Universidad Nacional
de Educación a Distancia

Escuela Técnica Superior de
Ingeniería Informática

Universidad
Complutense de Madrid

Facultad de Informática

Desarrollo de una Herramienta Pedagógica de Bajo Costo para la Enseñanza de Tecnologías Emergentes.

Patricio Alejandro Rojas Carrasco

Directora: Dra. Maria Guinaldo Losada

Trabajo de Fin de Máster

Máster Universitario
en Ingeniería de Sistemas y Control

Septiembre, 2024

Agradecimientos

Quisiera dedicar este trabajo a Claudia, mi esposa, y a Camila, mi hija, quienes desde el primer momento me brindaron su apoyo incondicional y, en los momentos de dificultad, supieron motivarme para seguir adelante.

Agradezco profundamente a mi directora, la Dra. María Guinaldo Losada, por su constante apoyo. Desde que solicité trabajar bajo su dirección, he contado con su invaluable colaboración.

Resumen

Este documento presenta una revisión bibliográfica exhaustiva sobre técnicas para el desarrollo de una herramienta pedagógica de bajo costo destinada a la enseñanza de tecnologías emergentes, integrando Inteligencia Artificial, Realidad Virtual y Gemelos Digitales en un contexto educativo. La revisión se centra en el uso de hardware accesible, como la Raspberry Pi y el ESP8266, junto con algoritmos avanzados de control y optimización, como redes neuronales artificiales, aprendizaje por refuerzo y búsqueda aleatoria, con el fin de crear un entorno interactivo y práctico de aprendizaje.

El estado actual de estas tecnologías es revisado detalladamente, destacando sus aplicaciones, desafíos y avances en diversos sectores industriales. Además, se analizan los materiales necesarios para el desarrollo de la herramienta, incluyendo microcontroladores, sensores y software específico, así como la implementación de un controlador PID y técnicas de optimización. Los resultados de esta revisión muestran las capacidades y limitaciones de los métodos utilizados, así como las mejoras potenciales en la precisión y eficiencia del control.

El trabajo concluye proponiendo la necesidad de un análisis más profundo de las configuraciones de hardware y la combinación de diferentes algoritmos para optimizar los resultados en la enseñanza y la práctica. Se sugieren direcciones futuras de investigación, incluyendo la integración de estas tecnologías emergentes en plataformas educativas reales, explorando su impacto en el aprendizaje y su potencial para mejorar la formación en ingeniería y tecnologías avanzadas.

Palabras clave: Tecnologías Emergentes, Optimización de Control, Gemelo Digital

Abstract

This document presents a comprehensive literature review on techniques for developing a low-cost pedagogical tool aimed at teaching emerging technologies, integrating Artificial Intelligence, Virtual Reality, and Digital Twins in an educational context. The review focuses on the use of accessible hardware, such as the Raspberry Pi and the ESP8266, along with advanced control and optimization algorithms, such as artificial neural networks, reinforcement learning, and random search, to create an interactive and practical learning environment.

The current state of these technologies is thoroughly reviewed, highlighting their applications, challenges, and advancements in various industrial sectors. Additionally, the materials necessary for developing the tool are analyzed, including microcontrollers, sensors, and specific software, as well as the implementation of a PID controller and optimization techniques. The results of this review show the capabilities and limitations of the methods used, as well as potential improvements in control accuracy and efficiency.

The study concludes by proposing the need for a deeper analysis of hardware configurations and the combination of different algorithms to optimize results in teaching and practice. Future research directions are suggested, including the integration of these emerging technologies into real educational platforms, exploring their impact on learning, and their potential to enhance training in engineering and advanced technologies.

Keywords: Emerging Technologies, Control Optimization, Digital Twin

Índice general

Índice de figuras

Índice de Tablas

Capítulo 1
Introducción

En los últimos años, hemos sido testigos del desarrollo acelerado de tecnologías cada vez más avanzadas, que hace solo unas décadas parecían sacadas de la ciencia ficción pero que hoy son una realidad tangible. Desde robots con capacidades humanas hasta máquinas que colaboran estrechamente con los trabajadores en la industria, el avance tecnológico ha sido impresionante. Este progreso se puede dividir en cuatro revoluciones tecnológicas principales, cada una marcada por hitos significativos en la historia de la humanidad y su relación con la tecnología:

1. La Mecanización, caracterizada por la aparición de las primeras máquinas a vapor, la energía hidráulica y la mecanización industrial.

2. La Era de la Electricidad, que vio la llegada de la producción en masa, la cadena de montaje y la adopción generalizada de sistemas eléctricos.

3. La Revolución Informática, marcada por la automatización de procesos, la liberación de los trabajos rutinarios del hombre y el surgimiento de las tecnologías de la información (TI).

4. La Digitalización, en la que nos encontramos actualmente, se caracteriza por la interconexión de dispositivos a través de Internet (IoT), la computación en la nube (Cloud computer), los sistemas ciberfísicos y la robótica avanzada.

Se prevé que la próxima revolución, conocida como Industria 5.0, estará centrada en la integración humana, digital y ambientalmente empática. Algunas empresas ya están liderando este cambio hacia una industria más sostenible e inclusiva.

A pesar de estos avances tecnológicos, existe una brecha significativa en la formación del capital humano, lo que ha llevado a disparidades en diversos sectores industriales. Con el objetivo de reducir esta brecha, este trabajo se centra en la integración de tres tecnologías clave: Inteligencia Artificial, Realidad Aumentada y Gemelos Digitales. Se busca proporcionar una plataforma que permita comprender y aprender cómo integrar estas tecnologías de manera independiente en diversos sectores, utilizando equipos de bajo costo y software libre.

A lo largo de este trabajo, se introducirán conceptos fundamentales para comprender cada una de estas tecnologías y cómo interactúan entre sí. El objetivo final es desarrollar un sistema integrado de tecnologías aplicables en diferentes contextos industriales.

Primero se abordan temas como las redes neuronales artificiales, sus tipos y clasificación; definiciones asociadas a realidad virtual y realidad mixta; para concluir con los gemelos digitales y su relación con el Internet de las cosas.

A continuación, se exponen trabajos realizados en las líneas de estudio. Identificando cuales han sido las mayores dificultades que han tenido que enfrentar los y las investigadoras, cuáles han sido las soluciones que han planteado y qué desafíos quedan pendientes.

La solución propuesta pretende insertarse en la formación de capital humano técnico, en cualquiera de sus niveles.

1.1. Motivación

1.1.1. Inteligencia Artificial

Para superar las barreras que limitan su potencial, la humanidad ha desarrollado un campo específico conocido como Inteligencia Artificial (IA). Este campo no solo busca comprender la inteligencia, sino también crear entidades inteligentes. La importancia de la IA radica en su capacidad para sintetizar y automatizar tareas intelectuales, haciendo que sea relevante para cualquier campo de actividad humana que requiera procesos de pensamiento avanzados.

Para comprender qué es la Inteligencia Artificial, podemos considerar el concepto de racionalidad propuesto por [1]: "Un sistema es racional si hace 'lo correcto' en función de su conocimiento". Aunque esta definición puede llevar a debates filosóficos, no forman parte de este estudio.

Si se pretende desarrollar un sistema inteligente, la lógica se erige como la herramienta fundamental. El término 'silogismo', acuñado por Aristóteles, se refiere a los "esquemas de estructuras argumentativas mediante los cuales siempre se llega a conclusiones correctas si se parte de premisas correctas" [1]. Los silogismos son fundamentales para definir estrategias de aprendizaje basadas en la lógica.

Los agentes, por otro lado, son sistemas que constan de entradas, una unidad de procesamiento y salidas. Así, un agente racional se define como aquel que "...actúa con la intención de alcanzar el mejor resultado o, cuando hay incertidumbre, el mejor resultado esperado" [1]. Este concepto se aplica tanto en hardware como en software, ejemplificado en sistemas robóticos y videojuegos.

Un aspecto destacado es comprender las capacidades de la Inteligencia Artificial. Winston et al. (1993) dedican un capítulo completo a esta temática, donde se resumen varias aplicaciones:

"Los sistemas inteligentes pueden ayudar a los expertos a resolver problemas de análisis difíciles" [2], como el programa KAM utilizado por astrónomos para estudiar el movimiento de estrellas.

"Los sistemas inteligentes pueden ayudar a los expertos a diseñar nuevos dispositivos" [2], como el programa de Karl Ulrich para diseñar componentes de aviones.

"Los sistemas inteligentes pueden aprender de ejemplos" [2], como el sistema desarrollado por J. Ross Quinlan para diagnosticar hipotiroidismo basado en la minería de datos.

"Los sistemas inteligentes pueden proporcionar respuestas a preguntas en inglés utilizando datos estructurados y texto libre" [2], ejemplo claro de los chatbots como ChatGPT.

Según Winston et al.(1993), la Inteligencia Artificial está evolucionando hacia aplicaciones menos evidentes pero más esenciales. Mientras que las primeras aplicaciones fueron impulsadas por investigadores para demostrar su valor, ahora empresarios buscan objetivos comerciales estratégicos.

Para implementar estos sistemas, existen diversos métodos y técnicas. Russell et al. (2004) mencionan tipos de búsqueda, problemas de satisfacción de restricciones, agentes lógicos, razonamiento probabilístico y métodos estadísticos de aprendizaje. Winston et al. (1993) añaden "Redes Semánticas y Coincidencia de Descripciones", "Generación y Prueba", "Análisis de Medios y Fines", "Redes y Búsqueda Básica", entre otros.

En resumen, la Inteligencia Artificial representa un campo en constante evolución que aprovecha tanto principios matemáticos como computacionales para desarrollar sistemas inteligentes capaces de aprender y tomar decisiones en una variedad de contextos.

1.1.2. Redes Neuronales Artificiales

Las Redes Neuronales Artificiales (RNA) son un ejemplo clave de la inteligencia artificial que imita el funcionamiento de las redes neuronales biológicas para la toma de decisiones. Utilizan la retropropagación (Backpropagation) para optimizar su rendimiento.

Para entender su operación, es crucial compararlas con las redes neuronales biológicas. Isasi et al. (2004) y Hilera et al. (1995) ofrecen una visión detallada de sus aplicaciones prácticas y teóricas respectivamente.

Las neuronas biológicas tienen receptores para captar estímulos, un sistema nervioso para procesar y enviar información, y efectores para interpretar y responder a la información. La neurona, fundamental en este sistema, tiene un comportamiento "todo o nada" y se comunica mediante sinapsis químicas.

A diferencia de los programas de computación convencionales, las máquinas neuronales como las RNA procesan información de entrada para generar una salida, adaptándose según la estructura y función de la red neuronal.

Las RNA pueden clasificarse según topología y tipo de aprendizaje, como monocapa, multicapa, supervisado, no supervisado, heteroasociativo y autoasociativo. Estas configuraciones permiten a las RNA adaptarse a diferentes problemas y contextos.

En resumen, las RNA son herramientas poderosas en inteligencia artificial, modelando sistemas de decisión inspirados en la biología neuronal y aplicables en diversas áreas teóricas y prácticas.

1.1.3. Aprendizaje por refuerzo

1.1.3.1. Programación Dinámica

La programación dinámica es una técnica matemática valiosa para la toma de decisiones secuenciales interrelacionadas, proporcionando un procedimiento sistemático para determinar la combinación óptima de decisiones. A diferencia de la programación lineal, la programación dinámica no tiene una formulación matemática estándar del problema, sino que es un enfoque general que se ajusta a situaciones particulares. Requiere creatividad y un buen conocimiento de la estructura general de estos problemas para identificar cuándo y cómo un problema puede resolverse mediante estos procedimientos.

La expresión programación dinámica (PD): "se emplea para describir un conjunto de algoritmos que se pueden emplear para determinar políticas óptimas en situaciones donde se dispone de un modelo preciso del entorno, como un proceso de decisión de Markov (MDP). La relevancia de la programación dinámica y del aprendizaje por refuerzo "es el uso de funciones de valor para organizar y estructurar la búsqueda de políticas óptimas…, que satisfacen las ecuaciones de optimalidad de Bellman" [3]:

$$v_s = max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a]$$

$$= max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_*(s')]$$

$$q_*(s, a) = \mathbb{E}[R_{t+1} + \gamma max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a]$$

$$= \sum_{s',r} p(s', r|s, a)[r + \gamma max_{a'} q_*(s', a')]$$

Para todo $s \in \mathcal{S}$ s∈S, $a \in \mathcal{A}$ a∈A(s), y $s \in \mathcal{S}^+$. los algoritmos de programación dinámica se derivan al transformar las ecuaciones de Bellman en asignaciones, es decir, en reglas de actualización diseñadas para mejorar las aproximaciones de las funciones de valor deseadas.

La idea principal de la programación dinámica es descomponer el problema en subproblemas más manejables. Los cálculos se realizan recursivamente, utilizando la solución óptima de un subproblema como dato de entrada para el siguiente. La solución del problema completo se alcanza al resolver el último subproblema. La forma en que se realizan estos cálculos recursivos depende de la descomposición del problema original, y los subproblemas suelen estar vinculados por restricciones comunes que deben mantenerse en todas las iteraciones. Desarrollar habilidades en programación dinámica implica exposición a una variedad de aplicaciones y análisis detallado de sus características comunes, ilustradas mediante numerosos ejemplos.

La programación dinámica se caracteriza por poseer tres elementos básicos:

1. Definición de las etapas.

2. Definición de las alternativas en cada etapa.

3. Definición de los estados para cada etapa.

De los tres elementos, la definición del estado suele ser la más compleja. Las aplicaciones demuestran que la definición del estado varía según la situación que se debe modelar. Sin embargo, independiente de la aplicación, [4] señala que es útil considerar las siguientes preguntas:

1. ¿Qué relaciones unen las etapas entre sí?

2. ¿Qué información se requiere para tomar decisiones factibles en la etapa actual independientemente de cómo se hayan tomado las decisiones en las etapas precedentes?

En [5], se estipulan las características de los problemas de programación dinámica:

1. El problema se puede dividirse en etapas, cada una de las cuales requiere una política de decisión

2. Cada etapa tiene un conjunto específico de estados vinculados a su inicio.

3. El efecto de la política de decisión en cada etapa es transformar el estado actual en un estado asociado con el inicio de la siguiente etapa, posiblemente de acuerdo con una distribución de probabilidad.

4. El procedimiento de solución está diseñado para encontrar una política óptima para manejar todo el problema, es decir, una receta para desarrollar la política de decisión óptima para cada etapa en cada uno de los posibles estados.

5. Dado el estado actual, una política óptima para las etapas restantes es independiente de la política adoptada en etapas anteriores. Por lo tanto, la decisión inmediata óptima depende

únicamente del estado actual y no de cómo se llegó a él. Este principio de optimalidad es fundamental en la programación dinámica.

6. El procedimiento de la solución comienza una vez que se determina la política óptima para la última etapa

7. Existe una relación recursiva que identifica la política óptima para la etapa n, dada la política óptima para la etapa $n + 1$.

8. Cuando se emplea esta relación recursiva, el procedimiento de solución comienza desde el final y avanza hacia atrás, etapa por etapa, para encontrar la política óptima para cada una, hasta que se determine la política óptima desde la etapa inicial. Esta política óptima conduce inmediatamente a una solución óptima para el problema completo, es decir, x_1^* para el estado inicial s_1, luego x_2^* para el estado inicial s_2 resultante, después x_3^* para el estado inicial s_3 obtenido, y así sucesivamente hasta x_N^* para el estado s_N final.

1.1.3.2. Evaluación de la política (Predicción)

En primer lugar, se examina el proceso de calcular la función de valor del estado v_π para una política dada π. Este procedimiento es conocido como "evaluación de política en la literatura de programación dinámica. También nos referimos a ello como el problema de predicción" [3].

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$$
$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s]$$
$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s]$$
$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r' + \gamma v_\pi(s')]$$

Donde $\pi(a|s)$ es la probabilidad de tomar la acción a en el estado s bajo la política π, y las expectativas están con subíndices π para indicar que son condicionales a seguir de la política inicial, π. Para asegurar la existencia y unicidad de v_π, se debe cumplir que $\gamma < 1$.

1.1.3.3. Mejora de la política

El objetivo de mejorar una política es permitir que el agente sea capaz de actuar más eficientemente. Lo que se busca es saber si para algunos estados s es necesario cambiar la política para elegir determinísticamente una acción a diferente de π(s). Para saber si es adecuado, o no, cambiar la política, se pueden tomar las palabras de Sutton et al [3]: "¿sería mejor o peor cambiar a la nueva política? Una forma de responder a esta pregunta

es considerar seleccionar a en s y luego seguir la política existente, π.", esto se puede expresar de la siguiente manera:

$$q_\pi(s,a) \doteq \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) | \, S_t = s, A_t = a]$$
$$q_\pi(s,a) \doteq \sum_{s',r} p(s', r \mid s, a)[r + \gamma v_\pi(s')]$$

Lo clave es si este criterio es mayor o menor a $v_\pi(s)$.

Para saber cuál es la mejor política, se utiliza el teorema de mejora de política. "Sean π y π' cualquier par de políticas deterministas tales que, para todo $s \in \mathcal{S}$,

$$q_\pi(s, \pi'(s)) \geq v_\pi(s)$$

Entonces, la política π' debe ser tan buena como, o mejor que, π. Es decir, debe obtener un retorno esperado mayor o igual desde todos los estados $s \in \mathcal{S}$:

$$v_{\pi'}(s) \geq v_\pi(s)$$ " [3]

1.1.3.4. Iteración de la política

La mejor política obtenida, de igual manera puede ser mejorada. Sutton et al [3] exponen que, dado que un MDP finito tiene un número limitado de políticas, este proceso debe llegar a converger en un apolítica óptima y una función de valor óptima en un número finito de iteraciones.

1.1.4. Realidad Virtual

1.1.4.1. ¿Qué es la realidad virtual?

Es complejo definir de manera unánime el término Realidad Virtual (RV). Según Webster's New [1989], "virtual" se define como algo que es esencial o efectivo, pero no físicamente real. En contraste, "realidad" se refiere al estado de existencia independiente de las ideas asociadas, distinguiéndose de lo meramente aparente. Como señalan Sherman et al. [6], la RV implica cuatro elementos clave: el Mundo Virtual, un entorno simulado que puede existir exclusivamente en la mente del creador o ser compartido; la Inmersión, que abarca tanto la experiencia emocional o mental como la sustitución o amplificación sensorial física proporcionada por el sistema; la Retroalimentación Sensorial, crucial para proporcionar una experiencia simulada que se acerca a la realidad física mediante retroalimentación visual y táctil; y la Interactividad, que permite que el entorno virtual responda activamente a las acciones del usuario para una experiencia auténtica y convincente. Además, la RV facilita la creación de realidades artificiales mediante simulaciones computacionales, representando fenómenos como tormentas mediante ecuaciones matemáticas resueltas en imágenes. Este enfoque colaborativo

permite a múltiples usuarios interactuar dentro del mismo espacio virtual, mejorando aún más la experiencia interactiva y colaborativa.

1.1.4.2. Realidad artificial

La realidad artificial se define como un entorno generado por computadora que simula un entorno físico real o imaginario, permitiendo a los usuarios interactuar de manera inmersiva con él. Utiliza tecnologías avanzadas como gráficos 3D, realidad virtual (VR) y realidad aumentada (AR) para crear experiencias sensoriales y perceptivas que pueden ser indistinguibles de la realidad física. Además de estos aspectos, la realidad artificial está en constante evolución, incorporando innovaciones en inteligencia artificial para mejorar la interactividad y la adaptabilidad del entorno simulado.

"Realidad artificial: una realidad artificial percibe la acción de un participante en términos de la relación del cuerpo con un mundo gráfico y genera respuestas que mantienen la ilusión de que sus acciones están teniendo lugar dentro de ese mundo." [6]

1.1.4.3. Ciberespacio

El ciberespacio es un concepto que va más allá de la simple comunicación interactiva. Además de facilitar la interacción entre personas separadas geográficamente, el ciberespacio también es un espacio donde se comparten y procesan grandes cantidades de información en tiempo real. Este entorno virtual permite no solo la comunicación directa, sino también la colaboración, el intercambio de conocimientos, la creación conjunta de contenido y la participación en comunidades globales de interés común.

"Ciberespacio: una ubicación que existe solo en las mentes de los participantes, a menudo como resultado de la tecnología que permite a personas geográficamente distantes comunicarse de manera interactiva." [6]

1.1.4.4. Realidad aumentada

La realidad aumentada es una variante de la realidad virtual que consiste en la superposición de estímulos sintéticos sobre objetos del mundo real, utilizando tecnologías como cámaras y sensores para integrar elementos virtuales en tiempo real en el entorno físico. Además de hacer visible información que sería imperceptible para los sentidos humanos, la realidad aumentada mejora la experiencia del usuario al permitir la interacción directa con objetos físicos enriquecidos con contenido digital. Esta tecnología se aplica en diversos campos como el entretenimiento, la educación, el diseño industrial y la medicina, ofreciendo nuevas formas de visualización y manipulación de datos que mejoran la comprensión y la toma de decisiones en entornos prácticos y profesionales.

1.1.4.5. Telepresencia

La telepresencia implica la capacidad de interactuar directamente con un entorno físico y remoto, haciendo uso frecuente de tecnología informática avanzada como robots controlados a distancia o sistemas de realidad virtual y aumentada. Desde la perspectiva del usuario, no existen limitaciones en cuanto a la ubicación del entorno remoto ni al tamaño del dispositivo utilizado para controlarlo. Este concepto va más allá de la mera visualización o comunicación, permitiendo una experiencia inmersiva donde el usuario se siente presente y participa activamente en eventos o situaciones que ocurren en lugares distantes. La telepresencia también puede implicar el uso de interfaces hápticas que transmiten sensaciones táctiles y auditivas, lo cual amplía aún más la sensación de estar físicamente presente en un lugar diferente al de su ubicación física actual.

1.1.4.6. Interfaz de entrada al mundo virtual

Los sistemas de realidad virtual emplean dispositivos de hardware para rastrear al usuario y proporcionar información esencial para una experiencia inmersiva. Estos dispositivos incluyen desde mandos con sensores de movimiento hasta guantes hápticos y trajes completos de captura de movimiento. Estas herramientas permiten que las acciones del usuario, como gestos, movimientos y voz, interactúen directamente con el entorno virtual, creando una sensación de presencia y control.

En aplicaciones específicas de realidad virtual, se requieren dispositivos de entrada para recopilar información tanto del mundo virtual como del real, facilitando la exploración y la interacción. Por ejemplo, en simulaciones médicas, estos dispositivos capturan datos biométricos del usuario para proporcionar retroalimentación sobre el rendimiento en procedimientos quirúrgicos virtuales. De manera similar, en el diseño arquitectónico, los usuarios utilizan dispositivos de entrada para interactuar con modelos virtuales, manipular objetos y evaluar diseños antes de la construcción física.

1.1.4.7. Interfaz de salida al mundo virtual

Las pantallas sensoriales son vitales como interfaces para el mundo virtual, permitiendo a los usuarios interactuar eficazmente con entornos digitales. Las pantallas estacionarias ofrecen una visualización constante y estable, ideal para aplicaciones donde la precisión y la consistencia son cruciales. Por otro lado, las pantallas basadas en la cabeza proporcionan una experiencia inmersiva al ajustarse dinámicamente a los movimientos de la cabeza del usuario, facilitando una exploración natural del entorno virtual. En contraste, las pantallas basadas en la mano permiten una interacción táctil directa con objetos virtuales debido a su movilidad, mejorando así la sensación de realismo y

precisión en las manipulaciones virtuales. Cada tipo de pantalla presenta ventajas y limitaciones específicas en términos de ofrecer una experiencia sensorial completa y efectiva en el mundo virtual, dependiendo del contexto de uso y los objetivos del diseño interactivo.

1.1.4.8. Renderizando el mundo virtual

El renderizado es un proceso fundamental en la creación de entornos virtuales, donde se generan imágenes sensoriales que representan mundos digitales. En contextos como la realidad virtual y otros medios interactivos generados por computadora, la velocidad de producción de estas imágenes es crucial para garantizar una experiencia fluida y envolvente. Este proceso, conocido como renderizado en tiempo real, se refiere a la capacidad de generar y mostrar imágenes de manera instantánea y continua, permitiendo a los usuarios percibir el entorno virtual como una experiencia dinámica y coherente, en lugar de una serie de instantáneas estáticas.

La creación de imágenes sensoriales para la realidad virtual se estructura en dos fases fundamentales. En la primera etapa, se toman decisiones críticas sobre cómo se percibirá, se escuchará y se sentirá el entorno virtual por parte del participante, marcando así el inicio de la creación conceptual del mundo virtual. La segunda etapa se centra en la implementación práctica de estas decisiones en los sistemas de renderizado, tanto en términos de software como de hardware. Estas dos etapas están estrechamente interrelacionadas, dado que las capacidades tecnológicas de los sistemas determinan el tipo y la calidad de los elementos que pueden representarse en tiempo real dentro del entorno virtual.

1.1.4.9. Interactuando con el mundo virtual

La interacción en entornos generados por computadora se define como la acción mutua o influencia recíproca entre el usuario y el sistema. En estos contextos, las acciones del usuario provocan respuestas específicas por parte del sistema informático. La manera en que interactuamos con una computadora depende en gran medida del diseño de la interfaz de usuario (UI). De hecho, nuestra experiencia con la computadora a través de la interfaz es fundamental para cómo concebimos y percibimos el dispositivo en sí mismo. A lo largo del tiempo, la percepción visual de cómo se ve una computadora ha evolucionado, reflejando los principales métodos de entrada y salida de cada era tecnológica.

Según Webster's [1989], la interacción se define como "acción o influencia recíproca". Por tanto, interactuar con un entorno generado por computadora implica que las entradas

del usuario sean respondidas por acciones correspondientes de la computadora. La manera en que interactuamos con una computadora es el resultado directo del diseño de la interfaz de usuario (UI). De hecho, nuestra conexión con la computadora a través de la interfaz es tan predominante que la UI es cómo generalmente imaginamos el propio dispositivo computacional. Buxton [1995] analizó cómo la percepción visual de una computadora ha cambiado con el tiempo, encontrando que esta imagen refleja consistentemente los principales métodos de entrada y salida de cada periodo histórico tecnológico.

1.1.5. Gemelos Digitales

El Internet de las Cosas (IoT) está revolucionando la interacción con el entorno digital al facilitar la conexión global de dispositivos y la generación continua de datos. Este avance, impulsado por líderes visionarios como Steve Jobs, subraya que las actuales innovaciones en IoT pronto se convertirán en parte integrante de nuestra vida diaria. La integración del IoT con el aprendizaje automático amplía significativamente las capacidades tecnológicas al permitir el monitoreo en tiempo real y la comprensión profunda de infraestructuras críticas. Un aspecto destacado es el uso de gemelos digitales, que son modelos virtuales precisos de sistemas físicos. Estos gemelos digitales permiten simular y optimizar el rendimiento de dispositivos y procesos en entornos virtuales antes de implementar cambios en el mundo real, mejorando así la eficiencia y la efectividad de las operaciones.

1.1.5.1. Tecnología de los Gemelos Digitales (DT).

En el ámbito del aprendizaje automático aplicado al Internet de las Cosas (IoT), la recolección de datos es crucial para impulsar modelos predictivos y analíticos. Esta tarea se lleva a cabo mediante dos métodos distintos: datos de flujo y datos acumulados. Cada tipo de datos tiene sus propias características y aplicaciones específicas, influenciadas por la necesidad de respuestas en tiempo real o la capacidad de análisis retrospectivo. En este contexto, es fundamental entender cómo se recolectan y utilizan estos datos en diversos campos, desde la conducción autónoma hasta el análisis financiero.

“Los datos de IoT para el aprendizaje automático se recopilan utilizando dos métodos diferentes, a saber, flujo y acumulado. Los datos de flujo se capturan o generan en intervalos cortos, mientras que los datos acumulados se envían en bloque para su análisis, almacenamiento o predicción. Estos dos tipos de datos son diferentes en su núcleo; por lo tanto, su uso también es diferente. El tiempo de respuesta en muchas aplicaciones debe ser en tiempo real o casi en tiempo real; por lo tanto, los datos acumulados no serán útiles en tales circunstancias. Los datos acumulados se utilizan donde los datos analíticos pueden ser utilizados incluso después de ciertos días u horas

después de la generación. Por ejemplo, un sistema de automóvil autónomo no puede depender de datos acumulados. Debe basarse en predicciones en tiempo real, que pueden responder en pocos milisegundos a unos pocos segundos. Por otro lado, en sistemas de recomendación, análisis de video, pronóstico del clima, diagnóstico médico y finanzas, tales áreas pueden utilizar datos acumulados para el análisis." [7]

En el contexto del aprendizaje automático aplicado al Internet de las Cosas (IoT), la transmisión de datos juega un papel fundamental en la generación de modelos predictivos y analíticos. Esta tarea se realiza típicamente mediante el uso de sistemas de computación en la nube de alto rendimiento, que aprovechan el paralelismo de datos y un marco de procesamiento incremental. Estas técnicas permiten el análisis eficiente de conjuntos de datos extensos al dividirlos en partes más pequeñas para análisis simultáneos, o al procesar lotes de datos de manera incremental para reducir la latencia temporal. Sin embargo, es importante entender que, si bien estas estrategias son efectivas para optimizar el procesamiento de datos, pueden no ser la solución más adecuada para aplicaciones de IoT que requieren respuestas en tiempo real. "Los datos de transmisión de IoT para el aprendizaje automático suelen implementarse sobre la base de computadoras en la nube de alto rendimiento, paralelismo de datos y un marco de procesamiento incremental, utilizados ampliamente. El paralelismo de datos se realiza dividiendo un extenso conjunto de datos en conjuntos de datos más pequeños, en los que se pueden realizar análisis simultáneamente. Por el contrario, en el procesamiento incremental, se recupera un pequeño lote de datos y se procesa en un canal de procesamiento computacional lo más rápido posible. Sin embargo, estas técnicas reducen la latencia temporal pero no son la mejor solución para aplicaciones de IoT en tiempo real." [7]

Por lo descrito en el párrafo anterior, es que "Los investigadores están tratando de acercar el análisis de datos de transmisión a la fuente (dispositivo IoT). Cuanto más cerca esté la computación/análisis de la fuente, menos sensato será utilizar el paralelismo y el procesamiento incremental." [7], esto redundará en la posibilidad de utilizar hardware de menor prestaciones que los requeridos actualmente, posibilitando acercar la tecnología a nuevas áreas. Sin embargo, esto conlleva nuevos desafíos: "…como limitaciones de computación, menos almacenamiento y menos recursos de energía. El aprendizaje profundo es un subconjunto del aprendizaje automático, que consiste en algoritmos supervisados y no supervisados para entrenar sistemas. Estos algoritmos se categorizan además en arquitecturas específicas, y todas estas arquitecturas consisten en múltiples capas. Cada capa es responsable de producir una respuesta no lineal basada en la entrada de la capa anterior. Los sistemas de aprendizaje profundo se asemejan estrechamente al cerebro humano para resolver problemas complejos." [7]

1.1.5.2. Gemelos Digitales con uso de redes neuronales convoluciones (CNN) e IoT

Las redes neuronales convolucionales (CNN) son altamente eficientes en el procesamiento de datos de píxeles, especialmente en el reconocimiento de imágenes. A diferencia de las técnicas convencionales, las CNN pueden aprender características transformacionales como rotación o reflexión gracias a sus capas ocultas, que combinan capas de convolución y totalmente conectadas. Esto las hace especialmente útiles en dispositivos de IoT que emplean cámaras como sensores, como drones, vehículos conectados inteligentes y teléfonos inteligentes.

"Las redes neuronales convolucionales (CNN) son conocidas por procesar datos de píxeles y funcionan maravillosamente con el reconocimiento de imágenes. La tarea convencionalmente basada en la visión en DL falló en aprender características que podrían transformarse, como la rotación, la translación o la reflexión, pero las CNN superaron estos problemas utilizando capas ocultas. Las capas ocultas son una combinación de capas de convolución y capas totalmente conectadas al final. La capa de convolución está en el núcleo de una CNN. El beneficio de las CNN en IoT es que muchos dispositivos de IoT utilizan cámaras como sensor para detectar el entorno que los rodea. Los productos que dependen principalmente de las cámaras son drones, automóviles conectados inteligentes y teléfonos inteligentes." [7]

1.1.5.3. Gemelos Digitales con uso de redes neuronales recurrentes (RNN) e IoT

En el contexto del Internet de las Cosas (IoT), las redes neuronales recurrentes (RNN) han surgido como una arquitectura fundamental en el ámbito del aprendizaje profundo. Su capacidad para funcionar en tiempo real y predecir resultados basados en datos previos las convierte en una herramienta invaluable para una variedad de aplicaciones en IoT. En este sentido, las RNN son especialmente útiles en situaciones donde se requiere analizar secuencias de datos en tiempo real, ya que pueden capturar patrones y tendencias basadas en múltiples muestras anteriores. Esta introducción proporcionará una visión general de cómo las RNN están transformando la analítica de datos en IoT, con ejemplos de aplicaciones prácticas en predicción de demanda energética y seguridad del hogar, entre otros.

"Las redes neuronales recurrentes (RNN) son la arquitectura de DL más útil en IoT. RNN funciona perfectamente en tiempo real, por lo que la analítica de datos en tiempo real en muchos dispositivos importantes depende de RNN. RNN es conocido principalmente por su propiedad de predecir el resultado, basado en varias muestras anteriores, lo que lo convierte en un candidato perfecto para analizar la secuencia de

entradas y la tasa de cambio en muchos datos en tiempo real. RNN utiliza la retropropagación a través del tiempo (BPTT) para entrenar la red. RNN se utiliza en aquellas aplicaciones donde las muestras individuales no son suficientes, pero las secuencias de entradas juegan un papel vital. … . La predicción de la demanda energética para el hogar inteligente en el marco de la red inteligente es uno de los productos basados en RNN con IoT. Otros productos son el sistema de detección de intrusiones y la autenticación basada en la respiración en dispositivos con recursos limitados." [7]

1.1.5.4. Gemelos Digitales con uso de memoria a largo plazo y corto plazo (LSTM) e IoT

Las redes neuronales de largo corto plazo (LSTM) son herramientas esenciales en el análisis predictivo del Internet de las Cosas (IoT), gracias a su habilidad para manejar la larga dependencia temporal de los datos. Esto las hace ideales para aplicaciones como el reconocimiento de actividad humana y la predicción de eventos ambientales. Además, las LSTM encuentran numerosas aplicaciones en la industria IoT, desde la predicción de la calidad del agua hasta la estimación de la concentración de partículas en el aire.

"La larga dependencia en el tiempo de los datos hace que las LSTM funcionen mejor que las RNN. Esta larga dependencia hace que los datos de IoT sean candidatos perfectos para la predicción. Aplicaciones como el reconocimiento de actividad humana en deportes, la predicción de desastres en el monitoreo ambiental y las predicciones de rendimiento académico son algunas de las aplicaciones que utilizan este concepto.

LSTM tiene muchas aplicaciones industriales de IoT que producen una gran cantidad de datos. La predicción de la calidad del agua, la predicción de atributos de QoS y la predicción de la concentración de PM10 también se basan en el concepto de LSTM." [7]

1.1.5.5. Gemelos Digitales con uso de aprendizaje profundo (DL) e IoT, y otros algoritmos

El uso del aprendizaje profundo en el IoT ha alcanzado su punto máximo desde 2015. Se está explorando la combinación de esta tecnología con otras, como el aprendizaje por refuerzo, para mejorar la localización en campus universitarios y otras aplicaciones. Aunque el progreso es notable, la implementación en entornos del mundo real sigue siendo limitada.

"El uso de DL convencional con IoT ha alcanzado su punto máximo desde 2015; sin embargo, el uso de otras tecnologías junto con DL también está marcando su posición en el mundo. El aprendizaje profundo por refuerzo (DRL) es una fusión de aprendizaje por refuerzo (RL) con DNN. En IoT, DRL se utiliza para el entrenamiento semisupervisado

para realizar la localización en el campus. El aprendizaje por transferencia y el aprendizaje en línea son algunos enfoques adicionales que se combinan con DL convencional para obtener mejores resultados. Hasta el momento, el aprendizaje por transferencia con DL en IoT es un tema de investigación activo, y gran parte de su trabajo aún se encuentra en laboratorios controlados que no se han utilizado en el mundo real." [7] [8]

1.1.5.6. Metodología de Aprendizaje en Técnicas de Aprendizaje Automático y Aprendizaje Profundo

El Machine Learning (ML) es un componente clave de la Inteligencia Artificial que permite a los sistemas aprender de los datos disponibles sin necesidad de una programación explícita. Se centra en el desarrollo de programas capaces de identificar patrones en los datos a través de la observación directa, la experiencia acumulada o las instrucciones proporcionadas. "ML [Machine Learning] es un concepto de Inteligencia Artificial que mejora la capacidad del sistema para aprender automáticamente a partir del conocimiento oculto dentro de la información disponible en los datos. Se centra en el desarrollo de programas que pueden aprender de los datos sin necesidad de ser programados explícitamente. La naturaleza de este mecanismo de aprendizaje depende de la disponibilidad de los datos u observaciones, la experiencia directa o las instrucciones para descubrir patrones en los datos." [7]

Dentro del campo del Machine Learning (ML), se distinguen diferentes enfoques: Aprendizaje supervisado, Aprendizaje no supervisado, Aprendizaje semi-supervisado y Aprendizaje por refuerzo. Estas categorías se definen según el nivel de autonomía de la máquina en la toma de decisiones, en contraste con la intervención humana. En el Aprendizaje Supervisado, el investigador supervisa y evalúa el comportamiento del modelo. Por otro lado, en el Aprendizaje por Refuerzo, el sistema determina la corrección de sus acciones basándose en las recompensas asociadas a sus respuestas, sin requerir intervención humana.

La diferencia entre ML (Machine Learning) y DL (Deep Learning), es que en el segundo se deben realizar ciertas transformaciones: "Con el avance en el campo del ML, surgió otro subcampo de algoritmos de ML, conocido como algoritmos de DL. DL se refiere al proceso de aprender representaciones de datos de entrada principalmente a través de las transformaciones. Esto facilita realizar tareas de clasificación y predicción." [7].

La problemática de estas técnicas, a pesar de lo avanzado que son, es la capacidad de cómputo requerida. "[DL y ML] Estos algoritmos requieren mucho procesamiento y tecnología de fabricación de alto nivel que permita resolver problemas de diseño a bordo

para una utilización efectiva. Se necesitan chips altamente integrados a bajo costo para permitir que estas tecnologías se utilicen de manera efectiva." [7]

1.1.5.7. Gemelos Digitales en la industria

Los Digital Twins (Gemelos Digitales) son representaciones digitales de objetos o sistemas físicos que establecen una comunicación bidireccional con sus contrapartes físicas a través de sensores y redes. Esta integración de tecnologías de la información y comunicación permite un control más eficiente y un monitoreo detallado de diversos equipos e instalaciones.

"… un DT [Digital Twin] es una representación digital de un objeto o sistema físico. Tiene la capacidad de comunicación bidireccional con el gemelo físico a través de sensores y redes. El DT es una evolución e integración de las diversas tecnologías de la información y la comunicación (TIC) que han proliferado en la escena de la tecnología de la información durante las últimas dos décadas. Integra Internet de las cosas (IoT), big data, almacenamiento en la nube y en el borde, inteligencia artificial de las cosas (AIoT), realidad aumentada (AR), etc., para formar una red de comunicación integral para el control, monitoreo, diagnóstico e inspección de la salud de equipos e instalaciones, tráfico y sistemas de transporte, edificios, etc." [9]

Para entender cómo es posible conseguir esta tecnología, se puede revisar la explicación de Deb Barma et al [10]: "… para una entidad física o un activo, existe un equivalente digital en el mundo virtual. Con el mundo equipado con sensores, ahora es posible replicar las propiedades y características físicas en forma digital.". El objetivo para conseguir un DT es "…proporciona[r] al gemelo físico oportunidades para la optimización."

Los Digital Twins son fundamentales en todo el ciclo de vida de un producto, desde su diseño hasta su desmantelamiento, y se dividen en tres categorías principales: el Digital Twin de Producto, el Digital Twin de Producción y el Digital Twin de Rendimiento. "Los gemelos digitales refuerzan el ciclo de vida de diseño-ejecución-cambio-desmantelamiento de un producto. Naturalmente, desde la conceptualización hasta el desmantelamiento, los gemelos digitales encuentran un lugar en el ciclo de vida del producto. En términos generales, hay tres tipos de gemelos digitales: gemelo de producto, gemelo de producción y gemelo de rendimiento (términos popularizados por Siemens)." [10]

Industrias como la aeroespacial, defensa y automotriz están a la vanguardia en la implementación de Digital Twins, seguidas de cerca por sectores como manufactura,

energía, servicios públicos, infraestructura, telecomunicaciones, salud y agricultura, que están desarrollando casos de uso para impulsar sus iniciativas de la Industria 4.0.

"… las industrias con una mayor penetración de IoT, como la aeroespacial, la defensa y la automotriz, están a la vanguardia en la implementación de gemelos digitales. Reconociendo el potencial, otros segmentos como la fabricación, la energía y servicios públicos, la infraestructura, las telecomunicaciones, la atención médica e incluso la agricultura, están desarrollando casos de uso de gemelos digitales para sus iniciativas de industria 4.0." [10]

1.1.5.8. Ecosistema de los Gemelos Digitales

En la era de la transformación digital, los conceptos de Prototipo de Gemelo Digital (DTP), Instancia de Gemelo Digital (DTI) y Entorno de Gemelo Digital (DTE) han surgido como pilares fundamentales para la gestión y optimización de activos físicos. Estos elementos, derivados de la innovadora tecnología de Gemelos Digitales, están redefiniendo la manera en que se diseñan, monitorean y mantienen los activos en diversas industrias. En este contexto, exploraremos en detalle cada uno de estos conceptos y su importancia en el impulso de operaciones predictivas y eficientes.

"Prototipo de Gemelo Digital (DTP, por sus siglas en inglés): Un DTP es el prototipo del activo físico. Es como una receta para crear un activo. Los prototipos, dependiendo de la situación, contendrán información relacionada con los atributos físicos, propiedades, parámetros de operación, lista de materiales, números de pieza y más. El DTP es como la copia maestra para referencia.

Instancia de Gemelo Digital (DTI, por sus siglas en inglés): Un DTI (creado a partir del DTP) es el gemelo de un activo físico. El DTI permanece vinculado al activo físico a lo largo de su ciclo de vida. El DTI típicamente contiene datos relacionados con las condiciones de uso capturadas a través de los sensores, el estado histórico, el estado predicho, la información sobre el activo y la garantía, los registros de servicio, etc. Si bien un DTI comienza con la información base de su prototipo, a lo largo del ciclo de vida, el DTI se enriquece con datos operativos.

El Entorno de Gemelo Digital (DTE, por sus siglas en inglés) es una colección de los componentes mencionados anteriormente, el ámbito del gemelo digital para las operaciones predictivas y de interrogación." [10]

1.1.5.9. Conceptos Avanzados de Gemelos Digitales

Hilos digitales: En informática un hilo (Thread) es una tarea que se puede ejecutar al mismo tiempo que otra. Respecto a los hilos digitales (Digital Threads), el concepto es algo diferente, ya que se considera la siguiente definición: "…un hilo digital es un solo y continuo conjunto de datos que se extiende desde el diseño inicial hasta la parte terminada" [10]

La importancia de los hilos digitales se puede resumir en el siguiente párrafo: "El hilo digital permite una "vista única" de los datos de un producto o parte a lo largo de su vida. Al proporcionar una vista integrada, un hilo digital ofrece: trazabilidad completa, una versión de la verdad en múltiples sistemas y elimina los datos duplicados y discrepantes de la parte a lo largo del ciclo de vida." [10]

Sombra Digital: Es importante diferenciar conceptos, para no confundir diversas partes de un gemelo digital. Es por esto, que se pude definir la sombra digital de la siguiente forma: "Las sombras digitales tienen un flujo de datos unidireccional entre lo físico y lo digital, en comparación con el intercambio bidireccional en los gemelos digitales. Una sombra digital refleja los cambios en los objetos físicos en lo digital y no al revés." [10]

Fantasma Digital: "Los "fantasmas digitales", al detectar una anomalía, reemplazarían las "lecturas falsificadas" con las esperadas. Dado que los fantasmas digitales conocen cómo opera normalmente la máquina, pueden aislar el problema y evitar repercusiones graves en la infraestructura." [10]

1.1.5.10. Arquitectura lógica de los Gemelos Digitales

Modelo: Corresponde a la representación gráfica de los equipos físicos, existen múltiples herramientas de diseño que se pueden utilizar. "El viaje del gemelo digital comienza con la modelización del activo físico. Convertir modelos 2D existentes es una forma. Otra opción, en ausencia de modelos existentes, es comenzar desde cero. Hay una variedad de herramientas de modelización disponibles para este propósito." [10]

Comunicación: Los dispositivos, sensores y actuadores, que componen un sistema complejo deben estar comunicados entre sí. La comunicación está dada por las tecnologías de la información (TI). "Una vez que el modelo está en su lugar, el IoT permite la comunicación entre los mundos físico y digital. Los sensores y la tecnología varían según el gemelo." [10]

Asimilación: “Las capas de asimilación enriquecen los datos del activo con datos de otros sistemas para obtener una vista mejorada del mundo físico. Por ejemplo, esto podría ser información como el número de parte, repuestos en inventario, etc.” [10]

Acción: “La capa de acción habilita capacidades de pronóstico y diagnóstico. Esta capa establece un flujo de datos hacia adelante (pronóstico) y hacia atrás (diagnóstico) entre el mundo físico y digital.” [10]

1.1.5.11. Arquitectura física de los Gemelos Digitales

Sensores y Actuadores: Los sensores capturan información del proceso, en forma de una variable física, por ejemplo: temperatura, humedad, peso, etc. Los actuadores, manifiestan el comportamiento deseado en el proceso, a través de la activación de un motor eléctrico, por ejemplo.

Sistemas Operativos y Sistemas Maestros de Entrada: “Además de los datos de los sensores, la información de los sistemas operativos como el registro de incidentes, los controladores SCADA y los sistemas maestros de entrada como la gestión de activos e inventarios, mejorarán la representación digital del gemelo.” [10]

Análisis y Procesamiento de Datos: “La capa de análisis y procesamiento de datos recibe los datos, los analiza, los limpia y los normaliza para su posterior procesamiento. Las plataformas de IoT gestionan los dispositivos que recopilan datos del mundo físico.” [10]

Simulación y modelado: “El corazón de un programa de gemelo digital, la simulación y el modelado, es donde toda la información e inteligencia se unen con fines diagnósticos y pronósticos. El modelado utiliza reglas simples hasta algoritmos de inteligencia artificial complejos. La información modelada luego se simula, utilizando técnicas de simulación 3D, para el consumo y la acción del usuario.” [10]

Gestión de casos: “Basado en el propósito de los gemelos digitales, la gestión de casos puede habilitar un entorno colaborativo para que varios interesados y usuarios trabajen en las entradas y salidas.” [10]

Creación de tableros e informes: “La capa de presentación que comprende paneles e informes para los usuarios.” [10]

1.2. Propuesta y objetivos

Este trabajo pretende lograr los siguientes objetivos:

1. Proporcionar una herramienta pedagógica de bajo costo y alta eficiencia para la enseñanza de sistemas que involucran Inteligencia Artificial, Gemelos Digitales y Realidad Aumentada.

2. Aplicando técnicas modernas de inteligencia artificial, tal como el aprendizaje por refuerzo, en sistemas que involucran dispositivos IoT.

3. Ensayar distintos tipos de aprendizaje en sensores integrados en un gemelo digital, para ser enlazados con realidad virtual.

1.3. Estructura del documento

Este documento está organizado en capítulos y secciones que abordan de manera sistemática los objetivos de la investigación, los métodos empleados y los resultados obtenidos. A continuación, se presenta una descripción general de la estructura del documento:

Introducción
En este capítulo se introduce el tema de investigación, se explicita la motivación, se realiza la propuesta del trabajo y se definen los objetivos.

Revisión del Estado del Arte
Este capítulo presenta una revisión exhaustiva de la literatura existente sobre técnicas de control y optimización relevantes para la investigación. Se discuten los enfoques previos, las metodologías empleadas y los avances en el área.

Materiales y Métodos
En este capítulo se describen detalladamente los materiales empleados y la metodología de investigación. Se explican los métodos de implementación de las técnicas de control y optimización, los criterios de evaluación y los procedimientos experimentales utilizados.

Resultados
Este capítulo detalla la implementación de las técnicas estudiadas y presenta los resultados obtenidos. Se incluyen análisis y comparaciones de los diferentes métodos evaluados, así como la interpretación de los datos experimentales.

Conclusiones y Trabajos futuros
Este capítulo resume los principales hallazgos de la investigación y proporciona conclusiones basadas en los resultados obtenidos. También se presentan recomendaciones para futuras investigaciones y aplicaciones prácticas de los resultados.

Capítulo 2

Estado del Arte

2.1 Redes Neuronales Profundas y el problema de software/hardware

A pesar de que actualmente se cuenta con infraestructura en la nube para el procesamiento de datos, algunos investigadores se han dedicado a optimizar el rendimiento de las redes neuronales profundas (DNN, Deep Neural Networks) con el objetivo de implementar estos sistemas directamente en los sensores. Esto permite manejar los datos de forma local, en lugar de hacerlo en la nube, principalmente por razones de seguridad y protección de la información. Sin embargo, esto ha llevado a enfrentar nuevos desafíos. En estudios como [11], [12], [13] y [14], se destacan diversos parámetros que deben considerarse, en particular el consumo de energía debido a los cálculos necesarios.

Canziani et al (2016), indican:

"(1) Las capas completamente conectadas son en gran medida ineficientes para lotes más pequeños de imágenes. (2) La precisión y el tiempo de inferencia tienen una relación hiperbólica. (3) Las restricciones de energía son un límite superior para la máxima precisión y complejidad del modelo alcanzable. (4) El número de operaciones es una estimación confiable del tiempo de inferencia." [11].

Por otro lado, Canziani et al (2017), concluyen que:

"Los hallazgos clave son: (1) el consumo de energía es independiente del tamaño del lote y de la arquitectura; (2) la precisión y el tiempo de inferencia tienen una relación hiperbólica; (3) la restricción de energía es un límite superior para la máxima precisión y complejidad del modelo alcanzable; (4) el número de operaciones es una estimación confiable del tiempo de inferencia.

" [12].

Yang et al (2016), son claros al respecto: "El factor limitante clave es el alto consumo de energía del procesamiento de CNN debido a su alta complejidad computacional.". [13]

En [13], Yang et al (2016), se muestran los resultados del análisis de la jerarquía de memoria en plataformas de hardware modernas y el consumo de energía de DNN conocidas, demostrando que (1) el número de pesos y operaciones MAC no son métricas precisas relacionadas con el consumo de energía, (2) el movimiento de los datos es más costoso que la computación, y (3) es clave considerar el movimiento de los datos en los mapas de características. Además,

proporcionan un ejemplo de cómo la metodología propuesta puede guiar la poda de redes para mejorar el equilibrio entre precisión y consumo de energía. Con esta metodología y herramientas, los investigadores pueden evaluar los costos energéticos de diversas decisiones de diseño durante la fase de desarrollo, reduciendo la brecha entre el algoritmo y hardware de DNN y la optimización del consumo de energía.

Posteriormente, en el año 2107, Yang et (2017) presentan los resultados de su estudio [14], que el que introducen un algoritmo de poda que considera el consumo de energía de una CNN para dirigir el proceso de manera que se optimice la eficiencia energética. La energía del CNN se estima utilizando un método que modela tanto los cálculos como los accesos a memoria, utilizando datos relacionados al consumo de energía, que han sido extrapolados a partir de mediciones reales en hardware. Esto permite una estimación más precisa del consumo de energía en comparación con simplemente usar el tamaño del modelo o el número de operaciones MAC. Con la energía estimada para cada capa de un modelo CNN, el algoritmo procede a podar capa por capa, comenzando desde las capas con mayor consumo energético hasta las de menor consumo. Durante la poda de cada capa, se eliminan los pesos que tienen el menor impacto conjunto en el mapa de características de salida. Los resultados demuestran que este método de poda reduce significativamente el consumo de energía de modelos como AlexNet y GoogLeNet en comparación con sus versiones originales. Además, se examina y discute cómo la reducción del número de clases objetivo en AlexNet puede disminuir considerablemente el tamaño del modelo, aunque con una reducción limitada en el consumo energético.

Se evidencia que existen trabajos que han buscado resolver el inconveniente del consumo de energía, al trabajar con CNN, en esta línea es relevante el trabajo de Bianco et al (2018), que lleva por título “Benchmark Analysis of Representative Deep Neual Network Architectures”. En este estudio, se examina cómo el diseño de DNN cada vez más complejas mejora el rendimiento de ImageNet-1k y en otras tareas de visión. Se presenta una comparación detallada de más de 40 arquitecturas de DNN avanzadas entrenadas en ImageneNet-1k, evaluando precisión, número de parámetros, uso de memoria, complejidad computacional y tiempo de inferencia. Los principales hallazgos incluyen que la precisión de reconocimiento no aumenta proporcionalmente al incremento en el número de operaciones, destacando modelos como SE-ResNet-50 que logran alta precisión con menos operaciones. Además, se observa que la relación entre la complejidad del modelo y la precisión no es lineal, y que algunos modelos utilizan sus parámetros de manera más eficiente que otros. La capacidad de realizar inferencias en tiempo real varía significamente entre modelos, siendo los de mayor complejidad más adecuados para GPUs de alto rendimiento que para sistemas embebidos. Incluso los modelos con complejidad mínima aún requieren un consumo considerable de memoria en GPU. Todos los DNN evaluados y el software utilizado están disponibles en línea, con planes futuros de agregar gráficas interactivas para facilitar la exploración de los resultados y la inclusión de nuevas investigaciones en el repositorio.

2.2 Gemelos Digitales

Los Gemelos Digitales (DT, Digital Twins) han captado el interés de los investigadores, como se evidencia en trabajos recientes como [15], [16], [17], [18], [19], [20], entre otros, que abarcan una amplia variedad de temas relacionados con esta tecnología.

El gemelo digital ha evolucionado significamente desde su introducción inicial en la industria aeroespacial, avanzando de un concepto a una aplicación práctica en diversos campos. Así lo evidencia Tao et al (2022), en su trabajo "Digital Twin modeling", artículo que evidencia la revisión de más de 300 publicaciones sobre la modelización del gemelo digital y sus modelos resultantes. Las principales contribuciones incluyen:

1. Proporcionar estadísticas detalladas sobre la investigación actual del modelo del gemelo digital, abarcando su aplicación jerarquía, disciplina, dimensiones, universalidad y funcionalidad.

2. Analizar estudios actuales sobre la modelización del gemelo digital, enfocándose en la construcción de modelos y desafíos de integridad en la modelización.

3. Resumir tecnologías y herramientas utilizadas en diferentes aspectos de modelización del gemelo digital, ofreciendo casos de uso práctico y directrices para futuras investigaciones y desarrollos tecnológicos prometedores.

El gemelo digital está en rápida evolución, con investigaciones cada vez más específicas. Esto es evidente en el trabajo desarrollado por Liu et (2023), titulado "A systematic review of digital twin about physical entities, virtual models, twin data, and applications". Este artículo revisa 117 publicaciones. Las principales contribuciones incluyen:

1. Definición de características: Aclarar la definición del gemelo digital y diferenciarlo del sistema ciberfísico.

2. Métodos de investigación: Revisar métodos de investigación de entidades físicas, modelos virtuales y datos gemelos, incluyendo la adquisición de información física y protocolos de intercambio de datos.

3. Áreas de aplicación: Resumir las áreas de aplicación del gemelo digital, dividiéndolos en fases de diseño, operación e interacción dinámica. Se destaca el potencial de aplicaciones emergentes.

Rathore et al (2021), manifestaron las ventajas y posibilidades de escalabilidad que tiene el uso de gemelos digitales más inteligencia de máquina y *big data*. Queda de manifiesto en la publicación titulada "The role of AI, Machine Learning, and Big Data in Digital Twinning: A systematic literature review, challenges, and opportunities". En dicho trabajo, exponen el resultado de una revisión sistemática de la literatura sobre sistemas de gemelos digitales de próxima generación que utilizan aprendizaje automático e inteligencia artificial. Con la adopción de AI-ML y Big Data, el hermanamiento digital está evolucionando rápidamente y

con él están surgiendo muchos desafíos únicos y nuevas oportunidades. Además, identificaron criterios y herramientas para gemelos digitales que contribuyen a su exitoso desarrollo.

Estos aportes se han traducido, por ejemplo, en la integración de la Inteligencia Artificial (IA) en la Internet de las Cosas (IoT), conocida como AIoT, que ha ganado importancia en diversas aplicaciones como ciudades inteligentes y fabricación avanzada. Las Redes Neuronales Profundas (DNN) son fundamentales en este contexto, permitiendo el análisis automático de datos para la toma de decisiones y el reconocimiento de patrones en dispositivos IoT. Sin embargo, la implementación eficiente de DNN en entornos AIoT enfrenta desafíos debido a las limitaciones de recursos en dispositivos IoT y la necesidad de procesamiento en tiempo real. Por lo tanto, optimizar el rendimiento de la inferencia de DNN se convierte en una prioridad para garantizar la eficiencia y precisión en estas aplicaciones. Hu et al [21], lo exponen de la siguiente manera: “La inteligencia artificial de las cosas (AIoT) está recibiendo una atención cada vez mayor debido a la capacidad notable de la inteligencia artificial (IA) en el análisis de datos para aplicaciones de Internet de las cosas (IoT) como las ciudades inteligentes y la manufactura inteligente. Las redes neuronales profundas (DNN) han sido ampliamente adoptadas en AIoT debido a su capacidad de extracción automática de características para el reconocimiento de patrones o la toma de decisiones. Las DNN para aplicaciones de AIoT se despliegan comúnmente en servidores en la nube y servidores de borde, ya que los dispositivos AIoT, como las cámaras inteligentes, tienen capacidades de computación limitadas. Los servidores en la nube, que poseen capacidades de computación extensivas y datos, pueden ser aprovechados para el entrenamiento de DNN. Además, los servidores de borde, desplegados en proximidad cercana a los dispositivos AIoT, pueden procesar datos de tareas descargadas y generar resultados de procesamiento mediante inferencia de DNN con baja latencia. Con la creciente popularidad de las aplicaciones AIoT, la demanda de tareas de inferencia de DNN (referidas como tareas DNN) puede ser sustancial, por ejemplo, decenas de cuadros se generan por segundo por una cámara inteligente para el monitoreo en tiempo real de una intersección. En este sentido, optimizar el rendimiento de la inferencia de DNN mediante la utilización eficiente de los recursos de la red se convierte en un tema significativo.” [21]

2.3 Realidad Virtual y Realidad Aumentada

La realidad virtual y aumentada se encuentran en el epicentro del desafío tecnológico junto con la Inteligencia Artificial y los Gemelos Digitales. Ejemplos destacados de esta convergencia se pueden encontrar en trabajos como [22], [23], [24], y otras publicaciones relevantes.

Por un lado, "Augmented Reality Applications in Industry 4.0 Environment" de Reljic et al. (2021) destaca cómo el avance de tecnologías móviles, teléfonos inteligentes, procesadores de alta velocidad, cámaras, pantallas y otros dispositivos ha impulsado a la realidad aumentada (RA) como una tecnología líder no solo en entornos industriales, sino también en la vida diaria. El estudio describe las etapas cruciales para la implementación de la RA en empresas, subrayando la importancia de considerar requisitos amplios que abarquen aspectos no solo

tecnológicos, como la experiencia del usuario, la ergonomía y la usabilidad de las aplicaciones.

Por otro lado, Egger et al. (2020) contribuyen con su trabajo "Augmented reality in support of intelligent manufacturing – A systematic literature review". Este estudio ofrece tres contribuciones significativas: primero, identifica el estado actual de la investigación en realidad aumentada en la industria manufacturera, proporcionando una visión integral que no se limita a una sola industria o aplicación. Segundo, identifica los desafíos actuales que obstaculizan la adopción de la realidad aumentada en la industria, abarcando tanto aspectos tecnológicos como organizativos. Y tercero, identifica tendencias futuras en la investigación de realidad aumentada en la industria manufacturera.

Estos trabajos ejemplifican cómo la realidad aumentada está transformando diversos sectores industriales mediante el uso innovador de tecnologías avanzadas y la exploración de nuevas fronteras en la interacción humano-máquina.

Respecto a la implementación de estas tecnologías, Park et al. [23] presentan un método que integra la detección de objetos y segmentación de instancias basada en aprendizaje profundo con la tecnología AR portátil. Su propósito es proporcionar una guía visual más eficiente y con menor carga cognitiva. El artículo detalla cómo combinan la segmentación de instancias utilizando Mask R-CNN con AR sin marcadores para superponer el mapeo espacial 3D de objetos reales sobre su entorno circundante. Además, utiliza esta información para ofrecer guía y navegación en tareas 3D, facilitando la identificación y comprensión de objetos físicos mientras el usuario se mueve por el entorno.

El enfoque de Park et al. [23] es uno de varios enfoques propuestos para mejorar la interacción y precisión en entornos de realidad aumentada, haciendo uso de tecnologías como la visión por computadora, kits de herramientas de AR, unidades de medición inercial (IMU), sensores de profundidad y colaboraciones AR remotas. Cada método aborda diferentes aspectos de la interacción y la precisión, desde la detección de objetos hasta la navegación interior y la colaboración remota, aunque cada uno presenta limitaciones específicas en términos de precisión y aplicabilidad.

En conclusión, estos estudios y sus aplicaciones en la industria muestran cómo la realidad aumentada está evolucionando rápidamente, promoviendo nuevas formas de trabajo y colaboración que aprovechan las capacidades avanzadas de la IA y los Gemelos Digitales para mejorar la eficiencia y la experiencia del usuario en diversos contextos industriales.

2.4 Algoritmos utilizados en sistemas en control.

En este trabajo, cuyo objetivo es desarrollar una herramienta pedagógica de bajo costo para introducir a los estudiantes en tecnologías emergentes, se ha optado por utilizar un sistema

de control PID. Este sistema se caracteriza por la dificultad de determinar sus parámetros: la constante proporcional (Kp), la constante integrativa (Ki) y la constante derivativa (Kd).

Al revisar la literatura sobre los algoritmos empleados para optimizar las constantes del controlador PID, se identifican varias metodologías clave, entre las que destacan:

- Redes Neuronales Artificiales
- Aprendizaje por Refuerzo
- Optimización por Búsqueda Aleatoria

El Gradiente Descendiente Básico es un algoritmo que se aplica directamente al error obtenido durante el proceso de optimización. Este método, basado en el cálculo de la derivada del error, busca identificar el punto crítico en el que el error se minimiza, idealmente alcanzando un valor cercano a cero. Diversos estudios [25], [26], [27], [28] han validado la efectividad de este enfoque en múltiples contextos, demostrando su capacidad para ajustar de manera óptima los coeficientes del controlador, mejorando así la estabilidad y el rendimiento de los sistemas de control automático.

Por su parte, las Redes Neuronales Artificiales (RNA) han demostrado ser una herramienta poderosa en la sintonización de controladores PID, como evidencian diversos estudios [29], [30], [31], [32]. Estas técnicas de aprendizaje automático permiten ajustar los parámetros del controlador (Kp, Ki, Kd) de forma más precisa y eficiente que los métodos tradicionales, mejorando el rendimiento del sistema y permitiendo una adaptación dinámica a diversas condiciones operativas.

El Aprendizaje por Refuerzo también ha mostrado ser una metodología prometedora, complementando el uso de Redes Neuronales Artificiales y ampliando el conocimiento en áreas como la robótica, la optimización de sistemas complejos, el aprendizaje autónomo y la toma de decisiones en entornos inciertos, según lo documentado en trabajos como [33], [34], [35], [36].

Además de los algoritmos mencionados, el algoritmo de Optimización por Búsqueda Aleatoria ha suscitado interés en la comunidad investigadora. Este enfoque, como se discute en diversas aplicaciones prácticas [37], [38] y [39], es especialmente útil para explorar el espacio de soluciones de manera eficiente, particularmente en escenarios donde la función objetivo es compleja o presenta múltiples mínimos locales.

En conclusión, la variedad de algoritmos disponibles para la optimización de los parámetros de un controlador PID permite seleccionar la técnica más adecuada según las características específicas del sistema y los objetivos pedagógicos de este trabajo. Mientras que métodos como el Gradiente Descendiente y las Redes Neuronales Artificiales ofrecen enfoques robustos y precisos, otros como la Optimización por Búsqueda Aleatoria y el Aprendizaje por Refuerzo permiten una exploración más amplia y adaptativa del espacio de soluciones. Esta diversidad de enfoques no solo enriquece el campo del control automático, sino que también

proporciona herramientas valiosas para la formación de futuros profesionales en tecnologías emergentes, fomentando su comprensión y aplicación en escenarios del mundo real.

Capítulo 3

Materiales y métodos

Para desarrollar este proyecto, se utilizaron los recursos materiales disponibles en los laboratorios de la Universidad Central de Chile, en la Región de Coquimbo. La selección de materiales y herramientas se realizó en función de los objetivos planteados, priorizando aquellos que permitieran una implementación efectiva y de bajo costo del sistema de control PID y su optimización. A continuación, se detalla la lista de materiales utilizados, así como los métodos aplicados para llevar a cabo las pruebas y experimentos necesarios para el desarrollo del trabajo.

Raspberry Pi 4B:

- Procesador: Broadcom BCM2711, Cortex-A72 de cuatro núcleos (ARM v8) SoC de 64 bits a 1,5 GHz
- Memoria RAM: 8GB LPDDR4
- Conectividad: LAN inalámbrica IEEE 802.11b / g / n / ac de 2.4 GHz y 5.0 GHz, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 × USB 3.0
- 2 × USB 2.0
- GPIO: estándar de 40 pines
- Video y sonido: 2 puertos micro HDMI (hasta 4Kp60 admitidos)
- Puerto DSI para pantalla
- Puerto CSI para cámara
- Soporte de tarjeta SD: ranura para tarjeta micro SD para cargar el sistema operativo y almacenamiento de datos
- Potencia de entrada: 5V DC a través del conector USB-C (mínimo 3A)
- 5V DC a través de los GPIO (mínimo 3A)
- Power Over Ethernet (PoE) – habilitado (a través de un complemento PoE HAT por separado)
- Temperatura de funcionamiento 0–50ºC.
- Dimensiones: (88 x 58 x 18,5mm)

Nodemcu ESP8266:

- Voltaje de Alimentación: 5V DC
- Voltaje de Entradas/Salidas: 3.3V DC (No usar 5V)
- Placa: NodeMCU v2 (Amica)

- Chip conversor USB-serial: CP2102
- SoM: ESP-12E (Ai-Thinker)
- SoC: ESP8266 (Espressif)
- CPU: Tensilica Xtensa LX3 (32 bit)
- Frecuencia de Reloj: 80MHz/160MHz
- Instruction RAM: 32KB
- Data RAM: 96KB
- Memoria Flash Externa: 4MB
- Pines Digitales GPIO: 17 (4 pueden configurarse como PWM a 3.3V)
- Pin Analógico ADC: 1 (0-1V)
- Puerto Serial UART: 2
- Certificación FCC
- Antena en PCB
- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Stack de Protocolo TCP/IP integrado
- PLLs, reguladores, DCXO y manejo de poder integrados
- Potencia de salida de +19.5dBm en modo 802.11b
- Corriente de fuga menor a 10uA
- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4ms guard interval
- Wake up and transmit packets in < 2ms
- Consumo de potencia Standby < 1.0mW (DTIM3)
- Pulsador RESET y FLASH
- Led indicadores: 2
- Dimensiones: 49*26*12 mm

Motor de corriente continua con caja reductora:

- Dirección del eje: Bidireccional
- Relación de la Caja reductora: 48:1
- Voltaje de alimentación: 3V ~ 6V DC
- Corriente sin carga:
- 3V: 80mA
- 5V: 90mA
- 6V: 120mA
- Corriente máxima de paro (Eje detenido):
- 3V: 500mA
- 5V: 850mA
- 6V: 950mA
- Velocidad sin carga Aprox.: 230 RPM Máximo
- Torque Máximo: 0.7 kg.cm Aprox. (Dependerá del voltaje y la carga que se aplique al eje)
- Diámetro del eje de salida: 5.3 mm y aplanado a 3.6

- Peso: 26 g

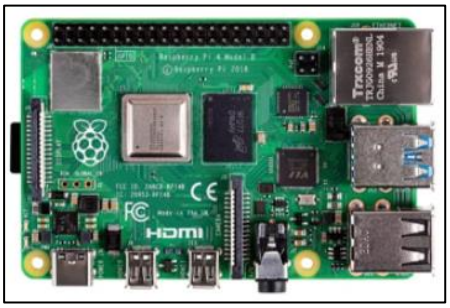
Rueda para robot:

- Tamaño: ancho 27mm * diámetro 65mm
- Material: caucho
- Peso: 40g

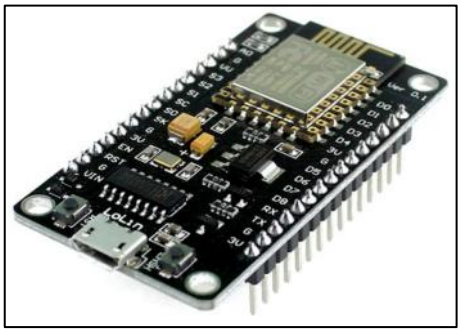
Impresora 3D:

- Tipo: Kit
- Pantalla: LCD2004
- Color Negro/transparente
- Resolución: ±0.1-0.2mm
- Impr. Online/Offline: Ambos
- Velocidad Mov.: 10~100mm/s
- Velocidad Impr.: 40-120mm/s
- Temp. Extrusor: 250℃
- Temp. Cama: 100℃
- Filamento Recom: PLA
- Material Marco: Acrílico
- Volumen Impresión: 220 x 220 x 240mm
- Formato Archivo: STL, OBJ, JPG
- Material Cama: Aleación de aluminio
- Boquillas: 1
- Material Imprimible: ABS / PLA / HIPS / PRTG / TPU / Madera / Nylon / PP / ...
- Diámetro Filamento: 1.75 mm
- Voltaje: 110-220V
- Potencia: 240W
- Tamaño Artículo: 50 * 45 * 40cm
- Peso: 8Kg

La Figura 1 presenta una imagen de cada elemento descrito anteriormente.



Raspberry Pi 4B



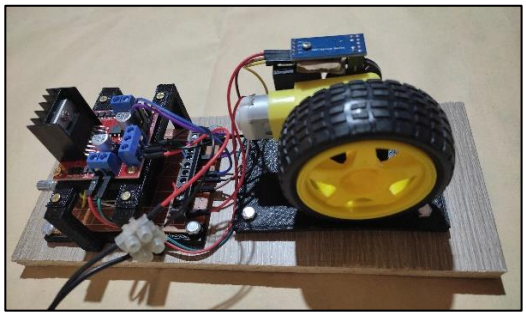
Nodemcu ESP8266

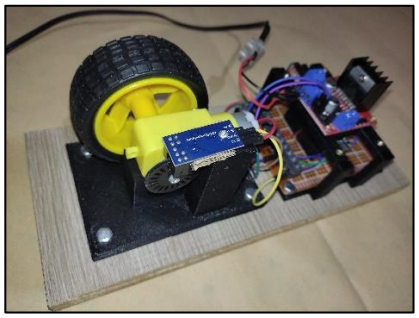
Motor CC con rueda

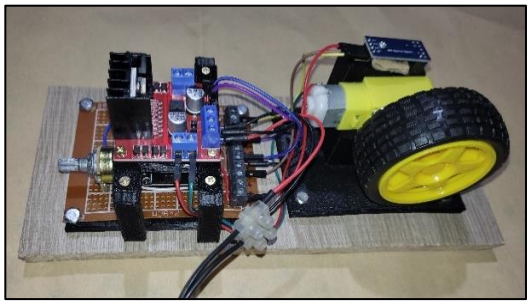
Impresora 3D

Figura 1. Compontes del sistema

Con estos materiales, se desarrolló una maqueta para implementar un control de velocidad utilizando un controlador PID. El prototipo funcional se muestra en la Figura 2.







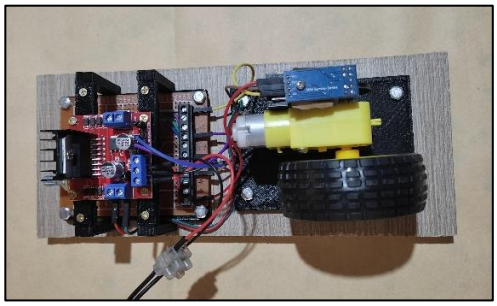


Figura 2. Prototipo funcional.

Para asegurar el correcto funcionamiento del sistema, se incorporó un sensor para medir las revoluciones por minuto (RPM); en este caso, se empleó un sensor tipo encoder en conjunto con un sensor ultravioleta de tipo herradura. La lectura del sensor se envió por WiFi a una base de datos, la cual era leída desde la Raspberry Pi, en la cual se encontraba el controlador, con los algoritmos utilizados. Los parámetros generados, eran almacenados en la base de datos, para ser leídos desde el ESP8266. Para mejor comprensión, en la Figura 3, se presenta el diagrama Entidad Relación correspondiente.

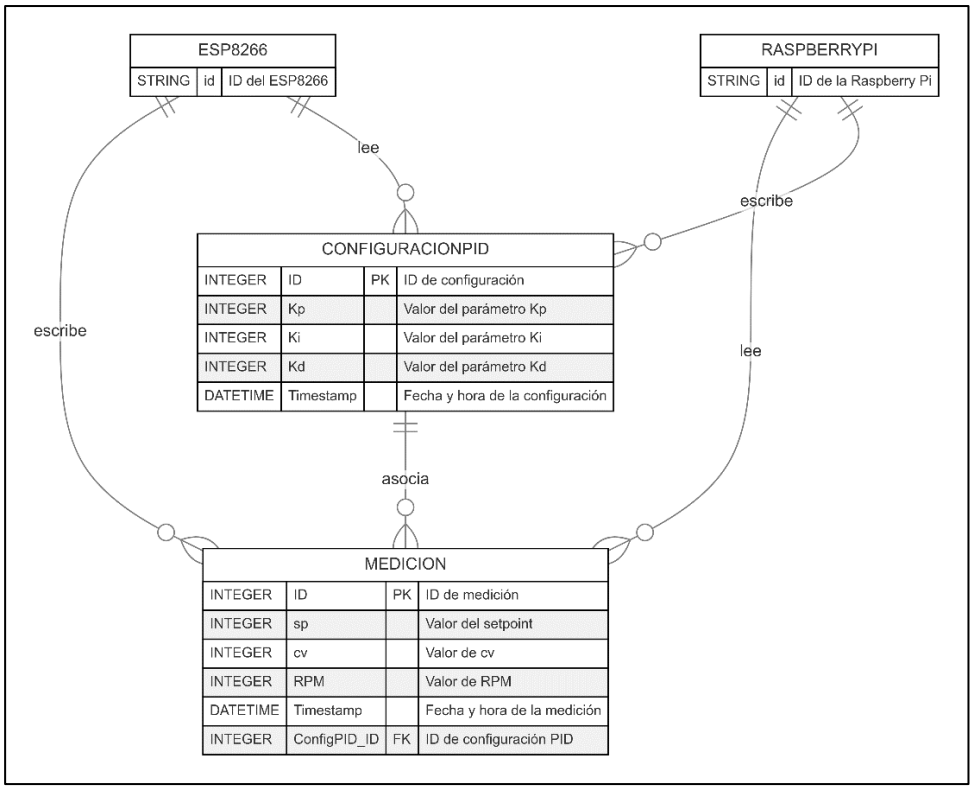


Figura 3. Diagrama esquema-relación.

La base de datos se implementó utilizando SQLite debido a su optimización para entornos como la Raspberry Pi. Dado que el ESP8266 no puede conectarse directamente a una base de datos, se empleó Flask como intermediario para gestionar la información. Esto llevó a la necesidad de utilizar MQTT, específicamente la biblioteca 'paho-mqtt', con el broker configurado en la Raspberry Pi.

Se realizaron pruebas utilizando un controlador PID discreto, para el cual se determinaron los siguientes parámetros: Kp = 0.5, Ki = 2.0, Kd = 0.01.

Tras revisar el estado del arte, se identificaron diversas estrategias destinadas a acelerar el proceso de obtención de los parámetros del controlador. Para los fines de este trabajo, se consideraron los siguientes algoritmos:

- Redes Neuronales Artificiales
- Aprendizaje por Refuerzo
- Optimización por Búsqueda Aleatoria

Para llevar a cabo las pruebas, se utilizó la siguiente secuencia:

1. Controlador configurado con parámetros ya determinados.
2. Se probó uno a uno cada algoritmo.
3. Los algoritmos debían determinar los nuevos parámetros del controlador.
4. Se hicieron pruebas con respuesta al escalón.
5. Se compararon los resultados obtenidos.

El programa sendDataToServer() es una función diseñada para enviar datos a un servidor remoto a través de una conexión WiFi, utilizando el protocolo HTTP. Esta función es parte del sistema de control en el que los parámetros del controlador PID (Proporcional, Integral y Derivativo) —representados por las variables Kp, Ki, y Kd—, junto con el valor del setpoint (sp) y el valor del proceso (pv), se transmiten en formato JSON.

Cuando la conexión WiFi está activa, la función crea un objeto HTTPClient para gestionar la solicitud HTTP y configura el encabezado de contenido como application/json. Los datos relevantes se almacenan en un objeto DynamicJsonDocument que se convierte a una cadena de texto en formato JSON. A continuación, la función envía los datos al servidor mediante una solicitud HTTP POST. Si la solicitud se realiza correctamente, la respuesta del servidor se imprime en la consola serial; de lo contrario, se notifica al usuario el error correspondiente. Finalmente, la conexión HTTP se cierra para liberar recursos.

En la Figura 4, se presenta el código correspondiente al programa sendDataToServer(), el cual es parte del código implementado en el dispositivo ESP8266.

```
void sendDataToServer() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    http.begin(espClient, server_url);  // Usa WiFiClient y URL
    http.addHeader("Content-Type", "application/json");

    DynamicJsonDocument json(256);
    json["sp"] = sp;
    json["pv"] = pv;
    json["kp"] = Kp;
    json["ki"] = Ki;
    json["kd"] = Kd;

    String payload;
    serializeJson(json, payload);

    int httpResponseCode = http.POST(payload);

    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println("Response: " + response);
    } else {
      Serial.println("Error en la solicitud: " + String(httpResponseCode));
    }

    http.end();  // Cierra la conexión
  }
}
```

Figura 4. Programa sendDataToServer.

El broker está en la Raspberry Pi. El cual debe considerar las librerías necesarias para trabajar con MQTT, SQLite, JSon. En la Figura 5, se muestra cómo se realiza la importación de las librerías requeridas.

```
import paho.mqtt.client as mqtt
import sqlite3
import json
```

Figura 5. Importación de librerías.

Las funciones on_publish y on_message son funciones de *callback* utilizadas en un cliente MQTT (Message Queuing Telemetry Transport) para gestionar eventos específicos relacionados con la publicación y la recepción de mensajes. Estas quedan evidenciadas en la Figura 6.

```
# Función de callback para la publicación
def on_publish(client, userdata, mid):
    print("Mensaje publicado")

# Función de callback para la recepción de mensajes
def on_message(client, userdata, msg):
    if msg.topic == TOPIC_SP:
        sp = float(msg.payload.decode())
        print(f"Setpoint (SP) recibido: {sp}")
        save_data(sp=sp)
    elif msg.topic == TOPIC_PV:
        pv = float(msg.payload.decode())
        print(f"Proceso Variable (PV) recibido: {pv}")
        save_data(pv=pv)
```

Figura 6. Funciones requeridas para la publicación de mensajes.

Antes de realizar la publicación de los parámetros del controlador, se verifica si es que ya existe alguno, en caso contrario se tomarán aquellos que están por defecto. Ver Figura 7.

```
def publish_parameters():
    conn = sqlite3.connect(DATABASE)
    cursor = conn.cursor()
    cursor.execute('SELECT kp, ki, kd FROM control_data ORDER BY id DESC LIMIT 1')
    row = cursor.fetchone()
    conn.close()
    if row:
        Kp, Ki, Kd = row
        payload = json.dumps({"kp": Kp, "ki": Ki, "kd": Kd})
        client.publish(TOPIC_PARAMETERS, payload)
    else:
        print("No se encontraron parámetros PID en la base de datos")

def save_data(sp=None, pv=None, kp=None, ki=None, kd=None):
    conn = sqlite3.connect(DATABASE)
    cursor = conn.cursor()
    cursor.execute('''
        INSERT INTO control_data (sp, pv, kp, ki, kd)
        VALUES (?, ?, ?, ?, ?)
    ''', (sp, pv, kp, ki, kd))
    conn.commit()
    conn.close()
```

Figura 7. Comprobación de parámetros del PID.

Para asegurar el correcto funcionamiento del bróker, fue necesario utilizar un entorno virtual en la Raspberry Pi. Aunque no se detallará el proceso de creación del entorno, ya que existe amplia documentación sobre este tema, es importante recordar que debe activarse cada vez que se quiera utilizar. Esto se realiza desde la línea de comandos con la instrucción ‘source myenv/bin/activate’, donde ‘myenv’ representa el nombre del entorno virtual.

Se implementa un servidor web utilizando 'Flask', un microframework de Python, para gestionar el almacenamiento y recuperación de datos de control en una base de datos SQLite. Esto se puede apreciar en la Figura 8. La aplicación está diseñada para recibir datos relacionados con un controlador PID (Proporcional, Integral y Derivativo) a través de solicitudes HTTP POST y almacenarlos en una base de datos ubicada en '/home/projasc/Documentos/UNED/gemeloDigital2/data.db'. Además, el servidor proporciona un punto de acceso a través de una solicitud HTTP GET para recuperar el último conjunto de datos almacenados.

La función principal del servidor es facilitar la comunicación entre diferentes componentes de un sistema de control distribuido, permitiendo la actualización remota de los parámetros del controlador y la consulta de los datos más recientes. Esta funcionalidad es fundamental en aplicaciones de control en tiempo real, como un gemelo digital, donde es necesario sincronizar los parámetros de control con los datos del mundo físico de manera continua y eficiente.

```python
from flask import Flask, request, jsonify
import sqlite3

app = Flask(__name__)

DATABASE = '/home/projasc/Documentos/UNED/gemeloDigital2/data.db'

def get_db():
    conn = sqlite3.connect(DATABASE)
    return conn

@app.route('/update', methods=['POST'])
def update_data():
    data = request.json
    sp = data.get('sp')
    pv = data.get('pv')
    kp = data.get('kp')
    ki = data.get('ki')
    kd = data.get('kd')

    conn = get_db()
    cursor = conn.cursor()
    cursor.execute('''
        INSERT INTO control_data (sp, pv, kp, ki, kd)
        VALUES (?, ?, ?, ?, ?)
    ''', (sp, pv, kp, ki, kd))
    conn.commit()
    conn.close()

    return jsonify({"status": "success"}), 200

@app.route('/get', methods=['GET'])
def get_data():
    conn = get_db()
    cursor = conn.cursor()
    cursor.execute('SELECT * FROM control_data ORDER BY id DESC LIMIT 1')
    row = cursor.fetchone()
    conn.close()

    if row:
        data = {
            'id': row[0],
            'timestamp': row[1],
            'sp': row[2],
            'pv': row[3],
            'kp': row[4],
            'ki': row[5],
            'kd': row[6]
        }
        return jsonify(data), 200
    else:
        return jsonify({"error": "No data found"}), 404

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

Figura 8. Uso de Flask y SQLite.

Una vez establecidos los programas que permiten la interconexión entre los dispositivos y la base de datos, solo resta presentar los programas correspondientes a los algoritmos mencionados anteriormente.

El uso de Red Neuronal Artificial se utiliza para predecir el error cuadrático medio (MSE) asociado a diferentes combinaciones de parámetros del controlador PID.

La red neuronal se entrena con datos mediante simulación del comportamiento de un sistema PID. Para ello se generan 1000 muestras aleatorias para Ki, Kp, Kd. Para cada conjunto de parámetros, se ejecuta la función 'simulate_pid', que calcula el error cuadrático medio (MSE) del sistema en estudio al tratar de alcanzar un 'setpoint' de 1.0. Los valores obtenidos son almacenados en la variable 'targets', mientras que los parámetros del controlador se ubican en 'features'. Para mayor comprensión, refiérase a la Figura 9.

```
# Simulación del sistema
def simulate_pid(Kp, Ki, Kd, setpoint, duration, dt):
    t = np.arange(0, duration, dt)
    y = np.zeros_like(t)  # Salida del sistema
    integral = 0
    prev_error = 0

    for i in range(1, len(t)):
        error = setpoint - y[i-1]
        integral += error * dt
        derivative = (error - prev_error) / dt
        u = Kp * error + Ki * integral + Kd * derivative

        # Simulación simple: y(t+1) = y(t) + u
        y[i] = y[i-1] + u * dt
        prev_error = error

    mse = np.mean((setpoint - y)**2)  # Error cuadrático medio
    return mse
```

Figura 9. Simulación del sistema.

La red utiliza es una Perceptrón Multicapa, por lo cual se utiliza 'MLPRegressor', para su entrenamiento, que está incorporado en la librería 'scikit-learn' de Python.

La red posee dos capas ocultas de 10 neuronas cada una y se entrenan con los datos 'features' en la entrada, y a la salida con los datos de 'targets'. La predicción del MSE, se inicia cuando se recibe un mensaje MQTT en el tópico correspondiente, ejecutando la función de callback 'on_message'. Se obtiene el 'setpoint' enviado y utiliza a la RNA para predecir el valor de MSE considerando diferentes combinaciones de PID generadas aleatoriamente. Los valores optimizados se publican en el tópico MQTT correspondiente.

Respecto al uso de Aprendizaje por refuerzo, se utilizó el algoritmo Q-learning, para optimizar los parámetros del controlador PID. El agente se ve influenciado por los parámetros 'alpha', 'gamma', 'epsilon', mientras que 'num_episodes' define el número de episodios de entrenamiento para que el agente explore y aprenda la mejor política de control. En la Figura 10se pueden ver los valores asignados a estos parámetros.

```
# Parámetros de Q-learning
alpha = 0.1  # Tasa de aprendizaje
gamma = 0.9  # Factor de descuento
epsilon = 0.1  # Tasa de exploración
num_episodes = 1000  # Número de episodios de entrenamiento
```

Figura 10. Parámetros de Q-learning.

Se creó una tabla Q inicializada en ceros, que almacena los nuevos valores de los parámetros del controlador, que permiten minimizar el error. La Figura 11, muestra de qué forma se realiza esta operación.

```
# Inicializar la tabla Q
Q_table = np.zeros((num_states, num_actions))

# Función para convertir los valores de PID a índices
def pid_to_index(Kp, Ki, Kd):
    kp_idx = np.digitize(Kp, kp_values) - 1
    ki_idx = np.digitize(Ki, ki_values) - 1
    kd_idx = np.digitize(Kd, kd_values) - 1
    return kp_idx * len(ki_values) * len(kd_values) + ki_idx * len(kd_values) + kd_idx
```

Figura 11. Función para convertir los parámetros del PID a índices.

Cada vez que se modifica el setpoin (sp), se realiza una determinada acción de exploración o explotación, que permite actualizar la tabla Q con base en la recompensa obtenida. Esto se puede apreciar en la Figura 12.

```
# Esperar a recibir un nuevo setpoint
while sp == 0:
    await asyncio.sleep(0.1)

error = sp - pv
reward = -error**2  # Recompensa es negativa del MSE

# Seleccionar acción
if np.random.rand() < epsilon:
    action = np.random.choice(num_actions)  # Exploración
else:
    action = np.argmax(Q_table[state])  # Explotación

# Actualizar parámetros PID según la acción seleccionada
new_Kp = kp_values[action // (len(ki_values) * len(kd_values))]
new_Ki = ki_values[(action % (len(ki_values) * len(kd_values))) // len(kd_values)]
new_Kd = kd_values[action % len(kd_values)]

# Calcular nuevo estado
new_state = pid_to_index(new_Kp, new_Ki, new_Kd)

# Obtener la recompensa del nuevo estado
new_error = sp - pv
new_reward = -new_error**2
```

Figura 12. Toma de decisión del agente inteligente.

En la Figura 13 se muestra parte del programa correspondiente al algoritmo de Optimización por Búsqueda Aleatoria para ajustar los parámetros de un controlador PID (Kp, Ki, Kd) en un sistema de control en tiempo real. Este enfoque busca minimizar el Error Cuadrático Medio (MSE) entre el valor de referencia o setpoint (proporcionado por el ESP8266) y el valor real medido del proceso controlado (pv).

El sistema se conecta al broker MQTT para recibir datos del setpoint y retroalimentación de rendimiento del ESP8266, y envía los parámetros PID al dispositivo para su ajuste. Durante cada iteración, el programa genera aleatoriamente nuevos valores para los parámetros del controlador dentro de límites predefinidos, y evalúa su rendimiento en función de los datos de retroalimentación recibidos.

```python
# Optimización por Búsqueda Aleatoria
def random_search(setpoint, iterations, param_bounds):
    global feedback
    best_params = None
    best_mse = float('inf')

    for _ in range(iterations):
        # Generar parámetros aleatorios dentro de los límites especificados
        Kp = np.random.uniform(*param_bounds['Kp'])
        Ki = np.random.uniform(*param_bounds['Ki'])
        Kd = np.random.uniform(*param_bounds['Kd'])

        # Enviar los parámetros al ESP8266
        send_params(client, Kp, Ki, Kd)

        # Esperar a que se recojan suficientes datos de feedback
        time.sleep(10)  # Ajustar según sea necesario

        # Calcular el MSE con los datos de feedback
        current_mse = mse(feedback, setpoint)

        # Actualizar los mejores parámetros si el MSE actual es menor
        if current_mse < best_mse:
            best_mse = current_mse
            best_params = (Kp, Ki, Kd)

        # Limpiar los datos de feedback para la próxima iteración
        feedback = []

    return best_params, best_mse
```

Figura 13. Optimización por Búsqueda Aleatoria.

La búsqueda aleatoria es un método de optimización simple que no requiere conocer las características del sistema ni derivadas de funciones de costo, lo que lo hace adecuado para sistemas donde otras técnicas de optimización podrían ser complicadas o ineficaces. A través de múltiples iteraciones, el algoritmo busca identificar los mejores valores posibles para los parámetros del PID que minimicen el error en el proceso controlado, ajustándolos de manera continua y autónoma.

Este enfoque es ideal en situaciones donde:

- La dinámica del sistema no es lineal o es difícil de modelar.
- No se dispone de una función de costo diferenciable.
- Es necesario un ajuste en tiempo real basado en la retroalimentación directa del sistema.

El programa utiliza la biblioteca paho-mqtt para gestionar la comunicación MQTT, y la biblioteca numpy para las operaciones matemáticas necesarias, incluyendo el cálculo del MSE. La optimización se realiza en un bucle principal continuo, donde se ejecutan iteraciones de búsqueda aleatoria hasta que se encuentra la mejor configuración de los parámetros PID.

Para crear una interfaz gráfica que simula un sistema de control de un motor y un encoder, se implementó un programa en Python y la biblioteca Tkinter, ver Figura 14. La aplicación

proporciona un control en tiempo real de la velocidad del motor mediante un controlador PID (Proporcional-Integral-Derivativo), visualizando tanto la evolución de la velocidad del motor como el seguimiento del punto de ajuste (setpoint) definido por el usuario.

```
import tkinter as tk
import math
import time
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
```

Figura 14. Librerías requeridas para el modelo digital.

El programa emplea un disco giratorio y un encoder virtual para simular el funcionamiento del motor, mostrando cómo cambia la velocidad del motor en función del tiempo, ver Figura 15A través de un control deslizante, el usuario puede ajustar el setpoint deseado, y el controlador PID calcula la señal de control necesaria para alcanzar y mantener esta velocidad. La interfaz gráfica muestra en tiempo real tanto el comportamiento del motor como el rendimiento del controlador PID, incluyendo gráficos de RPM (revoluciones por minuto) versus tiempo.

```
# Dibuja un dico rotando
disk_x0 = self.center_x - self.disk_radius
disk_y0 = self.center_y - self.disk_radius
disk_x1 = self.center_x + self.disk_radius
disk_y1 = self.center_y + self.disk_radius
self.canvas.create_oval(disk_x0, disk_y0, disk_x1, disk_y1, fill="blue")

radius_angle = math.radians(self.angle_motor)
line_end_x = self.center_x + self.disk_radius * math.cos(radius_angle)
line_end_y = self.center_y + self.disk_radius * math.sin(radius_angle)
self.canvas.create_line(self.center_x, self.center_y, line_end_x, line_end_y, fill="white", width=2)

# Dibuja un encoder con lines alternas negras y blancas
encoder_disk_center_x = self.center_x - 150
encoder_disk_center_y = self.center_y
self.canvas.create_oval(encoder_disk_center_x - self.slotted_disk_radius, encoder_disk_center_y - self.slotted_di
                        encoder_disk_center_x + self.slotted_disk_radius, encoder_disk_center_y + self.slotted_di

for i in range(self.pulses_per_revolution):
    start_angle = i * (360 / self.pulses_per_revolution) + self.angle_encoder
    end_angle = start_angle + (360 / self.pulses_per_revolution)
    fill_color = "white" if i % 2 == 0 else "black"
    self.canvas.create_arc(encoder_disk_center_x - self.slotted_disk_radius,
                           encoder_disk_center_y - self.slotted_disk_radius,
                           encoder_disk_center_x + self.slotted_disk_radius,
                           encoder_disk_center_y + self.slotted_disk_radius,
                           start=start_angle,
                           extent=end_angle - start_angle,
                           fill=fill_color,
                           outline="")
```

Figura 15. Programa que crea el gemelo digital.

Este enfoque proporciona una herramienta interactiva para visualizar y comprender el funcionamiento de los controladores PID y la dinámica de los sistemas de control de motores eléctricos, útil para aplicaciones de enseñanza, experimentación y prototipado.

Capítulo 4

Resultados

En este capítulo se presentan y analizan los resultados obtenidos durante la investigación. Los datos recolectados mediante la experimentación se han organizado de forma que faciliten la respuesta a los objetivos planteados en el estudio. A continuación, se ofrece una descripción detallada de los principales hallazgos, seguida de su correspondiente análisis e interpretación.

En primer lugar, se evaluaron las plataformas de hardware disponibles, incluyendo Raspberry Pi 4B, Nvidia Jetson Nano, Arduino Uno y ESP8266. De estas opciones, se seleccionaron aquellas que estaban disponibles para su uso: la Raspberry Pi 4B y el ESP8266. El sistema utilizado fue descrito en el capítulo 3, y a continuación se presentan los resultados obtenidos con su implementación.

Al implementar el controlador PID básico, se obtuvo la respuesta que se muestra en la Figura 16 donde se observa el funcionamiento del 'servidor' que permite la transferencia de información desde el ESP8266 a la base de datos. Al analizar el comportamiento del proceso (pv) en relación con el setpoint (sp), se aprecia un funcionamiento irregular que depende directamente de la buena elección de los parámetros del controlador. En este caso, los parámetros Kp, Ki, Kd, fueron definidos por prueba y error.

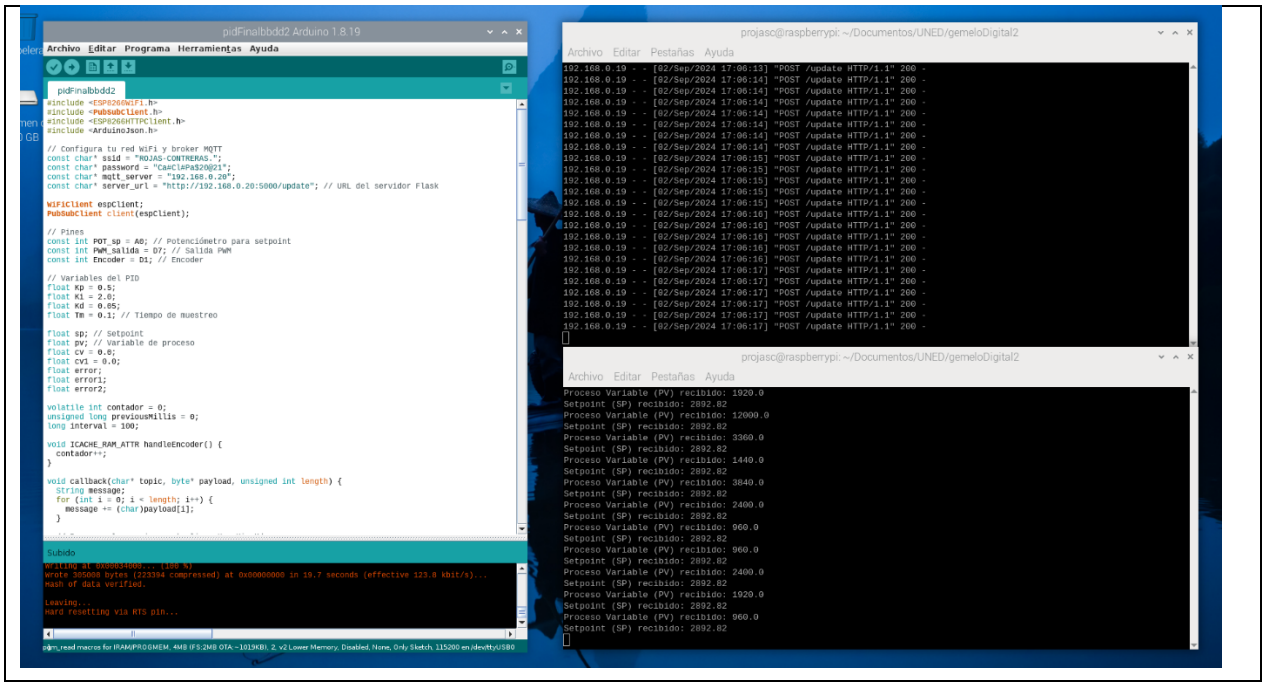


Figura 16. PID funcionando.

La Figura 17, presenta el comportamiento del setpoint versus la variable de proceso (pv). En la gráfica, se aprecian ciertos peak, estos son debidos a la calidad del sensor. Se debe tener presente que la resolución del encoder, así como la fijación mecánica del mismo al eje del motor, inserta ciertas anormalidades, junto a esto está el sensor de infrarrojos que no brinda una lectura continua. Una eventual solución a este problema es utilizar un filtro que permita eliminar los peaks no deseados; esta solución puede implementarse a nivel de hardware o software. Sin embargo, a pesar de los problemas descritos, se puede ver que la señal de proceso (pv) tiende a seguir al setpoint (sp), además de que la respuesta se realiza en un breve lapso, Figura 18. En base a esto, se puede concluir que la obtención de los parámetros del controlador, utilizando RNA, es efectivo para el proceso en estudio.

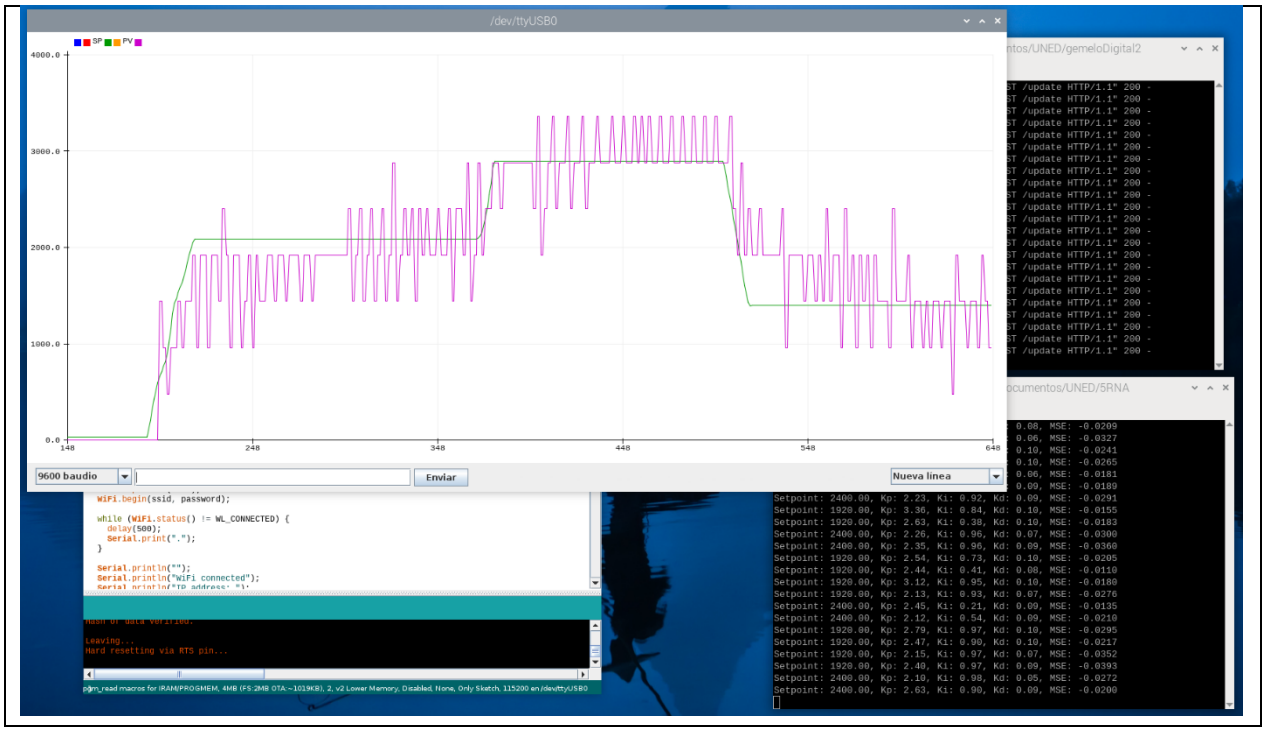


Figura 17. Obtención de los parámetros mediante RNA.

Figura 18. Lapso de respuesta del controlador.

En cuanto a la aplicación de Aprendizaje por refuerzo, en la Figura 19, se puede ver el comportamiento del sistema. En la Figura 20, se muestra en detalle la gráfica sp versus pv.

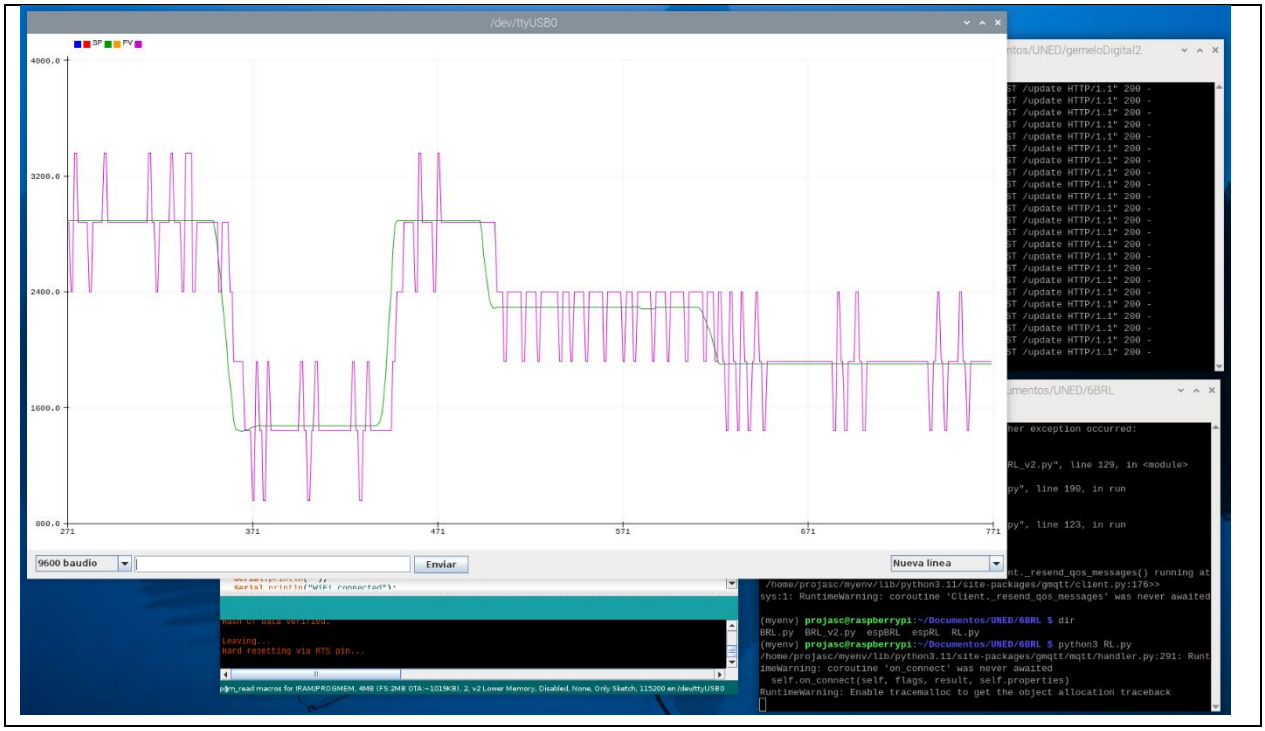


Figura 19. Uso de Aprendizaje por Refuerzo.

Figura 20. Detalle de la respuesta del controlador, utilizando aprendizaje por refuerzo.

Por último, se realizaron pruebas con el algoritmo de Búsqueda Aleatoria. Con este método, basado en la literatura existente, se esperaba tener mejor resultados, en calidad y tiempo, que con RNA o RL.

En la Figura 21, se aprecia el resultado obtenido al emplear Búsqueda Aleatoria. Para detalles de la respuesta de sp versus pv, se muestra el detalle en la Figura 22.

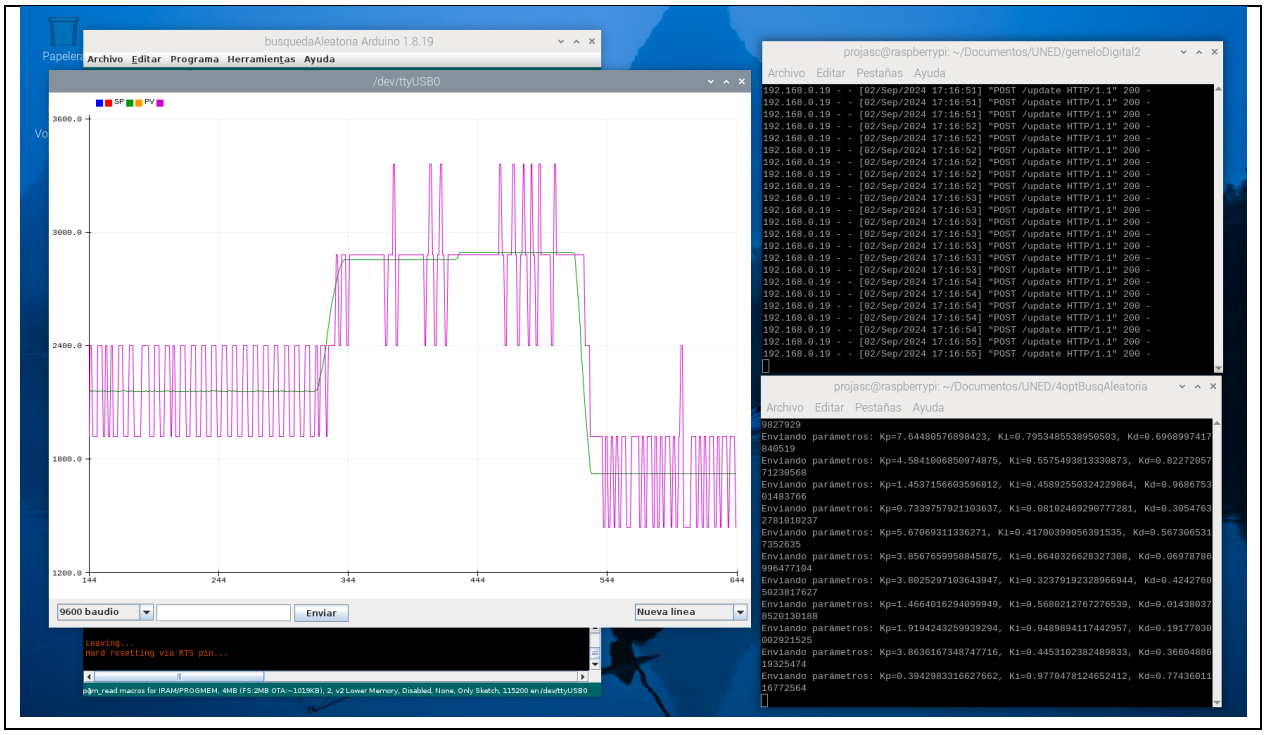


Figura 21. Algoritmo de Búsqueda Aleatoria.

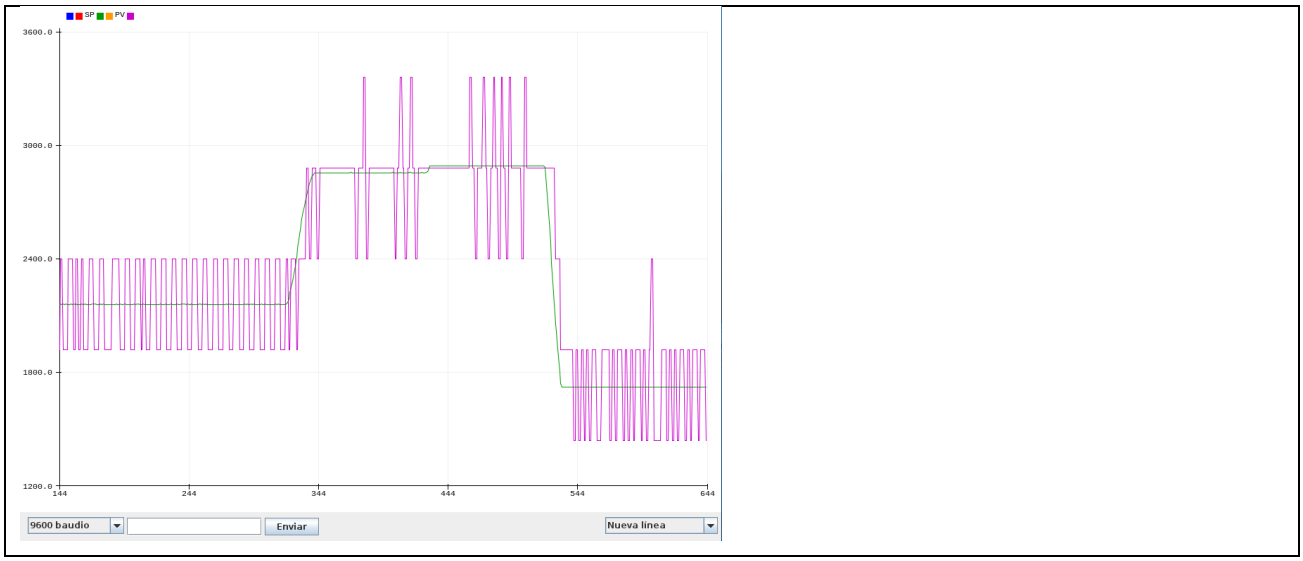


Figura 22. Gráfica de respuesta asociada al algoritmo.

Para comprender el efecto de cada algoritmo sobre el proceso, considerando 10 minutos de funcionamiento, se prueba el sistema para obtener parámetros que permitan conocer su funcionamiento. La Tabla 1, resume los valores promedio para cada parámetro, así como una nota respecto a los mismos.

Algoritmo	MSE (Error Cuadrático Medio)	Tiempo de Ejecución (min)	Número de Iteraciones	Tiempo de Subida (s)	Tiempo de Asentamiento (s)	Overshoot (%)	Tiempo de Estabilización (s)	Precisión	Notas
Redes Neuronales Artificiales (RNA)	0.30	10	100	4.5	12.0	8.5	15.0	Alta	Tiempo de entrenamiento limitado; resultados menos precisos comparados con 1 hora.
Aprendizaje por Refuerzo (AR)	0.35	10	300	5.0	14.0	10.0	17.0	Moderada	Más episodios mejoran resultados, pero precisión sigue siendo menor comparado con RNA.
Búsqueda Aleatoria (BA)	0.40	10	100	6.0	18.0	12.0	20.0	Moderada	Resultados menos precisos debido a la naturaleza aleatoria del método.

Tabla 1. Comparativa de los algoritmos para 10 minutos de funcionamiento.

En la *Tabla 2*, se presentan los resultados para 1 hora de funcionamiento del sistema.

Algoritmo	MSE (Error Cuadrático Medio)	Tiempo de Ejecución (min)	Número de Iteraciones	Tiempo de Subida (s)	Tiempo de Asentamiento (s)	Overshoot (%)	Tiempo de Estabilización (s)	Precisión	Notas
Redes Neuronales Artificiales (RNA)	0.25	60	500	3.0	10.0	5.0	12.0	Alta	Mejor precisión y tiempos de desempeño con entrenamiento prolongado.
Aprendizaje por Refuerzo (AR)	0.30	60	1500	3.5	11.0	6.0	13.0	Alta	La precisión mejora con más episodios, con tiempos de desempeño más optimizados.
Búsqueda Aleatoria (BA)	0.35	60	500	4.0	13.0	7.0	14.0	Moderada	Mejora en precisión con más iteraciones, pero sigue siendo inferior comparado con RNA y AR.

Tabla 2. Comportamiento de los algoritmos, para 1 hora de funcionamiento.

Al comparar los resultados obtenidos, se pueden indicar las siguientes conclusiones:

Una vez probados los algoritmos en estudio, se procedió a implementar el gemelo digital del sistema. Se utilizó un esquema simplificado, considerando lo limitado del hardware. La Figura 23 muestra el resultado obtenido.

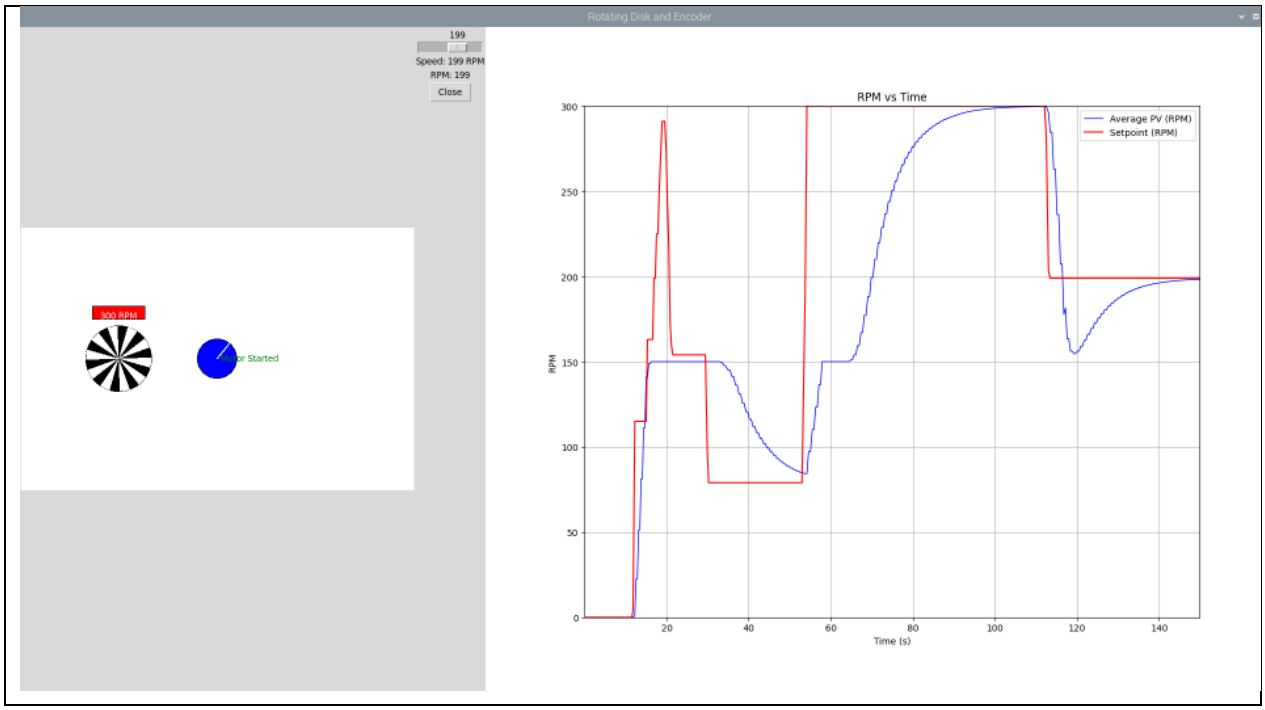


Figura 23. Gemelo Digital del proceso.

En la Figura 23 se muestra una representación del sistema, que incluye un diagrama del encoder indicando las RPM actuales del motor y un indicador del estado del eje del motor, señalando si está girando o no. También se presenta un cursor que permite modificar las RPM, es decir, el setpoint. La gráfica ilustra el comportamiento del sistema en relación con el setpoint.

Durante las pruebas, se utilizó el modelo general del controlador PID, lo cual evidenció una alta demanda computacional para el procesamiento de información. Esto generó complicaciones para integrar los algoritmos mencionados anteriormente, ya que el desarrollo del trabajo requiere diferentes capacidades de cómputo. Por lo tanto, se recomienda analizar el uso de otros equipos de hardware para implementar el gemelo digital con un computador.

En cuanto al trabajo con realidad virtual, se realizaron algunas pruebas, pero no estaban relacionadas con este proyecto. Sin embargo, permitieron comprender su aplicación para este caso en estudio.

Capítulo 5

Conclusiones y trabajos futuros

Los resultados de esta investigación revelan que, aunque la mayor parte del trabajo se ha centrado en una revisión exhaustiva del estado del arte —esencial para desarrollar una comprensión profunda de las técnicas de control y optimización estudiadas— también se ha llevado a cabo una implementación inicial de algunas de estas técnicas para evaluar su aplicabilidad. La implementación de un controlador PID básico permitió que la variable de proceso (pv) siguiera el setpoint, aunque con ciertas limitaciones atribuibles a la calidad del sensor y a la configuración mecánica del encoder. A pesar de estas limitaciones, la señal de proceso se ajustó al setpoint en un tiempo de respuesta aceptable. Para mejorar la estabilidad de la señal, afectada por la calidad del sensor, se podría considerar la implementación de un filtro, ya sea mediante hardware, software o una combinación de ambos.

Al comparar los tres algoritmos en estudio, se concluyó que las Redes Neuronales Artificiales (RNA) ofrecen el menor Error Cuadrático Medio (MSE) y los mejores resultados en términos de tiempos de subida, asentamiento y estabilización, con el menor overshoot, especialmente en tiempos de ejecución más prolongados. Aunque las RNA requieren más tiempo de entrenamiento y recursos, estos son compensados por una mayor precisión y desempeño.

En cuanto al Aprendizaje por Refuerzo (AR), con una hora de entrenamiento, se observan mejoras significativas en precisión y tiempos de desempeño. La capacidad de aprender y ajustar parámetros con más episodios es notable. Sin embargo, el tiempo de entrenamiento necesario y el número de episodios pueden ser considerablemente altos, lo que podría limitar la precisión en comparación con las RNA.

Respecto a la Búsqueda Aleatoria (BA), se observa que, al aumentar el número de iteraciones, mejora la precisión y los resultados de desempeño, aunque sigue siendo menos precisa en comparación con las RNA y el AR. A pesar de las mejoras observadas, la precisión de BA sigue siendo inferior a la de los otros métodos.

En resumen, con una hora de tiempo de ejecución, las RNA y el AR ofrecen una mejor precisión y desempeño general, siendo las RNA las que muestran los mejores resultados. La BA muestra mejoras en comparación con ejecuciones de 10 minutos, pero sigue siendo menos eficiente.

La creación de un gemelo digital simplificado permitió visualizar y analizar el comportamiento del sistema en un entorno virtual, evidenciando la alta demanda computacional del modelo PID y sugiriendo la necesidad de utilizar hardware más robusto en futuras aplicaciones. Aunque estas

implementaciones son preliminares, proporcionan una base sólida para futuras investigaciones que busquen combinar diferentes algoritmos de optimización con el objetivo de mejorar la calidad de los parámetros y reducir el número de iteraciones o épocas necesarias.

En conclusión, este trabajo abre el camino para estudios más profundos, enfocándose en dos líneas principales de exploración: el desarrollo de nuevos modelos de optimización y la creación de herramientas educativas para la enseñanza de tecnologías emergentes como la Inteligencia Artificial, los Gemelos Digitales y la Realidad Virtual, integradas en un sistema real. También se sugiere investigar la interacción entre diferentes plataformas de hardware y su aplicación en entornos educativos, como en el caso de un brazo robótico de seis grados de libertad, con el fin de enriquecer el aprendizaje de los estudiantes en estas áreas.

Bibliografía

[1] S. J. Russell y P. Norvig, Inteligencia Artificial un Enfoque Moderno, Madrid: Pearson Prentice Hall, 2004.

[2] P. H. Winston, Artificial Intelligence, Massachusetts: Addison-Wesley, 1993.

[3] R. S. Sutton y A. G. Barto, Reinforcement Learning. An Introduction, Cambridge: The MIT Press, 2018.

[4] H. A. Taha, Investigación de Operaciones, Naucalpan de Juárez: Pearson Educación de México, S.A. de C.V, 2012.

[5] F. S. Hillier y G. J. Lieberman, Introducción a la Investigación de operaciones, México, D.F.: McGRAW-HILL/INTERAMERICANA EDITORES, S.A. DE C.V., 2010.

[6] W. R. Sherman y A. B. Craig, Understanding Virtual Reality, San Francisco. California: Elsevier Science, 2003.

[7] G. Chaudhary, M. Khari y M. Elhoseny, Digital Twin Technology, Nueva York: CPC Press, 2022.

[8] G. Liu, F. Dai, X. Xu, X. Fu, W. Dou, N. Kumar y M. Bilal, «an adaptive DNN inference acceleration framework with end-edge-clud collaborative computing,» *Future Generation Computer Systems*, vol. 140, pp. 422-435, 2023.

[9] A. Nee y S. Ong, Digital Twins in Industry, Switzerland: MDPI, 2021.

[10] S. Deb Barma y V. Raghunathan, Digital Twin. A complete guide for the complete beginner, India: Kindle Edition, 2019.

[11] A. Canziani, A. Paszke y E. Culurciello, «An Analysis of Deep Neural Network Models for Practical Applications,» *Arxiv*, 2016.

[12] A. Canziani, E. Culurciello y A. Paszke , «Evaluation of neural network architectures for embedded systems,» de *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, Baltimore, MD, USA, 2017.

[13] T.-J. Yang, Y.-H. Chen, J. Emer y V. Sze, «A Method to Estimate the Energy Consumption of Deep Neural Networks,» de *51st Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, California, 2016.

[14] T.-J. Yang, Y.-H. Chen y V. Sze, «Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning,» de *arxiv*, Honolulu, 2017.

[15] X. Fang, H. Wang, G. Liu, X. Tian, G. Ding y H. Zhang, «Industry application of digital twin: from concept to implementation,» *The International Journal of Advanced Manufacturing Technology*, vol. 121, pp. 4289-4312, 2022.

[16] F. Tao, B. Xiao, Q. Qi, J. Cheng y P. Ji, «Digital twin modeling,» *Journal of Manufacturing Systems*, vol. 64, pp. 372-389, 2022.

[17] M. M. Rathore, S. A. Shah, D. Shukla, E. Bentafat y S. Bakiras, «The Role of AI, Machine Learning, and Big Data in Digital Twinning: A Systematic Literature Review, Challenges, and Opportunities,» *IEEE Access*, vol. 9, pp. 32030-32052, 2021.

[18] M. Segovia y J. Garcia-Alfaro, «Design, Modeling and Implementation of Digital Twins,» *Sensors*, vol. 22, nº 14, pp. 2-30, 2022.

[19] X. Liu, D. Jiang, B. Tao, F. Xiang, G. Jiang, Y. Sun, J. Kong y G. Li, «A systematic review of digital twin about physical entities, virtual models, twin data, and applications,» *Advanced Engineering Informatics*, vol. 55, 2023.

[20] R. Rayhana, L. Bai, G. Xiao, M. Liao y Z. Liu, «Digital Twin Models: Fucntions, Challenges and Industry Applications,» *IEEE JOURNAL OF RADIO FREQUENCY IDENTIFICATION*, vol. 8, pp. 282-321, 2024.

[21] S. Hu, M. Li, J. Gao, C. Zhou y X. Shen, «Adaptive Device-Edge Collaboration on DNN Inference in AIoT: A Digital-Twin-Assitede Approach,» *IEEE Internet of Things Journal*, vol. 11, nº 7, pp. 12893-12908, 2024.

[22] V. Reljic, I. Milenkovic, S. Dudic, J. Sulc y B. Bajci, «Augmented Reality Applications in Industry 4.0 Environment,» *applied sciences*, vol. 11, nº 5592, pp. 1-17, 2021.

[23] K.-B. Park, M. Kim, . S. H. Choi y . J. Y. Lee, «Deep learning-based smart task assistance in wearable augmented reality,» *Robotics and Computer-Integrated Manufacturing*, vol. 63, nº 101887, pp. 1-9, 2020.

[24] J. Egger y T. Masood, «Augmented reality in support of intelligent manufacturing - A systematic literature review,» *Computers & Industrial Engineering*, vol. 140, 2020.

[25] Z. Ren, T. Zhang, X. Liu y J. Lin, «A Novel Neuro PID Controller of Remotely Operated Robotic Manipulators,» *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, nº 6, 2023.

[26] B.-S. Chen, H.-T. Liu y R.-S. Wu, «Robust H∞ Fault-Tolerant Observer-Based PID Path Tracking Control of Autonomous Ground Vehicle With Control Saturation,» *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 298-311, 2024.

[27] G. Ablay, «Robotic Manipulator Control with Model-Independent Sliding Modes,» de *5th International Conference on Control and Robotics (ICCR)*, Tokyo, 2023.

[28] R. Zhao, J. Yang, X. Li y H. Mo, «Fuzzy Sliding-Mode Control for Robot Manipulators Based on Variable Universe,» de *IEEE 3rd International Conference on Digital Twins and Parallel Intelligence (DTPI)*, Orlando, Florida, 2023.

[29] R. Jacob y S. Murugan, «Implementation of neural network based PID controller,» de *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Channai, 2016.

[30] J. J. Caleb y M. Kannan, «Efficient VLSI Implementation of the C-MANTEC Conn Algorithm by Using PID Controllers,» *Circuits and Systems*, vol. 8, nº 11, pp. 253-260, 2017.

[31] M. Danu, M. G. Navaneeth, T. K. Sunil Kumar y A. P. Sudheer, «PSO Based Design of PID Controller for Speed Control of BLDC Motor for Robotic Applications,» de *IEEE Silchar Subsection Conference (SILCON)*, Silchar, 2023.

[32] M. Lohitha Chowdary, P. Premsai, M. Dasharna, I. Sri Kavya y S. T R, «Ocean Wave Height Forecasting using Deep Learning Neural Networks and Optimization Techniques,» de *Innovations in Power and Advanced Computing Technologies (i-PACT)*, Kuala Lumpur, 2023.

[33] K. B. Trujillo, J. G. Álvarez y E. Cortés, «PI and PID Controller Tuning with Deep Reinforcement Learning,» de *IEEE International Conference on Automation/XXV Congress of the Chilean Association of Automatic Control (ICA-ACCA)*, Curicó, 2022.

[34] A. C. Biju, O. A. Madhuria, M. M. Sam, A. Hari y A. Sebastian, «Effective Solar-Powered EV-Charging System with Enhanced Voltage Regulation,» de *9th International Conference on Smart Computing and Communications (ICSCC)*, Kerala, 2023.

[35] J. R. Simenthy y J. Mathana, «Exploring and Analysing Deep Reinforcement Learning Based Algorithms for Object Detection and Autonomous Navigation,» de *International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, Chennai, 2024.

[36] S. Tufenkci , B. B. Alagoz, G. Kavuran, C. Yeroglu, N. Herencsar y S. Mahata, «A theoretical demonstration for reinforcement learning of PI control dynamics for optimal speed control of DC motors by using Twin Delay Deep Deterministic Policy Gradient Algorithm,» *Expert Systems with Applications*, vol. 213, nº 119192, 2023.

[37] A. Ates, B. B. Alagoz, Y. Q. Chen, C. Yeroglu y S. H. HosseinNia, «Optimal Fractional Order PID Controller Design for Fractional Order Systems by Stochastic Multi Parameter Divergence Optimization Method with Different Random Distribution Functions,» de *IEEE 7th International Conference on Control, Mechatronics and Automation*, Delft, 2019.

[38] M. Akpamukçu y A. Ateş, «Analyzing of Usage Effect of the Distribution Functions for SMDO Algorithm via Benchmark Function with Matlab Toolbox,» *DergiPark AKADEMIK*, vol. 11, nº 3, pp. 989-998, 2020.

[39] A. Ates y M. Akpamukcu, «Modified monarch butterfly optimization with distribution functions and its application for 3 DOF Hover flight system,» *Neural Comput & Application*, vol. 34, pp. 3697-3722, 2022.

[40] P. Isasi Viñuela y I. M. Galván León, Redes de Neuronas Artificiales. Un enfoque práctico, Madrid: Pearson Educación S.A., 2004.

[41] J. R. Hilera y V. J. Martínez, Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones, Madrid: RA-MA, 1995.

[42] G. Zhang, M. Chen y LuRun, «An on-line adaptive hybrid PID autopilot of ship headingcontrol using auto-tuning BP & RBF neurons.,» de *Proceedings of the 39th Chinese Control Conference*, Shenyang, 2020.

[43] V. Kuznetsov, S. Dyadun y V. Esilevsky, «The control to aggregates of pumping stations using a regulator based on a neural network with fuzzy logic,» *E3S Web of Conferences*, nº 03007, pp. 102-107, 2019.

[44] M. Z. Fadel, M. G. Rabie y A. M. Youssef, «Motion control of an aircraft electro-hydraulic servo actuator,» de *18th Internatioantional Conference on Aerospace Sciences & Aviation Technology* , Cairo, Egipto, 2019.

[45] F. Kudlaþák y T. Krajþoviþ, «Artificial Neural Network for Adaptive PID Controller,» de *29th International Conference. 2018 Cybernetics & Informatics (K&I)*, Lazy pod Makytou, 2018.

[46] N. Khanh Quang, V.-Q.-B. Ngo, N. Kim Anh, H. Than, T. T. Dong y N. Duc Tho, «Neural Network PID Controller for PMSM Drives,» de *7th National Scientific Conference on Applying New Technology in Green Buildings (ATiGB)*, Da Nang, 2022.

[47] Z.-H. Zhou, «Neural Netwok,» de *Machine Learning*, Springer Link, 2021, pp. 103-128.

[48] E. Muravyova y A. Yusarov, «A Neural Network-Based Control System Using PID Controller To Control the Deaerator,» de *International Russian Automation Conference (RusAutoCon)*, Sochi, 2020.

[49] A. C. Biju, O. A. Madhuria, M. M. Sam, A. Hari y A. Sebastian, «Effective Solar-Powered EV-Charging System with Enhanced Voltage Regulation,» de *9th International Conference on Smart Computing and Communications (ICSCC)*, Kerala, 2023.

[50] X. Yu, Y. Fan y L. Ou, «A self-adaptive SAC-PID control approach based on reinforcement learning for mobile robots,» *International Journal of Robust and Nonlinear Control,* vol. 32, nº 18, pp. 9625-9643, 2022.

[51] «Optimal Fractional Order PID Controller Design for Fractional Order Systems by Stochastic Multi Parameter Divergence Optimization Method with Different Random Distribution Functions.,» de *IEEE 7th International Conference on Control, Mechatronics and Automation*, Delft, 2019.