



Comparativa de control automático PID vs control por redes neuronales artificiales

Comparative analysis of PID control vs. Artificial neural networks control

Carolina Michelle Suárez Pozo¹

¹Universidad Estatal Península de Santa Elena.
Facultad de Sistemas y Telecomunicaciones.
Electrónica y Automatización.
La Libertad - Santa Elena. Ecuador.
ORCID: <https://orcid.org/0009-0006-0161-6327>
Correo: carolina.suarezpozo@upse.edu.ec

Kevin Alexis Pozo Catuto²

²Universidad Estatal Península de Santa Elena.
Facultad de Sistemas y Telecomunicaciones.
Electrónica y Automatización.
La Libertad - Santa Elena. Ecuador.
ORCID: <https://orcid.org/0009-0009-6684-6205>
Correo: kevin.pozocatuto@upse.edu.ec

Alex Joel Mendoza Loo³

³Universidad Estatal Península de Santa Elena.
Facultad de Sistemas y Telecomunicaciones.
Electrónica y Automatización.
La Libertad - Santa Elena. Ecuador.
ORCID: <https://orcid.org/0009-0009-6483-4972>
Correo: alex.mendozaloor@upse.edu.ec

Marcos Guillermo Cruz Yagual⁴

⁴Universidad Estatal Península de Santa Elena.
Facultad de Sistemas y Telecomunicaciones.
Electrónica y Automatización.
La Libertad - Santa Elena. Ecuador.
ORCID: <https://orcid.org/0009-0002-4558-0128>
Correo: marcos.cruzyagual@upse.edu.ec

Carlos Alberto Saldaña Enderica⁵

⁵Universidad Estatal Península de Santa Elena.
Facultad de Sistemas y Telecomunicaciones.
Electrónica y Automatización.
La Libertad - Santa Elena. Ecuador.
ORCID: <https://orcid.org/0000-0002-5867-1734>
Correo: csaldana@upse.edu.ec

*Autor para correspondencia: carolina.suarezpozo@upse.edu.ec





Resumen

Este trabajo de investigación aborda la comparación entre dos enfoques de control, el Control Proporcional-Integral-Derivativo (PID) y Redes Neuronales Artificiales (RNA), aplicados a un motor de corriente continua (DC).

En la implementación del control PID, se emplea MATLAB y Simulink para diseñar y ajustar los parámetros del controlador, asegurando un seguimiento óptimo de la velocidad y posición del motor. Por otro lado, se utiliza una aproximación de Redes Neuronales Artificiales para modelar y controlar el sistema, permitiendo una adaptación dinámica y no lineal.

Los resultados obtenidos proporcionarán una comprensión profunda de las ventajas y limitaciones de cada método de control en el contexto específico del motor DC. Esta investigación contribuye al avance en el campo de la electrónica y automatización al proporcionar información valiosa para la selección y aplicación efectiva de estrategias de control en sistemas electromecánicos.

Palabras claves: Control PID; Redes Neuronales Artificiales; motor de corriente continua; MATLAB; Simulink; diseño de controladores; sistemas electromecánicos.

Abstract

This research work addresses the comparison of two control approaches: Proportional-Integral-Derivative (PID) control and Artificial Neural Networks (ANNs), as applied to a direct current (DC) motor. The primary objective is to evaluate and contrast the performance of these methods in terms of precision, stability, and efficiency.

In the implementation of PID control, MATLAB and Simulink are utilized to design and fine-tune the controller parameters, ensuring optimal tracking of the motor's speed and position. Conversely, the Artificial Neural Network approach is employed to model and control the system, facilitating dynamic and non-linear adaptation.

The results obtained will provide a comprehensive understanding of the advantages and limitations of each control method in the specific context of the DC motor. This research contributes to advancements in the fields of electronics and automation by offering valuable insights for the effective selection and application of control strategies in electromechanical systems.

Keywords: PID Control, Artificial Neural Networks, direct current motor, MATLAB, Simulink, controller design, electromechanical systems.

I. Introducción

El motor de corriente continua o motor de corriente directa es una máquina con la capacidad de convertir energía eléctrica en movimiento o trabajo mecánico a través de fuerzas electromagnéticas. [1]





Un motor con encoder es la integración de una máquina y un equipo encoder con capacidades para transformar el movimiento mecánico en acciones digitales, las cuales pueden ser descifradas por el sistema de electrónica de control integrado. Su principio de funcionamiento consiste en una retroalimentación de circuito cerrado que busca activar la velocidad y la posición, que servirán para la toma de decisiones en cuanto a los requisitos operacionales del dispositivo.

El diseño de sistemas de control para motores equipados con encoder desempeña un papel crucial, en donde, su simulación nos posibilita verificar de antemano la viabilidad de la aplicación práctica de ciertos métodos que se desea implementar. Cuanto más detallado sea el modelo por simular, incluyendo las funciones de transferencia de cada componente eléctrico, electrónico y mecánico, los resultados obtenidos durante la simulación serán más fieles a la realidad.

Para poder realizar la función de transferencia de este sistema, se realizó un modelado matemático del motor encoder, implementando un controlador de velocidad de PID junto a RNA.

La configuración de la red neuronal es un punto fundamental para que una aplicación produzca buenos resultados. Si los datos de entrenamiento son correctos, pero la arquitectura de la red no es la adecuada para la aplicación, la red neuronal no se comportará como se espera.[2]

En este artículo se muestra como las redes neuronales replican el comportamiento de un controlador PID para poder controlar las diferentes velocidades de un motor DC.

II. Materiales y métodos

Modelado del motor DC y controlador PID

A. Parámetros del motor

En primer lugar, se generó un modelo representado como una función de transferencia que abarca todas las componentes eléctricas y mecánicas del sistema físico esenciales para la ejecución del controlador. Este modelo inicial no incluyó las no linealidades y asumió que el motor estaba libre de carga. Para simular el modelo del motor, se requiere conocer los valores numéricos de cada una de las variables presentes en el diagrama de bloques.

Tabla 1. Parámetros del motor.

Parámetro	Símbolo	Valor	Unidad (SI)
Momento de Inercia	J	0.00567	$Kg \cdot m^2$
Constante de Fricción	B	0.000931	$N \cdot m \cdot s$
Constante de Fuerza Electromotriz	Ka	0.509	$N \cdot m/A$
Constante de par del Motor	Km	0.509	$V/rad \cdot s$
Resistencia de Armadura	Ra	7.101	Ω
Inductancia eléctrica	La	0.0034	H





Los parámetros del motor que impulsará el robot móvil de tipo diferencial se obtuvieron a partir de la información proporcionada en el datasheet del Motor DC EMG30

B. Circuito equivalente del motor DC

Tomando como punto de partida las ecuaciones en función del tiempo que describen nuestro sistema, se realiza la transformación al dominio de los números complejos mediante la transformada de Laplace. Durante este proceso, se llevan a cabo operaciones algebraicas para derivar las ecuaciones resultantes.

$$I(s) = \frac{U(s) - K_a \cdot W(s)}{L_a \cdot s + R_a}$$

$$T(s) = K_m \cdot I(s)$$

$$W(s) = \frac{T(s)}{J \cdot s + B}$$

$J \cdot s + B$

Con ayuda de MATLAB se generó un código para establecer el valor de los parámetros del motor para luego en Simulink modelar el motor DC:

```
%Parámetros del motor
J=0.00567; %Momento de inercia
B=0.000931; %Constante de fricción
Km=0.509; %Constante de par de motor
Ka=0.509; %Constante de fuerza electromotriz
Ra=7.101; %Resistencia de armadura
La=0.0034; %Inductancia eléctrica
```

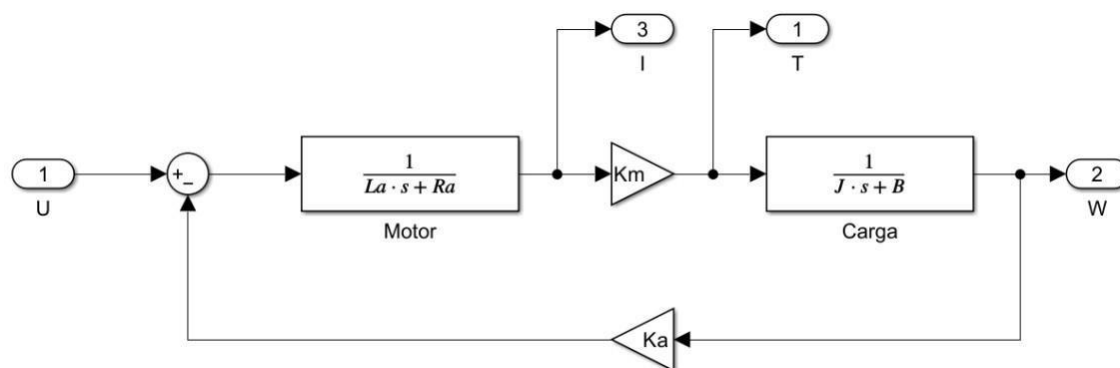


Figura 1. Diagrama de bloques de Simulink del modelo matemático del motor DC

A partir del diagrama de bloques previo, es posible inferir la función de transferencia en lazo abierto del sistema al establecer la relación entre la velocidad angular y el voltaje de entrada.

$$\frac{W(s)}{U(s)} = \frac{K_m}{(J \cdot L_a)s^2 + (B \cdot L_a + R_a \cdot J)s + (B \cdot R_a + K_a \cdot K_m)}$$





C. Modelado matemático y control PID

Un controlador PID (Proporcional, Integral, Derivativo) para un motor DC de velocidad es un dispositivo que ajusta la entrada de voltaje al motor con el objetivo de mantener la velocidad del motor en un valor deseado.

Proporcional (P): Proporciona una respuesta proporcional al error actual, es decir, la diferencia entre la velocidad deseada y la velocidad actual del motor.

Integral (I): Integra el error a lo largo del tiempo para abordar el acumulado de desviaciones persistentes, corrigiendo lentamente las discrepancias a largo plazo.

Derivativo (D): Responde a la tasa de cambio del error, anticipándose a posibles oscilaciones y contribuyendo a la estabilidad del sistema.

Existen diversas variantes de controladores PID, cuya selección depende de las características específicas del proceso a regular. En este contexto, se llevará a cabo la caracterización de un motor mediante la implementación de un controlador PID convencional. []

Al ajustar adecuadamente los parámetros P, I y D, un controlador PID puede proporcionar un control eficiente y preciso de la velocidad del motor, mejorando la capacidad de respuesta y reduciendo el error de seguimiento.

Para obtener nuestra función de transferencia de nuestro motor DC usamos el siguiente código MATLAB:

```
%Función de transferencia G(s)
num = [Km];
den = [J*La (f*La+Ra*J) (f*Ra+Ka*Km)];
G = tf(num,den)
```

$$G = \frac{0.509}{1.928e-05 s^2 + 0.04027 s + 0.2657}$$

Con esta función de transferencia podemos ver la respuesta del sistema ante una entrada escalón y podremos crear un controlador PID para compensar nuestra planta.

Para esto con ayuda de Simulink conectaremos un bloque Step, que será nuestra entrada; un bloque PID(s), que será nuestro controlador; Scope, para mostrarnos el comportamiento antes y después de compensar.

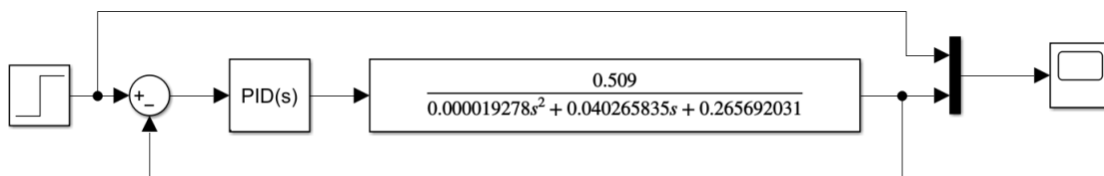


Figura 2. Diagrama de bloques de Simulink de la función de transferencia del motor DC.





Con ayuda del GUI PID Tuner de MATLAB podremos ver el comportamiento del sistema y ajustarlo a nuestro requerimiento. Así de esta manera establecerá automáticamente las constantes del controlador PID.

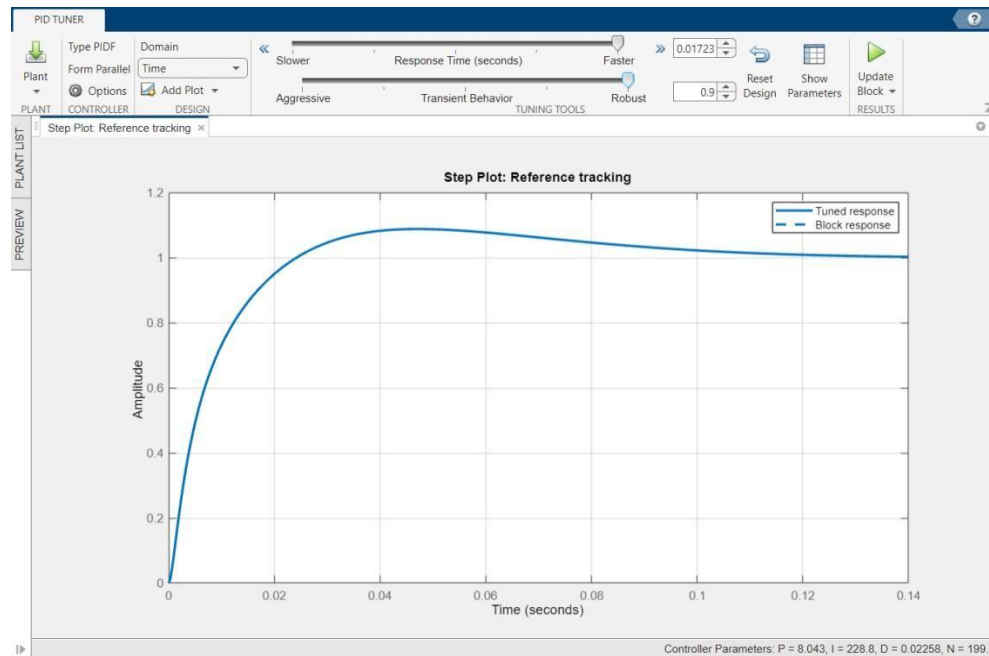


Figura 3. PID Tuner para compensar el sistema

Luego se configura los bloques de tal manera que podamos observar el sistema antes de ser compensado y después de compensar con el controlador PID.

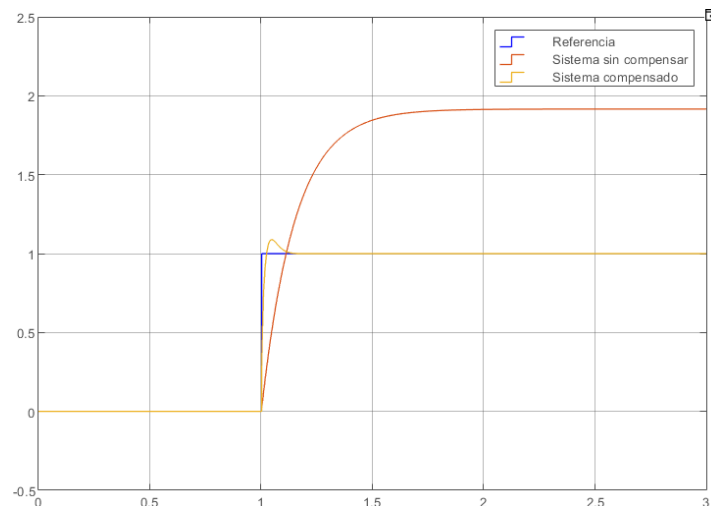


Figura 4. Muestreo del comportamiento de los sistemas ante una señal escalón

De la **Figura 4.** podemos observar que el sistema sin compensar tiene un error alrededor del 80% de la referencia, mientras que el sistema compensado se estabiliza con un error del 0% en milisegundos.





D. Control RNA

Para poder implementar control por Redes Neuronales Artificiales, se optó por usar control por modelo inverso. Este se basa en la idea de obtener un modelo matemático que represente el comportamiento de un sistema y utilizar ese modelo para diseñar un controlador. En lugar de medir las variables de salida del sistema y ajustar el controlador en consecuencia, como se hace en el control clásico, el control de modelo inverso utiliza el modelo matemático inverso del sistema para generar los comandos de entrada del controlador. [4]

El controlador no se basa en las mediciones de las variables del sistema, sino en el modelo matemático, que proporciona una forma mucho más eficiente de control. Además, es especialmente efectivo cuando se trata de sistemas que tienen una dinámica bien definida, con un modelo matemático establecido, y para los que se espera una respuesta rápida del controlador. [5] El control por modelo inverso puede presentar algunas limitaciones en situaciones donde el modelo matemático del sistema no es totalmente preciso y/o en situaciones con errores de medición. Además, el modelo matemático debe ser conocido y medido con precisión antes de poder ser utilizado para diseñar el controlador. En algunos casos esto puede resultar en una disminución del rendimiento del controlador. [6]

Durante el entrenamiento de la red neuronal, se proporcionan ejemplos de entrada-salida para el sistema que se desea controlar, y la red ajusta sus pesos internos para que su salida se acerque lo más posible a los ejemplos de salida deseados. [7]

Una vez que la red neuronal ha sido entrenada, se puede utilizar como controlador en el sistema que se desea controlar. En el caso del control de modelo inverso por RNA, se utiliza el modelo matemático inverso del sistema para generar entradas de la red neuronal, y la salida de la red se utiliza como entrada para el sistema. [8]

De esta manera, la red neuronal actúa como un controlador que ajusta las entradas al sistema en función del modelo matemático para producir la salida deseada. Este enfoque puede ser utilizado para sistemas de control de movimiento y posición, así como en una variedad de otras aplicaciones de control.

La forma en que funciona el control de modelo inverso por RNA es la siguiente:

1. En primer lugar, se genera un modelo matemático inverso del sistema que se desea controlar. El modelo matemático inverso es una expresión matemática que describe la relación entre las entradas del sistema y las salidas deseadas del sistema.
2. Después, se entrena una red neuronal artificial utilizando ejemplos de entrada-salida para el sistema que se desea controlar. Durante el entrenamiento, la red neuronal ajusta sus pesos internos para que su salida se acerque lo más posible a las salidas deseadas del modelo matemático inverso.
3. Una vez que la red neuronal ha sido entrenada, se utiliza el modelo matemático inverso del sistema para generar entradas para la red. La red neuronal procesa las entradas y produce una salida que representa la entrada óptima para el sistema que se desea controlar.
4. La salida de la red neuronal se utiliza como entrada para el sistema. El sistema procesa la entrada y produce una salida correspondiente. Si la salida del sistema no se corresponde con la salida deseada, se utiliza el error entre la salida deseada y la salida real para ajustar





el modelo matemático inverso y la red neuronal. Este proceso se repite hasta que se logre la salida deseada.

- De esta forma, la red neuronal actúa como un controlador que ajusta las entradas al sistema en función del modelo matemático para producir la salida deseada. Es un enfoque poderoso que se utiliza en una variedad de aplicaciones de control. [9]

Uno de los ejemplos para este modelo es el control difuso, que puede ser utilizado para ajustar los parámetros del controlador inverso del modelo en tiempo real, considerando las variaciones en las condiciones de operación o las perturbaciones externas. Esta combinación de técnicas ofrece una solución robusta y versátil para el control de manipuladores de dos eslabones, permitiendo un seguimiento preciso de trayectorias y una respuesta estable frente a condiciones cambiantes. [10] Otro ejemplo es donde un rayo experimenta una perturbación que induce vibraciones no deseadas, el control inverso del modelo calcula la señal de control adecuada para aplicar al sistema. Esta señal de control ajusta activamente los parámetros del sistema, como la fuerza aplicada, para contrarrestar y minimizar las vibraciones. En el caso de un rayo vibratorio amortiguado, el control inverso del modelo puede adaptarse dinámicamente a las condiciones cambiantes del entorno, proporcionando una solución efectiva para estabilizar el rayo y mantener su respuesta dentro de los límites deseados. Este enfoque es particularmente útil en situaciones donde el amortiguamiento del sistema puede variar, permitiendo un control preciso y adaptable de las vibraciones. [11]

Control PID por RNA: El control proporcional, integral y derivativo (PID) es un método de control clásico utilizado para controlar sistemas dinámicos. Sin embargo, en lugar de utilizar los algoritmos clásicos PID, se pueden emplear redes neuronales para ajustar los parámetros de este. Una ventaja del control de modelo inverso por RNA en este caso es que puede funcionar mejor cuando se tienen dinámicas no lineales o variables, en comparación con el control PID convencional. [12]

Se realiza una implementación en Simulink de un nuevo diagrama de bloques que incluye una señal de referencia de entrada, el controlador diseñado y el modelo del motor. Este modelo posibilita la exportación de los datos de entrada y salida del controlador al Workspace de MATLAB a través de las variables P1 y T1. Estas variables son esenciales para llevar a cabo un entrenamiento supervisado en una red neuronal artificial, cuyo diseño se realiza utilizando la interfaz gráfica de usuario NNtool. Su representación se presenta en la **figura 5**.

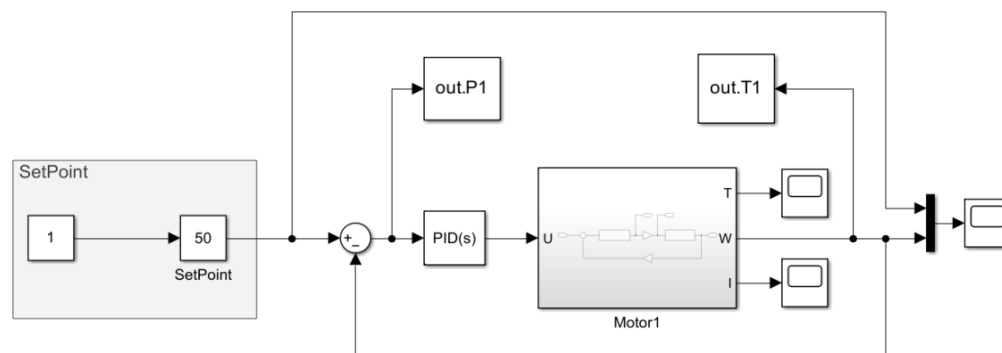
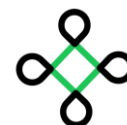


Figura 5. Diagrama de bloques de Simulink para obtener datos para entrenar RNA





Para generar datos en Simulink, fue necesario realizar ajustes en los parámetros de simulación para lograr 6000 datos por simulación. Para obtener esta cantidad de datos, se modificaron parámetros en Simulink, como cambiar la opción del solucionador a un paso fijo de 0.001 segundos. En este caso, la red neuronal operará con pasos de tiempo fijos de 0.001 segundos. Considerando el control PID que se emulará en la red neuronal, se implementó utilizando la herramienta NNtool de MATLAB. En la creación de la red neuronal, se definió la regla de aprendizaje de descenso de gradiente y se especificó la cantidad de épocas (el número de iteraciones realizadas antes de detenerse) durante las cuales se entrenó la red neuronal hasta alcanzar un nivel de error aceptable.

Después de completar el entrenamiento de las redes neuronales artificiales (RNA) utilizando la herramienta NNtool, se llevó a cabo la exportación de dichas redes como bloques de Simulink a través de la opción "export model". Posteriormente, fue posible reemplazar el controlador PID existente por este bloque en el sistema como se puede observar en la **Figura 6**.

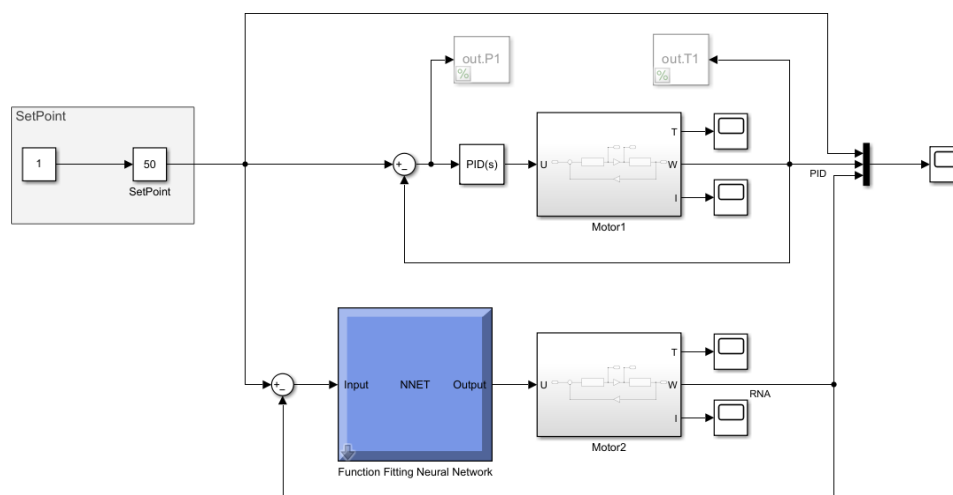


Figura 6. Diagrama de bloques de Simulink del sistema con control PID y RNA

Del diagrama de bloques de la figura 6. simulamos y excitamos el sistema para poder observar el comportamiento como se muestra en la figura 7. Donde se puede apreciar como color azul nuestro Setpoint; de color rojo el sistema controlado por PID; de color amarillo nuestro sistema controlado por RNA.

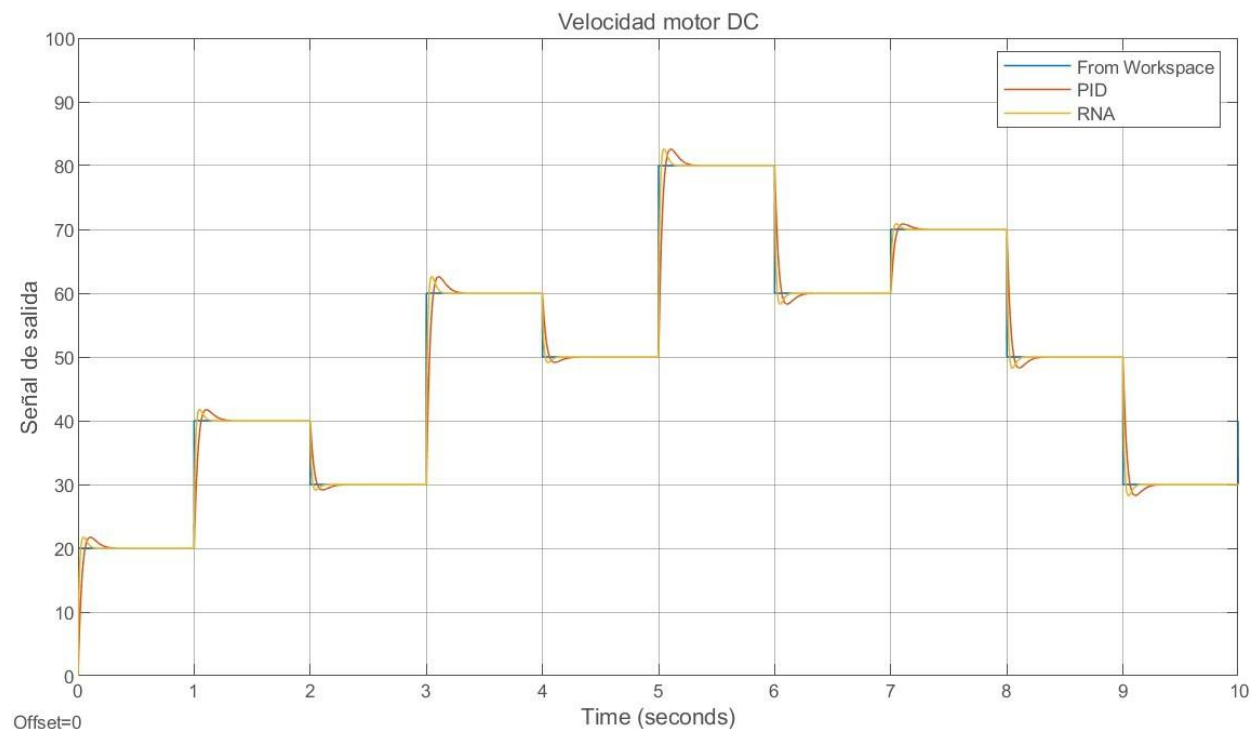


Figura 7. Comparación del comportamiento del sistema con los controladores PID y RNA

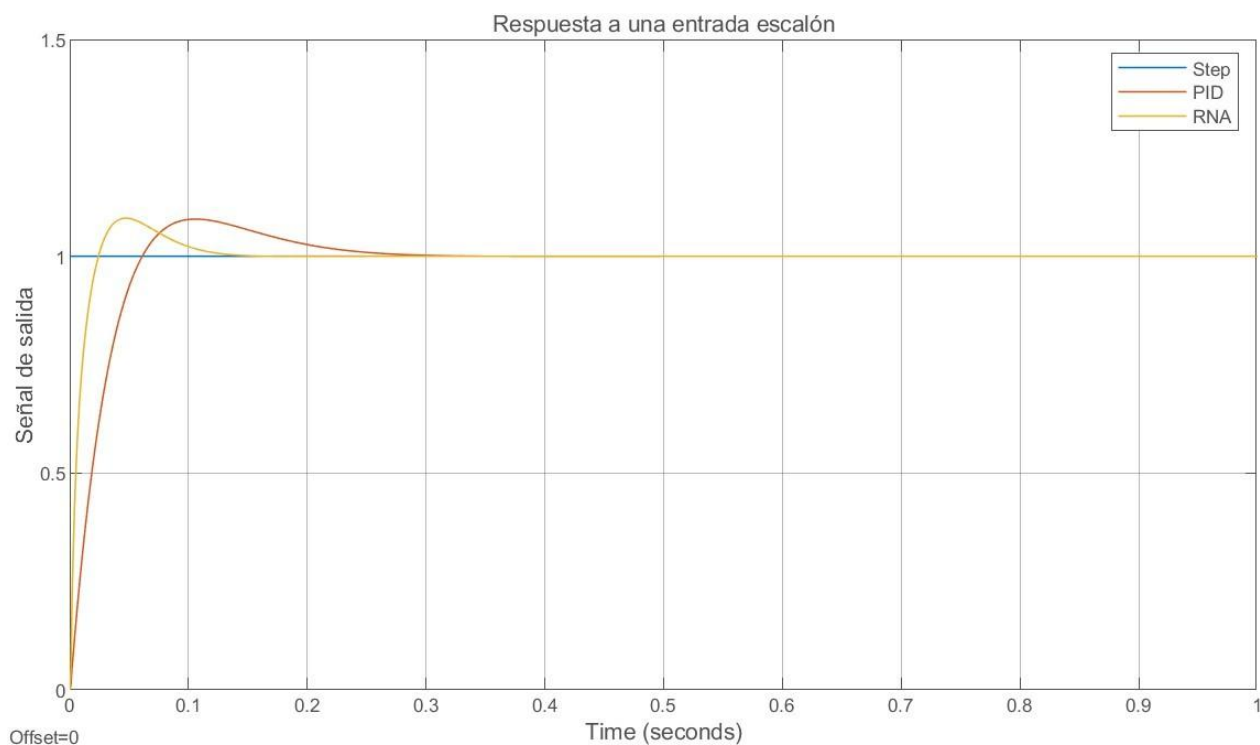


Figura 8. Comportamiento del sistema ante una señal escalón para los controladores PID y RNA





Tabla 2. Características de respuesta al escalón

Parámetro	PID	RNA
Overshoot	8%	9%
Tiempo de estabilización	0.471s	0.102s
Error de estado estacionario	0%	0%
Tiempo de subida	0.133	0.0158

Como se puede observar en la **Figura 7.** con los respectivos valores de la **Tabla 2.** Las características del análisis del tiempo de la señal escalón para los controladores PID y RNA son algo similares en cuanto al error de tiempo estacionario y en cierta medida al sobreimpulso. Pero en el tiempo de estabilización y el tiempo de subida, el control por RNA resulta tener mejor tiempo de respuesta.

IV. Conclusiones

El diseño y la implementación del controlador PID (proporcional-integral-derivativo) utilizando el software de MATLAB y Simulink demostró ser efectivo para el control de velocidad del motor, este mismo proporcionó una respuesta precisa y estable, permitiendo así el seguimiento de las referencias de velocidad de manera óptima, con un mínimo error y tiempo rápido de estabilización.

Las RNA usadas para la aproximación para modelar y controlar el sistema mostró una capacidad considerable para adaptación dinámica y no lineal del motor. Estas demostraron su utilidad al replicar el comportamiento del PID proporcionando resultados satisfactorios en predecir la velocidad.

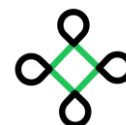
Ambos métodos ofrecieron resultados positivos en términos del control, el PID es más adaptado en el ámbito de ajustar parámetros, mostró una fácil implementación y ajuste, siendo así una opción confiable para sistemas con características bien comprendidas mientras que la RNA destaca por la capacidad de manejar sistemas no lineales y su adaptabilidad, estas demostraron ser más flexibles en la gestión de sistemas complejos(no lineales) pero su configuración adecuada es determinante para obtener los resultados deseados.

Por otro lado, en las gráficas podemos apreciar que la respuesta del PID muestra una rápida convergencia a la referencia, con una gran capacidad de minimizar el error, además refleja ausencia de oscilaciones en la respuesta escalón, lo que indica unos buenos parámetros en ajuste PID.

Si nos enfocamos ahora en la RNA muestra una adaptación suave y gradual a la referencia, destacando su capacidad para manejar cambios dinámicos y no lineales, además esta muestra una mayor respuesta y flexibilidad a perturbaciones, adaptación a variaciones inesperadas.

La comparación directa de las gráficas muestra que ambos métodos logran un seguimiento exitoso, PID tiende a tener una respuesta más rápida y agresiva, mientras que RNA una respuesta más suave y gradual, el método a elegir serán las consideraciones, enfocándose ya sea en velocidad de respuesta o la capacidad de adaptación.





Referencias

- [1] SDI. Motor de corriente continua. Obtenido de: <https://sdindustrial.com.mx/blog/motor-de-corriente-continua/>
- [2] Germán A. Hernández Millán; Luis Hernando Ríos González; Hernando Parra Lara. (2012). Implementación de un Controlador PID Mediante RNA para el Control de Motores D.C de Robots Moviles Diferenciales. Fundación Dialnet. Obtenido de: <https://dialnet.unirioja.es/servlet/articulo?codigo=4273550>
- [3] Hernández, U. M., Marcelo, B. A., Barrón, B. M., Miguel, J., & Paredes, H. (2019). Control PID Convencional con Filtro Pasa Bajas para el Control de Velocidad de un Motor de CD.
- [4] V. Santibañez, “Diseño de Controladores Basados en Modelo y Control por Adelanto-Atraso,” Capítulo 1, pp. 1-27, McGraw Hill, México, 2012.
- [5] K. Ogata, “Ingeniería de Control Moderna,” Quinta Edición Prentice-Hall, New Jersey. 2010.
- [6] A. İskender and M. Şahin, “Técnicas de Control Inverso para un Sistema de Refrigeración Basado en Métodos Inteligentes,” Journal of Food Engineering, vol. 120, no. 4, pp. 120-129, 2014.
- [7] Y. B. Shtessel et al., "Control de un UAV Quadrotor utilizando Redes Neuronales," 2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics.
- [8] M. S. Chen and S. J. Wang, "Control Inverso del Modelo para una Clase de Sistemas No Lineales usando Redes Neuronales," Proceedings del 27th Chinese Control Conference, pp. 1570-1574, 2008.
- [9] S. J. Kim and D. H. Shim, "Control de un Manipulador Accionado por un Músculo Neumático utilizando Control de Redes Neuronales Inverso," Proceedings del IEEE International Conference on Mechatronics and Automation, pp. 238-243, 2005.
- [10] G. Rodríguez-Ortiz et al., "Control Difuso de un Manipulador de Dos Eslabones utilizando un Enfoque de Control Inverso del Modelo," International Journal of Advanced Robotic Systems, vol. 9, no. 6, pp. 1-10, 2012.
- [11] H. Jiang et al., "Control Inverso del Modelo para un Rayo Vibratorio Amortiguado," Journal of Vibration and Control, vol. 23, no. 20, pp. 3241-3256, 2017.
- [12] J. D. Delgado et al., "Controlador PID mediante Redes Neuronales: comparativa con las técnicas PID y PID Auto sintonizado," Revista Cubana de Ingeniería, vol. 10, no. 1, pp. 15-26, 2019.
- [13] GSL Industrias, (2021). MOTOR CON ENCODER. Obtenido de: <https://industriasgsl.com/blogs/automatizacion/motor-con-encoder>
- [14] P. Ortega et al., "Control óptimo de la pisada humana en superficies blandas mediante un algoritmo de aprendizaje por refuerzo," Revista Iberoamericana de Automática e Informática Industrial, vol. 13, no. 4, pp. 502-510, 2016.





[15] C. Gómez et al., "Aprendizaje por Refuerzo Profundo aplicado a la Operación Inteligente de Redes de Jerarquía Eléctrica," Revista Facultad de Ingeniería Universidad de Antioquia, no. 84, pp. 83-98, 2018.

[16] C. Mehmet et al., "Control Inverso del Modelo de un Manipulador Montado sobre Rieles utilizando Control de Torque Computado No Lineal y Control Lineal Cuadrático," Journal of Mechanical Science and Technology, vol. 31, no. 10, pp. 4785-4799, 2017.

