



What is PID controls

Explorers FLL Team

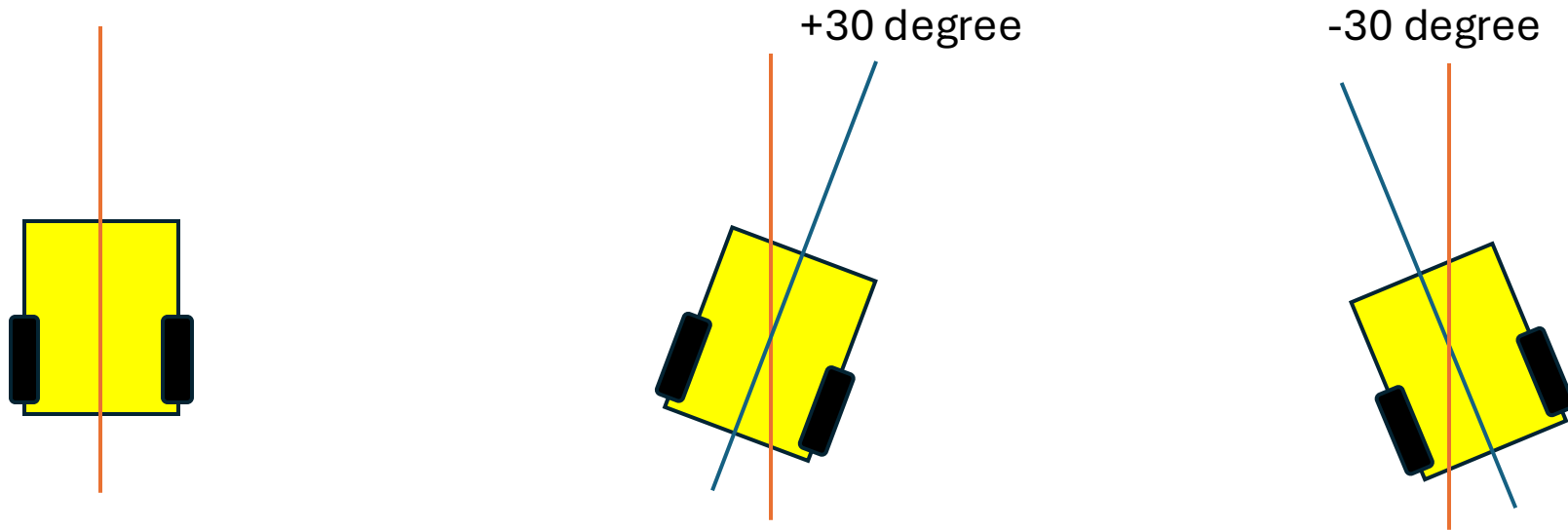
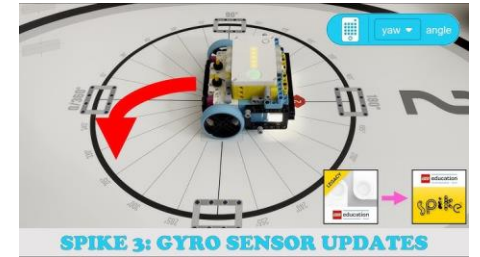
2025/12/28

About the Tutorials

- In FIRST LEGO League, small errors—like drifting during straight driving or overshooting a turn—can cause big point losses. Through studying and applying PID control, our team learned how to let the robot **automatically correct its motion** using sensor feedback (such as a gyro sensor).
- This tutorial is designed for:
 - Team members learning advanced robot control
 - New students who want to understand *why* the robot behaves better
 - Judges who want to see our **engineering learning process**, not just final results
- The goal of this document is not only to explain PID theory, but also to provide example to hand calculate the PID values.

PID controller (Use go straight as example)

- $\text{Error} = \text{desired_value} - \text{measured_value}$



How make it go straight:

Left Speed

+0

Right Speed

+0

Left Speed

-

Right Speed

+

Left Speed

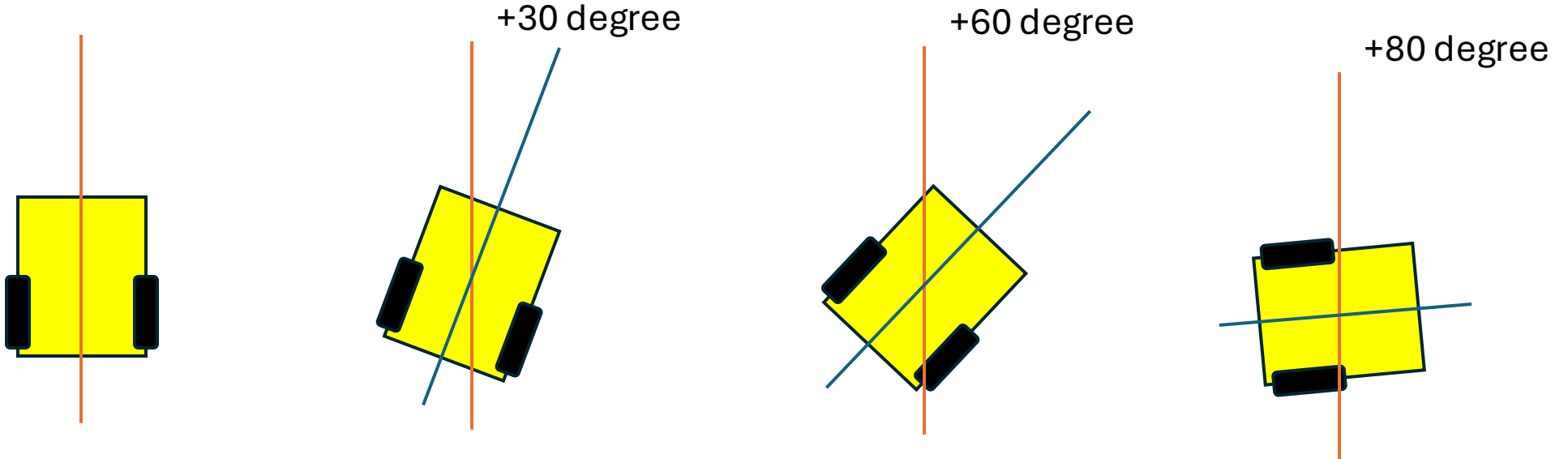
+

Right Speed

-

What is P (Proportional)

Base Speed = 200



Left Speed 200

200-**30**

200-**60**

200-**80**

Right Speed 200

200+**30**

200+**60**

200+**80**

What are P, I, D

Term	Name	Description
P	Proportional	Reacts to the current error . The larger the error, the stronger the correction.
I	Integral	Reacts to the accumulated error over time . Helps eliminate long-term drift.
D	Derivative	Reacts to the rate of change of the error . Helps predict and dampen overshoot.

Previous_error=5

Current_error = 10

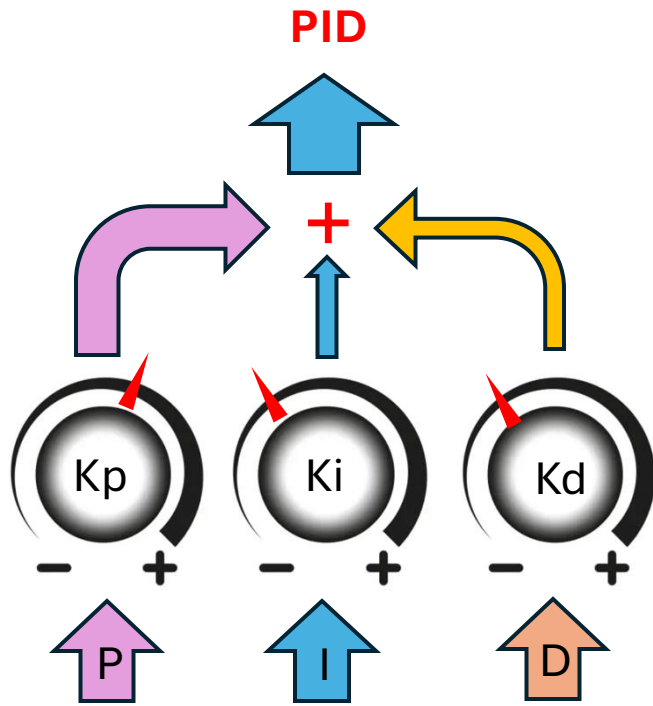
$P = \text{current_error}$

$I = \text{Sum(All errors)}$

$D = \text{current_error} - \text{previous_error}$

What is PID ?

$$\text{PID} = P * K_p + I * K_i + D * K_d$$



Example

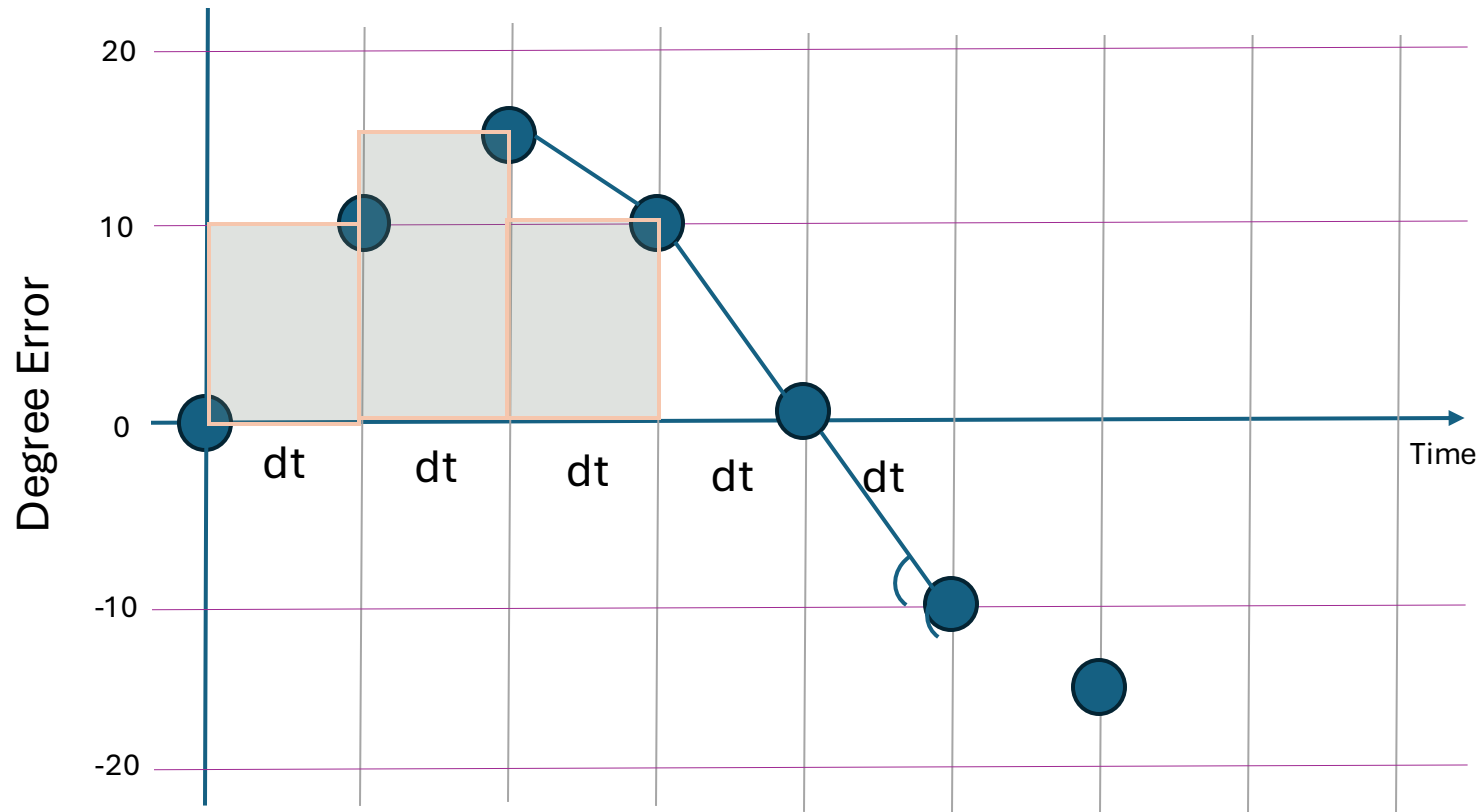
Time index	Error	P	I	D	PID
0	0				
1	10				
2	15				
3	10				
4	0				
5	-10				
6	-15				

$K_p=1$

$K_i=0.1$

$K_d=0.1$

More accurate PID (2) ?



Example

Time index	Error	P	I	D	PID
0	0	0	0	0	0
1	10	10	10	10	12
2	15	15	25	5	18
3	10	10	35	-5	13
4	0	0	35	-10	2.5
5	-10	-10	25	-10	-8.5
6	-15	-15	10	-5	-14.5

More accurate formula

$P = \text{current_error}$

$I = I + (\text{current_error}) * dt$

$D = (\text{current_error} - \text{previous_error}) / dt$

dt is the time of each round of while loop
For example, if you know the frequency of your while loop is 1000, the dt is $1/1000=0.001$

Please recalculate the PID using new formula

Time index	Error	P	I	D	PID
0	0				
1	10				
2	15				
3	10				
4	0				
5	-10				
6	-15				

Frequency =20

$K_p=1$

$K_i=0.1$

$K_d=0.1$

Tips to select P, I, D parameters

- Start with only the **P** term,
- Then add **D** to reduce overshoot.
- Add the **I** term last, and keep it small.
- Use **anti-windup mechanisms**, like:
 - Clamp the integral term to a min/max value.
 - Reset integral when the control output saturates or when a big step change occurs.



General Tuning Strategy

1. Set $K_i = 0$, $K_d = 0$
2. Increase K_p until system starts oscillating → back off a bit.
3. Add K_d to dampen the oscillations.
4. Finally add K_i slowly to eliminate steady-state error.
 1. Optional: Use **anti-windup** to control integral overshoot.