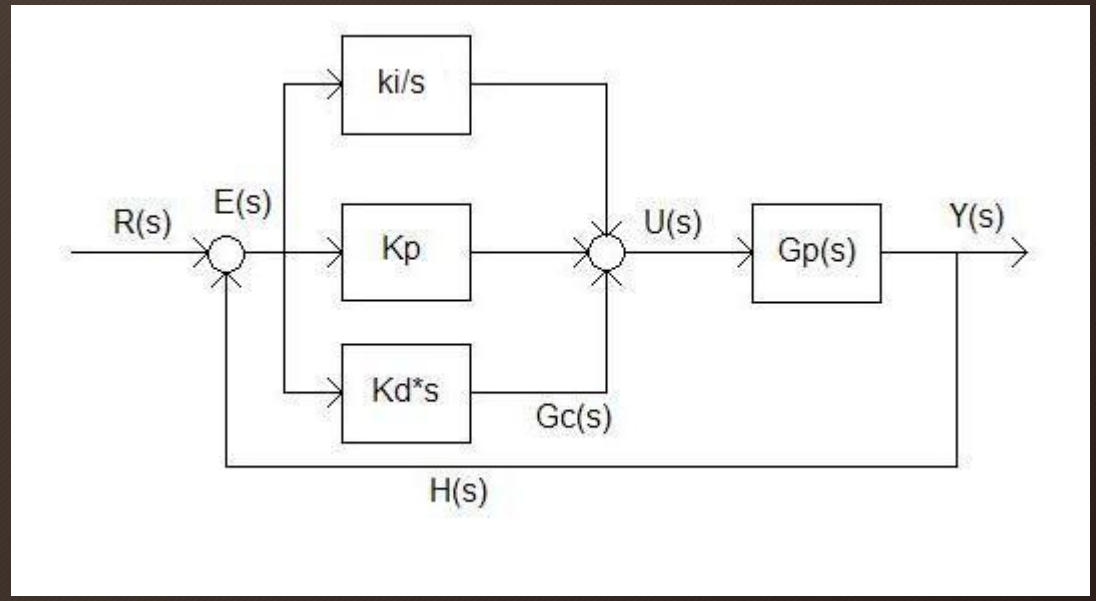
IMPLEMENTACIÓN DE ALGORITMOS PID

Rubén Espino San José

Índice

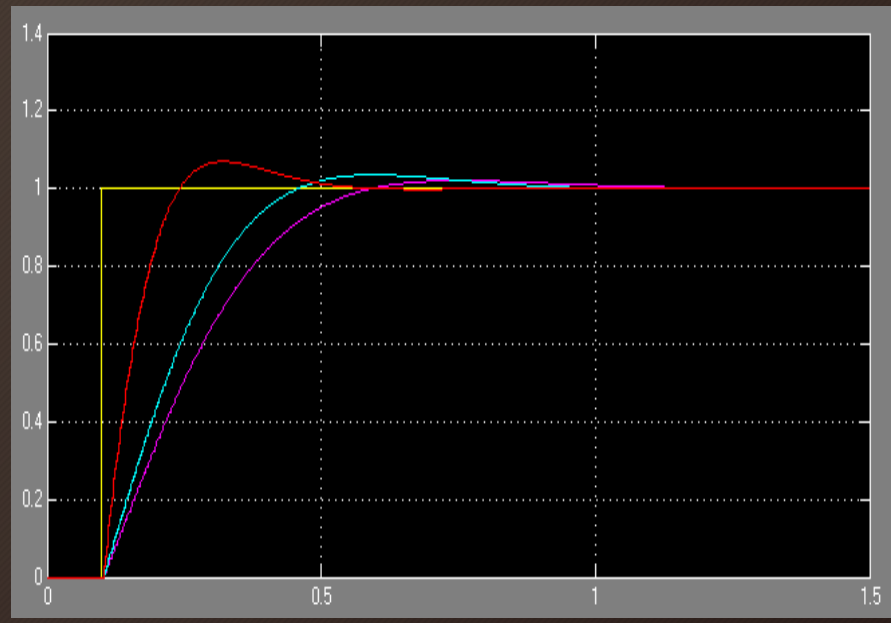
¿QUÉ ES UN CONTROLADOR PID?

- 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?

- Proporcional = posición_objetivo - posición_actual
- Integral = integral + proporcional
  - Saturar integral para no hacer inestable el algoritmo
- Derivativo = proporcional - proporcional_anterior
  - Actualizar proporcional_anterior = proporcional



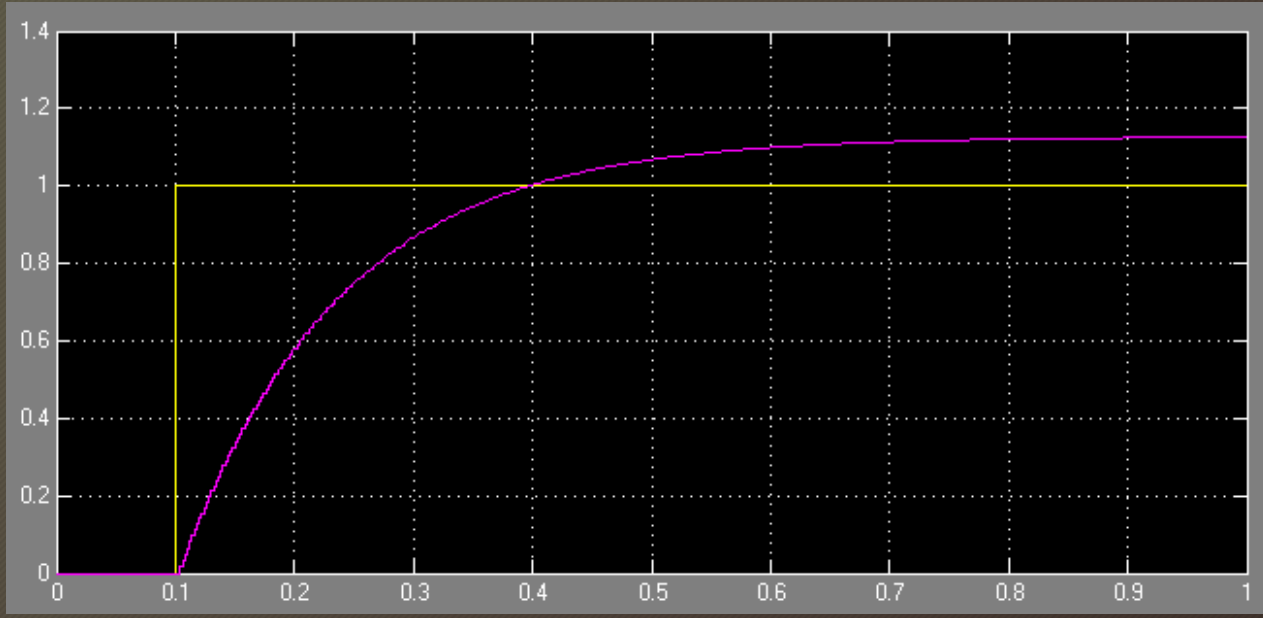
- Error = kp * proporcional + ki * integral + kd * derivativo

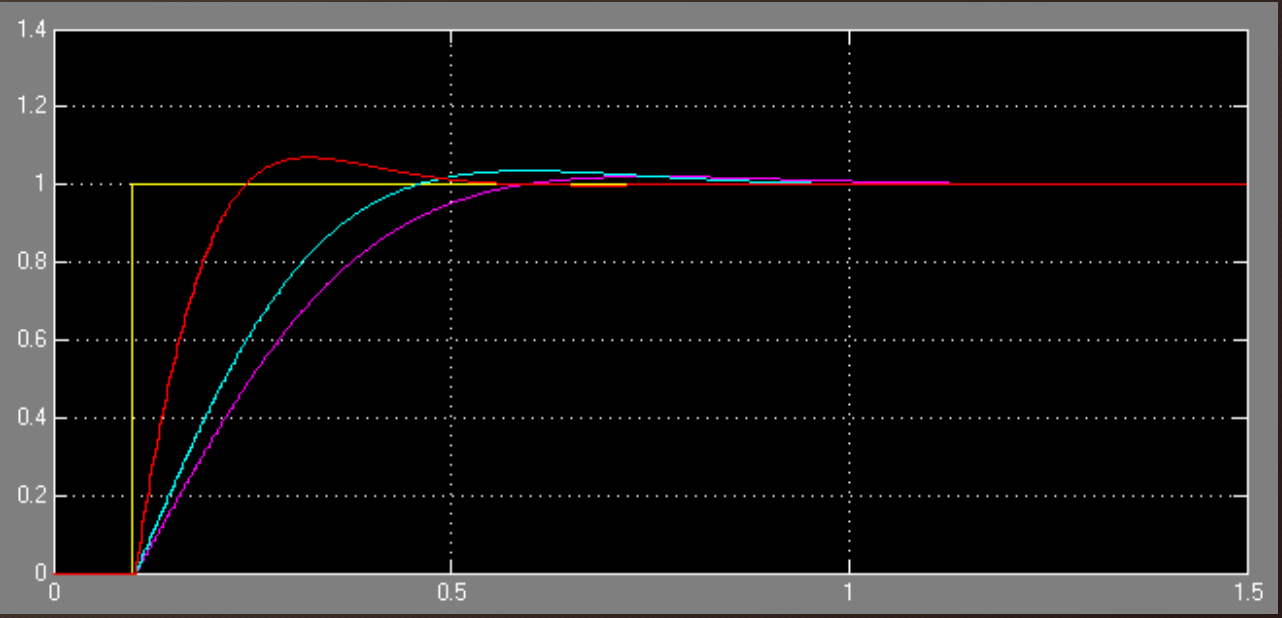
EJEMPLOS DE APLICACIÓN

- 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

- 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

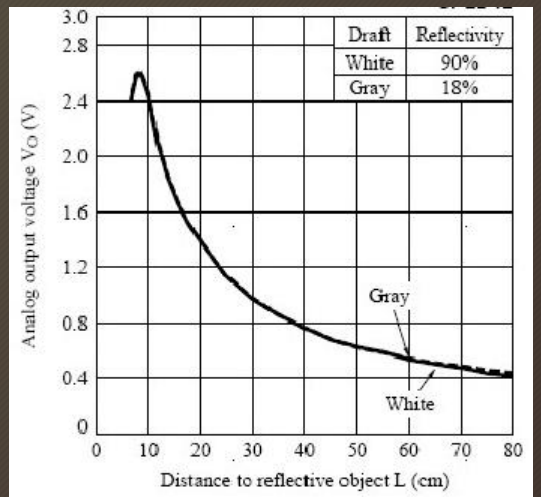
- Se trata de un control de velocidad angular

  velocidad_motor_izquierdo = velocidad_lineal + error

  velocidad_motor_derecho = velocidad_lineal - 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

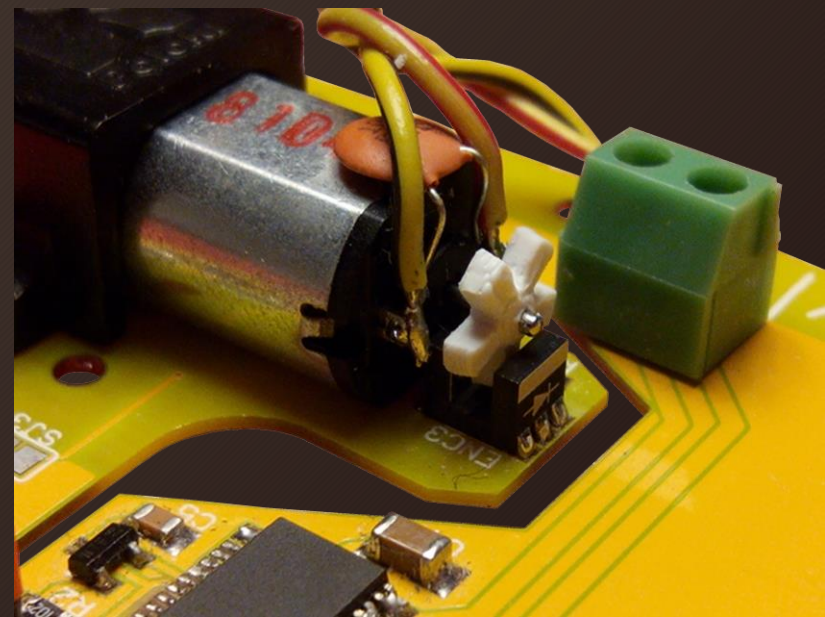
ROBOTS SIGUELÍNEAS: REBUFO EN CARRERAS

- 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



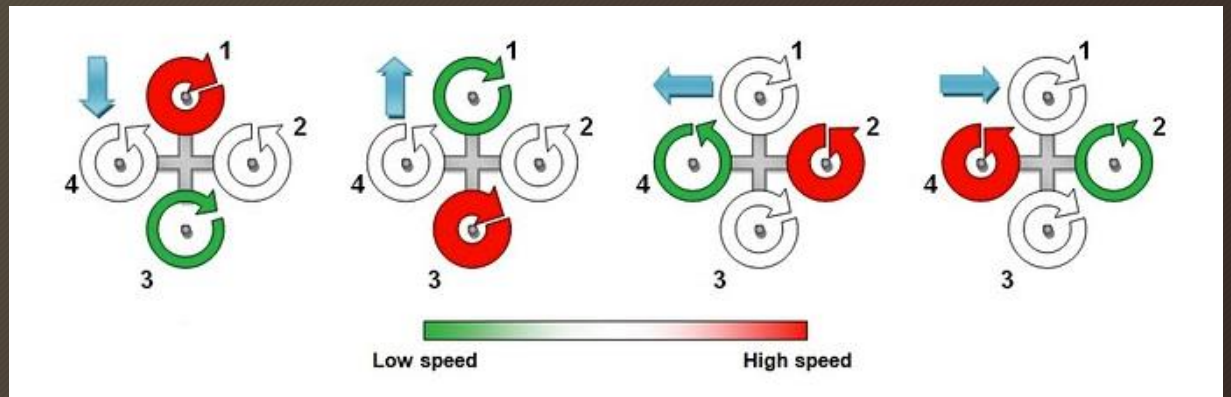
ROBOTS SIGUELÍNEAS: CONTROL DE VELOCIDAD LINEAL

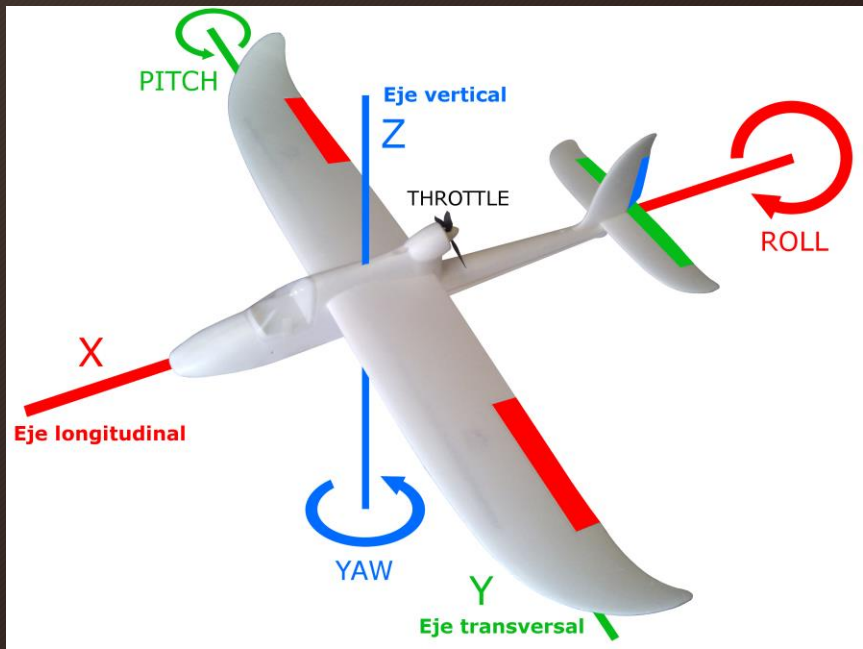
- Se incorpora un encoder en cada motor para leer la velocidad y cerrar el lazo de realimentación
- Se emplea para que las perturbaciones como los desniveles en la pista o el descenso de tensión de la batería afecten lo mínimo posible a la velocidad del robot
- Velocidad_lineal = (V_motor_izq + V_motor_der)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



DRONES: CONTROL DE ESTABILIDAD

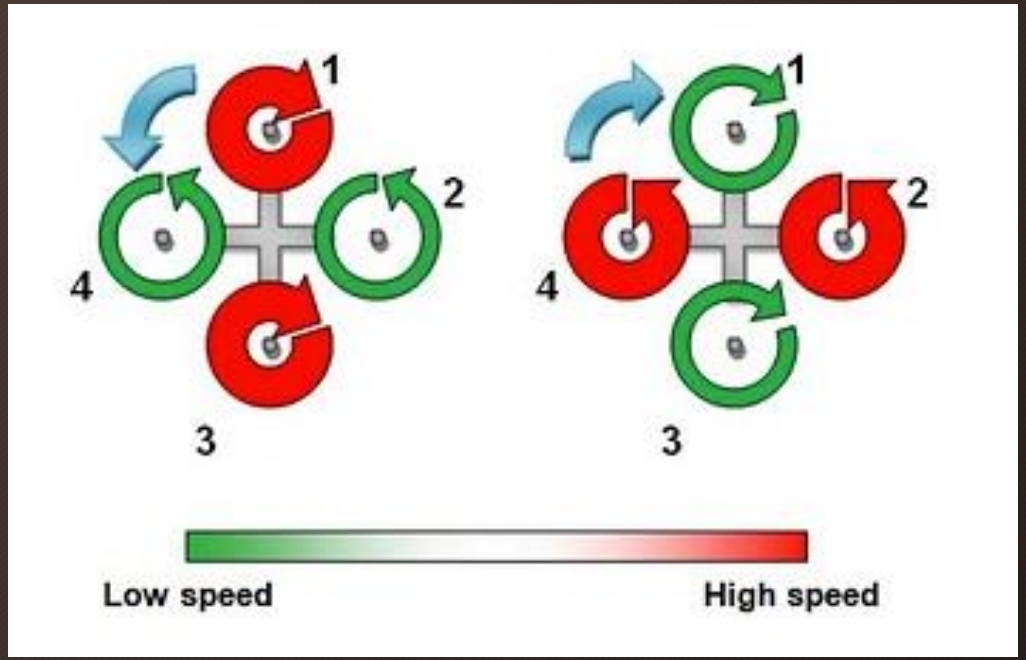
- Control de inclinación en *pitch* y *roll* con acelerómetro y giróscopo. Un PID para cada eje
- Acelerómetro: funciona como inclinómetro. Señala la dirección de la fuerza gravitatoria
- Giróscopo: da la velocidad de giro
- Se combinan para eliminar la deriva del giróscopo
- Demostración de control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





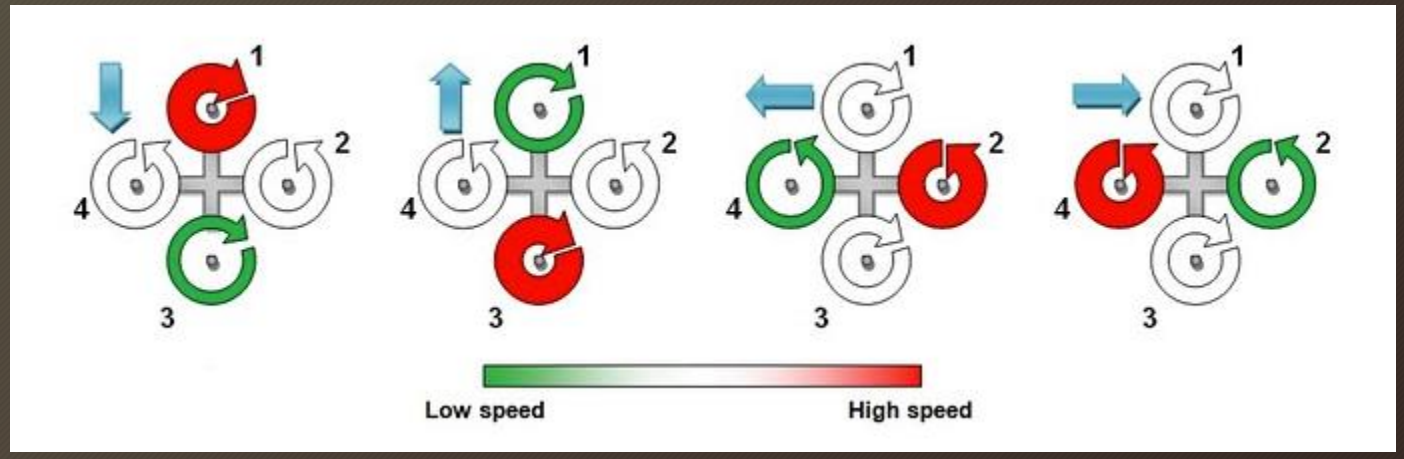
DRONES: CONTROL DE ORIENTACIÓN

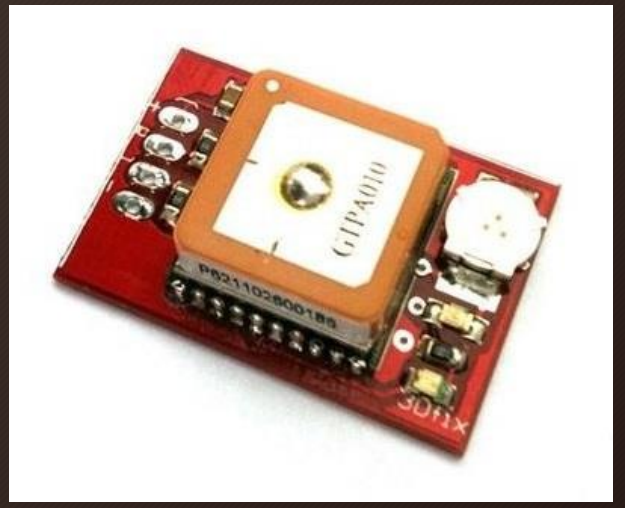
- Control de orientación (*yaw*) con brújula y giróscopo
- El giróscopo es más rápido calculando la orientación, pero la brújula elimina el error de deriva del giróscopo
- Similar al PID de seguimiento de líneas



DRONES: CONTROL DE POSICIÓN

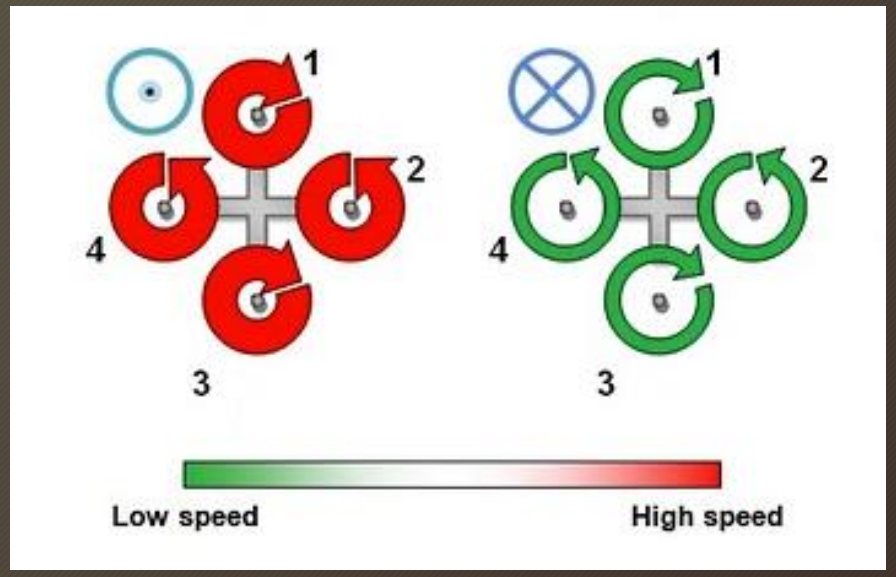
- Control de posición con GPS o con visión artificial. Un PID para cada eje del plano XY
- Eliminan la deriva en la posición del dron
- Demostración de control de posición con visión artificial

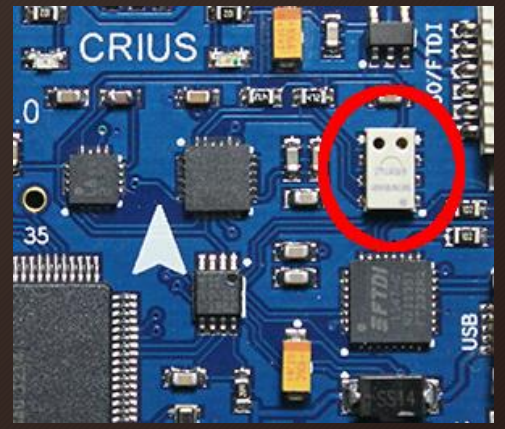




DRONES: CONTROL DE ALTURA

- 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





APÉNDICES

- 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
```

APÉNDICES

- Pasos para calibrar un PID manualmente:
  1. Poner todas las K's a cero
  2. Ir aumentando poco a poco Kp
  3. Cuando el robot empiece a cabecear, bajar un poco el valor de Kp y dejarlo fijo
  4. Realizar los pasos 2 y 3 para calibrar Kd
- 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

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

PUMA

PRIDE

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

¡¡Que los PIDs os acompañen!!