Line Follower Robot_with PID control

1st Bruno Chiabrando
Ingeniería Electrónica UTN-FRSFCO
Universidad Tecnológica Nacional
Facultad Regional San Francisco
San Francisco, Córdoba Argentina
bchiabrando@facultad.sanfrancisco.utn.edu.ar

2nd Angel Aguilar
Ingeniería Electrónica UTN-FRSFCO
Universidad Tecnológica Nacional
Facultad Regional San Francisco
San Francisco, Córdoba Argentina
aaguilar@facultad.sanfrancisco.utn.edu.ar

3rd Rodolfo Podadera
Ingeniería Electrónica UTN-FRSFCO
Universidad Tecnológica Nacional
Facultad Regional San Francisco
San Francisco, Córdoba Argentina
rpodadera@facultad.sanfrancisco.utn.edu.ar

Abstract—Line Follower Robot is controlled by a proportional-integral-derivative system known as PID is developed, the parameters of this system can be configured at any time through a simple user interface (UI), also the error obtained by the TCRT5000 sensor array is plotted in real time. The ESP32 microcontroller with FreeRTOS is used to be able to perform the multiple tasks simultaneously, it controls sensors, actuators and the communication with the web server through HTTP Request/Response. The final device has a correct operation and a high fidelity when it comes to stabilizing in the changes of the circuit to follow, in this way it meets the main objectives, reaching greater efficiency than conventional speed robots.

Keywords—PID, Robot, Follower, FreeRTOS, HTTP

I. INTRODUCCIÓN

El presente trabajo busca realizar un robot que sea capaz de recorrer cualquier circuito que posea una línea central continua de manera autónoma. El dispositivo cuenta con un microcontrolador, encargado de controlar a través de un sistema proporcional – derivativo – integral (PID) la velocidad de los motores lo cual nos permite alcanzar una mayor eficiencia y velocidad a la hora de seguir la línea. Los parámetros de dicho controlador PID pueden ser monitoreados y modificados desde una interfaz de usuario (UI) la cual se puede acceder de manera local a través de un servidor web creado por el microcontrolador ESP32. Para alcanzar una mayor eficiencia tanto en el funcionamiento como en la comunicación con el usuario se hace uso de FreeRTOS que nos permite ejecutar las distintas tareas necesarias en simultaneo, optimizando los recursos brindados por el microcontrolador seleccionado.

II. OBJETIVOS

- ✓ Elaborar un control PID para el manejo de velocidades de los motores que permita el seguimiento de una línea.
- ✓ Programar la comunicación entre cualquier dispositivo con conexión a la red local y el microcontrolador ESP32 para la calibración inalámbrica del controlador del robot seguidor de línea.
- ✓ Alcanzar luego de pruebas y re calibraciones una mayor eficiencia que robots velocistas convencionales.

III. METODOLOGÍA DE DESARROLLO

A. Robots seguidores de línea

Son capaces de seguir un camino trazado por una línea central continua. La cual es de un color que contrasta con el del piso, por lo general la línea es blanca y el resto del suelo es negro. Una de las virtudes de esta configuración es que se puede realizar de múltiples maneras y pueden ser tan complejos como la persona que lo realice quiera. Año tras año las tecnologías utilizadas para desarrollarlos evolucionan y uno de los motivos principales son las distintas competencias que se realizan. El robot desarrollado cumple con las reglas explícitas por la Liga Nacional de Robótica de Argentina por lo que se encuentra apto para competir en las próximas ediciones.

B. Chasis

Se fabricó de manera aditiva (impresora 3D) respetando las dimensiones máximas de los robots velocistas que deben ser de 20 cm de largo x 14 cm de ancho, con un límite en altura de 10cm. En la figura 1 se puede apreciar un render 3D del diseño.

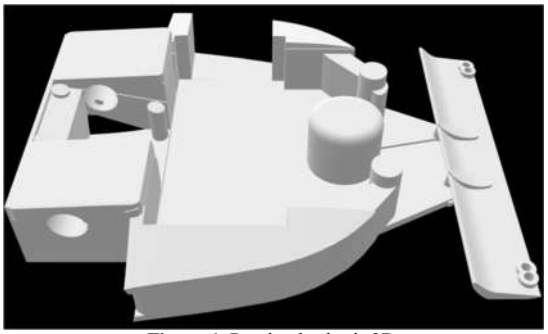

Figura 1. Render de chasis 3D.

C. Actuadores

Los mismos son los elementos principales del robot velocista, y de ellos depende el desempeño que se pueda alcanzar, aquí se utiliza un moto reductor pololu HP, 6V de corriente continua. El término reducción refiere a cuantas vueltas dará el motor para que los engranajes produzcan una vuelta a la salida del eje, de esta manera se incrementará el torque a costas de una reducción de velocidad.

Figura 2. Moto reductor.

D. Sensores

Para la detección de la línea y ubicación del robot se utilizaron ocho sensores ópticos infrarrojos TCRT5000 dispuestos en un circuito impreso y fueron trabajados de manera digital conectando cada uno de los sensores a una entrada digital y se los alimentó con 3.3 V.

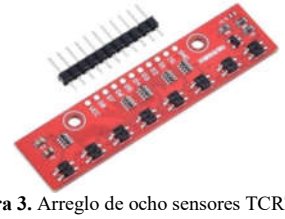

Figura 3. Arreglo de ocho sensores TCRT5000.

E. Circuito impreso de potencia

Para el control de los actuadores se diseñó un circuito encargado de recibir una señal de control PWM y alimentar a los motores en mayor o menor medida.

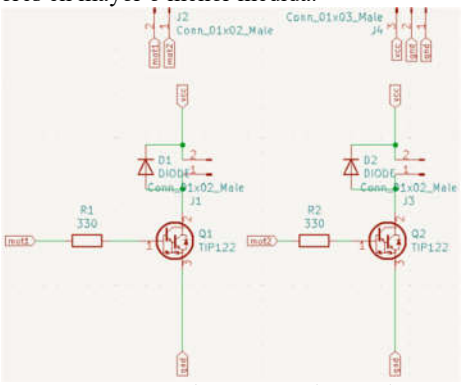

Figura 4. Esquemático para controlar actuadores.

El TIP122 está formado por un par de transistores NPN en conexión Darlington, su funcionamiento es igual al de un NPN. Excitando la base del mismo a través de una señal PWM podemos establecer un flujo de corriente entre emisor y colector y así alimentar a los motores.
En esta placa podemos encontrar el conexionado de los dos motores como la alimentación y la señal de PWM proveniente de la placa de control.

F. Microcontrolador

Como se dijo anteriormente para el control de este dispositivo se utiliza el microcontrolador de Espresif ESP32 en formato dev module. El mismo integra Wifi, Bluetooth, los GPIOS necesarios y tambièn brinda la posibilidad de uso de sistema operative FreeRTOS, que nos permite administrar un sistema multitareas robusto para obtener aprovechar de la manera más óptima el algoritmo.

Figura 5. Microcontrolador.

G. Circuito impreso de control

Para controlar el robot se diseñó una segunda placa en donde va ubicado el módulo ESP32 junto con sus periféricos, los distintos pines para conexionado de sensores, motores, y alimentación de la placa de potencia.

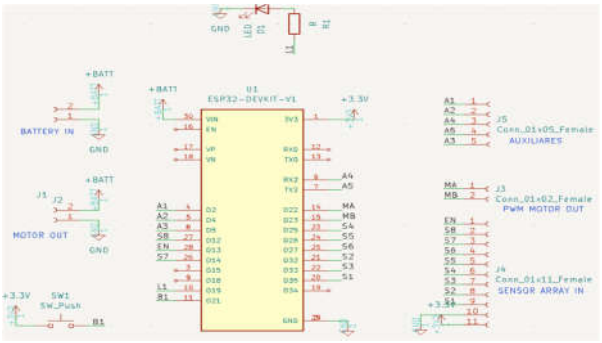

Figura 6. Esquemático Circuito de Potencia.

IV. CONTROLADOR PID

El control PID (control proporcional, integral y derivativo) es un control por realimentación que calcula la desviación o error entre un valor medido y el valor que se quiere obtener (set point), para aplicar una acción que corrija el proceso. En el caso del robot seguidor de línea, el controlador PID (rutina basada matemáticamente), procesa los datos del sensor y los utiliza para controlar el ángulo de giro a través de la velocidad enviada a cada motor, para de esta forma mantenerlo en curso.

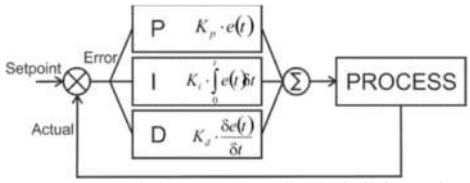

Figura 7. Esquema controlador PID (Aprendiendo facilelectrónica-2014).

Para el análisis de nuestro robot seguidor de línea se toma de base la estructura básica de un sistema de control de lazo cerrado. Lo cual en una representación en bloques queda como se observa.

Figura 8. Diagrama de Sistema de control a lazo cerrado.

Cada una de las etapas presentes se modela de manera virtual a través de MATLAB para poder realizar una simulación del robot en su totalidad.

A. Modelado de cinemática directa del robot

Comenzamos con el modelado del movimiento del robot, para esto se asumieron una serie de consideraciones para simplificar el análisis: el desplazamiento es causado únicamente por la rotación de las ruedas y la superficie en donde se desplaza es totalmente lisa. En este caso se toman en cuenta condiciones ideales para simplificar el modelo al considerar las pequeñas dimensiones del móvil. En una configuración diferencial se consideran tres grados de libertad: las distancias $\boldsymbol{x}$ e $\boldsymbol{y}$ de posición y el ángulo $\boldsymbol{\varphi}$ de orientación. Para poder lograr un movimiento controlado, se deben tener dominio sobre "Vr" la velocidad de la rueda derecha y "Vl" la velocidad de la rueda izquierda. El objetivo de modelar el robot es buscar una relación directa de cómo afectan las entradas "Vr y Vi", a los estados del sistema "x,

y, $\boldsymbol{\varphi}$". Podemos observar los distintos grados de libertad, y se resalta su dependencia de la velocidad de cada motor.

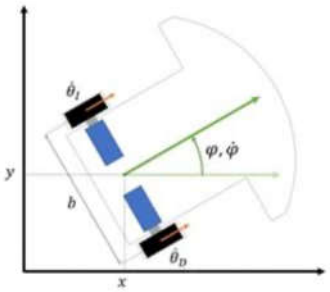

Figura 9. Movimiento de robot tipo tracción diferencial.

Una vez definidas las entradas y salidas de nuestro sistema, notamos que nuestro robot puede trasladarse con una velocidad lineal "v" y rotar con una velocidad angular "ω".

$$v = r\,(Vr + Vl)/\,2 \quad (1)$$
$$\omega = r\,(Vr - Vl)/\,b \quad (2)$$

Siendo r = Radio de la rueda. b = distancia entre las ruedas Vr = Velocidad motor derecha. Vl = Velocidad motor izquierda. Teniendo las velocidades definidas planteamos las distintas ecuaciones que describen el movimiento del robot.

$$x = v * \cos\varphi \quad (3)$$
$$y = v * \text{sen}\,\varphi \quad (4)$$
$$\varphi = \omega \quad (5)$$

Trabajando algebraicamente con las ecuaciones alcanzadas llegamos a las ecuaciones 6, 7 y 8 que determinan el movimiento a partir de la tracción diferencial.

$$x = [(Vr + Vl)\ \text{r}\cos\varphi]\ /2 \quad (6)$$
$$y = [(Vr + Vl)\ \text{r sen}\,\varphi]\ /2 \quad (7)$$
$$\varphi = [(Vr - Vl)\ \text{r}\ /b \quad (8)$$

Al integrar las ecuaciones (6), (7) y (8) se consigue la posición y orientación en un sistema de referencia absoluto. Utilizando programación a bloques en Simulink MATLAB modelamos la planta y nos queda como se observa en la figura 10.

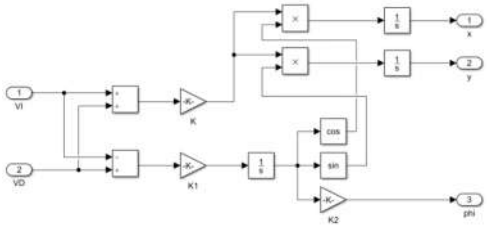

Figura 10. Subsistema, cinemática de la planta.

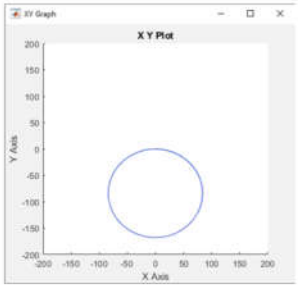

Figura 11. Desplazamiento en el plano.

Una de las primeras pruebas realizadas fue ingresar dos velocidades diferentes y graficar su posición en x e y a través de un gráfico XY y obtuvimos que el robot describe un movimiento circular tal como se observa en la figura 11 de esta manera verificamos que el modelado descripto responde correctamente con lo esperado.

B. Modelado de motores de corriente contínua

El movimiento del robot, se realiza con los motoreductores de 6V de corriente continua, cuyas características extraídas de hoja de datos se pueden apreciar en tabla 1:

Parámetro	Símbolo	Valor
Relación	η	30:1
Voltaje nominal	$V_{nominal}$	6V
RPM sin carga	Θ_{free}	104.72 rad/s
Corriente máxima	i_{max}	1.6 A
Corriente nominal	i_{free}	100 mA
Torque máximo	T_{max}	0.0557 Nm

Tabla 1. Tabla de valores extraídas de datacheet motor DC

Para obtener el modelo se analiza la velocidad angular de cada rueda a partir del circuito equivalente para un motor de corriente directa, tal como se muestra en la figura 12.

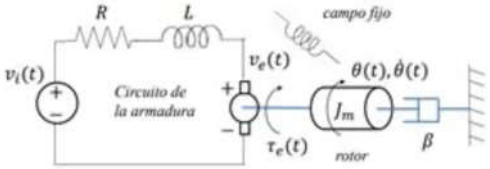

Figura 12. Circuito electromagnético de motor DC

Debido a que los parámetros proporcionados por el fabricante pertenecen a la salida del tren de engranes, se reducirá el estudio al considerar solamente un momento de inercia. A continuación, se obtienen las ecuaciones diferenciales de la parte eléctrica por la ley de voltajes de Kirchoff y de la parte mecánica por la ley de Newton.

$$\text{Ri}\,(t) + \text{L}\ \text{di}(t)/dt\ + v_e(t) = v_i(t) \quad (9)$$

Las ecuaciones del subsistema magnético son:

$$\boldsymbol{T}(t) = \text{P}(t) / \omega_m(t)\ = k_t\ \text{I}(t) \quad (10)$$
$$\text{e}(t) = k_e\ \omega_m(t)\ = k_e\ d\Theta(t)/dt \quad (11)$$

Reemplazando (11) aplicando Laplace a la ecuación (9), obtenemos la ecuación para la parte eléctrica.

$$\text{R I}(s) + \text{L s I}(s) + \text{ke s}\ \theta(s) = \text{Vi}\ (s) \quad (12)$$

Por otro lado, para la parte mecánica del motor aplicamos las leyes de Newton para movimientos mecánicos rotacionales y obtenemos la ecuación 13.

$$T(t) = J\text{m}\ d^2\Theta(t)/dt\ + b\ d\Theta(t)/dt \quad (13)$$

$$\mathcal{L}[T(t)] = T(s) = J\text{m}\ s^2\,\Theta(s) + b\ s\ \theta(s) \quad (14)$$

Reemplazando 10 en 14 se obtiene:

$$k\ I(s) = J\text{m}\ s^2\Theta(s) + bs\ \theta(s)\ \therefore\ k_t\ \text{I}(s)/(J_m\ s\ +b)s = \Theta(s) \quad (15)$$

Reemplazando 15 en 12 llegamos a:

$$\text{R I}(s) + \text{L s I}(s) + k_e\ [k_t\ \text{I}(s)\ /\ (J_m\ s\ +b)] = V_i\ (s) \quad (16)$$

Finalmente, para obtener la relación buscada entre la salida del motor y la entrada de voltaje tal como se observa en la ecuación 17.

$$\Theta(s)/V_i(s) = k_t / [(R + Ls)(J_m s + b) + k_e k_t] = k_t / [J_m L s^2 + (J_m R + b L) s + R b + k_e + k_t] \quad (17)$$

Las características eléctricas de Resistencia e Inductancia de armadura se obtienen a través de la medición del dispositivo físico, mientras que el voltaje de entrada será el valor nominal que especifica el fabricante. **R** = 31,3 Ω **L** = 5,1 mH **vi(t)** = 6V**.** Los valores de las constantes eléctricas y magnéticas los calculamos con los datos presentados en la tabla 1 correspondientes a los actuadores utilizados.

$$k_e = v_{nominal} / \Theta_{free} = 0.0573 \quad v / (rad/seg) \quad (18)$$

$$k_t = T_{max} / I_{max} = 0.034 \text{ N m /A} \quad (19)$$

Los parámetros mecánicos de momento de inercia y la constante de fricción al ser más difíciles de cuantificar, se hicieron de uso de unos valores pequeños obtenidos por un motor similar al utilizado para este trabajo, los valores son los siguientes:

$$J_m = 1x10^{-5}\, kgm^2 \qquad \beta = 1x10^{-7} Nm/(rad/s)$$

Una vez obtenidos los modelados de estos bloques que son de vital importancia para nuestro sistema, unimos cada uno de ellos y finalmente obtenemos un diagrama de nuestra función de transferencia final. Como vemos en la figura 13.
En nuestro diagrama se presenta un voltaje inicial base con el cual ambos motores se ven afectado en un inicio, nuestra retroalimentación va a afectar negativamente al voltaje de entrada del motor izquierdo y positivamente al voltaje de entrada del motor derecho y de esa manera logrará el giro deseado.

Figura 13. Simulación del sistema en Simulink

Al iniciar la simulación con un escalón unitario en su entrada obtenemos la respuesta que se puede observar en la figura 14, en la misma se presenta una oscilación elevada previa a su estabilización, esta es la que se corregirá cuando se aplique el controlador PID.

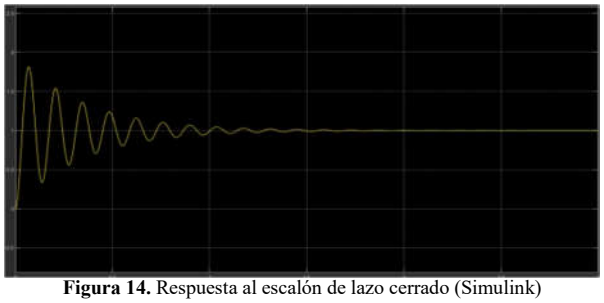

Figura 14. Respuesta al escalón de lazo cerrado (Simulink)

Para obtener una única función de transferencia de nuestro sistema se pueden aplicar técnicas de simplificación o algún software que facilite cálculos, dicha ecuación ya resuelta es la 20.

$$\frac{1.27e07s^2 + 7.814e10\, s + 4.872\, e\, 11}{s^5 + 1.227\, e04\, s^4 + 3.774\, e07\, s^3 + 4.696e08\, s^2 + 1.464e9s} \quad (20)$$

Como vemos en la figura 15 la respuesta al escalón calculada con Matlab es la misma a la obtenida anteriormente, por lo que podemos verificar que la función de transferencia obtenida es verdaderamente la que describe a nuestro sistema. Además, se observa la respuesta en lazo abierto en donde se ve como no converge en un punto.

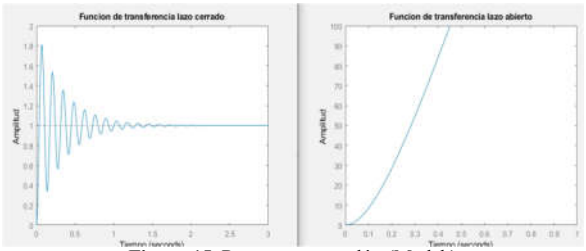

Figura 15. Respuesta a escalón (Matlab)

Una vez obtenida esta respuesta hacemos uso de nuestro controlador PID para mejorar la estabilización de nuestro sistema, a través de Matlab llegamos a la respuesta que se observa en la figura 16, esta respuesta fue obtenida sin hacer uso del término integrador del controlador ya que no existe error estacionario y con las constantes Kp = 0.01 y Kd = 0.001.

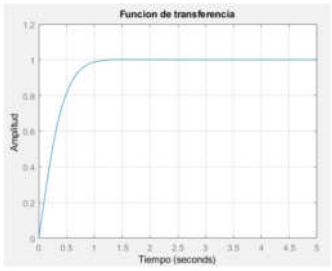

Figura 16. *Respuesta con controlador PID (Matlab)*

Junto con la respuesta obtenida se calculó el lugar geométrico de las raíces además del mapa de polos y ceros, estos gráficos se pueden observar en la figura 17, la cual nos va a ser útil para realizar algunos análisis.

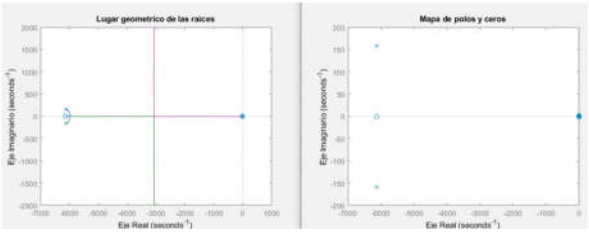

Figura 17. Gráficos de polos y ceros (Matlab)

En estos gráficos podemos observar tanto la posición de los polos y las raíces de nuestro sistema como los posibles valores que pueden tomar al modificar alguno de los parámetros. Al ser una función de transferencia de grado 5 el análisis puede llegar a ser más complejo de lo habitual, pero podemos notar que nuestro sistema no puede caer en la inestabilidad ya que los posibles valores que pueden tomar son negativos. Las respuestas que se pueden alcanzar son "Sub-amortiguado", "críticamente amortiguado" o "sobre amortiguado". Si analizamos, además, el origen vemos que existe la presencia de un polo en él, esto es así ya que en la función de transferencia (ecuación 20) no tenemos un término independiente en el denominador. En términos matemáticos esto significa que la función de transferencia ya posee un término integrador 1/s, lo que ocasiona que no sea necesario

utilizarlo en el controlador PID. Por otro lado, también se obtuvo el diagrama de bode correspondiente a nuestro Sistema como se aprecia en la figura 18, con el efecto de sus polos y ceros y donde se puede analizar la frecuencia de cambio de set point a la que se ve afectada la fase y la amplitud.

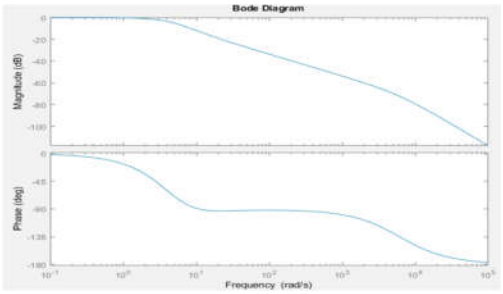

Figura 18. Diagrama de bode del sistema (Matlab)

Tal como se realizaron los cálculos y simulaciones para un tiempo continuo se hicieron también, para un tiempo discreto obteniendo respuestas similares (figura 19).

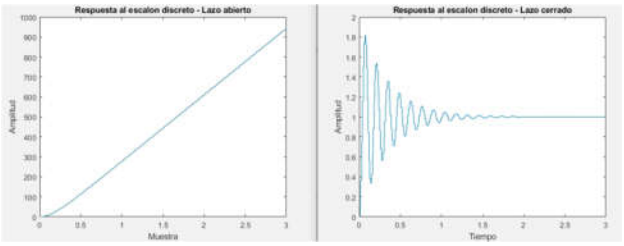

Figura 19. *Respuesta al escalon discreta (Matlab)*

De la misma manera que en la simulación continua, en la discreta le aplicamos un controlador PID lo calibramos y obtenemos una salida como la que se muestra en la figura 20. Las constantes de calibración alcanzadas son Kp=130, Kd=1 y nuevamente no se hizo uso del término integrador.

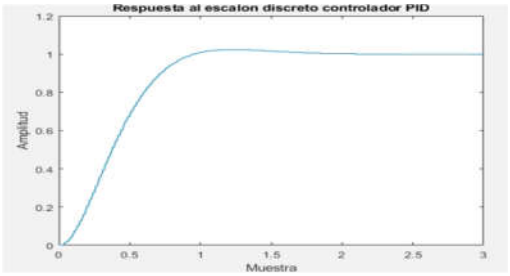

Figura 20. *Respuesta discreta con controlador PID*

Con todos los valores calculados, podemos finalizar la construcción de la simulación en Simulink, uniendo los bloques modelados junto al controlador PID como indica la figura 21. Se puede observar el ángulo de giro y el desplazamiento en el plano de nuestro robot simulado a través de XY ploter.

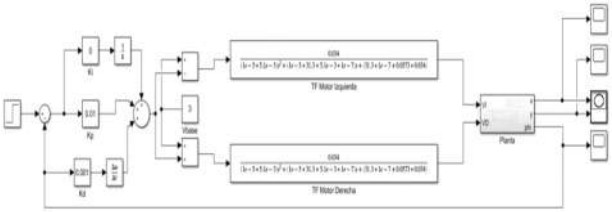

Figura 21. Line follower Robot modelado

V. FIRMWARE

El código se hizo de manera modular, separando las funcionalidades en librerías para un mejor manejo y simpleza a la hora de realizar modificaciones.

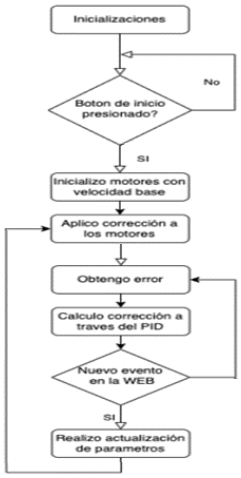

Figura 22. Diagrama de flujo del código

En la figura 22 podemos observar de una manera resumida las distintas tareas que tiene que ejecutar el micro. Para dar inicio al programa, por reglamentación de la Liga Nacional de Robótica, se debe presionar un pulsador y al liberarlo recién pueden inicializarse los motores. Para realizar este proceso se utilizó una máquina de estado encargada de eliminar los posibles rebotes al pulsar y de inicializar los motores en el momento que se libera el botón. Para el caso de la lectura de los sensores se hizo una librería utilizando programación orientada a objetos para facilitar la utilización de las distintas funciones a lo largo del código.

Figura 23. Interfaz de ususario

La interfaz de usuario mostrada en la figura 23 se realizó con HTML, estilizándolo con CSS, dándole funcionalidad con JS. Estos tres archivos se encuentran guardados en la memoria flash de la ESP32 a través de SPIFFS que es un sistema de archivos ligero creado para microcontroladores.

En la misma interfaz se graficará el error sensado por el robot en tiempo real, para así poder ver cómo responde la corrección de nuestro controlador PID. Una vez calculado el ajuste haremos uso de nuestra librería para el manejo de los motores, la misma le brinda a cada uno de ellos la misma velocidad de partida a través de dos canales de PWM con una frecuencia de 5Khz y 10 bit de resolución dando la posibilidad de un PWM de 0 a 1023.

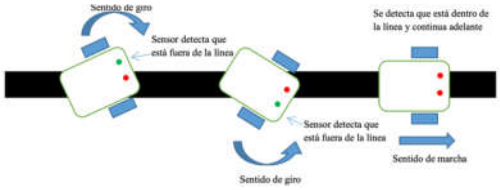

Figura 24. Esquema de detección de línea – Extraída de Martínez Ortiz, David (2015-2016)

El ajuste calculado por el PID se aplicará de manera positiva en el canal PWM correspondiente al motor de la derecha y negativamente al motor de la izquierda, debido a la manera en que planteamos el error.
En la figura 24, se puede observar cómo reacciona el robot al detectar que está fuera de línea (su set point), a través de la corrección calculada por el PID genera una diferencia de velocidad entre los dos motores logrando así un giro en un sentido determinado.
El algoritmo para realizar dicha corrección se ejecuta de manera continua, utilizando el tiempo transcurrido para calcular tanto el término derivativo como el integral, y el proporcional es obtenido con el error sensado. Para hacer uso de este código se necesita trabajar de manera asincrónica haciendo uso de distintos núcleos o de tareas en simultaneo.
Para el control general de las distintas etapas del robot se programó una máquina de estados encargada de monitorear y comandar al robot, esta es una de las tareas principales y corre en el núcleo 1 del microcontrolador.

```
int PIDUpdate(int error){
  currentTime = millis();
  elapsedTime = (double)(currentTime - previousTime);
  cumError += error * elapsedTime; // La integral es el error acumulativo en el tiempo
  rateError = (error - lastError)/elapsedTime; // La derivada es la tasa de cambio del
error

  int out = Kp*error + Ki*cumError + Kd*rateError;

  lastError = error;
  previousTime = currentTime;

  return out;
}
```

Código 1. Algoritmo de controlador PID

En simultáneo una tarea se ejecuta en el núcleo 2, para que permita obtener el error a través de los sensores, el cual será enviado al controlador PID para que realice los cálculos correspondientes y se obtenga la corrección de velocidad que será aplicada a los motores para alcanzar nuevamente el set point.
El prototipo final se puede ver en la figura 25. En el chasis se encuentran montados todos los periféricos que se nombraron anteriormente, los motores, el array de sensores, los circuitos impresos y las baterías interconectadas.

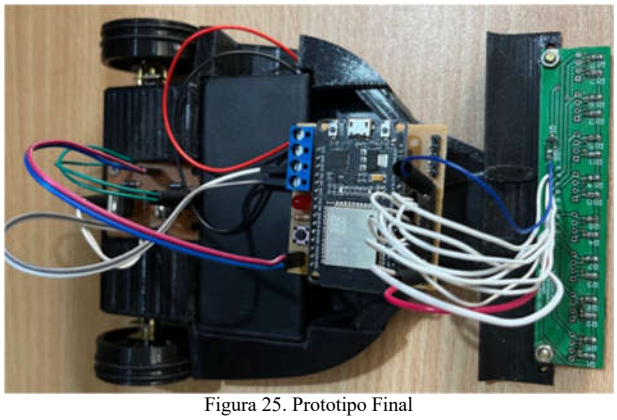

Figura 25. Prototipo Final

VI. RESULTADOS

En el primer encendido del prototipo se debió obtener a prueba y error la velocidad de partida necesaria para que el robot se traslade sobre una línea recta y que dicho valor no sea demasiado alto como para que salga de pista. Una vez configurado el valor de velocidad de partida se calibraron los parámetros del controlador PID, primeramente, la constante del término proporcional Kp, la misma presentó un correcto funcionamiento al valor de 50, en el caso de Ki al no existir error estacionario se dejó en 0, y Kd dio buenos resultados en un valor de 4. Esto se pudo calibrar de una manera sencilla desde la interfaz de usuario.

VII. CONCLUSIONES

La interfaz de usuario permite una configuración inalámbrica y facilita la rapidez y facilidad de la calibración.
La posibilidad de monitorear gráficamente el error actual en todo momento brinda facilidad para deducciones y nuevas calibraciones. Tanto el algoritmo del controlador PID como el de FreeRTOS permite una rápida acción de los motoreductores, ya que con cada lectura de los sensores en pista puede predecir los errores y usar los anteriores para mejorar su funcionamiento y, por tanto, la autonomía del movimiento de cada motor sea independiente según lo requiera.
Gracias al análisis de los polos y ceros se detectó que el termino integrador ya estaba en la función de transferencia que describía al sistema, ya que no existía término independiente en el denominador lo que generaba un polo en el origen. Esto describe un integrador mecánico, que en nuestro sistema se presenta ya que al realizar una corrección tras un error sensado no vuelve al origen si no que se mantiene en esa corrección. Al usar nuestro prototipo por un tiempo prolongado notamos que la respuesta de nuestro sistema varía, exigiendo una nueva calibración para volver a su comportamiento ideal. Esto es así ya que el sistema con el cual estamos trabajando no es invariante en el tiempo, luego de un determinado período su fuente de alimentación varía, ocasionado por la descarga de las baterías lo que lleva a una des calibración del sistema.

VIII. REFERENCIAS

[1] Parikh, P., Shah, H., & Sheth, S. (2014, June). A Mechatronics design of a line tracker robot using Ziegler Nichols control technique for P, PI and PID controllers. In International Mechanical Engineering Congress (IMEC-2014) (pp. 13-15).

[2] Martinez Ortiz, David (2015-2016). Robótica para seguimiento de líneas (robotics for line tracking). Escola Tecnica Superior d'Enginyeria de Telecomunicació de Barcelona (pp. 1-106).

[3] López Alarcón, Luis David (2021). Implementación de un algoritmo de control para un robot line follower usando red neuronal. Universidad Politécnica Salesiana Sede Guayaquil. (pp. 1-107).

[4] L. E. Solaque Guzmán, M. A. Molina Villa y E. L. Rodríguez Vásquez, «Seguimiento de Trayectorias con un Robot Móvil de Configuración Diferencial,» Ing. USBMed, vol. 5, nº 1, 2014.

[5] J. I. Collazo Cuevas, E. Gorrostieta Hurtado, J. C. Pedraza Ortega, U. G. Villaseñor Carrillo, R. A. Romero Torres y M. A. González Aguirre, «Modelación de un Robot Móvil de Dos Ruedas con Tracción Diferencial,» 2009.

[6] Aucatoma Tiban Jairo Patricio (Febrero 2022). Diseño E Implementación De Un Robot Velocista Controlado Mediante Wifi Utilizando Una Tarjeta De Desarrollo Esp32- Wroom. Instituto Superior Tecnológico Vida Nueva (pp. 27-40).

[7] Aprendiendofacilelectronica. (2014). «Robot Velocista de compentencia Algoritmo PID,» Diciembre 2014. [En línea]. Available: http://aprendiendofacilelectronica.blogspot.com/2014/12/robot-velocista-decompetencia_4.htm