Informaticae Abstracta
Año 2024, Vol. 2, No. 2, pp. 4-19 JUL | DIC 20244

Sistema de control mediante una red neuronal en motores de corriente directa

Adrian Camacho-Ramírez*; Marco Antonio Sánchez-Carmona

Información del Artículo

Recibido: 12-oct-2024
Aceptado: 17-dic-2024

Palabras clave: Control automático, red neuronal, Motor DC, ANN, PID

Resumen

Este estudio investiga la aplicación de redes neuronales artificiales en el control de velocidad de motores de corriente continua, con el objetivo de mejorar el rendimiento y la adaptabilidad en comparación con los controladores PID convencionales. La red neuronal artificial fue entrenada para ajustar dinámicamente la señal de control, logrando un seguimiento preciso de la velocidad sin necesidad de un modelo matemático detallado de la dinámica del motor, mostrando una alternativa robusta frente al control PID tradicional, el cual suele requerir un ajuste exhaustivo como su análisis de estabilidad y ajuste de las ganancias del controlador. Los resultados de simulación y experimentación confirman que el controlador basado en redes neuronales artificiales pueden mantener de manera adaptativa y precisa la velocidad deseada del motor.

1. Introducción

El control automático ha sido de gran importancia en la trasformación y evolución del ser humano debido a la constante necesidad de optimizar sus procesos o actividades en el menor tiempo posible, de manera eficiente y segura. Además, el control automático se volvió una parte importante e integral de los procesos modernos industriales y de manufactura. La automatización de procesos ha experimentado una evolución constante en los últimos años, provocando transformaciones significativas en nuestro país. En este contexto, es fundamental desarrollar procesos innovadores que modernicen y optimicen nuestros sistemas de producción automatizada, asegurando su competitividad y adaptación a las demandas tecnológicas actuales.

Los motores de corriente continua (CC) son fundamentales en la ingeniería de control y se utilizan ampliamente en aplicaciones domésticas, industriales y de investigación. Su gran versatilidad radica en su capacidad de control, ya que permiten ajustar con facilidad la velocidad, la posición y el par simplemente modificando el voltaje o la intensidad de corriente. Estas características los convierten en una opción ideal para el control de una amplia variedad de sistemas automatizados [1], [2], [3].

Aunque los motores de corriente alterna (CA), en particular los asíncronos, han ganado terreno debido a su control eficiente y costos más accesibles para los consumidores industriales [4], los motores de CC siguen siendo esenciales en aplicaciones que demandan alta potencia y precisión. Ejemplos de esto incluyen trenes, máquinas de precisión y micromotores, donde su control preciso y confiabilidad los hacen indispensables.

Existen varios métodos para controlar motores CC, una técnica común es la modulación por ancho de pulso (PWM) que es utilizado como señal de control para los controladores lineales a lazo cerrado, como los controladores PID (Proporcional-Integral-Derivativo); este último comúnmente regula el voltaje aplicado al motor, lo que resulta en un control eficiente y preciso de la velocidad. Por otro lado, los controladores PID se utilizan para mantener la velocidad o la posición deseada, compensando cualquier diferencia entre la salida real y el valor objetivo (setpoint). Esta capacidad de retroalimentación permite ajustar de manera continua la entrada al motor para corregir cualquier desviación en el comportamiento.

El uso de controladores lineales han dominado el control de motores CC, pero presentan limitaciones frente a sistemas no lineales, a la variación de parámetros y entornos cambiantes. Como resultado, la eficacia del control puede disminuir y el error aumentar bajo estas condiciones. Para abordar estos problemas, se pueden utilizar controladores robustos y no lineales, que son capaces de mejorar el rendimiento del sistema frente a dichas variaciones. Entre las técnicas de control utilizadas para resolver estos problemas podemos mencionar lógica difusa [5], control por modos deslizantes [6], aprendizaje iterativo [7], métodos de ajuste basados en inteligencia artificial [8]. La implementación de técnicas de aprendizaje profundo (deep learning), que son un conjunto de algoritmos de aprendizaje automático que intentan modelar los datos mediante el uso de componentes denominados *redes neuronales artificiales* (ANN, por sus siglas en inglés), para el control de motores CC son utilizados en [9], así como el uso de Matlab Simulink para el uso de ANN [10]. El principal componente de las ANN, lo encontramos con el término perceptrón, siendo este la ANN más antigua, y aun la más famosa [11]. En la Figura 1 se muestra la arquitectura de un perceptrón, donde se observan valores de entrada $x_1, x_2, \dots, x_n$ y un valor de salida $\overline{y}$; los valores de entrada vienen dados por la representación del conjunto de datos de entrenamiento y el valor de salida es la respuesta esperada para cada una de las muestras del conjunto de datos; para cada entrada x_i se asocia un peso w_i. El valor de salida de la neurona se calcula mediante la suma de los productos de las entradas por sus respectivos pesos, los cuales llegan mediante las conexiones de entrada aplicando una función de activación $\sigma(z)$, la cual dado un valor esta la transforma a un nuevo valor.

Si bien, el término ANN evoca la idea de máquinas que funcionan como cerebros y puede traer a la mente connotaciones de ciencia ficción, pero siendo más precisos en el término, una ANN es un conjunto interconectado de elementos de procesamiento simples, unidades o nodos, cuya funcionalidad se basa, de manera aproximada, en la neurona biológica. La capacidad de procesamiento de la red se almacena en las fortalezas de conexión entre las unidades, o pesos, que se obtienen a través de un proceso de adaptación o aprendizaje a partir de un conjunto de patrones de entrenamiento [12]. Las ANN tienen la capacidad de resolver problemas complejos en tiempo real, ya que funcionan mediante un entrenamiento previo (aprendizaje supervisado) y adaptarse continuamente a nuevas condiciones del problema. Una ventaja importante es que no es necesario desarrollar un

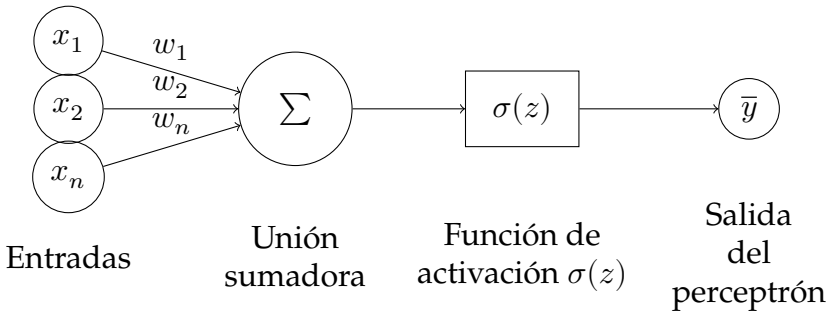


Figura 1: Estructura de un perceptrón

modelo a priori [13]. En la Figura 2 se puede observar una arquitectura básica de una ANN con tres capas; una capa de entrada con dos neuronas, una capa oculta con tres neuronas y una capa de salida con una única neurona.

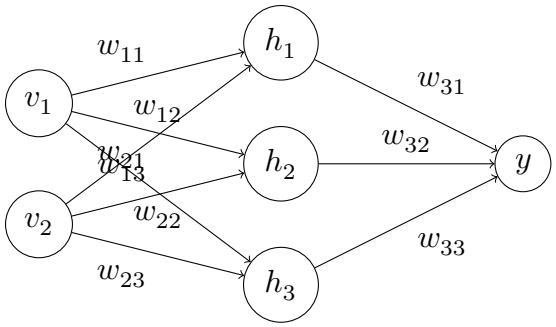


Figura 2: Arquitectura de una red neuronal artificial

Las ANN son entre todos los métodos disponibles, las más adecuadas para el reconocimiento de patrones en tiempo real, ya que operan en paralelo y actualizan todas sus instancias simultáneamente. Es fundamental señalar que esta característica se maximiza al implementar redes en hardware diseñado específicamente para el procesamiento paralelo.

Para el control automático de diferentes propósitos también se pueden utilizar ANN, incluyendo el control de fuerza para robots manipuladores [14], [15], control de postura y balanceo para robots bípedos [16], control de trayectoria para vehículo submarinos [17], para estimar la velocidad de vehículos longitudinales para aplicaciones dinámicas de control [18] y controles híbridos de posición y fuerza para controles adaptables para robots manipuladores como en [19].

Este trabajo de investigación explora la aplicación de ANN para el control de velocidad de un motor de CC, proponiendo una alternativa a los métodos tradicionales que requie-

ren un análisis exhaustivo para su diseño. El uso de esta tecnología es especialmente relevante en países emergentes como México. Las ANN se entrenarán utilizando tanto datos de simulaciones del sistema motor-control como resultados experimentales, y se espera que el control basado en ANN, entrenado para predecir el valor de control en función del error de velocidad, mejore la capacidad adaptativa y la robustez frente a perturbaciones en comparación con los controladores clásicos.

2. Materiales y Métodos

Los componentes utilizados, especificaciones y características se describen en las siguientes secciones, así como la arquitectura y componentes determinados para la ANN propuesta. También se detalla la estructura de los datos utilizados para la etapa de entrenamiento y pruebas tanto en simulación como los valores reales.

DC Motor

Se utiliza un Rotary Servo (RS), ver Figura 3, el cual es una plataforma intuitiva para experimentos de control de posición angular (grados) y control de velocidad angular (rpm, revoluciones por minuto). Es ideal para introducir conceptos y teorías de control básicos, el cual fue modificado para la adquisición de datos mediante un microcontrolador de bajo costo (Arduino Due).

Figura 3: Rotary Servo

La unidad base del RS es un sistema de servomecanismo con engranajes. Un motor CC impulsa un piñón pequeño, fijado a un engranaje de mayor diámetro que gira en torno al eje de interés. La posición angular del eje de interés se mide mediante un codificador óptico de alta resolución o encoder, ver Tabla 1. El encoder también se utiliza para estimar la velocidad angular del motor.

Tabla 1: Especificaciones del motor DC

Dimensiones	15cm x 15 cm x 18 cm
Masa	1.2 kg
Voltaje nominal	6 V
Corriente máxima	1 A
Resolución del enconder	4096 ppr (pulsos por revolución)
Velocidad máxima	63 rpm
Caja reductora	70:1

Arduino Duo

Arduino Due es una placa de desarrollo electrónica que se distingue por su potente microcontrolador SAM3X8E, basado en el núcleo ARM Cortex®-M3. Es una de las placas Arduino más avanzadas y potentes disponibles en el mercado. Con una velocidad de reloj de hasta 84 MHz y una generosa cantidad de memoria flash y RAM, el Arduino Due ofrece un rendimiento significativamente mejorado en comparación con las placas Arduino más antiguas. Esta placa es especialmente adecuada para proyectos que requieren un procesamiento de datos más rápido y una capacidad de cálculo más avanzada, como aplicaciones de control, robótica y procesamiento de señales [20].

Modulo controlador de motores L298N

El L298N es un circuito integrado comúnmente utilizado para controlar motores, actuando como interfaz entre los motores y una placa microcontroladora. Similar a otros controladores de la serie L293, el L298N está diseñado para manejar dos motores de manera simultánea. Este IC cuenta con dos puentes H (H-bridges) y tiene 15 pines, ofreciendo un mayor manejo de corriente que el L293D, lo que lo hace ideal para proyectos que requieren motores más potentes o mayores exigencias de control [21].

El driver L298N, conectado al motor, permite controlar tanto el sentido de giro como la velocidad angular a través de una entrada PWM. La señal de excitación (u) del motor depende del valor de PWM generado por el microcontrolador (Arduino Due), el cual tiene una resolución de 8 bits [0 – 255]. En este caso, el valor 255 corresponde al voltaje máximo aplicado al motor.

Arquitectura de la ANN

La ANN propuesta consta de 3 capas, en la primera capa de entrada se determinaron 10 neuronas con una entrada de 2 datos (x_1, x_2), la segunda capa, denominada oculta,

esta configurada igualmente por 10 neuronas y una tercera capa de salida es añadida a la arquitectura, esta capa consta únicamente de una neurona para dar la salida $\overline{y}$.

Función de activación

Para la capa 1 y 2 se estableció la función de activación ReLu (Rectified Linear Unit), siendo unas de las funciones de activación más comunes y utilizadas, debido a su efectividad en la aceleración del entrenamiento y en la mejora del rendimiento en la mayoría de los casos, la expresión matemática de esta función se muestra en la Ecuación 1.

$$ReLU(x) = \text{máx}(0, x) \tag{1}$$

La función ReLu devuelve el valor de entrada si es positivo y cero en caso contrario, por lo que hace que el cálculo sea eficiente en comparación con otras funciones de activación como la sigmoide o tangente hiperbólica que a diferencia de estas aplicada a la ANN, mejora la convergencia ya que se evita el problema del del desvanecimiento del gradiente, además su eficiencia computacional permite entrenar ANN de gran cantidad de capas y neuronas [22]. En la Figura 4 podemos observar el comportamiento de la función de activación ReLu, utilizada para la ANN propuesta en este trabajo.

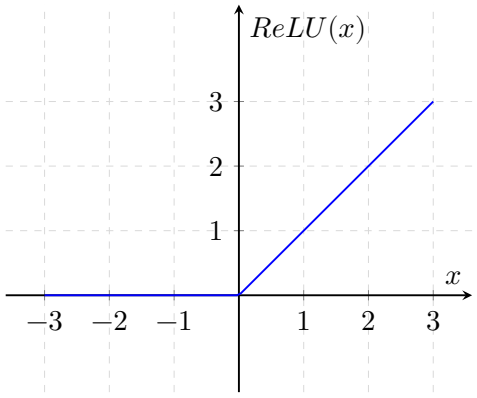


Figura 4: Gráfica de la función de activación ReLu

Optimizador

El optimizador seleccionado para la ANN es el algoritmo ADAM (Adaptative Moment Estimation), que es un algoritmo de optimización que mejora el proceso de aprendizaje de un modelo, debido a que utiliza una estimación del momento y de la magnitud de los gradientes anteriores para actualizar los parámetros, con esto ADAM adapta la tasa de aprendizaje de cada parámetro individualmente en función de su estimación del momento y la magnitud del gradiente [23]. Las principales ventajas del algoritmo ADAM conllevan a tener una adaptación de la tasa de aprendizaje, ya que ajusta dicha tasa de

cada parámetro individualmente con lo que se obtienen una mejor eficiencia, además utiliza el concepto de "corrección de sesgos"para evitar sesgos en los primeros pasos. En las Ecuaciones 2, 3 y 4 se expresan la forma de actualización de parámetros de ADAM, siendo m_t el promedio móvil de los gradientes, v_t el promedio móvil del cuadrado de los gradientes, β_1, β_2 factores de decaimiento exponencial y ϵ siendo un valor pequeño para evitar divisiones por 0.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{2}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{3}$$

$$\theta = \theta - \eta \frac{m_t}{\sqrt{v_t} + \epsilon} \tag{4}$$

Función de pérdida

La función de pérdida o costo establecida para la ANN es el Error Cuadrático Medio (MSE, por sus siglas en inglés), que puede definirse en la Ecuación 5, la ventaja de utilizar MSE para la ANN radica en que penaliza los errores más grandes que los pequeños debido a la naturaleza cuadrática de la función, con lo que prioriza la corrección donde hay grandes discrepancias entre la predicción y los valores reales [24].

$$mse = \frac{1}{n} \sum (y_n - \overline{y})^2 \tag{5}$$

Configuración para el entrenamiento

Para el entrenamiento se estableció una cantidad de épocas de $1,000$, utilizando el 80 % del conjunto de datos. Al utilizar el optimizar ADAM, se dejó la tasa de aprendizaje en $0{,}001$, dado el funcionamiento de este optimizador, la tasa de aprendizaje η se autoajusta como se observó en la Ecuación 4.

Estructura de los datos

Los datos utilizados, tanto para la etapa de entrenamiento como pruebas, se conforman de un dataset que contiene: Tiempo (medido en milisegundos), setpoint, valores de RPM, una valor de error y PWM, cada uno distribuido en una columna, los valores de tiempo, RPM y el error están almacenados como valores de punto flotante, por otro lado setpoint y PWM son valores enteros positivos. El total de registros utilizados se compone de 620 registros que representan los datos trabajados por un tiempo de 60 segundos (con una holgura). De este conjunto de datos, se utilizaron las RPM, el error como elementos de entrada X, mientras que PWM se utilizó como la salida deseada Y.

3. Implementación

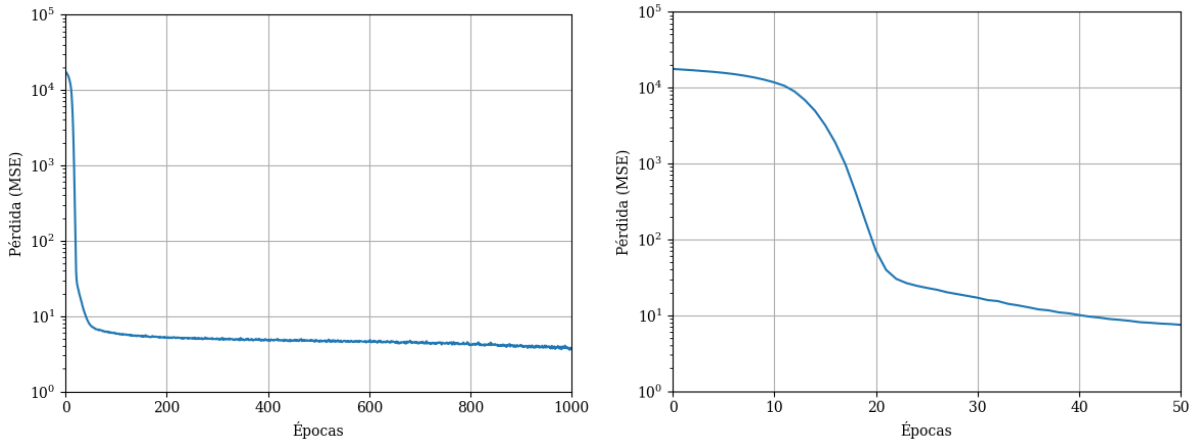
La ANN se implementó mediante la biblioteca Keras, siendo esta una biblioteca de redes neuronales de código abierto para el lenguaje Python, la ANN se creó mediante las configuraciones mencionadas, en la Tabla 2 se resume la configuración de la ANN utilizada.

Tabla 2: Resumen de la configuración de la ANN

Número de capas	3
Neuronas en la capa 1 y 2	10
Función de activación	ReLu
Optimizador	ADAM
Tasa de aprendizaje	0.001
Número de épocas	1000

Comportamiento de la ANN durante el entrenamiento

Después de la etapa de entrenamiento de la ANN propuesta, se muestra en la Figura 5 la gráfica de la función de pérdida MSE por cada época de entrenamiento, es decir, desde 1 hasta 1000 épocas. En la gráfica de la Figura 5a se observa el descenso de la función antes de llegar a la época número 200, en la gráfica de la Figura 5b se observa más a detalle este descenso en las primeras 50 épocas, ya que la curva empieza a converger con el eje X. El valor mínimo obtenido de MSE es de $0{,}0078$ que demuestra que la ANN tienen un margen de error bajo para las predicciones.



(a) Pérdida durante las 1000 épocas (b) Pérdida durante las primeras 50 épocas

Figura 5: Gráficas de la función de pérdida MSE durante el entrenamiento

Simulaciones con control ANN

Una vez entrenada, la ANN puede predecir la salida del motor en función de diferentes entradas, lo que facilita la toma de decisiones en tiempo real. La ANN puede ajustar las señales de control, minimizando el error entre la velocidad deseada y la real, mejorando así la precisión y el rendimiento del sistema.

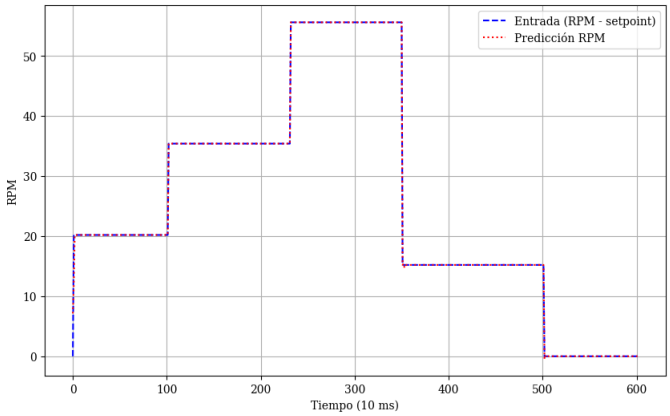


Figura 6: Predicción de la ANN para control de velocidad

Implementación con controlador PID

Para implementar el control PID, es fundamental determinar la función de transferencia del motor. Esto se logra excitando el motor con un voltaje específico y registrando su respuesta ante esta entrada. Los resultados obtenidos, que reflejan el comportamiento del sistema, se presentan en la gráfica de la Figura 7.

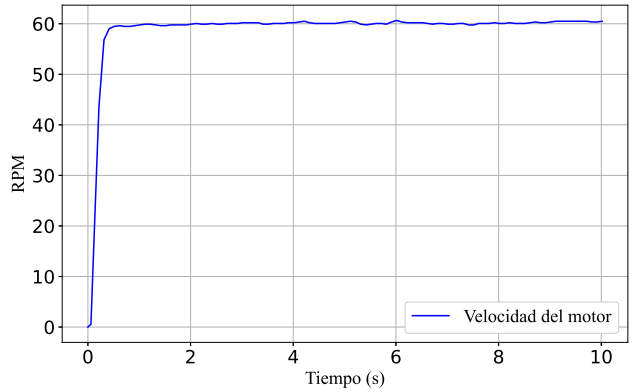


Figura 7: Curva de velocidad del motor a (PWM=255)

Los datos de entrada (voltaje) y de salida (RPM) fueron recolectados, y a través del método de System Identification en Matlab®, se obtuvo una aproximación matemática

del sistema RS en forma de una función de transferencia de primer orden, lo cual es una representación común para motores CC. Este enfoque permitió capturar con precisión la dinámica del motor, facilitando su modelado y control. La función de transferencia en la Ecuación 6 se obtuvo con entrada de control de PWM con valor de 255 y representa el modelo matemático del RS.

$$G(s) = \frac{4{,}673}{s + 4{,}676} \tag{6}$$

Un controlador PID aplicado a un motor CC permite regular su velocidad ajustando el voltaje de entrada. Esto se logra mediante el uso de un controlador L298N, que modula la señal de control generada por el Arduino a través de una señal PWM, permitiendo así el control preciso de la velocidad del motor.

La forma general del control por PID es:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{7}$$

Donde $u(t)$ es la señal de control, $e(t)$ es el error (setpoint - valor medido). K_p, Ki, K_d son los coeficientes de ganancia de los términos proporcional, integral y derivativo, respectivamente, los cuales son los encargados de proporcionar un control eficiente y preciso de la velocidad del motor, mejorando su capacidad de respuesta y reduciendo el error.

En términos del dominio de la frecuencia (s), el controlador PID se representa utilizando la transformada de Laplace de la Ecuación 7. Obteniendo así la siguiente función de transferencia $C(s)$:

$$C(s) = K_p + \frac{K_i}{s} + K_d s \tag{8}$$

Con esta función de transferencia el control mediante PID se realiza en lazo cerrado como se ve en la Figura 8.

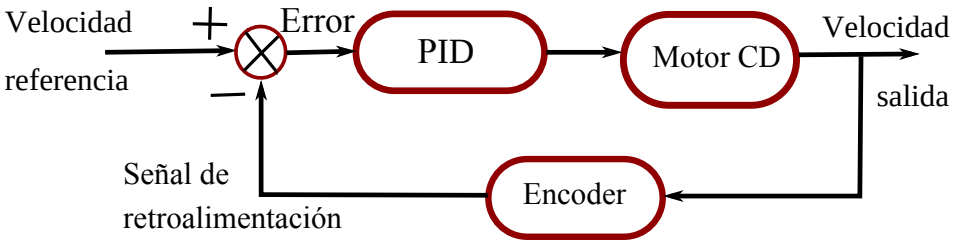


Figura 8: Diagrama de control

La implementación de este controlador al motor se realizo un análisis por medio del criterio de estabilidad Routh-Hurwitz [25], del cual se obtuvo la función de transferencia a lazo expresado en la Ecuación 9:

$$G_{PID}(s) = \frac{4{,}673(K_p s + K_i + K_d s^2)}{(s + 4{,}676)s + 4{,}673(K_p s + K_i + K_d s^2)} \tag{9}$$

La ecuación característica del sistema en lazo cerrado es:

$$(1 + 4{,}673K_d)s^2 + (4{,}676 + 4{,}673K_p)s + 4{,}673K_i = 0$$

Aplicando el criterio de Routh-Hurwitz, el sistema será estable si los valores de las ganancias K_p, K_i y K_d satisfacen las siguientes condiciones:

- $K_d \geq 0$
- $K_p > -1$
- $K_i > 0$

Estos valores aseguran que el sistema en lazo cerrado, compuesto por la función de transferencia del motor y el controlador PID, mantendrá su estabilidad.

Simulaciones con el controlador PID

Tomando como referencia estas condiciones, y mediante simulaciones en Simulink/-Matlab junto con una sintonización experimental, se determinaron las ganancias óptimas del controlador PID, las cuales se presentan en el Tabla 3.

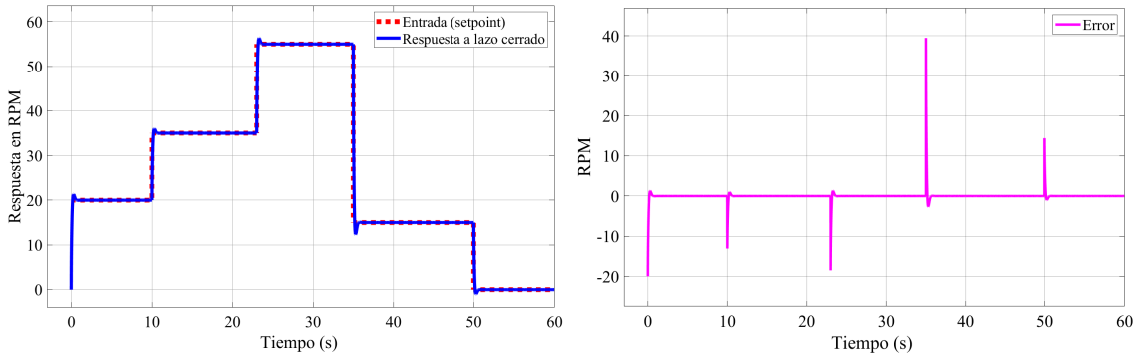
Tabla 3: Ganancias del PID

Ganancias	Valores
K_p	2.8
K_i	25
K_d	0.04

La respuesta del sistema en lazo cerrado se presenta en la Figura 9a, donde se observa en color rojo la evolución del setpoint, que varía la velocidad a lo largo del tiempo. La línea azul, por su parte, muestra el comportamiento del motor bajo el control PID, destacando cómo el sistema ajusta la velocidad del motor para seguir el valor de referencia. A medida que el setpoint cambia, el controlador PID ajusta el voltaje aplicado al motor para minimizar el error entre la velocidad deseada y la velocidad real del motor, lo que resulta en una respuesta rápida y estable. Asi mismo se muestra el error (ver Figura 9b)entre el setpoint y la velocidad de salida del motor en RPM.

4. Resultados

Los resultados experimentales en el sistema RS para la ANN y el controlador PID se basaron en los análisis discutidos en la sección anterior. Se utilizó la misma señal de referencia de velocidad propuesta en las simulaciones, con el objetivo de obtener una comparación más clara del comportamiento de ambos controladores aplicados al motor CC.



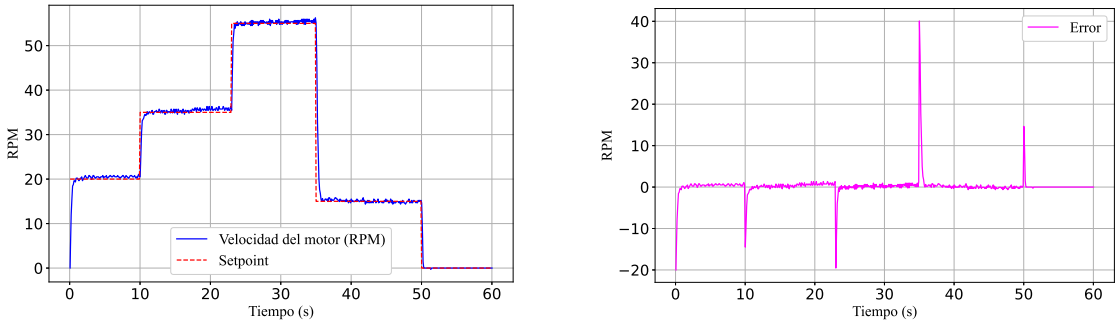
(a) Respuesta ante la entrada a lazo cerrado (b) Error

Figura 9: Simulación de control de velocidad con PID

Experimentación con ANN

La ANN ya entrenada se implementó con el microcontrolador Arduino Due, con la capacidad para realizar las predicciones en tiempo real realizadas por la ANN para ajustar el ciclo de trabajo del motor DC. Se realizó la conversión de los pesos de la ANN entrenada en Python a un formato compatible con el entorno Arduino para su uso en la predicción de PWM.

El código en Arduino se encargó de medir la velocidad del motor utilizando un encoder, calcular el error entre el setpoint y la velocidad medida, y aplicar el valor de PWM predicho por la red neuronal para controlar la velocidad del motor. La Figura 10 muestra el comportamiento del sistema RS utilizando la ANN como controlador. En la Figura 10a, se observa el seguimiento de velocidad (en azul) en comparación con el setpoint (en rojo), mientras que la Figura 10b presenta el error entre la velocidad medida y el setpoint.

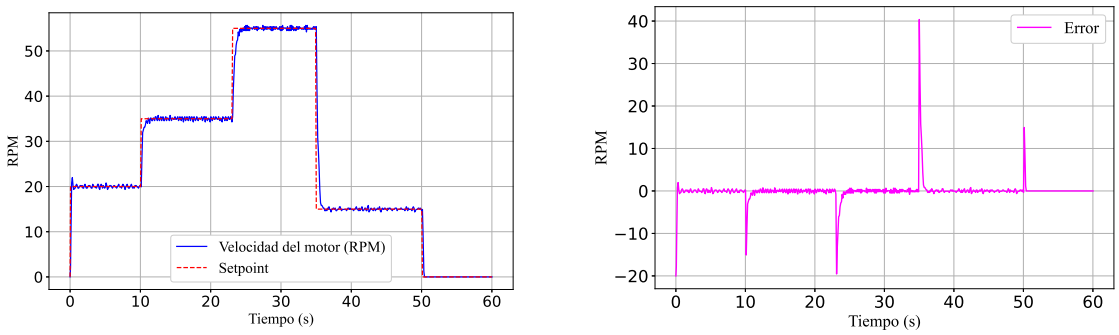


(a) Control de velocidad mediante ANN (b) Error

Figura 10: Control de velocidad del RS con ANN

Experimentación con PID

Con base en el análisis expuesto en la Sección 3, se implementó el control propuesto en el sistema RS, aplicando un ajuste de ganancias mediante una sintonización fina de forma experimental. Los valores ajustados fueron $K_d = 5$, $K_p = 15$ y $K_d = 0{,}01$, obteniendo los resultados moque se muestran en la Figura 11. En la Figura 11a, se presenta la velocidad del motor en respuesta a el setpoint determinado, mientras que la Figura 11b ilustra el error de seguimiento entre la señal de referencia y la señal medida.



(a) Control de velocidad mediante PID (b) Error

Figura 11: Control de velocidad del RS con PID

5. Conclusiones

Este estudio demuestra la efectividad de las ANN en el control de velocidad de motores de corriente continua, proporcionando una alternativa viable y adaptativa frente a métodos de control convencionales como el PID. La ANN implementada mostró un alto grado de precisión en el seguimiento de la velocidad deseada, ajustando dinámicamente la señal de control y reduciendo el error frente a cambios en el setpoint y variaciones externas. En comparación con el controlador PID, que requiere ajustes precisos de los parámetros de control, la ANN ofrece una mayor capacidad de adaptación sin depender de un modelo matemático explícito del sistema, lo cual es especialmente beneficioso para sistemas con dinámicas complejas o no lineales. Las simulaciones y experimentos realizados en el entorno de Arduino con el motor DC revelaron que la ANN y el controlador PID lograron un buen seguimiento de velocidad; sin embargo, la ANN presentó un mejor rendimiento ya que no genero sobreimpulsos en la respuesta del sistema y puede ser más robusto en presencia de perturbaciones y cambios en la dinámica del sistema. Estos resultados sugieren que, al integrar redes neuronales, los sistemas de control de motores

pueden optimizar su desempeño y fiabilidad, adaptándose mejor a variaciones en el entorno operativo. Futuros estudios podrían explorar el ajuste automático y la escalabilidad de ANN para el control de múltiples motores, así como la implementación en hardware especializado para maximizar la eficiencia en aplicaciones industriales.

Direcciones de correo de los autores

Adrian Camacho-Ramírez*: acamachor@uaemex.mx

Marco Antonio Sánchez-Carmona: msanchezc048@alumno.uaemex.mx

Referencias

[1] S. J. Hammoodi, K. S. Flayyih y A. R. Hamad, «Design and implementation speed control system of DC Motor based on PID control and matlab simulink,» en, *International Journal of Power Electronics and Drive Systems (IJPEDS)*, vol. 11, n.º 1, pág. 127, mar. de 2020, ISSN: 2722-256X, 2088-8694. DOI: 10.11591/ijpeds.v11.i1.pp127-134. visitado 9 de oct. de 2024. dirección: http://ijpeds.iaescore.com/index.php/IJPEDS/article/view/20153.

[2] M. Mahmud, S. M., A. H. y A. Nurashikin, «Control BLDC Motor Speed using PID Controller,» en, *International Journal of Advanced Computer Science and Applications*, vol. 11, n.º 3, 2020, ISSN: 21565570, 2158107X. DOI: 10.14569/IJACSA.2020.0110359. visitado 9 de oct. de 2024. dirección: http://thesai.org/Publications/ViewPaper?Volume=11&Issue=3&Code=IJACSA&SerialNo=59.

[3] S.-H. Kim, *Electric motor control: DC, AC, and BLDC motors*. Elsevier, 2017.

[4] S. Sakunthala, R. Kiranmayi y P. N. Mandadi, «A study on industrial motor drives: Comparison and applications of PMSM and BLDC motor drives,» en *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, IEEE, 2017, págs. 537-540.

[5] Y. A. Almatheel y A. Abdelrahman, «Speed control of DC motor using Fuzzy Logic Controller,» en *2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, ene. de 2017, págs. 1-8. DOI: 10.1109/ICCCCEE.2017.7867673. visitado 9 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/7867673.

[6] A. T. Hafez, A. A. Sarhan y S. Givigi, «Brushless DC Motor Speed Control Based on Advanced Sliding Mode Control (SMC) Techniques,» en *2019 IEEE International Systems Conference (SysCon)*, ISSN: 2472-9647, abr. de 2019, págs. 1-6. DOI: 10.1109/SYSCON.2019.8836754. visitado 9 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/8836754.

[7] D. Shen, «Iterative learning control with incomplete information: a survey,» *IEEE/-CAA Journal of Automatica Sinica*, vol. 5, n.º 5, págs. 885-901, sep. de 2018, Conference Name: IEEE/CAA Journal of Automatica Sinica, ISSN: 2329-9274. DOI: 10.1109/JAS.2018.7511123. visitado 9 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/8405347.

[8] P. Kofinas y A. I. Dounis, «Fuzzy Q-learning agent for online tuning of PID controller for DC motor speed control,» *Algorithms*, vol. 11, n.º 10, pág. 148, 2018.

[9] M. Alhanjouri, «Speed Control of DC Motor Using Artificial Neural Network,» en, *International Journal of Science and Research*, vol. 6, n.º 2, 2015, ISSN: 2319-7064. DOI: 10.21275/ART20172035.

[10] H. Sridhar, P. Hemanth, Pavitra, H. V. Soumya y B. G. Joshi, «Speed Control of BLDC Motor using Soft Computing Technique,» en *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, sep. de 2020, págs. 1162-1168. DOI: 10.1109/ICOSEC49089.2020.9215417. visitado 9 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/9215417.

[11] Departamento de Matemáticas e Informática de la Universidad de Barcelona, *El poder de los datos: Del big data al aprendizaje profundo*. National Geographic, 2017, ISBN: 978-84-473-8940-7.

[12] K. Gurney, *An Introduction to Neural Networks*. London: CRC Press, ene. de 2017, ISBN: 978-1-315-27357-0. DOI: 10.1201/9781315273570.

[13] Y.-c. Wu y J.-w. Feng, «Development and Application of Artificial Neural Network,» en, *Wireless Personal Communications*, vol. 102, n.º 2, págs. 1645-1656, sep. de 2018, ISSN: 1572-834X. DOI: 10.1007/s11277-017-5224-x. visitado 8 de oct. de 2024. dirección: https://doi.org/10.1007/s11277-017-5224-x.

[14] F. Lewis, «Neural network control of robot manipulators,» *IEEE Expert*, vol. 11, n.º 3, págs. 64-75, jun. de 1996, Conference Name: IEEE Expert, ISSN: 2374-9407. DOI: 10.1109/64.506755. visitado 8 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/506755.

[15] Z. Xu, S. Li, X. Zhou, S. Zhou, T. Cheng e Y. Guan, «Dynamic Neural Networks for Motion-Force Control of Redundant Manipulators: An Optimization Perspective,» *IEEE Transactions on Industrial Electronics*, vol. 68, n.º 2, págs. 1525-1536, feb. de 2021, Conference Name: IEEE Transactions on Industrial Electronics, ISSN: 1557-9948. DOI: 10.1109/TIE.2020.2970635. visitado 8 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/8984689.

[16] C. Sun, W. He, W. Ge y C. Chang, «Adaptive Neural Network Control of Biped Robots,» *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, n.º 2, págs. 315-326, feb. de 2017, Conference Name: IEEE Transactions on Systems, Man, and Cybernetics: Systems, ISSN: 2168-2232. DOI: 10.1109/TSMC.2016.2557223. visitado 8 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/7467542.

[17] R. Cui, C. Yang, Y. Li y S. Sharma, «Adaptive Neural Network Control of AUVs With Control Input Nonlinearities Using Reinforcement Learning,» *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, n.º 6, págs. 1019-1029, jun. de 2017, Conference Name: IEEE Transactions on Systems, Man, and Cybernetics: Systems, ISSN: 2168-2232. DOI: 10.1109/TSMC.2016.2645699. visitado 8 de oct. de 2024. dirección: https://ieeexplore.ieee.org/abstract/document/7812772.

[18] G. Napolitano Dell'Annunziata, V. M. Arricale, F. Farroni, A. Genovese, N. Pasquino y G. Tranquillo, «Estimation of Vehicle Longitudinal Velocity with Artificial Neural Network,» en, *Sensors*, vol. 22, n.º 23, pág. 9516, ene. de 2022, Number: 23 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 1424-8220. DOI: 10.3390/s22239516. visitado 8 de oct. de 2024. dirección: https://www.mdpi.com/1424-8220/22/23/9516.

[19] C. Cao, F. Wang, Q. Cao, H. Sun, W. Xu y M. Cui, «Neural network–based terminal sliding mode applied to position/force adaptive control for constrained robotic manipulators,» *Advances in Mechanical Engineering*, vol. 10, n.º 6, pág. 1 687 814 018 781 288, 2018.

[20] Arduino. «Arduino Due.» Accesado 19 de octubre, 2024.

[21] STMicroelectronics. «L298.» Accesado 19 de octubre, 2024.

[22] V. Nair y G. E. Hinton, «Rectified Linear Units Improve Restricted Boltzmann Machines,» en *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, págs. 807-814.

[23] D. P. Kingma y J. Ba, «Adam: A method for stochastic optimization,» *arXiv preprint arXiv:1412.6980*, 2014.

[24] I. Goodfellow, Y. Bengio y A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[25] K. Ogata, «Modern control engineering,» 2020.