

IMPLEMENTACIÓN DE ALGORITMOS PID

Rubén Espino San José

PUMA



PRIDE

Índice

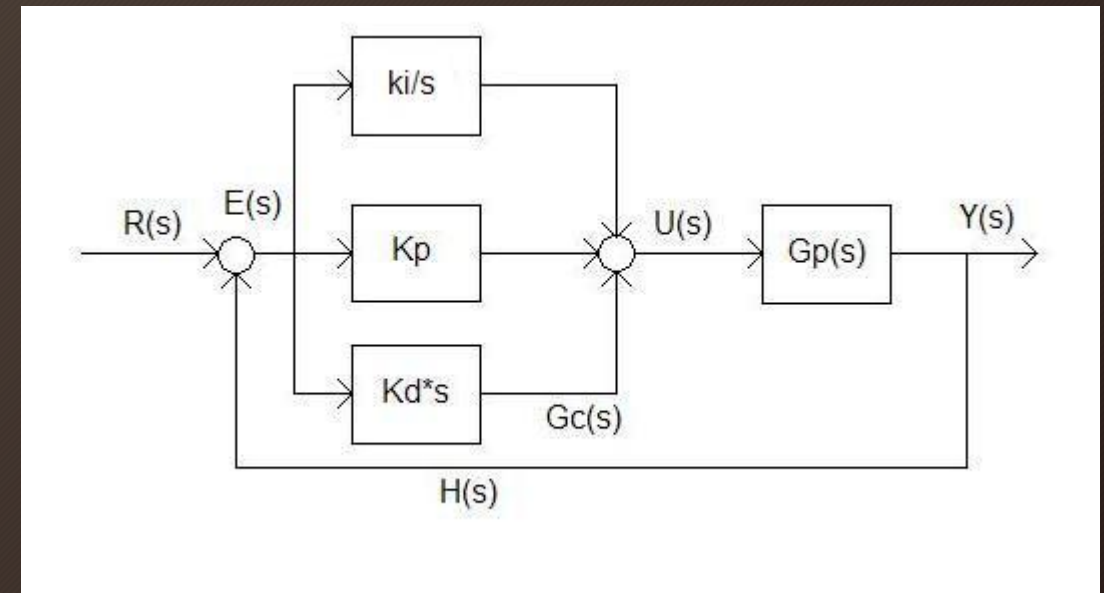
2

- ¿Qué es un controlador PID?
- ¿Cómo funciona un PID?
- Ejemplos de aplicación
- Apéndices
- Referencias

¿QUÉ ES UN CONTROLADOR PID?

3

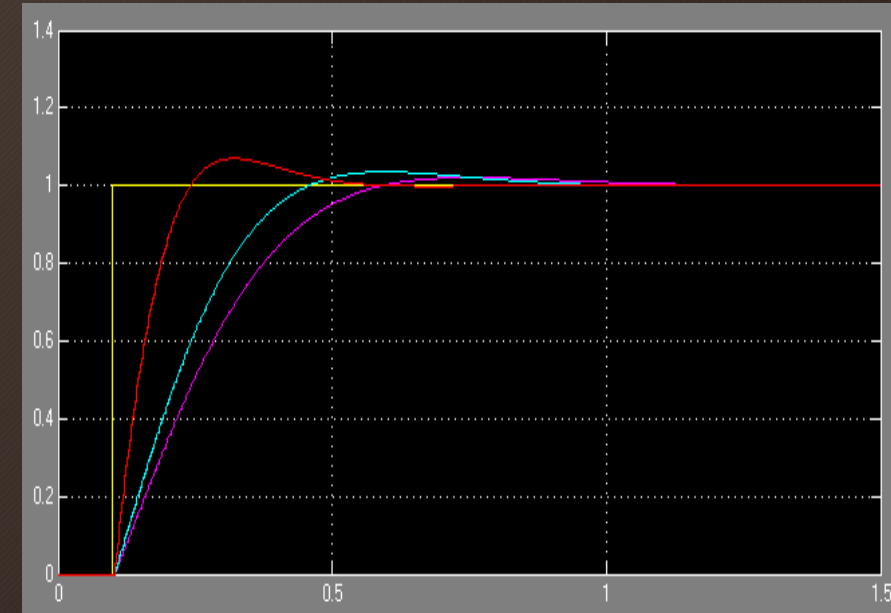
- Algoritmo que se emplea para contrarrestar los efectos de las perturbaciones en un sistema lineal
- Compuesto de las siguientes partes:
 - Proporcional
 - Detecta el error proporcional
 - Corrección de posición
 - Integral
 - Detecta el error acumulado
 - Oposición a las perturbaciones
 - Derivativo
 - Detecta la variación del error proporcional
 - Corrección de velocidad



¿CÓMO FUNCIONA UN PID?

4

- Proporcional = posición_objetivo - posición_actual
- Integral = integral + proporcional
 - Saturar integral para no hacer inestable el algoritmo
- Derivativo = proporcional - proporcional_anterior
 - Actualizar proporcional_anterior = proporcional
- Error = $k_p * \text{proporcional} + k_i * \text{integral} + k_d * \text{derivativo}$



EJEMPLOS DE APLICACIÓN

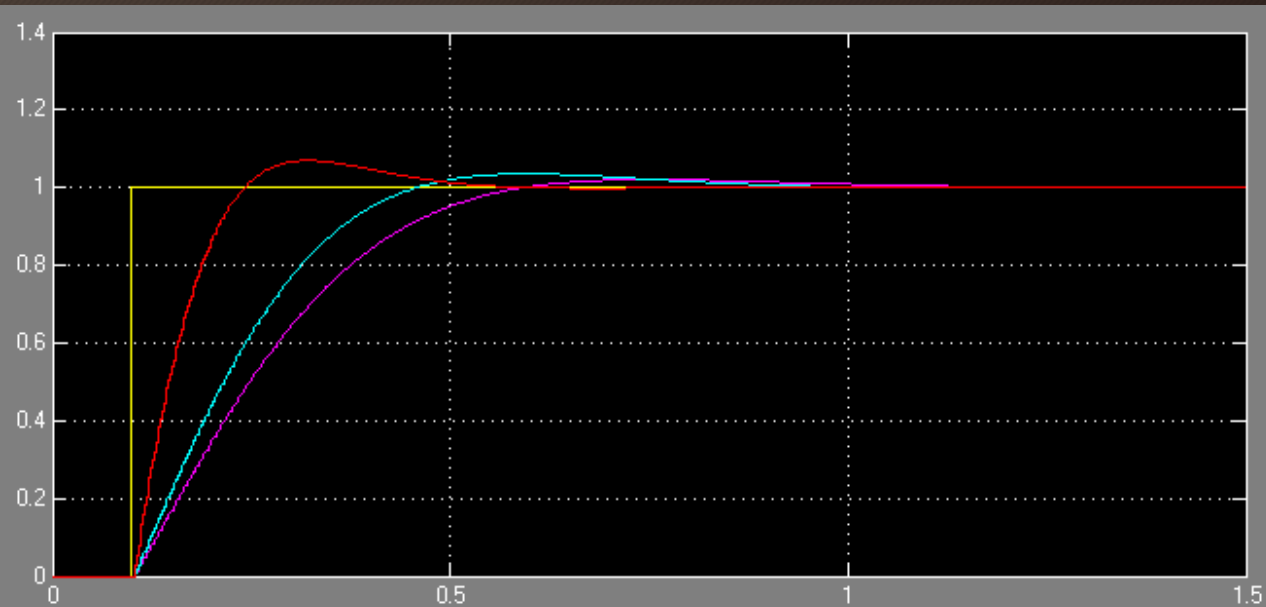
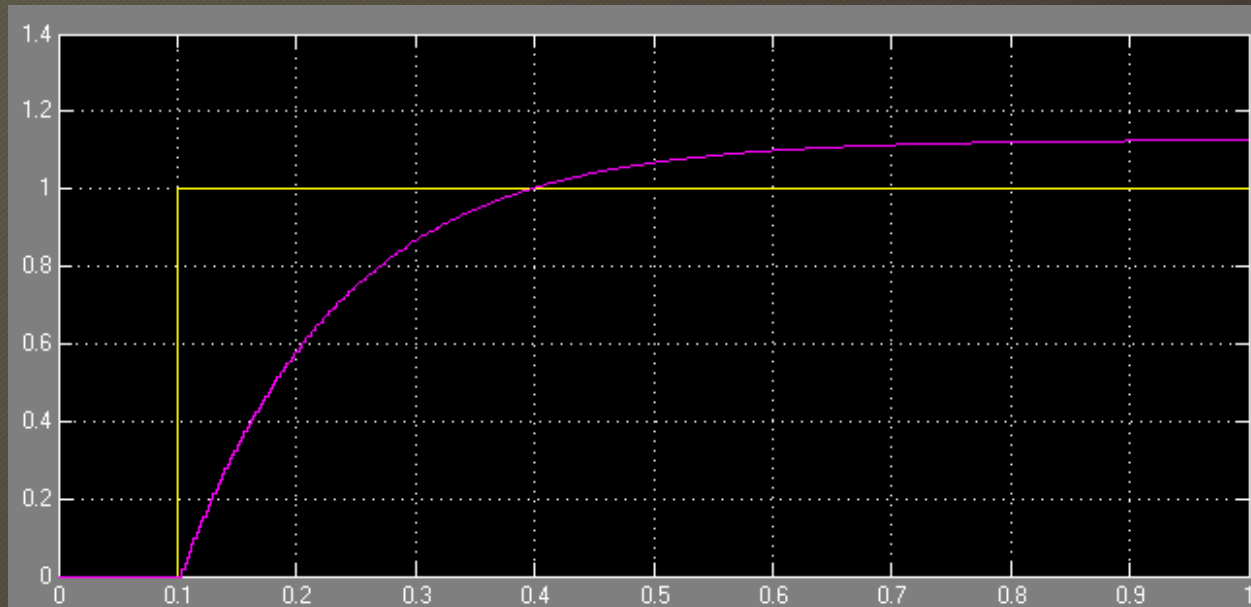
5

- Control de motores
- Robots siguelíneas
 - Seguimiento de líneas
 - Rebufo en carreras
 - Control de velocidad lineal
- Drones
 - Control de estabilidad
 - Control de orientación
 - Control de posición
 - Control de altura

CONTROL DE MOTORES

6

- Control P de velocidad de un motor con lazo abierto (sin realimentación)
 - Puede no alcanzar la posición objetivo
- Control PI de velocidad de un motor con lazo cerrado (con realimentación)
 - Alcanza la posición objetivo



ROBOTS SIGUELÍNEAS: SEGUIMIENTO DE LÍNEAS

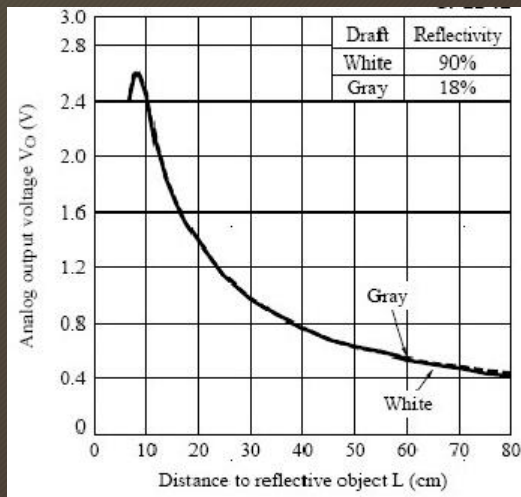
7

- Se trata de un control de velocidad angular
$$\text{velocidad_motor_izquierdo} = \text{velocidad_lineal} + \text{error}$$
$$\text{velocidad_motor_derecho} = \text{velocidad_lineal} - \text{error}$$
- [Demostración de corrección en estático](#)
- [Demostración con PID sobreamortiguado](#)
- [Demostración con PID subamortiguado](#)
- [Demostración con PID con amortiguamiento crítico](#)
- [Mi primer PID en un robot rastreador](#)
- [Desarrollo detallado del PID para el seguimiento de línea](#)

ROBOTS SIGUELÍNEAS: REBUFO EN CARRERAS

8

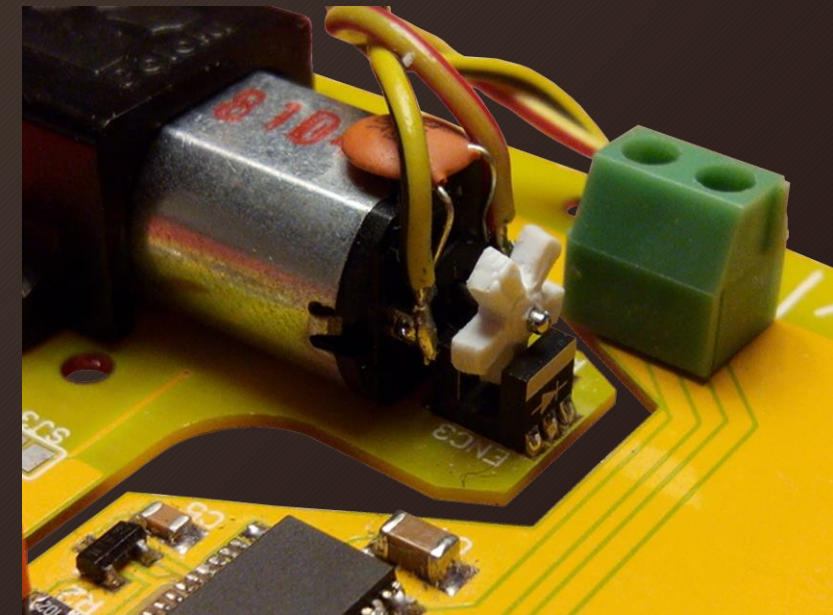
- Para seguir al oponente es necesario incorporar un sensor de distancia frontal analógico
 - [Pumatrón cogiendo el rebufo](#)
- Si la función de transferencia del sensor analógico no es lineal, hay que linealizarla
 - [Linealización de los sensores GP2D120 y GP2Y0A21](#)



ROBOTS SIGUELÍNEAS: CONTROL DE VELOCIDAD LINEAL

9

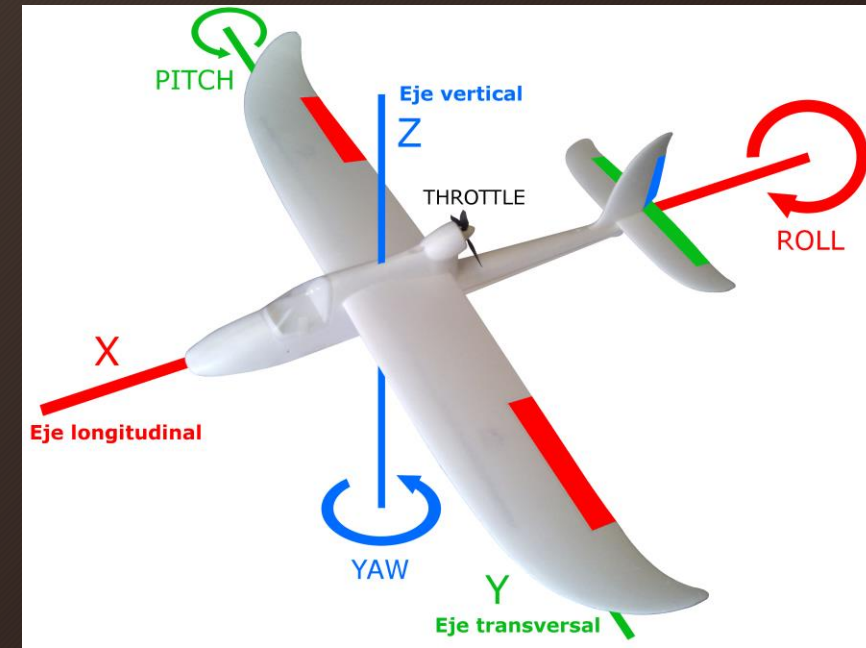
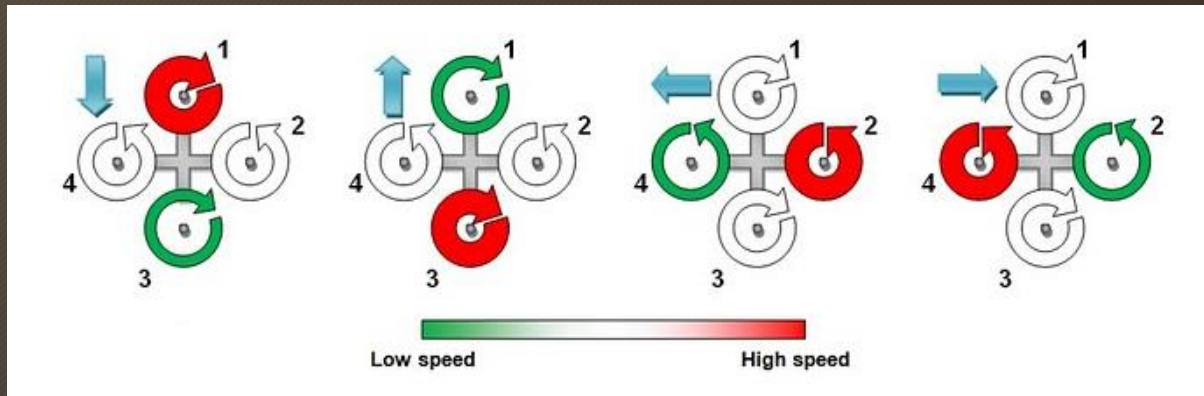
- Se incorpora un encoder en cada motor para leer la velocidad y cerrar el lazo de realimentación
- Se emplea para que las perturbaciones como los desniveles en la pista o el descenso de tensión de la batería afecten lo mínimo posible a la velocidad del robot
- $\text{Velocidad_lineal} = (\text{V_motor_izq} + \text{V_motor_der}) / 2$



DRONES: CONTROL DE ESTABILIDAD

10

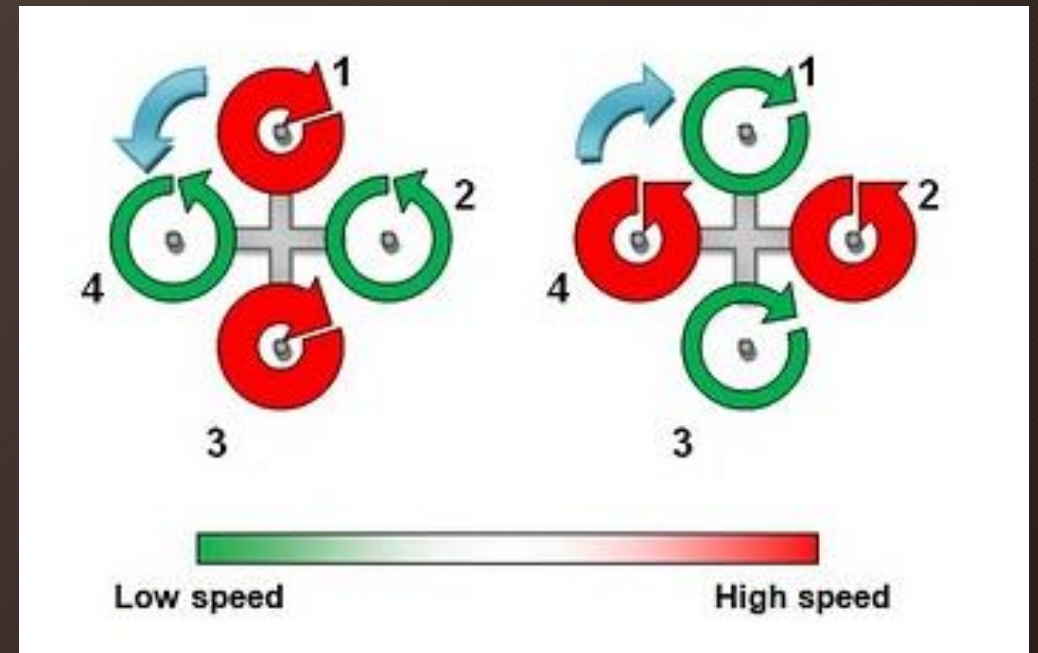
- Control de inclinación en *pitch* y *roll* con acelerómetro y giróscopo. Un PID para cada eje
- Acelerómetro: funciona como inclinómetro. Señala la dirección de la fuerza gravitatoria
- Giróscopo: da la velocidad de giro
- Se combinan para eliminar la deriva del giróscopo
- [Demostración de control de estabilidad](#)



DRONES: CONTROL DE ORIENTACIÓN

11

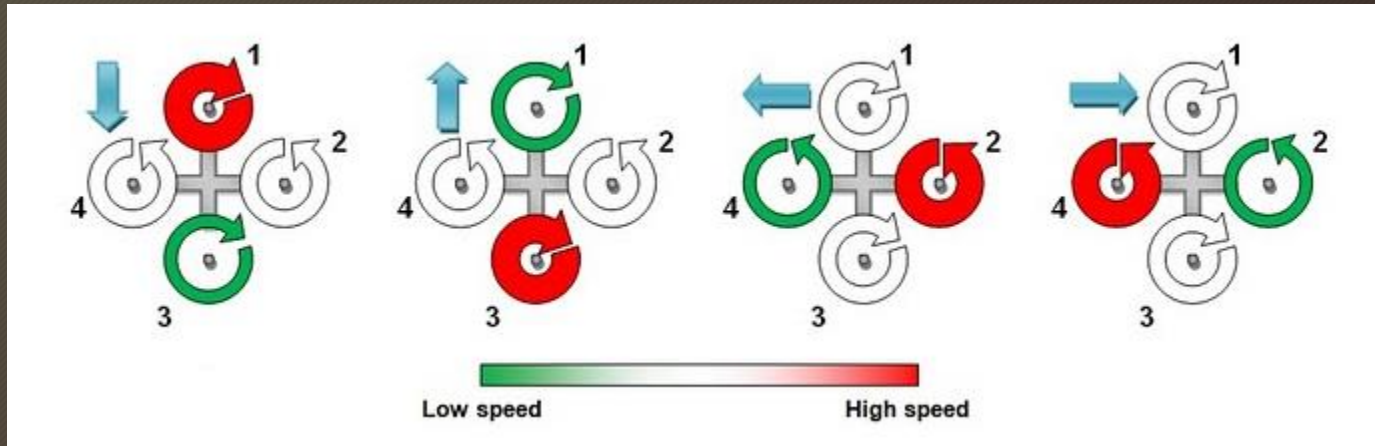
- Control de orientación (yaw) con brújula y giróscopo
- El giróscopo es más rápido calculando la orientación, pero la brújula elimina el error de deriva del giróscopo
- Similar al PID de seguimiento de líneas



DRONES: CONTROL DE POSICIÓN

12

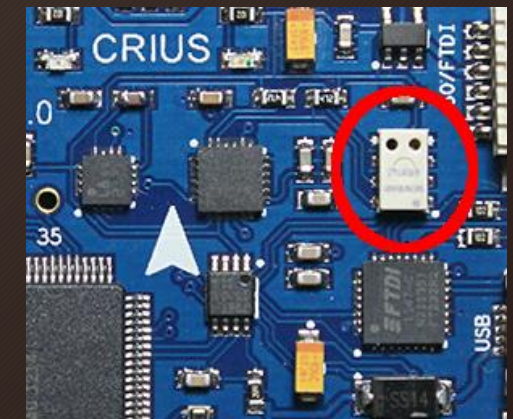
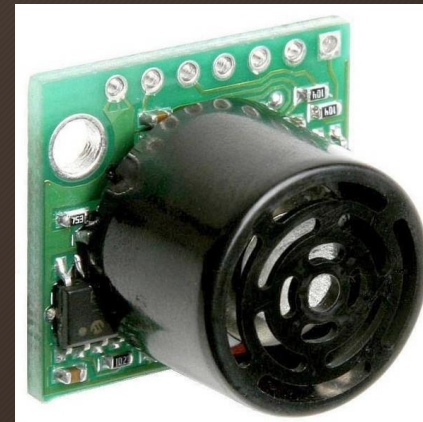
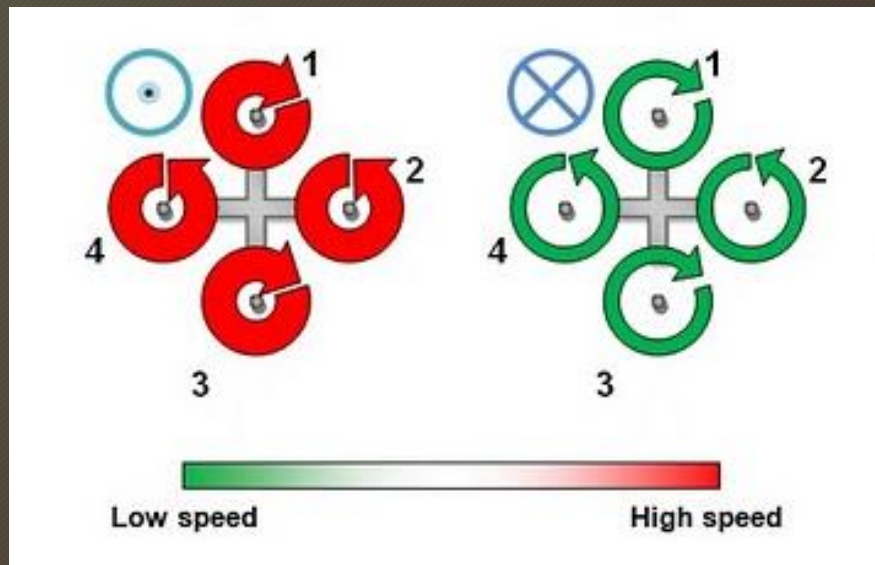
- Control de posición con GPS o con visión artificial. Un PID para cada eje del plano XY
- Eliminan la deriva en la posición del dron
- [Demostración de control de posición con visión artificial](#)



DRONES: CONTROL DE ALTURA

13

- Con barómetro: PID para el control de presión
- Con sensor de ultrasonidos: PID para el control de distancia al suelo
- [Demostración de control de altura con sensor de ultrasonidos](#)



- Cálculo de posición de línea para robots siguelíneas:

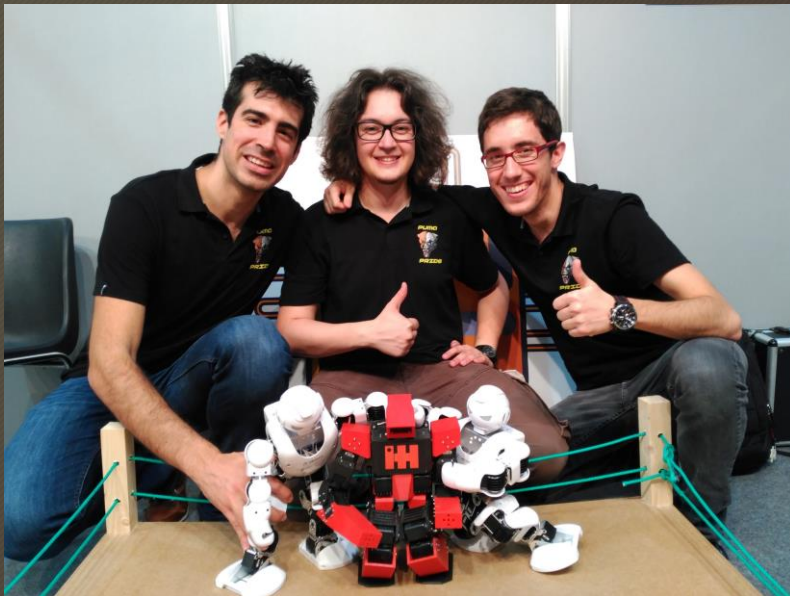
```
for(i = 0; i < N_sensores; i++)  
{  
    media += valor_sensor[i] * (i+1) * 1000;  
    suma += valor_sensor[i];  
}  
posicion = media / suma; // Posición con el cero en el extremo  
posicion_linea = posicion - (N_sensores + 1)*1000/2; // Para centrar la posición
```


- Pasos para calibrar un PID manualmente:
 1. Poner todas las K's a cero
 2. Ir aumentando poco a poco K_p
 3. Cuando el robot empiece a cabecear, bajar un poco el valor de K_p y dejarlo fijo
 4. Realizar los pasos 2 y 3 para calibrar K_d
- Poniendo como ejemplo un robot siguelíneas, la respuesta varía si se modifica su velocidad lineal, por lo que habrá que realizar el cálculo de las K's para cada velocidad

REFERENCIAS

16

- GitHub
 - Rubén Espino: Resaj
 - Javier Baliñas: supernudo
 - Javier Isabel: JavierIH



- Facebook
 - @pumaprideteam
- Twitter
 - Rubén Espino: @RugidoDePuma
 - Javier Baliñas: @supernudo
 - Javier Isabel: @JavierIH
- Correo
 - puma.pride@arc-robots.org

¡¡Que los PIDs os acompañen!!