



UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA
La Universidad Católica de Loja

ÁREA TÉCNICA

**TÍTULO DE INGENIERO EN ELECTRÓNICA Y
TELECOMUNICACIONES**

**Diseño, implementación y análisis de desempeño de un controlador
PID de orden fraccionario (FOPID) aplicado a control embebido de
sistemas mecatrónicos.**

TRABAJO DE TITULACIÓN.

AUTORES: Macas Peña, Edwin Stalin
Sarango Chamba, Roger Augusto

DIRECTOR: Calderón Córdova, Carlos Alberto, Ing.

LOJA – ECUADOR

2017



Esta versión digital, ha sido acreditada bajo la licencia Creative Commons 4.0, CC BY-NY-SA: Reconocimiento-No comercial-Compartir igual; la cual permite copiar, distribuir y comunicar públicamente la obra, mientras se reconozca la autoría original, no se utilice con fines comerciales y se permiten obras derivadas, siempre que mantenga la misma licencia al ser divulgada. <http://creativecommons.org/licenses/by-nc-sa/4.0/deed.es>

Septiembre, 2017

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Ingeniero

Carlos Alberto Calderón Córdova

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación: Diseño, implementación y análisis de desempeño de un controlador PID de orden fraccionario (FOPID) aplicado a control embebido de sistemas mecatrónicos, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, Mayo de 2017

f)

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

Nosotros Macas Peña Edwin Stalin, Sarango Chamba Roger Augusto declaramos ser los autores del presente trabajo de titulación: Diseño, implementación y análisis de desempeño de un controlador PID de orden fraccionario (FOPID) aplicado a control embebido de sistemas mecatrónicos, de la Titulación de Ingeniería en Electrónica y Telecomunicaciones, siendo el Ing. Carlos Alberto Calderón Córdova director del presente trabajo: y eximimos expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además, certificamos que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de nuestra exclusiva responsabilidad.

Adicionalmente declaramos conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: “Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad”.

f.....

Autor: Edwin Stalin Macas Peña

Cédula: 0704640440

f.....

Autor: Roger Augusto Sarango Chamba

Cédula: 1104128622

DEDICATORIA

Dedico el presente trabajo a Dios, por darme salud para terminar el proyecto en cuestión. A mis padres Jorge y Dalinda que siempre han estado pendientes en cada etapa de mi vida, por haberme inculcado buenos valores y por la confianza que me brindan. A mi hermano Adrián por ser como mi segundo padre. A la familia Camacho Macas, por haberme dado acogida, su afecto y apoyo durante mi carrera universitaria. A Aide por su apoyo incondicional.

Edwin Stalin

A mi padre, por ser la persona que me ha apoyado incondicionalmente a lo largo de mi vida. A mi esposa Marjorie por su apoyo y amor constante, quien día tras día me ayuda a ser mejor persona.

Roger Augusto

AGRADECIMIENTO

A nuestros padres, por su apoyo y motivación brindada en el transcurso de nuestra carrera universitaria.

A nuestro director de tesis, el Ing. Carlos Calderón por su ayuda durante la ejecución de este proyecto.

ÍNDICE DE CONTENIDOS

CARÁTULA	i
APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN.....	ii
DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS.....	iii
DEDICATORIA	iv
AGRADECIMIENTO	v
ÍNDICE DE CONTENIDOS.....	vi
ÍNDICE DE FIGURAS.....	ix
ÍNDICE DE TABLAS.....	xi
RESUMEN.....	1
ABSTRACT	2
INTRODUCCIÓN.....	3
CAPITULO I.....	6
1. ESTADO DEL ARTE DEL SOFTWARE Y HARDWARE PARA EL CONTROL EMBEBIDO DE MOTORES DC.....	6
1.1. Arquitectura de hardware.....	7
1.1.1. Motor DC.....	7
1.1.2. Reductor de engranaje recto.....	9
1.1.3. Encoder incremental.....	10
1.1.4. Driver (amplificador).....	11
1.1.5. sbRIO 9651.....	12
1.2. Arquitectura software.....	13
1.3. Controladores.....	14
1.3.1. Controladores PID y PI.....	14
1.3.2. Controladores FOPID y FOPI.....	16
1.3.3. Parámetros de la respuesta transitoria.....	17
1.3.3.1. Índices de error.....	17
1.3.3.2. Parámetros temporales.....	18
1.4. Herramienta computacional FOMCON.....	20
1.5. Aproximación de los operadores fraccionarios.....	23
CAPÍTULO II.....	26
2. DESCRIPCIÓN DEL SISTEMA Y DISEÑO DEL CONTROLADOR.....	26
2.1. Modelo matemático del motor DC.....	27
2.2. Metodología de diseño de controladores.....	31
2.2.1. Controladores enteros PID y PI.....	31
2.2.1.1. Identificación del sistema.....	32

2.2.1.2.	Ajuste de parámetros de controladores.....	35
2.2.2.	Controladores fraccionarios FOPID y FOPI.....	40
2.2.2.1.	Herramientas computacionales para control fraccionario.....	44
2.3.	Simulación de los controladores IOPID y FOPID.....	45
2.3.1.	Simulación de los controladores PID.....	45
2.3.1.1.	Controladores basados en la curva de reacción.....	46
2.3.1.2.	Controladores basados en el mínimo índice de desempeño.....	48
2.3.1.3.	Controladores basados en la robustez del sistema.....	51
2.3.1.4.	Controladores basados en filtrado derivativo.....	53
2.3.2.	Simulación de los controladores PI.....	55
2.3.3.	Simulación de los controladores FOPID.....	58
2.3.4.	Simulación de controladores FOPI.....	69
CAPÍTULO III.....		72
3.	IMPLEMENTACIÓN DEL SISTEMA DE CONTROL EMBEBIDO.....	72
3.1.	Arquitectura Hardware.....	73
3.1.1.	Conexión del sistema embebido.....	75
3.1.1.1.	Interfaz con el sensor Encoder incremental.....	75
3.1.1.2.	Interfaz sbRIO-9651 con Motor dc.....	76
3.2.	Desarrollo del proyecto en la plataforma sbRIO-9651.....	77
3.2.1.	Creación del proyecto.....	77
3.2.2.	Subprocesos en el FPGA.....	79
3.2.3.	Subprocesos en el Procesador Real Time.....	81
3.2.3.1.	Proceso de exportación de datos.....	84
3.3.	Síntesis de la Función de Transferencia del FOPID.....	86
CAPÍTULO IV.....		90
4.	ANÁLISIS DE RESULTADOS.....	90
4.1.	Análisis de desempeño de controladores.....	91
4.1.1.	Análisis y comparación de parámetros temporales.....	92
4.1.1.1.	Parámetros temporales de controladores PID.....	92
4.1.1.2.	Parámetros temporales de controladores PI.....	94
4.1.1.3.	Parámetros temporales de controladores FOPID.....	95
4.1.1.4.	Parámetros temporales de controladores FOPI.....	97
4.1.2.	Evaluación de desempeño basado en índices de error.....	99
4.1.2.1.	Índices de desempeño de controladores PID y PI.....	99
4.1.2.2.	Índices de desempeño de controladores FOPID Y FOPI.....	100
CONCLUSIONES.....		101

BIBLIOGRAFÍA..... 102

ÍNDICE DE FIGURAS

Figura 1.1. Arquitectura hardware del sistema embebido.	7
Figura 1.2. Sistema modular Maxon.	7
Figura 1.3. Conjunto Maxon, formado por motor, encoder y reductor.	9
Figura 1.4. Diseño esquemático de un encoder magnético.	10
Figura 1.5. Driver MC33926.	11
Figura 1.6. Arquitectura software.	13
Figura 1.7. Sistema de control PID en lazo cerrado.	15
Figura 1.8. Forma no interactuante del controlador.	15
Figura 1.9. Aportaciones de un PID tradicional y un FOPID.	17
Figura 1.10. PID fraccional.	17
Figura 1.11. Diagrama en lazo cerrado.	17
Figura 1.12. Respuesta transitoria.	19
Figura 1.13. Relación de FOMCON con paquetes similares.	20
Figura 1.14. Relación de los módulos de FOMCON.	21
Figura 1.15. Interface gráfica de la función iopid_tune.	22
Figura 1.16. Interface gráfica de la función fpid_tune.	22
Figura 1.17. Interface gráfica de la función fpid.	23
Figura 2.1. Sistema electromecánico del motor DC.	27
Figura 2.2. Representación de diagrama de bloques del motor DC en lazo abierto.	28
Figura 2.3. Respuesta al escalón del modelo matemático en lazo abierto.	31
Figura 2.4. Método de la tangente para la identificación del proceso.	33
Figura 2.5. Método de dos puntos para la identificación del proceso.	34
Figura 2.6. Respuesta al escalón de modelos del motor DC.	35
Figura 2.7. Diagrama de bloques del sistema de control PID.	37
Figura 2.8. Diagrama de bloques del sistema de control con filtrado derivativo.	39
Figura 2.9. Distribución de controladores PID.	45
Figura 2.10. Respuesta del controlador Ziegler-Nichols.	46
Figura 2.11. Respuesta del controlador Chien-Hrones-Reswick, servomecanismo.	47
Figura 2.12. Respuesta del controlador Chien-Hrones-Reswick, regulador.	47
Figura 2.13. Respuesta del controlador Cohen-Coon.	48
Figura 2.14. Respuesta del controlador Murrill-IAE.	49
Figura 2.15. Respuesta del controlador Arrieta Orozco-IAE.	49
Figura 2.16. Respuesta del controlador Murrill- ITAE.	50
Figura 2.17. Respuesta del controlador Smith-ITAE.	51
Figura 2.18. Respuesta del controlador Rivera.	52
Figura 2.19. Respuesta del controlador Brambilla.	52
Figura 2.20. Respuesta del controlador con filtrado derivativo de Alfaro Ruiz.	53
Figura 2.21. Respuesta del controlador con filtrado derivativo de Arrieta Orozco.	54
Figura 2.22. Respuesta del controlador con filtrado derivativo de Tavakoli.	54
Figura 2.23. Respuesta del controlador con filtrado derivativo de Arrieta Orozco.	55
Figura 2.24. Respuesta del controlador PI de Ziegler-Nichols.	56
Figura 2.25. Respuesta del controlador PI de Cohen-Coon.	56
Figura 2.26. Respuesta del controlador PI de Chien como regulador.	57
Figura 2.27. Respuesta del controlador PI de Chien como servomecanismo.	57
Figura 2.28. Código en Matlab de la FT del motor.	59
Figura 2.29. Identificación de la planta.	59
Figura 2.30. Importación de datos para la identificación.	60
Figura 2.31. Respuesta al escalón a la planta.	60
Figura 2.32. Función de transferencia obtenida de la identificación.	61
Figura 2.33. Parámetros obtenidos de la identificación.	61
Figura 2.34. Gráfica del modelo obtenido a partir de la identificación.	62
Figura 2.35. Sintonización PID convencional.	62

Figura 2.36. Sintonización FOPID.....	63
Figura 2.37. Selección de desempeño para el controlador.	63
Figura 2.38. Aproximación entera del FOPID.	63
Figura 2.39. Funcion de transferencia entera del FOPID.	63
Figura 2.40. Controlador Ziegler-Nichols, Nelder-Mead, IAE.	64
Figura 2.41. Controlador Ziegler-Nichols, Nelder-Mead, ISE.	65
Figura 2.42. Controlador Ziegler-Nichols, Nelder-Mead, ITSE.	65
Figura 2.43. Controlador Ziegler-Nichols, Nelder-Mead, ITAE.	66
Figura 2.44. Controlado Cohen-Coon, Nelder-Mead, ISE.	66
Figura 2.45. Controlador Cohen-Coon, Nelder-Mead, IAE.	67
Figura 2.46. Controlador Cohen-Coon, Nelder-Mead, ITSE.....	67
Figura 2.47. Controlador Cohen-Coon, Nelder-Mead, ITAE.....	68
Figura 2.48. Controlador FMINCON, INTERIOR-POINT, ISE.	68
Figura 2.49. Controlador FMINCON, INTERIOR-POINT, IAE.	69
Figura 2.50. Controlador FOPI CHIEN-Regulador, Nelder-Mead, ISE.....	69
Figura 2.51. Controlador FOPI CHIEN-Regulador, FMINCON, ISE.....	70
Figura 2.52. Controlador FOPI CHIEN-Regulador, FMINCON, ITAE.....	70
Figura 2.53. Controlador FOPI C-H-R, Nelder Mead, ISE.	71
Figura 3.1. Diagrama de bloques del sistema embebido de control.....	73
Figura 3.2. Arquitectura del sistema de control del motor DC.	74
Figura 3.3. Configuración experimental del sistema embebido.	74
Figura 3.4. Conexión de Encoder-PMOD3.	75
Figura 3.5. Conexión del controlador de potencia-PMOD4.	76
Figura 3.6. Toolkits instalados en el dispositivo sbRIO-9651.	78
Figura 3.7. Estructura del proyecto del sistema embebido.	78
Figura 3.8. Adquisición y procesamiento de la señal digital del encoder en FPGA.	79
Figura 3.9. Generación de la señal PWM.	80
Figura 3.10. Habilitación de pines de salida.	80
Figura 3.11. Comunicación entre el Procesador en Tiempo Real y FPGA.	81
Figura 3.12. Medición de velocidad angular.	82
Figura 3.13. Controlador PID.	83
Figura 3.14. Sistema embebido de control.	84
Figura 3.15. Muestreo de señal Set Point y señal Controlada.....	84
Figura 3.16. Interfaz gráfica para el control PID.	85
Figura 3.17. Datos obtenidos de la experimentación de controladores.....	85
Figura 3.18. Discretización de una FT continua en Matlab.....	87
Figura 3.19. Diagrama de bloques para discretizar una FT continua.....	88
Figura 3.20. FT discreto para controladores fraccionarios.	88
Figura 3.21. FOPID discreto en Real-Time.....	89
Figura 3.22. Ventana de configuración para el controlador discreto.	89
Figura 4.1. Configuración experimental del sistema de control.	91

ÍNDICE DE TABLAS

Tabla 1.1. Datos del motor A-max 26 353111	8
Tabla 1.2. Datos del reductor.	10
Tabla 1.3. Datos del sensor encoder.	11
Tabla 1.4. Datos del driver MC33926.	12
Tabla 1.5. Conexión del driver con la tarjeta controladora.	12
Tabla 2.1: Parámetros del motor DC.	30
Tabla 2.2: Pasos a realizar para el ajuste de controladores PID y PI.....	36
Tabla 2.3: Reglas de ajustes basadas en la curva reacción.....	37
Tabla 2.4: Reglas de ajustes basadas en el criterio de mínimo índice de desempeño.	38
Tabla 2.5: Reglas de ajustes basadas en el criterio de robustez.	39
Tabla 2.6: Reglas de ajustes basadas en el criterio de filtrado derivativo.	39
Tabla 2.7: Reglas de ajustes para controladores PI.	40
Tabla 2.8: Herramientas para el modelado de controladores fraccionarios.	44
Tabla 2.9. Parámetros de la respuesta temporal de controladores de orden entero.	58
Tabla 2.10: Parámetros de la respuesta temporal de controladores FOPID y FOPI.....	71
Tabla 3.1: Conexión del driver con la tarjeta sbRIO-9651.	76
Tabla 3.2: Conexión del driver, fuente de alimentación y motor DC A-max 26.	77
Tabla 4.1: Señales de respuesta de los controladores PID.	93
Tabla 4.2: Señales de respuesta de los controladores PI.	94
Tabla 4.3: Parámetros temporales de controladores de orden entero.	95
Tabla 4.4: Señales de respuestas de los controladores FOPID.	96
Tabla 4.5: Señales de respuestas de los controladores FOPI.	97
Tabla 4.6: Señales de respuesta de los controladores FOPI.	98
Tabla 4.7: Parámetros temporales de controladores de orden fraccionario.	99
Tabla 4.8: Índices de desempeño de los controladores PID y PI de orden entero.....	100
Tabla 4.9: Índices de desempeño de los controladores PID y PI de orden fraccionario.	100

RESUMEN

El presente trabajo forma parte del proyecto “Mano de Esperanza”, el objetivo es diseñar e implementar un controlador embebido PID de orden fraccionario (FOPID), aplicado a la regulación de velocidad de la articulación del codo en una prótesis robótica. El propósito es primeramente realizar una investigación preliminar sobre la factibilidad de implementar un controlador FOPID en un sistema embebido autónomo; finalmente, se busca analizar y comparar el desempeño de los controladores FOPID y PID clásico.

Como resultados del presente trabajo: se diseñaron e implementaron 7 controladores PID clásico y 10 controladores FOPID. La implementación se la realizó en una plataforma embebida FPGA-Procesador (sbRIO-9651) y en el lenguaje LabVIEW FPGA y RT. Finalmente se realizó la evaluación experimental y comparación de los controladores en base a los parámetros temporales: tiempo de establecimiento, porcentaje de sobreelongación; y a los índices de error: IAE, ITAE, ISE e ITSE. Los controladores embebidos de mejor desempeño fueron del tipo FOPID Z-N-ISE ($T_s=0.22s$, P.O.=7.18%) y PI Chien ($T_s=0.18s$, P.O.=0%).

PALABRAS CLAVES: PID, FOPID, control de velocidad del motor DC, sbRIO 9651.

ABSTRACT

The present work is part of the “Hand of Hope” Project, the objective is to design and implement an embedded Fractional Order PID controller (FOPID), applied to the elbow joint speed in a robotic prosthesis. The purpose is to, first, conduct a preliminary research about the feasibility of implementing a FOPID controller in an autonomous embedded system; second, we seek to analyze and compare the performance of classic PID and FOPID controllers.

As results of this work: seven classic PID controllers and ten FOPID controllers were designed and implemented. The implementation was performed in an embedded platform FPGA-Processor (sbRIO-9651) and in the LabVIEW FPGA & RT language.

Finally, experimental evaluation and comparison of controllers were performed, based on the time parameters: settling time, overshoot percentage; and on the error rates: IAE, ITAE, ISE and ITSE. The embedded controllers with the best performance were FOPID Z-N-ISE ($T_s = 0.22$ s, P.O. = 7.18 %) and PI Chien ($T_s = 0.18$ s, P.O. = 0 %).

KEYWORDS: PID, FOPID, DC motor speed control, sbRIO 9651.

INTRODUCCIÓN

Según el CONADIS (Consejo Nacional para la Igualdad de Discapacidades), hasta febrero de 2017 en el Ecuador existen 196.758 personas con discapacidad física-motriz, y de éste número, 48.271 personas están insertadas laboralmente, lo que representa que solamente el 24.5% ha tenido las condiciones y herramientas protésicas para poder desempeñar una actividad laboral [1]. En base al problema anterior nace el proyecto “Mano de Esperanza”, cuya misión es desarrollar prótesis robóticas de bajo costo con la finalidad de aportar a la inclusión social y laboral de personas con discapacidad motriz de sus extremidades superiores, con enfoque hacia países en desarrollo [2].

El presente trabajo forma parte del proyecto “Mano de Esperanza”, el objetivo es diseñar e implementar un controlador embebido PID de orden fraccionario (FOPID), aplicado a la regulación de velocidad de la articulación del codo en una prótesis robótica. El propósito es primeramente realizar una investigación preliminar sobre la factibilidad de implementar un controlador FOPID en un sistema embebido autónomo; finalmente, se busca analizar y comparar el desempeño de los controladores FOPID y PID clásico, implementados en el mismo sistema embebido.

Actualmente, los controladores de orden fraccionario son ampliamente usados en el entorno científico e industrial con el fin de lograr un desempeño robusto de un sistema dinámico [3]. El controlador PID fraccionario es una extensión del controlador PID clásico, el cual es muy popular por su simplicidad en el diseño y por su buen desempeño especialmente para sistemas lentos [4]. El controlador FOPID es menos sensible a los cambios en los parámetros y a las incertidumbres en general de un sistema controlado, es decir es más robusto y estable [3], [5], [6]; esta característica es deseada en nuestro sistema debido a que las condiciones de funcionamiento de la articulación del codo son altamente variables y dependen de la magnitud de carga mecánica que el usuario de la prótesis esté manipulando.

Los parámetros de un controlador FOPID son: constante proporcional (K_p), constante de integración (K_I), constante derivativa (K_D), orden de integración (λ) y orden del diferenciador (μ). Encontrar el conjunto óptimo de valores para estos cinco parámetros, con el fin de cumplir con las especificaciones de usuario, se necesita una optimización en un hiperespacio de cinco dimensiones [3], en conclusión la sintonización de un controlador FOPID requiere de un proceso sistemático y complejo. Además de la complejidad, existe una diversa gama de estructura de controladores y métodos de sintonización. De acuerdo a [6] y [7], estos métodos pueden ser divididos en tres

diferentes categorías: métodos basados en reglas (por ejemplo [6], [7]), métodos analíticos (por ejemplo [8], [9]) y métodos numéricos (por ejemplo [10], [11]).

Complementando al análisis anterior, en el entorno académico existen algunas herramientas software, por ejemplo *Ninteger*, *FOPID*, *FMCON*, *CRONE*, *FOTF*, entre otras [3], [12]. Estas herramientas se crearon para automatizar el análisis, la simulación y la implementación de sistemas de orden fraccionario, por lo tanto, son de utilidad para el objetivo perseguido en el presente trabajo, ya que diseñaremos una cantidad considerable de controladores basados en diferentes métodos de diseño y de sintonización.

Similar al problema del diseño de controladores FOPID, la implementación de estos controladores en un sistema de control en tiempo real, tiene sus particularidades, como por ejemplo determinar funciones de aproximación equivalentes de orden entero y analizar el desempeño del controlador embebido autónomo. Existe un número considerable de aplicaciones reales de los controladores FOPID en sistemas físicos basados en motores DC, sin embargo, las funciones de control no están completamente ejecutándose en el sistema embebido. Por ejemplo en [13] se utiliza la tarjeta Advantech PCI-1710-HG solamente para adquirir y emitir las señales desde y hacia el proceso, la función de control se encuentra ejecutándose en Simulink de MathWorks, de manera similar en [14] y [15] se implementó un controlador FOPID no autónomo basado en Simulink y en las tarjetas NI USB-6008 y NI USB-6259, respectivamente. Otra alternativa común para la implementación de funciones de control fraccionario es LabVIEW, por ejemplo en [16] y [17] se implementó un controlador completamente autónomo en equipos de alto desempeño NI cRIO-9014 y NI PXI-1031 y LabVIEW, sin embargo estos equipos son de alto costo (> 5.000 USD) y además se implementó una función de aproximación, más no la función analítica original. Finalmente en [18] y [19] se implementaron controladores FOPID no autónomos por medio de LabVIEW.

Basado en la problemática anterior, en el presente trabajo se implementó una función de control FOPID en un sistema embebido completamente autónomo, con la característica adicional de que la plataforma de desarrollo es un sistema de bajo costo basado en un FPGA y un Procesador (plataforma sbRIO-9651). Se definió utilizar un sistema FPGA como el hardware base para la implementación de un controlador embebido, debido a que según [16] posee: bajo consumo de potencia, tiempos de ejecución muy cortos, dimensiones reducidas de un dispositivo con alta velocidad computacional, implementación hardware más simple que otros procesadores, y la

posibilidad de implementar un desarrollo complejo en un dispositivo con alto nivel de portabilidad.

Para la demostración del sistema de control FOPID embebido se lo definió aplicar a un motor DC Maxon de 2940 rpm, el cual controlaría la flexión y extensión del codo que se adecuará a la prótesis documentada en [2] y [20].

Por último, para la evaluación experimental y comparación de los controladores embebidos PID clásico y FOPID se definió utilizar los parámetros temporales (tiempo de establecimiento, porcentaje de sobrelongación) y los índices de error (IAE, ITAE, ISE e ITSE).

Con respecto al desarrollo del presente proyecto, se encuentra estructurado de la siguiente manera. En el capítulo 1 se realiza el análisis de parámetros de los distintos elementos que van a ser usados en el proyecto, basándose en varios criterios para la selección del hardware. En el capítulo 2 se obtiene el modelo matemático que describe el funcionamiento dinámico del sistema físico, posteriormente se diseña y analiza los sistemas de control de orden entero y fraccionario, luego se describe las metodologías que fueron usadas y la demostración del funcionamiento mediante simulaciones de control usando herramientas de programación gráfica (LabVIEW). En el capítulo 3 se sintetiza los controladores PID, PI, FOPID y FOPI en la plataforma sbRIO-9651. Finalmente, en el capítulo 4 se implementa el sistema físico y se procede a realizar pruebas con los controladores, se analiza y compara los resultados de los parámetros temporales y desempeño.

CAPITULO I

1. ESTADO DEL ARTE DEL SOFTWARE Y HARDWARE PARA EL CONTROL EMBEBIDO DE MOTORES DC

1.1. Arquitectura de hardware.

En la presente sección se describirán los elementos y dispositivos de la arquitectura hardware que servirá para la implementación y evaluación de las funciones de control PID y FOPID. La arquitectura hardware del presente proyecto se muestra en la Figura 1.1. Además en esta sección se documenta la selección del Motor DC, del sensor de velocidad y de caja reductora, tomando en cuenta que su aplicación es agregar el grado de libertad perteneciente al codo para la prótesis robótica implementada en [2].

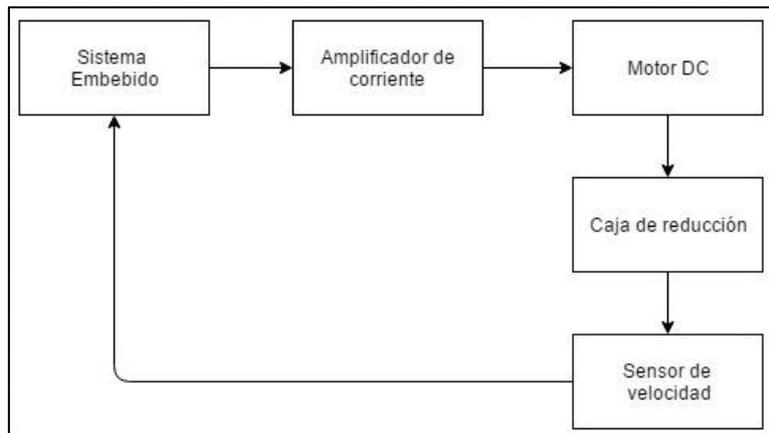


Figura 1.1. Arquitectura hardware del sistema embebido.

Fuente: Autores.

Elaboración: Autores.

1.1.1. Motor DC.

Para el desarrollo de este proyecto se escogió un motor comercial de imán permanente de la marca MAXON, de la gama A-max, esto debido a las siguientes razones: altas prestaciones, disponibilidad de los parámetros necesarios para modelar el motor DC, compatibilidad certificada con módulos de medición, cajas de engrajes y tarjetas controladoras. En la Figura 1.2 se muestra el motor DC con sus módulos compatibles.

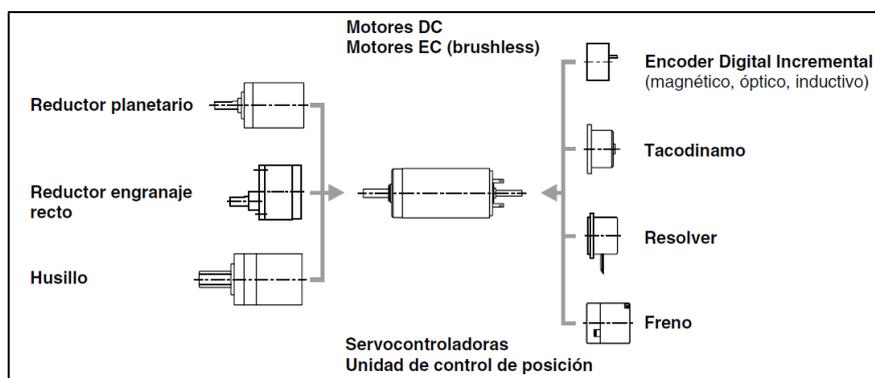


Figura 1.2. Sistema modular Maxon.

Fuente: [21].

Elaboración: Autores.

El sistema modular que ofrece la marca Maxon, permite una gran cantidad de combinaciones de motores, reductores, encoders, frenos y tarjetas controladoras, lo cual permite elegir los dispositivos adecuados para la aplicación que se requiera.

Para la selección del motor DC, se tomó en cuenta los siguientes criterios:

- Rendimiento superior al 80%.
- Voltaje de operación: 6 V.
- Par nominal mayor a 8 mNm.
- Corriente de arranque, máximo 2 A.
- Potencia de consumo menor a 5 W.

El dispositivo seleccionado fue el modelo A-max 26, de escobillas de metal precioso, de 44.7 mm de longitud y con una potencia de 4.5 W. En la Tabla 1.1 se detalla las especificaciones técnicas y los parámetros internos del motor. Todos los datos mostrados provienen del catálogo del fabricante, el motor seleccionado tiene la referencia 353111.

Tabla 1.1. Datos del motor A-max 26 353111

Valores de tensión nominal		
Tensión nominal	V	6
Velocidad en vacío	Rpm	4090
Corriente en vacío	mA	29.2
Velocidad nominal	Rpm	2940
Par nominal	mNm	11.3
Corriente nominal	A	0.84
Par de arranque	mNm	39.2
Corriente de arranque	A	2.83
Máximo rendimiento	%	81
Parámetros		
Resistencia en bornes	Ω	2.12
Inductancia en bornes	mH	0.227
Constante de par	mNm/A	13.9
Constante de velocidad	Rpm/V	689
Relación velocidad/par	Rpm/mNm	105
Constante de tiempo mecánica	ms	15
Inercia del rotor	gcm ²	13.6

Fuente: [21].

Elaboración: Autores.

El motor tiene dos grupos de cables para su conexión como se muestra en la Figura 1.3. El un grupo es para la alimentación y el otro es destinado al sensor de velocidad (encoder).



Figura 1.3. Conjunto Maxon, formado por motor, encoder y reductor.
Fuente: Autores.
Elaboración: Autores.

1.1.2. Reductor de engranaje recto.

Se seleccionó un reductor que permita el movimiento rotatorio del codo con alto par y con la correspondiente reducción de velocidad. Se requiere una velocidad máxima de 15 rpm, por lo que se busca en el catálogo provisto por el fabricante un reductor que cumpla con ciertos criterios de selección.

$$n_L = \frac{1 \text{ rev}}{4 \text{ seg}} * 1 \text{ seg}$$

$$n_L = 0.25 \frac{\text{rev}}{\text{seg}} * \frac{60 \text{ seg}}{1 \text{ min}}$$

$$\mathbf{n_L = 15 \text{ rpm}} \quad (1.1)$$

Para seleccionar el reductor, se divide la velocidad nominal del motor (entrada, n_{mot}) y la velocidad deseada (salida, n_L), obteniendo así la constante de reducción necesaria para la aplicación. La relación de reducción aproximada, se obtiene a partir de la siguiente fórmula [21]:

$$i_{max} = \frac{n_{mot}}{n_L} = \frac{2940 \text{ rpm}}{15 \text{ rpm}} = 195$$

$$\mathbf{195: 1} \quad (1.2)$$

Donde:

i_{max} : Relación de reducción máxima.

n_{mot} : Velocidad nominal del motor.

n_L : Velocidad deseada.

Siguiendo la recomendación del fabricante, se tiene varios posibles dispositivos para el sistema modular Maxon, de estos se escogió el reductor de engranaje recto GS 30 A de cinco etapas, cuya referencia es 110449 y su reducción es 200:1, cercano al valor deseado. En la Tabla 1.2 se muestra los datos del reductor seleccionado.

Tabla 1.2. Datos del reductor.

Característica	Unidad	Valor
Reducción		200:1
Número de etapas		5
Máximo rendimiento	%	60
Longitud del reductor	mm	30.5

Fuente: Autores.

Elaboración: Autores.

1.1.3. Encoder incremental.

Se seleccionó el encoder MR Tipo ML (1000 ppv) porque permite trabajar a velocidades altas; tiene disponibles canales con “line driver”, así se evita que las interferencias causen una pérdida en la señal digital del encoder.

El encoder incremental digital de la marca MAXON es bidireccional (dos canales de salida A y B) transmite señales de onda cuadrada, cuyos pulsos permiten detectar la dirección, la posición y velocidad de giro del eje. El desfase entre las señales de los canales A y B permite su comparación con el propósito de determinar el sentido de giro, el canal B está desfasado 90° eléctricos respecto al canal A. El canal Index (Canal I) puede ser usado como punto de referencia para determinar con precisión el ángulo de rotación recorrido [21].

Las características de este sensor son:

- Señal de posición relativa apta para posicionamiento.
- Reconocimiento del sentido de giro.
- Información de velocidad por la frecuencia de los pulsos.
- Solución estándar para muchas aplicaciones.

El encoder también genera las señales complementarias de \bar{A} , \bar{B} e \bar{I} , estas permiten eliminar interferencias, son usadas en entornos donde hay ruido y cuando se conectan cables de longitud larga. El sensor se basa en el principio magnético, en el que un pequeño imán permanente multipolar se coloca en el eje del motor como se muestra en la Figura 1.4.

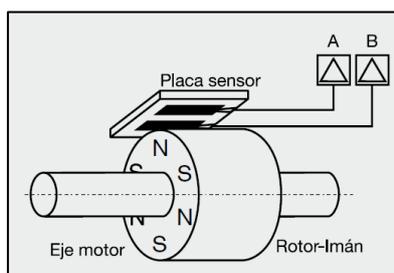


Figura 1.4. Diseño esquemático de un encoder magnético.

Fuente: [21].

Elaboración: Autores.

Las variaciones del flujo magnético son captadas por la placa sensor y se transmiten por los canales A y B a la electrónica de la tarjeta controladora. En la Tabla 1.3 se muestra la información relevante del encoder seleccionado.

Tabla 1.3. Datos del sensor encoder.

Características	Unidad	Valor
Número de pulsos por vuelta	ppv	1000
Número de canales	-	3
Máxima frecuencia de funcionamiento	kHz	200
Máxima velocidad	rpm	12000

Fuente: Autores.

Elaboración: Autores.

1.1.4. Driver (amplificador).

El controlador de motor DC MC-33926, fue seleccionado por ser un controlador para motores de corriente continua de tamaño medio, tiene rangos de operación que se ajustan con las necesidades del sistema de control presentado en este proyecto.

Es un driver de motor que suministra 3 A de corriente continua a un motor DC de 5 V – 28 V y puede soportar hasta corrientes pico de 5 A, este driver soporta modulación de ancho de pulso (PWM) de hasta 20 kHz, presenta una función que devuelve la corriente consumida por el motor conectando el pin analógico. El puente H (MC-33926) es un semiconductor de Freescale del fabricante Pololu [22]. Este driver se muestra en la Figura 1.5.

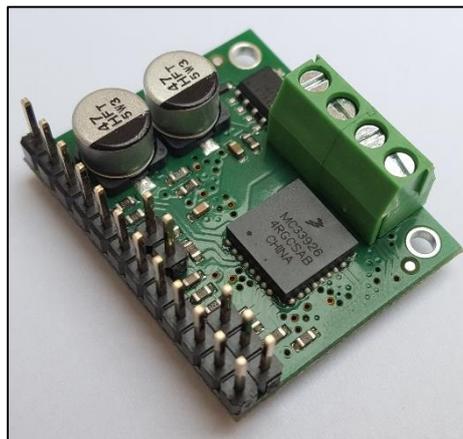


Figura 1.5. Driver MC33926.

Fuente: Autores.

Elaboración: Autores.

Para una aplicación de control de motor básica se usan seis líneas E/S para conectar el driver del motor a una tarjeta controladora, se conectan las entradas **IN1** e **IN2** para el control de dirección del motor; se desactiva **D1** o **D2**, para el control de velocidad PWM; el pin **SF** (status flag) supervisa los errores que pueda tener el driver durante su funcionamiento; el pin que supervisa el consumo de corriente **FB** se conecta a una

entrada del convertidor analógica/digital de la tarjeta controladora y la entrada **INV** para el control de dirección con IN1(en alto) e IN2(en bajo) o viceversa [22]. En la Tabla 1.4 se observa las principales especificaciones del driver.

Tabla 1.4. Datos del driver MC33926.

Características	Unidad	Valor
Canales para motor		1
Tensión de operación	V	5
Tensión máxima de operación	V	28
Corriente máxima por canal	A	5
Frecuencia PWM	KHz	20
Voltaje lógico mínimo	V	2.5
Voltaje lógico máximo	V	5.5

Fuente: Autores

Elaboración: Autores.

Los pines necesarios para la conexión del driver con el resto de dispositivos (plataforma sbRIO- 9651, motor DC) que se usaron en el proyecto se muestran en la Tabla 1.5.

Tabla 1.5. Conexión del driver con la tarjeta controladora.

Tarjeta controladora	Driver MC 33926
Salida PWM (Velocidad)	D2
Salida digital (1 o 0)	In1
Salida digital (0 o 1)	In2
Gnd	Gnd
5V	En
Gnd	D1

Fuente: Autores.

Elaboración: Autores.

1.1.5. sbRIO 9651.

El Kit de Desarrollo para el System-on-Module (SOM) es una tarjeta para el desarrollo de aplicaciones embebidas de monitoreo y control. El System-on-Module sbRIO-9651 proporciona un procesador embebido de tiempo real y una tarjeta FPGA reconfigurable, esta tarjeta provee interfaces de E/S y alimentación [23].

El kit de desarrollo contiene:

- SOM (System-on-Module).
- Tarjeta de referencia.
- Fuente de alimentación de escritorio.
- Cable USB.
- 20 monturas.
- Disipador de calor.

1.2. Arquitectura software.

La tarjeta sbRIO-9651 puede ser reconfigurado a través del software de National Instruments. Se requieren módulos para el control de dispositivos y diseño de aplicaciones. En la Figura 1.6 se muestra la arquitectura software, que consiste en todos los módulos requeridos para la implementación del sistema embebido en la plataforma sbRIO-9651.

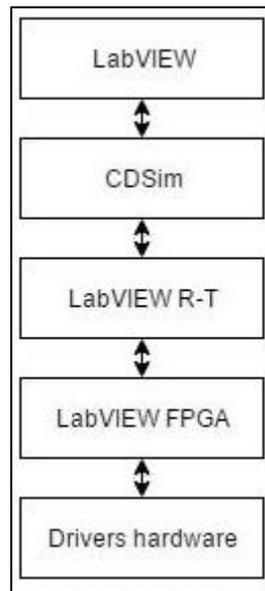


Figura 1.6. Arquitectura software.
Fuente: Autores.
Elaboración: Autores.

- LabVIEW

Constituye un entorno de programación gráfica destinada al desarrollo de aplicaciones que involucran adquisición, control, análisis y presentación de datos. LabVIEW emplea lenguaje de programación gráfica o lenguaje G para crear programas en diagramas de bloques, dichos programas se denominan Instrumentos virtuales (VIs) y estos a su vez tienen un panel frontal en el que se trata de una interfaz gráfica.

- CDSim

Es un módulo de LabVIEW, que permite analizar el comportamiento de modelos de lazo abierto, diseñar controladores en lazo cerrado y similar Sistema de control y realizar implementaciones de controladores en hardware.

- LabVIEW RT

El módulo Real -Time de National Instruments es un componente software complementario para el sistema de LabVIEW, una vez instalado, este software compila

el código gráfico y lo optimiza con el objetivo de que trabaje en tiempo real, este módulo incluye los paquetes de PID Control necesario para realizar el control embebido en la plataforma sbRIO-9651 [24].

- LabVIEW FPGA

El Módulo NI LabVIEW FPGA extiende la plataforma de desarrollo gráfico de LabVIEW para programar FPGA's en hardware de entradas/salidas reconfigurables (RIO) de National Instruments. Las siguientes características de LabVIEW no están disponibles en el FPGA [24]:

- Funciones de punto flotante.
- Arrays de tamaño variable y multidimensional.
- Clústeres o cadenas de errores.
- Cuadros de diálogo.
- E / S de archivos.
- Impresión.

Por este motivo, algunas funciones de programación deben ser implementadas en el procesador Real-Time.

1.3. Controladores.

El diseño de sistemas de control debe cumplir con ciertas especificaciones en el dominio del tiempo como sobrelongación, tiempo de levantamiento y tiempo de establecimiento para una entrada de escalón [25].

Los controladores para el control de un motor DC propuestos en este proyecto son PID (Proporcional-Integral-Derivativo) y FOPID (PID de orden fraccionario). El controlador más usado actualmente en la industria es el controlador PID.

Existen muchos métodos para el determinar los valores de los parámetros de un controlador, el proceso de selección de los parámetros se los conoce como de ajuste o sintonización de controladores.

1.3.1. Controladores PID y PI.

Los controladores PID son ampliamente usados en sistemas de control debido al bajo número de parámetros que se deben ajustar, y proporcionan señales de control que son proporcionales al error entre la señal de referencia y la señal de salida real.

El controlador PID está dada por la Ecuación (1.3) [26]:

$$U(t) = K_p e(t) + \frac{K_p}{T_i} \int_0^t e(t) dt + K_p T_d \frac{de(t)}{dt} \quad (1.3)$$

Donde $U(t)$ y $e(t)$ son las señales de control y error respectivamente, los parámetros a sintonizar son K_p, T_i, T_d y la correspondiente función de transferencia es:

$$K(s) = \frac{U(s)}{E(s)} = K_c \left(1 + \frac{1}{T_i s} + T_d s \right) \quad (1.4)$$

Donde K_p es la ganancia proporcional, T_i es el tiempo integral y T_d es el tiempo derivativo, para lograr un rendimiento de control óptimo se ajustan las ganancias de cada uno de estos parámetros. Las características del controlador PID es la anticipación de cambios en la salida con la acción derivativa y la eliminación del error de estado estacionario de la respuesta. En la Figura 1.7 se observa el sistema de control PID.

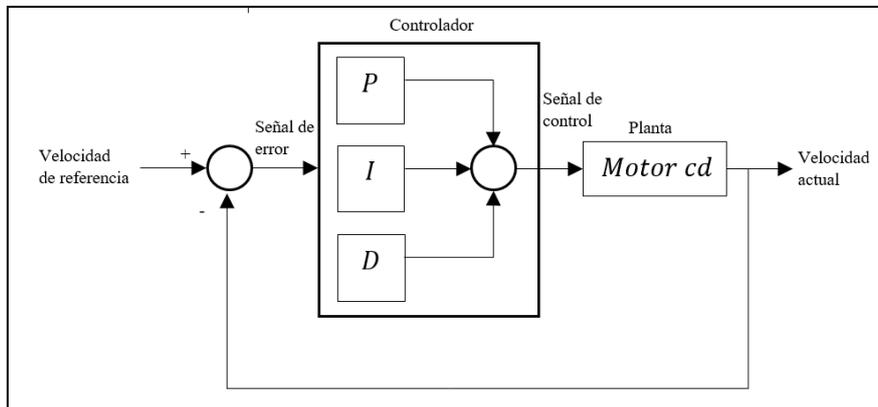


Figura 1.7. Sistema de control PID en lazo cerrado.

Fuente: [26].

Elaboración: Autores.

En la Figura 1.8, se muestra que la señal de control es la suma de los tres términos del controlador: el término P (proporcional al error), el término I (integral del error), cuya función principal es asegurar que la salida del proceso coincida con el punto de consigna en estado estacionario y el término D (derivada del error) que mejora la estabilidad en lazo cerrado [26].

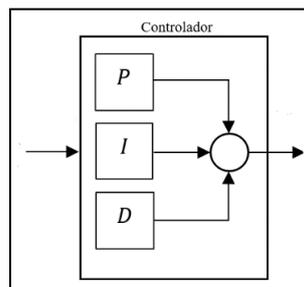


Figura 1.8. Forma no interactuante del controlador

Fuente: Autores.

Elaboración: Autores.

Los controladores PID no son los únicos que existen, hay algunas variaciones de estos controladores, como el controlador con acción proporcional e integral; estos parámetros al igual para los controladores PID son ajustables con reglas de ajustes. La función de transferencia de un controlador PI es:

$$K(s) = \frac{U(s)}{E(s)} = K_c \left(1 + \frac{1}{T_i s} \right) \quad (1.5)$$

1.3.2. Controladores FOPID y FOPI.

La ecuación integro-diferencial que define la acción de control del FOPID está dada por la Ecuación (1.6).

$$u(t) = K_p e(t) + K_i D^{-\lambda} e(t) + K_d D^{\mu} e(t) \quad (1.6)$$

De la Ecuación (1.6) se puede deducir la ecuación integro-diferencial del FOPI, en la que eliminando la parte derivativa se obtiene la Ecuación (1.7).

$$u(t) = K_p e(t) + K_i D^{-\lambda} e(t) \quad (1.7)$$

La transformada de Laplace aplicada a la Ecuación (1.6) con condiciones iniciales nulas, da como resultado la función de transferencia del controlador FOPID que se expresa en la Ecuación (1.8) [27]:

$$C_{\text{FOPID}}(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu} ; \quad \lambda, \mu \in \mathbb{R}^+ \quad (1.8)$$

Para obtener la función de transferencia del controlador FOPI se aplica la transformada de Laplace a la Ecuación (1.7) con condiciones nulas, dando como resultado la Ecuación (1.9).

$$C_{\text{FOPI}}(s) = K_p + \frac{K_i}{s^{\lambda}} ; \quad \lambda \in \mathbb{R}^+ \quad (1.9)$$

Donde λ es el orden integral y μ es el orden derivativo que son números reales positivos, K_p es la ganancia proporcional, K_i es la constante de integración y K_d es la constante de derivación [27].

En la Figura 1.9, se puede observar las posibilidades que ofrece un PID clásico donde se limitan solo a la selección de los vértices, mientras que las posibilidades que ofrece un PID de orden fraccional ocupa toda la superficie del cuadrado [27].

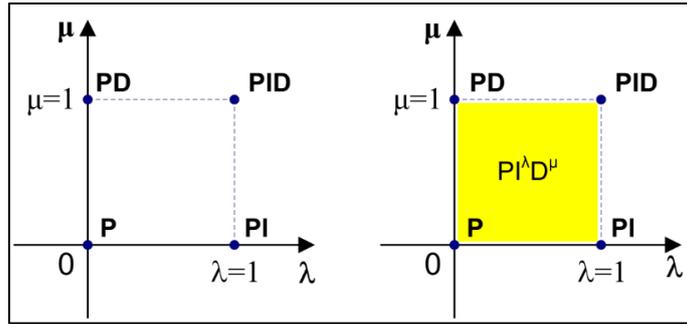


Figura 1.9. Aportaciones de un PID tradicional y un FOPID.
Fuente: [27].
Elaboración: Autores.

La Figura 1.10 muestra el esquema de control de un PID fraccional.

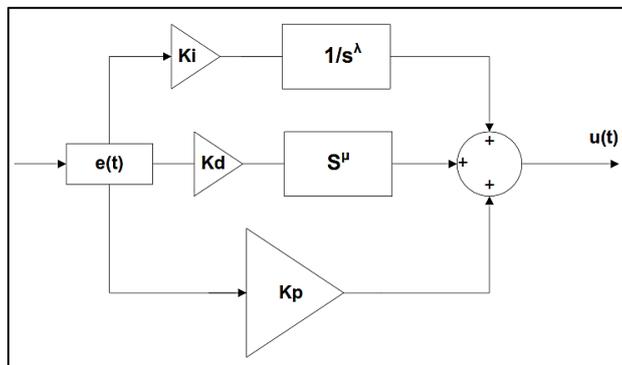


Figura 1.10. PID fraccional.
Fuente: [28].
Elaboración: Autores.

En la Figura 1.11 se muestra el diagrama en lazo cerrado con realimentación unitaria del controlador PID de orden fraccional con la función de transferencia de la planta en este caso del motor Maxon.

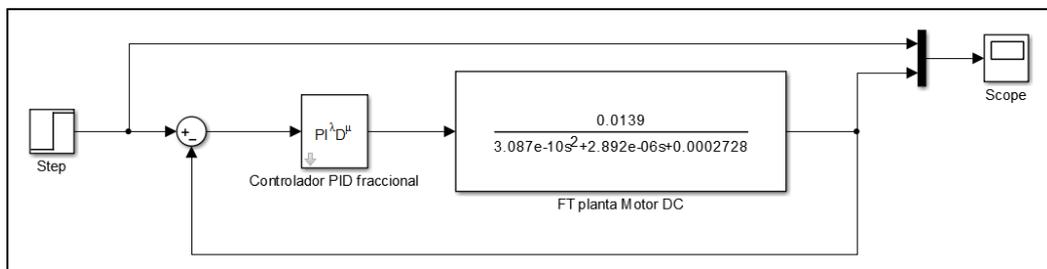


Figura 1.11. Diagrama en lazo cerrado.
Fuente: Autores.
Elaboración: Matlab/Simulink.

1.3.3. Parámetros de la respuesta transitoria.

Para determinar que controladores tienen mejores características, se ha propuesto analizarlos y compararlos basándonos en los métodos que se nombran a continuación

1.3.3.1. Índices de error.

Para cuantificar el comportamiento de los lazos de control existen los índices de desempeño que se basan en la señal de error $e(t)$, la cual es la diferencia entre el valor deseado de la variable controlada y su valor real. Los más conocidos son los llamados criterios integrales que se definen como [29]:

- Integral del error absoluto.

$$IAE = \int_0^{\infty} |e(t)| dt \quad (1.10)$$

- Integral del tiempo por el error absoluto.

$$ITAE = \int_0^{\infty} t|e(t)| dt \quad (1.11)$$

- Integral del error cuadrático.

$$ISE = \int_0^{\infty} e(t)^2 dt \quad (1.12)$$

- Integral del tiempo por el error cuadrático.

$$ITSE = \int_0^{\infty} te(t)^2 dt \quad (1.13)$$

En donde el error está dado por

$$e(t) = r(t) - y(t) \quad (1.14)$$

Para una planta dada, el objetivo es determinar los parámetros del controlador que minimizan la función de costo escogida, por lo que estos parámetros son óptimos bajo el criterio de desempeño establecido [29].

1.3.3.2. Parámetros temporales.

La respuesta transitoria de un sistema de control práctico, con frecuencia, muestra oscilaciones amortiguadas antes de alcanzar un estado estacionario. Si la salida de un sistema en estado estacionario no coincide exactamente con la entrada, se dice que el sistema tiene un error en estado estacionario. Este error indica la precisión del sistema [30]. En la Figura 1.12 se muestra la respuesta transitoria para una entrada de escalón unitario, las especificaciones de la respuesta transitoria son:

Tiempo de retardo, t_d : Tiempo requerido para que la respuesta alcance la primera vez la mitad del valor final.

Tiempo de subida, t_r : Tiempo requerido para que la respuesta pase del 10 al 90%, del 5 al 95% o del 0 al 100% su valor final.

Tiempo pico, t_p : Tiempo requerido para que la respuesta alcance el primer pico de sobreelongación.

Sobreelongación máxima, M_p : Es el máximo valor del pico (porcentaje) de la curva de respuesta, medido a partir de la unidad, esta cantidad indica de manera directa la estabilidad relativa del sistema.

Tiempo de establecimiento, t_s : Tiempo que se requiere para que la curva de respuesta alcance un rango alrededor del valor final del tamaño especificado por el porcentaje absoluto del valor final (por lo general, de 2 o 5%)

De las especificaciones anteriores, las que se ha considerado relevantes son la sobreelongación máxima y el tiempo de establecimiento. La primera porque nos permite proteger el motor al no aplicarle un controlador con una sobreelongación que pueda dañarlo. Y la segunda porque nos permite conocer el tiempo en llegar al margen (2 o 5%) de estabilización propuesto.

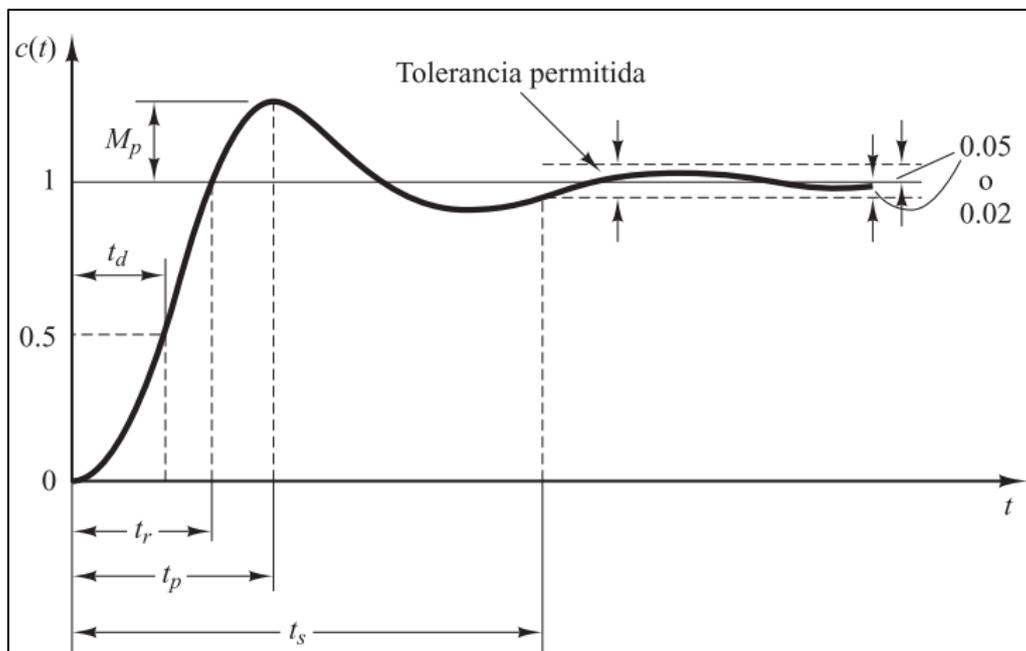


Figura 1.12. Respuesta transitoria.

Fuente: [30].

Elaboración: Autores.

1.4. Herramienta computacional FOMCON.

La herramienta FOMCON (Fractional Order Modeling and control), es un toolbox basado en cálculo de orden fraccionario para modelado de sistemas y diseño de control. El núcleo de esta herramienta se deriva de la caja de herramientas FOTF (Fractional Order Transfer Functions). El objetivo principal del análisis en FOMCON es una función de transferencia de orden fraccionario de la forma [31]:

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \dots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \dots + a_0 s^{\alpha_0}} \quad (1.15)$$

FOMCON está relacionado con otros Toolbox de MATLAB orientadas al cálculo de orden fraccionario, tales como CRONE y NINTEGER a través de las características de conversión del modelo de sistema o código compartido, esta relación se muestra en la Figura 1.13 [31].

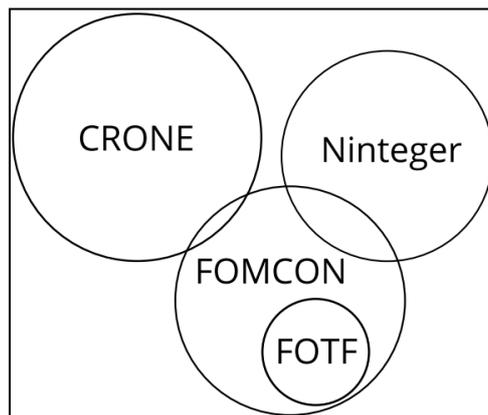


Figura 1.13. Relación de FOMCON con paquetes similares.
Fuente: [31].
Elaboración: Autores.

La característica de este toolbox es que se centra en los sistemas SISO (single input-single output) y LTI (linear time-invariant), además se compone de los siguientes módulos [31]:

- Módulo principal (análisis de sistemas fraccionales).
- Módulo de identificación (sistema de identificación en el dominio del tiempo y frecuencia).
- Módulo de control (Diseño del controlador PID fraccional, herramientas de ajuste y optimización, así como algunas características adicionales).
- Módulo de implementación (aproximaciones de tiempo continuo y discreto, implementación de los correspondientes filtros analógicos y digitales) [31].

Todos los módulos mencionados anteriormente están interconectados y se pueden acceder desde la GUI (Interfaz Gráfica de Usuario) del módulo principal como se puede observar en la Figura 1.14 [31].

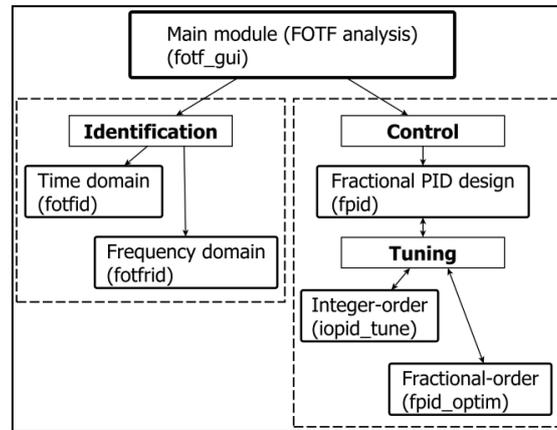


Figura 1.14. Relación de los módulos de FOMCON.
Fuente: [32].
Elaboración: Autores.

Las funciones destacadas para el diseño del controlador fraccionario son:

- Caja de herramientas del sistema de control, necesaria para la mayoría de las funciones.
- Caja de herramientas de optimización, necesaria para la identificación en el dominio del tiempo, la sintonización del PID convencional, y también parcialmente para la sintonización del controlador PID de orden fraccionario [31].

A continuación, se muestran las funciones usadas del toolbox FOMCON con su respectivo GUI para la sintonización del controlador fraccionario.

- Función *lopid_tune*.

Esta herramienta permite calcular el PID de manera computacional mediante los métodos: Ziegler-Nichols, Cohen-Coon, Astrom-Hagglund, etc. Que posteriormente se utilizarán como valores iniciales para la sintonización del PID de orden fraccional. La interfaz gráfica se muestra en la Figura 1.15.

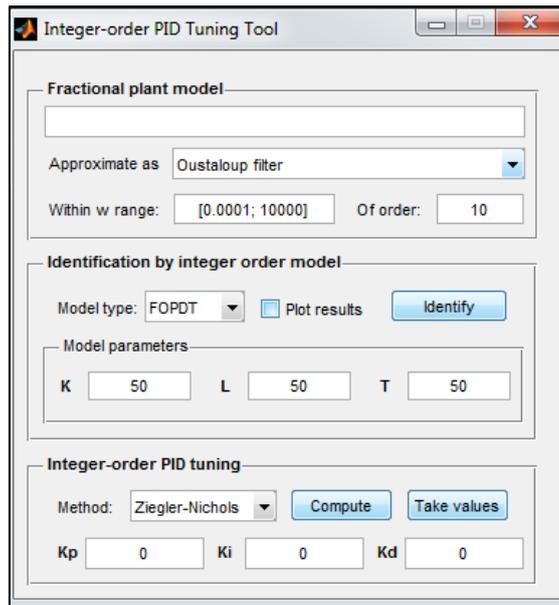


Figura 1.15. Interface gráfica de la función `iopid_tune`.
Fuente: Matlab.
Elaboración: Autores.

- Función `fpid_optim`.

Para sintonizar el PID de orden fraccionario se usa esta herramienta, donde se coloca en "LTI system" la variable con la que está declarada la función de transferencia de la planta en el workspace, luego se coloca el PID clásico antes calculado, posteriormente se selecciona entre los algoritmos de optimización sin restricción Nelder-Mead y el método con restricciones donde se usa la función FMINCON. La interfaz gráfica se muestra en la Figura 1.16.

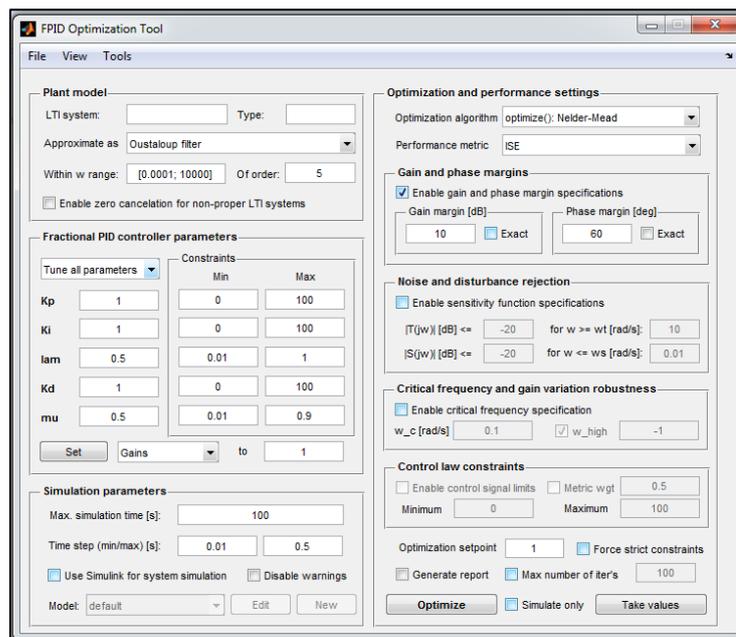


Figura 1.16. Interface gráfica de la función `fpid_tune`.
Fuente: Matlab.
Elaboración: Autores.

- Función *fpid*.

Esta herramienta permite simular el controlador PID de orden fraccionario calculado previamente con la función *fpid_optim*, para su uso se colocan los valores de los cinco parámetros del FOPID, luego, en la opción “system” se coloca la variable con la que está declarada la función de transferencia de la planta en el workspace. La interfaz gráfica se muestra en la Figura 1.17.

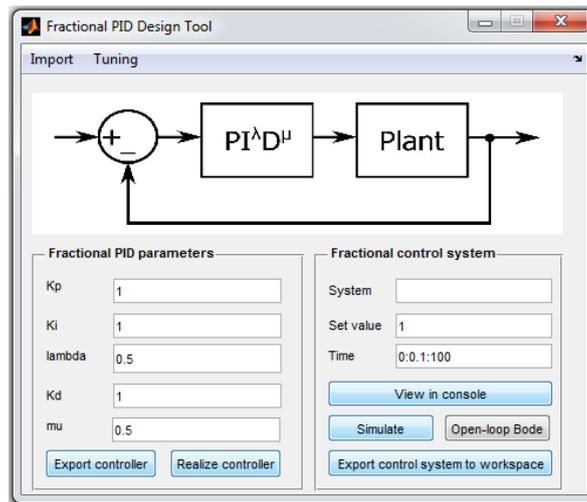


Figura 1.17. Interface gráfica de la función *fpid*.
Fuente: Matlab.
Elaboración: Autores.

1.5. Aproximación de los operadores fraccionarios.

La implementación de funciones de transferencia con operadores fraccionarios:

$$s^\alpha ; \alpha \in \mathbb{R}^n$$

No se puede realizar en el software LabVIEW por lo que necesariamente se debe aproximar los controladores de orden fraccionario, a controladores de orden entero con los que trabaja dicho software. No es posible decidir cuál aproximación es la mejor y por lo que no existe un criterio único, para su selección se optó por escoger entre las aproximaciones que se mencionan a continuación por ser las más usadas [33].

- Aproximación Crone.

Mediante una distribución recursiva de ceros y polos esta metodología permite aproximar de forma continua el operador fraccionario s^α . Esta distribución alterna los ceros y los polos en intervalos elegidos previamente, de tal forma que se pueden formar funciones de transferencia cuya ganancia varía linealmente con el logaritmo de la frecuencia y con fase prácticamente constante dentro de un intervalo definido (w_l, w_h) . Por tanto, el objetivo es adquirir un comportamiento en frecuencia lo más parecido

posible al comportamiento que tiene el operador fraccionario s^α . La aproximación de la función de transferencia tiene la forma [33]:

$$G(s) = s^\alpha = C_0 \prod_{n=1}^N \frac{1 + \frac{s}{w_{sn}}}{1 + \frac{s}{w_{pn}}} \quad (1.16)$$

- Aproximación de Matsuda.

Esta metodología que se basa en la aproximación no racional, por medio de una función racional obtenida mediante expansión en fracciones continuas fue propuesta por Matsuda y Fujii (1993). Sus valores coinciden con los de la función original en un conjunto de puntos logarítmicamente espaciados. La expresión de esta aproximación es la siguiente [33]:

$$G(s) = a_0 + \frac{s - \omega_0}{a_1 + \frac{s - \omega_1}{a_2 + \frac{s - \omega_2}{a_3 + \dots}}} = a_0 + \frac{s - \omega_0}{a_1 +} \frac{s - \omega_1}{a_2 +} \frac{s - \omega_2}{a_3 +} \quad (1.17)$$

- Aproximación de Carlson.

Esta metodología proporciona aproximaciones continuas de los operadores fraccionarios, fue propuesta por Carlson y Halijak (1964). Rescribiendo la función de transferencia en la forma (1.11), luego resolviendo esta ecuación mediante el método de Newton para aproximar la raíz α de un número de forma iterativa (1.13), tomando como punto de partida (1.12). Este método solo funciona para valores $1/\alpha \in \mathbb{Z}$, así que α sólo podrá tomar valores como: $1/2$, $1/4$, [33].

$$G_{\frac{1}{\alpha}}(s) = s \quad (1.18)$$

$$G_0(s) = 1 \quad (1.19)$$

$$G_i(s) = G_{i-1}(s) \frac{\left(\frac{1}{\alpha} - 1\right) G_{i-1}^{\frac{1}{\alpha}}(s) + \left(\frac{1}{\alpha} + 1\right) s}{\left(\frac{1}{\alpha} + 1\right) G_{i-1}^{\frac{1}{\alpha}}(s) + \left(\frac{1}{\alpha} - 1\right) s} \quad (1.20)$$

Las herramientas necesarias para transformar un PID de orden fraccional a una función de transferencia con potencias enteras de "S", nos brinda el toolbox "NINTEGER", mediante el uso del comando "nipid", el mismo con la siguiente sintaxis [34]:

$$C = \text{nipid}(K_p, K_d, \mu, K_i, \lambda, \text{ancho_de_banda}, n, \text{fórmula})$$

Donde:

$K_p, K_d, \mu, K_i, \lambda$: Son los parámetros del PID fraccional.

ancho_de_banda: Es el ancho de banda del controlador.

n: es el número de polos y ceros de la aproximación.

fórmula: Es el tipo de aproximación utilizada (en este caso crone) [34].

CAPÍTULO II

2. DESCRIPCIÓN DEL SISTEMA Y DISEÑO DEL CONTROLADOR

2.1. Modelo matemático del motor DC.

El elemento principal del sistema a controlar está formado por un motor DC Maxon de 2940 rpm, el cual formaría parte de la articulación del codo de una prótesis robótica de extremidades superiores. Los motores DC son los elementos comunes en los sistemas robóticos aplicados a prótesis, sin embargo, para su aplicación efectiva dentro del sistema se necesita primeramente determinar el modelo matemático para luego diseñar el sistema de control. Este capítulo iniciará con la determinación del modelo matemático del sistema y luego se abordará el diseño de los controladores PID de orden entero y de orden fraccionario.

Con respecto al modelo matemático, éste intenta aproximar la relación de las variables de entrada sobre las variables de salida del sistema, se construyen con la finalidad de que correspondan lo más cercano posible con el mundo real y siempre se lo debe considerar como una aproximación del sistema físico [35].

El motor DC es un sistema formado por dos componentes: eléctrico y mecánico. Los parámetros que comprenden las ecuaciones del subsistema eléctrico son la resistencia de armadura (R_a) y la inductancia (L_a), los parámetros de las ecuaciones del subsistema mecánico son el coeficiente de fricción (B_m) y el momento de inercia (J_m), estos dos componentes están acoplados por la constante de par (k_m) [36].

Una representación equivalente simplificada de los componentes del motor DC controlada por armadura se muestra en la Figura 2.1.

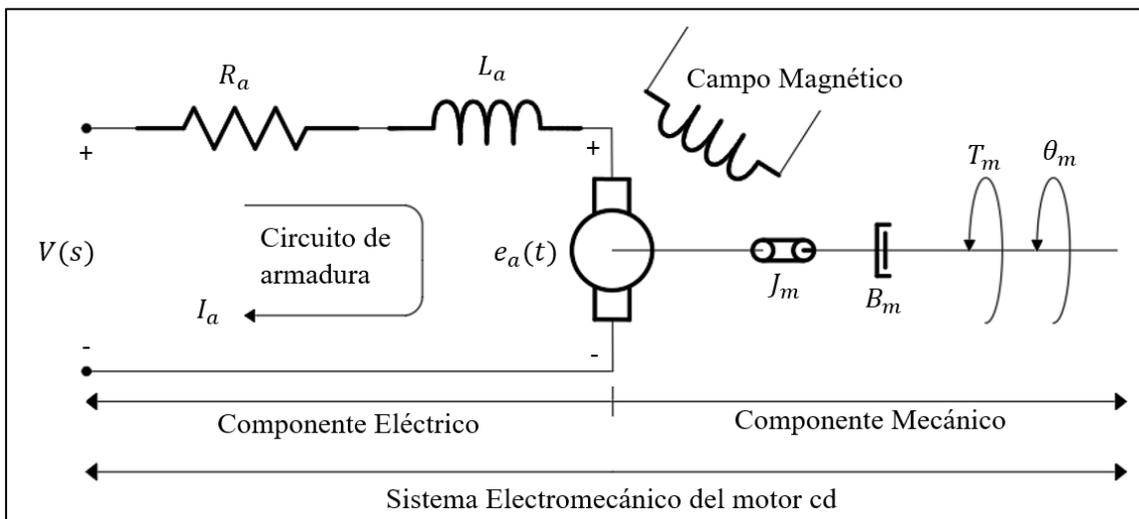


Figura 2.1. Sistema electromecánico del motor DC.

Fuente: [36], [37].

Elaboración: Autores.

Considerando los componentes eléctrico y mecánico del motor DC, el modelo matemático puede ser representado en el dominio de la frecuencia usando un diagrama de bloques como se muestra en la Figura 2.2, la función de transferencia de la velocidad con respecto al voltaje de armadura se obtiene mediante el análisis y reducción del diagrama de bloques, adicional se recalca que de entre las opciones: control del voltaje de campo y control de voltaje de armadura, se prefiere el segundo debido a que permite una amplia variación de velocidad de giro y par, además permite una alta suavidad en la regulación [38].

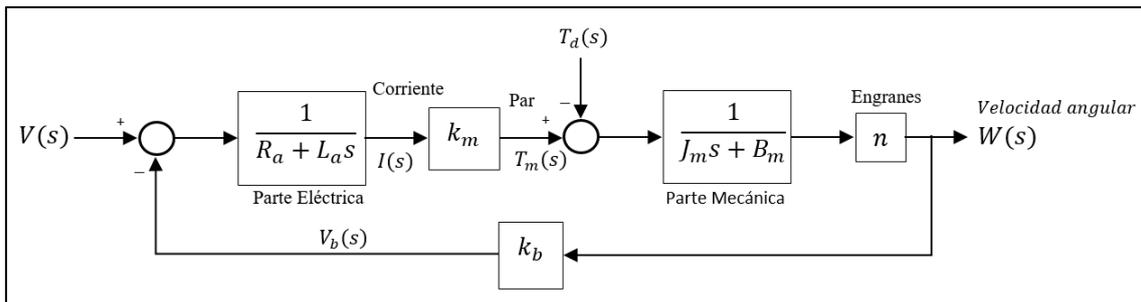


Figura 2.2. Representación de diagrama de bloques del motor DC en lazo abierto.

Fuente: [36].

Elaboración: Autores.

El motor DC es un sistema en lazo cerrado, para mantener la velocidad angular constante, este sistema presenta una realimentación de la fuerza contra-electromotriz, la cual es proporcional a la velocidad de rotación del eje [36]. A partir del diagrama de bloques de la Figura 2.2, se obtiene la función de transferencia del motor DC.

$$P_1(s) = G_1(s)G_2(s), \quad L_1(s) = -k_b G_1(s)G_2(s), \quad T_d(s) = 0 \quad (2.1)$$

$$\begin{aligned} G_p(s) &= \frac{P_1(s)}{1 - L_1(s)} \\ &= \frac{G_1(s)G_2(s)}{1 + k_b G_1(s)G_2(s)} \\ &= \frac{\left(\frac{k_m}{R_a + L_a s}\right) \left(\frac{1}{J_m s + B_m}\right)}{1 + k_b \left(\frac{k_m}{R_a + L_a s}\right) \left(\frac{1}{J_m s + B_m}\right)} \\ &= \frac{\frac{k_m}{(R_a + L_a s)(J_m s + B_m)}}{1 + \frac{k_b k_m}{(R_a + L_a s)(J_m s + B_m)}} \end{aligned}$$

$$\begin{aligned}
&= \frac{\frac{k_m}{(R_a + L_a s)(J_m s + B_m)}}{\frac{(R_a + L_a s)(J_m s + B_m) + k_b k_m}{(R_a + L_a s)(J_m s + B_m)}} \\
&= \frac{k_m [(R_a + L_a s)(J_m s + B_m)]}{[(R_a + L_a s)(J_m s + B_m)][(R_a + L_a s)(J_m s + B_m) + k_b k_m]} \\
G_p(s) &= \frac{V_m}{W_m} = \frac{k_m}{(J_m s + B_m)(L_a s + R_a) + k_b k_m} \quad (2.2)
\end{aligned}$$

La Ecuación (2.2) es la función de transferencia del motor DC, donde V_m es el voltaje aplicado a la armadura del motor, W_m es la velocidad angular, k_m es la constante de par, J_m la inercia del rotor, L_a es la inductancia de armadura y R_a es la resistencia de armadura; sin embargo, dos parámetros no se encuentran en el catálogo del fabricante, la constante de fuerza contra-electromotriz (k_b) y el coeficiente de fricción viscosa B_m , entonces es necesario calcular estos parámetros que faltan.

A partir de la constante de velocidad ($k_n = 689 \frac{rpm}{V}$), valor obtenido en [21], se calcula la constante de fuerza contra-electromotriz mediante la Ecuación (2.3).

$$\begin{aligned}
k_b &= \frac{1}{k_n} = \frac{1}{\left(689 \frac{rpm}{V}\right) * \frac{2\pi}{60}} \\
k_b &= 0.013859 \frac{V}{rad/s} \quad (2.3)
\end{aligned}$$

Los parámetros k_b y k_m son las constantes de fuerza contra-electromotriz y de par del motor DC, respectivamente; cuando éstas se expresan en unidades SI en la Ecuación (2.4), ambos valores deben ser iguales.

$$k_b \left[\frac{V}{rad/s} \right] = k_m \left[\frac{Nm}{A} \right] \quad (2.4)$$

$$0.013859 \cong 0.0139$$

Para obtener el coeficiente de fricción viscosa, se realiza una aproximación matemática [39], de B_m la cual resulta de la Ecuación (2.5).

$$J \frac{dW_n}{dt} = k_b i_a - B_m W_n \quad (2.5)$$

Despejando B_m , la ecuación resultante sería:

$$B_m = \frac{k_b * I_a}{W_n} \quad (2.6)$$

$$B_m = \frac{0.013859 * 0.84}{307.87608}$$

$$B_m = 0.00003781$$

Donde:

I_a : Corriente de armadura del motor, 0.84 A.

W_n : Velocidad nominal del motor, 2940 rpm = 307.87608 rad/s.

En la Tabla 2.1 se muestran los parámetros utilizados en la simulación del motor DC, la mayoría de estos valores fueron obtenidos del catálogo del fabricante Maxon:

Tabla 2.1: Parámetros del motor DC.

Parámetro	Descripción	Unidad	Valor
$W(t)$	Velocidad angular	rad/s	307.87608
$V_{in}(t)$	Tensión nominal	V	6.00
R_a	Resistencia de armadura	ohms	2.12
L_a	Inductancia de armadura	H	0.000227
J_m	Inercia del rotor	kg * m ²	0.00000136
B_m	Coeficiente de fricción viscosa	N * m / rad/s	0.00003781
k_b	Constante de fuerza contra electromotriz	V / rad/s	0.013859
k_m	Constante de par	N * m / A	0.0139
k_n	Constante de velocidad	rpm / V	689

Fuente: [21].

Elaboración: Autores.

Reemplazando los valores de la Tabla 2.1 en la Ecuación (2.2), la función de transferencia del motor A-max 26, 4.5 W es:

$$G_p(s) = \frac{0.0834}{3.0872e - 10s^2 + 2.89178e - 6s + 0.000272797} \quad (2.7)$$

Una vez obtenida la función de transferencia y al aplicar una señal de entrada tipo escalón de 6V, se obtiene la respuesta del motor en lazo abierto (Figura 2.3), la cual sirve para evaluar el comportamiento del modelo del motor de corriente continua.

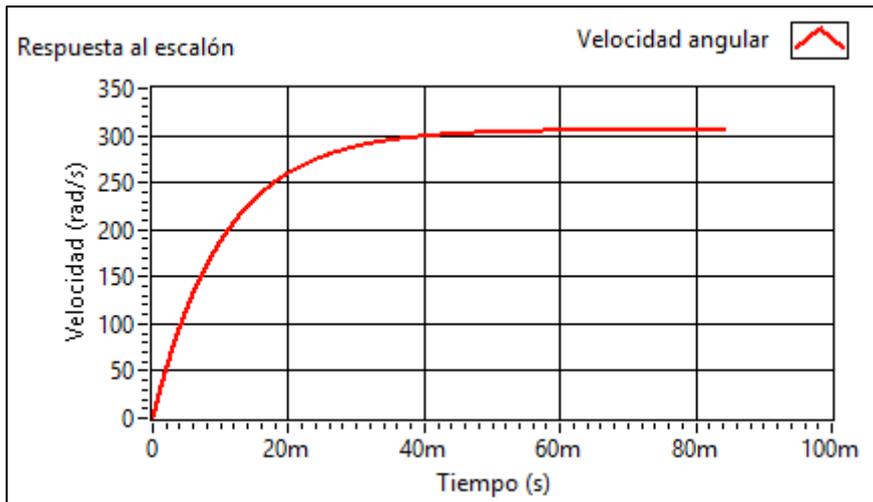


Figura 2.3. Respuesta al escalón del modelo matemático en lazo abierto.
 Fuente: Autores.
 Elaboración: Autores.

Como se observa en la Figura 2.3 el sistema alcanza una velocidad de $305.722 \frac{rad}{s}$, con un tiempo de establecimiento de $48 ms$. Dado que la respuesta del sistema está en $\frac{rad}{s}$, se desea obtener una respuesta en RPM, para ello se realiza una conversión mediante la Ecuación (2.8):

$$Velocidad(rpm) = 305.722 \frac{rad}{s} * \frac{1rev}{2\pi * rad} * \frac{60seg}{1min} \quad (2.8)$$

Comparando con los valores de [21], donde la velocidad nominal del motor A-max 26 es de $2940 rpm$; el resultado obtenido de la simulación y realizando la conversión con la Ecuación (2.8), se obtiene una velocidad de $2919.43 rpm$ y se puede verificar que ambos valores son similares.

2.2. Metodología de diseño de controladores.

Una vez obtenido el modelo matemático del motor DC, el siguiente paso es el diseño de los controladores PID de orden entero y orden fraccionario. La presente sección abordará las metodologías de diseño de controladores y finalmente la sintonización de las funciones de control. El procedimiento inicia con la identificación del proceso y luego se realiza el ajuste de controladores PID y FOPID usando diferentes técnicas disponibles en la literatura de control de procesos, y finalmente se desarrolla simulaciones para comprobar el funcionamiento de los controladores.

2.2.1. Controladores enteros PID y PI.

Después de construir el modelo matemático del motor DC se procede a diseñar los controladores de orden entero, el proceso de seleccionar los parámetros del controlador

que cumpla con las especificaciones de comportamiento del sistema en lazo cerrado se conoce como sintonía o ajuste del controlador [25].

Para el correcto diseño de un sistema de control se realizó dos etapas: identificación del sistema, en el que se obtiene la información de la dinámica del proceso (aproximación del modelo a un sistema de primer orden más tiempo de retardo) y la sintonización del controlador. Para la identificación del proceso, se basó en dos métodos de la curva de reacción del proceso (respuesta al escalón unitario), estos métodos son realizados en lazo abierto del modelo del motor DC, donde el controlador puede o no estar en el sistema y son procedimientos gráficos.

El método de lazo abierto se basa en la curva de reacción del proceso, y se caracteriza por los parámetros que sirven para la sintonización de los controladores [25]. El procedimiento a realizar para la prueba del escalón unitario es el siguiente:

- Se coloca el sistema en lazo abierto.
- Se aplica un escalón unitario en la entrada de la planta, se registra la señal de salida del proceso hasta que el sistema se estabilice.

Realizado esto, se procede a la identificación del sistema, la función de transferencia obtenida del motor DC es un sistema de segundo orden y puede ser aproximado mediante un modelo de primer orden más tiempo de retardo.

2.2.1.1. Identificación del sistema.

La obtención de los parámetros para la identificación de este sistema: la ganancia de la planta (k), el tiempo de retardo (T_m), la constante de tiempo (τ) y $a = \frac{kT}{\tau}$, se lo hace mediante la curva de reacción. Hay dos métodos que emplean esta técnica, el primer método se traza una recta tangente en el punto de inflexión de la curva y se determina las intersecciones de la tangente con el eje del tiempo [40].

La función de transferencia resultante de la identificación del motor DC, usando los métodos de la recta tangente y de dos puntos sería el siguiente modelo de primer orden más tiempo de retardo:

$$G_{pRT}(s) = e^{-T_m} \frac{k}{\tau s + 1}$$

Para el desarrollo del presente proyecto se dará prioridad al modelo de primer orden más tiempo de retardo.

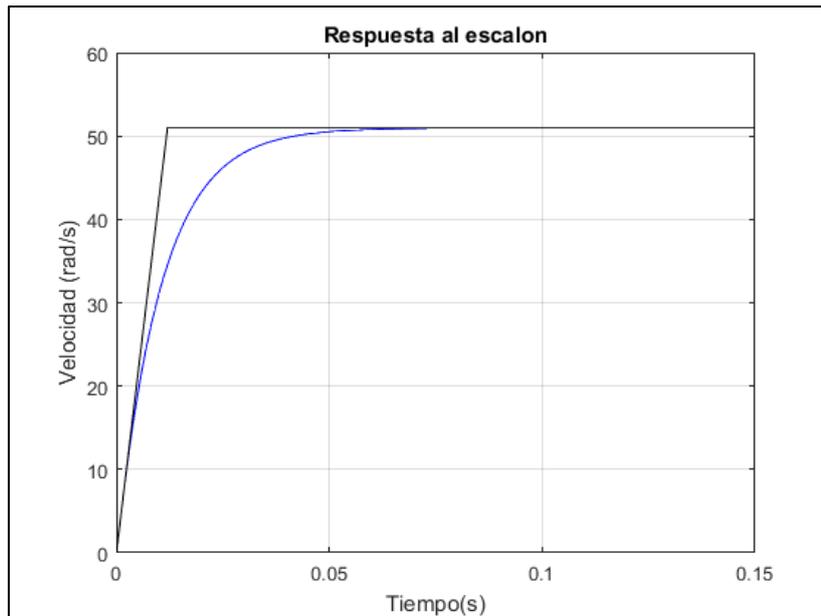


Figura 2.4. Método de la tangente para la identificación del proceso.

Fuente: Autores.

Elaboración: Autores.

La ganancia del sistema está dada por la relación entre el cambio en la salida y el cambio en la entrada, el tiempo muerto es el tiempo entre el instante en el que aplicó el escalón y el punto en que la tangente corte el eje del tiempo y la constante de tiempo es el intervalo de tiempo a partir del instante en que la tangente corta el eje del tiempo y el instante que corta el valor final de la salida [40].

Los parámetros identificados de la planta son:

$$k = 50.95$$

$$T_m = 0.000024366 \text{ s}$$

$$\tau = 0.012 \text{ s}$$

$$a = \frac{kT}{\tau} = 0.10345$$

La función de transferencia obtenida por el método de la recta tangente es:

$$G_{pRT}(s) = e^{-0.000024366} \frac{50.9536}{0.012s + 1} \quad 2.9$$

La segunda técnica es el método de dos puntos de la curva de reacción que se muestra en la Figura 2.5, donde el cálculo de la constante de tiempo y del tiempo muerto son obtenidas a partir de las ecuaciones presentadas en [41], estableciendo dos puntos porcentuales $t_1 = 25\%$ y $t_2 = 75\%$.

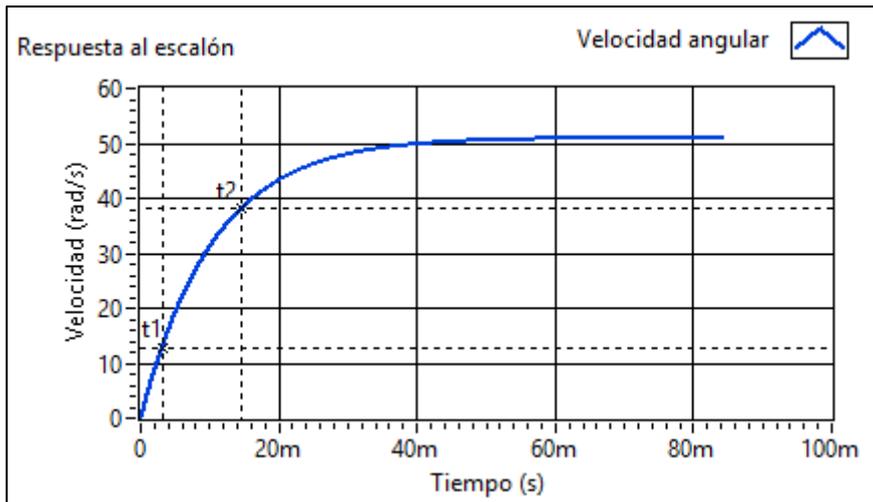


Figura 2.5. Método de dos puntos para la identificación del proceso
 Fuente: Autores.
 Elaboración: Autores.

Los parámetros del τ y T_m , se calculan midiendo el tiempo que tarda el sistema en alcanzar el 25% y 75% del valor final de la respuesta del sistema, se obtiene los valores $t_1 = 0.00313 \text{ s}$ y $t_2 = 0.01465 \text{ s}$, que corresponden a los dos puntos porcentuales (1/4 y 3/4) del cambio de respuesta del sistema ante una entrada de escalón unitario, se tienen los siguientes valores del modelo por el método de Alfaro:

$$k = 50.95$$

$$T_m = 1.262t_1 - 0.262t_2$$

$$T_m = 0.00011176 \text{ s}$$

$$\tau = 0.91(t_2 - t_1)$$

$$\tau = 0.0104832$$

La función de transferencia obtenida por el método de dos puntos es:

$$G_{pDP}(s) = e^{-0.00011176s} \frac{50.9536}{0.0104832s + 1} \quad (2.10)$$

Este segundo de método de identificación será usado para realizar el ajuste de parámetros de tres de los controladores con criterios basados en filtrado derivativo.

Con el objetivo de analizar y validar los modelos matemáticos identificados, se realiza una comparación con el modelo matemático teórico. Esta comparación se efectúa entre los modelos obtenidos de las Ecuaciones (2.7), (2.9) y (2.10), como se observa en la Figura 2.6.

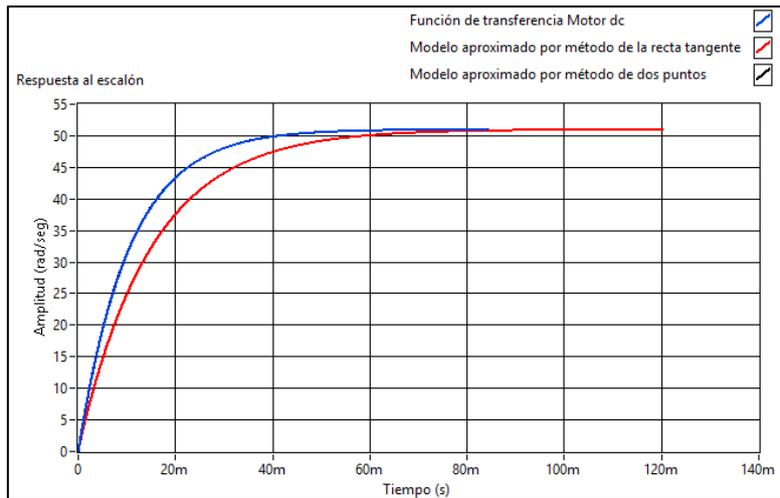


Figura 2.6. Respuesta al escalón de modelos del motor DC.
 Fuente: Autores.
 Elaboración: Autores.

El modelo que se aproxima más a la función de transferencia teórica del motor DC, con los valores de los parámetros obtenidos de [21], es el método de dos puntos propuesto por Alfaro.

2.2.1.2. Ajuste de parámetros de controladores.

Después de identificar y aproximar la dinámica del proceso por medio de los métodos mencionados, se procede a realizar el ajuste de los parámetros (K_p, T_i, T_d) en el caso del PID y los parámetros (K_p, T_i) para el PI, con el objetivo de obtener un desempeño adecuado del sistema de control. Para el desarrollo del presente proyecto se usó la configuración del controlador PID, cuya función de transferencia es:

$$C(s) = K_p + \frac{K_i}{s} + K_d s \quad 2.11$$

En el caso del controlador PI, la función es:

$$C(s) = K_p + \frac{K_i}{s} \quad 2.12$$

Los métodos de sintonización de los controladores PID y PI se clasifican en dos categorías: lazo abierto y lazo cerrado, en este trabajo se trabajó con la técnica de lazo abierto. Existen varios métodos de ajustes que son muy usados en procesos de control, se clasifican en aquellos basados en requerimientos de uso y aquellos basados en criterio de desempeño [42]. En la Tabla 2.2 se muestran los pasos que se realizaron para diseñar controladores PID y PI.

Tabla 2.2: Pasos a realizar para el ajuste de controladores PID y PI.

Paso	Descripción
1	Obtener la función de transferencia por medio del diagrama de bloques del sistema electromecánico.
2	Analizar y validar el modelo matemático del motor DC.
3	Graficar la función de transferencia del motor como respuesta a una entrada de escalón unitario, usar los dos métodos de identificación (recta tangente y dos puntos de la curva del proceso).
4	Determinar los parámetros T , τ , k y a del proceso.
5	Calcular los parámetros del controlador, escoger los métodos de ajuste basados en la curva de la reacción del proceso.
6	Determinar la función de transferencia del controlador PID con la estructura en paralela.
7	Establecer la respuesta del sistema de lazo cerrado de la planta y el controlador con realimentación unitaria.
8	Obtener los datos paramétricos de la respuesta temporal del sistema.
9	Determinar cuáles son los controladores que presentan las mejores respuestas.

Fuente: Autores.

Elaboración: Autores.

Siguiendo los pasos de la Tabla 2.2, se logra obtener la sintonización de los parámetros de los dos controladores, tomando como base [41] se recoge de la literatura de control automático, todos los métodos disponibles de diferentes autores. Además, se aplica algunas variaciones de la formulas del ajuste de los parámetros PID y PI que se acoplen con el sistema.

Las reglas de sintonización de los controladores PID están organizadas de la siguiente manera:

- Reglas de ajuste basadas en la respuesta al escalón o llamada método de curva de reacción.
- Reglas de ajuste basadas en el criterio de mínimo índice de desempeño.
- Reglas de ajuste basadas en filtrado derivativo.
- Reglas de ajuste robustas en estabilidad y desempeño.

Para controladores PID, en las Tablas (2.3), (2.4), (2.5) y (2.6) se muestran las fórmulas para el ajuste de parámetros.

El diagrama de bloques del sistema presentado en la Figura 2.7 sirve para las reglas de ajuste de los métodos basados en la curva de reacción, para las reglas basadas en el criterio de mínimo índice de desempeño, y también para las reglas basadas en la robustez del sistema.

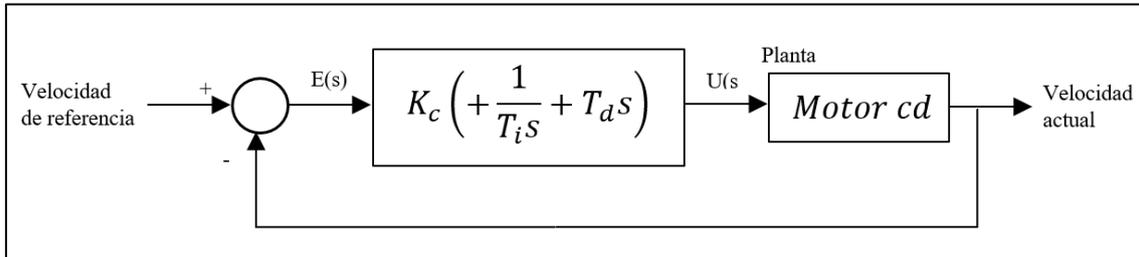


Figura 2.7. Diagrama de bloques del sistema de control PID.
Fuente: [41].
Elaboración: Autores.

Las reglas de ajuste de la Tabla 2.3 corresponden a un controlador que funciona como regulador y cuenta con la siguiente restricción para que pueda ser aplicada al sistema del motor DC:

$$0.1 \leq a \leq 1$$

Como ya se mencionó $a = 0.10345$, es una constante del sistema, por lo que las reglas si son aplicables al modelo del motor.

Tabla 2.3: Reglas de ajustes basadas en la curva reacción.

Método Curva de reacción			
Regla	K_c	T_i	T_d
Ziegler-Nichols	$1.2 \frac{\tau}{T_m}$	$2T_m$	$0.5T_m$
Chien-Hrones-Reswick	$\frac{1.2\tau}{kT_m}$	2τ	0.42τ
Chien-Hrones-Reswick	$\frac{0.95}{a}$	1.4τ	0.47τ
Cohen-Coon	$\frac{1}{k} \left(1.35 \frac{\tau}{T_m} + 0.25 \right)$	$T_m \left(\frac{2.5 \left(\frac{T_m}{\tau} \right) + 0.46 \left(\frac{T_m}{\tau} \right)^2}{1 + 0.61 \left(\frac{T_m}{\tau} \right)} \right)$	$\frac{0.37T_m}{1 + 0.19 \left(\frac{T_m}{\tau} \right)}$

Fuente: [40], [41].
Elaboración: Autores.

Para los métodos de Arrieta Orozco (IAE), Murrill (ITAE) Smith (ITAE) de la Tabla 2.4, que funcionan como reguladores, la restricción es:

$$0.1 \leq a \leq 1$$

Dado el valor de a estas reglas de sintonización sin son aplicables al sistema.

Tabla 2.4: Reglas de ajustes basadas en el criterio de mínimo índice de desempeño.

Mínimo Índice de Desempeño			
Regla	K_c	T_i	T_d
Murrill (IAE)	$\frac{1.435}{k} \left(\frac{\tau}{T_m}\right)^{0.921}$	$\frac{\tau}{0.878} \left(\frac{T_m}{\tau}\right)^{0.749}$	$0.482\tau \left(\frac{T_m}{\tau}\right)^{1.137}$
Arrieta Orozco (IAE)	$\frac{1}{k} \left(0.2068 + 1.1597 \frac{\tau}{T_m}\right)^{1.0158}$	$\tau \left[0.2228 + 1.3009 \left(\frac{T_m}{\tau}\right)^{0.5022}\right]$	$0.3953 \left(\frac{T_m}{\tau}\right)^{0.8469}$
Murrill (ITAE)	$\frac{1.357}{k} \left(\frac{\tau}{T_m}\right)^{0.947}$	$\frac{\tau}{0.842} \left(\frac{\tau}{T_m}\right)^{0.738}$	$0.381\tau \left(\frac{\tau}{T_m}\right)^{0.995}$
Smith (ITAE)	$\frac{0.965}{k} \left(\frac{\tau}{T_m}\right)^{0.855}$	1.26τ	$0.308T_m$

Fuente: [41].

Elaboración: Autores.

Para los controladores robustos, el valor de λ es un parámetro que mejora la robustez del sistema, para el método de Rivera se considera que la restricción de este valor debe estar entre:

$$0.5T_m \leq \lambda \leq 3T_m$$

Se toma un valor constante para λ entre 0.5 y 3:

$$\begin{aligned} &0.8T_m \\ \lambda &\geq 0.8(0.000024366) \\ \lambda &= \mathbf{0.0000194928} \end{aligned}$$

Por lo tanto, el valor de λ si cumple con la restricción establecida para las reglas de ajustes del método de Rivera.

$$0.5T_m \leq 0.0000194928 \leq 3T_m$$

Para el método de Brambilla la restricción es:

$$0.1 \leq a \leq 10$$

Y se toma el valor de λ , de acuerdo a la relación:

$$\begin{aligned} 0.1\tau_m &\leq \lambda \leq 0.5\tau_m \\ \lambda &= \mathbf{0.0036} \end{aligned}$$

Se muestra en la Tabla 2.5 las reglas de ajustes de basados en criterios de robustez para controladores PID.

Tabla 2.5: Reglas de ajustes basadas en el criterio de robustez.

Robusto			
Regla	K_c	T_i	T_d
Rivera, $\lambda = 0.0000194928$	$\frac{\tau + 0.5T_m}{k(\lambda + 0.5T_m)}$	$\tau + 0.5T_m$	$\frac{\tau T_m}{2\tau + T_m}$
Brambilla, $\lambda = 0.0036$	$\frac{\tau + 0.5T_m}{k(\lambda + T_m)}$	$\tau + 0.5T_m$	$\frac{\tau T_m}{2\tau + T_m}$

Fuente: [41].

Elaboración: Autores.

Las reglas basadas en criterio de filtrado derivativo [41], se toma como valor $N = 10$, esto permite eliminar el ruido del sistema, este controlador se basa en la Figura 2.8.

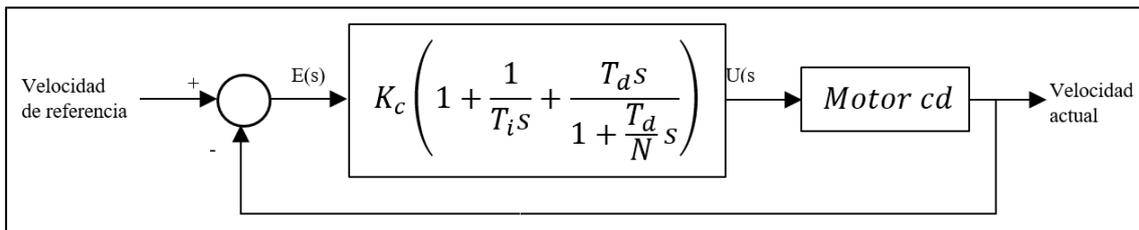


Figura 2.8. Diagrama de bloques del sistema de control con filtrado derivativo.

Fuente: [41].

Elaboración: Autores.

En la Tabla 2.6 se muestran diferentes reglas de ajuste del controlador usando un filtro derivativo, basadas en índices de mínimo desempeño.

Tabla 2.6: Reglas de ajustes basadas en el criterio de filtrado derivativo.

Filtrado Derivativo			
Regla	K_c	T_i	T_d
Alfaro Ruiz (IAE)	$\frac{1}{k} \left[0.2068 + 1.597 \left(\frac{\tau}{T_m} \right)^{1.0158} \right]$	$\tau \left[-0.2228 + 1.3009 \left(\frac{T_m}{\tau} \right)^{0.5022} \right]$	$0.3953 \left(\frac{T_m}{\tau} \right)^{0.8469}$
Arrieta Orozco (IAE)	$\frac{1}{k} \left[0.1050 + 1.2432 \left(\frac{\tau}{T_m} \right)^{0.9946} \right]$	$\tau \left[-0.2512 + 1.3581 \left(\frac{T_m}{\tau} \right)^{0.4796} \right]$	$\tau \left[-0.0003 + 0.3838 \left(\frac{T_m}{\tau} \right)^{0.9479} \right]$
Tavakoli (ITAE)	$\frac{1}{k} \left(\frac{0.8}{\frac{T_m}{\tau} + 0.1} \right)$	$\tau \left(0.3 + \frac{\tau}{T_m} \right)$	$\tau \left(\frac{0.06}{\frac{T_m}{\tau} + 0.04} \right)$
Arrieta Orozco (ITAE)	$\frac{1}{k} \left[0.1749 + 0.8355 \left(\frac{\tau}{T_m} \right)^{0.9462} \right]$	$\tau \left[0.9581 + 0.3987 \left(\frac{T_m}{\tau} \right)^{0.6884} \right]$	$\tau \left[-0.0169 + 0.3126 \left(\frac{T_m}{\tau} \right)^{0.7417} \right]$

Fuente: [41].

Elaboración: Autores.

Las dos primeras reglas de ajustes de controladores de la Tabla 2.6 actúan como reguladores y las siguientes dos reglas actúan como servomecanismos.

En la Tabla 2.7 se presentan algunas reglas de ajuste para controladores PI, para realizar la sintonización de los parámetros de estos controladores se usan los valores obtenidos del método de la recta tangente.

Tabla 2.7: Reglas de ajustes para controladores PI.

Método Curva de reacción		
Regla	K_c	T_i
Ziegler Nichols	$\frac{0.9T_m}{k\tau}$	3.33τ
Cohen-Coon	$\frac{1}{k} \left(0.9 \frac{\tau}{T_m} + 0.83 \right)$	$\tau \left[\frac{3.33 \left(\frac{T_m}{\tau} \right) + 0.31 \left(\frac{T_m}{\tau} \right)^2}{1 + 2.22 \left(\frac{T_m}{\tau} \right)} \right]$
Chien (regulador)	$\frac{0.6\tau}{kT_m}$	$4T_m$
Chien (servo)	$\frac{0.35\tau}{kT_m}$	1.17τ

Fuente: [41].

Elaboración: Autores.

Dentro del diseño de los controladores presentados en este trabajo, existen dos modos de operación del sistema de control, el primer modo es como servomecanismo en el cual la perturbación del sistema es el mismo valor deseado, y el segundo modo es el regulador en donde se requiere minimizar el efecto de las perturbaciones sobre el sistema. En la sección 2.3 se mostrará los resultados de la sintonización y simulación de los controladores PID y PI basados en los métodos expuestos en la presente sección.

2.2.2. Controladores fraccionarios FOPID y FOPI.

En la presente sección se expondrán las metodologías y herramientas para el diseño de controladores FOPID, así como también las técnicas y procedimientos para la implementación de los controladores de este tipo.

Basándose en el análisis de la respuesta en frecuencia de lazo abierto, tenemos que la función de transferencia inicia a partir de la Ecuación (2.13) [43].

$$F(j\omega) = C(j\omega)G_p(j\omega) \quad (2.13)$$

Donde:

$F(j\omega)$: Es la función de transferencia en lazo abierto.

$C(j\omega)$: Es la función del controlador FOPID.

$G_p(j\omega)$: Es la función de la Planta.

Para lograr el ajuste del PID y PI de orden fraccionario se lo ha hecho mediante minimización, la cual está basada en dos métodos de optimización que se mencionan a continuación:

- Método de optimización sin restricción.
- Método de optimización con restricción.

Método de optimización sin restricción: Algunas veces se necesita encontrar el mínimo (números reales) de una función determinada en todo el espacio n-dimensional. Para estos problemas se encuentran métodos de optimización llamados de búsqueda directa, tal como el método de Nelder-Mead [44]. También se lo denomina método simplex de Nelder-Mead, el cual se encarga de resolver problemas de optimización sin restricciones de la forma como se expresa la Ecuación (2.14):

$$\min_x F(x); \quad x \in \mathbb{R}^n \quad (2.14)$$

Este es un método que realiza una búsqueda directa, por lo tanto, es muy adecuado para optimizar funciones en las cuales sus derivadas son desconocidas o inexistentes. En [31] se describe este método.

Primeramente, un simplex inicial se construye fijando n+1 vértices junto con los correspondientes valores de $F(x_k)$. La k-ésima iteración consiste en las siguientes etapas [31]:

1) Ordenar

Ordenar los n+1 vértices del simplex como en la Ecuación (2.15).

$$F(x_1) \leq F(x_2) \dots \leq F(x_{n+1}) \quad (2.15)$$

2) Reflejar

Calcular el punto de reflexión.

$$x_r = \bar{x} + \rho(\bar{x} - x_{n+1}) \quad (2.16)$$

$$\bar{x} = \sum_{i=1}^n x_i/n \quad (2.17)$$

Donde \bar{x} es el mejor centroide de los n mejores vértices.

Evaluar $F_r = F(x_r)$. Si $F_1 \leq F_r < F_n$, establece que $x_{n+1} = x_r$ es un nuevo vértice del simplex, elimina el peor vértice y termina la iteración.

3) Expandir

Si $F_r < F_1$, calcular el punto de expansión.

$$x_e = \bar{x} + \chi(x_r - \bar{x}), \quad (2.18)$$

Evaluar $F_e = F(x_e)$. Si $F_e < F_r$, establece que $x_{n+1} = x_e$, se termina la iteración. De otra manera, establece que $x_{n+1} = x_r$, elimina el peor vértice y termina la iteración.

4) Contraer

Si $F_r \geq F_n$ realiza una contracción entre \bar{x} y el mejor entre x_{n+1} y x_r :

a) Contracción externa

Si $F_n \leq F_r < F_{n+1}$, calcula

$$x_c = \bar{x} + \gamma(x_r - \bar{x}) \quad (2.19)$$

Y evalúa $F_c = F(x_c)$. Si $F_c \leq F_r$, establece $x_{n+1} = x_c$ y termina la operación. De otra manera se dirige al paso 5.

b) Contracción interna

Si $F_r \geq F_{n+1}$, realiza una contracción interna, calcula

$$x'_c = \bar{x} + \gamma(\bar{x} - x_{n+1}) \quad (2.20)$$

Y evalúa $F'_c = F(x'_c)$. Si $F'_c \leq F_{n+1}$ establece que $x_{n+1} = x'_c$, se termina la iteración. De otra manera ir al paso 5.

5) Encoger

Define n nuevos vértices de

$$x_i = x_1 + \sigma(x_i - x_1), \quad i = 2, \dots, n+1, \quad (2.21)$$

Y evalúa F en estos puntos [31].

Método de optimización con restricción: Este algoritmo de optimización trata de que el sistema controlado cumpla con los siguientes criterios de robustez:

a) Margen de fase φ_m

$$\arg(F(j\omega_c)) = -\pi + \varphi_m \quad 2.22$$

b) Ganancia en la frecuencia de cruce ω_c

$$|F(j\omega_c)| = 0 \text{ dB} \quad 2.23$$

c) Robustez ante variaciones de ganancia de la planta

$$\left. \frac{d \arg(F(j\omega))}{d\omega} \right|_{\omega=\omega_c} = 0 \quad 2.24$$

d) Rechazo de ruido de alta frecuencia

$$\left| T(j\omega) = \frac{F(j\omega)}{1+F(j\omega)} \right|_{dB} \leq A \text{ dB}, \forall \omega \geq \omega_t \text{ rad/s} \quad 2.25$$

e) Rechazo de las perturbaciones en la salida

$$\left| S(j\omega) = \frac{1}{1+F(j\omega)} \right|_{dB} \leq B \text{ dB}, \forall \omega \leq \omega_s \text{ rad/s} \quad 2.26$$

Donde S es la función de sensibilidad.

De acuerdo a la Ecuación (2.13) se pueden cumplir los cinco criterios de robustez mencionados anteriormente, ya que tiene cinco parámetros para sintonizar. En otras palabras, el diseño del controlador FOPID se basa en resolver el sistema de cinco ecuaciones no lineales correspondientes a los criterios de robustez y los cinco parámetros del FOPID no conocidos K_p , K_i , K_d , λ , y μ .

Sin embargo, la complejidad de este conjunto de ecuaciones no lineales es muy significativa cuando se necesita encontrar la mejor solución que se basa en la optimización global que es la tarea de encontrar el conjunto absolutamente mejor de condiciones admisibles para lograr un objetivo bajo ciertas limitaciones. Gracias a los métodos actuales algunos problemas de optimización global han sido resueltos, y existen disponibles una serie de paquetes de software que los solucionan de forma fiable en su gran mayoría [43].

Sin embargo, encontrar el mínimo global que sería lo ideal es un problema difícil (muy dependiente de las condiciones iniciales) comparado con encontrar un mínimo local, a pesar de la importancia del mínimo global en muchas aplicaciones prácticas no es esencial, ya que cualquier punto factible suficientemente bueno es útil y usualmente una mejora sobre lo que está disponible sin optimización [43].

Teniendo en cuenta el análisis anterior y considerando que el conjunto de funciones a minimizar es continuo y sólo puede presentar un mínimo en la región factible, cualquiera de los métodos de optimización disponibles podría ser eficaz. Por esta razón y considerando que Matlab es una herramienta apropiada para el análisis y diseño de sistemas de control, en [43] los autores utilizaron el toolbox de optimización para llegar a la mejor solución, dicho toolbox presenta la función FMINCON que se encarga de encontrar el mínimo de la función objetivo bajo las restricciones de funciones propuestas.

2.2.2.1. Herramientas computacionales para control fraccionario.

Entre las principales herramientas computacionales para el modelado, análisis y diseño de los controladores PID de orden fraccionario, basadas en MATLAB especialmente para aplicaciones de control se muestran en la Tabla 2.8:

Tabla 2.8: Herramientas para el modelado de controladores fraccionarios.

Herramienta	Aplicaciones/Módulos	GUI	Comandos para interfaz	Sistemas con retardo
@fotf	Análisis y control de sistemas fraccionarios (orientado a objetos). Disponibile en [45], [43].	No	-	Sí
CRONE Toolbox	- Crone CSD: diseño de sistemas de control fraccionario (tipo CRONE). - ooCrone: análisis de sistemas fraccionarios (orientado a objetos). Disponibile en [46].	Sí	crone_control	No
		No		No
FOMCON	- Análisis de sistemas fraccionarios. - Diseño de PID fraccionarios. - Optimización de PID fraccionarios. - Diseño de PID clásicos. - Identificación de sistemas fraccionarios (dominio del tiempo y frecuencia). - Implementación de controladores fraccionarios (dominio de la frecuencia). Disponibile en [31].	Sí	fomcon	Sí
		Sí	fpid	
		Sí	fpid_optim	
		Sí	iopid_tune	
		Sí	fotfid/ fotfrid	
NINTEGER	- Análisis de sistemas fraccionarios (dominio de la frecuencia). - Diseño de controladores PID fraccionarios y tipo CRONE. - Identificación de sistemas fraccionarios (dominio de la frecuencia). Disponibile en [34].	No	ninteger	Sí
		Sí		
		No		

Fuente: [47].

Elaboración: Autores.

De las herramientas de MATLAB mencionadas en la Tabla 2.8, las que se escogieron para la sintonización del PID de orden fraccionario, fueron FOMCON (Fractional Order Modeling and Control) e NINTEGER. La primera por disponer de interfaz de usuario de fácil uso la cual la convierte en la herramienta más utilizada en su tipo, además es aplicable para sistemas con retardo como es el caso de nuestra planta, y finalmente su completa documentación [31] es de utilidad para la aplicación de la herramienta en nuestro proyecto. La segunda herramienta NINTEGER se la seleccionó debido a que permite convertir una función de transferencia de orden fraccionaria a una de orden entera, y de esta a una función de transferencia discreta para la implementación en hardware. Las otras herramientas @fotf y CRONE toolbox, se descartaron porque poseen aplicaciones limitadas respecto a las mencionadas anteriormente como se puede observar en la Tabla 2.8.

2.3. Simulación de los controladores IOPID y FOPID.

Con el fin de observar y analizar las respuestas del sistema de control de velocidad para el motor DC, se realiza simulaciones de todos los controladores de orden entero y fraccionario, para implementar estas simulaciones se lleva a cabo mediante el software LabVIEW.

2.3.1. Simulación de los controladores PID.

Se desarrolló 14 controladores PID, en la Figura 2.9 se muestra la distribución.

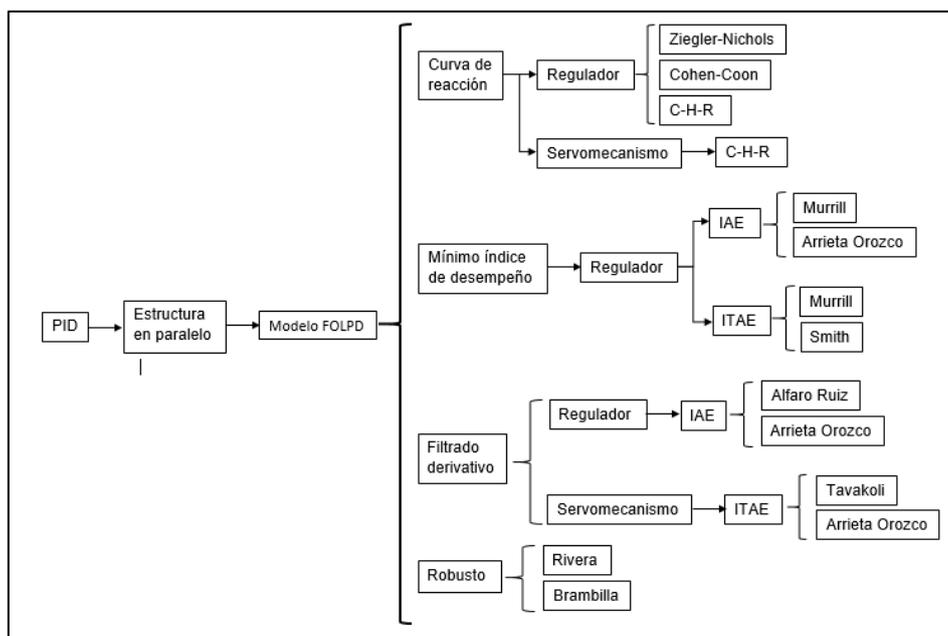


Figura 2.9. Distribución de controladores PID.

Fuente: Autores.

Elaboración: Autores.

2.3.1.1. Controladores basados en la curva de reacción.

Controlador Ziegler y Nichols

Estos autores fueron los primeros en desarrollar de forma empírica un procedimiento para ajustar los parámetros, las características del método son:

- Identificación del modelo: Método de la recta tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Decaimiento de un cuarto.

Función de transferencia del controlador:

$$C(s) = 590.9874 + \frac{12127296.2324}{s} + 0.0071999 s$$

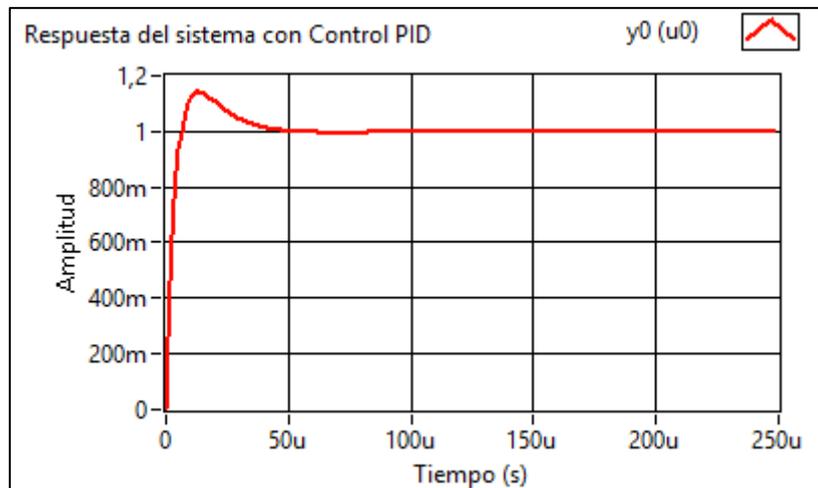


Figura 2.10. Respuesta del controlador Ziegler-Nichols.
Fuente: Autores.
Elaboración: Autores.

Controlador Chien-Hrones-Reswick

Las características del método son:

- Identificación del modelo: Método de la recta tangente.
- Funcionamiento del controlador: Servomecanismo.
- Criterio de desempeño: Respuesta rápida, sin sobrelongación o con sobrelongación máximo de 20%.

Función de transferencia del controlador:

$$C(s) = 9.1828 + \frac{546.5952}{s} + 0.005179 s$$

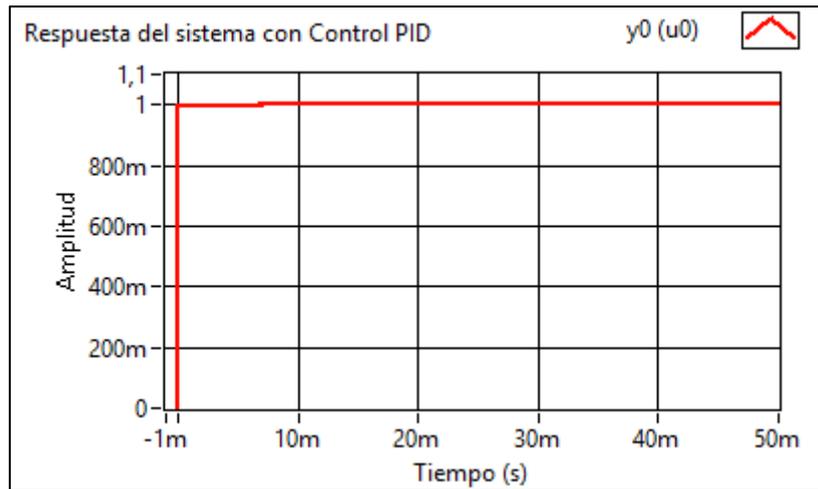


Figura 2.11. Respuesta del controlador Chien-Hrones-Reswick, servomecanismo.
Fuente: Autores
Elaboración: Autores.

Controlador Chien-Hrones-Reswick

Las características del método son:

- Identificación del modelo: Método de la recta tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Respuesta rápida, sin sobreelongación o con sobreelongación máximo de 20%.

Función de transferencia del controlador:

$$C(s) = 11.5993 + \frac{483.304}{s} + 0.05846 s$$

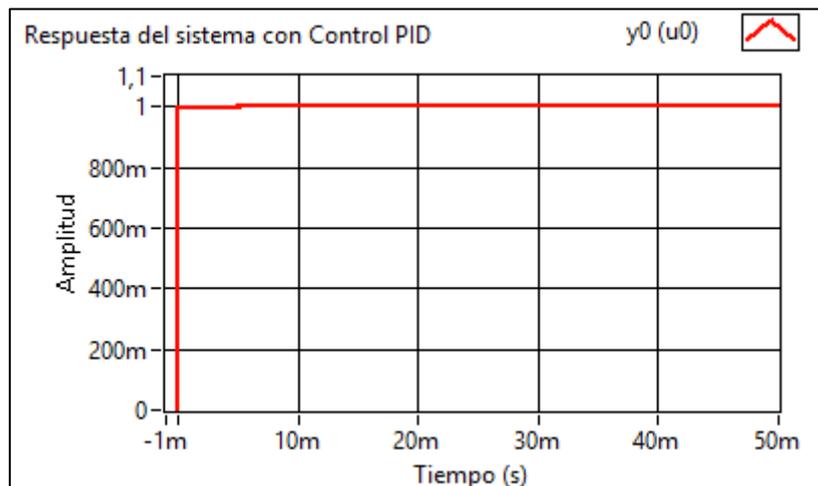


Figura 2.12. Respuesta del controlador Chien-Hrones-Reswick, regulador.
Fuente: Autores
Elaboración: Autores.

Controlador Cohen Coon

El método de Cohen-Coon considera al proceso como auto-regulado y se modifican las ecuaciones del método de ajuste, las principales características del controlador son:

- Identificación de planta: Método de la tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Decaimiento de un cuarto.

Función de transferencia del controlador:

$$C(s) = 13.0778 + \frac{215.769}{s} + 0.000117629 s$$

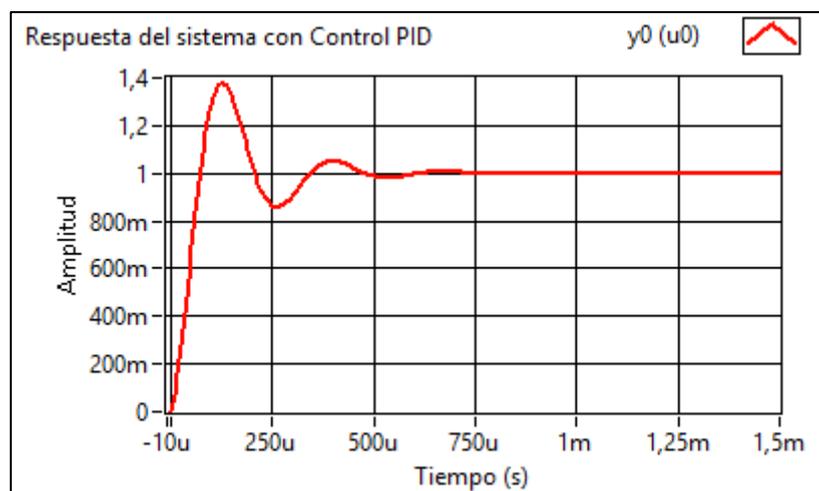


Figura 2.13. Respuesta del controlador Cohen-Coon.
Fuente: Autores
Elaboración: Autores.

2.3.1.2. Controladores basados en el mínimo índice de desempeño.

Controlador Murrill (IAE)

Controlador basado en el establecimiento del índice de desempeño de la señal de error (Integral de Error absoluto). Características del controlador:

- Identificación de la planta: Método de la tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Mínimo IAE

Función de transferencia del controlador:

$$C(s) = 8.49973 + \frac{64617.10815}{s} + 0.000042$$

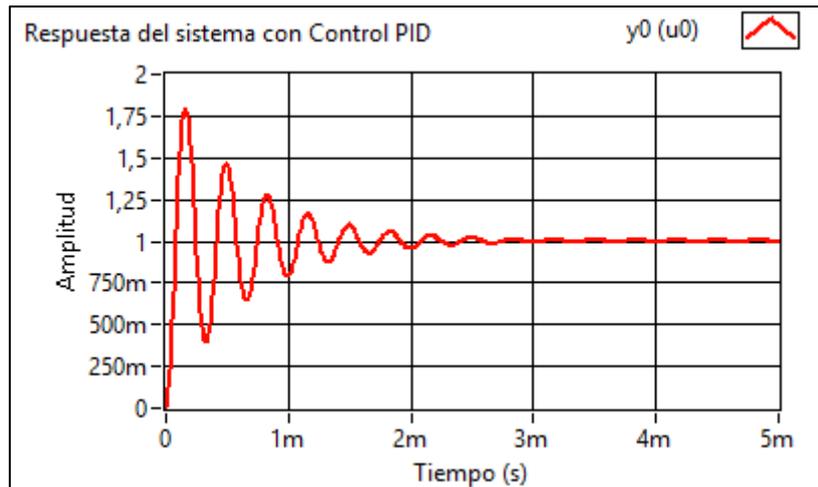


Figura 2.14. Respuesta del controlador Murrill-IAE.
 Fuente: Autores.
 Elaboración: Autores.

Controlador Arrieta Orozco

Características del controlador:

- Identificación de la planta: Método de la tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Mínimo IAE

Función de transferencia del controlador:

$$C(s) = 12.3674 + \frac{6247.17}{s} + 0.000307747 s$$

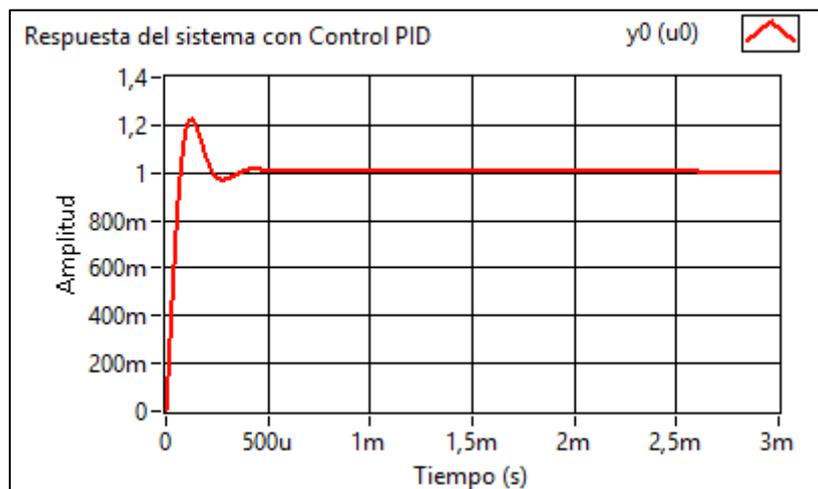


Figura 2.15. Respuesta del controlador Arrieta Orozco-IAE.
 Fuente: Autores.
 Elaboración: Autores.

Controlador Murrill (ITAE)

Controlador Basado en el establecimiento del índice de desempeño de la señal de error (Integral del Tiempo por el Error absoluto). Características del controlador:

- Identificación de la planta: Método de la tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Mínimo ITAE

Función de transferencia del controlador:

$$C(s) = 9.44355 + \frac{6.82776}{s} + 20.6146 s$$

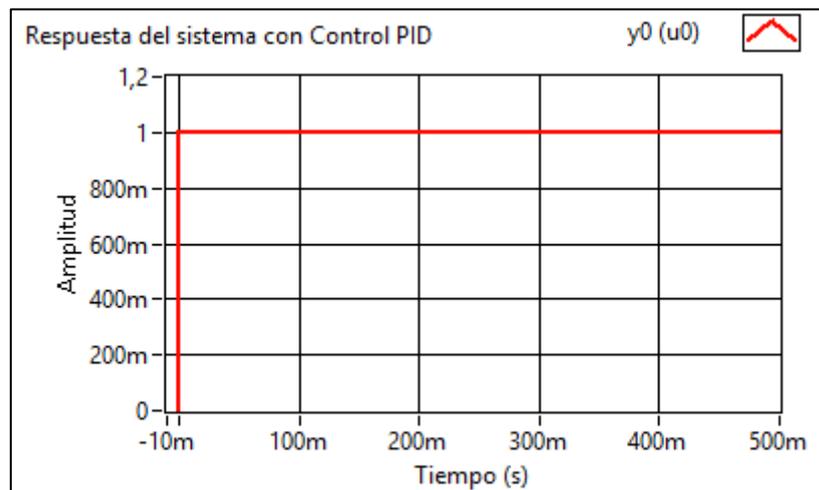


Figura 2.16. Respuesta del controlador Murrill- ITAE.

Fuente: Autores.

Elaboración: Autores.

Controlador Smith

Controlador Basado en el establecimiento del índice de desempeño basado en la señal de error (Integral del Tiempo por el Error absoluto). Características del controlador

- Identificación de la planta: Cualquier método de identificación (recta tangente).
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Mínimo ITAE.

Función de transferencia del controlador:

$$C(s) = 3.79649 + \frac{251.0906}{s} + 0.0000284916s$$

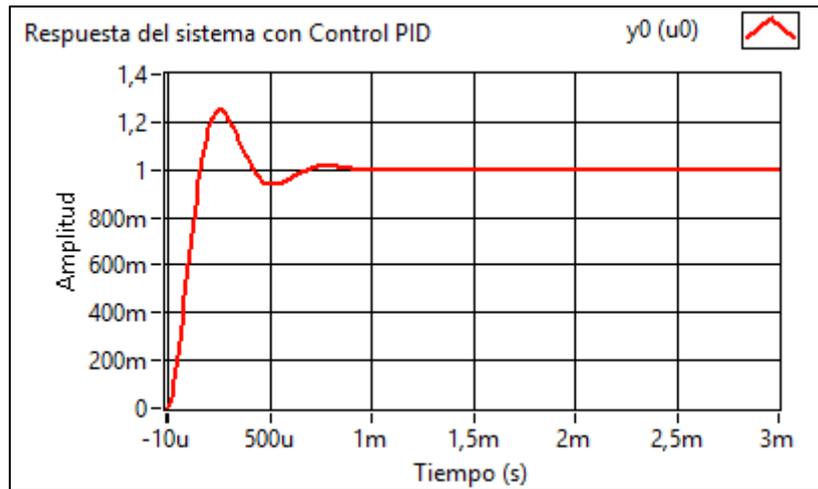


Figura 2.17. Respuesta del controlador Smith-ITAE.
Fuente: Autores.
Elaboración: Autores.

2.3.1.3. Controladores basados en la robustez del sistema.

Usando las reglas de sintonía parametrizadas se mejora la estabilidad del sistema en lazo cerrado, obteniendo así los parámetros del controlador PID con un grado de robustez [48].

Controlador Rivera

Características del controlador:

- Identificación de la planta: Cualquier método de identificación (recta tangente).
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Robusto.

Función de transferencia del controlador:

$$C(s) = 7.44593 + \frac{619.625}{s} + 0.000905849 s$$

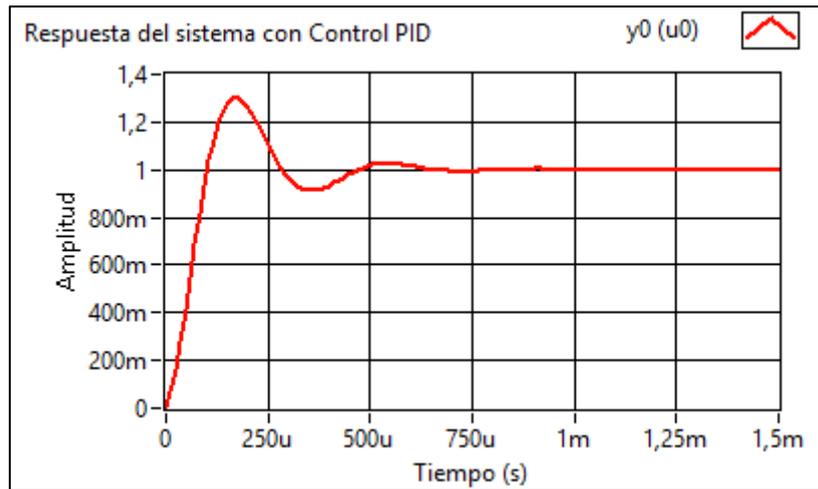


Figura 2.18. Respuesta del controlador Rivera.
 Fuente: Autores.
 Elaboración: Autores.

Controlador Brambilla

Características del controlador:

- Identificación de la planta: Cualquier método de identificación (recta tangente).
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Robusto.

Función de transferencia del controlador:

$$C(s) = 0.06505 + \frac{5.4154}{s} + 0.00000079169 s$$

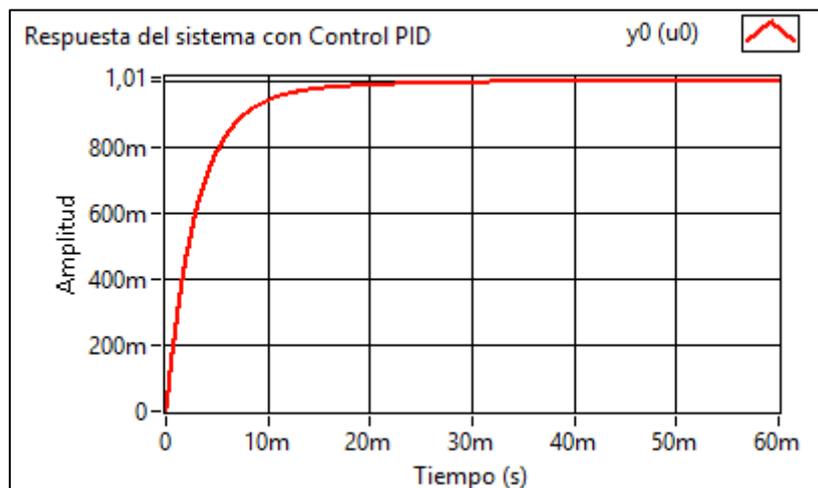


Figura 2.19. Respuesta del controlador Brambilla.
 Fuente: Autores.
 Elaboración: Autores.

2.3.1.4. Controladores basados en filtrado derivativo.

Método de Alfaro Ruiz

Características del controlador:

- Identificación de la planta: Método de dos puntos de la curva de reacción.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Índice de mínimo desempeño (IAE).

Función de transferencia es:

$$C(s) = 3.1629 + \frac{3359.4264}{s} + 0.0267 s$$

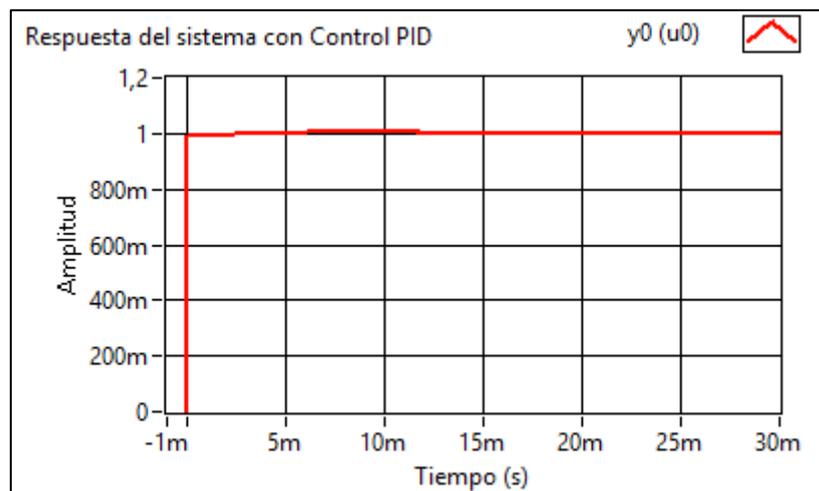


Figura 2.20. Respuesta del controlador con filtrado derivativo de Alfaro Ruiz.
Fuente: Autores.
Elaboración: Autores.

Método de Arrieta Orozco

Características del controlador:

- Identificación de la planta: Método de dos puntos de la curva de reacción.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Índice de mínimo desempeño (IAE).

Función de transferencia es:

$$C(s) = 2.2354 + \frac{2191.5686}{s} + 0.0001143 s$$

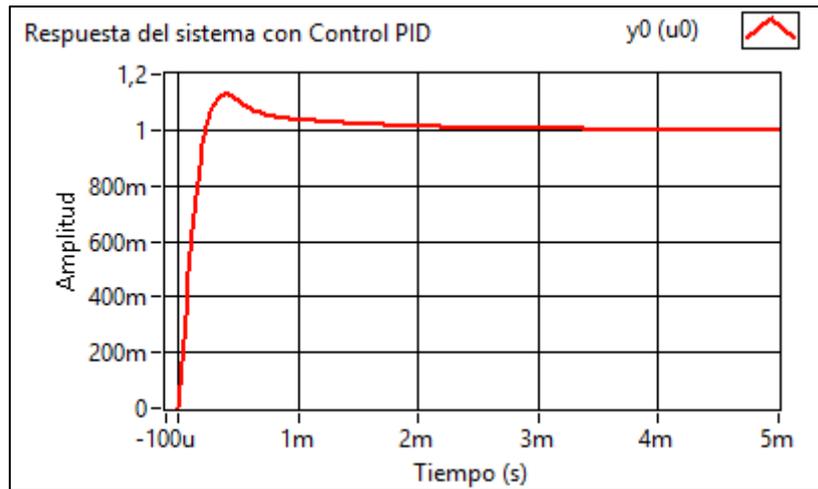


Figura 2.21. Respuesta del controlador con filtrado derivativo de Arrieta Orozco.
 Fuente: Autores.
 Elaboración: Autores.

Método de Tavakoli

Características del controlador:

- Identificación de la planta: Método de la tangente.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Índice de mínimo desempeño (ITAE).

Función de transferencia es:

$$C(s) = 0.15389 + \frac{12.8167}{s} + 0.0000535278 s$$

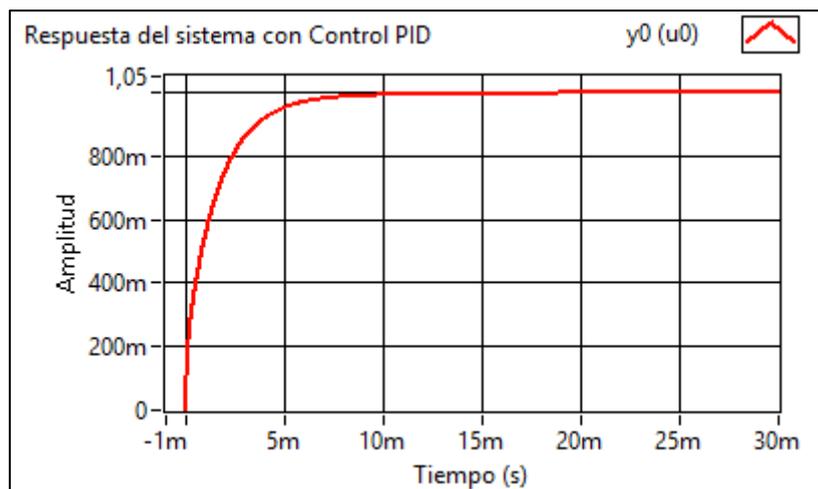


Figura 2.22. Respuesta del controlador con filtrado derivativo de Tavakoli.
 Fuente: Autores.
 Elaboración: Autores.

Método de Arrieta Orozco

Características del controlador:

- Identificación de la planta: Método de dos puntos de la curva de reacción.
- Funcionamiento del controlador: Regulador.
- Criterio de desempeño: Índice de mínimo desempeño (ITAE).

Función de transferencia es:

$$C(s) = 1.2082 + \frac{118.2192}{s} + 0.00007239 s$$

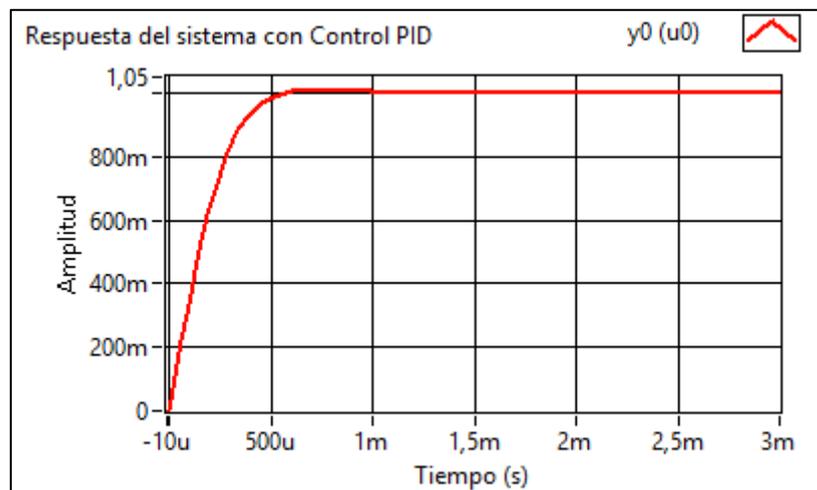


Figura 2.23. Respuesta del controlador con filtrado derivativo de Arrieta Orozco.
Fuente: Autores.
Elaboración: Autores.

2.3.2. Simulación de los controladores PI.

Usando la misma metodología para controladores PID, se usa el método de la recta tangente para realizar el ajuste los parámetros de controladores PI, en esta sección se muestra el resultado de las simulaciones de los controladores PI en el software LabVIEW, usando el toolkit de "Control & Simulation".

Método de Ziegler-Nichols

Función de transferencia es:

$$C(s) = 8.6996 + \frac{107219.514}{s}$$

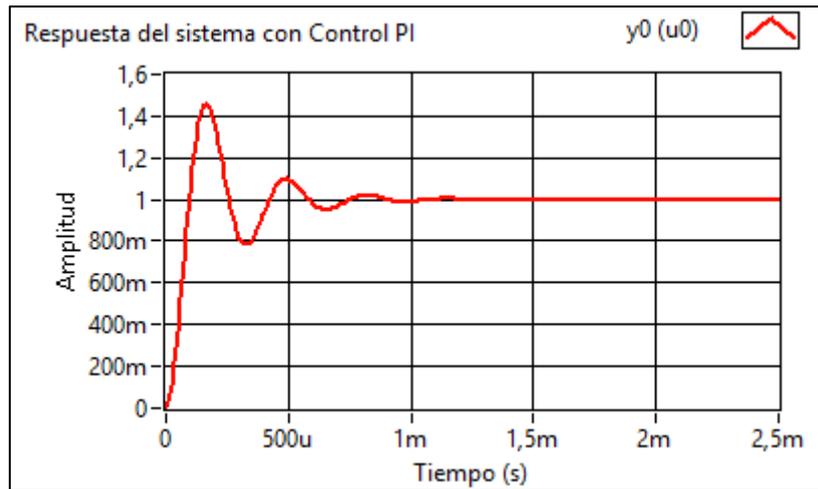


Figura 2.24. Respuesta del controlador PI de Ziegler-Nichols
 Fuente: Autores.
 Elaboración: Autores.

Método de Cohen-Coon

Función de transferencia es:

$$C(s) = 443.3236 + \frac{5487737.484}{s}$$

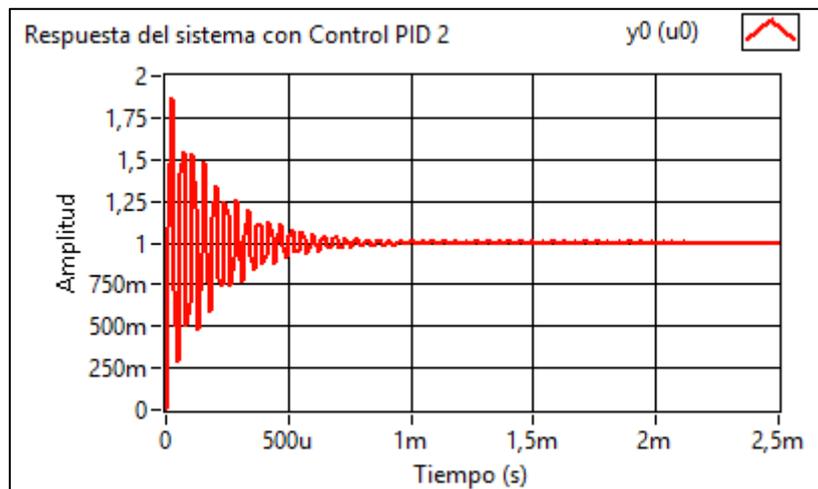


Figura 2.25. Respuesta del controlador PI de Cohen-Coon.
 Fuente: Autores.
 Elaboración: Autores.

Método de Chien (Regulador)

Función de transferencia es:

$$C(s) = 49.6628 + \frac{509550.193}{s}$$

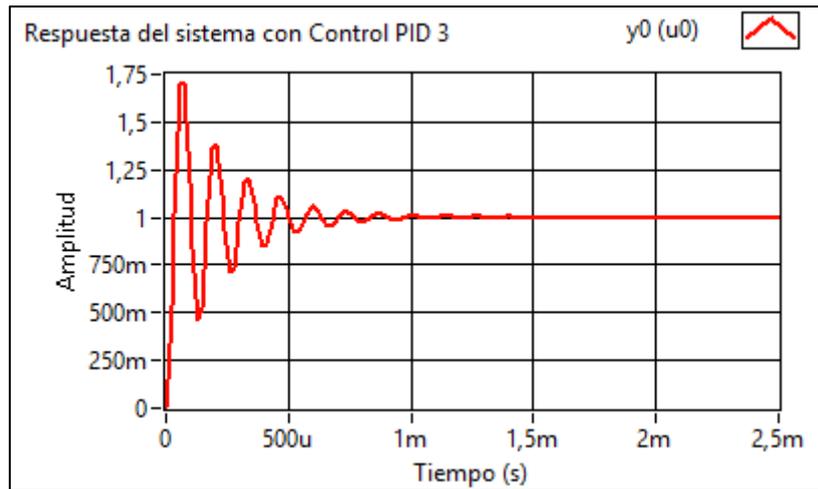


Figura 2.26. Respuesta del controlador PI de Chien como regulador.
 Fuente: Autores.
 Elaboración: Propia.

Método de Chien (Servomecanismo)

Función de transferencia es:

$$C(s) = 3.3831 + \frac{240.9648}{s}$$

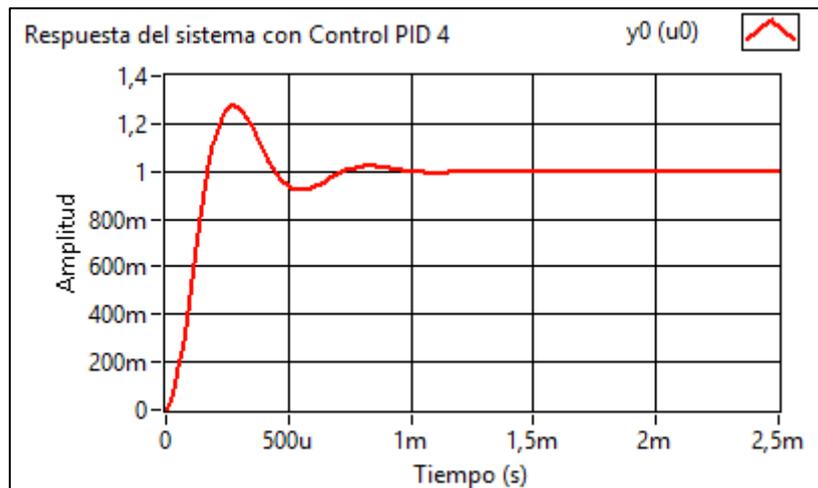


Figura 2.27. Respuesta del controlador PI de Chien como servomecanismo.
 Fuente: Autores.
 Elaboración: Propia.

En la Tabla 2.9 se muestran los resultados de los parámetros temporales de los controladores de orden entero realizados en simulación en el programa de LabVIEW.

Tabla 2.9. Parámetros de la respuesta temporal de controladores de orden entero.

	Método de ajuste	Sobrelongación (%)	Tiempo de establecimiento (s)
PID	Ziegler-Nichols, Regulador	14.0044	0.000041
	C-H-R, Servo	0.0044	0.000002
	C-H-R, Regulador	0.0000	0.000002
	Cohen-Coon, Regulador	37.6739	0.000587
	Murrill, IAE	79.6840	0.002853
	Arrieta Orozco, IAE	22.4188	0.000470
	Murrill, ITAE	0.0254	0.000139
	Smith, ITAE	24.8117	0.000835
	Alfaro Ruiz, Filtrado, IAE	2.0976	0.320229
	Arrieta Orozco, Filtrado, IAE	2.8039	0.029618
	Tavakoli, Filtrado, ITAE	0.0000	0.009443
	Arrieta Orozco, Filtrado, ITAE	0.0000	0.074170
	Rivera, Robusto	29.8288	0.000610
	Brambilla, Robusto	0.0000	0.023538
PI	Ziegler-Nichols	46.0200	0.001020
	Cohen-Coon	87.3600	0.000960
	Chien (regulador)	70.8700	0.000948
	Chien (servo)	27.6700	0.000926

Fuente: Autores.

Elaboración: Autores.

En base a los resultados de la simulación en lazo cerrado de los controladores (Tabla 2.9), se observa que los mejores controladores del sistema son los controladores PID de los métodos de C-H-R (Regulador) y Murrill (ITAE); se escogió estos dos controladores ya que presentan mejores respuestas en los parámetros de sobrelongación y tiempo de establecimiento.

2.3.3. Simulación de los controladores FOPID.

Para la sintonización del PID de orden fraccionario se siguieron los siguientes pasos en Matlab:

Paso 1. Mediante los parámetros del fabricante del motor Maxon se logró obtener la función de transferencia del motor, a la cual se le aplicó un escalón y de esta respuesta al escalón se obtuvo las variables de entrada y salida como muestra la Figura 2.28.

```

1 - entrada= ones(1,101);
2 - Va = 1; % voltaje aplicado
3 - J = 0.00000136; %Inercia mecánica
4 - B = 0.00003781; %Fricción viscosa de los rodamientos
5 - Kt = 0.0139; %Constante mecánica del motor
6 - Ra = 2.12; %Resistencia del circuito de armadura
7 - La = 0.000227; %Inductancia del circuito de armadura
8 - Kb = 0.013859; % Constante de fuerza contraelectromotriz
9 - num = Kt;
10 - den = [(J*La) ((J*Ra)+(La*B)) ((B*Ra)+(Kt*Kb))];
11 - M = tf(num, den) % Respuesta al escalon
12 - pp = pole(M)
13 - dt = 0.001;
14 - t = 0:dt:0.1;
15 - y = step(Va*M,t)';
16 - dy = diff(y)/dt;
17 - [m,p] = max(dy);
18 - d2y = diff(dy)/dt;
19 - yi = y(p);
20 - ti = t(p);
21 - L = ti-yi/m; %Retardo
22 - Tao = (y(end)-yi)/m+ti-L; %Constante de tiempo
23 - %-----
24 - plot(t,y,'b',[0 L L+Tao t(end)],[0 0 y(end) y(end)], 'k');
25 - title('Respuesta al escalon')
26 - xlabel('Tiempo(s)');
27 - ylabel('Velocidad (rad/s)');
28 - legend('Exacta', 'Aproximacion lineal')
29 - grid on
30 - [Gm, Pm, Wg, Wp] = margin(M)

```

Figura 2.28. Código en Matlab de la FT del motor.

Fuente: Autores.

Elaboración: Autores.

Donde “entrada” es la variable de entrada e “y” es la variable de salida.

Paso 2. Identificamos la planta mediante un sistema de primer orden con retardo, para esto se coloca la función “ident” en el workspace, posteriormente se abre la interfaz como la Figura 2.29.

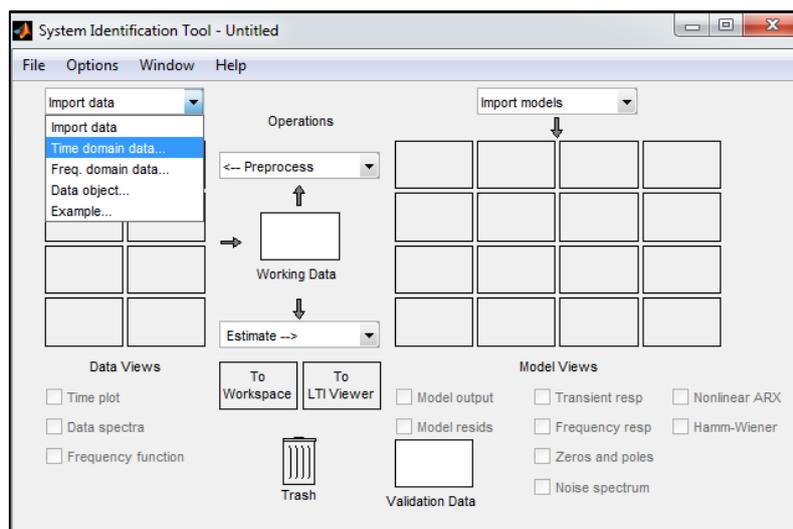


Figura 2.29. Identificación de la planta.

Fuente: Autores.

Elaboración: Autores.

Luego se selecciona “Time domain data” donde se coloca las variables de entrada y salida como se muestra en la Figura 2.30, posteriormente se selecciona la opción “import”, y aceptamos los mensajes que aparecen.

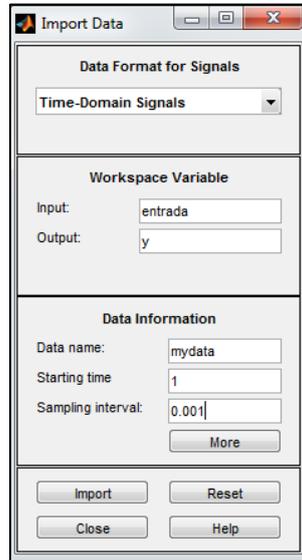


Figura 2.30. Importación de datos para la identificación.

Fuente: Autores.

Elaboración: Autores.

Posteriormente de manera automática se redirige a la interfaz de la Figura 2.29, en la que podemos seleccionar la opción “Time plot” que nos muestra la gráfica de la entrada y de la salida como en la Figura 2.31.

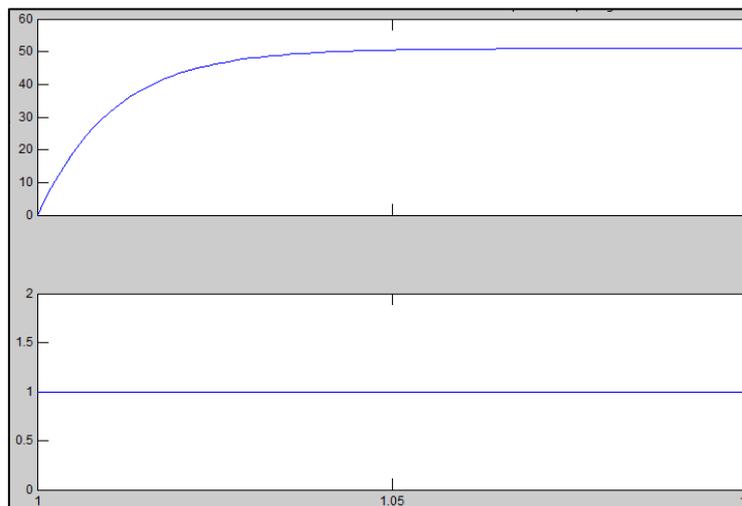


Figura 2.31. Respuesta al escalón a la planta.

Fuente: Autores.

Elaboración: Autores

Luego en la interfaz de la Figura 2.29 en la opción de “estimate” seleccionamos “Process Models” la cual nos abre la ventana que se muestra la Figura 2.32.

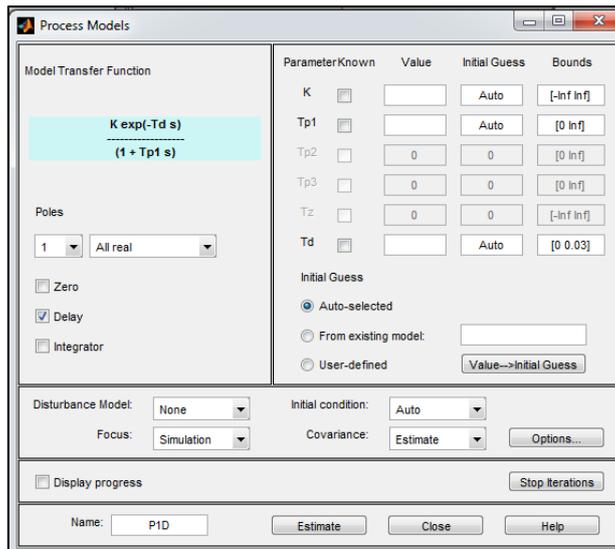


Figura 2.32. Función de transferencia obtenida de la identificación.
Fuente: Autores.
Elaboración: Autores

Seguidamente seleccionamos la opción “estimate” que nos dará los parámetros de la identificación como muestra en la Figura 2.33.

Parameter Known	Value	Initial Guess	Bounds
K	50.9536	Auto	[-Inf Inf]
Tp1	0.010493	Auto	[0 10000]
Tp2	0	0	[0 Inf]
Tp3	0	0	[0 Inf]
Tz	0	0	[-Inf Inf]
Td	0.000108	Auto	[0 0.03]

Figura 2.33. Parámetros obtenidos de la identificación.
Fuente: Autores.
Elaboración: Autores

Estos parámetros son los de un sistema de primer orden más retardo como muestra la Ecuación 2.27.

$$\frac{50.9536}{0.010493s + 1} e^{-0.000108s} \quad 2.27$$

Para comprobar la fidelidad de la identificación, en la interfaz de la Figura 2.29 seleccionamos la opción “model output” y obtenemos que la señal identificada tiene el 100% de igualdad que la del motor, esto se puede apreciar en la Figura 2.34.

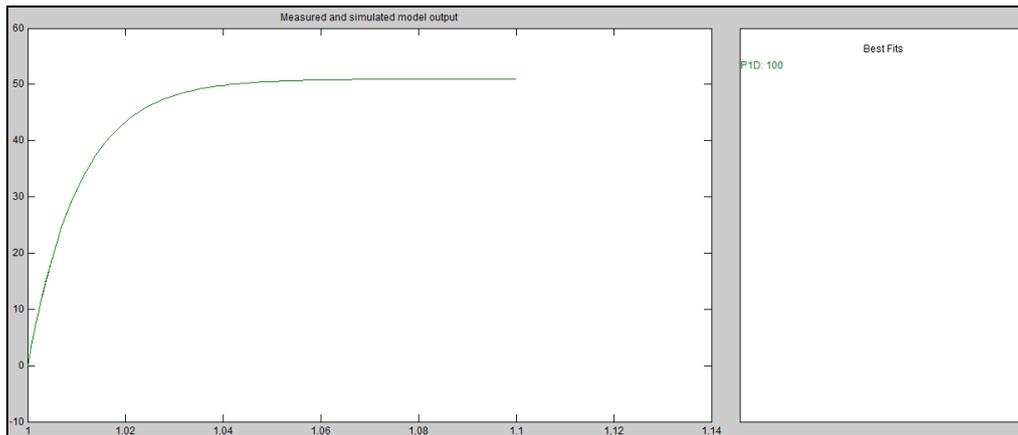


Figura 2.34. Gráfica del modelo obtenido a partir de la identificación.

Fuente: Autores.

Elaboración: Autores

Paso 3. Usando la función “iopid_tune” se abre la interfaz de la Figura 2.35, colocamos los parámetros que se muestran en la Figura 2.33, luego en la opción “Method” seleccionamos el método PID clásico a sintonizar, luego seleccionamos la opción compute y obtenemos los valores como indica la Figura 2.35.

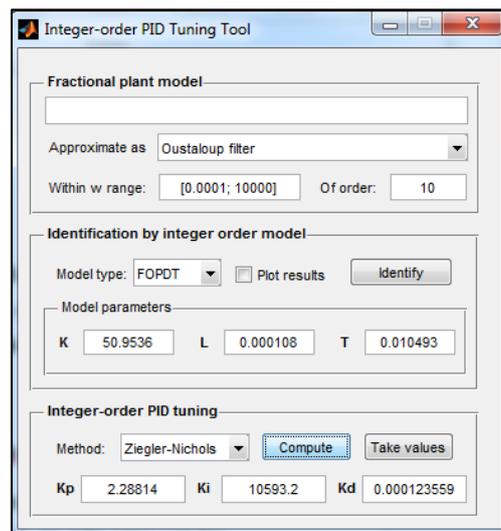


Figura 2.35. Sintonización PID convencional.

Fuente: Autores.

Elaboración: Autores

Paso 4. Para la sintonización del PID fraccionario se usa la función “fpid_optim”, la cual abre la interfaz de la Figura 2.35. En la opción “LTI system” se coloca la variable que contiene la función de transferencia que se expresa en la Ecuación (2.27).

En la opción “Fractional PID controller parameters”, como se muestra en la Figura 2.36, se coloca como parámetros iniciales las constantes calculadas del PID clásico de la Figura 2.35, y en los parámetros de λ y μ se colocan valores arbitrariamente desde donde se desea que se inicie el ajuste.

Fractional PID controller parameters		
		Constraints
		Min Max
Kp	2.28814	0 100
Ki	10593.2	0 100
lam	0.9	0.01 2
Kd	0.000123559	0 100
mu	0.8	0.01 2

Figura 2.36. Sintonización FOPID.

Fuente: Autores.

Elaboración: Autores

En la opción que se muestra en la Figura 2.37, se coloca el algoritmo de optimización y el índice de desempeño.

Optimization and performance settings	
Optimization algorithm	optimize(): Nelder-Mead
Performance metric	IAE

Figura 2.37. Selección de desempeño para el controlador.

Fuente: Autores.

Elaboración: Autores

Paso 5. Finalmente se realiza una aproximación entera del controlador PID fraccionario con el comando *nipid* que nos brinda el toolbox NINTEGER, como muestra en la Figura 2.38.

```

1  % Aproximación entera del controlador PID fraccionario
2  - kp=1.3437;
3  - ki=99.9797;
4  - kd=0.00091695;
5  - lam=0.89826;
6  - u=0.80168;
7  - cc=nipid(kp,kd,u,ki,lam,[2 400],2,'crone') % Continuo

```

Figura 2.38. Aproximación entera del FOPID.

Fuente: Autores.

Elaboración: Autores

Y se obtiene como resultado la función de transferencia continua “Ziegler-Nichols, Nelder-Mead, IAE” como se muestra en la Figura 2.39. Para el resto de métodos se sigue los mismos pasos.

$$\frac{0.01223 s^3 + 4.712 s^2 + 487.3 s + 7617}{0.008704 s^3 + 2.381 s^2 + 76.24 s}$$

Continuous-time transfer function.

Figura 2.39. Funcion de transferencia entera del FOPID.

Fuente: Autores.

Elaboración: Autores

En las siguientes secciones se documentan las funciones de control FOPID resultantes de la aplicación de los métodos de diseño de controladores de orden fraccionario, de los métodos de optimización y sus respectivos índices de desempeño. Para mostrar cada controlador se nombra primero el método de sintonización, método de optimización y el índice de desempeño usado para este controlador.

Controlador Ziegler-Nichols, Nelder-Mead, IAE

FOPID:

$$1.3437 + \frac{99.9797}{s^{0.8926}} + 0.00091695s^{0.80168}$$

Aproximación FOPID

$$\frac{0,01223s^3 + 4,712s^2 + 487,3s + 7617}{0,008704s^3 + 2,381s^2 + 76,24s}$$

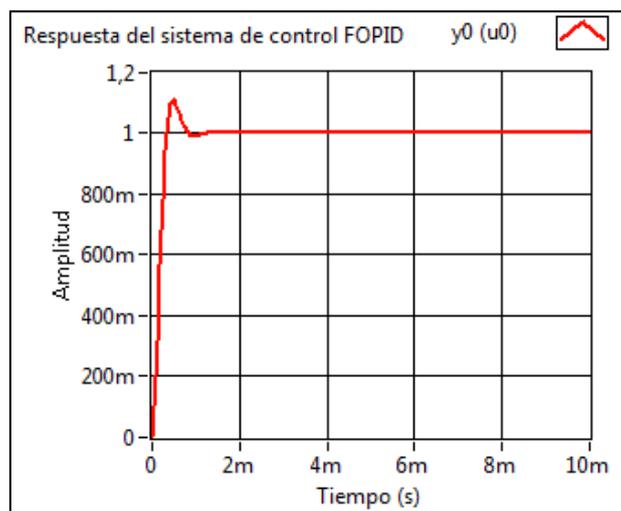


Figura 2.40. Controlador Ziegler-Nichols, Nelder-Mead, IAE.

Fuente: Autores.

Elaboración: Autores

Controlador Ziegler-Nichols, Nelder-Mead, ISE

FOPID:

$$1.1944 + \frac{98.098}{s^{0.9461}} + 0.0013562s^{0.73498}$$

Aproximación FOPID

$$\frac{0,3641s^3 + 78,59s^2 + 4992s + 78910}{0,3017s^3 + 35,09s^2 + 804,6s}$$



Figura 2.41. Controlador Ziegler-Nichols, Nelder-Mead, ISE.
Fuente: Autores.
Elaboración: Autores

Controlador Ziegler-Nichols, Nelder-Mead, ITSE

FOPID:

$$1.0657 + \frac{99.0391}{s^{0.99053}} + 0.0014063s^{0.75225}$$

Aproximación FOPID

$$\frac{0,02079s^3 + 6,546s^2 + 564,1s + 10880}{0,01826s^3 + 4,318s^2 + 109,9s}$$



Figura 2.42. Controlador Ziegler-Nichols, Nelder-Mead, ITSE.
Fuente: Autores.
Elaboración: Autores

Controlador Ziegler-Nichols, Nelder-Mead, ITAE

FOPID:

$$1.3419 + \frac{99.9806}{s^{0.89833}} + 0.0008107s^{0.80153}$$

Aproximación FOPID

$$\frac{0,01217s^3 + 4,709s^2 + 487,4s + 7622}{0,008714s^3 + 2,383s^2 + 76,28s}$$

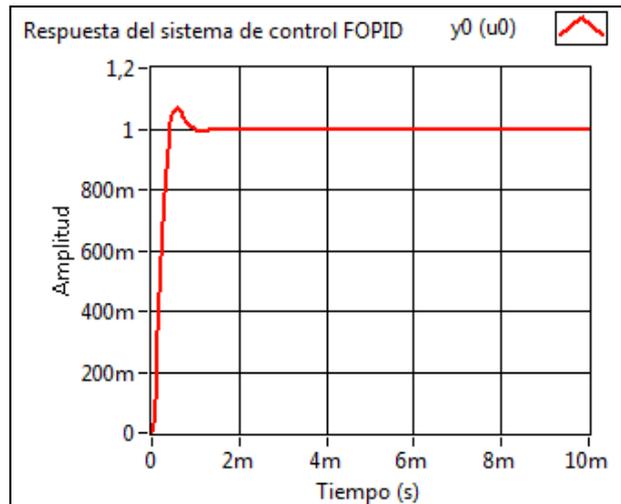


Figura 2.43. Controlador Ziegler-Nichols, Nelder-Mead, ITAE.
Fuente: Autores.
Elaboración: Autores

Controlador Cohen-Coon, Nelder-Mead, ISE

FOPID:

$$1.2806 + \frac{99.995}{s^{0.89978}} + 0.00011335s^{0.90183}$$

Aproximación FOPID

$$\frac{0,00686s^3 + 3,25s^2 + 374,8s + 6029}{0,005304s^3 + 1,832s^2 + 60,34s}$$



Figura 2.44. Controlado Cohen-Coon, Nelder-Mead, ISE.
Fuente: Autores.
Elaboración: Autores

Controlador Cohen-Coon, Nelder-Mead, IAE

FOPID:

$$1.3799 + \frac{99.9912}{s^{0.79545}} + 0.0021312s^{0.71345}$$

Aproximación FOPID

$$\frac{0,01168s^3 + 4,967s^2 + 577,3s + 7219}{0,007942s^3 + 1,873s^2 + 72,31s}$$

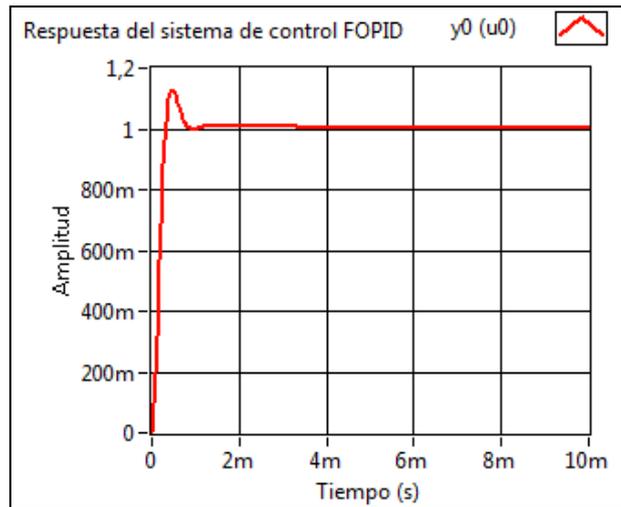


Figura 2.45. Controlador Cohen-Coon, Nelder-Mead, IAE.
Fuente: Autores.
Elaboración: Autores

Controlador Cohen-Coon, Nelder-Mead, ITSE

FOPID:

$$1.3097 + \frac{99.935}{s^{0.84464}} + 0.00080899s^{0.80468}$$

Aproximación FOPID

$$\frac{0,00881s^3 + 3,86s^2 + 463,5s + 6558}{0,006455s^3 + 1,815s^2 + 65,69s}$$

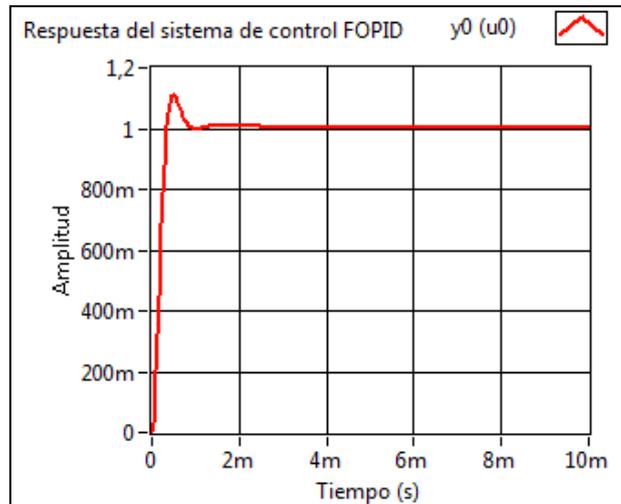


Figura 2.46. Controlador Cohen-Coon, Nelder-Mead, ITSE.
Fuente: Autores.
Elaboración: Autores

Controlador Cohen-Coon, Nelder-Mead, ITAE

FOPID:

$$1.0796 + \frac{99.9957}{s^{0.98234}} + 0.0031903s^{0.67508}$$

Aproximación FOPID

$$\frac{0,03101s^3 + 8,545s^2 + 701,9s + 13050}{0,02603s^3 + 5,175s^2 + 130,5s}$$

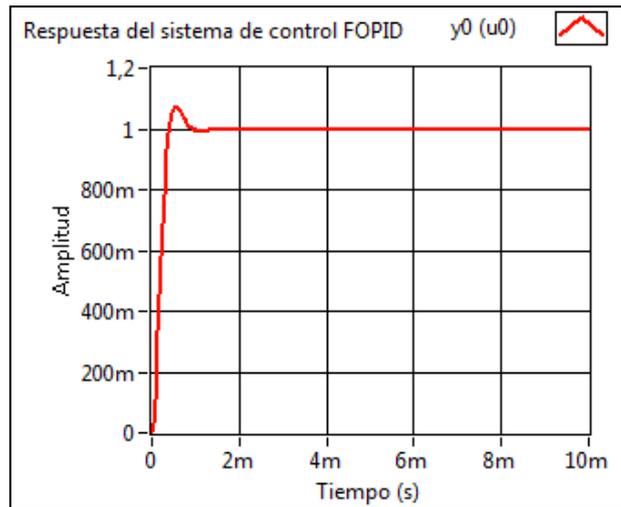


Figura 2.47. Controlador Cohen-Coon, Nelder-Mead, ITAE.
Fuente: Autores.
Elaboración: Autores

Controlador FMINCON, INTERIOR-POINT, ISE

FOPID:

$$2.3248 + \frac{98.9683}{s^{0.90374}} + 0.00013383s^{0.80186}$$

Aproximación FOPID

$$\frac{0,01089s^3 + 7,162s^2 + 561s + 7647}{0,008952s^3 + 2,445s^2 + 77,32s}$$

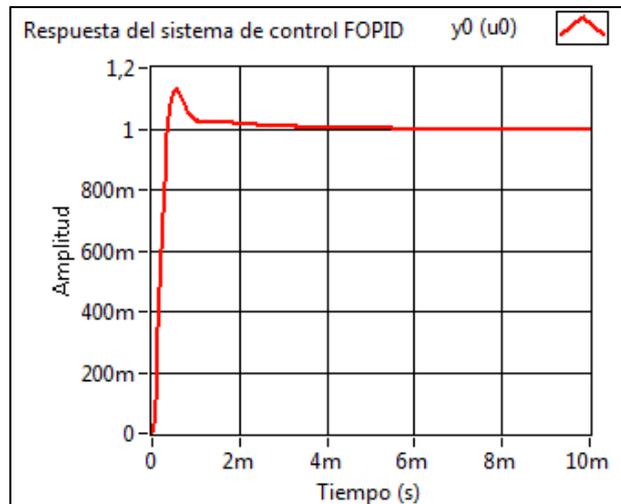


Figura 2.48. Controlador FMINCON, INTERIOR-POINT, ISE.
Fuente: Autores.
Elaboración: Autores

Controlador FMINCON, INTERIOR-POINT, IAE

FOPID:

$$2.3248 + \frac{98.9683}{s^{0.90374}} + 0.00013383s^{0.80186}$$

Aproximación FOPID

$$\frac{0,02089s^3 + 7,162s^2 + 561s + 7647}{0,008952s^3 + 2,445s^2 + 77,32s}$$



Figura 2.49. Controlador FMINCON, INTERIOR-POINT, IAE.
Fuente: Autores.
Elaboración: Autores

2.3.4. Simulación de controladores FOPI.

Para la sintonización del controlador PI de orden fraccionario se siguieron los pasos antes mencionados para la sintonización del FOPID.

Controlador CHIEN-Regulador, Nelder-Mead, ISE

FOPI:

$$2.0642 + \frac{236.96}{s^{1.1133}}$$

Aproximación FOPI

$$\frac{0,02006s^3 + 5,944s^2 + 237s + 639,8}{0,009718s^3 + 2,879s^2}$$

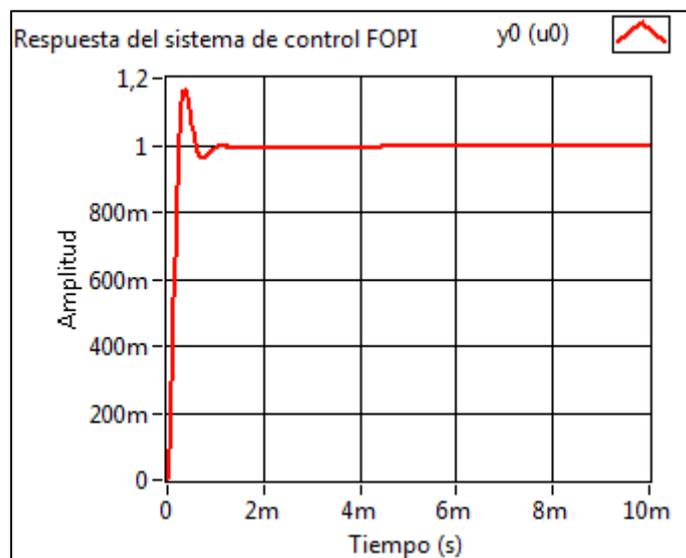


Figura 2.50. Controlador FOPI CHIEN-Regulador, Nelder-Mead, ISE.
Fuente: Autores.
Elaboración: Autores

Controlador CHIEN-Regulador, FMINCON, ISE

FOPI:

$$2.07 + \frac{299.9985}{s^{0.9492}}$$

Aproximación FOPI

$$\frac{1,582s^2 + 351,2s + 7417}{0,7643s^2 + 24,73s}$$



Figura 2.51. Controlador FOPI CHIEN-Regulador, FMINCON, ISE.
Fuente: Autores.
Elaboración: Autores

Controlador CHIEN-Regulador, FMINCON, ITAE

FOPI:

$$0.83852 + \frac{240.9854}{s^{0.9586}}$$

Aproximación FOPI

$$\frac{0,6736s^2 + 262,2s + 6108}{0,8033s^2 + 25,35s}$$

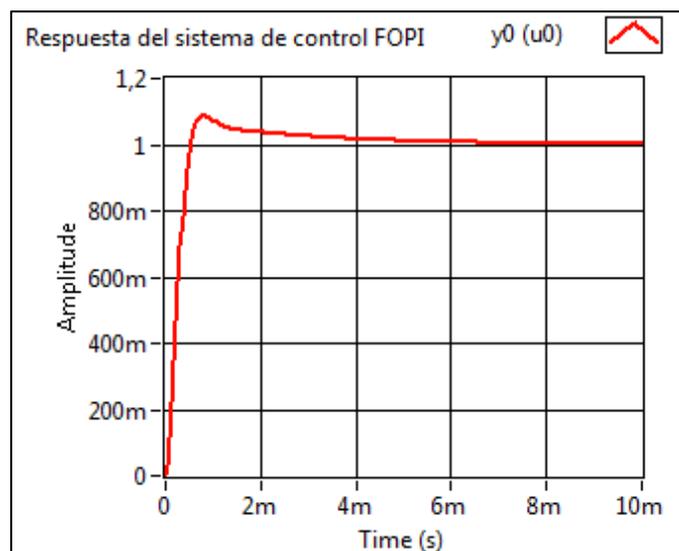


Figura 2.52. Controlador FOPI CHIEN-Regulador, FMINCON, ITAE.
Fuente: Autores.
Elaboración: Autores

Controlador C-H-R, Nelder mead, ISE

FOPI:

$$2.0702 + \frac{416.28}{s^{1.0815}}$$

Aproximación FOPI

$$\frac{0,01719s^3 + 5,54s^2 + 416,3s + 1033}{0,008302s^3 + 2,676s^2}$$

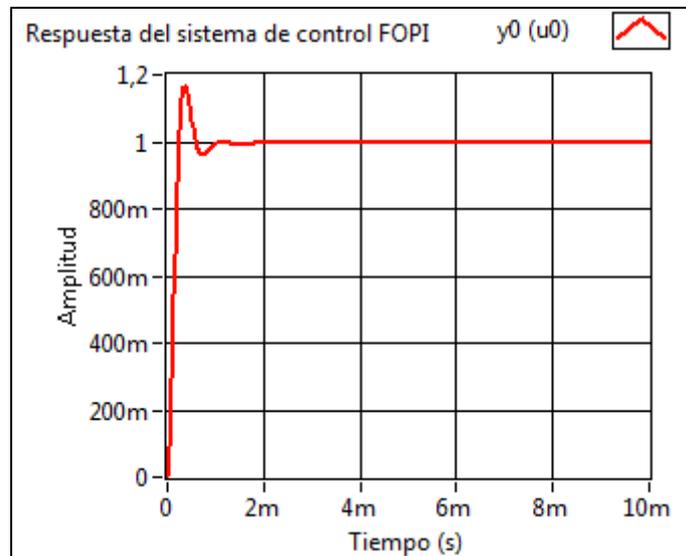


Figura 2.53. Controlador FOPI C-H-R, Nelder Mead, ISE.

Fuente: Autores.

Elaboración: Autores

A continuación, en la Tabla 2.10 se realiza una comparación de los principales parámetros de la respuesta temporal de los sistemas de lazo cerrado obtenidos en la simulación para controladores de orden fraccional.

Tabla 2.10: Parámetros de la respuesta temporal de controladores FOPID y FOPI.

	Método de ajuste	Sobrelongación (%)	Tiempo de establecimiento (s)
FOPID	Ziegler-Nichols, IAE	10.759	0.000774
	Ziegler-Nichols, ISE	8.530	0,000843
	Ziegler-Nichols, ITSE	6.595	0.000878
	Ziegler-Nichols, ITAE	10.681	0.000809
	Cohen-Coon, ISE	9.609	0.000859
	Cohen-Coon, IAE	12.841	0.001592
	Cohen-Coon, ITSE	11.019	0.000825
	Cohen-Coon, ITAE	7.303	0.000845
	FMINCON, ISE	0.379	0.001282
	FMINCON, IAE	19.952	0.000874
FOPI	CHIEN-Regulador, ISE	16.514	0.000967
	CHIEN-Regulador, ISE	19.165	0.001256
	CHIEN-Regulador, ITAE	8.719	0.005255
	C-H-R, ISE	16.603	0.000965

Fuente: Autores.

Elaboración: Autores

CAPÍTULO III

3. IMPLEMENTACIÓN DEL SISTEMA DE CONTROL EMBEBIDO

3.1. Arquitectura Hardware.

La plataforma sbRIO-9651 es un sistema embebido que incluye dos componentes: un procesador (ARM Cortex-A9) y un FPGA (Xilinx Artix-7). El procesador es el que ejecuta el sistema operativo en tiempo real (RTOS), el cual ofrece un comportamiento fiable para la implementación del sistema de control del motor DC. El FPGA es el hardware reconfigurable que posee módulos E/S para la interconexión con dispositivos periféricos, así también es el encargado de realizar las tareas que requieren una ejecución a alta velocidad. En el presente proyecto el Procesador RT (Real-Time) se encarga de ejecutar las funciones de control y cálculos adicionales, y el FPGA se encarga de adquirir la señal del sensor de velocidad rotacional y de generar la señal PWM que controla la velocidad del Motor DC.

En la Figura 3.1 se observa el diagrama de bloques del sistema embebido implementado para el control de velocidad de un motor DC.

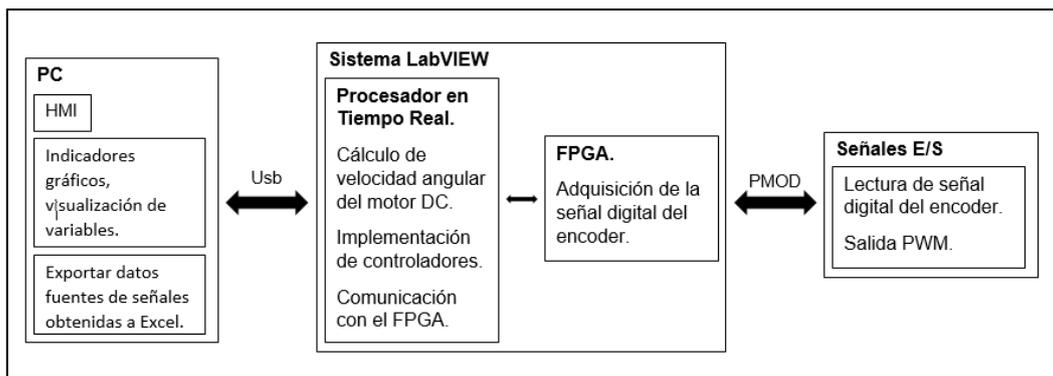


Figura 3.1. Diagrama de bloques del sistema embebido de control.

Fuente: Autores.

Elaboración: Autores.

El procesador RT ejecuta la función de control en alto nivel (función de transferencia en tiempo discreto). El FPGA ejecuta las operaciones de control en bajo nivel, permite la configuración de los módulos E/S y la adquisición de la señal del encoder incremental. Cabe recalcar que el sistema de control embebido opera autónomamente, sin embargo, para efectos de cuantificar el desempeño se agregó al sistema un computador, el cual proporciona una interfaz de usuario que permite al operador interactuar con el sistema, visualizar y registrar los datos y señales útiles para procedimientos de post-procesamiento de análisis de desempeño del controlador.

En la Figura 3.2 se muestra la arquitectura que se utilizó para el sistema de control, en donde la interfaz humano-maquina se encuentra en la PC, junto con la plataforma RT-FPGA sbRIO-9651 y los módulos E/S.

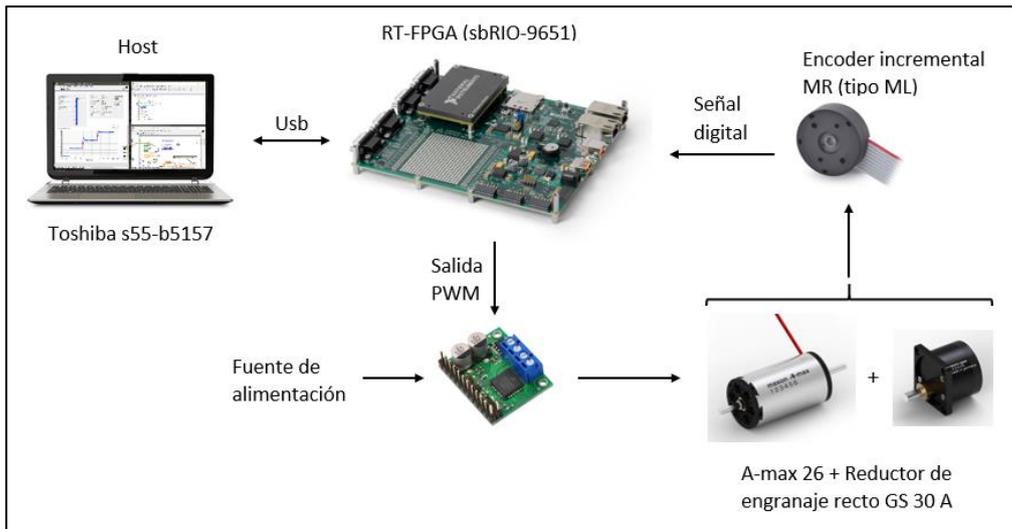
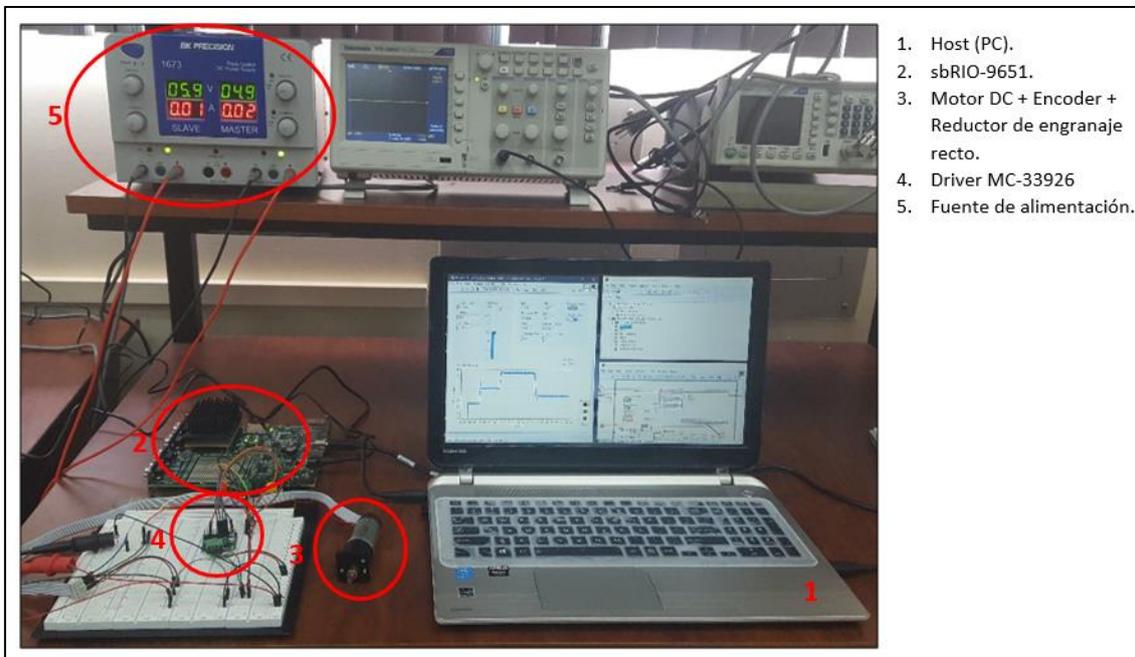


Figura 3.2. Arquitectura del sistema de control del motor DC.

Fuente: Autores.

Elaboración: Autores.

La FPGA posee cuatro conectores Pmod de 12 pines, estos son interfaces de E/S digitales que permiten la conexión de sensores y dispositivos digitales. Se usa un pin del Pmod3 para conectar el encoder digital, el Pmod4 es destinado para la conexión con el driver. En la Figura 3.3 se observa todos los dispositivos conectados de acuerdo a la arquitectura mostrada en la Figura 3.2.



1. Host (PC).
2. sbRIO-9651.
3. Motor DC + Encoder + Reductor de engranaje recto.
4. Driver MC-33926
5. Fuente de alimentación.

Figura 3.3. Configuración experimental del sistema embebido.

Fuente: Autores.

Elaboración: Autores.

3.1.1. Conexión del sistema embebido.

Se requiere dos interfaces para la conexión entre todos los dispositivos que componen el sistema embebido, el primero como entrada para la lectura del sensor y segundo como salida para controlar el motor DC, el PMOD3 es destinado para la conexión del encoder y el PMOD4 para conectar el driver.

3.1.1.1. Interfaz con el sensor Encoder incremental.

El encoder incremental tiene tres salidas digitales (canal A, canal B, Index), en la implementación del proyecto se usa la salida digital del canal A, para leer los datos de este canal se usó el Pin 1 del Pmod3 como entrada para la conexión del sensor con la sbRIO-9651. El sensor debe alimentarse con una fuente de 5 V, debido a que el FPGA no cuenta con este requisito, se utiliza una fuente de alimentación externa, el pin 6 del sensor corresponde a la señal digital del canal A.

En la Figura 3.4 se muestra la conexión que se realizó en la implementación del sistema de control, la lectura de la velocidad del motor se la realizó de forma digital, obteniendo los pulsos de la señal del encoder incremental y procesándolos en el FPGA.

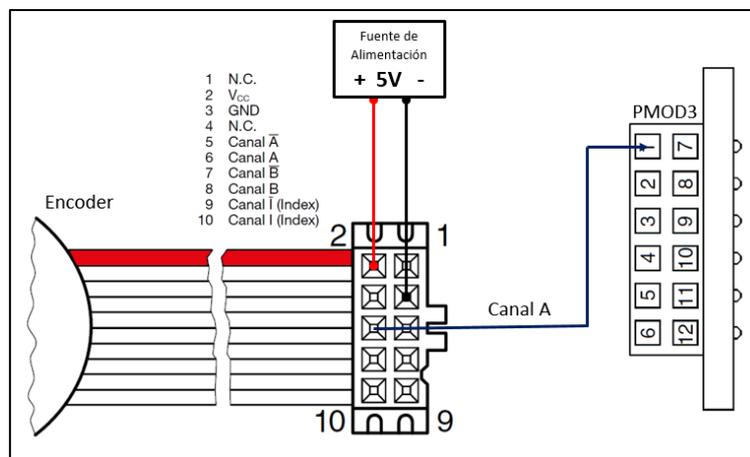


Figura 3.4. Conexión de Encoder-PMOD3.

Fuente: Autores.

Elaboración: Autores.

El sensor encoder MR, del sistema Maxon está basado en el principio magnetoresistivo, en donde un pequeño imán permanente multipolar se coloca en el eje del motor DC, según el fabricante, la salida de este dispositivo entrega 1000 pulsos por cada vuelta que realiza el eje del motor [21].

3.1.1.2. Interfaz sbRIO-9651 con Motor dc.

La sbRIO-9651 no puede ser conectada directamente con el motor DC, esto debido que el actuador necesita una etapa de potencia (amplificación de corriente) para operar adecuadamente, este driver recibe la señal PWM de la salida digital del FPGA y luego la amplifica para enviarla al Motor DC.

Como se observa en la Figura 3.5, se requieren pocos pines para la conexión entre motor-driver-FPGA, para controlar el sentido de giro del motor del motor se usan los pines IN1 y IN2, el D2 se usa para enviar la señal PWM del FPGA al driver y controlar la velocidad del motor, finalmente el pin D1 es conectado a tierra y el pin EN se conecta a un valor lógico en alto para activar el driver.

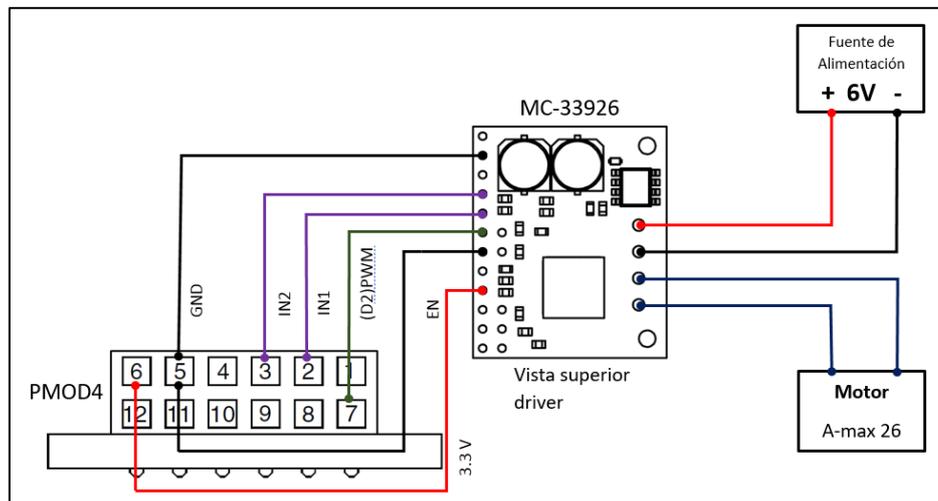


Figura 3.5. Conexión del controlador de potencia-PMOD4.

Fuente: Autores.

Elaboración: Autores.

El conector Pmod4 fue usado como salida digital en donde se envía la señal de control PWM al driver, en la Tabla 3.1 se muestra la conexión de los pines de entrada necesarios para la aplicación que se necesita implementar.

Tabla 3.1: Conexión del driver con la tarjeta sbRIO-9651.

Pmod4	MC-33926	
PIN7	PWM/D2	Pin para la señal de control PWM.
PIN5-GND	PWM/D1	Entrada deshabilitada.
PIN2	IN1	Entradas de control lógico para el sentido de giro.
PIN3	IN2	
PIN6-3.3V	EN	Entrada para habilitar el driver.
PIN5-GND	GND	Conexión de tierra

Fuente: Autores

Elaboración: Autores.

El motor DC es conectado a las salidas del driver OUT1 y OUT2, se requiere una fuente de alimentación externa para alimentar al dispositivo electromecánico, esta fuente debe suministrar 6 V y al menos 1 A para alimentar al motor.

La Tabla 3.2 muestra los pines usados para la conexión del motor con el driver y la alimentación del motor DC con una fuente de 6 V.

Tabla 3.2: Conexión del driver, fuente de alimentación y motor DC A-max 26.

Motor DC		MC-33926	
PIN1	OUT1	Pines de salida para el motor.	
PIN2	OUT2		
Fuente de Alimentación		MC-33926	
Vcc	VIN	Conexión para la alimentación del motor de 5 V-28 V.	
GND	GND	Conexión de tierra para la fuente de alimentación del motor.	

Fuente: Autores.

Elaboración: Autores.

3.2. Desarrollo del proyecto en la plataforma sbRIO-9651.

El enfoque del proyecto es la implementación de los controladores de orden entero y fraccionario utilizando programación gráfica de LabVIEW y la plataforma sbRIO-9651 que incluye el procesador en tiempo real (RT) y el chasis del FPGA desarrollado por National Instruments.

El proyecto está compuesto de tres instrumentos virtuales (VI), uno es el programa que se compila, traduce y transfiere en el FPGA, en este se configura las entradas y salidas digitales a través de los módulos Pmod; los otros dos VI's desarrollados en el procesador Real Time están encargados de establecer comunicación entre el Procesador y el FPGA desde una interfaz gráfica en la que se permitirá procesar las variables del sistema de control y en los cuales se implementan los controladores de orden entero y fraccionario.

3.2.1. Creación del proyecto.

El VI del FPGA sirve para los dos tipos de controladores (PID y FOPID) a ejecutarse en el procesador Real-Time. El VI del FPGA realiza la adquisición y procesamiento de la señal digital del encoder y genera la señal PWM para control de velocidad.

Antes de implementar los controladores en el procesador Real Time, se debe cargar los Toolkits necesarios [49] para la tarjeta sbRIO-9651, en este caso se usó el módulo de *Control Design and Simulation 16.0* (CDSim), para realizar este procedimiento se hace uso del software NI MAX. En la Figura 3.6 se muestra que el módulo CDSim está

correctamente instalado en la plataforma sbRIO-9651, esto se verifica en la pestaña “Sistemas remotos” dentro del NI MAX.

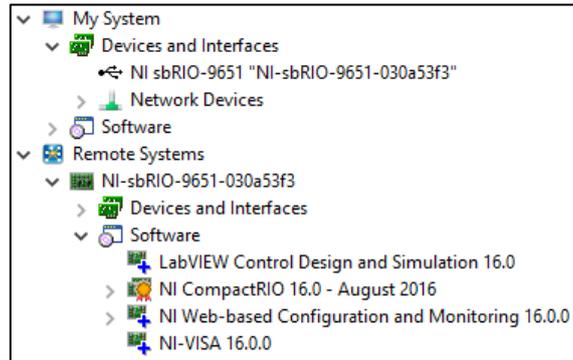


Figura 3.6. Toolkits instalados en el dispositivo sbRIO-9651.
Fuente: Autores.
Elaboración: Autores.

Una vez instaladas las herramientas necesarias para la implementación del proyecto se procede a crear los VI's en el FPGA y en el procesador Real Time. Se organizan estos archivos mediante la generación de un proyecto, en el Real Time se implementan los controladores y todos los subprocesos de procesamiento de datos, mientras en el FPGA se realiza la adquisición de señales digitales y generación de PWM.

La estructura del proyecto consiste en tres partes importantes: el host, Real-Time y el FPGA, en la Figura 3.7 se muestra la distribución de los VI's usados en el proyecto. La conexión entre el PC (Host) y la sbRIO-9651 se realizó mediante USB.

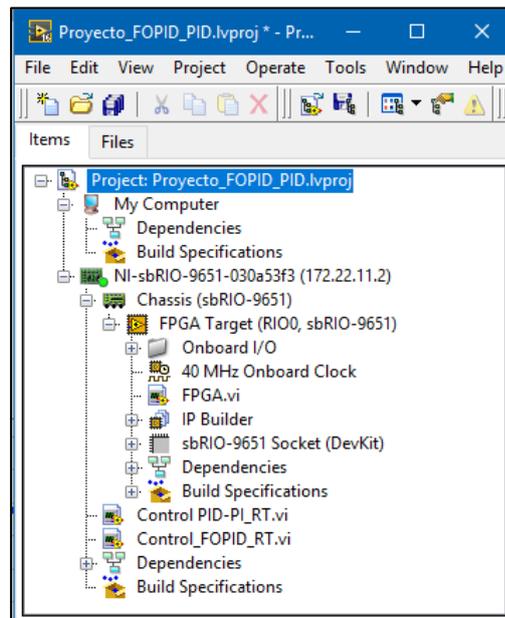


Figura 3.7. Estructura del proyecto del sistema embebido.
Fuente: Autores.
Elaboración: Autores.

Cada VI tiene subprocesos que cumplen diferentes funciones y están interconectados entre ellos para poder formar un sistema y permitir de esta manera la implementación de un control PID, PI, FOPID y FOPI de un motor DC desde LabVIEW a través de la plataforma sbRIO-9651. A continuación, se detallan las estructuras y funcionamiento de cada uno de los VI's implementados y de los procesos que se realizan dentro de cada uno de ellos.

3.2.2. Subprocesos en el FPGA.

La programación del FPGA consiste en la configuración de entradas/salidas de los conectores Pmod y los subprocesos de adquisición de la señal digital del encoder y la generación de la señal PWM para el control de potencia administrada al driver del motor.

La entrada del Pmod que corresponde a la señal del encoder, se la procesa mediante lógica booleana, se determina cuando se produce un cambio de borde ascendente en la señal cuadrada del sensor, dentro del CASE se usa un reloj FPGA para determinar la cantidad de pulsos (pulsaciones por intervalo - PPI) existentes en cada periodo; mediante un registro de desplazamiento, la cantidad de pulsos del periodo actual se restan de la cantidad de pulsos del periodo anterior.

Posteriormente en el procesador Real Time se realizó las operaciones de conversión para obtener la frecuencia de la señal digital y la velocidad angular del rotor. En la Figura 3.8 se muestra el código en el diagrama de bloques para la adquisición de la señal digital.

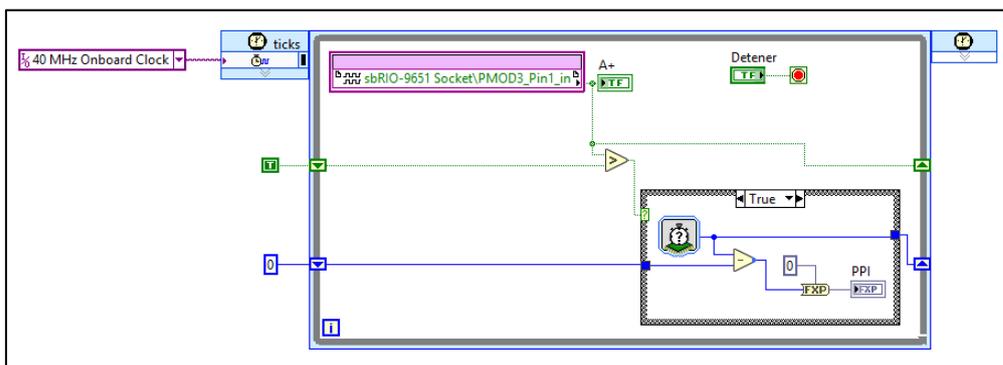


Figura 3.8. Adquisición y procesamiento de la señal digital del encoder en FPGA.

Fuente: Autores.

Elaboración: Autores.

Se usa PWM (Modulación por ancho de Pulso) para controlar el circuito analógico MC-33926, se varía la duración del ciclo de trabajo de los pulsos digitales de frecuencia fija para determinar la potencia que será entregada al circuito. La señal PWM se utiliza para controlar la velocidad del motor DC. La relación del ancho del pulso para el periodo se conoce como ciclo de trabajo de la señal [50].

En la Figura 3.9 se muestra cómo se controla en el FPGA el ancho de los tiempos en altos y bajo de los pulsos. Los valores de “ T_{off} ” y “ T_{on} ” serán procesados en el Real Time a la salida del controlador con el respectivo ciclo de trabajo.

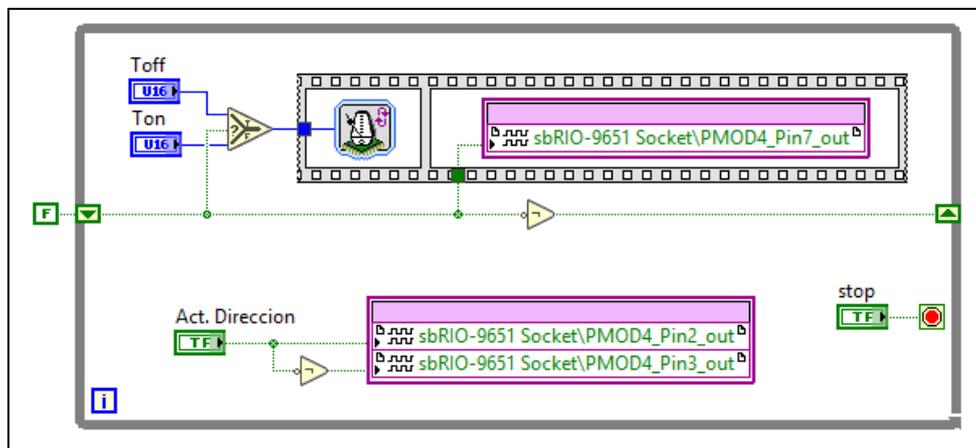


Figura 3.9. Generación de la señal PWM.

Fuente: Autores.

Elaboración: Autores.

Se usan los pines 2 y 3 para controlar el sentido de giro del motor, el pin 7 es usado para enviar la señal PWM, para habilitar las salidas digitales del Pmod4 del FPGA se da un valor lógico alto, la habilitación de las salidas digitales se muestra en la Figura 3.10, estas salidas serán conectadas al driver para controlar al motor DC.

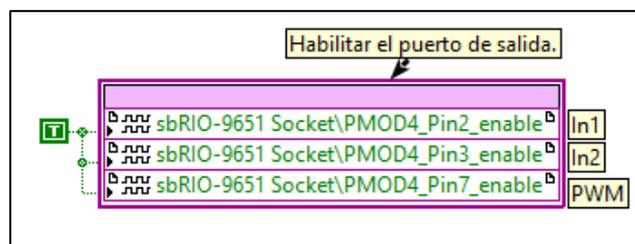


Figura 3.10. Habilitación de pines de salida.

Fuente: Autores.

Elaboración: Autores.

Terminado el código en el diagrama de bloques del FPGA se procede a compilar el programa, existen dos formas para la compilación de un VI en el FPGA, el primero es instalar: “LabVIEW 2016 FPGA Module Xilinx Compilation Tool for ISE 14.7”, este módulo permite la compilación dentro de máquinas con versiones iguales o inferiores a Windows 7. La segunda opción es la compilación de un VI en el FPGA, usando el servicio de compilación en la nube.

La compilación en un equipo con Windows 8 o 10 no es posible debido a que el compilador de Xilinx no es compatible con estos sistemas operativos. Se menciona esto, debido a que el proyecto se desarrolló usando un PC con sistema operativo Windows 10, la solución a este inconveniente es usar el servicio de compilación en la nube.

Para empezar a usar este servicio, se requiere crear una cuenta en el portal del servicio de compilación en la nube, este servicio es gratuito y tiene una duración de uso de 90 días [51]. Al compilar el VI en el FPGA se selecciona la opción “LabVIEW FPGA Compile Cloud Service”, aquí se pide iniciar sesión para acceder al servicio de compilación en la nube de National Instruments.

3.2.3. Subprocesos en el Procesador Real Time.

La plataforma RT es responsable de la comunicación con el FPGA, la aplicación que se ejecuta en el PC y también de enviar la señal de control al generador PWM en la salida de controlador PID/FOPID.

Se requiere enlazar el procesador Real Time con el FPGA, en donde se realizará la lectura y escritura de todas las variables que se usaron en el sistema de control. En la Figura 3.11 se observa parte de código necesario para realizar esta comunicación, desde el VI del Real-Time se abre una referencia al VI del FPGA, después se realiza la lectura y escritura de todos los parámetros de entrada y salida del VI del FPGA, y finalmente se cierra la referencia.

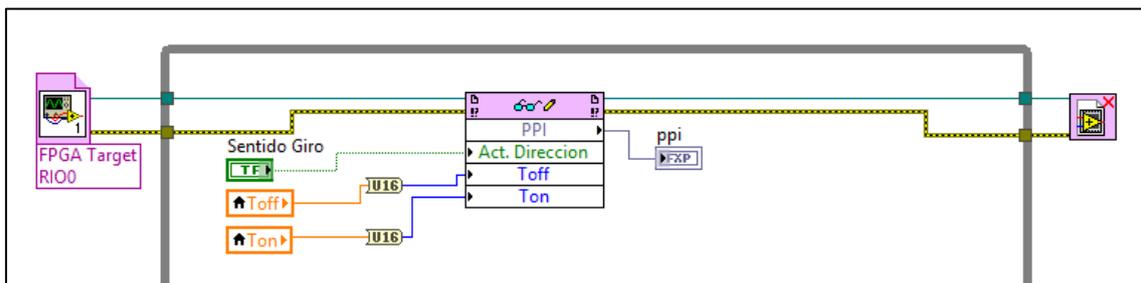


Figura 3.11. Comunicación entre el Procesador en Tiempo Real y FPGA.

Fuente: Autores.

Elaboración: Autores.

Debido a que el valor de la variable “PPI”, tiende a variar demasiado, se procede a realizar un promedio de los valores obtenidos para suavizar el comportamiento de la señal en el tiempo, para realizar este paso se crea un vector de longitud fija en el que se insertan los valores de la variable PPI en el vector, cuando se inserta el valor en la última posición del vector el proceso comienza de nuevo en el índice cero. Durante cada iteración se suman todos los elementos y se divide por la longitud del vector, este procedimiento se muestra en la Figura 3.12.

Después, se procede a obtener la frecuencia de la señal del encoder, para esto se considera la frecuencia con la que opera la sbRIO-9651 (40 MHz) [52], y se aplica la siguiente relación:

$$f_{se} = \frac{f_t}{ppi} \quad (3.1)$$

Donde:

f_{se} : Frecuencia la señal del encoder.

f_t : Frecuencia en la que trabaja la sbRIO.

ppi : Pulsos por intervalo.

Para realizar la conversión de “pulsos por periodo” a “revoluciones por minuto” se toma en cuenta la constante del sensor denominada pulsos por vuelta, para este caso el valor es de 1000 pulsos por vuelta. La medición de la velocidad angular es dada por:

$$RPM = f_{se} * \frac{1 rev}{ppv} * \frac{60 seg}{min} \quad (3.2)$$

Donde:

f_{se} : Frecuencia de la señal del encoder.

ppv : Numero de pulsos por vuelta del encoder.

Otro aspecto que hay que tener en cuenta, es la caja de reducción que posee el sistema sensor-motor-reductor, la velocidad de rotación obtenida se la divide por 200 (200:1, relación de reducción) debido a que el sensor se encuentra ubicado en el eje del motor y no en la salida del reductor. En la Figura 3.12 muestra el código realizado en LabVIEW para obtener el valor de velocidad en RPM a partir del número de pulsos por intervalo.

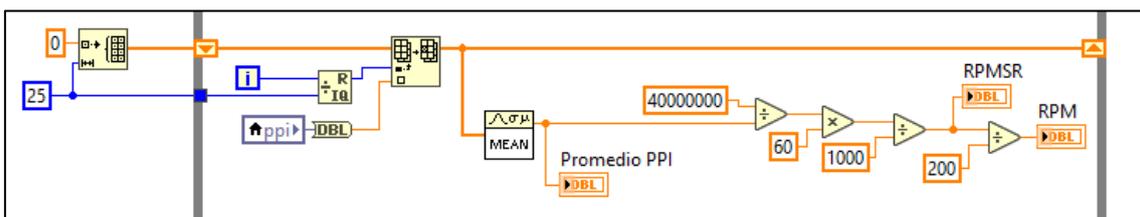


Figura 3.12. Medición de velocidad angular.

Fuente: Autores.

Elaboración: Autores.

Para la implementación del control de velocidad se usó las herramientas de software “PID Structure Conversion” y “PID” de LabVIEW, para la implementación de los controladores PID y PI se procedió a convertir los controladores diseñados en el Capítulo 2 de estructura en paralelo a estructura académica.

De acuerdo a la Figura 3.13 la salida del controlador está limitada en el rango de 0-100 y está dado en términos de porcentaje, a este valor se lo divide para 100 para obtener el ciclo de trabajo de PWM cuyo valor están dentro del rango de 0-1.

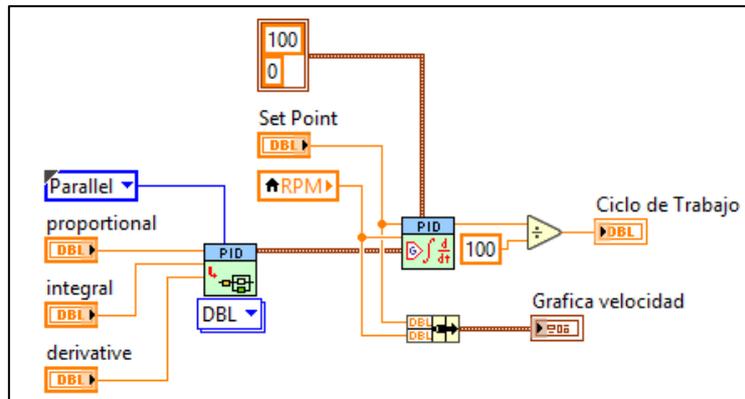


Figura 3.13. Controlador PID.
Fuente: Autores.
Elaboración: Autores.

Una vez obtenido el ciclo de trabajo se realiza un procedimiento para calcular el “Tiempo encendido” y “Tiempo apagado” de los pulsos de la señal. La potencia que será entregada al driver por medio de la señal PWM depende de la frecuencia de trabajo, para este proyecto se obtuvo de manera experimental que la frecuencia de trabajo para el motor DC debe ser de 25 Hz.

Los cálculos para la generación de la señal PWM son:

El periodo de la señal es dado por:

$$T = T_{on} + T_{off} = \frac{1}{f} \quad (3.3)$$

El “tiempo encendido” del pulso en alto es:

$$T_{on} = T * Ciclo\ de\ Trabajo \quad (3.4)$$

El “tiempo apagado” del pulso en bajo es:

$$T_{off} = T - T_{on} \quad (3.5)$$

La variación de velocidad del motor está determinada por el ciclo de trabajo que varía entre 0 (0%) y 1 (100%) en la salida del controlador; a partir de este valor el código genera los tiempos en alto y en bajo (encendido y apagado) usando las ecuaciones (3.3), (3.4) y (3.5), este cálculo en LabVIEW se muestra en la Figura 3.14.

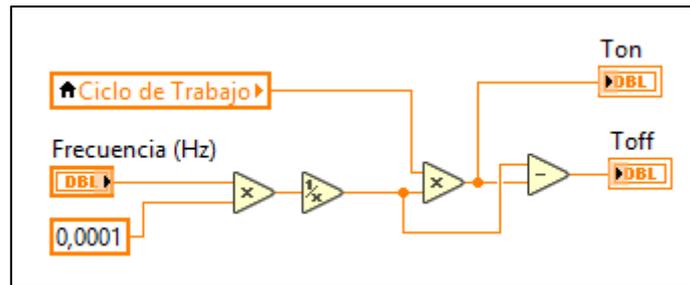


Figura 3.14. Sistema embebido de control.
 Fuente: [53].
 Elaboración: Autores.

Finalmente, se realiza el análisis de los parámetros temporales del sistema de control, para ello se utiliza las señales de entrada y salida: la señal de la velocidad deseada y la señal de la velocidad medida.

3.2.3.1. Proceso de exportación de datos.

El motor DC trabaja a una velocidad máxima de 15 rpm, se toma un cambio de escalón cuya amplitud es del 90% de la velocidad máxima, es decir 13.5 rpm. Después de realizar esto, se toma muestras por 5 segundos (a 100 muestras/segundo) a partir del cambio de escalón y posteriormente se exportan a Excel para determinar el porcentaje de sobrelongación y el tiempo de establecimiento de la señal de salida. En la Figura 3.15 se muestra el código usado para el muestreo de las señales de entrada y salida.

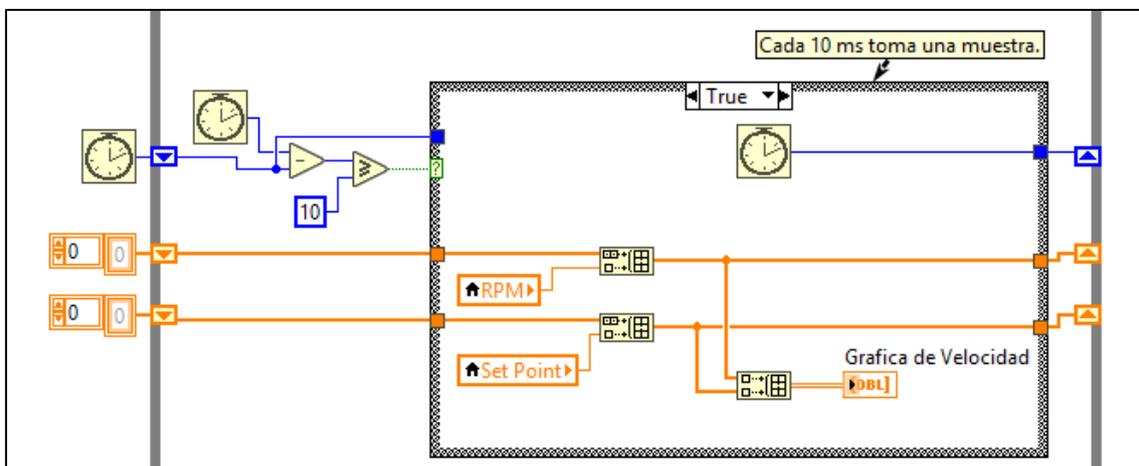


Figura 3.15. Muestreo de señal Set Point y señal Controlada.
 Fuente: Autores.
 Elaboración: Autores.

En la Figura 3.16 se puede observar una interfaz gráfica en donde se muestra algunas variables tales como los parámetros del controlador, el valor de velocidad obtenida del sensor encoder (RPM), un control mediante Set Point, parámetros necesarios para generar la señal PWM (Ton, Toff, ciclo de trabajo), un botón para el control del sentido

de giro del motor y un indicador gráfico para representar las señales de consigna y controlada del sistema.

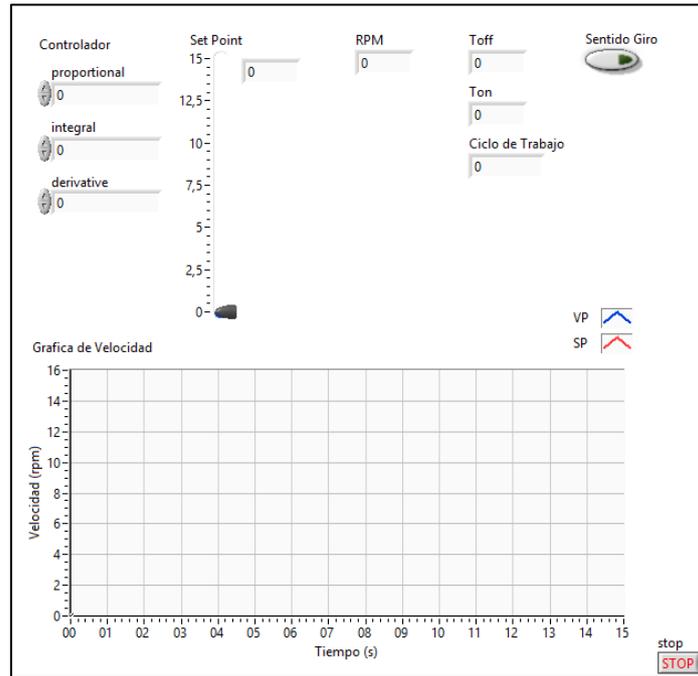


Figura 3.16. Interfaz gráfica para el control PID.
Fuente: Autores.
Elaboración: Autores.

Se observa en la Figura 3.17 los resultados experimentales de la respuesta del controlador ante varios cambios realizados en la señal de set Point.

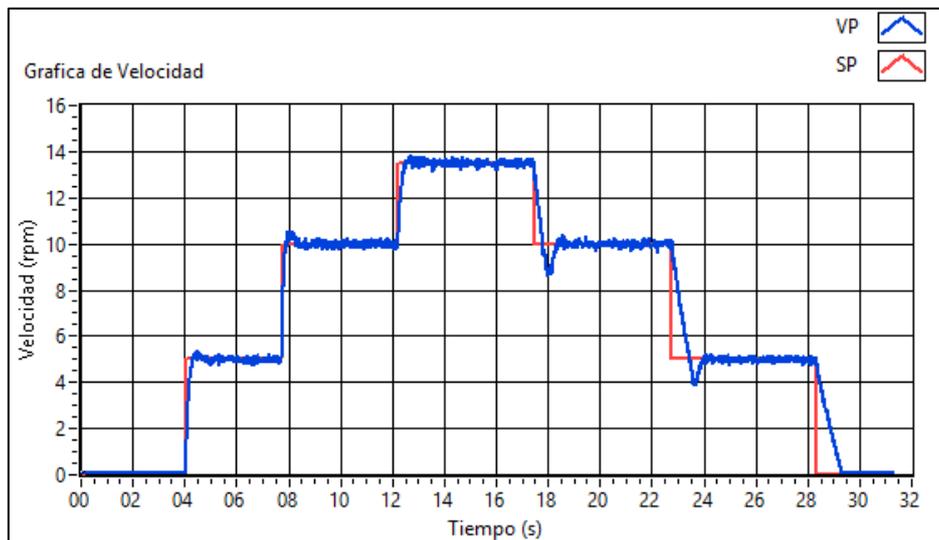


Figura 3.17. Datos obtenidos de la experimentación de controladores.
Fuente: Autores.
Elaboración: Autores.

3.3. Síntesis de la Función de Transferencia del FOPID.

La clave de la implementación de un controlador de orden fraccionario es la discretización de la función de transferencia, en comparación con los controladores de orden entero, los controladores de orden fraccionario tienen una memoria ilimitada, tomando en cuenta esta característica la discretización de los controladores es un aspecto importante y más adecuado para la implementación en hardware [16], [54].

Se realiza el ajuste de los parámetros del controlador de orden fraccionario en tiempo continuo y luego la función de transferencia es discretizada, se usó el software LabVIEW para discretizar la ecuación y se contrastó el resultado con lo obtenido en Matlab.

En la implementación de controladores FOPID y FOPI del presente proyecto, se reutiliza el mismo código de los VI's del Real-Time y el FPGA; la variante, es la forma de implementar el controlador. Al igual que los controladores PID y PI, dentro del módulo CDSim instalado existe la herramienta "CD Discrete Transfer Function" que permite discretizar una función de transferencia de orden entero. Las funciones de transferencia de los controladores FOPID y FOPI se transformaron a su forma discreta. En función del parámetro s , el FOPID queda de la siguiente manera:

$$C_{FOPID}(s) = K_p \left(1 + \frac{1}{T_i s^\lambda} + T_d s^\mu \right) \quad (3.6)$$

Y la función de transferencia del FOPI es [54]:

$$C_{FOPI}(s) = K_p \left(1 + \frac{1}{T_i s^\lambda} \right) \quad (3.7)$$

Para modelos SISO, las funciones de transferencia discretas usan la siguiente ecuación para calcular la salida:

$$H_{CFO}(z) = \frac{b_0 + b_1 z + \dots + b_{m-1} z^{m-1} + b_m z^m}{a_0 + a_1 z + \dots + a_{n-1} z^{n-1} + a_n z^n} \quad (3.8)$$

Los parámetros de los controladores de orden fraccionario se obtuvieron a partir de la herramienta computacional FMINCON, un toolbox del software Matlab. Una vez obtenida la función de transferencia continua del controlador se procede a discretizar la ecuación de cada controlador FOPID y FOPI para el control de velocidad de un motor DC.

Para implementar el controlador de orden fraccionario en la plataforma sbRIO-9651, se requiere que la función de transferencia del controlador esté en su forma discreta [16].

Para validar que la discretización de los controladores, se procedió a verificar la discretización de la función de transferencia usando los programas realizados en MATLAB y LabVIEW. Se observa en la Figura 3.18 el código usado en Matlab para la discretización de controladores FOPID, en [43] se menciona varios métodos de discretización:

- ZOH: Retención de orden cero en las entradas.
- FOH: Interpolación lineal de entradas.
- TUSTIN: Aproximación bilinear.
- MATCHED: Método coincidente de polos y ceros (sólo para sistemas SISO).

```

%% Discretizacion de Funcion de transferencia continua
% Metodos: zoh, foh, tustin, prewarp, matched
num = [0.01223 4.712 487.3 7617];
den = [0.008704 2.381 76.24 0];
Gs = tf(num,den);
Ts = 0.001;
[Nz,Dz] = c2dm(num,den,Ts,'zoh');
y = tf(Nz,Dz,Ts);

```

FOPID Discreto	$\frac{1.405 z^3 - 3.711 z^2 + 3.256 z - 0.9498}{z^3 - 2.753 z^2 + 2.514 z - 0.7607}$ <p style="font-size: small; margin: 0;">Sample time: 0.001 seconds Discrete-time transfer function.</p>
----------------	---

Figura 3.18. Discretización de una FT continua en Matlab.
Fuente: Autores.
Elaboración: Autores.

De los métodos mencionados se escogió el método de retención de orden cero (ZOH), porque proporciona una aproximación exacta entre los sistemas continuos y discretos y permitió implementar los controladores discretos en el sistema embebido correctamente. Este método en Matlab utiliza la función “c2d()” cuya sintaxis es:

$$H_d = c2d(H_c, T, \text{Método})$$

Donde:

H_d : La función de transferencia resultante.

H_c : La función de transferencia continua a discretizar.

T: El periodo de muestreo.

Método: El método usado para la discretización.

Se usó el mismo método de discretización que en Matlab, en la Figura 3.19 se observa el VI creado en LabVIEW, para obtener la función de transferencia discreta de un controlador FOPID.

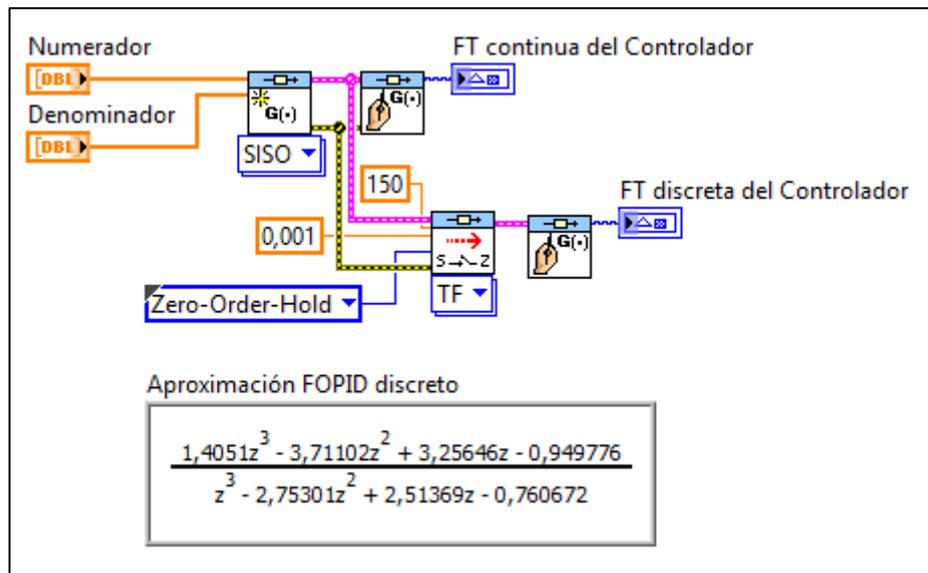


Figura 3.19. Diagrama de bloques para discretizar una FT continua.

Fuente: Autores.

Elaboración: Autores.

Como se mencionó anteriormente se requiere tener instalado módulo de “Control Design and Simulation 16.0” en la sbRIO-9651, una vez obtenida la función de transferencia discreta se utiliza la herramienta que permite implementar un controlador discreto para un sistema en tiempo real.

La función “CD Discrete Transfer Function” de la Figura 3.20 permite la transformación de cada función en tiempo continuo del controlador FOPID y FOPI en una función discreta.

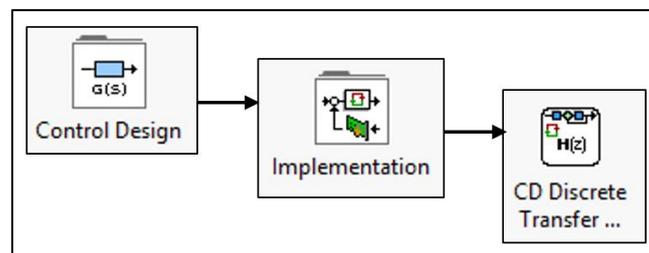


Figura 3.20. FT discreto para controladores fraccionarios.

Fuente: Autores.

Elaboración: Autores.

Se ingresa la señal de error en la entrada del controlador de orden fraccionario, así como se muestra en la Figura 3.21, y la salida del controlador es el ciclo trabajo que servirá para la generación de la señal PWM.

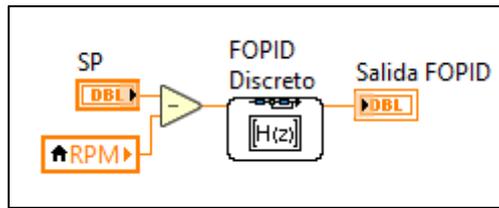


Figura 3.21. FOPID discreto en Real-Time.
 Fuente: Autores.
 Elaboración: Autores.

Para implementar cada uno de los controladores FOPID y FOPI, se configura el numerador y denominador de la función de transferencia discreta de cada uno de estas ecuaciones, se observa en la Figura 3.22 la implementación de la función de transferencia discreta de un controlador de orden fraccionario en el que se define el numerador y el denominador de la función de transferencia.

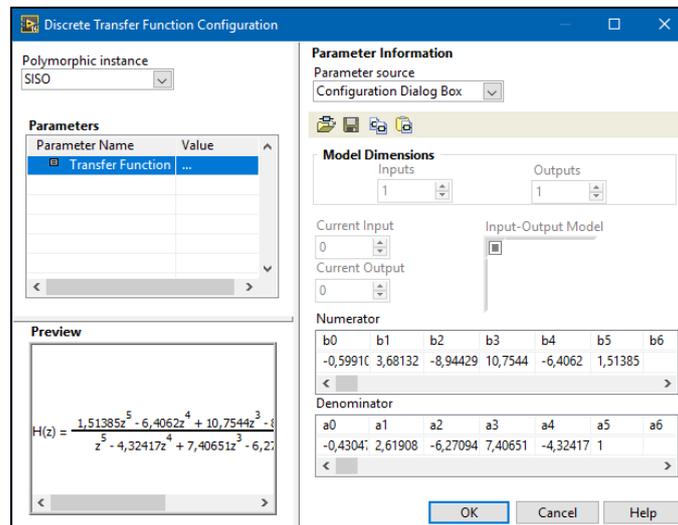


Figura 3.22. Ventana de configuración para el controlador discreto.
 Fuente: Autores.
 Elaboración: Autores.

CAPÍTULO IV

4. ANÁLISIS DE RESULTADOS

4.1. Análisis de desempeño de controladores.

En el presente capítulo se presentan y analizan los resultados del comportamiento de los controladores PID y FOPID implementados en el sistema embebido, el análisis se basa en los parámetros temporales y en los índices de desempeño de la respuesta temporal de los controladores. La configuración experimental para la obtención de los resultados obtenidos se muestra en la Figura 4.1, esta configuración está formada por: tarjeta de desarrollo FPGA-RT sbRIO-9651, tarjeta amplificadora de corriente MC-33926, motor DC A-max 26, encoder incremental MR (tipo ML, 1000 ppv), fuente de alimentación dc BK-Precision 1673 con triple salida y finalmente la computadora Toshiba s55-b5157 que se utilizó para la visualización y registro de las señales de respuesta del sistema.

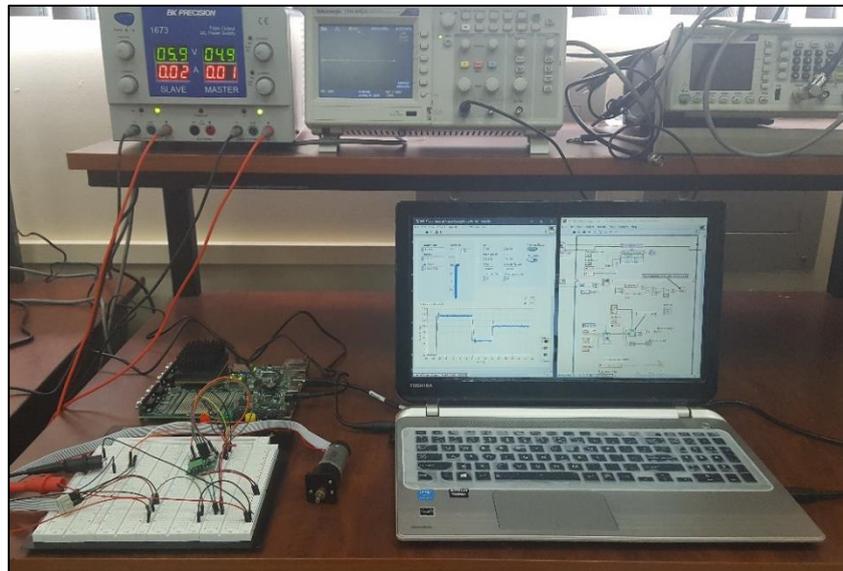


Figura 4.1. Configuración experimental del sistema de control.

Fuente: Autores.

Elaboración: Autores.

Para caracterizar el desempeño de los controladores se obtuvieron los valores de los parámetros temporales y de las integrales de error de la respuesta del sistema de control, la respuesta es obtenida mediante la implementación del sistema en lazo realimentado entre el controlador y el motor DC. Basado en la configuración experimental de la Figura 4.1, los pasos a seguir para la evaluación del sistema de control se citan a continuación:

- Realizar un cambio de escalón del valor de velocidad deseado (SP), el valor inicial es 0 rpm y el valor final es 13.5 rpm, este valor corresponde al 90 % de la velocidad máxima de trabajo del subsistema motor DC con el reductor de

engranaje recto GS 30 A. Esta señal de escalón se la envía a la entrada del sistema de control en lazo cerrado.

- Registrar la señal de salida del sistema de control, esta señal es captada por el encoder incremental.
- Exportar los datos fuente de la señal obtenida a Excel, estos valores corresponden a la señal de consigna, la señal de salida del sistema y el tiempo dado en segundos.
- A partir de la señal adquirida, determinar los parámetros temporales de las respuestas de cada uno de los controladores PID y FOPID: tiempo de establecimiento y el sobrelongación de la señal.
- A partir de la señal adquirida, determinar los índices de error de las respuestas de cada uno de los controladores enteros y fraccionarios: IAE, ITAE, ISE e ITSE.

En el Capítulo 2 se diseñó una cantidad considerable de controladores, 14 controladores PID, 4 controladores PI, 10 controladores FOPID y 4 controladores FOPI, luego se implementaron en hardware todos los controladores, sin embargo, en el presente análisis se seleccionó aquellos controladores con características de desempeño aceptables. A continuación, los resultados de los controladores de orden fraccionario se comparan con los de controladores de orden entero y se determina que controladores presentan las mejores respuestas en la implementación real del sistema de control de velocidad del motor DC.

4.1.1. Análisis y comparación de parámetros temporales.

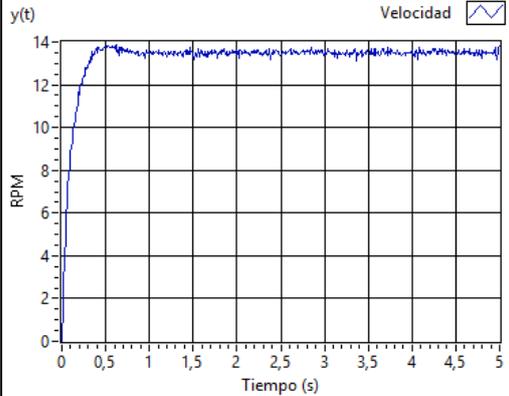
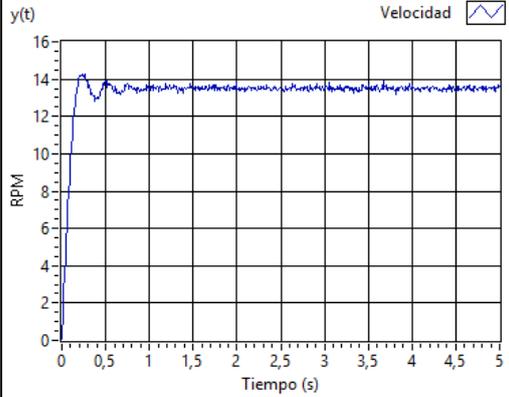
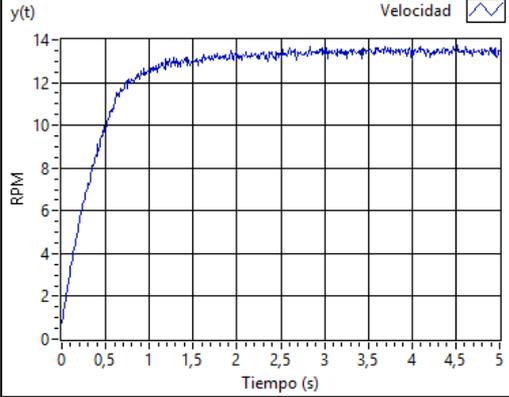
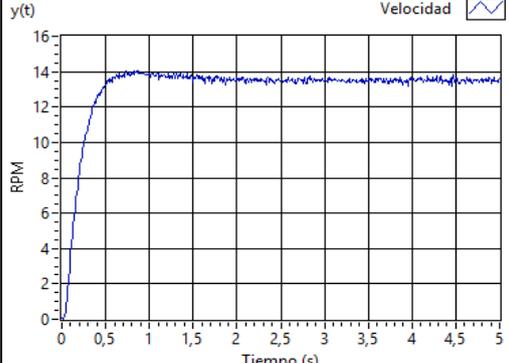
En la presente sección, las respuestas obtenidas de la implementación se organizan en dos grupos de controladores: controladores de orden entero (PI, PID) y controladores de orden fraccionario (FOPI, FOPID). Todos los ensayos se llevaron a cabo para una señal de referencia tipo escalón de amplitud 13.5 rpm. Para el análisis de los resultados se consideran los parámetros temporales más importantes: tiempo de establecimiento y porcentaje de sobrelongación, estos parámetros interesan ya que son la medida de la exactitud de seguimiento entre la señal de consigna y la señal de salida del sistema [55].

4.1.1.1. Parámetros temporales de controladores PID.

Se seleccionan los controladores que mejor respuesta presentan en los resultados de implementación, los criterios de selección fueron: tiempo de establecimiento menor a dos segundos ($< 2\text{seg}$) y el porcentaje de sobrelongación menor al siete por ciento ($< 7\%$). A continuación, en la Tabla 4.1 se muestra las señales de respuesta de los

controladores PID implementados que cumplen con los requerimientos establecidos en un sistema embebido, ante un cambio de escalón de 0 a 13.5 rpm.

Tabla 4.1: Señales de respuesta de los controladores PID.

Respuesta del sistema de control	Descripción del sistema de control
	<ul style="list-style-type: none"> - Ajuste de controlador: Método de Arrieta Orozco. - Índice de rendimiento mínimo: IAE. - Funcionamiento: Regulador. - Parámetros: <ul style="list-style-type: none"> $K_p = 12.3674$ $K_i = 6247.17$ $K_d = 0.000307747$
	<ul style="list-style-type: none"> - Ajuste de controlador por método de Murrill. - Índice de rendimiento mínimo: IAE. - Funcionamiento: Regulador. - Parámetros: <ul style="list-style-type: none"> $K_p = 8.49973$ $K_i = 64617.1$ $K_d = 0.00004269$
	<ul style="list-style-type: none"> - Ajuste de controlador: método de Rivera. - Regla de ajuste robusto. - Parámetros: <ul style="list-style-type: none"> $K_p = 7.44593$ $K_i = 619.625$ $K_d = 0.0000905849$
	<ul style="list-style-type: none"> - Ajuste de controlador: método de Arrieta Orozco. - Regla de ajuste: Filtrado derivativo. - Parámetros: <ul style="list-style-type: none"> $K_p = 2.2354$ $K_i = 2191.5686$ $K_d = 0.0001143$

Fuente: Autores.

Elaboración: Autores

4.1.1.2. Parámetros temporales de controladores PI.

Se realiza el mismo procedimiento de evaluación experimental realizado en los controladores PID para la obtención de los parámetros temporales de los controladores PI. En la Tabla 4.2 se muestra las señales de respuesta de los controladores PI implementados en un sistema embebido, ante un cambio de escalón de 0 a 13.5 rpm.

Tabla 4.2: Señales de respuesta de los controladores PI.

	<ul style="list-style-type: none"> - Ajuste de controlador: Método de Ziegler-Nichols. - Parámetros: <ul style="list-style-type: none"> • $K_p = 8.6996$ • $K_i = 107220$
	<ul style="list-style-type: none"> - Ajuste de controlador: Método de Cohen-Coon. - Parámetros: <ul style="list-style-type: none"> • $K_p = 443.324$ • $K_i = 5487737.484$
	<ul style="list-style-type: none"> - Ajuste de controlador: Método de Chien. - Parámetros: <ul style="list-style-type: none"> • $K_p = 49.6628$ • $K_i = 509550$

Fuente: Autores.
Elaboración: Autores

El tiempo de establecimiento es el tiempo requerido por el sistema para que el valor de la variable de salida se mantenga en un rango de variación del $\pm 2\%$ del valor deseado, en la Tabla 4.3 se presenta los valores obtenidos para el porcentaje de sobrelongación

y el tiempo de establecimiento de los controladores de orden entero PID y PI. De esta tabla se determina que los controladores PI presentan mejores resultados en comparación con los controladores PID.

Tabla 4.3: Parámetros temporales de controladores de orden entero.

	Método	Sobrelongación (%)	Tiempo de establecimiento (ms)
PID	Arrieta Orozco-IAE	2.947	800
	Murrill-IAE	5.895	680
	Rivera-Robusto	0.000	28700
	Arrieta Orozco-Filtrado	4.275	1550
PI	Ziegler-Nichols	5.530	870
	Cohen-Coon	0.000	260
	Chien-Regulador	0.000	180

Fuente: Autores.

Elaboración: Autores

Se concluye que los controladores de orden entero con mejores respuestas de los parámetros temporales son los controladores PI sintonizados en base a los métodos de Cohen-Coon y Chien, estos controladores PI presentan mejores resultados con respecto a los controladores PID.

4.1.1.3. Parámetros temporales de controladores FOPID.

A continuación, se muestran las respuestas temporales experimentales del PID de orden fraccionario de acuerdo a los diferentes métodos propuestos como se indica en las Tablas 4.4 y 4.5. En estas tablas se muestra las señales de respuesta de los controladores FOPID implementados en un sistema embebido, ante un cambio de escalón de 0 a 13.5 rpm.

Se observa en la Tabla 4.4 los controladores FOPID del método de Ziegler-Nichols, basados en los cuatro índices de desempeño (IAE, ISE, ITAE, ISE). Mostrando una variación entre las respuestas transitorias de cada uno de los controladores sobre la planta del motor DC.

Tabla 4.4: Señales de respuestas de los controladores FOPID.

	<ul style="list-style-type: none"> - Método de sintonización: Ziegler-Nichols - Aproximación: Nelder-Mead - Índice de desempeño: IAE <p>Parámetros:</p> <ul style="list-style-type: none"> • $K_p= 1.3437$ • $K_i= 99.9797$ • $K_d= 0.00091695$ • $\lambda= 0.89826$ • $\mu= 0.80168$
	<ul style="list-style-type: none"> - Método de sintonización: Ziegler-Nichols - Aproximación: Nelder-Mead - Índice de desempeño: ISE - Parámetros: <ul style="list-style-type: none"> • $K_p= 1.1944$ • $K_i= 98.0979$ • $K_d= 0.0013562$ • $\lambda= 0.9461$ • $\mu= 0.73498$
	<ul style="list-style-type: none"> - Método de sintonización: Ziegler-Nichols - Aproximación: Nelder-Mead - Índice de desempeño: ITSE - Parámetros: <ul style="list-style-type: none"> • $K_p= 1.0657$ • $K_i= 99.0391$ • $K_d= 0.0014063$ • $\lambda= 0.99053$ • $\mu= 0.75225$
	<ul style="list-style-type: none"> - Método de sintonización: Ziegler-Nichols - Aproximación: Nelder-Mead - Índice de desempeño: ITAE - Parámetros: <ul style="list-style-type: none"> • $K_p= 1.3419$ • $K_i= 99.9806$ • $K_d= 0.0008107$ • $\lambda= 0.89833$ • $\mu= 0.80153$

Fuente: Autores.

Elaboración: Autores.

Se muestra en la Tabla 4.5 las respuestas temporales experimentales del PID de orden fraccionario de los métodos de Cohen-Coon y FMINCON ante un cambio de escalón de 0 a 13.5 rpm implementado en un sistema embebido.

Tabla 4.5: Señales de respuestas de los controladores FOPID.

	<ul style="list-style-type: none"> - Método de sintonización: Cohen-Coon - Aproximación: Nelder-Mead - Índice de desempeño: ITAE - Parámetros: <ul style="list-style-type: none"> • $K_p = 1.0796$ • $K_i = 99.9957$ • $K_d = 0.0031903$ • $\lambda = 0.98234$ • $\mu = 0.67508$
	<ul style="list-style-type: none"> - Aproximación: FMINCON INTERIOR-POINT - Índice de desempeño: ISE - Parámetros: <ul style="list-style-type: none"> • $K_p = 2.3248$; • $K_i = 98.9683$ • $K_d = 0.00013383$ • $\lambda = 0.90374$ • $\mu = 0.80186$
	<ul style="list-style-type: none"> - Aproximación: FMINCON INTERIOR-POINT - Índice de desempeño: IAE - Parámetros: <ul style="list-style-type: none"> • $K_p = 2.3248$ • $K_i = 98.9683$ • $K_d = 0.00013383$ • $\lambda = 0.90374$ • $\mu = 0.80186$

Fuente: Autores.

Elaboración: Autores.

4.1.1.4. Parámetros temporales de controladores FOPI.

El mismo procedimiento aplicado para el PID de orden fraccionario, fue aplicado para desarrollar el controlador FOPI. Los datos experimentales se muestran en la Tabla 4.6.

Tabla 4.6: Señales de respuesta de los controladores FOPI.

	<ul style="list-style-type: none"> - Método de sintonización: Chien-Regulador. - Aproximación: FMINCON INTERIOR-POINT. - Índice de desempeño: ISE. - Parámetros: <ul style="list-style-type: none"> • $K_p = 2.07000$ • $K_i = 299.9985$ • $\lambda = 0.9492$
	<ul style="list-style-type: none"> - Método de sintonización: Chien-Regulador - Aproximación: FMINCON INTERIOR-POINT - Índice de desempeño: ITAE - Parámetros: <ul style="list-style-type: none"> • $K_p = 0.8385$ • $K_i = 240.9854$ • $\lambda = 0.9586$
	<ul style="list-style-type: none"> - Método de sintonización: Ziegler-Nichols. - Aproximación: FMINCON INTERIOR-POINT - Índice de desempeño: ISE - Parámetros: <ul style="list-style-type: none"> • $K_p = 2.11534$ • $K_i = 309.0034$ • $\lambda = 1.1723$

Fuente: Autores.

Elaboración: Autores.

Finalmente se tabula los valores obtenidos de los parámetros temporales, producto de las evaluaciones experimentales reales de los controladores fraccionarios en un sistema embebido, estos resultados se observan en la Tabla 4.7.

Cabe recalcar que los resultados mostrados de la Tabla 4.7 sólo corresponden a los controladores que presentaron características de desempeño aceptables.

Tabla 4.7: Parámetros temporales de controladores de orden fraccionario.

	Método	Sobrelongación (%)	Tiempo de establecimiento (ms)
FOPID	Ziegler-Nichols / IAE	11.70	190
	Ziegler-Nichols / ISE	7.18	220
	Ziegler-Nichols ITSE	14.95	590
	Ziegler-Nichols / ITAE	37.47	910
	Cohen-Coon / ITAE	13.86	480
	FMINCON / ISE	38.35	1490
	FMINCON / IAE	14.517	No existe
FOPI	Chien-Regulador / ISE	16.57	1980
	Chien-Regulador / ITAE	37.8	No existe
	Ziegler-Nichols / ISE	37.16	1790

Fuente: Autores.

Elaboración: Autores.

De la Tabla 4.7 se observa que los dos mejores controladores PID de orden fraccionario implementados en el sistema embebido basado en la plataforma sbRIO-9651, son aquellos que fueron sintonizados por el método de Ziegler-Nichols, con índices de desempeño ISE e IAE.

4.1.2. Evaluación de desempeño basado en índices de error.

La evaluación del desempeño del sistema de control basado en los índices de error se basa en la cuantificación del error acumulado, el error es la diferencia entre el valor deseado y la variable de salida, mientras menor sea el error acumulado, mejor será el desempeño del controlador. Los criterios usados son las siguientes integrales del error:

- Integral del valor absoluto del error (IAE).
- Integral del tiempo por el valor absoluto del error (ITAE).
- Integral del cuadrado del error (ISE).
- Integral del tiempo por el cuadrado del error (ITSE).

En esta sección se muestra los resultados del desempeño de controladores de orden entero y fraccionario obtenidos de la configuración experimental.

4.1.2.1. Índices de desempeño de controladores PID y PI.

Los valores de los índices de desempeño IAE, ITAE, ISE e ITSE obtenidos, se listan en la Tabla 4.8 para los sistemas de control PID y PI. Los valores obtenidos entre los métodos de sintonización de Arrieta-Orozco y Murrill no muestran una diferencia significativa entre ellos, mostrando que el método de Rivera presenta una ventaja sobre los otros dos controladores.

Para controladores PI, los resultados obtenidos para los índices de desempeño se muestran en la Tabla 4.8. Se puede observar que el método de sintonización de Chien, que funciona como regulador, presenta una ventaja sobre los otros dos controladores PI.

Tabla 4.8: Índices de desempeño de los controladores PID y PI de orden entero.

	Método	IAE	ITAE	ISE	ITSE
PID	Arrieta Orozco-IAE	61.2932	156.1873	759.4911	1952.1206
	Murrill-IAE	61.6109	156.2342	766.8145	1953.1601
	Rivera-Robusto	57.1205	153.3838	680.2216	1888.1384
	Arriera Orozco-Filtrado	60.3382	156.3027	748.0775	1956.5489
PI	Ziegler-Nichols	61.6686	156.2220	767.2643	1952.8111
	Cohen-Coon	62.0188	157.7299	776.7625	1990.7781
	Chien-Regulador	61.5923	156.2752	765.8859	1954.1675

Fuente: Autores.

Elaboración: Autores.

De acuerdo a los valores obtenidos de los índices de desempeño reales de la Tabla 4.8, se muestra que las mejores respuestas pertenecen a los controladores sintonizados por el método de método de Rivera con reglas de ajuste robustas. Sin embargo, el controlador PID por el método de Arrieta Orozco con índice de desempeño IAE presenta valores bajos en los índices de ITAE e ITSE. Por el contrario, el controlador PID del método de Arrieta Orozco con filtrado derivativo muestra valores bajos en los índices de desempeño de IAE e ISE.

4.1.2.2. Índices de desempeño de controladores FOPID Y FOPI.

Los índices de desempeño de los diferentes métodos de sintonización de controladores de orden fraccionario se muestran en la Tabla 4.9.

Tabla 4.9: Índices de desempeño de los controladores PID y PI de orden fraccionario.

	Método	IAE	ITAE	ISE	ITSE
FOPID	Ziegler-Nichols, IAE	62.2225	156.2237	777.4197	1952.7950
	Ziegler-Nichols, ISE	61.9273	156.0065	771.2896	1947.4089
	Ziegler-Nichols, ITSE	62.5108	156.3666	784.3882	1956.5494
	Ziegler-Nichols, ITAE	63.0333	156.4876	802.9420	1960.6085
	Cohen-Coon, ITAE	62.2213	156.2288	778.8858	1953.0947
	FMINCON, ISE	63.1142	156.4029	807.2389	1959.7764
	FMINCON, IAE	62.5747	156.7369	788.9099	1967.5135
FOPI	Chien-Servomecanismo, ISE	62.5436	156.2501	786.8995	1954.3092
	Chien-Servomecanismo, ITAE	63.4061	157.2240	816.6179	1984.8189
	Ziegler-Nichols, ISE	62.8744	156.0980	803.5343	1953.5004

Fuente: Autores.

Elaboración: Autores.

A partir de la Tabla 4.9, se observa que los dos controladores que presentan un mejor rendimiento son los controladores sintonizados por el método de Ziegler-Nichols con índices de desempeño de IAE e ISE.

CONCLUSIONES

- Con el objetivo de evaluar el desempeño entre las funciones de control PID clásico y FOPID se diseñaron y simularon 18 controladores PID y 14 controladores FOPID, tomando como referencia metodologías de diseño plateadas por diversos autores. Para el diseño y simulación de los controladores se utilizó el Módulo *Control Design & Simulation Module* para LabVIEW y el Toolbox *FOMCOM* para Matlab.
- En base a los resultados de la simulación en lazo cerrado de los controladores PID se concluye que los de mejor desempeño son los controladores PID basados en los métodos de Chien-Hrones-Reswick (Regulador) y Murrill (ITAE). Con respecto a las simulaciones de los controladores FOPID se concluye que aquellos de mejor desempeño están basados en los métodos Ziegler-Nichols-ISE y Cohen-Coon-ITAE.
- Se implementaron 7 controladores PID clásico y 10 controladores FOPID en un sistema embebido completamente autónomo y de bajo costo. La implementación se la realizó en una plataforma embebida FPGA-Procesador (sbRIO-9651) y en el lenguaje LabVIEW FPGA y RT. La plataforma de desarrollo está basada en la combinación de un FPGA Xilinx Artix-7 y un Procesador ARM Cortex-A9 (plataforma sbRIO-9651).
- La configuración experimental para evaluar los controladores embebidos estuvo formada por: tarjeta de desarrollo FPGA-RT NI sbRIO-9651, motor DC Maxon A-max 26, tarjeta amplificadora de corriente Pololu MC-33926, encoder incremental Maxon MR (tipo ML, 1000 ppv), fuente de alimentación BK-Precision 1673 y finalmente un computador Toshiba s55-b5157 que se utilizó para la visualización y registro de las señales de salida del sistema.
- En base a la evaluación experimental y comparación de los controladores en base a los parámetros temporales: tiempo de establecimiento, porcentaje de sobrelongación; y a los índices de error: IAE, ITAE, ISE e ITSE. Los controladores embebidos de mejor desempeño fueron: Controlador FOPID basado en el método de sintonización Ziegler-Nichols-ISE, aproximación Nelder-Mead, los parámetros obtenidos fueron $T_s=0.22s$ y $P.O.=7.18\%$; y, controlador PI basado en el método Chien-Regulador, para este controlador los parámetros obtenidos fueron $T_s=0.18s$, $P.O.=0\%$.

BIBLIOGRAFÍA

- [1] C. N. p. l. l. d. Discapacidades. (2017). *Estadísticas de Discapacidad*. Available: <http://www.consejodiscapacidades.gob.ec/estadistica/index.html>
- [2] C. A. Calderon, C. Ramirez, V. Barros, and G. Punin, "Design and Deployment of Grasp Control System applied to robotic hand prosthesis," *IEEE Latin America Transactions*, vol. 15, pp. 181-188, 2017.
- [3] P. Shah and S. Agashe, "Review of fractional PID controller," *Mechatronics*, vol. 38, pp. 29-41, 2016.
- [4] K. J. Åström and T. Häggglund, "PID controllers: theory, design, and tuning," 1995.
- [5] M. S. Tavazoei and M. Haeri, "A note on the stability of fractional order systems," *Mathematics and Computers in Simulation*, vol. 79, pp. 1566-1576, 2009.
- [6] D. Valério and J. S. Costa, "Introduction to single-input, single-output fractional control," *IET control theory & applications*, vol. 5, pp. 1033-1057, 2011.
- [7] D. Valerio and J. S. da Costa, "A review of tuning methods for fractional PIDs," in *4th IFAC Workshop on Fractional Differentiation and its Applications, FDA*, 2010.
- [8] Y. Luo, H. Chao, L. Di, and Y. Chen, "Lateral directional fractional order (PI) π control of a small fixed-wing unmanned aerial vehicles: controller designs and flight tests," *IET control theory & applications*, vol. 5, pp. 2156-2167, 2011.
- [9] S. Das, I. Pan, S. Das, and A. Gupta, "Improved model reduction and tuning of fractional-order PI λ D μ controllers for analytical rule extraction with genetic programming," *ISA transactions*, vol. 51, pp. 237-261, 2012.
- [10] L.-Y. Chang and H.-C. Chen, "Tuning of fractional PID controllers using adaptive genetic algorithm for active magnetic bearing system," *WSEAS Trans. Syst*, vol. 8, pp. 158-167, 2009.
- [11] M. P. Aghababa, "Optimal design of fractional-order PID controller for five bar linkage robot using a new particle swarm optimization algorithm," *Soft Computing*, vol. 20, pp. 4055-4067, 2016.
- [12] Z. Li, L. Liu, S. Dehghan, Y. Chen, and D. Xue, "A review and evaluation of numerical tools for fractional calculus and fractional order controls," *International Journal of Control*, pp. 1-17, 2016.
- [13] V. Mehra, S. Srivastava, and P. Varshney, "Fractional-order PID controller design for speed control of DC motor," in *Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference on*, 2010, pp. 422-425.
- [14] J. Viola, L. Angel, and J. Sebastian, "Design and robust performance evaluation of a fractional order PID controller applied to a DC motor," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, pp. 304-314, 2017.
- [15] S. H. HosseinNia, I. Tejado, and B. M. Vinagre, "Fractional-order reset control: Application to a servomotor," *Mechatronics*, vol. 23, pp. 781-788, 2013.
- [16] C. I. Muresan, S. Folea, G. Mois, and E. H. Dulf, "Development and implementation of an FPGA based fractional order controller for a DC motor," *Mechatronics*, vol. 23, pp. 798-804, 2013.
- [17] K. Rana, V. Kumar, N. Mitra, and N. Pramanik, "Implementation of fractional order integrator/differentiator on field programmable gate array," *Alexandria Engineering Journal*, vol. 55, pp. 1765-1773, 2016.
- [18] E. Midhun and S. K. TK, "LabVIEW based real time implementation of Fractional Order PID controller for a magnetic levitation system," in *Power Electronics, Intelligent Control and Energy Systems (ICPEICES), IEEE International Conference on*, 2016, pp. 1-6.
- [19] Y. Jin, Y. Luo, C. Wang, and Y. Chen, "LabVIEW based experimental validation of fractional order motion controllers," in *Control and Decision Conference, 2009. CCDC'09. Chinese*, 2009, pp. 323-328.

- [20] C. Calderon-Cordova, C. Ramírez, V. Barros, P. A. Quezada-Sarmiento, and L. Barba-Guamán, "EMG signal patterns recognition based on feedforward Artificial Neural Network applied to robotic prosthesis myoelectric control," in *Future Technologies Conference (FTC)*, 2016, pp. 868-875.
- [21] E. Maxon, "Programa 2016/17: Micromotores y sistemas de alta precisión," M. Motor, Ed., ed, 2016.
- [22] Pololu. (2016). *MC33926 Motor Driver Carrier*. Available: <https://www.pololu.com/product/1212>
- [23] N. Instruments. (2015). *Embedded Control and Monitoring Using LabVIEW Course Manual (Enero 2015 ed.)*.
- [24] N. Instruments. (2016). *LabVIEW Real-Time y FPGA* Available: http://www.ni.com/gate/gb/GB_EVALTLKTEMBDES/US
- [25] K. Ogata, *Ingeniería de control moderna*, Quinta ed.: Pearson Education, 2010.
- [26] K. J. Åström and T. Hägglund, *Control PID avanzado*: Pearson, Madrid, 2009.
- [27] B. M. Vinagre and C. A. Monje, "Introducción al control fraccionario," *Revista Iberoamericana de Automática e Informática Industrial*, vol. 3, pp. 5-23, 2006.
- [28] O. D. Hernandez Arboleda, "Metodología para modelar y controlar un sistema de combustion utilizando calculo fraccional," 2014.
- [29] O. A. Orozco and V. M. A. Ruiz, "Sintonización de controladores PI y PID utilizando los criterios integrales IAE e ITAE," *Journal of Tropical Engineering*, vol. 13, 2011.
- [30] K. Ogata, "Modelado matemático de sistemas de control," *Ingeniería de control moderna*, vol. 5, 2010.
- [31] A. Tepljakov, *Fractional-order modeling and control of dynamic systems*: Springer, 2017.
- [32] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," *International Journal of Microelectronics and Computer Science*, vol. 2, pp. 51-62, 2011.
- [33] M. Romero Hortelano, "Control predictivo de orden fraccionario," 2010.
- [34] D. P. M. de Oliveira Valério, "Ninteger v. 2.3 Fractional control toolbox for MatLab," *Universidade tecnica de lisboa instituto superior tecnico*, 2005.
- [35] H. MARTÍNEZ, "Análisis modelado, simulación en computadora del motor de corriente directa tipo serie," *Universidad tecnológica de la mixteca*, 2009.
- [36] A. A. Mahfouz, M. Mohammed, and F. A. Salem, "Modeling, Simulation and Dynamics Analysis Issues of Electric Motor, for Mechatronics Applications, Using Different Approaches and Verification by MATLAB/Simulink," *International Journal of Intelligent Systems and Applications*, vol. 5, p. 39, 2013.
- [37] E. D. Ruiz-Rojas, J. L. Vázquez-González, R. Alejos-Palomares, A. Z. Escudero-Uribe, and J. R. Mendoza-Vázquez, "Mathematical model of a linear electric actuator with prosthesis applications," in *Electronics, Communications and Computers, 2008. CONIELECOMP 2008, 18th International Conference on*, 2008, pp. 182-186.
- [38] M. R. Reis, B. Alvarenga, W. G. da Silva, C. A. Ganzaroli, W. P. Calixto, W. R. de Araujo, et al., "Heuristic and deterministic strategies applied on a PID controller tuning for speed control of a DC motor," in *Environment and Electrical Engineering (EEEIC), 2013 13th International Conference on*, 2013, pp. 223-228.
- [39] G. Garcés, L. Angel, and J. L. Rincón Gaviria, "Control PID para el control de velocidad de un motor DC," 2014.
- [40] P. Meshram and R. G. Kanojiya, "Tuning of PID controller using Ziegler-Nichols method for speed control of DC motor," in *Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on*, 2012, pp. 117-122.
- [41] A. O'Dwyer, *Handbook of PI and PID controller tuning rules*: World Scientific, 2009.
- [42] M. Shahrokhi and A. Zomorodi, "Comparison of PID controller tuning methods," *Department of Chemical & Petroleum Engineering Sharif University of Technology*, 2013.

- [43] C. A. Monje, Y. Chen, B. M. Vinagre, D. Xue, and V. Feliu-Batlle, *Fractional-order systems and controls: fundamentals and applications*: Springer Science & Business Media, 2010.
- [44] E. A. Pilotta, "El método de Nelder-Mead para minimización irrestricta sin derivadas," *Revista de Educación Matemática*, vol. 17, 2002.
- [45] Y. Chen, I. Petras, and D. Xue, "Fractional order control-A tutorial," in *American Control Conference, 2009. ACC'09.*, 2009, pp. 1397-1411.
- [46] C. group. (2010). *Crone toolbox*. Available: http://archive.ims-bordeaux.fr/CRONE/toolbox/pages/accueilSITE.php?guidPage=home_page
- [47] B. M. Vinagre, V. Feliu-Batlle, and I. Tejado, "Control fraccionario: fundamentos y guía de uso," *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 13, pp. 265-280, 7// 2016.
- [48] R. Vilanova and V. M. Alfaro, "Control PID robusto: Una visión panorámica," *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 8, pp. 141-158, 2011.
- [49] L. Caisaguano, V. Gabriela, P. Anchatipán, and A. Darwin, "Diseño e implementación de un algoritmo de control predictivo para una planta de flujo utilizando un controlador de automatización programable para el Laboratorio de Redes Industriales y Control de Procesos de la Universidad de las Fuerzas Armadas-ESPE Extensión Latacunga," Universidad de las Fuerzas Armadas ESPE Extensión Latacunga. Carrera de Ingeniería en Electrónica e Instrumentación., 2015.
- [50] N. Instruments. (2016). *Ocho maneras de utilizar un módulo digital en un sistema CompactRIO*. Available: <http://www.ni.com/white-paper/14549/en/>
- [51] N. Instruments. (2016). *Compile Faster with the LabVIEW FPGA Compile Cloud Service*. Available: <http://www.ni.com/white-paper/52328/en/>
- [52] M. A. Bevilaqua, A. Nied, and J. de Oliveira, "Labview FPGA FOC implementation for synchronous Permanent Magnet Motor Speed Control," in *Industry Applications (INDUSCON), 2014 11th IEEE/IAS International Conference on*, 2014, pp. 1-8.
- [53] M. Narvidas. (2016). *La lectura de valores analógicos y PWM con LabVIEW FPGA*. Available: <https://www.allaboutcircuits.com/technical-articles/reading-analog-values-and-pwm-with-labview-fpga/>
- [54] C. I. Muresan, G. Mois, S. Folea, and C. Ionescu, "Alternative implementations of a fractional order control algorithm on FPGAs," in *Reconfigurable Computing and FPGAs (ReConFig), 2013 International Conference on*, 2013, pp. 1-6.
- [55] R. C. B. Dorf, R. H. R. C. Dorf, and R. H. Bishop, *Sistemas de control moderno*: Pearson Educación, 2005.