UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE INGENIERÍA ELECTRÓNICA

ROBOT MÓVIL DE DESTREZA PARA LABERINTO Y SEGUIDOR DE LÍNEA UTILIZANDO INTELIGENCIA ARTIFICIAL

Trabajo de titulación previo a la obtención del
Título de Ingeniero Electrónico.

AUTOR: Joel Alexander Díaz Bonilla

TUTOR: Carmen Johanna Celi Sánchez

Quito-Ecuador

2023

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Joel Alexander Díaz Bonilla con documento de identificación N° 1725042236; manifesto que:

Soy el autor y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 23 de febrero del año 2023

Atentamente,

Joel Alexander Diaz Bonilla
1725042236

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, Joel Alexander Díaz Bonilla con documento de identificación N° 1725042236, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy el autor del Proyecto Técnico: "Robot móvil de destreza para laberinto y seguidor de línea utilizando inteligencia artificial", el cual ha sido desarrollado para optar por el título de: Ingeniero Electrónico, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana

Quito, 23 de febrero del año 2023

Atentamente,

Joel Alexander Diaz Bonilla
1725042236

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo , Carmen Johanna Celi Sánchez con documento de identificación N° 1717437808, docente de la Universidad Politécnica Salesiana declaro que bajo mi dirección y asesoría fue desarrollado el trabajo de titulación: ROBOT MÓVIL DE DESTREZA PARA LABERINTO Y SEGUIDOR DE LÍNEA UTILIZANDO INTELIGENCIA ARTIFICIAL, realizado por Joel Alexander Díaz Bonilla, con documento de identificación N° 1725042236, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto Técnico, que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 23 de febrero del año 2023

Atentamente,

Ing. Carmen Johanna Celi Sánchez Mgtr.
1717437808

DEDICATORIA

Dedico el presente proyecto a mis padres Jorge y Maribel por su apoyo incondicional durante toda esta etapa universitaria, por darme fuerzas en los malos momentos y compartir los buenos, por confiar en mí y en mis metas ayudándome a cumplir cada objetivo que me propuse, agradezco a mi padre por compartir su sabiduría y ayudarme en todo este proceso, a mi madre por presionarme para no desmayarse ante los obstáculos que se prestaron y aprender de estos. A mi hermana Melanie que me apoyo y me escucho cuando más lo necesitaba y mi novia Daysi por ser la persona que me impulso todos los días para cumplir con mis objetivos, por hacerme entender que el camino, aunque sea difícil no será imposible y sobre todo a darme fuerzas y enseñarme que cualquier cosa que me proponga lo conseguiré con esfuerzo y humildad.

Joel Alexander Díaz Bonilla

AGRADECIMIENTO

Agradezco a la Universidad Politécnica Salesiana por brindarme las herramientas para sobresalir y cumplir mis metas, por compartir sus conocimientos y brindarme una excelente educación humana y académica, además agradezco a mis profesores que resolvieron todas mis dudas y me ayudaron a cultivar valores y como ser un buen profesional.

De igual manera agradezco a la Ing. Carmen Johanna Celi Sánchez MSc. Por brindarme su apoyo incondicional y experiencia para poder solventar los errores y problemas que se presentaron durante la realización de este proyecto logrando cumplir con las metas propuestas.

ÍNDICE DE CONTENIDO

DEDICATORIA vi
AGRADECIMIENTO vii
ÍNDICE DE CONTENIDO viii
ÍNDICE DE FIGURAS xi
RESUMEN xiv
ABSTRACT xv
INTRODUCCIÓN xvi
CAPITULO 1 1
ANTECDENTES 1
1.1 Planteamiento del problema 1
1.2 Justificación del proyecto 1
1.3 Objetivos 3
1.3.1 Objetivo General 3
1.3.2 Objetivo específicos 3
CAPITULO 2 4
MARCO TEÓRICO 4
¿Qué es Robótica? 4
2.1 Robot Autónomo 5
2.2 Robot Destreza seguidor de línea 5
2.2.1 Arquitectura del Robot seguidor de línea 7
2.3 Robot laberinto 9
2.4 Microcontrolador 10
2.5 Inteligencia Artificial 12
2.6 Red Neuronal 13
2.7 Tipos de redes neuronales 13
2.7.1 Arquitectura de una red neuronal 14

2.7.2 Perceptron..... 15
2.7.3 Red neuronal MLP 16
2.7.4 Red neuronal Adaline y Madaline..... 17
CAPITULO 3 19
DISEÑO E IMPLANTACIÓN DEL ROBOT AUTÓNOMO MODO SEGUIDOR DE LÍNEA DESTREZA..... 19
3.1 Diseño y construcción del robot autónomo. 19
3.1.1 Sistema Sensorial 19
3.1.2 Sistema de control 21
3.1.3 Motores y ruedas 21
3.1.4 Fuente de alimentación del robot 22
3.1.5 Diseño esquemático..... 23
3.2 Diseño de la red neuronal Artificial para el aprendizaje del robot seguidor de línea de destreza. 25
3.3 Creación de la red Neuronal perceptrón modo seguidor de línea. 27
3.4 Programación del microcontrolador para robot modo seguidor de línea de destreza..... 30
3.5 Programación del PID para el robot seguidor línea de destreza. 32
DISEÑO E IMPLEMENTACIÓN DEL ROBOT AUTÓNOMO MODO LABERINTO 33
3.6 Diseño y creación de la red neuronal Adaline..... 33
3.7 Diseño y Creación de la red neuronal Adaline..... 36
CAPITULO 4 40
PRUEBAS Y RESULTADOS 40
4.1 Prueba de entrenamiento y desarrollo de la red neuronal artificial para el control de Robot modo seguidor de línea. 40
4.1.1 Creación de la red Backpropagation con función de transferencia tangente sigmoidal y lineal. 40

4.1.2 Creación de la red Backpropagation con función de transferencia en la primera capa línea y en la segunda lineal. 41
4.1.3 Prueba de aprendizaje con la creación de la red Adaline. 41
4.1.4 Resultados finales. 42
CONCLUSIONES 46
RECOMENDACIONES 48
REFERENCIAS 50
Bibliografía 50
ANEXOS 53

ÍNDICE DE FIGURAS

Figura 2.1 Robot 3PI. 7
Figura 2.2 Robot seguidor de línea. 9
Figura 2.3 Pista de robot laberinto 10
Figura 2.4 Microcontrolador 12
Figura 2.5 Estructura de una red neuronal 14
Figura 2.6 Estructura Simpe del Perceptron 15
Figura 2.7 Estructura del MLP. 16
Figura 2.8 Estructura del Adaline 17
Figura 3.1 Sensor ultrasónico HC-SR04 20
Figura 3.2 Sensor infrarrojo HW-201 20
Figura 3.3 Raspberry pi pico 21
Figura 3.4 Motor 12 voltios con Driver TB6612fng 22
Figura 3.5 llanta omnidireccional 22
Figura 3.6 Diseño esquemático. 24
Figura 3.7 Placa PCB 25
Figura 3.8 Estructura de la red neuronal perceptrón para el R.M.S 26
Figura 3.9 Matriz de entrada R.M.S 26
Figura 3.10 Matriz de salida R.M.S 27
Figura 3.11 Creación del perceptrón. 28
Figura 3.12 Estructura del perceptrón. 28
Figura 3.13 Valores de consulta del perceptrón 29
Figura 3.14 obtención de los pesos de la red artificial 29
Figura 3.15 Respuesta del perceptrón 30
Figura 3.16 subrutina de la red neuronal 31
Figura 3.17 Acondicionamiento de la salida R.M.S 31
Figura 3.18 PID R.M.S 32

Figura 3.19 Control de motores R.M.S 33
Figura 3.20 Localización de los sensores ultrasónicos. 34
Figura 3.21 Creación y limitación del vector de entrada. 35
Figura 3.22 Creación de la red neuronal Adaline. 36
Figura 3.23 Creación de la red Adaline 37
Figura 3.24 Implementación de la red Adaline 38
Figura 3.25 Programación de dirección al frente. 39
Figura 3.26 Programación de dirección derecha. 39
Figura 4.1 Prueba de velocidad del robot modo seguidor de línea. 43
Figura 4.2 Prueba de velocidad del robot modo laberinto 45

ÍNDICE DE TABLAS

Tabla 2.1 Tabla de características del robot seguidor de línea. 8

Tabla 2.2 Tabla de características del robot seguidor de línea. 11

Tabla 3.1 Pines de conexión del microcontrolador. 23

Tabla 3.2 Matriz de entradas y salidas. 35

Tabla 4.1 Pruebas del robot modo seguidor de línea. 42

Tabla 4.2 Pruebas del robot modo seguidor laberinto. 44

RESUMEN

La robótica ha tenido una acogida muy grande en los últimos tiempos, lo que permitió que muchos estudiantes desde los colegios hasta la educación superior formando clubes de robótica y competencias para poner a prueba sus diferentes prototipos, sin embargo, el costo de elaboración de un robot suele ser elevado debido a los componentes y la calidad de los mismo, por este motivo los competidores se limitan a una sola categoría, por esta razón el presente trabajo propone un prototipo que permita desempeñarse en los ámbitos de seguidor de línea y resuelve laberintos, con el fin de reducir costos y aplicar nuevas técnicas de diseño y programación.

Los robots autónomos poseen una estructura similar para competir en diferentes categorías y constan de motores, sensores y sistemas de control los que permiten controlar el robot. Estos sensores se utilizarán dependiendo de qué tipo de competencia se está realizando sea seguidor de línea o resuelve laberinto.

Para que el robot pueda concursar en las categorías de seguidor de línea y laberinto se diseñó un prototipo modular, este dispone de sensores ultrasónicos para el modo laberinto y de sensores ópticos para seguidor de línea, los cuales se pueden retirar para un mejor manejar del robot en la pista.

Para el sistema de control no se realizó un sistema PID o similares, se utilizó redes neuronales artificiales, las cuales se encargan de realizar el control del robot en base a un sistema de datos que se ingresó en la red neuronal artificial, los sensores proporcionan la información a la red neuronal y evalúa que decisión debe tomar si girar o seguir.

Palabras claves: Inteligencia artificial, redes neuronales, robot, multi - categoría.

ABSTRACT

Robotics has been very well received in recent times, which allowed many students from schools to higher education to form robotics clubs and competitions to test their different prototypes, however, the cost of developing a robot It is usually high due to the components and the quality of the same, for this reason the competitors are limited to a single category, for this reason the present work proposes a prototype that allows to perform in the areas of line follower and solves mazes, with in order to reduce costs and apply new design and programming techniques.

Autonomous robots have a similar structure to compete in distinct categories and consist of motors, sensors and control systems that allow the robot to be controlled. These sensors will be used depending on what type of competition is being carried out, be it a line follower or a maze solver.

In order for the robot to compete in the line follower and maze categories, a modular prototype was designed. It has ultrasonic sensors for the maze mode and optical sensors for the line follower, which can be removed for better handling of the robot. on the track.

For the control system, a PID or similar system was not made, artificial neural networks were used, which are in charge of controlling the robot based on a data system that was entered into the artificial neural network, the sensors provide the information to the neural network and evaluates what decision to make whether to turn or continue.

Keywords: Artificial intelligence, neural networks, robot, multi-category.

INTRODUCCIÓN

En el capítulo 1, se presentan los antecedentes del proyecto los cuales plantean la problemática, la justificación y los objetivos de este.

El Capítulo 2, se enfoca en la definición de robot autónomo además de mostrar que es un robot seguidor de línea de destreza y sus principales características, de igual manera que es un robot resuelve laberinto y su funcionamiento.

Se define los conceptos de microcontroladores, la inteligencia artificial y que es una red neuronal y su arquitectura.

Para finalizar se muestra el diseño y construcción del prototipo, en el cual se explica las partes que lo componen como sensores, motores y como están distribuidos.

El capítulo 3, se centra en la creación de la red neuronal para reconocer las diferentes figuras del seguidor de línea y determinar qué acción debe tomar el robot, parar esto tiene un entrenamiento y la implementación del robot.

Se diseña la red neuronal Adaline que permite ingresar valores, los cuales serán utilizados en la red neuronal para realizar el direccionamiento, según se determine en la lógica de programación para resolver el laberinto, de igual manera consta con dos algoritmos, los cuales son el de consulta y entrenamiento, finalizando con la aplicación en el robot.

En el capítulo 4 se realizaron las pruebas en las dos modalidades laberinto y seguidor de línea de destreza, se evalúan los resultados obtenidos en las dos pistas, además de corregir errores y evaluar las desventajas y ventajas que se obtiene al implementa este tipo de control.

Por último, se definen las conclusiones y recomendaciones obtenidas de las pruebas y puntos que se debe tomar en cuenta a la hora de diseñar y poner en práctica el robot.

CAPITULO 1
ANTECDENTES

1.1 Planteamiento del problema

La tecnología ha tenido un avance significativo en los últimos años, siendo una herramienta fundamental para la educación, sin embargo, estas tecnologías y herramientas tiene un costo elevado, no obstante, la robótica abre posibilidades con hardware simple y de bajo costo con los que se puede realizar varios procesos gracias a la programación y la inteligencia artificial, que permite realizar operaciones complejas con recursos limitados. Se han creado aplicaciones de bajo costo las cuales se desarrollan con el fin de implementar la inteligencia artificial en la robótica, lo que permite el estudio y desarrollo de nuevas tecnologías. (Herrera & Daniel, 2010)

Los estudiantes de la carrera de electrónica tienen la destreza y la motivación de realizar sus propios prototipos y participan en concursos internos y externos en categoría de seguidor de línea de destreza y laberinto, pero a pesar de que desean participar los costos de los robots son elevados, lo que restringe la participación en más categorías.

La falta de instrucción de cómo diseñar y programar los robots es un factor importante para la poca participación, debido a que no existe un laboratorio que permita aprender, verificar diferentes métodos de pruebas y la corrección de errores, lo que cual dificulta el desarrollo de la robótica. (Burgos, 2020)

1.2 Justificación del proyecto

Hoy en día la industria 4.0 (término utilizado por primera vez en la Feria de Hanover, Alemania, en el año 2011) se incluye en todos los procesos digitales y avance de procesamiento de datos, software inteligente, etc.

La inteligencia artificial se define como el entendimiento en que se relacionan los hechos y las cosas, pero es preciso no confundir la capacidad de memorizar ni la sabiduría, ya que la inteligencia artificial se enfoca en la capacidad de analizar y recordar datos lo cuales permiten relacionarlos para obtener la salida deseada. (Adriana, 2020)

Gracias al avance de la tecnología e inteligencia artificial se obtienen mejores resultados, si bien los robots seguidores de línea utilizan lógica de programación convencional, estos

se ven afectados en circunstancia que no se tomaran en cuenta en la programación, sin embargo con la inteligencia artificial se tiene la ventaja de adaptarse a las circunstancia, ya que al poseer un algoritmo que aprende y se adapta se obtiene mejores resultados, que no serán afectados por eventos externos, como ejemplo si se utiliza lógica de programación convencional en un robot laberinto, si este no está en parámetros en los cuales se programaron como son la cantidad de luz o el tipo de laberinto tiende a fallar, en cambio con redes neuronales se puede entrenar para que se acople a diferentes circunstancias y el proceso de adaptación es innato en una red neuronal que consiste en la capacidad del robot para tomar decisiones acertadas en un momento de circunstancias desconocidas para el algoritmo y en el cual el robot responderá tal cual lo haría un ser humano.

Los robots laberinto y seguidores de línea de destreza son robots autónomos por ende su estructura es similar, posen sensores y motores para realizar sus respectivas operaciones, sin embargo con el avance de nuevas tecnologías ahora los microcontroladores tiene una mayor capacidad y pueden realizar varios procesos, es decir con un microcontrolador se puede utilizar un solo prototipo que en su estructura tenga sensores para los seguidores de línea y a la vez los sensores para el laberinto lo que permite que por medio de software se elija en que categoría se va concursar, con la ayuda de la inteligencia artificial se puede obtener un mejor rendimiento ya que permite realizar mejores procesos de analizar y realizar un mejor control.

La inteligencia artificial sin bien es objeto de estudio en las carreras de ingeniería no se ha dado de forma práctica, como introducirla en un robot y que este pueda realizar las acciones deseadas, ya que se estudia de manera teórica y muy poca de manera práctica por eso es esencial desarrollar proyectos utilizando IA, para que así se ponga en práctica y se desarrollen las habilidades de programación, esto ayudará a que existan mejores avances en el área de IA con robótica, además de que los estudiantes podrán obtener mejores conocimientos en la práctica ayudando a que exista mayores proyectos que involucren la inteligencia artificial y sus ventajas.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un robot de destreza para participar en la categoría de seguidor de línea y la categoría de laberinto utilizando algoritmos de redes neuronales.

1.3.2 Objetivo específicos

- Investigar el estado de la tecnología de hardware y software para la implementación de redes neuronales dentro de los microcontroladores o los microprocesadores.

- Programar redes neuronales que permitan el manejo autónomo e inteligente del robot tanto en la categoría de laberinto y seguidor de línea de destreza.

- Construir un robot móvil para su implementación con redes neuronales en la tarjeta electrónica de control.

- Realizar las pruebas de funcionamiento en el circuito de un laberinto y en una pista de seguidor de línea de destreza, para la determinación de las ventajas del prototipo.

CAPITULO 2
MARCO TEÓRICO

¿Qué es Robótica?

La robótica es la ciencia para desarrollar y diseñar un robot permitiendo la formación profesional de la ingeniería, empleadas de las carreras de automatización, mecánica, informática y mecatrónica (Valverde, 2020).

Los robots desempeñan tareas humanas mediante la programación y el uso de la inteligencia artificial.

Por su generación se clasificación de robots:

- Primera Generación: Son aquellos que tienen un sistema para ser controlados de manera manual.
- Segunda Generación: Son robots que reproducen secuencias de movimientos realizados anteriormente por humanos.
- Tercera Generación: Se controlan por medio de sensores, los que a su vez son guiados por un software, mediante señales y actuadores (Esneca, 2018).
- Cuarta Generación: En esta generación los robots son inteligentes capaces de controlar los cuales poseen sistemas sensoriales.
- Quinta Generación: Nace la inteligencia artificial y empieza la nueva era creando robots autónomos con capacidades de poder imitar las funciones cognitivas que posee la mente del ser humano (Esneca, 2018).

También se clasifican por la estructura del robot (ICCIUTEM, 2014):

- Polis articulados: Varias piezas con movimientos.
- Móviles: Son rodantes.
- Zoomórficos: Similitudes con animales.
- Antropomórficos: Similitudes con seres humanos.

Aportes positivos de la robótica:

Productividad: Crecimiento significativo dentro de empresas y fábricas, gracias a que estos robots realizan tareas complejas en menos tiempo.

Hostilidad: Permiten el acceso a terrenos hostiles en las que el ser humano no puede ingresar, por ejemplo, Chernóbil debido a su alta radioactividad.

Medicina: Realizan intervenciones quirúrgicas mediante máquinas con software que le permiten exactitud.

2.1 Robot Autónomo

Estos tipos de robots se han vuelto indispensables dentro del área industrial 4.0, ya que cumplen efectivamente sus labores, pero ¿Qué son estos robots?, tiene la habilidad de realizar, ejecutar tareas, sin hacer uso de comandos, ni controles de una manera explícita por parte del ser humano.

Estos robots autónomos son prácticos dentro de las industrias y del comercio, además de ser útiles en diferentes elementos ya sea: aire, agua y también el espacio. Estos tienen la capacidad de realizar tareas con autonomía propia, en 194 inició el desarrollo de estos, cuando William Grey diseñó un par de robots a los cuales denominó Elmer y Elsie, a los cuales apodó como tortugas debido a su lento movimiento, capaces de desempeñar actividades en un entorno distinto (Ripipsa, 2020).

parámetros reconocidos durante estos años de desarrollo para el robot autónomo son:

- Obtener información de forma permanente de los distintos ambientes, y de esta manera desempeñar su actividad.
- Realiza trabajos por largo tiempo, sin la participación directa del ser humano.
- Se moviliza mediante la función de su área de operación, sin la necesidad de que el ser humano sea partícipe.
- Evita riesgos en base a sus propias decisiones.
- Logran coordinar actividades con otros robots autónomos, gracias a los parámetros impuestos de las redes neuronales, algoritmos genéticos, lógica borrosa y aprendizaje por esfuerzo.

2.2 Robot Destreza seguidor de línea

Este tipo de robots tienen como objetivo completar un trayecto complicado, el cual presenta algunos obstáculos, mediante líneas que se encuentran plasmadas en un área

designada. El robot detecta la línea por medio de sensores infrarrojos, lo que le permite mantenerse sobre esta.

Los eventos más importantes que se han realizado son:

- Micro Mouse de 1979 se llevó a cabo la primera competencia en New York Pequeños robots capaces de completar un laberinto, en ella hubo 15 participantes, siendo campeón Arthur Boland. John Billinsley (robotsystems.net, 2014).
- Micro Mouse de 1980, llevo la competencia a Europa en esta competencia el laberinto está formado por celdas conformando un cuadrado, el cual consta de paredes color blanco y su superficie es de color negro (robotsystems.net, 2011).
- Japón 2009 una nueva versión aparece denominada Half-Size, siendo este más pequeño que el tradicional, el laberinto empleado para este tipo de robots

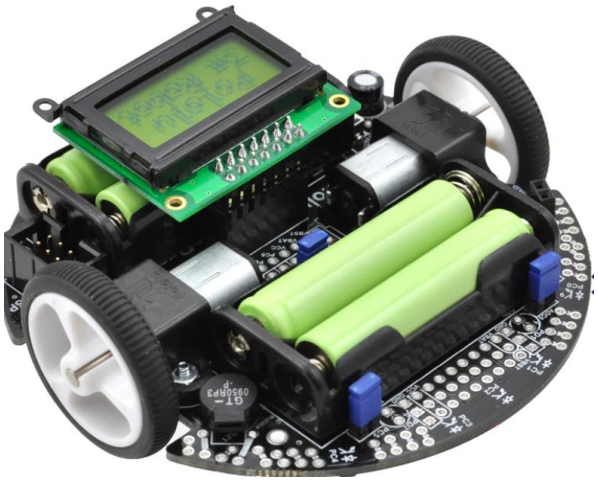
Estos robots tienen un buen rendimiento ya que poseen una gran velocidad (5 m/s), al pasar del tiempo el rendimiento de estos robots ha ido acrecentándose, en el año 2015 el robot que se declaró ganador llego a la velocidad de 10 m/s tanto de freno como de aceleración,

Los robots más recientes, han sido equipados con un ventilador capaz de generar un vacío parcial debajo del micro ratón. Gracias a este ingenio, el rendimiento ha aumentado de forma notable logrando aceleraciones de más de 6g (6 veces la fuerza de la gravedad) en curvas y aceleraciones en línea recta que fácilmente exceden las 2,5g.

Robot 3Pi

Este robot es una plataforma, este está constituido por dos motores micro metálicos, sensores reflectantes, es un robot que tiene una gran velocidad, algunas características de este robot son: un microcontrolador de microchip que funciona a 20 MHz de igual manera posee una memoria 32kb y un controlador como muestra la Figura 2,1 (Pololu, 2018).

Figura 2.1 Robot 3PI.



Pololu 3pi Robot, Elaborado por: (Pololu, 2018)

2.2.1 Arquitectura del Robot seguidor de línea

Este robot se encuentra conformado por carcaza, microcontroladores, llantas y varios componentes electrónicos, lo que se debe tener en cuenta al realizar este tipo de robots es que deben ser livianos.

Se puede contar con diversos materiales para la elaboración de este como por ejemplo aluminio, latón, plástico, madera y soportes de aluminio.

Sensores infrarrojos.

Se emplearon 5 y son instrumentos optoelectrónicos que permiten al robot identificar el camino trazado en la superficie, gracias al emisor de infrarrojo, que reflecta la señal sobre el área que se está censando.

Motores Dc de 12 voltios.

Este tipo de motores tienen alta velocidad, son silenciosos y su consumo es mínimo.

Puente h

Circuitos electrónicos que posibilitan a un motor DC antes mencionado, girar en ambos sentidos, al igual que le permite avanzar y retroceder.

Las señales transmitidas por los sensores son análogas y se convierten en digitales para que de esta manera el microcontrolador sea capaz de manejar dichas señales de una manera más específica, por ello se emplea comparadores de señal, lo que hace posible el cambio de señales analógicas a digitales, pasando así por el sensor.

El robot se guía por medio del algoritmo de red neuronal el cual toma una decisión basándose en las señales que provienen de los sensores, entregando los valores de salida que determinaran los valores de velocidades para los motores

Existen competencias de robots seguidores de línea, los cuales se fabrican para cumplir un recorrido.

Para llevarse a cabo estas competencias, se deben considerar algunos puntos para participar, como:

Área de la pista: Según (IEEE), determina que esta se debe realizar en dos pistas de 2m x 2m con distintos itinerarios o rutas, así mismo estas para poder ser calificadas será divididas en 4 secciones según su dificultad como se observa en la Tabla 2.1.

Tabla 2.1 Tabla de características del robot seguidor de línea.

Características principales de una pista para el Robot Seguidor de Línea	
Dimensiones de la pista	2 m × 2m
Color de la línea o trayectoria a seguir	Negro
Ancho de la línea a seguir	2 cm (20mm)
Color del fondo de la pista	Blanco
Número de rampas	2 máximo
Ángulo de las pendientes	15° máximo
Longitud aproximada de la trayectoria	Menor a 10 metros

Tabla de características del robot seguidor de línea Fuente: (IEEE, 2018)

También como da a conocer (IEEE), esta ruta trazada debe contar con algunas dificultades: curvas cerradas (ángulo mínimo de 30°), secciones ausentes de línea hasta 2,5 cm, cruces, rampas con inclinación de máximo 15° y túneles de las medidas 30 cm ancho y 17 cm alto.

En la competencia se debe respetar la normativa la cual es que el robot competirá en el momento justo que sea nombrado no después, la competencia inicia en el preciso

momento en el que inicie su robot y el tiempo correrá a partir del cruce de la línea de inicio y terminara dicho conteo al culminar la ruta o pista, se considera que al no regresar el robot sobre el recorrido la trayectoria en 10 segundos se culminará.

Para determinar el ganador se califican de acuerdo con los puntos acumulados en el menor tiempo, para cerrar una trayectoria dispondrá de 5 minutos, se les otorgan a 3 reinicios por cada pista.

Como en toda competencia en esta no falta las amonestaciones las cuales son:

Ingresar a la pista sin que el juez lo haya autorizado, movilizar el robot sin autorización previa, deteriorar la pista, ocasionar daños a contrincantes de manera internacional (IEEE, 2018).

Tal como se aprecia en la Figura 2.2 se observa un prototipo de robot seguidor de línea usado para competencias.

Figura 2.2 Robot seguidor de línea.

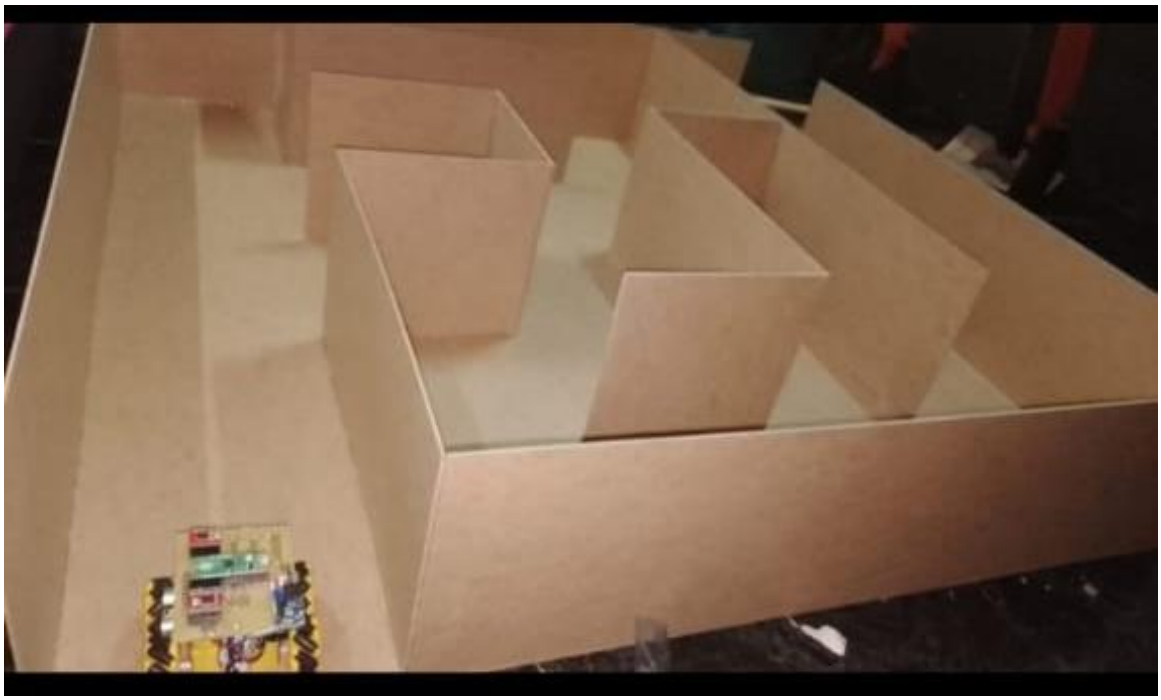
Construcción de robot seguidor de línea. Fuente: (Carrillo, Cardona, Arvizo, & Rodríguez, 2016)

2.3 Robot laberinto

Gracias al avance de la tecnología se han buscado formas automatizadas para resolver problemas o realizar tareas, lo que nos ha facilitado la vida, estos robots contienen hardware (físico del robot) y software (lógica).

Debe existir un equilibrio en el funcionamiento de estos ya que, si el uno es menos eficiente, este no podrá desarrollar su potencial; para que el robot logre aprender existen varios algoritmos, de modo que se implemente aquello algoritmo que se acople mejor al hardware.

Figura 2.3 Pista de robot laberinto.



Pista de robot laberinto, Elaborado por: Joel Díaz

2.4 Microcontrolador

Creados en Texas Instruments en 1970, aunque en esa época solo eran microprocesadores que poseían la funcionalidad de una memoria, en el año 1971 Gary Boone y Michael Cochran crearon el TMS 1000, el cual estaba conformado por 4 bits de ROM y RAM, este se utilizaba en diversos productos hasta el año 1974. Su desarrollo fue mejorando con el pasar del tiempo gracias a ello fueron vendidos a industrias, En 1976 Intel también fabricó uno de los mejores microcontroladores como el 8048, el cual fue empleado como procesador de un teclado, en 1990 aparecieron microcontroladores como las ahora conocidas como flash memory, estos permiten grabar, borrar y regrabar datos, gracias a

su avance se destinan estos microcontroladores para automatizar máquinas, apoyo dentro de la comunicación y otros dispositivos, en 2010 Atmel Flash lanzó un nuevo microcontrolador más pequeño el cual podría ser empleado en productos más pequeños.

Fabricantes y aportes importantes son mencionados a continuación en la Tabla 2.2

Tabla 2.2 Tabla de características del robot seguidor de línea.

Microship	PIC 8 bits
Atmel Corporation	8051, AT91SAM, dispositivos de radiofrecuencia, memorias EEPROM, flash, ASIC y WIMAX.
Texas Instruments	Circuitos integrados para celulares.
Intel	1974 microprocesador 8080
Freescale Semiconductor	Productos de conectividad
Zilog Inc	Microcontrolador Zilog Z80 8 bits
Motorola	Microprocesador 6800 8 bits

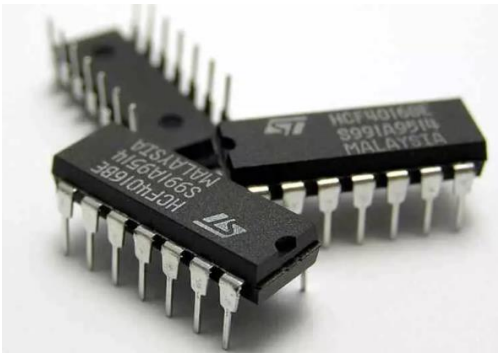
Tabla de características del robot seguidor de línea Fuente: (IEEE, 2018)

Estos microcontroladores tal como se indican en la Figura 2.4 cuentan con un CPU, unidad de memoria RAM y ROM. Pueden controlar un LCD, diseñar planes para ahorrar energía, también permite comunicarse por medio de Bluetooth, Wifi para recepción y envío de datos, procesa los datos de entrada y como resultado da una salida la cual nos permite realizar una actividad como, por ejemplo: en mecánica, robótica, automóviles, medicina e industria.

Algunas de sus características son:

- Capaces de compaginar y de esta forma menorar el gasto de energía eléctrica.
- Procesamiento central con mínimos circuitos externos.
- Cuenta con capacidades alrededor de 264kb de procesamiento.

Figura 2.4 Microcontrolador.



Construcción de carro seguidor de línea. Fuente: (Roca, 2021)

2.5 Inteligencia Artificial

Inicia en 1943 empieza gracias al escrito de Warren McCullogh y Walter Pitt, encontraron que el sistema nervioso tiene una gran similitud con el software de las computadoras, en 1949 Donal O. Hebb público su libro "La organización de la conducta" lo que se utilizó para que las neuronas artificiales aprendieran por medio de algoritmos (Rouhiainen, 2018).

Se clasifica en:

- Débil. - Aquella IA que solo pretende o aparenta razonar, es decir que no actúa como humano de una forma consciente si no como le programaron para reaccionar frente a un estímulo.
- Fuerte. - Algunas personas que se dedican a desarrollar la IA consideran que estas máquinas podrían llegar a desarrollar mente y sentimientos propios y que será capaz de imaginar, soñar y razonar como un humano.

Uno de los principales temas dentro de la IA es el aprendizaje autónomo, que quiere decir que aprenda algo para lo cual no se encuentra programado.

Este aprendizaje autónomo emplea algoritmos los cuales permiten que estos aprendan de patrones y estos se clasifican en:

- Supervisado: Dentro de la programación se ingresan datos por el ser humano para que pueda realizar funciones, tareas y actividades.

- No vigilancia: En su programación no cuenta con datos, que este recopila o aprende después mientras acrecentar su experiencia junto al usuario.
- Refuerzo: Quiere decir que obtendrán un pase verde cuando realicen un movimiento por su cuenta que acierte.

Gracias a la integración de la IA se ha evidenciado un ahorro de tiempo y optimización de respuestas, ya que es precisa gracias al aprendizaje mediante almacenamiento de información a través de las redes neuronales, como son las plataformas que se encuentran en los dispositivos celulares chatbots que permiten realizar tareas solo por medio de reconocimiento de voz como pedir apagar y encender luces, revisar alarmas, agenda, realizar llamadas sin tocar el celular esto siendo muy funcional a la hora de conducir, y también es útil para los más pequeños de la casa ya que al no saber escribir aún, por medio de la voz solicitan música, cuentos, videos, entre otros para su recreación (Rouhiainen, 2018).

2.6 Red Neuronal

Las redes neuronales artificiales simulan el funcionamiento del sistema nervioso del cuerpo humano, el cual permite respuestas y acciones frente a un estímulo o información, como las redes neuronales naturales se comunican a través de impulsos nerviosos, y el cerebro emite una respuesta, del mismo modo las neuronas artificiales se comunican a través de un conjunto de entradas que al multiplicarse determinan o clasifican un impulso, esta carga es procesada por la célula y emite un valor o respuesta denominado salida de la neurona artificial .

La capa es un grupo de neuronas, en la cual su información es recibida del exterior o conjunto de neuronas anteriores.

2.7 Tipos de redes neuronales

- Mono Capa: Se conforma por una capa que obtiene información del exterior, esta asimila el estímulo recibido y emite una respuesta, las neuronas de esta capa se encuentran desconectadas entre sí.

- Multicapa: Su extensión es mayor al mono capa, presenta la capa que recepta los estímulos y existen capas intermedias que se encargan de evaluar los datos y la siguiente capa se encarga de emitir la respuesta a una siguiente capa.
- Convolucional: Tiene una similitud al mono capa y multi capa, diferenciándose en que sus capas están semi conectadas a las de la siguiente capa y no se relacionan las neuronas dentro de la misma capa.
- Concurrentes: Estas neuronas se relacionan entre sí de manera aleatoria, sin estructura gracias a ello se genera una especie de memoria.

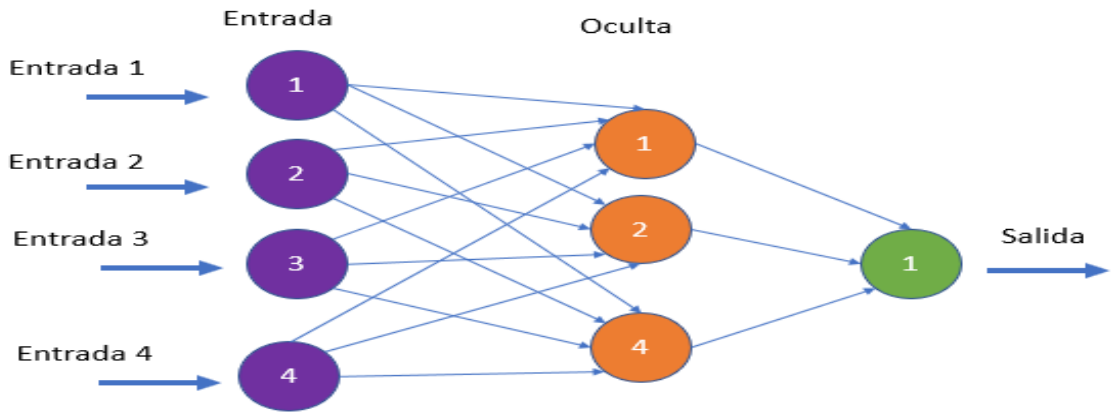
Algunos usos que se les asignan a estas redes neuronales pueden ser reconocimiento de voz como: Alexa, Siri, Cortana, automóviles autómatas como Tesla y Uber, sistemas de seguridad por medio de cámaras.

Para instruir una neurona debemos realizar el ajuste de los pesos de entrada y a su vez generen salidas.

2.7.1 Arquitectura de una red neuronal

La arquitectura de una red neuronal está organizada en capas la cual tiene entradas y salidas como se aprecia en la Figura 2.3.

Figura 2.5 Estructura de una red neuronal



Estructura de una red neuronal, Elaborado por: Joel Diaz

2.7.2 Perceptron

Es una arquitectura de las más simples como se observa en la Figura 2.4, ya que su estructura está basada en tan solo una entrada y una salida, su función es clasificar e identificar diferentes modelos, se maneja por medio de estímulos ópticos, presentó algunos inconvenientes en el caso que requiere obtener una respuesta no lineal, debido a la estructura básica que posee.

La forma de aprender del Perceptron es mediante un algoritmo el cual educa los pesos de manera automática y de esa forma decidir una entrada para generar una respuesta, Esta recepta señales de entrada cuando la suma de las entradas que registra es mayor a la entrada determinada, de esta forma se activa si el valor determinado es positivo.

Existen dos tipos de Perceptron el cual puede poseer una o varias capas, diferenciando que al solo tener una capa esta funciona sobre una función lineal de forma separada, al contrario de una multicapa que posibilita un mejor cálculo (Berzal, 2020).

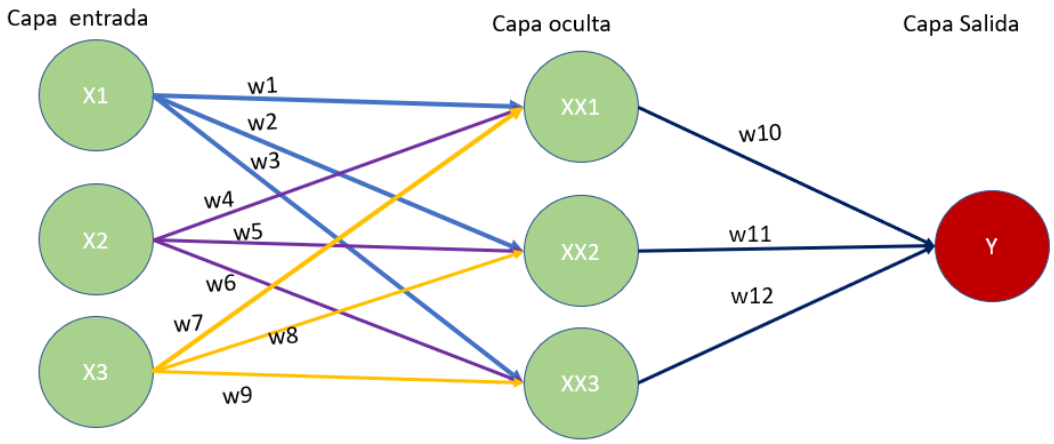
Figura 2.6 Estructura Simpe del Perceptron

Estructura del Perceptron, Elaborado por: Joel Diaz

2.7.3 Red neuronal MLP

Red multicapa Perceptrones consiste en la formación de nodos por capas, cada nodo puede recibir solo la entrada situada en la capa inferior, cada nodo realiza un cálculo de una combinación lineal para cada entrada que ingresa añadiendo un sesgo, aplicando una función de activación, conocida como la función de transferencia, en esta se realiza las operaciones para determinar el rango de entradas, dando así una salida que puede ingresar a un nuevo nodo como entrada, de la manera que se muestra la Figura 2.5 (Menacho, 2014)

Figura 2.7 Estructura del MLP.



Estructura de una red neurona MLP, Elaborado por: Joel Diaz

En el MLP pose diferente estructura de manera que poseer de una o varias salidas y cada una de estas poseen sus propios pesos y sesgo correspondiente, de modo general para la activación de la red se suele utilizar la misma función de transferencia, sin embargo, no es obligatorio en algunos casos se puede utilizar diferentes funciones de activación para cada neurona (Menacho, 2014).

2.7.4 Red neuronal Adaline y Madaline

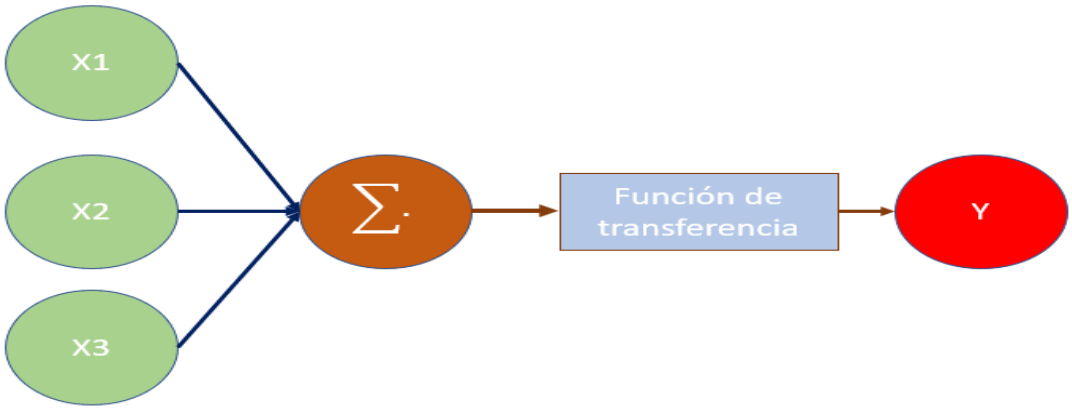
La red Neuronal Adaline fue creada por Bernard Widrow en el año de 1959. Su función se basa en la sumatoria lineal de la matriz de entradas la cual genera una función umbral para proporcionar el resultado de la suma como se visualiza en la Figura 2.6 (Koldo, 2018).

De igual manera la red neuronal Madaline fue creada por Bernard Widrow y su estructura es la unión de dos o más Adaline interconectadas.

Su funcionalidad ha sufrido diversas variaciones con el paso de los años en estas dos redes neuronales artificial cambiando su manera de aprendizaje, al igual que las aplicaciones en las cuales son utilizadas, siendo para la eliminación de ruido y reconocimiento de patrones de las señales (Acquatella, 2012).

Sin embargo, ambas redes tienen la misma limitación igual al Perceptron, la cual es que al poseer una sola capa su capacidad se ve limitada para el ajuste de pesos (Basoqain, 2008).

Figura 2.8 Estructura del Adaline



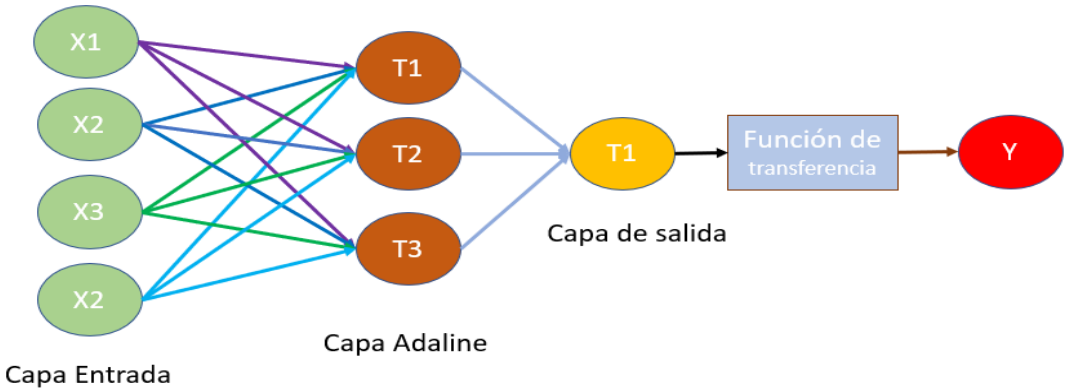
Estructura de una red neurona Adaline, Elaborado por: Joel Diaz

Para obtener el valor de salida se tiene que realizar un proceso para obtener el valor deseado, esto se consigue realizando una función del umbral, el cual determina si el valor es mayor o menos al umbral dando un resultado de -1 o 1

La estructura de Madaline está formado por Adaline, por ende, posen un sistema similar a diferencia que todos los Adaline están interconectadas, como muestra la Figura 2.7 se

observa cómo están conectados los Adaline siendo una capa secundaria y estas se conectan a una capa de salida la cual se denomina Madaline.

Figura 2.9 Estructura del Madaline



Estructura de una red neurona Madaline, Elaborado por: Joel Diaz

Para entrenar la red Madaline utiliza un sistema parecido a como se entrena el Perceptron, realiza un conjunto de patrones tomando en cuenta la matriz de entrada y la de salida, evalúa en cada interacción con la salida y si este cumple almacena el dato hasta obtener todos los pesos de la red neuronal tomando en cuenta el bias, caso contrario esta repite el proceso hasta llegar a la salida deseada.

CAPITULO 3
DISEÑO E IMPLANTACIÓN DEL ROBOT AUTÓNOMO MODO SEGUIDOR DE LÍNEA DESTREZA

3.1 Diseño y construcción del robot autónomo.

Para el diseño del robot se pensó en una estructura simple y de bajo costo y según las normas del CER (Concurso Ecuatoriano de Robótica) que no sobre pase el volumen de 15x15x15 cm del robot, con la función de que permita manejarse en las categorías de seguidor de línea de destreza y resuelve laberinto.

El diseño consiste en un sistema de dos ruedas y una rueda loca, este es muy usado en los robots de velocidad de seguidor de línea, no obstante, tiene restricciones a la hora de girar, además de que el robot también debe poderse manejarse en modo laberinto por esta razón se opta por un sistema de cuatro ruedas, cada una de estas tendrán su propio controlador.

Para tener un mejor de maniobrabilidad se decide utilizar ruedas omnidireccionales, estas permitirán una movilidad en todas direcciones, lo cual permite un control más versátil, en modo laberinto estas ruedas ayudan ha mitigar contactos con los muros.

En la distribución de los sensores se incorporan cinco frontales de tipo infrarrojos para la detección de línea, son colocados se forma paralela al robot con el fin de detectar la localización de la línea, para la detección cada sensor es calibrado de manera manual con el potenciómetro incorporado en los mismos, estos sensores poseen un convertidor analógico digital, lo cuales permiten enviar un señal de 0 o 5 voltios si detecta la luz infrarroja, esto permite utilizar las entradas digitales del microcontrolador Raspberry Pi Pico.

Para medir mejores distancias de recorrido se utiliza sensores ultrasónicos, ya que son más precisos y su detección no se ve afectados por la luz ambiental, no obstante, se debe tomar en cuenta que no detectan objetos blandos.

3.1.1 Sistema Sensorial

El sistema sensorial que utiliza el robot modo laberinto son los sensores HC-SR04, son sensores ultrasónicos como se muestra en la Figura 3.1, son de este tipo ya que son muy precisos y los factores ambientales no perturban su lectura, la desventaja es que su

funcionamiento falla cuando debe detectar objetos blandos, sin embargo, los laberintos son de materiales sólidos como la madera lo que es perfecto para este tipo de sensores.

Figura 3.1 Sensor ultrasónico HC-SR04

HC-SR04, fuente (Cytron, 2013)

Para el robot en modo seguidor de línea, se utilizan sensores ópticos digitales como se muestra en la Figura 3.2 para la lectura de la línea, sea negra o blanca en este caso se utiliza para la línea negra este tipo de sensores son los HW-201, normalmente se utilizan para la detección de objetos cercanos, no obstante, su funcionamiento es similar que los sensores utilizados habitualmente en este tipo de robot, se escogieron estos porque su detección es más precisa y no son muy afectados por la iluminación.

Figura 3.2 Sensor infrarrojo HW-201

Sensor infrarrojo HW-201, fuente (Llamas, 2016)

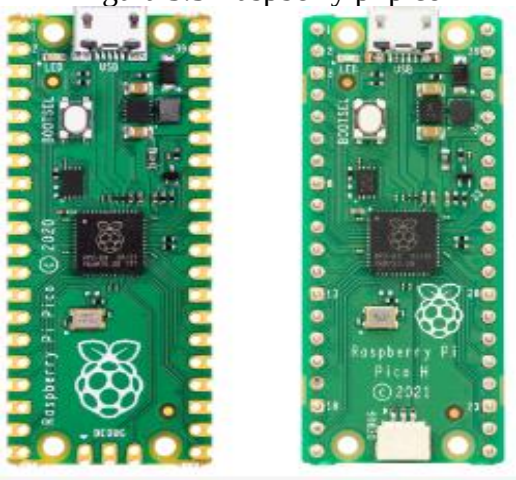
3.1.2 Sistema de control

Para el sistema de control se utilizó Raspberry pi pico como se muestra en la Figura 3.3

La principal razón para utilizar este chip fue su procesador y el número de entradas y salidas siendo 26, esto permite conectar todos los sistemas de control del puente H y los sensores infrarrojos y ultrasónicos, no obstante, al manejarse con valores de 3.3 voltios ocasionó problemas con los sensores, debido a que estos funcionan con 5 voltios.

A pesar de que la Raspberry pi pico posee su propio IDE de programación siendo una versión Python reducida llamada MicroPython, no se utilizó debido a que no poseen librerías que faciliten el código por este motivo se decidió utilizar IDE de Arduino, se crearon librería para el control de motores y sensores.

Figura 3.3 Raspberry pi pico

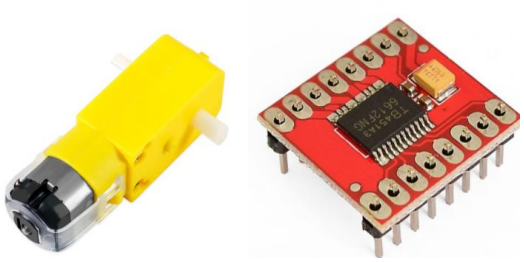


Raspberry pi pico, fuente (raspberry, 2021)

3.1.3 Motores y ruedas

Los motores utilizados son motores reductores de 12 voltios, poseen mayor fuerza y un consumo de corriente bajo, esto permite ahorrar carga de la batería. Para su control se utilizó el Driver TB6612fng permitiendo el control de dos motores con PWM y direccionamiento digital para cada motor, se utilizan para el control de cuatro motores, además de estas ventajas sus dimisiones permiten reducir el tamaño del robot como se indica en la Figura 3.4.

Figura 3.4 Motor 12 voltios con Driver TB6612fng



Motor Dc con caja reductora con Driver TB6612fng, fuente (Naylamp, 2021)

Las ruedas utilizadas son omnidireccionales, Estas permiten no solo desplazarse hacia delante o atrás, sino hacia la derecha o izquierda, como se observa en la Figura 3.5, con estas combinaciones se puede mover el robot en todas direcciones, para girar hacia adelante ambas ruedas deben girar hacia al frente y de igual manera para ir hacia atrás, en el caso que se requiera ir hacia la izquierda se debe direccionar las ruedas de la izquierda hacia adentro y los de la derecha hacia fuera, se aplica la misma lógica para ir hacia la derecha. Para su perfecto funcionamiento se debe tener en cuenta que las cuatro ruedas deben estar bien niveladas caso contrario sus movimientos no serán los esperados.

Figura 3.5 llanta omnidireccional

Llanta omnidireccional MECANUM Derecha, fuente (Electro Estore, 2021)

3.1.4 Fuente de alimentación del robot

Como fuente se tiene una batería tipo LI-PO de dos celdas de 7.4 voltios de 500 mA, estas batería están compuestas de litio y polímero, su principal característica es que son de tamaño reducido y de alta capacidad, no obstante su tiempo de vida es de alrededor de 3 años, se debe tomar en cuenta que las baterías son altamente explosivas, por eso es

esencial respetar los tiempos de carga y con sus cargadores específicos para este tipo de baterías, cada celda pose 3.7 voltios al ser de dos celadas su voltaje final es de 7.3 voltios.

3.1.5 Diseño esquemático

Para realizar el circuito esquemático se utiliza el programa de diseño eléctrico Proteus, en este se elabora diagrama de conexiones utilizando como microcontrolador la antes mencionada Raspberry pi pico, a continuación se indica la conexión de los pines con los sensores y actuadores como se muestra en la Tabla 3.1.

Tabla 3.1 Pines de conexión del microcontrolador.

Pines digitales del controlador	Pines de sensores y actuadores.	
GP0	TRIGGER_sensor_1	Sensor ultrasónico izquierda
GP1	ECHO_sensor_1	
GP2	TRIGGER_sensor_2	Sensor ultrasónico centro
GP3	ECHO_sensor_2	
GP4	TRIGGER_sensor_3	Sensor ultrasónico derecha
GP5	ECHO_sensor_3	
GP6	I_PWM_1	Controlador de motores (puente h) de lado izquierdo frontal y posterior
GP7	I_In_B_pin_1	
GP8	I_PWM_1	
GP9	I_pin_STBY	
GP10	I_In_A_pin_2	
GP11	I_In_B_pin_2	
GP12	I_PWM_2	
GP13	D_PWM_1	Controlador de motores (puente h) de lado derecho frontal y posterior
GP14	D_in_B_pin_1	
GP15	D_in_A_pin_1	
GP16	D_pin_STBY	
GP17	D_in_B_pin2	
GP18	D_in_A_pin_2	

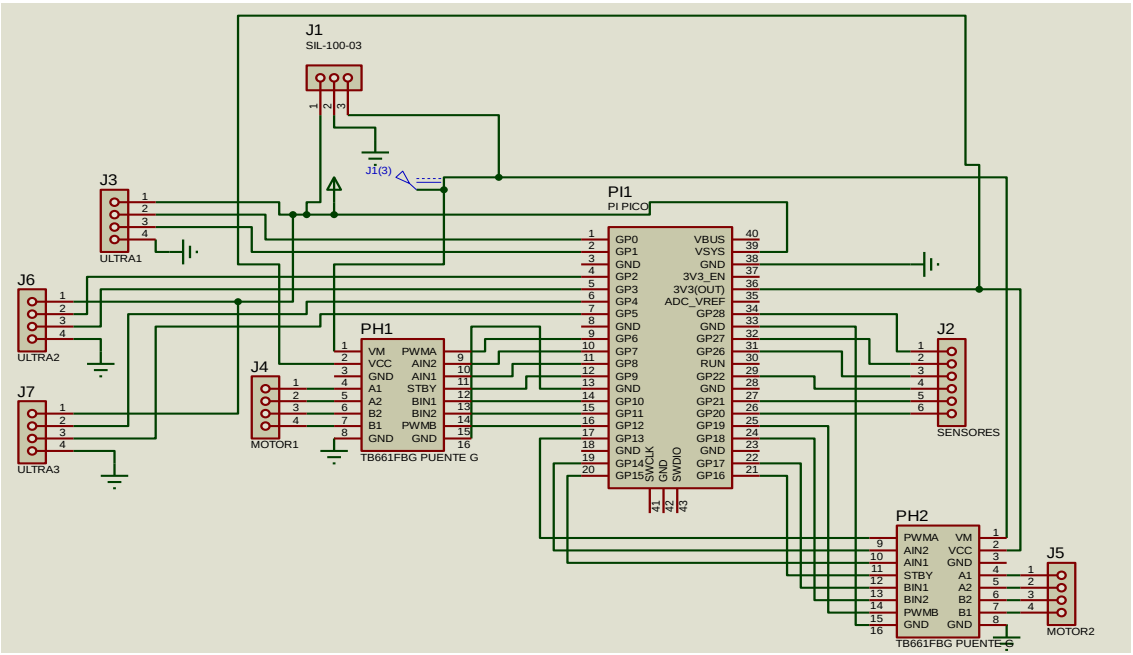
GP19	D_PWM_2	
GP20	Sensor_óptico_1	Entrada de los pines digitales de los sensores infrarrojos
GP21	Sensor_óptico_2	
GP22	Sensor_óptico_3	
GP26	Sensor_óptico_4	
GP27	Sensor_óptico_5	
GP28	Sensor_óptico_6	

Pines de conexión del microcontrolador, Elaborado por: Joel Diaz

Los sensores ultrasónicos funcionan con 5 voltios, por este motivo se debe regular desde la batería con un sistema reductor de voltaje, para que entregue 5 voltios de alimentaran a los sensores infrarrojos y los ultrasónicos.

Para las conexiones de los motores estos van enlazados a sus respectivos controladores como muestra la Figura 3.6, el cual se ingresa un voltaje de entrada directamente de la batería. Tener en cuenta que voltaje soportan estos drivers en caso de que se requiera motores de mayor consumo. Además de que se observa como está realizada la conexión de todos los componentes según la Tabla 3.1.

Figura 3.6 Diseño esquemático.

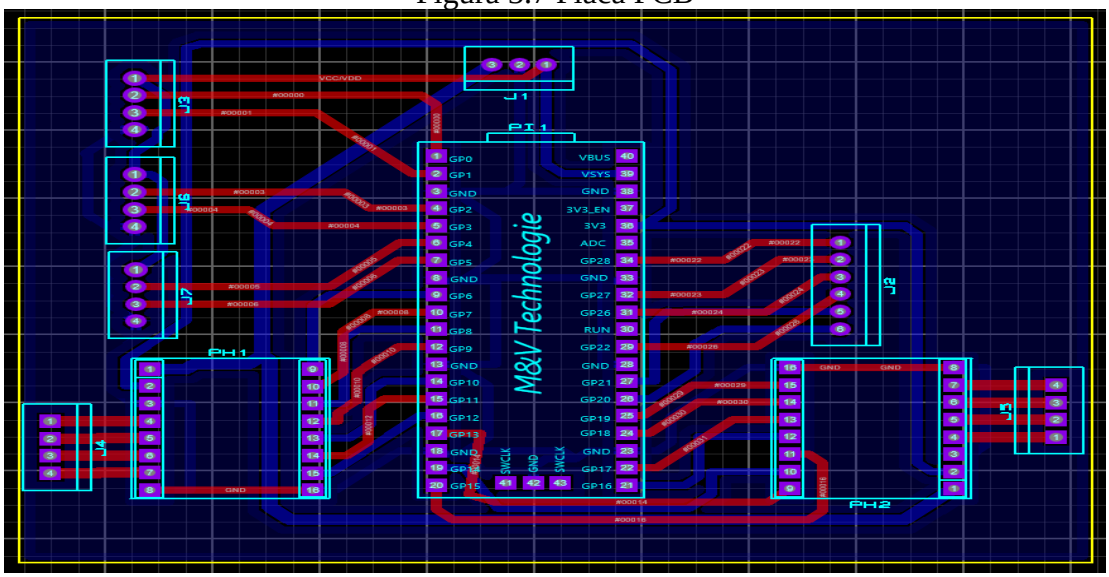


Diseño eléctrico, fuente Joel Diaz

Para el diseño de la placa PCB en necesario la utilización de una PCB de dos capas para reducir el tamaño de la placa, además aislar las pistas con GND para evitar errores de lectura en los sensores como muestra la Figura 3.7, además se puede observar que se utilizan pistas de mayor grosor para la conexión de los motores.

Para realizar la PCB se utiliza una baquelita de dos capas y se imprime en papel fotográfico el diseño de los circuitos para los dos lados, mediante el método de planchado se obtiene la PCB con sus pistas diseñadas, se realizan sus respectivas perforaciones y se ensambla cada componente en sus respectivos lugares.

Figura 3.7 Placa PCB

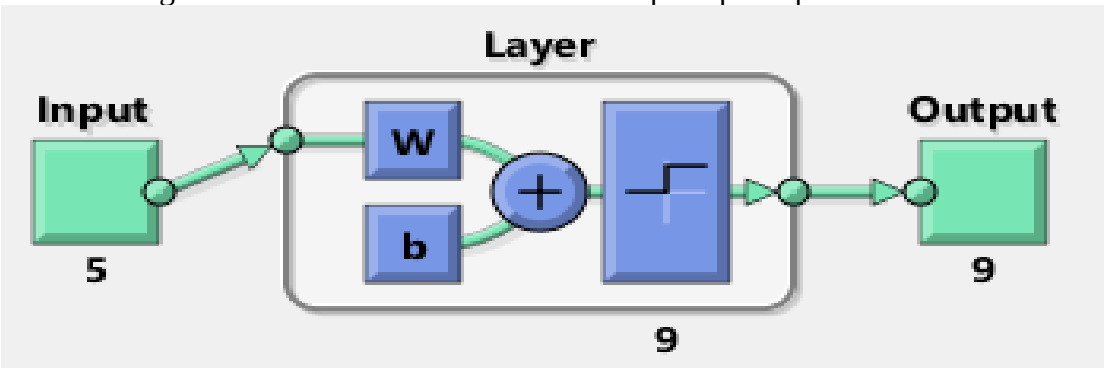


Placa PCB, fuente Joel Diaz

3.2 Diseño de la red neuronal Artificial para el aprendizaje del robot seguidor de línea de destreza.

La estructura de la red neuronal artificial al ser un perceptrón solo pose dos capas una de entrada y salida, para el caso de las entradas son 5 por los sensores utilizados y la salida son 9, estos permitirán identificar en qué lugar se encuentra la línea y hallar el error como se mutra en la Figura 3.8

Figura 3.8 Estructura de la red neuronal perceptrón para el R.M.S



Estructura del Perceptron para R.M.S, Elaborado por: Joel Diaz

El robot seguidor de línea pose 5 entradas digital las cuales se encuentran en la parte frontal, estos detectan la intensidad de luz infrarroja que rebota dependiendo del color de la superficie, los colores oscuros en este caso el negro bloquea el paso de luz, de esta manera se obtiene una señal digital de 0 o 1 dependiendo de cómo este configurado el transductor de cada sensor infrarrojo, en este caso si detecta color blanco da un valor de 1 y negro una valor de 0, con estos valore se crea una matriz de entrada como se muestra en la Figura 3.9.

La matriz de entradas está compuesta por un array de 5 números de 0 o 1 que determinan la ubicación de la línea, tomando esto en cuenta se platean 13 tipos de combinaciones, que su función es direccionar el robot tomando en cuenta la ubicación de la línea de la pista, para mejorar el aprendizaje cada combinación se repite mínimo 4 veces para cada salida.

Figura 3.9 Matriz de entrada R.M.S

```
Entrada=[1 1 1 1 0;1 1 1 1 0;1 1 1 1 0;1 1 1 1 0;1 1 0 0 0;1 1 0 0 0;1 1 0 0 0;1 1 0 0 0;
         1 1 1 0 0;1 1 1 0 0;1 1 1 0 0;1 1 1 0 0;1 1 1 0 0;1 1 1 0 0;1 1 1 0 0;1 1 1 0 0;
         1 1 1 0 1;1 1 1 0 1;1 1 1 0 1;1 1 1 0 1;1 1 1 0 1;1 1 1 0 1;1 1 1 0 1;1 1 1 0 1;
         1 1 0 0 1;1 1 0 0 1;1 1 0 0 1;1 1 0 0 1;1 1 0 0 1;1 1 0 0 1;1 1 0 0 1;1 1 0 0 1;
         1 1 0 1 1;1 1 0 1 1;1 1 0 1 1;1 1 0 1 1;1 1 0 1 1;1 1 0 1 1;1 1 0 1 1;1 1 0 1 1;
         1 0 0 1 1;1 0 0 1 1;1 0 0 1 1;1 0 0 1 1;1 0 0 1 1;1 0 0 1 1;1 0 0 1 1;1 0 0 1 1;
         1 0 1 1 1;1 0 1 1 1;1 0 1 1 1;1 0 1 1 1;1 0 1 1 1;1 0 1 1 1;1 0 1 1 1;1 0 1 1 1;
         0 0 1 1 1;0 0 1 1 1;0 0 1 1 1;0 0 1 1 1;0 0 1 1 1;0 0 1 1 1;0 0 1 1 1;0 0 1 1 1;
         0 1 1 1 1;0 1 1 1 1;0 1 1 1 1;0 1 1 1 1;0 0 -1 1 1;0 0 0 1 1;0 0 0 1 1;0 0 0 1 1
```

Matriz de entrada R.M.S, Elaborado por: Joel Dia

Para la salida de la red neuronal se desea obtener valores lineales de entre -4 a 4, con el fin de que estos valores entran en un sistema de control en este caso un PID, este se encargará de direccionar y controlar la velocidad de los motores para realizar los giros dependiendo de la entrada de la red neuronal, sin embargo, se realizó pruebas para identificar que valores de salida serían los idóneo para el pleno aprendizaje de la red neuronal, se realizó con salidas de números enteros para que sea más fácil el ingreso al control PID, para esto se utilizó la red neuronal Backpropagation ya que su capa oculta ayudaría a un aprendizaje eficiente, no obstante al ingresar los valores al toolbox de Matlab la red no aprendió y se tuvieron varios errores, por este motivo se decide utilizar salidas entre 0 y 1 con el fin de acoplar a una red neuronal Perceptrón como se muestran en la Figura 3.10

Figura 3.10 Matriz de salida R.M.S

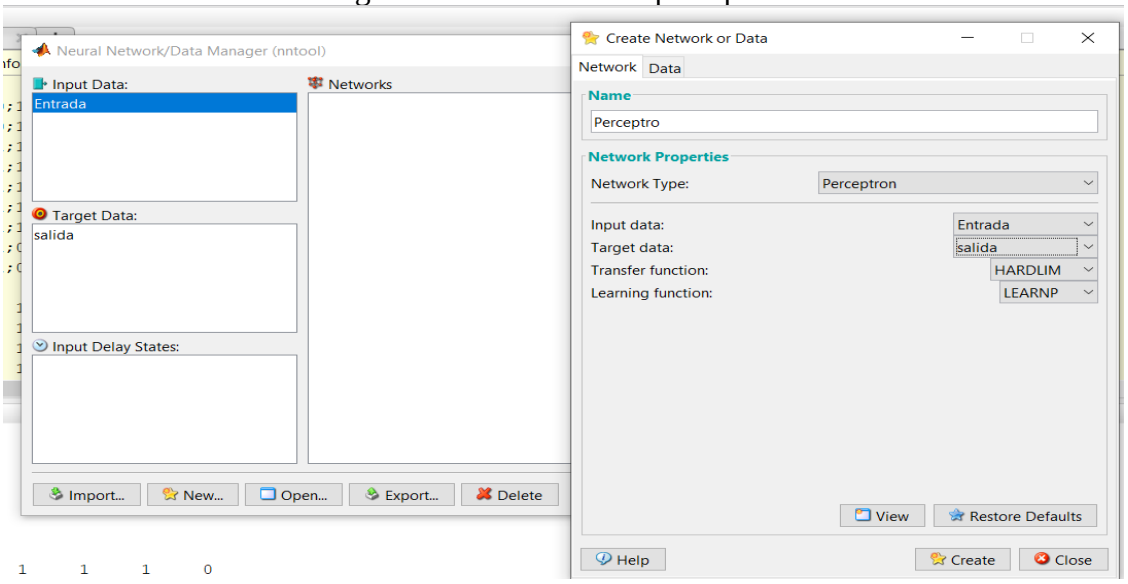
```
salida=[ 1 1 1 1 1 1 1 1 0 ; 1 1 1 1 1 1 1 1 0 ; 1 1 1 1 1 1 1 1 0 ; 1 1 1 1 1 1 1 1 0 ;
         1 1 1 1 1 1 1 0 1 ; 1 1 1 1 1 1 1 0 1 ; 1 1 1 1 1 1 1 0 1 ; 1 1 1 1 1 1 1 0 1 ;
         1 1 1 1 1 1 0 1 1 ; 1 1 1 1 1 1 0 1 1 ; 1 1 1 1 1 1 0 1 1 ; 1 1 1 1 1 1 0 1 1 ;
         1 1 1 1 1 0 1 1 1 ; 1 1 1 1 1 0 1 1 1 ; 1 1 1 1 1 0 1 1 1 ; 1 1 1 1 1 0 1 1 1 ;
         1 1 1 1 0 1 1 1 1 ; 1 1 1 1 0 1 1 1 1 ; 1 1 1 1 0 1 1 1 1 ; 1 1 1 1 0 1 1 1 1 ;
         1 1 1 0 1 1 1 1 1 ; 1 1 1 0 1 1 1 1 1 ; 1 1 1 0 1 1 1 1 1 ; 1 1 1 0 1 1 1 1 1 ;
         1 1 0 1 1 1 1 1 1 ; 1 1 0 1 1 1 1 1 1 ; 1 1 0 1 1 1 1 1 1 ; 1 1 0 1 1 1 1 1 1 ;
         1 0 1 1 1 1 1 1 1 ; 1 0 1 1 1 1 1 1 1 ; 1 0 1 1 1 1 1 1 1 ; 1 0 1 1 1 1 1 1 1 ;
         0 1 1 1 1 1 1 1 1 ; 0 1 1 1 1 1 1 1 1 ; 0 1 1 1 1 1 1 1 1 ; 0 1 1 1 1 1 1 1 1 ;
```

Matriz de salida R.M.S, Elaborado por: Joel Diaz

3.3 Creación de la red Neuronal perceptrón modo seguidor de línea.

Para la creación de red neuronal se utiliza el toolbox de Matlab tomando en cuenta la matriz de entrada y salida de las Figuras 3.9 y 3.10, estos valores ingresan en toolbox y crear la red neuronal Perceptrón como se indica la Figura 3.11.

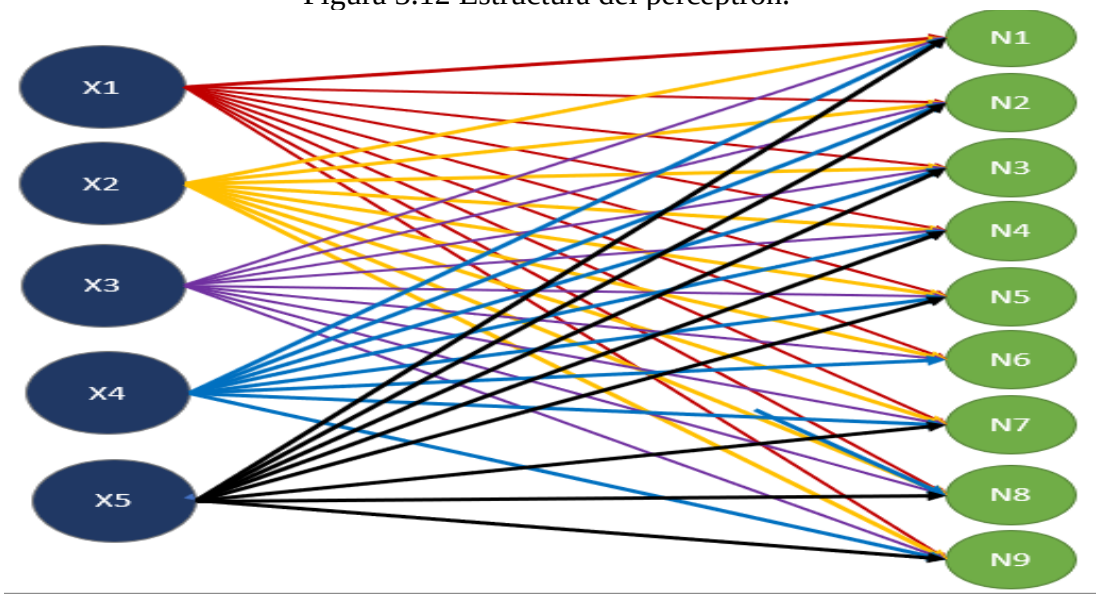
Figura 3.11 Creación del perceptrón.



Ajustes de perceptrón, Elaborado por: Joel Diaz

Al ser un perceptrón es una red mono capa obtenida una estructura mostrada en la Figura 3.12, se puede ver que se ingresan cinco entradas proviene de los sensores ópticos, los cuales se multiplican con los pesos y se suman los bias obteniendo la salida esperada en este caso se tiene 9 salidas, porque se necesitan valores -4 a 4 siendo una salida línea de 9 valores, debido a que el perceptrón busca el 0 se crean 9 salidas binarias.

Figura 3.12 Estructura del perceptrón.



Estructura de perceptrón, Elaborado por: Joel Diaz

Una vez obtenidos los valores se procede a realizar la fase de consulta, siguiendo la estructura de la red se procede a programar y realizar pruebas de funcionamiento, como se puede observar en la Figura 3.13 se insertan los valores obtenidos del entrenamiento del perceptrón.

Figura 3.13 Valores de consulta del perceptrón

```
%% pesos de la capa 1
wc1= [7 -4 3 -1 -2;
 1 4 -1 0 0;
 -4 4 -2 0 0;
 -2 1 4 -1 -1;
 1 -4 3 -1 -1;
 -1 -1 1 4 -1;
 1 1 -4 3 -2;
 0 0 -1 4 4;
 -1 -1 3 -4 7];
```

```
 %% bias 1
b1=[0;
 0;
 5;
 3;
 4;
 2;
 3;
 0;
 -1];
```

Valores del perceptrón, Elaborado por: Joel Diaz

Para hallar la salida se crea una multiplicación de matrices mostrada en la Figura 3.14, se observa como cada entrada se multiplica con un valor del peso y la suma de estos se le incrementa el bias para cada salida, como se tiene 9 salidas se multiplica 9 veces cada entrada para determinar cada salida.

Figura 3.14 obtención de los pesos de la red artificial

```
T1= ((entrada(1)*wc1(1,1))+   (entrada(2)*wc1(1,2))+ (entrada(3)*wc1(1,3))+ (entrada(4)*wc1(1,4))+(entrada(5)*wc1(1,5)) + b1(1));
T2= ((entrada(1)*wc1(2,1))+   (entrada(2)*wc1(2,2))+ (entrada(3)*wc1(2,3))+ (entrada(4)*wc1(2,4))+(entrada(5)*wc1(2,5)) + b1(2));
T3= ((entrada(1)*wc1(3,1))+   (entrada(2)*wc1(3,2))+ (entrada(3)*wc1(3,3))+ (entrada(4)*wc1(3,4))+(entrada(5)*wc1(3,5)) + b1(3));
T4= ((entrada(1)*wc1(4,1))+   (entrada(2)*wc1(4,2))+ (entrada(3)*wc1(4,3))+ (entrada(4)*wc1(4,4))+(entrada(5)*wc1(4,5)) + b1(4));
T5= ((entrada(1)*wc1(5,1))+   (entrada(2)*wc1(5,2))+ (entrada(3)*wc1(5,3))+ (entrada(4)*wc1(5,4))+(entrada(5)*wc1(5,5)) + b1(5));
T6= ((entrada(1)*wc1(6,1))+   (entrada(2)*wc1(6,2))+ (entrada(3)*wc1(6,3))+ (entrada(4)*wc1(6,4))+(entrada(5)*wc1(6,5)) + b1(6));
T7= ((entrada(1)*wc1(7,1))+   (entrada(2)*wc1(7,2))+ (entrada(3)*wc1(7,3))+ (entrada(4)*wc1(7,4))+(entrada(5)*wc1(7,5)) + b1(7));
T8= ((entrada(1)*wc1(8,1))+   (entrada(2)*wc1(8,2))+ (entrada(3)*wc1(8,3))+ (entrada(4)*wc1(8,4))+(entrada(5)*wc1(8,5)) + b1(8));
T9= ((entrada(1)*wc1(9,1))+   (entrada(2)*wc1(9,2))+ (entrada(3)*wc1(9,3))+ (entrada(4)*wc1(9,4))+(entrada(5)*wc1(9,5)) + b1(9));
```

obtención de los pesos de la red artificial, Elaborado por: Joel Diaz

Se realizan pruebas con cada una de las entradas y se obtiene resultados esperados, dando las salidas que se desea como se muestra la Figura 3.15, sin embargo, este resultado es número binario y no un entero por ende se debe acondicionar la salida para que convierta este número binario en número entero lineal de -4 a 4 para el análisis del error del sistema PID.

Figura 3.15 Respuesta del perceptrón

```
Ingrese 5 datos a la entrada para consultar:
Elemento(1): 1
Elemento(2): 1
Elemento(3): 1
Elemento(4): 1
Elemento(5): 0

Respuesta =

     1     1     1     1     1     1     1     1     0
```

Fase de consulta perceptrón, Elaborado por: Joel Diaz

3.4 Programación del microcontrolador para robot modo seguidor de línea de destreza

El controlador utilizado es la raspberry pi pico debido a su velocidad de procesamiento y los número de pines digitales que posee, esto permite controlar los drivers de motores y los sensores, este microcontrolador tiene su proprio IDE para su programación, no obstante, al ser una nueva versión de raspberry utiliza una versión reducida Python llamada MicroPython, pero al ser nueva no dispone de muchas librerías por este motivo se utiliza IDE de Arduino, ya que este reduce la programación y es compatible con este microcontrolador, utilizando las librerías que ayudan a facilitar el código.
como muestra la Figura 3.16.

Figura 3.16 subrutina de la red neuronal

```
  entrada[1]=x1;
  entrada[2]=x2;
  entrada[3]=x3;
  entrada[4]=x4;
  entrada[5]=x5;
int Respuesta=0;
T1= ((entrada[1]*( 7)) +  (entrada[2]*(-4)) + (entrada[3]*(3 )) +  (entrada[4]*(-1)) + (entrada[5]*(-2)) + 0);
T2= ((entrada[1]*( 1)) +  (entrada[2]*(4 )) + (entrada[3]*(-1)) +  (entrada[4]*( 0)) + (entrada[5]*(0 )) + 0);
T3= ((entrada[1]*(-4)) +  (entrada[2]*(4 )) + (entrada[3]*(-2)) +  (entrada[4]*(0 )) + (entrada[5]*(0 )) + 5);
T4= ((entrada[1]*(-2)) +  (entrada[2]*(1 )) + (entrada[3]*(4 )) +  (entrada[4]*(-1)) + (entrada[5]*(-1)) + 3);
T5= ((entrada[1]*(1 )) +  (entrada[2]*(-4)) + (entrada[3]*(3 )) +  (entrada[4]*(-1)) + (entrada[5]*(-1)) + 4);
T6= ((entrada[1]*(-1)) +  (entrada[2]*(-1)) + (entrada[3]*(1 )) +  (entrada[4]*(4 )) + (entrada[5]*(-1)) + 2);
T7= ((entrada[1]*(1 )) +  (entrada[2]*(1 )) + (entrada[3]*(-4)) +  (entrada[4]*(3 )) + (entrada[5]*(-2)) + 3);
T8= ((entrada[1]*(0 )) +  (entrada[2]*(0 )) + (entrada[3]*(-1)) +  (entrada[4]*(4 )) + (entrada[5]*(4 )) + 0);
T9= ((entrada[1]*(-1)) +  (entrada[2]*(-1)) + (entrada[3]*(3 )) +  (entrada[4]*(-4)) + (entrada[5]*(7 )) - 1);
```

subrutina de la red neuronal, Elaborado por: Joel Diaz

Se puede observar que es el mismo código de consulta que se obtuvo del programa de Matlab, sin embargo, se debe incorporar la fase de acondicionamiento y que la salida de la red es un valor binario, para esto se crean condiciones en las cuales se determina un número entero para cada número binario que proporciona la salida de perceptrón como se muestra en la Figura 3.17

Figura 3.17 Acondicionamiento de la salida R.M.S

```
if((T1==1) && (T2==1) && (T3 ==1) && (T4==1) && (T5==1) && (T6==1) && (T7==1) && (T8==1) && (T9==0)){
Respuesta=4;
}if((T1==1) && (T2==1) && (T3 ==1) && (T4==1) && (T5==1) && (T6==1) && (T7==1) && (T8==0) && (T9==1)){
Respuesta=3;
} if((T1==1) && (T2==1) && (T3 ==1) && (T4==1) && (T5==1) && (T6==1) && (T7==0) && (T8==1) && (T9==1)){
Respuesta=2;
} if((T1==1) && (T2==1) && (T3 ==1) && (T4==1) && (T5==1) && (T6==0) && (T7==1) && (T8==1) && (T9==1)){
Respuesta=1;
} if((T1==1) && (T2==1) && (T3 ==1) && (T4==1) && (T5==0) && (T6==1) && (T7==1) && (T8==1) && (T9==1)){
Respuesta=0;
} if((T1==1) && (T2==1) && (T3 ==1) && (T4==0) && (T5==1) && (T6==1) && (T7==1) && (T8==1) && (T9==1)){
Respuesta=-1;
} if((T1==1) && (T2==1) && (T3 ==0) && (T4==1) && (T5==1) && (T6==1) && (T7==1) && (T8==1) && (T9==1)){
Respuesta=-2;
} if((T1==1) && (T2==0) && (T3 ==1) && (T4==1) && (T5==1) && (T6==1) && (T7==1) && (T8==1) && (T9==1)){
Respuesta=-3;
} if((T1==0) && (T2==1) && (T3 ==1) && (T4==1) && (T5==1) && (T6==1) && (T7==1) && (T8==1) && (T9==1)){
Respuesta=-4;
}
return Respuesta;
```

Acondicionamiento de la salida, Elaborado por: Joel Diaz

Una vez obtenidos los valores en número enteros de -4 a 4 se ingresan al PID, ya que estos valores son el error detectado por los sensores siendo -4 cuando esta al extremo izquierdo, 4 al extremo derecho y el error será 0 cuando el robot este en el centro de la línea, de esta manera se determina el error para el manejo del robot seguidor de línea de destreza.

3.5 Programación del PID para el robot seguidor línea de destreza.

Con el error obtenido de la red neuronal se procede a elaborar un sistema PID para el control del robot, el sistema PID maneja tres parámetros como se visualiza en la Figura 3.13 los cuales son:

- Posición: este valor es entregado por la salida acondicionada de la red neuronal, también conocido como el error de posición.
- Set point: es el número del cual partirá el PID por defecto es 0, si cambia este valor se pueden obtener resultados más altos lo cual se podrían usar para controlar la variación de datos del PID
- Límite: el límite ayuda a establecer la cantidad máxima de PID.

Estos datos controlaran la variación del PWM de los motores, siendo los valores de entre 0 a 255 como se muestra en la Figura 3.18.

Figura 3.18 PID R.M.S

```
long PID(int POS, int setpoint, int lim) {

  proportional = int(POS) - setpoint;
  derivative = proportional - last_proportional;
  last_proportional = proportional;
  int power_difference = (proportional * KP) + (derivative * KD);

  if (power_difference > lim)
    power_difference = lim;
  else if (power_difference < -lim)
    power_difference = -lim;

  return power_difference;
```

PID de la red neuronal, Elaborado por: Joel Diaz

Una vez obtenido el valor de PID dependiendo del error de posición como muestra en la Figura 3.8, se programa la señal PWM de los motores para esto se utiliza un valor base, este se sumará o se restará si es para el control de los motores izquierdo o derecho como se muestra en la Figura 3.19.

Figura 3.19 Control de motores R.M.S

```
int posicion = map(salida, -4, 4, -255, 255);

int pid=PID( posicion, setpoint, velocidad_giro);

vel_izquierda= (base +pid );
vel_derecha= (base -pid );


DmotorGo(0,1,vel_derecha);//1er motor
DmotorGo(1,1,vel_derecha);//2do motor
ImotorGo(0,1,vel_izquierda);//1er motor
ImotorGo(1,1,vel_izquierda);//2do motor
```

Control de motores, Elaborado por: Joel Diaz

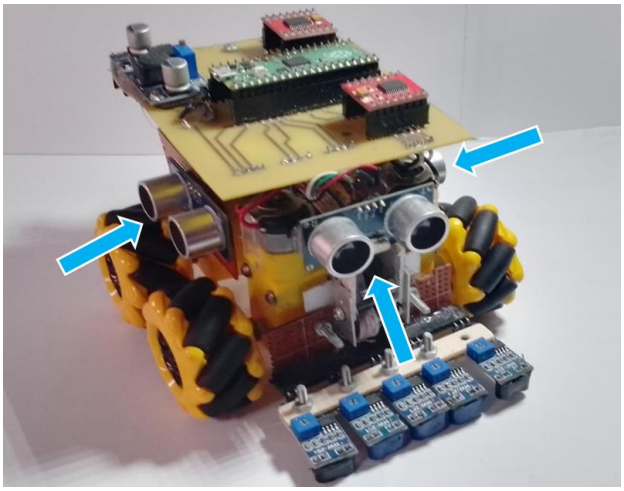
Gracias a la red neuronal se puede establecer diferentes parámetros de entrada, esto ayuda a no cambiar todo el algoritmo de programación, ya que solo se entrena la red neuronal, con estos nuevos valores para obtener los resultados esperados, para esto basta con cambiar los pesos obtenidos del aprendizaje.

DISEÑO E IMPLEMENTACIÓN DEL ROBOT AUTÓNOMO MODO LABERINTO

3.6 Diseño y creación de la red neuronal Adaline.

Para el diseño de la red se debe tomar en cuenta las entradas y las salidas que requiere, en el caso de robot modo laberinto utiliza 3 sensores ultrasónicos localizados en los extremos del robot como muestra la Figura 3.20, estos se encargan de medir las distancias y proporcionar estos datos a la red neuronal.

Figura 3.20 Localización de los sensores ultrasónicos.



Robot seguidor de línea y laberinto, Elaborado por: Joel Diaz

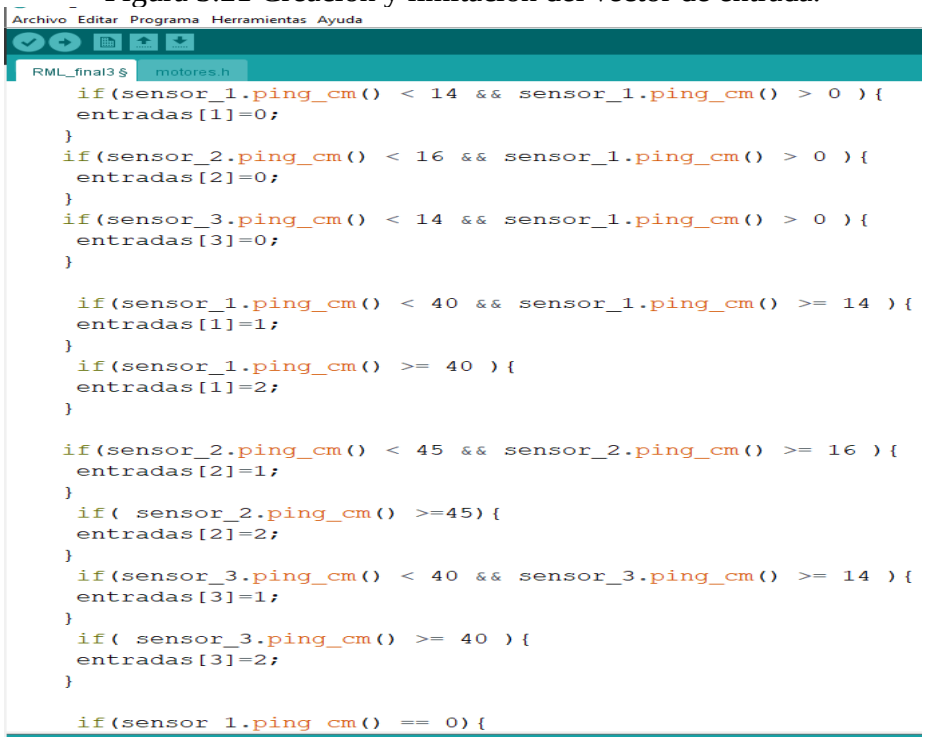
Para ingresar los valores a la red neuronal se estandarizan las medidas de los sensores ultrasónicos en tres valores de la siguiente manera:

- Para mediciones de 0 a 14 cm se determina el valor de 0, esto significa que detecta un muero sin espacios.
- En el rango de 14 a 40 cm el valor asignado será 1, lo que implica que tiene un espacio de hasta detectar la pared.
- Por último, para mayores 40 cm se asigna el número 2, utilizado para identificar que no detecta colisión,

Se debe tomar en cuenta que se utiliza una condición debido a un error con los sensores, ya que su funcionamiento es de 5 voltios y el microcontrolador funciona con 3.3 voltios, por este motivo cuando supera un rango de detección no recibe señal, para solventar este error se coloca una condición con el fin de que, si existe este error su resultado sea 2, así el funcionamiento de los sensores es correcto como muestra la Figura 3.21.

Para la salida solo se requiere una neurona debido a que esta entregará valores enteros de entre -4 a 4, los cuales permitirán al PWM controlar los motores y realizar el control.

Figura 3.21 Creación y limitación del vector de entrada.



Programación de limitación de entradas, Elaborado por: Joel Diaz

Una vez obtenido el vector de entrada se crea una tabla con los datos de aprendizaje, en esta se ingresan los diferentes casos en el que robot se encuentre y se los asocia con una salida, la salida será un número del 1 al 4 estos números corresponden a los estados que los motores deberán tomar, siendo 4 adelante, 3 derecha, 2 izquierda y 1 se usará para casos especiales como por ejemplo en el caso que se tenga las entradas 0 2 2 su salida correspondiente será 4, lo que significa que el robot se moverá hacia al frente, para los demás casos se puede ver en la Tabla 3.2.

Tabla 3.2 Matriz de entradas y salidas.

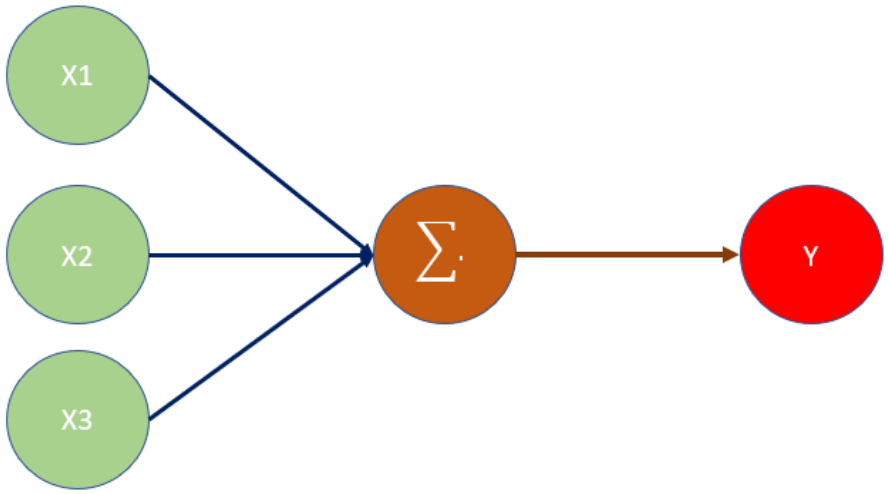
Sensor izquierdo	Sensor central	Sensor derecho	Salida
0	1	0	4
0	2	0	4
0	2	2	4
0	2	1	4
0	1	2	3
0	1	1	3

2	0	0	2
2	0	1	2
2	1	1	2
1	0	0	2

Matriz de entradas y salidas de la red neuronal Adaline, Elaborado por: Joel Diaz

Con estos datos se identificará cuantas neuronas se requiere para la creación de la red Adaline las cuales son tres neuronas en la capa de entrada y una en la salida como se muestra en la Figura 3.22.

Figura 3.22 Creación de la red neuronal Adaline.



Esquema de la red neuronal, Elaborado por: Joel Diaz

3.7 Diseño y Creación de la red neuronal Adaline.

Para la creación de la red neuronal se la utilizó el programa Matlab, primero se crea una matriz de entrada y un vector de salida con los datos de la Tabla 2.1, los parámetros de aprendizaje siendo LR de 0.005 y un bias de 1 como se muestra en la Figura 2.4, se crea un bucle While el que se romperá una vez que la red aprenda, dentro del ciclo se aplica la fórmula de Adaline para cada entrada con su respectivo peso como indica la Figura 2.3, se crea un ciclo for el cual se va encargar de ejecutar la fórmula para actualizar los pesos como se muestra en la Figura 3.23

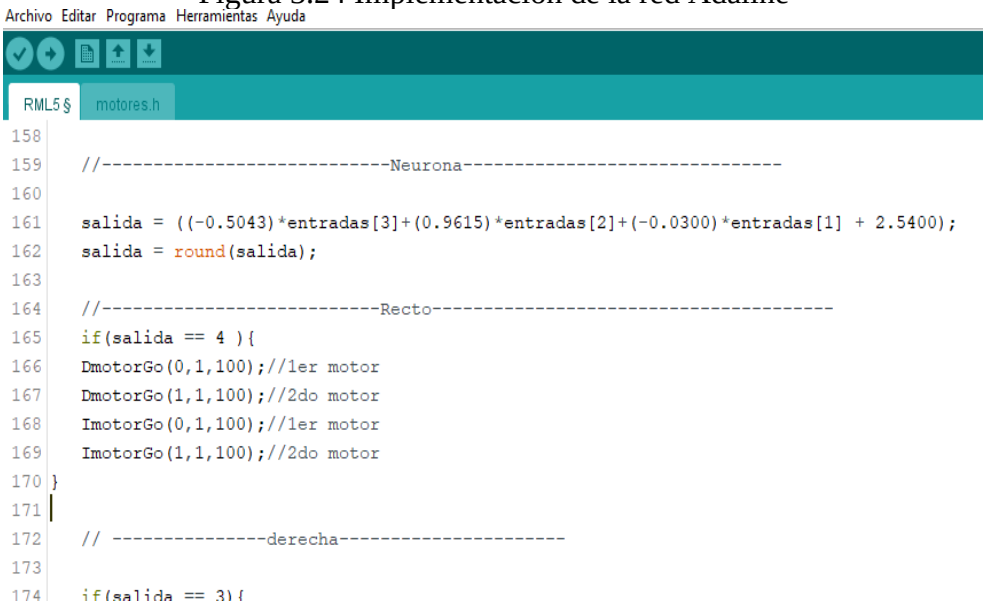
Figura 3.23 Creación de la red Adaline

```
clear all; clc;
Entradas=[0 1 0 ; 0 2 0; 0 2 2;0 2 1;0 1 2;0 1 1; 2 0 0;2 0 1;2 1 1; 1 0 0]
salida=[4 4 4 4  3 3 2 2 2 2];
LR= 0.005; var=1; suma=0; iteracion=0; bias=1;
p= (abs(0.5 - rand(3,1)))/10;

while var ==1
    for i= 1:10
        I(i) = ((Entradas(i,1)*p(1))+(Entradas(i,2)*p(2))+  (Entradas(i,3)*p(3))) + bias;
        I1(i) = round(I(i));

        p(1)= p(1) + (LR*(salida(i)-I1(i))*Entradas(i,1));
        p(2)= p(2) + (LR*(salida(i)-I1(i))*Entradas(i,2));
        p(3)= p(3) + (LR*(salida(i)-I1(i))*Entradas(i,3));

        bias= bias +(2*LR*(salida(i)-I1(i)));
        error(i)=abs(salida(i)-I1(i));
        suma= suma+error(i);

    end
     if suma==0
            disp('La neurona ya aprendio')
            var=10;
     end

     if suma ~= 0 %diferente de cero
            disp('La neurona no aprende todavia')
            var=1;
     end

        suma=0;
        iteracion=iteracion+1;
 end
```

Creación de la red Adaline, Elaborado por: Joel Diaz

La red neuronal artificial realizó 126 iteraciones para adquirir los pesos correspondientes para obtener la salida deseada para este caso se obtuvo en el peso 1 = -0.5043, el peso 2 = 0.9615, el peso 3 = -0.0300 y un bias de 2.5400, estos datos se ingresan a la programación del microcontrolador como se indica en la Figura 3.24.

Figura 3.24 Implementación de la red Adaline



Programación de la neurona artificial, Elaborado por: Joel Diaz

Una vez obtenida la ecuación correspondiente del entrenamiento de la red neuronal se procede a programar las funciones de los motores según el estado de la salida de la red neuronal, para el control de los motores se crea una librería que facilita su manejo, esta función se encarga de direccionar los motores mediante tres variables, la primera variable indica si el motor izquierdo o derecho, la segunda variable es la dirección y la tercera es la salida PWM que controla la velocidad de los motores.

Con la librería creada se programa para los diferentes casos tomando en cuenta la salida de la red, por ejemplo, si la salida de la red es 4 este debe ir hacia delante como se muestra en la Figura 3.25, se realiza la misma lógica para el direccionamiento si la salida es 3 corresponde a la derecha, si la salida es 2 debe girar a la izquierda, para esto se creó una librería que permita ingresar cual motor se va utilizar sea el delantero represando con 0 o trasero representado con 1, la dirección siendo 1 hacía al frente o 2 hacia atrás y por último la velocidad o PWM del motor.

Figura 3.25 Programación de dirección al frente.

RML5 § motores.h

```
185
186   }
187    //---------------------------Recto----
188    if(salida == 4 ){
189    DmotorGo(0,1,100);//1er motor
190    DmotorGo(1,1,100);//2do motor
191    ImotorGo(0,1,100);//1er motor
192    ImotorGo(1,1,100);//2do motor
```

Programación de direccionamiento, Elaborado por: Joel Diaz

Con la ayuda de los retardos se realizan los giros hacia la derecha o izquierda, una vez realizado el giro se programa al robot que avance para pasar a la siguiente evaluación de los sensores, de esta manera el robot se desplace y gire dependiendo a las entradas que ingresaron en la red neuronal como muestra la Figura 3.26.

Figura 3.26 Programación de dirección derecha.

```
if(salida == 3){
DmotorGo(0,2,0);//1er motor
DmotorGo(1,1,0);//2do motor
ImotorGo(0,2,0);//1er motor
ImotorGo(1,1,0);//2do motor
DmotorGo(0,1,60);//1er motor
DmotorGo(1,1,60);//2do motor
ImotorGo(0,1,60);//1er motor
ImotorGo(1,1,60);//2do motor
delay(550);
DmotorGo(0,2,0);//1er motor
DmotorGo(1,1,0);//2do motor
ImotorGo(0,2,0);//1er motor
ImotorGo(1,1,0);//2do motor
DmotorGo(0,2,60);//1er motor
DmotorGo(1,2,60);//2do motor
ImotorGo(0,1,60);//1er motor
ImotorGo(1,1,60);//2do motor
delay(600); // tiempo de giro
DmotorGo(0,2,0);//1er motor
DmotorGo(1,1,0);//2do motor
ImotorGo(0,2,0);//1er motor
ImotorGo(1,1,0);//2do motor
// avansas despues de girar
DmotorGo(0,1,60);//1er motor
DmotorGo(1,1,60);//2do motor
ImotorGo(0,1,60);//1er motor
ImotorGo(1,1,60);//2do motor
delay(1100);
DmotorGo(0,2,0);//1er motor
DmotorGo(1,1,0);//2do motor
ImotorGo(0,2,0);//1er motor
ImotorGo(1,1,0);//2do motor
}
```

Programación de direccionamiento derecha, Elaborado por: Joel Diaz

CAPITULO 4
PRUEBAS Y RESULTADOS

4.1 Prueba de entrenamiento y desarrollo de la red neuronal artificial para el control de Robot modo seguidor de línea.

4.1.1 Creación de la red Backpropagation con función de transferencia tangente sigmoidal y lineal.

Para obtener un mejor resultado se utiliza el Toolbox de Matlab seleccionado el parámetro Backprop como se indica en Anexo 2, de esta manera se crea la red neuronal. Para esta prueba se utiliza como función de transferencia en la primera capa la función tangente sigmoidal y una función lineal en la segunda capa.

Al realizar los parámetros de entrenamiento de la red neuronal artificial, se puede observar que la red aprende 7 iteraciones, como se muestra en el Anexo 2 lo cual es muy poco probable que su aprendizaje sea el esperado. Además de que su salida de la red neuronal son números enteros, esto no se puede dar debido a que esta red se majea con números decimales, por esta razón no se tiene la certeza de que los pesos y la red tenga un buen funcionamiento.

Con los pesos obtenidos se prueba en el algoritmo de fase de consulta, en los cuales se inserta los pesos y el bias obtenidos del toolbox, y se ingresan los valores [1 1 1 1 0] y el resultado esperado seria 9, sin embargo, se obtiene un numero de -0.84801 lo cual es una respuesta muy lejana de la esperada.

En la fase de consulta se puede observar que con estos valores los resultados son negativos y no se pueden utilizar en el control del robot en modo seguidor de línea, además de que sus los valores de salida solo varían de -1 a 1, esto se debe a que la función de transferencia es tangente sigmoidal, y la salida que se requiere son valores mayores a uno, para obtener los valores deseados se debe utilizar una función de acondicionamiento o probar con diferentes funciones que permitan valores mayor a uno.

4.1.2 Creación de la red Backpropagation con función de transferencia en la primera capa línea y en la segunda lineal.

Utilizando el mismo procedimiento con la función de transferencia tangente sigmoidal se prueba con una función de transferencia lineal en ambas capas, para de esta manera el resultado no se restringe a valores de entre -1 y 1, debido a la función de transferencia tangente sigmoidal de la anterior prueba con este cambio se espera tener resultados más eficientes.

En la fase de entrenamiento se observa en el Anexo 3 que la red neuronal artificial solo tiene alrededor de 5 iteraciones, esto no brinda una buena señal del aprendizaje y por ende de obtener buenos resultados, en la salida de la red se evidencia que la red no logra aprender y que posee mucha confusión.

A pesar de que no se muestran buenos resultados en la salida se implementan los pesos obtenidos, se cambia el bias, obtenidos en el aprendizaje de la red neuronal.

Una vez creada y configurada la red se realizan las pruebas y se obtiene malos resultados, de igual manera se prueba con valores de entrada [1 1 1 1 0] pero la salida da un número negativo -0.36073, aún que se realizan varias pruebas cambiando los valores de entrada no se logra aprender eficazmente y los resultados son erróneos.

A pesar de que la red neuronal Backpropagation utiliza una capa oculta para enlazar los pesos y tener mejores resultados no se logra conseguir las salidas esperadas, se probó cambiado los valores de salida, sin embargo, al final la red no aprende y no se utiliza para el control del robot modo seguidor de línea.

4.1.3 Prueba de aprendizaje con la creación de la red Adaline.

Para la creación de la red Adaline al ser una red mono capa, se tiene dificultades para aprender debido a que posee cinco entradas como muestra el Anexo 4, sin embargo, se

procede hacer la matriz de entradas como se utilizó en las anteriores pruebas y como salida se maneja el error, que será utilizo para el sistema PID, que es encargado de controlar los motores en base al error el cual maneja valores de -4 a 4, siendo esta la salida de la matriz de la red neuronal.

Para la creación de la red se utilizaron los parámetros de aprendizaje de la siguiente manera, con una tasa de aprendizaje de 0.0001 y un bias de 0.01.

En la primera prueba la red no logra aprender y tiene confusión con los números negativos, dando como resultado un bucle infinito donde la red neuronal no aprende como muestra el Anexo 5, esto es porque la salida son números reales y a la red Adaline se complica procesar estos datos, sin embargo, si se cambiarán las salidas por un código binario sería más sencillo de procesar y los resultados serán eficaces.

4.1.4 Resultados finales.

Para finalizar con los resultados en la categoría seguidor de lineal se utilizó la red neuronal Perceptron con una matriz de salida binaria, de tal manera que la red logre aprender y su respuesta sea la esperada como indica en al apartado 3.3, realizando pruebas de funcionamiento y velocidad se determina un promedio de 95% en las pruebas al completar la pista con un tiempo promedio de 2 minutos y 45 segundos, el tiempo más rápido en que tarda en concluir el recorrido fue de 2 minutos como se observa en la Tabla 4.1, la pista posee una dificultad moderada como se muestra en la Figura 4.15.

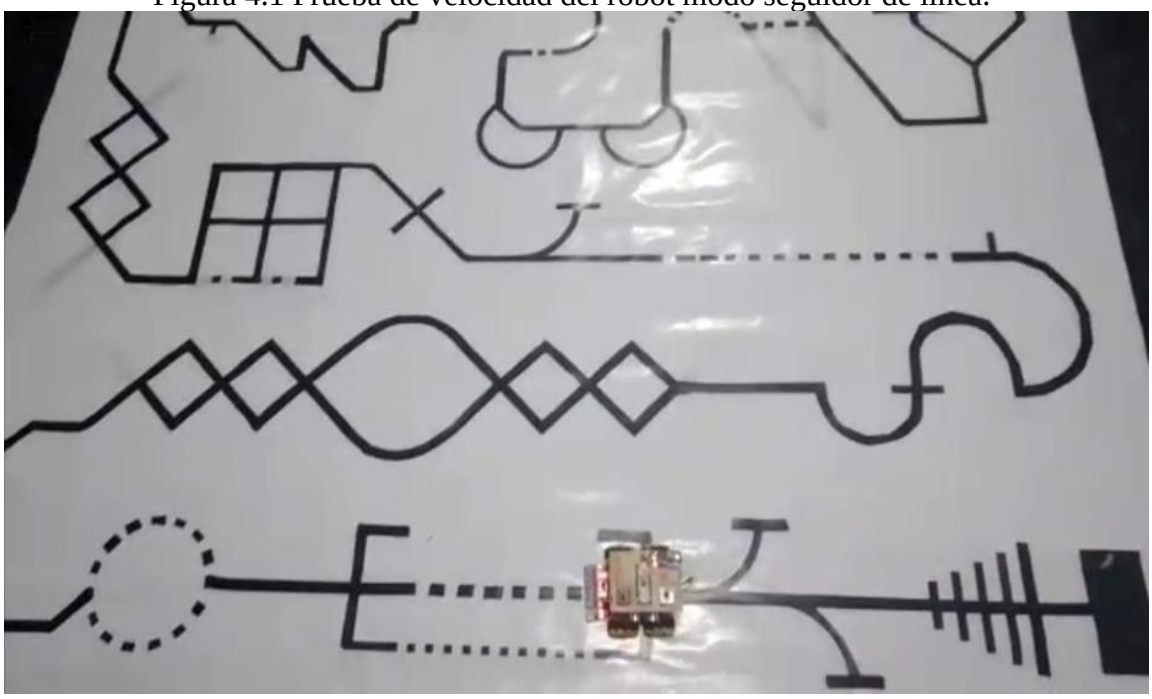
Tabla 4.1 Pruebas del robot modo seguidor de línea.

Pruebas realizadas	Porcentaje de pista realizada	Tiempo recorrido en la pista
1	80 %	3 minutos y 40 segundos
2	75%	3 minutos y 25 segundos
3	85%	3 minutos y 45 segundos
4	85%	3 minutos y 40 segundos
5	90%	3 minutos y 50 segundos

6	65%	2 minutos y 20 segundos
7	70%	2 minutos y 55 segundos
8	75%	2 minutos y 55 segundos
9	80%	2 minutos y 30 segundos
10	100%	3 minutos
11	100%	3 minutos y 20 segundos
12	95%	2 minutos y 45 segundos
13	100%	2 minutos y 20 segundos
14	100%	2 minutos y 20 segundos
15	95%	2 minutos y 25 segundos
16	100%	2 minutos
17	100%	2 minutos
18	100%	2 minutos y 25 segundos
19	100%	2 minutos y 15 segundos
20	100%	2 minutos

Pruebas del robot modo seguidor de línea, Elaborado por: Joel Diaz

Figura 4.1 Prueba de velocidad del robot modo seguidor de línea.



Prueba de velocidad del robot modo seguidor de línea, Elaborado por: Joel Diaz

En el caso del robot modo resuelve laberintos se utilizó la red neuronal artificial Adaline, debido a que permite obtener valores de salida mayores a uno, además de que en este

modo el robo tiene una estructura simple de 3 neuronas de entradas y una neurona de salida como se determina en el apartado 3.6, en este se logró observar que realizó alrededor de 120 iteraciones para hallar los pesos correspondientes para la salida deseada.

En las pruebas realizadas se determinó un promedio del 90% de recorrido de la pista con un tiempo aproximado de 45 segundos, el tiempo más rápido en el que robot resuelve el laberinto es de 30 segundos como se muestra en la Tabla 4.2, la pista posee una dificultad media, su tamaño es de 1.5 m^2 cómo se observa en la Figura 4.16.

Tabla 4.2 Pruebas del robot modo seguidor laberinto.

Pruebas realizadas	Porcentaje de pista realizada	Tiempo recorrido en la pista
1	70 %	45 segundos
2	75%	60 segundos
3	85%	80 segundos
4	70%	50 segundos
5	85%	45 segundos
6	75%	45 segundos
7	75%	50 segundos
8	75%	40 segundos
9	80%	45 segundos
10	100%	40 segundos
11	100%	35 segundos
12	95%	35 segundos
13	95%	35 segundos
14	100%	30 segundos
15	95%	30 segundos
16	100%	35 segundos
17	100%	30 segundos
18	100%	30 segundos
19	100%	30 segundos
20	100%	30 segundos

Pruebas del robot modo seguidor laberinto, Elaborado por: Joel Diaz

Figura 4.2 Prueba de velocidad del robot modo laberinto

Prueba de velocidad del robot modo laberinto, Elaborado por: Joel Diaz

CONCLUSIONES

Se decidió crear las redes neuronales artificiales y entrenarlas en Matlab con su herramienta nntool, debido a que posee una aplicación que permite crear y entrenar las redes neuronales artificiales llamado nntool, en esta se crea las redes como MPL, Adaline, etc. Como se evidencia en el apartado 4.1.1 se tuvieron problemas al crear la red Backpropagation en la fase de consulta para el robot modo seguidor de línea, por este motivo se decidió realizar con la red artificial Perceptron ya que la fase de consulta es lineal y por este motivo se utilizó una función de transferencia lineal, lo que facilita la implantación en el microcontrolador, realizando estos cambios se logró obtener los resultados deseados.

Para la construcción del robot se decidió incorporar ruedas omnidireccionales, estas permitieron un manejo autónomo eficaz cuando está en el modo laberinto, permitiendo que el robot se mantenga alineado para realizar los giros de 90° grados, a diferencia de que si el robot fuera diferencial se tendría que realizar más movimiento para que este se encuentre alineado con la pared, de este modo con ruedas omnidireccionales los giros son más precisos y no requiere de complejos movimientos para alinearse y girar, además de que si el robot se acercase demasiado a una pared, este se moverá al lado contrario para evitar choques y no perder puntos de competencia.

Los sensores utilizados para el robot seguidor de línea son de mayor sensibilidad, esto permite que los factores ambientales como la luz no afecta al reconocimiento de la línea, evitando falsos positivos. Se usaron 5 sensores que dependiendo de cual se active generan un error negativo si está a la izquierda o positivo si está a la derecha de esta manera el robot corregirá la trayectoria para que continue con su recorrido, si bien el funcionamiento fue el esperado si se incorpora más sensores ópticos se tendría un mejor control, debido a que así se podrían crear más casos para la toma de decisiones.

En las pruebas de funcionamiento en modo seguidor de línea se tuvieron problemas con los sensores, debido a que se ven afectados por la luz ocasionado resultados erróneos,

además de que la altura de separación del suelo al sensor también era un factor que alteraba la detección del sensor óptico, al utilizar sensores de diferente tipo la falla persistía. Sin embargo, al colocar sensores de mayor tamaño permitió una mejor detección y evitar falsos positivos entregando los valores digitales correctos a la matriz de entrada.

Realizando las pruebas del robot modo seguidor de línea, este completó el recorrido de manera efectiva con un tiempo de 2 minutos, en el que el robot tiene un desempeño excelente utilizando la red neuronal Adaline, para el robot modo laberinto se obtuvo un tiempo de 30 segundos en el cual acaba la pista sin problema alguno gracias a la red neuronal Perceptron.

RECOMENDACIONES

Como principal recomendación es utilizar un microcontrolador que su funcionamiento sea de 5 voltios y no 3.3 voltios, debido a que en la actualidad los sensores que operan a 3.3 voltios son escasos y la mayoría de 5 voltios no son compatibles en su totalidad con controladores que manejar voltajes de 3.3 voltios, este problema se presentó al utilizar los sensores ultrasónicos ya que al funcionar con 5 voltios en el momento de utilizarlos con 3.3 los sensores tenían fallas, lo que no ocurrió con microcontroladores que manejar voltajes de 5 voltios, si bien presentan fallas se puede implementar restricciones en las mediciones para resolver este tipo de inconvenientes, pero para evitar estos problemas se recomienda utilizar microcontrolador de 5 voltios de operación así no existirían problemas de adaptabilidad con lo sensores y otros periféricos, sin embargo se puede observar que se están creando nuevas tecnologías que funciona con menor consumo y mayor capacidad de procesamiento, por lo que en el futuro cercano los sensores serán más eficientes como muestra se puede evidenciar con el microcontrolador usado en este proyecto que su consumo mínimo es 1.8 v y su capacidad de procesamiento es de 133MHz.

Para el robot en modo laberinto se presentó un problema a la hora de realizar los giros de 90 grados, debido que al no ser precisos estos tendían a desviar al robot, se probó con diferentes métodos como control PID, sensores ópticos en los lados para detectar si hay colisión, pero esto ocasionó que robot pierda la trayectoria y entre en bucle de giros, la solución parcial fue realizar pruebas de giro con Delay, de esta manera se calculó que tiempo se necesitan para hacer los giros dando un funcionamiento parcial pero efectivo al control del robot modo laberinto.

Se recomienda utilizar motores paso o servos motores para controlar los giros de manera más precisa y tener un funcionamiento más eficaz, de tal manera que no dependa de tiempos para su funcionamiento ya que estos se ven afectados por la cantidad de batería que dispone el robot.

Por último, se recomienda por el momento utilizar Ide de Arduino ya que con la ayuda de sus librerías facilitan el código, no obstante al ser relativamente nueva la raspberry pi pico no posee muchas librerías y su comunidad es pequeña, a diferencia de que lo posee Arduino, pero con el tiempo se está encontrando nuevas aportaciones y avances en la creaciones librerías y aplicaciones para el chip RP 2040, con el uso de leguajes de programación utilizados originalmente como son MicroPython o c/c++.

REFERENCIAS

Bibliografía

Acquatella, P. (15 de Marzo de 2012). *Tutorial Madaline Rule II - Un algoritmo de entrenamiento para redes neuronales*. Obtenido de mathworks: https://www.mathworks.com/matlabcentral/fileexchange/35658-tutorial-madaline-rule-ii-un-algoritmo-de-entrenamiento-para-redes-neuronales?tab=discussions

Adriana, P. (2020). La inteligencia artificial y la robótica: sus dilemas sociales, éticos y jurídicos. *Scielo*.

Basoqain, X. (2008). *REDES NEURONALES ARTIFICIALES Y SUS APLICACIONES.* Bilbao: Escuela Superior de Ingeniería de Bilbao.

Berzal, F. (2020). *Backpropagation.* Granada: UNiversidad de Granada.

Burgos, H. (2020). *PROTOTIPO DE ROBOT EDUCATIVO DESMONTABLE DE BAJO COSTO PARA COMPETENCIAS DE SEGUIDOR DE LINEA Y LABERINTO.* Guayaquil: FACULTAD DE INGENIERÍA INDUSTRIAL CARRERAINGENIERÍAENTELEINFORMÁTICA.

Carrillo, M., Cardona, J., Arvizo, G., & Rodríguez, F. (2016). *Sistema de control y arquitectura de un robot seguidor de línea.* Chihuahua: Universidad Tecnológica de Chihuahua.

Cytron. (Mayo de 2013). *HC-SR04 Datasheet*. Obtenido de datasheet4u: https://datasheet4u.com/datasheet-pdf/Cytron/HC-SR04/pdf.php?id=1291829

Electro Estore. (23 de Marzo de 2021). *LLANTA OMNIDIRECCIONAL MECANUM DERECHA 80MM*. Obtenido de Electro Estore: https://grupoelectrostore.com/shop/estructuras-y-partes-no-electronicas-de-robots/llantas-estructuras-y-partes-no-electronicas-de-robots/llanta-omnidireccional-mecanum-derecha-80mm/

Esneca. (3 de Agosto de 2018). *Esneca*. Obtenido de Clasificación de los robots según su función: https://www.esneca.com/blog/clasificacion-de-los-robots-segun-su-funcion/

Herrera, Y., & Daniel, R. (2010). *ESTADO DEL ARTE DE LA ROBOTICA EDUCATIVA EN EL.* Bogotá: Universidad Minuto de Dios.

ICCIUTEM. (22 de Abril de 2014). *Robots según estructura*. Obtenido de ICCIUTEM: https://icciutem2014.wordpress.com/2014/04/22/robots-segun-estructura/

IEEE. (2007). *Competencia de Robots Seguidores de Línea.* Lima: Universidad Nacional de Ingeniería.

Johnson, D. (29 de Octubre de 2022). *Back Propagation in Neural Network: Machine Learning Algorithm*. Obtenido de guru99: https://www.guru99.com/backpropogation-neural-network.html

Koldo. (3 de Julio de 2018). *Adaline – The Perceptron Evolution*. Obtenido de Koldopina: https://koldopina.com/adaline/

Llamas, L. (2 de Junio de 2016). *DETECTOR DE OBSTÁCULOS CON SENSOR INFRARROJO Y ARDUINO*. Obtenido de luisllamas: https://www.luisllamas.es/detectar-obstaculos-con-sensor-infrarrojo-y-arduino/

LRP electronic GmbH. (2021). *MANUAL DE INSTRUCCIONES Baterias LIPO.* LRP electronic GmbH: Schondorf. Obtenido de Dynamo.

Menacho, C. (2014). *Modelos de regresión lineal con redes neuronales.* Lima: Universidad Nacional Agraria La Molina.

Mesias, I., & Hidalgo, B. (2020). ESTUDIO COMPARATIVO DE LOS ALGORITMOS BACKPROPAGATION (BP) Y MÚLTIPLE LINEAR REGRESSION (MLR) A TRAVÉS DEL ANÁLISIS ESTADÍSTICO DE DATOS APLICADO A REDES NEURONALES ARTIFICIALES. *REVISTA BOLETÍN REDIPE* , 144-152.

Naylamp. (9 de Mayo de 2021). *MOTOR DC CON CAJA REDUCTORA CON EJE DE METAL*. Obtenido de naylampmechatronic: https://naylampmechatronics.com/motores-dc/607-motor-dc-con-caja-reductora-con-eje-de-metal-6v-200rpm.html

Pololu. (4 de Noviembre de 2018). *Pololu 3pi Robot*. Obtenido de Pololu: https://www.pololu.com/product/975

raspberry. (6 de OCtubre de 2021). *Documentación de Raspberry Pi*. Obtenido de raspberry: https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html

Ripipsa. (6 de Agosto de 2020). *ROBOTS AUTÓNOMOS: QUÉ SON, CÓMO FUNCIONAN Y QUE VENTAJAS OFRECEN.* Obtenido de Ripipsa Tecnologia al servicio de la indrustria : https://ripipsacobots.com/robots-autonomos/

robotsystems.net. (18 de Marzo de 2011). *robotsystems.net*. Obtenido de Micromouse - the early years, 1980: http://davidbuckley.net/RS/mmouse/micromouse80.htm

robotsystems.net. (21 de MArzo de 2014). *robotsystems.net*. Obtenido de Microcomputers: The Great Electronic Mouse Race: http://davidbuckley.net/RS/mmouse/micromouse78.htm

Roca, J. (16 de Marzo de 2021). *Todos los chips tienen uno pero, ¿sabes qué son los microcontroladores?* Obtenido de Hardzone: https://hardzone.es/reportajes/que-es/microcontroladores/

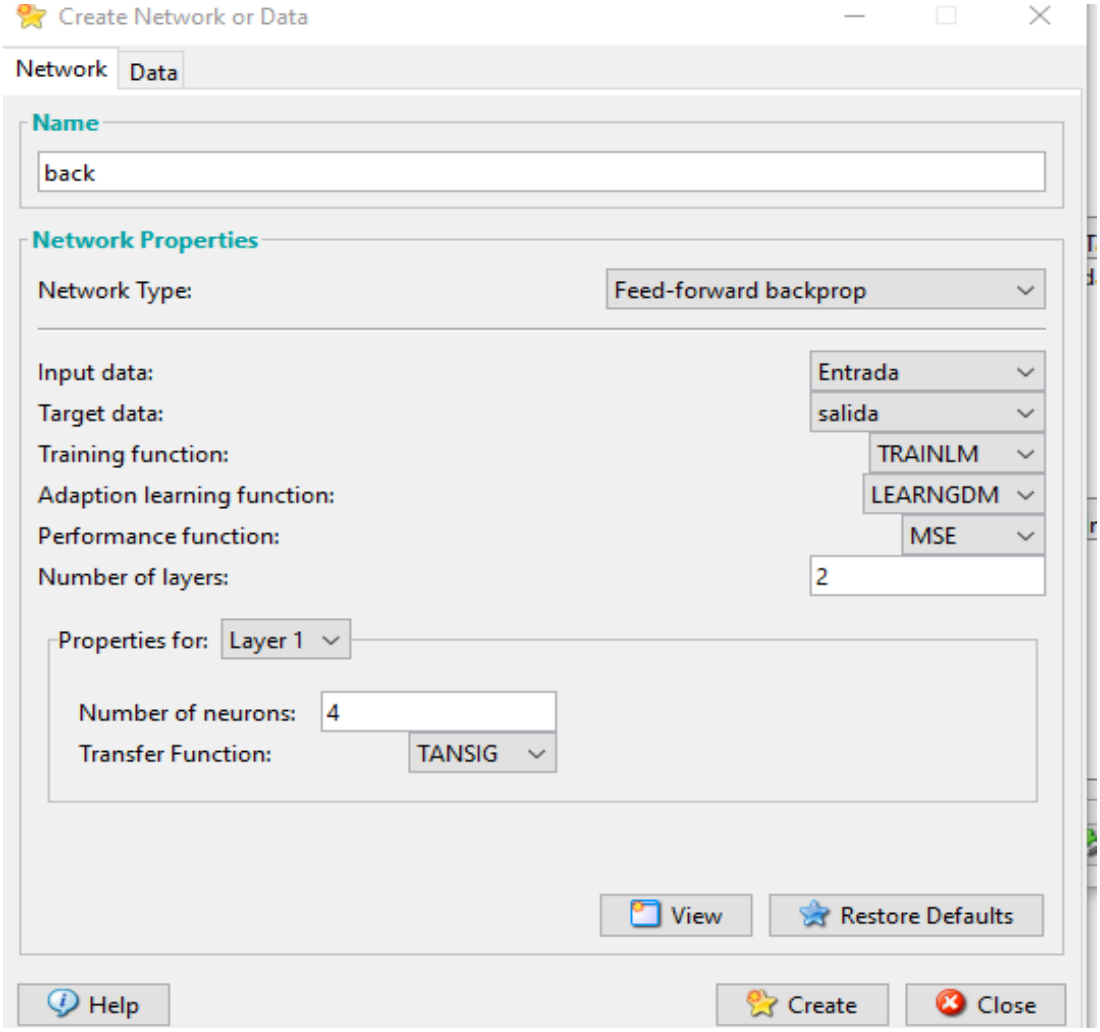
Rouhiainen, L. (2018). *INTELIGENCIA ARTIFICIAL 101 COSAS QUE DEBES SABER HOY SOBRE NUESTRO FUTURO.* España: Editorial Planeta, S.A.

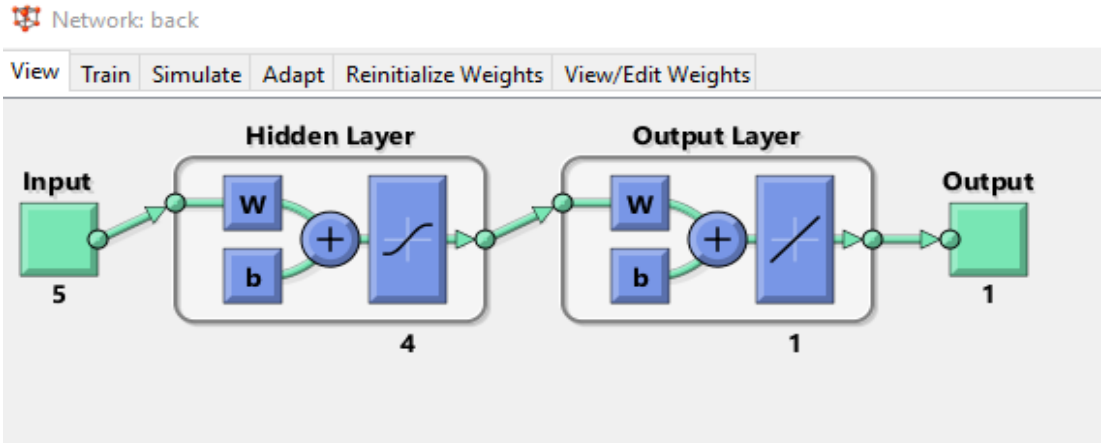
Valverde, B. (21 de agosto de 2020). La importancia de la Robótica como eje en el desarrollo de la sociedad. *POLO DEL CONOCIMIENTO*, págs. 1368-1377.

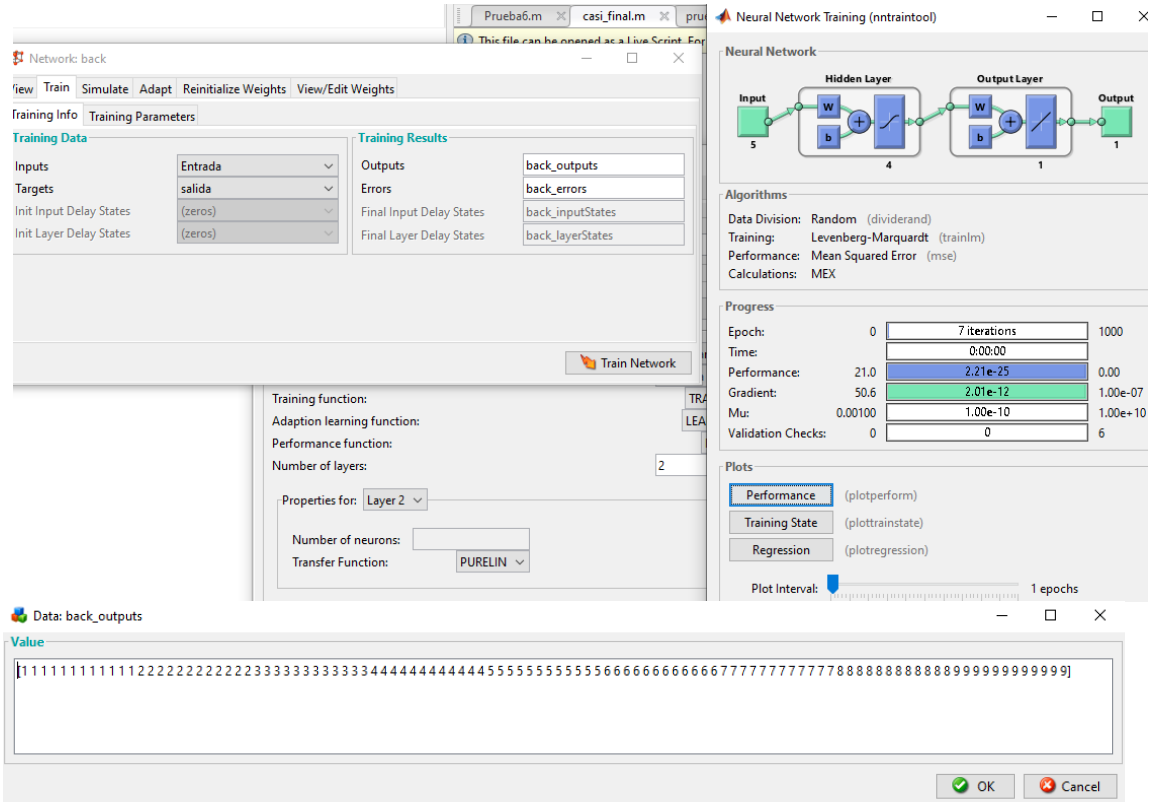
ANEXO I: Tabal de presupuesto para la elaboración del prototipo

ACTIVIDAD	DETALLES		PRECIO
	MATERIALES	Precio	
Placa PCB	2 driver TB6612fng	20 $	38.5 $
	raspberry pi pico	15 $	
	Lm2596 Modulo Reductor De Voltaje	3.5	
Arquitectura del robot	1 armazón de aluminio	0.5 $	71.5 $
	Batería LI-PO 7.4v 2 celdas 500mA	30 $$	
	4 motores DC + ruedas omnidireccionales	40 $	
	Cables	1 $	
Sensores	3 sensores ultrasónico Hc-sr04	7.5 $	22.5 $
	5 sensores infrarrojos	15 $	
Mano de obra	Un mes de lunes a viernes de 5 horas laborables	500 $	500 $
Costo total del proyecto			**632.5$**

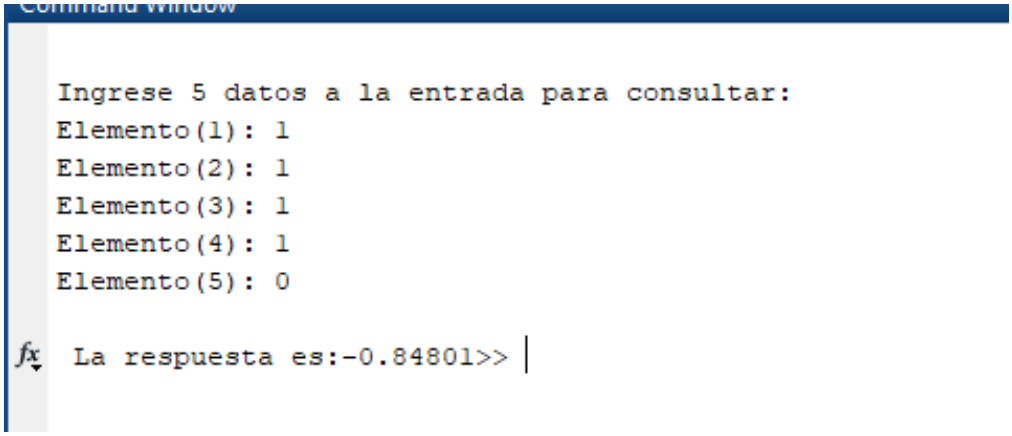
ANEXO 2: Creación de la red neuronal Backpropagation con función de transferencia tangente sigmoidal y lineal.



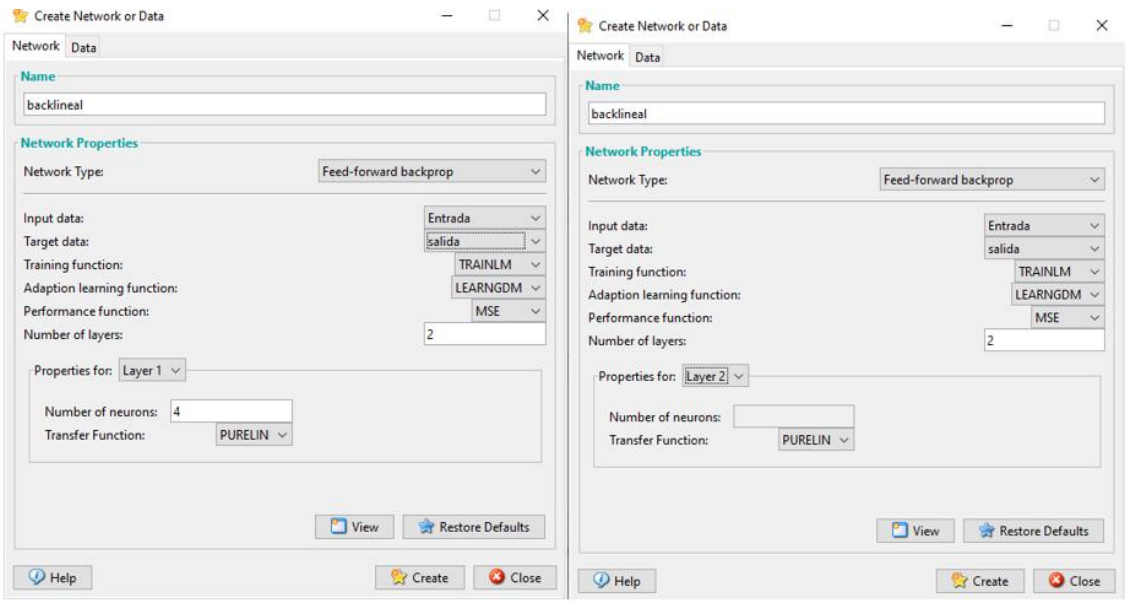


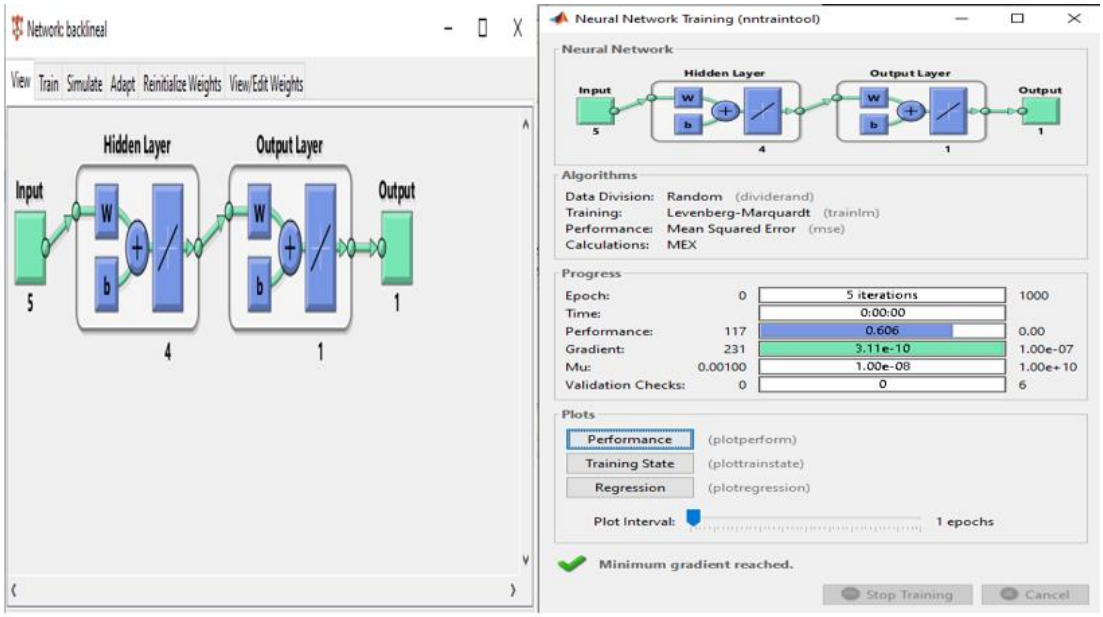


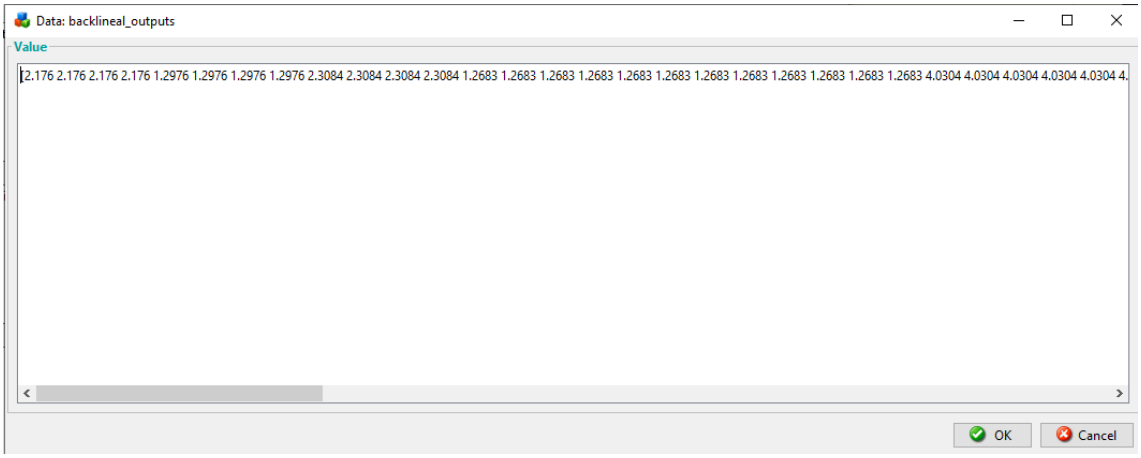

```
%% pesos de la capa 1
wc1= [0.80956 0.14123 1.9607 0.26597 1.4096;
 1.2963 -0.91843 1.3426 0.38952 0.86528;
 -0.63717 -0.46908 -0.030959 0.27236 1.134;
 -0.11238 -0.10309 0.96898 -0.36329 1.6918]
 %% bias 1
b1=[-1.7927;
 -0.69082;
 -0.45142;
 1.7318]
%% pesos de la capa 2
wc2= [0.16165 -0.30533 0.81537 0.27595]
%% bias 2
b2 =-0.15087                          %% bias 2
```



ANEXO 3: Creación de la red neuronal Backpropagation con función de transferencia en la primera capa línea y en la segunda lineal.

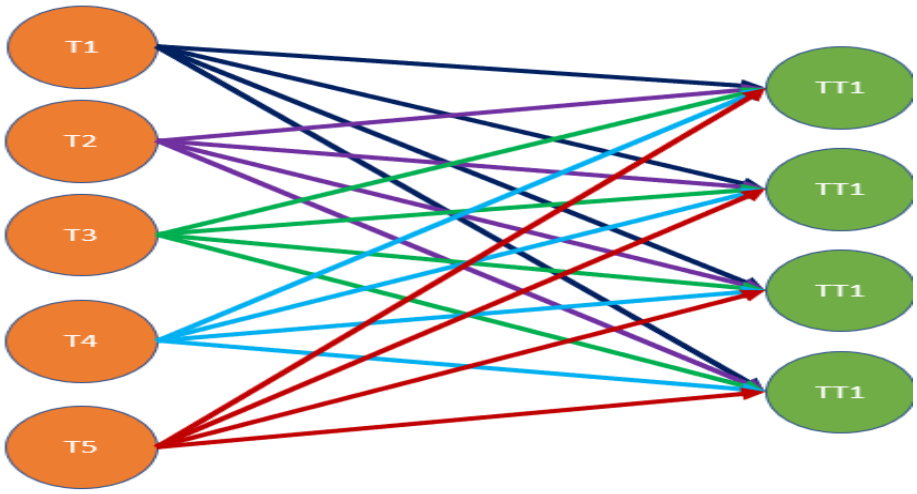







```
%% pesos de la capa 1
wc1= [0.13807 0.85283 -0.77634 -0.67044 -0.83155;
 -0.2232 0.2976 -0.13498 -0.39079 0.87166;
 -0.3923 -0.40904 -0.62713 -0.74443 -0.18165;
 -0.7568 -0.081499 -0.29889 0.73696 -0.87952];
 %% bias 1
b1=[-0.21405;
 0.80222;
 0.28758;
 -0.0084478];
%% pesos de la capa 2
wc2= [-0.23881 0.44472 0.083381 0.25675];
%% bias 2
b2 =-0.41462  ;                    %% bias 2
```

ANEXO 4: Estructura de le red neuronal artificial Adaline



ANEXO 5: Programación de la red artificial Adaline

```
LR= 0.0001; var=1; suma=0; iteracion=0; bias=0.01;%el bias inicial en el adalain es 1   1-0.1-0.01
p= (abs(rand(5,1)))/10000000;
s = length(salida)
while var ==1
    for i= 1:s
        I(i) = (Entrada(i,1)*p(1)) + (Entrada(i,2)*p(2))+ (Entrada(i,3)*p(3))+ (Entrada(i,4)*p(4)) + (Entrada(i,5)*p(5))   + bias
        I1(i) = round(I(i))

        p(1)= p(1) + (LR*(salida(i)-I1(i))*Entrada(i,1));
        p(2)= p(2) + (LR*(salida(i)-I1(i))*Entrada(i,2));
        p(3)= p(3) + (LR*(salida(i)-I1(i))*Entrada(i,3));
        p(4)= p(4) + (LR*(salida(i)-I1(i))*Entrada(i,4));
        p(5)= p(5) + (LR*(salida(i)-I1(i))*Entrada(i,5));


        bias= bias +(2*LR*(salida(i)-I1(i)));
        error(i)=abs(salida(i)-I1(i));
        suma= suma+error(i);

    end
     if suma==0
           disp('La neurona ya aprendio')
           var=10;
     end
```