0 Controlador Proporcional Integral Derivado (PID)

Clubes de Ciencia y Tecnología

Nombre:_____

Objetivo: Aplicar el control proporcional integral derivativo (PID) para el seguimiento de línea, con entusiasmo y perseverancia.

Propósito: Para aumentar la velocidad y mejorar los resultados del seguimiento de línea del Hermes.

Ahora que ya tenemos nuestro controlador proporcional definido, hablemos de la integral.

Lo primero es saber que no nos vamos a enfocar en la definición más matemática del concepto, sino que quedémonos con la idea de que la integral va a ir almacenando los errores que vamos leyendo de manera continua, quedando algo así.

```
integral = integral + error
```

¿Y de qué nos sirve esto? Bueno, por un lado, si el error se mantiene del mismo signo por mucho rato (la línea se queda constantemente de un lado del robot), la integral va a seguir sumando, lo que en la práctica nos va a dar una correción de error más agresiva para contrarrestar este efecto y ayudar a seguir la línea. Por otro lado, cómo va sumando constantemente, nos va a ayudar a corregir errores pequeños que de otra forma no influiría en el cálculo del giro. Entonces, añadiendolo al cálculo del giro se tiene algo así:

```
giro = Kp * error + Ki * integral
```

Donde se añade la integral con su respectiva constante **Ki,** la cual usualmente se inicializa en 0.1, y luego se va ajustando según las necesidades que se vayan viendo.

Ahora con esto claro, podemos modificar el algoritmo proporcional simple que ya teníamos, quedando así:

```
//Inicializar variables
Kp = 0,39 //Constante Proporcional
Ki = 0,1    //Constante Integral
ref = 0  //Este es el valor al que queremos llegar
Tp = 100  //Velocidad base
integral = 0 //Se inicia la variable donde se almacenará la integral

Loop:
      posicion = lecturaSensores()
      error = posicion-ref
      integral = integral + error
      giro = Kp * error + Ki * integral
      velocidadIzq = Tp + giro
      velocidadDer = Tp - giro
      moverMotores(velIzq, velDer)
```

Llegados a este punto tenemos el proporcional que corrige el error actual y la integral que trata de corregir los errores pasados, entonces, ¿Habrá una forma de mirar hacia el futuro y tratar de predecir el error que viene?

La respuesta es sí, y aquí es donde entra el concepto de la derivada, la cual viene del mundo de las matemáticas avanzadas pero para suerte nuestra, lo que vamos a utilizar es bastante simple.

Podemos mirar hacia el futuro asumiendo que el próximo cambio en el error es el mismo que el último cambio en el error, lo que significa que se espera que el siguiente error va a ser igual a la diferencia de los 2 errores anteriores. Esta diferencia es lo que se conoce como la derivada y se puede pensar como la pendiente de una recta en un punto específico.

Aterrizando la idea, se tiene lo siguiente:

```
derivada = error - lastError
```

Y al igual que con la integral, este error futuro tiene su propia constante, la cual llamaremos Kd, y en este caso se inicializa en 0.01 (es un valor arbitrario y se debe ir ajustando), ahora, teniendo esto, el cálculo del giro queda así:

```
giro = Kp * error + Ki * integral + Kd * derivada
```

Por último, al añadirlo al algoritmo nos queda algo así:

```
//Inicializar variables
Kp = 0,39 //Constante Proporcional
Ki = 0,1    //Constante Integral
Kd = 0,01  //Constante Derivada
ref = 0  //Este es el valor al que queremos llegar
Tp = 100  //Velocidad base
error = 0 //Se inicia la variable donde se almacenará el error
integral = 0 //Se inicia la variable donde se almacenará la integral
lastError = 0 //Se inicia la variable para el último error
derivada = 0 //Se inicia la variable donde se almacenará la derivada

Loop:
      posicion = lecturaSensores()
      error = posicion-ref
      integral = integral + error
      derivada = error  - lastError
      giro = Kp * error + Ki * integral + Kd * derivada
      velocidadIzq = Tp + giro
      velocidadDer = Tp - giro
      moverMotores(velIzq, velDer)
      lastError = error  //Se guarda el ultimo error
```