Universidad Autónoma de Querétaro

Facultad de Informática

Developing an algorithm based on Soft and Adaptive computing for Direct Current Motors that Syntonizes Membership Functions to Smartly Control Motor Output

Tesis

Que como parte de los requisitos para obtener el Grado de

Doctor en Ciencias de la Computación

Presenta

M.T. A. José Alfonso Niembro Ceceña

Dirigido por:

Dr. Juvenal Rodríguez Reséndiz

Co-Director:

Dr. Ákos Odry

Queretaro, Qro. a 1 de diciembre de 2024

Universidad Autónoma de Querétaro

Facultad de Informática

Doctorado en Ciencias de la Computación

Developing an algorithm based on Soft and Adaptive computing for Direct Current Motors that Syntonizes Membership Functions to Smartly Control Motor Output

TESIS

Que como parte de los requisitos para obtener el grado de
Doctor en Ciencias de la Computación

Presenta:
M.T.A. José Alfonso Niembro Ceceña

Dirigido por:
Dr. Juvenal Rodríguez Reséndiz

Codirigido por:
Dr. Ákos Odry

SINODALES

Dr. Juvenal Rodríguez Reséndiz
Presidente

Dr. Ákos Odry
Secretario

Dr. Hugo Jiménez Hernández
Vocal

Dr. Fausto Abraham Jacques García
Suplente

Dr. Julio Alejandro Romero González
Suplente

Centro Universitario, Querétaro, Qro.
Diciembre 2024
México

This work is dedicated to my loved ones.

Resumen

Este trabajo presenta un método de control novedoso que tiene la intención de controlar la velocidad de la flecha de un motor de corriente directa mediante el uso de cómputo suave auto sintonizado y asistido de métodos adaptivos en la medida de lo posible. La idea detrás de este proyecto viene como una extensión de los estudios que este autor realizó para obtener el grado de maestría desarrollando un control difuso (control mediante el uso de lógica difusa) e incrementar la comprensión, además de estudiar en el estado del arte, que la sintonización de las funciones de membresía es realizada a prueba y error (método heurístico) con la finalidad de hacerlo rápido. Basado en estos eventos se encontró que hacer esto no es rápido y carece de precisión. Es entendido que por sí mismo el control difuso es tolerante a los cambios y tiende a ser benevolente con la acción de control, sin embargo, en el trabajo de investigación presente se desarrolló un concepto con el que se busca implementar un mecanismo inteligente computacionalmente para tomar la decisión de usar un tipo de control difuso u otro (basado en el método de Takagi-Sugeno). Finalmente, se implementó esta propuesta en un microcontrolador de 8-bits para comprobar la factibilidad de incluir este método como una alternativa de control, pero principalmente con la idea de realizar las pruebas y presentar los resultados comparativos.

Palabras clave: Motor de corriente directa, control de motores, cómputo suave, control difuso.

Abstract

This work presents a novel control method that intends to control the shaft speed of a direct current motor by using self-tuned soft computing and assisted by adaptive methods when possible. The idea behind this project comes as an extension of the studies that this author carried out to obtain the master's degree developing fuzzy control (control through the use of fuzzy logic) and increasing understanding, in addition to studying the state of the art, that the tuning of the membership functions is carried out by trial and error (heuristic method) in order to do it quickly. Based on these events it was found that doing this is not fast and lacks precision. It is understood that by itself fuzzy control is tolerant to changes and tends to be benevolent with control action. However, in the present research work, a concept was developed which seeks to implement a computationally intelligent mechanism to take the decision to use one type of fuzzy control or another (based on the Takagi-Sugeno). Finally, this proposal was implemented in an 8-bit microcontroller to verify the feasibility of including this method as a control alternative, but mainly with the idea of performing tests and presenting comparative results.

Keywords: Direct Current Motor, Motor Control, Soft Computing, Fuzzy Control.

Acknowledgements

I want to thank to Consejo Nacional de Humanidades Ciencia y Tecnología (CONAHCYT) for the support they have granted to me. In addition, I want to thank to Universidad Autonoma de Queretaro (UAQ) for accepting this project as part of the research projects cluster and Dra. Ana and Dra. Diana for all the support. I want to thank my director Dr. Juvenal and co-director Dr. Ákos for their great encouragement and advice, to my committee Dr. Hugo, Dr. Abraham and Dr. Julio for their valuable feedback during the reviews.

I want to thank my family, my father Alfonso and mother Emma for their great example, to my children Edgar, Pablo and Abraham for being an everyday motivation and my dear Roxanna for being who she is and staying with me.

General Index

Figures Index

Tables Index

CHAPTER 1

Introduction

Machines that are smart enough to mimic humans, this is a science fiction premise since while ago that makes people think that eventually machines will behave as human beings do. Adaptive, smart, soft computing, bio-inspired algorithms are concepts that are applied to different engineering disciplines; control engineering is not the exception. In the last 30 years research into topics for making the controllers adjust automatically, if not smartly, to changes have been made to produce this human-like type of behavior. In this work, it is not explored the fact that humans do the best work possible, it is assumed that this is the case, then making a machine to think as a human does is the correct instruction. When talking about bio-inspired algorithms, the researchers are not looking to replicate humans, but animals, especially, bee's colonies, wolf pack, ants' social structures and so on; these types of implementations are looking essentially the be as optimum as possible, based on the belief that nature does the best work possible maximizing benefit while using minimum effort. So, in general sense, it is thought that when being close to nature, in operation, it can produce the best results. It cannot be assured when this type of thinking started, but control engineering has its start while ago. The word "feedback" was introduced in 1920 at Bell Laboratories, and this was used equivalently to words "reset", in United Kingdom, and "closed cycle", in United States, the same year (Bennet, 2008). Regulation of movement was developed in the early stages of what we could think was the baseline of engineering processes in which the idea is to control the movement, for instance, of a windmill with specific purposes. All these ideas became fundamental parts of a bigger complex system that has been built nowadays, for instance, the James Web Space Telescope (JWST).

Control engineering is being considered a mature discipline in 1969 with the publication of Otto Mayr´s book titled, in English, as “The origins of feedback control”. This was extended from Germany to the United States and USSR. Starting at the twentieth century, the Control of the Systems and the foundational basis of the Control theory have become relevant to successfully achieve the automation and self-regulation of the processes. Since then, the Control systems theory has positively impacted the automatic process (Ogata, 2008). Control engineering includes Control Theory, if not considered synonyms, and Control Engineering involves various concepts that are worth mentioning:

- It is applied to Systems with a single input and a single output.
- It consists of three types of controllers, also known as the control laws, which are proportional, integral and derivative.
- It is designed to meet controller requirements (steady state error, overshoot and response/rise time).
- It is used majorly for deterministic Systems (those that can be represented mathematically); however, stochastic systems can be controlled.

James Watt pioneered the implementation of speed regulators to the steam engine in eighteenth century assistance by mechanical regulators, as an indication that the first control systems were conceived in a totally different manner to what can be seen nowadays. Ogata (2008) mentions that the methodologies to keep a System operating in a certain regime to keep up with the requirements in an automatic way was later named Automatic Control. Control Theory was based on significant contributions made in the first half of the twentieth century mentioned in Table 1.1.

Table 1.1.

Control Theory contributions from 1920 to 1950.

Year	Author	Contribution	Name of publication

1922	Minorsky	The stability of the system is found through the solution of the differential equations that define the system. For all the non-linear systems, the Taylor´s expansion can be applied to linearize them.	"Directional Stability of Automatic Steered Bodies"
1932	Nyquist	A procedure to determine stability in closed loop was designed and proposed considering a sinusoidal input in open loop. This work also introduced the use of the frequency domain rather than the time domain and added in the stability criteria determination.	"Regeneration Theory"
1934	Hazen	Defined the term "servomechanism" and contributed to its development as a feedback element in the system. It studied the implementation of a servomechanism to axis position.	"Theory of Servomechanism"
1934	Black	Achieved a gains system that behaved as a linear system despite the non-linearities intrinsically associated.	"Stabilized feedback amplifiers"
1940	Bode	Defined phase and gain margins and Bode's diagram developed by means of logarithmic graphic representations to highlight important control data.	"Relations Between Attenuation and Phase in Feedback Amplifier Design"
1942	Ziegler and Nichols	Defined a method to fine tune Classic PID Controller gains.	"Optimum Settings for Automatic Controllers"
1947	Harris	Transfer function concept introduced to a System that includes feedback elements.	"The analysis and design of Servomechanisms"

Note. Data collected by author on the 1st of November 2024, based on the analysis done of the publications mentioned.

This document follows a structure that allows the reader to follow the process according to logical methodology; the purpose is to let the reader walk on these pages to capture the information that was intended to be shown. The document includes the following sections:

- Chapter 1. This chapter presents introductory information for the reader to get contextualized in the research topic by including the problem statement, justification, the hypothesis, objectives and factual insights from the research.
- Chapter 2. Here the state of the art is introduced to show the latest and greatest research in subject matter, putting emphasis on soft computing, control and adaptive control and artificial intelligence.
- Chapter 3. This chapter includes the methodology followed by the author and an explanation of the problem is presented in more detail.
- Chapter 4. In this chapter, the materials and testing scheme are shown to let the reader understand what is done and how the hypothesis is tested.
- Chapter 5. This chapter is dedicated to showing the results of the experimentation as well as the discussion and findings of the research.
- Chapter 6. In this chapter the conclusions obtained are presented in addition to what is expected for continuing future research topics.

1.1 Problem Description

Direct Current (DC) motors are old, some can arguably reference the initial step to have functional DC motor was after Michael Faraday discovered electromagnetic induction in 1831, the backbone of the operation of this electromechanics device. It is widely spread and accepted that Faraday is the inventor of such magnificent technology; however, the first invention is credited to William Sturgeon who developed this device to power machinery using a commutator for the first time in 1832. Thomas Davenport was first awarded with the patent of the DC motor in 1837, he based his work on Sturgeon's although his invention showed reliability problems. Then, by 1887, Nicholas Tesla introduced the Alternating Current (AC) motor and its patent one year later (Tong, 2014). Today, the electric motors are manufactured in a wide variety of types to fit to different applications; Tong (2014) argues that by 2030 the energy consumption from these electric motors will rise to more than 13,000-terawatt hour (TWh). It is clear electric motors are essential to keep the lifestyle, and it is even more evident that there is interest in continuing the manufacturing of such energy conversion mechanisms.

With the operation of these motors, particularly the DC ones, few challenges are faced, starting with the shaft speed and shaft torque regulation. Different control strategies are used to complete the regulation task for both, starting from classic control theory and the wide used Proportional-Integral-Derivative (PID), soft computing assisted by Fuzzy Logic (FL), adaptive control, and bio-inspired algorithms for auto-tune of the PID gains, also known as applied Artificial Intelligence (AI). The Fuzzy Logic Control/Controller (FLC) is focusing on the adjustment of the PID Controller in some architectures; there are other FLCs, highly tolerant to non-linearities, that directly control the input to the DC motor, in other words that do the regulation of the shaft speed. It is important to mention one of the key factors is that the model that represents the motor is linearized. All controllers start with gains assuming linear models, the reality Is quite beyond this, after a period the DC motor operates, the speed regulation loses accuracy.

Specifically, all DC motors are prone to not meet control requirements while operating. Currently several techniques of adjustment are used, but not many of them are practical nor implemented commercially since the technicians normally adjust the gains at site. FLC is a control technology used widely but with the inconvenience MF is defined by experienced users heuristically. This research focuses on the idea of using basic Artificial Neural Network (ANN) to select between a different Membership Function (MF) shape to improve DC motor response and adapt to changes the DC motors experience in time.

1.2 Hypothesis

There is an ANN that can select a shape of a MF considering the error the shaft speed experiences while is in operation to keep this parameter within the Control requirements.

The error is defined as the absolute value of the arithmetic difference between the reference speed and the current speed (measured).

1.3 Justification

There are plenty of applications that are using DC motors in those days, the need of having a precise regulation is mandatory in some applications. DC motors are part of the automation process, mechatronic systems, robotics and much more. In the context of the Universidad Autonoma de Queretaro (UAQ), the Instrumentation and Control department in the Engineering Faculty does research in the Control of DC motors (brushed or brushless) by implementing different controls.

Interest is given to the use of FLCs in open architecture, implementation in FPGAs and other flexible gates arrangements, but also this is implemented in other chips. However, there is no identified method to determine and set the MFs of the FLCs, those are defined heuristically. This situation is something that is not ideal, primarily because there is no mechanism to evaluate if proposed control is good enough for the application. It is needed to determine a mechanism, standardize, that can add in favor of making the determination of the type of MFs more rigorously.

1.4 Objectives

1.4.1 General

It is desired to develop an algorithm that is used to select either one shape-type of a MF or another assisted by basic ANN, as an intelligent mechanism, to keep up with the Control requirements specified for the operation-type either speed regulation or position of the mechanism or both.

1.4.2 Specific Goals

Analyze the state of the art to figure out the latest and greatest methods and technology applied to speed and position regulation as applicable.

Understand and implement a FLC either on or off the operation line in DC motors and the application of the AI to set up these if any.

Determine the Control requirements and the testing scheme and scenarios for determining the capacity of the proposed Controller.

Develop and implement an algorithm and embedded code to test the proposed methodology, which shall operate online.

Collect data from testing, analyze and indicate if it fulfills Control requirements.

1.5 Impact and limitations

The following impacts are considered:

Innovation and technical innovation: This research work intends to add to the state of the art by completing a method to apply ANNs to make decisions based on system performance. Although some of the ideas expressed here can be improved and refined.

Advancing knowledge: Using any AI add value to making the Controller to be smart. It is necessary to investigate more to consider this as a promising technique that can contribute to Adaptive or Intelligent Control.

Informed decision-making: Auto-select from different MF shapes for FLC using an algorithm that bases its decision on data. This research exhibits initial steps toward this topic.

This work is limited in its goal, even though there is a desire to explore as much as possible, there are still limitations in knowledge, experience and infrastructure to complete a full theory made of this proposal.

1.6 Methodology

The methodology implemented in this research is based on Descartes' that dictates the following steps:

- Problem definition.
- Theoretical framework.
- Refinement of the ideas.
- Testing the hypothesis.
- Make conclusions.

CHAPTER 2

Background and State of the Art

For a long period, humans have considered themselves as thoughtful, a species that owns intelligence, and for several years humans have tried to understand the process behind thinking and how intelligence works. One of the fundamental questions is, how is it possible to think? So far, there is not a quite straightforward answer to this question. Artificial Intelligence faces this exact same unanswered question, but even more, how to recreate the intelligence into a machine? How to build smart not-human intelligence?

In recent years, the AI is used in several applications and it is seen as a solution for everything. There are benefits in using AI, but it is necessary to distinguish what AI does and does not do. However, the initial idea was different to what it is today. In the old days, McCulloch y Pitts were investigating a new way to do computing parallelly to what was investigated by Von Neumann and others. They realized that using a simplified model of a human neuron in the form of addition of inputs and thresholding them would be sufficient to mimic the real neuron. Rosenblatt did more research with this ANN, and he is credited to give the ability to learn to this. Later, MITs' professors Minsky and Papert brought the XOR incident (single layered ANNs are not able to learn simple mathematical functions), then AI was put in the freezer for several years (Howard & Gugger, 2020).

AI can be seen from different perspectives, at least there are two relevant to mention: thinking process and reasoning, and behavioral aspects. AI is compared with humans or compared against what is thought to be the best way to think, then AI decides on what and how to show depending on its learning. There are four classifications used described as follows:

- Acting as a human being (Turing´s test).
- Thinking as a human being (cognitive model).
- Thinking rationally (thinking laws).
- Acting rationally (rational approximation).

These classifications are widely used by people as collective knowledge to think what AI is capable of and this is where the implementation of AI struggles and does not support research and daily activities as thought it should mean to do it. Control Systems are not the exception in the race for applying AI to control and regulate a variable that is possibly controllable. In recent years, several research investigated the use of AI to update Controller gains.

It is convenient to review highlights for Control Engineering, Control laws and advancements in Control to deep dive into the subject.

2.1 Control Engineering Background

One of the most important contributors to control engineering is Norbert Wiener, who was a mathematician that worked with Bertrand Russell, and contributed to control of mechanical systems. Wiener and others developed mechanisms that were intentionally biased by using regulators to minimize the error. He published his book named “Cybernetics” in 1948 which introduced the concept of having machines which are artificially intelligent.

Jumping a bit from the beginning to the most modern in the use of AI in control theory, optimal control for stochastic systems has its objective in designing a system that maximizes the objective function over time and this makes an excellent connection with the definition that could be given to AI, which is: designing systems that behave optimally (Rusell, 2010).

Based on what was stated before, it is pertinent to go to the beginnings of control theory in order to understand a little more about this area of engineering that has played a fundamental role in production processes in recent years.

Control theory has its origins in the 18th century through the work of James Watt in his attempt to control the speed of the steam engine (something that became known as the centrifugal speed governor). From that moment on, one of the fields of knowledge began that has shown great usefulness in many topics of interest in the field of engineering and is known as systems control. Systems control has had a predominant relevance in the 20th century, it has served a lot for the development of the industry and has positively impacted automatic processes (Ogata, 2008).

First, it is imperative to make known some fundamental aspects of classical control theory, and these are, among others, the following:

- The control is used for one-input, one-output systems; however, it also applies to multiple-input, multiple-output systems.
- In classical control there are 3 types of control, they are also known as control laws, and these are: proportional, integral and derivative. All of them are applied to error.
- Systems control must meet certain performance requirements called control requirements, and in general, they refer to the system response time, type and percentage of damping, and percentage of error with respect to the reference signal.
- Control is preferably applied to deterministic systems; However, there are also techniques that are applied to stochastic systems.
- The control is based on the solution of the differential equations that represent the system.

James Watt's work to control the speed of the steam engine was based on the use of mechanical means. This is how the first control systems were developed, but currently, these are very different since mechatronics is used for their development and implementation. Ogata (2008) mentioned that at the beginning of the 20th century, work was done to find methods to keep systems in operation and meeting control requirements automatically and this is known as automatic control.

In classical control theory, feedback control systems can be classified in various ways, depending on the purpose of said classification. According to the analysis and design method, these are defined as: linear and non-linear and/or time-variant or invariant. According to the types of signals used in the system, these are in continuous time or discrete time or in modulated and non-modulated systems. They are often classified according to their primary purpose, for example a position control system or a speed control system.

The controlled variable is the quantity or condition that is measured and controlled. The manipulated variable is the quantity or condition that the controller modifies to affect the value of the controlled variable.

The purpose of a control system is to obtain a desired response from one or more variables within a system. This can be achieved by using an open loop system or a closed loop system. Closed loop systems are those where the control determines the input value, that is, it indicates the reference value for the process and uses the feedback signal to measure the output.

Feedback control is essential to keep process variables close to the desired values, regardless of whether the system is exposed to disturbances or changes in its dynamics (Visioli, 2006).

Proportional, integral and derivative (PID) control is a three-term controller that has a long history in the field of automatic control and has its beginnings at the beginning of the last century. Due to its characteristic simplicity and the fact that it is very intuitive, this controller has become the standard for many industrial applications.

Considering technological advances today, this type of controller is implemented electronically in almost all cases; contrary to what was done before with mechanical, pneumatic or electrical implementation.

As mentioned before, PID control consists of terms, and at this point it is convenient to rename them to control actions that are correspondingly: proportional, integral and derivative.

Proportional action is proportional to the current error of the controller and is defined as:

$$u_t = k_p e(t) = k_p[r(t) - y(t)] \quad \text{E1.}$$

Where k_p is the proportional gain, $e(t)$ is the error, also represented by the additive inverse of the reference value with the system output value. In terms of the controller, the proportional action can be written as:

$$c(s) = k_p \quad \text{E2.}$$

The integral action is proportional to the integral of the current error of the controller and is defined as:

$$u_t = k_i \int_0^t e(\tau)d\tau \quad \text{E3.}$$

Where k_i is the integral gain and the integrand is responsible for taking into account and integrating the previous error values. The presence of the pole at the origin of the complex plane, through integral action, allows the reduction of the steady state error to 0, even after the system experiences a disturbance. In other words, the control action can correct the value of the integration constant resulting from integrating from 0 to t. The integral action can also be written as:

$$c(s) = k_i \frac{1}{s} \quad \text{E4.}$$

Finally, the derivative action tries to consider the prediction of the value of the control error. Therefore, the derivative action is proportional to the derivative of the current error of the controller and is defined as:

$$u_t = k_d \frac{de}{dt} \quad \text{E5.}$$

Where k_d is the derivative gain. In terms of the controller, the derivative action can be written as:

$$c(s) = k_d s \tag{E6.}$$

The set of the three control actions, to form the PID controller, can be ideally represented as follows:

$$c(s) = k_p \left(1 + \frac{1}{T_i s} + T_d s\right) \tag{E7.}$$

Where k_p is the proportional gain, T_i is the integral time constant and T_d is the derivative time constant. It is convenient to define the following equivalences:

$$k_i = \frac{k_p}{T_i} \tag{E8.}$$

$$k_d = k_p T_d \tag{E9.}$$

Visioli (2006) defines this representation relates the integral and derivative gains to the Proportional gain; However, it is not the only representation.

Continuing with the relevant aspects of classical control theory, during the Second World War there were great advances in its application. During this period, the engineers focused on solving the shooting problem, which consists of detecting and tracking the target, predicting its trajectory, and placing the cannon in the firing position.

Substantial progress was also made in controlling the position of radars for locating and tracking targets, among some other elements and/or devices (Bennet S. , 1993).

Some of the most important contributions were made by:

- Brown (1946) in his work “Dynamic behavior and design of servomechanism.”
- Harris (1942) in his work “The frequency response of automatic control.”
- Hall (1946) in his work application of circuit theory to the design servomechanisms.”

- Weiss (1939) in his work “Constant speed control theory.”

To control the position and/or speed of the system, it is necessary to use feedback signals supplied by the control system itself using the correct instrumentation.

Feedback is done through a servomechanism. This term is used for a particular class of automatic control system that allows the system to be controlled to always follow a reference signal regardless of frequency and speed (Hall, 1946).

A servomotor is a servomechanism that consists of an armature, a commutator, brushes, a permanent magnet, and a detector. Basically, it is the normal constitution of a direct current motor, it operates under the same physical principles (a current that is introduced into a fixed winding that rotates the mobile winding placed on the rotor. To make it work as direct current, use is made of the brushes to commutate); However, it has the sensor that allows the angular position of the arrow to be obtained, and consequently, the speed can be calculated or measured directly using a tachometer. It is well known that the current injected into the motor produces a torque on the output shaft that is directly proportional through the so-called electromechanical constant of the motor (Suh, Kang, Chung, & Stroud, 2008).

As indicated before, a servomotor is capable of being controlled when it has a feedback system; therefore, it is considered a closed loop. Currently, there are different methodologies for the control of servomotors, although at an industrial level reliability and proven, functional and simple techniques are sought such as proportional, proportional and integral, proportional and integral and derivative control or variants not mentioned.

Currently you can see different classical control proposals; However, they make use of sophisticated implementations, including embedded software for the servomotor.

Ogata (2008) mentions that technological advances have allowed control to reach new dimensions and functionalities while preserving the classic control theory that has proven to be effective, simple and functional.

2.2 Contributions in Control Engineering

Some of the most recent contributions related to servomotor control are listed in Table 2.1 shown next.

Table 2.1.

Recent contributions to servomotor control.

Year	Author	Contribution
2008	Zhou	The proposal includes the construction of a “driver” with the purpose of being able to embed the control code using the proportional, integral and derivative laws.
2015	Rathore, Singh, Chauhan	The implementation of a method to minimize the error in the calculation of the PID gains of a controller is proposed and the results are compared against the traditional Ziegler-Nichols methodology.
2016	Salim Qureshi, Swarnkar, Gupta	Use of a robust control (sliding model) and a proportional and integral control to understand the variation of the parameters and their behavior with disturbances, in the same way, verify that the proposed control strategy works to maintain the desired behavior.

Note. Data collected by author on the 1st of November 2024, based on the analysis done of the publications mentioned.

Among modern control techniques is adaptive control. This is used in systems where most of the parameters are unknown, it even has the peculiarity of calculating some of them when establishing the control action.

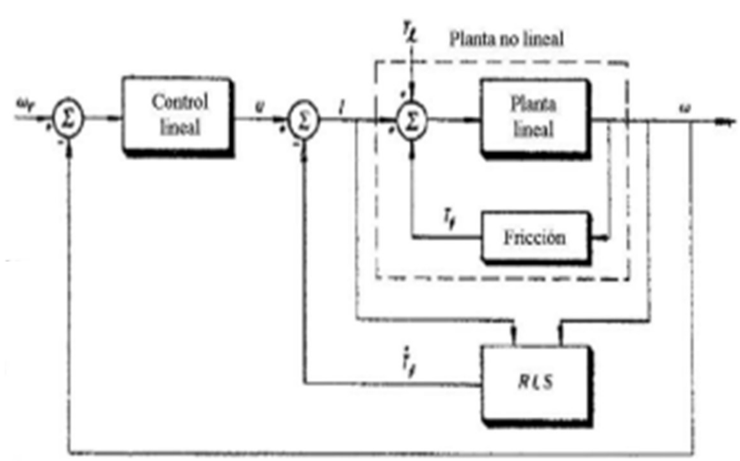
Some methodologies use a control law with difference equations, taking as a starting point the second-order model for control systems that involve the natural frequency and the damping coefficient. The linear part is the direct current servomotor model and its closed loop with the friction model becomes the nonlinear part.

One example proposes a control scheme where the non-linear effects of friction are adaptively compensated. It is established that a standard method can be implemented to design the regulator of said plant, where the regulator must be understood as the control. They propose a combination of fixed linear control and an adaptive part that compensates for the nonlinear effects of friction.

It is mentioned that friction is antisymmetric, whether static or dynamic. This characteristic depends on the direction of rotation, as established. This particularity is considered in the mathematical model used in this control proposal. Friction is a phenomenon that is present in mechanisms, and in many cases, affects the performance of systems (Canudas, Amströn, & Braun, 1986), as shown in Figure 2.1.

Figure 2.1.

Block diagram of the proposed control of a DC motor with adaptive friction compensation.



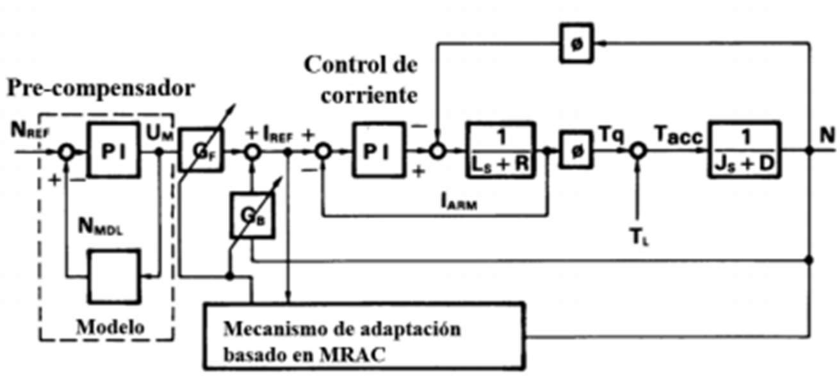
Note. This figure was taken from the article “Adaptive Friction Compensation in DC Motors Drives” by Canudas, Amströn and Braun, 1986, IEEE. Copyright 1986 by IEEE.

Continuing with adaptive control, it is established that for a speed regulator proposed for a direct current servomotor, this technique can be used by reference model. One of the main features is that it can be implemented on a microprocessor.

The implementation, as shown in Figure 2.2, also uses difference equations. The conventional electromechanical model of the servomotor that converts current into torque was used. The output speed as feedback to the control loop and also to the reference model, which depends on this speed and the reference current readjusts the gains (Naitoh & Tadakuma, 1987).

Figure 2.2.

Block diagram of the proposed control, based on reference model.



Note. This diagram is extracted from the work published by IEEE from authors Naitoh and Tadakuma, 1987, IEEE transactions on Industry Applications. Copyright by IEEE.

2.2.1 Fuzzy Logic in Control Engineering

The first approach in the development of fuzzy logic was presented in 1937 by M. Black in his article "Vagueness". In this work he proposes an alternative to represent the inaccuracy of the real world (Black, 1937).

Black (1937) defined the following: "Vagueness should not be compared with equality, since the former is approximate but not exact, so more than a defect of language it is an important source of creativity."

Lofti Asker Zadeh published an article titled "From Circuit Theory to System Theory" in 1962 where he highlighted the inability of conventional mathematics to deal with real systems (Zadeh, 1962).

Zadeh (1962) defined the following: "to deal with the analysis of biological systems and deal effectively with these, which are generally orders of magnitude more complex than systems made by humans, a radically different kind of mathematics is needed, mathematics that uses fuzzy quantities that can be described in terms of probability distributions."

It is said that the emergence of fuzzy logic was in 1965 when Zadeh introduced the idea of fuzzy sets. Fuzzy logic is then known as an extension of classical set theory and is considered a way to approximately represent the way of reasoning of the human mind. In fuzzy logic the truth value of a premise can vary in the range of $[0,1]$, in such a way that it can take truth values in that range. These values represent the degree of truth of said variable with respect to its membership function.

Fuzzy logic uses rules to describe a system; these rules are based on common sense. A problem that fuzzy logic has is that although the definition of the rules and the creation of membership functions is a simple task, the adjustment of these is not simple, it is required to use the trial-and-error method to modify the shape of the membership functions and generally many iterations are needed to define the optimal shape. The application of fuzzy logic for automatic control was first mentioned in 1972 by Zadeh.

Uncertainty is one of the paradigmatic changes it is faced in this past century and the current one. A lot of sciences and disciplines contribute to the view of all the unknowns faced while developing understanding (Klir & Yuan, 1995).

Ebrahim H. Mamdani, in 1974, applied fuzzy logic to the control of a steam engine, developing a model where inference is made from data through a rule base, that is, a rule-based system. The control algorithm interprets a set of rules expressed as fuzzy conditional statements (El-Sharkawi, 2015).

The technique proposed by Mamdani was the use of 2 algorithms, one was responsible for controlling the heat and the other for controlling the throttle of the steam engine. Each of these algorithms assigns the correction value based on the output variables of the steam engine (pressure and speed) considering the interactive nature of the process.

It should be noted that in the work presented by Mamdani it is argued that the calculations carried out have the advantage of speed of execution compared to other artificial intelligence methods. Some basic concepts that correspond to the vocabulary of fuzzy logic are:

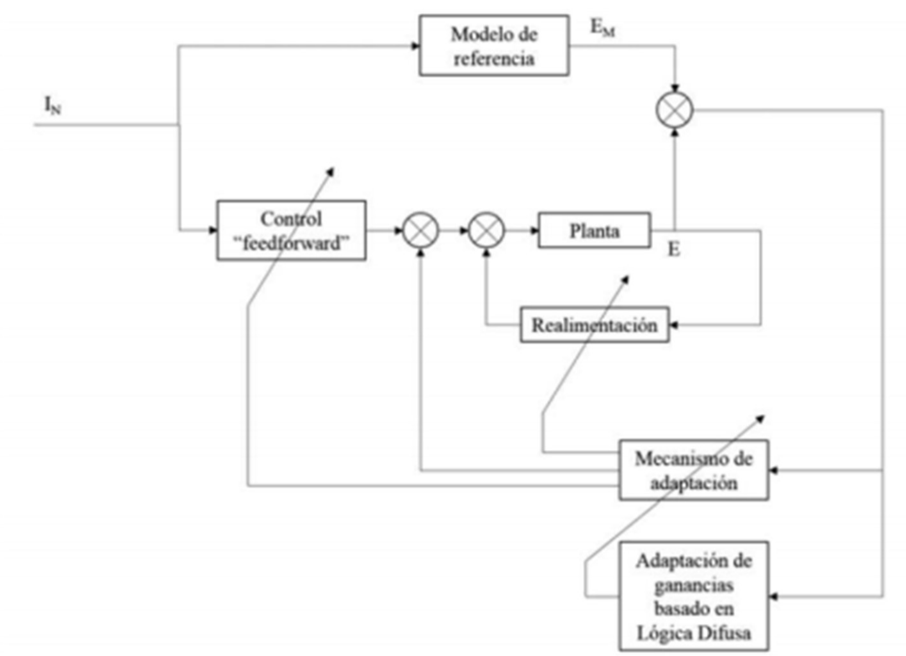
- The system is defined as a collection of elements that interact to perform a common task.
- Approximate reasoning is defined as a reasoning that is characterized by making decisions based on premises that are neither completely true nor false.
- Crisp value is considered in traditional logic, and in mathematics in general, values are used whose symbolic representation is associated with a real value. These values are known as crisp in fuzzy logic.

- Fuzzy values are the values used in fuzzy systems that correspond to sets of real values, but without a specific definition. Fuzzy values may vary depending on the perception of the interpreter.
- Linguistic variable are the values of fuzzy sets are usually expressed by words, a variable that contains fuzzy values is known as linguistic variable.
- Degree of membership or membership is a value that corresponds to a crisp entry when evaluated in a fuzzy set. This value is between $[0,1]$ and is directly related to how much it is or is not part of the set.

The use of fuzzy logic in conjunction with adaptive control by reference model has been investigated (Cheung, Cheng, & Kamal, 1996). Fuzzy logic is used to tune the reference model as an adaptation mechanism which in turn adjusts the value of the feedback gains, as shown in Figure 2.3.

Figure 2.3.

Typical FLMRAC block diagram.



Note. This diagram is extracted from the work published by IEEE from authors Cheung, Cheng and Kamal, 1996, IEEE Power Electronics and Variable Drives. Copyright by IEEE.

In the case of fuzzy control, there are techniques where the use of fractional control is proposed, that is, using fractional PID control (FOPID) in conjunction with a traditional fuzzy control (FLC) to create a fuzzy fractional PID control (FFPID).

They base the development of their transfer function on the use of fractional calculus (introduced by Leibniz in 1695), but additionally incorporate the use of genetic algorithms to recalculate the values of K_p, K_i, K_d and the fractional exponents. This proposal is considered to control a direct current servomotor using 2 control techniques (FOPID and FLC) concurrent to a third control technique (FFPID).

Servomotors are highly reliable, can operate in a wide power range, and are widely used in industry today. The purpose of mixing the two control techniques FOPID and FLC into a single FFPID by determining whether the latter provides a better response to the disturbances that an engine may face. The use of mathematical definitions of fractional calculus is proposed as follows:

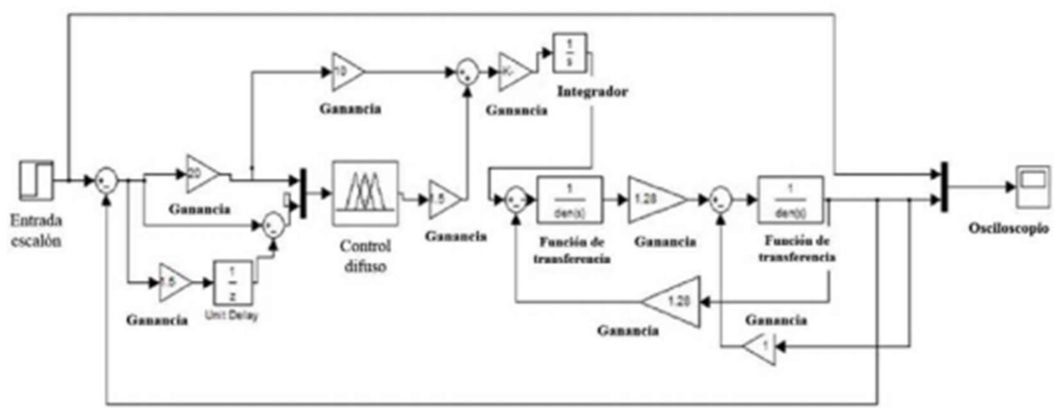
- Definition of Riemann-Liouville.
- Definition of Caputo.
- Grunwald-Letkinov definition.

The first two include the Euler gamma function. It is shown that the FFPID provides better performance due to an underdamped response (Gupta & Varshney, 2013).

Fuzzy logic, as such, is also used to control the speed of a direct current servomotor, using the error and the change in error as input parameters to the fuzzy set. A mathematical model of the servomotor can be developed using the mechanical and electrical equations. The mathematical model can be used to propose a PID control, where the gains are calculated using Simulink® auto-tune. Fuzzy control is used instead of PID control. Figure 2.4 shows this proposal. This control proposal integrates the total output of the FLC to avoid the error in steady state (Almatheel & Abdelrahman, 2017).

Figure 2.4.

Block diagram includes the use of fuzzy logic for control.



Note. This figure is extracted from the publication from IEEE, credited to Almatheel and Abdelrahman, 2017. Copyright by IEEE.

Some other proposals make use of fuzzy sets to be able to transform a crisp value into a fuzzy number according to the corresponding membership function, make use of the inference mechanism and the basis of the rules to carry out the defuzzification process. and return a functional crisp value with the ability to perform a relevant control action. It is important to mention that a crisp value or number is simply a real number, only in fuzzy logic they are called that way.

The general aspects of how fuzzy control is proposed have been mentioned and these are:

- It must be determined which elements will be worked with as inputs to the FLC (control based on fuzzy logic). At this stage you can have one or more inputs. The common thing for control systems is to use two: the error and the reason for change of error.

- Second, membership functions must be defined for these entries and membership degrees assigned considering the shape that these fuzzy sets will have. This stage is very important, since the output values of the FLC depend largely on the shape of these membership functions. Membership functions can take, among others, the following forms:
  - Triangular
  - Trapezoidal
  - Gaussian
  - Sigmoidal
- There is also the option of using singleton functions that map a given crisp input on a closed interval $[a, b]$ to a single fuzzy output value.
- Third, an inference mechanism must be created, which will consist of the use of semantic rules with the intention of carrying out a selection process of the corresponding fuzzy value depending on the degree of membership that the entry has to the different fuzzy sets.
- Finally, a defuzzification method that is based on the rules of fuzzy logic is used to determine the calculated value. Here it is important to mention that a surface is created, a space of possible values that is derived from the inference mechanism and therefore the system is very tolerant of nonlinearities.

Currently, these are the stages that are used to design a fuzzy control that is responsible for calculating the values of the PID gains on a recurring basis to make it respond better independent of the loads imposed on the system. But this is where an area of opportunity is found, the controls in general are diffuse, classical are to a certain degree static, the adaptive control is subject to an excellent model of the system conditions and this in general only covers a range of operations of the system.

Even when talking about fuzzy control, the values of the gains will be limited to the finite space produced on the defuzzification surface, there will be no room for more values.

Fuzzy controls are normally used to calculate the gains of a classic PID control. It is in this portion where the concepts of fuzzy logic come into action. There are alternatives in which PID control is proposed together with the FL with its membership functions and a speed curve or profile with the intention of showing the capabilities and power of this controller. Programming is done in C/C++ and FPGAs are used to implement some of the alternative functionalities of the concept, embedded code. Within these proposals, a rapid convergence to the optimal solution is sought, with the smallest number of iterations, but this will depend on the computational capabilities of the system. A disadvantage is that settings are made offline (Cruz-Miguel, García-Martínez, Rodríguez-Reséndiz, & Carrillo-Serrano, 2020).

2.2.2 Other Controllers, towards using AI in Control Engineering

Table 2.2 shows a synthetic analysis of the more detailed works regarding the application of more modern control strategies to servomotors.

Table 2.2.

Application of modern control strategies to servomotors.

Year	Author	Contribution
2019	Potnuru, Mary and Babu	The implementation of a bio-algorithm is proposed; It is inspired based on the pollination of flowers for the speed control of a brushless servomotor. The bio-inspired algorithm is used for the dynamic calculation of the control PID gain values with the purpose of reducing error. The results are compared to other control strategies based on AI and the Z-N method.
2019	Boonpramuk, Tunyasirut,	The use of AI is proposed to optimize a PID controller based on two algorithms. The first is the ATS and the second is the ICS, with this it is possible to optimize the speed response of brushless servo motor. The results show to be efficient compared to the Z-N and GA techniques.
2019	Yoon, Baek	An optimization algorithm is developed based on a Robust design to control the torque produced by a brushless direct current servomotor. The algorithm makes extensive use of the design space for a conventional controller. Design levels are based on the physical characteristics of the engine.
2019	Chen, Liu	They propose the use of a technique without feedback to do position control using BLDC servomotors, they take into consideration the effects that back-electromotive voltage has on their calculations to account for the better.

Note. Data collected by author on the 1st of November 2024, based on the analysis done of the publications mentioned.

It could be said that diffuse control itself is not control in itself; However, the use of these techniques is highly useful to provide classical control with a mechanism for adaptation to changes in the system. This is where the most attractive functionality lies when it is said that control adapts and evolves. But this control still has its limits, and they are determined by the form and number of membership functions. Therefore, the use of fuzzy logic can be optimized to find the best performance at the lowest cost.

2.3 Direct Current Motors

DC motors are devices that convert electrical energy into mechanical energy. They are made up of two parts called Stator and Rotor. The stator is responsible for supporting the motor elements and has the main windings, these are also known as the poles and are responsible for to form the permanent magnetic field.

The rotor, cylindrical in shape, is made of windings and a core, these around the shaft which is the one that is responsible for transmitting mechanical energy, these windings are supplied with current through some configurations, the most common being fixed brushes.

2.3.1 Electromechanical aspects of DC motors

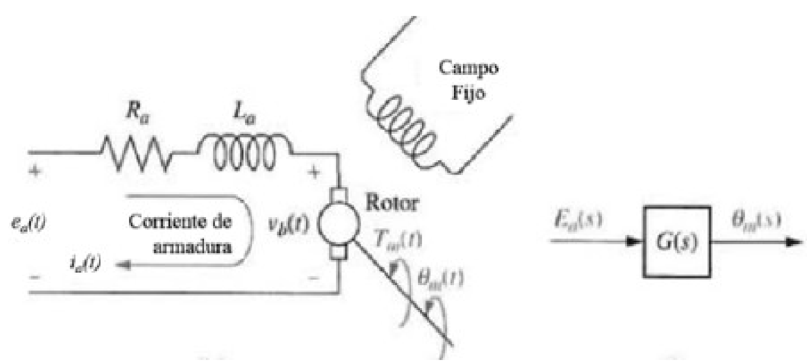
It is known that the force produced on the motor shaft is proportional to the intensity of the flux, the length of the winding and the current flowing through it. It is necessary to understand the laws that govern the movement produced by inducing an electric current in a winding that is within a deep magnetic field. It was mentioned previously that the DC motor is one that converts the electrical energy it receives from an alternating source (voltage and current); However, the direct current motor has the switch that is responsible for converting the alternating voltage into direct voltage and which ends up being transformed into mechanical energy in the form of rotational movement in its shaft. The DC motor is composed of two fixed magnetic fields (north and south) and one or several rotating coils (s) mounted around a rotating axis, see Figure 2.5 for more details. Within these devices it is important to consider the reluctance of the air and the magnetic fluxes, the distance between the rotor and the stator must be minimum, optimal and uniform to take advantage of the mechanical energy produced to the maximum, it is then said that at uniform reluctance magnetic flux is expected. constant at all points within the polar faces of the dc motor.

The induced voltage generated in the machine is a function of the magnetic flux, the rotation speed and a constant that represents the physical characteristics of the motor. The torque induced in the rotating loop is a function of the magnetic flux, the current and a constant that represents the physical characteristics of the motor. A variant of direct current motors is the so-called servomotor, also known as servo, which has a special accessory called encoder, it is a counter that helps locate the position of the arrow, the position is determined within the number of counts.

The encoders are discs adapted with slots/perforations (these perforations are the ones used to count using the so-called counts). There is a direct relationship between the number of counts and the position (the nature of the encoder can be, for example: 1000 counts per 360°).

Figure 2.5.

Electromechanical equivalent circuit of a DC motor and block model.



Note. Figure extracted from Ogata, 2008, from Prentice Hall.

Servomotors are widely used today in industry. Encoders allows the control of the position and/or speed of the motor. Another additional and very important feature is that these motors are currently sold with an interface that offers the opportunity to adjust said motor to meet the needs that arise in the field.

It is convenient to have an interface that helps with the adjustment and start-up of the motor, that supports the processing and helps with the interpretation of the signals.

2.3.2 Mathematical Model of the DC Motor

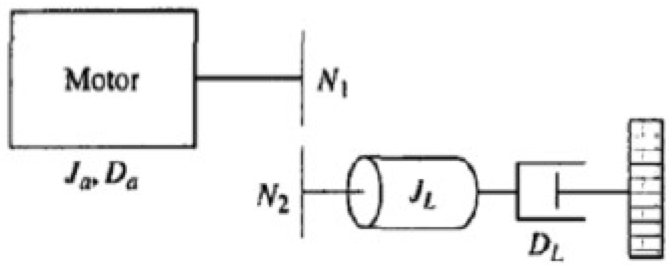
Mechanical energy enters or leaves the machine through a rotating shaft. Usually, this energy is measured per unit of time, or that is, based on the concept of power, so that its components would be the moment (or torque) and the angular velocity: $Pm = T$. The units of power, In the international system, they are watts, torque is measured in Newton per meter and speed is measured in radians per second. Electrical energy leaves the machine, or it enters it, through cables and electrical connections. For this reason, it is also handled as direct current electrical power, so that its components would be the voltage and the intensity of the current: $Pe = VI$. The unit of electrical power is measured in watts, voltage in volts, and current intensity in amperes. From a mechanical point of view, all electrical machines are built by two large parts: the stator and the rotor. From the functional point of view, two groups are also considered: the magnetic circuit and the armature. It has been seen both in generator mode and in the motor, there is an electromotive force and an electromagnetic torque. This relationship is the energy conversion formula and is given by: $EIa = Te$. These equations form the fundamental basis of the electromechanical relationship of the direct current motor.

One of the problems that those who work in the control or identification of processes traditionally must face lies in the existence of a diversity of mathematical tools to study, analyze and design these systems.

There are different types of models, in some of them, such as scale models, the aim is to reproduce the real physical system through a more manageable model or system on which you can experiment and extract results, Figure 2.6 exhibits the model of a DC motor connected to a load. These models, although interesting, are not the ones that are most interesting in control, but rather those where the predictions are made on a certain mathematical model of the system. The mathematical model can be represented with an equivalent electromechanical circuit. And from there it is considered whether the engine has any mechanical element connected to it.

Figure 2.6.

DC motor connected to a load.



Note. Figure extracted from Ogata, 2008.cOpyright by Prentice Hall.

CHAPTER 3

Methodology and Development

In this section of the thesis defines the systematic approach and processes undertaken to achieve the objectives of the proposed study by exhibiting the strategies employed to ensure the results are valid and accurate as much as possible. It is important to mention this is using trial and error with a defined experimental set to get test data to substantiate the analysis.

3.1 Analysis

Specifically, all DC motors are prone to not meet control requirements while operating. Currently several techniques of adjustment are used, but not many of them are practical nor implemented commercially since the technicians normally adjust the gains at site. FLC is a control technology used widely but with the inconvenience MF is defined by experienced users heuristically.

3.2 DC Motor Identification

The transfer function can be represented as follows in simple terms in the Laplace domain and simplified to be defined as a first order equation when it comes to speed control. The selected engine is 36JX30K/38ZY63 Dongyangcorp brand. It has a reducer with a 51:1 ratio and is adapted to the E4P “Encoder” with 360 counts per revolution, the DC motor is shown in Figure 3.1.

Figure 3.1.

DC motor selected for the experimentation that includes gear reduction.

Note. This image is recovered from Phidgets on 1st November 2022. https://www.phidgets.com/?tier=2&catid=20&pcid=17

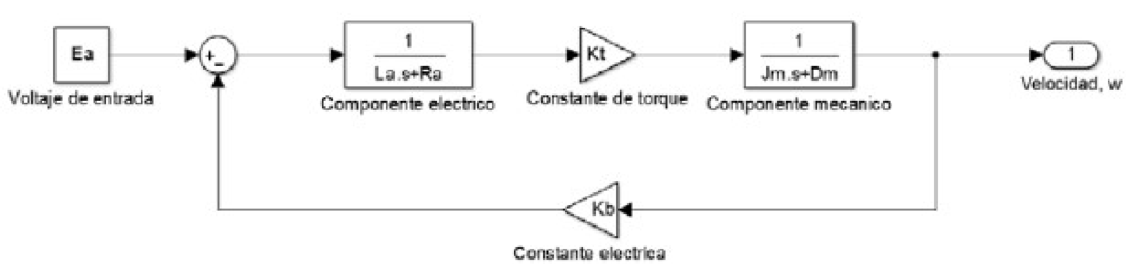
The methodology for identifying motor parameters in an open loop (in the time domain) proposes the use of the following transfer function:

$$\frac{w(s)}{E(s)} = \frac{K_m}{T_m s+1} \qquad \text{E10.}$$

It must be induced in the motor with a voltage input to obtain the speed response and at the same time verify the response time, the block diagram is depicted in Figure 3.2.

Figure 3.2.

Block representation of DC motor used for experimentation.

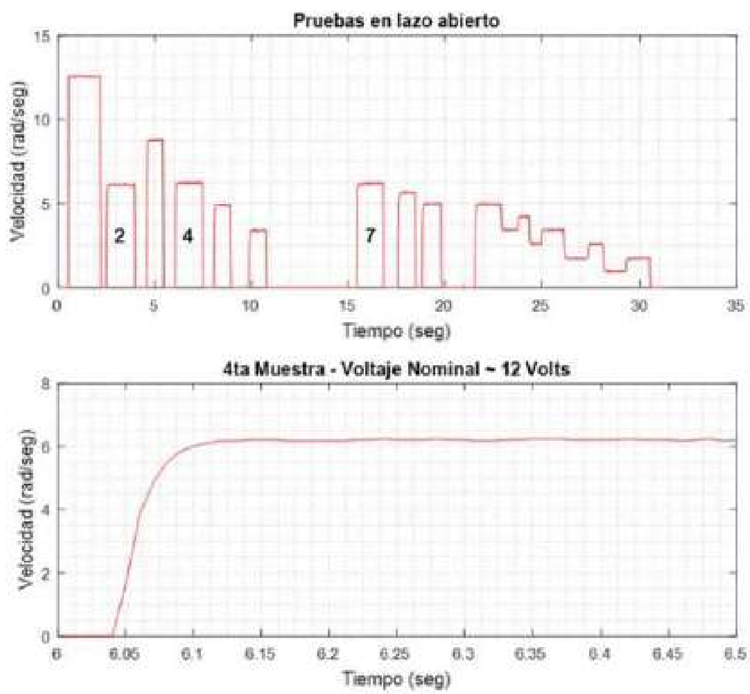


Note. Block Diagram was developed by the author to represent the DC motor model. This is based on different Control Engineering literature.

Several voltage pulses were performed to verify the response of the DC motor, and the results are shown in the following Figure 3.3.

Figure 3.3.

DC motor identification using open loop method and transfer function.



Note. These graphics were developed by the author.

Due to the identified transfer function, the natural and forced response of the system can be known for a step of 12.15 volts. The resulting transfer function is as follows:

$$\frac{w(s)}{E(s)} = \frac{23.228}{s+45.5} \quad \text{E11.}$$

Where $K_m = 0.511$ and $T_m = 0.022$.

3.2.1 Classic Control Theory highlights

Feedback control systems can be classified in various ways, depending on the purpose of the classification. According to the analysis and design method, these are defined as: linear and non-linear and/or time-variant or time-invariant. According to the types of signals used in the system, these are in continuous time or discrete time or in modulated and non-modulated systems. They are often classified according to their primary purpose, for example a position control system and a speed control system.

The controlled variable is the quantity or condition that is measured and controlled. The manipulated variable is the quantity or condition that the controller modifies to affect the value of the controlled variable.

The purpose of a control system is to obtain the desired response from one or several variables within a system. This can be achieved by using an open loop system or a closed loop system. Closed loop systems are those where the control determines the input value, that is, it indicates the reference value for the process and uses the feedback signal to measure the output.

Feedback control is essential to keep process variables close to the desired values regardless of whether the system is exposed to disturbances or changes in its dynamics.

3.2.1.1 Control PID

PID control is a 3-term controller that has a long history in the field of automatic control and has its beginnings at the beginning of the last century. Due to its characteristic simplicity and the fact that it is very intuitive, this controller has become the standard for many industrial applications. Considering technological advances, today this type of controller is implemented electronically in almost all cases, contrary to what was done before, with a pneumatic or electrical implementation.

As mentioned before, PID control consists of 3 terms, and at this time it is convenient to rename them to the 3 control actions which are: proportional, integral and derivative.

The proportional action is proportional to the current error of the controller and is defined as:

$$u_t = k_p e(t)$$

Where k_p is the proportional gain and $e(t)$ is the error, also represented by the additive inverse of the reference value with the system output value. In terms of the controller, the proportional action can be written as:

$$c(s) = k_p$$

The integral action is proportional to the integral of the current error of the controller and is defined as:

$$u_t = k_i \int_0^t e(\tau) d\tau$$

Where k_i is the integral gain, and the integrand is responsible for considering integrating the previous error values. The presence of the pole at the origin of complex plane, through integral action, allows the reduction of the error in stable state to 0, even after the system experiences a disturbance. In other words, the control action can correct the value of the constant of integration resulting from integrating from 0 to t.

The integral action can also be written as:

$$c(s) = k_i \frac{1}{s}$$

Finally, the derivative action tries to consider the prediction of the value of the control error. Therefore, the derivative action is proportional to the derivative of the current driver error and is defined as:

$$u_t = k_d \frac{de}{dt}$$

Where k_d is the derivative gain. In terms of the controller, the action derivative can be written as:

$$c(s) = k_d s$$

The set of the three control actions is the well-known PID and this can be represented ideally as follows:

$$c(s) = k_p \left(1 + \frac{1}{Ti_s} + T_d s\right)$$

All those equations were defined in the precious chapter. There is a new set of equivalents that need to be defined as follows:

$k_i = \frac{k_i}{T_i}$ E12.

$k_d = k_p T_d$ E13.

This representation relates the integral and derivative gains with the proportional gain; however, is not the only representation.

3.2.1.2 PID Control proposal

The method for controller design depends largely on the dynamics of the process to control. In 1946 Ziegler and Nichols presented two methods, which are now called classics, and are widely used today for defining a controller.

These methods characterize the behavior of the system based on two parameters, the first is related to the gain of the process and the second describes how fast the process is. The method of frequency response is based on the characterization of the dynamics of the process that you want to control.

The design depends on knowing the point on the curve of Nyquist, resulting from the transfer function that represents the process, where it intercepts the negative real axis. Since its genesis, this point is characterized by two parameters K_u and T_u that are known as the gain last and the last period respectively.

To determine the parameters K_u and T_u it is necessary to connect a controller proportional to the process and increase the gain slowly until the process begins to oscillate. In this way the values of the profit and the last period.

This method was used to obtain the ultimate gains from experimentation, the response obtained is shown in Figure 3.4.

Figure 3.4.

Oscillatory response of the system to the last gain value.

Note. This figure is developed by the author based on testing data obtained. The methodology is leveraged from Ogata (2008) published by Prentice Hall.

Table 3.1.

PID Controller Parameters from Ziegler-Nichols based on frequency.

Controller	k_p	T_i	T_d
P	$0.5K_u$		
PI	$0.45K_u$	$T_u/1.2$	
PID	$0.6K_u$	$T_u/2$	$T_u/8$

Note. This table is developed by the author basing the information in Ogata (2008) released by Prentice Hall.

Therefore, the PID controller gains are: $k_p = 0.084$, $k_i = 1.2$ and $k_d = 0.0015$, using the estimation from Table 3.1.

3.2.1.3 Stability Analysis

Stability is one of the most important specifications for systems of control. When a system is unstable, no control can be designed to obtain a specific transient response, nor to guarantee an error in steady state. A system is said to be stable if the natural response approaches zero when time tends to infinite, this applies to systems linear and time invariant.

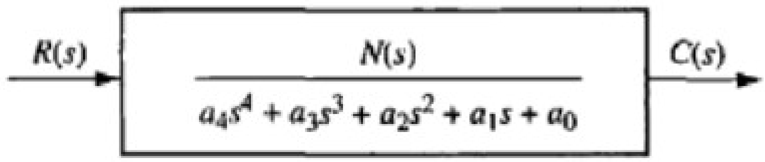
The Routh-Hurwitz method allows us to identify the poles in both half-planes and on the jw axis and it is not required to solve the closed loop system. This consists of two steps:

- Generate the Routh table.
- Interpret the Routh table to identify how many poles are in any of the half planes or on the same jw axis.

To generate the Routh data-table, it is essential to know the function of equivalent transfer of the entire system and take into account the denominator of the same, since the analysis is based on the poles of the system. To illustrate how to construct the Routh table, suppose the Transfer Function (TF), in Figure 3.5, is the equivalent TF.

Figure 3.5.

Equivalent TF of the system.



Note. This figure is extracted from Ogata (2008) published by Prentice Hall. Copyright to Prentice Hall.

From this function, the characteristic polynomial (denominator) is considered and the coefficients to construct the Routh. For further details on how to develop the Routh table please consult Ogata (2008).

If the transfer function has all its poles on the side of the half-plane left, then it is said to be stable. Therefore, it is inferred from the table of Routh that, if there is no sign change in the first column, the system is stable.

Knowing the values of the gains and the form of the PID control, the stability analysis using the Routh-Hurwitz criterion. The matrix is created Routh considering the coefficients of the characteristic equation. Therefore, this characteristic equation is: $1.018\, s^2 + 46.5\, s + 14.21$, the poles meet at $p_1 = -45.24$ and $p_2 = -0.61$, this shows that the system is stable, since the poles are in the left real half plane, and it is also shown that the system is overdamped since there are no complex conjugate poles. The Routh matrix is constructed, and it is shown that the system is stable since there are no sign changes.

3.2.1.4 Fuzzy Control

In the first instance, Fuzzy Logic Control (FLC) is based on Fuzzy Logic (FL). Fuzzy Control uses statements, also known as rules, to control a process. This controller can receive multiple inputs and have databases experience (known in English as "expert knowledge").

The interface included in the FLC is in natural language terms (with Natural language should be understood more like the common/everyday language used by people) and this is the main distinction of this type of control compared to other control techniques.

FLC can be linear or nonlinear (this depends on the form of the functions or rather the type of function which is incorporated into the fuzzy set). However, this must remain true design procedure to be implemented. The proposed steps are as follows:

1. Design a PID control,
2. Change it to a linear fuzzy control,
3. Make this control non-linear and
4. Adjust the resulting control.

Some examples of consumer products that include FLC are automatic washing machines, video cameras and cars. In the industrial part we have examples such as: cement mills, trains and robots.

A first approximation to the definition of FLC is the following: an automatic control with a self-regulating mechanism to control an object according to a desired behavior.

What is fuzzy control? It is from general knowledge that computers could take decisions, but within a set of 2 values (binary), that is, they can evaluate two states: high and low (true and false, yes and no, 1 and 0).

FL, on the other hand, allows the use of intermediate truth values, that is values that are between true and false (1 and 0). An example is controlling the air conditioning for a room, it can be expressed as follows:

- for a two-state control: above or below the set point,
- for fuzzy control of two states: hot and cold.

FLC must react based on rules, and these rules are what are responsible for determining the actions to be executed by the controller, a rule for this control strategy can be the following:

If the temperature of the room is hot and is increasing slightly, then it will increase cooling.

All values/numbers that are contained in the “hot” set represent the mathematical basis of the control (at this point it is worth mentioning fuzzy logic).

The heart of fuzzy control is represented by the collection of linguistic rules of the form “if this – then this” these rules are responsible for making the reasoning used in computers, for example, closely resemble human reasoning.

Continuing with the previous example (air conditioning control for a room but being more specific; the desired temperature in the room) it is known that the operating principle of control is the measurement of the temperature. However, more information is needed to determine if the room is warm. Therefore, a way must be incorporated to be able to make this distinction, something more like what a human could classify based on the temperature perception.

With the argument above it is convenient to make the following associations: a very hot room has a value of 1, a very cold room corresponds to a value of 0.

A characteristic of fuzzy control is that it is (to a great extent) more suitable than PID control in higher order processes, systems with long periods dead time or systems with oscillatory modes. However, it is convenient to review some points against others in favor of using an FLC against a PID control.

Fuzzy control uses an inference mechanism that maps the control input to its output. The inference mechanism is composed of:

- Fuzzification defines how state variables represent the dynamics of the system.
- Rule base is the decision-making process based on rules of control and linguistic variables.
- An inference engine is the programming code that is responsible for processing the rules/cases contained in the rule base.
- Defuzzification, converts the linguistic variable back to a value in the domain of real numbers. Depending on the fusion method, it can consider several alternatives to do the inverse process.

Because the control strategy is made up of rules of the type “if this, then this” is a very easy to read for any user/operator. The Rules can be constructed based on colloquial language with words such as such as: “high, low, much, little, increase, decrease” among others. The users/operators have the freedom/ease of being able to incorporate the experience that they have of the process in this control.

Fuzzy control can include multiple inputs and outputs. The variables can be combined with “if this, then this” rules and “AND” and “OR” connections. The rules are executed in parallel, which results in the calculation of several possible corrective actions that will conflict, but the controller will oversee taking the best one as an output.

Fuzzy control allows non-control specialists to be able to design/adjust/work with fuzzy control.

CHAPTER 4

Results and Discussion

This chapter shows the characteristics of the proposed work. They address the issues related to the requirements established for the control, the driver, the test prototype, the software and its interface.

4.1 Design of the Control System

Speed control in a system depends on the use of a servo-actuator that allows the transmission of movement. A particular case of servo-actuator is the composed of the direct current motor, the “gearbox” (amplification system or attenuation) of the speed/torque and the speed measurement mechanism.

This speed processing mechanism is an important aspect because it is known as the feedback system and as such it allows the formation of the closed loop. Here are some complications derived from the closed loop.

(1) the dynamics of the system changes, (2) noise is introduced into the system due to the sensor signal.

In general, these fundamental aspects must be considered to calculate the profit margins that allow the control system to guarantee stability, you should think about observing these signals.

Having said the above, it is known that “Drivers” are fundamental for the processes control and in general for productive transformation processes, these helps increase the performance of the actuator systems within this transformation. The “drivers” allow the management of ascent loads and descent and through the control signal make the actuator return to work to the pre-established parameters. In the case of electromechanical ones, it is known that suddenly the loads imposed on the arrows of said actuators.

An essential step is to create prototypical systems of such devices with the intention to understand and improve the industrial implementation of said components (Ponce, et al. 2020), Ponce, et al. mentions “the design of prototypes for testing can end up achieving excellent results when said are used for the purpose of experiment.”

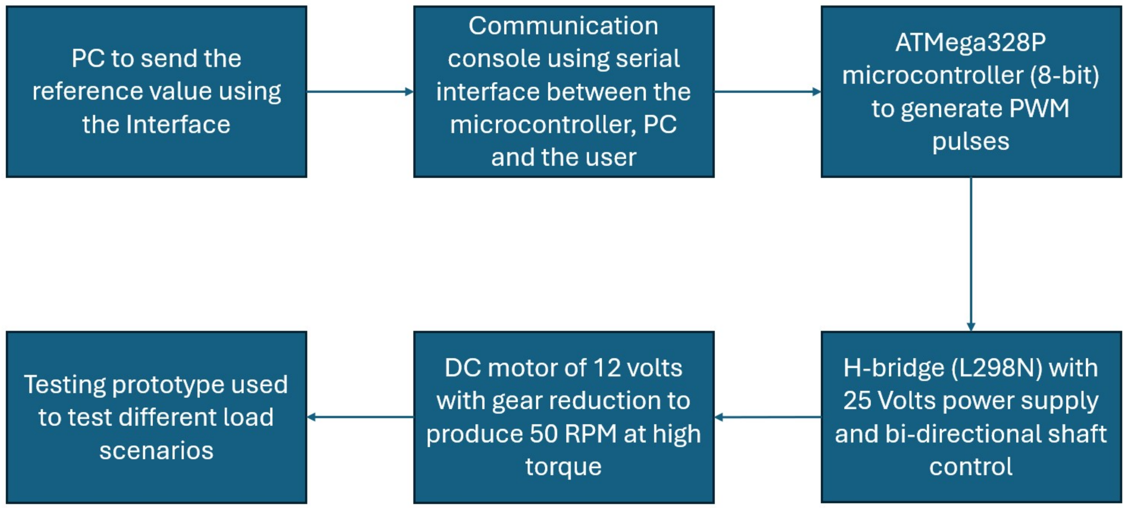
The modern control and some other techniques such as adaptive control or the use of artificial intelligence for control to update profits are used as alternatives for adapting these to the changes that the system experiences in the time. These metaheuristic techniques contribute to the continuous development of the modern control systems. One can also argue the use of series of time and its analysis to be able to create a control that adapts to changes by studying the behavior of the system a few moments in the future with a moving average model (Niembro-Ceceña, et al., 2022).

Finally, it must be specified that the “driver” must be prepared to be able to process the control strategy for which it is intended, therefore, it is of vital importance to define the essential concepts to which this should be subject device with the idea of being able to be used effectively. To understand the control requirements, you must know its design, which has been set out in previous chapters.

The high-level system components are presented in Figure 4.1.

Figure 4.1.

Concept of the System.



Note. This figure was created by the author.

4.2 System Requirements Definition

It is necessary to define the system requirements beyond the control requirements, since these occupy an even more predominant place in this case. First, you must understand the operation of the system and its interactions. Subsequently, the "driver" test must be done and then talk about software requirements. Some fundamental aspects are the control signal (command), the communication console, the microcontroller, the H-bridge, the DC motor, and the testing prototype.

1. The command signal refers to the reference signal which in this case is the angular velocity.

2. The control signal is the result of the arithmetic difference of the command signal and the feedback signal.

3. The feedback signal is the signal collected from the sensor system, in this case the one coming from the tachometer.

4. The communication console is the human-machine interface through which you can interact with the system.

5. The microcontroller is the neuralgic unit of the system that is responsible for calculating and commanding signals.

6. The H-bridge is responsible for the power of the system and therefore reverses the direction of rotation if necessary.

7. The test prototype is the device with which the corresponding tests will be carried out to determine if the controller complies with the requirements.

Once the elements that make up the system are known, defining its requirements becomes a simpler task. From the perspective of product, it is understood that the motor must rotate its shaft at a certain speed regardless of the load that is attached to it and as long as it is within the limits allowed by the capacity of the motor itself, there is a range of speeds at which the motor can operate according to a range speeds defined by the manufacturer.

Additionally, the speed requirement has an associated error band for said speed in a stable state and this should not typically exceed 0%. Obviously, the production process will determine this error range more precisely.

The response time and overshoot are also some parameters that become relevant when talking about system requirements and for the first one can be defined from 0.5 seconds up to 2 seconds depending on the case and in the case of sub-damped systems that should not exceed 10% at peak moment.

This concept includes the elements that make up the system and are the microcontroller, the H-bridge, the tachometer, the DC motor, and the test cell with their intrinsic connections.

It is understood that the type of “driver” depends greatly on the application of which it will be part, which is why it must be considered that the nominal voltage, the current of the DC motor and the maximum operating conditions, in the same way if the rotation of the arrow is bidirectional, power consumption, signal processing and so on.

Proposal of the driver, shown in Figure 4.2, depending on the type of control. In the currently proposed situation, it is known that the motor operates nominally at 12 volts but has the capacity to manage up to 24 volts, it accepts a maximum current of 3.5 Amps and the microcontroller could handle 5 volts TTL. In general, the following can be said:

• The “driver” must assist in regulating the angular speed.

• The microcontroller is responsible for processing the error signal through the arithmetic difference of the reference and the measured signal.

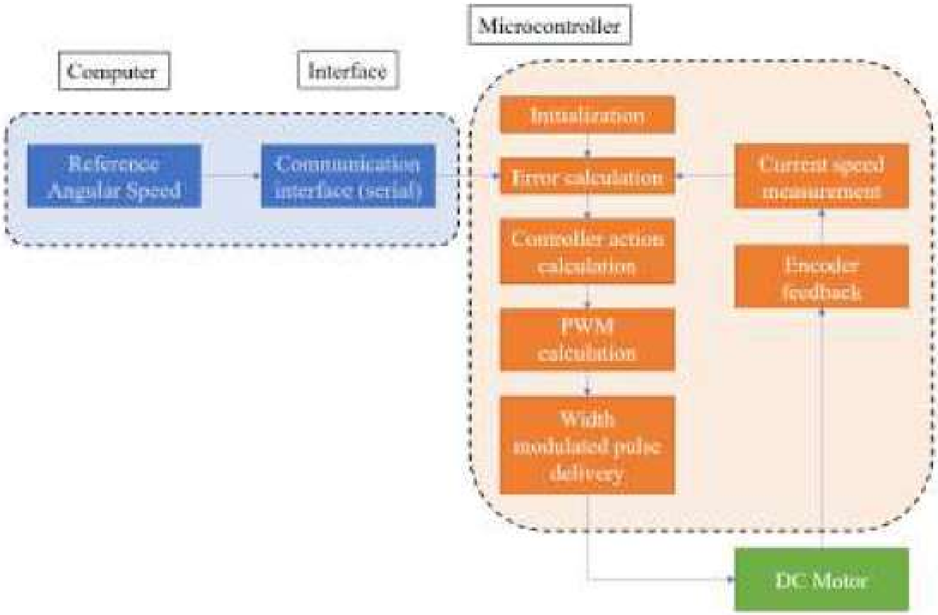
• The reference signal is entered through the console.

• Once the error is calculated, the control signal is calculated and passed to the PWM calculator.

• The PWM signal is managed by the H-bridge and the power stage is made to supply the motor with the corresponding proportional voltage until the error is 0.

Figure 4.2.

High-level description of the System.



Note. This figure was developed by the author.

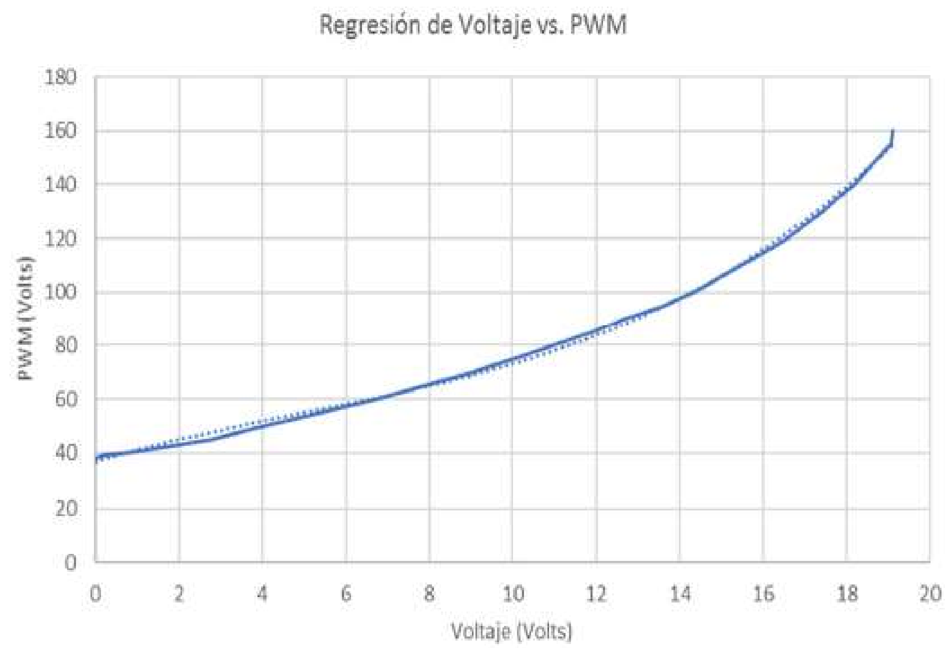
4.2.1 PWM Characterization

The response produced from the PWM to the voltage is not linear. Therefore, measurements were made of the voltage values produced by different pulses and It was found that the relationship is not linear. A polynomial regression method based on the least squares error was used. R2 must tend to 1 to guarantee a reliable regression.

The regression equation was found to be $y = 0.0192x^3 - 0.2807x^2 + 4.5101x + 36.958$, with an $R^2 = 0.9981$. The variables are y represents the PWM pulse and x the input voltage. This is shown in Figure 4.3.

Figure 4.3.

PWM Characterization.



Note. This figure was developed by the author.

4.2.2 Control Software

The control software is composed of several modules. The first module is made up of all the configuration parameters and global variables.

The second module is responsible for initializing the microcontroller, establishing serial communication and "setting" the registers to determine the duty cycle.

The third part is the main body of the program also known as the main function and is responsible for executing the program iteratively.

In this section the reference speed value is received and processed against the current value, the error is calculated, and the control is made. Using a "char" variable the type of control is selected between the PID and the FP+I.

It is important to note that the control must be calculated in this part of the program to ensure that the

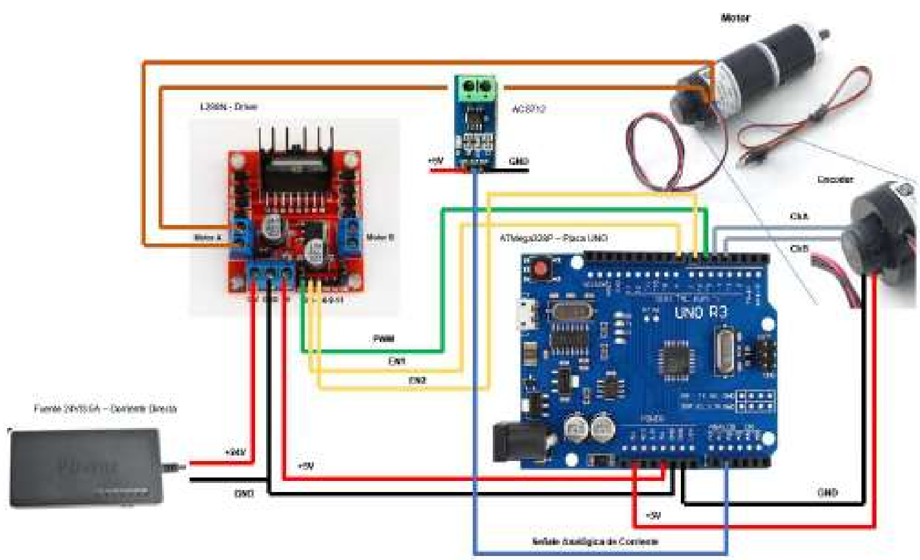
The integral part adds up the error and the pertinent control action can be executed. This module also has an interface, through serial printing, with the which is interacted through the serial communication screen.

The last module is composed of all the functions created to make necessary calculations such as: reading the encoder, calculating the current speed, error calculation, calculation of the PWM pulse value based on the input voltage, among other functions. These calculations are separated to improve its management. For further reference to the code see Appendix A.

The schematic of the proposed system is Figure 4.3. All corresponding elements have been identified and their functions defined, therefore, the driver is complete. The code implemented without However, it is important to mention that according to the simulations, there are nonlinearities that make the control signal limited; this occurs when the error is large, and the proportional part has a negative effect in this regard.

Figure 4.4.

Schematic of the proposed solution.



Note. This figure was developed by the author.

4.3 Results

Today, maintaining key performance indicators (KPIs) for the entire lifespan of DC motors involves a range of advanced methodologies, including soft computing, fuzzy logic, artificial intelligence (AI), fractional calculus, classic control theory, and adaptive control techniques (such as time series analysis). DC motor drivers are increasingly incorporating novel technologies. While more innovative companies may integrate AI-based solutions, most drivers still rely on traditional PI or PID controllers. However, during system operation, various factors, including overloads, operating conditions, and wear, pose challenges to sustaining optimal performance and meeting KPI targets.

The main limitation of PI and PID controllers lies in their reliance on gains calculated based on an initial DC motor model, which represents ideal conditions as a best-case scenario. However, as the system operates, various challenges cause these gains to fall short of maintaining the required KPIs over time. Consequently, much research has focused on strategies to update these gains dynamically within a stable operational range. This process continues until the system can no longer maintain zero steady-state errors, control overshoot or undershoot (depending on the design and system behavior), or achieve the desired response time.

Such iterative techniques aim to adjust the gains to align with the desired KPIs continually. Additionally, some research explores the impacts of wear, hysteresis, heat, and other electromechanical interactions on performance. Simplifying the system model is often preferred, though some applications always require strict adherence to high-performance standards.

In scenarios where performance must be maintained, advanced technologies are employed. These are not exclusive to such applications but offer significant value. This is where the contribution of control software becomes noteworthy, leveraging MCUs to provide the following benefits. Embedded software offers significant potential for exploring alternative controller strategies, contributing to advancements in control engineering research. Its key advantages include:

- Cost Savings: Employs straightforward electronic circuitry that is easy to maintain and replace, reducing expenses.
- Legacy Programming Support: Leverages established programming languages that enhance the capabilities of control software.
- Improved User Interaction: When properly partitioned, the software can support advanced graphical management for enhanced usability.
- Multiprocessing Capability: Facilitates communication between multiple MCUs for efficient multiprocessing.
- Scalability and Feature Expansion: Enables the addition of enhanced, reusable features, with the possibility of exporting code where applicable.

4.3.1 Testing results

The results obtained were compared with those of a PI controller, revealing notable advantages in certain test cases. For example, 62.5% of the results exhibited less than 10% overshoot when using the FLS-TS approach. At lower speeds, variability was higher, which is expected due to the unique inertia effects across different loads, making shaft rotation behavior distinct for each case. Both controllers faced challenges in this scenario, but the FLS-TS approach reduced overshoot variability at reference speeds, particularly after group 3. This improvement is attributed to the influence of the three distinct membership functions (MFs) on the proportional gain.

While some tests showed room for improvement, these can be addressed by refining the shape of the MFs thereby enhancing the system’s response. The Control Software is a control software designed to integrate fuzzy logic controllers (FLCs) based on the Takagi-Sugeno (TS) model into the core control framework of a DC motor driver. Its notable features include:

- Embedded MF implementation within an MCU, enabling scalability and updates.
- Tolerance to system nonlinearities in control actions.
- Development in C language, making the code portable and reusable across similar platforms.

Control Software supports leveraging fuzzy logic soft computing within MCUs, opening new avenues for research and practical applications. This capability aligns with the growing adoption of artificial intelligence (AI) in control engineering.

The software considers the dynamic changes induced by closing the control loop and includes control law modules that are open to updates. It also integrates precise PWM resolution to suit the system and power supply requirements. The ability to directly program AVR (Alf and Vegard's RISC) processors enhances flexibility and potential, allowing for custom coding of control laws. This flexibility supports the inclusion of:

- Complex nonlinear functions for MFs.
- Additional rules with linear or nonlinear patterns.
- Filtering for the derivative (D) component in PID controllers, or setting D to zero for PI control, all achievable through a single MCU re-flash.

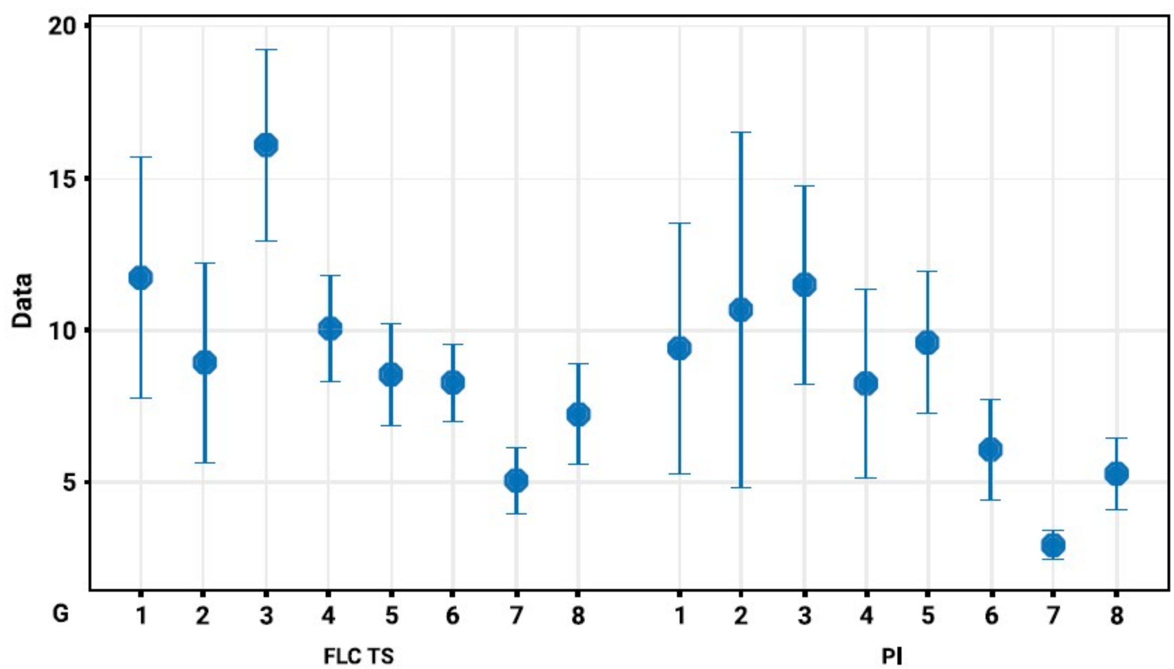
AVR processors are a powerful and cost-effective option for both control systems research and commercial applications. They offer high-level programming capabilities in contrast to programmable logic controllers (PLCs), which often require specialized expertise in less common languages.

Soft computing, a subset of AI, addresses the challenges posed by the nonlinearities inherent in DC motor drivers that affect motor performance. This software exemplifies this approach, offering reusable, adaptable code for researchers and developers exploring FLCs for alternative control strategies or commercial purposes.

While integration of fuzzy logic as an AI-based strategy within AVR MCUs propels the research field forward, it also underscores the potential for expanding these methodologies to new horizons. Results are shown in Figure 4.5.

Figure 4.5.

Statistically analyzed testing results from experiments.



Note. Figure developed by the author.

The use of the ANN was included based on perceptron to assist with the selection among a variety of control strategies including different MFs characteristics but also including switching between other control strategies. Based on the results, it was shown that an option of adding a decision based controller is plausible in a AVR using this technique, more details to come in a paper that is being worked.

CHAPTER 5

Conclusions

Developing control software can be particularly challenging when system nonlinearities are not fully understood. While some control engineers, especially those proficient in adaptive control, can address these complexities effectively, others rely on software designed to handle nonlinearities inherently. Control Software exemplifies this approach by utilizing a fuzzy logic controller (FLC) based on the Takagi-Sugeno (TS) model.

The TS model was chosen for its ability to produce a curve-based response (PWM dependent on voltage error), as opposed to the surface-based responses of Mamdani models, which are more complex and harder to implement on microcontroller units (MCUs). The experimental results demonstrated certain benefits of this approach, with opportunities for further improvement, particularly in refining the shape of membership functions (MFs).

The code is openly available and easily adaptable, compatible with C compilers, code snippets, or Arduino applications. It is primarily intended for integration into AVR MCU-based applications, serving as an application programming interface (API) for DC motor drivers. While initially designed for in-house or research DC motor drivers, its versatility supports broader use cases.

The computational cost of Control Software is not prohibitive, with latency remaining within acceptable limits. Experimental data showed that the proposed controller achieved reference speed in 100% of test cases (on average) across varying loads (within defined boundaries). This contrasts with the PI controller, which succeeded in 85% of cases. Additionally, the proposed controller reduced overshoot by up to 16% compared to the PI controller.

Control is a critical component of DC motor drivers, as it determines the actions necessary to maintain shaft operation over time. The architecture proposed by Control Software supports the inclusion of more complex mathematical expressions, further enhancing system performance and responsiveness.

Fuzzy logic controllers are already widely used in products like washing machines, electro-discharge machining (EDM) equipment, and other consumer goods. This highlights the importance of soft computing, which has substantial potential to integrate artificial intelligence into control software, paving the way for smarter and more efficient control systems.

5.1 Future Work

The research and development of control software like Control Software open several avenues for further exploration and improvement.

Dynamic Membership Function (MF) Optimization: Investigate advanced algorithms, such as machine learning or evolutionary optimization techniques, to dynamically adjust the shape and parameters of MFs during runtime. This would enhance the system’s adaptability to changing conditions and improve overall performance.

Integration with Advanced AI Techniques: Expand the incorporation of artificial intelligence by integrating neural networks or hybrid AI models with fuzzy logic controllers. This could improve the system's ability to handle complex nonlinearities and unforeseen operational scenarios.

Hardware Scalability: Explore the implementation of Control Software on other MCU architectures or platforms beyond AVR, such as ARM-based processors, to evaluate its scalability and compatibility with more powerful systems.

Real-Time Performance Metrics: Develop real-time monitoring tools to analyze performance metrics like latency, overshoot, and response time under various loads and conditions. This would provide deeper insights into the controller's efficiency and areas for improvement.

Multi-MCU Collaboration: Investigate the potential for intercommunication between multiple MCUs to enable distributed control systems. This could lead to more robust and fault-tolerant designs for industrial applications.

Power Efficiency Optimization: Analyze the controller's impact on energy consumption and explore strategies to enhance power efficiency, making it suitable for battery-operated or energy-sensitive applications.

Fuzzy Logic Benchmarking: Conduct comparative studies of different fuzzy logic models, such as Mamdani, TS, and hybrid approaches, to establish benchmarks for various performance criteria in DC motor control.

Advanced Nonlinearity Modeling: Develop more comprehensive models to account for complex electromechanical interactions, such as hysteresis, wear, and thermal effects, to further enhance the accuracy and reliability of control actions.

These directions will not only strengthen the capabilities of Control Software but also contribute to the broader field of intelligent control systems, making them more versatile, efficient, and widely applicable.

BIBLIOGRAPHY

Almatheel, Y. A., & Abdelrahman, A. (2017). Speed control of a DC motor using fuzzy logic controller. *International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*.

Bennet, S. (1993). *A History of Control Engineering 1930-1955.* IET Digital Library.

Bennet, S. (2008). *A History of Control Engineering 1800 - 1930.* London, United Kingdom: Institution of Engineering Technology.

Black, M. (1937). Vagueness: An exercise in logical analysis. *Philosophy of science, 4*(4), pp. 427 - 455.

Canudas, C., Amströn, K. J., & Braun, K. (1986). Adaptive Friction Compensation in DC.

Cheung, J. Y., Cheng, K. W., & Kamal, A. S. (1996). Motor speed control by using a fuzzy logic model Reference Adaptive Controller. *IEEE Power Electronics and Variable Speed Drives*.

Cruz-Miguel, E., García-Martínez, J., Rodríguez-Reséndiz, J., & Carrillo-Serrano, R. (2020). A New Methodology for a Retrofitted Self-tuned Controller with Open-Source FPGA. *Sensors*.

El-Sharkawi, M. (2015). *Electric Energy: An Introduction.* United States: CRC Press Taylor and Francis Group.

Gupta, S. K., & Varshney, P. (2013). Fractional fuzzy PID controller for speed control of DC Motor. *Third International Conference on Advances in Computing and Communication*.

Hall, A. C. (1946). Application of circuit theory to the design of servomechanism.

Howard, J., & Gugger, S. (2020). *Deep Learning for Coders with Fastai and PyTorch.* United States: O'Reilly Media.

Klir, G., & Yuan, B. (1995). *Fuzzy sets and fuzzy logic* (Vol. 4). New Jersey, United States: Prentice Hall.

Naitoh, H., & Tadakuma, S. (1987). Microprocessor-based adjustable-speed DC motor. *IEEE TRANSACTIONS ON INDUSTRY APPLICATIONS*.

Ogata, K. (2008). *Modern Control Engineering.* Pearson.

Rusell, S. (2010). *Artificial Intelligence. A Modern Approach.* New Jersey, United States: Prentice Hall.

Suh, S. H., Kang, S. K., Chung, D. H., & Stroud, I. (2008). Theory and Design of CNC.

Tong, W. (2014). *Mechanical Design of Electric Motors.* United States: CRC Press.

Visioli, A. (2006). *Practical PID Control, Series in Advanced Engineering.* London: Springer.

Zadeh, L. (1962). From Circuit Theory to System Theory. *Proceedings of the IRE , 50*(5), pp. 856 – 865.

APPENDIX A

```
/*
Core control software for a DC motor driver
Ver: 1.0
Autor: Jose Niembro
Fecha: 11/01/2023

Closed-loop speed control for a DC motor: 38ZY63-1230/36JX30KG8D20.
Controllers: PID (Kp, Ki, Kd), Fuzzy logic controller for proportional gain with
Integral (FP+I - FKi = 15.0), PID (Ki = 1.2, Kp = 0.085, Kd = 0.0015).
Encoder: E4P / 360 CPR.
*/

// Pins configuration
const byte IN1 = 7; // IN1 for H-Bridge
const byte IN2 = 8; // IN2 for H-Bridge
const byte PWM = 5; // PWM output
const byte ChA = 2; // Encoder ChA
const byte ChB = 4; // Encoder ChB

// Current sensor setting7 config
int Iter = 0, Counts = 75; // Iterations to calculate an Electrical Current
average, 100 samples to average
double Sens = 0.125; // Sensor Sensibility
double MsVoltageACS = 0.0;

// Init output for PWM
int Speed = 0;
int Output = 0;

// MCU timer setting
int Tmr_Ctr = 0;
```

```
// General settings
double MtrCurrentSpeed = 0.0;
double Err = 0.0;
double Voltage_Ref = 0.0;
double PWM_Ref = 0.0;
double RefSpeed = 0.0;
double Voltage_fbk = 0.0;
double PWM_fbk = 0.0;
unsigned long t;

// PID controller gains set up
double Kp = 0.09, Ki = 1.1, FKi = 0.9, Kd = 0.0015;

// Init for loop parameters
double PastErr = 0.0;
double SumErr = 0.0;
double IntegralTerm = 0.0;

// Integers
int Encoder = 0;

int interruptFlag = 0;

// General function for MCU setting
void setup() {

  // "Set up" reading E4P / 360 CPR
  pinMode (ChA, INPUT_PULLUP);
  pinMode (ChB, INPUT_PULLUP);

  // H-bridge config
  pinMode (IN1, OUTPUT);
  pinMode (IN2, OUTPUT);
  pinMode (PWM, OUTPUT);
```

```
  // IN1 HIGH & IN2 LOW is CCW
  digitalWrite (IN1, HIGH);
  digitalWrite (IN2, LOW);

  attachInterrupt(digitalPinToInterrupt(ChA), Detect_ChA, CHANGE);

  // Serial comm set up
  Serial.begin (9600);
  noInterrupts (); // No interrputs setting

  // ATMega328P config
  TCCR1A = 0;
  TCCR1B = 0;
  Tmr_Ctr = 64286; // (0.02 seg)

  TCNT1 = Tmr_Ctr;
  TCCR1B |= (1 << CS12);
  TIMSK1 |= (1 << TOIE1);
  interrupts ();

}

// Controller selector
char ctrl = 'p'; // 'p' is PID, 'f' is Fuzzy, 'a' is PI

double Out = 0.0;
double Proportional = 0.0;
double Derivada = 0.0;
double Integral = 0.0;
double PWM2 = 0.0, Otp = 0.0;
double PIDPastErr = 0.0;
double FPWM = 0.0, Fuzzy = 0.0;
double f1 = 0.0, f2 = 0.0, f3 = 0.0;
double sug1 = 0.0, sug2 = 0.0, sug3 = 0.0;
```

```
double FIntegral = 0.0;
const double encoderScaling = 60.0/360.0/0.02;

// Loop execution
void loop() {

  // fixed freq
  if(interruptFlag == 1)
  {
    MtrCurrentSpeed = Encoder;
    Encoder = 0;
    MtrCurrentSpeed = MtrCurrentSpeed*encoderScaling;

    // Cast from int to double
      RefSpeed = (double)Speed;

      // Voltage calc for reference speed
      Voltage_Ref = Calc_Voltage (RefSpeed);

      // PWM calc depending on voltage reference
      //PWM_Ref = Calc_PWM (Voltage_Ref);

      // Current DC motor speed calc
      double GearCurrentSpeed = MtrCurrentSpeed / 50.89; // Eq 1

      // Current DC motor speed calc in volts
      Voltage_fbk = Calc_Voltage (GearCurrentSpeed); // Function 1

      // PWM feedback cacl
      //PWM_fbk = Calc_PWM (Voltage_fbk);

      // Error calc
      double VErr = Voltage_Ref - Voltage_fbk;

      // Control selector
```

```
switch (ctrl) {
  case 'p':
    Integral += Ki * VErr;
    Proportional = VErr * Kp;
    Derivada = (VErr - PastErr) * Kd;
    PWM2 = Integral + (VErr * Kp) + ((VErr - PastErr) * Kd);
    if (PWM2 < 19.1 & PWM2 > 0){
      Otp = PWM2;
    }
    else {
      if (PWM2 > 19.1) {
        Otp = 19.1;
      }
      else {
        Otp = 0.0;
      }
    }

    PIDPastErr = VErr;
    Serial.print ("Integral: ");
    Serial.print (Integral);
    Serial.print (", ");
    Serial.print ("Proportional: ");
    Serial.print (Proportional);
    Serial.print (", ");
    Serial.print ("Derivative: ");
    Serial.print (Derivada);
    Serial.print (", ");
    Serial.print ("PID_u: ");
    Serial.print (Otp);
    Serial.print (", ");
    Out = Calc_PWM (Otp);
    break;

  case 'f':
```

```
if (VErr <= -10.0) {
  f1 = 1.0;
  f2 = 0.0;
  f3 = 0.0;
}
else if (VErr > -10.0 & VErr < 0) {
  f1 = -0.1 * VErr;
  f2 = (0.1 * VErr) + 1;
  f3 = 0.0;
}
else if (VErr >= 0.0 & VErr < 10.0) {
  f1 = 0.0;
  f2 = (-0.1 * VErr) + 1;
  f3 = 0.1 * VErr;
}
else {
  f1 = 0.0;
  f2 = 0.0;
  f3 = 1.0;
}
sug1 = -0.1 * VErr;
sug3 = 0.1 * VErr;

Serial.print ("f1: ");
Serial.print (f1);
Serial.print (", ");
Serial.print ("f2: ");
Serial.print (f2);
Serial.print (", ");
Serial.print ("f3: ");
Serial.print (f3);
Serial.print (", ");

Fuzzy = ((sug1 * f1) + (sug2 * f2) + (sug3 * f3)) / (f1 + f2 + f3);
```

```
    Serial.print ("Fuzzy: ");
    Serial.print (Fuzzy);
    Serial.print (", ");

    FIntegral += FKi * VErr;

    Serial.print ("FIntegral: ");
    Serial.print (FIntegral);
    Serial.print (", ");

    FPWM = FIntegral + Fuzzy;

    Serial.print ("FP+I_u: ");
    Serial.print (FPWM);
    Serial.print (", ");

    if (FPWM < 19.1 & FPWM > 0){
      Otp = FPWM;
    }
    else {
      if (FPWM > 19.1) {
        Otp = 19.1;
      }
      else {
        Otp = 0.0;
      }
    }

    Out = Calc_PWM (Otp);
    break;
}

//double Out = Calc_PWM (Output);
```

```
  // Casting PWM value
  int OPWM = (int) Out;

  if (OPWM > 255) {
    analogWrite (PWM, 255);
  }
  else if (OPWM < 255 & OPWM > 0) {
    analogWrite (PWM, OPWM);
  }
  else {
      analogWrite (PWM, 0);
  }

t = millis ();
  Serial.print ("Time: ");
  Serial.print (t);
  Serial.print (", ");
  Serial.print ("Referencia: ");
  Serial.print (Speed);
  Serial.print (", ");
  //Serial.print ("RefSpeed: ");
  //Serial.print (RefSpeed);
  //Serial.print (", ");
  Serial.print ("Voltage: ");
  Serial.print (Voltage_Ref);
  Serial.print (", ");
  //Serial.print ("PWMRef: ");
  //Serial.print (PWM_Ref);
  //Serial.print (", ");
  Serial.print ("GearBox_Speed: ");
  Serial.print (GearCurrentSpeed, 2);
  Serial.print(",");
  Serial.print ("Voltage_fbk: ");
  Serial.print (Voltage_fbk);
  Serial.print (", ");
```

```
    //Serial.print ("PWM_fbk: ");
    //Serial.print (PWM_fbk);
    //Serial.print (", ");
    Serial.print ("Err_Voltaje: ");
    Serial.print (VErr);
    Serial.print (", ");
    //Serial.print ("Error_PWM: ");
    //Serial.print (Err);
    //Serial.print (", ");
    Serial.print ("OPWM: ");
    Serial.println (OPWM);

  interruptFlag = 0;
}


// Lectura del puerto serial
if (Serial.available() > 0) {
    String data = Serial.readString();
    Speed = data.toInt();
    }

    // Current reading
    // MsVoltageACS = GetMeasuredVoltage (Counts);
    // double Current = (((MsVoltageACS - 2.5) / Sens) + 0.065) * 1.0;



    //analogWrite (PWM, 50);



    // "Set up" for PWM value display
```

```
}

double Calc_Voltage (double RefSpeed) {

  double Ref = 0.0;
  Ref = RefSpeed * 0.216;

  return (Ref);
}

double Calc_PWM (double Voltage_Ref) {

  double refPWM = 0.0;
  float VR1 = Voltage_Ref * Voltage_Ref * Voltage_Ref;
  float VR2 = Voltage_Ref * Voltage_Ref;

  if (Voltage_Ref <= 0.5) {
    refPWM = 0.0;
  }
  else if (Voltage_Ref > 0.5 & Voltage_Ref < 19.1) {
    refPWM = (0.0192 * VR1) - (0.281 * VR2) + (4.51 * Voltage_Ref) + 36.96;
  }
  else if (Voltage_Ref >= 19.1) {
    refPWM = 255;
  }

  return (refPWM);
}

/*
double GetMeasuredVoltage (int Counts) {
    double ACS712_Voltage = 0.0;

    for (int i=0; i < Counts; i++) {
```

```
        ACS712_Voltage = ACS712_Voltage + analogRead (A1) * (5.0 / 1023);
      }

      ACS712_Voltage = ACS712_Voltage / Counts;
    return (ACS712_Voltage);
  }
*/

// Detecting positive flange for ChA
// X2 decoding
void Detect_ChA ()
{
     if(digitalRead(chA))
     { // rising edge
       if(digitalRead(ChB)
         Encoder -= 1;
       else
         Encoder += 1;
     }
     else
     { // falling edge
       if(digitalRead(ChB)
         Encoder += 1;
       else
         Encoder -= 1;
     }
}

// Rutina de interrupción
ISR(TIMER1_OVF_vect) {
    interruptFlag = 1;
    TCNT1 = Tmr_Ctr;
  }

/*
```

```
// Current sensor reading
double GetCurrent (int Iter) {
    double Volts;
    double SumCurrent = 0.0;

    for (int i=0; i < Iter; i++) {
        Volts = analogRead (A1) * 5.0 / 1024.0;
        SumCurrent += (Volts - 2.5) / Sens;
      }

    return (SumCurrent / Iter);
  }
  */
```