Metodología para entrenamiento de redes neuronales artificiales utilizadas como controladores inteligentes en procesos industriales

Paul Enrique Arias-Lizárraga, Julio César Picos-Ponce,
Guillermo Javier Rubio-Astorga, David Enrique Castro-Palazuelos

Tecnológico Nacional de México/Instituto Tecnológico de Culiacán,
México

{M24171313, julio.pp}@culiacan.tecnm.mx

Resumen. El uso de aplicaciones de inteligencia artificial (IA) en la industria 4.0 son imprescindibles en la actualidad. Sin embargo, el hardware estándar con el que se implementan las aplicaciones de automatización y control en la industria quedan limitados para tal fin. Hoy en día, existe hardware especializado capaz de ejecutar aplicaciones complejas de IA. La diversidad de dispositivos que existen en entornos industriales y la necesidad de que estos funcionen junto a dispositivos modernos se presenta como un reto. En este trabajo se busca integrar estas dos tecnologías: el hardware estandarizado para aplicaciones industriales con todas sus ventajas inherentes como modularidad, diversidad de protocolos de comunicación, robustez, entre otros, con esta nueva arquitectura de hardware capaz de ejecutar algoritmos complejos de IA como control inteligente. Para lograr esto, se desarrolló una metodología donde se enlazan un proceso a controlar, un dispositivo de control convencional (controlador lógico programable) y un dispositivo moderno (Nvidia Jetson Orin Nano Super) comunicados vía Modbus TCP/IP. Esto con el fin de generar una arquitectura capaz de adquirir y procesar datos, además de ejecutar tareas de control con redes neuronales artificiales. Se adquirieron datos de un proceso, los cuales se utilizaron para el entrenamiento de una red neuronal artificial. Posteriormente se implementó un control inteligente con la misma infraestructura de hardware y se comparó con una implementación de control PID. Los resultados de la comparación a través de pruebas estadísticas muestran que el control con redes neuronales artificiales es posible con la metodología utilizada.

Palabras clave: Control inteligente, industria 4.0, base de datos.

Methodology for Training Artificial Neural Networks Used as Intelligent Controllers in Industrial Processes

Abstract. The use of artificial intelligence (AI) applications in Industry 4.0 is essential today. However, the standard hardware used to implement automation and control applications in industry is limited for this purpose. Today, specialized hardware exists that can run complex AI applications. The diversity of devices in industrial environments and the need for them to work alongside modern devices

presents a challenge. This work seeks to integrate these two technologies: standardized hardware for industrial applications, with all its inherent advantages such as modularity, diverse communication protocols, and robustness, among others, with this new hardware architecture capable of running complex AI algorithms for intelligent control. To achieve this, a methodology was developed that links a process to be controlled, a conventional control device (programmable logic controller), and a modern device (Nvidia Jetson Orin Nano Super) communicating via Modbus TCP/IP. This aims to generate an architecture capable of acquiring and processing data, as well as executing control tasks with artificial neural networks. Data was acquired from a process and used to train an artificial neural network. Subsequently, intelligent control was implemented using the same hardware infrastructure and compared to a PID control implementation. The results of the comparison, obtained through statistical tests, demonstrate that control with artificial neural networks is feasible using the methodology employed.

Keywords: Intelligent control, industry 4.0, database.

1. Introducción

La industria 4.0 se ha caracterizado por la integración de tecnologías emergentes como lo son el Internet de las cosas, la Inteligencia Artificial (IA), la computación en la nube, entre otras[1]. En este marco se ha visto una vasta cantidad de aplicaciones de la IA, las cuales utilizan a las redes neuronales artificiales (RNA) como uno de sus núcleos [2].

Además, ha cobrado relevancia el concepto de Edge Computing, la cual se caracteriza por la cercanía al usuario, así como por un tratamiento local de los datos. De la mano de compañías como Nvidia se han desarrollado dispositivos que caben dentro de dicho concepto, como la tarjeta de desarrollo Jetson Orin Nano Super (JONS), además, poseen la capacidad para ejecutar RNA. Por otro lado, las herramientas de software moderno han permitido facilitar el desarrollo de aplicaciones basadas en RNA.

Los enlaces de comunicación industrial modernos deben ser estables, tener la capacidad para el almacenamiento de grandes cantidades de datos y tener la capacidad de integrar herramientas modernas de hardware y software. En consecuencia, en el área de instrumentación industrial se ha identificado como un reto la integración de las nuevas tecnologías con las ya existentes. Esta problemática se origina, en gran medida, debido a la diversidad de dispositivos utilizados en una red de control industrial [3].

Se han encontrado ventajas en el uso de aplicaciones de RNA para el control de procesos, ya sea como complemento a sistemas de control más tradicionales con el fin de mejorar el comportamiento de un sistema [4] o en la generación de algoritmos de control adaptable a través de la creación de bases de datos adecuadas [5].

Con la finalidad de aumentar la cantidad de aplicaciones de RNA en la industria, debido a las ventajas que presentan, se debe buscar la integración de hardware y software adecuado para dichos entornos. Además, se deben establecer metodologías claras para la creación de bases de datos para los entrenamientos, así como para las aplicaciones de RNA.

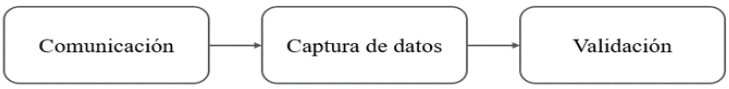

Fig. 1. Proceso para generación de base de datos.

2. Trabajos relacionados

Al ser necesaria la integración de múltiples dispositivos para la recopilación y uso de grandes cantidades de datos, surgen arquitecturas de sistemas con el fin de cubrir esa necesidad. Una de ellas es la presentada por Radlbauer, en la cual se prioriza la integración de diferentes protocolos de comunicación con el fin de utilizar tecnologías modernas como antiguas, centradas en la recolección y visualización de los datos [6].

En aplicaciones como el mantenimiento predictivo, como lo mostrado por Halenar, se han desarrollado sistemas donde se adiciona hardware con la capacidad para cubrir los requerimientos de almacenamiento y monitoreo de datos. Se muestran las capacidades del hardware moderno, se utilizan dispositivos de Edge computing, para la recopilación de datos útiles con la finalidad de un uso posterior por redes neuronales [7].

Los dispositivos de *Edge computing*, permiten adicionar o facilitar el uso de ciertas herramientas de software, por lo tanto, se vuelven muy útiles no solo para un tratamiento local de datos, sino también para funcionar como un intermediario entre sistemas locales y sistemas en la nube. Con el fin de poder escalar el almacenamiento de datos, Magnus presenta un sistema donde combina un sistema de *Edge computing* con almacenamiento en la nube [8].

Las herramientas de software desarrolladas para la integración de sistemas son muy útiles, sin embargo, aquellas que son cerradas y dependen de una compañía, como las utilizadas por Radlbauer, pueden tener limitaciones. La metodología de agregar capas a un proceso ya establecido con la finalidad de añadir funcionalidades nuevas, como la predicción del mantenimiento predictivo presentado por Halener, sin necesidad de alterar aquello que ya funciona se ve limitada cuando se quiere agregar funcionalidades directamente relacionadas con los sensores y actuadores del proceso.

Complementar sistemas locales con herramientas como la computación en la nube tiene ventajas, como la escalabilidad de almacenamiento de datos y el poder desplegarlos de maneras que un usuario pueda entenderlo con facilidad en cualquier dispositivo con acceso a la nube. Sin embargo, en situaciones donde no se tenga un acceso estable a estos sistemas, la selección de un hardware, software y métodos adecuados para optimizar el manejo de datos necesario para que herramientas como las redes neuronales funcionen adecuadamente, cobre relevancia.

Por lo anterior, en este trabajo se busca presentar una arquitectura donde se integre hardware, como controladores lógicos programables (PLC por sus siglas en inglés) y dispositivos que permitan el tratamiento cercano de los datos, así como protocolos de comunicación estandarizados, además de una metodología para recopilación de datos que sea útil para el entrenamiento de redes neuronales para control. Finalmente, se

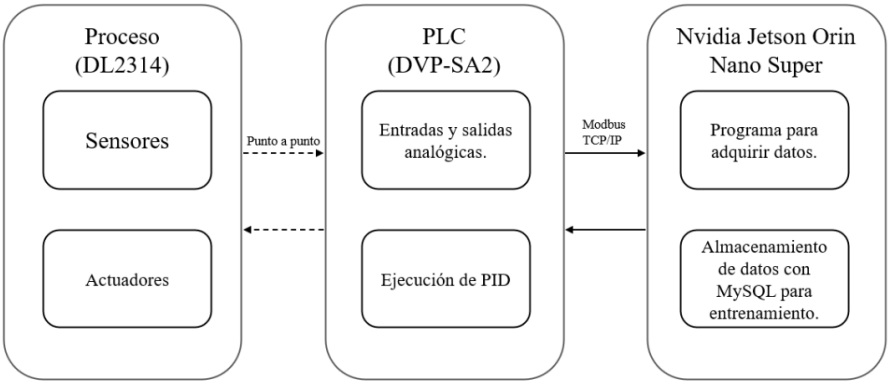

Fig. 2. Arquitectura de comunicación para toma de datos.

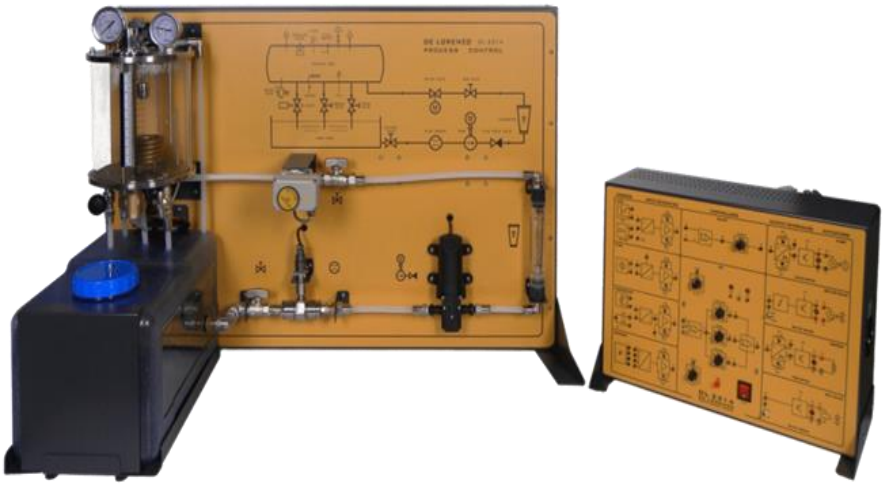

Fig. 3. Proceso utilizado, marca De Lorenzo modelo DL2314 [10].

prueba esta arquitectura con la implementación de un control inteligente de una variable en un proceso.

3. Metodología propuesta

El trabajo realizado se encuentra dividido en tres bloques, en la Fig. 1 representa las etapas que se deben realizar para la generación de una base de datos útil para el entrenamiento de una red neuronal. Las siguientes secciones se encuentran divididas de manera correspondiente a los bloques del sistema propuesto. En cada sección se detalla lo necesario para cada bloque, su funcionamiento y el resultado proporcionado.

3.1 Comunicación

En este bloque se integran los diferentes instrumentos de hardware necesarios para el almacenamiento y procesamiento de la información. Para ello se necesita un nodo con la capacidad de leer sensores y activar actuadores, además de poseer la capacidad de trasmitir los datos a través de un puerto de red, un nodo con la capacidad de procesar datos, además de una planta para utilizar sus respectivos sensores y actuadores.

La planta proporciona las señales de sensores, son capturadas por un PLC. Dicho PLC funciona como un servidor comunicado por el protocolo Modbus TCP/IP, lo cual permite el envío de datos a otro dispositivo en red que lo solicite y utilice el mismo protocolo de comunicación. Se seleccionó el protocolo al mostrar buenos resultados para el envío de datos [9].

La planta es un sistema de control de procesos, modelo DL2314 de la compañía De Lorenzo, se puede ver en la Fig. 3, el cual cuenta con un tanque equipado con sensores para medición de nivel con salida analógica de 0 a 10 vcd, además, para el llenado del tanque cuenta con una bomba, tres salidas de agua de activación manual y una salida con activación por electroválvula de dos posiciones [10]. La conexión con el hardware encargado de leer los datos proporcionados por el sensor es del tipo punto a punto.

Para la lectura de sensores y activación de actuadores se realiza con un PLC marca DELTA, modelo DVP-12SA2, adicionado con el módulo DVP-06XA-S para manipular señales analógicas y el módulo DVPEN01 para comunicación por red.

El nodo encargado del almacenamiento de los datos es una tarjeta de desarrollo de la marca Nvidia, modelo Jetson Orin Nano Super (JONS), el procesador es 6-core Arm (R) Cortex (R)-A78AE v8.2, cuenta con 8 GB RAM [11]. Para lograr la comunicación se configura como un cliente Modbus TCP/IP con el uso de Python y la librería pymodbus versión 2.1.0.

3.2 Captura de datos

Para obtener datos que una red neuronal pueda utilizar para replicar un control, se debe tener un sistema que los pueda generar. Con este fin se utilizó un control PID (proporcional-integral-derivativo) sintonizado a través del método de Ziegler-Nichols [12]. Entre los datos que se almacenaron, por cada marca de tiempo, para el posterior entrenamiento de la red neuronal se encuentran el error, la derivada del error, el valor deseado y la potencia de activación del actuador.

Para la ejecución del algoritmo de control PID y obtención de los datos, se programó el PLC con el software ISPSoft versión 3.21, en la Fig. 4 se muestra el diagrama de flujo. Una vez que se genera el dato es leído por el cliente y almacenado en una base de datos, para la gestión de esta última se utilizó MySQL.

El conjunto de datos de entrenamiento se generó a partir de la ejecución del algoritmo PID en un periodo de tiempo de 2400 segundos con un periodo de muestreo de 4 segundos. Durante este tiempo, se hicieron 30 cambios aleatorios del valor deseado (*setpoint*), con el fin de capturar la dinámica del sistema PID, dando un total de 600 conjuntos de datos almacenados en JONS. No fue necesario realizar ningún tratamiento a los datos posterior a su almacenamiento.

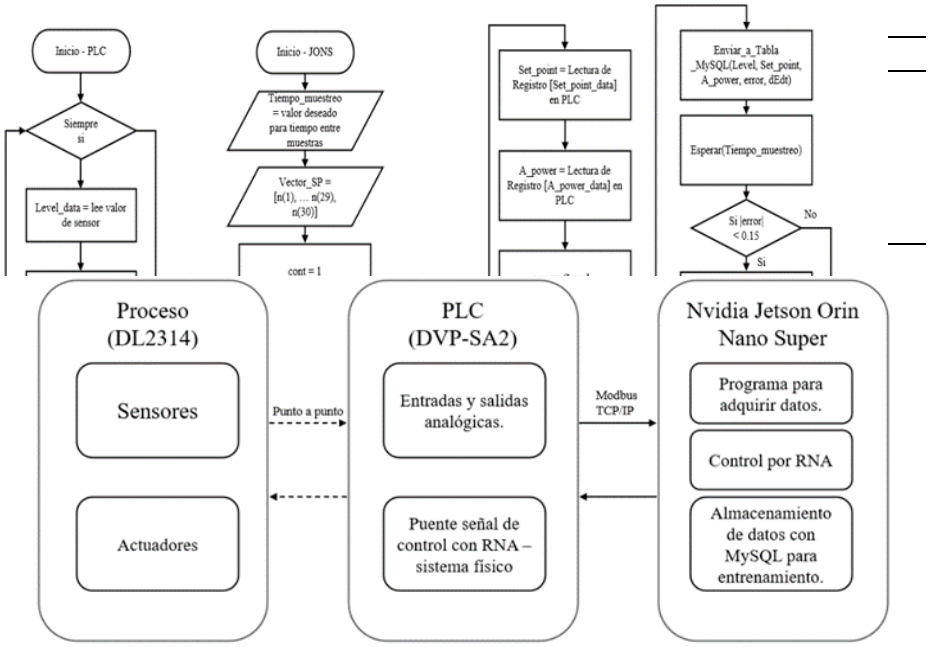

Fig. 5. Arquitectura de comunicación para ejecución de control inteligente con RNA.

Tiempo de muestreo

Con la finalidad de optimizar el almacenamiento, se debe seleccionar el tiempo de muestreo adecuado, se debe seguir un criterio que establezca el límite máximo del tiempo entre muestras. Para este caso, al ser la planta un tanque de almacenamiento de agua se considera como un sistema de primer orden [13], de tal modo que el intervalo se puede obtener a partir del ancho de banda del sistema, el tiempo de muestreo debe permitir tomar datos entre 20 y 40 veces el valor considerado del ancho de banda [14].

A partir de lo anterior, se establece el tiempo para la frecuencia del ancho de banda cuando el valor de la señal indique el 63.2% del cambio total, dada una entrada del tipo escalón con la potencia para permitir que el sistema llegue a un estado estable. Dicho valor de tiempo se obtuvo de manera experimental. De lo anterior se obtuvo un tiempo de 4 segundos entre muestras a guardadas en la base de datos. En la Tabla 1 se encuentra un resumen de la configuración utilizada para el entrenamiento de la red neuronal.

4. Validación

Para la validación de la base de datos como funcional para el entrenamiento de redes neuronales capaces de ejecutar tareas de control, se sustituyó el control PID ejecutado por el PLC con la red neuronal entrenada en ejecución sobre la tarjeta JONS, sobre la misma estructura de comunicación. se puede ver un esquema en la Fig. 5. Para la

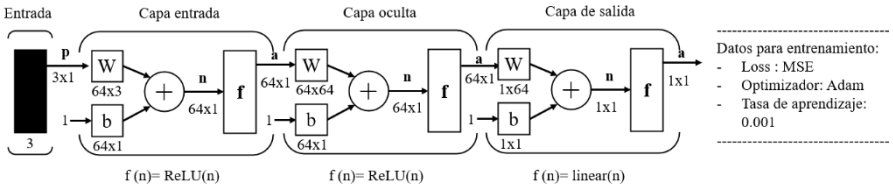

Fig. 6. Arquitectura de RNA utilizada para ejecución de control inteligente.

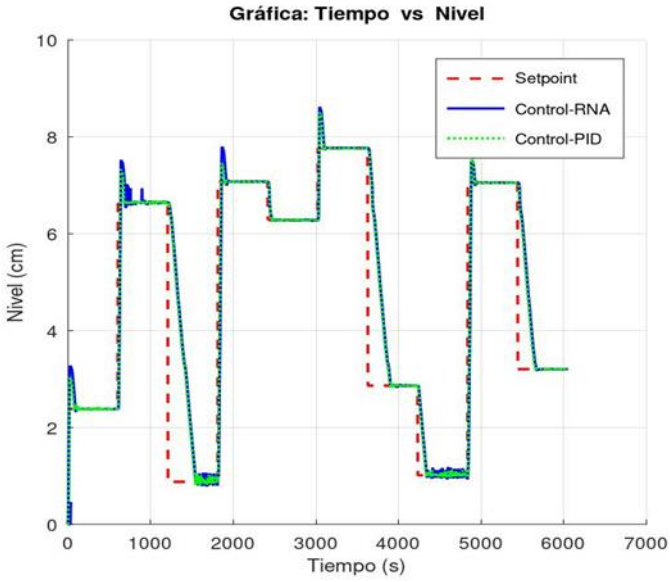

Fig. 7. Gráfica comparativa *Setpoint* /Control-RNA/Control-PID.

configuración de la red neuronal se utilizó Python y la librería Tensorflow versión 2.20. La red neuronal se entrenó un total de 809 épocas, por el mismo equipo de hardware encargado de ejecutarla. La estructura de la red neuronal utilizada se puede ver en la Fig. 6.

4.1 Resultados

Una vez entrenada la red neuronal se realizaron pruebas sobre diez valores deseados distintos, elegidos de forma aleatoria. Se comparo con el control PID, el comportamiento de ambos se puede ver en la Fig. 7. Para evaluar el comportamiento se utilizaron criterios vistos en trabajos relacionados [15], en este caso se utilizó el error cuadrático medio (MSE), el error absoluto medio (MAE) y el error estándar (SE). Además, se agregó la evaluación de resultados mediante la prueba de rangos con signos de Wilcoxon sobre el error absoluto de cada controlador respecto al *setpoint* [16].

Tabla 2. Comparativa de control con RNA y PID, con parámetro MSE.

Valor deseado (cm)	Control con RNA (cm^2)	Control con PID (cm^2)
2.380	0.00003	0.00003
6.640	0.00028	0.00042
0.875	0.00454	0.00372
7.065	0.00003	0.00003
6.280	0.00024	0.00010
7.755	0.00012	0.00003
2.860	0.00006	0.00036
1.010	0.00405	0.00209
7.045	0.00002	0.00001
3.200	0.00002	0.00003

Tabla 3. Comparativa de control con RNA y PID, con parámetro MAE.

Valor deseado (cm)	Control con RNA (cm)	Control con PID (cm)
2.380	0.0043	0.0028
6.640	0.0128	0.0178
0.875	0.0525	0.0480
7.065	0.0043	0.0035
6.280	0.0130	0.0080
7.755	0.0108	0.0043
2.860	0.0053	0.0158
1.010	0.0533	0.0335
7.045	0.0030	0.0020
3.200	0.0038	0.0043

Tabla 4. Comparativa de control con RNA y PID, con parámetro SE.

Valor deseado (cm)	Control con RNA (cm)	Control con PID (cm)
2.380	0.0011	0.0012
6.640	0.0038	0.0045
0.875	0.0147	0.0113
7.065	0.0009	0.0012
6.280	0.0032	0.0022
7.755	0.0007	0.0013
2.860	0.0017	0.0043
1.010	0.0139	0.0094
7.045	0.0009	0.0007
3.200	0.0008	0.0012

Tabla 5. Resumen de resultados de prueba de rangos con signo de Wilcoxon. Para la prueba se consideró Error_PID – Error_RNA.

Parámetro	Valores
Numero de diferencias no nulas	1229
W^+	582477
W^-	173358
W	173358
p	< 0.0001

Para los cálculos de evaluación en las tablas se consideraron las ultimas 20 muestras previas al cambio de valor deseado.

En la Tabla 2 se puede observar como la red neuronal supera al 30% y empata 20% de los casos al control PID. Por otro lado, en la Tabla 3, la red neuronal solo supera en 30% de los casos al control PID. Con el parámetro del error estándar plasmado en la Tabla 4, en 60% de los casos el control PID es superado por el control con RNA.

La prueba de rangos con signo de Wilcoxon aplicada sobre el error absoluto de ambos controladores, resumida en la Tabla 5, muestra que los valores representados por W^+ son mayores a los representados por W^-. Al representar a los errores de los controladores PID y la RNA respectivamente, se indica que el controlador PID presenta mayor error en la mayoría de los puntos evaluados.

5. Conclusión

En el presente trabajo se estableció una arquitectura de comunicación que permite, a través de un protocolo de comunicación abierto, integrar tres sistemas individuales: un proceso con sensores y actuadores, un controlador lógico programable como servidor con la capacidad de lectura de sensores y activación de actuadores y un tercer dispositivo como cliente capaz de almacenar y procesar datos de manera local, así como de ejecutar tareas de control inteligente.

Se generó una base de datos con información suficiente para el entrenamiento de una red neuronal, basándose en información generada por un control PID. La información fue almacenada en una tabla relacional gestionada por MySQL. Además, se estableció un tiempo de muestreo adecuado para optimizar el almacenamiento de los datos.

Se creó un control con redes neuronales artificiales entrenado con los datos del PID, posteriormente se implementó, evaluó y comparó con el mismo control utilizado para entrenarse. Los resultados muestran que la metodología para la creación de una base de datos es efectiva al alcanzar valores, con el control con RNA, cercanos a los valores del control utilizado para su entrenamiento, inclusive superándolo en algunos casos.

Agradecimientos. Se agradece al Centro de Investigación, Innovación y Desarrollo Tecnológico (CIIDETEC) en Culiacán por facilitar el uso de instalaciones y equipos para el desarrollo de este trabajo.

Referencias

1. Sharma, M., Tomar, A., Hazra, A.: Edge Computing for Industry 5.0: Fundamental, Applications, and Research Challenges. IEEE Internet Things J., Vol. 11, pp. 19070–19093 (2024). Doi:10.1109/JIOT.2024.3359297.
2. Abiodun, O.I., Jantan, A., Omolara, A.E., Dada, K.V., Mohamed, N.A.E., Arshad, H.: State-of-the-art in artificial neural network applications: A survey. Heliyon. 4, e00938 (2018). Doi:10.1016/j.heliyon.2018.e00938.
3. Sumarlin, T., Kusumajaya, R.A.: AI Challenges and Strategies for Business Process Optimization in Industry 4.0: Systematic Literature Review. Journal of Management and Informatics. Vol. 3, pp. 195–211 (2024). Doi:10.51903/jmi.v3i2.25.
4. Sahrir, N.H., Basri, M.A.M.: Intelligent PID Controller Based on Neural Network for AI-Driven Control Quadcopter UAV. International Journal of Robotics and Control Systems. Vol. 4, pp. 691–708 (2024). Doi:10.31763/ijrcs.v4i2.1374.
5. Yakout, A.H., Dashtdar, M., Aboras, K.M., Ghadi, Y.Y., Elzawawy, A., Yousef, A., Kotb, H.: Neural Network-Based Adaptive PID Controller Design for Over-Frequency Control in Microgrid Using Honey Badger Algorithm. IEEE Access. Vol. 12, pp. 27989–28005 (2024). Doi: 10.1109/ACCESS.2024.3367288.
6. Radlbauer, E., Moser, T., Wagner, M.: Designing a System Architecture for Dynamic Data Collection as a Foundation for Knowledge Modeling in Industry. Applied Sciences (Switzerland). Vol. 15, pp. 1–28 (2025). Doi: 10.3390/app15095081.
7. Halenar, I., Halenarova, L., Tanuska, P., Vazan, P.: Machine Condition Monitoring System Based on Edge Computing Technology. Sensors. Vol. 25, (2025). Doi:10.3390/s25010180.
8. Wahlgård, M., Krajnik, P., Stahre, J.: A Scalable Edge Computing Solution for Real-Time Process Monitoring and Optimization in Industrial IoT. Procedia CIRP., Vol. 134, Vol. 903–908 (2025). Doi:10.1016/j.procir.2025.02.219.
9. Tapia, E., Sastoque-Pinilla, L., Lopez-Novoa, U., Bediaga, I., López de Lacalle, N.: Assessing Industrial Communication Protocols to Bridge the Gap between Machine Tools and Software Monitoring. Sensors, Vol. 23, pp. 1–15 (2023). Doi:10.3390/s23125694.
10. Procesos, E.D.E.C.D.E.: Entrenador de control de procesos DL 2314 (2024)
11. Nvidia Corporation: NVIDIA Jetson Orin Nano Developer Kit (2023)
12. Ziegler, J.G., Nichols, N.B.: Optimum settings for automatic controllers. Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME, Vol. 115, pp. 220–222 (1993). Doi:10.1115/1.2899060.
13. Ogata, K.: Dinamica de sistemas. Pretince Hall (2010)
14. Franklin, G., Powell, D., Emami-Naeini, A.: Feedback Control of Dynamic Systems. Pearson (1986). Doi:10.1109/MCS.1986.1105084.
15. Gerardo-Parra, C., Barreto-Salazar, L.E., Flores-López, L.M., Picos-Ponce, J.C., Castro-Palazuelos, D.E., Rubio-Astorga, G.J.: Development of a Neural Network-Based Controller for a Greenhouse Irrigation System at Laboratory Scale. Agriculture, Vol. 16, pp. 245 (2026). Doi:10.3390/agriculture16020245.
16. Montgomery, Douglas C; Runger, G.C.: Probabilidad y Estadistica aplicadas a la ingeniería (2000)