ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA MECÁNICA

"DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA PARA EL ESTUDIO COMPARATIVO DE UN CONTROLADOR PREDICTIVO (MPC) CON CONTROLADORES PID Y FUZZY, APLICADO AL CONTROL DE VELOCIDAD DE UN MOTOR DC"

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE MAGÍSTER EN MECATRÓNICA y ROBÓTICA

LUIS EDUARDO MAYORGA MIRANDA
lucho_mayorga@hotmail.com

DIRECTOR: Ph.D. JORGE ANDRÉS ROSALES ACOSTA
andres.rosales@epn.edu.ec

CO-DIRECTOR: Ph.D. OSCAR EDUARDO CAMACHO QUINTERO
oscar.camacho@epn.edu.ec

Quito, Noviembre 2018

CERTIFICACIÓN

Certificamos que el presente trabajo fue desarrollado por el señor **LUIS EDUARDO MAYORGA MIRANDA**, bajo nuestra supervisión.

_____

Ph.D. Andrés Rosales

DIRECTOR DE PROYECTO

_____

Ph.D. Oscar Camacho

CO-DIRECTOR DE PROYECTO

DECLARACIÓN

Yo, **Luis Eduardo Mayorga Miranda**, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondiente a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.

_____

Luis Eduardo Mayorga Miranda

DEDICATORIA

A Dios, a mi abuelito Octavio por ser siempre ejemplo e inspiración, a mis padres Washington y Elsa por el apoyo incondicional que siempre me han brindado, a mi hermana Diana y mi sobrina Dome por el cariño que siempre me han demostrado, a mi esposa Pamela y a nuestro hijo por su apoyo incondicional.

Luis.

AGRADECIMIENTO

Agradezco a Dios, por las bendiciones que siempre me ha brindado y por ser la luz en mi camino. A mis padres por el apoyo, los consejos y el cariño que siempre me han demostrado.

A los doctores Andrés Rosales y Oscar Camacho por las enseñanzas y concejos durante el postgrado y en la elaboración de este proyecto.

A mis amigos Erick Herrera, Jonathan Loor y Geovanny Romero los “Gurus”, por su gran amistad y todas las aventuras que compartimos durante este tiempo, que a pesar de que algunas no fueron positivas siempre el cariño de amigos no saco adelante.

A mi amigo José Luis Pulloquinga por ser cómplice de una carrera más.

De manera especial a mi gran amigo Fernando Saá, por ser un hermano en todos los aspectos de mi vida.

Al ingeniero Diego Ortiz por la amistad, los concejos y su ayuda.

Luis.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En el presente trabajo se desarrolló una plataforma para el estudio comparativo de un controlador predictivo (MPC) con controladores PID y FUZZY, aplicados al control de velocidad de un motor DC. Esta plataforma se desarrolló utilizando el modelado matemático de un motor DC, donde se analizaron las propiedades eléctricas constantes y variables físicas, para poder obtener una representación matemática abstracta del sistema, con lo cual, se obtuvieron el modelo en el espacio de estados y una función de transferencia en el dominio s. A continuación, se simularon subsistemas de control, para verificar su comportamiento. En el motor DC, se aplicó los controladores para poder tener una velocidad estable y preestablecida por el usuario; el motor DC está acoplado a un encoder incremental para la medición de la velocidad. El modelo matemático del motor DC implementado en un PC mediante el software LabVIEW permitió que el controlador MPC a través de la tarjeta Arduino MEGA controle el motor DC. Además, esto permitió comparar los controladores tradicionales como el PID y controladores modernos como el FUZZY, los cuales también fueron diseñados e implementados. Finalmente, a través de una interfaz de usuario desarrollada en LabVIEW se puede cambiar las variables, ajustar el SET, y de esta manera, ver y comparar las diferentes respuestas de cada controlador. El comportamiento de los controladores sobre el motor se analizó mediante dos métodos, con fuerza constante sobre el eje y sin fuerza sobre el eje, permitiendo establecer las diferencias, efectividad, desempeño entre los controladores y, así obtener el comportamiento más óptimo del sistema en análisis.

Palabras Clave: Motor DC, espacio de estados, función de transferencia, control predictivo basado en modelo, control PID, control FUZZY.

ABSTRACT

In the present work, a platform for the comparative study of a predictive controller (MPC) with PID and FUZZY controllers, applied to the speed control of a DC motor was developed. This platform was developed by performing the mathematical model of a DC motor, where the constant electrical properties and physical variables were analyzed in order to obtain an abstract mathematical representation of the system, which proceeded to obtain the model in the state space and a transfer function in the s domain.

Then we proceeded to simulate control subsystems in order to verify their behavior. In the DC motor the controllers were applied in order to have a preset and stable speed defined by the user; the DC motor is coupled to an incremental encoder for the measurement of the speed. The mathematical model of the DC motor implemented in a PC through the LabVIEW software allowed the MPC controller through the Arduino MEGA card to control the DC motor, as well as being able to compare the traditional controllers like the PID or FUZZY which were also designed and implemented.

Finally, with a user interface developed in LabVIEW, it is possible to change the variables, to adjust the SET, and by the way, different responses of each controller can be seen and compared. The behavior of the controllers on the motor was analyzed by means of two methods, with constant force on the axis and without force on the axis, allowing to establish the differences, effectiveness, performance between the controllers and, by this way, to obtain the optimum behavior of the analyzed system.

Keywords: DC Motor, state- space model, transfer function, model predictive control, PID control, FUZZY control.

“DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA PARA EL ESTUDIO COMPARATIVO DE UN CONTROLADOR PREDICTIVO (MPC) CON CONTROLADORES PID Y FUZZY, APLICADO AL CONTROL DE VELOCIDAD DE UN MOTOR DC”

INTRODUCCIÓN

Actualmente, el control en máquinas eléctricas se ha convertido en un campo de investigación de la ingeniería de amplio interés, dado las numerosas aplicaciones industriales específicamente el control de velocidad y posición de motores eléctricos de todo tipo.

Al ser escasas las herramientas de prueba de algoritmos de control eficientes como bancos de pruebas de motores fuera de línea, la calibración del mismo viene a representar deficiencias en optimización de tiempos de puesta en marcha una vez que ya estén instalados en alguna línea de producción o un sistema robótico. (Páucar, 2000) (Alvarado M. S., 2012)

Además, en el control de velocidad de los motores de corriente continua, se presenta el inconveniente que debido al fuerte acoplamiento entre las variables que intervienen en el proceso de operación del mismo, se tiene un sistema no lineal, con un rango limitado de comportamiento lineal (desde cero rpm y por debajo de la velocidad nominal); el sistema es multivariable, ya que por lo general el flujo magnético, el par electromagnético, velocidad y la posición son variables que comúnmente se presentan como salidas en el motor.

Otra deficiencia que se presenta en el control de motores de corriente continua, es encontrar un controlador óptimo con respuestas temporales requeridas, aplicando técnicas de control clásico o técnicas de control moderno, avanzado e inteligente. (Saá Tapia, 2017)

La costumbre de la utilización de controladores clásicos como el control ON/OFF y PID en la industria limitan la investigación de nuevos controladores sin poder identificar, comparar y analizar las ventajas y/o desventajas que el estudio y la implementación de un control avanzado podrían lograr. (Ati Villamarín, 2011)

Pregunta de Investigación

Con la implementación de un controlador avanzado y controladores clásicos se podrá comparar, mediante la interfaz de LabVIEW, el comportamiento diferente del motor DC con cada controlador, permitiendo así al usuario analizar y experimentar el comportamiento de la señal de control y velocidad del motor realizando modificaciones en cada controlador; y, de esta manera, poder determinar qué tipo de control sería el más óptimo para la aplicación.

Objetivo general

Diseñar e implementar una plataforma para el estudio comparativo de un controlador predictivo (MPC) con controladores PID y FUZZY, aplicado al control de velocidad de un motor DC.

Objetivos específicos

- Analizar e interpretar los conceptos y elementos básicos del control predictivo.
- Modelar un motor DC en variables de estado y función de transferencia, estableciendo los límites y restricciones de la planta.
- Diseñar un algoritmo de control predictivo, un algoritmo de control PID y un algoritmo de control FUZZY.
- Implementar los algoritmos de control diseñados en un PC mediante el software LabVIEW, para lo cual, se empleará la tarjeta electrónica Arduino MEGA cómo tarjeta de adquisición de datos.
- Desarrollar una interfaz HMI para toda la plataforma que permita manejar los controladores desarrollados.
- Realizar las pruebas necesarias y la comparación de los diferentes tipos de controladores.

Alcance

Para cumplir con los objetivos del proyecto se realizarán las siguientes actividades:

- Se realizará la recopilación de la información sobre el funcionamiento y operación del motor de corriente continua.
- Se desarrollará el modelado de la planta en variables de estado y función de transferencia en el dominio s mediante el software MATLAB;

- Se diseñará el algoritmo de control predictivo.
- Se diseñarán los controladores PID y FUZZY.
- Se realizarán simulaciones y correcciones de los algoritmos de control.
- Los controladores MPC, PID y FUZZY serán implementados en un PC mediante el software LabVIEW, para lo cual, se empleará la tarjeta electrónica Arduino MEGA cómo tarjeta de adquisición de datos.
- Se realizará una interfaz HMI para el manejo de la plataforma propuesta.
- Se realizará el análisis y comparación de las respuestas obtenidas con los diferentes tipos de controladores.

1. MARCO TEÓRICO

1.1. Sistemas de control

Las teorías de control que se manejan normalmente son la teoría de control clásica (también llamada teoría de control convencional) y la teoría de control moderno. Estas teorías están pensadas para cada tipo de sistemas, se tiene para el control clásico sistemas continuos, lineales e invariantes en el tiempo y para el control moderno sistemas digitales, lineales o no lineales, que generalmente usan sistemas en espacios de estados. (Córdova Mendoza, 2013)

El control automático ha desempeñado un rol esencial en el desarrollo de la ingeniería y la ciencia. El control automático se ha transformado en una parte importante en los sistemas de vehículos espaciales, en los sistemas robóticos, en los procesos modernos de fabricación y en cualquier operación industrial que requiera el control de temperatura, presión, humedad, flujo, velocidad, etc. (Ogata, 2010)

En la actualidad los sistemas de control industriales tienen que satisfacer criterios económicos, asociados con el mantenimiento de las variables de proceso en sus referencias, minimizando dinámicamente una función de coste de operación, criterios de seguridad y medioambientales, y de calidad en la producción, la cual debe satisfacer ciertas especificaciones sujetas a una demanda normalmente variable. (Chacón, Szigeti, & Camacho, 1996)

Por eso se puede considerar que, actualmente el principal objetivo de todo sistema de control consiste en actuar sobre las variables manipuladas de tal manera que puedan satisfacerse múltiples y cambiantes criterios de funcionamiento (económicos, de seguridad, medioambientales o de calidad) en presencia de cambios en las características del proceso.

La diferencia entre las diversas técnicas radica básicamente en los compromisos hechos en la formulación matemática de los criterios de funcionamiento y en la elección de la manera de representar el proceso. (Bolaños Paredes & Mayorga Miranda, 2015)

A continuación, la Figura 1.1 muestra las expectativas y posibilidades técnicas de las distintas técnicas de control.

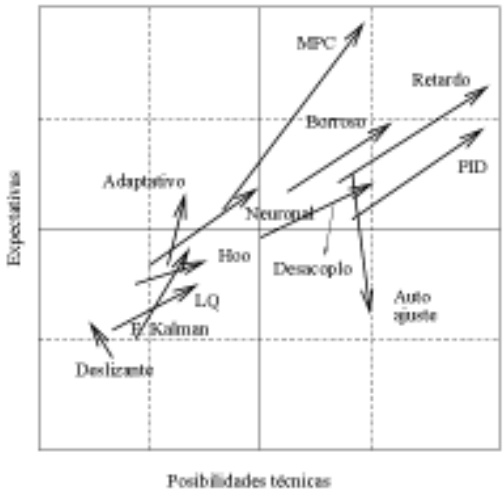

Figura 1.1 Expectativas y posibilidades técnicas
Fuente: (Bordons Alba, 2000)

1.2. Controlador PID

La mayoría de los controladores industriales que se utilizan en la actualidad usan esquemas de control PID o PID modificado.

La mayoría de los controladores PID se ajustan en el sitio, por lo que existen diversas reglas de sintonización, las cuales permiten llevar a cabo una sintonización delicada y fina de los controladores PID en el sitio. Se han creado métodos automáticos de sintonización y algunos controladores PID poseen la capacidad de sintonización automática en línea.

Dado que la utilización de los controladores PID estriba en que se aplican en forma casi general en la mayoría de los procesos industriales que tienen sistemas de control, pese al sorprendente desarrollo de la teoría de control y el soporte tecnológico que se utiliza para su implementación, el control PID se emplea casi con exclusividad en el ambiente industrial de todo el mundo. Generalmente, cuando el modelo matemático de la planta no se conoce y, no es posible emplear métodos de diseño analíticos, es cuando los controles PID resultan más útiles.

En el área de control de procesos, es un hecho común que los esquemas de control PID básicos y modificados han demostrado su utilidad para aportar un control satisfactorio, sin embargo, que en muchas situaciones específicas no aporten un control óptimo. (Pajarón Pérez, 2012)

Para el diseño de los controladores PID se puede utilizar las reglas de sintonía de Ziegler y Nichols, el método convencional de respuesta en frecuencia y el método de optimización computacional.

1.2.1. Reglas de Ziegler-Nichols para la sintonía de controladores PID

La Figura 1.2 muestra un control PID en una planta. Si se puede conseguir un modelo matemático de la planta, es posible la aplicación de varias técnicas de diseño con la finalidad de determinar los parámetros del controlador los cuales tienen que cumplir las especificaciones del transitorio y del estado estacionario del sistema en lazo cerrado. Sin embargo, si no se logra obtener el modelo matemático de la planta, tampoco es posible un método analítico para el diseño de un controlador PID. En este caso, se debe acudir a procedimientos experimentales para la sintonía de los controladores PID.

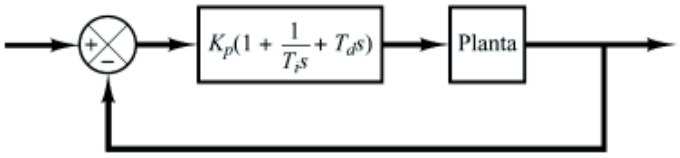

Figura 1.2 Control PID de una planta
Fuente: (Ogata, 2010)

El proceso para la selección de los parámetros del controlador que cumplan con las descripciones de comportamiento dadas, se conoce como sintonía del controlador.

Ziegler y Nichols recomendaron reglas para sintonizar los controladores PID (esto significa dar valores a las constantes Kp, Ti y Td) basándose en las respuestas escalón experimentales o en el valor de Kp que produce estabilidad marginal solo cuando se utiliza la acción de control proporcional. Estas reglas son muy convenientes cuando no se conocen los modelos matemáticos de las plantas, por su puesto, se puede utilizar para el diseño de sistemas conocidos.

Estas reglas recomiendan un conjunto de valores de Kp, Ti y Td que darán una operación estable al sistema. Sin embargo, el resultado del sistema puede presentar un gran sobreimpulso en su respuesta escalón, de manera que el resultado sea no aceptable. En este caso se deberá realizar una serie de ajustes finos hasta que se obtenga el resultado deseado.

Las reglas de Ziegler-Nichols para la sintonización controladores PID determinan los valores de la ganancia proporcional Kp, del tiempo integral Ti y del tiempo derivativo Td, basándose en las características de respuesta transitoria de una planta conocida. (Ogata, 2010)

1.3. Controlador FUZZY

Los controladores con Lógica Difusa aportan un mecanismo de inferencia que posibilita simular los procedimientos de razonamiento humano en sistemas basados en el conocimiento. Teóricamente la lógica difusa proporciona un marco matemático que posibilita modelar la incertidumbre de los procesos cognitivos humanos de forma que puedan ser tratados por medio de un computador. (González Morcillo, 2009)

La lógica difusa está basada en la teoría de conjuntos difusos propuestos por Zadeh en 1965. La cuál indica que, en muchos casos, las clases de objetos del mundo real no poseen criterios de membresía precisamente definidos. El elemento de un conjunto clásico pertenece o no a un determinado conjunto, el elemento de un conjunto difuso admite grados de pertenencia y permite formalizar conceptos tales como “alto”, “bajo”, “rápido”, “frío” comúnmente usados por esencias no precisa, el conjunto difuso permite representar de una mejor manera ciertos tipos de incertidumbre. (Iglesias Sánchez, García, Sanjuan, Camacho, & Smith, 2007)

Inicialmente para construir un sistema difuso se debe obtener una colección de reglas difusas basadas en el conocimiento humano. Posteriormente, se debe combinar estas reglas en un sistema simple. Los diferentes sistemas difusos emplean diferentes principios de esta combinación.

Existen tres tipos de sistemas difusos empleados comúnmente:

- Sistemas difusos puros o de Mamdani,
- Sistemas difusos Takagi-Sugeno-Kang (TSK), y
- Sistemas difusos con fuzzyficador y defuzzyficador.

No solo se tiene como aplicación a los sistemas de control, la Lógica Difusa se puede utilizar en distintos tipos de instrumentos, máquinas y en diversos ámbitos de la vida cotidiana. (Vera, 2011)

1.4. Controlador MPC (Model Predictive Control)

El Control Predictivo Basado en Modelo es parte de las técnicas de control avanzado que han tenido injerencia en la industria, este permite la utilización de varios algoritmos, los cuales utilizan un modelo dinámico del proceso a controlar, para establecer una predicción de las acciones de control futuras en la salida, las mismas que se determinan con la minimización del error predicho sujeto a las restricciones de operación del sistema.

El MPC no es una estrategia de control específica, más bien se abarca un campo muy amplio de métodos de control desarrollados en torno a ciertas ideas comunes. Las ideas que surgen en mayor o menor medida en toda la familia de controladores predictivos son básicamente:

- Uso explícito de un modelo para predecir la salida del proceso en futuros instantes de tiempo (horizonte).
- Cálculo de las señales de control minimizando una cierta función objetivo.
- Estrategia deslizante, de forma que en cada instante el horizonte se va desplazando hacia el futuro, lo que implica aplicar la primera señal de control en cada instante y desechar el resto, repitiendo el cálculo en cada instante de muestreo.

El Control Predictivo es un tipo de control de naturaleza abierta dentro del cual se han desarrollado muchas realizaciones, encontrando gran aceptación en aplicaciones industriales.

En la actualidad existen numerosas aplicaciones de controladores predictivos funcionando con éxito, tanto en la industria de procesos como en control de motores o robótica. El buen funcionamiento de estas aplicaciones muestra la capacidad del MPC para conseguir sistemas de control de elevadas prestaciones capaces de operar sin apenas intervención durante largos períodos de tiempo. (Bordons Alba, 2000) (Bolaños Paredes & Mayorga Miranda, 2015)

Una manera más simple de entender el MPC es mediante la representación de ejemplos. Como se ilustra en Figura 1.3 Analogía del MPC con la conducción de un automóvil., un conductor de un automóvil que en el instante *k* sabe cuál es su trayectoria deseada para un horizonte de tiempo finito *hp*, tomando en cuenta las características del vehículo y del entorno, el conductor realiza un modelo mental y decide que secuencia de acciones de control realizar (acelerar, frenar y maniobrar) para seguir la trayectoria deseada sobre la ruta, teniendo en cuenta que solo la primera acción de control es tomada, a continuación

el procedimiento se repite en el instante siguiente *k+1*. (Limón Marruedo, Alamo, & Camacho, 2002)

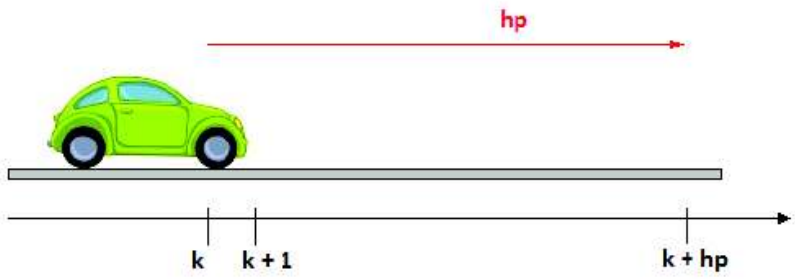

Figura 1.3 Analogía del MPC con la conducción de un automóvil.
Fuente: (Limón Marruedo, Alamo, & Camacho, 2002)

1.5. Motor de Corriente Continua

También llamado motor de corriente directa, motor CC o motor DC, es una máquina que transforma la energía eléctrica en energía mecánica, provocando un movimiento rotatorio, gracias a la acción de un campo magnético. En algunas modificaciones, ejercen tracción sobre un riel, a estos motores se les conoce como motes lineales.

El motor DC se compone de dos partes: el estator que es el que da soporte mecánico al aparato y contiene los polos de la máquina, que pueden ser de dos tipos: devanados de hilo de cobre sobre un núcleo de hierro o imanes permanentes y el rotor que es generalmente de forma cilíndrica, también devanado y con núcleo, alimentado con corriente continua a través de delgas, que están en contacto alternante con escobillas fijas (también llamadas carbones).

Algunas aplicaciones de los motores pueden verse en los motores lineales, servomotores y motores paso a paso, también existen motores de DC sin escobillas los cuales son comúnmente utilizados en el aeromodelismo por su bajo par motor y su gran velocidad.

Es posible controlar la velocidad y el par de estos motores utilizando diversas técnicas de control. (Geekbot electronics, 2018)

Figura 1.4 Ejemplo de motor de corriente continua.
Fuente: https://es.rs-online.com/web/p/motores-dc/5366046/

El modelado matemático del motor DC requiere de dos ecuaciones, una ecuación mecánica y otra ecuación eléctrica. Estas ecuaciones están relacionadas y se fundamentan en las Leyes de la dinámica y de Kirchhoff. La ecuación mecánica modela principalmente el movimiento del rotor y la ecuación eléctrica modela lo que ocurre en el circuito eléctrico del inducido. (Alvarado M. S., 2012)

1.6. Sensores de Velocidad

El sensor, es un dispositivo que posee la capacidad de detectar acciones o estímulos externos y responder en consecuencia. Son capaces de transformar las magnitudes físicas o químicas en magnitudes eléctricas.

Los sensores, en resumen, son instrumentos que permiten obtener información del entorno e interactuar con ella. Un ejemplo, los seres humanos apelan a su sistema sensorial para dicha tarea, las máquinas y robots necesitan de sensores para la interacción con el medio en el que se encuentran. Un ejemplo de aplicación de los sensores sería en la Industria automotriz, robótica, industria aeroespacial, medicina, industria de manufactura, etc.

Los sensores se pueden conectar a un computador para obtener ventajas como son el acceso a la toma de valores desde el sensor, una base de datos, etc. (Definición, 2018)

1.6.1. Encoder

El encoder es un transductor rotativo, que mediante una señal eléctrica se puede indicar la posición angular de un eje, la velocidad y la aceleración del rotor de un motor.

Un encoder se compone de un disco conectado a un eje giratorio. El disco está hecho de vidrio o plástico y se encuentra “codificado” con unas partes transparentes y otras opacas que bloquean el paso de la luz emitida por la fuente de luz (típicamente emisores

infrarrojos). En la mayoría de los casos, estas áreas bloqueadas (codificadas) están arregladas en forma radial.

Mientras el eje rota, el emisor infrarrojo emite luz que es recibida por el sensor óptico (o foto-transistor) generando los pulsos digitales a medida que la luz cruza a través del disco o es bloqueada en diferentes secciones de este. Esto produce una secuencia que puede ser usada para controlar el radio de giro, la dirección del movimiento e incluso la velocidad.

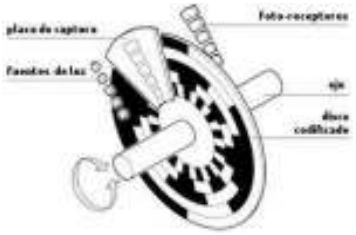

Figura 1.5 Representación básica de un encoder.
Fuente: http://www.ingmecafenix.com/automatizacion/encoder/

Un tipo de encoder es el encoder incremental, el cuál es un encoder que determina el ángulo de posición por medio de la realización de cuentas incrementales.

Es decir, el encoder incremental provee una posición estratégica desde donde siempre comenzará la cuenta. La posición actual del encoder es incremental cuando es comparada con la última posición registrada por el sensor. (Ingmecafenix, 2018)

Figura 1.6 Ejemplo de un encoder incremental de cuadratura.
Fuente: http://www.ingmecafenix.com/automatizacion/encoder/

1.7. Modelado de Sistemas

Los modelos matemáticos son hoy en día un medio de trabajo imprescindible para el control de procesos. Se puede realizar un modelo del proceso a regular, de su entorno y de sus leyes de control. No existe un modelo único, sino una serie de modelos. Los modelos más sencillos nos muestran el comportamiento, los más complejos reproducen el comportamiento del sistema real con mayor fidelidad.

Para el diseño de un modelo para cualquier sistema se debe partir de una predicción de su funcionamiento antes del diseño a detalle del sistema. La predicción se basa en una descripción matemática de las características dinámicas del sistema. A esta descripción matemática se la llama modelo matemático.

Generalmente el modelo matemático trata de una serie de ecuaciones diferenciales que describen el comportamiento del sistema (modelo teórico).

1.7.1. Sistemas lineales y no lineales:

- *Sistema Lineal:* Las ecuaciones que describen el modelo son lineales, se aplica el principio de superposición (ante dos entradas la salida es la suma de las respuestas individuales).
- *Sistema No lineal:* No se aplica el principio de superposición. Existe dificultad matemática y normalmente se les aproxima a modelos matemáticos lineales.

1.7.2. Sistemas dinámicos y estáticos:

- *Sistema Dinámico:* Si su salida en el presente depende de una entrada en el pasado.
- *Sistema Estático:* Su salida en curso depende de la entrada en curso. En este caso la salida no cambia si la entrada no cambia. En el dinámico la salida cambia con el tiempo cuando no está en equilibrio.

Validación del Modelo

Tanto en el análisis teórico como en el experimental, después de obtener el modelo es necesaria su coincidencia con el modelo teórico real. Esto se denomina la validación.

Los métodos para validar un modelo pueden ser:

- Analizar la respuesta del modelo (al escalón, al impulso, etc.).
- Análisis de polos y ceros del sistema.
- Calcular determinadas relaciones estadísticas.
- Investigar las variaciones de aquellas magnitudes que sean especialmente sensibles a cambios en los parámetros del modelo. (Crespo & Pendino, 2015)

1.8. Función de Transferencia

Se denomina función de transferencia a la relación entre las transformadas de la salida y de la entrada, sin tener en cuenta el efecto de las condiciones iniciales:

$$G(s) = \frac{B(s)}{A(s)} = \frac{Y(s)}{U(s)}_{c.i.\ nulas}$$

Se denominan polos a las raíces del denominador y ceros a las raíces del numerador de la función de transferencia. El orden de esta debe coincidir con el grado del polinomio característico *A(s)* y corresponde al orden de la ecuación diferencial asociada. Un sistema es de orden mínimo si no hay cancelaciones entre polos y ceros en la función de transferencia.

Una función de transferencia es propia si el orden del numerador es menor o igual que el del denominador. Si es menor, es estrictamente propia. La mayoría de los sistemas reales tienen funciones de transferencia propias. (Cartagena, 2018)

1.9. Espacios de Estado

El estado en un sistema dinámico es el conjunto de variables (denominadas variables de estado), el conocimiento de esas variables en un determinado instante t_0 junto con el conocimiento de los valores de la señal de entrada para los instantes $t \geq t_0$, permite determinar el comportamiento y evolución del sistema para cualquier instante de tiempo $t \geq t_0$.

Las variables de estado se agrupan en un *vector de estado* y el espacio *n-dimensional* que determinan los posibles valores de esas variables, se denomina espacio de estados. La dinámica de un sistema se puede describir en función del valor del vector de estados y de la señal de entrada

De forma general la representación en espacio de estados de un sistema lineal, se escribe de la siguiente manera:

$$\dot{X}(t) = A(t)x(t) + B(t)u(t)$$

$$Y(t) = C(t)x(t) + D(t)u(t)$$

dónde: *x(t)* vector de estados, *Y(t)* vector de salida, *u(t)* vector de entradas (o control), *A(t)* matriz de estados, *B(t)* matriz de entrada, *C(t)* matriz de salida y *D(t)* es la matriz de

transmisión directa que por simplicidad normalmente se toma como la matriz cero. (Rodríguez Ramírez & Bordóns Alba, 2005)

2. METODOLOGÍA

2.1. Requerimientos de Hardware y Software

2.1.1. Motor de corriente continua POLOLU

Para la implementación de los controladores PID, FUZZY y MPC, se dispone de un motor de corriente continua POLOLU, el cual es un motor con caja de reducción 30:1 integrado a un encoder de cuentas por revoluciones (CPR) o incremental de cuadratura, el cual provee una resolución de 64 conteos por revolución en el eje del motor, que corresponden a 1920 conteos por revolución en el eje a la salida de la caja reductora.

Tabla 2.1 Especificaciones Generales del motor POLOLU 37D

Descripción	Características
Alimentación	**12 Vdc**
Motoreductor	30:1
Velocidad sin carga @ 12Vdc	350 rpm
Corriente sin carga @ 12Vdc	300 mA
Tamaño	37D x 68L mm
Peso	215 g
CPR a la salida de la Caja de engranajes	1920

(Fuente: Autor)

En la Figura 2.1 Motor POLOLU 37D mm con caja reductora y encoder de 64 CPR.se puede observar el motor de corriente continua que será utilizado en este proyecto.

Figura 2.1 Motor POLOLU 37D mm con caja reductora y encoder de 64 CPR.
Fuente: https://www.pololu.com/product/2823/specs

2.1.2. Tarjeta ARDUINA MEGA

Para la comunicación entre motor DC y la PC se utilizará la tarjeta de microcontrolador basado en el ATmega2560 (ARDUINO MEGA 2560), la cual cuenta con 54 pines de

entrada / salida digital (de los cuales 15 se pueden usar como salidas PWM), 16 entradas analógicas, 4 UART (puertos serie de hardware), un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, un cabezal ICSP, y un botón de reinicio. (Arduino Store, 2018)

Figura 2.2 Tarjeta ARDUINO MEGA 2560
Fuente: https://store.arduino.cc/usa/arduino-mega-2560-rev3

2.1.3. Driver IBT-2 de ARDUINO

Este driver utiliza dos chips Infineon BTS 7960 de medio puente para alta corriente para aplicaciones de motores.

La conectividad con un microcontrolador o Arduino se hace sencilla con la utilización de este controlador que cuenta con sensores de corriente, ajuste de velocidad de giro y protecciones contra sobre temperaturas, sobre tensión, bajo voltaje, sobre intensidad y corto circuito.

Este driver de tamaño pequeño proporciona una solución de costo optimizado para motores que utilizan PWM. En la Figura 2.3 Driver IBT-2 para motores DC. se puede observar una imagen referencial de este driver. (Electronics, 2018)

Figura 2.3 Driver IBT-2 para motores DC.
Fuente: (Electronics, 2018)

2.1.4. Computadora y programas de Software

Para el diseño, programación, control, comparación y obtención de resultados se utilizar la computadora y los programas de Software que se describen en la Tabla 2.2 *Características de PC y Softwares*.

Tabla 2.2 Características de PC y Softwares.

Descripción	Características
Computador	Toshiba Satellite S55-A5169, Intel Core i7-4700MQ 2.4GHz, 12 GB DDR3.
Software LabVIEW	NI LabVIEW 2013 SP1
Software MATLAB	MATLAB R2015b
Software ARDUINO IDE	LIFA_Base 1.8.3

Fuente: Autor

2.2. Diseño de los algoritmos de control

2.2.1. Modelado del sistema

Para realizar el modelo matemático de la planta (Motor DC POLOLU), primero se realiza la adquisición de datos, para poder obtener los datos hay que realizar las conexiones electrónicas como se muestran en el ANEXO 1. El modelamiento de la planta tiene como objetivo adquirir un comportamiento lo más real posible del sistema.

En la actualidad, con la ayuda de los diferentes tipos de Software que se disponen, es posible realizar un modelo dinámico de cualquier sistema que se desee controlar, ya que realizarlo de forma manual, resulta tedioso y poco confiable.

2.2.1.1 Adquisición de datos

Cuando se desea obtener un modelo lo más real de un motor DC, el primer paso es controlarlo de forma manual para así variar su velocidad (PWM) y alcanzar los límites mínimos y máximos de operación del encoder (CPR), para después poder tabularlos y realizar el análisis de los datos obtenidos.

En una VI de LabVIEW se realiza el HMI y la programación para la adquisición de datos. La Figura 2.4 Programación del Diagrama de Bloques para la adquisición de datos. muestra la programación del diagrama de bloques para la obtención y tabulación de los datos.

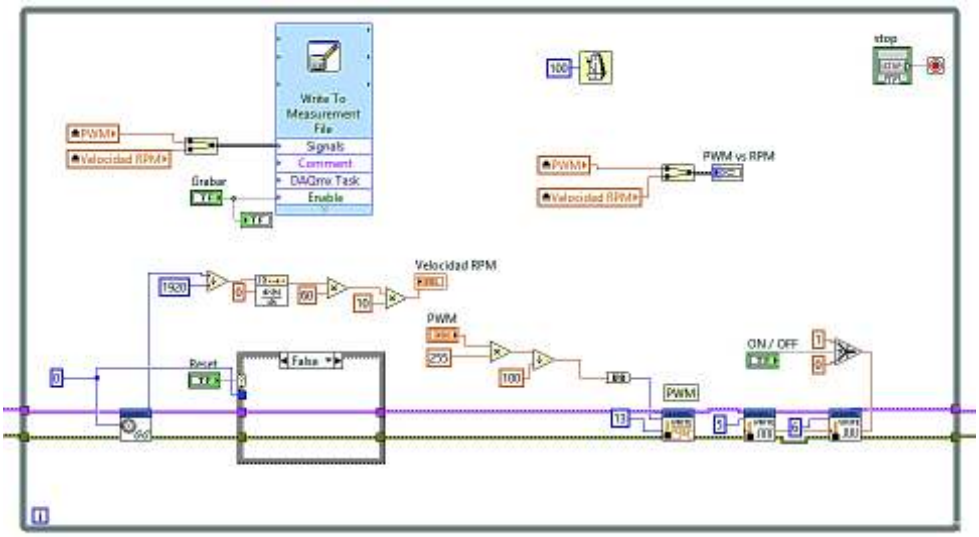

Figura 2.4 Programación del Diagrama de Bloques para la adquisición de datos.
Fuente: Autor

Al generar los datos a través de LabVIEW, estos son exportados a un archivo de Excel (.xml), En donde se debe depurar los primeros y los últimos datos ya que estos generalmente son valores basura que se producen al momento de iniciar/finalizar la comunicación entre la tarjeta ARDUINO MEGA 2560 y la PC.

La Figura 2.5 Depuración manual de datos basura indica un segmento de los resultados al inicio de la lectura obtenidos, los cuales deben ser depurados.

TIEMPO [Seg]	PWM [0-100%]	RPM [0-350 RPM]
5.199801	0	0
5.299469	0	0
5.399624	0	0
5.500261	0	0
5.60017	0	0
5.69959	0	0
5.80084	0	0
5.899789	0	0
5.999733	0	0
6.099689	0	0
6.199623	0	0
6.30051	0	0
6.399575	0	0
6.499506	0	0
6.600616	0	0
6.699629	0	0
6.800317	0	0
6.899375	0	0
6.99951	0	0
7.099334	0	0
7.200168	25	0
7.299445	25	0

Figura 2.5 Depuración manual de datos basura (en rojo) producidos al inicio de la lectura de datos.
Fuente: Autor

También se debe depurar los datos que son producto de los errores de la comunicación y de la lectura del encoder, ya que estos se encuentran fuera del rango de tolerancia y pueden alterar el resultado del modelo final. En la se muestra la depuración de estos errores.

TIEMPO [Seg]	PWM [0-100%]	RPM [0-350 RPM]
15.500042	50	170.78125
15.599469	75	171.09375
15.700148	75	170.46875
15.80051	75	185.9375
15.899415	75	224.0625
15.999648	75	247.5
16.09978	75	251.71875
16.200037	75	254.0625
16.299939	75	254.21875
16.412436	75	254.21875
16.499703	75	270.46875
16.599843	75	254.21875
16.701215	75	237.8125
16.800265	75	255.9375
16.89942	75	255
16.999474	75	252.1875
17.099373	75	253.4375
17.217135	75	254.375
17.305722	75	276.5625
17.400194	75	262.65625
17.499358	75	233.125
17.60092	75	246.25

Figura 2.6 Depuración manual de datos basura producto de la comunicación y el encoder.
Fuente: Autor

2.2.1.2 Modelado en MATLAB

Posterior a la adquisición, análisis y depuración de los datos obtenidos del motor DC, se procede a ingresarlos al Workspace de MATLAB, para así poder utilizarlos en el toolbox de IDENT del software MATLAB y realizar el modelamiento del motor DC.

Para poder tener disponible la información en el Workspace de MATLAB, se debe realizar los siguientes pasos:

- Crear dos variables nuevas en MATLAB (PWM y RPM). Para esto se debe dirigir a la barra de tarea de MATLAB de la siguiente manera *HOME>New Variable*.
- Ingresar los datos depurados de cada variable (Tabla de Excel) según correspondan a cada columna de las nuevas variables creadas en MATLAB.

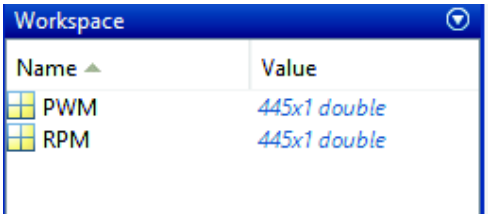

Figura 2.7 Variables PWM y RPM creadas en MATLAB.
Fuente: Autor

Una vez se tiene las variables del proceso, se debe escribir en el *Command Window* de MATLAB la palabra **IDENT**, para que se ejecute la herramienta de sistema de identificación que tienen MATLAB.

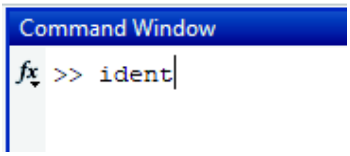

Figura 2.8 Ejecución de la herramienta IDENT en MATLAB
Fuente: Autor

Posterior a la ejecución de la herramienta IDENT, para poder importar y tener a disposición los datos ingresados en las variables que se crearon en el inicio de este punto, se debe realizar los siguientes pasos:

- Seleccionar la opción de *Time domain data*, para que se ejecute la ventana de importación de datos.

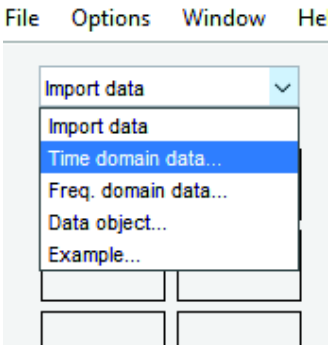

Figura 2.9 Selección de la opción "Datos en dominio del tiempo" dentro de la herramienta IDENT.
Fuente: Autor

- Una vez ejecutada la ventana de importación de datos se debe ingresar las variables creadas anteriormente, con los nombres según corresponda a la entrada y salida de nuestro sistema (Entrada: PWM y Salida: RPM). De igual manera y muy

importante se debe colocar el tiempo de muestreo (Sample time) que en este caso corresponde a un valor de 100 mili segundos que es el tiempo con el que se adquirió los datos en LabVIEW. Se coloca el tiempo de inicio (Starting time) en cero ya que en la obtención de datos se partió desde que el motor estaba parado. Se puede colocar un nombre específico para nuestro sistema, así como definir las unidades de su entrada y salida si se presiona la opción *more*.

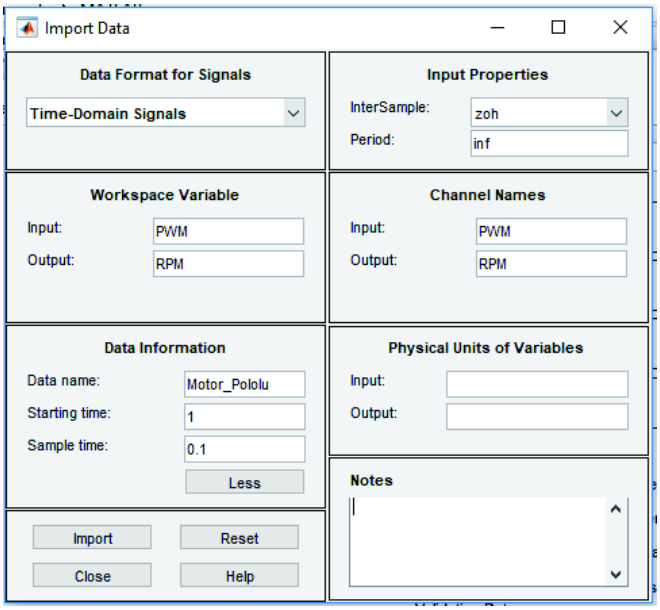

Figura 2.10 Configuración de la ventana de importación de datos.
Fuente: Autor

- Se procede a verificar que los datos ingresados estén correctos de acuerdo al comportamiento del motor DC. Esto se realiza señalando los datos del modelo nuevo que se creó en el sistema de identificación y colocando un visto en la opción de *Time plot.* En la se puede observar el comportamiento del motor a los diferentes cambios en su entrada (PWM).

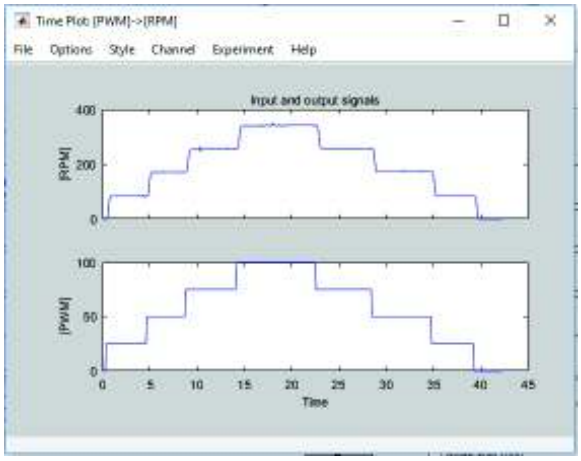

Figura 2.11 Gráfica del comportamiento del motor DC.
Fuente: Autor

2.2.1.3 Función de Transferencia

Una vez ingresado los datos del motor DC al sistema de identificación IDENT de MATLAB, se procede a la obtención de la función de transferencia la cual se utilizará para el controlador PID. La cual se debe obtener realizando los siguientes pasos:

- Seleccionar la opción *Process Models* en la pestaña de estimación del IDENT.

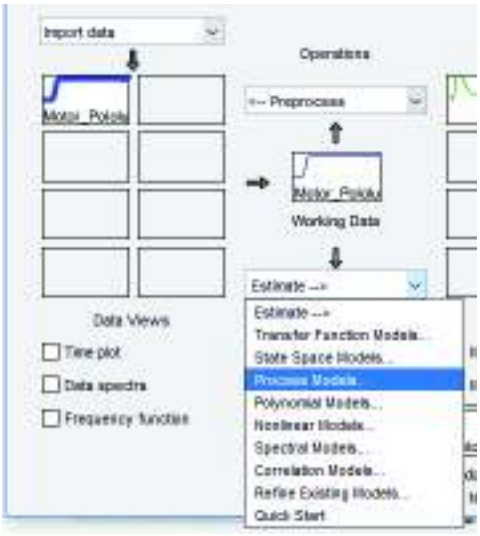

Figura 2.12 Selección del "Modelo de Proceso".
Fuente: Autor

- Ejecutada la ventada de modelado de proceso, se debe quitar el visto en la opción de *Delay*, así como en *Disturbance Model* seleccionar la opción *Order 1*, ya que se

conoce que la mayoría de motores de corriente continua son sistemas de primer orden. Se pone *Estimar* y obtendremos los valores para la función de transferencia.

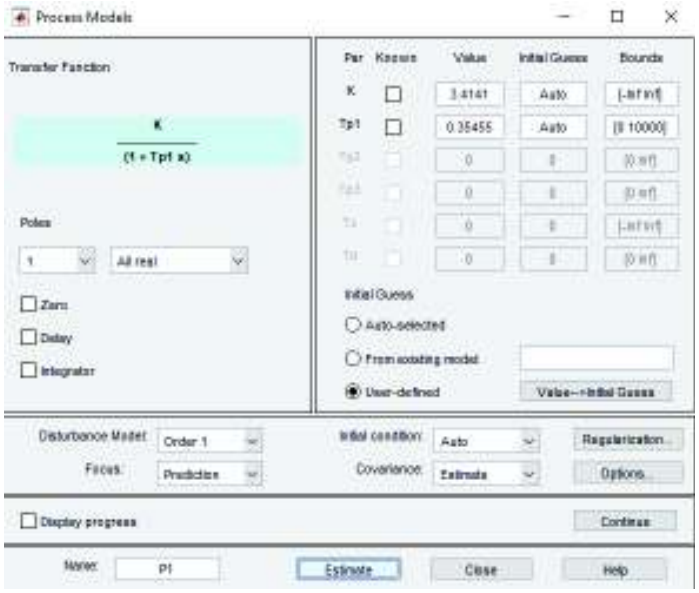

Figura 2.13 Configuración del modelo de proceso.
Fuente: Autor

- Se procede a la validación de la función de transferencia con la salida de la planta, señalando el modelo p1 y colocando un visto en la opción *Model output* en la ventana principal del IDENT.

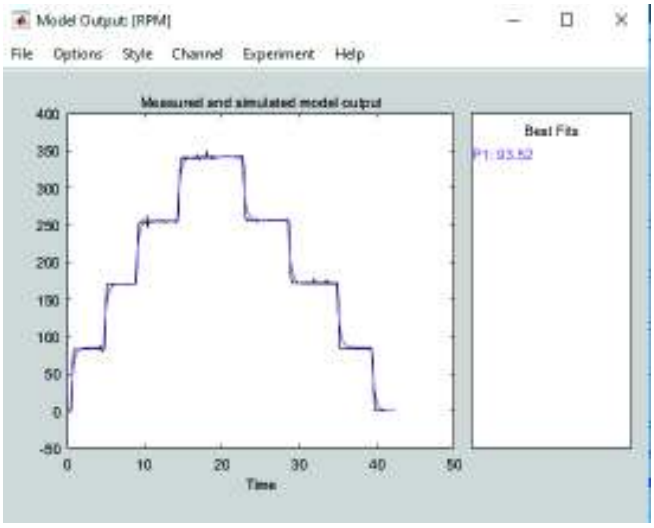

Figura 2.14 Validación del modelo de la Función de Transferencia con la salida de la planta.(Negro: Salida de la planta – Lila: Salida estimada del modelo de la función de transferencia)
Fuente: Autor

- Como se obtuvo un resultado de 93.52% en la validación del modelo, se puede trabajar con la función de transferencia obtenida. En la Figura 2.15 Función de Transferencia del motor DC POLOLU obtenida mediante la herramienta IDENT de MATLAB. se observa la función de transferencia final.

```
Process model with transfer function:
                Kp
  G(s) = ----------
           1+Tpl*s

         Kp = 3.4141
        Tpl = 0.35455
```

Figura 2.15 Función de Transferencia del motor DC POLOLU obtenida mediante la herramienta IDENT de MATLAB
Fuente: Autor

2.2.1.4 Modelado Espacios de Estado

Para la obtención del modelo del motor DC POLOLU en espacios de estado (State-Space) que se utilizara en el controlador MPC se debe realizar los siguientes pasos:

- Seleccionar la opción *State Space Models* en la pestaña de estimación del IDENT.

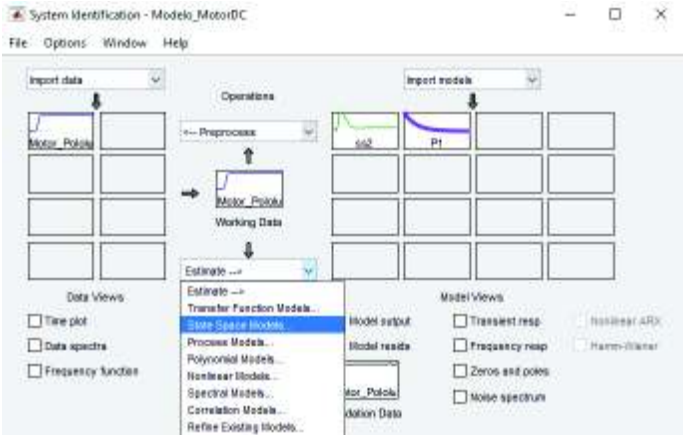

Figura 2.16 Selección del "Modelo en Espacios de Estado"
Fuente: Autor

- Ejecutada la ventana de Modelo en Espacios de Estado, se debe seleccionar la opción *Pick best value in the range* y la opción *Discrete-time (Ts=0.1)* esta opción de tiempo se ajusta al tiempo de muestra (100 mili segundos) que se utilizó en la adquisición de datos en LabVIEW. Se hace click en la opción *Estimate* y se espera la respuesta.

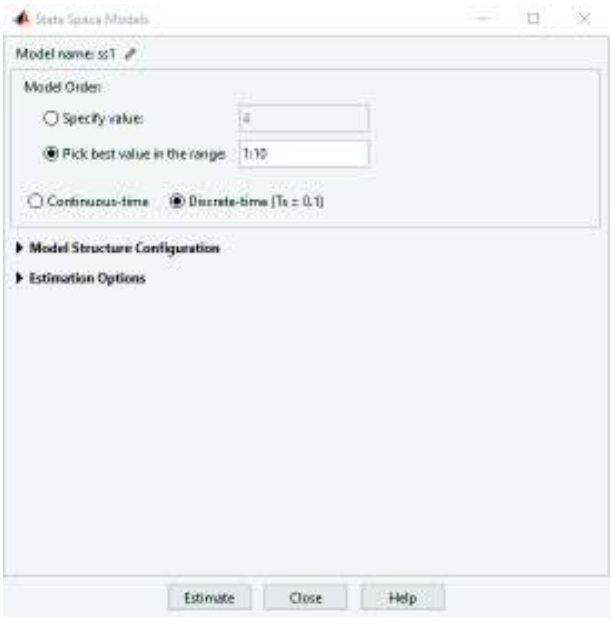

Figura 2.17 Configuración del Modelo en Espacios de Estado.
Fuente: Autor

- Se abre una nueva ventada en donde se debe seleccionar el orden del modelo. Por Default el programa recomienda un modelo de orden 4, sin embargo, al tratarse de un motor DC y si la validación de modelos inferiores al orden que se recomienda tiene un porcentaje alto, es recomendable elegir un modelo de orden inferior al sugerido por el programa. En este caso se procede a estimar un modelo de orden 4 y de orden 2. Colocando el número de orden del modelo en la opción *order* y dando click en el botón *Insert*.

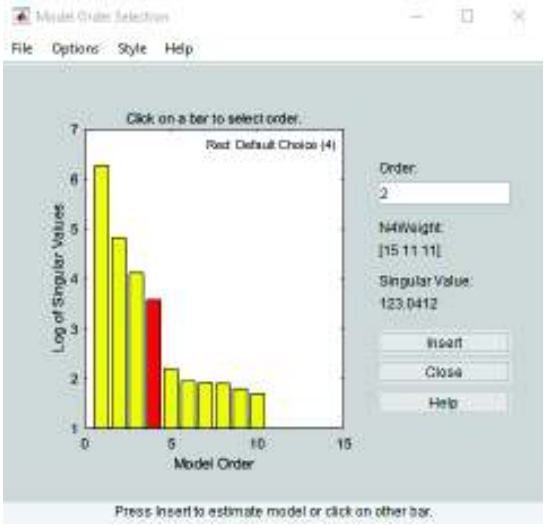

Figura 2.18 Selección del Orden del Modelo en Espacios de Estado.
Fuente: Autor

- Se procede a la validación del modelo en espacios de estado con la salida de la planta, señalando los modelos *ss4* y *ss2* y colocando un visto en la opción *Model output* en la ventana principal del IDENT

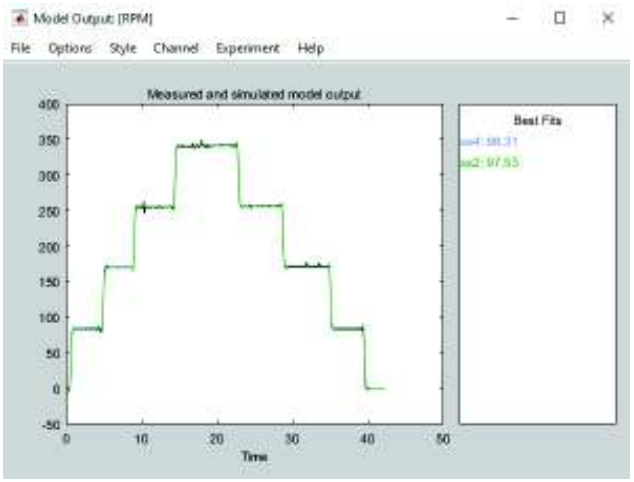

Figura 2.19 Validación de los modelos de Espacios de Estado con la salida de la planta. (Negro: Salida de la planta – Verde: Salida estimada del modelo en espacio de estado)
Fuente: Autor

- Comparando el porcentaje en la validación de los modelos obtenidos, se puede observar que en los dos casos son menores al 5%, por lo que se decidió trabajar con un modelo de orden 2 el cual tiene un porcentaje de validación del 97.53%. En la Figura 2.20 Modelo en Espacio de Estados del motor DC POLOLU obtenida mediante la herramienta IDENT de MATLAB.se observa las matrices del modelo final en espacios de estado.

```
Discrete-time identified state-space model:
  x(t+Ts) = A x(t) + B u(t) + K e(t)
     y(t) = C x(t) + D u(t) + e(t)
A =
         x1      x2
 x1  0.7027  -0.187
 x2  0.6573  0.2576
B =
         [PWM]
 x1  -0.001455
 x2    0.01343
C =
             x1      x2
 [RPM]  -376.5  -59.47
D =
        [PWM]
 [RPM]      0
```

Figura 2.20 Modelo en Espacio de Estados del motor DC POLOLU obtenida mediante la herramienta IDENT de MATLAB.
Fuente: Autor

2.2.2. Diseño del controlador PID

Para el diseño y simulación del controlador PID, se utiliza el software LabVIEW.

2.2.2.1 Simulación del Controlador PID

La simulación se realiza en un Virtual Instrumentation (VI) de LabVIEW. En el diagrama de bloques se crea una estructura de simulación denominada *Control & Simulation Loop*, dentro de esta estructura se agrega: la función de transferencia encontrada anteriormente, una señal escalón y el subVI de PID, este último permite agregar las ganancias, Set, retroalimentación del sistema para realizar la simulación.

En la Figura 2.21 Programa en el diagrama de bloques en LabVIEW de la simulación del control PID. se puede observar el diagrama de bloques de la programación para la simulación del controlador PID. La función de trasferencia ingresada es única del sistema y no se puede cambiar mientras se ejecuta la simulación, en el panel frontal se puede realizar los cambios para los valores de las ganancias del PID, al igual que reinicializar la simulación.

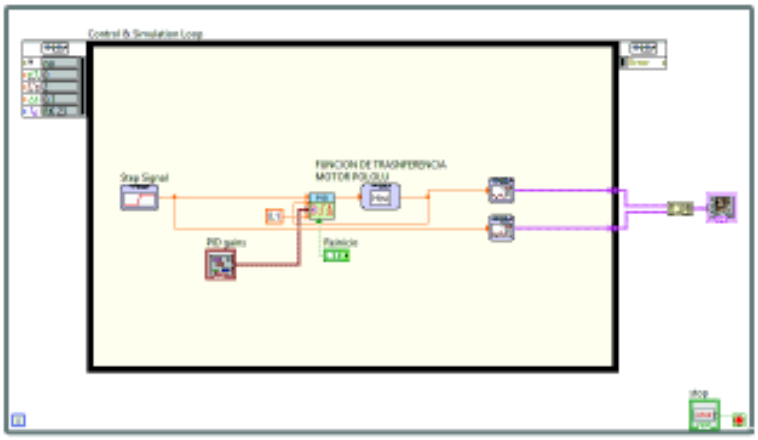

Figura 2.21 Programa en el diagrama de bloques en LabVIEW de la simulación del control PID.
Fuente: Autor

2.2.2.2 Implementación del control PID

Para la implementación del controlador PID y para realizar la plataforma comparativa con los otros dos controladores (FUZZY y MPC), se crea un proyecto nuevo en LabVIEW, el cual tendrá como programa principal el VI nombrado **"HMI"** que será el programa que se ejecuta en la computadora y mediante el cual se podrá realizar la comunicación entre la tarjeta Arduino y el Computador, así como también será el encargado de procesar los diferentes algoritmos de los controladores.

Figura 2.22 Proyecto creado en LabVIEW que contiene el VI del HMI.

Fuente: Autor

Una vez creado el Interfaz hombre-máquina (HIM), en el diagrama de bloques se realiza la programación para la lectura y escritura de la tarjeta Arduino, lectura del encoder y PWM del motor. En la Figura 2.23 Sección de la programación de la comunicación con la tarjeta Arduino MEGA 2560. se puede observar la programación para la comunicación con la tarjeta Arduino, el acondicionamiento del encoder y la salida del PWM.

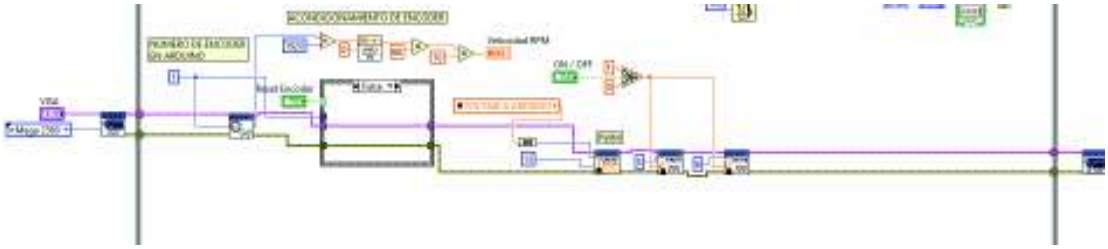

Figura 2.23 Sección de la programación de la comunicación con la tarjeta Arduino MEGA 2560.
Fuente: Autor

Se realiza una pestaña definida con el nombre **"MANUAL"**, en donde al iniciar la ejecución del programa permite comprobar la comunicación con la tarjeta Arduino MEGA 2560, el correcto envió de la señal PWM hacia el motor y el funcionamiento del encoder.

Figura 2.24 HMI de la opción "MANUAL".
Fuente: Autor

La implementación del controlador PID se empieza con la creación de una pestaña definida con el nombre **"PID"** en el Interfaz hombre-máquina (HIM), en donde se podrá ajustar el valor del Set y varias las ganancias del controlador.

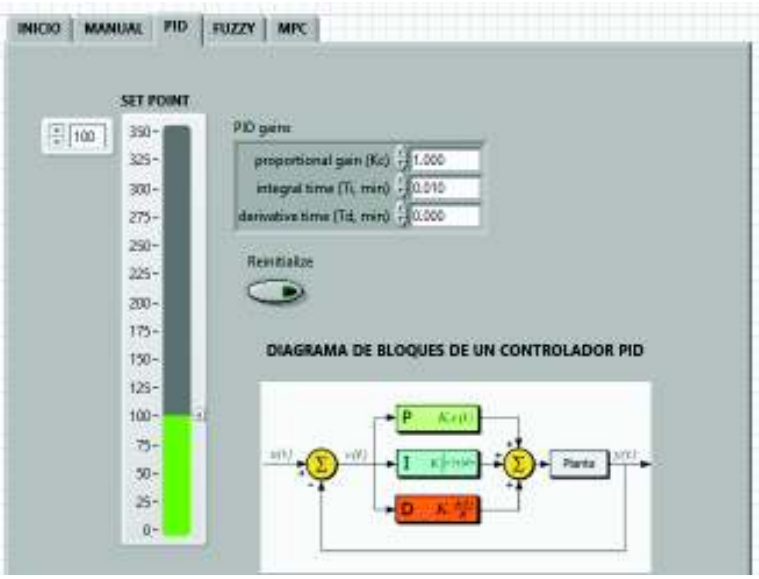

Figura 2.25 HMI del control PID
Fuente: Autor

Para la programación del controlador PID en el diagrama de bloques de LabVIEW, se trabaja con el *subVI "PID"* propio de LabVIEW, el cual permite configurar el rango necesario para la salida en este caso un PWM (0-100), modificar las ganancias del controlador, ingresar el valor deseado del Set y la lectura de la retroalimentación de la velocidad en rpm.

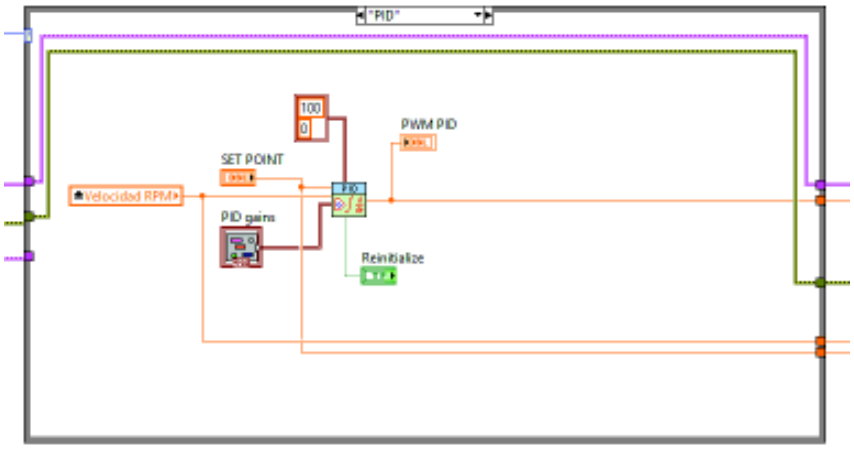

Figura 2.26 Diagrama de Bloques de la programación del control PID.
Fuente: Autor

2.2.2.3 Parámetros del controlador PID

Para la obtención de los parámetros del controlador PID, se trabaja mediante el toolbox *Simulink* de MATLAB. En este se realiza el diagrama de bloques del motor DC POLOLU

implementando un controlador PID como se indica en la Figura 2.27 Diagrama de bloques del motor DC POLOLU representado en Simulink.

Figura 2.27 Diagrama de bloques del motor DC POLOLU representado en Simulink.
Fuente: Autor

Para obtener los parámetros de control se ejecuta la opción de *Tuning* como se muestra en

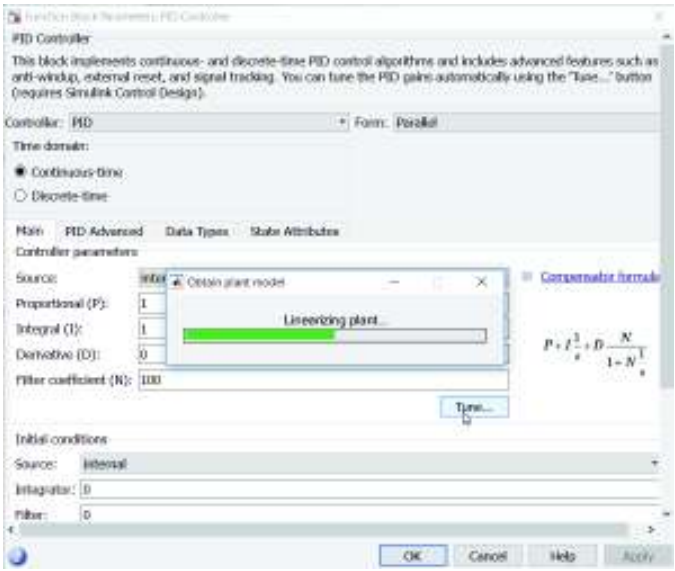

Figura 2.28 Linealización del sistema para obtener los parámetros del control PID mediante el uso de la opción Tuning.
Fuente: Autor

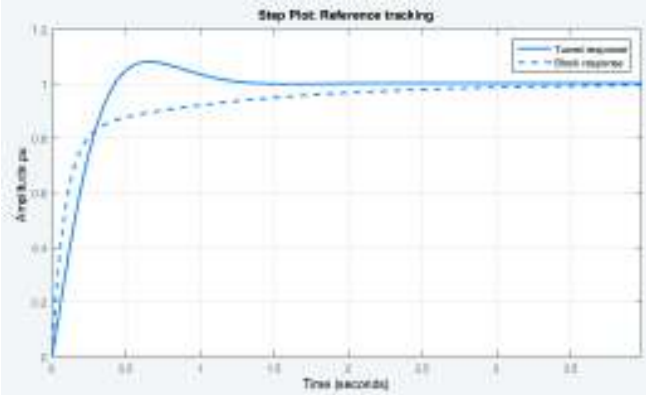

Figura 2.29 Respuesta de la planta luego de la obtención de los parámetros del control PID mediante la opción Tuning.
Fuente: Autor

En la Figura 2.30 Parámetros de control obtenidos mediante PID Tuning. se muestran los parámetros de control que arrojó el PID Tuning para la función de transferencia que modela la planta.

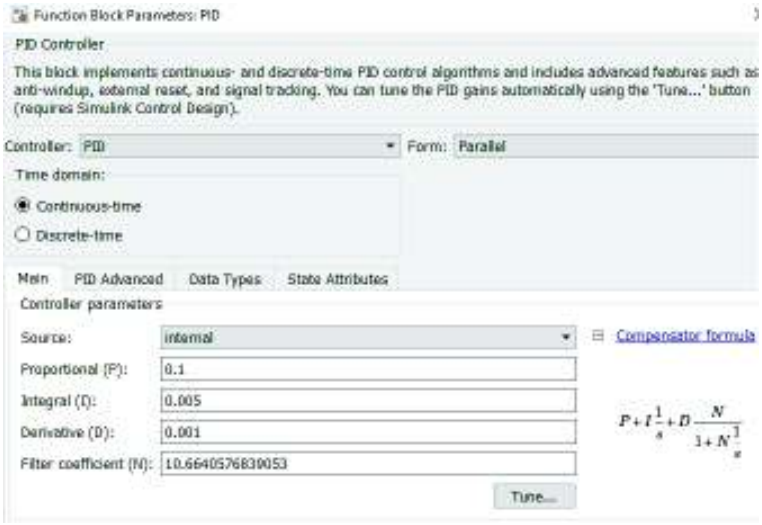

Figura 2.30 Parámetros de control obtenidos mediante PID Tuning.
Fuente: Autor

2.2.3. Diseño del controlador FUZZY LOGIC

Para el diseño del controlador FUZZY se utilizó el toolbox de MATLAB *FUZZY* el cual permite la creación de los conjuntos difusos y la desfuzificación de los mismos. Para la implementación se ocupa el proyecto creado en LabVIEW anteriormente y se trabajara en el mismo HMI del PID.

2.2.3.1 Diseño del controlador FUZZY

Se debe definir las entradas y salidas del sistema. Para este proyecto se va a utilizar 2 entradas y 1 salida, donde las entradas son la velocidad deseada (Set rpm) y el Error, la salida será el PWM que va al motor.

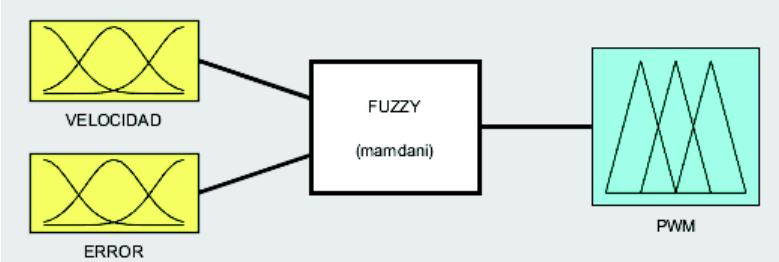

Figura 2.31 Definición de entradas y salida para el control FUZZY.
Fuente: Autor

Se debe definir los rangos de las entradas y salidas, así como asignar funciones de pertenencia y asignar las variables lingüísticas.

Como se muestra en la Figura 2.32 Definición de rangos y funciones de pertenencia de la Entrada 1: Velocidad. la entrada Velocidad está definida a lo largo del rango [0,350] rpm y consta de 7 funciones de pertenencia.

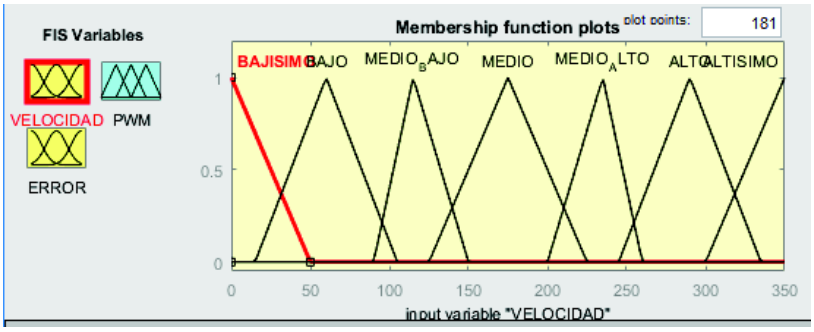

Figura 2.32 Definición de rangos y funciones de pertenencia de la Entrada 1: Velocidad.
Fuente: Autor

Como se muestra en la Figura 2.33 Definición de rangos y funciones de pertenencia de la Entrada 2: Error. la entrada Error está definida a lo largo del rango [-300,+300] y consta de 3 funciones de pertenencia. El ERROR es calculado restando los rpm de lectura del encoder menos los rpm del valor del SET.

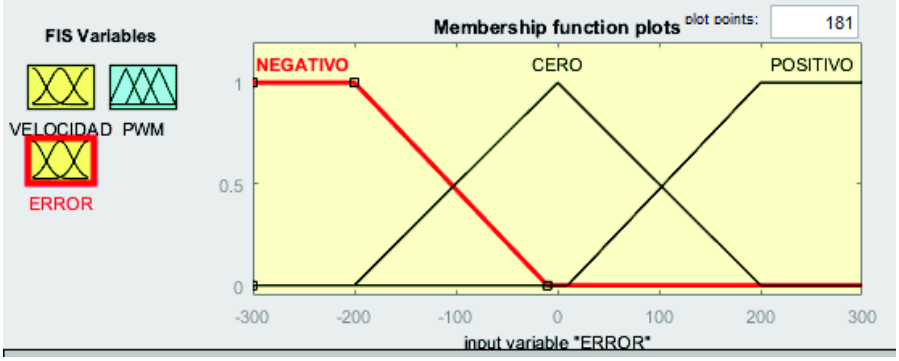

Figura 2.33 Definición de rangos y funciones de pertenencia de la Entrada 2: Error.
Fuente: Autor

Como se muestra en la Figura 2.34 Definición de rangos y funciones de pertenencia de la Salida: PWM. la salida PWM está definida a lo largo del rango [0,100] PWM y consta de 7 funciones de pertenencia.

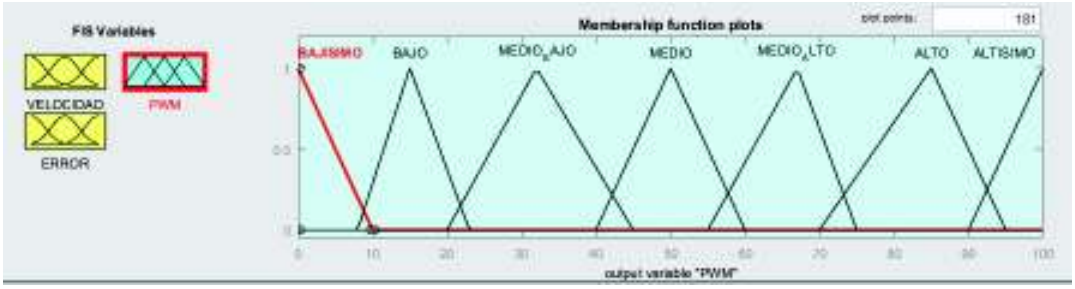

Figura 2.34 Definición de rangos y funciones de pertenencia de la Salida: PWM.
Fuente: Autor

Para el diseño del controlador se consideraron 21 reglas que fueron definidas como se muestra en la Tabla 2.3 *Matriz para determinar las reglas de pertenencia.*

Tabla 2.3 Matriz para determinar las reglas de pertenencia.

ERROR / RPM	BAJISIMO	BAJO	MEDIO BAJO	MEDIO	ALTO	MEDIO ALTO	ALTISIMO
NEGATIVO	BAJO	MEDIO BAJO	MEDIO	MEDIO ALTO	ALTO	ALTISIMO	ALTISIMO
CERO	BAJISIMO	BAJO	MEDIO BAJO	MEDIO	MEDIO ALTO	ALTO	ALTISIMO
POSITIVO	BAJISIMO	BAJISIMO	BAJISIMO	BAJISIMO	BAJO	BAJO	BAJO

Fuente: Autor

Posterior a realizar la matriz para determinar las reglas de pertenencia, se procede a agregar al *Rule Editor* como se indica en la Figura 2.35 Reglas agregadas al Rule Editor.

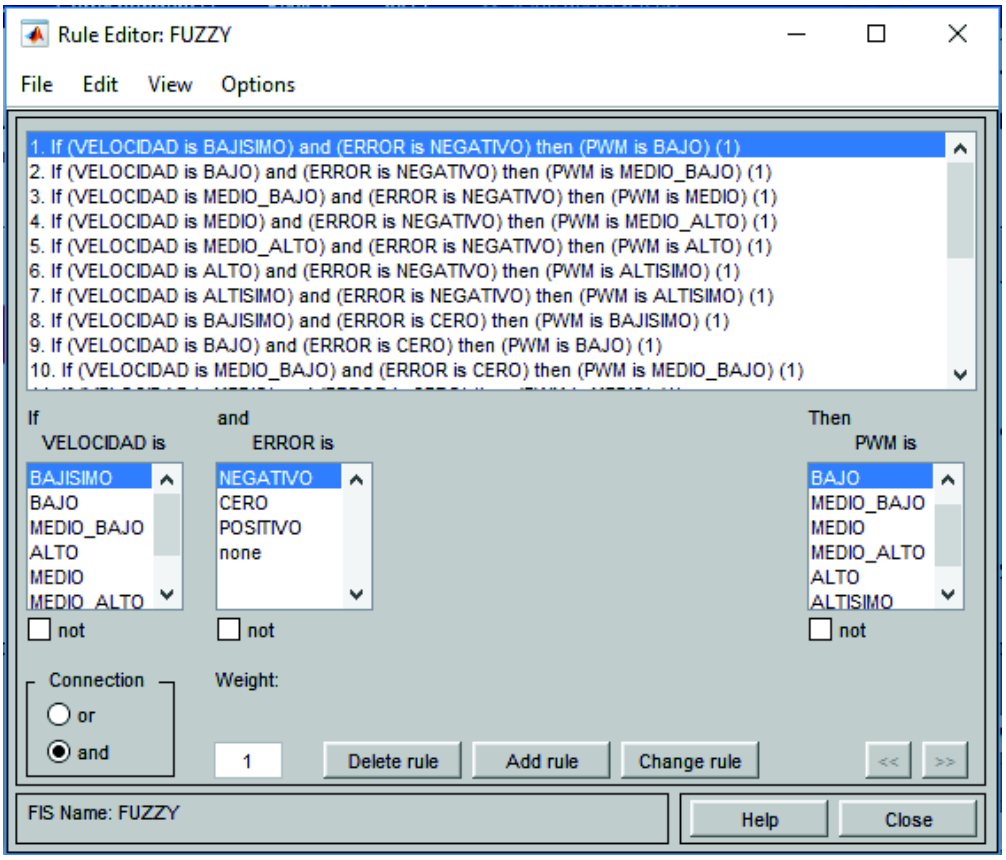

Figura 2.35 Reglas agregadas al Rule Editor.
Fuente: Autor

En la Figura 2.36 Surface del controlador. se puede observar el proceso de Desfuzificación mediante el método del centroide.

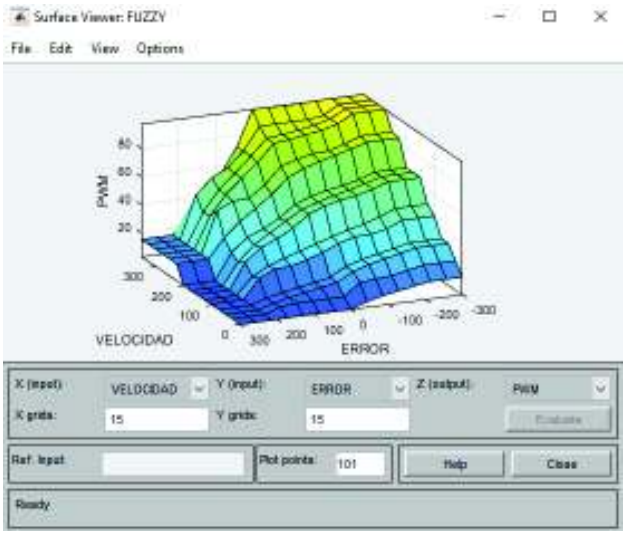

Figura 2.36 Surface del controlador.
Fuente: Autor

2.2.3.2 Implementación del controlador FUZZY

Para la implementación del controlador FUZZY se utiliza el proyecto de LabVIEW ya creado anteriormente. Se crea una pestaña definida con el nombre **"FUZZY"** en el HMI, en donde se podrá ajustar el valor del Set y cargar el controlador diseñado anteriormente.

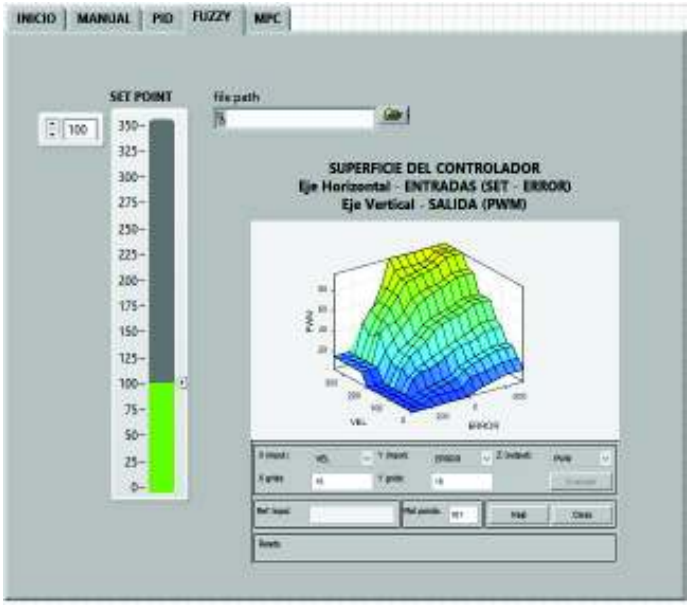

Figura 2.37 HMI del control FUZZY.
Fuente: Autor

Para la programación del controlador FUZZY en el diagrama de bloques de LabVIEW, se trabaja con el *subVI "FUZZY"* propio de LabVIEW, se debe seleccionar la configuración **MISO** (Multiple Input – Single Output) ya que vamos a trabajar con dos entradas y una salida. Se debe crear todos los controles que irán conectados al subVI como son: entradas, archivo de configuración y la salida del PWM.

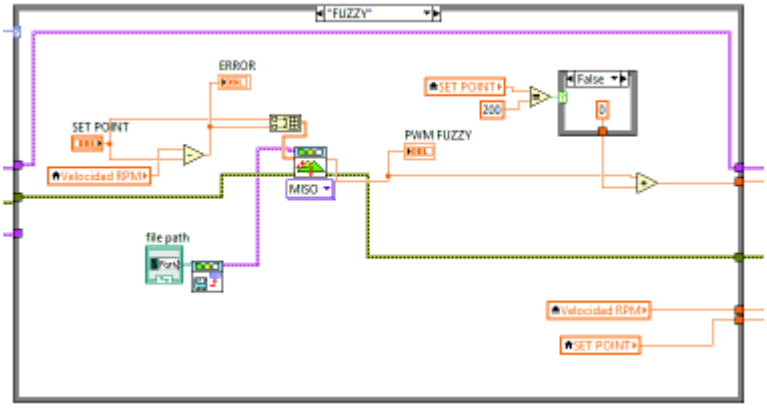

Figura 2.38 Diagrama de Bloques de la programación del control FUZZY.
Fuente: Autor

2.2.4. Diseño del controlador MPC (Model Predictive Control)

Para el diseño y simulación del controlador MPC, se utiliza el software LabVIEW.

2.2.4.1 Simulación del control MPC

La simulación se realiza en un VI de LabVIEW. En el diagrama de bloques se crea una estructura de simulación denominada Control & Simulation Loop, dentro de esta estructura se agrega los siguientes subVIs propios para el control MPC:

- **CD Create MPC Controller:** Tiene como objetivo la creación del modelo MPC controlado por un modelo de EE. Entre sus parámetros está:
    - Parámetros del Controlador:
        - *Horizonte de predicción:* Indica el número de muestras futuras que el controlador considera para predecir la salida del sistema.
        - *Horizonte de control:* Es el número de muestras futuras que el controlador considera para calcular la acción de control.
    - Modelo en Espacios de Estado: Indica la matriz de espacios de estado en tiempo discreto del modelo de la planta que se va a controlar.
    - Matriz de Costos: Representa el peso de las matrices que van a ser usadas en la función de costo.

- Matriz de Restricciones (DUAL): Representa las restricciones, se definen utilizando el método de optimización DUAL. Cada parámetro especifica un límite mínimo o máximo sobre: la acción de control inicial o final ***u***, la salida de la planta ***y***, el cambio en la acción de control ***du***. Se puede indicar solo un límite mínimo o máximo para un parámetro de los definidos anteriormente, LabVIEW asumirá como una constante.

- **CD Set MPC Controller:** Actualiza los parámetros específicos del modelo de control predictivo (MPC) controlados por el modelo de espacios de estado.
- **CD Implement MPC Controller:** Calcula la Acción de control **u(k)** para aplicar a la planta. Este VI utiliza la salida de la ventana de referencia, la ventana de perturbación y la ventana de referencia de la acción de control para calcular la acción de control a lo largo del horizonte de control en el momento **k**.
- **CD Step Forward MPC Window (Solo Simulación):** Calcula el segmento apropiado o ventana, de los perfiles del punto de Set.
- **CD Update MPC Window (Solo Implementación):** Calcula el segmento apropiado, o ventana, del punto de Set o el perfil de perturbación de la señal desde el momento **k** hasta el tiempo **k + horizonte de predicción.**

En la se puede observar el diagrama de bloques de la programación para la simulación del controlador MPC. La matriz de espacios de estado, restricciones y los demás parámetros deben ser ingresados previo a la ejecución de la simulación.

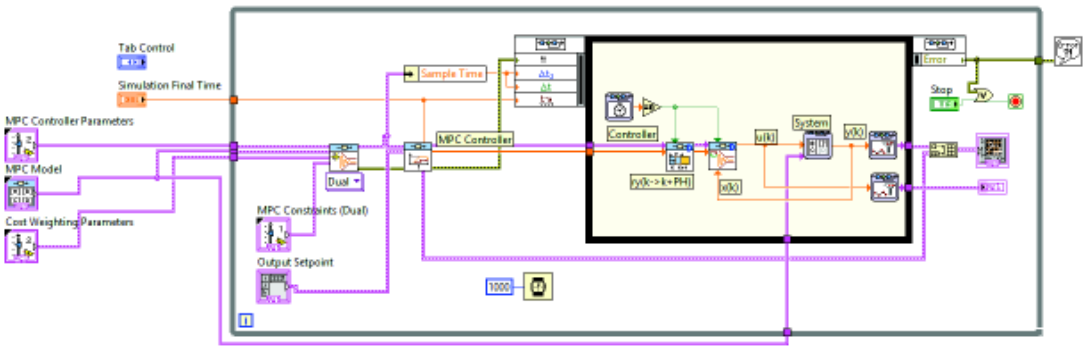

Figura 2.39 Programa en el diagrama de bloques en LabVIEW de la simulación del control MPC.
Fuente: Autor

2.2.4.2 Implementación del control MPC

Para el diseño del controlador MPC se utiliza el proyecto creado en LabVIEW anteriormente y se trabajara en el mismo HMI del PID y el FUZZY.

La programación se realiza en el diagrama de bloques utilizando los subVIs propios de LabVIEW para el controlador MPC. Se crea una pestaña definida con el nombre “MPC” en el HMI, en donde se podrá ajustar los diferentes parámetros y restricciones del controlador, así como el valor del Set y visualizar le modelo en espacios de estado de la planta.

En las siguientes figuras: Figura 2.40 Modelo en Espacio de Estados del control MPC., Figura 2.41 Parámetros de Control del controlador MPC. Figura 2.42 Parámetro de Restricciones del controlador MPC.y Figura 2.43 HMI del control MPC. se indica cada sección del controlador MPC y sus parámetros.

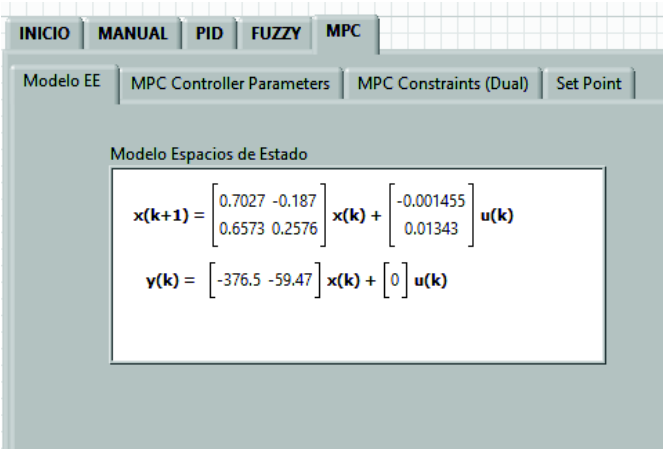

Figura 2.40 Modelo en Espacio de Estados del control MPC.
Fuente: Autor

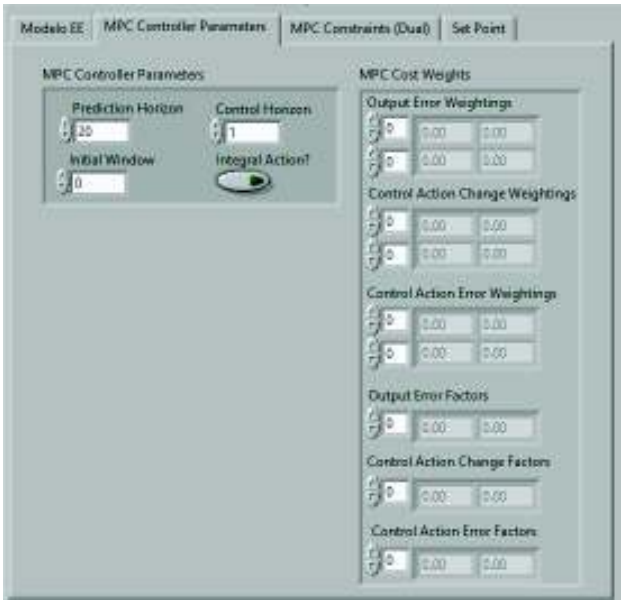

Figura 2.41 Parámetros de Control del controlador MPC.
Fuente: Autor

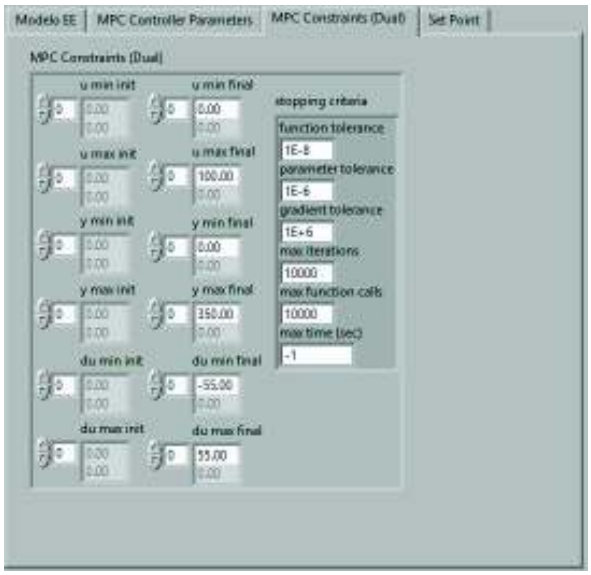

Figura 2.42 Parámetro de Restricciones del controlador MPC.
Fuente: Autor

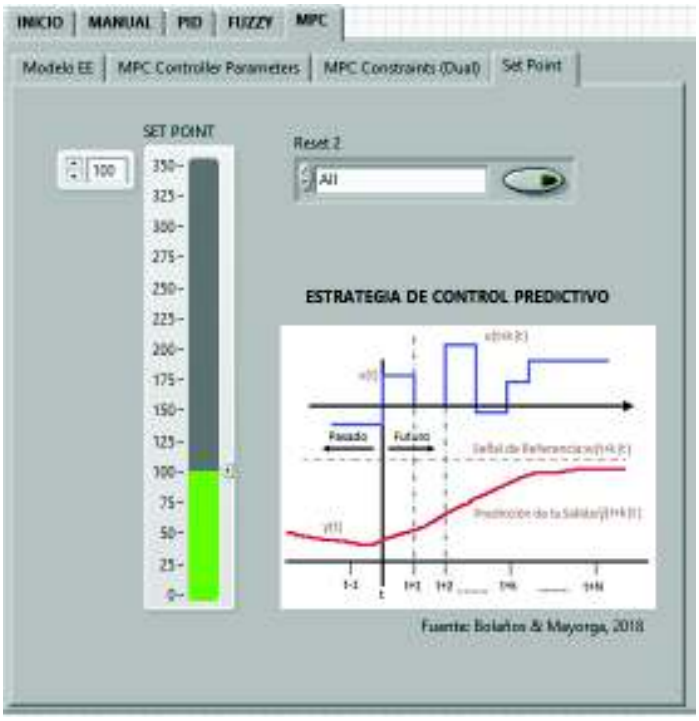

Figura 2.43 HMI del control MPC.
Fuente: Autor

3. RESULTADOS

Se presentan los resultados para el análisis y el estudio comparativo de los controladores PID, FUZZY y MPC, se empieza con la simulación de los controladores, para continuar con las respuestas de la implementación de los controladores.

En la implementación se realizan las pruebas de dos maneras, como se muestra en la Figura 3.1 Motor DC POLOLU a) Con eje libre b) Con fuerza constante en el eje. Se prueba con el motor DC POLOLU con el eje libre y con el eje con una fuerza constante (Peso).

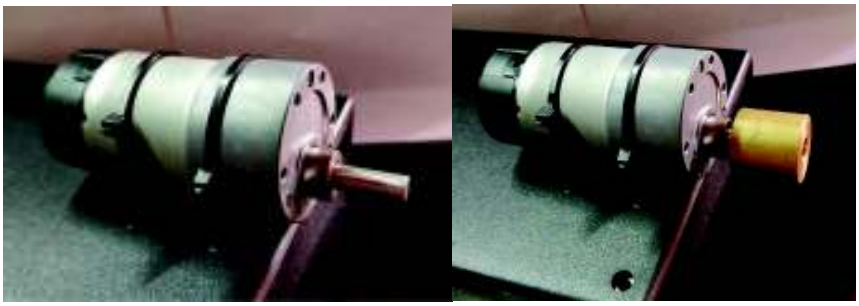

Figura 3.1 Motor DC POLOLU a) Con eje libre b) Con fuerza constante en el eje.
Fuente: Autor

3.1. Simulación de los controladores

3.1.1. Simulación Control PID

Se ejecuta la simulación del control PID, en el diagrama de bloques se ingresó la función de trasferencia del motor, el controlador con los parámetros encontrados mediante el auto tuning de MATLAB.

En la Figura 3.2 Gráfica de la respuesta de la simulación del control PID. se puede observar la respuesta a la simulación del control PID.

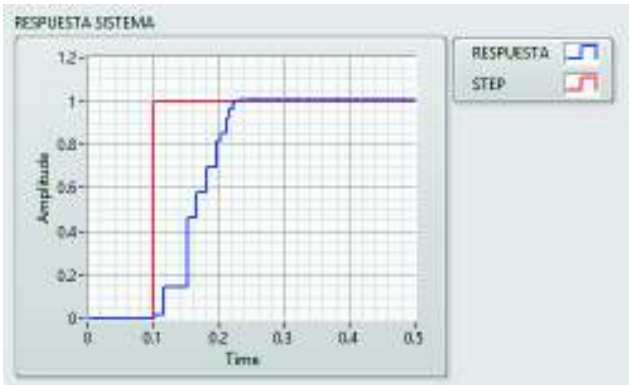

Figura 3.2 Gráfica de la respuesta de la simulación del control PID.
Fuente: Autor

En la Figura 3.2 Gráfica de la respuesta de la simulación del control PID. se observa el resultado de la simulación del control PID, se puede observar que la salida del sistema tiene una respuesta óptima ya que no presenta sobre impulso y llega a su estabilización.

3.1.2. Simulación control FUZZY

A continuación, se realiza la simulación del control FUZZY, con el diseño del controlador que se realizó en el capítulo anterior.

En la Figura 3.3 Comportamiento del sistema a los valores de las entradas: Set:175 y Error:-50 da como resultado una salida PWM: 54.2. y Figura 3.4 Comportamiento del sistema a los valores de las entradas: Set:250 y Error:-10 da como resultado una salida PWM: 70.3. se muestra el comportamiento del sistema a diferentes valores de ajuste de Set.

Figura 3.3 Comportamiento del sistema a los valores de las entradas: Set:175 y Error:-50 da como resultado una salida PWM: 54.2.
Fuente: Autor

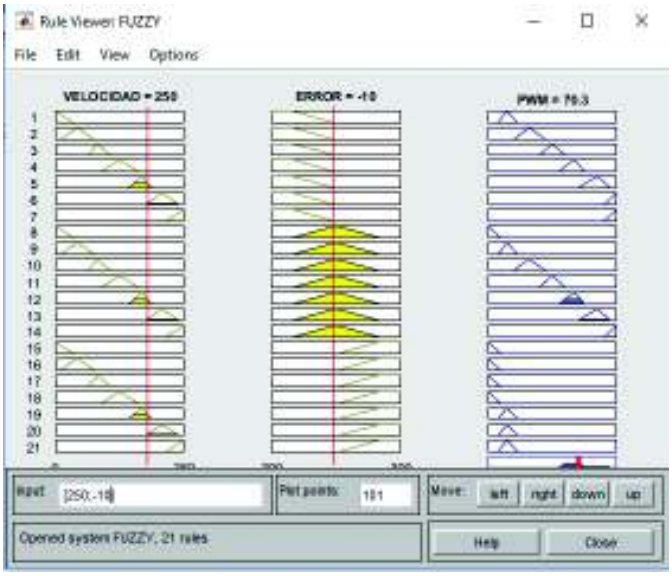

Figura 3.4 Comportamiento del sistema a los valores de las entradas: Set:250 y Error:-10 da como resultado una salida PWM: 70.3.
Fuente: Autor

En la Figura 3.3 Comportamiento del sistema a los valores de las entradas: Set:175 y Error:-50 da como resultado una salida PWM: 54.2. y Figura 3.4 Comportamiento del sistema a los valores de las entradas: Set:250 y Error:-10 da como resultado una salida PWM: 70.3. se puede observar que el proceso de desfuzificación está acorde a las características del diseño del controlador FUZZY. Se tiene una respuesta coherente de acuerdo a los valores de las entradas que se presentaron.

3.1.3. Simulación control MPC

A continuación, se realiza la simulación del control MPC, con el diseño del controlador que se realizó en el capítulo anterior.

En la Figura 3.5 Gráfica de la respuesta de la simulación del control MPC a diferentes valores de Set. se muestra la respuesta del sistema a diferentes valores de Set.

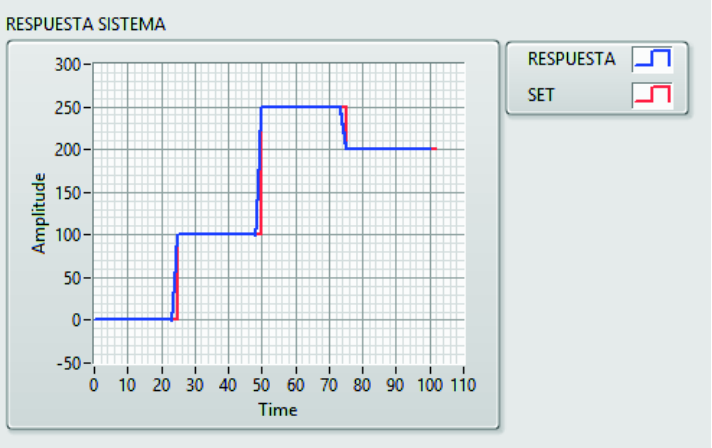

Figura 3.5 Gráfica de la respuesta de la simulación del control MPC a diferentes valores de Set.
Fuente: Autor

En la Figura 3.5 Gráfica de la respuesta de la simulación del control MPC a diferentes valores de Set. se observa el resultado de la simulación del control MPC, se puede observar que la salida del sistema sigue la trayectoria de acuerdo a los diferentes puntos de Set que se simularon, obteniendo una respuesta rápida y estable.

3.2. Implementación de los Controladores

A continuación, se prueba los diferentes controladores, con la implementación de estos en la computadora y la comunicación a través de la tarjeta Arduino y el driver para poder controlar el motor DC.

Para cada controlador se prueba dos puntos de ajuste y con el motor con peso y sin peso en el eje.

3.2.1. Implementación control PID

3.2.1.1 Implementación control PID en el motor con el eje libre

La Figura 3.6 Respuesta del sistema con control PID a un Set de 100 rpm. y Figura 3.7 Respuesta del sistema con control PID a un Set de 200 rpm. muestran la respuesta del controlador a los valores de Set de 100 y 200 rpm aplicados al motor con el eje libre.

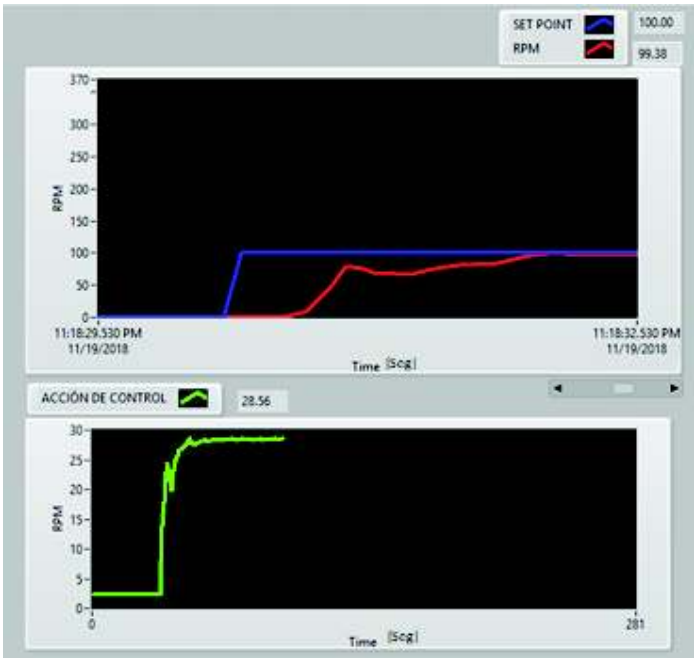

Figura 3.6 Respuesta del sistema con control PID a un Set de 100 rpm.
Fuente: Autor

Figura 3.7 Respuesta del sistema con control PID a un Set de 200 rpm.
Fuente: Autor

La Figura 3.6 Respuesta del sistema con control PID a un Set de 100 rpm. y Figura 3.7 Respuesta del sistema con control PID a un Set de 200 rpm. muestran la implementación del control PID,

se obtiene resultados óptimos ya que la salida del sistema llega al punto de Set sin presentar sobre impulsos y se mantiene estable después. La acción de control indica que la respuesta del sistema es óptima.

3.2.1.2 Implementación control PID en el motor con fuerza sobre el eje.

La Figura 3.8 Respuesta del sistema con control PID a un Set de 100 rpm con peso en el eje. y Figura 3.9 Respuesta del sistema con control PID a un Set de 200 rpm con peso en el eje. muestran la respuesta del controlador a los valores de Set de 100 y 200 rpm aplicados al motor con fuerza sobre el eje (Peso).

Figura 3.8 Respuesta del sistema con control PID a un Set de 100 rpm con peso en el eje.
Fuente: Autor

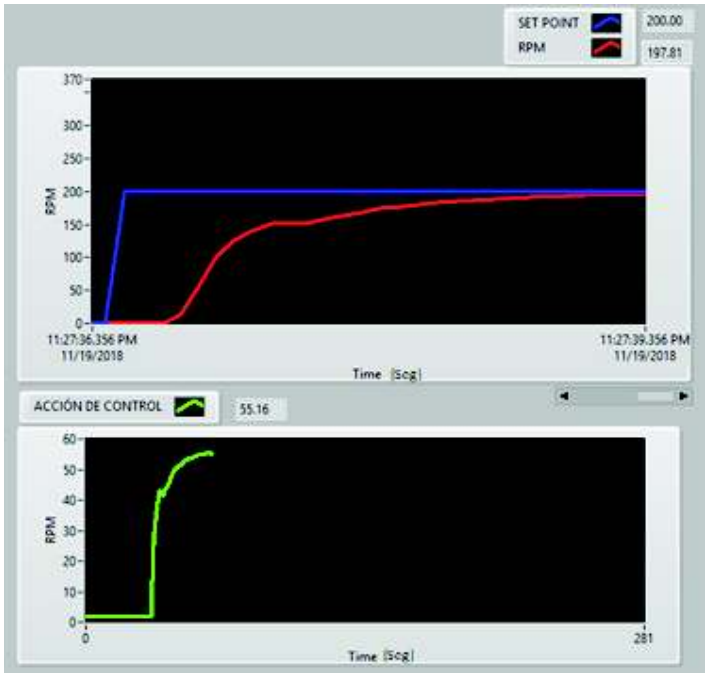

Figura 3.9 Respuesta del sistema con control PID a un Set de 200 rpm con peso en el eje.
Fuente: Autor

La Figura 3.8. y Figura 3.9. muestran la implementación del control PID con peso en el motor, la respuesta del sistema mostrada indica que a pesar de tener un peso la acción de control sigue su curso y logra estabilizar sin presentar sobre impulsos. La salida del sistema sigue el valor deseado.

3.2.2. Implementación control FUZZY

3.2.2.1 Implementación control FUZZY en el motor con el eje libre

La Figura 3.10 Figura 3.8 Respuesta del sistema con control FUZZY a un Set de 100 rpm. y Figura 3.11 Respuesta del sistema con control FUZZY a un Set de 200 rpm. muestran la respuesta del controlador a los valores de Set de 100 y 200 rpm aplicados al motor con el eje libre.

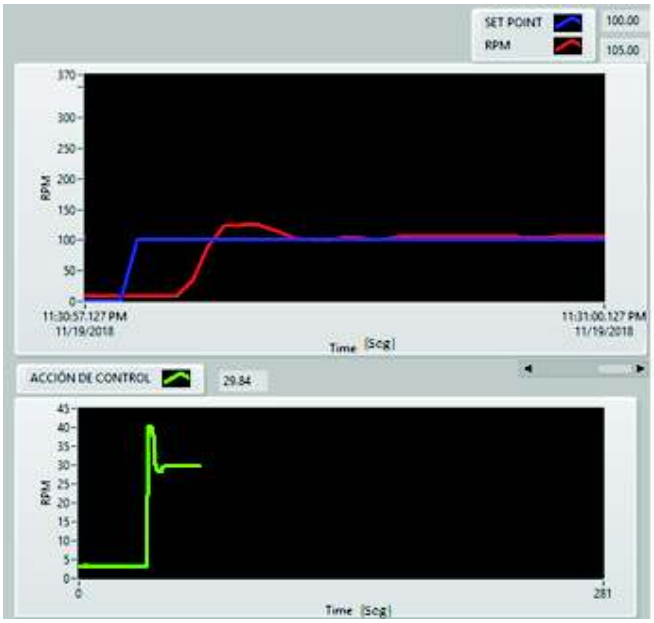

Figura 3.10 Figura 3.8 Respuesta del sistema con control FUZZY a un Set de 100 rpm.
Fuente: Autor

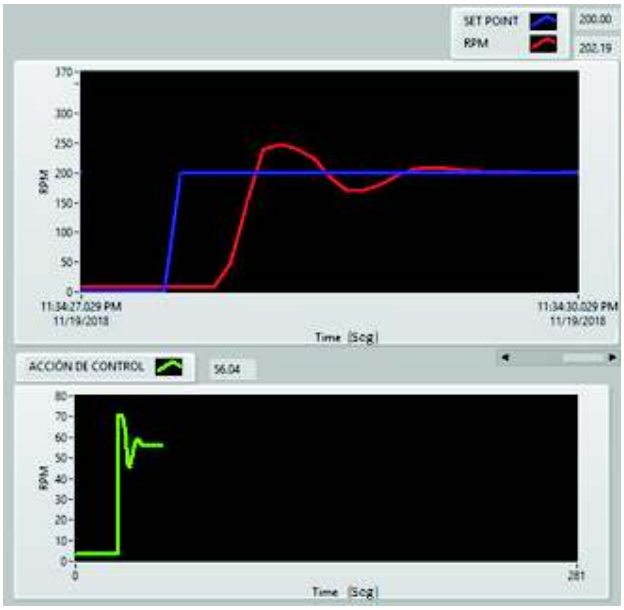

Figura 3.11 Respuesta del sistema con control FUZZY a un Set de 200 rpm.
Fuente: Autor

La Figura 3.10 Figura 3.8 Respuesta del sistema con control FUZZY a un Set de 100 rpm. y Figura 3.11 Respuesta del sistema con control FUZZY a un Set de 200 rpm. muestran la implementación del control FUZZY, en ambas respuestas se puede observar que la señal

del sistema presenta un ligero sobre impulso por lo que su respuesta se clasificaría como subamortiguado. La acción de control indica de una mejor manera la respuesta del sistema, presentando oscilación hasta llegar a su punto de estabilización.

3.2.2.2 Implementación control FUZZY en el motor con fuerza sobre el eje.

La Figura 3.12 Respuesta del sistema con control FUZZY a un Set de 100 rpm con peso en el eje. y Figura 3.13 Respuesta del sistema con control FUZZY a un Set de 200 rpm con peso en el eje. muestran la respuesta del controlador a los valores de Set de 100 y 200 rpm aplicados al motor con fuerza sobre el eje (Peso).

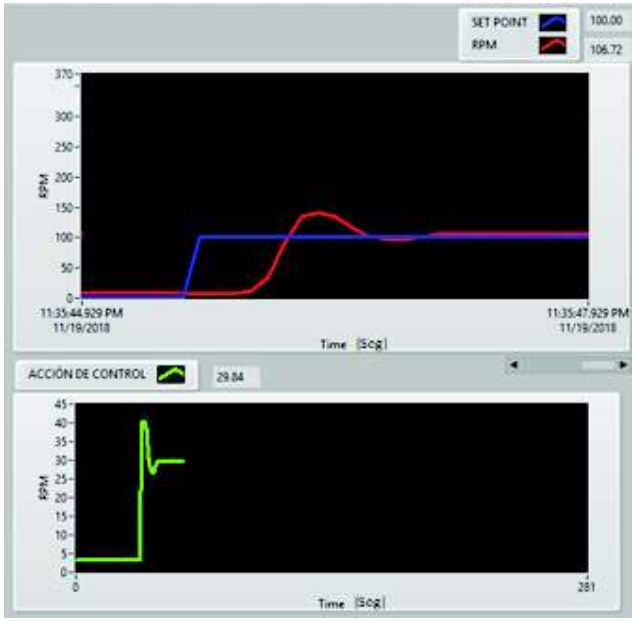

Figura 3.12 Respuesta del sistema con control FUZZY a un Set de 100 rpm con peso en el eje.
Fuente: Autor

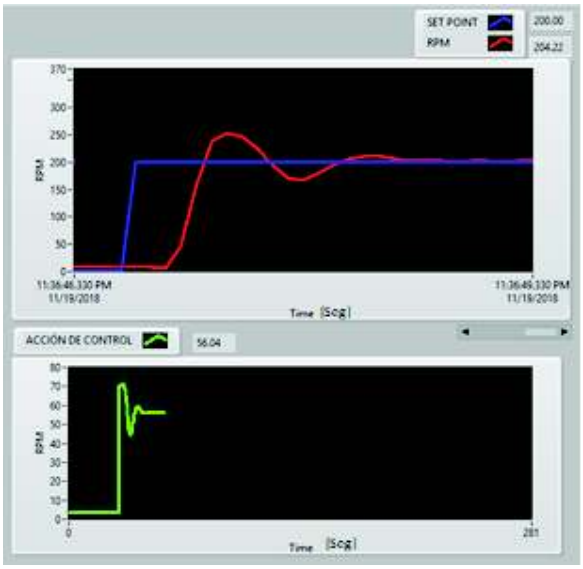

Figura 3.13 Respuesta del sistema con control FUZZY a un Set de 200 rpm con peso en el eje.
Fuente: Autor

La Figura 3.12 Respuesta del sistema con control FUZZY a un Set de 100 rpm con peso en el eje. y Figura 3.13 Respuesta del sistema con control FUZZY a un Set de 200 rpm con peso en el eje. muestran la implementación del control FUZZY con peso en el eje del motor, en este caso la respuesta se mantendría similar a la del sistema anterior presentando una respuesta subamortiguada, en donde la acción de control presenta varias oscilaciones antes de llegar a su estabilización.

3.2.3. Implementación control MPC

3.2.3.1 Implementación control MPC en el motor con el eje libre

La Figura 3.14 Respuesta del sistema con control MPC a un Set de 100 rpm. y Figura 3.15 Respuesta del sistema con control MPC a un Set de 200 rpm. muestran la respuesta del controlador a los valores de Set de 100 y 200 rpm aplicados al motor con el eje libre.

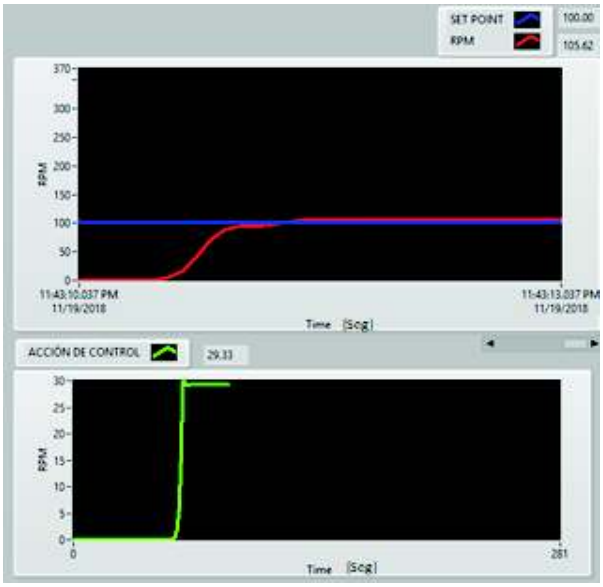

Figura 3.14 Respuesta del sistema con control MPC a un Set de 100 rpm.
Fuente: Autor

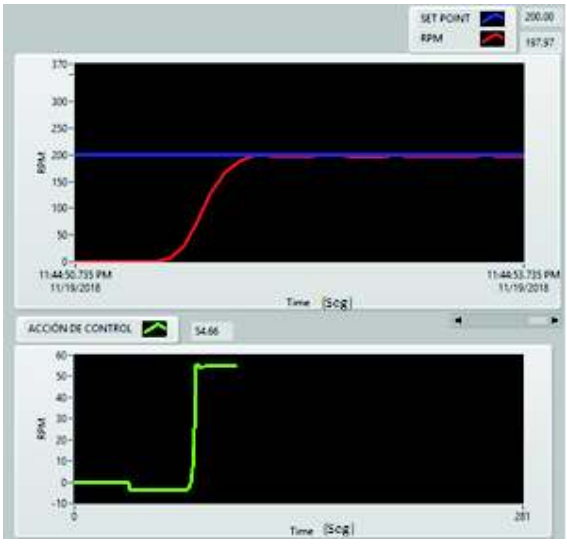

Figura 3.15 Respuesta del sistema con control MPC a un Set de 200 rpm.
Fuente: Autor

La Figura 3.14 Respuesta del sistema con control MPC a un Set de 100 rpm. y Figura 3.15 Respuesta del sistema con control MPC a un Set de 200 rpm. muestran la implementación del controlador MPC. Los cuales presentan una respuesta óptima ya que no presentan sobre impulsos y el sistema sigue el punto de set al cual se le ajusto. La acción de control presenta una respuesta rápida.

3.2.3.2 Implementación control MPC en el motor con fuerza sobre el eje.

La Figura 3.16 Figura Respuesta del sistema con control MPC a un Set de 100 rpm con peso en el eje. y Figura 3.17 Respuesta del sistema con control MPC a un Set de 200 rpm con peso en el eje. muestran la respuesta del controlador a los valores de Set de 100 y 200 rpm aplicados al motor con fuerza sobre el eje (Peso).

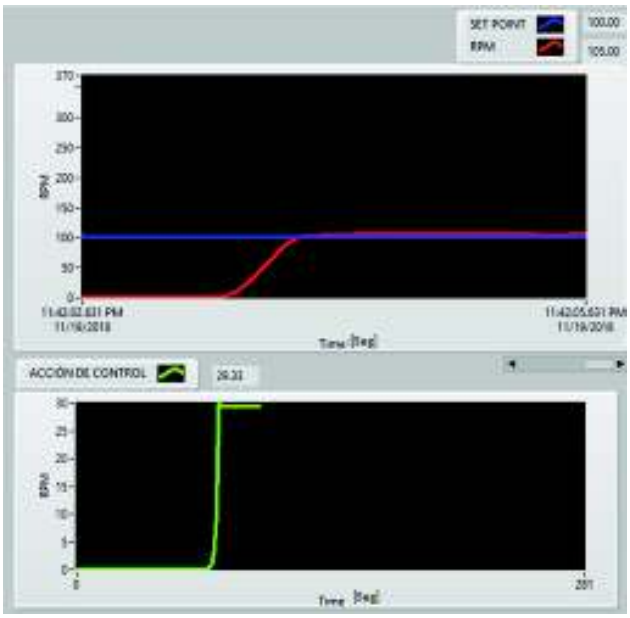

Figura 3.16 Figura Respuesta del sistema con control MPC a un Set de 100 rpm con peso en el eje.
Fuente: Autor

Figura 3.17 Respuesta del sistema con control MPC a un Set de 200 rpm con peso en el eje.
Fuente: Autor

La Figura 3.16 Figura Respuesta del sistema con control MPC a un Set de 100 rpm con peso en el eje. y Figura 3.17 Respuesta del sistema con control MPC a un Set de 200 rpm con peso en el eje. muestran la implementación del controlador MPC con peso en el eje del motor. El sistema es similar al anterior analizado, pero en este caso en particular para el Set de 200 rpm la señal del sistema presenta una pequeña variación hasta llegar a su valor de set, sin embargo, no presenta sombre impulso. La acción de control presenta una respuesta rápida.

4. DISCUSIÓN

En este capítulo se realizará el análisis de los resultados de la implementación de los controladores PID, FUZZY y MPC.

Cuando se diseña un sistema de control para un proceso físico, se debe seleccionar uno o más parámetros para analizar el mejor rendimiento. Utilizando una medida conocida como índice de rendimiento. Los criterios comunes para el índice de rendimiento se dan utilizando 4 expresiones, que van en función de la señal de error, las cuales son: (Voice, 2018)

- Integral Square Error (ISE)
- Integral Absolute Error (IAE)
- Integral Time Squared Error (ITSE)
- Integral Time Absolute Error (ITAE)

La señal de error ($e(t)$) en un Sistema de Control está dada por:

$$e(t) = r(t) - c(t)$$

donde:

$r(t)$ = entrada o señal de referencia

$c(t)$ = señal de salida

Todas las medidas requieren que se realice un experimento fijo en el sistema (es decir, un punto de ajuste fijo o un cambio de perturbación) y las integrales se evalúan durante un período de tiempo fijo (en teoría hasta el infinito, pero generalmente hasta un tiempo lo suficientemente largo para que las respuestas se resuelvan).

ISE integra el cuadrado del error en el tiempo. ISE penalizará los errores grandes más que los pequeños (ya que el cuadrado de un error grande será mucho más grande). Los sistemas de control especificados para minimizar el ISE tenderán a eliminar grandes errores rápidamente, pero tolerarán los pequeños errores que persisten durante un largo período de tiempo. Generalmente, esto conduce a respuestas rápidas, pero con una oscilación considerable y de baja amplitud.

IAE integra el error absoluto en el tiempo. No agrega peso a ninguno de los errores en una respuesta de sistemas. Tiende a producir una respuesta más lenta que los sistemas ISE, pero generalmente con una oscilación menos sostenida.

ITAE integra el error absoluto multiplicado por el tiempo a lo largo del tiempo. Pondera los errores que existen después de mucho tiempo mucho más que aquellos al comienzo de la respuesta. El ajuste de ITAE produce sistemas que se liquidan mucho más rápidamente que los otros dos métodos de ajuste. La desventaja de esto es que la sintonización ITAE también produce sistemas con respuesta inicial lenta (necesaria para evitar la oscilación sostenida). (Vissim, 2016)

Para el análisis del índice de rendimiento en específico se utilizará la expresión ISE.

4.1. Comparación de las especificaciones del dominio de tiempo para el punto de set.

En la Tabla 4.1 Resultados *con Set: 100 y eje de motor libre.* se muestra la comparación de los resultados de las respuestas de los controladores PID, FUZZY y MPC con un punto de SET de 100 rpm con el eje del motor libre.

Tabla 4.1 Resultados con Set: 100 y eje de motor libre.

	PID	FUZZY	MPC
Tiempo de estabilización (Segundos)	2.0	2.7	2.3
Sobre impulso (%)	0	0.05	0.02
Error Estado Estacionario (%)	0.02	0.05	0.02

Fuente: Autor

En la Tabla 4.1 *Resultados con Set: 100 y eje de motor libre.* se observa que el PID tiene un tiempo de estabilización más rápido comparado con el FUZZY y el MPC, el sobre impulso tiene un porcentaje muy bajo en los tres controladores al igual que el error en estado estacionario.

En la Tabla 4.2 *Resultados con Set: 200 y eje de motor libre.* se muestra la comparación de los resultados de las respuestas de los controladores PID, FUZZY y MPC con un punto de SET de 200 rpm con el eje del motor libre.

Tabla 4.2 Resultados con Set: 200 y eje de motor libre.

	PID	FUZZY	MPC
Tiempo de estabilización (Segundos)	2.6	2.5	2.4
Sobre impulso (%)	0	0.05	0
Error Estado Estacionario (%)	0.03	0.02	0.02

Fuente: Autor

En la Tabla 4.2 *Resultados con Set: 200 y eje de motor libre.* se observa que el MPC tiene un tiempo de estabilización más rápido comparado con el PID y el FUZZY, el sobre impulso es cero en el MPC y el PID y se tiene un porcentaje muy bajo en el FUZZY, el error en estado estacionario tiene un valor bajo en los tres controladores.

En la Tabla 4.3 *Resultados con Set: 100 con peso en el eje del motor.* se muestra la comparación de los resultados de las respuestas de los controladores PID, FUZZY y MPC con un punto de SET de 100 rpm con peso en el eje del motor.

Tabla 4.3 Resultados con Set: 100 con peso en el eje del motor.

	PID	FUZZY	MPC
Tiempo de estabilización (Segundos)	2.7	2.4	2.3
Sobre impulso (%)	0	0.06	0.04
Error Estado Estacionario (%)	0.02	0.05	0.03

Fuente: Autor

En la Tabla 4.3 *Resultados con Set: 100 con peso en el eje del motor.* se observa que al implementar los controladores con peso en el eje del motor tiene variaciones en el tiempo de estabilización, el sobre impulso y el error en estado estacionario mantienen sus porcentajes bajos.

En la Tabla 4.4 *Resultados con Set: 200 con peso en el eje del motor.* se muestra la comparación de los resultados de las respuestas de los controladores PID, FUZZY y MPC con un punto de SET de 200 rpm con peso en el eje del motor.

Tabla 4.4 Resultados con Set: 200 con peso en el eje del motor.

	PID	FUZZY	MPC
Tiempo de estabilización (Segundos)	2.6	2.7	2.5
Sobre impulso (%)	0	0.06	0
Error Estado Estacionario (%)	0.04	0.04	0.02

Fuente: Autor

En la Tabla 4.4 *Resultados con Set: 200 con peso en el eje del motor.* se observa que al implementar los controladores con peso en el eje del motor se tiene un ligero incremento en el tiempo de estabilización, se mantiene en cero el sobre impulso en el PID y el MPC con un porcentaje bajo en el FUZZY y una variación muy pequeña en el porcentaje del error en estado estacionario.

4.2. Índices de desempeño de los controladores implementados.

En la Tabla 4.5 *Comparación del índice de desempeño para un Set:100 con el eje del motor libre.* se muestra el índice de desempeño de la expresión ISE para los controladores PID, FUZZY y MPC con un punto de SET de 100 rpm con el eje del motor libre.

Tabla 4.5 Comparación del índice de desempeño para un Set:100 con el eje del motor libre.

	PID	FUZZY	MPC
ISE	0.2245	0.3212	0.3576

Fuente: Autor

En la Tabla 4.6 *Comparación del índice de desempeño para un Set:200 con el eje del motor libre.* se muestra el índice de desempeño de la expresión ISE para los controladores PID, FUZZY y MPC con un punto de SET de 200 rpm con el eje del motor libre.

Tabla 4.6 Comparación del índice de desempeño para un Set:200 con el eje del motor libre.

	PID	FUZZY	MPC
ISE	0.272	0.1038	0.0408

Fuente: Autor

La Tabla 4.5 *Comparación del índice de desempeño para un Set:100 con el eje del motor libre.* y la Tabla 4.6 *Comparación del índice de desempeño para un Set:200 con el eje del motor libre.* muestran el índice de desempeño (ISE) para los controladores PID, FUZZY y MPC con sets de 100 y 200 rpm, se puede observar que el control PID mantiene su valor con una

diferencia muy baja, no así el caso del FUZZY y el MPC que disminuyen sus valores significativamente para una velocidad mayor.

En la Tabla 4.7 *Comparación del índice de desempeño para un Set:100 con peso en el motor.* se muestra el índice de desempeño de la expresión ISE para los controladores PID, FUZZY y MPC con un punto de SET de 100 rpm con peso en el eje del motor.

Tabla 4.7 Comparación del índice de desempeño para un Set:100 con peso en el motor.

	PID	FUZZY	MPC
ISE	0.2343	0.3871	0.3104

Fuente: Autor

En la Tabla 4.8 *Comparación del índice de desempeño para un Set:200 con peso en el motor.* se muestra el índice de desempeño de la expresión ISE para los controladores PID, FUZZY y MPC con un punto de SET de 200 rpm con peso en el eje del motor.

Tabla 4.8 Comparación del índice de desempeño para un Set:200 con peso en el motor.

	PID	FUZZY	MPC
ISE	0.2713	0.1188	0.0463

Fuente: Autor

La Tabla 4.7 *Comparación del índice de desempeño para un Set:100 con peso en el motor.* y la Tabla 4.8 *Comparación del índice de desempeño para un Set:200 con peso en el motor.* muestran el índice de desempeño (ISE) para los controladores PID, FUZZY y MPC con sets de 100 y 200 rpm con peso en el eje del motor, al igual que el análisis en las tablas con el eje del motor sin peso, se observa que el control PID mantiene su valor con una diferencia muy baja, a diferencia del FUZZY y el MPC que disminuyen sus valores significativamente para una velocidad mayor.

5. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- Se logró cumplir con el objetivo fundamental del proyecto al diseñar e implementar una plataforma para el estudio comparativo de un controlador predictivo (MPC) con controladores PID y FUZZY, aplicado al control de velocidad de un motor DC.
- Se analizó e implemento los conceptos y elementos básicos del control predictivo, se realizó el modelo un motor DC POLOLU en variables de estado y función de transferencia, estableciendo los límites y restricciones de la planta.
- Se diseñó un algoritmo de control predictivo, un algoritmo de control PID y un algoritmo de control FUZZY, los cuales se implementaron en un PC mediante el software LabVIEW, para lo cual, se empleó la tarjeta electrónica Arduino MEGA cómo tarjeta de adquisición de datos. Se desarrolló una interfaz HMI para toda la plataforma que permite manejar los controladores desarrollados.
- Después de realizar el modelamiento del motor DC POLOLU, se compara los modelos obtenidos tanto de la función de transferencia como el modelo en espacio de estados, con la salida del sistema. Estos deben seleccionarse de acuerdo al porcentaje más alto de similitud independiente para cada caso. Los modelos obtenidos permiten tener el comportamiento más cercano al real, y de este modo poder realizar las parametrizaciones para los controladores, los mismos que permiten obtener la mejor respuesta para cada uno.
- Después de realizar las simulaciones se puede observar la salida aproximada para cada sistema. Al tener una buena respuesta para cada uno se puede proceder a realizar la implementación, ya que la simulación permite tener un panorama casi real a lo que podría ser la implementación, y con las respuestas obtenidas en la simulación se puede proceder a la implementación en donde solo habrá que realizar pequeños ajustes a cada parámetro de los controladores.
- Para encontrar los parámetros de cada controlador, se trabaja de manera independiente y diferente para cada uno. En el caso del PID se obtiene a través de MATLAB, los cuales también se pueden comprobar de manera manual a través del primer método de Ziegler-Nichols. Para el control FUZZY se debe ir probando con diferentes números de funciones de pertenencia, al igual que ir variando los rangos de acuerdo a la respuesta que se tenga, esto con el fin de llegar a la respuesta de mejor desempeño. Para el control MPC, al tener el motor una respuesta rápida ante cualquier señal, se debe empezar por ajustar en valores bajos el horizonte de

predicción, hasta llegar a una respuesta estable, el horizonte de control al tener el motor una respuesta rápida se debe mantener en uno (1) ya que valores superiores producirán oscilación en la respuesta del sistema.

- Para las comparaciones de los controladores con el eje del motor libre en sets de 100 y 200 rpm, se puede observar que el que presenta un mayor sobre impulso es el control FUZZY, siendo casi o escaso el sobre impulso en los otros controladores. El tiempo de estabilización más bajo presenta el control PID seguido del MPC por una diferencia muy baja, en el análisis del error en el estado estacionario se presentan valores muy similares para los tres controladores, por lo que se podría decir que el desempeño de éstos se nota en el sobre impulso que puedan presentar. El tiempo de estabilización de los controladores solo nos indica cual es el que tiene una mejor reacción en caso de algún cambio en el sistema.
- Para las comparaciones de los controladores con peso en el eje del motor en sets de 100 y 200 rpm, se observa que se sigue manteniendo el sobre impulso en el control FUZZY siendo escaso o casi nulo en los otros controladores, se observa también que el tiempo de estabilización aumenta un poco para el control PID, así como también se ve diferencias en el error en estado estacionario para los controles FUZZY y PID, en este caso el error en estado estacionario se mantiene en el MPC, ya que éste trabaja en base al comportamiento futuro que puede tener el sistema y así mantener el error en estado estacionario más estable.
- Para el análisis del índice de desempeño mediante la expresión ISE, se puede notar que éste varía de acuerdo a cada punto de ajuste y las condiciones que tiene el motor, el que presenta un mejor índice de desempeño en general de acuerdo al ISE es el control MPC, seguido y con valores muy cercanos el PID y el FUZZY.
- En este proyecto se puede definir que los tres controladores se encuentran dentro de un rango de desempeño óptimo, ya que los datos y el estudio realizado en este caso en específico demuestran que para aplicaciones donde se necesite el uso de motores de corriente continua sería de decisión del operario o diseñador el tipo de control que se va a utilizar, esto dependería específicamente de la aplicación en la cual se va a implementar y observando todas las variables del entorno que podrían afectar el desempeño de cada controlador. El sobre impulso mostrado en el control Fuzzy no sobrepasa el 5%, por lo que, su desempeño es bueno y puede aplicarse al igual que los otros controladores tomando en cuenta las variables del entorno y los equipos que se utilizaran para la implementación, así como el motor.
- La realización de esta plataforma permitió que se analice detalladamente el estudio de los controladores PID, FUZZY y MPC para una planta en específico, en donde

se puede obtener mejores destrezas para encontrar los diferentes parámetros, así como ver las diferentes respuestas frente a perturbaciones o valores distintos de los parámetros encontrados para la estabilización del sistema.

5.2. Recomendaciones

- Tomar en cuenta que los controladores son cargados en la PC, por lo que para la adquisición de datos y la ejecución de los controladores se recomienda tener cargado solo el software LabVIEW, para poder tener una mejor comunicación en la PC y la tarjeta Arduino.
- Se recomienda siempre depurar los valores iniciales obtenidos en la adquisición de datos, ya que por lo general los primeros datos obtenidos muestran datos basura y pueden alterar el modelo de la planta. Así también es recomendable eliminar todos los picos o datos erróneos que se pueden producir en el tiempo de muestreo.
- Se recomienda antes de la ejecución de los controladores en el HMI, probar el motor de forma manual para poder validar que la comunicación con la tarjeta este correcta, que el encoder lea adecuadamente y que no exista retardos en la lectura.
- Para un mejor análisis del desempeño de los controladores es recomendable el análisis en las mismas condiciones para cada controlador, es decir mismo punto de Set, mismo peso y así poder determinar de una manera equitativa los resultados.

REFERENCIA BIBLIOGRÁFICAS

Alvarado, M. S. (2012). Modelo matemático de un motor de corriente continua separadamente excitado: Control de velocidad por corriente de armadura. *IEEE*.

Alvarado, M. S. (2012). Modelo matemático de un motor de corriente continua separadamente excitado: Control de velocidad por corriente de armadura.

Arduino Store. (2018). Obtenido de https://store.arduino.cc/usa/arduino-mega-2560-rev3

Ati Villamarín, S. (2011). *Análisis, Diseño e Implementación de Controladores Predictivos.* Sangolquí - Ecuador: ESPE.

Bolaños Paredes, D. F., & Mayorga Miranda, L. E. (2015). *Diseño e implementación de un algoritmo de control predictivo multivariable de temperatura y nivel para la estación de control de procesos FESTO en el laboratorio de mecatrónica.* Latacunga: ESPE.

Bordons Alba, C. (2000). *Control Predictivo: metodología, tecnología y nuevas perspectivas.* Almería - España.

Cartagena. (2018). *Cartagena99*. Obtenido de http://www.cartagena99.com/recursos/alumnos/apuntes/Cap_3_SD.pdf

Chacón, E., Szigeti, F., & Camacho, O. (1996). Integral automation of industrial complexes based on hybrid systems. *ISA Transactions*, Volumen 35, Issue 4, Pages 305-319.

Córdova Mendoza, P. (2013). Instrumentación y control de procesos. *SlideShare*.

Crespo, M., & Pendino, C. (2015). En *Introducción al modelado de sistemas físicos.* Rosario - Argentina.

Definición. (2018). Obtenido de https://definicion.de/sensor/

Electronics, F. (2018). Obtenido de www.fut-electronics.com

Feroldi, D. (2012). Control Predictivo Basado en Modelo con Restricciones. Rosario, Argentina.

Geekbot electronics. (2018). Obtenido de http://www.geekbotelectronics.com/motores-de-dc/

González Morcillo, C. (2009). Lógica Difusa: Una introducción práctica. Castilla, España: Universidad de Castilla.

Iglesias Sánchez, E. J., García, Y., Sanjuan, M., Camacho, O., & Smith, C. (Febrero de 2007). Fuzzy surface-based sliding mode control. *ISA Transactions*, Volumen 46, Issue 1, Pages 73-83.

Ingmecafenix. (2018). Obtenido de http://www.ingmecafenix.com/automatizacion/encoder/

Limón Marruedo, D., Alamo, T., & Camacho, E. (10-13 de Diciembre de 2002). Input-to-state stable MPC for constrained discrete-time nonlinear systems with bounded additive uncertainties. Las Vegas, Nevada, USA.

Morcillo, C. G. (s.f.). Lógica Difusa - Una introducción práctica.

Ogata, K. (2010). *Ingeniería de Control Moderna.* Madrid: Pearson Educación.

Pajarón Pérez, R. (2012). *Control Borroso Industrial: Uso de la IEC 1131-7 para el control de plantas industriales.* Sevilla.

Páucar, A. G. (2000). *Teoría y Análisis de Máquinas Eléctricas.* Lima - Perú.

Rodríguez Ramírez, D., & Bordóns Alba, C. (2005). *Análisis y control de sistemas en espacio de estado identificación de sistemas.* España. Obtenido de https://es.wikipedia.org/wiki/Espacio_de_estados

Saá Tapia, F. D. (2017). *Plataforma para pruebas de algoritmos de control de velocidad de un motor de corriente continua Pololu.* Riobamba: ESPOCH.

Vera, O. Z. (18 de Mayo de 2011). Lógica Difusa. México.

Vissim. (2016). *Online Courses Vissim*. Obtenido de http://www.online-courses.vissim.us/Strathclyde/measures_of_controlled_system_pe.htm

Voice, E. (2018). *Electrical Voice*. Obtenido de https://electricalvoice.com/performance-indices-in-control-system-ise-iae-itse-itae/

ANEXOS