UNIVERSIDAD PRIVADA ANTENOR ORREGO

FACULTAD DE INGENIERÍA

PROGRAMA DE ESTUDIO DE INGENIERÍA ELECTRÓNICA

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE

INGENIERO ELECTRÓNICO

Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal

---

Línea de investigación: Robótica y programación avanzada

Autores:

García Henriquez, Paul Jhon

Saucedo Zafra, Jorge Armando

Jurado evaluador:

Presidente : Linares Vertiz, Saúl Noel

Secretario : Vargas Diaz, Luis Alberto

Vocal : Mendoza Chomba, Jorge Esteban

Asesor:

Ms. De La Cruz Rodríguez, Oscar Miguel

Código Orcid: https://orcid.org/0000-0001-9207-8558

Trujillo–Perú

2025

Fecha de Sustentación:2025/07/14

Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal

por Paul Jhon Garcia Henriquez

Fecha de entrega: 10-jul-2025 12:56p. m. (UTC-0500)
Identificador de la entrega: 2712949329
Nombre del archivo: Tesis_-_Garc_a_Saucedo.pdf (2.06M)
Total de palabras: 10008
Total de caracteres: 54280

Ms. Ing. Oscar De La Cruz Rodríguez
CIP: 85598

Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal

INFORME DE ORIGINALIDAD

10%	10%	2%	6%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

Ms. Ing. Oscar De La Cruz Rodrígu
CIP: 85598

FUENTES PRIMARIAS

1	repositorio.untels.edu.pe Fuente de Internet	1%
2	repositorio.umsa.bo Fuente de Internet	1%
3	www.jstor.org Fuente de Internet	1%

Excluir citas	Apagado	Excluir coincidencias	< 1%
Excluir bibliografía	Activo		

Ms. Ing. Oscar De La Cruz Rodríguez
CIP: 85598

Jurado de sustentación Oral

Linares Vertiz Saúl Noel

N° CIP 142213

Presidente

Vargas Diaz Luis Alberto

N° CIP 104175

Secretario

Mendoza Chomba Jorge Esteban

N° CIP 338937

Vocal

Entregado el:

García Henriquez Paul Jhon

DNI 47587235

Aprobado por:

Saucedo Zafra Jorge Armando

DNI 46401428

De La Cruz Rodríguez Oscar Miguel

Asesor de Tesis

UNIVERSIDAD PRIVADA ANTENOR ORREGO

FACULTAD DE INGENIERÍA

PROGRAMA DE ESTUDIO DE INGENIERÍA ELECTRÓNICA

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE

INGENIERO ELECTRÓNICO

Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal

---

Línea de investigación: Robótica y programación avanzada

Autores:

García Henriquez, Paul Jhon

Saucedo Zafra, Jorge Armando

Jurado evaluador:

Presidente : Linares Vertiz, Saúl Noel

Secretario : Vargas Diaz, Luis Alberto

Vocal : Mendoza Chomba, Jorge Esteban

Asesor:

Ms. De La Cruz Rodríguez, Oscar Miguel

Código Orcid: https://orcid.org/0000-0001-9207-8558

Trujillo–Perú

2025

Fecha de Sustentación:2025/07/14

DECLARACION DE ORIGINALIDAD

Yo, Oscar Miguel De La Cruz Rodríguez, docente del Programa de Estudio de Pregrado de la Universidad Privada Antenor Orrego, asesor de la tesis titulada “Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal”, de los autores Paul Jhon García Henriquez y Jorge Armando Saucedo Zafra.

• El mencionado documento tiene un índice de puntuación de similitud del **10%**. Así lo consigna el reporte de similitud emitido por el software Turnitin el día 10 de julio del 2025.

• He revisado con detalle dicho reporte de la tesis “Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal” y no se advierte indicios de plagio.

• Las citas a otros autores y sus respectivas referencias cumplen con

las normas establecidas por la Universidad.

Trujillo, 10 de julio del 2025

GARCÍA HENRIQUEZ PAUL JHON
DNI 47587235

SAUCEDO ZAFRA JORGE ARMANDO
DNI 46401428

DE LA CRUZ RODRÍGUEZ OSCAR MIGUEL
DNI: 40545044
ORCID: 0000-0001-9207-8558

Dedicatoria

Dedico esta tesis principalmente a Dios por ser el inspirador
y darme fuerza para continuar durante todo este proceso
para obtener uno de los anhelos más deseados.
A mis padres Carlos Garcia Rodriguez y Esther Henriquez Álvarez,
por su amor, trabajo y sacrificio en todos estos años,
gracias a ustedes he logrado llegar hasta aquí y
convertirme en lo que soy.
Agradezco profundamente su apoyo y motivación,
sin los cuales no habría podido alcanzar este logro.

Paul Garcia

Agradecimientos

Expreso mi más sincero agradecimiento
a quienes hicieron posible este sueño.
Esta mención en especial a Dios por su protección.
A mis padres por sus consejos valores y principios que me inculcaron.
A mi asesor de tesis Oscar de la Cruz Rodríguez por su valiosa orientación
y apoyo constante durante la realización de este trabajo.
Agradezco a mis docentes de la Escuela de Ingeniería Electrónica
de la Universidad Privada Antenor Orrego, por haber compartido sus conocimientos
a lo largo de la preparación en mi profesión,
gracias por su apoyo y confianza

Paul García

A nuestro creador por darnos la sabiduría, paciencia y
fortaleza para lograr culminar este reto.
A mis padres, por el apoyo constante ante las adversidades que se
presentaron en mi camino y con perseverancia lograr culminar los
estudios.
A mi esposa e hijos, por ser el estímulo para seguir adelante y lograr un
mejor futuro para nuestra familia.
A mis amigos, que fueron la voz que dieron aliento para poder escalar este
peldaño pendiente por años.
A nuestro asesor de Tesis, Ing. Oscar Miguel de la Cruz Rodríguez,
por su apoyo constante, tiempo y dedicación para poder culminar
este proyecto indispensable para nuestra vida profesional.
Jorge Saucedo

Resumen

El objetivo del presente trabajo es diseñar una red neuronal en un PLC S7-1200 para la detección de color, en procesos industriales de clasificación.

En principio en el trabajo se describen los aspectos del diseño de investigación y justificación del estudio, para luego presentar las bases teóricas que permita comprender la implementación de las redes neuronales en el controlador SIEMENS S7-1200. A partir del análisis de la información obtenida se evalúan aspectos técnicos para determinar la eficiencia del sistema basado en redes neuronales aplicados en controladores industriales. Se logro diseñar la red neuronal basada en una capa con tres neuronas con función de activación SIGMOIDE y con tres entradas y tres salidas. Las cuales determinan los colores de detección basado en las frecuencias de entrada a la red neuronal.

Por último, se logró determinar la eficiencia promedio de la red con 16.33 aciertos de 20 tomas. Lo que permite una eficiencia 81%.

Palabras Claves: Proceso, controlador, neurona

Abstract

The objective of this work is to design a neural network in an S7-1200 PLC for color detection in industrial sorting processes.

Initially, the work describes the research design and justification of the study, and then presents the theoretical basis for understanding the implementation of neural networks in the SIEMENS S7-1200 controller. Based on the analysis of the information obtained, technical aspects are evaluated to determine the efficiency of the neural network-based system applied to industrial controllers. The neural network was designed based on a layer with three neurons with a SIGMOID activation function and with three inputs and three outputs. These determine the detection colors based on the input frequencies to the neural network.

Finally, the average efficiency of the network was determined, with 16.33 correct answers out of 20 inputs, resulting in an efficiency of 81%.

Keywords: Process, controller, neuron

Presentación

Señores miembros del Jurado:

De conformidad con lo estipulado en el Reglamento de Grados y Títulos de la Universidad Privada Antenor Orrego, ponemos a su disposición el informe de tesis titulado “Diseño de un sistema para selección de colores a través del controlador lógico programable S7-1200 utilizando una red neuronal” para que sea revisado y evaluado y de ser aprobado pueda ser defendido oralmente para optar el título profesional de Ingeniero Electrónico.

De antemano, nos excusamos de los errores involuntarios en que se hubiera incurrido en el desarrollo y redacción del misma, esperando del honorable jurado un justo dictamen.

García Henriquez Paul Jhon

Saucedo Zafra Jorge Armando

Tabla de contenido

I. INTRODUCCIÓN

1.1. Problema de investigación

En la industria manufacturera, la detección de color es un factor clave en procesos como clasificación de productos, control de calidad y embalaje. Actualmente, los sistemas de detección de color en PLCs utilizan sensores ópticos que operan bajo umbrales predefinidos, lo que los hace poco flexibles ante variaciones en iluminación, textura o degradación del color. Esto puede generar errores en la identificación de productos, afectando la calidad y eficiencia de la producción.

Los controladores lógicos programables, como el *Siemens S7-1200*, son ampliamente utilizados en la automatización industrial, pero su programación tradicional basada en lógica booleana y comparaciones fijas limita su capacidad para manejar variaciones complejas en el reconocimiento de color. Esto implica una necesidad constante de calibración manual y ajustes en los parámetros, lo que incrementa costos y tiempos de inactividad.

Las redes neuronales artificiales (RNA) han demostrado ser altamente efectivas en el procesamiento de imágenes y la clasificación de patrones, ya que pueden aprender a identificar colores con mayor precisión y adaptarse a variaciones en tiempo real. Sin embargo, la implementación de redes neuronales en un *PLC S7-1200* presenta desafíos técnicos, como restricciones de procesamiento y compatibilidad con los lenguajes de programación estándar de estos dispositivos.

Por ello, es necesario investigar la *implementación de una red neuronal en un PLC S7-1200 para la detección de color*, con el fin de mejorar la precisión, reducir errores y optimizar los procesos industriales que dependen de esta tecnología. Esta solución permitiría una mayor adaptabilidad y autonomía en la identificación de colores, avanzando hacia sistemas de automatización más inteligentes y eficientes.

1.2. Objetivos

1.2.1. Objetivo general

Diseñar una red neuronal en un PLC S7-1200 para la detección de color, en procesos industriales de clasificación.

1.2.2. Objetivos específicos

- ✓ Diseñar un modelo de red neuronal artificial capaz de identificar y clasificar colores en entornos industriales.
- ✓ Implementar la red neuronal en un PLC S7-1200, integrándola con sensores de color y evaluando su compatibilidad.
- ✓ Determinar la eficiencia del sistema basado en redes neuronales aplicados en controladores industriales.

1.3. Justificación del estudio

En lo académico:

La implementación de redes neuronales en *PLCs* es un área emergente dentro de la automatización industrial y la inteligencia artificial aplicada. Esta investigación contribuirá al desarrollo del conocimiento en *procesamiento de señales, control automático e inteligencia artificial*, proporcionando un

enfoque innovador para la detección de color en entornos industriales. Además, servirá como referencia para futuras investigaciones en el campo de la automatización inteligente, impulsando el aprendizaje interdisciplinario entre la ingeniería electrónica, la computación y la inteligencia artificial.

En lo tecnológico:

Los sistemas de detección de color actuales en PLCs dependen de umbrales fijos y requieren calibraciones manuales constantes. Integrar una *red neuronal en un PLC S7-1200* permitirá desarrollar un sistema *más preciso, adaptable y autónomo, optimizando los procesos industriales de clasificación, inspección y control de calidad. Este avance fortalecerá la aplicación de técnicas de **aprendizaje automático en la industria 4.0*, promoviendo la evolución de los sistemas de automatización y su integración con tecnologías avanzadas.

En lo social:

Desde una perspectiva social, esta investigación contribuirá a mejorar la eficiencia productiva y la calidad de los bienes fabricados, reduciendo desperdicios y errores en la clasificación de productos. Esto impactará positivamente en la competitividad de las industrias, generando empleo en sectores tecnológicos y favoreciendo la adopción de soluciones innovadoras. Además, permitirá una producción más sostenible al minimizar la tasa de defectos y mejorar el uso eficiente de los recursos.

En conclusión, este estudio es relevante en los ámbitos académico, tecnológico y social, ya que impulsa el desarrollo del conocimiento, mejora la

eficiencia en la industria y contribuye a la innovación en la automatización inteligente.

II. MARCO DE REFERENCIA

2.1. Antecedentes del estudio

Gonzales del Valle Romero y Neyra Espinoza (2022), en la investigación "Implementación de una red neuronal artificial convolucional para la detección y conteo de vehículos en una sección del estacionamiento de la Universidad Ricardo Palma", el propósito del trabajo fue desarrollar un sistema de visión artificial que utilice redes neuronales para detectar y contabilizar vehículos en tiempo real. La investigación llegó a los siguientes resultados: se logró una detección eficiente mediante los modelos YOLO v5 y Faster R-CNN con alta precisión en escenarios reales. El principal aporte al trabajo de investigación es la validación del uso de redes neuronales convolucionales para aplicaciones de monitoreo automatizado en espacios universitarios.

Pampamallco Jara (2019), en la investigación *"Diseño e implementación de controladores basados en lógica difusa para sistemas de control industrial"*, el propósito del trabajo fue aplicar técnicas de control inteligente como la lógica difusa para mejorar el desempeño de procesos industriales. La investigación llegó a los siguientes resultados: se logró una respuesta más estable y adaptativa frente a perturbaciones externas en comparación con los controladores clásicos. El principal aporte al trabajo de investigación es la demostración de la viabilidad de incorporar técnicas de inteligencia computacional en sistemas de control industrial programables, abriendo paso a soluciones como redes neuronales.

Medina Carrillo y Zevallos Peña (2020), en la investigación *"Diseño de un sistema de clasificación de maracuyá según su estado de madurez utilizando*

visión artificial y PLC S7-1200", el propósito del trabajo fue automatizar la clasificación de maracuyá mediante el reconocimiento de color y madurez usando visión artificial integrada a un PLC S7-1200. La investigación llegó a los siguientes resultados: se desarrolló un sistema funcional capaz de diferenciar frutas maduras de inmaduras con una precisión mayor al 85%. El principal aporte al trabajo de investigación es la combinación de visión artificial con controladores lógicos programables en procesos agroindustriales, mostrando el potencial de integrar redes neuronales en futuras versiones.

Chávez Huamán y Gonzales Gutiérrez (2021), en la investigación *"Propuesta de implementación de un sistema automático IoT para controlar parámetros* de *un generador de rayos X convencional mediante Node-RED y PLC S7-1200"*, el propósito del trabajo fue desarrollar un sistema de automatización remota que permita monitorear y controlar parámetros de operación de un generador de rayos X, integrando tecnologías IoT con el PLC S7-1200. La investigación llegó a los siguientes resultados: se logró una comunicación efectiva entre el entorno gráfico Node-RED y el PLC, permitiendo la operación segura y precisa del equipo. El principal aporte al trabajo de investigación es demostrar la viabilidad de integrar tecnologías emergentes como IoT con PLCs de gama media, abriendo posibilidades para futuras aplicaciones con inteligencia artificial o redes neuronales.

Ramírez Alarcón (2020), en la investigación *"Implementación de un control neuronal directo en el controlador lógico programable S7-1200 para aplicaciones industriales"*, el propósito del trabajo fue diseñar un sistema de control

automático basado en una red neuronal entrenada, capaz de tomar decisiones en tiempo real a través de un PLC Siemens S7-1200. La investigación llegó a los siguientes resultados: se logró una integración funcional de la red neuronal con el PLC mediante bloques de función, y se comprobó una mejora en la eficiencia del control frente a métodos tradicionales. El principal aporte al trabajo de investigación es la implementación práctica de una red neuronal dentro de un sistema automatizado, validando su aplicabilidad en la industria nacional.

2.2. Marco teórico

Redes neuronales artificiales

La Agencia de Investigación de Proyectos Avanzados de Defensa (DARPA), define una red neuronal artificial como un sistema compuesto de muchos elementos simples de procesamiento los cuales operan en paralelo y cuya función es determinada por la estructura de la red y el peso de las conexiones, donde el procesamiento se realiza en cada uno de los nodos o elementos de cómputo. (Caicedo B y López S, 2009).

El conocimiento es obtenido por la red a través de un proceso de aprendizaje. Las conexiones entre neuronas, conocidas como pesos sinápticos, con utilizadas para almacenar dicho conocimiento.

Las redes neuronales imitan la estructura y la función del cerebro humano mediante el procesamiento de información no lineal y la computación paralela a gran escala con una capacidad de almacenamiento distribuido. Cuando se trata con varios tipos de información, las redes neuronales aprenden por sí mismas y ajustan constantemente sus sistemas para adaptarse a un entorno

variable. Por lo tanto, se usan ampliamente en el reconocimiento de patrones, para predecir cambios en las cosas, optimizar decisiones, mejorar el control del proceso y otras tareas. (Ye y Kim, 2018) Según Haykin, una red neuronal es un procesador paralelo masivamente distribuido que tiene una facilidad natural para el almacenamiento de conocimiento obtenido de la experiencia para luego hacerlo utilizable. Se parece al cerebro en dos aspectos:(Caicedo B y López S, 2009).

El algoritmo Algoritmo Levenberg-Marquardt (LM) es una red neuronal utilizada en un algoritmo de entrenamiento no lineal, donde combina el método de descenso de gradiente y el método Quasi-Newton para garantizar la velocidad de convergencia localmente rápida y mantener un mejor rendimiento general. La idea básica de este algoritmo es que cada iteración no es más larga de lo necesario a lo largo de un solo gradiente en la dirección negativa y permite buscar el error en la dirección de descenso. Además, los pesos de la red pueden optimizarse mediante el ajuste adaptativo entre el método de descenso de gradiente más pronunciado y el método de Gauss-Newton, que permite que la red converja de manera efectiva y esto mejora en gran medida la velocidad de convergencia y la generalización de la red. (Marulanda Gonzalez, 2010)

Las redes neuronales se pueden utilizar para implementar controladores altamente no lineales con pesos o parámetros internos que se autodeterminan mediante un proceso de aprendizaje, al cambiar el peso de un elemento alterará el comportamiento de la red. El objetivo es encontrar los pesos de la

red para lograr una relación entrada/ salida deseada. (Nguyen y Widrow, 1990)

La ventaja de las redes neuronales artificiales frente a sistemas matemáticos o expertos es que la función gana complejidad cuanto mayor es el número y las combinaciones de estas. Al igual que las neuronas biológicas, la muerte o deterioro de una neurona afecta cuantitativamente, pero no cualitativamente. Esto les confiere características que las hacen muy adecuadas para la realización de tareas tales como identificación, reconocimiento de patrones y sobre todo control. (Marulanda Gonzalez, 2010)

La distribución de las neuronas dentro de una red neuronal artificial se realiza formando niveles de un número de neuronas determinado. Si un conjunto de neuronas artificiales recibe simultáneamente el mismo tipo de información, lo denominaremos capa. En una red podemos diferenciar tres tipos de niveles:(Caicedo B y López S, 2009)

Entrada: Es el conjunto de neuronas que recibe directamente la información proveniente de las fuentes externas de la red.

- Oculto: Corresponde a un conjunto de neuronas internas a la red y no tiene contacto directo con el exterior.
- Salida: Es el conjunto de neuronas que transfieren la información que la red ha procesado hacia el exterior.

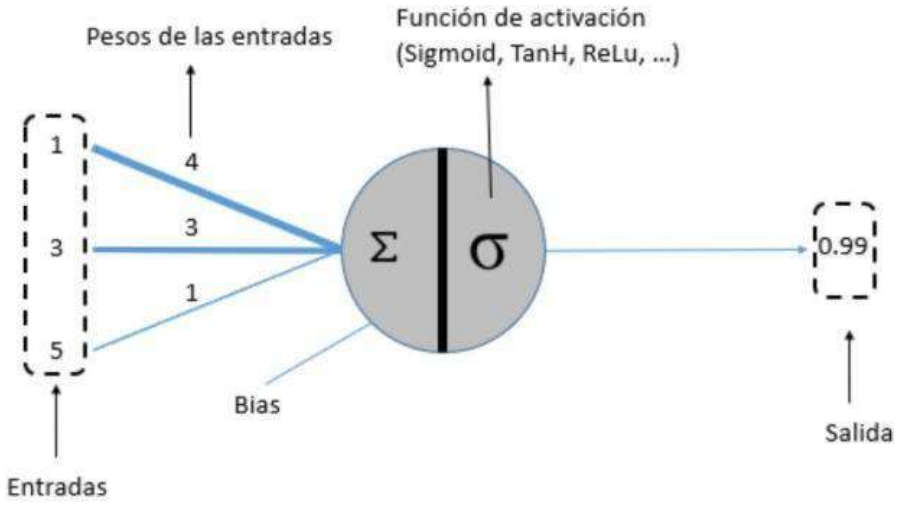
La arquitectura de una red neuronal artificial ver figura 4.4 es la forma como se organizan las neuronas en su interior y está estrechamente ligada al algoritmo de aprendizaje usado para entrenar la red. Dependiendo del número

de capas, definimos las redes como monocapa y multicapa. (Caicedo B y López S, 2009)

Una red neuronal artificial es capaz de aprender por si misma para controlar sistemas dinámicos no lineales (es decir, que sus parámetros cambian conforme pasa el tiempo), esto es quizá una de las razones por las que más se utiliza este tipo de control. Una red neuronal de múltiples capas, aprende a identificar las características dinámicas del sistema. El controlador, otra red neuronal multípara, que aprende a controlar al emulador. El controlador autodidacta se usa para controlar el sistema dinámico real. El proceso de aprendizaje continúa a medida que el emulador y el controlador mejoran y rastrean el proceso físico el cual es el cálculo procedente al relacionar las entradas y salidas del sistema. (Nguyen y Widrow, 1990)

Figura 1

Red Neuronal



Nota: Estructura de una red neuronal (Caicedo y cols)

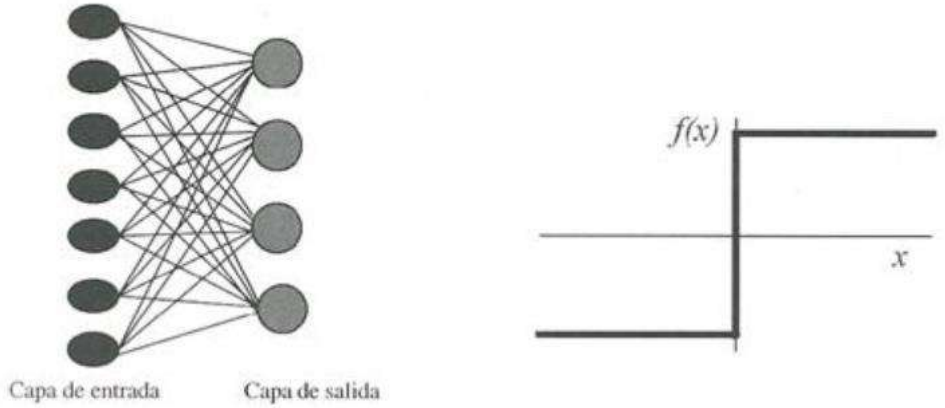
El Perceptrón

El Perceptrón fue el primer modelo de RNA presentado a la comunidad científica por el psicólogo Frank Rosenblatt en 1958. Como es natural despertó un enorme interés en la década de los años sesenta, debido a su capacidad para aprender a reconocer patrones sencillos con una superficie de separación lineal, razón por la cual fue también objeto de severas críticas que terminaron por dejar en el olvido la propuesta de Rosenblatt. La estructura del Perceptrón es supremamente sencilla: en su entrada posee varias neuronas lineales que se encargan de recibir el estímulo externo de la red y a la salida presenta una única capa de neuronas, con función de activación binaria o bipolar lo cual trae como consecuencia que la salida de cada neurona esté entre dos posibles valores. (Caicedo B y López S, 2009)

La estructura del perceptrón ver figura 4.5 se inspira en las primeras etapas de procesamiento de los sistemas sensoriales de los animales (por ejemplo, el de visión), en los cuales la información va atravesando sucesivas capas de neuronas, que realizan un procesamiento progresivamente de más alto nivel. El perceptrón simple es un modelo unidireccional, compuesto por dos capas de neuronas, una sensorial o de entradas, y otra de salida. (Martín del Brio y Sanz, 2001)

Figura 2

Perceptrón simple



Nota: Perceptrón simple y función de transferencia de una neurona

(Caicedo y cols)

Las neuronas de entrada no realizan ningún computo, únicamente envían la información a las neuronas de salida. La función de activación de las neuronas de la capa de salida es de tipo escalón o Heaviside. El perceptrón puede utilizarse tanto como clasificador, como para la representación de funciones booleanas, pues su neurona es esencialmente de tipo MacCulloch-Pitts, de salida binaria. Cada neurona del perceptrón representa una determinada clase, de modo que, dado un vector de entrada, una cierta neurona responde con 0 si no pertenece a la clase que representa, y con un 1 si si pertenece. Es fácil ver que una neurona tipo perceptrón solamente permite discriminar entre dos clases linealmente separables (es decir, cuyas regiones de decisión pueden ser separadas mediante una única condición lineal o hiperplano). (Martín del Brio y Sanz, 2001)

Por lo tanto, pese a su gran interés, el perceptrón presenta serias limitaciones, pues solamente puede representar funciones linealmente separables. Así, aunque pueda aprender automáticamente a representar complejas funciones booleanas o resolver con éxito muchos problemas de clasificación, en otras ocasiones fallará estrepitosamente. (Martín del Brio y Sanz, 2001).

Controlador Lógico Programable:

Un autómata programable industrial es un equipo electrónico, programable en lenguaje no informático, diseñado para controlar en tiempo real y en ambiente de tipo industrial procesos secuenciales. El autómata programable también se conoce como PLC, que es la sigla de Programmable Logic Controller. Tal y como se resume en la definición, se trata de un computador especial, tanto en el software como en el hardware. En el software, porque se programa en un lenguaje especial diseñado específicamente para generar de forma sencilla el programa que implementa el algoritmo de control de procesos secuenciales (de sistemas de eventos discretos), y porque el algoritmo de control programado es ejecutado de forma periódica en un ciclo temporal que es lo bastante breve como para poder controlar los procesos en tiempo real.

En el hardware, porque utiliza componentes robustos que soportan condiciones de trabajo adversas, como las que se dan en ambientes industriales (polvo, temperatura, vibraciones, etc.), y porque su constitución física incluye los circuitos de interfaz necesarios para conectarlo de forma directa a los sensores y actuadores del proceso. (Sanchis Llopis y cols., 2010)

El PLC es un instrumento electrónico que sirve de herramienta para dar solución a problemas de automatización especialmente en el ámbito industrial, dentro de los lenguajes de programación soportados están: el lenguaje escalera, bloques funcionales y texto estructurado. (Quiña, 2014).

Como computador que es, tiene un procesador que es el que ejecuta el programa almacenado en la memoria de programa. La memoria de programa y la de datos están físicamente separadas, constituyendo una arquitectura tipo Harvard. Además, la memoria de datos está separada en dos tipos, que en la figura se denominan memoria de datos y memoria interna. Esta última se utiliza para almacenar los valores de las señales de entrada y salida, por lo que están conectadas con los módulos de entradas y salidas, que son los elementos de interfaz donde se conectan los sensores y actuadores del proceso. También dispone de periféricos para comunicas con otros dispositivos, como pantallas táctiles, ordenadores u otros autómatas.

Lenguaje de Programación

En la siguiente tabla 1 se encuentran mencionados los lenguajes de PLC's más utilizados en la actualidad. En el siguiente apartado se verán los lenguajes de programación en PLC's que se tratarán en esté trabajo.

Texto Estructurado (ST)

ST es un lenguaje de programación de alto nivel, similar a la Programación Pascal. ST es desarrollado y publicado por la Comisión Internacional Electro-técnica (IEC) en IEC 61131-3 International Satandard en 1993. El estándar consiste de 5 lenguajes de programación PLC, donde la Programación

LADDER es la más conocida y utilizada. La programación ST para los Controles de PLC ha sido publicada en varias ocasiones desde aproximadamente 2010; y desde 2015. (Antonsen, 2019). En los PLC de Siemens, la programación es llamada Lenguaje de Control Estructurado (SCL), el cual incluye algunas diferencias en relación al lenguaje de programación Estructurado de Texto (ST).

ST es un lenguaje de programación muy flexible y universal. El código del programa ST puede ser fácilmente replicado entre diferentes tipos de PLC y puede ser enviado vía e-mail, debido a que se encuentra basado en texto y no en gráficas como la programación LADDER lo hace. El código del programa ST es similar a las oraciones de texto y el trabajo es llevado a cabo como en un programa procesador de textos (como, por ejemplo, Microsoft Word); lo cual lo hace más fácil trabajar con él. Consecuentemente, los mismos métodos de trabajo son aplicados como en un programa procesador de textos. (Antonsen, 2019)

Debido a su muy estructurada naturaleza, ST es ideal para tareas basadas en matemáticas complejas, reutilización de códigos o toma de decisiones (por ejemplo, optimización automática de energía, algoritmos, recolección y regulación de datos en plantas de proceso). Contando con la experiencia con programación PLC, la transición hacia otros lenguajes de programación con Control PLC y automatización, deberá ser más sencilla; es decir, robótica de programación o programación Visual Basic.

En los últimos años, aún más compañías han cambiado a la Programación ST, lo cual se debe al hecho de que el ST provee una serie de ventajas comparado con los otros cuatro lenguajes de programación PLC (LADDER, SFC, FBD e IL). (Antonsen, 2019) Las ventajas son las siguientes:

- El código de programación ST puede ser replicado relativamente fácil entre diferentes tipos de PLC.
- Es el lenguaje PLC más sencillo para cálculos matemáticos, fórmulas y algoritmos 2) y una gran cantidad de datos (big-data).

Las soluciones PLC son más demandantes ahora que 20 años atrás.

Muchos lenguajes generalizados de programación de PC (C++, C, VB, PASCAL) recuerda mucho la estructura del programa ST.

Los otros lenguajes PLC (LAD, SFC y FBD) requieren que algunas partes de ellos estén programadas en ST.

Utiliza menos espacio cuando el código PLC debe ser documentado o descrito.

Es el lenguaje de PLC más sencillo para el control de versiones a través de comentarios en el código de programa. (Antonsen, 2019)

Lenguaje de Control Estructurado (SCL)

SCL “Structured Control Language” es el lenguaje de texto estructurado para PLC’s de Siemens y está basado en el lenguaje de alto nivel PASCAL. Permite una fácil integración en el contexto de una solución global para un problema de automatización ya que un bloque programado en SCL puede ser llamado desde un bloque escrito en Ladder, en grafcet, en AWL o en FUP y a la

inversa, un bloque escrito, por ejemplo, en KOP puede ser llamado desde un bloque escrito en SCL. Al mismo tiempo SCL trata todas las áreas de memoria del PLC como variables globales, lo que permite, como en el resto de lenguajes, intercalar una dirección absoluta de memoria (entrada, salida, marca, DB, periferia, etc.) en el área de instrucciones del bloque como si se tratara de una variable del bloque. (Barnes, 2007)

SCL está especialmente indicado para la resolución de los siguientes tipos de problemas:

- Problemas matemáticos complejos como los siguientes ejemplos:

  Función de normalizado de variables analógicas.

  Cálculo de volúmenes y pesos de depósitos cónico-cilíndricos.

  Cálculo de tiempos de activación o funcionamiento de dispositivos.

- Problemas de tratamiento de datos, como los siguientes ejemplos:

  Filtrado de una variable analógica.

  Determinación de media y valores extremos en un grupo de valores.

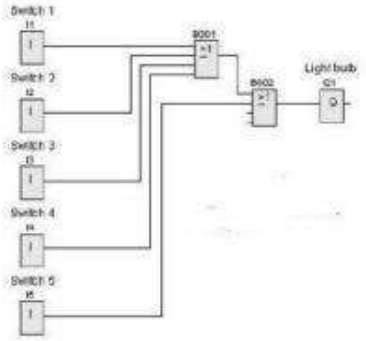
  Detección de errores repetidos por desviaciones fuera de tolerancia en distintos instrumentos.

- Manejo de datos en matrices, como los siguientes ejemplos.
- Determinación de la posición de carga y descarga de piezas en un almacén.
- Mantenimiento de un histórico de eventos en un PLC.
- Impresión de un histórico de valores de producción en un dosificador de colas.

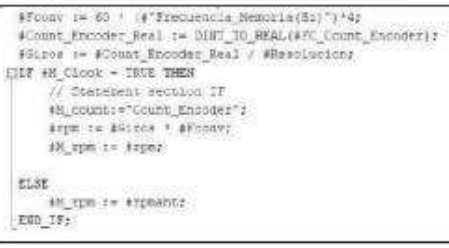
- Gestión de la presentación en pantalla de datos de mantenimiento y funcionamiento de válvulas y motores. •
- Gestión de contadores de producto en máquina plegadora de sábanas. (Barnes, 2007)

Figura 3

Lenguaje de Programación en PLC

Lenguaje	Siglas	Ejemplo
Escalera (Ladder)	LD	
Bloques de Funciones	FBD	
Lista de Instrucciones	IL	**Instruction list language** LD %I1.1 R %C8 LD %I1.2 AND %M0 CU %C8 LD %C8.D ST %Q2.0

Texto Estructurado	ST	**IF S1 = TRUE THEN**
		K1:= TRUE;
		END_IF;
		
		IF S2 = TRUE THEN
		K1:= FALSE;
		END_IF;

Lenguaje de Control Estructurado	SCL	

Sensor de color

Un sensor de color es un dispositivo óptico que permite detectar la composición cromática de un objeto u superficie mediante la medición de la luz reflejada en las bandas espectrales de rojo (R), verde (G) y azul (B). Su funcionamiento se basa en fotodiodos sensibles a longitudes de onda específicas y filtros ópticos que separan la luz incidente para cada componente cromática, generando señales eléctricas proporcionales a la intensidad detectada. Estas señales son posteriormente procesadas para identificar el color observado.

Según el fabricante SICK AG (2022), los sensores de color industriales están diseñados para reconocer colores con alta precisión, incluso cuando existen ligeras variaciones de tono, brillo o superficie. Esto se logra mediante la emisión de luz blanca o RGB y el análisis espectral reflejado. La comparación de los valores RGB capturados con una base de datos predefinida permite la clasificación automática de objetos por color.

Tipos de sensores de color industriales

Los sensores de color industriales pueden clasificarse en:

Sensores RGB estándar: Capturan la intensidad de las componentes R, G y B del espectro visible. Son los más comunes en aplicaciones básicas de reconocimiento de color.

Sensores espectrales avanzados: Miden en múltiples longitudes de onda y pueden diferenciar colores muy similares (por ejemplo, tonalidades de blanco o azul). Utilizados en industrias de impresión, textiles y farmacéutica.

Sensores con aprendizaje adaptativo: Incorporan algoritmos de calibración automática y aprendizaje, permitiendo mayor precisión en ambientes cambiantes.

Aplicaciones industriales

Los sensores de color se utilizan ampliamente en entornos industriales para las siguientes tareas:

- Clasificación de productos: En líneas de producción automatizadas, los productos se separan según su color (por ejemplo, cápsulas farmacéuticas, botellas de plástico o frutas).
- Control de calidad: Verificación del color correcto en etiquetas, pinturas o componentes, detectando defectos visuales.
- Empaque automático: Detección de códigos de color para dirigir productos a diferentes líneas de envasado.
- Industria alimentaria: Detección del grado de madurez o deterioro de frutas y verduras.

Integración con sistemas automatizados

Estos sensores pueden integrarse directamente a Controladores Lógicos Programables (PLC) mediante interfaces industriales como PNP/NPN, salida analógica, RS232, IO-Link o Ethernet, dependiendo del modelo. Su combinación con controladores como el Siemens S7-1200 permite ejecutar acciones automáticas según el color detectado, como la activación de servomotores, cilindros neumáticos o señales visuales.

2.3. Marco conceptual

Sensor de Color

Es un dispositivo óptico que detecta el color de una superficie u objeto mediante la medición de la luz reflejada en los componentes rojo, verde y azul (RGB). Se utiliza para clasificar objetos, controlar calidad o automatizar procesos industriales según características cromáticas (SICK AG, 2022).

Software MATLAB

MATLAB es un entorno de programación y cálculo numérico desarrollado por MathWorks. Está orientado al análisis matemático, simulación de sistemas, visualización de datos y desarrollo de algoritmos. Incorpora herramientas como Neural Network Toolbox para el modelado de redes neuronales artificiales (MathWorks, 2023).

Software TIA Portal

TIA Portal (Totally Integrated Automation Portal) es el entorno de desarrollo integral de Siemens para la automatización industrial. Permite programar, configurar y simular dispositivos como PLCs, HMIs y redes industriales, integrando todos los componentes en una sola plataforma (Siemens, 2020).

2.4. Sistema de hipótesis

La implementación de una red neuronal entrenada e integrada al PLC S7-1200 permite la detección de colores en controladores industriales

2.5. Variables e indicadores

Tabla 1

Operacionalización de la variable independiente: Red neuronal

Definición Conceptual	Definición Operacional	Dimensiones	Indicadores	Instrumento
Modelo computacional inspirado en el funcionamiento del cerebro humano, compuesto por unidades elementales llamadas *neuronas artificiales (Haykin, 2009).*	Una **red neuronal** se entiende como un modelo diseñado, entrenado y simulado en el entorno MATLAB, específicamente para clasificar colores captados por un sensor RGB.	Precisión de clasificación de color Tiempo de procesamiento del sistema	Nro de capas Nro de neuronas Data de aprendizaje	Reporte de diseño

Tabla 3

Operacionalización de la variable dependiente: Detección de color

Definición Conceptual	Definición Operacional	Dimensiones	Indicadores	Instrumento
Es un dispositivo electrónico que permite identificar los colores de un objeto o superficie mediante la medición de la luz reflejada o transmitida (SICK AG, 2022)	Es un sensor óptico que capta la intensidad de las longitudes de onda correspondientes a los colores primarios (RGB) de un objeto en movimiento dentro de una línea de producción.	Precisión de detección Velocidad de respuesta	Falsos positivos Falsos negativos Detecciones correctas	Reporte de diseño

III. METODOLOGÍA EMPLEADA

3.1. Tipo y nivel de investigación

Tipo de investigación

Aplicada: A partir de conocimientos establecidos se dará solución a un problema planteado

Nivel de investigación

Explicativo

3.2. Población y muestra de estudio

3.2.1. Población

Es definida por Hernández y Mendoza (2018) como el conjunto de individuos, acciones, procesos u otras características que permiten representarlo en un conjunto que las cuales las asocian a una unidad de estudio. La población de estudio será la gama de colores

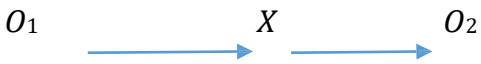
3.2.2. Muestra

La muestra de estudio será la selección de tres colores (rojo, verde y azul)

3.3. Diseño de investigación

La medición es de enfoque cuantitativo, por lo que Azuero (2019), indica que la medición numérica obtenida se procesa en un análisis estadístico e inferencial para validar las hipótesis de estudio.

El diseño de contrastación es no experimental, que según Hernández y Mendoza (2018) indica que este tipo de investigación de corte transversal se fundamenta en la medición de la variable en un tiempo establecido.



Leyenda:

$\boldsymbol{O_1}$**:** Colores

$\boldsymbol{X}$: Red neuronal

$\boldsymbol{O_2}$: Detección de colores

3.4. Técnicas e instrumentos de investigación

Descripción de las técnicas e instrumentos

Para el estudio se requerirá de técnicas que permitan la recolección de la información, las técnicas que se utilizaran se definen de la siguiente manera:

Observación: Se utiliza para recopilar información y datos de manera confiable ya que interactúa con los sentidos para un registro sistemático, permitiendo validar las respuestas (Azuero, 2019). Como resultado, el enfoque se utilizará para recopilar información sobre los colores a detectar.

Análisis documental: Según Azuero (2019), indica que es una investigación en medios digitales, artículos o libros sobre la actividad investigadora planificada; en este sentido, la presente investigación recogerá información sobre los colores a detectar.

Asimismo, para la recolección de la información necesaria para la presente investigación, a continuación, se detallan los instrumentos de recolección de datos:

Guía de análisis documental: Según Hernández y Mendoza (2018) mencionan que las guías de análisis documental son factibles en la captación de las especificaciones necesarias para el trabajo de investigación utilizando el enfoque de análisis documental, por lo que, la presente investigación utilizará registros de información para la selección de los detectores de color.

3.5. Procesamiento y análisis de datos

A continuación, se precisa los procedimientos a realizar en la investigación a fin de cumplir con los objetivos trazados.

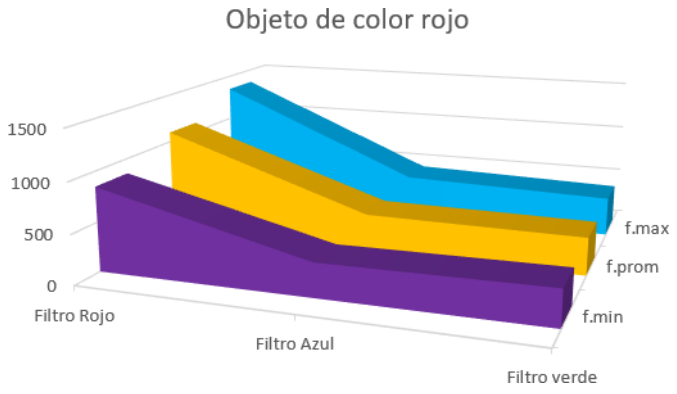
Recolección de datos del sensor (ambiente controlado)

Tabla 2
Recolección de datos color rojo

	Objeto de color rojo			
	f.min	<f.prom<	f.max	
Filtro R	851	1100.5	1350	Hz
Filtro A	340	397.5	455	
Filtro V	360	380	400	

Nota: Recolección de datos para el objeto de color rojo medidos en frecuencia en función de los filtros rojo, azul y verde. Se aprecia para el filtro rojo una frecuencia mínima de 851Hz y una máxima de 1.350KHz, entonces el promedio de ambos es de 1.1005KHz. Para el filtro azul se observa una frecuencia mínima de 340Hz y una máxima de 455Hz, entonces el promedio de ambos es de 397.5Hz. Para el filtro verde se observa una frecuencia mínima de 360Hz y una máxima de 400Hz, entonces el promedio de ambos es de 380Hz.

Figura 4
Comportamiento de la frecuencia - color rojo



Nota: Comportamiento de la frecuencia máxima, promedio y mínima en función de los filtros aplicados para el objeto de color rojo. Para este caso se aprecia como la frecuencia mínima, máxima y promedio tienen mayor valor de frecuencia para el filtro rojo, corroborando asi el propósito del filtrado correspondiente.

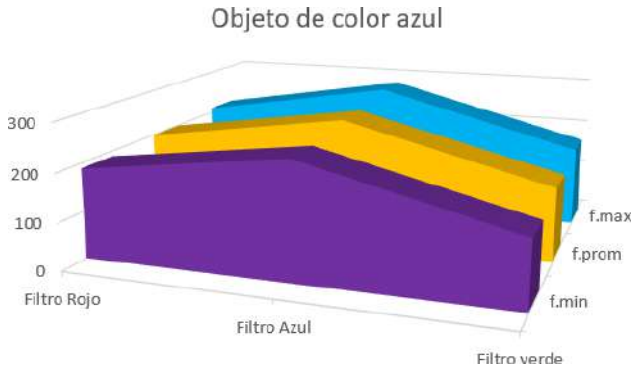
Tabla 3
Recolección de datos color azul

	Objeto de color azul			
	f.min	<f.prom<	f.max	
Filtro R	189	199.5	210	Hz
Filtro A	245	263.5	282	
Filtro V	139	154.5	170	

Nota: Recolección de datos para el objeto de color azul medidos en frecuencia en función de los filtros rojo, azul y verde. Se aprecia para el filtro rojo una frecuencia mínima de 189Hz y una máxima de 210Hz, entonces el promedio de ambos es de 199.5Hz. Para el filtro azul se observa una frecuencia mínima de 245Hz y una máxima de 282Hz, entonces el promedio de ambos es de 263.5Hz. Para el filtro verde se observa una

frecuencia mínima de 139Hz y una máxima de 170Hz, entonces el promedio de ambos es de 154.5Hz.

Figura 5
Comportamiento de la frecuencia - color azul



Nota: Comportamiento de la frecuencia máxima, promedio y mínima en función de los filtros aplicados para el objeto de color azul. Para este caso se aprecia como la frecuencia mínima, máxima y promedio tienen mayor valor de frecuencia para el filtro azul, corroborando así el propósito del filtrado correspondiente.

Tabla 4
Recolección de datos color verde

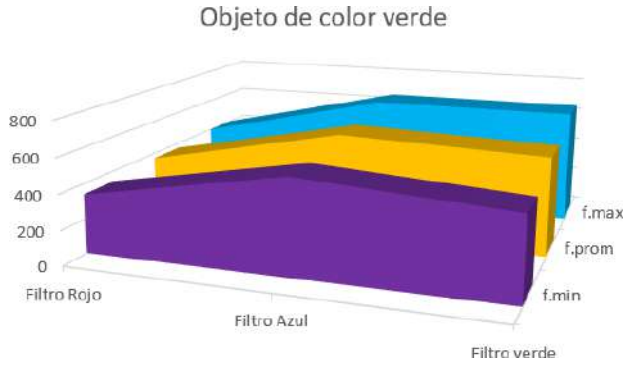
	Objeto de color verde			
	f.min	<f.prom<	f.max	
Filtro R	340	370	400	Hz
Filtro A	545	603.5	662	
Filtro V	470	565	660	

Nota: Recolección de datos para el objeto de color verde medidos en frecuencia en función de los filtros rojo, azul y verde. Se aprecia para el filtro rojo una frecuencia mínima de 340Hz y una máxima de 400Hz, entonces el promedio de ambos es de 370Hz. Para

el filtro azul se observa una frecuencia mínima de 545Hz y una máxima de 662Hz, entonces el promedio de ambos es de 603.5Hz. Para el filtro verde se observa una frecuencia mínima de 470Hz y una máxima de 600Hz, entonces el promedio de ambos es de 565Hz.

Figura 6

Comportamiento de la frecuencia - color verde



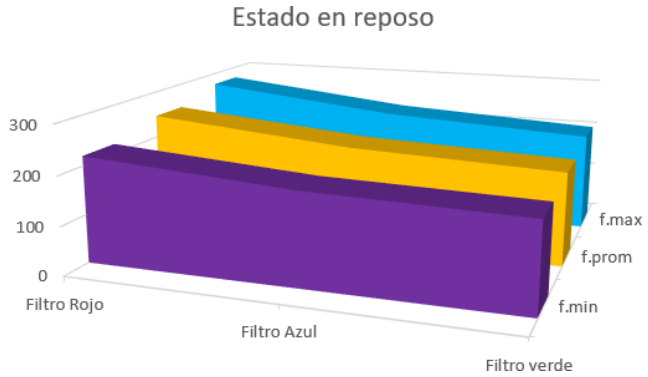
Nota: Comportamiento de la frecuencia máxima, promedio y mínima en función de los filtros aplicados para el objeto de color verde. Para este caso se aprecia como la frecuencia mínima, máxima y promedio tienen mayor valor de frecuencia para el filtro azul y verde, teniendo diferente sistema de trabajo de filtraje en comparación a los anteriores filtros.

Tabla 5
Recolección de datos sin color

	Estado en reposo			
	f.min	<f.prom<	f.max	
Filtro R	220	247	274	Hz
Filtro A	190	209	228	
Filtro V	180	192.5	205	

Nota: Recolección de datos para el estado en reposo sin color medidos en frecuencia en función de los filtros rojo, azul y verde. Se aprecia para el filtro rojo una frecuencia mínima de 220Hz y una máxima de 274Hz, entonces el promedio de ambos es de 247Hz. Para el filtro azul se observa una frecuencia mínima de 190Hz y una máxima de 228Hz, entonces el promedio de ambos es de 209Hz. Para el filtro verde se observa una frecuencia mínima de 180Hz y una máxima de 205Hz, entonces el promedio de ambos es de 192.5Hz.

Figura 7
Comportamiento de la frecuencia - sin color



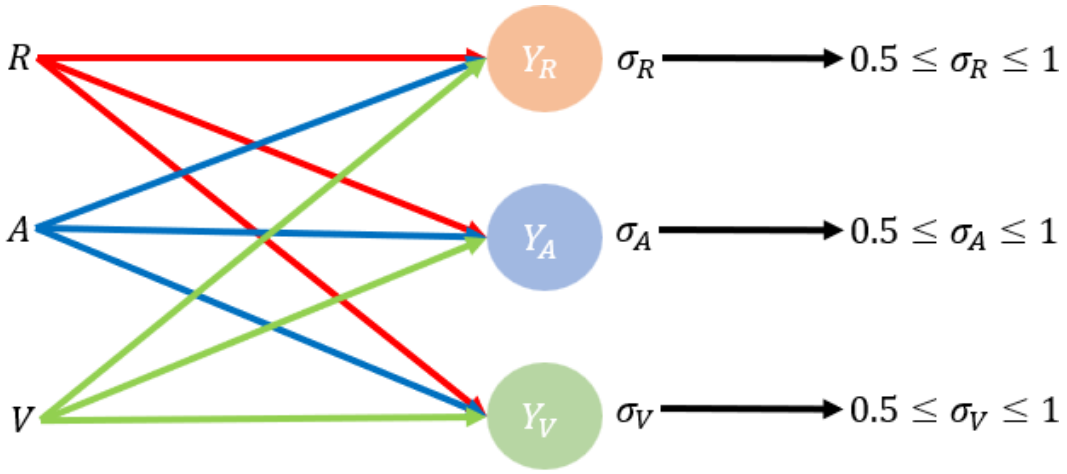
Nota: Comportamiento de la frecuencia máxima, promedio y mínima en función de los filtros aplicados para el estado en reposo sin color. Para este caso se aprecia como la

frecuencia mínima, máxima y promedio tienden a valores iguales de frecuencia para todos los filtros aplicados, corroborando asi el propósito de los filtrados correspondientes.

Leyenda:

- f.min: frecuencia en Hercios mínima de trabajo del objeto
- f.max: frecuencia en Hercios máxima de trabajo del objeto
- f.prom: frecuencia promedio de la frecuencia máxima y mínima
- Filtro R: filtro de color rojo del sensor
- Filtro A: filtro de color azul del sensor
- Filtro V: filtro de color verde del sensor

Figura 8
Diseño de la red neuronal



- $Y_R = R * w_{R1} + A * w_{R2} + \mathrm{V} * w_{R3} + b_R$
- $Y_A = R * w_{A1} + A * w_{A2} + \mathrm{V} * w_{A3} + b_A$
- $Y_V = R * w_{V1} + A * w_{V2} + \mathrm{V} * w_{V3} + b_V$

Nota: El diseño de clasificación de colores consta de la capa de entrada y salida, donde sus entradas están directamente conectadas a la capa de salida. Ilustrado en la figura, las entradas "R", "A" y "V" están en conjunto conectadas a cada neurona, teniendo en

cuenta los pesos "w" y lo sesgos "b" correspondientes. La función de activación para cada neurona es de tipo sigmoide, con su umbral de activación independiente al color.

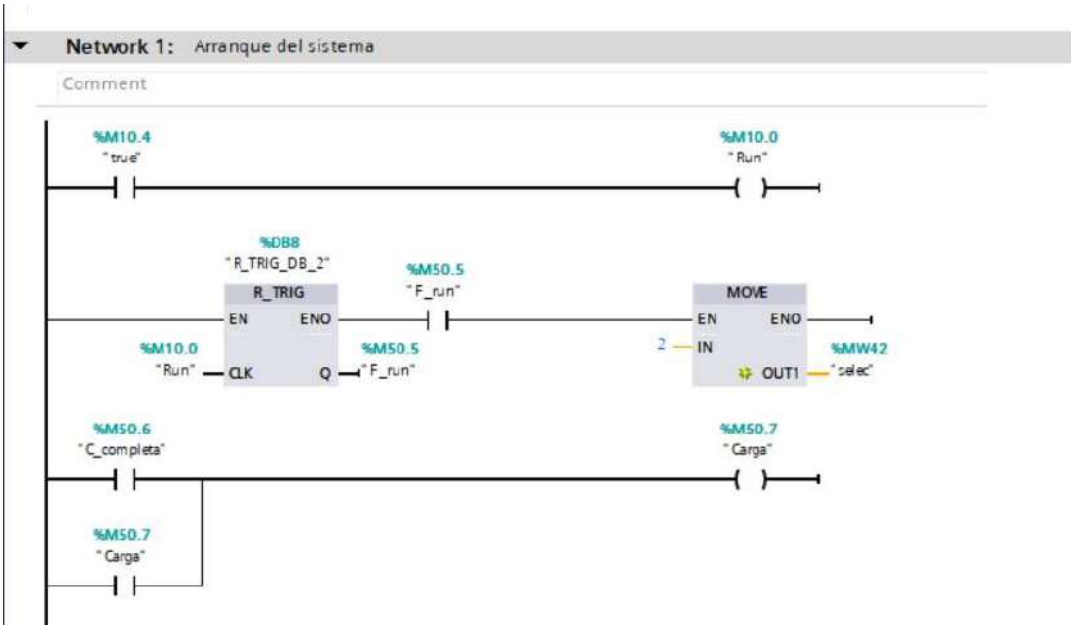
$$\boldsymbol{Función\ de\ activación\ \sigma} = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e}^{-\boldsymbol{y}}}$$

Programación de bloques del PLC

Arranque del sistema:

Figura 9

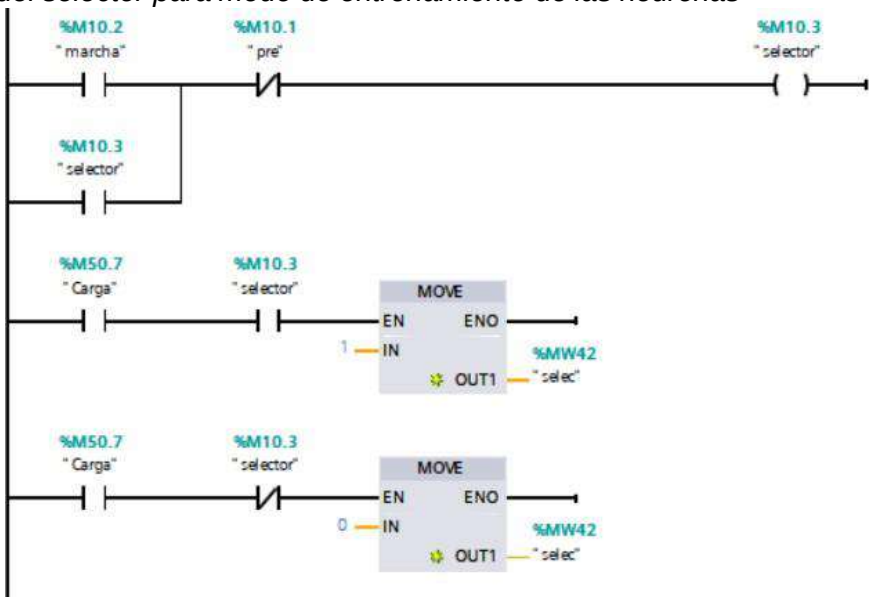
Arranque del sistema



Nota: Se aprecia el algoritmo de arranque del sistema. Este sistema inicia con "Run" controlado y monitoreado desde el HMI, causando la carga de valor "2" para la variable "selec" causando la carga de pesos de las neuronas presentadas en la interrupción cíclica. Y cuando la carga se complete, la variable "Carga" se pone en valor alto.

Figura 10

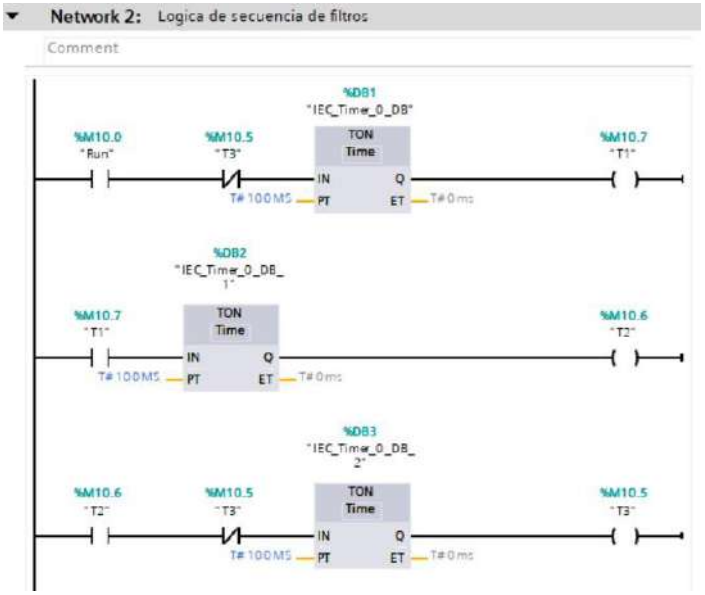
Configuración del selector para modo de entrenamiento de las neuronas



Nota: Mientras "selector" se mantenga en valor bajo, se carga "0" a "selec" que es el modo de entrenamiento de las neuronas. Y si el "selector" se mantiene en valor alto, se carga "1" a "selec" que es el modo de puesta en marcha de las neuronas.

Figura 11

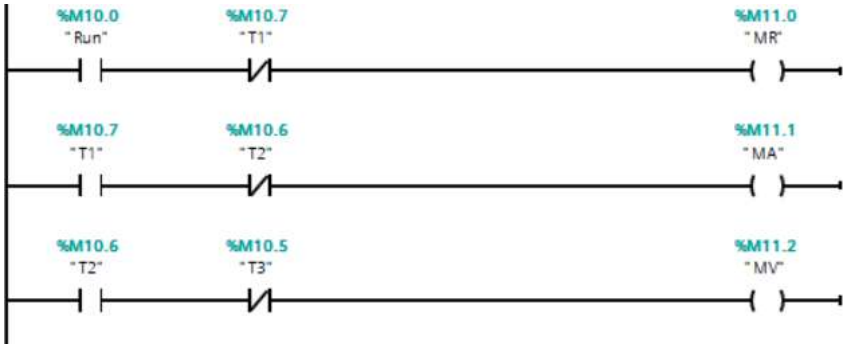
Lógica de secuencia de filtros



Nota: Se aprecia el sistema de lógica de secuencia para el intercambio de filtros, dentro de este esquema, se trabaja con temporizadores guardados en variables T1, T2 y T3.

Figura 12

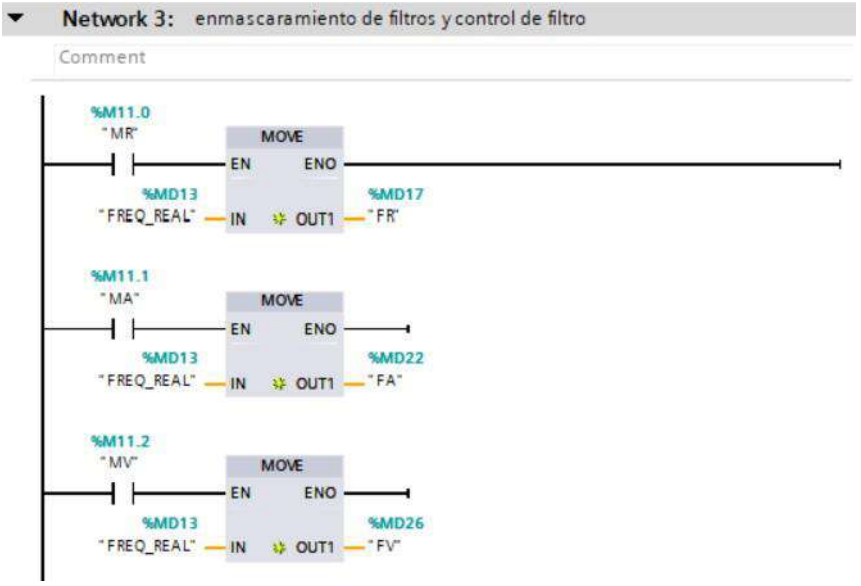
Secuencia de intercambio de filtros



Nota: para la secuencia de intercambio de los tres filtros, cada ecuación estará en función de los temporizadores antes mencionados. La secuencia comienza con el filtro rojo "MR", luego con el filtro azul "MA" y finalmente con el filtro verde "MV" y se repite en forma de bucle.

Figura 13

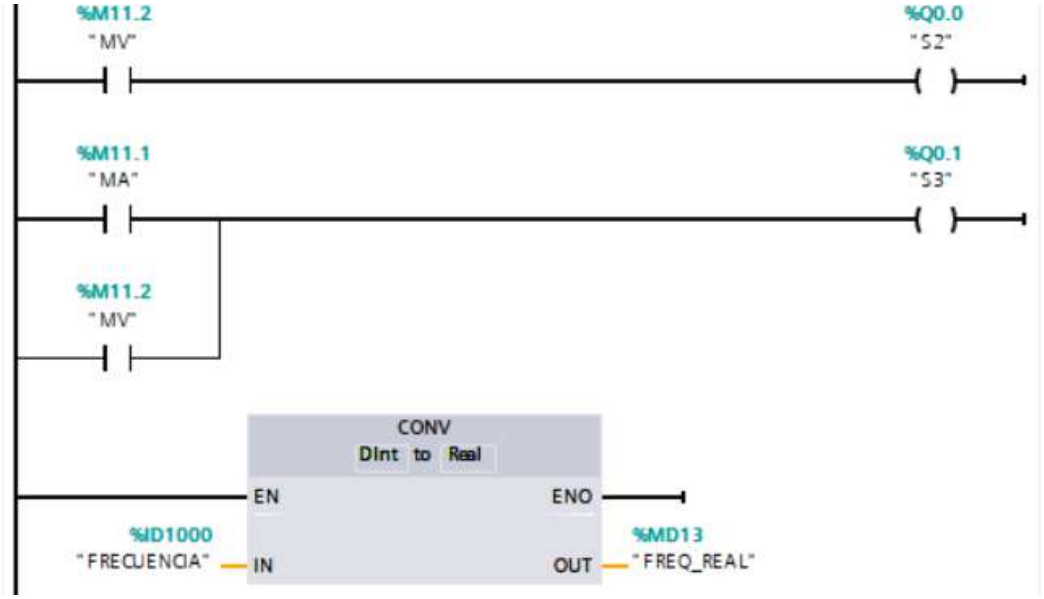
Enmascaramiento de filtros y control de filtro



Nota: La parte de enmascaramiento sigue la misma secuencia de intercambio de filtros. Cuando se activa cada filtro correspondiente, el valor de la frecuencia medida del PLC se mueve a la respectiva variable indicada en los bloques MOVE de la salida OUT1, y estas variables son guardadas en "FR", "FA" y "FV" individualmente.

Figura 14

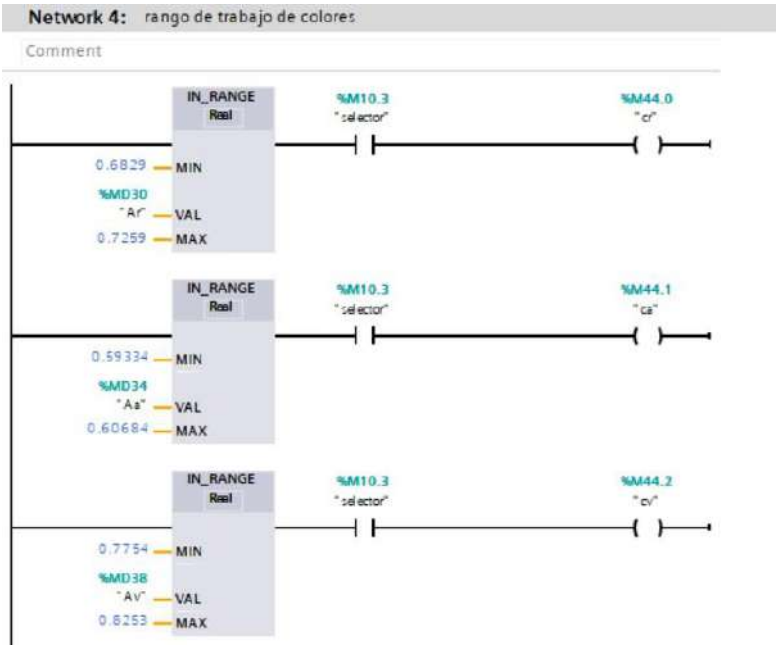
Salidas físicas en función de la activación



Nota: Se visualiza las salidas físicas S2 y S3 en función de la activación de los filtros. Y a la vez se convierte el valor de la frecuencia medida del PLC "FRECUENCIA" a un valor de tipo real almacenada en "FREQ_REAL".

Figura 15

Rango de trabajo de colores

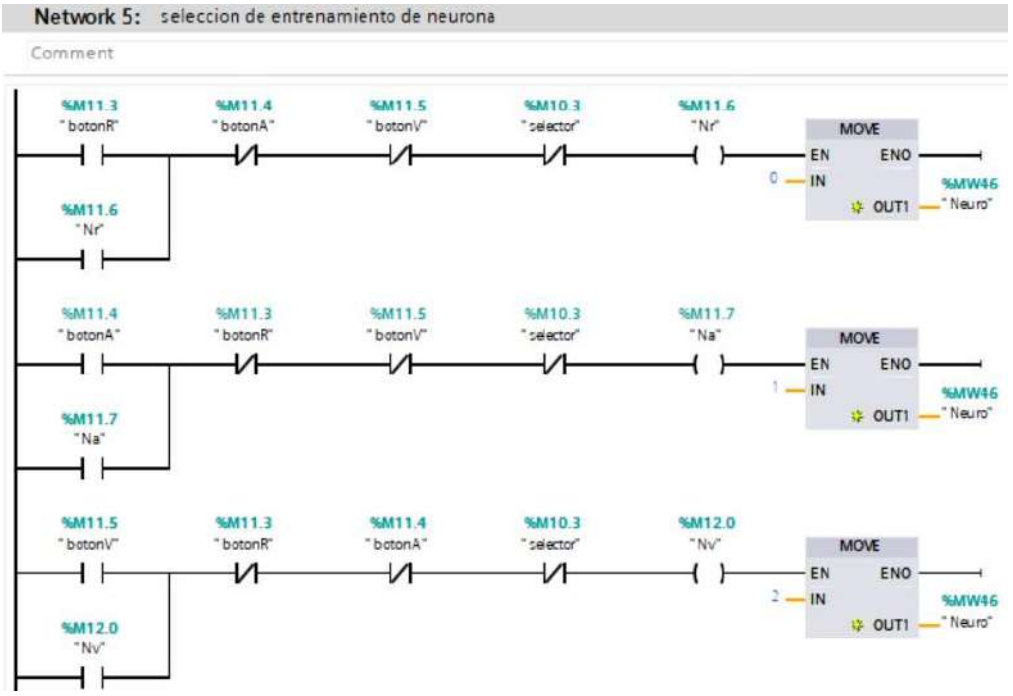


Nota: Se visualiza el rango de los valores de las funciones de activación de cada neurona, donde si el valor de la función de activación está dentro del rango establecido, se activa la variable correspondiente conectada al bloque, estas salidas son "cr", "ca" y "cv" que indican el estado del color de tipo booleano.

:

Figura 16

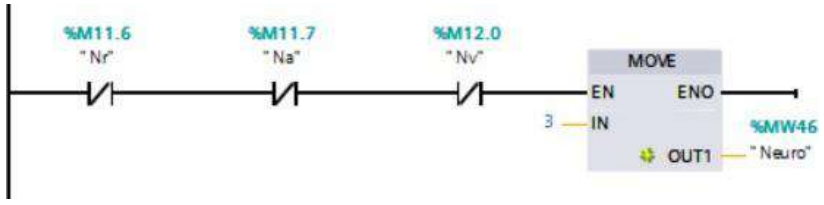
Selección de entrenamiento de neurona



Nota: Para la selección de neurona a entrenar, cada neurona tendrá un valor único donde estará almacenada en "Neuro", y ese mismo valor se almacenará a partir de las funciones de las salidas "Nr", "Na" y "Nv".

Figura 17

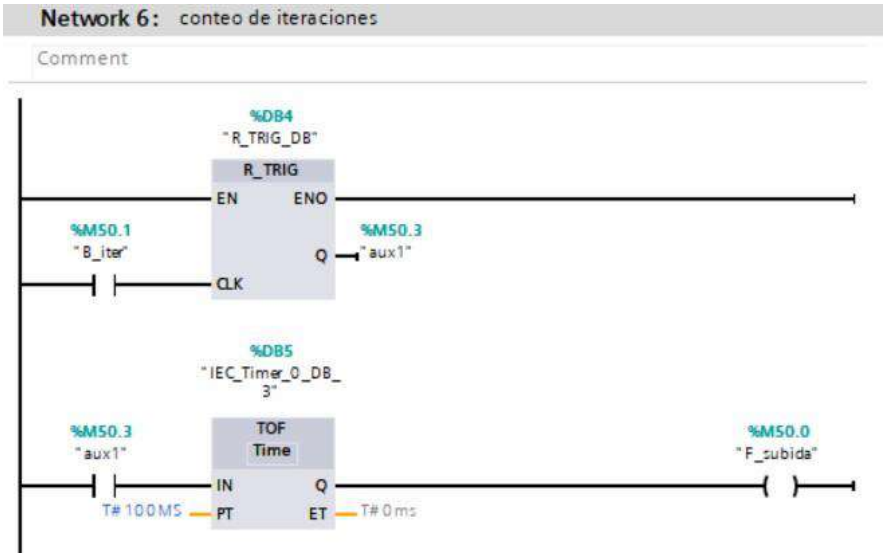
Bloque Move para neurona



Nota: Mientras ninguna neurona esté activa para entrenar, el valor de "3" se cargará a "Neuro".

Figura 18

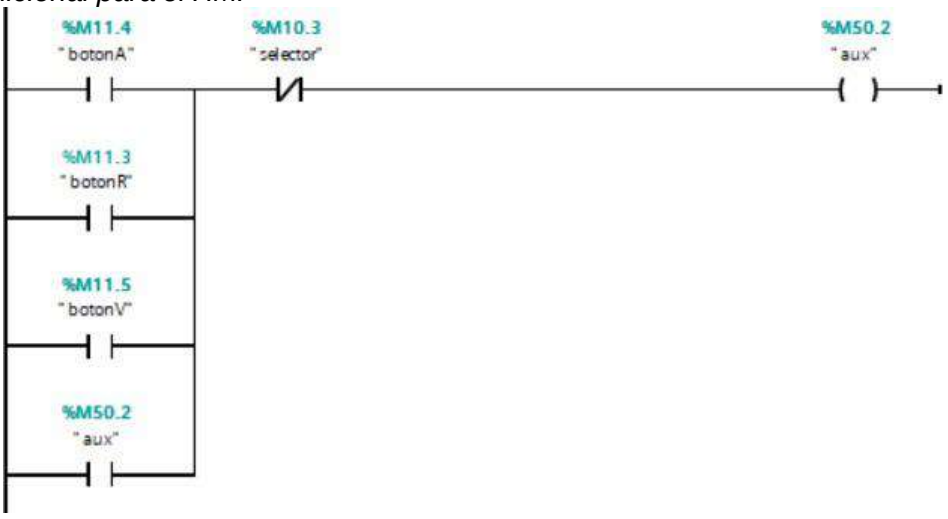
Conteo de iteraciones



Nota: Para censar si el botón "B_iter" fue presionado desde el HMI para la interrupción cíclica, este mismo se lleva a un bloque de flanco de subida, y luego se lleva a un temporizador TOF para mantener ese estado del botón por 100ms donde finalmente se lleva a la salida "F_subida".

Figura 19

Algoritmo adicional para el HMI



Nota: Algoritmo adicional para el HMI, consiste en la aparición de bloques como el ingreso de iteraciones máximas y visualización de la iteración actual del HMI a partir del enclavamiento de la variable "aux".

Programación de interrupción cíclica del PLC:

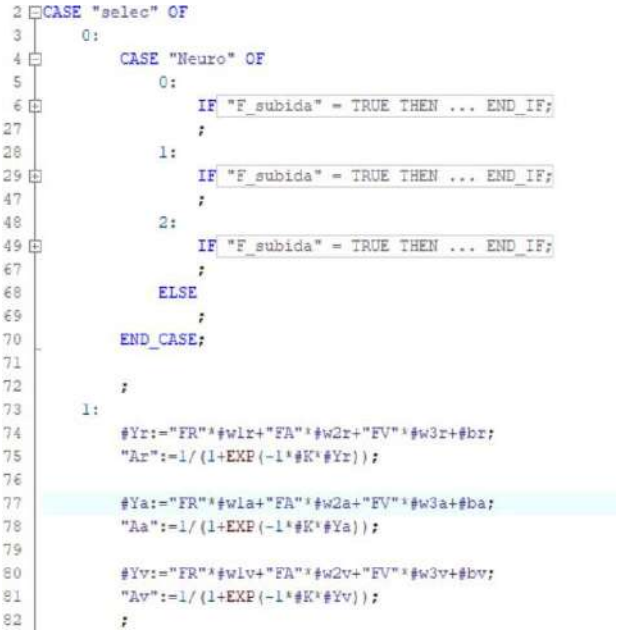
Figura 20

Programación de la neuronal

```
83      ELSE
84          #w1r := 4.6410E-5;
85          #w2r := 2.8070E-4;
86          #w3r := 3.3562E-4;
87          #br := 9.1772E-7;
88          #w1a := 2.4654E-4;
89          #w2a := 2.3350E-4;
90          #w3a := 1.5919E-4;
91          #ba := 1.4853E-6;
92          #w1v := 2.4466E-4;
93          #w2v := 3.3376E-4;
94          #w3v := 3.1490E-4;
95          #bv := 7.3944E-7;
96          #K := 3;
97          #B := 0.000001;
98          "R" := 0;
99          "C_completa" := TRUE;
100         ;
101 END_CASE;
```

Nota: Cuando el sistema arranca con el "START" del HMI, hace que el sistema cargue previamente los pesos y sesgos para cada neurona correspondiente, y cuando termine, envía un "TRUE" para indicar que se terminó la carga de datos.

Figura 21
Modo entrenamiento



```
CASE "selec" OF
    0:
        CASE "Neuro" OF
            0:
                IF "F_subida" = TRUE THEN ... END_IF;
                ;
            1:
                IF "F_subida" = TRUE THEN ... END_IF;
                ;
            2:
                IF "F_subida" = TRUE THEN ... END_IF;
                ;
            ELSE
                ;
        END_CASE;

        ;
    1:
        #Yr:="FR"*#w1r+"FA"*#w2r+"FV"*#w3r+#br;
        "Ar":=1/(1+EXP(-1*#K*#Yr));

        #Ya:="FR"*#w1a+"FA"*#w2a+"FV"*#w3a+#ba;
        "Aa":=1/(1+EXP(-1*#K*#Ya));

        #Yv:="FR"*#w1v+"FA"*#w2v+"FV"*#w3v+#bv;
        "Av":=1/(1+EXP(-1*#K*#Yv));
        ;
```

Nota: Este sistema general se basa en casos, por lo tanto, si “selec” es “0” el sistema entrara en modo entrenamiento. Y si “selec” es “1” el sistema entrara en modo puesta en marcha, donde los pesos ya entrenados serán netamente constantes en las ecuaciones de las salidas “Yr”, “Ya” y “Yv” (Yr: Salida de color rojo; Ya: Salida de color azul; Yv: Salida de color verde) al igual que las funciones de activación “Ar”, “Aa” y “Av” (Ar: Activación de color rojo; Aa: Activación de color azul; Av: Activación de color verde).

Figura 22

Entrenamiento para neurona roja

```
CASE "selec" OF
    0:
        CASE "Neuro" OF
            0:
                IF "F_subida" = TRUE THEN
                    #Yr := "FR" * #w1r + "FA" * #w2r + "FV" * #w3r + #br;
                    "Ar" := 1 / (1 + EXP(-1 * #K * #Yr));
                    #error := 0.7 - "Ar";
                    #w1r := #w1r + #B * #error * "FR" * "Ar" * #K * (1 - "Ar");
                    "FR_ANTERIOR" := "FR";
                    #w2r := #w2r + #B * #error * "FA" * "Ar" * #K * (1 - "Ar");
                    "FA_ANTERIOR" := "FA";
                    #w3r := #w3r + #B * #error * "FV" * "Ar" * #K * (1 - "Ar");
                    "FV_ANTERIOR" := "FV";
                    #br := #br + #B * #error * "Ar" * #K * (1 - "Ar");
                    "R":= "R" + 1;
                    "FR_ANTERIOR" := "FR";
                    "FA_ANTERIOR" := "FA";
                    "FV_ANTERIOR" := "FV";
                    IF "R" = "i_maxR" THEN
                        "R" := 0;
                        ;
                    END_IF;
                    ;
                END_IF;
                ;
```

Nota: Para el entrenamiento de la neurona roja, la variable “Neuro” debe valer “0”. Para que este entrenamiento se lleve a cabo, el flanco de subida debe ser verdadero. Dentro de esta condición, se suma a la variable “R” la unidad para indicar la iteración actual de entrenamiento, y si este es igual a la iteración máxima establecida en el HMI, la variable “R” se reinicia a cero.

Figura 23

Entrenamiento para neurona azul

```
1:
    IF "F_subida" = TRUE THEN
        #Ya := "FR" * #w1a + "FA" * #w2a + "FV" * #w3a + #ba;
        "Aa" := 1 / (1 + EXP(-1 * #K * #Ya));
        #error := 0.6 - "Aa";
        #w1a := #w1a + #B * #error * "FR" * "Aa" * #K * (1 - "Aa");
        #w2a := #w2a + #B * #error * "FA" * "Aa" * #K * (1 - "Aa");
        #w3a := #w3a + #B * #error * "FV" * "Aa" * #K * (1 - "Aa");
        #ba := #ba + #B * #error * "Aa" * #K * (1 - "Aa");
        "A" := "A" + 1;
        "FR_ANTERIOR" := "FR";
        "FA_ANTERIOR" := "FA";
        "FV_ANTERIOR" := "FV";
        IF "A" = "i_maxA" THEN
            "A" := 0;
            ;
        END_IF;
        ;
    END_IF;
    ;
```

Nota: Para el entrenamiento de la neurona azul, la variable "Neuro" debe valer "1". Para que este entrenamiento se lleve a cabo, el flanco de subida debe ser verdadero. Dentro de esta condición, se suma a la variable "A" la unidad para la indicar la iteración actual de entrenamiento, y si este es igual a la iteración máxima establecida en el HMI, la variable "A" se reinicia a cero.

Figura 24

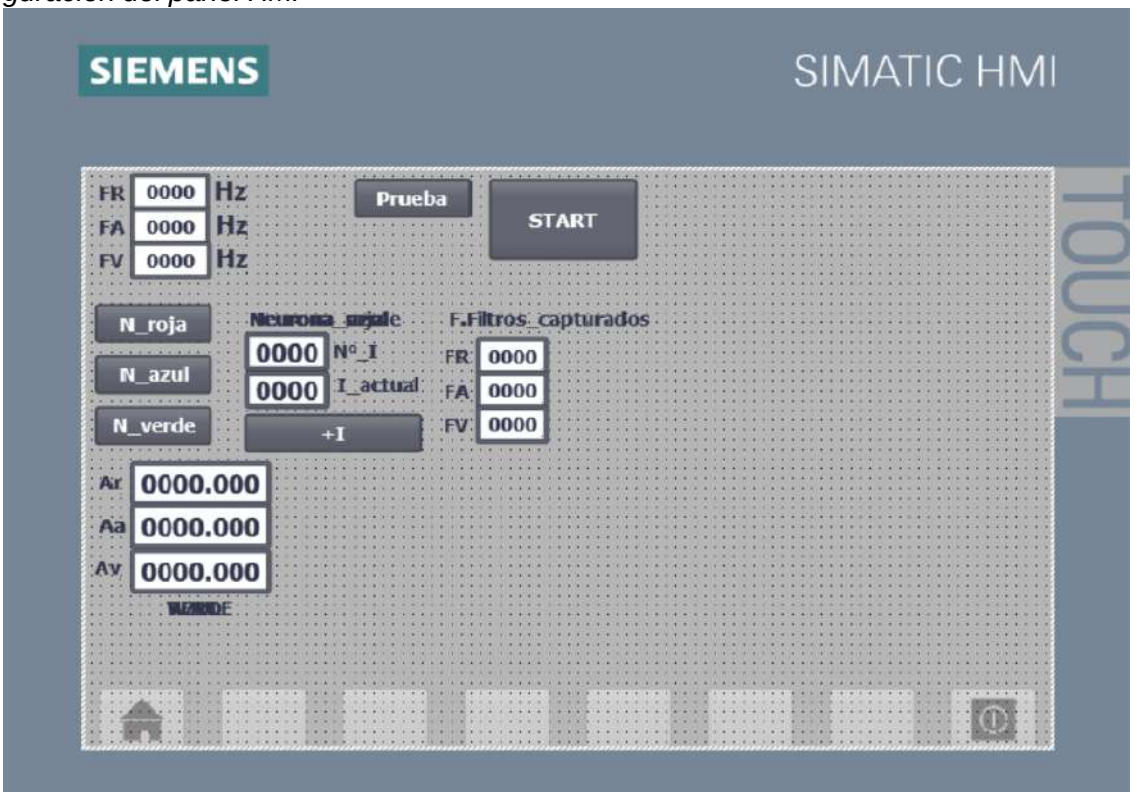
Entrenamiento para neurona verde

```
2:
    IF "F_subida" = TRUE THEN
        #Yv := "FR" * #w1v + "FA" * #w2v + "FV" * #w3v + #bv;
        "Av" := 1 / (1 + EXP(-1 * #K * #Yv));
        #error := 0.8 - "Av";
        #w1v := #w1v + #B * #error * "FR" * "Av" * #K * (1 - "Av");
        #w2v := #w2v + #B * #error * "FA" * "Av" * #K * (1 - "Av");
        #w3v := #w3v + #B * #error * "FV" * "Av" * #K * (1 - "Av");
        #bv := #bv + #B * #error * "Av" * #K * (1 - "Av");
        "V" := "V" + 1;
        "FR_ANTERIOR" := "FR";
        "FA_ANTERIOR" := "FA";
        "FV_ANTERIOR" := "FV";
        IF "V" = "i_maxV" THEN
            "V" := 0;
            ;
        END_IF;
        ;
    END_IF;
    ;
```

Nota: Para el entrenamiento de la neurona verde, la variable "Neuro" debe valer "2". Para que este entrenamiento se lleve a cabo, el flanco de subida debe ser verdadero. Dentro de esta condición, se suma a la variable "V" la unidad para la indicar la iteración actual de entrenamiento, y si este es igual a la iteración máxima establecida en el HMI, la variable "V" se reinicia a cero.

Figura 25

Configuración del panel HMI

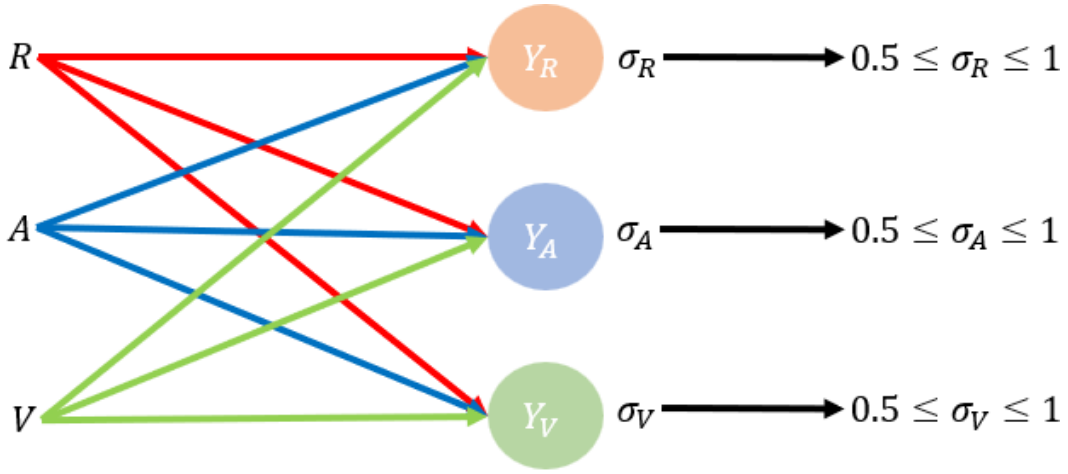


Nota: En la parte superior del HMI se aprecia la visualización de las frecuencias con su respectivo filtro (FR: filtro rojo; FA: filtro azul; FV: filtro verde), también el botón de "Prueba" que esta superpuesto al botón de puesta en marcha junto al botón de arranque "START". En la sección del medio, la parte izquierda se aprecia los botones de cada neurona (N_roja: Neurona roja; N_azul: Neurona azul; N_verde: Neurona verde), en la parte derecha restante se aprecia el ingreso de datos de la iteración máxima "Nº_I" y la visualización numérica de la iteración actual "I_actual" y el botón de incremento de la iteración actual "+I acompañado en la parte lateral de las frecuencias capturadas con sus respectivos filtros. Y en la parte inferior se encuentra las funciones de activación "Ar", "Aa" y "Av" como visualizadores numéricos.

IV. PRESENTACIÓN DE RESULTADOS

Diseño de la red neuronal

Figura 26
Diseño de la red neuronal



- $Y_R = R * w_{R1} + A * w_{R2} + V * w_{R3} + b_R$
- $Y_A = R * w_{A1} + A * w_{A2} + V * w_{A3} + b_A$
- $Y_V = R * w_{V1} + A * w_{V2} + V * w_{V3} + b_V$

Nota: El diseño de clasificación de colores consta de la capa de entrada y salida, donde sus entradas están directamente conectadas a la capa de salida. Ilustrado en la figura, las entradas "R", "A" y "V" están en conjunto conectadas a cada neurona, teniendo en cuenta los pesos "w" y lo sesgos "b" correspondientes. La función de activación para cada neurona es de tipo sigmoide, con su umbral de activación independiente al color.

$$\textbf{Función de activación } \sigma = \frac{1}{1 + e^{-y}}$$

Implementación de la red neuronal en el PLC s7-1200

Figura 27

Programación de la neurona

```
    ELSE
        #w1r := 4.6410E-5;
        #w2r := 2.8070E-4;
        #w3r := 3.3562E-4;
        #br := 9.1772E-7;
        #w1a := 2.4654E-4;
        #w2a := 2.3350E-4;
        #w3a := 1.5919E-4;
        #ba := 1.4853E-6;
        #w1v := 2.4466E-4;
        #w2v := 3.3376E-4;
        #w3v := 3.1490E-4;
        #bv := 7.3944E-7;
        #K := 3;
        #B := 0.000001;
        "R" := 0;
        "C_completa" := TRUE;
        ;
END_CASE;
```

Nota: Cuando el sistema arranca con el "START" del HMI, hace que el sistema cargue previamente los pesos y sesgos para cada neurona correspondiente, y cuando termine, envía un "TRUE" para indicar que se terminó la carga de datos.

Figura 28

Modo entrenamiento

```
CASE "selec" OF
    0:
        CASE "Neuro" OF
            0:
                IF "F_subida" = TRUE THEN ... END_IF;
                ;
            1:
                IF "F_subida" = TRUE THEN ... END_IF;
                ;
            2:
                IF "F_subida" = TRUE THEN ... END_IF;
                ;
            ELSE
                ;
        END_CASE;

        ;
    1:
        #Yr:="FR"*#w1r+"FA"*#w2r+"FV"*#w3r+#br;
        "Ar":=1/(1+EXP(-1*#K*#Yr));

        #Ya:="FR"*#w1a+"FA"*#w2a+"FV"*#w3a+#ba;
        "Aa":=1/(1+EXP(-1*#K*#Ya));

        #Yv:="FR"*#w1v+"FA"*#w2v+"FV"*#w3v+#bv;
        "Av":=1/(1+EXP(-1*#K*#Yv));
        ;
```

Nota: Este sistema general se basa en casos, por lo tanto, si "selec" es "0" el sistema entrara en modo entrenamiento. Y si "selec" es "1" el sistema entrara en modo puesta en marcha, donde los pesos ya entrenados serán netamente constantes en las ecuaciones de las salidas "Yr", "Ya" y "Yv" (Yr: Salida de color rojo; Ya: Salida de color azul; Yv: Salida de color verde) al igual que las funciones de activación "Ar", "Aa" y "Av" (Ar: Activación de color rojo; Aa: Activación de color azul; Av: Activación de color verde).

Figura 29

Entrenamiento para neurona roja

```
CASE "selec" OF
    0:
        CASE "Neuro" OF
            0:
                IF "F_subida" = TRUE THEN
                    #Yr := "FR" * #w1r + "FA" * #w2r + "FV" * #w3r + #br;
                    "Ar" := 1 / (1 + EXP(-1 * #K * #Yr));
                    #error := 0.7 - "Ar";
                    #w1r := #w1r + #B * #error * "FR" * "Ar" * #K * (1 - "Ar");
                    "FR_ANTERIOR" := "FR";
                    #w2r := #w2r + #B * #error * "FA" * "Ar" * #K * (1 - "Ar");
                    "FA_ANTERIOR" := "FA";
                    #w3r := #w3r + #B * #error * "FV" * "Ar" * #K * (1 - "Ar");
                    "FV_ANTERIOR" := "FV";
                    #br := #br + #B * #error * "Ar" * #K * (1 - "Ar");
                    "R":= "R" + 1;
                    "FR_ANTERIOR" := "FR";
                    "FA_ANTERIOR" := "FA";
                    "FV_ANTERIOR" := "FV";
                    IF "R" = "i_maxR" THEN
                        "R" := 0;
                        ;
                    END_IF;
                    ;
                END_IF;
                ;
```

Nota: Para el entrenamiento de la neurona roja, la variable "Neuro" debe valer "0". Para que este entrenamiento se lleve a cabo, el flanco de subida debe ser verdadero. Dentro de esta condición, se suma a la variable "R" la unidad para indicar la iteración actual de entrenamiento, y si este es igual a la iteración máxima establecida en el HMI, la variable "R" se reinicia a cero.

Figura 30

Entrenamiento para neurona azul

```
1:
    IF "F_subida" = TRUE THEN
        #Ya := "FR" * #w1a + "FA" * #w2a + "FV" * #w3a + #ba;
        "Aa" := 1 / (1 + EXP(-1 * #K * #Ya));
        #error := 0.6 - "Aa";
        #w1a := #w1a + #B * #error * "FR" * "Aa" * #K * (1 - "Aa");
        #w2a := #w2a + #B * #error * "FA" * "Aa" * #K * (1 - "Aa");
        #w3a := #w3a + #B * #error * "FV" * "Aa" * #K * (1 - "Aa");
        #ba := #ba + #B * #error * "Aa" * #K * (1 - "Aa");
        "A" := "A" + 1;
        "FR_ANTERIOR" := "FR";
        "FA_ANTERIOR" := "FA";
        "FV_ANTERIOR" := "FV";
        IF "A" = "i_maxA" THEN
            "A" := 0;
            ;
        END_IF;
        ;
    END_IF;
    ;
```

Nota: Para el entrenamiento de la neurona azul, la variable “Neuro” debe valer “1”. Para que este entrenamiento se lleve a cabo, el flanco de subida debe ser verdadero. Dentro de esta condición, se suma a la variable “A” la unidad para la indicar la iteración actual de entrenamiento, y si este es igual a la iteración máxima establecida en el HMI, la variable “A” se reinicia a cero.

Figura 31

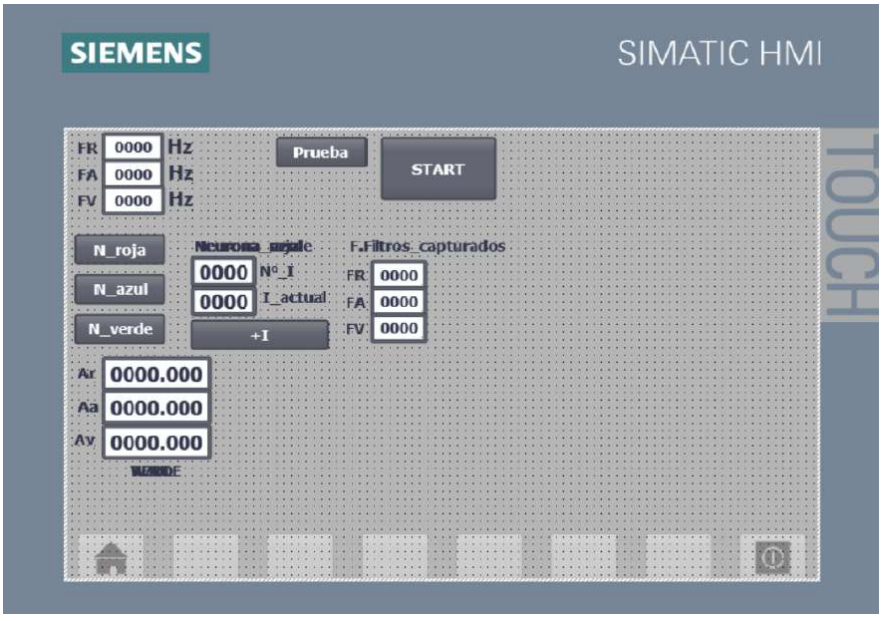
Entrenamiento para neurona verde

```
2:
    IF "F_subida" = TRUE THEN
        #Yv := "FR" * #w1v + "FA" * #w2v + "FV" * #w3v + #bv;
        "Av" := 1 / (1 + EXP(-1 * #K * #Yv));
        #error := 0.8 - "Av";
        #w1v := #w1v + #B * #error * "FR" * "Av" * #K * (1 - "Av");
        #w2v := #w2v + #B * #error * "FA" * "Av" * #K * (1 - "Av");
        #w3v := #w3v + #B * #error * "FV" * "Av" * #K * (1 - "Av");
        #bv := #bv + #B * #error * "Av" * #K * (1 - "Av");
        "V" := "V" + 1;
        "FR_ANTERIOR" := "FR";
        "FA_ANTERIOR" := "FA";
        "FV_ANTERIOR" := "FV";
        IF "V" = "i_maxV" THEN
            "V" := 0;
            ;
        END_IF;
        ;
    END_IF;
    ;
```

Nota: Para el entrenamiento de la neurona verde, la variable “Neuro” debe valer “2”. Para que este entrenamiento se lleve a cabo, el flanco de subida debe ser verdadero. Dentro de esta condición, se suma a la variable “V” la unidad para la indicar la iteración actual de entrenamiento, y si este es igual a la iteración máxima establecida en el HMI, la variable “V” se reinicia a cero.

Figura 32

Configuración del panel HMI



Nota: En la parte superior del HMI se aprecia la visualización de las frecuencias con su respectivo filtro (FR: filtro rojo; FA: filtro azul; FV: filtro verde), también el botón de "Prueba" que esta superpuesto al botón de puesta en marcha junto al botón de arranque "START". En la sección del medio, la parte izquierda se aprecia los botones de cada neurona (N_roja: Neurona roja; N_azul: Neurona azul; N_verde: Neurona verde), en la parte derecha restante se aprecia el ingreso de datos de la iteración máxima "N°_I" y la visualización numérica de la iteración actual "I_actual" y el botón de incremento de la iteración actual "+I acompañado en la parte lateral de las frecuencias capturadas con sus respectivos filtros. Y en la parte inferior se encuentra las funciones de activación "Ar", "Aa" y "Av" como visualizadores numéricos.

Entrenamiento de la red neuronal

Neurona de color rojo: Y_R

Tabla 6

Entrenamiento de neurona color rojo: Y_R

Parámetros de entrenamiento	Valores
w_{R1}	$4.641x10^{-5}$
w_{R2}	$2.807x10^{-4}$
w_{R3}	$3.3562x10^{-4}$
b_R	$9.1772x10^{-7}$
N° de iteraciones	100

Nota: Como se aprecia en la tabla, se registra los 3 pesos y el sesgo ya corregidos para la neurona de detección en objetos de color rojo. Para este caso, el entrenamiento tuvo un máximo de 100 iteraciones o épocas para minimizar el error.

Neurona de color azul: Y_A

Tabla 7

Entrenamiento de neurona color azul: Y_A

Parámetros de entrenamiento	Valores
w_{A1}	$2.4654x10^{-4}$
w_{A2}	$2.335x10^{-4}$
w_{A3}	$1.5919x10^{-4}$
b_A	$1.4853x10^{-6}$
N° de iteraciones	100

Nota: Como se aprecia en la tabla, se registra los 3 pesos y el sesgo ya corregidos para la neurona de detección en objetos de color azul. Para este caso, el entrenamiento tuvo un máximo de 100 iteraciones o épocas para minimizar el error.

Neurona de color verde: Y_V

Tabla 8

Entrenamiento de neurona color verde: Y_V

Parámetros de entrenamiento	Valores
w_{V1}	$2.4466x10^{-4}$
w_{V2}	$3.3376x10^{-4}$
w_{V3}	$3.149x10^{-4}$
b_V	$7.3944x10^{-7}$
N° de iteraciones	100

Nota: Como se aprecia en la tabla, se registra los 3 pesos y el sesgo ya corregidos para la neurona de detección en objetos de color verde. Para este caso, el entrenamiento tuvo un máximo de 100 iteraciones o épocas para minimizar el error.

Determinación de la eficiencia:

Eficiencia del sistema basado en redes neuronales en entornos industriales. (t STUDENT)

Prueba de error:

V: Resultado de estado verdadero

F: Resultado de estado falso

Parámetros de valor de muestra para población infinita

Tabla 9

Parámetros de valor de muestra para población infinita

Parámetros	Valor
Z	1.645
p	0.9
q	0.1
e	0.11
n	20.13

Nota: números de muestras de 20 ítems con un nivel de confianza de 90%

Detección del objeto color rojo:

Tabla 10

Detección de objeto color rojo

N° de muestra	Estado de la salida
1	V
2	V
3	V
4	V

5	V
6	V
7	F
8	V
9	V
10	V
11	V
12	V
13	V
14	F
15	F
16	V
17	V
18	V
19	V
20	V

Nota: Como se aprecia en la tabla, se hace una prueba de 20 muestras para averiguar la proporción de eventos para los resultados de estado verdadero y falso aplicado a un objeto de color rojo. Para este caso, la proporción de resultados falsos es de 0.15%, mientras la proporción de resultados verdaderos es de 0.85%

Detección del objeto color azul:

Tabla 11

Detección de objeto color azul

N° de muestra	Estado de la salida
1	V
2	F
3	V
4	V
5	F
6	F
7	V
8	V
9	V
10	V
11	V
12	V
13	V
14	V
15	V
16	V
17	V
18	V
19	V

20	F

Nota: Como se aprecia en la tabla, se hace una prueba de 20 muestras para averiguar la proporción de eventos para los resultados de estado verdadero y falso aplicado a un objeto de color azul. Para este caso, la proporción de resultados falsos es de 0.2%, mientras la proporción de resultados verdaderos es de 0.8%

Detección del objeto color verde:

Tabla 12

Detección de objeto color verde

N° de muestra	Estado de la salida
1	F
2	F
3	V
4	F
5	V
6	V
7	V
8	V
9	V
10	V
11	V
12	V
13	V

14	V
15	V
16	V
17	V
18	F
19	V
20	V

Nota: Como se aprecia en la tabla, se hace una prueba de 20 muestras para averiguar la proporción de eventos para los resultados de estado verdadero y falso aplicado a un objeto de color verde. Para este caso, la proporción de resultados falsos es de 0.2%, mientras la proporción de resultados verdaderos es de 0.8%

Por lo tanto, la eficiencia promedio es 81%. Teniendo en cuenta la red con 16.33 aciertos de 20 tomas.

V. DISCUSIÓN DE RESULTADOS

El modelo diseñado consistió en una red neuronal de arquitectura simple, conformada por una capa con tres neuronas, configuradas con función de activación. Las entradas corresponden a los valores normalizados de frecuencia generados por el sensor de color, mientras que las salidas representan la activación correspondiente a cada color previamente entrenado (azul, rojo y verde).

Este diseño permitió un balance adecuado entre simplicidad de implementación y capacidad de clasificación**,** características fundamentales en entornos industriales donde los recursos computacionales son limitados. La estructura adoptada mostró un comportamiento estable y coherente, con una capacidad de aprendizaje adecuada para las condiciones del entorno de prueba, logrando identificar correctamente los colores con una precisión aceptable.

Dado que el PLC Siemens S7-1200 no está diseñado para ejecutar modelos de inteligencia artificial de manera nativa, se optó por una implementación directa de la lógica neuronal en el propio controlador, utilizando el lenguaje de programación estructurado SCL (Structured Control Language). El modelo fue codificado dentro de un bloque de interrupción cíclica, con un intervalo de activación de 100 ms, permitiendo una ejecución periódica y determinista del algoritmo de clasificación.

La lectura de datos del sensor de color se integró exitosamente al sistema, permitiendo una comunicación efectiva entre el módulo de entrada analógica y la lógica de control implementada. Esta integración demostró que es posible adaptar el PLC como una plataforma ejecutora de algoritmos de clasificación neuronal, siempre que se utilicen modelos livianos y estructuras lógicas optimizadas.

Durante la etapa de validación, se realizaron 20 pruebas consecutivas de detección de color bajo condiciones estables de iluminación y disposición de los objetos. De estas, 16.33 fueron aciertos promedio, lo que representa una eficiencia del 81%. Este nivel de desempeño se considera aceptable para aplicaciones básicas de clasificación, especialmente si se toma en cuenta la limitación de recursos del PLC y la sencillez de la red neuronal implementada.

El comportamiento observado confirma que la red neuronal fue capaz de generalizar adecuadamente, aunque en algunos casos se detectaron ligeras confusiones cuando los valores de entrada se aproximaban a los límites de decisión entre dos colores.

CONCLUSIONES

Se diseñó la red neuronal basada en una capa con tres neuronas con función de activación SIGMOIDE y con tres entradas y tres salidas. Las cuales determinan los colores de detección basado en las frecuencias de entrada a la red neuronal.

$$\boldsymbol{Función\ de\ activación\ \sigma = \frac{1}{1 + e^{-y}}}$$

Se implementó la red neuronal en un bloque de interrupción cíclica, con un tiempo de interruptor de 100ms. Desarrollada en lenguaje SCL. como de evidencia en el capítulo IV.

Se determinó la eficiencia promedio de la red con 16.33 aciertos de 20 tomas. Lo que permite una eficiencia 81%.

RECOMENDACIONES

Se recomienda realizar pruebas con una red que tenga al menos una capa oculta para determinar si mejora la eficiencia.

VI. REFERENCIAS BIBLIOGRÁFICAS

1. Chávez Huamán, C., & Gonzales Gutiérrez, J. (2021). *Propuesta de implementación de un sistema automático IoT para controlar parámetros de un generador de rayos X convencional mediante Node-RED y PLC S7-1200* [Tesis de licenciatura, Universidad Peruana de Ciencias Aplicadas]. Repositorio UPC.
2. Gonzales del Valle Romero, G. F., & Neyra Espinoza, W. J. (2022). *Implementación de una red neuronal artificial convolucional para la detección y conteo de vehículos en una sección del estacionamiento de la Universidad Ricardo Palma* [Tesis de licenciatura, Universidad Ricardo Palma]. Repositorio URP.
3. Haykin, S. (2009). *Neural Networks and Learning Machines* (3.ª ed.). Pearson Education.
4. Lindao Chavarría, J. A. (2022). *Diseño y análisis de un sistema automatizado basado en redes industriales para el control del proceso de etiquetado y almacenamiento de cajas de leche mediante un controlador robusto y un Gateway para la comunicación de datos* [Tesis de licenciatura, UPSE].
5. MathWorks. (2023). *Neural Network Toolbox Documentation*. https://www.mathworks.com/help/nnet
6. Medina Carrillo, J., & Zevallos Peña, R. (2020). *Diseño de un sistema de clasificación de maracuyá según su estado de madurez utilizando visión artificial y PLC S7-1200* [Tesis de licenciatura, Universidad Nacional de Trujillo].
7. Ogata, K. (2010). *Ingeniería de control moderna* (5.ª ed.). Pearson Educación.

8. Pampamallco Jara, J. (2019). *Diseño e implementación de controladores basados en lógica difusa para sistemas de control industrial* [Tesis de licenciatura, Universidad de Ingeniería y Tecnología]. Repositorio UTEC.

9. Ramírez Pozo, S. A. (2022). *Diseño y simulación de la automatización del proceso de fabricación de néctar de mango* [Tesis de licenciatura, UPSE].

10. Ramírez Rodríguez, J. (2020). *Implementación de un control neuronal directo en el controlador lógico programable S7-1200 para aplicaciones industriales* [Tesis de grado, Universidad de Pamplona]. https://repositoriodspace.unipamplona.edu.co/jspui/handle/20.500.12744/4349

11. Rockwell Automation. (2020). *Color and contrast sensors*. https://www.rockwellautomation.com

12. Salgado Rubiano, J. S. (2023). *Diseño de un sistema automatizado para el control y monitoreo del proceso de fabricación de hielo en la empresa MOSATEC utilizando herramientas de la industria 4.0* [Tesis de licenciatura, UPSE].

13. Santos Florencia, R. M. (2024). *Diseño de un simulador de sistema SCADA del proceso de conservación de granos de maíz en la etapa de almacenamiento usando PLC, HMI y Wonderware Intouch* [Tesis de licenciatura, UPSE].

14. Siemens AG. (2020). *SIMATIC S7-1200 System Manual*. https://support.industry.siemens.com

15. Siemens AG. (2021). *TIA Portal – Totally Integrated Automation Portal*. https://new.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal.html

16. SICK AG. (2022). *Color Sensors CS8-2*. https://www.sick.com

17. The MathWorks, Inc. (2023). *MATLAB Documentation*. https://www.mathworks.com/help/matlab

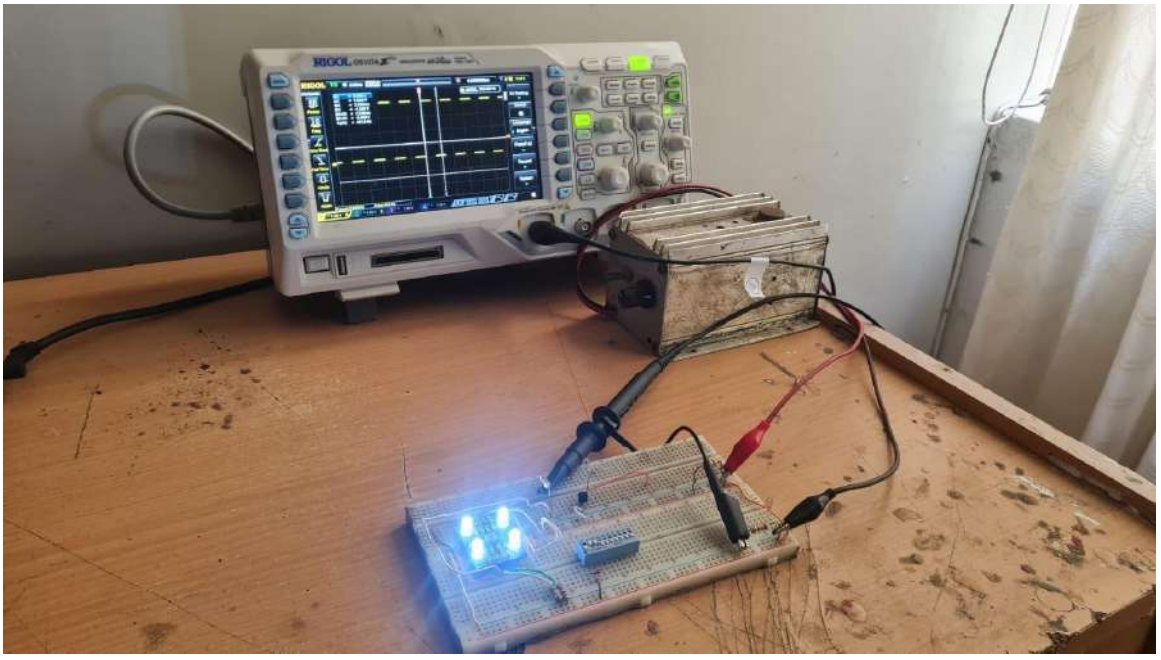
18. Villao Paredes, K. A. (2023). *Diseño de un prototipo de sistema de monitoreo y predicción del consumo eléctrico en zonas residenciales usando redes neuronales artificiales* [Tesis de licenciatura, UPSE].

19. Vishay Intertechnology. (2021). *Color sensors for industrial automation*. https://www.vishay.com

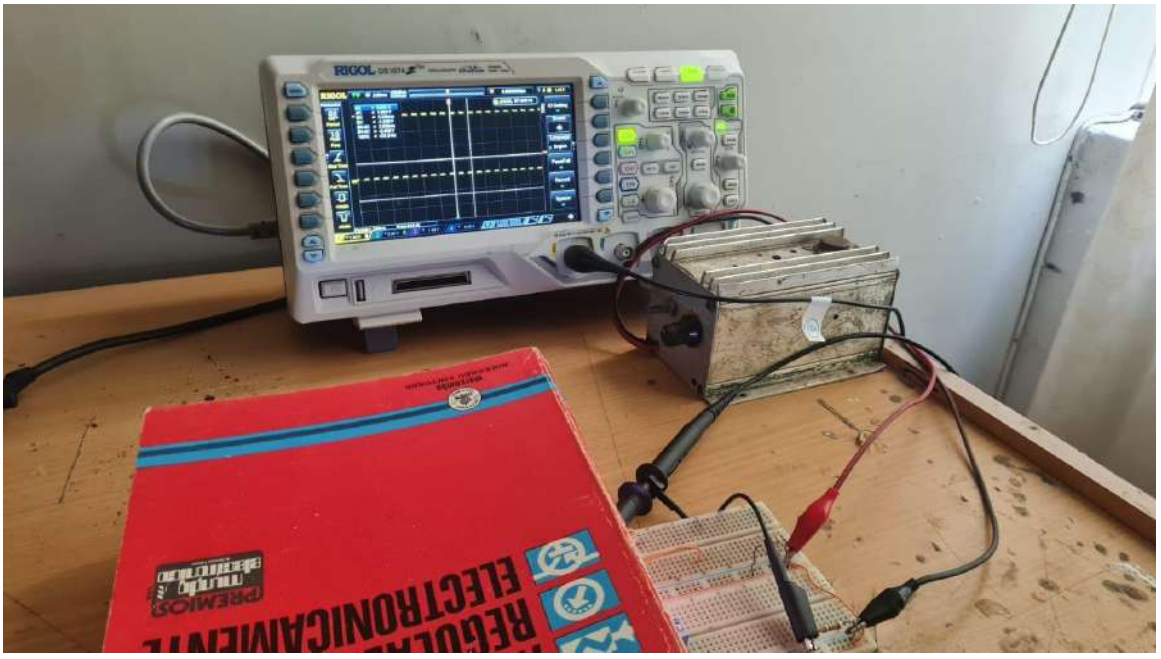
20. Zhang, Y., & Zhao, H. (2019). Industrial color detection and classification based on RGB color model. *IEEE Sensors Journal, 19*(14), 5278–5285. https://doi.org/10.1109/JSEN.2019.2910479

ANEXOS

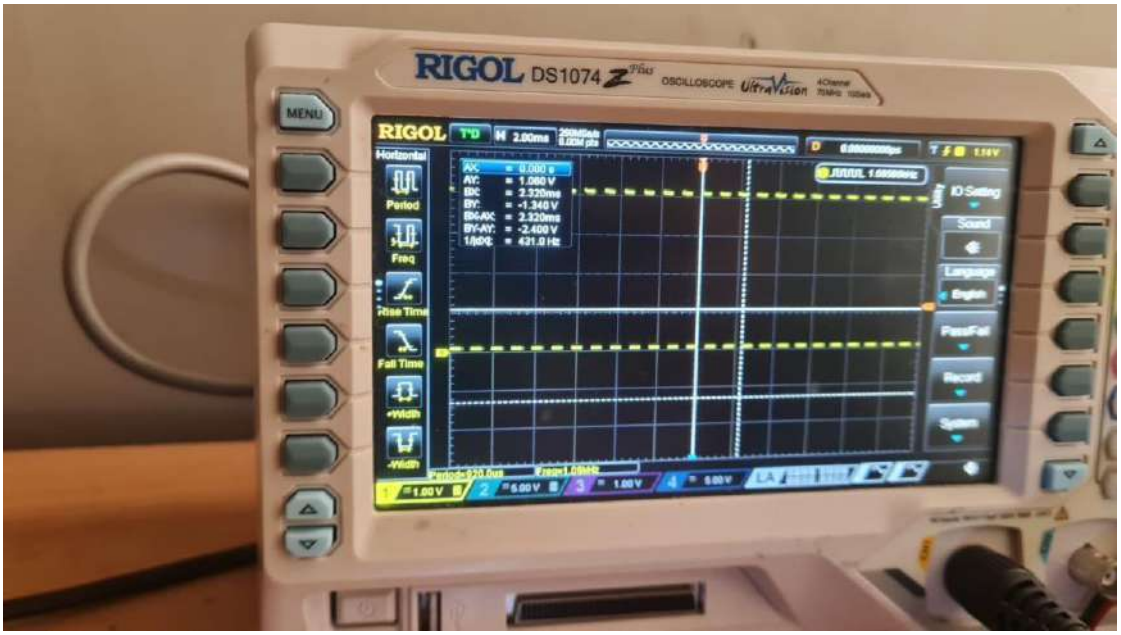
Anexo 1: Circuito experimental



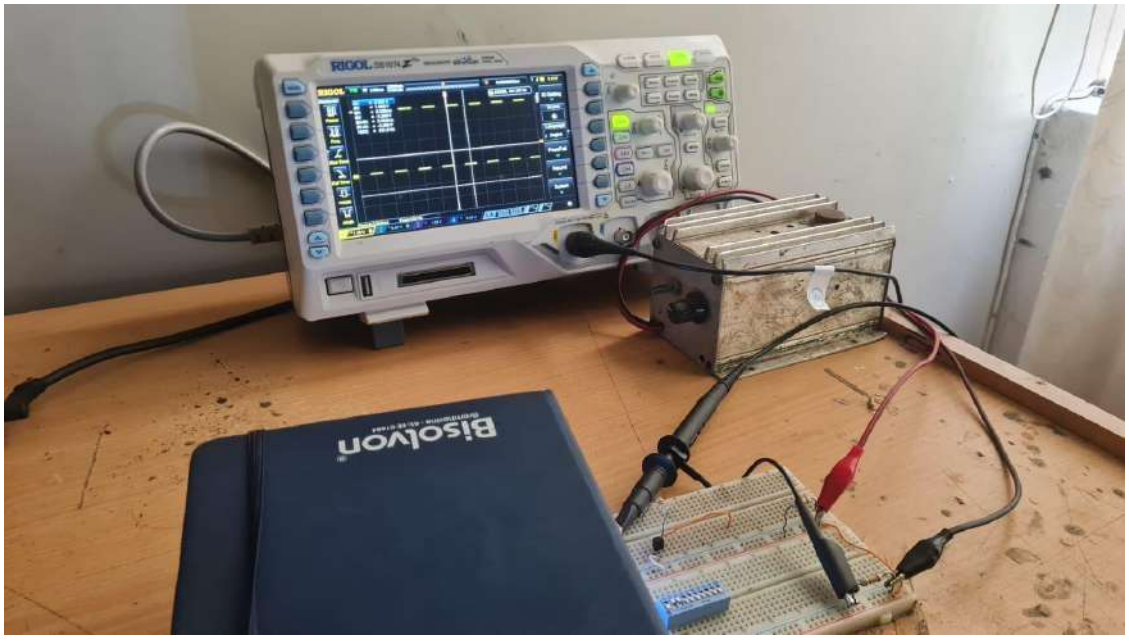
Anexo 2: Prueba para el objeto de color rojo



Anexo 3: Respuesta para el filtro de color rojo

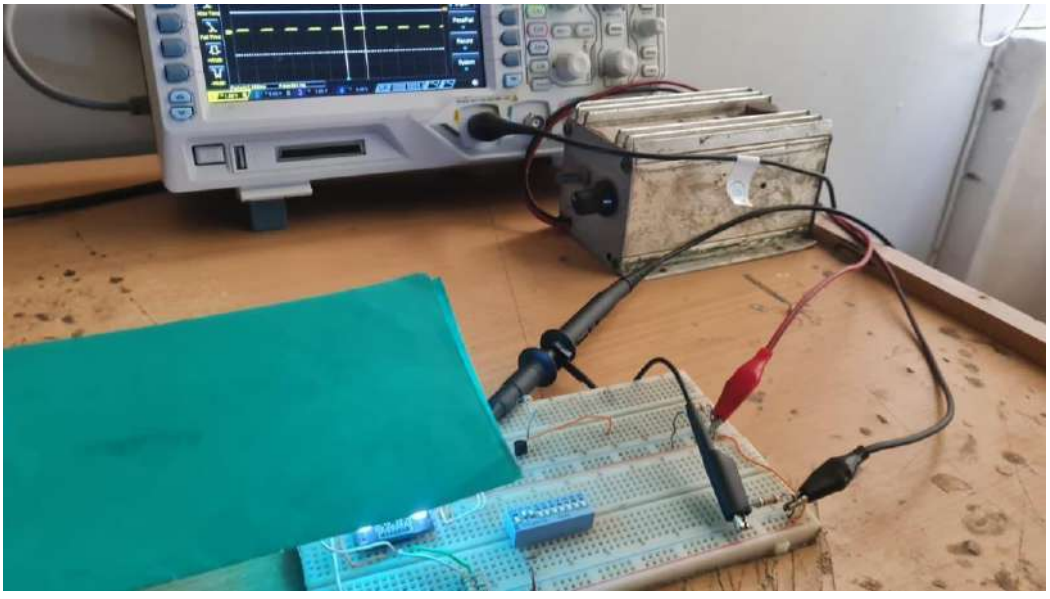


Anexo 4: Prueba para el objeto de color azul



Anexo 5: Respuesta para el filtro de color azul

Anexo 6: Prueba para el objeto de color verde



Anexo 7: Respuesta para el filtro de color verde