



MICROMOUSE

Richard Gómez, Santiago Alarcón

*Escuela de Ciencias Exactas e Ingeniería, Universidad Sergio Arboleda
Bogotá, Colombia*

richard.gomez01@usa.edu.co, santiago.alarcon01@usa.edu.co

Abstract

El presente proyecto desarrolla un sistema autónomo de navegación para un robot Micromouse capaz de explorar, mapear y resolver un laberinto utilizando algoritmos de control, sensado y planificación basados en técnicas clásicas de robótica móvil. Se implementó un esquema completo de control de movimiento con perfiles trapezoidales de velocidad, regulación mediante PID y odometría con encoders, garantizando desplazamientos precisos entre celdas del laberinto. Para la percepción del entorno se emplearon sensores infrarrojos dispuestos frontal y lateralmente, permitiendo la detección de paredes y la construcción dinámica del mapa interno. La planificación de rutas se realizó mediante el algoritmo Flood-Fill, que calcula campos de distancia desde el objetivo para determinar el camino óptimo durante la fase de exploración y la posterior ejecución del recorrido rápido (Fast Run).

INTRODUCCIÓN

Inicialmente, se midieron los pulsos del encoder en tiempo real utilizando el modo encoder de los temporizadores del microcontrolador, y se realizó la conversión de pulsos a rad/s y posteriormente a mm/s. Principalmente la interfaz Qt permitió seleccionar el puerto serial, enviar consignas de velocidad, visualizar en tiempo real la respuesta del sistema mediante un gráfico de tiempo vs velocidad y registrar los datos en un archivo CSV para su análisis posterior.

Por otra parte, se implementó un **filtro FIR** para mejorar la calidad de la señal de velocidad y se usaron funciones matemáticas para convertir voltaje a PWM y velocidad deseada a voltaje, optimizando así la respuesta del sistema. La parte final de proyecto, consistió en la caracterización

la relación velocidad-voltaje, evaluando el tiempo de estabilización según la gráfica.

Para el control del desplazamiento se implementaron estrategias basadas en perfiles trapezoidales de velocidad, permitiendo movimientos suaves, estables y reproducibles. La medición precisa de la posición y velocidad se efectuó mediante encoders conectados a temporizadores configurados en modo contador cuadratura, mientras que el control de los motores se realizó mediante modulación por ancho de pulso (PWM) regulada por controladores PID. Estas técnicas garantizan que el robot avance de forma controlada entre celdas del laberinto, cumpliendo las condiciones necesarias para la exploración sistemática.

METODOLOGÍA

Para medir con precisión la velocidad real del motor, se utiliza un sensor encoder, el

cual genera pulsos que indican cuánto ha girado la rueda. A partir de esos pulsos se calcula la velocidad y se compara con la velocidad deseada. Por otra parte, El motor fue controlado mediante señales PWM generadas por la tarjeta STM32. El encoder se conectó a los pines de un temporizador configurado en **modo encoder**, lo que permitió contar pulsos a medida que el motor giraba. La lectura de los pulsos del encoder se utilizó para calcular la **velocidad angular** y posteriormente la **velocidad lineal** de la rueda en mm/s. Esta conversión se realizó a partir del número de pulsos por vuelta y el diámetro de la rueda, relacionando el giro físico con el desplazamiento lineal. Se implementó un **control FeedForward**, que consistió en aplicar un valor de PWM previamente calculado en función de la velocidad deseada. Para ello, se caracterizó la respuesta del motor midiendo la velocidad real para diferentes niveles de voltaje, obteniendo así una relación lineal que permitió estimar el PWM adecuado para alcanzar una velocidad específica.

Con el fin de mejorar la calidad de la señal de velocidad, se implementó un **filtro FIR**, reduciendo fluctuaciones y ruido en las mediciones. Luego, los datos fueron analizados para evaluar el comportamiento del motor, el tiempo de estabilización y la precisión del control.

Calculo para el encoder

$$RPM = \frac{encoder_{motor1} \cdot 60}{pulsos_{vuelta} \cdot (tiempobase \cdot 100 \cdot 10^{-6})}$$

encoder_motor_1 = número de pulsos contados en el intervalo de muestreo.

PULSOS_VUELTA = cantidad de pulsos que genera el encoder por cada vuelta completa.

tiempo_base = intervalo de muestreo (en unidades base del sistema).

100e-6 = factor de corrección (por ejemplo, si el tiempo base está en pasos de 100 µs).

60 = factor de conversión de revoluciones por segundo a revoluciones por minuto (RPM).

RESULTADOS Y DISCUSIONES

Al principio, la conversión de pulsos a mm/s no consideraba correctamente los pulsos por vuelta, lo que producía valores exagerados, por lo tanto, Se recalculó la relación pulsos → rad/s → mm/s utilizando el número exacto de pulsos por vuelta y el diámetro real de la rueda, eliminando lecturas incorrectas.

Por otra parte, se registraron datos incoherentes en la gráfica, Al iniciar las primeras pruebas, la gráfica mostraba valores fuera de escala o picos irregulares que no coincidían con el comportamiento real del motor. Para reducir el ruido, se aplicó un filtro FIR simple que suavizó la señal de velocidad sin afectar el tiempo de respuesta.

CONCLUSIONES

Se encontró que la conexión de los LED's rojos de chorro no fue exitosa debido al ruido electromagnético y malas conexiones en la placa PCB, entonces se decidió conectar unos leds IR para mejorar la precisión en la medida y además asegurar una buena conexión en una nueva placa realizada.

Para la calibración de los ADC se utilizó una pared blanca a modo de simulación del laberinto, sin embargo, se presentaron fallas en los receptores debido a la falta de control desde el software. Para futuras referencias, se confirmó que las pruebas de los ADC sobre todo deben realizarse como parte esencial del desarrollo del proyecto. Debido a la complejidad de la parte analógica.

Referencias

- [1] R. Magalhães, “Control de velocidad y dirección del motor de CC con microcontrolador STM32F103 y controlador de motor L293D,” *COMPRACO Indústria e Construção*, Jun. 14, 2024. [En línea]. Disponible en: <https://compraco.com.br/es/blogs/tecnologia-e-desenvolvimento/controle-de-velocidade-e-direcao-do-motor-dc-com-microcontrolador-stm32f103-e-driver-de-motor-l293d?srltid=AfmBOoq7tGrIQo0YKO4YZsy7YQ5pAvbf2K4TYqgGLvpxBNC5YWeSVLKN>
- [2] T. E. y-desenvolvimento, “Setting Up – Qt Plotting Widget QCustomPlot,” *QCustomPlot*, 2025. [En línea]. Disponible en: <https://www.qcustomplot.com/index.php/tutorials/settingup>
- [3] R. Lozano, “¿Cómo medir RPM encoder 400 pulsos?,” *Talos Electronics Tutoriales*, 22 Ene. 2023. [En línea]. Disponible en: <https://www.taloselectronics.com/blogs/tutoriales/como-medir-rpm-encoder-400-pulsos>

ANEXOS