UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR	**FORMATO DE GUÍA DE PRÁCTICA DE LABORATORIO / TALLERES / CENTROS DE SIMULACIÓN – PARA DOCENTES**
CARRERA: Electrónica y Automatización	**ASIGNATURA:** Inteligencia Artificial
NRO. PRÁCTICA: 1	**TÍTULO PRÁCTICA**: Control de un motor DC con un grupo de perceptrones

Fecha: 27/11/2023	**Estudiantes:** • Darwin Andrés Chacha Criollo • Alexis Valentín Borja Palacios	**Responsable:** Dr. Christian Salamea P.

Resumen— *La práctica implica entrenar dos neuronas con un perceptrón convencional para controlar la velocidad y dirección de un motor mediante interruptores. Al aprovechar la capacidad de las neuronas para aprender y adaptarse, se logra un control preciso y eficiente del motor.*

Abstract— *The practice involves training two neurons with a conventional perceptron to control the speed and direction of a motor using switches. By harnessing the ability of neurons to learn and adapt, precise and efficient motor control is achieved.*

I. OBJETIVOS

Objetivo general.

- Entrenar eficientemente dos neuronas con el fin de lograr un control preciso y óptimo de un motor.

Objetivo Específicos.

- Diseñar circuitos con amplificadores operacionales para gestionar las señales de salida y lograr un control efectivo del motor.
- Verificar el funcionamiento de las neuronas mediante la evaluación de sus tablas de verdad, asegurando respuestas coherentes y precisas.
- Analizar el comportamiento individual de cada neurona, identificando sus características específicas de velocidad y dirección.

II. EQUIPOS, INSTRUMENTOS Y SOFTWARE

1. Op-Amp LM741
2. Driver LN298
3. Software Colaboratory
4. Arduino Uno
5. Motor DC

III. MARCO TEÓRICO

E El Perceptrón: ¿Qué es y para qué sirve?

El perceptrón, inventado por Frank Rosenblatt en 1957 en el laboratorio aeronáutico de Cornell, es una neurona artificial que opera como una unidad fundamental en las redes neuronales. Este algoritmo realiza cálculos para identificar características o patrones en los datos de entrada [1]

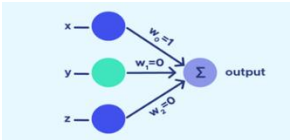

Fig. 1. Componentes de un perceptrón

Red Neuronal

Una red neuronal se forma cuando una colección de nodos o neuronas se interconectan a través de conexiones sinápticas. Hay tres capas en cada red neuronal artificial: capa de entrada, capa oculta y capa de salida [2]

Algoritmo de aprendizaje del perceptrón

Se utiliza como algoritmo o clasificador lineal para facilitar el aprendizaje supervisado de clasificadores binarios.

Un clasificador lineal en el que se clasifica el perceptrón es un algoritmo de clasificación, que se basa en una función predictora lineal para hacer predicciones. Sus predicciones se basan en una combinación que incluye pesos y vector de características [2]

A continuación, se enumeran los componentes principales de un perceptrón [3]:

1. Entradas: Valores de entrada al perceptrón.
2. Pesos: Valores asociados a cada entrada.
3. Bias (sesgo): Valor adicional sumado a la suma ponderada antes de la función de activación.
4. Suma ponderada: Resultado de multiplicar cada entrada por su peso y sumar los productos.
5. Función de activación: Función matemática aplicada a la suma ponderada.
6. Salida: Resultado final del perceptrón tras la función de activación.

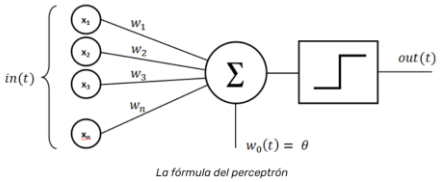

Fig. 2. Fórmula para un perceptrón

La fórmula para la salida de un perceptrón se puede expresar de la siguiente manera [3]:

$$\boldsymbol{y(t) = f[\omega_1 X_1 + \omega_2 X_2 - \omega_\theta X_\theta]}$$

Donde:

$\boldsymbol{y(t)} = Salida\ del\ perceptrón.$
$\boldsymbol{f} = Es\ la\ funcion\ de\ activación.$
$\boldsymbol{\omega} = Es\ el\ vector\ de\ pesos.$

$\omega_\theta = Es\ el\ sesgo\ nergativo\ que\ representa\ el\ umbral\ de\ activación.$

IV. DESARROLLO DE LA PRACTICA.

Al emplear el modelo de entrenamiento del perceptrón, obtendremos un modelo capaz de separar linealmente patrones, lo que implica que puede decir si una entrada pertenece a una de las dos clases. Durante el proceso, se utilizaron las siguientes tablas de verdad:

$1 \rightarrow izquierda \qquad 0 \rightarrow derecha$

V1	V2	Salida
0	0	0
0	1	0
1	0	1
1	1	1

Tabla 1. Tabla de verdad para la neurona de dirección.

$1 \rightarrow rapido \qquad 0 \rightarrow lento$

V1	V2	Salida
0	0	0
0	1	0
1	0	0
1	1	1

Tabla 2. Tabla de verdad para la neurona de velocidad

Buscaremos los pesos que están relacionados con cada una de las entradas y con un umbral de $\theta = 2$:

Pesos para la neurona de dirección:

Pesos Iniciales	Pesos finales
$\omega_1 = 1$ $\omega_2 = 1$ $\omega_\theta = 1$	$\omega_1 = 3$ $\omega_2 = 0.5$ $\omega_\theta = 1.2$

Pesos para la neurona de velocidad:

Pesos Iniciales	Pesos finales
$\omega_1 = 1$ $\omega_2 = 1$ $\omega_\theta = 1$	$\omega_1 = 0.55$ $\omega_2 = 0.45$ $\omega_\theta = 1.14$

- **Neurona de dirección.**

0	0	→ 0	y = -1.2 (2) = **-2.4**
0	1	→ 0	y = 0.5 (5) – 1.2 (2) = **0.1**
1	0	→ 1	y = 3 (5) – 1.2 (2) = **12.6**
1	1	→ 1	y = 3 (5) + 0.5 (5) – 1.2 (2) = **15.2**

- **Neurona de velocidad.**

0	0	→ 0	y = -1.14 (2) = **-2.2**
0	1	→ 0	y = 0.45 (5) – 1.14 (2) = **0.03**
1	0	→ 1	y = 0.55 (5) – 1.14 (2) = **0.47**
1	1	→ 1	y = 0.55 (5) + 0.45 (5) – 1.14 (2) = **2.72**

Mediante la configuración de Op-Amp´s y el calculo de sus respectivos valores de componentes obtenemos las siguientes configuraciones.

- **Neurona de dirección.**

Para la configuración de la neurona de dirección se implemento un amplificador operacional inversor, el cual nos permite invertir la polaridad de nuestra señal de entrada en este caso Vumbral el cual es de -2.4V.

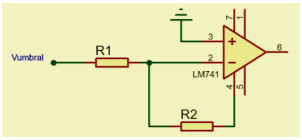

Fig 3. Configuración con inversor Op-Amp

Datos:

$V_{in} = 2\ V$
$V_{out} = -2.4\ V$
$R_1 = 10\ K\Omega$

$$R_2 = -\frac{R_1}{V_{out}} V_{in} = -\frac{10K}{(-2.4)}(2) = 8.3K$$
$$R_2 = 8.3\ K\Omega \approx 8.2\ K\Omega \rightarrow valor\ comercial$$

Configuramos un amplificador operacional sumador inversor para la sumatoria de pesos.

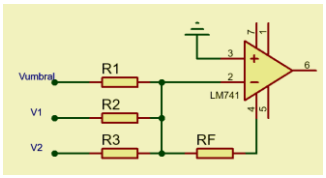

Fig 4. Configuración del sumador inversor Op-Amp

Datos para el caso de V_1:

$V_{in} = 5\ V\ (1\ equivale \rightarrow 5\ V) \quad V_{out} = 15\ V\ (\omega_\theta = 3)$
$R_f = 10\ K\Omega$

$$R_2 = \frac{R_f}{V_{out}} V_{in} = \frac{10K}{(15)}(5) = 3.3K\Omega$$
$$R_2 = 3.3\ K\Omega$$

Datos para el caso de V_2:

$V_{in} = 5\ V\ (1\ equivale \rightarrow 5\ V) \quad V_{out} = 2.5\ V\ (\omega_\theta = 0.5)$
$R_f = 10\ K\Omega$

$$R_3 = \frac{R_f}{V_{out}} V_{in} = \frac{10K}{(2.5)}(5) = 20\ K\Omega$$
$$R_3 = 20\ K\Omega \approx 18\ K\Omega \rightarrow valor\ comercial$$

- **Neurona de velocidad.**

Se implementó un amplificador operacional inversor en la neurona de velocidad para invertir la polaridad de la señal de entrada (Umbral, -2.28V), similar a la configuración de la neurona de dirección.

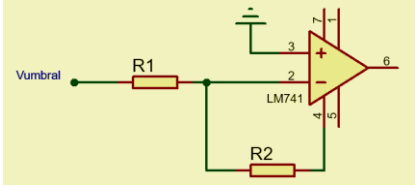

Fig5. Configuración con inversor Op-Amp

Datos:

$V_{in} = 2\ V \quad V_{out} = -2.28\ V \quad R_1 = 10\ K\Omega$

$$R_2 = -\frac{R_1}{V_{out}} V_{in} = -\frac{10K}{(-2.28)}(2) = 8.7K$$

$$R_2 = 8.7\ K\Omega \approx 9.1\ K\Omega \rightarrow valor\ comercial$$

Configuramos un amplificador operacional sumador inversor para la sumatoria de pesos multiplicados.

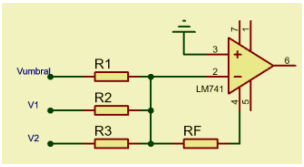

Fig 6. Configuración del sumador inversor Op-Amp

Datos para el caso de V_1:

$V_{in} = 5\ V\ (1\ equivale \rightarrow 5\ V)$

$V_{out} = 2.75\ V\ (\omega_\theta = 0.55)$

$R_f = 10\ K\Omega$

$$R_2 = \frac{R_f}{V_{out}} V_{in} = \frac{10K}{(2.75)}(5) = 18.18\ K\Omega$$

$$R_2 = 18.18\ K\Omega \approx 18\ K\Omega \rightarrow valor\ comercial$$

Datos para el caso de V_2:

$V_{in} = 5\ V\ (1\ equivale \rightarrow 5\ V) \quad V_{out} = 2.25\ V\ (\omega_\theta = 0.45)$

$R_f = 10\ K\Omega$

$$R_3 = \frac{R_f}{V_{out}} V_{in} = \frac{10K}{(2.25)}(5) = 22.2\ K\Omega$$

$$R_2 = 22.2\ K\Omega \approx 22\ K\Omega \rightarrow valor\ comercial$$

Las salidas de los sumadores en ambas neuronas tienen el signo opuesto para eso se agrega un inversor corrigiendo el voltaje de salida de ambos sumadores en ambos casos.

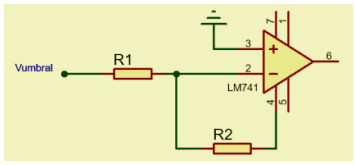

Fig 7. Configuración del inversor Op-Amp

$$V_{out} = -\frac{R_2}{R_1} V_{in}$$

Datos:

- V_in: Valor obtenido del sumador, corresponde a la salida de V1 y V2.
- V_out: Valor igual a V_in pero con signo opuesto.
- R_1 = 10 KΩ
- R_2 = 10 KΩ

Para ambos grupos de neuronas, la salida se evalúa mediante una función utilizando una configuración de Op-Amp en comparador.

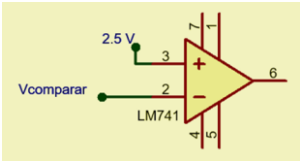

Fig 8. Configuración inversor Op-Amp

El valor que sale del último inversor se compara con 2.5V, si el valor es mayor se va a 1 (5V) y si el valor es menor a 2.5V se va a 0 (0V).

A continuación, se muestra el diseño completo para cada una de las neuronas con sus respectivos valores calculados:

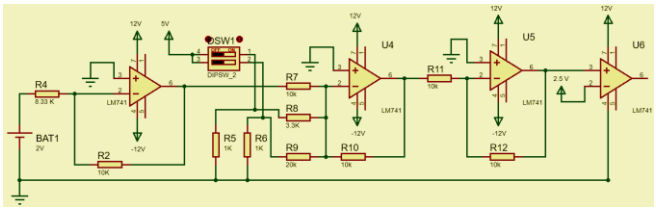

Fig 9. Red de neuronas para controlar la dirección.

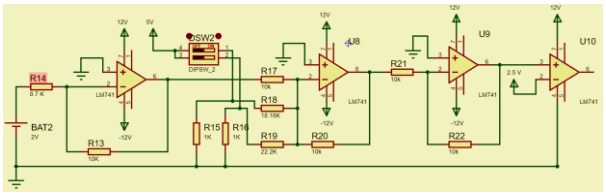

Fig 10. Red de neuronas para controlar la velocidad.

Finalmente se desarrolla la parte del controlador del motor mediante la implementación de un microcontrolador Arduino y un driver L298N para controlar el funcionamiento de nuestro motor.

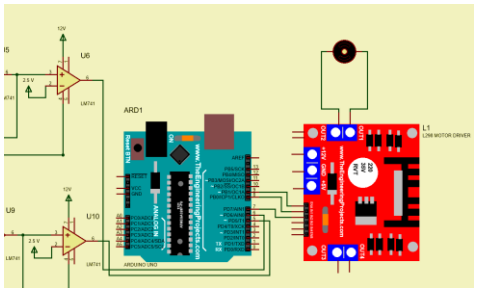

Fig 11. Conexión de la salida del grupo de neuronas con el microcontrolador, el puente H y el motor.

V. CONCLUSIONES.

En este experimento, se empleó el algoritmo de perceptrones, una técnica de aprendizaje automático supervisado lineal que sienta las bases para las redes neuronales. Se estableció un sistema de control mediante el entrenamiento de dos neuronas utilizando el modelo convencional de perceptrón. Estas neuronas tomaron decisiones sobre la dirección y velocidad de un sistema con interruptor y motor. La implementación de amplificadores operacionales ofreció una solución práctica para controlar el sistema en tiempo real.

Estos amplificadores convirtieron las salidas binarias de las neuronas en señales que regularon con precisión la dirección y velocidad del sistema, asegurando un control eficiente.

VI. BIBLIOGRAFIA

[1] Perceptron : qu'est-ce que c'est et à quoi ça sert ? (2022, March 7). Formation Data Science | Datascientest.com. https://datascientest.com/es/perceptron-que-es-y-para-que-sirve

[2] Ponce Cruz, P. (2011). Inteligencia Artificial con Aplicaciones a la Ingeniería (1.ª ed., p. 376). Ciudad Mexico: Alfaomega Grupo Editor, S.A. Ciudad Mexico

[3] Redes Neuronales. (2020, April 14). Amazonaws.com. https://rstudio-pubs-static.s3.amazonaws.com/599210_4306c5d3d6a34a6c829566ca11b7e27a.html#20

VII. ANEXOS.

▪ Código Entrenamiento del perceptrón en C++

```
#include <stdio.h>
int main(){
    float a=0,b=0, d=0,e=1,l,x,y,z,net,sum,c=0,r1,r2,r3,r4,e1=1;
    printf("Dame los valores iniciales\n");
    printf("x=");
    scanf("%f",&x);
    printf("y=");
    scanf("%f",&y);
    printf("z=");
    scanf("%f",&z);
    printf("l=");

    scanf("%f",&l);


    while(e!=0)
    {
        a=0,b=0,d=0;
        //printf("%f\n",x);
        sum=(a*x)+(b*y)-(2*z);
        if(sum>2.5)net=5;
        else net=0;
        e1=e;
        e=d-net;
        r1=e;
        x=x+l*e*a;
        y=y+l*e*b;
        z=z+l*e*2;

        a=0,b=5,d=0;
        sum=(a*x)+(b*y)-(2*z);
        if(sum>2.5)net=5;
        else net=0;
        e=d-net;
        r2=e;
        x=x+l*e*a;
        y=y+l*e*b;
        z=z+l*e*2;

        a=5,b=0,d=0;
        sum=(a*x)+(b*y)-(2*z);
        if(sum>2.5)net=5;
        else net=0;
        e=d-net;
        r3=e;
        x=x+l*e*a;
        y=y+l*e*b;
        z=z+l*e*2;

        a=5,b=5,d=5;
        sum=(a*x)+(b*y)-(2*z);
        if(sum>2.5)net=5;
        else net=0;
        e=d-net;
        r4=e;
        x=x+l*e*a;
           y=y+l*e*b;
        z=z+l*e*2;

        sum=(a*x)+(b*y)-(2*z);
        if(sum>2.5)net=5;
        else net=0;
        c++;

    }

    printf("x=%f\n",x);

    printf("y=%f\n",y);

    printf("z=%f\n",z);

}
```

▪ Código Implementado en Arduino

```
int motorInputA = 8;
int motorInputB = 7;
int motorEnable = 9;
int rotationSensor = 6;
int speedSensor = 5;

void setup() {
  pinMode(motorInputA, OUTPUT);
  pinMode(motorInputB, OUTPUT);
  pinMode(motorEnable, OUTPUT);
  pinMode(rotationSensor, INPUT);
  pinMode(speedSensor, INPUT);

  Serial.begin(9600);
}
void loop() {
  bool isClockwise = digitalRead(rotationSensor);
  bool isHighSpeed = digitalRead(speedSensor);

  if (!isClockwise) {
    if (!isHighSpeed) {
      digitalWrite(motorInputA, HIGH);
      digitalWrite(motorInputB, LOW);
      analogWrite(motorEnable, 50);
    }
  }

  if (isClockwise) {
    if (!isHighSpeed) {
      digitalWrite(motorInputA, LOW);
      digitalWrite(motorInputB, HIGH);
      analogWrite(motorEnable, 50);
    }
  }
  if (!isClockwise) {
    if (isHighSpeed) {
      digitalWrite(motorInputA, HIGH);
      digitalWrite(motorInputB, LOW);
      analogWrite(motorEnable, 100);
    }
  }

  if (isClockwise) {
    if (isHighSpeed) {
      digitalWrite(motorInputA, LOW);
      digitalWrite(motorInputB, HIGH);
      analogWrite(motorEnable, 100);
    }

  }

}
```

Armado del circuito

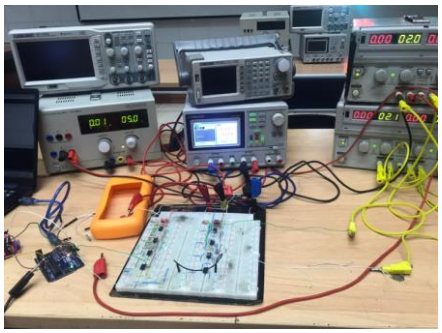

Fig 12. Armado de las redes neuronales y su conexión al microcontrolador y el driver del motor.

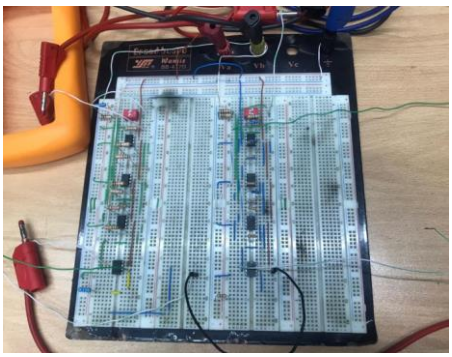

Fig 13. Armado de las redes neuronales con los circuitos operacionales.