Informe laboratorio #9

Daniel Avila

Yeison Lara

Cristian Mora

¿Que se usó?

- Robot móvil ensamblado con sus motores y sensores
- Cable delgado para conexión UART
- Programador STLink
- Teleplot o Qt para visualización

¿Que se realizó?

Primero diseñamos e implementamos un perfil de velocidad trapezoidal dinámico que permite especificar la velocidad inicial, la velocidad máxima y la aceleración, tanto para desplazamientos lineales como para giros sobre el eje. A continuación, creamos la librería `PID_MM.c`, que incorpora dos controladores PID independientes: uno lineal y otro angular. Cada controlador se inicializa con sus propias constantes `Kp` y `Kd`, lo que facilita la calibración específica para cada tipo de movimiento; además, el PID angular incluye un límite en la salida de control para evitar saturaciones.

En la rutina principal, el sistema ejecuta en un bucle una secuencia de al menos cinco movimientos configurados en cola, activada con un botón físico. En cada iteración, se calcula la velocidad de referencia mediante la función `trapecio_perpetuo`, y luego se evalúa el error entre esta referencia y la velocidad real de cada motor para generar las señales de control PID (una para el motor izquierdo y otra para el derecho). Estas señales se envían a los motores, logrando un seguimiento muy cercano de la velocidad deseada. Simultáneamente, cada 0.005 s el robot transmite por UART —a 921600 bps— tanto la velocidad deseada como la real y los valores de salida del PID, que Teleplot grafica en tiempo real.

Durante la prueba, al presionar el botón de inicio el robot avanzó de acuerdo con el perfil trapezoidal, corrigió con suavidad cualquier desviación usando el PID lineal y, cuando correspondía, giró sobre sí mismo empleando el PID angular con su clamping de salida. El resultado fue un seguimiento estable y preciso de los movimientos programados: la velocidad real casi coincidía con la deseada, y las correcciones del PID disminuían rápidamente una vez alcanzado el régimen. Al completar los cinco movimientos, el

sistema regresó de forma automática a estado de espera, sin necesidad de reinicios manuales.

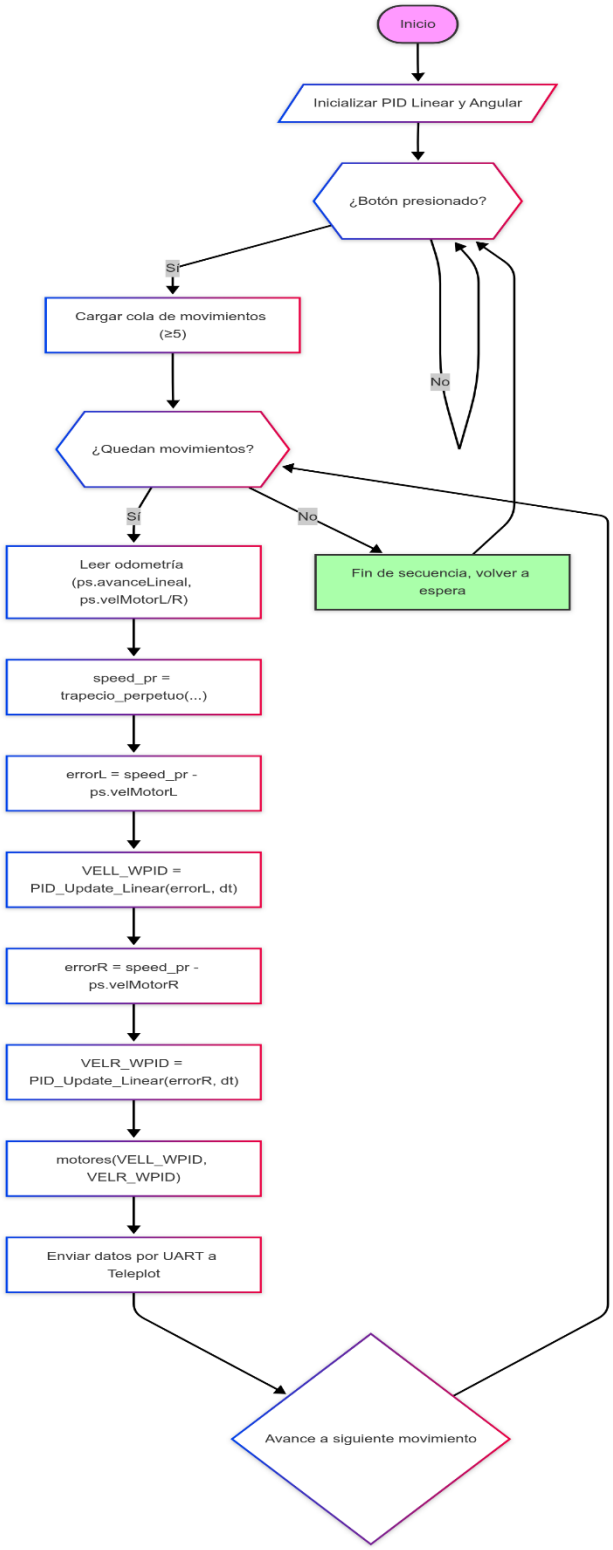
¿Qué sucedió?

Al presionar el botón de inicio, el robot ejecutó la cola de movimientos: avanzó en línea recta, luego giró sobre su eje según los ángulos programados, y así sucesivamente hasta completar los cinco movimientos. Gracias al control PID:

- La **velocidad real** de cada motor siguió muy de cerca la **referencia trapezoidal**, reduciendo oscilaciones.
- Durante los giros, el **clamping** en el PID angular evitó que la señal saturara los motores, logrando giros suaves y precisos.
- El sistema completó el ciclo sin necesidad de reinicios y volvió a modo espera automáticamente.

En **Teleplot** pudimos ver gráficas en tiempo real que confirmaron el buen desempeño: la línea de velocidad real prácticamente se superponía a la deseada, y las correcciones PID eran mínimas una vez alcanzado el régimen.

Diagrama de flujo:



Conclusión:

En este laboratorio se logró construir una **librería modular de control PID** que controla de forma independiente la velocidad lineal y angular del robot, integrando perfiles trapezoidales dinámicos. La **separación de controladores** (PID lineal y PID angular) permitió ajustar finamente cada comportamiento, mejorar la precisión y evitar saturación del actuador en giros. La **visualización en tiempo real con Teleplot** dio evidencia gráfica de la eficacia del control, mostrando un seguimiento casi perfecto de las referencias y correcciones mínimas. El sistema completó los cinco movimientos programados sin fallos y retornó a estado de espera automáticamente, demostrando estabilidad y robustez en la implementación.