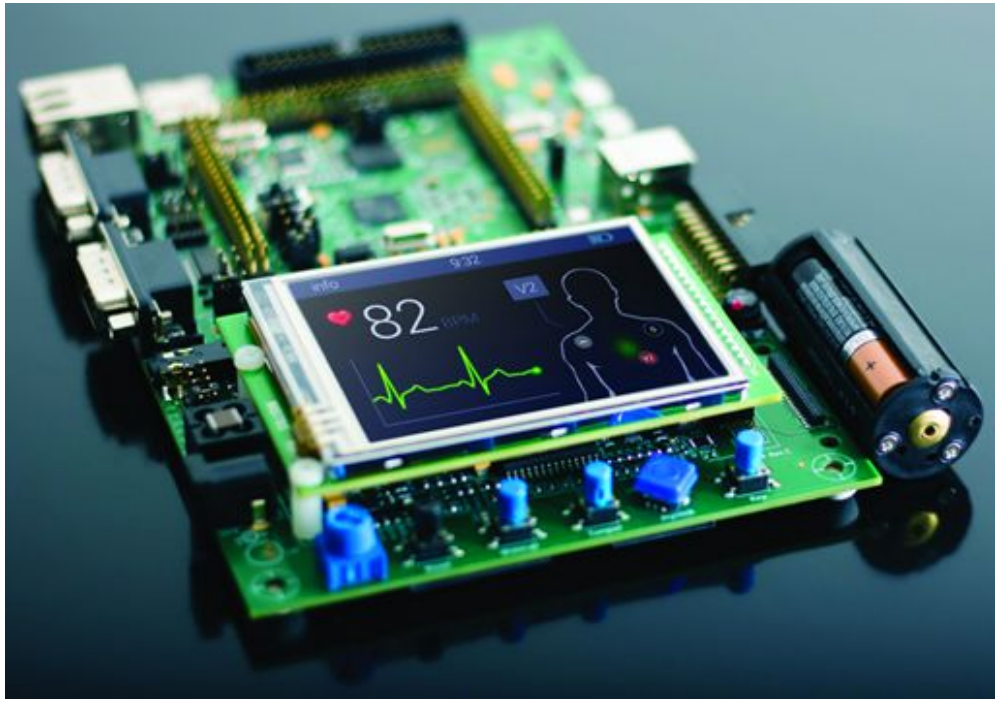
Embedded Software

CS 145/145L



Caio Batista de Melo

Announcements (2022-05-17)

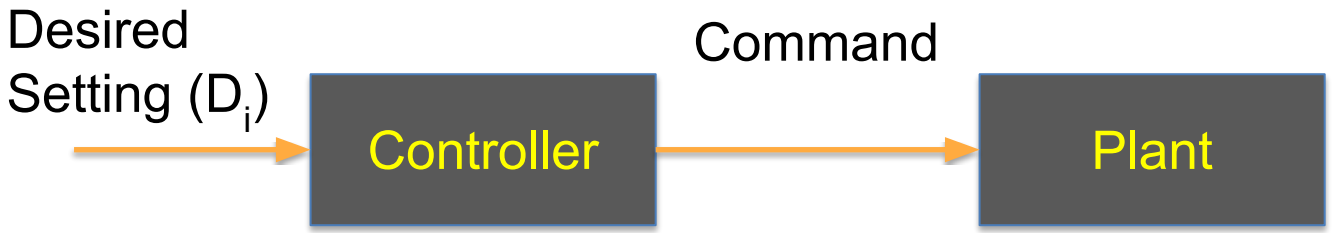
- Guest speaker on Thursday; in-person!
  - Topic: Real-time systems
  - Recent paper from speaker: https://ieeexplore.ieee.org/abstract/document/9470238
  - If you have early questions for them: https://dotstorming.com/b/627e93e7d34fa625f5e0ac9d
- Project 4 and Homework 4 are due at the end of this week
- Project 5 page should be visible now
  - https://canvas.eee.uci.edu/courses/45047/assignments/929276

Agenda

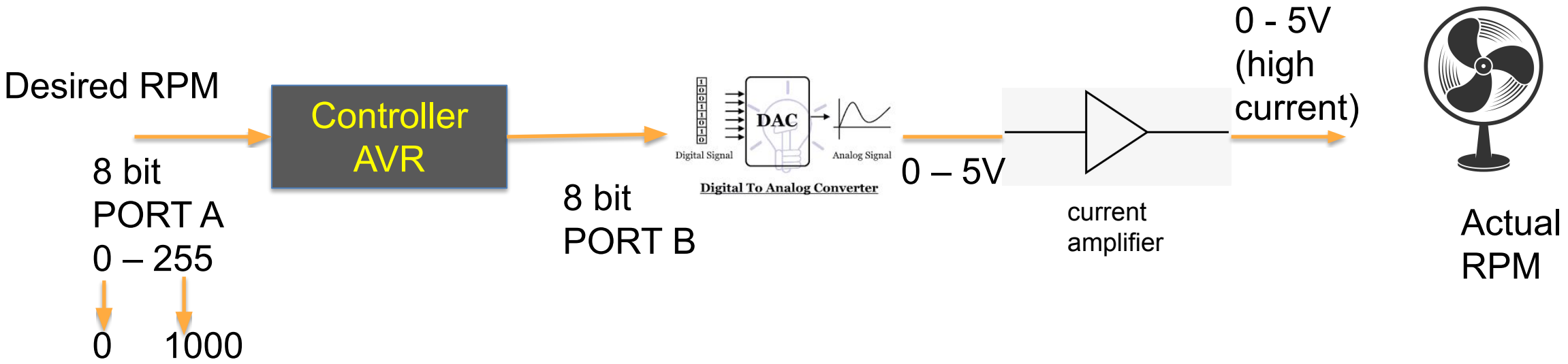
- Control Systems
- Open-Loop Control
- Close-Loop Control
  - Proportional Controller
  - Proportional Integral Controller
  - Proportional Integral Derivative Controller
- Examples

Control Systems

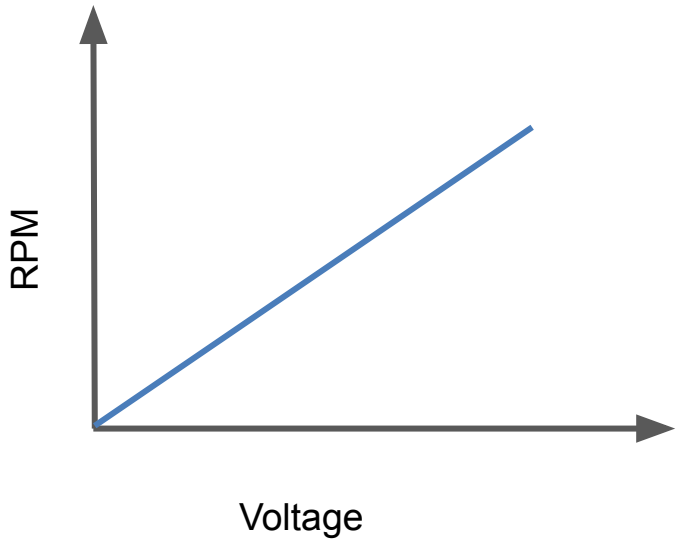
Control Systems



Example:

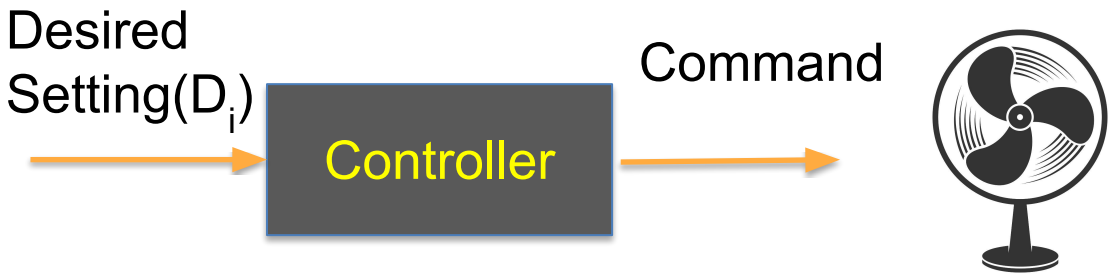


Modeling The Plant



Input	Output
0 V	0 RPM
… V	… RPM
5 V	1000 RPM

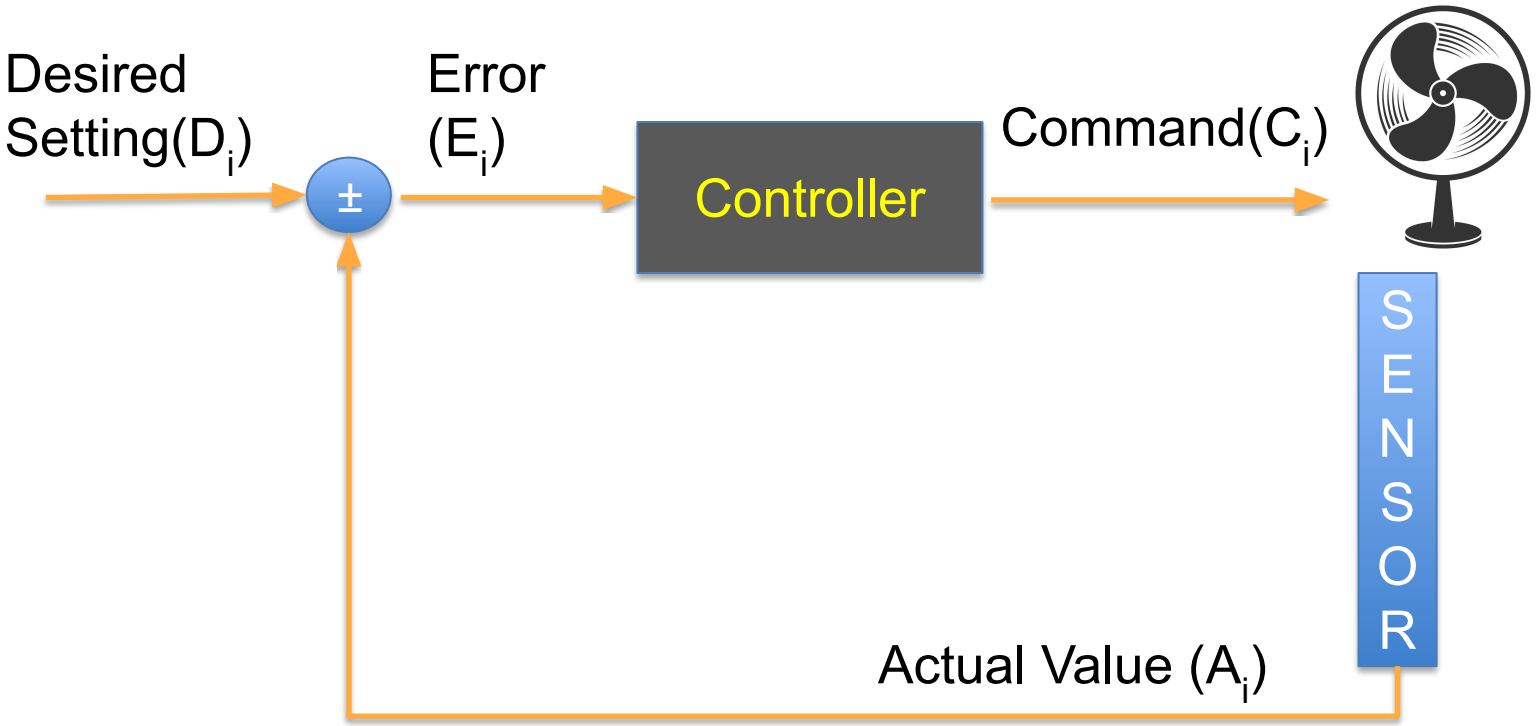
Open-Loop Controller



Input	Output
0 V	0 RPM
… V	… RPM
5 V	1000 RPM

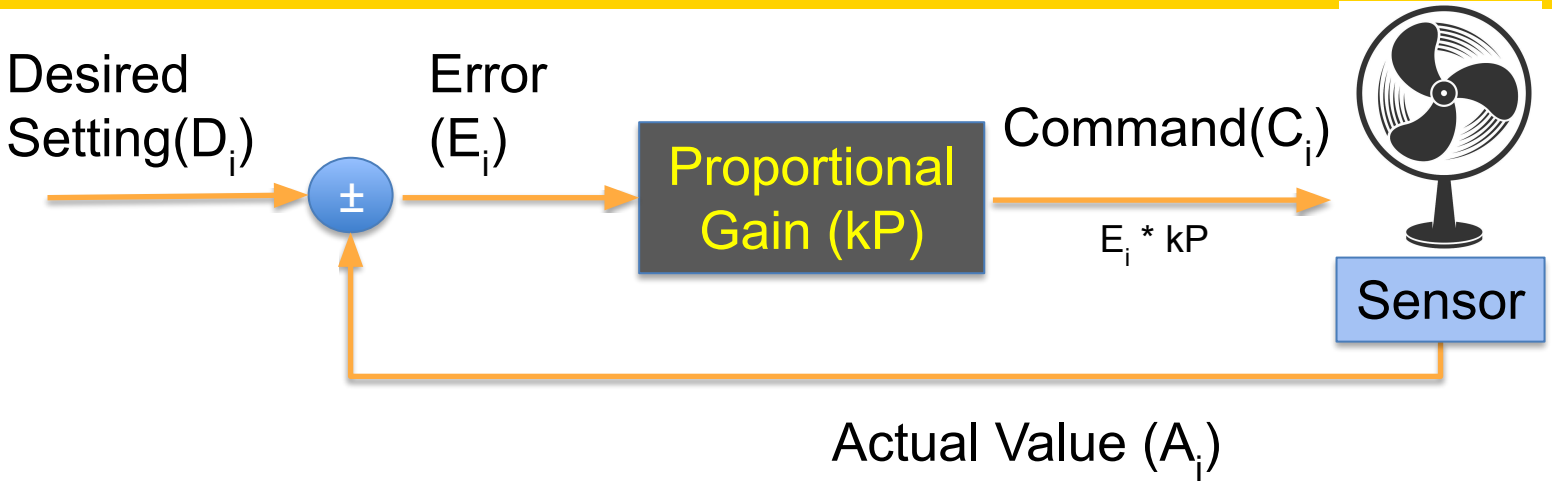
```
int main() {
  int desired;
  float new_value;
  while (1) {
    desired = get_desired();
    new_value = desired / 1000.0 * 5;
    command(new_value);
    avr_wait(100);
  }
  return 0;
}
```

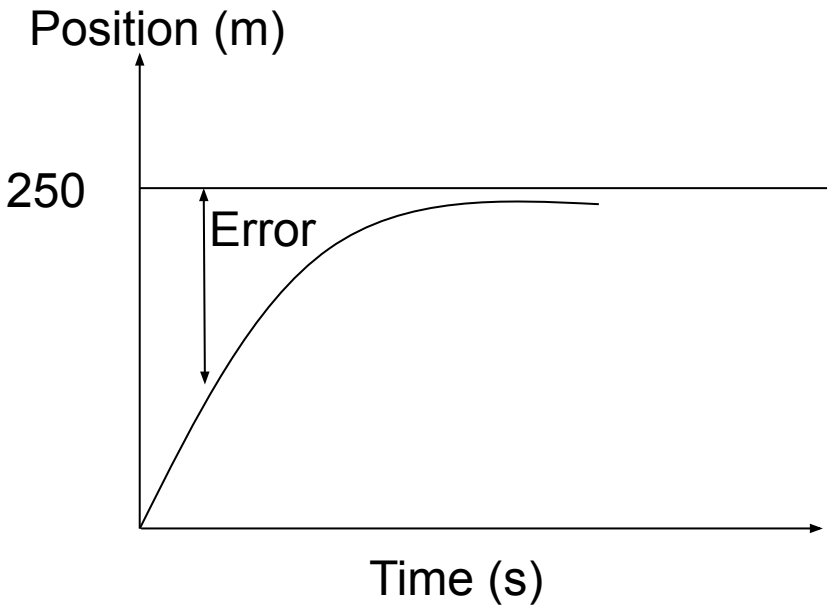
Closed-Loop Controller

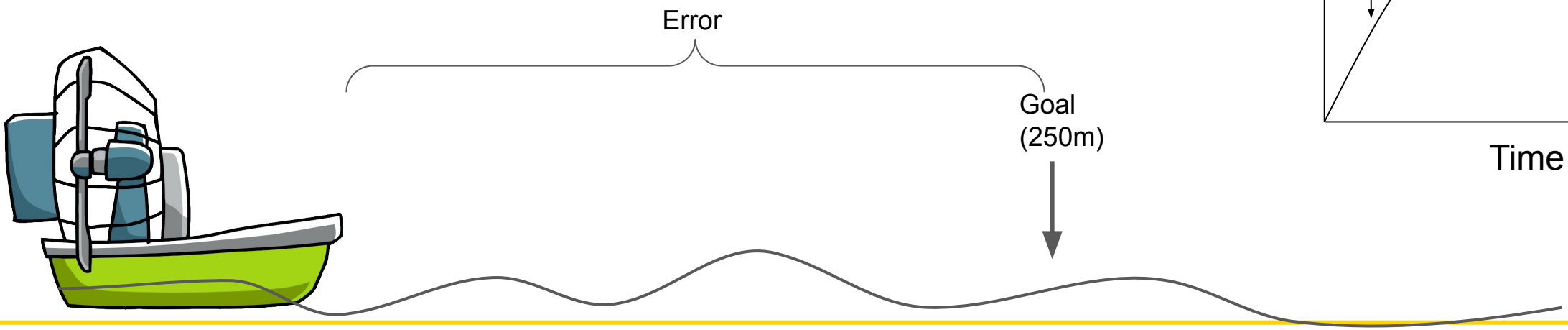


Goal: Minimize Error

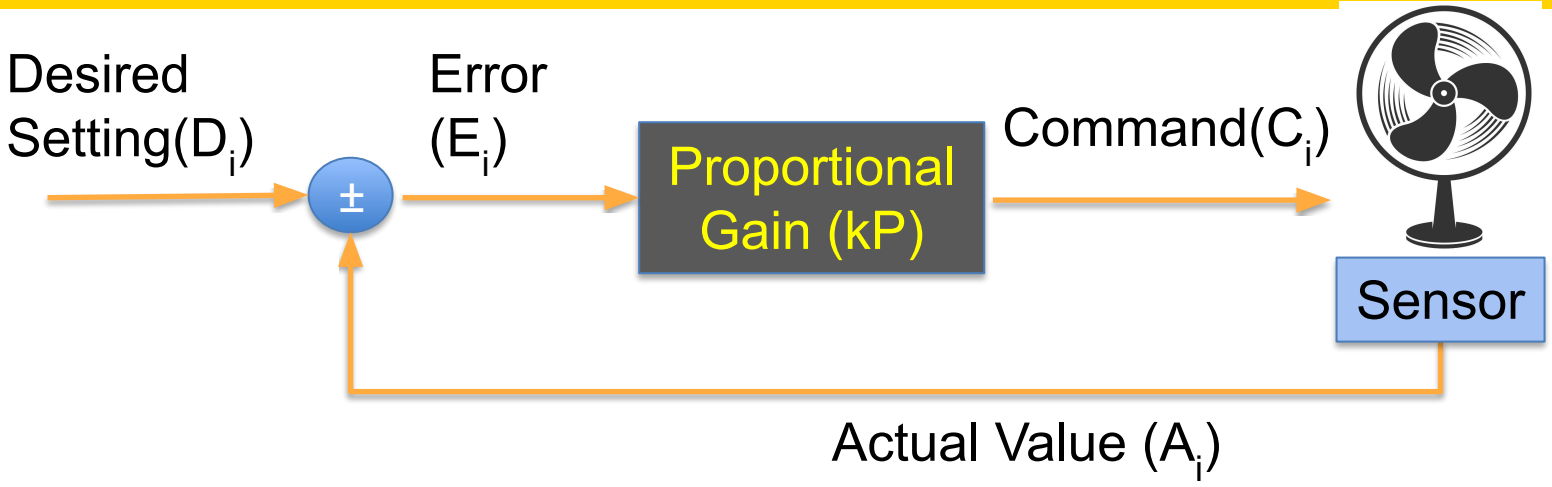
Proportional Controller



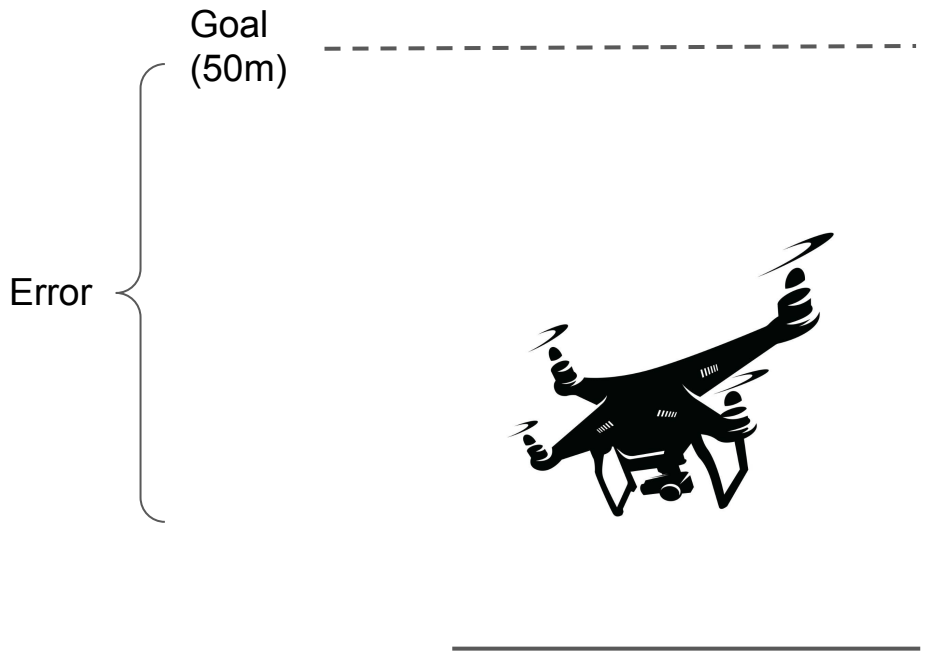




Proportional Controller



What's the problem here?
Once error is 0, the drone will fall!

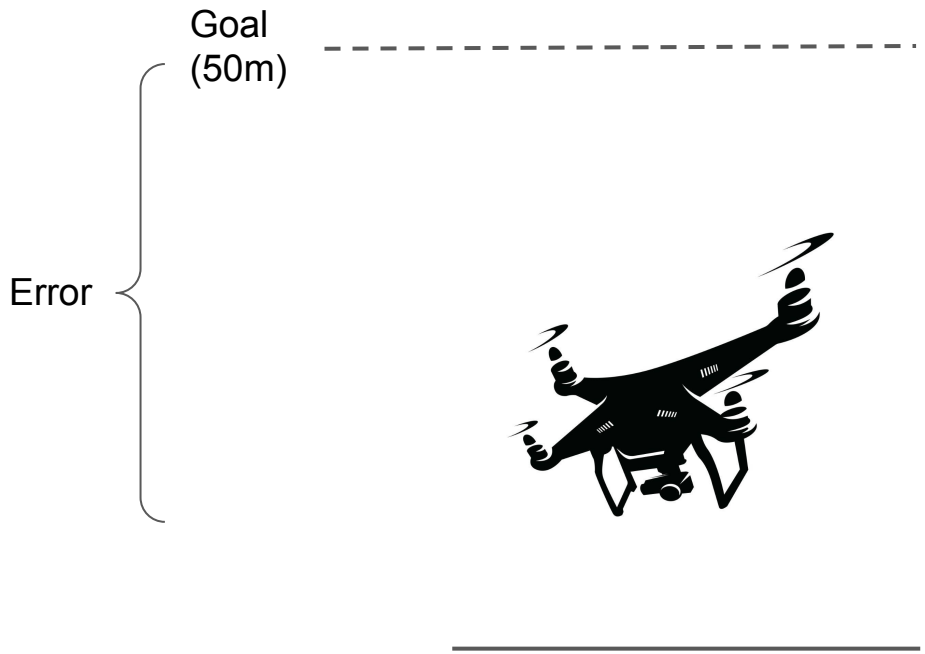


Steady State Error

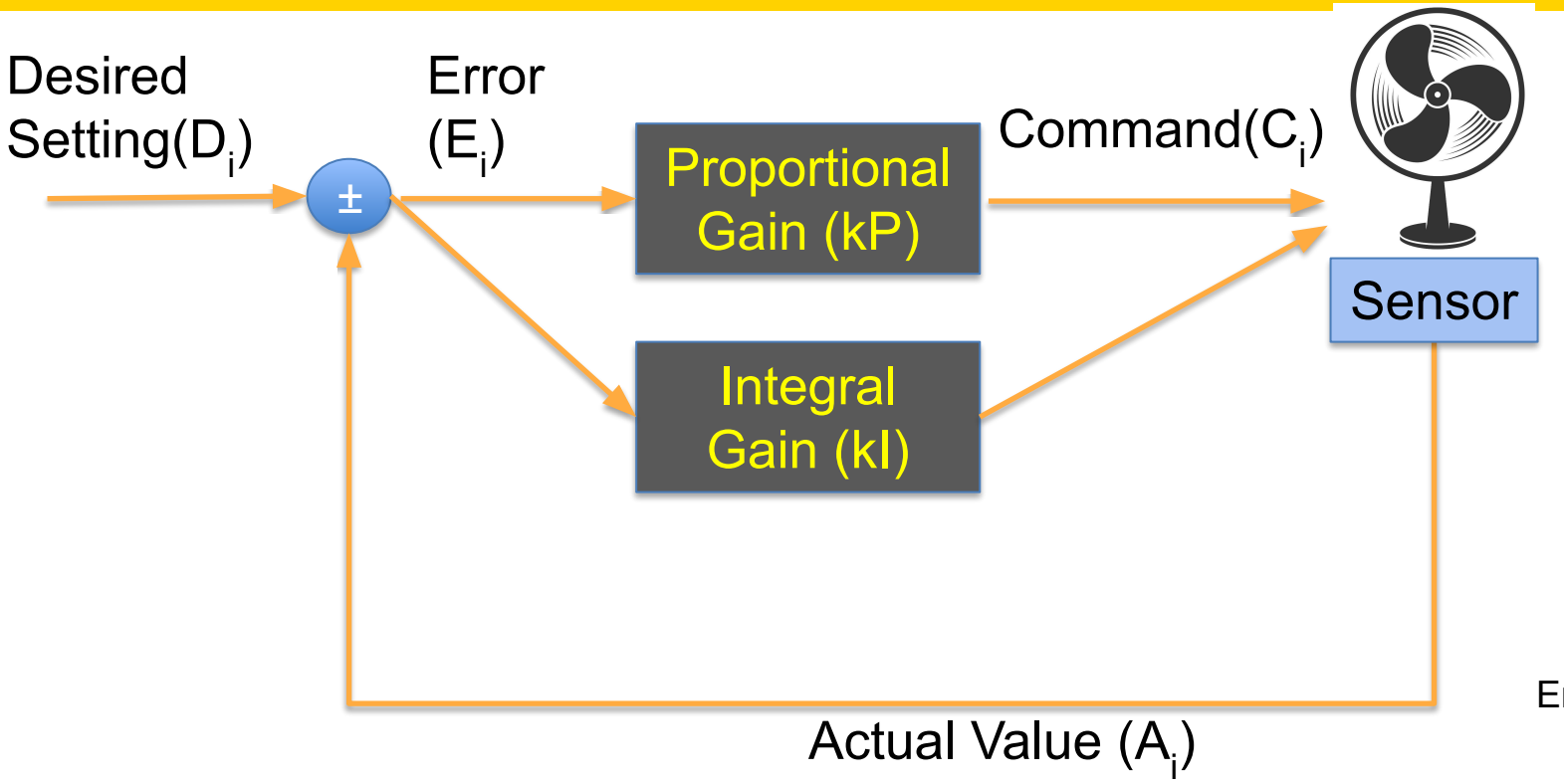
- Let's say the drone hovers at 20 RPM
- Also, let's assume kP = 1

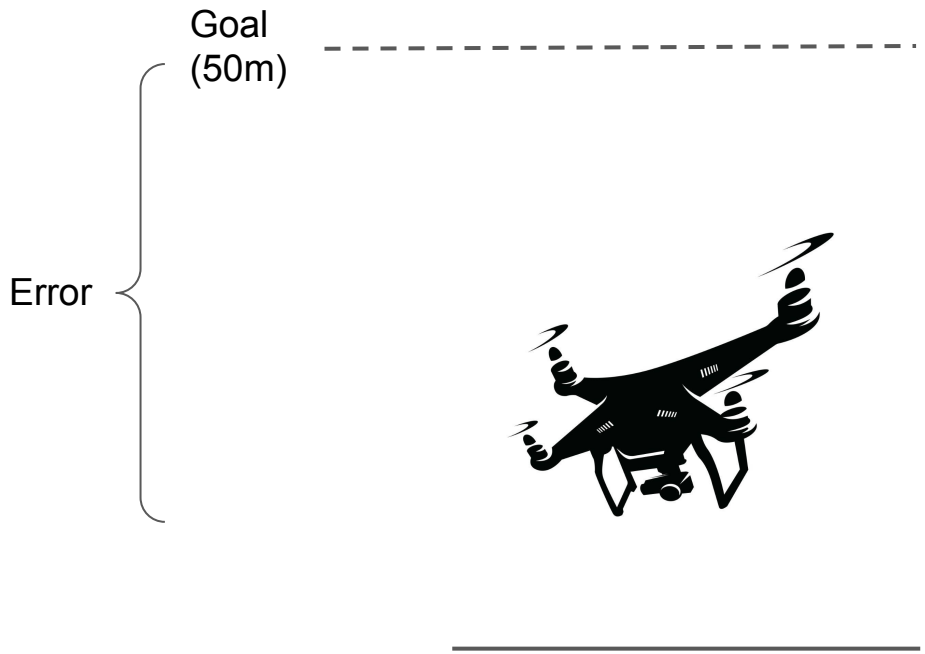
- Whenever altitude == 30
  - error will be 20
  - command will be 20 RPM

- If we change the value of kP, can we fix this?
  - No!
- kP = 100 -> altitude == 48m
  - error will be 2
  - command will be 20 RPM



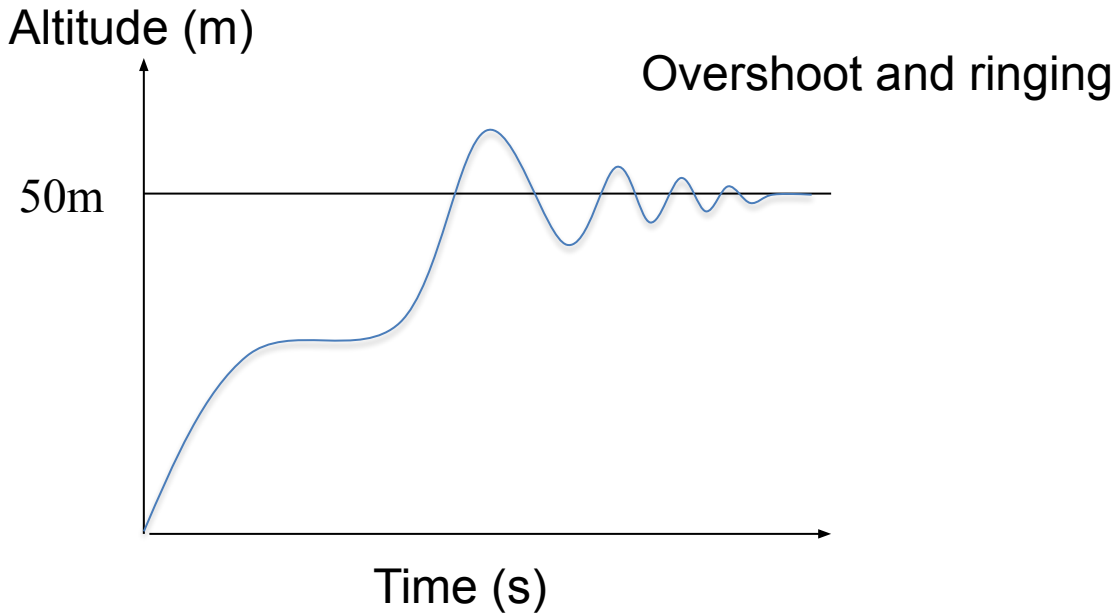
Proportional Integral Controller

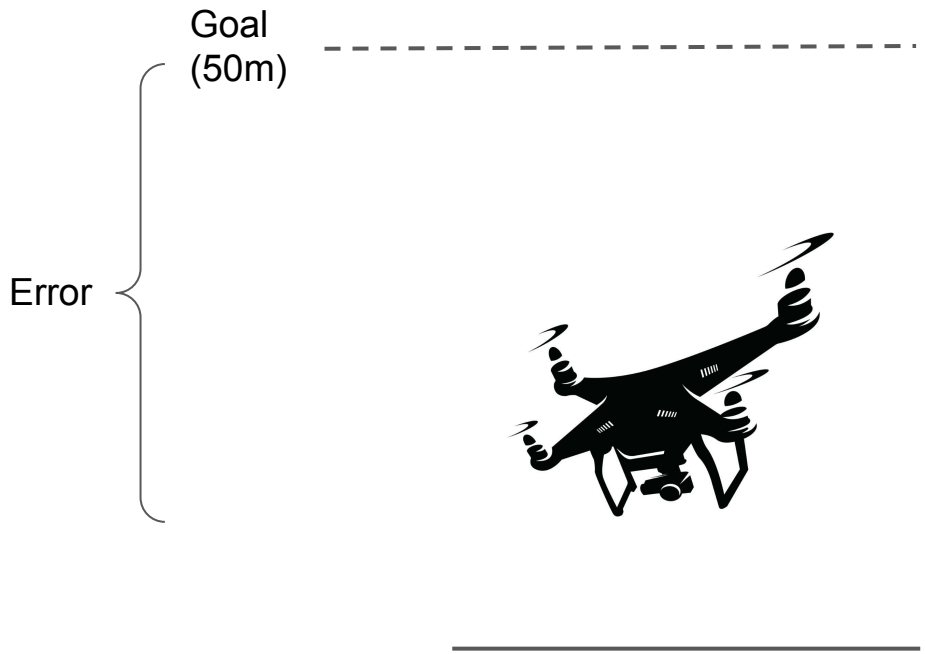




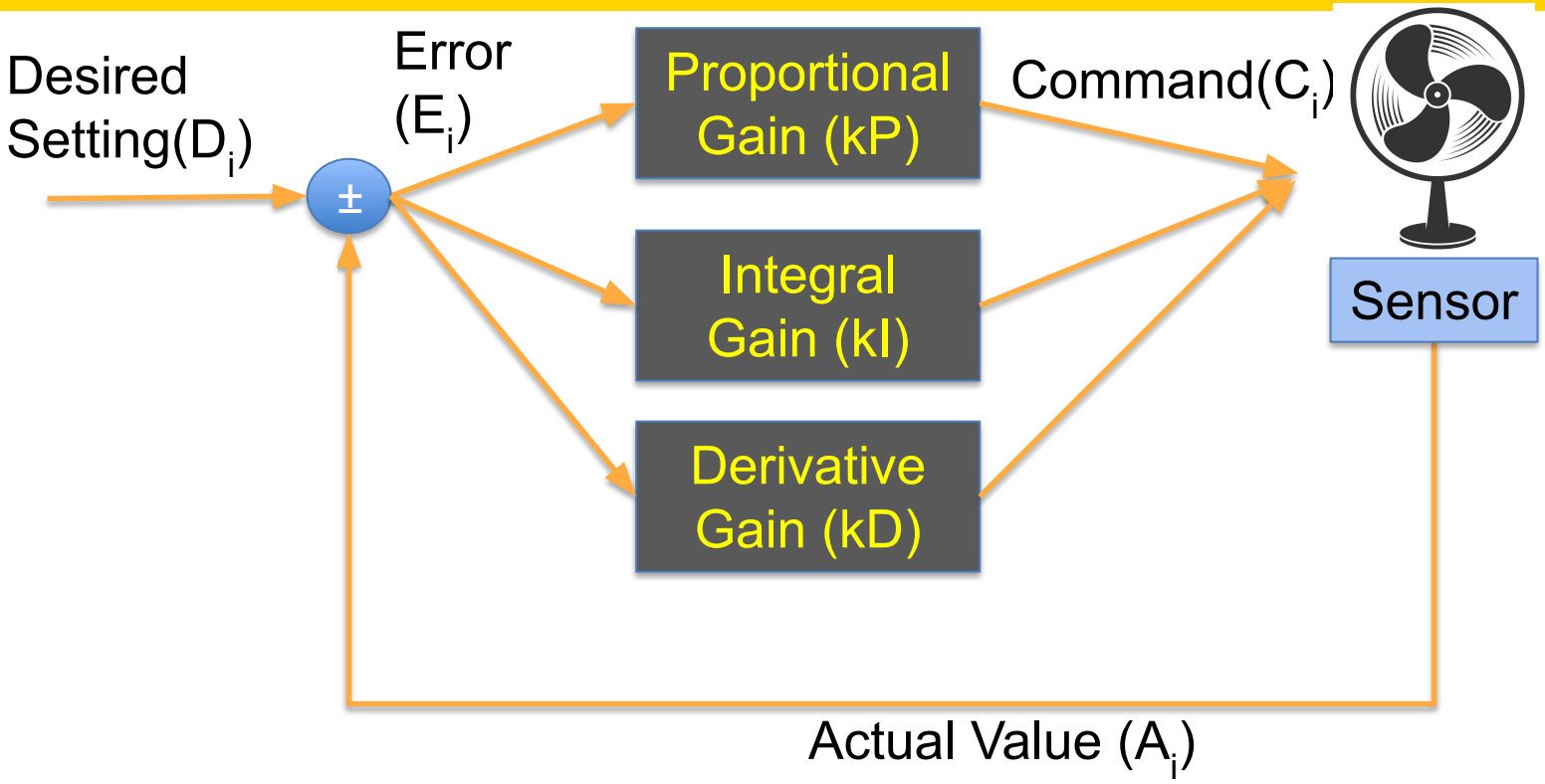
$$Command(C_i) = \{ E_i * kP \} + \{ (E_0 + E_1 + \ldots + E_i) * kI \}$$

Proportional Integral Controller





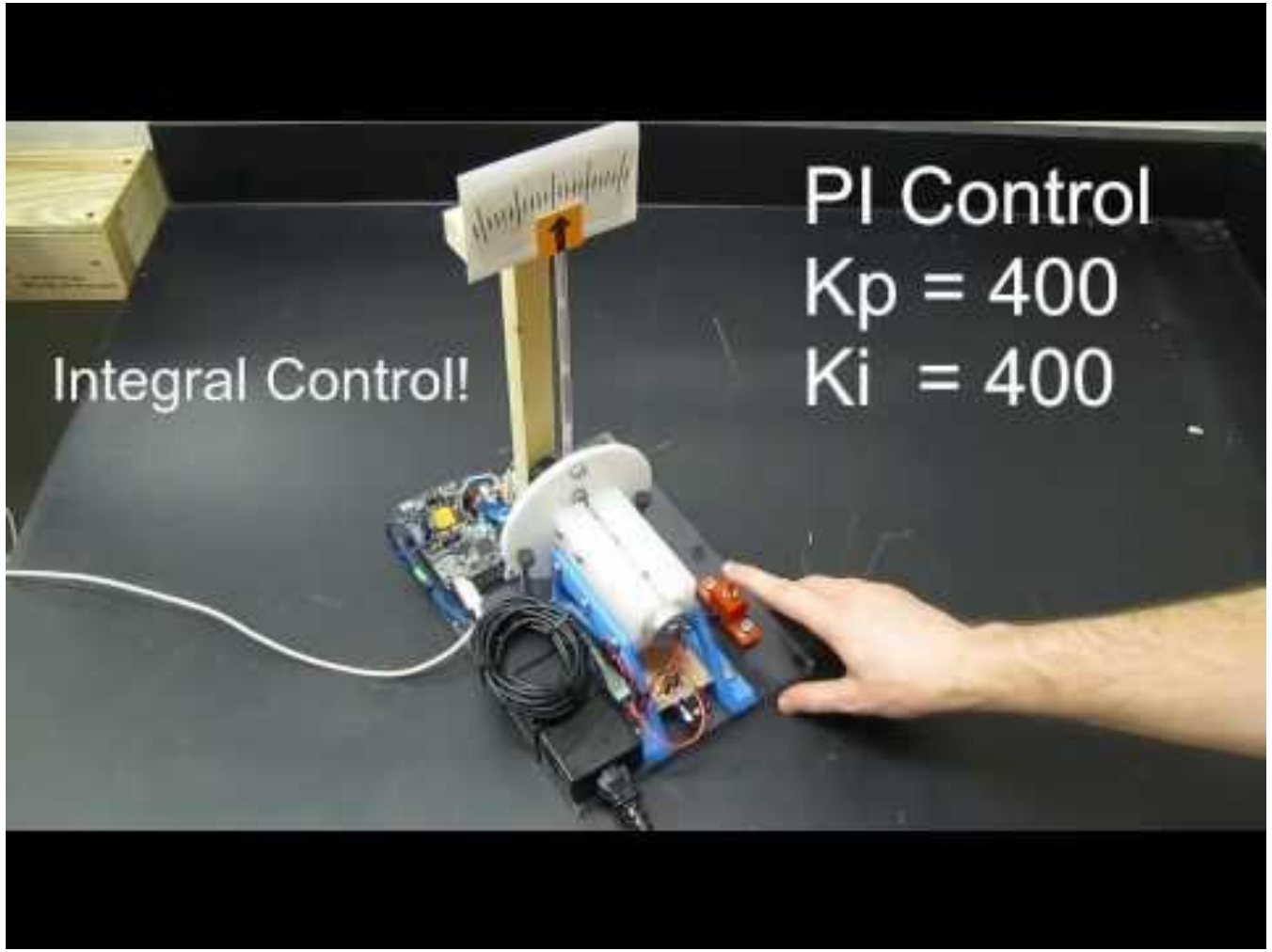
Proportional Integral Derivative Controller



- Derivative looks at how fast the error is changing;
- If error is changing quickly, we decrease the change
  - Prevents overshooting

$$\text{Command}(C_i) = \{ E_i * kP \} + \{ (E_0 + E_1 + \ldots + E_i) * kI \} + \{ (E_i - E_{i-1}) * kD \}$$

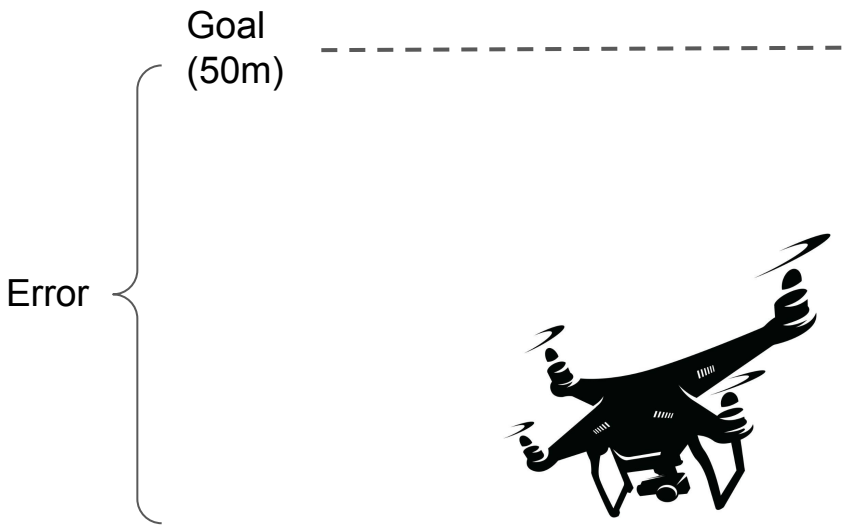
Video Example of a PID



https://www.youtube.com/watch?v=fusr9eTceEo

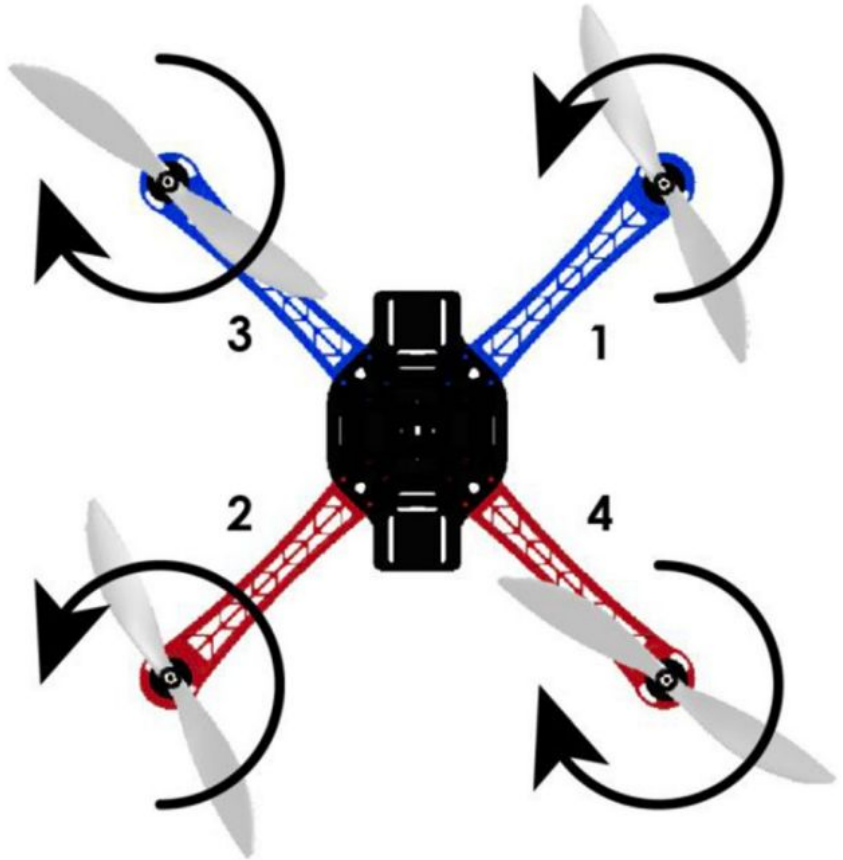
Integral Windup

- Windup happens when the Integral part thinks we're acting more than we actually are!
- Imagine we have a max of 5k RPM
- If error is 50, and kI is 1000, it'll command 50k RPM
- So whenever we reach the goal, it thinks we used 50k RPM, and will try to correct those large errors.



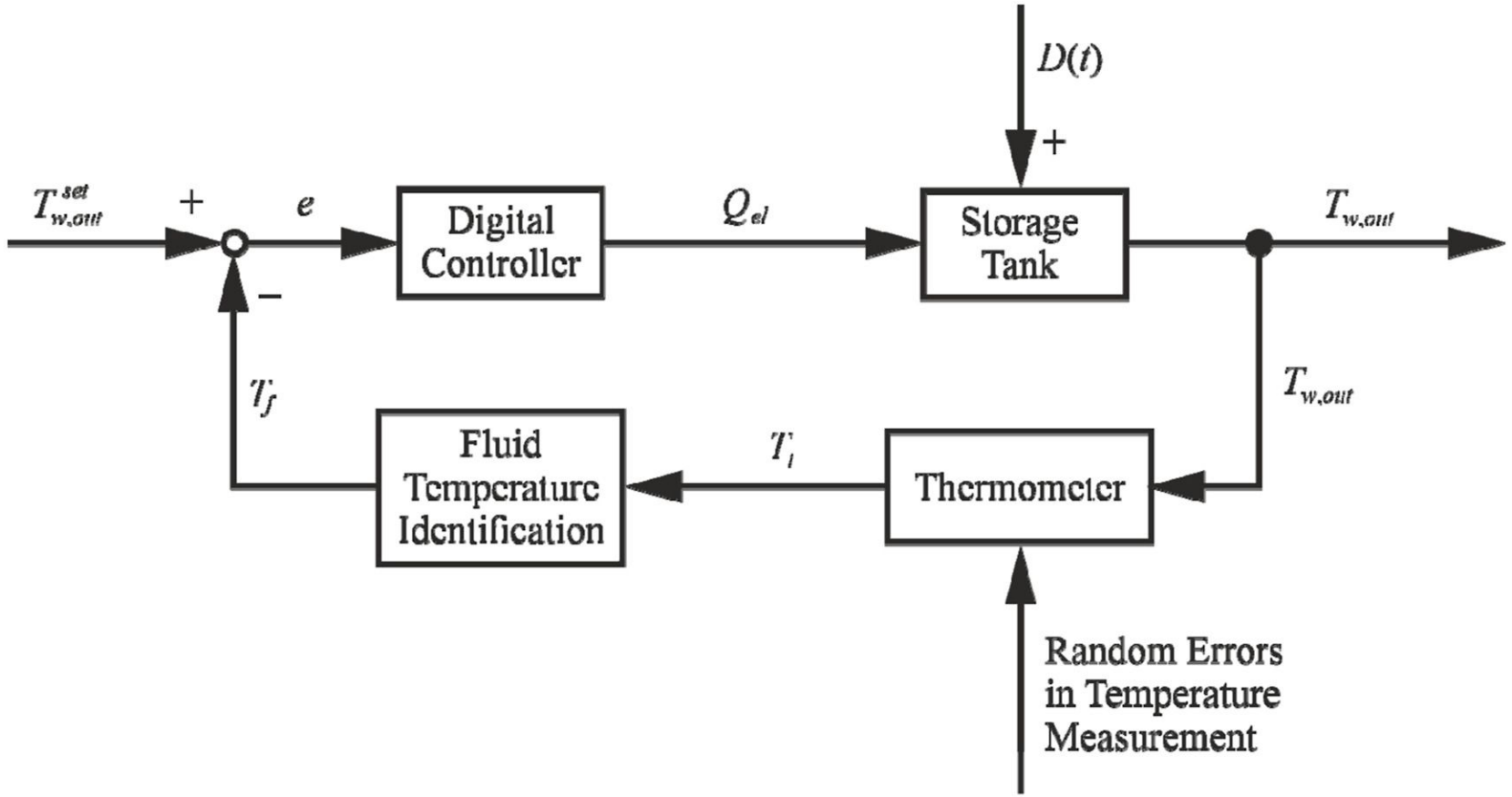
Examples

Adaptive PID Controller for Quadrotor UAV



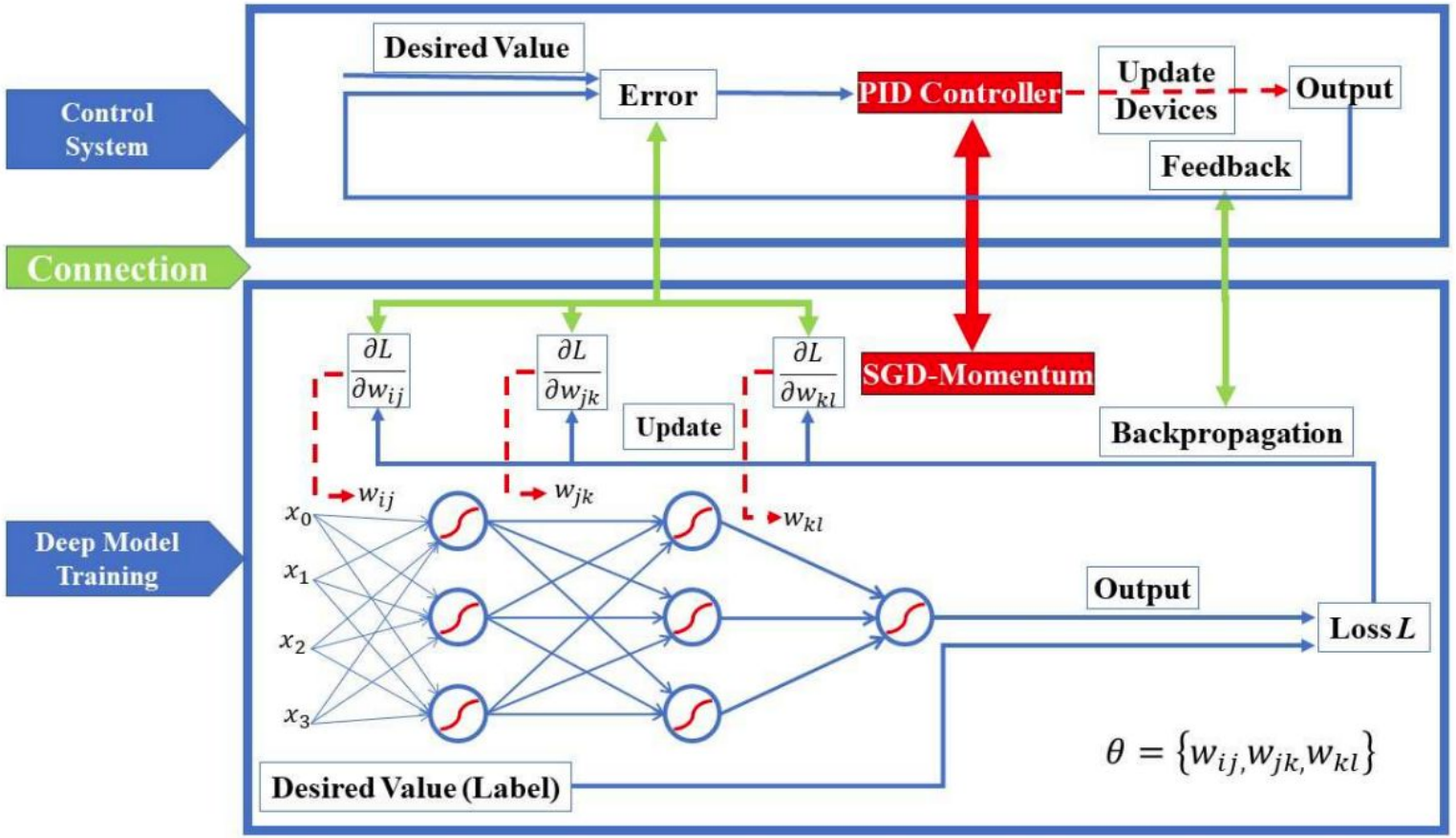
https://link.springer.com/article/10.1007/s13369-020-04742-w

Hot Liquid Temperature Control with PID Controller



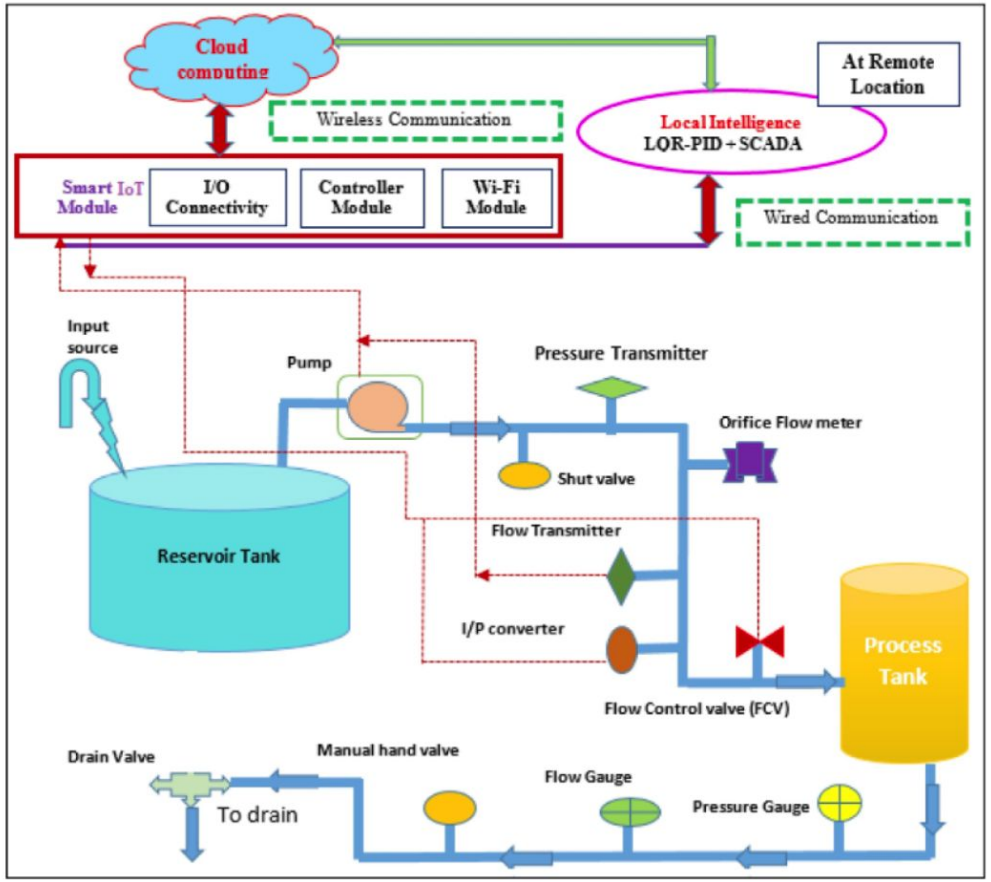
https://www.sciencedirect.com/science/article/pii/S0360544221030206

PID Controller for Optimization of Deep Networks



https://ieeexplore.ieee.org/document/8578987

PID Controller for Fluid Transportation System



https://www.sciencedirect.com/science/article/pii/S2452414X20300030

See you next time :)

Q & A