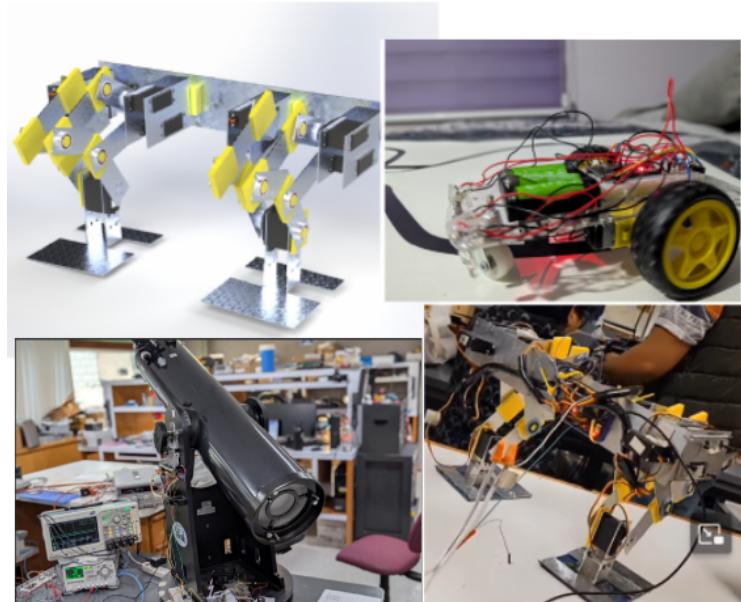


Alumno de:

Universidad Autónoma de Ciudad Juárez

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación



Portafolio de Constancias, Reconocimientos y Proyectos de:

Angel Hiram Alvidrez Hernandez

Índice

Constancias y Reconocimientos	2
Constancia Verano Científico OAN	2
Constancia Fundamentos de Semiconductores	3
Reconocimiento Como Estudiante Integral	4
Reconocimiento por Impartir conferencia: Realmente Estamos solos en el universo?	5
Reconocimiento por Impartir conferencia: Las muertes del Universo	6
Proyectos y Prácticas de Ingeniería	7
Bípedo subactuado Basado en un Mecanismo de 5 Barras	7
Actualización de la Electrónica de Control de un Telescopio	23
Amplificador de Guitarra mediante Transistores	24
Control de Posición de un carrito con PID	29
Carrito Seguidor de Líneas	42
Encendido y Apagado de un Foco con Aplausos	53
Sistema de Riego IoT (ESP32 + Blynk)	69



Instituto de
Astronomía

La Unidad Académica en Ensenada del Instituto de Astronomía y el Observatorio Astronómico Nacional San Pedro Martir,

Instituto de Astronomía, de la Universidad Nacional Autónoma de México,
junto al Comité Organizador del

XXXIV Verano Científico del Observatorio Astronómico Nacional San Pedro Martir

extienden la siguiente

Constancia

a

Ángel Hiram Alvídrez Hernández

Por su valiosa participación como **estudiante** en el Programa del XXXIV Verano Científico OAN-SPM, atendiendo cursos, talleres y conferencias, y desarrollando el proyecto de investigación que le fue asignado en el periodo comprendido entre el 16 de junio y el 4 de julio de 2025.



INSTITUTO DE ASTRONOMÍA
OBSERVATORIO ASTRONÓMICO
NACIONAL DE S.P.M.
ENSENADA, B.C.

Dr. Michael Richer

Jefe de la Unidad Académica Ensenada
del Instituto de Astronomía,
Universidad Nacional Autónoma de México

David Hiruart García

Dr. **David Hiruart García**
Jefe del Observatorio Astronómico Nacional
San Pedro Martir, Instituto de Astronomía,
Universidad Nacional Autónoma de México

Dra. Lucía Adame Villanueva
En representación del Comité Organizador
XXXIV Verano Científico OAN-SPM



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

INADET
INSTITUTO DE APOYO AL DESARROLLO
TECNOLÓGICO

CENALTEC
CENTRO DE ENTRENAMIENTO EN ALTA TECNOLOGÍA



**SECRETARÍA DE EDUCACIÓN PÚBLICA
SISTEMA EDUCATIVO NACIONAL**

DIRECCIÓN GENERAL DE CENTROS DE FORMACIÓN PARA EL TRABAJO

**INSTITUTO DE CAPACITACIÓN PARA EL TRABAJO
ORGANISMO DESCENTRALIZADO**

Unidad de capacitación: **CENTRO DE ENTRENAMIENTO EN ALTA TECNOLOGÍA** Con CCT: 08EIC0001B
IAD-990630-MK7-0013

OTORGА LA PRESENTE

CONSTANCIA

ANGEL HIRAM ALVIDREZ HERNANDEZ

con Clave Única de Registro de Población: **AIHA040515HCHLRNA2**

En virtud de que acreditó los conocimientos, habilidades, destrezas y actitudes del curso

FUNDAMENTOS DE SEMICONDUCTORES

impartido en 90 horas, de conformidad al programa de Extensión.

De acuerdo a la información académica que obra en los archivos del Centro Educativo.

El presente se expide en **JUÁREZ, CHIHUAHUA**
A los **DOCE** días del mes de **DICIEMBRE** del dos mil **VEINTICUATRO**

DO 6770
Folio


DGCFT


BRENDA AIDEЕ RODRIGUEZ PINALES

Director (a)

ESTADOS UNIDOS MEXICANOS
Sep Sems


D.G.C.F.T.
CENTRO DE ENTRENAMIENTO
EN ALTA TECNOLOGÍA
CCT. 08EIC0001B
STPS. IAD-990630-MK7-0013

SELLO

Otorga el presente

RECONOCIMIENTO

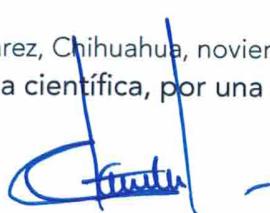
A: **Ángel Hiram Alvírez Hernández**

del programa

Ingeniería Mecatrónica

**Por distinguirse como ESTUDIANTE INTEGRAL, con su excelente desempeño
académico y por haber realizado actividades a favor de la comunidad
durante el semestre enero-junio 2025.**

Ciudad Juárez, Chihuahua, noviembre 2025.
Por una vida científica, por una ciencia vital



Dr. Daniel Alberto Constandse Cortez
Rector

Dr. Salvador David Nava Martínez
Secretario General

Dr. Pedro Enrique Yáñez Camacho
Director General de Bienestar Universitario



UACJ | IIT

**La Universidad Autónoma de Ciudad Juárez
a través del Instituto de Ingeniería y Tecnología**

**Otorga el presente
RECONOCIMIENTO**

A

Ángel Hiram Alvidrez Hernández
Por su participación como ponente de la Conferencia:
“¿Estamos solos en el universo?”,
durante la XXX Semana de Ingeniería.
Septiembre 2024

“Por una vida científica,
Por una ciencia vital”

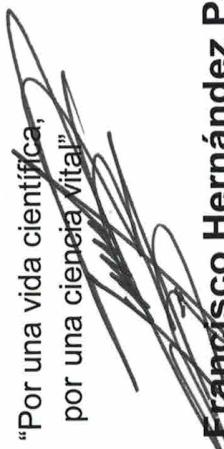
Dr. Juan Francisco Hernández Paz
Director del Instituto de Ingeniería y Tecnología

La Universidad Autónoma de Ciudad Juárez
a través del Instituto de Ingeniería y Tecnología

Otorga el presente

RECONOCIMIENTO al Club de Astronomía

Por impartir la conferencia:
“Las muertes del universo”.
Durante la XXIX Semana IIT


“Por una vida científica,
por una ciencia vital”

Dr. Juan Francisco Hernández Paz

Director del Instituto de Ingeniería y Tecnología

Universidad Autónoma de Ciudad Juárez

Laboratorio de Robótica

Departamento de Ingeniería Eléctrica y Computación



Proyecto semestral

Motores y controladores
Edgar Alonso Martínez García

Integrantes del Equipo:

Ingeniería Mecatronica: Alan Quintana Quezada - 214621

Ingeniería Mecatrónica: Angel Hiram Alvidrez Hernandez - 225581

Ingeniería Mecatrónica: Viridiana Ahumada Velázquez - 225243

Fecha: 21 de noviembre 2025

Índice

1. Introducción	1
2. Mecanismo de 5 barras	2
2.1. Diseño en CAD	2
2.2. Cinemática del mecanismo	2
3. Cicloide	7
3.1. Movimiento de los servos	7
3.1.1. Comunicación Maestro → Esclavo	7
3.1.2. Velocidad angular	8
3.2. Cicloide generada	8
4. Sensado	9
4.1. Fusión de datos	9
4.1.1. Comunicación Maestro ← Esclavo	10
5. Estrategia de Control	11
6. Conclusiones	13

1. Introducción

Este proyecto presenta la elaboración de un bípedo subactuado basado en mecanismos de cinco barras por extremidad, con un diseño mecánico modelado en CAD y un ensamble perfeccionado mediante sucesivas etapas de ajuste y mejora del prototipo orientado a lograr movimientos cílicos controlados y pruebas iniciales de caminata. También se utiliza un Orange Pi como computadora principal y se integra una IMU para registrar orientación y aceleraciones durante postura estática, trayectorias cicloides y caminata.

2. Mecanismo de 5 barras

Se implementaron dos mecanismos de cinco barras dispuestos en paralelo, uno por extremidad. Este es un mecanismo subactuado que posee dos grados de libertad. Las articulaciones activas son accionadas mediante un par de servomotores MG996R, mientras que las articulaciones pasivas se lograron utilizando baleros. Se agregaron dos servomotores mas por extremidad, en la conexión a la cadera y la conexión del pie del robot.

2.1. Diseño en CAD

En la figura 1 se muestra de forma aislada uno de los mecanismos de cinco barras correspondientes a la parte superior de la extremidad. Por otro lado, en la figura 2 se presentan las cinco barras con un código de colores, descrito en la tabla 1, junto con las longitudes de cada una expresadas en milímetros (mm). Una vez terminado el diseño CAD del mecanismo de 5 barras, se termino el diseño de todo el robot bipedo, mostrado en la figura 3

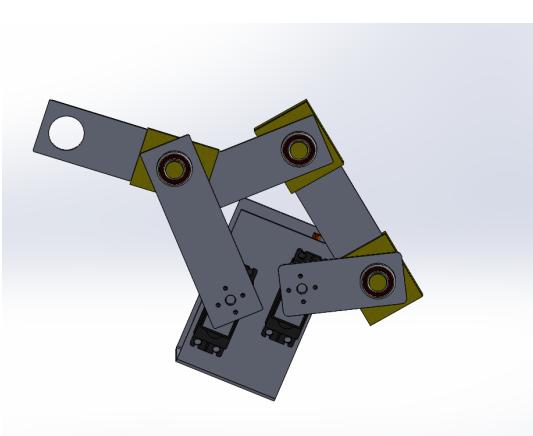


Figura 1: Diseño 1.

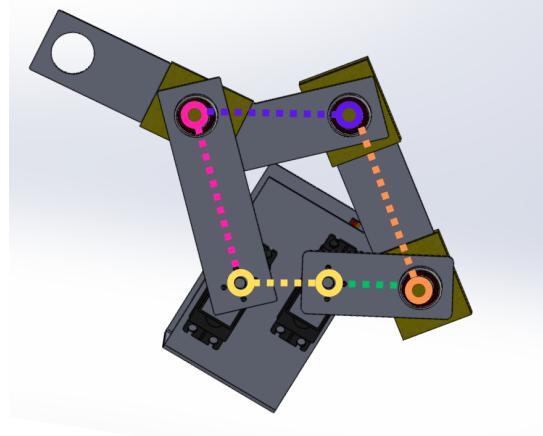


Figura 2: Definición de las 5 barras.

2.2. Cinemática del mecanismo

Una vez definidas las cinco barras del mecanismo, se procede al análisis cinemático con el objetivo de determinar la posición, en milímetros, de la punta superior del mecanismo a partir de los ángulos de los motores θ_1 y θ_4 , como se muestra en la figura 4.

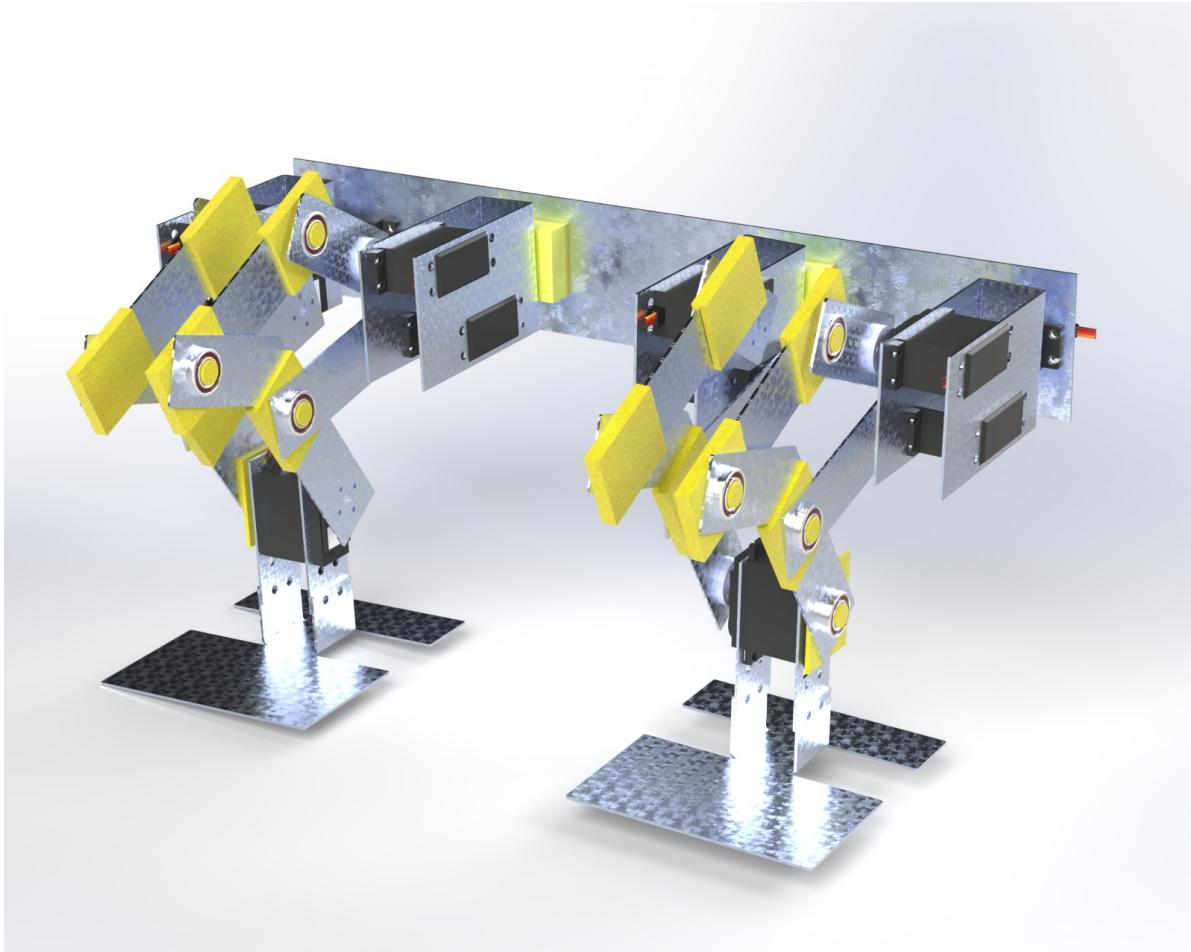


Figura 3: Render del diseño.

Tabla 1: Título de la tabla

Barra	Longitud	Color
L_0	40.84 mm	Amarillo
L_1	43.5 mm	Verde
L_2	88.67 mm	Naranja
L_3	67.07 mm	Purpura
L_4	83 mm	Rosa

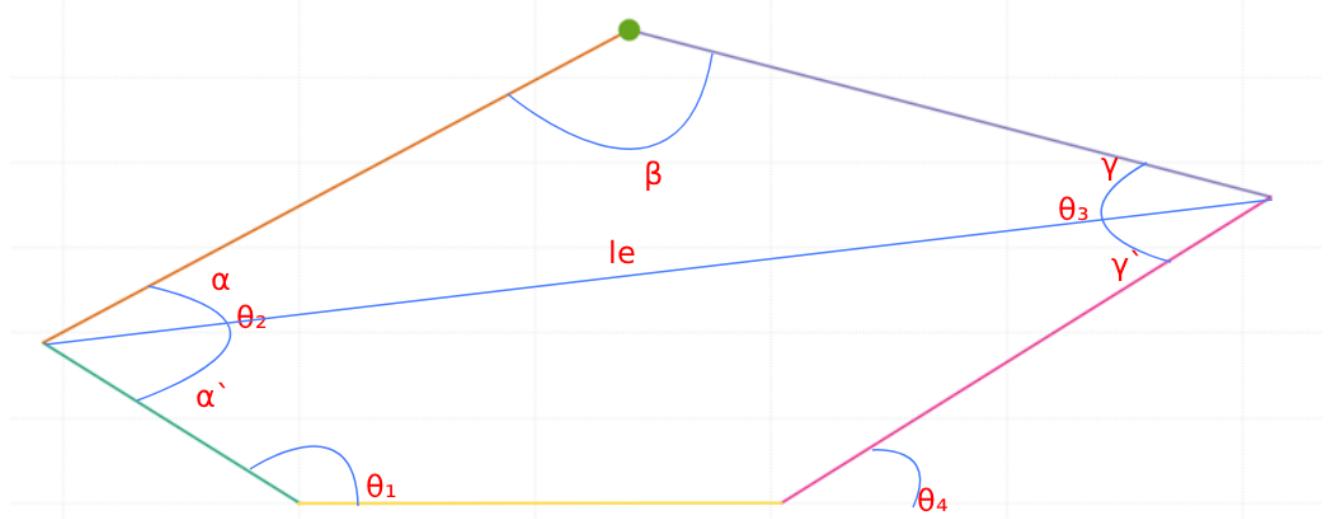


Figura 4: Visualización y nomenclatura de los ángulos del mecanismo.

Se comienza definiendo la posición en los ejes X y Y del extremo del eslabón L_1 y L_4 mediante las siguientes expresiones:

$$X_1 = L_1 \cos(\theta_1); \quad Y_1 = L_1 \sin(\theta_1);$$

$$X_4 = L_4 \cos(\theta_4); \quad Y_4 = L_4 \sin(\theta_4);$$

Estas posiciones son necesarias para conocer la longitud de nuestra "Barra imaginaria" L_e , definida por:

$$L_e = \sqrt{(X_1 - X_4)^2 + (Y_1 - Y_4)^2}$$

L_e nos permite ver si la combinación de ángulos de θ_1 y θ_4 es posible sin romper nuestro mecanismo, pues si la sumatoria de L_2 y L_3 es menor que nuestra recién calculada L_e los motores serían incapaces de llegar a los ángulos indicados sin destruir el mecanismo o a ellos mismos.

Continuando con el análisis, podemos utilizar la ley de senos cosenos y despejarla para conocer

los ángulos internos β , α y γ con las expresiones:

$$\beta = \arccos\left(\frac{L_e^2 - L_2^2 - L_3^2}{-2 \times L_2 \times L_3}\right)$$

$$\alpha = \arcsin\left(\frac{L_3}{L_e} \times \sin(\beta)\right)$$

$$\gamma = \arcsin\left(\frac{L_2}{L_e} \times \sin(\beta)\right)$$

Para poder continuar tenemos que declarar otro triangulo dibujando una línea imaginaria, el cual va de la punta superior de L_1 hasta la punta derecha de L_0 , como se puede observar en la figura 5.



Figura 5: Visualización de L_{01} como la linea punteada.

Y utilizando la ley de cosenos podemos encontrar la longitud de esta linea imaginaria.

$$L_{01} = \sqrt{L_0^2 + L_1^2 - 2L_0L_1\cos(\theta_1)}$$

con la cual podemos calcular la versión local de el angulo γ utilizando:

$$\gamma' = \arccos\left(\frac{L_{01}^2 - L_4^2 - L_e^2}{-2L_eL_4}\right)$$

la cual nos sirve para conocer el angulo interno θ_3 sumando γ con γ' y también necesitaremos el complementario de estos ángulos. Con los cuales podemos llegar finalmente a las posiciones cartesianas del efector final del mecanismo de 5 barras con:

$$P_X = L_0 + L_4 \times \cos(\theta_4) + L_3 \times \cos(\theta_4 + \theta_3')$$

$$P_y = L_0 + L_4 \times \sin(\theta_4) + L_3 \times \sin(\theta_4 + \theta_3')$$

Una vez conocida una condición que limite nuestro mecanismo, podemos calcular todos las posibles localizaciones de nuestro mecanismo y tener nuestro espacio de trabajo seguro, mostrado en la figura 6

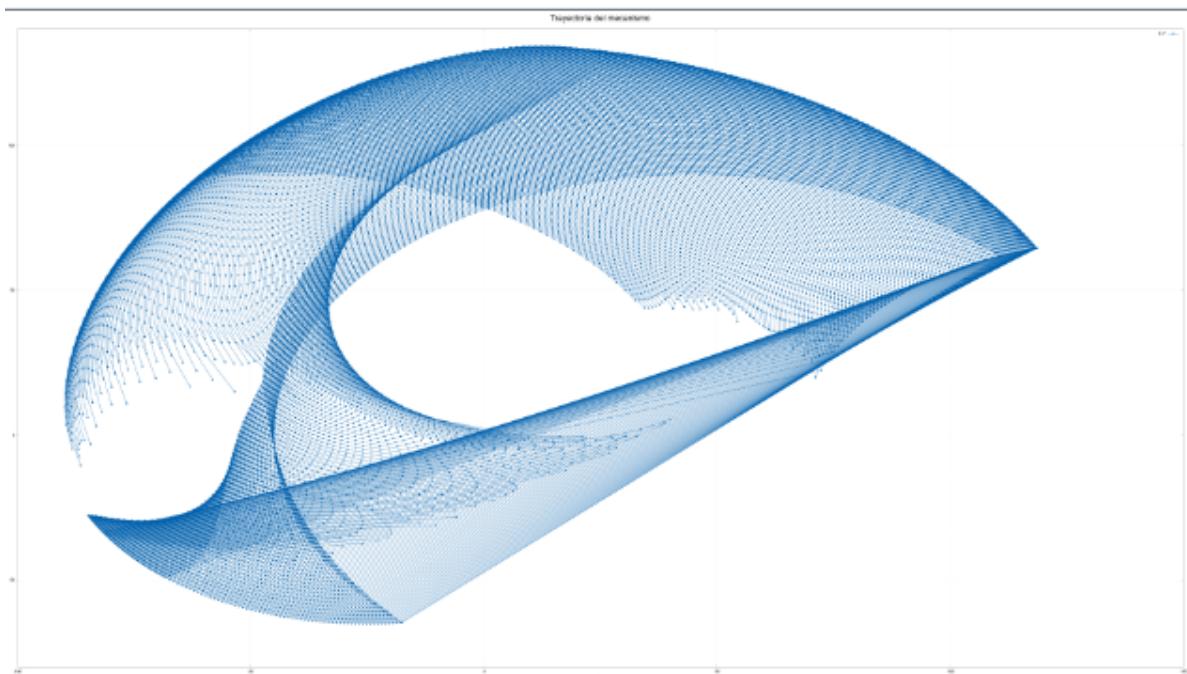


Figura 6: Espacio de trabajo.

3. Cicloide

La cicloide es la forma de movimiento que el mecanismo de cinco barras sigue para simular una caminata.

Para que el mecanismo siga el movimiento deseado, se aplicaron las siguientes ecuaciones en función del tiempo (t):

$$\theta_1 = a + c \times \sin(t) \quad (1)$$

$$\theta_4 = b + c \times \sin\left(t + \frac{\pi}{2}\right) \quad (2)$$

dónde:

a : Ángulo inicial de θ_1 ,

b : Ángulo inicial de θ_4 ,

c : Amplitud de oscilación.

3.1. Movimiento de los servos

Una vez definidos los movimientos articulares, se enviaron los comandos a los servomotores mediante una ESP-WROOM-32 y una tarjeta expansora con fuentes de alimentación independientes para control y potencia.

3.1.1. Comunicación Maestro → Esclavo

La computadora embebida al bípedo le manda las señales de movimiento mediante puerto serial utilizando los puertos USB de la computadora y la tarjeta de control y sensado.

Se manda un telegrama en el cual se especifica que servomotor se va a mover a que ángulo y con que velocidad.

3.1.2. Velocidad angular

Para el manejo de la velocidad angular de los servos, se optó inicialmente por el uso de librerías, pero estas no lograban un buen control de varios servos simultáneos. Para solucionar esto se desarrolló un algoritmo;

$$\theta_t = \theta_0 + \sum_{k=1}^N \text{sgn}(\theta_{i+N} - \theta_i(k-1)) \Delta\theta_i$$

Siendo $\Delta\theta_i$ los intervalos angulares que toma el servo, calculados por:

$$\Delta\theta_i = \frac{1}{\text{vel}_i}$$

Los intervalos de tiempo fueron calculados usando un contador de tiempo del microcontrolador, en vez de utilizando interrupciones, para así poder manejar varios servos simultáneamente.

Todo esto fue hecho para crear una velocidad constante y suave en los servomotores, enviándoles a estos ángulos mediante modulación de ancho de pulsos, pues es el único dato que pueden recibir los servos.[1]

3.2. Cicloide generada

Una vez conocemos la cinemática de nuestro mecanismo y tenemos control de los actuadores de este, se procedió a utilizar la función de la cicloide y a realizar varias iteraciones en los parámetros de la ecuación 1 y 2 hasta tener una respuesta que permitiera al robot desplazarse manteniendo su balance. La respuesta del efecto final del mecanismo de 5 barras es mostrado en la figura 7

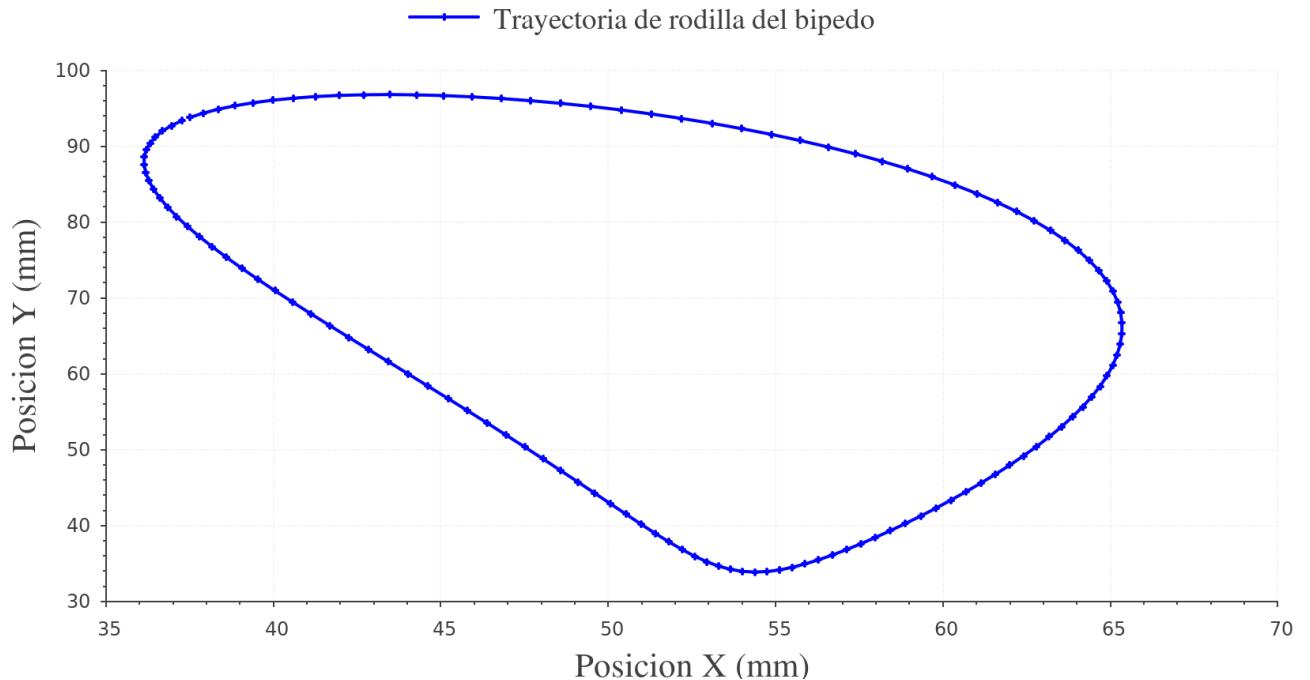


Figura 7: Efector final del mecanismo de 5 barras.

4. Sensado

Para evitar que el Bípedo se caiga, este tiene que saber que esta cayendo, por lo que se le implemento una IMU.

La IMU seleccionada fue la BNO080X, cuyos sensores son mostrados en la tabla 2

4.1. Fusión de datos

La BNO08x es un Sistema en Paquete (SiP) que incluye un microcontrolador dedicado y sus propios sensores. El secreto de su rendimiento radica en que el microcontrolador ejecuta el algoritmo de fusión dentro del chip, sin necesidad de intervención del procesador principal del usuario. [2]

Tabla 2: IMU BNO08X

SENSOR	MIDE	PROBLEMA PRINCIPAL
Acelerómetro	Gravedad y aceleración lineal.	Es sensible al ruido de vibración
Giroscopio	Velocidad angular (rotación).	Sufre de derivación (drift)
Magnetómetro	Campo magnético terrestre (brújula).	Es muy sensible a la interferencia de metales.

4.1.1. Comunicación Maestro ← Esclavo

Una vez definida nuestra IMU, esta se comunicará con el ESP-32 mediante el protocolo I²C, y estará leyendo los datos de la IMU constantemente. Hasta que la Orange PI le mande un telegrama donde solicite los datos de la IMU, entonces el ESP-32 enviará los últimos datos que haya guardado en memoria. Este algoritmo de comunicación se ejemplifica visualmente en la figura 8.

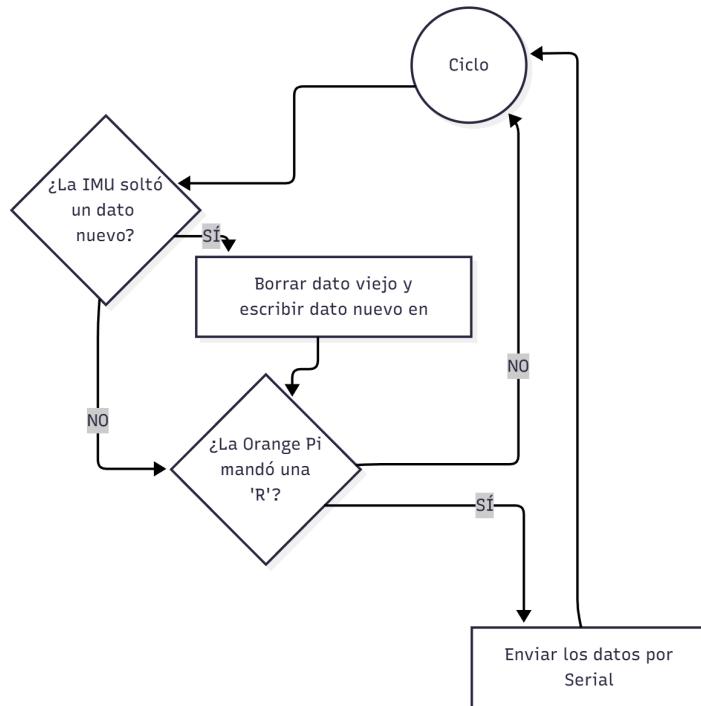


Figura 8: Diagrama de secuencia de la comunicación Esclavo → Maestro.

5. Estrategia de Control

El robot monitorea activamente su inclinación en tiempo real para tomar decisiones de seguridad, alternando entre un modo de Caminata y un modo de Recuperación.

El algoritmo de control implementado en C++ ejecuta el siguiente ciclo lógico en cada iteración, mostrado también en la figura 9:

1. **Adquisición de Datos (Feedback):** El Computador envía el telegrama de adquisición de datos al módulo de sensado. El subsistema de la IMU responde enviando el ángulo de inclinación actual (*Pitch*).
2. **Evaluación de Estabilidad:** El algoritmo verifica si el robot se encuentra dentro de un rango seguro de operación, definido experimentalmente como $\pm 20^\circ$ de inclinación en el eje de Pitch.
3. **Toma de Decisiones:**
 - **Caso Estable ($|Pitch| < 20^\circ$):** Si la inclinación es segura, el sistema ejecuta el siguiente paso de la trayectoria cicloide. Se calculan las coordenadas (*X*, *Y*), se resuelve la cinemática inversa y se mueven los servos para avanzar.
 - **Caso Inestable (Caída Detectada):** Si el umbral es superado, se interrumpe temporalmente la generación de la cicloide y se activa un Reflejo de Seguridad":
 - Si $Pitch > 20^\circ$ (Caída frontal): La extremidad libre se desplaza rápidamente hacia adelante para ampliar el polígono de soporte.
 - Si $Pitch < -20^\circ$ (Caída trasera): La extremidad se desplaza hacia atrás para contrapesar la caída.

- 4. Reanudación de Marcha:** El sistema mantiene la posición de seguridad hasta que las lecturas de la IMU regresan al rango de $\pm 20^\circ$. Una vez estabilizado, el algoritmo retoma la ejecución de la caminata cicloide desde el punto donde se interrumpió.

Este enfoque permite una caminata cíclica predecible, delegando la estabilidad al diseño mecánico y al filtrado pasivo de las perturbaciones.

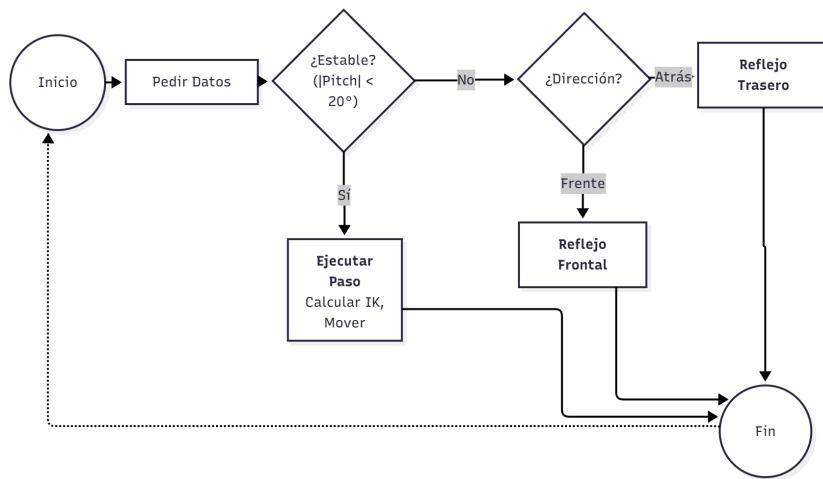


Figura 9: Diagrama de secuencia de la caminata

Con esta arquitectura de Control, aunque sea un poco "primitiva", logró dar una caminata en la que el bípedo se mantuvo estable sin necesitar de apoyo, logrando así caminar solo. [3].

Mientras la caminata se efectuaba, los datos de la IMU eran almacenados para ser posteriormente graficados, los cuales se pueden apreciar en la figura 10

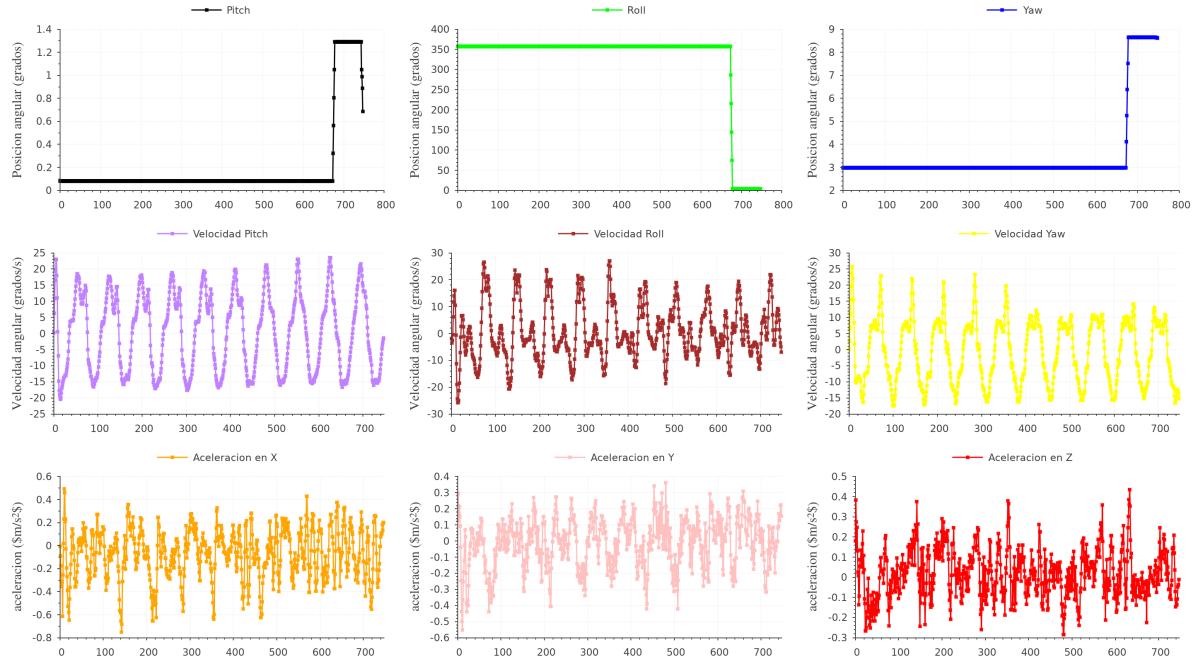


Figura 10: Diagrama de secuencia de la caminata

6. Conclusiones

- **Conclusión 1:** El proyecto a realizar fue el desarrollo de un robot bípedo caminante, implementando una arquitectura de cinco barras para la cinemática del sistema, pero tuvimos problemas con la estructura del bípedo por unas dimensiones y los pesos que llegaba soportar, lo mas que nos llevo tiempo en trabajar en el bípedo fue en la cicloide ya que mientras buscábamos los ángulos correctos teníamos que estar a prueba y error para que llegara hacer la caminata como nosotros quisiéramos para que se mantuviera recto y pudiera hacer su trayectoria sin que se troncara por su peso
- **Conclusión 2 :**En resumen, el proyecto fue una gran experiencia trabajar en hacer el bípedo ya que requiere tanto la solución de problemas de ingeniería mecánica (peso y estructura) como la precisión en la implementación de las funciones matemáticas (la cicloide) para lograr una buena caminata.

- Conclusión 3: En general, el proyecto del bípedo logró cumplir con su objetivo principal: generar una caminata estable a partir de un mecanismo de cinco barras controlado mediante una arquitectura simple, aunque aún quedan muchas mejoras por implementar.

Durante el desarrollo se identificaron varias oportunidades que podrían optimizar tanto el rendimiento como la eficiencia del sistema. Por ejemplo, no se aprovechó la capacidad de los múltiples puertos seriales disponibles en la Orange Pi ni en el ESP32, lo cual habría permitido separar la comunicación del sensado y del control de los servomotores, evitando cuellos de comunicación y posibles bloqueos.

Tampoco se explotó el potencial de los dos núcleos del ESP32 ni de la Orange Pi, donde podría haberse asignado un núcleo exclusivamente a la adquisición de datos y otro al control de movimiento, logrando un control verdaderamente paralelo y más fluido.

Otro punto pendiente fue la implementación de un filtro de Kalman, que habría permitido suavizar los datos de la IMU y generar una caminata más precisa y controlada, en lugar de depender de una estrategia algo “a fuerza bruta”.

Finalmente, me hubiera gustado realizar un análisis dinámico más profundo mediante el formalismo de Lagrange, para que el bípedo aprovechara mejor la gravedad y lograra un movimiento más natural y eficiente.

Aun con estas limitaciones, el proyecto sirvió como una base sólida para comprender la integración entre la parte mecánica, electrónica y de control de un robot bípedo, y deja un buen punto de partida para versiones más avanzadas en el futuro.

Referencias

- [1] Tower Pro. Mg996r high torque metal gear dual ball bearing servo. https://www.electronicoscaldas.com/datasheet/MG996R_Tower-Pro.pdf, 2018. Hoja de datos, accedido: noviembre 2025.
- [2] CEVA Hillcrest Labs. Imu de la serie bno08x. <https://www.digikey.com.mx/es/product-highlight/h/hillcrest-labs/bno080-imu>, 2024. Accedido: noviembre 2025.
- [3] Angel Alvidrez. Video de caminata del bipedo. <https://www.youtube.com/shorts/CoHYQln-zsA>, 2025. Youtube Short.

Actualización de la electrónica de control de un telescopio Orión de 8 Pulgadas

El presente proyecto consistió en la modernización de la electrónica de un telescopio Orión cuyo controlador original dejó de funcionar por desgaste ambiental. Este telescopio cuenta con dos codificadores por eje, uno de alta resolución para las trayectorias y seguimiento fino del objeto y el de baja resolución para el apuntado. Dado que los motores y los codificadores aún estaban en buen estado, se optó por reutilizarlos e implementar un nuevo sistema de control basado en una arquitectura modular.

Se utilizaron dos microcontroladores ESP32, uno para cada eje (altitud y azimut), los cuales se encargan de controlar los movimientos de cada motor con ayuda de un puente H L298N, a su vez de procesar en tiempo real las señales de codificadores ópticos. Para ello fue necesario diseñar una etapa de acondicionamiento electrónico a cada codificador, que incluye amplificación de señal mediante un LM324 en configuración no inversora, corrección de offset con un potenciómetro, y digitalización utilizando compuertas inversoras 74HCT05.

El circuito de acondicionamiento fue probado en protoboard y se pasó a un circuito impreso que se realizó en el mismo laboratorio.

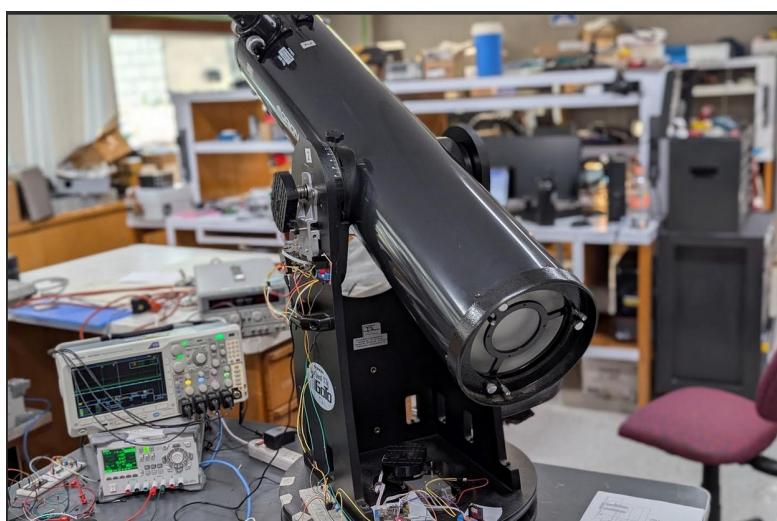
Cada ESP32 calcula trayectorias en base a los pulsos del codificador y transmite la información de posición y estado a una Raspberry Pi mediante comunicación serial. Esta Raspberry funciona como nodo central del sistema, ejecutando un script en Python que recibe coordenadas y permite enviar comandos al sistema.

Además, se está integrando el software Stellarium el cual permitirá al usuario seleccionar objetos celestes en tiempo real. Stellarium enviará las coordenadas directamente al Raspberry Pi. Esta función se encuentra en etapa final de integración y pruebas

Autores:

Ángel Alvídrez – Instituto de Ingeniería y Tecnología, Universidad Autónoma de Ciudad Juárez - aalvidrezhdz@gmail.com

Francisco Murillo - Asesor



Amplificador de audio basado en transistores

Karla V. Suárez, *Estudiante, UACJ, IIT, Depto. Ing. Eléctrica y Computación,*
 Ángel H. Alvidrez, *Estudiante, UACJ, IIT, Depto. Ing. Eléctrica y Computación,*
 Viridiana A. Velázquez, *Estudiante, UACJ, IIT, Depto. Ing. Eléctrica y Computación*

Abstract—Este documento presenta el diseño, construcción y simulación de un amplificador de audio basado en transistores utilizando una configuración de divisor de voltaje. El circuito incorpora un transistor NPN 2N2222A y un transistor TIP32C para amplificar señales de audio de baja potencia destinadas a dispositivos de salida como altavoces. La metodología incluye cálculos detallados y procedimientos para lograr un diseño eficiente. Aunque se logró amplificar las señales de audio de manera satisfactoria, el prototipo revela áreas de mejora, como la gestión térmica, la reducción de la distorsión y la precisión en la selección de componentes, lo que sugiere futuras optimizaciones para un mejor rendimiento y confiabilidad. Entre los temas abordados se encuentran la polarización de transistores, la respuesta en frecuencia y los desafíos prácticos en la implementación de circuitos electrónicos.

Index Terms—Amplificador de audio, transistores, divisor de voltaje, electrónica, polarización de transistores, frecuencia, altavoces, capacitores, resistencias.

I. INTRODUCCIÓN

Un amplificador de audio basado en transistores es un circuito electrónico diseñado específicamente para aumentar la potencia de las señales de audio que reciba, esto permite garantizar su reproducción en dispositivos de salida o en altavoces, permitiendo que el audio sea con mayor volumen y claridad. Como su nombre lo menciona, estos dispositivos se basan en el principio de amplificación, donde un transistor controla una corriente mayor en función de una señal de entrada mucho más débil [1], [2]. Este mismo principio suele tener distintas aplicaciones en áreas de tecnología como sistemas de sonido, instrumentación musical, sonido en celulares, entre otros.

Una de las mayores características de un amplificador es su gran capacidad para procesar señales de baja potencia y convertirlas en señales más fuertes que puedan ser controladas por medio de altavoces. Esto se logra mediante el uso de transistores permitiendo realizar múltiples configuraciones, en las que se destacan el *emisor común*, *colector común* y configuraciones push-pull. Cada una de estas configuraciones tiene características específicas que las hacen adecuarse para distintas etapas de amplificación [3], [4].

A. Principio de funcionamiento de un transistor

El transistor entra en la categoría de ser un dispositivo semiconductor que cuenta con tres terminales; *emisor*, *base* y *colector* [1]. En la Fig. 1 se observa el diagrama visual de transistores bipolares.

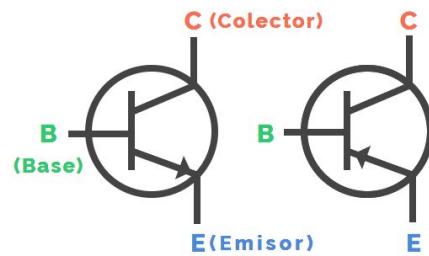


Fig. 1: Diagrama de símbolos de transistores bipolares: NPN (izquierda) y PNP (derecha), mostrando sus terminales: Base (B), Colector (C) y Emisor (E) [5].

La función principal de un transistor es controlar el flujo de corriente, actuando como interruptor o bien, como amplificador. Al momento de amplificar una señal, el transistor es capaz de aumentar la amplitud de una señal débil aplicada en su base, dando resultado a una señal de mayor potencia en el colector [1].

La relación entre un transistor y un amplificador, es que el transistor cuenta con sus tres terminales que pueden diseñarse como amplificadores con características específicas de ganancia, impedancia y frecuencia [3].

Mientras que un transistor es un componente individual capaz de controlar y amplificar señales, un amplificador es un circuito completo que funciona con transistores, entre otros componentes, con el fin de aumentar la amplitud de una señal de entrada según las necesidades deseadas [4].

II. MARCO TEÓRICO: AMPLIFICADOR DE AUDIO POR MEDIO DE UN DIVISOR DE VOLTAJE

Existen distintas configuraciones al momento de amplificar una señal, en esta sección se habla sobre un amplificador de audio basado en un divisor de voltaje, que básicamente es un circuito que utiliza esta configuración en específico para establecer el nivel de polarización necesario en el transistor, permitiendo que así, opere de manera eficiente en la región activa. Esta configuración se caracteriza por ser simple y confiable, ya que el divisor de voltaje, formado por dos resistencias en serie, divide la fuente de alimentación para generar un voltaje constante que controla el comportamiento que cuenta el transistor [2].

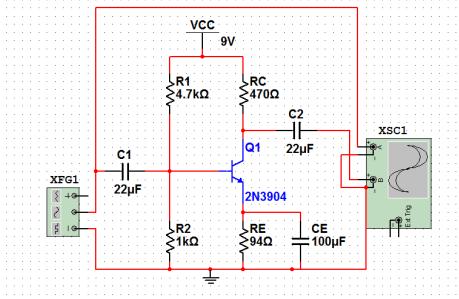


Fig. 2: Esquema de un amplificador de audio basado en un divisor de voltaje utilizando un transistor 2N3904. Incluye resistencias para la polarización (R_1 y R_2), resistencia de colector (R_C), resistencia de emisor (R_E), un capacitor de desacoplo (CE) y capacitores de acople (C_1 y C_2) para señal de entrada y salida, alimentado por una fuente de 9V [6].

En la Fig. 2 se presenta el esquema de un amplificador de audio basado en un divisor de voltaje utilizando un transistor NPN modelo 2N3904. El circuito emplea resistencias para polarización, componentes de acople y desacoplo, y una fuente de alimentación de 9V.

Un divisor de voltaje está compuesto típicamente por dos resistencias en serie conectadas entre el voltaje de alimentación (V_{CC}) y tierra. La tensión en el punto de unión de estas resistencias se utiliza para polarizar la base del transistor. Si R_1 y R_2 son las resistencias del divisor, la tensión de polarización está dada por:

$$V_B = \frac{R_2}{R_1 + R_2} V_{CC} \quad (1)$$

Este voltaje determina la corriente de base (I_B) del transistor, que a su vez controla la corriente de colector (I_C) según la relación:

$$I_C = \beta I_B \quad (2)$$

donde β es la ganancia de corriente del transistor.

A. Componentes

El circuito también incluye resistencias y capacitores adicionales para mejorar su desempeño:

- R_E y R_C : La resistencia R_E en el emisor ayuda a estabilizar el punto de operación, mientras que R_C determina la ganancia del amplificador.
- C_E : El capacitor de desacoplo C_E elimina la componente de corriente alterna en R_E , incrementando la ganancia del amplificador.
- C_1 y C_2 : Los capacitores de acople permiten que solo las señales de corriente alterna pasen entre las etapas de entrada y salida, bloqueando cualquier componente de corriente directa.

La respuesta en frecuencia del amplificador está determinada por los capacitores y resistencias del circuito. La frecuencia de corte inferior (f_c) de un capacitor C en serie con una resistencia R se calcula como:

$$f_c = \frac{1}{2\pi RC}. \quad (3)$$

Este parámetro es crucial para garantizar que el amplificador funcione en el rango de frecuencias (20 Hz - 20 kHz).

III. METODOLOGÍA

Para llevar a cabo este proyecto, se siguió un proceso estructurado pero práctico [7], que abarca desde la idea inicial hasta la validación del diseño final. A continuación, se describen las etapas realizadas.

A. Diseño del circuito

Para el diseño de circuito de un amplificador de audio basado en transistores, se utilizó como base un circuito amplificador de audio por medio de un divisor de voltaje. Para obtener resultados más eficientes en el proyecto, se realizaron dos amplificadores, es decir, se dividió el diseño del circuito en dos fases.

1) *Primera etapa de amplificación*: En la primera etapa de amplificación, se realizó el boceto del circuito (Fig. 3.) y después los cálculos pertinentes. En esta fase, se encuentra la entrada a la guitarra, seguido de la conexión de un transistor en división de voltaje con capacitores.

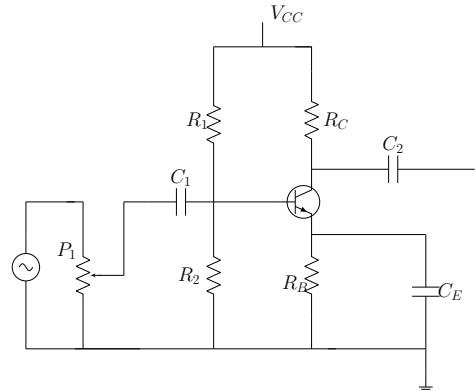


Fig. 3: Esquema inicial de la primera etapa de amplificación. Este circuito preliminar define la configuración básica para la amplificación de una señal, incluyendo el uso de un transistor como elemento activo y componentes pasivos como resistencias y capacitores para estabilizar y controlar la ganancia. Los valores de los componentes, como R_1 , R_2 , R_E , C_1 , C_2 , y C_E , aún no han sido determinados, lo que permite su ajuste posterior para cumplir con las especificaciones de diseño.

En la primera etapa de amplificación se emplea el uso de un transistor 2N2222A como un preamplificador, el cual se espera que consiga una ganancia de corriente en el colector de 100, para esto se escogieron las resistencias adecuadas para asegurar una corriente de colector de 15 mA y una corriente de base de 150 μ A. Para escoger estas resistencias se empezó el análisis haciendo que en los pines de colector y emisor exista una tensión de el 50% de nuestra fuente para tener un rango de señal sin tanta distorsión, en este caso 18V, en la resistencia del emisor se espera un 10% de la fuente. Por lo tanto, para

determinar el valor de la resistencia necesaria, se llevó a cabo el siguiente cálculo.

$$R_E = \frac{10\% \times 18V}{15mA} = 120\Omega \quad (4)$$

Para la resistencia del colector se asumió que la corriente en el colector y en el emisor es la misma ($I_E = I_C$), para conseguir el 50% del voltaje de la fuente en emisor colector, a la resistencia del colector le tiene que caer el 40% de la fuente, haciendo un cálculo similar al anterior, obtenemos:

$$R_C = \frac{40\% \times 18V}{15mA} = 480\Omega \quad (5)$$

Para las resistencias del divisor de tensión ubicado en la base del transistor nos aseguramos de que R_2 sea 10 veces más grande que la de emisor garantizando una corriente de base menor que en el colector y emisor.

$$R_2 = 10 \times R_e = 1.2K\Omega \quad (6)$$

Y para la última resistencia se realizó el análisis de el voltaje que caerá en la base del transistor, esta tensión se conoce por las leyes de Kirchhoff pues el voltaje en el pin emisor del transistor escogimos que sería el 10% de nuestra fuente de voltaje, y la caída de voltaje que hay de base a emisor es de aproximadamente 0.7V, con este voltaje podremos calcular la caída de voltaje esperada en la primera resistencia y así calcularla con la relación de voltajes entre base, la caída esperada y R_2 .

$$V_{be} \approx 0.7V, V_e = 1.8V \quad (7)$$

$$V_b = 1.8 + 0.7 = 2.5V \quad (8)$$

$$V_{R1} = 18 - 2.5 = 15.5V \quad (9)$$

$$R_1 = \frac{15.5}{2.5} \times R_2 = 10.33K\Omega \quad (10)$$

Finalmente calculamos la potencia que disipará cada resistencia para asegurarnos de comprar el tipo adecuado.

$$P_{RE} = 1.8V \times 15mA = 27mW \quad (11)$$

$$P_{RC} = 7.2V \times 15mA = 108mW \quad (12)$$

$$P_{R2} = \frac{2.5V^2}{1200\Omega} = 5.2mW \quad (13)$$

$$P_{R1} = \frac{15.5V}{10.333K\Omega} = 23.25mW \quad (14)$$

2) Segunda etapa de amplificación: Para el amplificador de poder, escogimos el transistor TIP32-C pues tiene una capacidad de corriente mayor y una dissipación de poder moderada, pero que puede ser mejorada hasta 20 veces con un disipador de calor, a este transistor se le colocó una corriente de $I_C = 150mA$ en su colector y su corriente de base fue ajustada a $I_B = 15mA$ para ser conectado a la primera etapa de amplificación.

El procedimiento utilizado para determinar los valores de los componentes en la primera etapa de amplificación se aplicó, de manera análoga, para calcular los valores correspondientes a los componentes en la segunda etapa de amplificación. Esto permitió garantizar la continuidad en el diseño y la coherencia en el desempeño del circuito.

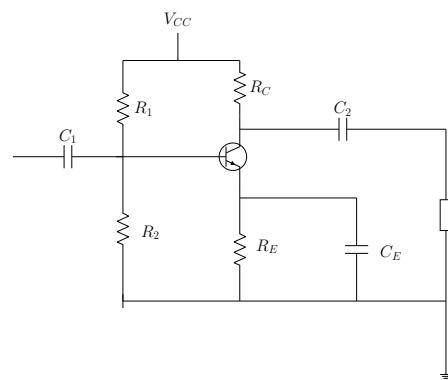


Fig. 4: Esquema del circuito final de amplificación. Este diseño muestra la configuración completa de la segunda etapa amplificadora, donde R_1 y R_2 forman un divisor de voltaje para polarizar el transistor. Los capacitores C_1 y C_2 actúan como acopladores para la señal de entrada y salida, respectivamente, mientras que C_E proporciona una mayor ganancia al desacoplar la resistencia R_E . El circuito está diseñado para proporcionar una amplificación estable y eficiente.

3) Diseño final de circuito: El diseño final del circuito (Fig. 5.) fue el resultado de un proceso de ajustes basado en las etapas de amplificación previas. El objetivo principal era optimizar la ganancia total del sistema y garantizar que el circuito fuera estable y eficiente. Este diseño incorpora componentes cuidadosamente seleccionados para cumplir con las necesidades específicas de amplificación.

B. Simulación del circuito

La simulación del circuito en su etapa final se realizó utilizando el software gratuito Tinkercad, es una herramienta muy útil para diseñar y simular circuitos electrónicos de una manera visual e interactiva. El diseño implementado incluye un transistor 2N2222A y un TIP32C, los cuales cumplen el rol fundamental para la amplificación del audio, los cuales fueron seleccionados gracias a que sus características de ganancia y capacidad de manejo de potencia (Fig. 6.).

La configuración del circuito incluye el divisor de voltaje que se mencionaba anteriormente, formado por resistencias en la base del transistor 2N2222A, pues esto establece el punto de polarización adecuado para que el transistor funcione

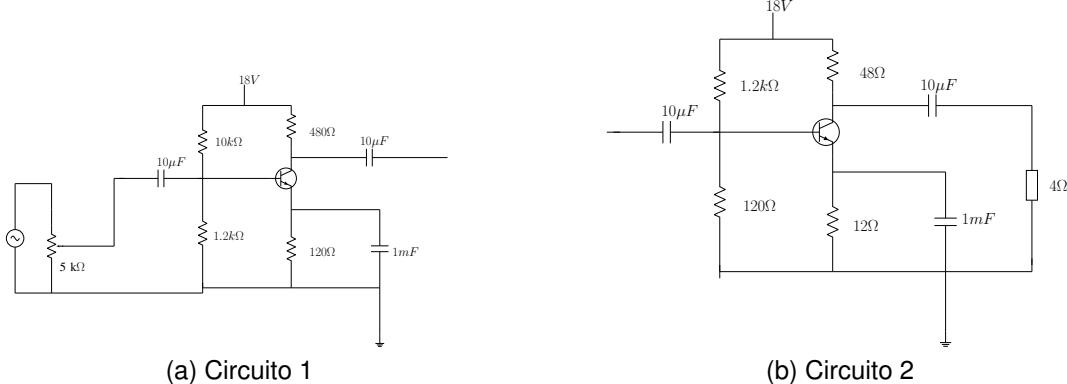


Fig. 5: Circuito de amplificación en dos etapas. (a) Primera etapa de amplificación: se presenta el diseño inicial del circuito, donde los valores de los componentes, como resistencias y capacitores, fueron calculados previamente para asegurar un aumento eficiente de la señal inicial. Esta etapa establece la ganancia básica del sistema. (b) Segunda etapa de amplificación: muestra el diseño final del circuito, donde los valores calculados de los componentes optimizan la ganancia total y la estabilidad del sistema. Esta etapa refuerza la amplificación lograda en la primera, asegurando una señal de salida adecuada para el propósito deseado.

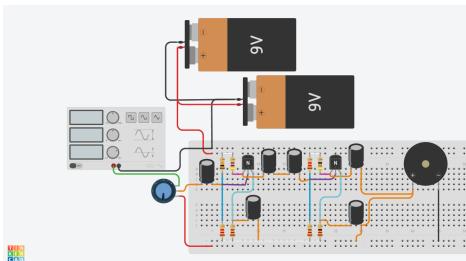


Fig. 6: Visualización de la simulación de la etapa final del circuito del amplificador, hecha en Tinkercad.



Fig. 7: El transistor de la imagen tiene el mismo encapsulado (TO220) que el TIP32C utilizado en el diseño del circuito [8].

adecuadamente. Por otra parte, se utilizaron capacitores de acoplamiento (C_1 y C_2) para asegurar una transferencia limpia de la señal de corriente alterna entre cada capa, así, eliminando la corriente directa no deseada de otros componentes. Y por otro lado, se integró un capacitor de desacoplamiento (C_E) el cual nos ayuda a eliminar la caída de voltaje en la resistencia del emisor, así maximizando la ganancia.

Para la fuente de alimentación, se determinó utilizar 18V, por lo que se optó por utilizar dos baterías 9V cada una, las cuales suministran la energía necesaria para el funcionamiento del circuito.

C. Construcción del circuito

Tras simular el circuito y determinar los valores óptimos para cada componente, se procedió a ensamblarlo en el protoboard, realizando posteriormente las pruebas y ajustes necesarios para garantizar su correcto funcionamiento.

IV. RESULTADOS

Con la construcción del amplificador de audio, se realizaron las pruebas prácticas para evaluar su comportamiento en condiciones reales. El circuito fue conectado a un altavoz

Radox 065-176 de 5 pulgadas con capacidad de 15W y una impedancia de 4Ω , reproduciendo señales de audio de una guitarra eléctrica. Los resultados mostraron que logra exitosamente amplificar la señal de audio, permitiendo su salida con un volumen perceptible. Sin embargo, se distingue distorsión en la salida. Adicionalmente, se observó un aumento en la temperatura de los transistores y algunas resistencias durante el funcionamiento prolongado. Incluso con el uso de un disipador de calor, no es suficiente para que la temperatura se mantenga en un rango óptimo, evidenciando la necesidad de mejoras en la gestión térmica del circuito. Esta elevada disipación térmica afecta la eficiencia y la durabilidad del amplificador, requiriendo el uso de disipadores más potentes. En general, el amplificador cumple con su función, pero presenta aspectos que requieren mejora en la calidad de la salida y el desempeño durante uso prolongado sin calentarse tanto.

V. ANÁLISIS DE RESULTADOS

El amplificador de audio construido mostró un desempeño aceptable, logrando amplificar señales de audio de una guitarra eléctrica. Sin embargo, el análisis de resultados nos llevó a identificar las áreas en las que hay que mejorar para que su rendimiento sea mucho más óptimo y confiable.

Por ejemplo, aunque la amplificación fue suficiente para reproducir un buen volumen de sonido, se detectó distorsión armónica. También que durante el uso prolongado, el transistor TIP32C presentó calentamiento, junto con las resistencias más cercanas incluso con un disipador de calor y esto podría reducir significativamente la vida útil del componente y el correcto funcionamiento del amplificador. O la robustez del diseño que aunque nos permitió validar la funcionalidad del circuito, la protoboard no es tan práctica, lo ideal sería un diseño pequeño y compacto en una PCB, la cual tenga conexiones mucho más estables y se evite el ruido o los falsos contactos. Sumado a esto, durante la etapa de construcción mientras se elegían los componentes no fue posible conseguir resistencias de 48Ω exactas, por lo que se utilizaron dos resistencias de 100Ω conectadas en paralelo para obtener un valor aproximado a los 48Ω . De manera similar, para las resistencias de 12Ω se tuvo que utilizar unas de 24Ω en paralelo. Son modificaciones que afectan levemente el comportamiento del circuito y que deben ser mejoradas. En pocas palabras, aunque se cumplieron los objetivos mejorar todos estos aspectos hará que el amplificador sea más eficiente y estable, adecuado para usos más exigentes.

VI. CONCLUSIÓN

El diseño, simulación y construcción del amplificador de audio basado en transistores logró cumplir su objetivo principal de amplificar las señales de audio para su reproducción a un volumen adecuado en dispositivos de salida, como altavoces. La implementación de las etapas de amplificación, utilizando los transistores 2N2222A y TIP32C, permitieron obtener una ganancia correcta y un rendimiento satisfactorio. No obstante, durante el análisis de resultados se determinó cuales áreas son las que deben ser mejoradas para un mejor funcionamiento del amplificador. Durante las pruebas se observó el calentamiento del transistor TIP32C, incluso integrando un disipador de calor, lo cual compromete a la eficiencia térmica del circuito. Asimismo, la calidad de la señal de salida presentó ligeras distorsiones. Estas cuestiones son significativas para modificar el diseño para una mejor resistencia térmica. Otro aspecto importante fue la selección de componentes, específicamente las resistencias. Pues el no haber utilizado las resistencias exactas que fueron calculadas debido a las complicaciones relacionadas a la disponibilidad en tiendas, llevó al uso de combinaciones en paralelo, lo cual tiene leves repercusiones en la precisión del diseño. Finalmente, aunque el uso del protoboard sea práctico para comprobar la funcionalidad del amplificador, para un uso más confiable y duradero se determinó que lo ideal es trasladarlo a una PCB para hacerlo más compacto y con conexiones más precisas para evitar falsos contactos. En conclusión, el desarrollo de este amplificador de audio basado en transistores posibilitó la exploración de tanto los principios fundamentales de la amplificación como los desafíos prácticos que surgen en la implementación de electrónica. Aunque el proyecto logró una funcionalidad básica satisfactoria, evidenció varios aspectos técnicos que requieren mejoras para incrementar el desempeño del sistema.

REFERENCIAS

- [1] A. S. Sedra and K. C. Smith, *Microelectronic Circuits*, 7th ed. New York, NY, USA: Oxford Univ. Press, 2015.
- [2] R. L. Boylestad and L. Nashelsky, *Electrónica: Teoría de Circuitos y Dispositivos Electrónicos*, 11th ed. Ciudad de México, México: Pearson Educación, 2009.
- [3] P. Horowitz and W. Hill, *The Art of Electronics*, 3rd ed. Cambridge, U.K.: Cambridge Univ. Press, 2015.
- [4] A. Malvino and D. J. Bates, *Principios de Electrónica*, 7th ed. New York, NY, USA: McGraw Hill, 2016.
- [5] Mielecánica Fácil, "Transistor." [Online]. Disponible: <https://mielecnicafacil.com/componentes/transistor/>. [Accedido: Nov. 18, 2024].
- [6] S. S. Mohapatra, "¿Cómo aumentar la potencia de salida para este amplificador de audio?", Electronica.guru, 2015. [En línea]. Disponible en: <https://electronica.guru/questions/84301/como-aumentar-la-potencia-de-salida-para-este-amplificador-d>. [Accedido: 18-nov-2024].
- [7] "Diseño de amplificadores a transistores con pocas matemáticas," YouTube. <https://www.youtube.com/watch?v=CkFdLhFLLLA> (accedido: Nov. 20, 2024).
- [8] "Disipador de calor para transistor TO-220," Electrónica Gabriel. [En línea]. Disponible en: <https://electronicagabriel.com/producto/disipador-de-calor-para-transistor-to-220/>. [Accedido: 20-nov-2024].

Universidad Autónoma de Ciudad Juárez

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación



Tarea 14 Control de posición de un carrito

Sensores e Instrumentación

Ángel Alvídrez 225581:

Fecha: 30 de Abril 2025



Índice

1. Objetivo	2
2. Introducción	2
3. Definiciones y marco conceptual	3
3.1. Sensor ultrasónico	3
3.2. Control PID	4
3.3. Comunicación Bluetooth	4
3.4. Motores de Corriente Directa (CD)	4
3.5. Señales PWM	5
3.6. Puente H (L298N)	5
4. Desarrollo	6
5. Código	7
6. Resultados	10
7. Conclusión	11



1. Objetivo

Desarrollar un sistema de control de posicion para un carrito motorizado, utilizando un sensor de ultrasonido y un controlador PID implementado en Arduino, con el fin de mantener una distancia fija respecto a una pared.

2. Introducción

En el ambito de la mecatronica y el control automatico, uno de los retos fundamentales es lograr que un sistema se mantenga en una posicion o trayectoria deseada ante posibles perturbaciones externas. En esta practica se implementa un sistema de control de posicion para un carrito con motores de corriente directa, cuyo objetivo es mantener una distancia constante respecto a una pared utilizando un sensor ultrasónico HC-SR04.

El sistema se controla mediante un algoritmo PID (Proporcional, Integral, Derivativo), el cual ajusta dinamicamente la velocidad y direccion del carrito para minimizar el error entre la distancia actual y la deseada. Esta ultima se establece de forma remota a traves de una interfaz Bluetooth, permitiendo experimentar en tiempo real con distintos valores de referencia.

El uso de control PID en sistemas de posicion es ampliamente utilizado en la industria debido a su simplicidad y efectividad. Esta practica permite al estudiante comprender de manera practica el funcionamiento de un controlador en lazo cerrado, asi como la importancia del ajuste adecuado de sus parametros para obtener una respuesta estable y precisa.



3. Definiciones y marco conceptual

3.1. Sensor ultrasónico

Este sensor permite medir distancias a través del tiempo de vuelo de un pulso ultrasónico. Al enviar un pulso sonoro de alta frecuencia y detectar el eco reflejado en un objeto, es posible calcular la distancia mediante la fórmula:

$$\text{distancia} = \frac{340 \text{ m/s} \cdot t(\mu\text{s})}{2} \cdot \frac{1 \text{ s}}{1,000,000 \mu\text{s}} \cdot \frac{100 \text{ cm}}{1 \text{ m}}$$

La cual simplificada queda como

$$\text{distancia (cm)} = t(\mu\text{s}) \times 0,01715$$



Figura 1: Sensor Ultrasónico HC-SR04



3.2. Control PID

El controlador PID (Proporcional, Integral, Derivativo) es uno de los algoritmos más utilizados en el control de sistemas. Su salida se basa en tres componentes:

- Proporcional (P): responde de manera proporcional al error actual.
- Integral (I): acumula el error en el tiempo para corregir errores persistentes.
- Derivativo (D): anticipa la tendencia del error para reducir oscilaciones.

La combinación de estas tres acciones permite lograr un equilibrio entre rapidez, estabilidad y precisión. En este caso, el PID se implementa con la librería PID v1 de ARDUINO en el ESP 32, configurando los parámetros de manera empírica hasta lograr un comportamiento deseado.

3.3. Comunicación Bluetooth

La comunicación Bluetooth permite enviar y recibir datos de forma inalámbrica. En esta práctica, se utiliza el módulo integrado del ESP32 junto con la librería BluetoothSerial para recibir el valor del setpoint en tiempo real desde un dispositivo externo, permitiendo modificar la referencia de posición sin necesidad de reprogramar el sistema utilizando una aplicación Android creada en MIT App Inventor.

3.4. Motores de Corriente Directa (CD)

Los motores de corriente directa convierten energía eléctrica en movimiento rotatorio. Son ampliamente utilizados por su facilidad de control y bajo costo. En esta práctica se utilizaron motores de 6V, los cuales permiten manejar velocidades aceptables para un entorno de laboratorio o salón.



Figura 2: Motor de 6V DC

3.5. Señales PWM

La modulación por ancho de pulso (PWM, por sus siglas en inglés) es una técnica que permite controlar la velocidad de un motor de CD ajustando el ciclo de trabajo (duty cycle) de una señal digital. Cuanto mayor sea el ciclo de trabajo, mayor será la energía suministrada al motor, y por ende, su velocidad.

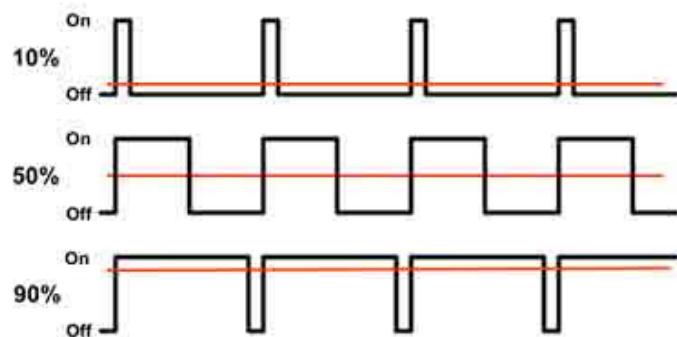


Figura 3: Gráfica de PWM

3.6. Puente H (L298N)

El puente H es un circuito que permite cambiar la polaridad de alimentación de los motores, lo cual permite invertir su sentido de giro. El L298N es un puente doble que



permite controlar dos motores de manera independiente, y también puede controlar su velocidad mediante la señal PWM enviada desde un microcontrolador como el Arduino.

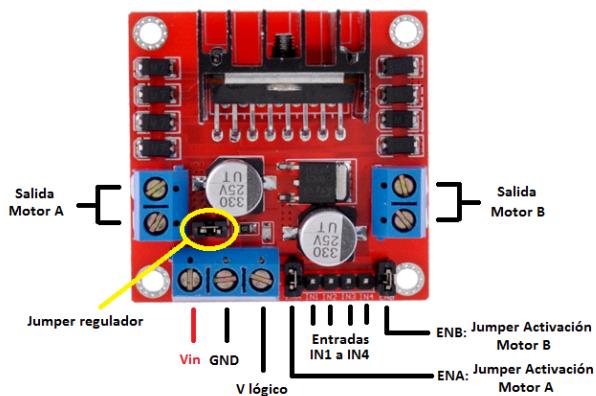


Figura 4: Puente H L298N

4. Desarrollo

El sistema de control de posición se implementó en una placa ESP32, la cual se encargó tanto de la lectura del sensor ultrasónico HC-SR04 como del control de los motores a través del puente H L298N. La programación se realizó en el entorno de Arduino para controlar un ESP32, utilizando la librería PID_v1.h para implementar el controlador PID y la librería BluetoothSerial.h para recibir datos desde un dispositivo móvil o computadora mediante Bluetooth.

El flujo del programa es el siguiente:

- Inicialización de la comunicación serial y Bluetooth.
- Configuración de los pines para el control de los motores y el sensor ultrasónico.
- Espera de un valor de setpoint enviado por Bluetooth, el cual representa la distancia deseada en centímetros entre el carrito y la pared.
- Lectura constante de la distancia utilizando el tiempo de vuelo del sensor ultrasónico.
- Cálculo de la señal de control usando el algoritmo PID.



- Aplicación de la señal de control a los motores, utilizando PWM para ajustar la velocidad según la magnitud del error.
- Si el PWM calculado es muy bajo (pero no cero), se impone un valor mínimo para asegurar que los motores reaccionen.

El código se fue ajustando empíricamente, principalmente los valores de las constantes del PID y el umbral mínimo de PWM necesario para que el carrito se mueva, pues con valores muy bajos el carrito no se movía . Se observó que, con las constantes $K_p = 1,0$, $K_i = 0,0$ y $K_d = 0,2$, el comportamiento del sistema era suficientemente estable y respondía de forma adecuada a las correcciones de posición.

5. Código

Este fue el código utilizado explicado anteriormente:

Listing 1: Código de control PID para el carrito

```
#include <PID_v1.h>
#include "BluetoothSerial.h"
BluetoothSerial SerialBT;

// Pines motores L298N
const int M1_A = 16, M1_B = 17;
const int M2_A = 18, M2_B = 19;

// Pines HC-SR04
const int trigPin = 27, echoPin = 26;
long duration;
double leerDistancia() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2); // Opcional, pero puede ayudar a estabilizar
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    long duration = pulseIn(echoPin, HIGH); // sin timeout
    return duration * 0.01715;
```



```
}

// PID

double Setpoint, Input, Output;
double Kp = 1.0, Ki = 0.0, Kd = 0.2;
PID myPID(&Input, &Output, &Setpoint, Kp, Ki, Kd, DIRECT);

// Umbral m nimo de motor
const int motorMinPWM = 135;

void setup() {
    Serial.begin(115200);
    while (!Serial) delay(10);
    Serial.println("\n--- Control PID de distancia ---");

    SerialBT.begin("ESP32_Carrito"); // Nombre Bluetooth

    // Pines motores
    pinMode(M1_A, OUTPUT); pinMode(M1_B, OUTPUT);
    pinMode(M2_A, OUTPUT); pinMode(M2_B, OUTPUT);
    // Pines ultrasonido
    pinMode(trigPin, OUTPUT); pinMode(echoPin, INPUT);

    // Esperar primer Setpoint por Bluetooth
    Serial.println("Esperando Setpoint por Bluetooth (1 byte en cm)...");
    while (SerialBT.available() == 0) delay(10);
    Setpoint = SerialBT.read();
    Serial.print("Setpoint inicial recibido: ");
    Serial.print(Setpoint);
    Serial.println(" cm");

    // Init PID
    Input = leerDistancia();
    myPID.SetMode(AUTOMATIC);
    myPID.SetOutputLimits(-255, 255);

    Serial.println("Setup completo. Iniciando control...");
```



```
}
```

```
void loop() {
    // Actualizar Setpoint si llega uno nuevo
    if (SerialBT.available() > 0) {
        Setpoint = SerialBT.read();
        Serial.print("Nuevo Setpoint recibido: ");
        Serial.print(Setpoint);
        Serial.println(" cm");
    }

    // 1) Leer distancia
    Input = leerDistancia();
    Serial.print("Dist actual: "); Serial.print(Input); Serial.println(" cm");

    // 2) Calcular PID
    myPID.Compute();

    // 3) Control de motores
    int pwm = abs((int)Output);
    bool adelante = (Output > 0);

    if (pwm > 0 && pwm < motorMinPWM) pwm = motorMinPWM; // Aplicar mínimo PWM

    if (adelante) {
        analogWrite(M1_A, pwm); analogWrite(M2_A, pwm);
        analogWrite(M1_B, 0); analogWrite(M2_B, 0);
    } else {
        analogWrite(M1_B, pwm); analogWrite(M2_B, pwm);
        analogWrite(M1_A, 0); analogWrite(M2_A, 0);
    }

    // 4) Debug en monitor serial
    Serial.print(" SP:"); Serial.print(Setpoint);
    Serial.print(" PV:"); Serial.print(Input);
    Serial.print(" Err:"); Serial.print(Setpoint - Input);
    Serial.print(" Out:"); Serial.print(Output);
```



```
Serial.print("  PWM:"); Serial.println(pwm);

delay(100);

}
```

6. Resultados

Durante las pruebas del sistema se utilizó un rango de distancias deseadas entre 10 cm y 50 cm, enviado dinámicamente mediante Bluetooth. En este rango, el carrito mantuvo una posición estable frente a la pared, corrigiendo su distancia de forma continua mediante el controlador PID.

Se observó que, al aumentar la distancia deseada más allá de los 50 o 60 cm, el sensor ultrasónico comenzaba a presentar errores de medición. Esto probablemente se debe a que el sensor opera con un ángulo de detección de aproximadamente 30° , por lo que a distancias grandes puede comenzar a detectar el piso en lugar de la pared, afectando la precisión de la lectura.

Dentro del rango óptimo (10–50 cm), el comportamiento del sistema fue suave, sin oscilaciones muy marcadas ni movimientos muy bruscos. También se comprobó que el uso de un umbral mínimo de PWM (`motorMinPWM`) fue necesario para evitar que los motores quedaran estáticos con señales muy pequeñas, especialmente cuando el error era bajo pero aún necesitaba corrección.

La señal de salida del PID se ajustaba bien a los cambios de setpoint, lo que permitía al carrito adaptarse a nuevas posiciones con una transición rápida y estable. Todo el comportamiento del sistema se podía monitorear en tiempo real desde el monitor serial.



7. Conclusión

El sistema implementado logró controlar efectivamente la posición de un carrito respecto a una pared utilizando un sensor ultrasónico y un controlador PID. La integración del módulo Bluetooth permitió modificar dinámicamente el setpoint, lo cual facilitó la experimentación con distintas distancias objetivo sin necesidad de modificar el código o reiniciar el sistema.

Se comprobó que el sensor ultrasónico es confiable en un rango aproximado de 10 a 50 cm, pero que comienza a fallar a distancias mayores debido a su forma de detección cónica, que puede provocar lecturas erróneas al captar ecos del piso u otros objetos. Esto resalta la importancia de conocer las limitaciones físicas de los sensores al momento de diseñar un sistema de control.

El uso del controlador PID permitió suavizar el comportamiento del carrito, ajustando su posición de forma continua sin necesidad de detenerse por completo. Ajustes empíricos a las constantes K_p , K_i y K_d fueron clave para lograr un desempeño estable y responsivo. Finalmente, esta práctica fue útil para reforzar conceptos de control automático, electrónica y programación aplicada, mostrando cómo integrar múltiples componentes en un sistema funcional.

Universidad Autónoma de Ciudad Juárez

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación



Tarea 16 Carrito seguidor de lineas

Sensores e Instrumentación

Ángel Alvídrez 225581:

Fecha: 30 de Abril 2025



Índice

1. Objetivo	2
2. Introducción	2
3. Definiciones y marco conceptual	3
3.1. Fotodiodo	3
3.2. Fototransistor	3
3.3. Motores de Corriente Directa (CD)	4
3.4. Señales PWM	4
3.5. Puente H (L298N)	5
4. Desarrollo	6
5. Código	7
6. Resultados	9
7. Conclusión	10



1. Objetivo

Desarrollar un prototipo funcional de un carrito seguidor de línea empleando sensores infrarrojos y motores de corriente directa controlados mediante PWM, utilizando un microcontrolador ESP32. El sistema deberá ser capaz de detectar una trayectoria en una pista y seguirla de manera autónoma, demostrando el principio de retroalimentación entre sensores y actuadores.

2. Introducción

Un seguidor de línea es un tipo de robot móvil que utiliza sensores ópticos para detectar y seguir una trayectoria definida, usualmente una línea de color contrastante (negra sobre fondo blanco). Estos sistemas son una de las aplicaciones más comunes en la robótica educativa debido a su capacidad de demostrar conceptos fundamentales de electrónica, control y programación de una manera práctica.

En esta tarea se desarrolló un prototipo de carrito seguidor de línea utilizando un microcontrolador ESP32, sensores infrarrojos compuestos por un par fotodiodo-fototransistor, y motores de corriente directa controlados mediante modulación por ancho de pulso (PWM). El sistema fue diseñado para seguir una pista, reaccionando a los cambios de reflectividad del suelo para ajustar el movimiento del vehículo en tiempo real.



3. Definiciones y marco conceptual

3.1. Fotodiodo

El FD-IR333C es un diodo emisor de luz infrarroja que opera típicamente a una tensión directa de aproximadamente 1.2V. Su función es emitir radiación infrarroja que será reflejada por la superficie bajo el sensor. Superficies claras reflejan más luz, mientras que las oscuras absorben gran parte de la radiación. En este proyecto, el fotodiodo se conecta en serie con una resistencia limitadora de corriente de 100Ω , formando parte del par sensor que detecta la línea sobre la pista.



Figura 1: Foto diodo transparente (FD-IR333C)

3.2. Fototransistor

El PT331C es un fototransistor que responde a la radiación infrarroja emitida por el fotodiodo. Cuando la luz infrarroja reflejada incide sobre su base, el fototransistor permite el paso de corriente del colector al emisor. En esta configuración, el colector se conecta a una resistencia pull-up de $10k\Omega$ hacia 5V y también a la entrada analógica del ESP32, mientras que el emisor va conectado a tierra. Este arreglo permite detectar cambios en la cantidad de luz reflejada como variaciones en el voltaje de salida, que el microcontrolador interpreta para determinar la posición del carrito respecto a la línea.



Figura 2: Fototransistor (PT331C)

3.3. Motores de Corriente Directa (CD)

Los motores de corriente directa convierten energía eléctrica en movimiento rotatorio. Son ampliamente utilizados por su facilidad de control y bajo costo. En esta práctica se utilizaron motores de 6V, los cuales permiten manejar velocidades aceptables para un entorno de laboratorio o salón.



Figura 3: Motor de 6V DC

3.4. Señales PWM

La modulación por ancho de pulso (PWM, por sus siglas en inglés) es una técnica que permite controlar la velocidad de un motor de CD ajustando el ciclo de trabajo (duty



cycle) de una señal digital. Cuanto mayor sea el ciclo de trabajo, mayor será la energía suministrada al motor, y por ende, su velocidad.

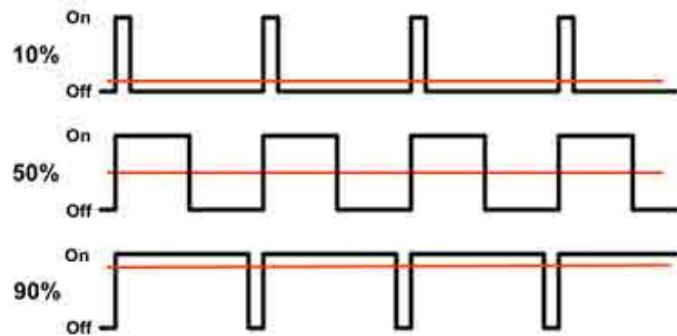


Figura 4: Gráfica de PWM

3.5. Puente H (L298N)

El puente H es un circuito que permite cambiar la polaridad de alimentación de los motores, lo cual permite invertir su sentido de giro. El L298N es un puente doble que permite controlar dos motores de manera independiente, y también puede controlar su velocidad mediante la señal PWM enviada desde un microcontrolador como el Arduino.

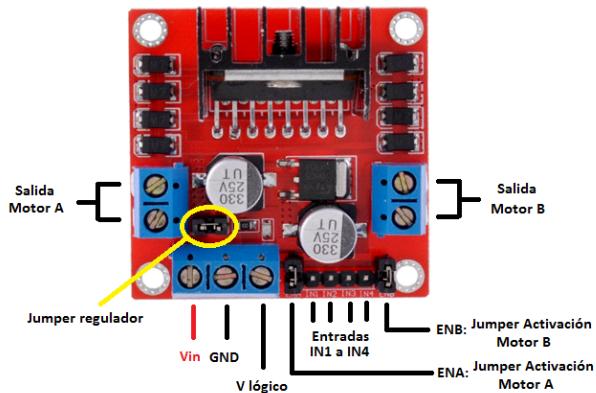


Figura 5: Puente H L298N



4. Desarrollo

Para la detección de línea, se utilizaron dos pares sensor: cada uno compuesto por un fotodiodo FD-IR333C y un fototransistor PT331C.

Para los fotodiodos, se utiliza una resistencia limitadora de 100Ω con una alimentación de 5V y una caída de tensión aproximada de 1.2V y estando ambos diodos en paralelo que divide la corriente para cada led a la mitad. Aplicando la Ley de Ohm:

$$V_{led} = 5V - 1,2V = 3,8$$

$$I_{led} = \frac{V_{led}}{R} \times \frac{1}{2} = \frac{3,8V}{100} \times \frac{1}{2} \approx 20mA$$

Esta configuración asegura un funcionamiento estable y un consumo de corriente adecuado para proteger tanto los LEDs como el ESP32.

Los fototransistores se conectaron con el colector hacia una resistencia pull-up de $100k\Omega$ que va a 5V, y también a una entrada analógica del ESP32 (GPIO34 y GPIO35).

El emisor de cada fototransistor se conecta directamente a tierra. Así, cuando el sensor detecta una superficie clara (que refleja más IR), el fototransistor conduce y el voltaje en el pin de entrada disminuye. Cuando detecta línea negra (menos reflexión), la conducción disminuye y el voltaje de entrada sube, permitiendo distinguir claramente la posición de la línea.

El sistema cuenta con dos motores de corriente directa, uno por cada rueda. Estos motores se conectaron directamente a los pines GPIO16, GPIO17, GPIO18 y GPIO19 del ESP32 a través de un driver que permite la conmutación H-Bridge para cada motor.

Cada motor recibe dos señales: una de control PWM para la velocidad, y otra lógica para el sentido de giro. El motor izquierdo está controlado por los pines 18 y 19, y el derecho por los pines 16 y 17.

El algoritmo principal consiste en leer constantemente los valores analógicos de los dos sensores y comparar sus valores contra un umbral (aproximadamente 1900). Si el sensor izquierdo detecta línea y el derecho no, el carrito gira a la izquierda; si ocurre lo contrario, gira a la derecha. Si ambos sensores detectan línea o fondo, el carrito se detiene. La velocidad de los motores se controla mediante PWM, con un valor de 255 (máxima velocidad).



5. Código

Este fue el código utilizado explicado anteriormente:

Listing 1: Código carrito seguidor de linea

```
#include <Arduino.h>

// Sensores anal gicos
const int sensorIzq      = 35;      // GPIO34 (ADC1_CH6)
const int sensorDer       = 34;      // GPIO35 (ADC1_CH7)

// Motor A (rueda izquierda)
const int motorA_IN1     = 18;      // GPIO18
const int motorA_IN2     = 19;      // GPIO19

// Motor B (rueda derecha)
const int motorB_IN1     = 16;      // GPIO16
const int motorB_IN2     = 17;      // GPIO17

// ===== PARAMETROS =====
const int UMBRAL         = 1900;
const int VELOCIDAD        = 255;

void setup() {
    Serial.begin(115200);
    pinMode(sensorIzq, INPUT);
    pinMode(sensorDer, INPUT);
    pinMode(motorA_IN1, OUTPUT);
    pinMode(motorA_IN2, OUTPUT);
    pinMode(motorB_IN1, OUTPUT);
    pinMode(motorB_IN2, OUTPUT);
}

void loop() {
    int valIzq = analogRead(sensorIzq);
    int valDer = analogRead(sensorDer)-300;
```



```
Serial.printf("Sensor Izq=%4d  Sensor Der=%4d\n", valIzq, valDer);

if (valIzq < UMBRAL && valDer > UMBRAL) {
    giraIzq();
}

else if (valIzq > UMBRAL && valDer < UMBRAL) {
    giraDer();
}

else {
    parar();
}

delay(1);
}

void giraIzq() {
    analogWrite(motorA_IN1, 0);           digitalWrite(motorA_IN2, LOW);
    analogWrite(motorB_IN1, VELOCIDAD);  digitalWrite(motorB_IN2, LOW);
}

void giraDer() {
    analogWrite(motorA_IN1, VELOCIDAD);  digitalWrite(motorA_IN2, LOW);
    analogWrite(motorB_IN1, 0);           digitalWrite(motorB_IN2, LOW);
}

void parar() {
    digitalWrite(motorA_IN1, LOW);  digitalWrite(motorA_IN2, LOW);
    digitalWrite(motorB_IN1, LOW);  digitalWrite(motorB_IN2, LOW);
}
```



6. Resultados

Al finalizar la implementación del carrito seguidor de línea, se comprobó su funcionamiento sobre una pista con un trazo negro sobre fondo blanco. El carrito fue capaz de identificar correctamente la línea y reaccionar de acuerdo con la lógica programada. Cuando el sensor izquierdo detectaba la línea, el carrito giraba hacia la izquierda; cuando era el derecho, giraba hacia la derecha; y cuando ambos sensores estaban fuera de la línea, el carrito se detenía. La respuesta fue rápida y estable durante las pruebas.

Los sensores infrarrojos respondieron con lecturas claras y distinguibles: valores altos cuando no había línea (fondo blanco) y valores bajos al detectar la línea negra. Esto permitió establecer un umbral confiable de detección en 1900, con una pequeña calibración adicional al sensor derecho (-300) para compensar diferencias en la sensibilidad. Además, el control PWM proporcionó una velocidad constante y adecuada para la prueba sin causar movimientos bruscos ni desviaciones fuera de la pista.

Durante las pruebas, el carrito completó múltiples recorridos sin perder el trazo ni necesitar intervención externa. El sistema demostró ser eficiente para tareas básicas de seguimiento de línea, cumpliendo con los objetivos establecidos para la práctica.

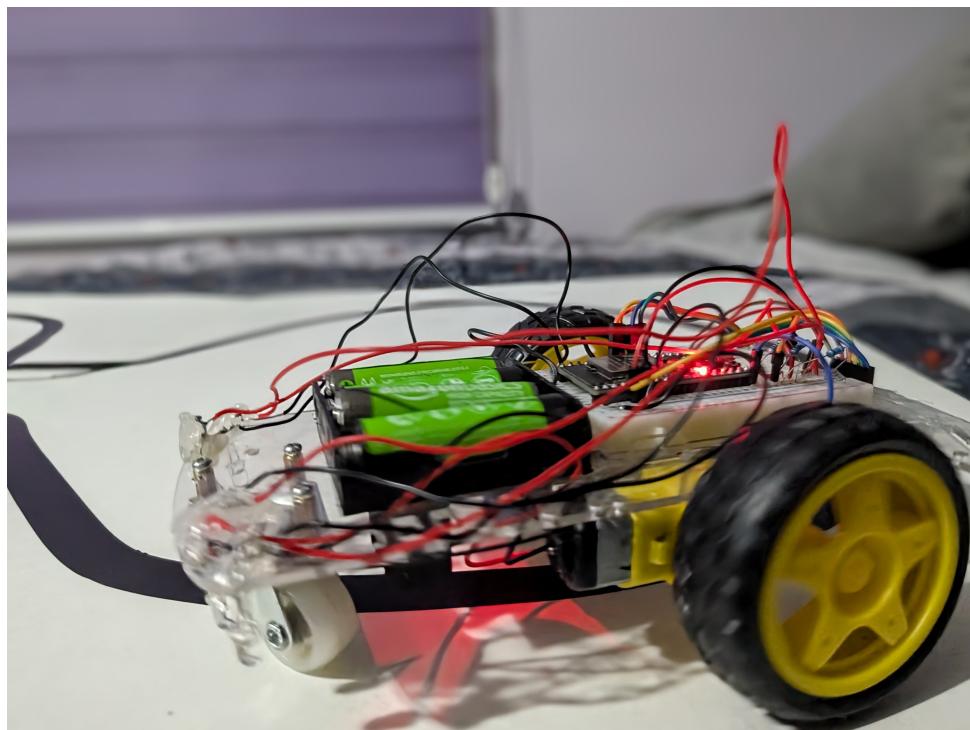


Figura 6: Carrito en funcionamiento



7. Conclusión

El proyecto del carrito seguidor de línea cumplió con los objetivos establecidos de manera satisfactoria. La combinación de sensores infrarrojos (FD-IR333C y PT331C), el microcontrolador ESP32 y los motores de corriente directa permitió crear un sistema de navegación autónoma capaz de seguir una línea con precisión.

El uso de la modulación por ancho de pulso (PWM) facilitó un control eficiente de la velocidad de los motores, lo que mejoró la estabilidad del carrito. La lectura analógica de los sensores infrarrojos permitió detectar de manera fiable las variaciones entre fondo blanco y la línea negra, lo que hizo posible que el carrito reaccionara correctamente ante las distintas situaciones que se presentaron en la pista.

El sistema mostró un comportamiento consistente en las pruebas, realizando giros precisos y deteniéndose cuando era necesario. Aunque el sistema es funcional, se pueden realizar mejoras, como la implementación de un control de velocidad adaptativo o la incorporación de sensores adicionales para mejorar la precisión en curvas más cerradas.

En general, el proyecto fue exitoso en su implementación y representa una base sólida para futuros proyectos de robótica y automatización.

Universidad Autónoma de Ciudad Juárez

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación



Tarea 15 Encendido y apagado de un foco con aplausos

Sensores e Instrumentación

Ángel Alvídrez 225581:

Fecha: 6 de Mayo 2025



Índice

1. Introducción	2
2. Definiciones y marco conceptual	3
2.1. Micrófono KY-038	3
2.2. Amplificador operacional TL081	4
2.3. Filtro pasabanda activo	4
2.4. Comparador con nivel de referencia ajustable	5
2.5. Optoacoplador MOC3022	6
2.6. TRIAC 2N6071	6
3. Desarrollo	7
3.1. Amplificación con op amp (TL081)	7
3.2. Filtro pasabanda activo (3kHz – 6kHz)	7
3.3. Comparador de umbral	9
3.4. Procesamiento Digital	10
3.5. Etapa de potencia	10
4. Código	11
5. Resultados	13
6. Conclusión	15



1. Introducción

En la presente práctica se diseña e implementa un sistema capaz de encender y apagar un foco de 110 V a través de la detección de aplausos. Para ello, se emplea una cadena de procesamiento de señales que va desde la captación del sonido hasta la conmutación de un triac mediante un Arduino Uno, se espera que el circuito:

- Detecte el sonido de los aplausos y convertirlo en una señal eléctrica procesable.
- Filtre y amplifique únicamente la banda de frecuencias donde predominan los aplausos (3 kHz - 6 kHz).
- Genere un pulso digital ajustable para el Arduino, en función de la intensidad del aplauso.
- Cuente el número de aplausos en un intervalo de 3 segundos para decidir el encendido (2 aplausos) o apagado (4 aplausos) del foco.
- Commute un triac (2N6071) con un optoacoplador MOC3022 para aislar la señal de control de la línea de 110 VAC.

La ganancia se ajusta con un potenciómetro en la red de realimentación (R_f), permitiendo controlar la amplitud de la señal proveniente del micrófono KY-038. Se alimentará el circuito con +5 V provenientes del Arduino y -5 V generados con un regulador L7905 conectado a una pila de 9V.

El filtro fue implementado con un segundo TL081 diseñado para atenuar todas las frecuencias fuera del rango de interés, asegurando que sólo los aplausos generen una señal significativa. Un tercer TL081 funciona como comparador de ventana simple. El umbral de disparo se ajusta con otro potenciómetro, de modo que la salida se satura y entrega un nivel digital (HIGH) cuando la señal filtrada supera el nivel de referencia.

La salida del comparador se conecta al pin 2 del Arduino, que detecta flancos de subida para incrementar un contador de aplausos.

Se mide un intervalo de 3 segundos (iniciado con el primer aplauso) dentro del cual: 2 aplausos el Arduino enciende el LED de simulación (y envía la señal al MOC3022 para encender el triac).



4 aplausos el Arduino apaga el LED (y deja de disparar el triac). Valores excedentes de aplausos o fin de los 3 segundos reinician el conteo. El Arduino controlará un MOC3022, que optoaisla la señal de control y dispara el triac 2N6071, encargado de comutar el foco de 110 VAC.

2. Definiciones y marco conceptual

2.1. Micrófono KY-038

Es un módulo sensor de sonido con salida analógica y digital. En este proyecto solo se utiliza la salida analógica, que proporciona una señal proporcional a la intensidad del sonido captado. El sensor consiste en un micrófono de condensador electret y un pequeño amplificador integrado. La señal es débil y requiere amplificación adicional para ser útil en procesamiento de señales.

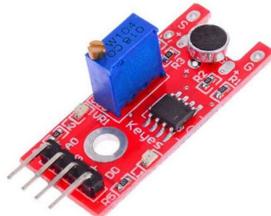


Figura 1: Sensor Ultrasónico HC-SR04



2.2. Amplificador operacional TL081

Es un amplificador operacional de bajo ruido con entrada diferencial y salida simple. Se utilizó este componente para todas las etapas analógicas del circuito:

- Amplificación inicial de la señal del micrófono
- Filtro pasabanda
- Comparador de nivel

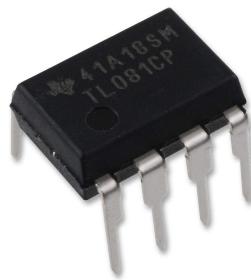


Figura 2: Op Amp TL081

2.3. Filtro pasabanda activo

Los aplausos tienen un componente fuerte de energía entre los 3kHz y 6kHz. Para asegurarnos de que solo estas frecuencias pasen y el sistema no reaccione ante ruidos ambientales u otras voces, se diseñó un filtro pasabanda activo con op amps.

Este filtro atenúa las señales fuera de esa banda, disipando ruido bajo (habla, ambiente) y alto (chispeos, interferencia).

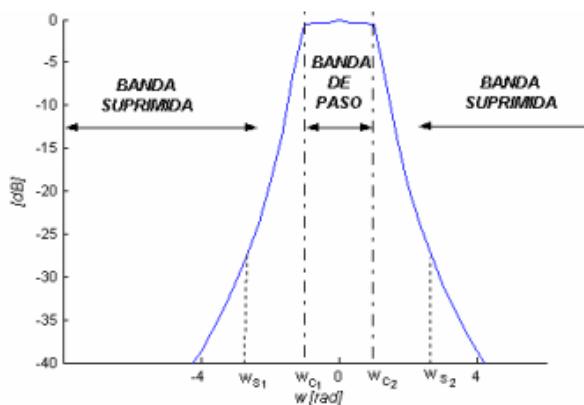


Figura 3: Salida de un filtro pasa banda

2.4. Comparador con nivel de referencia ajustable

La señal filtrada pasa a un comparador construido con otro TL081, la cuál se compara contra un voltaje de referencia ajustable con potenciómetro, cuando la señal supera ese umbral (por un aplauso), el op amp satura y su salida cambia bruscamente a nivel alto (5 V), actuando como un detector de aplauso.

La salida es una señal digital que el Arduino puede leer fácilmente.

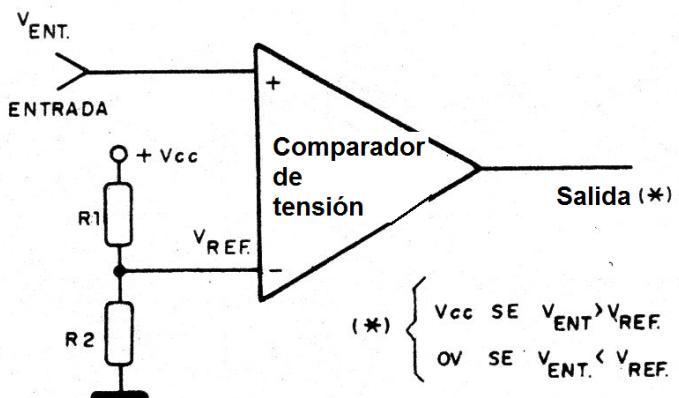


Figura 4: Ejemplo de un comparador de voltaje



2.5. Optoacoplador MOC3022

El MOC3022 es un optoacoplador con salida de TRIAC diseñado para controlar cargas de corriente alterna sin necesidad de conexión directa entre la parte de control y la parte de potencia. Consiste en un diodo LED interno que, al activarse, dispara un TRIAC integrado.

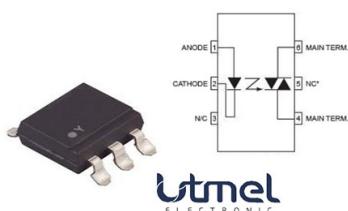


Figura 5: Optoacoplador MOC3022

2.6. TRIAC 2N6071

El 2N6071 es un TRIAC, un dispositivo semiconductor que permite el control de corriente alterna en ambas polaridades de la onda de CA. Funciona como un interruptor controlado por una pequeña señal de disparo en su compuerta

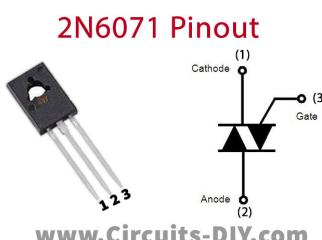


Figura 6: Triac 2N6071



3. Desarrollo

Para el diseño del circuito se dividió todo en bloques funcionales, cada uno con una tarea clara. El primero es la etapa de captación de sonido, que se hizo con un micrófono KY-038. Este módulo tiene dos salidas, una digital y una analógica; para este proyecto se usó la analógica porque entrega una señal proporcional a la intensidad del sonido. Esta señal es de muy baja amplitud, apenas en milivoltios, por lo que se necesitó una etapa de amplificación para poder procesarla bien.

3.1. Amplificación con op amp (TL081)

La amplificación se logró usando un TL081 en configuración no inversora, alimentado con una fuente simétrica de $\pm 5V$. Los $+5V$ los tomamos directamente del Arduino, y para los $-5V$ usamos un regulador L7905 con una fuente externa de $-9V$. En esta etapa también se colocó un potenciómetro de precisión como resistencia variable de realimentación, para ajustar la ganancia del amplificador y así poder controlar qué tan fuerte debía sonar un aplauso para que se detectara.

3.2. Filtro pasabanda activo (3kHz – 6kHz)

Antes de escoger las frecuencias para el filtro, se decidió utilizar el programa Audacity para analizar las frecuencias que hay en los aplausos, por lo que se grabó un aplauso y se aplicó la herramienta de análisis de frecuencia.



Figura 7: Análisis de una frecuencia en Audacity



Práctica num 15

Tarea 15 Encendido y apagado de un foco con aplausos)

Viendo que una gran parte de las frecuencias de los aplausos se aplicó en el mismo programa, un filtro con una curva simulando la que realizará el filtro del circuito.

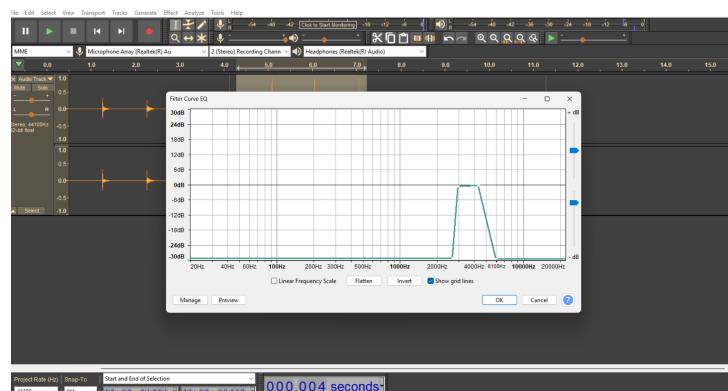


Figura 8: Simulación del filtro en Audacity



Figura 9: Salida del filtro en Audacity

Una vez aplicado el filtro, se puede observar que se filtran los demás sonidos de fondo del audio y quedan unos picos en el espectrograma del audio, mostrado en la figura 9 demostrando que los aplausos prevalecen después del filtro.

Una vez seleccionadas unas frecuencias para el filtro, se empezó con el diseño de este. El capacitor fue seleccionado de forma práctica, eligiendo un valor de 1nF ya que es un valor fácil de conseguir. A partir de ese valor, se calcularon las resistencias necesarias para centrar el filtro en la frecuencia media de:

$$f_r = \sqrt{f_H \cdot f_L} = \sqrt{6 \text{ kHz} \cdot 3 \text{ kHz}} = 4,242 \text{ kHz}$$



La banda pasante se definió como:

$$B = f_H - f_L = 6 \text{ kHz} - 3 \text{ kHz} = 3 \text{ kHz}$$

Y la calidad del filtro como:

$$Q = \frac{f_r}{B} = \frac{4,242}{3} = 1,414$$

Con estos datos, y usando la configuración de un filtro pasa banda activo con amplificador operacional, se usaron las fórmulas correspondientes para calcular los valores de las resistencias, con la constante del filtro dada por:

$$\frac{1}{RC} = 2\pi B$$

$$R = \frac{1}{2\pi BC} \approx \frac{0,1591}{3 \text{ kHz} \times 1 \text{ nF}} \approx 53,033 \text{ k}\Omega$$

Para la siguiente resistencia se utilizó la expresión

$$f_r = \frac{\sqrt{1 + \frac{R}{R_r}}}{2\sqrt{2} \times \pi \times R \times C}$$

$$R_r = R \left(\left(\frac{f_r \cdot RC}{0,1125} \right)^2 - 1 \right)^{-1}$$

$$R_r \approx 17,684 \text{ k}\Omega$$

3.3. Comparador de umbral

Después del filtro, la señal se envió a una tercera etapa: un comparador de voltaje hecho también con un TL081. Aquí se comparaba la señal del filtro contra un voltaje de referencia ajustable con potenciómetro. Si la señal superaba ese umbral, el comparador saturaba a su voltaje alto (cercano a 5V), generando una salida digital que representaba la detección de un aplauso. Esta salida se conectó directamente a una entrada digital del Arduino



Uno, que se encargó de procesar los eventos Estas 3 etapas anteriores fueron simuladas en Proteus 10 para observar su funcionamiento mostrado en la figura 11.

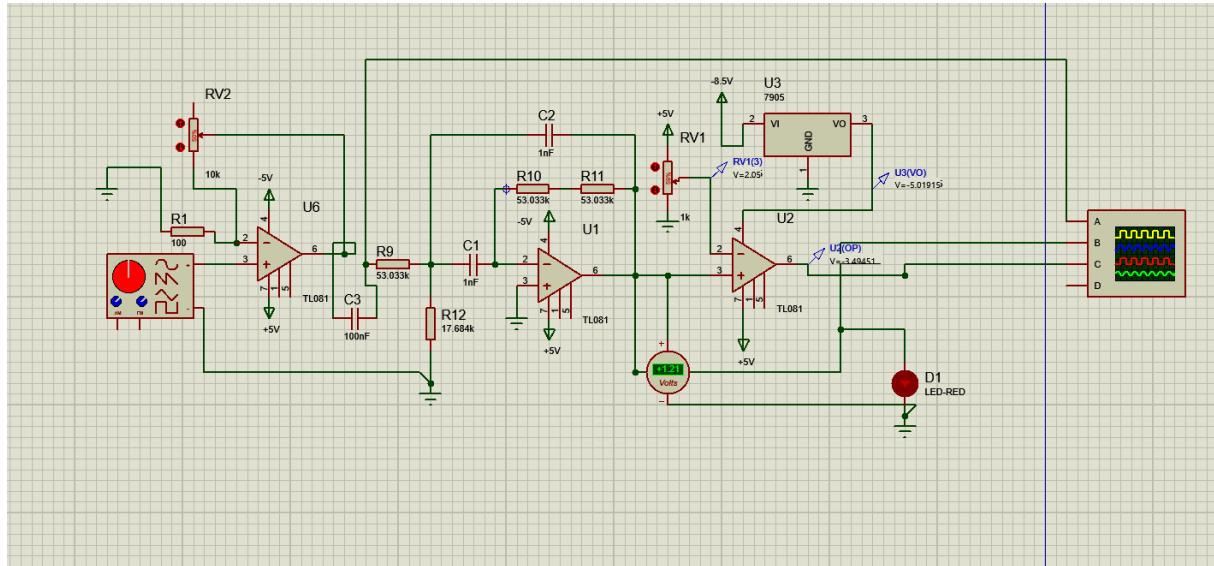


Figura 10: Análisis de una frecuencia en Audacity

3.4. Procesamiento Digital

El Arduino fue programado para detectar los flancos de subida, es decir, cuando la señal pasaba de bajo a alto. Al detectar un aplauso, iniciaba una ventana de 3 segundos para contar los siguientes.

Si dentro de ese tiempo se contaban 2 aplausos, se encendía el foco. Si eran 4, se apagaba. En caso de que pasaran más de 3 segundos sin alcanzar una cantidad válida o se detectaran más de 4 aplausos, el contador se reiniciaba automáticamente.

3.5. Etapa de potencia

La etapa final fue el disparo del foco de 110V con un TRIAC 2N6071. Para aislar el Arduino del voltaje de la red, se utilizó un MOC3022 como optoacoplador. El Arduino mandaba la señal al LED interno del MOC3022, y cuando se activaba, el TRIAC se disparaba, permitiendo el paso de corriente alterna al foco.

Todo esto se integró en una sola tarjeta perforada con los tres TL081, el MOC3022, el TRIAC y los componentes pasivos, dejando el Arduino y la fuente conectados por cables.



4. Código

Este fue el código utilizado explicado anteriormente:

Listing 1: Código de control PID para el carrito

```
const int pinEntrada = 2;      // Entrada del comparador
const int pinLED = 4;           // LED (simula el foco)

int contadorAplausos = 0;
unsigned long tiempoInicio = 0;
bool contando = false;
bool focoEncendido = true;

void setup() {
    pinMode(pinEntrada, INPUT);
    pinMode(pinLED, OUTPUT);
    digitalWrite(pinLED, HIGH);    // El foco inicia encendido
    Serial.begin(115200);
    Serial.println("Contador de aplausos iniciado...");
}

void loop() {
    static bool estadoAnterior = LOW;
    bool estadoActual = digitalRead(pinEntrada);

    // Detectar flanco de subida
    if (estadoActual == HIGH && estadoAnterior == LOW) {
        if (!contando) {
            tiempoInicio = millis();
            contando = true;
        }

        contadorAplausos++;
        Serial.print(" Aplauso detectado! Total: ");
        Serial.println(contadorAplausos);

        // Acciones inmediatas
    }
}
```



```
if (contadorAplausos == 2 && !focoEncendido) {  
    digitalWrite(pinLED, HIGH);  
    focoEncendido = true;  
    Serial.println(">> FOCO ENCENDIDO <<");  
}  
  
else if (contadorAplausos == 4 && focoEncendido) {  
    digitalWrite(pinLED, LOW);  
    focoEncendido = false;  
    Serial.println(">> FOCO APAGADO <<");  
}  
  
else if (contadorAplausos > 4) {  
    Serial.println(">> M s de 4 aplausos. Reiniciando contador. <<");  
    contadorAplausos = 0;  
    contando = false;  
}  
  
delay(200); // Anti-rebote  
}  
  
estadoAnterior = estadoActual;  
  
// Si se pasa el tiempo de espera  
if (contando && (millis() - tiempoInicio >= 3000)) {  
    Serial.print(">> Tiempo agotado. Aplausos contados: ");  
    Serial.print(contadorAplausos);  
    Serial.println(". Reiniciando contador. <<");  
    contadorAplausos = 0;  
    contando = false;  
}  
}
```



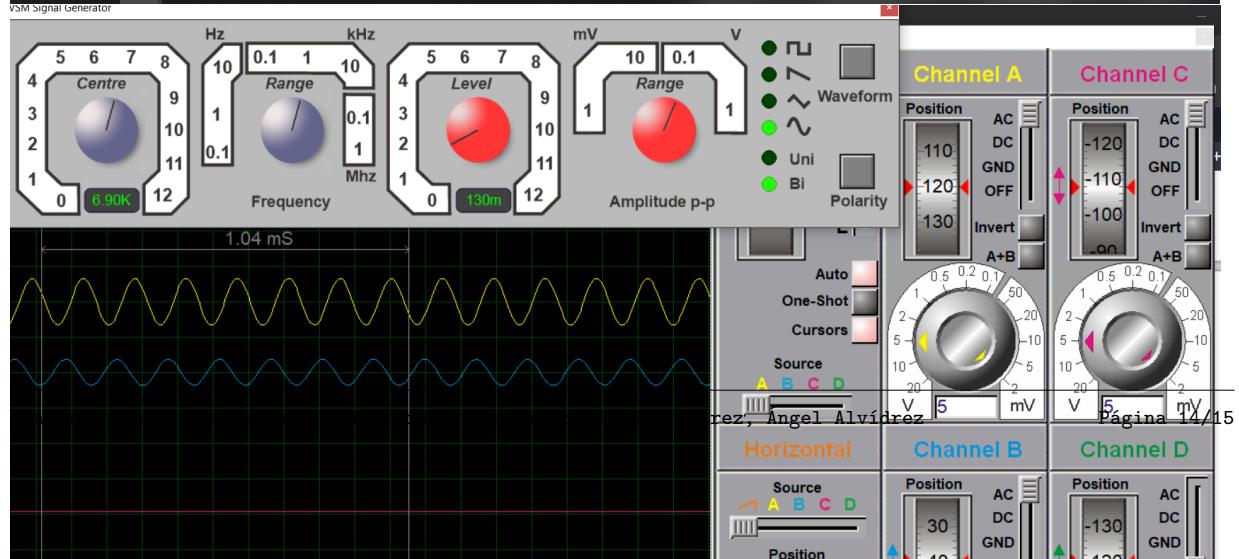
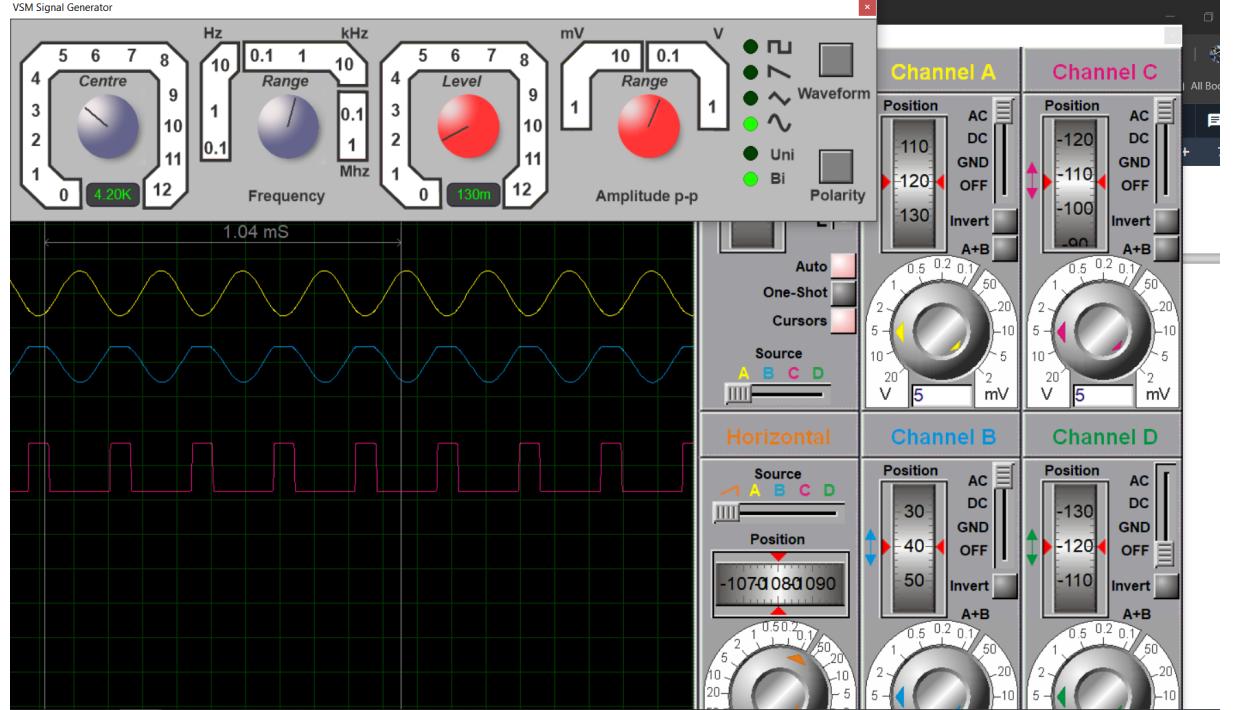
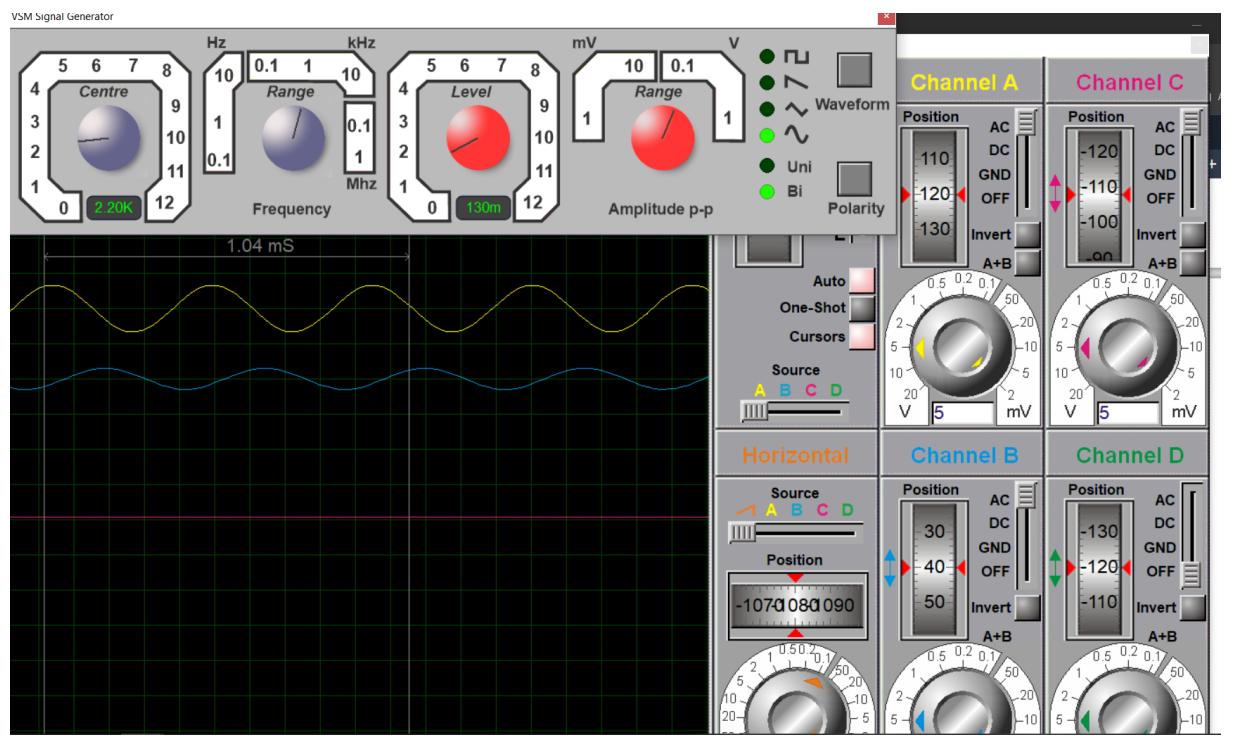
5. Resultados

El circuito fue simulado inicialmente en Proteus, sustituyendo el micrófono por una generadora de señales. En esta simulación el comportamiento fue correcto: el filtro permitía el paso de la señal en el rango deseado y atenuaba frecuencias fuera de banda tal como se esperaba. Sin embargo, en la implementación física surgieron varios detalles que obligaron a modificar el diseño. El módulo KY-037 utilizado como micrófono introducía un offset de aproximadamente 2.5V en la señal, lo cual afectaba directamente el funcionamiento del filtro. Para mitigar este efecto, se incorporó un capacitor cerámico en la entrada del circuito con el fin de bloquear ese componente de DC, ajuste que también se aplicó posteriormente en la simulación de Proteus para reflejar el comportamiento real.



Práctica num 15

Tarea 15 Encendido y apagado de un foco con aplausos)





jjjjjjjjujjjjjjjjjjjjjjjjjj Además, se detectó que el comparador, en su estado de salida baja, introducía un pequeño voltaje negativo que resultaba no deseado al momento de excitar el gate del TRIAC. Este problema se resolvió colocando un diodo que bloquea ese valor negativo y asegura que la señal de control permanezca limpia y sin interferencias en estado de reposo. Con estos ajustes, tanto el funcionamiento en simulación como en la práctica fue estable y el circuito cumplió con su objetivo de detectar señales acústicas dentro del rango específico de frecuencias.

6. Conclusión

Este proyecto permitió integrar múltiples conceptos de electrónica analógica y digital para desarrollar un sistema funcional de detección de frecuencias específicas usando un filtro pasabanda activo con amplificadores operacionales TL081 y un circuito de disparo con TRIAC controlado por Arduino. A lo largo del desarrollo, se comprobó la importancia de considerar no solo el diseño teórico, sino también los detalles prácticos que surgen al implementar componentes reales, como el offset inesperado del módulo KY-037 o el comportamiento del comparador en estado bajo.

La simulación en Proteus fue una herramienta clave para validar el diseño antes del armado físico, pero también se evidenció que ciertos elementos no se comportan de forma idéntica entre la simulación y la práctica. Esto obligó a realizar ajustes directos sobre el circuito, como la inclusión de un capacitor para eliminar el offset de DC o un diodo para asegurar niveles de señal compatibles con el disparo del TRIAC. Estas soluciones simples pero efectivas muestran cómo una buena comprensión del comportamiento de cada bloque funcional permite resolver problemas sin rehacer todo el diseño.

Finalmente, se logró un sistema que detecta señales en el rango deseado y responde adecuadamente encendiendo una carga de corriente alterna, demostrando el uso práctico de filtros activos, comparadores y dispositivos de potencia en una aplicación de control automatizado.

Universidad Autónoma de Ciudad Juárez

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación



Tarea 18 Sistema de riego por IoT

Sensores e Instrumentación

Ángel Alvídrez 225581:

Fecha: 21 de Mayo 2025



Índice

1. Introducción	2
2. Marco Teórico	3
2.1. Microcontrolador ESP32	3
2.2. Sensor de Humedad del Suelo	3
2.3. Sensor DHT11	4
2.4. Sensor PIR (Sensor de Movimiento)	4
2.5. Módulo Relé de 5V	5
2.6. Bomba de Agua y Control de Flujo	5
2.7. Plataforma Blynk	6
2.8. Pantalla LCD	6
2.9. Comunicación I ² C y módulo PCF8574	7
3. Desarrollo	8
3.1. Microcontrolador y comunicación	8
3.2. Sensores y actuadores	8
3.3. Lógica del programa	8
3.4. Visualización de datos	9
3.5. Funcionamiento del sistema	9
4. Código	10
5. Resultados	16
6. Conclusión	16



1. Introducción

La escasez de agua y el uso ineficiente de los recursos hídricos son problemas persistentes en la agricultura y jardinería urbana. En este contexto, los sistemas de riego automatizado representan una solución eficiente para reducir el desperdicio de agua y mejorar el cuidado de las plantas.

El presente proyecto tiene como objetivo desarrollar un sistema de riego inteligente utilizando una placa de desarrollo ESP32, sensores de humedad de suelo y la plataforma Blynk para monitoreo remoto. Este sistema permite detectar en tiempo real el nivel de humedad en la tierra y activar el riego automáticamente cuando sea necesario. Además, el usuario puede visualizar el estado de la planta desde su dispositivo móvil, gracias a la conectividad Wi-Fi y la interfaz amigable que ofrece Blynk [1].

La idea fue inspirada por proyectos previos de código abierto, como el desarrollado por *Viral Science* [2] , el cual mostró un enfoque sencillo pero funcional para el monitoreo de plantas mediante la integración de sensores y controladores conectados a Blynk A partir de esta base, se desarrolló una solución personalizada, adaptada a los recursos disponibles y necesidades específicas del entorno donde se implementará.



2. Marco Teórico

El desarrollo de sistemas inteligentes para el monitoreo y control de plantas ha tomado gran relevancia en el campo de la automatización y el Internet de las Cosas (IoT). Estos sistemas permiten optimizar el uso de recursos como el agua, mejorando la eficiencia y facilitando el cuidado de las plantas mediante el uso de sensores y actuadores interconectados.

2.1. Microcontrolador ESP32

El ESP32 es un microcontrolador de doble núcleo desarrollado por Espressif Systems, que integra conectividad Wi-Fi y Bluetooth, y ofrece un alto rendimiento con bajo consumo energético [3]. Gracias a su versatilidad y capacidad de procesamiento, es ampliamente utilizado en proyectos de IoT que requieren comunicación inalámbrica y control en tiempo real, como el sistema de riego inteligente presentado en este proyecto.

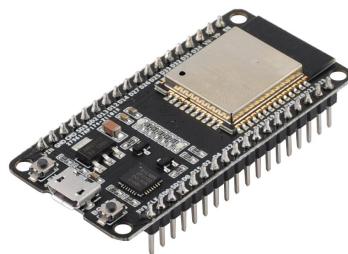


Figura 1: Placa de desarrollo ESP32

2.2. Sensor de Humedad del Suelo

El sensor de humedad del suelo empleado mide la conductividad eléctrica entre dos electrodos sumergidos en el sustrato. La resistencia varía en función del contenido de agua, permitiendo determinar el nivel de humedad del suelo mediante una señal analógica que el microcontrolador puede interpretar . Este dato es fundamental para activar la bomba de agua y mantener un riego adecuado.

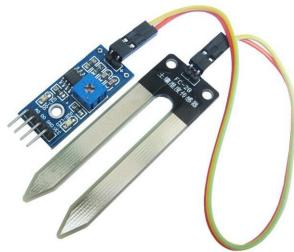


Figura 2: Sensor de humedad del suelo HR0040

2.3. Sensor DHT11

El sensor DHT11 es un dispositivo digital que mide la temperatura y la humedad relativa del aire. Cuenta con un sensor capacitivo para la humedad y un termistor para la temperatura, entregando valores en formato digital con una resolución suficiente para aplicaciones ambientales básicas. En el sistema, este sensor permite monitorear las condiciones atmosféricas que influyen en el estado de la planta.

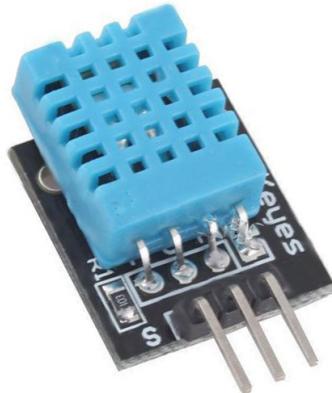


Figura 3: Sensor de humedad y temperatura del aire

2.4. Sensor PIR (Sensor de Movimiento)

El sensor PIR (Passive Infrared Sensor) detecta movimiento basado en cambios en la radiación infrarroja emitida por objetos en su campo de visión. Es utilizado para detectar la presencia o el paso de personas o animales cerca de la planta, habilitando alertas y reportes en la aplicación móvil cuando el sensor está activo .



Figura 4: Sensor de movimiento PIR

2.5. Módulo Relé de 5V

El módulo relé de 5V permite controlar dispositivos de corriente alterna o corriente continua de mayor voltaje y corriente, como la bomba de agua, utilizando la señal de baja tensión generada por el microcontrolador. Este módulo actúa como un interruptor electrónico que aísla el microcontrolador de las cargas eléctricas elevadas, garantizando la seguridad del sistema.

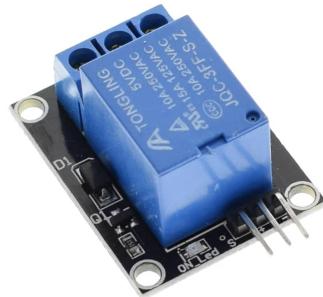


Figura 5: Modulo relevador de 5V

2.6. Bomba de Agua y Control de Flujo

La bomba de agua utilizada es una bomba de corriente continua de 5 V, diseñada para suministrar un flujo constante de agua adecuado para el riego de plantas pequeñas y macetas. La característica clave de esta bomba es su flujo nominal de aproximadamente 1.6 litros por minuto [4] (0.0267 litros por segundo), lo cual permite calcular con precisión el consumo total de agua durante su operación. La activación de la bomba se realiza mediante



un módulo relé de 5 V que funciona como interruptor electrónico, controlado directamente por el ESP32 para encender o apagar el riego según las condiciones detectadas por los sensores.

El control preciso del flujo de agua es esencial para evitar tanto el riego insuficiente como el exceso, optimizando el uso del recurso hídrico y promoviendo un desarrollo saludable de las plantas. Además, el sistema calcula y muestra en tiempo real el consumo acumulado de agua durante el funcionamiento de la bomba, facilitando un monitoreo eficiente.



Figura 6: Bomba de agua de 9V

2.7. Plataforma Blynk

Blynk es una plataforma IoT que permite la creación rápida de interfaces gráficas para el control y monitoreo remoto de dispositivos embebidos. A través de su aplicación móvil, los usuarios pueden visualizar datos, recibir notificaciones y controlar actuadores en tiempo real mediante widgets personalizables [1]. En este proyecto, Blynk facilita la supervisión de las condiciones ambientales y el manejo del sistema de riego de manera remota.

2.8. Pantalla LCD

El sistema utiliza una pantalla LCD alfanumérica de 16x2 caracteres, la cual está basada en el controlador estándar HD44780. Esta pantalla permite mostrar información importante como temperatura, humedad, humedad del suelo y consumo de agua de forma clara y legible para el usuario.



Figura 7: Pantalla LCD de 16x2

2.9. Comunicación I²C y módulo PCF8574

Para simplificar la conexión entre el microcontrolador ESP32 y la pantalla LCD, se utiliza un módulo adaptador I²C basado en el chip PCF8574. Este módulo reduce el número de pines necesarios para controlar la pantalla, usando solo dos líneas de datos, SDA y SCL. El chip PCF8574 actúa como un expansor de puertos que permite controlar los pines del LCD a través del protocolo I²C, facilitando la comunicación y disminuyendo la complejidad del cableado en el proyecto.

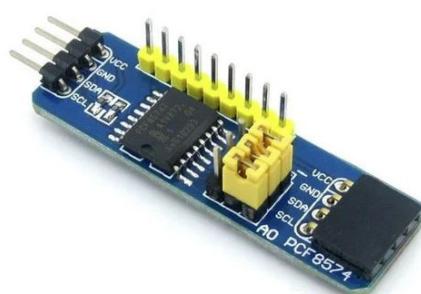


Figura 8: Módulo PCF



3. Desarrollo

El sistema de riego inteligente se desarrolló utilizando un microcontrolador ESP32, sensores ambientales, un módulo relé y una pantalla LCD para la visualización de datos. A continuación se describen los elementos principales y la lógica de funcionamiento implementada.

3.1. Microcontrolador y comunicación

Se empleó un ESP32 debido a su capacidad para conectarse a redes WiFi y ejecutar la plataforma Blynk, que permite el monitoreo remoto y control del sistema a través de una aplicación móvil. La comunicación con los sensores y actuadores se realiza mediante pines digitales y analógicos, así como por el bus I²C para la pantalla LCD.

3.2. Sensores y actuadores

El sistema integra los siguientes dispositivos:

- Sensor de humedad de suelo conectado al pin analógico GPIO36, que mide el porcentaje de humedad presente.
- Sensor de temperatura y humedad DHT11 conectado al GPIO4, que proporciona las condiciones ambientales.
- Sensor PIR conectado al GPIO27, detecta movimiento para funciones adicionales de monitoreo.
- Módulo relé de 5 V conectado al GPIO26, que controla el encendido y apagado de la bomba de agua.
- Botón físico en el GPIO33 para activar o desactivar manualmente el relé.

3.3. Lógica del programa

El código utiliza la biblioteca Blynk para conectar el dispositivo a la nube y permitir la interacción remota. Además, emplea temporizadores para leer periódicamente los sensores y actualizar la pantalla LCD.



La bomba de agua se activa o desactiva según el estado del botón físico o virtual. Durante su funcionamiento, se calcula el consumo aproximado de agua basándose en el tiempo que permanece encendida y un flujo estimado de 1.6 litros por minuto.

3.4. Visualización de datos

La pantalla LCD muestra en tiempo real la temperatura, humedad ambiental, humedad del suelo y el consumo acumulado de agua. La información es actualizada cada segundo para mantener una interfaz clara y útil para el usuario.

3.5. Funcionamiento del sistema

El sistema inicia la conexión WiFi y se sincroniza con la plataforma Blynk. Se configuran los pines y sensores, y se establecen las tareas periódicas mediante temporizadores para lectura y actualización de datos.

En el ciclo principal (`loop`), se evalúa el estado del sensor PIR para activar indicadores de movimiento, se ejecutan las tareas de Blynk y se monitorean las interacciones del usuario a través de botones físicos y virtuales.

Esta arquitectura modular permite que el sistema sea flexible, eficiente y escalable para futuras mejoras.



4. Código

Este fue el código utilizado explicado anteriormente:

Listing 1: Código de velocidad PID

```
// Viral Science www.viralsciencecreativity.com www.youtube.com/c/viralscience
// Blynk IOT Smart Plant Monitoring System

#define BLYNK_TEMPLATE_ID "TMPL2I5BNgZpU"
#define BLYNK_TEMPLATE_NAME "Sistema de riego inteligente"
#define BLYNK_AUTH_TOKEN "sP_bnZb903gxC06MNO-h4YMteKdv925c"

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <DHT.h>

#define BLYNK_PRINT Serial

// LCD
LiquidCrystal_I2C lcd(0x20, 16, 2);

// Pines de I2C
const int SDA_PIN = 21;
const int SCL_PIN = 22;

// WiFi y Blynk
char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "*****";
char pass[] = "*****";

// Blynk Timer
BlynkTimer timer;

// Pines
```



```
#define SOIL_SENSOR_PIN      36
#define PIR_PIN                27
#define RELAY_PIN_1             26
#define PUSH_BUTTON_1_PIN       33
#define VPIN_BUTTON_1           V12
#define VPIN_CONSUMO            V13

// DHT11
#define DHTPIN 4
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

// Estados
int PIR_ToggleValue = 0;
int relay1State     = LOW;
int pushButton1State = HIGH;

// Consumo
unsigned long tiempoEncendido = 0; // en segundos
float consumoLitros = 0.0;
const float LITROS_POR_SEGUNDO = 1.6 / 60.0; // 0.0267

void setup() {
    Wire.begin(SDA_PIN, SCL_PIN);
    Serial.begin(9600);

    lcd.begin(16, 2);
    lcd.backlight();
    lcd.setCursor(0, 0);
    lcd.print("  Initializing  ");
    for (int a = 5; a <= 10; a++) {
        lcd.setCursor(a, 1);
        lcd.print(".");
        delay(500);
    }
    lcd.clear();
}
```



```
pinMode(PIR_PIN, INPUT);
pinMode(RELAY_PIN_1, OUTPUT);
digitalWrite(RELAY_PIN_1, relay1State);
pinMode(PUSH_BUTTON_1_PIN, INPUT_PULLUP);

dht.begin();
Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);

timer.setInterval(100L, soilMoistureSensor);
timer.setInterval(2000L, DHT11sensor);
timer.setInterval(500L, checkPhysicalButton);
timer.setInterval(1000L, calcularConsumo); // cada segundo
}

// Lectura sensor humedad suelo
void soilMoistureSensor() {
    float value = analogRead(SOIL_SENSOR_PIN);
    value = map(value, 0, 4095, 0, 100);
    value = (value - 100) * -1;
    Blynk.virtualWrite(V3, value);
    lcd.setCursor(0, 1);
    lcd.print("S:");
    lcd.print(value);
    lcd.print(" ");
    Serial.print("HUMEDAD SUELO: ");
    Serial.println(value);
}

// Lectura sensor DHT11
void DHT11sensor() {
    float h = dht.readHumidity();
    float t = dht.readTemperature();
    if (isnan(h) || isnan(t)) {
        Serial.println("Error leyendo DHT11");
        return;
    }
}
```



```
Blynk.virtualWrite(V0, t);
Blynk.virtualWrite(V1, h);

lcd.setCursor(0, 0);
lcd.print("T:");
lcd.print(t);
lcd.print("C ");
lcd.print("H:");
lcd.print(h);
lcd.print("% ");
Serial.print("Temp: "); Serial.print(t);
Serial.print(" C     Hum: "); Serial.print(h); Serial.println("%");
}

// Sensor PIR
void PIRsensor() {
    bool value = digitalRead(PIR_PIN);
    Serial.print("PIR Sensor Value: ");
    Blynk.virtualWrite(V5, value);
    Serial.println(value);

    if (value) {
        Blynk.logEvent("pirmotion", "WARNING! Motion Detected!");
        WidgetLED LED(V5);
        LED.on();
    } else {
        WidgetLED LED(V5);
        LED.off();
    }
}

// Botón falso
void checkPhysicalButton() {
    if (digitalRead(PUSH_BUTTON_1_PIN) == LOW) {
        if (pushButton1State != LOW) {
            relay1State = !relay1State;
            digitalWrite(RELAY_PIN_1, relay1State);
        }
    }
}
```



```
Blynk.virtualWrite(VPIN_BUTTON_1, relay1State);
delay(500);
}
pushButton1State = LOW;
} else {
pushButton1State = HIGH;
}
}

// Estado del PIR desde Blynk
BLYNK_WRITE(V6) {
PIR_ToggleValue = param.asInt();
}

// Boton virtual conectado a rele
BLYNK_CONNECTED() {
Blynk.syncVirtual(VPIN_BUTTON_1);
}

BLYNK_WRITE(VPIN_BUTTON_1) {
relay1State = param.asInt();
digitalWrite(RELAY_PIN_1, relay1State);
}

// Funcion para calcular consumo de agua
void calcularConsumo() {
if (relay1State == HIGH) {
tiempoEncendido++;
consumoLitros = tiempoEncendido * LITROS_POR_SEGUNDO;
consumoLitros = round(consumoLitros * 100.0) / 100.0; // redondear a 2 decimales
}
}

Blynk.virtualWrite(VPIN_CONSUMO, consumoLitros);

lcd.setCursor(11, 1);
lcd.print("L:");
lcd.print(consumoLitros);
```



```
lcd.print("  "); // limpiar sobrantes

Serial.print("CONSUMO AGUA: ");
Serial.print(consumoLitros);
Serial.println(" L");

}

// Loop principal
void loop() {
    if (PIR_ToggleValue == 1) {
        lcd.setCursor(5, 1);
        lcd.print("M:ON ");
        PIRsensor();
    } else {
        lcd.setCursor(5, 1);
        lcd.print("M:OFF");
        WidgetLED LED(V6);
        LED.off();
    }

    Blynk.run();
    timer.run();
}
```



5. Resultados

Durante la implementación y pruebas del sistema de riego inteligente, se logró monitorear en tiempo real la humedad del suelo, la temperatura y humedad ambiental, así como el consumo aproximado de agua mediante el tiempo de encendido de la bomba.

El cálculo del consumo se basó en un flujo estimado de 1.6 litros por minuto, obtenido de especificaciones técnicas de la bomba. Sin embargo, para obtener mediciones más precisas, hubiera sido ideal utilizar un caudalímetro que permitiera registrar el flujo real de agua. No obstante, los caudalímetros suelen ser dispositivos costosos de integrar en proyectos de bajo presupuesto, por lo que en esta etapa se optó por el método indirecto basado en el tiempo de operación del relé.

Los datos obtenidos mostraron una respuesta adecuada del sistema a las condiciones del suelo y del ambiente, con la capacidad de activar o desactivar la bomba manualmente.

La pantalla LCD proporcionó información clara y continua al usuario, facilitando la supervisión del estado del sistema sin necesidad de conexión remota.

En conjunto, los resultados confirman que el sistema cumple con los objetivos planteados de monitoreo y control eficiente del riego, con margen para futuras mejoras en la precisión del consumo de agua.

6. Conclusión

El desarrollo del sistema de riego inteligente basado en ESP32 demostró ser una solución efectiva para el monitoreo y control automatizado de las condiciones del suelo y del ambiente, optimizando el uso del agua mediante la activación precisa de la bomba.

El uso de sensores como el DHT11 y el sensor de humedad de suelo permitió obtener datos confiables que se integraron exitosamente con la plataforma Blynk para su supervisión remota, mientras que el control local mediante un relé y un botón físico aumentó la versatilidad del sistema.

Aunque el cálculo del consumo de agua fue aproximado debido a la ausencia de un caudalímetro, el método basado en el tiempo de operación resultó suficiente para estimar el volumen utilizado, manteniendo costos accesibles.



La implementación de la pantalla LCD y el módulo PCF facilitó la interacción directa del usuario con el sistema, mejorando la experiencia y el control en sitio.

Finalmente, el proyecto abre la puerta para futuras mejoras, como la incorporación de sensores más precisos y la integración de algoritmos inteligentes para optimizar aún más el riego, contribuyendo a un uso sostenible de los recursos hídricos.



Referencias

- [1] Blynk Inc. Blynk documentation. <https://docs.blynk.io/en>, 2024. Documentación oficial para integrar sensores y dispositivos IoT con Blynk.
- [2] Viral Science - The home of Creativity. New blynk iot smart plant monitoring system. https://www.youtube.com/watch?v=PTJ9sAk2c2I&t=1s&ab_channel=ViralScience-ThehomeofCreativity, 2022. Video guía para implementar monitoreo de planta con ESP32 y Blynk.
- [3] Espressif Systems. Esp32 series datasheet. https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf, 2023. Ficha técnica del microcontrolador ESP32 usado en el sistema.
- [4] Electrozone México. Mini bomba de agua sumergible 3-5v mb-35. https://electrozone.com.mx/tblarticulos_fulltext.php?field=Descripcion&key1=MB-35, 2023. Especificaciones técnicas de la bomba utilizada en el proyecto.