 15).

[11]J. Pan, G. L. Shi, and X. M. Zhu. “Force tracking control for an electro-hydraulic actuator based on an intelligent feed forward compensator”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 224.4 (2010), pp. 837–849 (cit. on p. 15).

[12]Young Rae Ko and Tae Hyoung Kim. “Feedforward plus feedback control of an electro-hydraulic valve system using a proportional control valve”. In: *Actuators* 9.2 (2020), pp. 1–14 (cit. on p. 15).

[13]Gorazd Karer, Gašper Mušič, Igor Škrjanc, and Borut Zupančič. “Feedforward control of a class of hybrid systems using an inverse model”. In: *Mathematics and Computers in Simulation* 82.3 (2011), pp. 414–427 (cit. on p. 15).

[14]Kamaruzaman Jusoff, MF Rahmat, S Md Rozali, and N Abdul Wahab. “Modeling and Controller Design of an Electro-Hydraulic Actuator System”. In: *American Journal of Applied Sciences* 7.8 (2010), pp. 1100–1108 (cit. on p. 17).

[15]Bin Yao, Fanping Bu, John Reedy, and George T.C. Chiu. “Adaptive robust motion control of single-rod hydraulic actuators: Theory and experiments”. In: *IEEE/ASME Transactions on Mechatronics* 5.1 (2000), pp. 79–91 (cit. on p. 17).

[16]Jan Komsta, Nils van Oijen, and Peter Antoszkiewicz. “Integral sliding mode compensator for load pressure control of die-cushion cylinder drive”. In: *Control Engineering Practice* 21.5 (2013), pp. 708–718 (cit. on p. 17).

[17]Davide Cristofori and Andrea Vacca. “The modeling of electrohydraulic proportional valves”. In: *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME* 134.2 (2012), pp. 1–13 (cit. on p. 17).

[18]Md Abu Ayub Siddique, Wan Soo Kim, Yeon Soo Kim, et al. “Effects of temperatures and viscosity of the hydraulic oils on the proportional valve for a rice transplanter based on PID control algorithm”. In: *Agriculture (Switzerland)* 10.3 (2020) (cit. on p. 17).

[19]Bjorn Wittenmark, Johan Nilsson, and Martin Torngren. “Timing problems in real-time control systems”. In: *Proceedings of the American Control Conference* 3.May (1995), pp. 2000–2004 (cit. on p. 18).

[20]Muhammad Ali Al-Marhoun. “The Coefficient of Isothermal Compressibility of Black Oils”. In: *Proceedings of the Middle East Oil Show* 13.10 (2003), pp. 173–179 (cit. on p. 20).

[21]Donald F Specht. “A general regression neural network”. In: *IEEE transactions on neural networks* 2.6 (1991), pp. 568–576 (cit. on p. 23).

[22]Matti Maltamo and Annika Kangas. “Methods based on k-nearest neighbor regression in the prediction of basal area diameter distribution”. In: *Canadian Journal of Forest Research* 28.8 (1998), pp. 1107–1115 (cit. on p. 23).

[23]Theodore B. Trafalis and Huseyin Ince. “Support vector machine for regression and applications to financial forecasting”. In: *Proceedings of the International Joint Conference on Neural Networks* 6.May 2016 (2000), pp. 348–353 (cit. on p. 23).

[24]Carl Edward Rasmussen. “Gaussian processes in machine learning”. In: *Summer school on machine learning*. Springer, 2003, pp. 63–71 (cit. on p. 23).

[25]Antonio Visioli. “A new design for a PID plus feedforward controller”. In: *Journal of Process Control* 14.4 (2004), pp. 457–463 (cit. on p. 26).

[26]Marc Peter Deisenroth and Carl Edward Rasmussen. “PILCO: A model-based and data-efficient approach to policy search”. In: *Proceedings of the 28th International Conference on Machine Learning, ICML 2011* (2011), pp. 465–472 (cit. on p. 29).

[27]Sergey Levine and Pieter Abbeel. “Learning Dynamic Manipulation Skills under Unknown Dynamics with Guided Policy Search”. In: *Advances in Neural Information Processing Systems 27* (2014), pp. 1–3. arXiv: `arXiv:1501.05611v1` (cit. on p. 29).

[28]Masaru Uchiyama. “Formation of high speed motion pattern of mechanical arm by trial”. In: *Transactions of the Society of Instrumentation and Control Engineers* 19 (1978), pp. 709–712 (cit. on p. 31).

[29]Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. “Bettering operation of Robots by learning”. In: *Journal of Robotic Systems* 1.2 (1984), pp. 123–140 (cit. on p. 31).

[30]Suguru Arimoto. “Learning control theory for robotic motion”. In: *International Journal of Adaptive Control and Signal Processing* 4.6 (1990), pp. 543–564 (cit. on pp. 31, 33, 45, 87).

[31]Samer S. Saab. “Stochastic P-type/D-type iterative learning control algorithms”. In: *International Journal of Control* 76.2 (2003), pp. 139–148 (cit. on pp. 33, 34).

[32]Haluk Elci, Richard W Longman, Minh Q Phan, Jer-Nan Juang, and Roberto Ugoletti. “Simple learning control made practical by zero-phase filtering: Applications to robotics”. In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 49.6 (2002), pp. 753–767 (cit. on p. 33).

[33]Tingting Meng, Jiangang Li, Depeng Zheng, and Zexiang Li. “The design of iterative learning control scheme for CNC machine tools”. In: *2016 IEEE International Conference on Information and Automation, IEEE ICIA 2016* August (2017), pp. 665–670 (cit. on p. 33).

[34]Kasper Vinther, Vikas Chandan, and Andrew G Alleyne. “Learning / Repetitive Control for Building Systems with Nearly Periodic Disturbances”. In: *European Control Conference (ECC)* (2013), pp. 1198–1203 (cit. on pp. 33, 82).

[35]Yang Liu, Li Li, Xiaofeng Yang, and Jiubin Tan. “Enhanced kalman-filtering iterative learning control with application to a wafer scanner”. In: *Information Sciences* 541 (2020), pp. 152–165 (cit. on p. 33).

[36]Danwei Wang, Yongqiang Ye, and Bin Zhang. *Practical Iterative Learning Control with Frequency Domain Design and Sampled Data Implementation*. Ilc. 2014, pp. 539–548 (cit. on pp. 33, 35).

[37]Longman Richard W. “Iterative learning control and repetitive control for enginnering practice”. In: *International journal of control* 73 (2010), pp. 930–954 (cit. on pp. 33, 41).

[38]Mikael Norrlöf. *Iterative Learning Control Analysis, Design, and Experiments*. 2000 (cit. on p. 33).

[39]Wei Zhou, Miao Yu, and De Qing Huang. “A high-order internal model based iterative learning control scheme for discrete linear time-varying systems”. In: *International Journal of Automation and Computing* 12.3 (2015), pp. 330–336 (cit. on p. 33).

[40]DOUGLAS BRISTOW, MARINA THARAYIL, and ANDREW ALLEYNE. “A Learning-Based Method for High-Performance Tracking Control”. In: *Ieee Control Systems Magazine* 1066.033X/06 (2006), pp. 96–114 (cit. on pp. 33, 35, 36, 82, 87).

[41]Chih-Keng Chen and Wei-Cheng Zeng. “The iterative learning control for the position tracking of the hydraulic cylinder”. In: *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing* 46.2 (2003), pp. 720–726 (cit. on p. 33).

[42]Shou Kun Wang, Jun Zheng Wang, and Jiang Bo Zhao. “Application of PD-type iterative learning control in hydraulically driven 6-DOF parallel platform”. In: *Transactions of the Institute of Measurement and Control* 35.5 (2013), pp. 683–691 (cit. on p. 33).

[43]Lingjun Li, Uwe Poms, and Thomas Thurner. “Accurate position control of a servo-hydraulic test cylinder by iterative learning control technique”. In: *Proceedings - UKSim-AMSS 8th European Modelling Symposium on Computer Modelling and Simulation, EMS 2014* (2014), pp. 297–302 (cit. on p. 34).

[44]P. H. Gøytil, M. R. Hansen, and G. Hovland. “Iterative learning applied to hydraulic pressure control”. In: *Modeling, Identification and Control* 39.1 (2018), pp. 1–14 (cit. on pp. 34, 41).

[45]James Ratcliffe, Lize Van Duinkerken, Paul Lewin, et al. “Fast norm-optimal iterative learning control for industrial applications”. In: *Proceedings of the American Control Conference* 3.July (2005), pp. 1951–1956 (cit. on p. 34).

[46]Dong Wei and Razvan Panaitescu. “An implementation of iterative learning control in industrial production machines”. In: *4th IEEE Conference on Automation Science and Engineering, CASE 2008* (2008), pp. 472–477 (cit. on p. 34).

[47]T. J. Harte, J. Hätönen, and D. H. Owens. “Discrete-time inverse model-based iterative learning control: Stability, monotonicity and robustness”. In: *International Journal of Control* 78.8 (2005), pp. 577–586 (cit. on pp. 36, 41).

[48]David H. Owens, Chris T. Freeman, and Bing Chu. “Multivariable norm optimal iterative learning control with auxiliary optimisation”. In: *International Journal of Control* 86.6 (2013), pp. 1026–1045 (cit. on p. 36).

[49]D. H. Owens and J. Hätönen. “Iterative learning control - An optimization paradigm”. In: *Annual Reviews in Control* 29.1 (2005), pp. 57–70 (cit. on p. 36).

[50]Svante Gunnarsson and Mikael Norrlöf. “On the design of ILC algorithms using optimization”. In: *Automatica* 37.12 (2001), pp. 2011–2016 (cit. on p. 36).

[51]Jay H. Lee, Kwang S. Lee, and Won C. Kim. “Model-based iterative learning control with a quadratic criterion for time-varying linear systems”. In: *Automatica* 36.5 (2000), pp. 641–657 (cit. on p. 36).

[52]Jayati Ghosh and Brad Paden. “Pseudo-Inverse Based Iterative Learning Control for Linear Nonminimum Phase Plants with Unmodeled Dynamics”. In: *Journal of Dynamic Systems, Measurement, and Control* 126.3 (2004), p. 661 (cit. on p. 36).

[53]Santosh Devasia, Degang Chen, and Brad Paden. “Nonlinear inversion-based output tracking”. In: *IEEE Transactions on Automatic Control* 41.7 (1996), pp. 930–942 (cit. on p. 36).

[54]Jurgen van Zundert and Tom Oomen. “On inversion-based approaches for feedforward and ILC”. In: *Mechatronics* 50.November 2016 (2018), pp. 282–291 (cit. on p. 36).

[55]Nard Strijbosch, Tom Oomen, and Lennart Blanken. “Frequency domain design of iterative learning control and repetitive control for complex motion systems”. In: *Samcon* 2 (2018), pp. 1–2 (cit. on pp. 36, 37, 140).

[56]Benjamin T. Fine, Sandipan Mishra, and Masayoshi Tomizuka. “Model inverse based iterative learning control using finite impulse response approximations”. In: *Proceedings of the American Control Conference* July (2009), pp. 931–936 (cit. on pp. 36, 37, 39, 45, 52, 55, 79).

[57]Je Sung Yeon, Jong Hyeon Park, Seung Woo Son, and Sang Hun Lee. “Model-based iterative learning control for industrial robot manipulators”. In: *Proceedings of the 2009 IEEE International Conference on Automation and Logistics, ICAL 2009* August (2009), pp. 24–28 (cit. on pp. 37, 45, 52, 56, 79).

[58]K. S. Lee, S. H. Bang, and K. S. Chang. “Feedback-assisted iterative learning control based on an inverse process model”. In: *Journal of Process Control* 4.2 (1994), pp. 77–89 (cit. on pp. 37, 45, 52, 56, 79).

[59]M Norrlöf and S Gunnarsson. “A model based Iterative Learning Control method applied to an industrial robot”. In: *Conference on Computer Science and Systems Engineering* October (2000), pp. 73–78 (cit. on pp. 37, 45, 52, 56, 79).

[60]Mojtaba Haghi, Yusheng Yao, Dip Goswami, and Kees Goossens. “Parallel implementation of iterative learning controllers on multi-core platforms”. In: *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 1704–1709 (cit. on p. 39).

[61]Chung Li Jiang. “Sufficient condition for the asymptotic stability of interval matrices”. In: *International Journal of Control* 46.5 (1987), pp. 1803–1810 (cit. on p. 41).

[62]P. Batra. “On necessary conditions for real robust Schur-stability”. In: *IEEE Transactions on Automatic Control* 48.2 (2003), pp. 259–261 (cit. on p. 41).

[63]Hyo Sung Ahn, Kevin L. Moore, and Yang Quan Chen. “Schur stability radius bounds for robust iterative learning controller design”. In: *Proceedings of the American Control Conference* 1 (2005), pp. 178–183 (cit. on p. 41).

[64]Hyo Sung Ahn, Kevin L. Moore, and Yang Quan Chen. “Stability analysis of discrete-time iterative learning control systems with interval uncertainty”. In: *Automatica* 43.5 (2007), pp. 892–902 (cit. on p. 41).

[65]Koji Kinosita, Takuya Sogo, and Norihiko Adachi. “Iterative learning control using adjoint systems and stable inversion”. In: *Asian Journal of Control* 4.1 (2002), pp. 60–67 (cit. on p. 41).

[66]Jorge Rodriguez-Guerra, Carlos Calleja, Iker Elorza, et al. “A Methodology for Real-Time HiL Validation of Hydraulic-Press Controllers Based on Novel Modeling Techniques”. In: *IEEE Access* 7 (2019), pp. 110541–110553 (cit. on pp. 42, 96, 123).

[67]Joerg Grabbel and Monika Ivantysynova. “An Investigation of Swash Plate Control Concepts for Displacement Controlled Actuators”. In: *International Journal of Fluid Power* 6.2 (2005), pp. 19–36 (cit. on p. 58).

[68]Pornjit Pratumsuwan, Santi Hutamarn, and Watcharin Po-Ngaen. “Energy saving in electro-hydraulic system using a MIMO fuzzy controller”. In: *Advanced Materials Research* 622 (2013), pp. 75–79 (cit. on p. 69).

[69]Pornjit Pratumsuwan and Aphaiwong Junchangpood. “Force and position control in the electro-hydraulic system by using a MIMO fuzzy controller”. In: *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 2013, pp. 1462–1467 (cit. on p. 69).

[70]Yongseob Lim, Ravinder Venugopal, and A. Galip Ulsoy. “Multi-input multi-output (MIMO) modeling and control for stamping”. In: *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME* 132.4 (2010), pp. 1–12 (cit. on p. 69).

[71]Arne Jansson, Petter Krus, and Jan-Ove Plamberg. “Decoupling of response and pressure level in a hydraulic actuator”. In: *The 4th Bath International Fluid Power Workshop* January 1992 (1991), pp. 1–16 (cit. on p. 69).

[72]M Elfving and J O Palmberg. “Distributed control of pressure controlled fluid power actuators-decoupled chamber pressure controlled cylinder”. In: *Fluid Power Systems, Ninth Bath International Fluid Power Workshop*. 1996 (cit. on p. 70).

[73]Simon Schröders, Clemens C. Maier, Wolfgang Ebner, Alexander Fidlin, and Christoph Hametner. “Two-degree-of-freedom MIMO control for hydraulic servo-systems with switching properties”. In: *Control Engineering Practice* 95.December 2019 (2020), p. 104246 (cit. on p. 70).

[74]Xiao Wang, Dacheng Cong, Zhidong Yang, Shengjie Xu, and Junwei Han. “Iterative learning control with complex conjugate gradient optimization algorithm for multiaxial road durability test rig”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 233.7 (2019), pp. 2349–2360 (cit. on p. 70).

[75]Xiao Wang, Dacheng Cong, Zhidong Yang, Shengjie Xu, and Junwei Han. “Modified Quasi-Newton Optimization Algorithm-Based Iterative Learning Control for Multi-Axial Road Durability Test Rig”. In: *IEEE Access* 7 (2019), pp. 31286–31296 (cit. on p. 70).

[76]Tino Müller and Christian Endisch. “An estimation based iterative learning control approach for cross-coupled vehicle test systems”. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM* (2017), pp. 1375–1381 (cit. on p. 70).

[77]Shangcheng Chen and Christopher T. Freeman. “Decentralised Collaborative and Formation Iterative Learning Control for Multi-Agent Systems”. In: *Proceedings of the American Control Conference* 2020-July (2020), pp. 2729–2734 (cit. on p. 70).

[78]Deokkyun Yoon, Xinyi Ge, and Chinedum E. Okwudire. “Optimal inversion-based iterative learning control for overactuated systems”. In: *IEEE Transactions on Control Systems Technology* 28.5 (2020), pp. 1948–1955 (cit. on p. 70).

[79]Mircea Bogdan Radac, Radu Emil Precup, and Emil M. Petriu. “Model-Free Primitive-Based Iterative Learning Control Approach to Trajectory Tracking of MIMO Systems With Experimental Validation”. In: *IEEE Transactions on Neural Networks and Learning Systems* 26.11 (2015), pp. 2925–2938 (cit. on p. 70).

[80]Ye Ren, Zhongsheng Hou, Isik Ilber Sirmatel, and Nikolas Geroliminis. “Data driven model free adaptive iterative learning perimeter control for large-scale urban road networks”. In: *Transportation Research Part C: Emerging Technologies* 115.February (2020), p. 102618 (cit. on p. 70).

[81]Chems Eddine Boudjedir, Mohamed Bouri, and Djamel Boukhetala. “Model-Free Iterative Learning Control with Nonrepetitive Trajectories for Second-Order MIMO Nonlinear Systems - Application to a Delta Robot”. In: *IEEE Transactions on Industrial Electronics* 68.8 (2021), pp. 7433–7443 (cit. on p. 70).

[82]Notker Amann, David H. Owens, and Eric Rogers. “Predictive optimal iterative learning control”. In: *International Journal of Control* 69.2 (1998), pp. 203–226 (cit. on p. 70).

[83]Lennart Blanken and Tom Oomen. “Multivariable iterative learning control design procedures: From decentralized to centralized, illustrated on an industrial printer”. In: *IEEE Transactions on Control Systems Technology* 28.4 (2019), pp. 1534–1541 (cit. on pp. 70, 71).

[84]Jian-Xin Xu and Ying Tan. *Linear and nonlinear iterative learning control*. Vol. 291. Springer, 2003 (cit. on p. 72).

[85]Hongfeng Tao, Xiaohui Li, Wojciech Paszke, Vladimir Stojanovic, and Huizhong Yang. “Robust PD-type iterative learning control for discrete systems with multiple time-delays subjected to polytopic uncertainty and restricted frequency-domain”. In: *Multidimensional Systems and Signal Processing* 32.2 (2021), pp. 671–692 (cit. on p. 72).

[86]Željko Šitum. “Force and position control of a hydraulic press”. In: *Krmiljenje Hidravlične Stiskalnice* 17.4 (2011), pp. 314–320 (cit. on p. 76).

[87]Jian ming Zheng, Sheng dun Zhao, and Shu guo Wei. “Application of self-tuning fuzzy PID controller for a SRM direct drive volume control hydraulic press”. In: *Control Engineering Practice* 17.12 (2009), pp. 1398–1404 (cit. on p. 76).

[88]Mihály Héder. “From NASA to EU: The evolution of the TRL scale in Public Sector Innovation”. In: *Innovation Journal* 22.2 (2017), pp. 1–23 (cit. on p. 95).

[89]European Commission. “HORIZON 2020 WORK PROGRAMME 2014 - 2015. (5i) Leadership in enabling and industrial technologies - Information and Communication Technologies. Revised {European Commission Decision C (2014) 4995 of 22 July 2014}”. In: 2015.July 2014 (2014) (cit. on pp. 95, 96).

[90]Gunnar Prytz. “A performance analysis of EtherCAT and PROFINET IRT”. In: *2008 IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE, 2008, pp. 408–415 (cit. on p. 123).

[91]Paolo Ferrari, Alessandra Flammini, Daniele Marioli, and Andrea Taroni. “Experimental evaluation of PROFINET performance”. In: *IEEE International Workshop on Factory Communication Systems, 2004. Proceedings.* IEEE, 2004, pp. 331–334 (cit. on p. 123).

[92]Siemens Ag and I D T Mc. “SIMOTION Technology Packages System Functions”. In: (2010), pp. 1–1512 (cit. on p. 124).

[93]Deyuan Meng and Kevin L. Moore. “Robust iterative learning control for non-repetitive uncertain systems”. In: *IEEE Transactions on Automatic Control* 62.2 (2017), pp. 907–913 (cit. on p. 137).

[94]Xu Jin. “Iterative learning control for non-repetitive trajectory tracking of robot manipulators with joint position constraints and actuator faults”. In: *International Journal of Adaptive Control and Signal Processing* 31.6 (2017), pp. 859–875 (cit. on p. 137).

[95]Xu Jin. “Fault Tolerant Nonrepetitive Trajectory Tracking for MIMO Output Constrained Nonlinear Systems Using Iterative Learning Control”. In: *IEEE Transactions on Cybernetics* 49.8 (2019), pp. 3180–3190 (cit. on p. 137).

[96] Sandipan Mishra and Masayoshi Tomizuka. “Projection-based iterative learning control for wafer scanner systems”. In: *IEEE/ASME Transactions on Mechatronics* 14.3 (2009), pp. 388–393 (cit. on p. 140).

[97] Amine Meziou, Taoufik Wassar, Majdi Chaari, Matthew A. Franchek, and Reza Tafreshi. “Model-based design and analysis of a subsea high integrity pressure protection system (HIPPS)”. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM* 2016-Septe (2016), pp. 1216–1221 (cit. on p. 147).

[98] Oscar Pena and Michael J Leamy. “A SELF-CONTAINED ARCHITECTURE FOR ENERGY RECOVERY IN HYDRAULIC”. In: (2017), pp. 1–6 (cit. on p. 147).

List of Figures

List of Tables