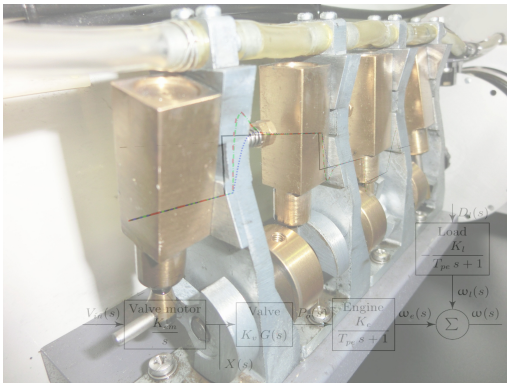
Escola Superior Náutica Infante D. Henrique
Departamento de Engenharia Marítima



Engine Speed Control

João Pedro Sousa Dias Pardelhas

Thesis to obtain the Master of Science Degree in

Engenharia de Máquinas Marítimas

Supervisor: Professor Luís Mendonça
Co-Supervisor: Professor Miguel Silva

December 2015

ESCOLA SUPERIOR NÁUTICA INFANTE D.HENRIQUE

Abstract

Departamento de Engenharia Marítima

Master of Science in Engenharia de Máquinas Marítimas

Engine speed control

by João Pardelhas

The purpose of this dissertation is to implement and compare speed controllers on the CE 107 pneumatic engine apparatus. The controllers are implemented on a developed hardware that includes a microcontroller and a dedicated motor drive circuit capable to interface with the DC motors. A software was created to plot the data in real time.

Firstly, the system static characteristics and transfer functions are identified so that controllers can be tuned and simulations can be done.

Secondly, feedback, cascade, internal model, feedforward and model free controllers are tuned, simulated and implemented. A change on a tuning rule and a 2DOF control structure that aims to separate the setpoint from the load disturbance responses are proposed, simulated and implemented.

Finally, the controllers performances are compared and conclusions are drawn.

Keywords: PID controller, speed control, 2DOF control, cascade control, internal model control, model-free control.

ESCOLA SUPERIOR NÁUTICA INFANTE D.HENRIQUE

Resumo

Departamento de Engenharia Marítima

Mestrado em Engenharia de Máquinas Marítimas

Engine speed control

por João Pardelhas

Os objetivos desta dissertação são a implementação e comparação de controladores de velocidade no motor pneumático CE 107. Os controladores são implementados num *hardware* desenvolvido que inclui um microcontrolador e um circuito dedicado para efetuar a interação com os motores DC. Foi criado um *software* para representar gráficamente os parâmetros em tempo real.

Inicialmente são identificadas os parâmetros e funções de transferência dos vários elementos do sistema. Esta identificação tem como principais objectivos a obtenção de modelos para ajustar os controladores e efectuar as simulações. O motor pneumático e a válvula foram aproximados como sendo processos de primeira ordem.

Seguidamente são ajustados e simulados controladores de velocidade em atraso, em controlo em cascata, *internal model*, em avanço e *model free*, sendo que o desempenho de cada simulação é quantificado de acordo com quatro índices de desempenho. A regra para ajustar controladores em atraso que teve o melhor desempenho foi modificada e introduzida num controlador com dois graus de liberdade onde se verificou uma melhoria geral de acordo com todos os índices de desempenho.

O controlador *internal model* possui um único parâmetro de ajuste. Idealmente, estes controladores possuem um desempenho óptimo quando este parâmetro é igual ao atraso do sistema. Este fenómeno não se verificou nas simulações, possivelmente devido à discretização do controlador. A estrutura do controlador em avanço permite uma resposta completamente diferente para mudanças de referência e atenuação de disturbios. Uma alteração a esta estrutura é proposta através da introdução de um preditor Smith. A utilização de estruturas ao avanço para separar a resposta a alterações de referência e atenuação de perturbações foi alterada, através da introdução de um preditor Smith, onde se observou uma ligeira melhoria na resposta a alterações de referência e uma degradação significativa na atenuação de perturbações. Aproximar o atraso no modelo do processo como sendo puro ou segundo Padé não tem um impacto significativo no desempenho. A simulação da estrutura *model free* apresenta

uma resposta superior e é capaz de eliminar o erro quando a mudança de referência ocorre em rampa. As simulações mais promissoras foram implementadas no motor pneumático.

Finalmente, é efetuada uma comparação experimental entre os controladores. A resposta do anel de controlo em cascata é melhor do que a resposta do controlador em atraso. O controlador *internal model* deu origem a uma resposta fraca, o que foi interpretado como um sinal de que o modelo do processo é impreciso. O controlador em avanço não foi capaz de superar o desempenho do controlo em cascata. O controlador *model free* obteve a melhor resposta.

Palavras-chave: controlador PID, regulador de velocidade, controlo com 2 graus de liberdade, controlo em cascata, *internal model control*, *model-free control*.

Acknowledgements

Foremost, I would like to express my deep gratitude to my supervisors Professor Luís Mendonça and Professor Miguel Silva for always keeping their doors open. It was their patient support, good nature and expertise that made this work possible.

I wish to thank all members of DEM that, one way or another, contributed to this work, specially Professor José Costa for introducing me to Model-Free Control and Professor Jorge Trindade for all the help with the algorithms.

The hard work to develop the hardware was substantially lessened by my dear friend João Lisboa.

The completion of this work would have been all the more difficult if it weren't for the help of João Marques, Inês Oliveira and Pedro Teodoro. Thank you for keeping your sense of humour when I lost mine.

To all my family and friends, thank you for carrying me on your shoulders with nothing but love in your heart.

Contents

List of Figures

List of Tables

Abbreviations

2DOF	**2** **D**egrees of **F**reedom
AMIGO	**A**pproximated **M**-constrained **I**ntegral **G**ain **O**ptimization
FB	**F**eed**B**ack
FF	**F**eed**F**orward
FOPTD	**F**irst **O**rder **P**lus **T**ime **D**elay
CC	**C**ascade **C**ontrol
CC-SP	**C**ascade **C**ontrol with a **S**mith **P**redictor
IAE	**I**ntegral of **A**bsolute **E**rror
IMC	**I**nternal **M**odel **C**ontroller
ISE	**I**ntegral of **S**quared **E**rror
ITAE	**I**ntegral of **T**ime multiplied **A**bsolute **E**rror
ITSE	**I**ntegral of **T**ime multiplied **S**quared **E**rror
MFC	**M**odel **F**ree **C**ontrol
PI	**P**erformance **I**ndices
PID	**P**roportional, **I**ntegral and **D**erivative
PWM	**P**ulse-**W**idth **M**odulation
SOPTD	**S**econd **O**rder **P**lus **T**ime **D**elay
SP	**S**mith **P**redictor
SPFF	**S**et**P**oint **F**ollowing through **F**eed**F**orward
SPFF-SP	**SPFF** with a **S**mith **P**redictor
VV	**V**itecková and **V**itecek PID tuning rule
VV-M	**M**odified **V**itecková and **V**itecek PID tuning rule

Symbols

D_l	Voltage input load disturbance
G_c	Controller transfer function
G_{ffsp}	Feedforward setpoint controller transfer function
G_l	Load disturbance transfer function
G_p	Process transfer function
G_{pe}	Pneumatic engine transfer function
$G_{p_{est}}$	Real time estimation of G_p
G_{sp}	Adjustable setpoint response transfer function
G_v	Air supply valve transfer function
I	Inertia
$K_{A/D}$	Analog to Digital converter gain
K_c'	Interactive PID controller proportional gain
K_{cs}	Complete system steady state gain
K_d	Controller derivative gain
K_e	Pneumatic engine steady state gain
K_{em}	Electric motor steady state gain
K_i	Controller integral gain
K_l	Load disturbance steady state gain
K_{load}	Load disturbance steady state gain
K_p	Controller proportional gain
K_{pe}	Pneumatic engine steady state gain
K_{pot}	Potentiometer steady state gain
K_{PWM}	PWM gain
K_{ss}	Tachometer steady state gain

K_t	Anti-windup reset gain
K_v	Air supply valve steady state gain
L	Time delay
P_o	Air pressure at valve output
τ_f	Friction torque
τ_l	Load disturbance torque
τ_{pe}	Pneumatic engine torque
T_{ar}	Time average residence
T_{asv}	Air supply valve time constant
T_d	Derivative time
T_i	Integral time
T_s	Sampling period
X	Air supply valve position
α	Dimensionless constant
b	Proportional setpoint weight
c	Derivative setpoint weight
e	Error
iP	Intelligent proportional controller
w	Angular velocity
u	Controller output
u_{max}	Maximum controller output
y	Process output
y_{sp}	Process setpoint

Chapter 1

Introduction

1.1 Motivation

Speed control of rotating machinery is of great importance to the industry. The CE 107 pneumatic engine is an apparatus which aims to aware to some control problems regarding speed control and non-linearities. The apparatus requires an interface that must read sensors and controls the electric motors. The development of a flexible interface, not only for this apparatus but possibly for many others, is important.

There is a wide range of control structures and tuning rules with different levels of complexity and working principles suitable for distinct needs. Even though it is not possible to compare all of them in a generic way, some of the most well known and promising ones can be selected, tested and compared on the pneumatic engine.

1.2 Objectives and contributions

The main objectives and contributions of this work are:

- To develop an economically accessible interface between a computer and the CE107 pneumatic engine. The interface should be flexible enough so that several control structures can be tested. Changing the gains and obtaining data should be easily carried out.
- To implement a stable PID controller algorithm on a microcontroller.
- To obtain the pneumatic engine air and supply valve dynamic model.
- To simulate and implement different control structures and tuning rules.

1.3 Thesis outline

The work is organized in the following way:

- **Chapter 2** is dedicated to a brief historical review of speed control, its importance to control engineering and some speed control architectures are also presented.
- In **Chapter 3** a description of the system and the modelling of its main components is carried out so that controllers can be tuned and simulations can be executed.
- In **Chapter 4** the selected control structures and tuning rules are presented and accompanied by simulations. The results are analysed and their performance is quantified by four performance indices. The best performances are identified. Some modifications are carried out on tuning rules and control structures, showing improvements on the performance.
- In **Chapter 5** the simulations for each control structure that has the most promising performances are implemented on the pneumatic motor and the results are analysed through the same four performance indices.
- In **Chapter 6** the conclusions and future work are presented.

Chapter 2

A review of speed control

In this chapter a brief historical review of speed control and its importance to control engineering is presented. Some of the most well known control structures and tuning rules are identified.

2.1 A brief historical review

Speed control has been under development for over 220 years and had a huge impact on control engineering [1]. It was first introduced by Miekle in 1745 to control the speed of windmills through spring-regulated sails [2]. A major improvement was made in 1788 by James Watt. He adapted Miekle invention and used it to control the steam flow to a steam engine. The Centrifugal Speed Governor, as Watt named it, is schematically represented in figure 2.1 [3]. It consists on a set of two revolving balls attached to a vertical spindle by link arms. The controlling force is the dependency between the weight of the balls and their rotational speed. Increasing the rotational speed results in higher centrifugal force, which in turn will lead to a reduction of the supplied steam, and with it the speed decreases. It combined the sensed angular speed with the valve actuation to change the steal flow supplied to the engine.

In this early stage the control loop was prone to instability and faced several challenges such as friction, sluggish response and backlash. The integral action was introduced, by the Pèrrier brothers in 1790 [1], through the use of an hydraulic mechanism. In 1845 the Siemens brothers implemented integral action using differential gears and derivative action based on the inertia of a coupled wheel [1].

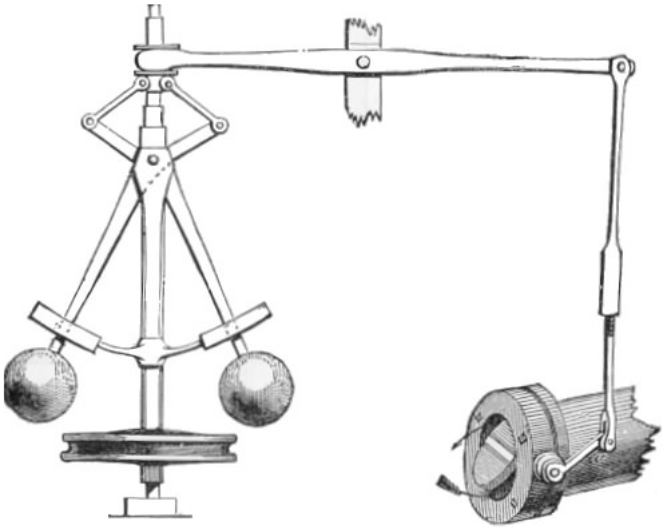


FIGURE 2.1: Centrifugal Speed Governor [3].

In turn, the first stability discussion on a closed-loop system dates to 1840 by G.B. Airy during the development of a speed controller for a rotating telescope [4]. However, it was Maxwell, in 1868, that developed and described the mathematical analysis of the governor stability in the classic seminal paper "On Governors" [5]. It is considered a central paper of the early days of control theory and marks the starting point of automatic control as a new engineering branch. He also discovered that the stability of a closed-loop system was related to the roots of the characteristic equation [6]. Later on, Maxwell delivered his work to Routh. Stodola, who also encountered problems with the steam turbine speed regulation, suggested the stability importance to Hurwitz. Routh and Hurwitz developed, independently, a stability criteria using different methods. The combined results are known as the Routh-Hurwitz criteria [7].

Nowadays, the internal combustion engine governor does not exclusively control the amount of fuel supplied to the engine but also on the injection timing [8], fuel-air ratio [9] and, in some cases, the type of fuel supplied to the engine [10]. Furthermore, modern speed controllers must be designed to maintain the air pollution within a certain legal range, provide good drive-ability and efficiency as high as possible, at the expense of additional control variables [11]. As a result, the controllers complexity increased.

2.2 Review of speed control architectures

Most controllers are Proportional and Integral (PI) or Proportional, Integral and Derivative (PID). Even more sophisticated control structures are, most of the times, proportional and integral or PID controllers at their lowest level [12]. There are a number of reasons for their popularity such as the long list of successful operations in many applications over time, their easy understanding and implementation without the need of advanced hardware or computational algorithms and their ability to provide satisfactory control for most industrial processes [13].

PID control increasingly attracted the community interest for the past years, [14], but improper tuning is still quite common [15] and factory tuning is widely and inappropriately applied [15]. PID controllers are extensively used in distinct industries [16–18] and are a crucial part of the control loop.

Even though PID controllers only have three parameters, finding their suitable values can be quite difficult [19]. Tuning the controller is one of the most important aspects, if not the most important one. The first step is to understand the plant static and dynamic behaviours. Afterwords, tuning can be achieved by two different approaches.

The heuristic approach consists on choosing some initial controller parameters. The initial parameters can be set up randomly or based on the previous experience obtained with similar processes [20]. Observing the system changes, the parameters can be progressively adjusted until satisfactory performance is achieved.

The analytical approach consists on developing mathematical models that describe the system behaviour and then applying a tuning method.

Regardless of the technique used to achieve the tuning, there is always a primary objective. Most common ones are:

- Attenuation of load disturbances.
- To reduce the sensitivity of measurement to noise.
- To augment the robustness to model uncertainty.
- To obtain satisfactory setpoint tracking.

A complete response analysis for a simple feedback control loop requires six transfer functions. These transfer functions are known as the Gang of Six [21]. It is not possible to have a top performance on all of the mentioned objectives and a balance must be reached that is unique to the system [22].

Furthermore, to partially achieve the proposed objectives some restrictions must be taken into consideration such as system dynamics, non-linearities and internal and external disturbances. Extremely fast responses can not always be translated into better performance. A typical example is the fuel valve of an internal combustion engine. Very fast responses will lead to premature fuel valve worn-out and higher average fuel consumption without any guarantee of a significant positive impact on the engine performance [23].

The sensitivity to measurement noise is of special concern on controllers with derivative action because it can be a performance drawback [24]. First order filters are usually implemented but higher order ones have proved to have a better performance [25]. Special care must be taken when a PID is tuned and a filter is introduced afterwords because the controller zeros may have changed position. An iterative calculation is proposed to attenuate the location shift [26].

The robustness to model uncertainty and process changes is quantified by the gain and the phase margins of the PID controller [27, 28]. It is possible to tune a PID to provide the desired gain and phase margins [29, 30].

The deviation from the setpoint is fundamental for a feedback controller. Without it, no change in the control action will be generated. However, when the disturbances can be measured it is possible to start acting on the plant before a bigger impact is made. If the controller and actuator can act much faster than the process, the measurable disturbance can even be nullified while maintaining an error free process [31].

However, the use of feedforward requires a good mathematical model of the process and the disturbance. Process uncertainty should be taken into consideration [32]. The feedforward controller is not a source of instability because it is not affected by the process output, however it is more sensitive to modelling errors. An additional advantage of using both feedback and feedforward is the reduction of the importance of modelling errors because they can be treated as the effect of a load disturbance

[33].

Feedforward can also be used to improve setpoint response. It was first proposed by Horowitz in 1963 [34] and transferred to the PID technology by Araki in 1984 [35]. Later, more complex structures were introduced [21, 36–38]. A review and comparison can be found in [39].

Most methods that aim to a better segregation of disturbance rejection and changes to setpoint responses require tuning, most of which through trial and error. Complete response detachment is proposed by Åström and Hägglund in [21]. With their feedforward technique the response to setpoint changes can be given by any desirable function, under a certain limited range of conditions, regardless of the tuning of the PID controller. A trade-off between disturbance response and robustness on the feedback controller stills exists.

Tuning rules for PID controllers have been developed over the years [1, 6, 20, 21]. The tuning parameters can be obtained by several methods such as the step response [12, 40–42], frequency response [12, 40, 43, 44], analytical design [21, 26, 31, 45–48], optimization methods [49–51] and pole placement. An excellent compilation and comparison of tuning rules can be found in [52] and [21] and the numerous references therein.

A very well known and noteworthy tuning rule was created by Ziegler and Nichols. The heuristic tuning rule was developed in 1942 through a performance comparison of a batch of transfer functions [40]. The rules are still used nowadays [52]. Simplicity might be the main cause of such longevity.

Two variants of the tuning rule were developed. One based on the step response and the other based on frequency response methods [53]. For each method two parameters can be obtained and, based on this information, the tuning parameters can be computed for different controllers. Initially, a quarter amplitude damping was expected but not assured [53]. The Ziegler-Nichols tuning rules have been improved for several years with the introduction of interesting variations such as no overshoot or faster response with 25 % overshoot as well as the significant refinement of the tuning formula [12].

Aside from the PID internal structure and possible tuning rules the control loop

structure is of high importance. Control loop structures, that can have or not a PID within its core, have been under development. Some of the most notorious ones that are still being improved are the Internal Model Controller (IMC) [54], the Smith Predictor (SP) [55, 56], Fuzzy controllers [57], Linear-Quadratic Regulator [58], Model Predictive Control [59], and Model-Free Controllers (MFC) [60]. Some control loop structures were already used together [61, 62].

Tuning rules based on other control structures are also subject of research. The SIMC, also known as Skogestad IMC or Simple IMC, is a tuning rule developed for a IMC by Skogestad [63]. The only tuning parameter is the closed-loop time constant that allows a simple trade-off between performance and robustness. Axel Holene in [64] expanded the SIMC to more models, including simulations with varying time delay models. He compared the results with a SP tuned by the same rule. The SIMC tuning rules were a rediscovery of the lambda tuning rules that were originated in early computer control [22].

A well known control structure is cascade control. Cascade control can be used when multiple measured signals are used to control a single variable and its main objective is to respond faster to disturbances. The control system can have as many loops as intermediate variables. The variable that is intended to be controlled is in the outer loop, also refereed as the primary loop. The situation where the inner loop, or secondary loop, is a feedback around the actuator is widely used [26, 31]. In order to use cascade control there must be a clear relationship between the primary and secondary variables.

The selected tuning rules for the inner loop vary significantly. It can be a pure proportional or combined with derivative action. The static error in this loop does not have a serious consequence because the outer loop will eliminate the steady-state error of the controlling variable, which has integral action and a high proportional gain. If integral action is used in the inner loop the proportional action in the outer one must be reduced. This is has a considerable performance decay and should be avoided [20, 21, 31, 65].

Extraordinary care must be taken into consideration when implementing a cascade control loop such as the saturation of the inner loop, tracking of the inner control

loop variable to the outer loop in order to avoid windup, changeover of the inner control loop from manual to auto and vice-versa [21, 26, 31].

Chapter 3

System modelling

The main objective of modelling is to capture the process aspects that are relevant for control [66]. The estimation of the process transfer function is a crucial part for most tuning rules and they are needed for simulations [52]. This chapter is dedicated to the modelling of the actuator, the pneumatic engine, the tachometer and the load disturbance.

3.1 Apparatus description

The CE107 pneumatic engine is an apparatus, shown in Figure 3.1, designed by TecQuipment, that models an internal combustion engine, where the speed and mechanical power are controlled by the amount of air supplied, instead of fuel. The dynamics of the CE107 are similar to an ignition compression engine in a dynamometer controlled test bed. It was conceived to reveal problems regarding speed control, especially the control of non-linear systems [67]. The basic principle is to control the position of the air supply valve, which is motorised by a d.c. motor, and so to regulate the engine speed under varying load conditions.

The CE107 apparatus is divided into five main components. The air supply valve, the valve position sensor, the pneumatic engine, the tachometer and the load disturbance. The valve position is determined through a potentiometer. The load on the engine is introduced on the d.c. generator[1] connected to its output. The pneumatic

[1]Actually, it is a d.c. motor because electric power is being supplied to it and not from it. However, considering its purpose and the users manual designation, it is from now on referred as a d.c. generator.

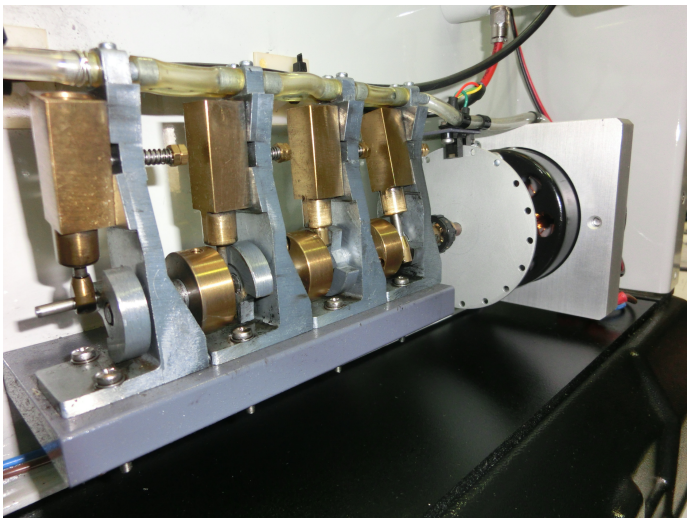


FIGURE 3.1: Engine speed control apparatus.

TABLE 3.1: Main characteristics of the *CE107 pneumatic engine*.

Engine	Four-cylinder industrial air motor, compressed air driven
Cylinder dimensions	Bore 8 mm, stroke 19 mm, swept volume 3.8 cm^3
Supply valve motor	24 V, 500 mA
Speed transducer	Optical sensor (LED/phototransistor)
Speed range	From 800 rpm to 2000 rpm
Air supply pressure	From 5 bar to 10 bar
Air supply flow	From 0 litres/minute to 10 litres/minute

engine has its characteristics presented on table 3.1. The working speed range depends on the lubrication system installed on the pneumatic engine and should not be exceeded [68].

The rotating speed of the apparatus is sensed through the combination of a disk, perforated on the periphery, located on the flywheel, and an optical sensor assembly. The optical sensor assembly is made of a light-emitting diode and a phototransistor. When it detects the holes on the perforated disk it generates an electrical pulse. The frequency at which the holes pass between the LED and the photo transistor is proportional to the engine speed. The signal conditioning circuit converts this frequency into an analogue signal voltage level [67]. The compressed air, supplied by an external air compressor, is filtered within the CE107 unit and its pressure is regulated at the inlet to attenuate air pressure variations while running.

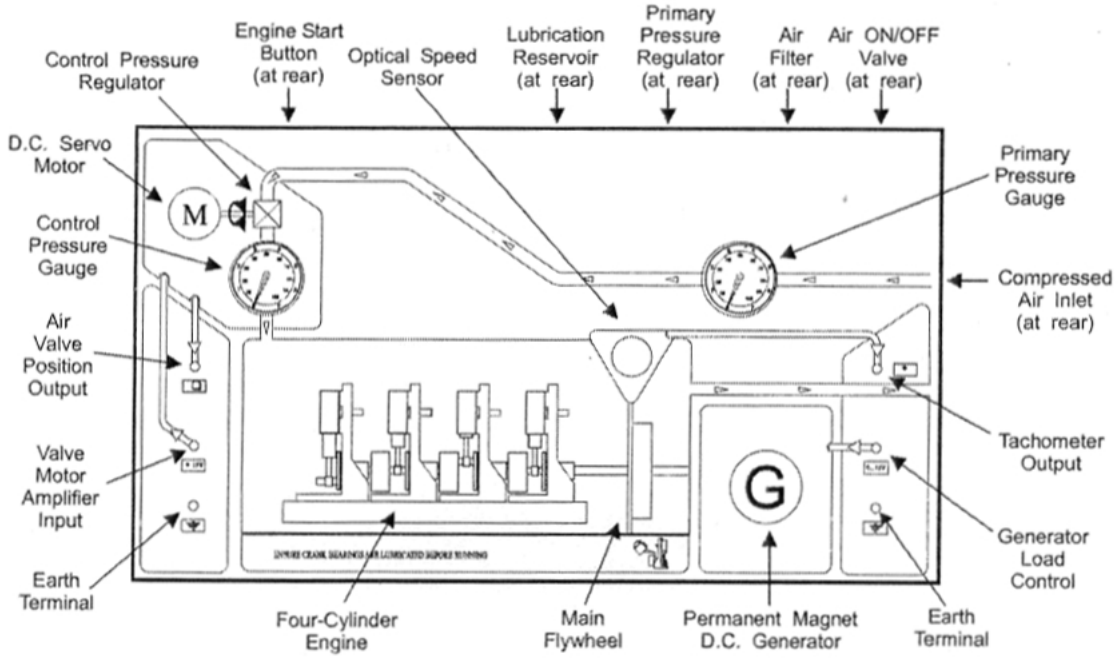


FIGURE 3.2: CE107 front panel layout [67].

The air supply valve position will affect the air pressure at the pneumatic engine inlet. The resistant torque that can be applied to the engine output demands a certain amount of mechanical power from the engine. A change in the resistant torque will result in a change of the engine speed and, after a period of time, the engine will reach a constant speed. Under these conditions, the pneumatic power supplied to the engine used to overcome the resistant torque, the frictional losses and to accelerate the engine flywheel.

When the apparatus was in its original conditions all of the power supplied to drive the system as well as signal conditioning circuits were fully buffered and contained within the base of the CE107 unit, as can be seen in figure 3.2 [67]. The front panel is provided with a schematic functional detail, easy and clear interface with the transducers, d.c. motor responsible for moving the valve and d.c. generator that will introduce a load on the engine.

TecQuipment recommends the digital interface, CE120, or the self-contained analogue and computer based controller, CE122, to interact with the pneumatic engine. Both hardware include a Control Software with editable, pre-made, control experiments. Alternatively, a microcontroller is used to control and send relevant information to a personal computer. Nowadays, microcontrollers have enough computational

resources and interface solutions so that they can be used to control a wide range of processes [69–74] and are available at low prices.

3.2 Components modelling

3.2.1 Air supply valve

The air supply valve is a complex set of components. Its layout is illustrated in figure 3.3 and its block diagram in figure 3.4. It includes an electric motor that will move a shaft, a gear box that transmits the movement from the electric motor shaft to the valve shaft and a potentiometer on the valve shaft, to read the valve position. The valve itself is made of a body, spindle, seat, diaphragm and two springs.

A change in the valve position will cause a change in the springs tension, resulting in variations of the forces F_1 and F_2. This forces are exercised, directly or indirectly, on the valve diaphragm. On the diaphragm an air pressure P_o should balance the forces F_1 and F_2. When the valve opens the force F_2 increases and F_1 decreases. The air flow through the air supply valve to the pneumatic engine increases as well. As a result, the outlet air supply valve pressure P_o, that acts upon the diaphragm's surface, increases. When the spring forces F_1, F_2 and the force resulting from the outlet pressure P_o are in equilibrium the air flow becomes stable. In other words, the valve itself has a mechanical feedback control system and the output air pressure P_o is proportional to valve position $X(s)$ by a gain K_v, i.e.,

$$P_o(s) = K_v X(s)\,. \tag{3.1}$$

The air supply valve is operated by a d.c. electric motor which turns at a rate proportional to the input voltage. The valve opens when the voltage supplied to the electric motor is positive and closes when the voltage is negative. Supplying constant input voltage should result in a continuous change in the valve position, until it reaches its maximum or minimum positions. However, this phenomena is not always observed due to the friction forces present in the air supply valve and also due to the

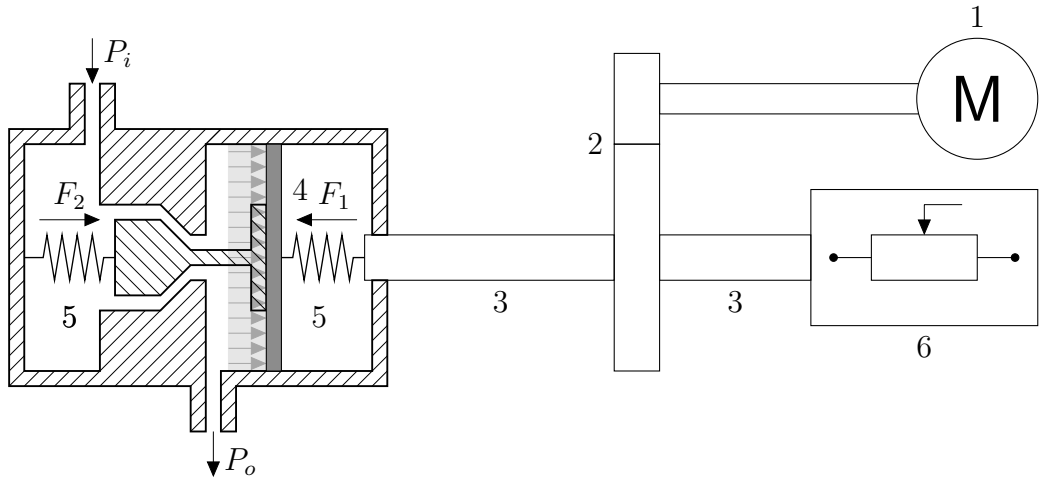


FIGURE 3.3: Air supply valve layout. 1 – D.c. electric motor. 2 – Gear box. 3 – Shaft. 4 – Diaphragm. 5 – Springs. 6 – Potentiometer.

effect of the springs, that are not eliminated by the air pressure $P_o(s)$. As a result, the air supply valve transfer function is approximated as a self-regulating process instead of an integrating process. A mathematical representation by differential equations could not be evaluated and the valve transfer function is given by $K_v\, G_v(s)$.

Two step inputs are carried out on the actuator to observe if the difference between the opening and closing transient states are significant. One step input was carried out with the valve fully open and another one with the valve fully closed. To open the valve a 24 V step input was applied and to close it a -24 V one. The air supply valve responses to step inputs are shown in figure 3.5. The difference between the transient responses is minimal and therefore neglected in this work.

In order to reach a speed of 800 rpm the air supply valve must be at approximately at 47 % of its position, i.e., there is a significant dead-band. The valve working range is significantly narrower than what is represented in figure 3.5. A new step input

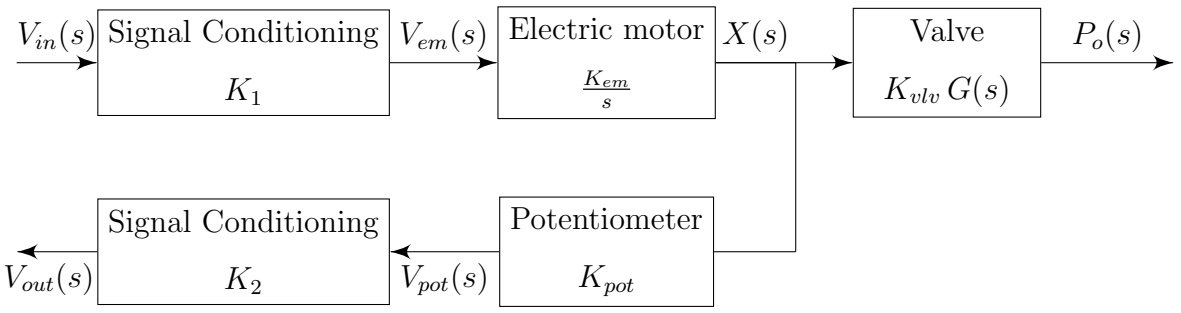


FIGURE 3.4: Air supply valve block diagram.

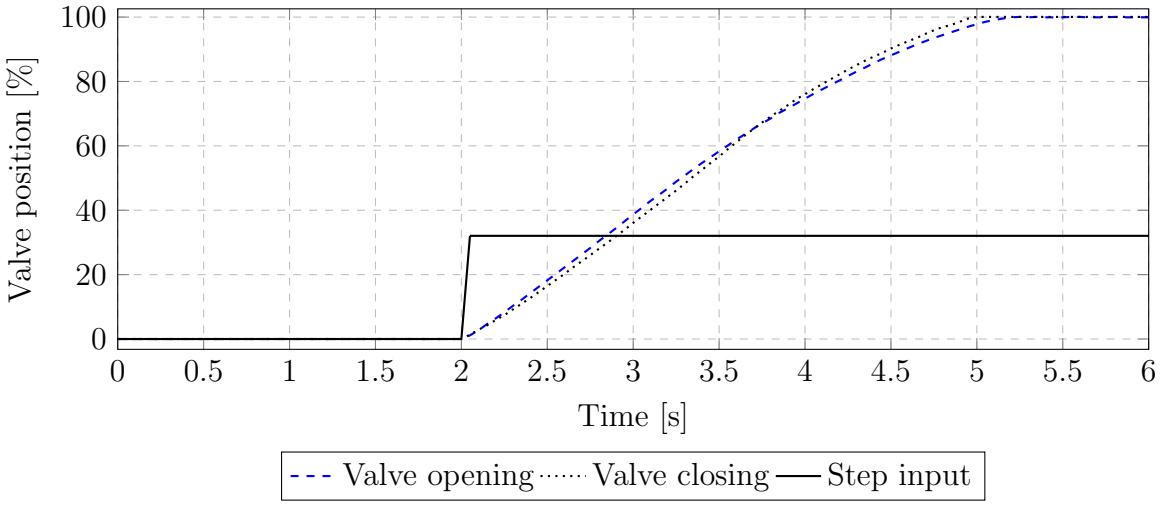


FIGURE 3.5: Actuator responses to step inputs in both directions.

is carried out, but this time with the valve position corresponding to a pneumatic engine speed of 800 rpm. A first order approximation was carried out. The steady state gain is

$$K_v = \frac{\Delta V}{\Delta PWM} = \frac{841 - 490}{255 - 0} = 1.38\,, \tag{3.2}$$

where ΔV is the position change, read at the potentiometer output through the Analog to Digital converter, and ΔPWM is the input Pulse With Modulation signal. The gain was already affected by the signal conditioning circuit. This will be explained with more detail on chapter 4. The time constant was selected as the time it took the process to reach $T = 0.632\,K_v$.

The dead time is smaller than a sampling period. It is assumed that it has the same magnitude, i.e. $L = 0.1$.

The air supply valve transfer function is given by

$$G_v(s) = \frac{X(s)}{V_{in}(s)} = \frac{1.38}{0.4\,s + 1} e^{-0.1s}\,, \tag{3.3}$$

where $X(s)$ is the valve position and $V_{in}(s)$ is the input voltage. The experimental results and the first order approximation are presented in figure 3.6.

Due to the static friction of the actuator there is a minimum supply voltage required before it begins to move. In order to determine the dead-band the actuator must be stopped in an expected working position. Then, the input voltage on the air

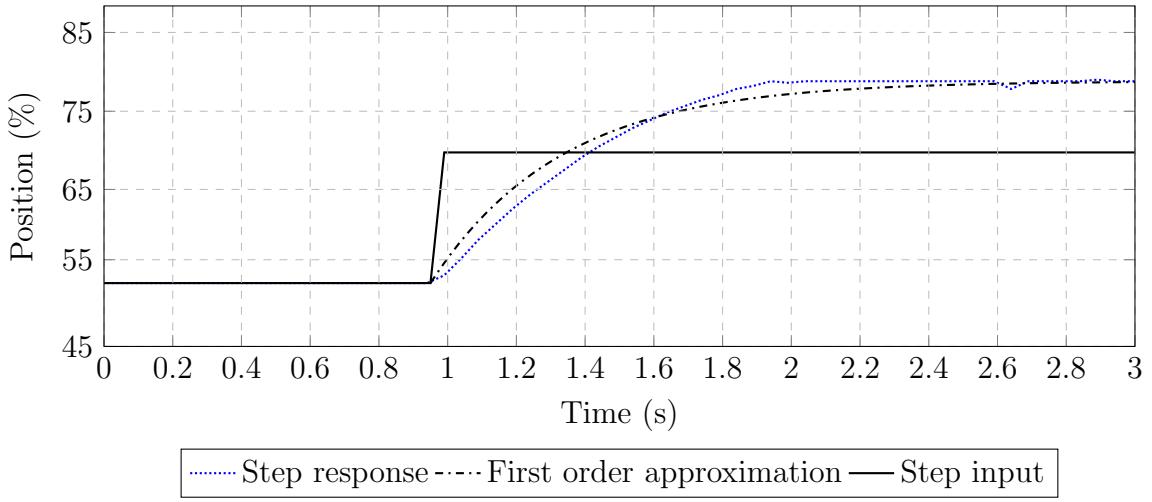


FIGURE 3.6: Actuator response to a step input on a narrow working range.

TABLE 3.2: Actuator opening and closing dead-bands.

	Deadband	
Position	Opening	Closing
70 %	2.3 V	1.7 V

supply valve is slowly and constantly increased until it starts to move in one direction. At that moment, the minimum supply voltage to move the actuator in that direction is known. The other dead-band is determined the same way, but with the polarity of the supply voltage inverted. The valve was manually positioned at 70 % of its position range and subjected to continuously increasing step inputs. Simultaneously, the valve position was registered. The dead-bands are presented in table 3.2. The dead-band to open the valve, is higher than the one to close due to the springs effect.

Another static experiment was carried out. The air supply pressure was set to 5 bar, its normal operating condition, and the air supply valve was opened by hand until the pneumatic engine minimum speed was achieved. Then, the air supply valve was slowly opened until a significant change in the pneumatic engine speed was verified. The air supply valve position and pneumatic engine speed were registered. This procedure was repeated until the maximum speed was achieved. Afterwards, the air supply valve was closed by hand and stopped at the same positions. The

results are represented in figure 3.7. Due to hysteresis is not possible to control the speed solely based on the actuator position, even when no load was applied.

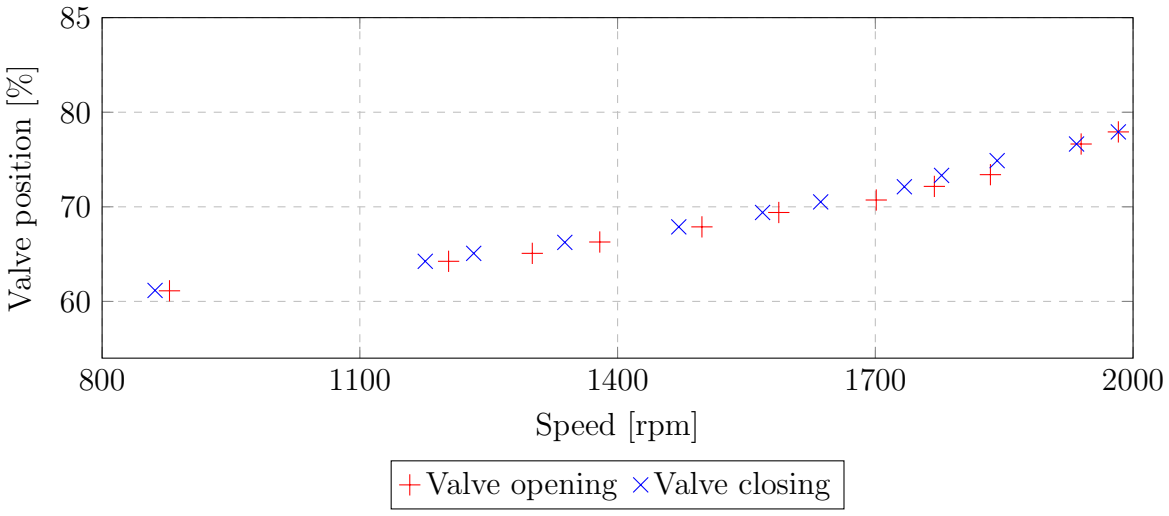


FIGURE 3.7: Static relationship between the valve position and engine speed.

3.2.2 Pneumatic engine

The air supplied by the valve applies a pressure, p_o, on the engine piston crown which is converted into torque, τ_{pe}, at the engine crankshaft. The torque produced by an internal combustion engine cylinder varies significantly over a single crankshaft revolution. However, this is not so significant on the pneumatic engine because the change in the air pressure, that naturally occurs when the inlet and outlet ports open, is small when compared to the pressure changes of the internal combustion engine [67].

Because the engine has four cylinders and a high rotational speed it is assumed that the engine torque, τ_{pe}, is linearly proportional to the air pressure. Hence,

$$\tau_{pe} = K_{pe}\, p_o\,, \tag{3.4}$$

where K_{pe} is the pneumatic engine assembly gain.

The engine torque produced this way is used to overcome frictional losses, τ_f, accelerate the engine flywheel and overcome its inertia, I, and to supply the engine load, τ_l. Assuming the process is linear, if w is the engine speed, applying the Newton's second law to rotational movement

$$I\,\frac{dw}{dt} = \tau_{pe} - \tau_l - \tau_f \tag{3.5}$$

where the term $I\,\frac{dw}{dt}$ is the rate of change of flywheel momentum. The torque consumed to overcome frictional losses, τ_f, is considered to be proportional to the engine speed with a gain of K_f:

$$\tau_f = w\,K_f \tag{3.6}$$

and the load torque, τ_l, is proportional to the voltage applied to the d.c. generator, d_l, connected to the engine shaft

$$\tau_l = K_{load}\,d_l\,, \tag{3.7}$$

where K_{load} is the load gain.

Combining the equations 3.4, 3.5, 3.6 and 3.7

$$I\,\frac{dw}{dt} + w\,K_f = K_{pe}\,p_o - K_{load}\,d_l\,. \tag{3.8}$$

Using the Laplace transform on equation 3.8,

$$W(s) = \frac{K_{pe}}{s\,I + K_f}\,P_o(s) - \frac{K_{load}}{s\,I + K_f}\,D_l(s) = \frac{K_e}{s\,T_{pe} + 1}\,P_o(s) - \frac{K_l}{s\,T_{pe} + 1}\,D_l(s)\,, \tag{3.9}$$

where $K_e = K_{pe}/K_f$, $T_{pe} = I/K_f$ and $K_l = K_{load}/K_f$. The pressure $P_o(s) = K_v\,X(s)$ was already established in subsection 3.2.1, on equation 3.1.

The equation 3.9 represents a first order system with load disturbance. The block diagram of the pneumatic engine assembly is represented in figure 3.8.

The pressure $P_o(s)$ is converted into the torque τ_e. The load disturbance D_l will generate a resistant torque τ_l and the frictional losses a resistant torque τ_f. The combined torques will determine the pneumatic engine speed w. The pneumatic engine speed is converted to voltage $V(s)$ by the tachometer with a gain of K_{ss}.

A step input is used to estimate the parameters of the pneumatic engine transfer function based on a time response approach. The step input is applied on the valve. Precautions must be taken to avoid overspeed. The manufacturer's instructions of

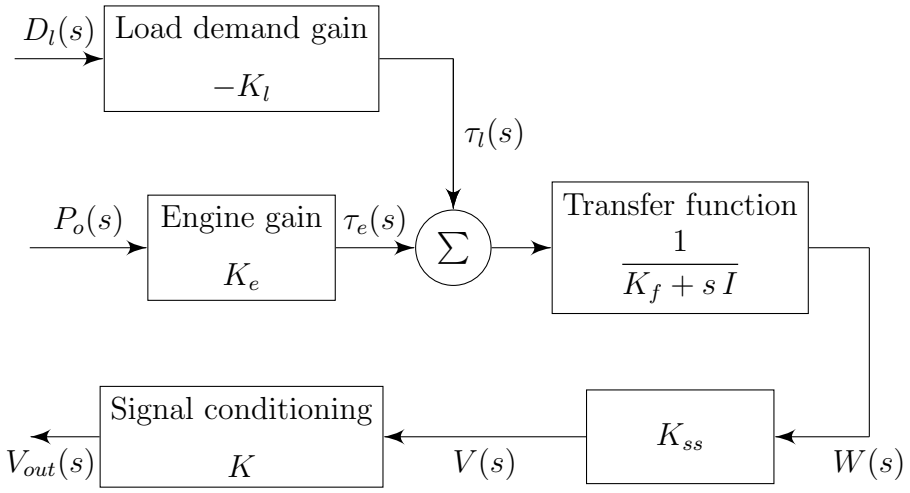


FIGURE 3.8: Pneumatic engine and load block diagram.

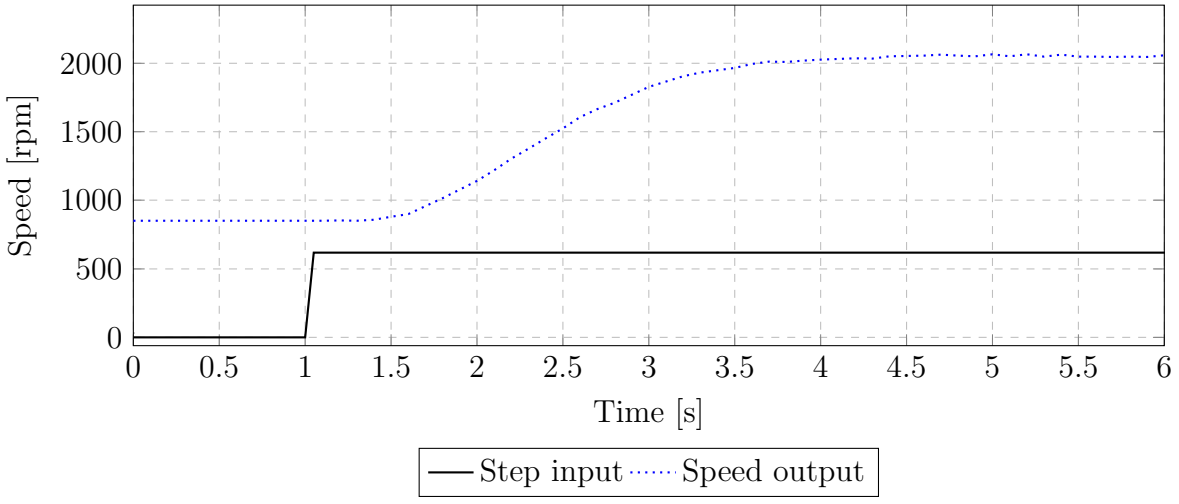


FIGURE 3.9: Pneumatic engine step input response.

the apparatus refer that under this working conditions the maximum pneumatic engine speed is 2000 rpm but it can be temporarily exceeded, as long as proper prelubrication is carried out. The procedure is described in [68]. The air supply valve was continuously opened by hand until a speed of 800 rpm is achieved. A step input was applied and the obtained results are presented in figure 3.9.

A transfer function is estimated from the step response input. The step input is applied on the air supply valve, which is estimated as a first order system. The response to step input of the system includes the air supply valve and the pneumatic engine, which are, when combined, approximated by a second order transfer function.

There are several ways to estimate a second order transfer function [20, 21]. It is

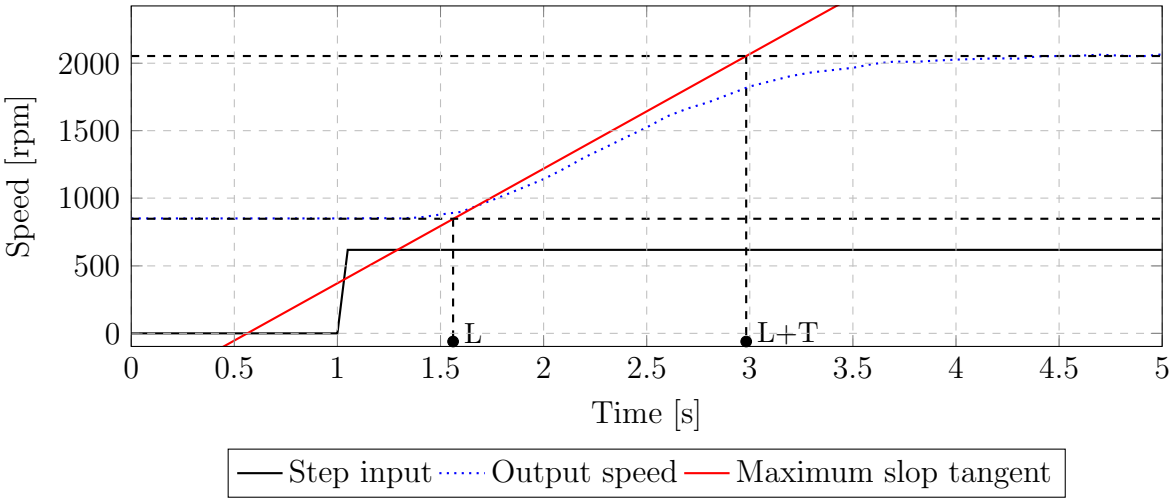


FIGURE 3.10: Step input response and its maximum slope tangent.

assumed that each pole is provided by each main component, i.e., the air supply valve and the pneumatic engine. The complete system is estimated as a Second Order Plus Time Delay (SOPTD) process. A SOPTD transfer function is given by

$$\text{SOPTD} = \frac{K}{(T_1\, s + 1)(T_2\, s + 1)}\, e^{-Ls}\,. \tag{3.10}$$

A SOPTD approximation by the tangent method is proposed by Shinskey, [52], where T_2 is assumed known. If T_2 is the time constant of the first order approximation of the air supply valve transfer function, then $T_2 = 1.4$ seconds. The tangent method consists on finding the maximum slope of the response to a step input and drawing a tangent line to it. This tangent line reaches the minimum and maximum process output. After the response to the step input initiates, the time at which the tangent line intersects with the minimum process output is the dead-time L and the time at which the line intersects the maximum process output is given by $T + L$, like it is presented in figure 3.10.

The tangent intersection with the minimum occurs at 1.52 seconds and with the maximum at 2.99 seconds. The step input was carried out at 1 second so

$$\begin{aligned} L &= 1.52 - 1 = 0.52 \text{ s, and} \\ T &= 2.99 - 1 - 0.52 = 1.47 \text{ s.} \end{aligned} \tag{3.11}$$

TABLE 3.3: Air supply valve plus pneumatic engine SOPTD approximations.

	Transfer function	RMSE
Shinskey method	$\frac{1.95}{(1.47s+1)(0.4s+1)}\,e^{-0.52s}$	0.4376
Method of moments	$\frac{1.95}{(0.6s+1)(0.4s+1)}\,e^{-0.52s}$	0.1202

According to the Shinskey method,

$$\begin{aligned} T_1 &= T = 1.47 \text{ s}, \\ T_2 &= 0.4 \text{ s}, \\ L &= 0.52 \text{ s}. \end{aligned} \tag{3.12}$$

Another method to estimate the unknown pneumatic engine time constant was used. The method of moments can be applied to determine the parameters of the transfer function. The average residence time T_{ar} is the time it takes a process to reach 63.2 % of its gain. If a step input is applied, it can be obtained by

$$T_{ar} = \frac{G'(0)}{G(0)}\,. \tag{3.13}$$

Hence, for a SOPTD,

$$T_{ar} = T_1 + T_2 + L\,. \tag{3.14}$$

Since the step occurs at 1 second, $T_{ar} = 1.52$ s, $T_2 = 0.4$ s and $L = 0.52$ s,

$$T_1 = T_{ar} - T_2 - L = 0.6\,\text{s}. \tag{3.15}$$

The transfer functions are presented in table 3.3. They are SOPTD approximations of the process. The obtained transfer functions differ significantly. The Root Mean Squared Error (RMSE) is used to measure the error between the experimental data

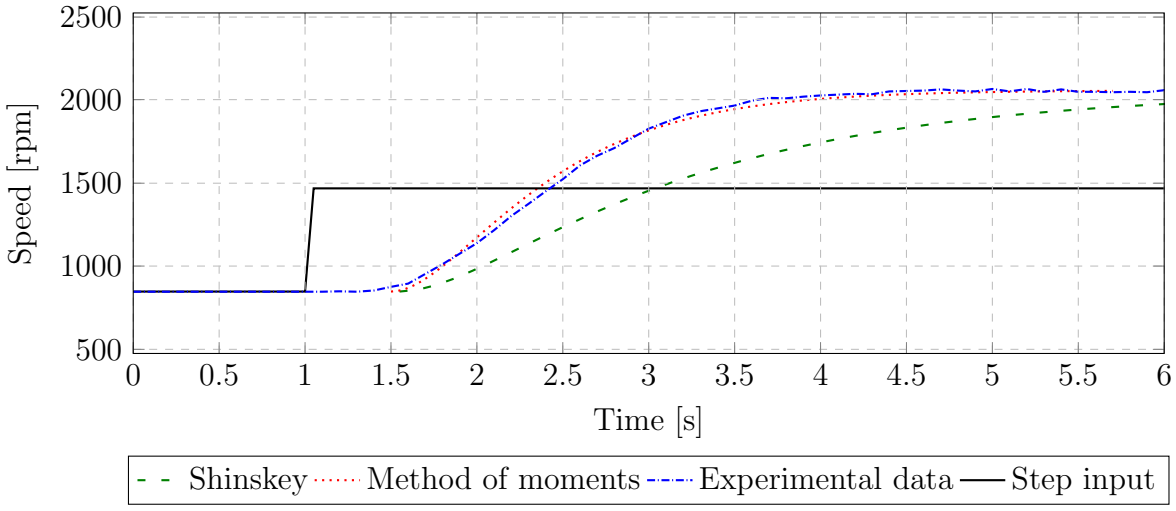


FIGURE 3.11: Step input response and SOPTD approximations.

and the SOPTD approximations. Lower values of RMSE mean an approximation with less error. The RMSE is given by

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{n}(y_t - \hat{y})}{n}} \tag{3.16}$$

where y_t is the experimental data at time t and $\hat{y}$ is the approximation. The RMSE for the SOPTD approximations are presented in table 3.3. The method of moments has the smallest RMSE. The SOPTD approximations and the experimental data are represented in figure 3.11. It can be visually observed that the best approximation is achieved through the method of moments.

Finally, the pneumatic engine transfer function is estimated. The steady state gain of the SOPTD approximation, i.e., the complete system, is

$$K_{cs} = \frac{\Delta A/D}{\Delta PWM} = \frac{802 - 304}{255 - 0} = 1.95\,, \tag{3.17}$$

where $\Delta A/D$ is the voltage change, read at the tachometer output through the Analog to Digital converter and ΔPWM is the Pulse With Modulation input signal. The gain is already affected by the signal conditioning circuit. It will be explained with more detail on chapter 4. The pneumatic engine steady state gain is given by

$$K_e = \frac{K_{cs}}{K_v} = \frac{1.95}{1.38} = 1.41\,. \tag{3.18}$$

The time delay is

$$L = L_1 + L_2 \tag{3.19}$$

where L_1 is the pneumatic engine time delay and L_2 the air supply valve. Therefore,

$$L_1 = L - L_2 = 0.52 - 0.1 = 0.42\,. \tag{3.20}$$

The First Order Plus Time Delay (FOPTD) transfer function of the pneumatic engine becomes

$$G_{pe} = \frac{w(s)}{X(s)} = \frac{1.41}{0.6s + 1} e^{-0.42s}. \tag{3.21}$$

3.2.3 Tachometer

The pneumatic engine tachometer characteristics were determined. The supply air pressure was set to 5 bar and the air supply valve was continuously opened by hand until the minimum speed was reached, i.e., 800 rpm. The pneumatic engine speed was measured with a stroboscopic light. The air supply valve was opened by hand and, after the transient, the tachometer output voltage and frequency of the stroboscopic light were registered. This procedure was repeated for seven different speeds. The relation between the tachometer output voltage and pneumatic engine speed is represented in figure 3.12.

The sensor presents a linear behaviour and no adjustments were carried out[2]. The linear approximation is given by

$$y \simeq 0.00403\, x\,. \tag{3.22}$$

The engine tachometer gain K_{ss} is

$$K_{ss} = 0.00403\, \frac{V(s)}{W(s)} \tag{3.23}$$

where $V(s)$ is the voltage input and $W(s)$ is the speed output.

[2]The apparatus is equipped with three potentiometers, that can be found inside the casing, to re-adjust the sensor characteristics.

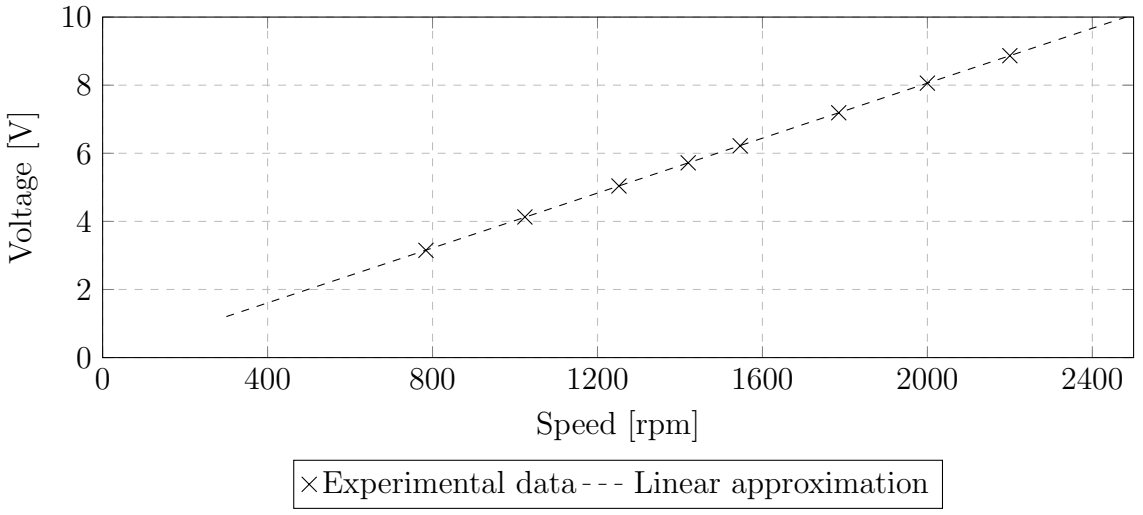


FIGURE 3.12: Pneumatic engine tachometer static characteristic.

3.2.4 Load disturbance

According to equation 3.9 the only difference between the pneumatic engine transfer function and load disturbance is the gain. The load disturbance transfer function could not be determined due to hardware limitations. However, some simulations were carried out where a load disturbance was introduced. Therefore, the load disturbance transfer function is assumed to be

$$G_l(s) = \frac{W(s)}{D_l(s)} = -\frac{1}{0.6s + 1}\, e^{-0.42s}\,. \tag{3.24}$$

The load demand D_l is a sensed disturbance variable that, even though is not used to control the speed, it can supply valuable information to improve the controller performance [21].

3.3 Complete system

The apparatus components that were presented and modelled in this chapter can be combined into to a complete system model. The voltage applied to the electric motor of the air supply valve will change the valve position. The valve consists, among many more internal components, of a spring and a diaphragm that together regulate the air pressure at the valve outlet as function of the valve position.

The air pressure that will act on the piston crown is converted into torque on the crankshaft, which in turn will be consumed to overcome frictional losses, accelerate

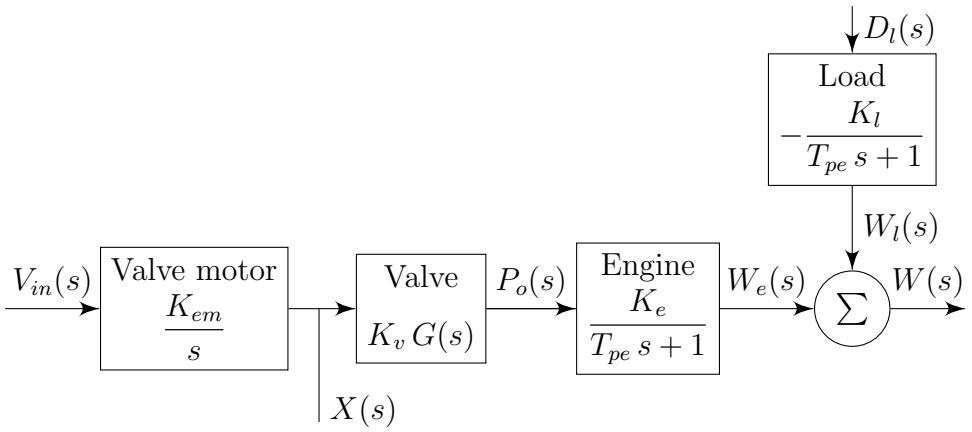


FIGURE 3.13: Complete system block diagram.

the engine flywheel and supply the engine load.

A load disturbance can be applied on the pneumatic engine shaft, producing a resistant torque. The pneumatic engine speed will depend on the power supplied by the air, the friction forces and the load demand. When this parameters remain constant the pneumatic engine speed will be, after a transient, constant. The overall system block diagram is presented in figure 3.13. The gains are already affected by the sensors and signal conditioning circuits.

From the block diagram, and making first order approximations to the air supply valve and electric motor, the combined load and engine function is

$$w(s) = \frac{K_{em}\,K_{vlv}\,K_e}{(T_{pe}\,s+1)(T_v s+1)}\,V(s) - \frac{K_l}{T_{pe}\,s+1}\,D_l(s)\,. \tag{3.25}$$

The complete system transfer function is given by

$$G_{cs} = \frac{W(s)}{V_{in}(s)} = \frac{1.95}{(0.6s+1)(0.4s+1)}\,e^{-0.52s} - \frac{1}{0.6s+1}\,e^{-0.42s}\,. \tag{3.26}$$

Chapter 4

Comparison of different controllers

The main objectives of this chapter consists on simulating and comparing the performance of different controllers and tuning rules, applied to the process models obtained in chapter 3. A setpoint tracking test is proposed and performance indices are used to quantify the performance.

4.1 Controllers

Several control structures and tuning rules have been developed over the centuries. They vary in their working principle and complexity. Based on the models of the air supply valve and pneumatic engine, obtained in chapter 3, simulations can be carried out for different control structures and tuning rules. For each one, a description is presented on its respective section, further in this chapter. The selected control structures are:

- Feedback Control (FC);
- Cascade Control (CC);
- Internal Model Control (IMC);
- Feedback and Feedforward (FF);
- Model-Free Control (MFC).

Before tackling into each control structure, there are some features most of them have in common that are analysed, in this section, such as:

- Signal conditioning;
- Discretization;
- Sampling frequency;
- Deadband compensation;
- Algorithm foundations;
- Anti-windup.

4.1.1 Signal conditioning

The controllers are implemented on the microcontroller and the input/output signals are affected by the signal conditioning circuit steady state gains. If signal condition transient exists, it was already included in the air supply valve and pneumatic engine transfer functions.

Regarding the input signals, i.e., process output signals and microcontroller input ones, they are affected by the respective sensor gain and Analog to Digital (A/D) converter. For the potentiometer, used to read the valve position, the gain is

$$K_{pot} = \frac{\Delta V}{\Delta X} = \frac{5-0}{100-0} = 0.05\,, \tag{4.1}$$

where ΔV is the output voltage range and ΔX is the position range in units of length which perfectly fit the valve stroke. The tachometer steady state gain was obtained in chapter 3. The 10 bit A/D converter steady state gain is

$$K_{A/D} = \frac{\text{Resolution}}{\Delta V} = \frac{2^{10}}{5-0} = 204.8\,, \tag{4.2}$$

where ΔV is the input voltage. Regarding the output signal, i.e., the signal that leaves the microcontroller, it is affected by the Pulse With Modulation (PWM) signal and motor drive gain. The controllers output ranges from -1 to 1 while the PWM signal ranges from -255 to 255 and the input voltage to the d.c. electric motor from

-12 V to 12 V. Therefore, the Digital to Analog (D/A) converter steady state gain becomes

$$K_{D/A} = \frac{\Delta V}{\Delta PWM} = \frac{12 - (-12)}{255 - (-255)} = 0.05\,. \tag{4.3}$$

The air supply valve steady state gain affected by the signal conditioning circuit was obtained in section 3.2.1 and is $K_v = 1.38$. It is affected by K_{PWM} at the input and by K_{pot} and $K_{A/D}$ at the output. Hence, the signal conditioning steady state gain for the air supply valve becomes

$$K = \frac{K_{pot}\,K_{A/D}}{K_{D/A}} = 217.87\,. \tag{4.4}$$

For the pneumatic engine speed, the signal conditioning steady state gain is

$$K = \frac{K_{ss}\,K_{A/D}}{K_{D/A}} = 17.56\,. \tag{4.5}$$

4.1.2 Discretization

The control architectures and tuning rules are initially designed in the s domain, and later on converted to the z domain using the Tustin or bilinear method, given by

$$s = \frac{2}{T_s}\,\frac{z-1}{z+1}\,, \tag{4.6}$$

where T_s is the sampling period. Some control structures, described further in this chapter, require the use of specific transfer functions, such as controllers, models and inverse of models. All transfer functions that are uploaded to the microcontroller must be discrete. The discretizations were carried out in the same way both on simulations and implementations.

The discretization of PID is carried out as follows. The discretization of the integral term is achieved through the bilinear approximation and the derivative term through the backward difference. The backward difference is used instead of the bilinear approximation due to the introduction, by the later one, of a pole at $z = -1$.

Therefore, the discrete PID controller can be written as:

$$u(k) = K_p\, e(k) + K_i\, T_s \sum_{k=1}^{n} (e(k)) + K_d \left(\frac{e(k) - e(k-1)}{T_s} \right) \tag{4.7}$$

where e is the error computed in the current sampling period K, $\sum_{k=1}^{n} (e(k))$ is the accumulated error and $\frac{e(k)-e(k-1)}{T_s}$ is the first order derivative of the error.

4.1.3 Sampling frequency

The sampling frequency was selected based on the time constant of the fastest component, the air supply valve. Sampling period T_s can be related to time constant T in the following way [75]:

$$0.21\, T < T_s < 1.25\, T\,. \tag{4.8}$$

The time constant $T = 0.4$ and therefore T_s should range between 0.08 and 0.50, which is equivalent to a sampling frequency f_s, in Hz, of

$$12.5 > f_s > 2\,. \tag{4.9}$$

The sampling frequency is chosen as $f_s = 10$ Hz.

4.1.4 Deadband compensation

Deadband is a common source of poor performance in control loops [26]. It is a typical control system actuator problem and can be solved in the following ways, providing the deadband is not too large [26]:

- Use of a large gain in a feedback system;
- Adding a dither signal to the controller output;
- Set a minimum controller output.

If a large gain is used, then a small error will result in a high controller output, and the deadband is overcome. However, the dynamic behaviour of the actuator will be

closer to instability.

The dither signal is a high frequency periodic gain added to the controller output. The amplitude of the dither signal is equivalent to the deadband. Higher dither signal frequencies result in a smaller difference between the slope of the deadband region and non-deadband region. It remains a non-linear characteristic but it is much easier to handle than a deadband.

Forcing the controller output to a minimum level, which can be different for each direction, places the output voltage much closer to the deadband limits. Therefore, a small controller output shall be able to move the valve. This is the chosen solution because it is easy to implement with a PWM signal and it does not require considerable computational power when compared to the dither signal. Deadband compensation without reducing the controller resolution can be given by

$$u = u\left(\frac{u_{max} - \text{deadband}}{u_{max}}\right) + \text{deadband} \tag{4.10}$$

where u is the controller output at a given moment, u_{max} is the maximum controller output and deadband is the required controller output to overcome the process deadband. The deadband compensation is not represented in the different control architectures but it is employed.

4.1.5 Algorithm foundations

PID control plays an increasing crucial role in several industries [76] and they are an important part of the control loop. Furthermore, the development of a stable and versatile PID algorithm is an objective of this work.

The bulk of the algorithm is used in all control architectures. The algorithm comprises the following characteristics:

- Constant and controllable sampling frequency: On simulations, constant sampling frequency is introduced within each MatLab®\SimuLink® block [77];
- Direct and reverse acting: Direct and reverse acting can be selected and its implementation consists on multiplying the PID gains K_p, K_i and K_d by -1;

- Smooth transition between switching from manual to auto and vice versa: The smooth transition between switching from manual to auto and vice versa is based on the algorithm proposed in [31]. The setpoint and process output are still known to the PID, so computation is carried out continuously and therefore the control actions should be appropriate to the situation. The output of the controller is switched-off while in manual mode.

- Bumpless changes in the tuning parameters while running: A bumpless parameter change can be achieved by placing the integral gain K_i inside the integral term. By doing so, a change in the controller integral gain K_i will not affect previous accumulated errors that were already affected by the older integral gain;

- Reduced bump due to setpoint changes through setpoint weights.

The reduced bump due to setpoint changes through setpoint weights is based on the following. The PID controller can be tuned to provide a good response to setpoint changes or to disturbance rejection. When they are tuned to provide an adequate disturbance rejection the setpoint changes can be ramped through time, smoothing the transition. A PID algorithm with a simultaneous good response to load disturbance rejection and setpoint changes allows a more unrestricted and versatile tuning.

$$\begin{aligned} e_p &= b\, y_{sp} - y\,, \\ e_i &= y_{sp} - y\,, \\ e_d &= c\, y_{sp} - y\,. \end{aligned} \tag{4.11}$$

This structure is obtained by treating the proportional, derivative and integral error like they are represented in equation 4.11. The constants b and c are called setpoint weights and do not have any impact on the response to load disturbances or noise measurement. The integral action uses true control error to eliminate steady-state errors. PID controllers with setpoint weights are called Two Degrees of Freedom (2DOF) PID controllers because they have different responses to setpoint and disturbance rejections of analog magnitude.

In a typical feedback control loop with a PID controller, if setpoint changes are extremely fast the derivative of this change is very high and u increases drastically. This is known as *derivative kick*. When the setpoint remains constant its derivative is zero. The derivative of the error can be calculated accordingly with equation 4.12 where the coefficient c on equation 4.11 becomes zero.

$$\frac{d\,e}{d\,t} = -\frac{d\,y}{d\,t}\,. \tag{4.12}$$

This method is known as *derivative on measurement*. It is not used on controllers located in the inner loops of cascade systems because the change in the setpoint is controlled by the outer loop controllers. This does not mean that the parameter c on equation 4.11 cannot be tuned, it just usually assumes a non-unitary value [78]. The *derivative kick* is the most significant cause of bumping when parameters are changed but the proportional kick is also present. The setpoint weight b ranges between one and zero being one a response with more overshoot than zero [31].

So, it is possible through setpoint weights to achieve a better performance to both disturbance rejection and setpoint changes. The setpoint weights do not need to have a fixed value. As a matter of fact, variable setpoint weighting is shown in [26]. A very interesting setpoint following method is proposed in [33]. It consists on keeping the control signal at its maximum until the output reaches a percentage of the desired output. After that, the controller resumes its normal operation and leads the process, without overshoot, to its steady-state. When the process model is known, setpoint tracking and response to load disturbance can be completely separated as it is shown in [21].

$$G_c(s) = K_p + \frac{K_i}{s} + K_d\,s \tag{4.13}$$

Controllers with setpoint weights cannot be represented in a block diagram like a typical feedback control loop because the PID internal structure is not acting exclusively on the true control error. A block diagram for a system using a PID controller with setpoint weights is presented in figure 4.1 where G_p is the process, $G_c(s)$ is given by equation 4.13 and $G_{sp}(s)$ is

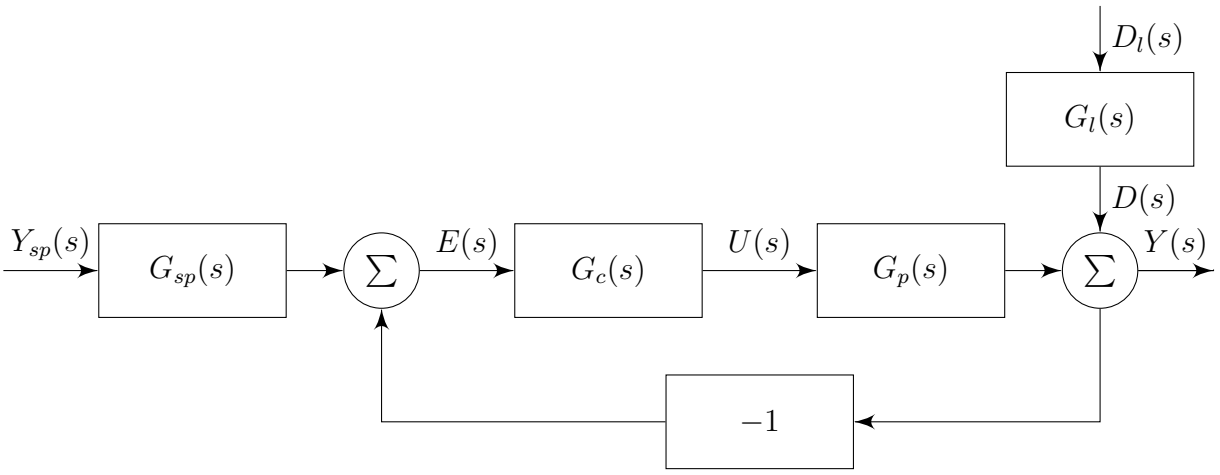


FIGURE 4.1: Two degrees of freedom feedback PID controller.

$$G_{sp}(s) = \frac{b + \dfrac{K_i}{s} + s\,c\,K_d}{1 + \dfrac{K_i}{s} + s\,K_d} = \frac{c\,K_d\,s^2 + b\,s + K_i}{K_d\,s^2 + s + K_i}\,. \tag{4.14}$$

This way, it is easier to tune disturbances and setpoint changes responses separately. The controller G_c is designed in such a way that it gives good disturbance rejection and robustness while G_{sp} is conceived to provide good, but still limited, setpoint following. Within this work the PID controllers with 2DOF are compared to the 1DOF ones in the feedback section, even tough they have a feedforward component.

Summarizing, the feedback control loop, figure 4.1, works in the following way. A process G_p generates an output y, that can be affected by a disturbance $D(s)$. The control error, $e = y_{sp} - y$, quantifies the difference between the process output and the setpoint y_{sp}. Based on the value of the error a controller output u is computed by the process controller G_c and it is fed to the process in an attempt to eliminate the error. If there are any setpoint changes and the setpoint weights b and c are different than one, then G_{sp} will originate an output different then y_{sp}.

However, if the source of error is a load disturbance or noise, than G_{sp} will not have any impact, regardless of the setpoint weights value b or c. The controller output, u, is based on three terms. The P term, proportional to the error, the I term, proportional to the integral of the error, and the D term, proportional to the derivative of the error. Together they act on the present (P) error, on the past error (I) and,

through linear extrapolation, on the future error (D).

4.1.6 Anti-windup

One important performance setback is the occurrence of windup. Windup happens when the controller output exceeds the actuator limits and there will be an undesired discrepancy between the real and the expected actuator output. The system saturates when the controller output is outside the proportional band limits. Additionally, the closed control loop can behave as an open loop because the actuator will remain at its limit regardless of the process output.

If integral action is used in the controller any non-zero error might contribute to an accumulating controller output. It is then required that the error must remain with an opposite signal during an extraordinary length of time until this effect has vanished. During this time, any changes in the process will not have any effect on the actuator. The windup effects can be reduced by introducing some changes in the algorithm such as setpoint limitation, conditional integration, integrator clamping and back-calculation and tracking [79].

Back-calculation and tracking is the only method that does not limit the dynamics of both process and control. It has an advantage due to the fact that the integral action itself will help the controller output to fade away from the saturation limits [26].

Anti-windup techniques can be use alone or combined [80–82]. Other variable structures that have been developed as well as a comparison between the anti-windup methods on two processes can be found in [26]. The results alert that even though some methods behave better than others under some conditions, they might not do so on different processes and controllers.

The back-calculation and tracking, figure 4.2, had a top overall performance for a batch of processes and tunings [26], and for that reason was selected as the anti-windup structure in the controllers. The tracking gain K_t determines how fast the reset is carried out. Smaller values of tracking gain K_t will result in a response with more overshoot. K_t should be larger than $1/(K_p\,K_d)$ and smaller than K_i/K_p. It is suggested at [21] to choose $K_t = |K_p|/\left/(\sqrt{K_i\,K_d}\right)\right.$. If $K_t < K_d\,K_p$ a small change in

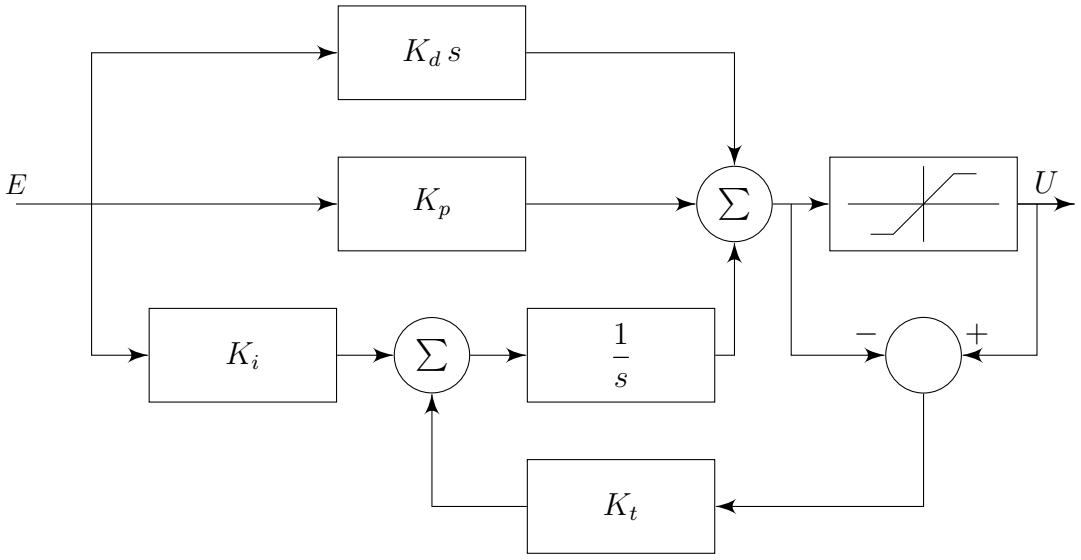


FIGURE 4.2: Block diagram of a PID controller with back-calculation and tracking.

the error can result in a saturated output which might reset the integral action [21].

Wherever a control structure that is not a PID is used, the anti-windup is achieved through clamping. Clamping consists on limiting the controller output before it outreaches the upper or lower limits. During this period of time the controller output is kept constant and near the limits.

4.2 Performance indices and simulation tests

The performance evaluation is of crucial importance in this stage because only the tuning rules with the best performances in simple architectures are used in simulations with more complex architectures. A small description of performance evaluation and proposed tests is carried out.

It can be hard to compare two similar performances or between a significant batch of them. The performance can be assessed in a mathematical way through performance indices. They allow the comparison between the outcome of a controlled process and the desired performance. The most common performance indices are represented in table 4.1.

The ISE was developed to penalize larger errors more than smaller ones. The IAE was presented in [83] in order to penalise persistent oscillations. The ITSE and ITAE are performance indices which aim to emphasise how long it takes for the error to be eliminated [26]. Lower values of performance indices under similar working conditions

TABLE 4.1: Performance indices.

Integral of squared error (ISE)	$\int_0^\infty e(t)^2\, d(t)$
Integral of absolute error (IAE)	$\int_0^\infty \lvert e(t)\rvert\, d(t)$
Integral of time multiplied squared error (ITSE)	$\int_0^\infty t\, e(t)^2\, d(t)$
Integral of time multiplied absolute error (ITAE)	$\int_0^\infty t\, \lvert e(t)\rvert\, d(t)$

can be associated with better performances.

Setpoint tracking and load disturbance rejection are the controllers design primary objectives. A setpoint change response test is proposed, figure 4.3. The load disturbance rejection is a step input applied when no steady-state error is present. The tests are carried out on MATLAB®\SIMULINK®.

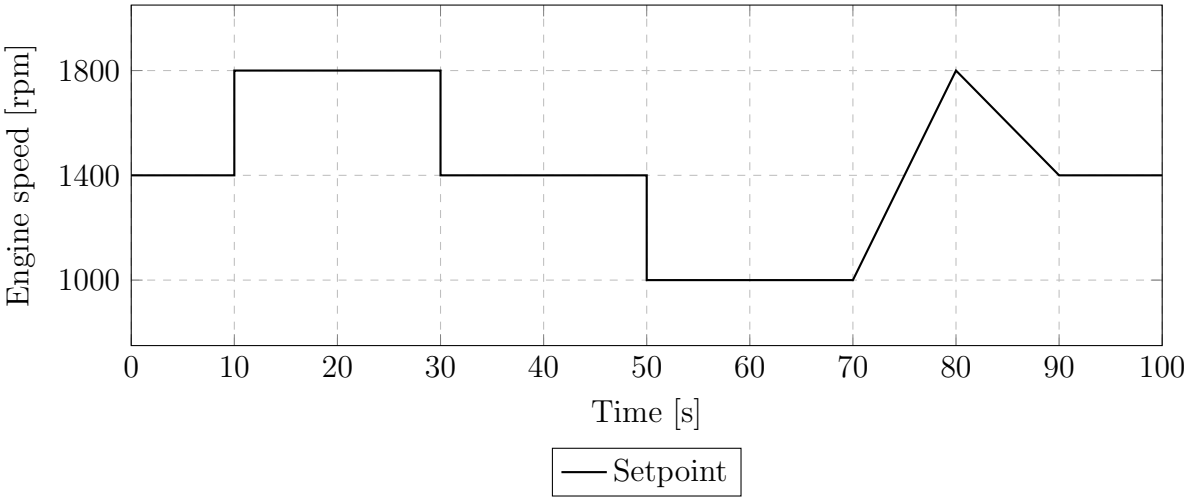


FIGURE 4.3: Setpoint following performance test.

4.3 Control structures

In this section the control structures are presented, tuned and evaluated through simulations. Performance is mathematically quantified using performance indices.

4.3.1 Feedback control

Tuning rules are designed to adjust controller parameters for a specific control structure, a typical process model and a desired response. Only some tuning rules are presented here, even though there are many more [52].

For a SOPTD transfer function given by

$$G(s) = \frac{K}{(T_1 s + 1)(T_2 s + 1)} e^{-sL} = \frac{K}{T_m^2 s^2 + 2\zeta T_m s + 1} e^{-sL} \tag{4.15}$$

the selected tuning rules for PID control to be simulated are:

- Ziegler-Nichols (1942) [40] (ZN): The tuning parameters arise from several tests to a batch of transfer functions. A quarter amplitude damping is expected but not ensured [53]. Modelling is done through tangent method.

$$\begin{aligned} K_c' &= \frac{1.2\,(T_1 + T_2 + L)}{K\,L} \\ T_i &= 2\,L \\ T_d &= 0.5\,L \end{aligned} \tag{4.16}$$

- Lee et al (1998) [52] (Lee): PID tuning rules with a varying closed-loop response that is based on a single parameter, λ. Optimization of the parameter λ is developed in [84]. Modelling parameters are not specified. It is assumed that they are known.

$$\begin{aligned} K_c' &= \frac{T_1}{K(2\lambda + L)} \\ T_i &= T_1 + T_2 - \frac{2\lambda^2 - L^2}{2(2\lambda + L)} \\ T_d &= T_i - T_1 - T_2 + \frac{T_1 + T_2}{T_i} \\ \lambda &= \text{maximum}[0.25\,L; 0.2\,T_1] \end{aligned} \tag{4.17}$$

- Bi et al (1999) [85] (Bi): Tuning rules derive from an auto-tuner algorithm and adapted to a SOPTD process. Modelling technique is not specified. It is assumed that they are known.

$$\begin{aligned} K_c' &= 1.0128\,\zeta\frac{T_m}{K\,L} \\ T_i &= 1.9747\,K\,L \\ T_d &= 0.5064\,\frac{T_m^2}{K\,L} \end{aligned} \tag{4.18}$$

- Vitecková and Vitecek (2000) [86] (VV): Tuning parameters for SOPTD processes are developed based on the desired overshoot. A 10 % overshoot was selected because the Suyama tuning rule [87] is conceived to achieve the same purpose and therefore the performance comparison between them can be interesting to carry out.

$$\begin{aligned} K_c' &= X1\,\frac{T_1+T_2}{K\,L} \\ T_i &= T_1+T_2 \\ T_d &= \frac{T_1\,T_2}{T_1+T_2} \\ X_1 &= 0.581 \end{aligned} \tag{4.19}$$

- Gorez (2003) [88]: PID tuning rules are derived from IMC similarly to the SIMC (see below) where perfect modelling is assumed. Quantification of the tuning parameters differs with the normalized dead time τ. The closed-loop maximum sensitivity is a tuning parameter. Modelling technique is not specified. It is assumed that model parameters are known.

$$\begin{aligned} K_c' &= \frac{2\zeta T_m}{K\,(2\,\zeta\,T_m+L)} \\ T_i &= 2\,\zeta\,T_m \\ T_d &= \frac{T_m}{2\,\zeta} \end{aligned} \tag{4.20}$$

- Suyama (1992) [87] (SYM): Tuning is based on model matching and pole-zero cancellation. The main objectives are simple algebraic parametrization and

provide adequate closed-loop step response with 10 % overshoot. Model parameters are assumed known.

$$\begin{aligned} K_c' &= \frac{T_1 + T_2}{2\,K\,L} \\ T_i &= T_1 + T_2 \\ T_d &= \frac{T_1\,T_2}{T_1 + T_2} \end{aligned} \tag{4.21}$$

- Skogestad (2003) [89] (SIMC): The main objective of Skogestad is to simplify the tuning of PID controllers. Tuning are analytically derived from IMC, where the only tuning parameter is the closed loop time constant. Also known as SIMC (Simple IMC or Skogestad IMC). It is recommended to choose the closed loop time constant equal to the time delay. Model parameters are assumed to be known.

$$\begin{aligned} K_c' &= \frac{2\zeta\,T_m}{K\,T_{CL} + K\,L} \\ T_i &= 2\,\zeta\,T_m \\ T_d &= \frac{T_m}{2\,\zeta} \end{aligned} \tag{4.22}$$

- C.R. Madhuranthakam (2006) [90] (C.R.M.): Minimization of the performance index IAE to estimate optimal tuning parameters of PID controllers for different processes, including a SOPTD.

$$\begin{aligned} K_c' &= \frac{0.5723}{K}\left(\frac{L}{L + T_1 + T_2}\right)^{-1.0409} \\ T_i &= \frac{L}{T_1}0.2476(L + T_1 + T_2)\left(\frac{L}{L + T_1 + T_2}\right)^{-1.6501} \\ T_d &= \frac{0.0943(L\,T_2 + L^2)}{T_1}\left(\frac{L}{L + T_1 + T_2}\right)^{-1.4636} \end{aligned} \tag{4.23}$$

- AMIGO (2004) [12, 45, 91–93]: Developed by Karl J. Aström and Tore Hägglund, the AMIGO stands for an heuristic Approximation of the Maximum Integral Gain Optimization tuning rules, which aim maximizing the integral gain with a constrained closed-loop maximum sensitivity ranging from 1.2 to 2.0. The

normalized dead time τ should be higher than 0.3. Setpoint weights are conservatively used and assume fixed values for different normalized dead times $\tau = (L)/(L + T_{ar})$, where L is the dead time and T_{ar} is the time the process takes to reach 63.2 % of the process steady state gain when a step input is carried out.

$$\begin{aligned} K_p &= \frac{1}{K}\left(0.19 + \frac{0.37\,T_1}{L} + \frac{0.18\,T_2}{L} + \frac{0.02\,T_1\,T_2}{L^2}\right) \\ K_i &= \frac{1}{K}\left(\frac{0.48}{Le + 0.03\,(T_1/L^2)} - 0.0007\frac{T_2}{L^2} + 0.0012\frac{T_1\,T_2}{L^3}\right) \\ K_d &= \frac{1}{K}\left(0.29\,L + 0.16\,T_1 + 0.2\,T_2 + 0.28\,\frac{T_1 + T_2}{L}\frac{T_1 + T_2}{T_1 + T_2 + L}\right) \end{aligned} \tag{4.24}$$

Since $\tau \leq 0.5$ then, $b = 0$ and $c = 0$.

The tuning rules for the feedback control loop are based on the complete system SOPTD transfer function, estimated on chapter 3, with a gain $K_{cs} = 1.95$, time constants $T_1 = 0.6$, $T_2 = 0.4$, time delay $L = 0.52$ and maximum sensitivity $M_s = 1.4$. The transfer function G_{sp} is given by equation 4.14. Controller gains that will be used to control the pneumatic engine are in the table 4.2. Some tuning rules were not developed for the parallel PID but a conversion can be carried out because

$$\begin{aligned} K_p &= K_c', \\ K_i &= \frac{K_c'}{T_i}, \\ K_d &= K_c'\,T_d, \end{aligned} \tag{4.25}$$

where K_c' is the PID proportional gain in the ideal structure.

Setpoint following test is simulated for each tuning rule. Performance indices are presented in table 4.3. A small modification is proposed on the VV tuning rule because it was the one that had the best performance. A 2DOF controller is introduced and the setpoint weights are $c = 0$ and $b = 0.85$. The proportional, integral and derivative gains are kept constant. The modified tuning rule is identified by VV-M. The modified rule presents a better setpoint response performance. Performance index for VV-M is also presented in table 4.3. It is worth noting that using setpoint

TABLE 4.2: Controller gains for SOPTD PID tuning rules.

	K_p	K_i	K_d	b	c
ZN	1.42	1.39	0.38	n/a	n/a
Lee	0.39	0.34	0.14	n/a	n/a
Bi	0.50	0.25	0.06	n/a	n/a
VV	0.57	0.57	0.14	n/a	n/a
Gorez	0.34	0.34	0.08	n/a	n/a
SYM	0.49	0.49	0.12	n/a	n/a
SIMC	0.49	0.49	0.11	n/a	n/a
C.R.M.	0.90	0.47	0.32	n/a	n/a
AMIGO	0.40	0.51	0.15	0	0

weights do not always translate into lower performance indices. However, lower setpoint weights result in less, or even none, overshoot [21].

In the figure 4.4 the top three step response performances are presented and the respective performance indices are in table table 4.4. The tuning rule VV-M has the best performance according to the performance indices and therefore it is, from now on, used in all control loops that attempt to tune the SOPTD process. A load disturbance rejection is simulated using VV, VV-M and SIMC tuning rules. The response is presented in figure 4.5 and its evaluation in table 4.5.

The VV and VV-M tuning rules have the exactly same response to load disturbance rejection as it was expected because they only differ in setpoint weights. The SIMC tuning rule is not as good as VV and VV-M.

There are some aspects of tuning rules that, even though they do not have the best performance, are worth comparing due to their common objectives or tuning principle.

TABLE 4.3: Setpoint following test performance indices for SOPTD tuning rules.

	IAE	ISE	$ITAE$	$ITSE$
ZN	6.98	1.91	310.51	69.98
Bi	7.05	1.95	368.40	86.44
Lee	5.53	1.73	295.00	73.01
VV	4.92	1.62	252.85	63.89
VV-M	4.18	1.34	207.59	48.99
Gorez	5.63	1.85	301.54	79.04
SYM	5.43	1.83	277.05	74.81
SIMC	5.45	1.83	278.07	75.08
C.R.M.	5.62	1.82	281.4	69.08
AMIGO	5.88	1.94	307.79	79.40

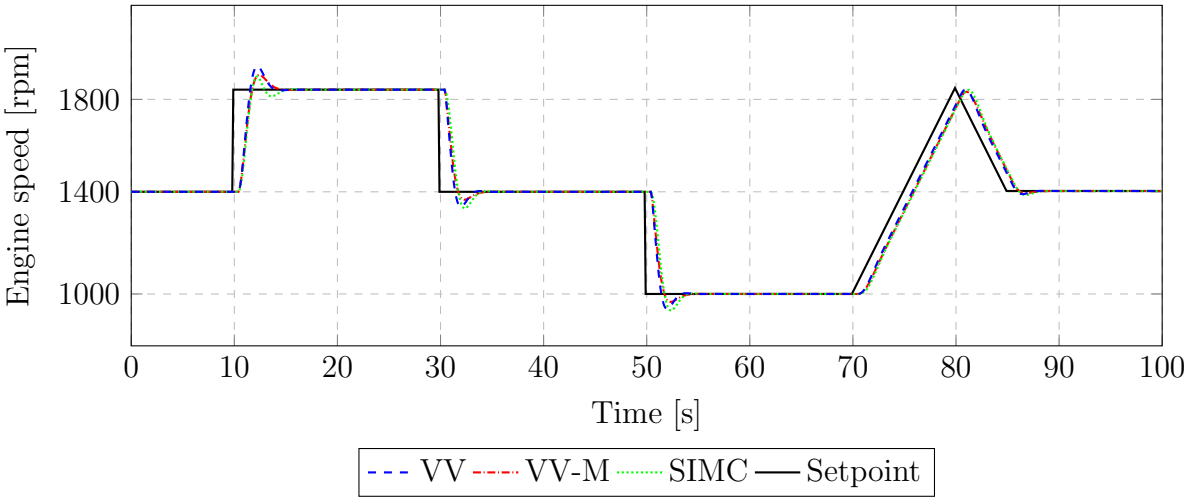


FIGURE 4.4: Setpoint following simulation of a feedback control loop tuned with VV, VV-M and SIMC.

TABLE 4.4: Simulation performance indices comparison for PID feedback control.

	IAE	ISE	$ITAE$	$ITSE$
SIMC	5.63	1.84	301.54	79.02
	↓12.61 %	↓11.95	↓16.15 %	↓19.15 %
VV	4.92	1.62	252.85	63.89
	↓15.04 %	↓17.21 %	↓17.93 %	↓23.31 %
VV-M	4.18	1.34	207.59	48.99

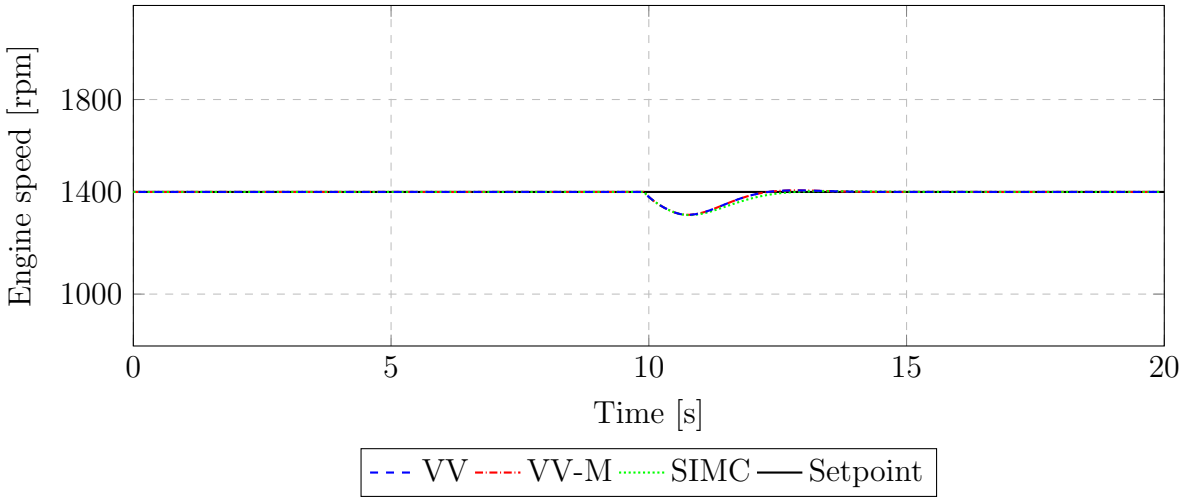


FIGURE 4.5: Load disturbance rejection simulation of a feedback control loop tuned with VV, VV-M and SIMC.

The VV and SYM were conceived to have a response with 10% overshoot. For both of them, T_i and T_d are obtained exactly in the same way and K_c' in a very similar manner. It can be observed that VV has a higher overshoot, however the SYM and the SIMC present an undershoot that contributes to a performance indices increase. The Gorez, SYM and SIMC are an attempt to transfer the IMC principles into PID tuning. The SYM and SIMC have very similar responses. The only difference between the Gorez and SIMC is K_c' that, under these working conditions, produced has a smaller value that manifested higher values of performance indices.

Table 4.5: Load disturbance simulation performance indices comparison for PID feedback control.

	IAE	ISE	$ITAE$	$ITSE$
SIMC	0.22	0.02	2.48	0.25
	↓9.12 %	—	↓10.14 %	↓12.03 %
VV	0.20	0.02	2.23	0.22
	—	—	—	—
VV-M	0.20	0.02	2.23	0.22

4.3.2 Cascade control

One of the main advantages of the cascade control is to provide an effective disturbance rejection on the inner loop at the cost of additional sensors and control loop complexity [21]. Other important advantage is the transfer of actuator non-linearities to the inner controller. Cascade control can be used when there are more measurable signals than primary control variables.

There is a clear relationship between the primary and secondary control variables, the latter ones have influence on the former and the inner loop responds three to four times faster than the outer loop [31]. In the pneumatic engine, the control variable is the speed and the extra measurable signal is the valve position. So, the primary control variable is the pneumatic engine speed and the secondary is the valve position.

In this section two cascade control structures are compared. The simulations and performance indices are presented in the same figure and table. A block diagram of a typical cascade control structure with two loops is presented in figure 4.6. In cascade control, the controllers and process with the number 1 are located in the outer loop and the ones with the number 2 in the inner loop.

Even though there are some developed approaches to tune cascade controllers [20, 21, 26, 31], tuning rules based on the process parameters are not so common.

A performance comparison carried out in [94] uses tuning rules for PID controllers in a simple feedback control loop to obtain the proportional gain for the inner loop

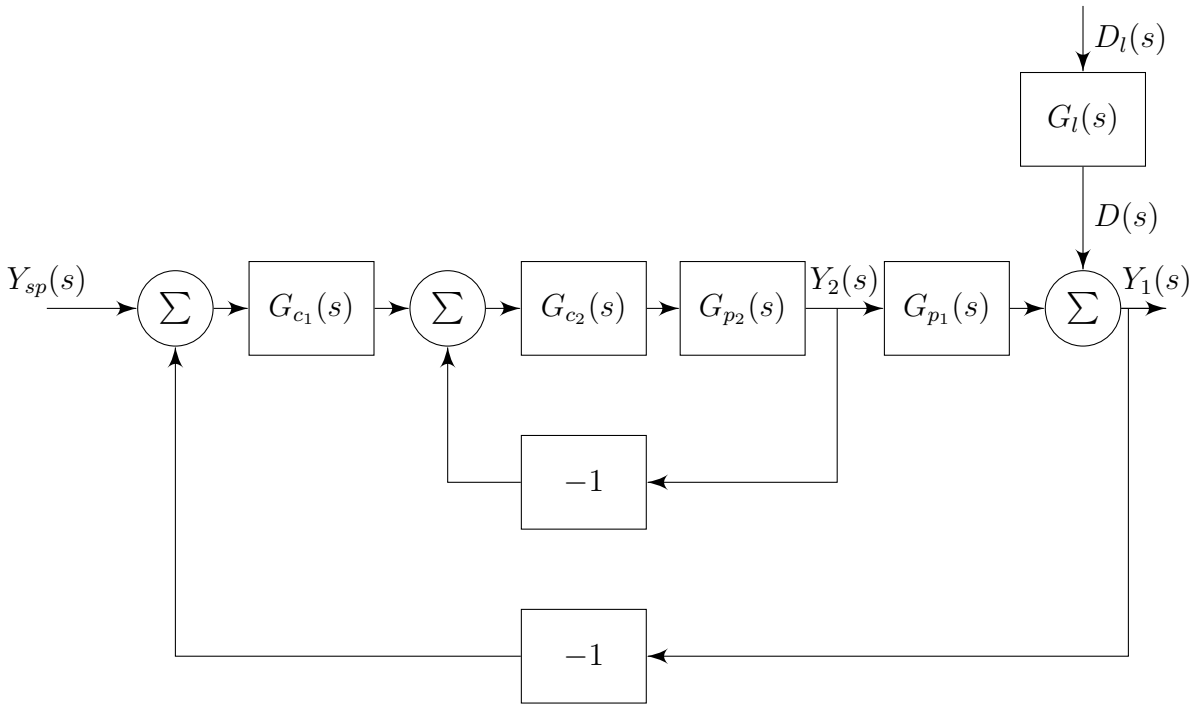


FIGURE 4.6: Cascade control structure.

controller. However, the aim of this dissertation is to compare control structures and tuning rules developed specifically for them and therefore this approach was not selected.

A tuning rule for Cascade Controllers (CC) is shown in [26]. It was developed for First Order Plus Time Delay (FOPTD) processes where the inner loop transfer function is given by

$$G_{p1} = \frac{K_1}{T_1 s + 1} e^{-L_1 s} \tag{4.26}$$

and the outer loop one by

$$G_{p2} = \frac{K_2}{T_2 s + 1} e^{-L_2 s} \,. \tag{4.27}$$

Tuning parameters for the inner loop controller are

$$\begin{aligned} K_{p2} &= \frac{T_2 + \frac{L_2^2}{2(\lambda_2 + L_2)}}{K_2(\lambda_2 + L_2)} \,, \\ T_{i2} &= T_2 + \frac{L_2^2}{2(\lambda_2 + L_2)} \,, \\ T_{d2} &= \frac{L_2^2}{6(\lambda_2 + L_2)} \left(3 - \frac{L_2}{T_2 + \frac{L_2^2}{2(\lambda_2 + L_2)}} \right) \,. \end{aligned} \tag{4.28}$$

Tuning parameters for the outer loop controller are

$$\begin{aligned} K_{p1} &= \frac{T_1 + \lambda_2 + \frac{(L_1+L_2)^2}{2(\lambda_1+L_1+L_2)}}{K_1(\lambda_1 + L_1 + L_2)}\,, \\ T_{i1} &= T_1 + \lambda_2 + \frac{(L_1 + L_2)^2}{2(\lambda_1 + L_1 + L_2)}\,, \\ T_{d1} &= \frac{\lambda_2 T_1 - \frac{(L_1+L_2)^3}{6(\lambda_1+L_1+L_2)}}{T_1 + \lambda_2 + \frac{(L_1+L_2)^2}{2(\lambda_1+L_1+L_2)}} + \frac{(L_1 + L_2)^2}{2(\lambda_1 + L_1 + L_2)}\,. \end{aligned} \tag{4.29}$$

Parameters λ_1 and λ_2 allow a trade-off between aggressiveness and robustness. Based on simulations [26] the suggested values are

$$\begin{aligned} \lambda_2 &= 0.5L_2 \\ \lambda_1 &= 0.5(L_1 + L_2) \end{aligned} \tag{4.30}$$

which yields the following gains for the inner loop controller

$$\begin{aligned} K_{p2} &= 2.09\,, \\ K_{i2} &= 4.83\,, \\ K_{d2} &= 0.06\,, \end{aligned} \tag{4.31}$$

and for the outer ones

$$\begin{aligned} K_{p1} &= 0.75\,, \\ K_{i1} &= 0.91\,, \\ K_{d1} &= 0.13\,. \end{aligned} \tag{4.32}$$

The other control structure is a scheme based on a Cascade Control loop with a Smith Predictor (CC-SP), proposed in [95], figure 4.7. The Smith predictor is employed to compensate for the delay term in the primary control loop, which is the source of most of the time delay in the CE107 apparatus.

The simulated responses to setpoint test are presented in figure 4.8 and respective performance indices in table 4.6

Introducing a Smith Predictor in the cascade control loop improved the response to setpoint changes. Furthermore, and even though overshoot is not an evaluation criteria, it is worth noting that it disappears when the Smith Predictor is introduced.

A load disturbance rejection simulation is carried out for both CC and CC-SP, presented in figure 4.9 and performance indices in table 4.7.

The CC-SP did not improve the load disturbance rejection when it is evaluated through the selected performance indices

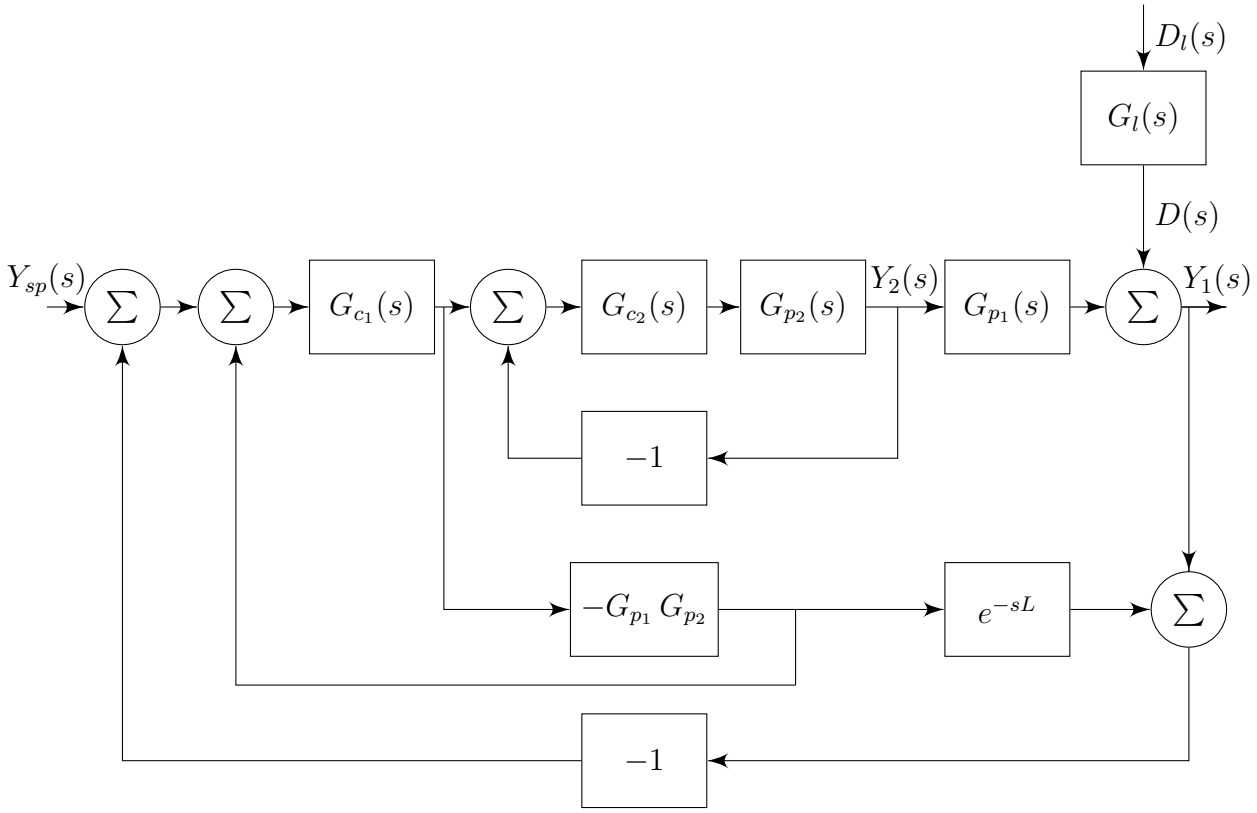


FIGURE 4.7: Smith predictor based cascade control structure.

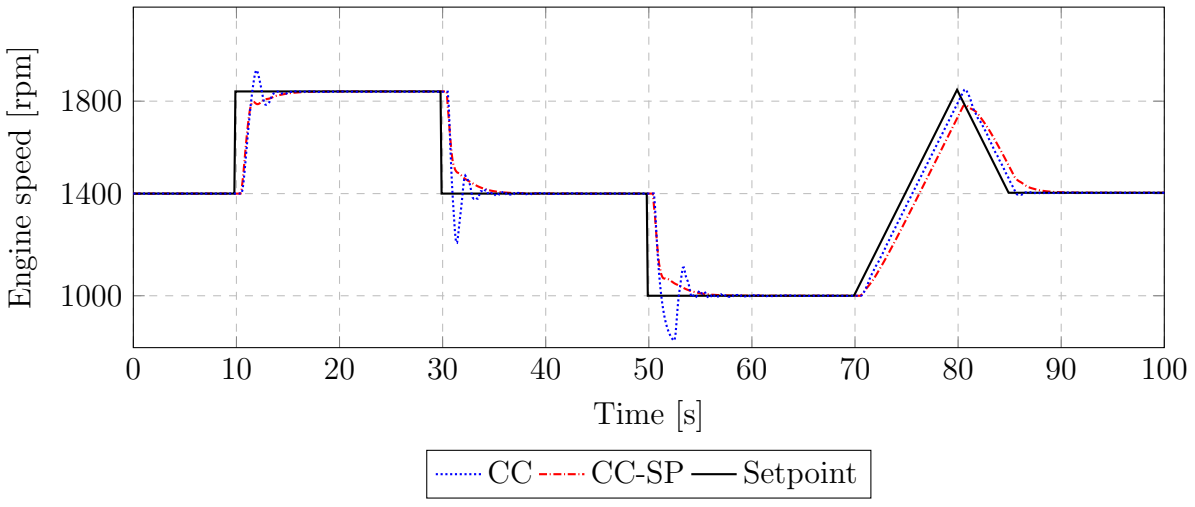


FIGURE 4.8: Setpoint following simulated results of CC and CC-SP.

TABLE 4.6: Setpoint following simulated performance indices for CC and CC-SP.

	IAE	*ISE*	*ITAE*	*ITSE*
CC	4.95	1.43	264.10	59.48
	↓9.12 %	—	↓10.14 %	↓12.03 %
CC-SP	4.19	1.33	196.64	46.35

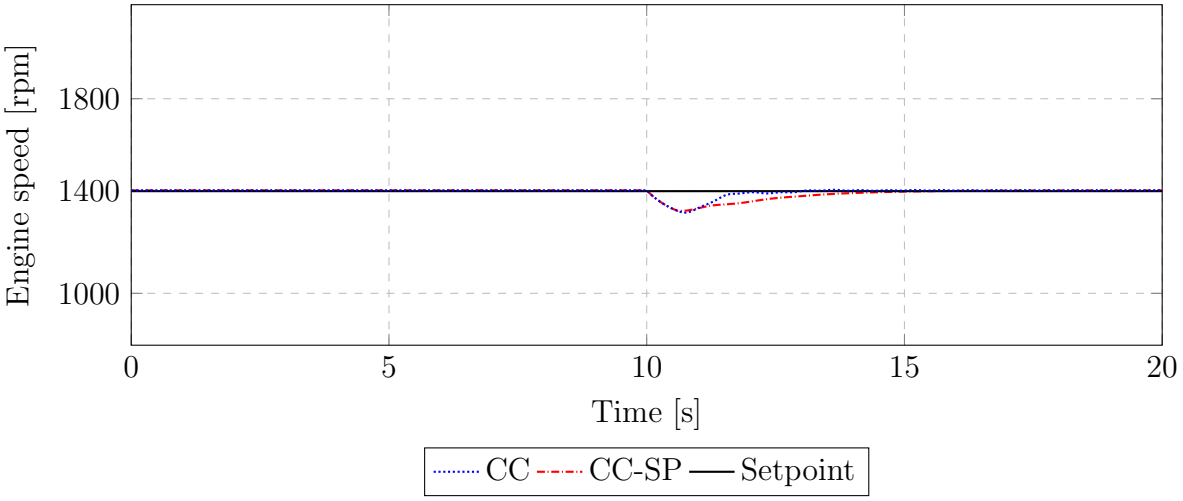


FIGURE 4.9: Load disturbance rejection simulated results of a cascade control structure tuned with VV and VV-M.

TABLE 4.7: Load disturbance rejection simulated performance indices for CC and CC-SP.

	IAE	*ISE*	*ITAE*	*ITSE*
CC	0.16	0.01	1.48	0.12
	↑68.75 %	↑100.00 %	↑466.89 %	↑1641.67 %
CC-SP	0.27	0.02	8.39	2.09

4.3.3 Internal Model Controller

The complexity and uncertainty of several processes made it very difficult to implement more traditional tuning methods. In the late 1970s model and disturbance uncertainty limitations were overcome by the internal mode control theory [75]. It is still widely use to control a variety of processes [96–98]. Its structure is represented in the figure 4.10.

The process G_p is controlled by a low-pass filter G_f, its model $\hat{G}_p$ and the inverse of the model $\hat{G}_p^{-1}$. It has some known drawbacks such has the slow response to load disturbances [20, 21, 26, 31].

The first order low-pass filter is given by

$$G_f(s) = \frac{1}{s\,T_f + 1}\,. \tag{4.33}$$

The inverse of a SOPTD model can be given

$$G_p^{-1} = \frac{(sT_1 + 1)(sT_2 + 1)}{K(sT_1/N + 1)(sT_2/N + 1)} e^{-sL}\,, \tag{4.34}$$

where the time delay e^{-Ls} is not inverted. The parameter N gives the frequency range where the inversion is valid. A value of 10 is common [21]. Hence, and combining

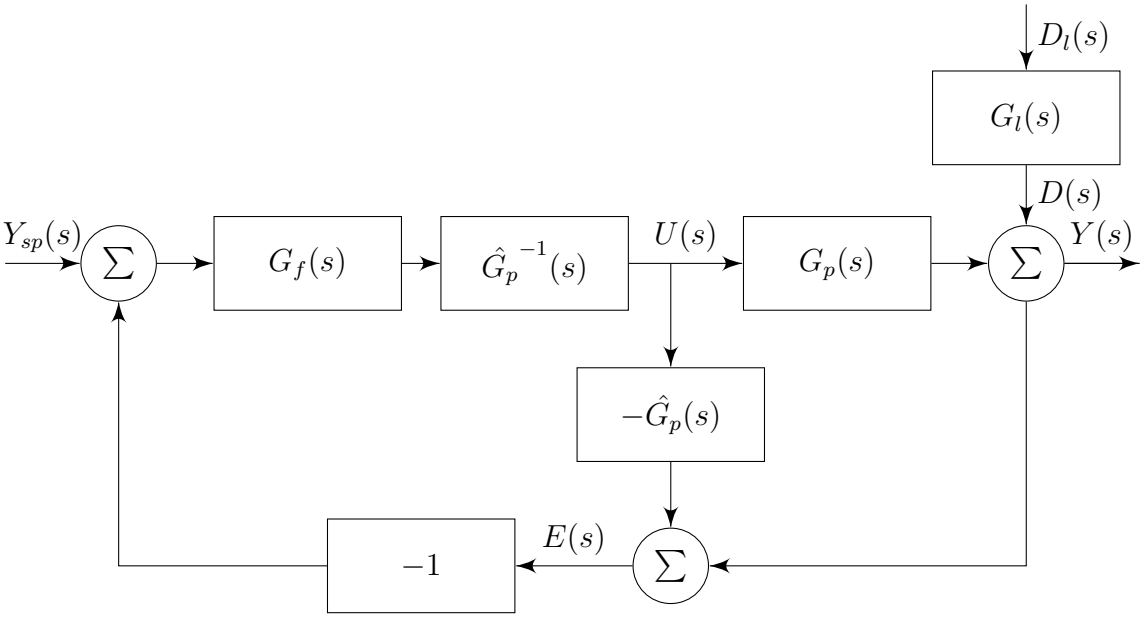


FIGURE 4.10: Block diagram of a process G_p controlled by an IMC.

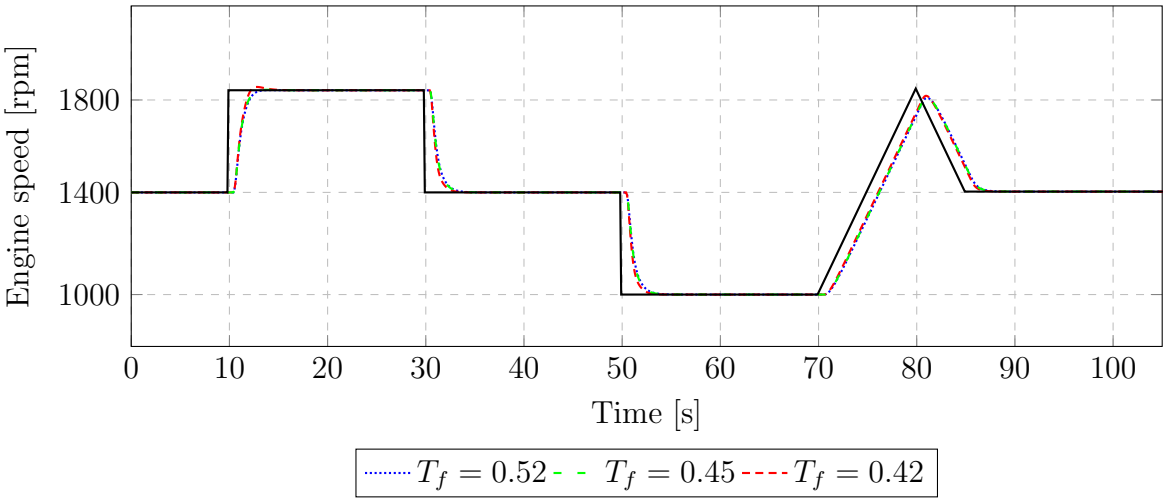


FIGURE 4.11: Simulated IMC control structure setpoint step response for different values of T_f.

equations 4.39, 4.38 and 4.34, G_{ffsp} becomes

$$\begin{aligned} G_{ffsp} &= \frac{1}{10s+1}\frac{(0.6s+1)(0.4s+1)}{1.95\left((0.06s+1)(0.04s+1)\right)}e^{-0.52s} \\ &= \frac{60\,s^2+250\,s+250}{1.17\,s^3+49.92\,s^2+536.25\,s+487.50}e^{-0.52s} \end{aligned} \tag{4.35}$$

Anti-windup is achieved through clamping and not through back-calculation, like it was previously used in the other structures, because no integral and derivative gains are being used here. Anti-windup must be exactly the same for the process G_p and process model $\hat{G}_p$ [31, 75].

The IMC response to setpoint changes and load disturbances can be easily adjusted if T_f is considered a tuning parameter. The simulations were carried out for the different values for the time constant T_f. A better performance is achieved when $T_f = L$ [89]. The setpoint tracking test was carried out for $T_f = 0.45$, $T_f = 0.52$ and $T_f = 0.55$, figure 4.11 and the performance indices under this conditions are in table 4.8. A better performance was achieved when $T_f = 0.45$ which is lower than the process time delay.

A load disturbance is simulated using the same time constants as peer the setpoint following simulations. The results are presented in figure 4.12 and performance indices in table 4.9. Once again, the best performance is achieved when $T_f = 0.45$ and not when $T_f = L$ like it is proposed in [89].

TABLE 4.8: Simulated performance indices for the setpoint following test using IMC.

	IAE	ISE	$ITAE$	$ITSE$
$T_f = 0.55$	4.98	1.61	267.23	66.8
	↓2.42 %	↓1.87 %	↓2.27 %	↓2.45 %
$T_f = 0.52$	4.86	1.58	261.01	65.13
	↓4.32 %	↓4.41 %	↓4.89 %	↓6.13 %
$T_f = 0.45$	4.65	1.51	248.30	61.17
	↑2.80 %	↑2.65 %	↑7.83 %	↑3.33 %
$T_f = 0.42$	4.78	1.55	267.74	63.21

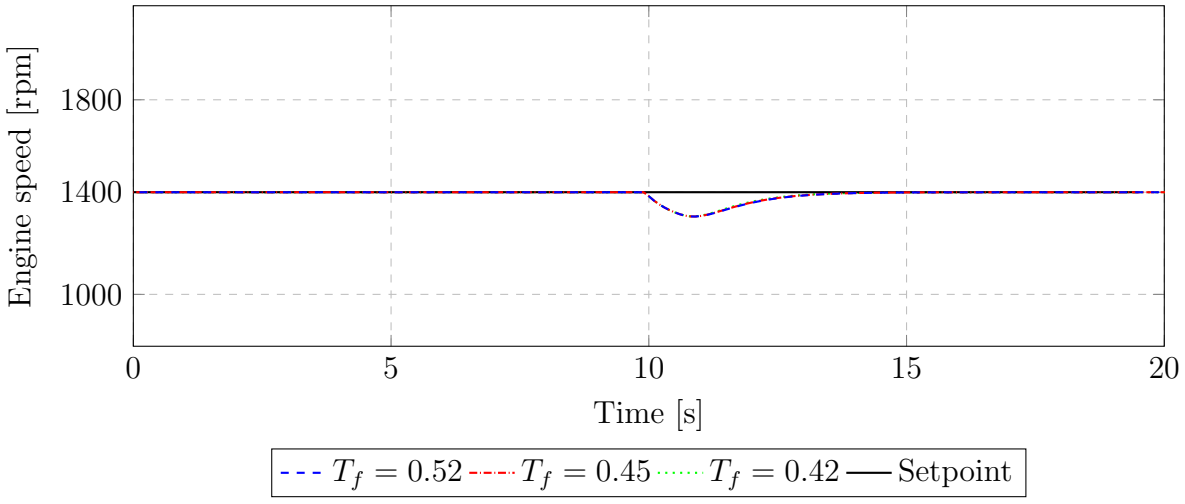


FIGURE 4.12: Load disturbance rejection simulation of IMC for different values of T_f.

TABLE 4.9: Simulated performance indices comparison of IMC load disturbance for $T_f = 0.42$, $T_f = 0.45$ and $T_f = 0.52$.

	IAE	ISE	$ITAE$	$ITSE$
$T_f = 0.52$	0.28	0.03	3.43	0.34
	$\downarrow$7.14 %	—	$\downarrow$7.28 %	$\downarrow$8.82 %
$T_f = 0.45$	0.26	0.03	3.18	0.31
	$\uparrow$3.85 %	—	$\uparrow$5.97 %	$\uparrow$6.45 %
$T_f = 0.42$	0.27	0.03	3.37	0.33

4.3.4 Feedback and Feedforward

Feedforward (FF) is a control technique that complements feedback (FB) and can be used to improve setpoint response. In this section it will be shown a complete separation from load disturbance rejection to setpoint response.

The SetPoint response through FeedForward (SPFF) is a two degree of freedom control structure and can be achieved by introducing the feedforward controller G_{ffsp} like its is done in figure 4.13 [21]. The FF controller action is added to the FB one before the anti-windup structure takes action, which means the controller output $U(s) = U_{fb}(s) + U_{ff}(s)$ remains within the working range.

The controllers $G_{sp}(s)$ and $G_{ffsp}(s)$ compose the feedforward compensator. $G_{sp}(s)$ is tuned to give the desired response to step-point changes, i.e., is a transfer function of any order, with or without time delay.

The arrangement works as follows. Assume that $E(s) = 0$. When a change in the setpoint occurs, $G_{ffsp}(s)$ will generate an output $U_{ff}(s)$, which in turn will contribute to the desired $G_p(s)$ controller output, $U(s)$. Load is introduced, like in all previous structures, after the process.

The transfer function from setpoint to the process output is given by

$$\frac{Y}{Y_{sp}} = G_{sp} + \frac{G_p\, G_{ffsp} + G_{sp}}{1 + G_p\, G_c}\,. \tag{4.36}$$

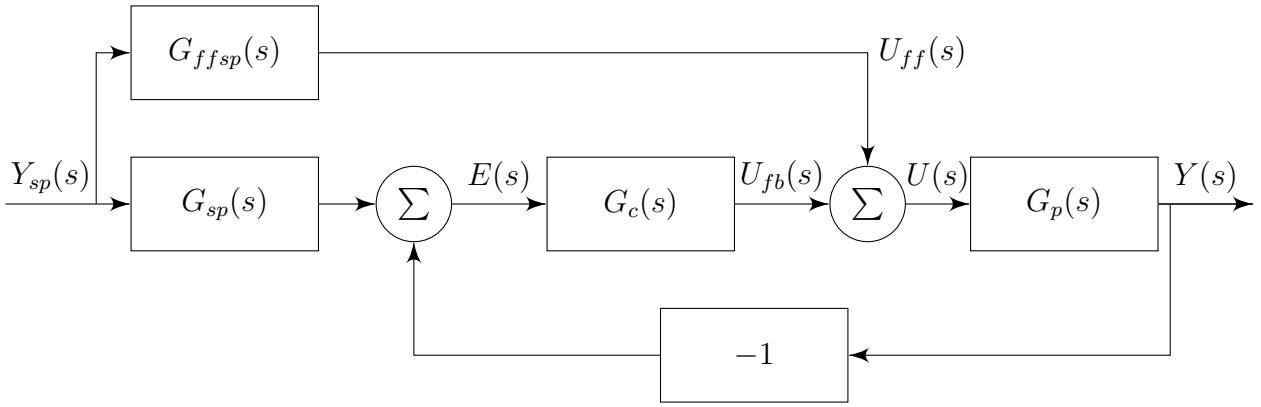


FIGURE 4.13: Two degrees of freedom PID controller with feedback and feedforward designed to improve setpoint response.

The term G_{sp} is the desired setpoint response. The other term can be made small by making $G_p\,G_c$ large and/or by making $G_p\,G_{ffsp} + G_{sp}$ very small. The ideal feedforward controller is achieved when

$$G_{sp} = G_p\,G_{ffsp}\,, \tag{4.37}$$

which means that G_{ffsp} depends of the inverse of the process model, i.e.,

$$G_{ffsp} = G_{sp}\,G_p^{-1}\,. \tag{4.38}$$

In an initial approach $G_{sp}(s)$ is given by

$$G_{sp}(s) = \frac{1}{10\,s+1}\,. \tag{4.39}$$

and the inverse of the model was already obtained in the IMC section, equation 4.35.

The SPFF does not work very well in process with high normalized dead time $\tau = L/(L+T)$. The pneumatic engine was approximated by a SOPTD on chapter 3, where the dead time was assumed to be pure, i.e., it was given by $e^{-0.52\,s}$. In this control structure a first order Padé approximation, shown in [31], is proposed and is given by

$$e^{-s\,L} = \frac{-\frac{L}{2}s+1}{\frac{L}{2}s+1}\,. \tag{4.40}$$

The SOPTD and becomes

$$G_p = \frac{1.95}{(0.6s+1)(0.4s+1)} \frac{-\frac{0.52}{2}s+1}{\frac{0.52}{2}s+1} = \frac{-0.507s+1.95}{0.0624s^3+0.5s^2+1.26s+1}. \tag{4.41}$$

Hence, combining equations 4.35, 4.38 and 4.39, the function G_{ffsp} is given by

$$\begin{aligned} G_{ffsp} &= \frac{6000\,s^2+25000\,s+25000}{117\,s^2+4875\,s+48750} \frac{-\frac{0.52}{2}s+1}{\frac{0.52}{2}s+1} \\ &= \frac{780\,s^3+250\,s^2-9250\,s-12500}{15.21\,s^3+692.25\,s^2+8775\,s+24375}. \end{aligned} \tag{4.42}$$

The response to a setpoint step change based on SPFF using pure dead time and a first order Padé approximation is shown in figure 4.14. The process takes 10 seconds to reach 62.3% of its steady state gain, which is as expected because it the time constant T in G_{sp}. In the same simulation a load disturbance is introduced after the setpoint reaches a steady state value. Performance indices for this simulations are presented in table 4.10. The values of performance indices does not include the load disturbance response, so that they can be used to compare with the experimental results. Approaching the time delay by a first order Padé approximation did not improve the performance.

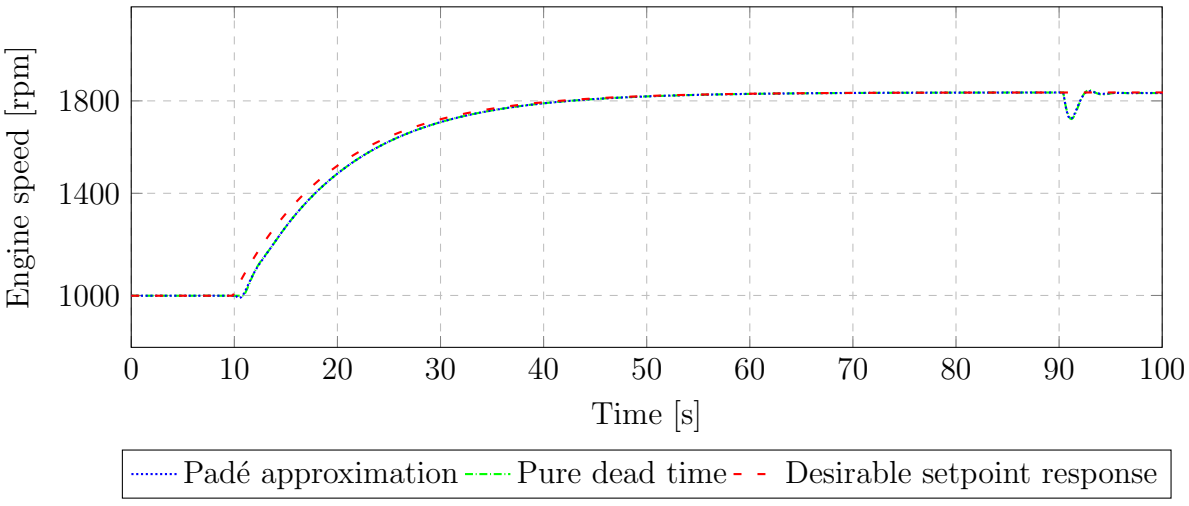


FIGURE 4.14: Setpoint following through SPFF using a pure and first order Padé approximations of the time delay. A load disturbance is introduced at 90 seconds.

TABLE 4.10: Simulated performance indices comparison for SPFF using pure and Padé time delay.

	IAE	ISE	$ITAE$	$ITSE$
Pure	1.25	0.07	25.64	1.18
	↑0.83 %	↑14.37 %	↑0.46 %	↑0.8 %
Padé	1.26	0.08	25.75	1.19

A change in SPFF is proposed. It consists on the introduction of a Smith predictor in order to attempt to reduce deviations from the desirable response and attenuate the effects of dead-time. This new structure is referred to as SPFF-SP and is presented in figure 4.15.

Similarly to the SPFF, a setpoint following test with a time constant of 10 sec-

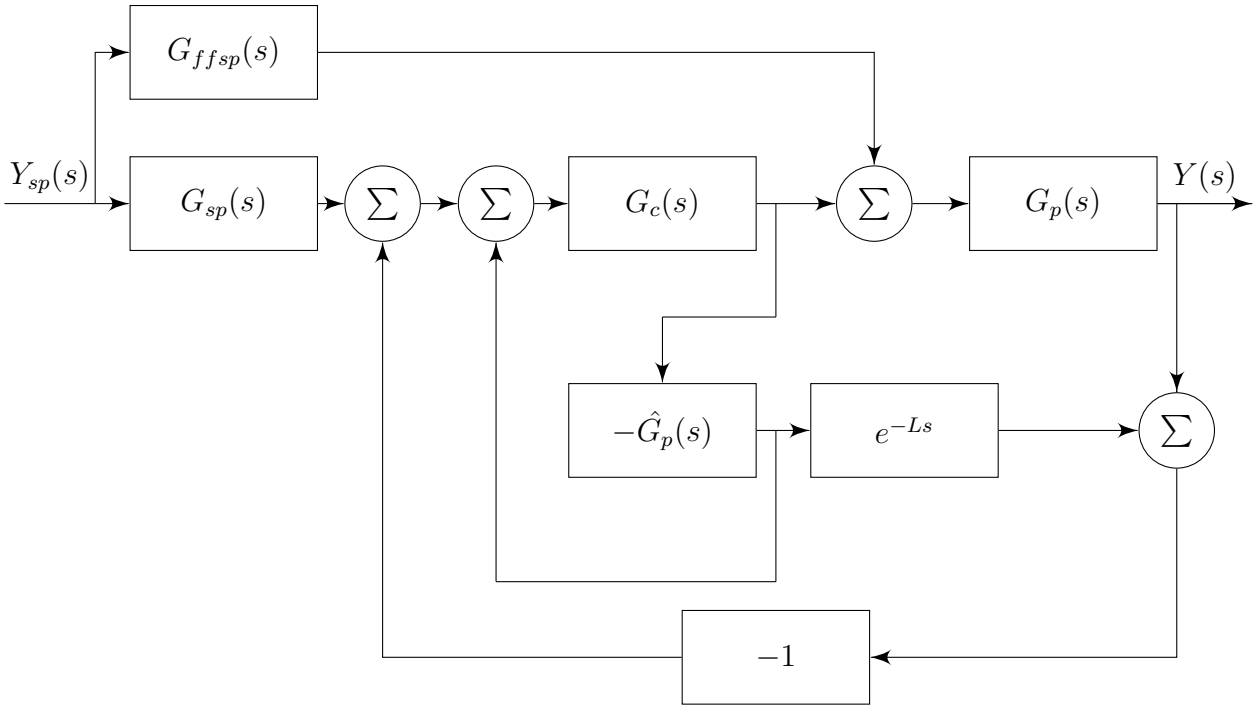


FIGURE 4.15: Two degrees of freedom PID controller with feedback, feedforward and a Smith predictor designed to improve setpoint response.

onds is carried out for both pure time delay and first order Padé approximations. The results are presented in figure 4.16 and the performance indices on table 4.11. Once again, the Padé approximation increased the performance indices.

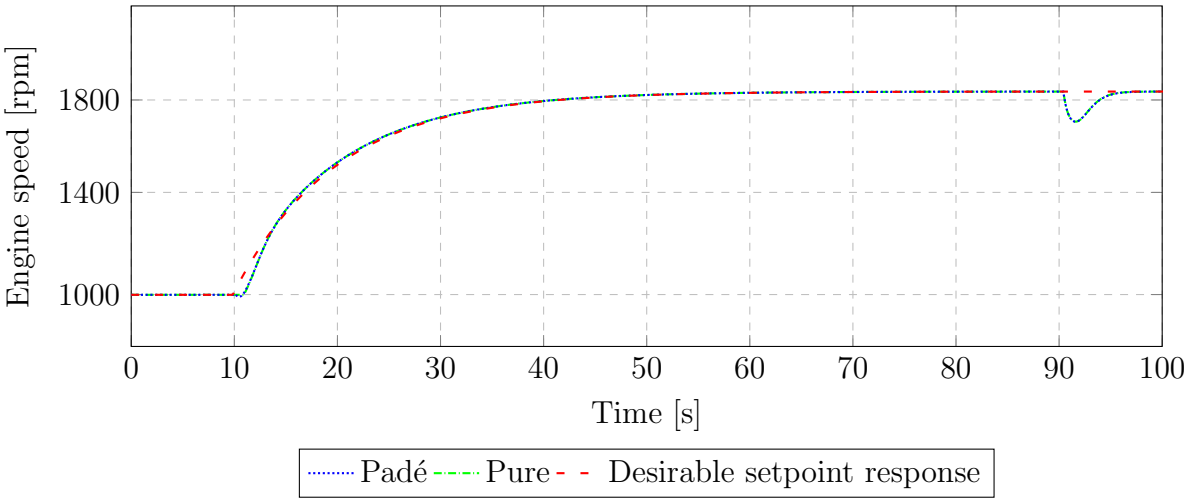


FIGURE 4.16: Setpoint following through feedforward action using SPFF-SP.

TABLE 4.11: Simulated performance indices comparison for SPFF-SP using pure and Padé time delays.

	IAE	*ISE*	*ITAE*	*ITSE*
Pure	0.51	0.03	9.33	0.32
	↑1.95 %	—	↑3.62 %	↑3.18 %
Padé	0.52	0.03	9.67	0.33

Finally, a comparison between SPFF and SPFF-SP is carried out. The performance was better for both while using pure dead time. The introduction of the Smith predictor improved the performance indices, table 4.12.

TABLE 4.12: Simulated performance indices comparison between SPFF and SPFF-SP using pure time delay.

	IAE	*ISE*	*ITAE*	*ITSE*
SPFF	1.25	0.07	25.64	1.18
	↓60.24 %	↓57.13 %	↓63.68 %	↓27.15 %
SPFF-SP	0.51	0.03	9.33	0.32

Setpoint test is carried out for SPFF-SP with pure dead time since it is when best

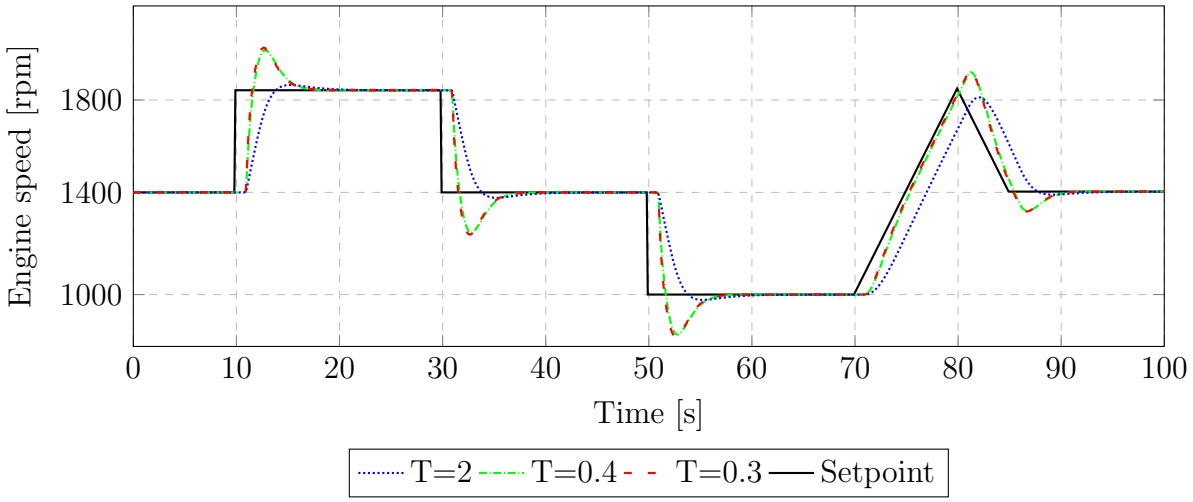


FIGURE 4.17: SPFF-SP setpoint following test for different values of T.

TABLE 4.13: Simulated performance indices comparison for a range of T in SPFF-SP.

	IAE	*ISE*	*ITAE*	*ITSE*
T=2	8.46	3.09	451.7	141.15
	↓25.44 %	↓33.36 %	↓35.37 %	↓45.05 %
T=0.4	6.31	2.06	292.39	77.7
	↑3.36 %	↑2.48 %	↑4.94 %	↑3.93 %
T=0.3	6.52	2.11	306.8	80.74

performance is achieved. The time constant T was changed based on trial an error in an attempt to reduce performance indices. The results are presented in figure 4.17 and the performance indices in 4.13.

A load disturbance test is simulated for both SPFF and SPFF-SP, figure 4.18 and table 4.14. The introduction of a Smith predictor significantly decreased the performance of the architecture to load disturbance rejection.

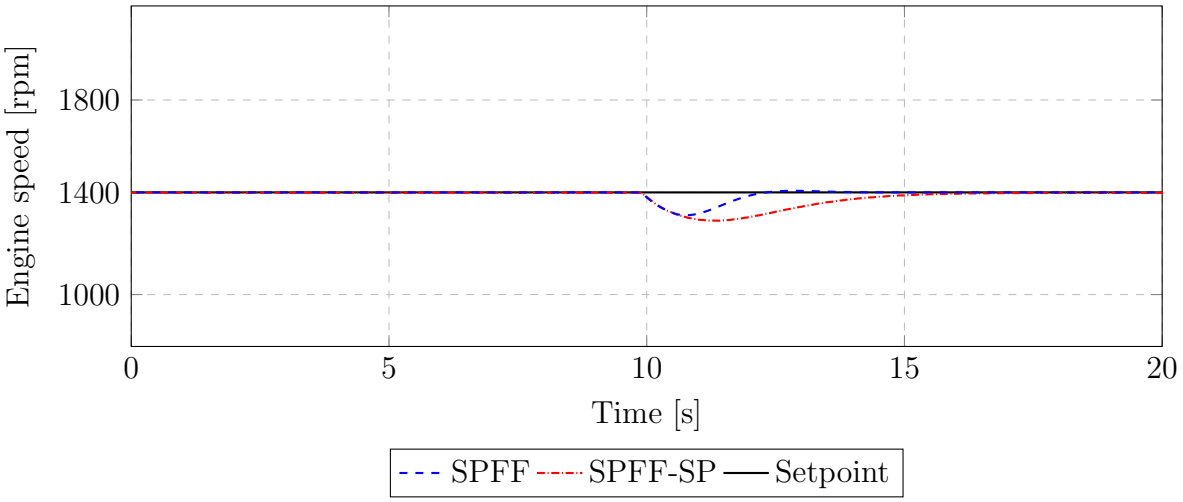


FIGURE 4.18: Load disturbance rejection simulation of SPFF and SPFF-SP.

TABLE 4.14: Simulated performance indices comparison of SPFF and SPFF-SP load disturbance.

	IAE	*ISE*	*ITAE*	*ITSE*
SPFF	0.20	0.02	2.23	0.22
	↑160.07 %	↑200.01 %	↑24.65 %	↑27.31 %
SPFF-SP	0.52	0.06	2.78	0.28

4.3.5 Model-Free controller

The model-free control was initially developed by Fliess and Join, see [60, 99–102] where a very good compilation of their work, and of course of many others, as well as an incredible list of successful applications can be found, free of charge. Most of this section in based on these sources of information.

Consider a SISO system with a control variable u, output y and an unknown or partially known process G_p. The ultra-local model is given by

$$y^v = G_p + \alpha\, u\,, \tag{4.43}$$

where y^v is the derivative of order v of y selected by the practitioner[1], α is a non-physical constant parameter obtained through trial and error[2] and G_p comprehends

[1]Practical applications show that a value for v of one or two provides satisfactory performance.
[2]The magnitude of $\dot{y}$ should be the same of αu.

the unknown parts of the process and possible disturbances without any distinction between them.

The equation 4.43 is not valid for long time lapses and therefore it must be continuously updated. A high sampling frequency is recommended.

Assuming $v = 2$ in equation 4.43, then

$$\ddot{y} = G_p + \alpha\, u\,. \tag{4.44}$$

Closing the control loop with an *intelligent PID* (*iPID*) controller,

$$u = -\frac{G_p - \ddot{y} + K_p\, e + K_i \int e + K_d\, \dot{e}}{\alpha}\,. \tag{4.45}$$

Combining equations 4.44 and 4.45 it is obtained

$$\ddot{e} + K_d\, \dot{e} + K_p\, e + K_i \int e = 0\,. \tag{4.46}$$

Special attention must be given to the fact that G_p, the unknown parts and the possible disturbances, no longer appear in equation 4.46 and all that remains is a linear differential equation with constant parameters. No attempt is made to find out the unknown parts and possible disturbances of G_p.

The parameter $v = 1$, in equation 4.43 will produce an iPI controller given by

$$u = -\frac{G_p - \dot{y} + K_p\, e + K_i \int e}{\alpha}\,, \tag{4.47}$$

where K_i can be set to zero in order to produce an iP controller. An iP controller avoids anti-windup algorithms and it is here implemented, where $K_p = K_{pVV-M} = 0.57$ and $K_i = K_{iVV-M} = 0.57$.

The iP controller output is given by

$$u(t) = u(t - T_s) - \frac{e(t) - e(t - T_s)}{h\alpha} + \frac{K_p}{\alpha} e(t)\,, \tag{4.48}$$

which is equivalent to a velocity algorithm PI controller output,

$$u(t - T_s) = u(t - T_s) + K_p(e(t) - e(t - h)) + K_i T_s e(t)\,, \tag{4.49}$$

if

$$\begin{aligned} K_p &= -\frac{1}{h\alpha} \\ K_i &= \frac{K_p}{h\alpha}\,. \end{aligned} \tag{4.50}$$

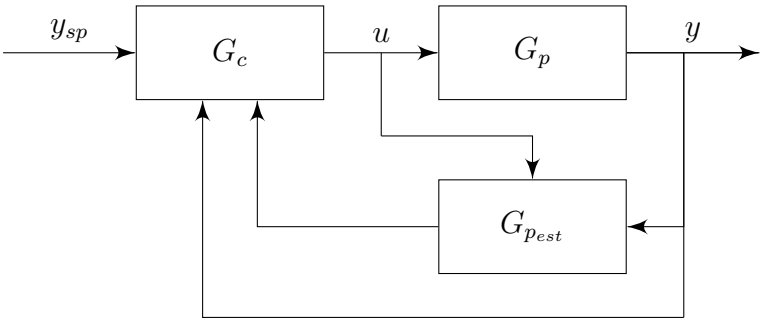


FIGURE 4.19: Block diagram of a process G_p controlled by a model-free controller.

An extraordinarily interesting comparison must be highlighted here. An iP controller is equivalent to a PI one, where no steady-state error is present in a response to step input. This controllers are not arranged in the same way as the feedback ones. A block diagram is presented in figure 4.19.

It was already stated that this control technique does not depend on the unknown parts of the process, equation 4.46. However, in equation 4.45, one can confirm that the controller output still depends on G_p, or better, an estimation of it. So, total independence from the unknown parts is only valid when perfect estimation is achieved.

This may appear quite uncomfortable and not advantageous at all, because it might look that one needs to know the unknown in order to make it work and it would fall into an algebraic loop. However, good estimation of G_p, from now on called $G_{p_{est}}$, was already developed.

One possible estimation is to close the loop through an iP controller and a first order ultra-local model. It yields

$$G_{p_{est}}(t) = \frac{1}{\psi}\left[\int_{t-\psi}^{t} (\dot{y} - \alpha u - K_p e)\, dt\right], \tag{4.51}$$

which is quite different from the traditional model identifications. The parameter ψ depends on the sampling frequency and noise intensity and it must assume a very small value. Here, a sampling frequency of 10 Hz is used and $\psi = 0.1$ seconds. The great advantage is the exclusive need to tune two more constants[3] and obtain a robust controller. The loop can be closed with any other i controller or higher order ultra-local models. In any other case, the changes in equation 4.51 are minor.

[3]The model-free control is robust to process changes without the need to retune any parameters or change the estimation of G_p.

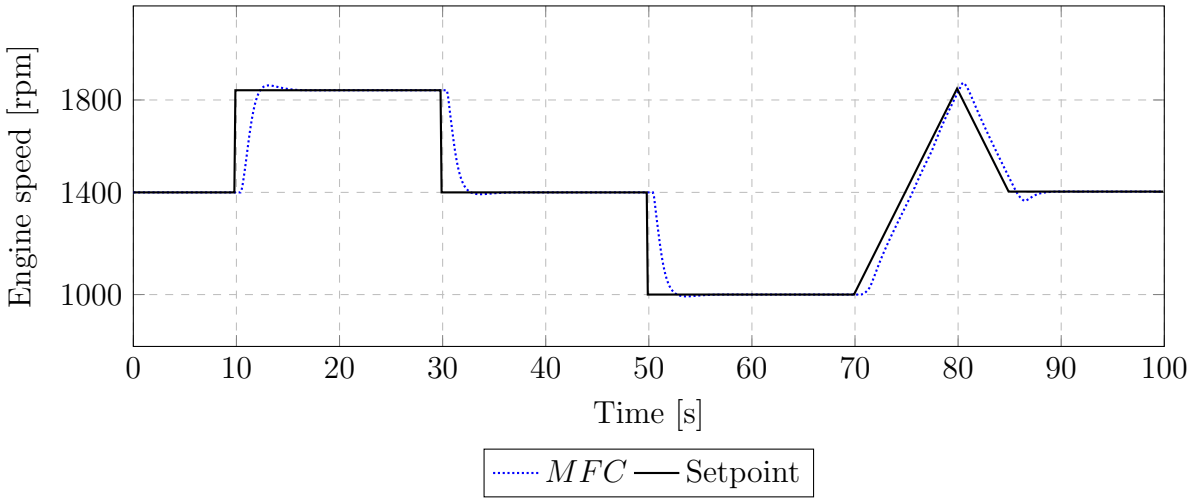


FIGURE 4.20: MFC setpoint following simulated response.

TABLE 4.15: MFC setpoint following performance indices.

	IAE	ISE	$ITAE$	$ITSE$
MFC	4.08	1.40	193.26	48.61

TABLE 4.16: MFC load disturbance performance indices.

	IAE	ISE	$ITAE$	$ITSE$
MFC	0.26	0.03	3.01	0.30

Simulation is presented in figure 4.20 and performance indices in table 4.15.

Similarly to other control structures a load disturbance was introduced, figure 4.21 and its response evaluated through performance indices, table 4.16.

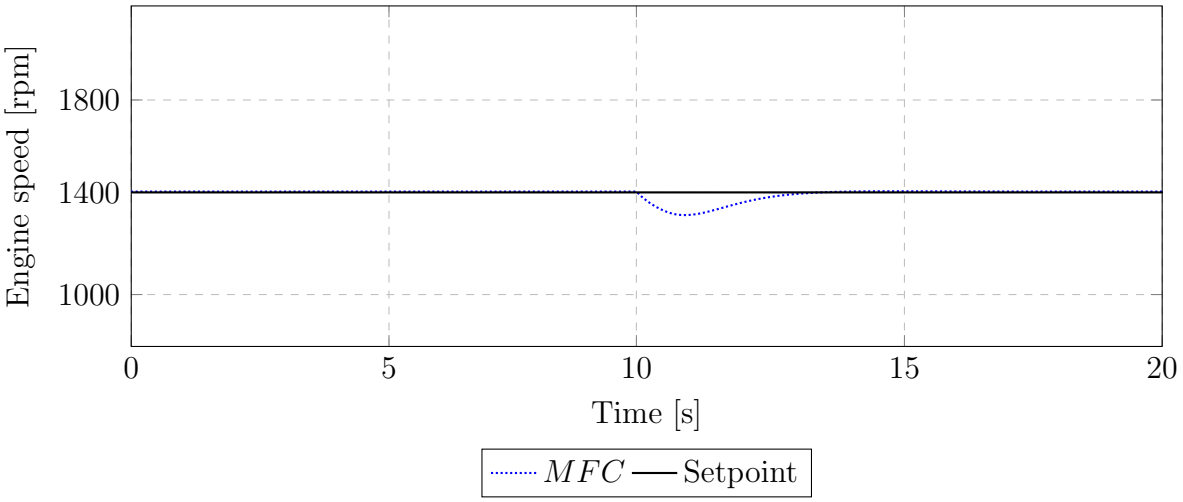


FIGURE 4.21: MFC load disturbance simulated response.

4.4 Comparison of simulation results

In this chapter simulations were carried out for different controllers and tuning rules. A comparison was carried out within each controller subsection in order to select the best performances, evaluated through performance indices, and now an overall one is presented.

The selected controllers are based on different working principles. Feedback control is based on the difference between the setpoint and process output. Tuning can be achieved in different ways. However, only tuning rules that are based on the model of the process were used. Even so, several tuning rules were developed based on distinct concepts. Model parameters are used to estimate PID gains.

Cascade control is similar to feedback. The difference between the setpoint and process output determines, along with the PID gains and process characteristics, the closed-loop response to setpoint changes and load disturbances. Tuning rules that are simple to implement and similar to the feedback ones were harder to find.

IMC is a control structure based on the model of the process which is not used to estimate tuning parameters but instead to control the process. Based on this approach some control structures were developed, even though only one is analysed here.

SPFF is introduced because it uses feedforward action to provide completely different responses to setpoint changes and load disturbances, allowing a more unrestricted

TABLE 4.17: Best setpoint following performance indices for each control structure.

	IAE	ISE	$ITAE$	$ITSE$
VV-M	4.18	1.34	207.59	48.99
	↑0.37 %	↓0.72 %	↓5.33 %	↓5.45 %
CC-SP	4.19	1.33	196.64	46.35
	↑11.05 %	↑13.50 %	↑26.39 %	↑32.03 %
IMC	4.65	1.51	248.30	61.17
	↑35.74 %	↑36.47 %	↑17.83 %	↑27.04 %
SPFF-SP	6.31	2.06	292.39	77.7
	↓35.38 %	↓32.02 %	↓33.99 %	↓37.40 %
MFC	4.08	1.40	193.26	48.61

tuning. The introduction of a Smith Predictor is proposed. In both structures time delay is approximated as being a pure one and through a first order Padé approximation. The results are very similar.

Finally, MFC is based on the estimation of the unknown parts of the process in real time. It is the only controller that is not based on previous mathematical modelling of the process and model is not used to tune or control the process.

Performance indices evolution throughout this chapter are presented in table 4.17 in the order they were simulated. The best overall performance is based on MFC structure where the main difference between it and VV-M is the capability of the former to eliminate the error to ramp inputs, which reduces performance indices considerably.

Regarding load disturbance rejection the performance indices are presented in table 4.18. The best load disturbance rejection is achieved while using the cascade control structure. The use of a Smith predictor decreased the performance for load disturbance rejection, specially in the proposed SPFF-SP structure.

Table 4.18: Best load disturbance rejection performance indices for each control structure.

	IAE	ISE	$ITAE$	$ITSE$
VV-M	0.20	0.02	2.23	0.22
	↓20.00 %	↓50.00 %	↓44.65 %	↓45.54 %
CC	0.16	0.01	1.48	0.12
	↑162.57 %	↑300.00 %	↑214.93 %	↑258.39 %
IMC	0.26	0.03	3.18	0.31
	↓23.12 %	↓33.33 %	↓29.91 %	↓29.15 %
SPFF	0.2	0.02	2.23	0.22
	↑130.03 %	↑150.00 %	↑135.00 %	↑136.47 %
MFC	0.26	0.03	3.01	0.30

Cascade control without a Smith predictor has the best load disturbance rejection. The VV-M is slightly worst, and of course is exactly the same as SPFF, being followed by the MFC.

It is worth noting that even though some control structures do not have a better performance, they can have other advantages that were not focused within this performance evaluation.

Chapter 5

Experimental results and analysis

The control structures and tuning that had the best performance in the simulations are now put to test and compared.

5.1 Implementation

Before entering into the experimental results some details regarding the PID algorithm, the software developed for real time plotting and the designed hardware are shown.

5.1.1 PID algorithm

The implemented algorithm is based on the Arduino library [103] where the following was already developed:

- Constant and controllable sampling frequency;
- Setpoint weight on derivative action;
- Bumpless changes in the tuning parameters while running;
- Anti-windup through clamping;
- Direct and reverse acting.

The bulk of the library is used in all control structures due to its robustness and organization.

The sampling frequency in the original algorithm cannot be higher than 10 Hz for a typical feedback control loop, i.e., it includes a single PID controller with one analog read (to obtain the sensor output), one PWM output signal and one value through serial. The sampling frequency is inversely proportional to the the computational load enforced on the Arduino. Within all the tasks, the analog read takes roughly 70 % of the elapsed algorithm time. If a cascade control is used the maximum sampling frequency decreases to roughly 5 Hz.

In an attempt to increase the sampling frequency some variables types and loops were replaced in order to reduce the code size and, with it, reduce computation time. The optimization was achieved through trial and error with the help of published Arduino code optimizations, most of them also carried out through trial an error [104–109]. The major source of improvement is carried out on the time required to perform an analog read [110]. After the optimization it is possible to use a sampling frequency of 80 Hz, instead of 5 Hz, in a structure with two PID algorithms, two analog reads, 2 PWM output signals, send two values through serial and change, one at a time, both PID gains and setpoints. Sampling frequency was established in chapter 4 and is 10 Hz.

The anti-windup method was changed from clamping to back-calculation and tracking, which was presented on chapter 4.

5.1.2 Real time plot

On the computer a PYTHON script was developed to receive multiple variables and simultaneously save them on separated arrays and plot them in real time on a single plot. The real time plot, figure 5.1, presents a delay that could not be eliminated. Furthermore, it is quite stiff since the number of variables, its format and even USB port must be manually introduced.

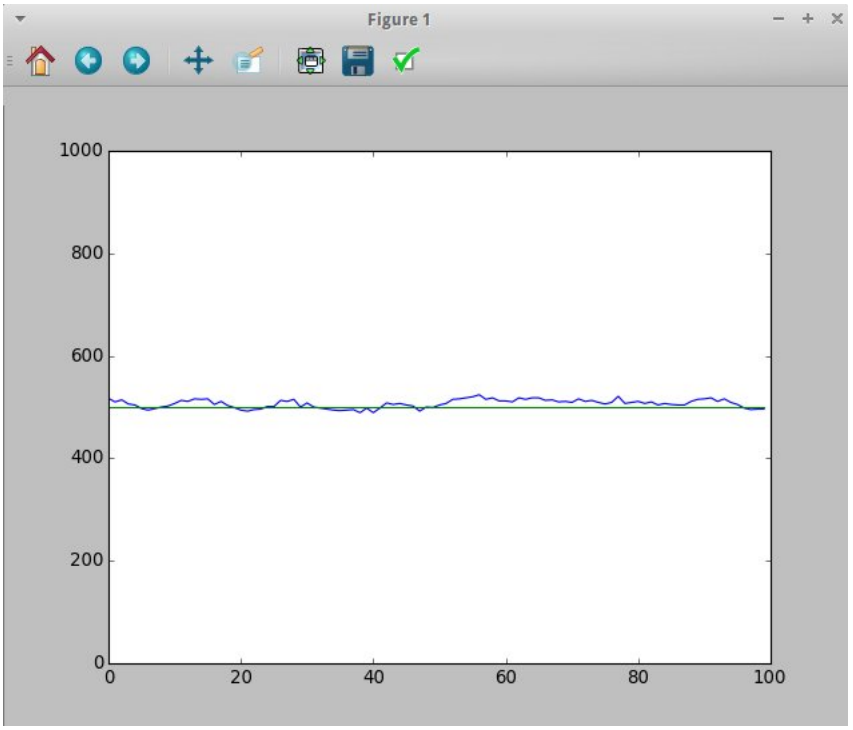


FIGURE 5.1: Real time plotting software.

5.1.3 Hardware

An interface based on an Arduino Mega, figure 5.2, was developed to meet the following criteria:

- Read the potentiometer and tachometer output voltage;
- Supply electric power to d.c. electric motor responsible for moving the air supply valve;
- Supply electric power to the d.c. generator which allows introducing a resistive load on the pneumatic engine;
- Send information through serial connection to a computer.

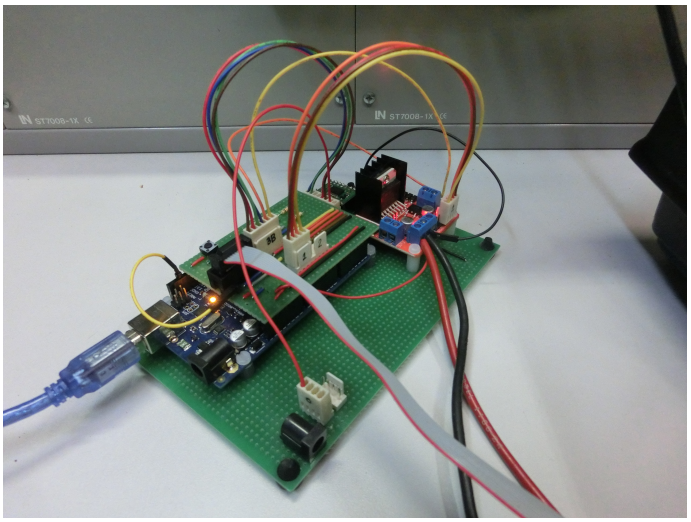


FIGURE 5.2: Hardware developed to interface with the CE107.

5.2 Experimental results

In this section the controllers with the best theoretical performance are implemented on the pneumatic engine. Setpoint following test is carried out and performances are mathematically evaluated using performance indices. A comparison between simulations and experiments is accomplished.

5.2.1 Feedback control

The implementation of a simple feedback control loop is carried out on the complete system tuned with VV-M. The experimental result are presented in figure 5.3 and performance indices in table 5.1.

TABLE 5.1: VV-M experimental performance indices.

	IAE	ISE	$ITAE$	$ITSE$
VV-M	28.86	10.41	498.5	177.88

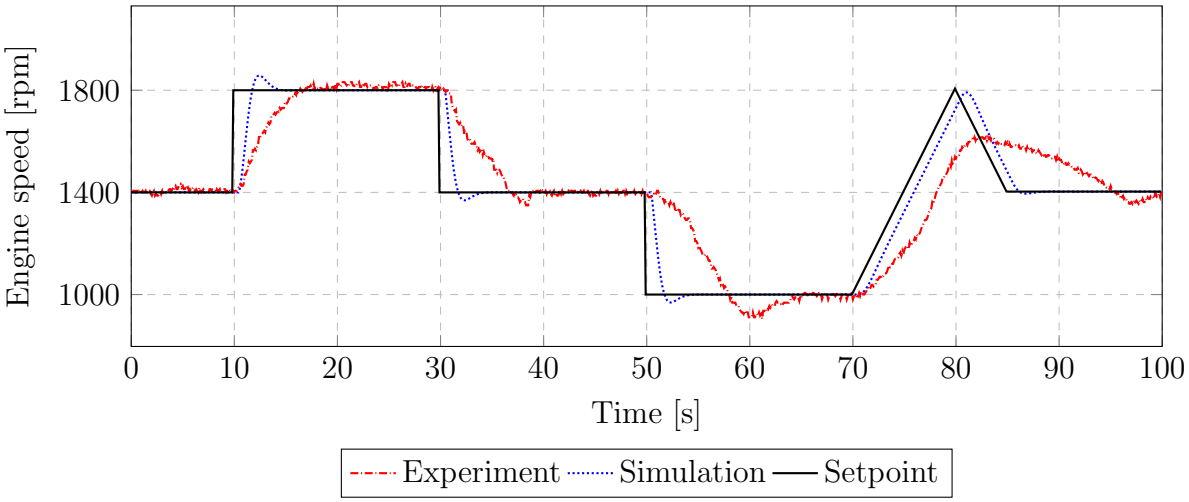


FIGURE 5.3: VV-M experimental results.

5.2.2 Cascade control

The cascade control structure with a Smith Predictor is implemented on the microcontroller and setpoint following is carried out. All air supply valve non-linearities are transferred to the air supply valve controller. The experimental result are shown in figure 5.4 and the respective performance indices in table 5.2.

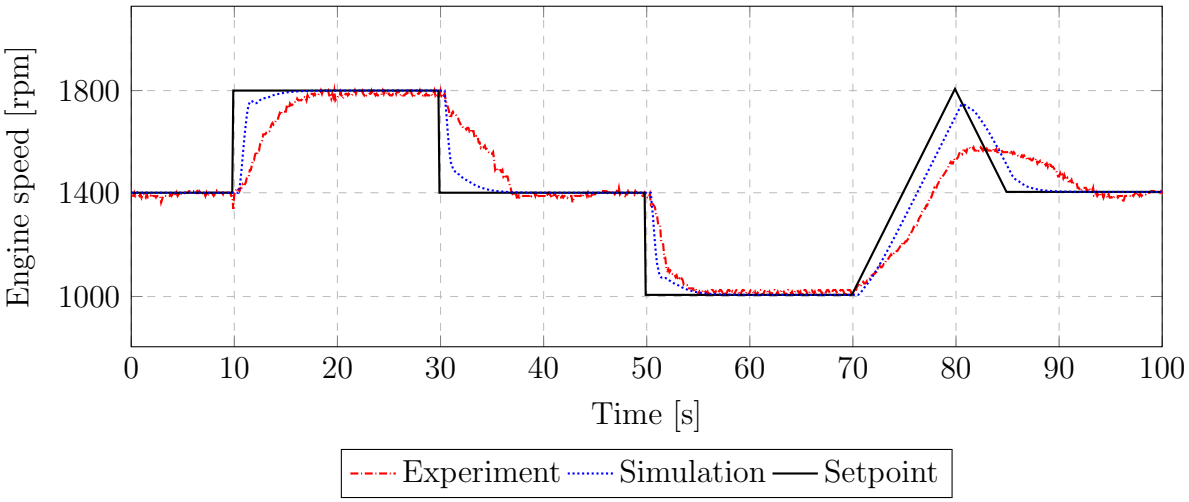


FIGURE 5.4: CC-SP experimental performance indices.

TABLE 5.2: CC-SP experimental performance indices.

	IAE	*ISE*	*ITAE*	*ITSE*
CC-SP	25.22	7.34	419.8	118.46

5.2.3 Internal Model Controller

The Internal model controller is implemented. Anti-windup is executed through clamping and not through back-calculation and tracking because no PID gains are present. The experiment was carried out with IMC for $T_f = 0.45$. The results are presented in figure 5.5 and the performance indices on table 5.3.

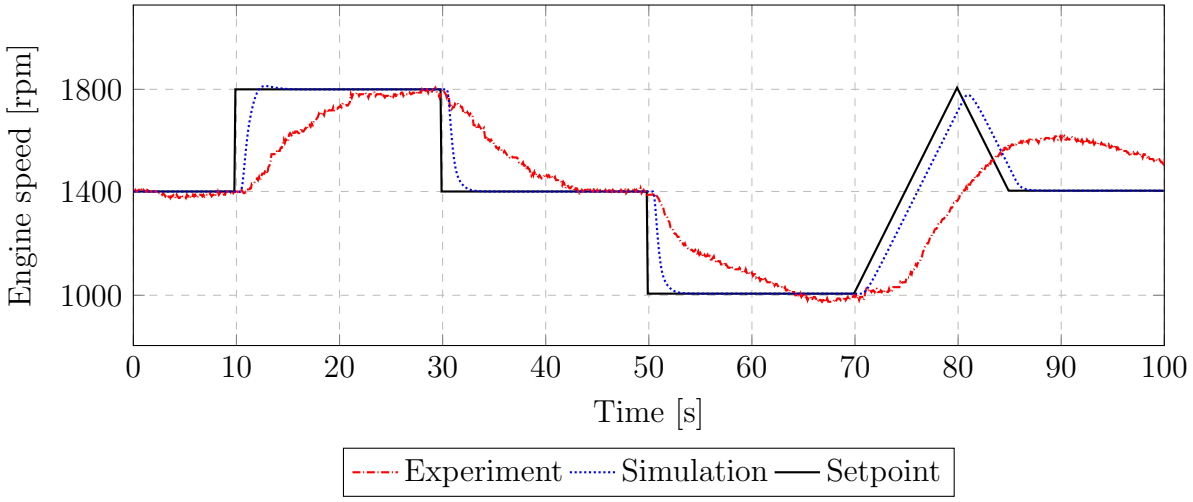


FIGURE 5.5: IMC experimental results.

TABLE 5.3: IMC experimental performance indices.

	IAE	ISE	$ITAE$	$ITSE$
$T_f = 0.45$	38.47	15.07	524.46	193.04

5.2.4 Feedback and Feedforward

The use of feedback and feedforward to improve setpoint following is implemented. The SPFF-SP control structure is used and the model inverse time delay was approximated as being a pure one. The results are in figure 5.6 and table 5.4.

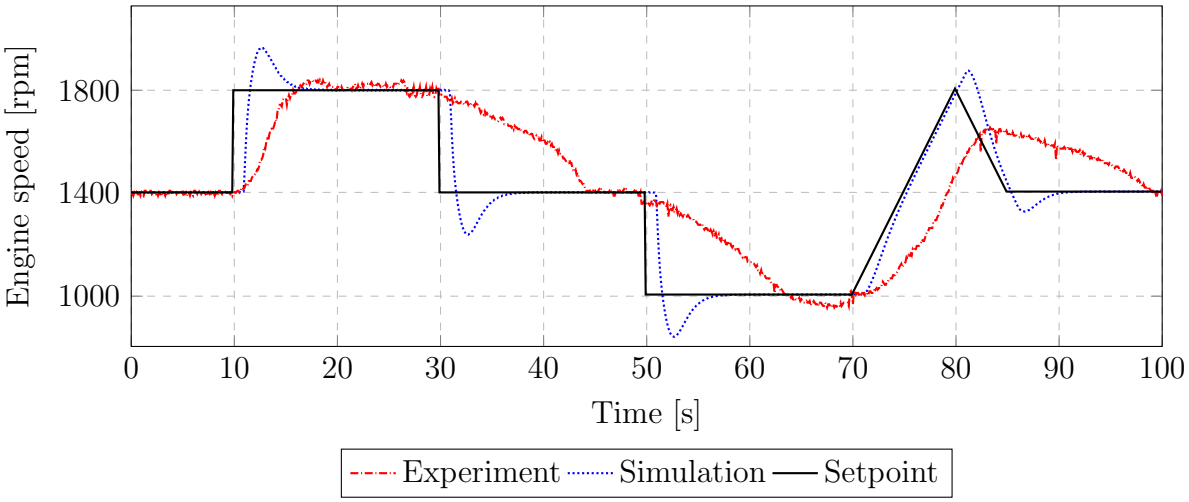


FIGURE 5.6: SPFF-SP experimental results.

TABLE 5.4: SPFF-SP experimental performance indices.

	IAE	*ISE*	*ITAE*	*ITSE*
SPFF-SP	57.45	18.26	594.12	201.02

5.2.5 Model-Free controller

Model-free controller is implemented and the results are in figure 5.7 and table 5.5.

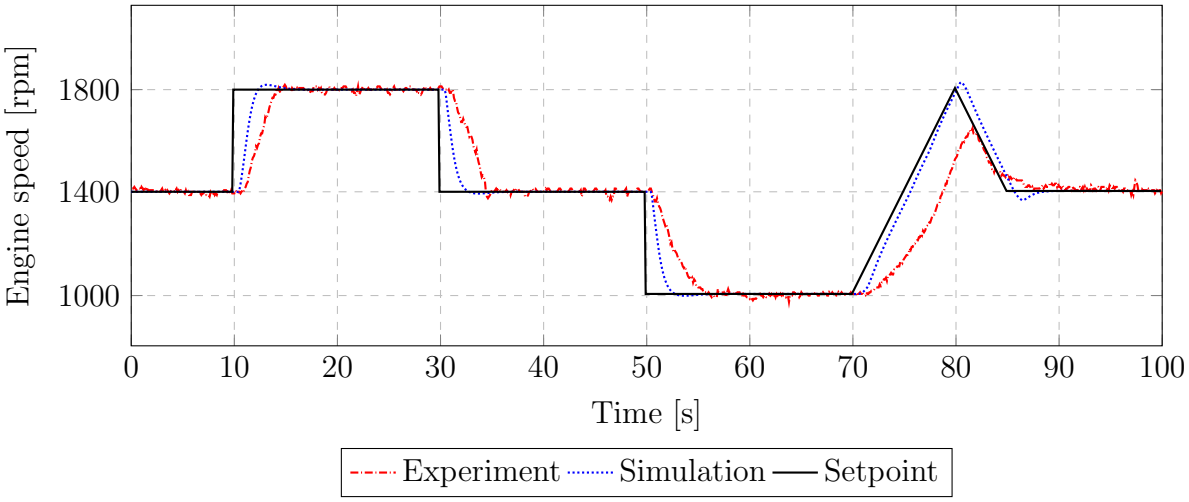


FIGURE 5.7: MFC experimental results.

Table 5.5: MFC experimental performance indices.

	IAE	ISE	$ITAE$	$ITSE$
MFC	15.33	5.47	322.15	139.71

5.3 Analysis of experimental results

Looking at the experimental data, the process is much slower and reacts with less overshoot than expected. Furthermore, simulated control structures that could eliminate the steady state error during ramp inputs failed to do so in the experiments carried out. This could be due to the higher than estimated time constant of the process and the ramp slope is too high. Even so, no control structure gave any sign that it would eliminate the error under the tested conditions.

When analysing the response to step inputs it is possible to verify that even though they all have the same magnitude the transient differs. Usually, the time constant is higher when the speed is changed from 1400 rpm to 1000 rpm than from 1800 rpm to 1400 rpm. The unexpected result gave rise to the following test, which purpose is to analyse the relationship between the magnitude of the step inputs on the air supply valve and the change in the pneumatic engine speed. The air supply valve was subjected to a sequence of step inputs, figure 5.8, with an amplitude increased by 33 %. If the relationship between the valve position and engine speed is linear, the steady state gain and time constants should be the same. However, the steady state gains do not increase in a constant way. Moreover, the time constant, i.e., the time it takes the process to reach 63.2 % of its steady state value, changes with the valve position. There is not a linear relationship between the amplitude of the valve input signal and the output signal of the pneumatic engine speed. The changes in the gain and time constant could be the reason of such discrepant behaviour between simulations and experiments. It is expected that an input signal with an amplitude 33 % would move the valve from one extremity to the other. However, this phenomena was not verified and therefore the electric motor does not behave as an integrator. The torque supplied by the electric motor is not high enough to smoothly move the air supply valve.

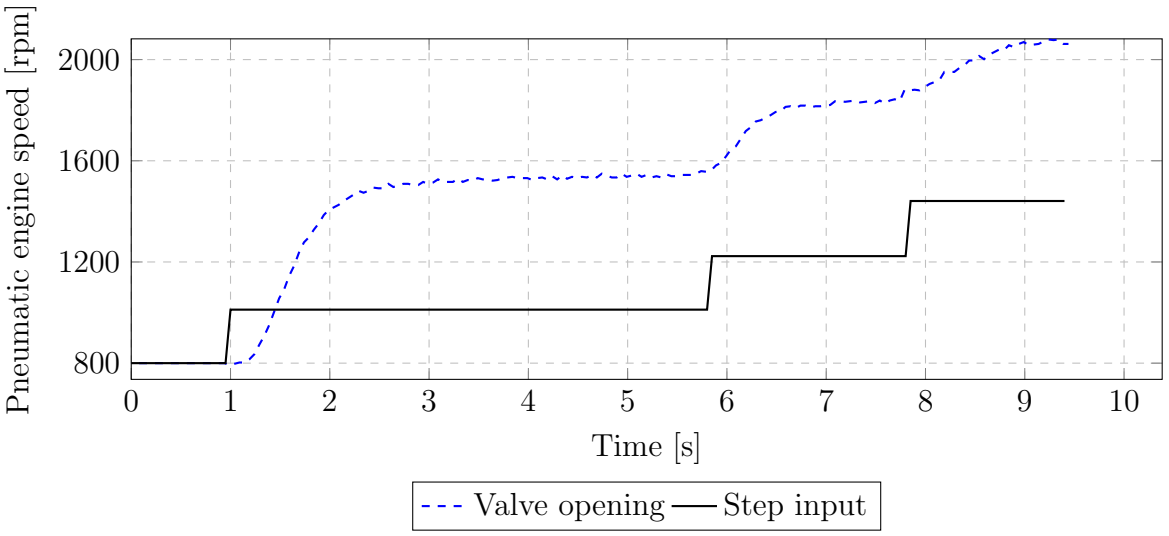


FIGURE 5.8: Actuator response to a sequence of step inputs.

Five control architectures were implemented and performance indices were computed for each of them and are presented in table 5.6. The best performance is obtained when using the MFC. The MFC experimental response is also the most similar to the simulated ones. There is an improvement on the performance indices when CC-SP is used instead of VV-M. Using IMC resulted in a poor performance, which can be due to the discrepancy between the process and its model.

To conclude, the MFC was the best control architecture, according not only with performance indices but also when comparing the simulated response with the experimental one.

TABLE 5.6: Experimental setpoint following performance indices for all implemented control architectures.

	IAE	ISE	$ITAE$	$ITSE$
VV-M	28.86	10.41	498.53	177.88
	↓12.70 %	↓29.49 %	↓15.78 %	↓33.40 %
CC-SP	25.22	7.34	419.84	118.46
	↑52.54 %	↑105.31 %	↑24.92 %	↑62.96 %
IMC	38.47	15.07	524.46	193.04
	↑149.34%	↑121.12 %	↑113.18 %	↑104.13 %
SPFF-SP	57.45	18.26	594.12	201.02
	↓73.31 %	↓70.04 %	↓45.78 %	↓30.50 %
MFC	15.33	5.47	322.15	139.71

Chapter 6

Conclusions and future work

6.1 Conclusions

In chapter 3 the description of the system and modelling of its main components are carried out. The air supply valve and pneumatic engine were approximated as first order processes. The SOPTD approximation to the step input response trough the method of moments is significantly better than the Shinskey method, even though it is assumed in both of them that the air supply valve contributes with one pole and the pneumatic engine with the other.

In chapter 4 a description of the algorithm, control structures and tuning rules is presented. Simulations are carried out on MatLab®\Simulink®. The performance is evaluated through the performance indices IAE, ISE, ITAE and ITSE. A feedback, cascade, cascade with Smith, IMC and MFC control structures are simulated and tuned. Complete segregation to setpoint and load disturbance rejection response is carried out. A change in this control structure is proposed. It consists on introducing a Smith predictor in an attempt to attenuate the effects of the process time delay. In both structures, simulations showed that approximating the time delay as a pure one or through a first order Padé approximation had little impact on the performance. The introduction of the Smith predictor slightly improved the response to setpoint following but considerably decayed the response to load disturbance. The MFC has a top response and the main difference between it and VV-M is its capability to eliminate offset errors during ramp inputs.

Chapter 5 is exclusively dedicated to the experiments and analysis of its outcomes. MFC provides the best experimental performance. Cascade control has a

better response than VV-M. In order to attempt to explain the discrepancy between simulations and experiments the air supply valve was subjected to another test that revealed its non-linear behaviour. The fitted electric motor responsible for moving the air supply valve needs to be replaced by another with higher available torque. The lack of torque, and therefore a slower response of the system, can be the reason of the high discrepancy between the simulations and experiments.

6.2 Suggestions for future work

First of all, an upgrade on the apparatus is proposed, more specifically on the d.c. electric motor responsible for moving the air supply valve. Also, an upgrade on hardware used in this work is recommended because studying load disturbance responses is of utmost importance to control engineering and this was only carried out in simulations.

Within the pneumatic engine used for this work there are some topics that can be studied with more detail, such as:

- Improvement of the model;
- Performance comparison on anti-windup algorithms;
- 2DOF PID control with variable setpoint weights;
- Use a velocity algorithm and compare with the position one used here;
- Test the robustness of the control loops by introducing changes in the process, such as changes in the air pressure and voltage supplied to the air supply valve electric motor.

Performance comparison in a more generic way can easily become complex. Even so, more simulations and experiments can be interestingly executed on the following topics:

- Compare the control structures SPFF and SPFF-SP on a wide range of normalize dead time constant for processes of different orders, unstable ones and non-minimum phase systems;

- Try to find a way to estimate the setpoint weights b and c on the 2DOF PID controller, regardless of the method used to tune it.

Finally, the software developed was exclusively tested on Ubuntu Linux. Considering that most use a different operating system a conversion might be needed. The software cannot plot, in real time, more than two variables. A new approach to real time plotting is needed as well as some development in order to make it closer to the plug-and-play experience.

Bibliography

[1] S. Bennett. A history of control engineering 1800-1930. *London: Peter Peregrinus Ldt*, 1979.

[2] John P. Shaw. Andrew Meikle (1719–1811), Millwright and inventor of the threshing machine., 11 2014. URL `http://www.oxforddnb.com/view/printable/18516`.

[3] James Watt Steam Engine Diagram, 03 2015. URL `https://manufacturingpearls.files.wordpress.com/2011/11/centrifugal-governor.jpg`.

[4] A. T. Fuller. The early development of control theory. *ASME: J. Dyn. Syst. Meas. Control*, 38:109–118, 224–235, 1976.

[5] J.C. Maxwell. On Governors. *Proceedings of the Royal Society*, 1868.

[6] Karl J. Åström and P.R. Kumar. Control: A perspective. *Automatica*, 50:3–43, 2013.

[7] Franklin Powell and Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall, 1986.

[8] Kiran Nath. Operational Information The MAN B&W MC Engine VIT Fuel Pump, 11 2014. URL `http://www.scribd.com/doc/46037096/Man-b-w-Mc-Engine-Vit-Fuel-Pump`.

[9] Matúš Kopačka, Peter Šimončič, Jozef Csambál. Real-time Air/Fuel Ratio Model Predictive Control of a Spark Ignition Engine. *Proceedings of the 18th International Conference on Process Control*, page 57–462, 2011.

[10] ARTEMIS. Dual Fuel Management, 11 2014. URL `http://www.heinzmann.com/en/engine-and-turbine-management/dual-fuel`.

[11] Rolf Isermann. *Engine Modeling and Control: Modeling and Electronic Management of Internal Combustion Engines*. Springer-Verlag Berlin Heidelberg, 2014.

[12] Karl J. Åström and Tore Hägglund. Revisiting the Ziegler-Nichols step response method for PID control. *Process Control*, 14:635–650, 2004.

[13] Hélène Panagopoulos. *PID Control. Design, Extension, Application*. PhD thesis, Lund Institute of Technology, 2000.

[14] S Bennett. The past of PID controllers. *Annual Reviews in Control*, 25:43–53, 2000.

[15] W. L. Bialkowski. Dream vs. reality - a view from both sides of the gap. *Pulp & Paper Canada*, 11:19–27, 1994.

[16] S. Yamamoto and I. Hashimoto. Present status and future needs: the view from japanese industry. *In Chemical Process Control*, TX, 1991.

[17] A. Ingimundarson and Tore Hägglund. Robust automatic tuning of an industrial PI controller for dead-time systems. *IFAC Workshop of Digital Control. Past, present and future of PID Control*, pages 149–154, 2000.

[18] G.P. Liu and S. Daley. Optimal-tuning PID controller design in the frequency domain with application to a rotary hydraulic system. *Control Engineering Practice*, 7:821–830, 1999.

[19] Mingwei Suna. PID pitch attitude control for unstable flight vehicle in the presence of actuator delay: Tuning and analysis. *Journal of the Franklin Institute*, 351:5523–5547, 2014.

[20] Jaques F. Smuts. *Process Control for Practitioners*. OptiControls Inc., 2012.

[21] Karl J. Åström and Tore Hägglund. *Advanced PID control*. ISA, 2005.

[22] Karl Johan Åström, Olof Garpinger, Tore Hägglund. Performance and robustness trade-offs in PID control. *Journal of Process Control*, 24:568–577, 2014.

[23] Todd A. Williamson. Modeling and implementation of pid control for autonomous robots. Master's thesis, NAVAL POSTGRADUATE SCHOOL. MONTEREY, CALIFORNIA, 2007.

[24] V. Romero Segovia, T. Hägglund, K.J. Åström. Measurement noise filtering for common PID tuning rules. *Control Engineering Practice*, 32:43–63, 2014.

[25] Aleksandar Micića, Miroslav Mataušekb. Optimization of PID controller with higher-order noise filter. *Journal of Process Control*, 24:694–700, 2014.

[26] Antonio Visioli. *Practical PID control*. Springer Science, 2006.

[27] De-Jin Wang. A PID controller set of guaranteeing stability and gain and phase margins for time-delay systems. *Journal of Process Control*, 22:1298–1306, 2012.

[28] Petr Hušek. PID controller design for hydraulic turbine based on sensitivity margin specifications. *International Journal of Electrical Power & Energy Systems*, 55:460–466, 2014.

[29] Sergei S. Mikhalevicha, Sergey A. Baydalia, Flavio Manentib. Development of a tunable method for PID controllers to achieve the desired phase margin. *Journal of Process Control*, 25:28–34, 2014.

[30] Yuan-Jay Wang. Determination of all feasible robust PID controllers for open-loop unstable plus time delay processes with gain margin and phase margin specifications. *ISA Transactions*, 53:628–646, 2014.

[31] Karl J. Åström and Tore Hägglund. *PID Controllers: Theory, Design, and Tuning*. ISA, 1995.

[32] Antonio Visioli. A new design for a PID plus feedforward controller. *Journal of Process Control*, 14:457–463, 2004.

[33] A. Wallen. *Tools for autonomous process control*. PhD thesis, Lund Institute of Technology, 2000.

[34] Isaac M. Horowitz. *Synthesis of Feedback Systems.* Academic Press, New York, 1963.

[35] M. Araki. PID control systems with reference feedforward (PID-FF control systems). *SICE Annual Conference*, page 31, 1984.

[36] M. Araki. On two-degree-of-freedom PID control systems. *SICE Research Committee on Modeling and Control Design of Real Systems*, 984.

[37] M. Araki. Two-degree-of-freedom control system: part i. *Systems and Control*, 29:649–656, 1985.

[38] K. Hiroi. Two-degree-of-freedom PID control system and its application. *Instrumentation*, 29:39–43, 1986.

[39] M. Araki and H. Taguchi. Two-Degree-of-Freedom PID Controllers. *International Journal of Control, Automation, and Systems*, 1:401–411, 2003.

[40] J.G. Ziegler, N.B. Nichols. Optimum setting for automatic controllers. *Trans. ASME*, 64:759–768, 1942.

[41] Min-Sen Chiu Jyh-Cheng Jeng, Wan-Ling Tseng. A one-step tuning method for PID controllers with robustness specification using plant step-response data. *Chemical Engineering Research and Design*, 92:545–558, 2014.

[42] Danijel Pavkovi, Siniš Polak, Davor Zorc. PID controller auto-tuning based on process step response and damping optimum criterion. *ISA Transactions*, 53: 85–96, 2014.

[43] Md Nishat Anwar, Somnath Pan. A frequency response model matching method for PID controller design for processes with dead-time. *ISA Transactions*, 55: 175–187, 2015.

[44] Md Nishat Anwar, Somnath Pan. A new PID load frequency controller design method in frequency domain through direct synthesis approach. *Electrical Power and Energy Systems*, 67:560–569, 2015.

[45] K.J. Åström, H. Panagopoulos, and T. Hägglund. Design of PI Controllers Based on Non-Convex Optimization. *Automatica*, 34:585–601, 1998.

[46] Neng-Sheng Pai, Shih-Chi Chang, Chi-Tsung Huang. Tuning PI/PID controllers for integrating processes with deadtime and inverse response by simple calculations. *Journal of Process Control*, 20:726–733, 2010.

[47] Q. Bi, W.-J. Cai, et al. Analytical IMC-PID design in terms of performance/robustness tradeoff for integrating processes: From 2-Dof to 1-Dof. *Journal of Process Control*, 24:22–32, 2014.

[48] Salvador Alcantara. *Analytical design of feedback compensators based on Robustness/Performance and Servo/Regulator trade-offs*. PhD thesis, UNIVERSITAT AUTONOMA DE BARCELONA, 2011.

[49] Guo-Qiang Zeng, Jie Chen, Min-Rong Chen, Yu-Xing Dai, Li-Min Li, Kang-Di Lu, Chong-Wei Zheng. Design of multivariable PID controllers using real-coded population-based extremal optimization. *Neurocomputing*, 151:1343–1353, 2015.

[50] Sheng Wu. State space predictive functional control optimization based new PID design for multivariable processes. *Chemometrics and Intelligent Laboratory Systems*, 143:16–27, 2015.

[51] R. Vilanova. IMC based Robust PID design: Tuning guidelines and automatic tuning. *Journal of Process Control*, 18:61–70, 2008.

[52] Aidan O'Dwyer. *Handbook of PI and PID Controller Tuning Rules*. Imperial College Press, 2006.

[53] Jens Graf. *PID control: Ziegler Nichols Tuning*. Sinus Engineering, 2013.

[54] Jeffrey E. Arbogast, Douglas J. Cooper. Extension of IMC tuning correlations for non-self regulating (integrating) processes. *ISA Transactions*, 46:303–311, 2007.

[55] Dola Gobinda Padhan, Somanath Majhi. Modified Smith predictor based cascade control of unstable time delay processes. *ISATransactions*, 51:95–104, 2012.

[56] Ibrahim Kaya. Obtaining controller parameters for a new PI-PD Smith predictor using autotuning. *Journal of Process Control*, 13:465–472, 2003.

[57] Vijay Bhaskar Semwal, Pavan Chakraborty, G.C. Nandi. Less computationall yintensive fuzzy logic (type-1)-based controller for humanoid push recovery. *Robotics and Autonomous Systems*, 63:122–135, 2015.

[58] Ruth Curtaina, Orest Iftime, Hans Zwartc. A comparison between LQR control for a long string of SISO systems and LQR control of the infinite spatially invariant version. *Automatica*, 46:1604–1615, 2010.

[59] Ridong Zhanga, Furong Gaob. An improved decoupling structure based state space MPC design with improved performance. *Systems & Control Letters*, 75: 77–83, 2015.

[60] Michel Fliess, Cédric Join. Model-free control, 2 2015. URL `http://arxiv.org/pdf/1305.7085.pdf`.

[61] Tufan Kumbasar, Ibrahim Eksin, Mujde Guzelkaya, Engin Yesil. Interval type-2 fuzzy inverse controller design in nonlinear IMC structure. *Engineering Applications of Artificial Intelligence*, 26:996–1005, 2011.

[62] Ibrahim Kaya. IMC based automatic tuning method for PID controllers in a Smith predictor configuration. *Computers and Chemical Engineering 28 (2004) 281–290*, 28:281–290, 2004.

[63] Sigurd Skogestad. Simple analytic rules for model reduction and PID controller tuning. *Journal of Process Control 13 (2003) 291–309*, 13:291–309, 2003.

[64] Axel Lødemel Holene. Performance and Robustness of Smith Predictor Control and Comparison with PID Control. Master's thesis, Norwegian University of Science and Technology, 2013.

[65] Peter Wolf. *Chemical Engineering Process Dynamics and Controls Open Textbook*. University of Michigan, 2013.

[66] Karl J. Åström and R.M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2006.

[67] *CE107 Engine Speed Control User Guide*. TecQuipment Ltd, 2009.

[68] *CE107 Engine Speed Control Safety Guide*. TecQuipment Ltd, 2009.

[69] David Zwick. Arduino Ball on Beam PID Balance, 04 2015. URL `https://www.youtube.com/watch?v=r5VaAP09BPA`.

[70] SpartanIIMark6. Arduino self balancing robot, 04 2015. URL `https://www.youtube.com/watch?v=_afq1DTAJZo`.

[71] strangedev. An arduino sketch for controlling a quadcopter. Uses RC and the MPU-6050 chip., 04 2015. URL `https://github.com/strangedev/Arduino-Quadcopter`.

[72] mihaipop11. Line following robot with PID algorithm, 04 2015. URL `http://www.instructables.com/id/Line-following-robot-with-PID-algorithm/`.

[73] Lon Glazner. PID Motor Control with an Arduino, 04 2015. URL `http://blog.solutions-cubed.com/pid-motor-control-with-an-arduino/`.

[74] James Bruce. Make Your Own Temperature Controller with an Arduino, 04 2015. URL `http://www.makeuseof.com/tag/make-your-own-temperature-controller-with-an-arduino/`.

[75] Karl J. Åström, Michael Athans, John Bailieul, Robert Bitmead, Peter Kokotovic, Michael J. Piovoso, Wilson J. Rugh. *The control handbook*. CRC press, 2000.

[76] Karl J. Åström and Tore Hägglund. The future of PID control. *Control Engineering Practice 9 (2001) 1163–1175*, 9:1163–1175, 2001.

[77] how to specify the sample time, 12 2014. URL `http://www.mathworks.com/help/simulink/ug/how-to-specify-the-sample-time.html`.

[78] H. Taguchi and M. Araki. Two-degree-of-freedom PID controllers - their functions and optimal tuning. *IFAC Workshop of Digital Control. Past, present and future of PID Control*, 2000.

[79] Karl Åström and L. Rundqwist. Integrator windup and how to avoid it. *Proceedings of the American Control Conference*, pages 1693–1698, 1989.

[80] G. Howes. Control of overshoot in plastic-extruder barrel zones. *EI Technology*, N.º 3:16–17, 1986.

[81] L.H. Dreinhofer. Controller tuning for a slow nonlinear process. *IEEE*, 8:2: 56–60, 1988.

[82] F.G. Shinskey. *Process-Control Systems. Application, Design, and Tuning 4th edition.* McGraw-Hill, New York, 1996.

[83] Tore Hägglund. A control-loop performance monitor. *Control Engineering Practice*, 3:1543–1551, 1995.

[84] Panda RC, Yu CC, Huang HP. PID tuning rules for SOPDT systems: review and some new results. *ISA Transactions*, 43:283–295, 2004.

[85] Q. Bi, W.-J. Cai, et al. Advanced controller auto-tuning and its application in HVAC systems. *Control Engineering Practice, 8, pp. 633-644.*, 8:633–644, 2000.

[86] M. Viteckova, A. Vitecek, and L. Smutny. Controller tuning for controlled plants with time delay. *Preprints of Proceedings of PID '00: IFAC Workshop on digital control*, pages 238–288, 2000.

[87] Koichi Suyama. A Single Design method for Sampled Data PID Control Systems with adequate step responses. *Proceedings of the Annual Conference of the IEEE Industrial Electronics Society*, pages 1117–1122, 1992.

[88] Raymond Gorez. New design relations for 2-DOF PID-like control systems. *Automatica*, 39:901–908, 2003.

[89] Sigurd Skogestad. Probably the best simple PID tuning rules in the world. *Jounal of Process Control*, 13:291–309, 2003.

[90] C.R. Madhuranthakam, A. Elkamel, H. Budman. Optimal tuning of PID controllers for FOPTD, SOPTD and SOPTD with lead processes. *Chemical Engineering and Processing 47 (2008) 251–264*, 47:251–264, 2008.

[91] T. Hägglund, K.J. Åström. Revisiting the Ziegler–Nichols tuning rules for PI control. *Asian Journal of Control*, 4:364–380, 2002.

[92] H. Panagopoulos, K.J. Åström, T. Hägglund. Design of PID controllers based on constrained optimisation. *IEE Proceedings - Control Theory and Applications*, 149:1:32–40, 2002.

[93] A. Wallén, K. J. Åström and T. Hägglun. Loop-Shaping Design Of Pid Controllers With Constant Ti/Td ratio. *Asian Journal of Control*, 4:403–409, 2008.

[94] Yongho Park and Moonyong Lee. PID controller tuning to obtain desired closed loop responses for cascade control systems. *Industrial and Engineering Chemistry Research*, 37:1859–1865, 1998.

[95] I. Kaya. Improving performance using cascade control and a smith predictor. *ISA Transactions*, 40:223–234, 2001.

[96] A. Mawire, M. McPherson. A feedforward IMC structure for controlling the charging temperature of a TES system of a solar cooker. *Energy Conversion and Management*, 49:3143–3154, 2008.

[97] Dazi Li, Fanyou Zeng, Qibing Jin, Lideng Pan. Applications of an imc based pid controller tuning strategy in atmospheric and vacuum distillation units. *Nonlinear Analysis: Real World Applications*, 10:2729–2739, 2009.

[98] Thomas F. Edgar Juergen Hahn, Thomas Edison. Adaptive IMC control for drug infusion for biological systems. *Control Engineering Practice*, 10:45–56, 2002.

[99] Pierre-Antoine Gédouin, Emmanuel Delaleau, Jean-Matthieu Bourgeot, Cédric Join, Shabnam Arbab-Chirani, Sylvain Calloch. Experimental comparison of classical pid and model-free control: position control of a shape memory alloy active spring., 2 2015. URL https://hal.inria.fr/inria-00563941/document.

[100] Michel Fliess, Cédric Join. Model-free control and intelligent PID controllers, 2 2015. URL http://www.sysid2009.org/plenaries/fliess.pdf.

[101] Michel Fliess, Cédric Join. Model-free control and intelligent PID controllers: Towards a possible trivialization of nonlinear control?, 2 2015. URL https://hal.inria.fr/inria-00372325/PS/St_malo_Pl_VF.ps.

[102] Frédéric Lafont, Jean-François Balmat, Nathalie Pessel, Michel Fliess. A model-free control strategy for an experimental greenhouse with an application to fault accommodation. *Computers and Electronics in Agriculture*, 110:139–149, 2015.

[103] Brett Beauregard. Improving the beginners PID, 1 2015. URL http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/.

[104] JustinHoMi. Topic: optimizing code?, 12 2014. URL http://forum.arduino.cc/index.php?topic=44924.0.

[105] Bill Earl. Optimizing SRAM, 12 2014. URL https://learn.adafruit.com/memories-of-an-arduino/optimizing-sram.

[106] Peter R. Bloomfield. What can I do if I run out of Flash memory or SRAM?, 12 2014. URL http://arduino.stackexchange.com/questions/221/what-can-i-do-if-i-run-out-of-flash-memory-or-sram.

[107] abcminiuser. GCC and the PROGMEM Attribute, 12 2014. URL http://www.avrfreaks.net/forum/tut-c-gcc-and-progmem-attribute?name=PNphpBB2&file=viewtopic&t=38003.

[108] Store data in flash (program) memory instead of SRAM, 12 2014. URL http://arduino.cc/en/Reference/PROGMEM.

[109] Define, 1 2015. URL `http://arduino.cc/en/Reference/Define`.

[110] jmknapp. Faster Analog Read, 2 2015. URL `http://forum.arduino.cc/index.php?topic=6549.0`.