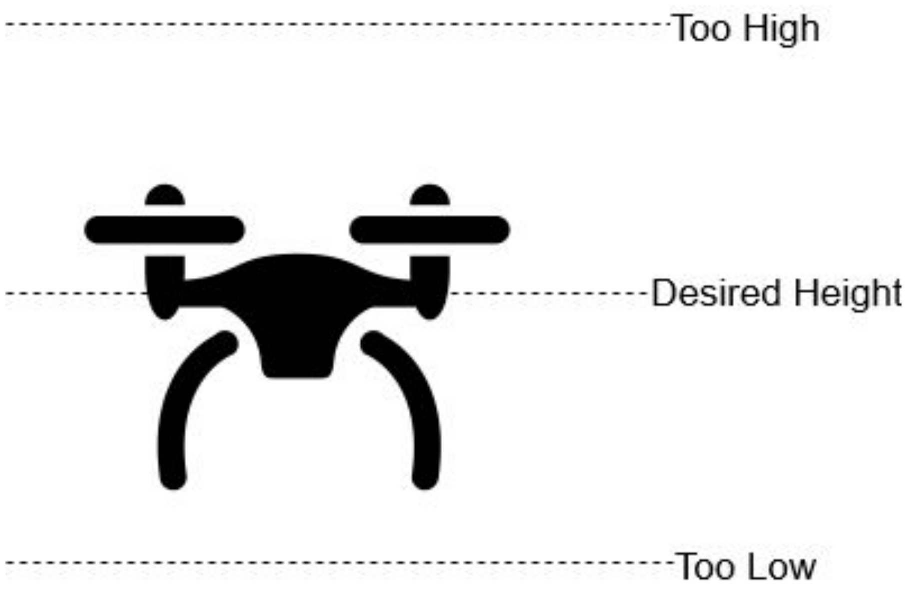
Basic Robotics Course

Class 3 - Automatic Control

How did you make the drone hover on the simulation?

What did you do?



- Throttle Up when the drone was Too Low.
- Throttle Down when the drone was Too High.

What did you do?

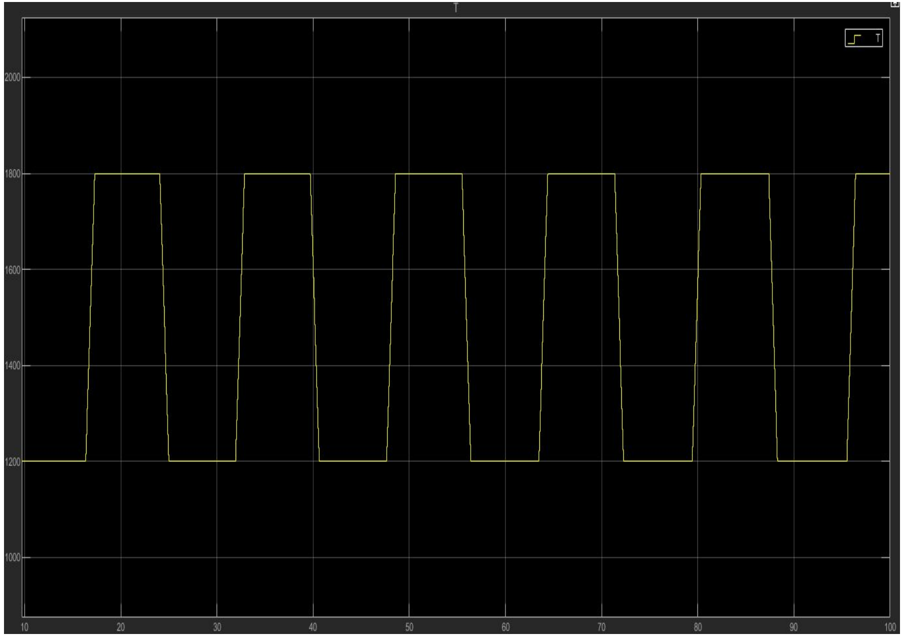


Chart 1: Throttle on Time

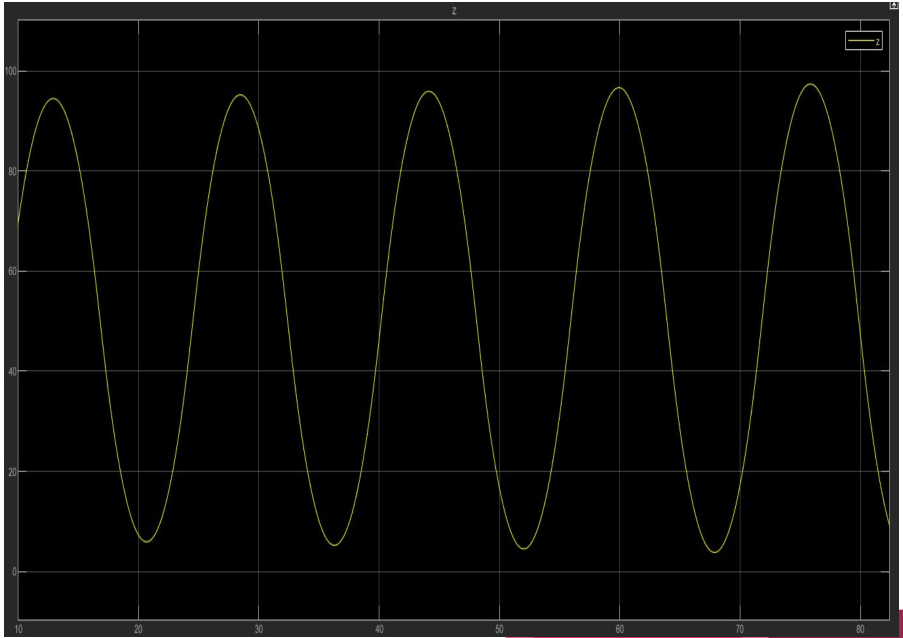
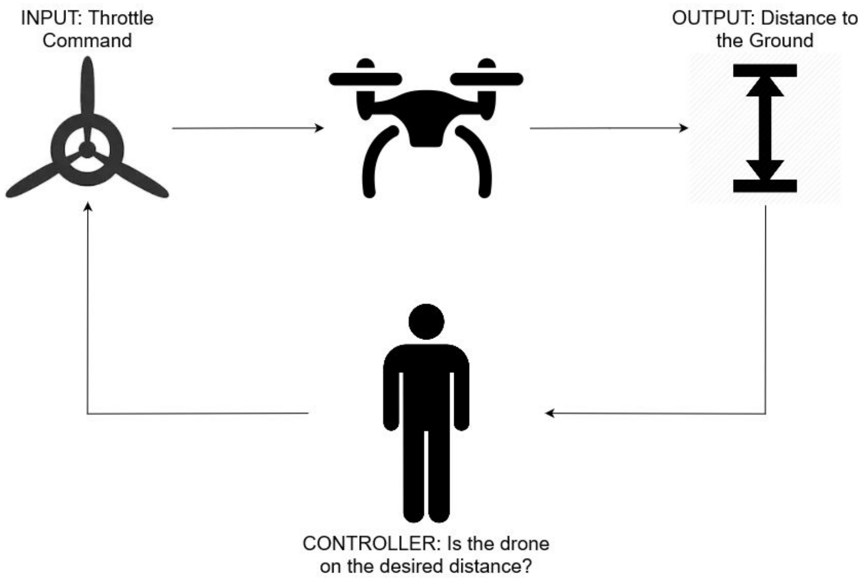


Chart 2: Distance to ground on Time

Control Theory



- The controller monitors the controlled process variables (the current distance to the ground).
- Compares with the reference or set point (desired distance).
- Generates a control action to bring the controlled process variable to the same value as the reference.

Open-loop Control

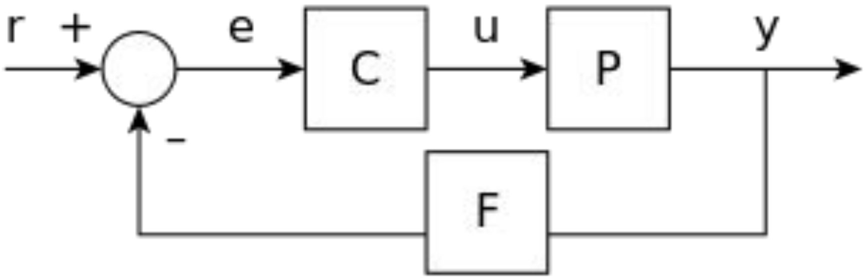
- The control action doesn’t depend on the output.
- There’s no feedback.
- Example: a microwave device that has heat as output, but the control action doesn’t depend on the heat or the temperature, depending just on a timer.

Closed-loop

- The control action depends on the output.
- There's feedback.
- Example: an air conditioner device that has the temperature as output and it's control action depends on the sensed temperature.

Closed-loop Control

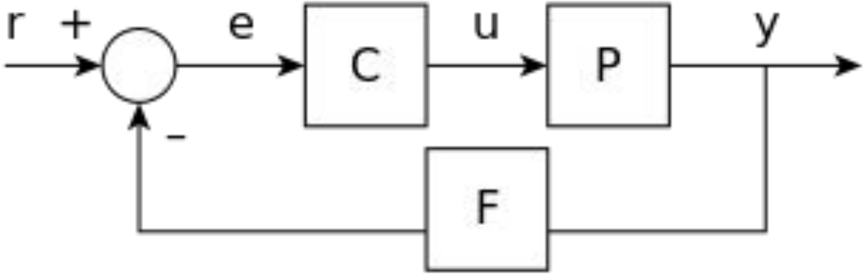
- **P** is the **Plant**, the system being controlled (the air conditioner).
- **C** is the **Controller** (thermostat).
- **y** is the **Output** (temperature).
- **F** is the **Feedback Element**, generally a **Sensor** that measures the output (thermometer).



Typical block diagram of a SISO (Single Input Single Output) control system

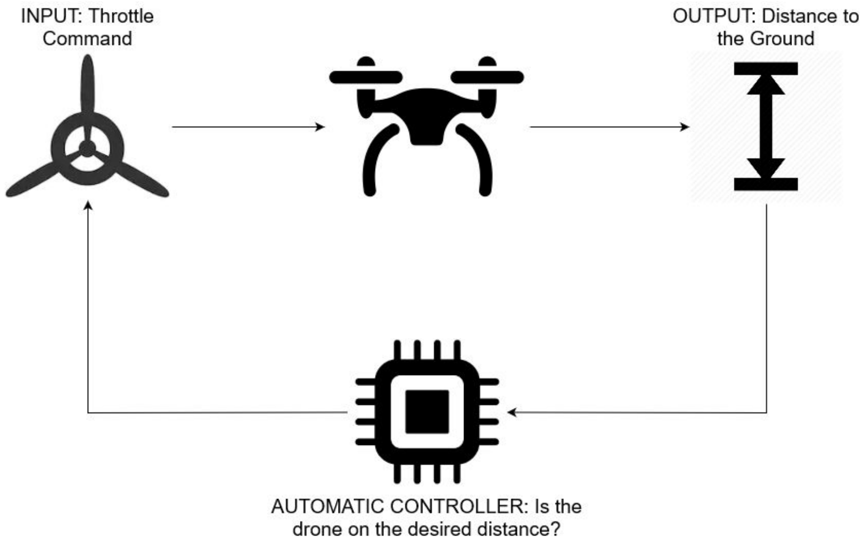
Closed-loop Control

- **r** is the **Reference Value** (desired temperature).
- **e** is the **Error** between the reference value and the desired value.
- **u** is the **Input** of the system being controlled (turn on or turn off the cooling device).



Typical block diagram of a SISO (Single Input Single Output) control system

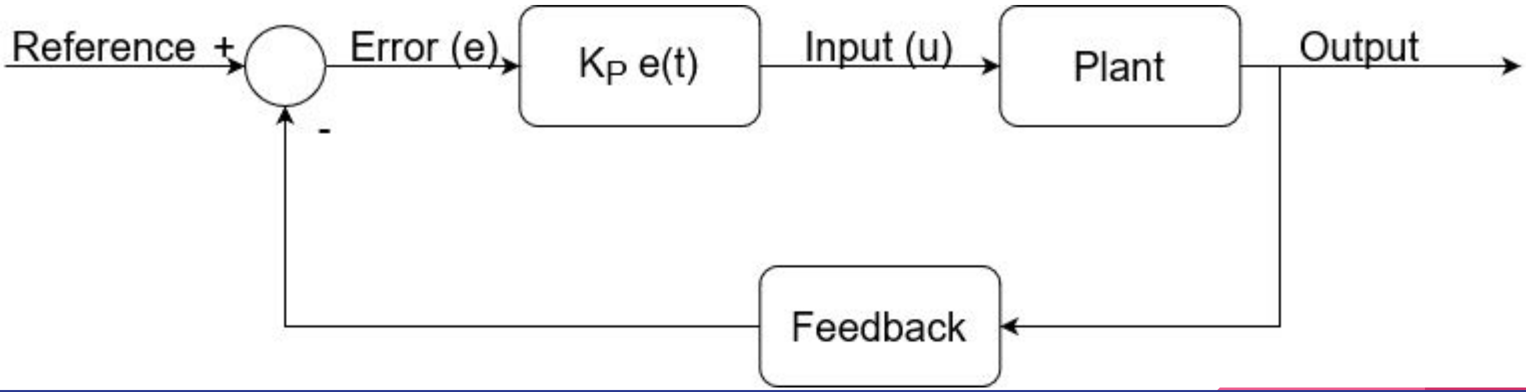
Automatic Control



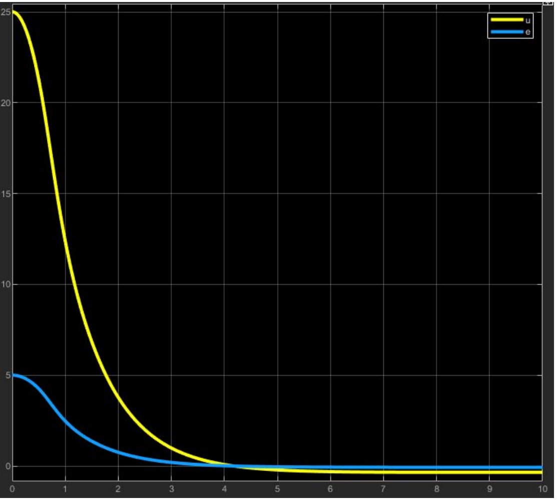
- How to make a computer control our process automatically?
- What logic adopt?

Proportional Controller

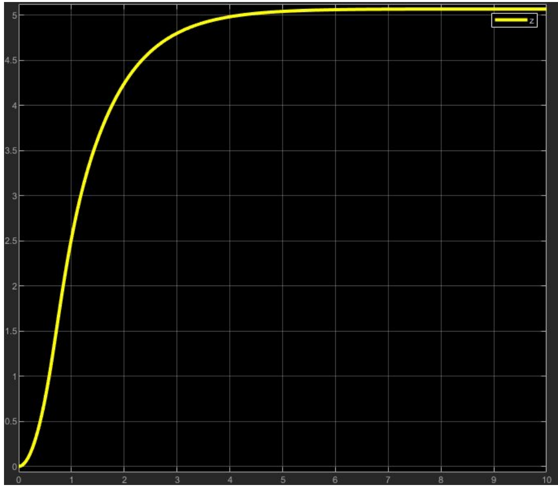
- The Input is proportional to the error.
- Only the current error matters.
- $u(t) = K_P\, e(t)$.



Proportional Controller - Example



- The Plant is the drone, the Input is the vertical velocity (vz), and the Output is the vertical position (z).
- The drone starts at z = 0 and must go to z = 5.
- A P controller (K_P = 5) will make the velocity proportional to the position error.

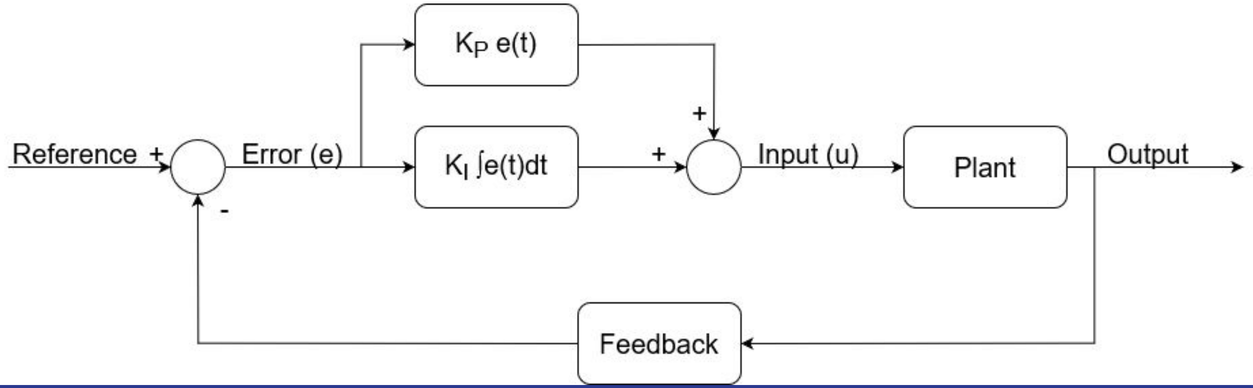


Proportional Controller

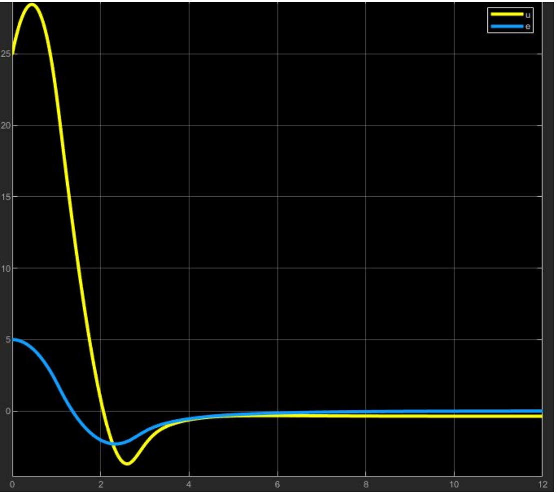
- The output will reach a near value to the desired, but won’t reach it.
- The difference in the time infinity is called **steady-state error**.

PI Controller

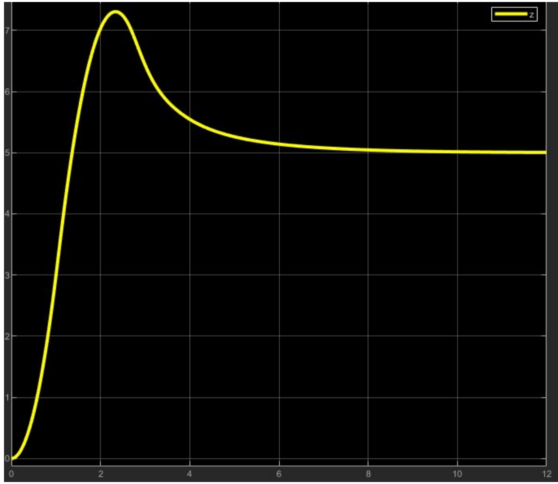
- Have an Integral component to generate the Input.
- That way, the former errors are used on the calculus, correcting the steady-state error. The current and past errors matter.
- $u(t) = K_P\, e(t) + K_I \int e(t)dt$.



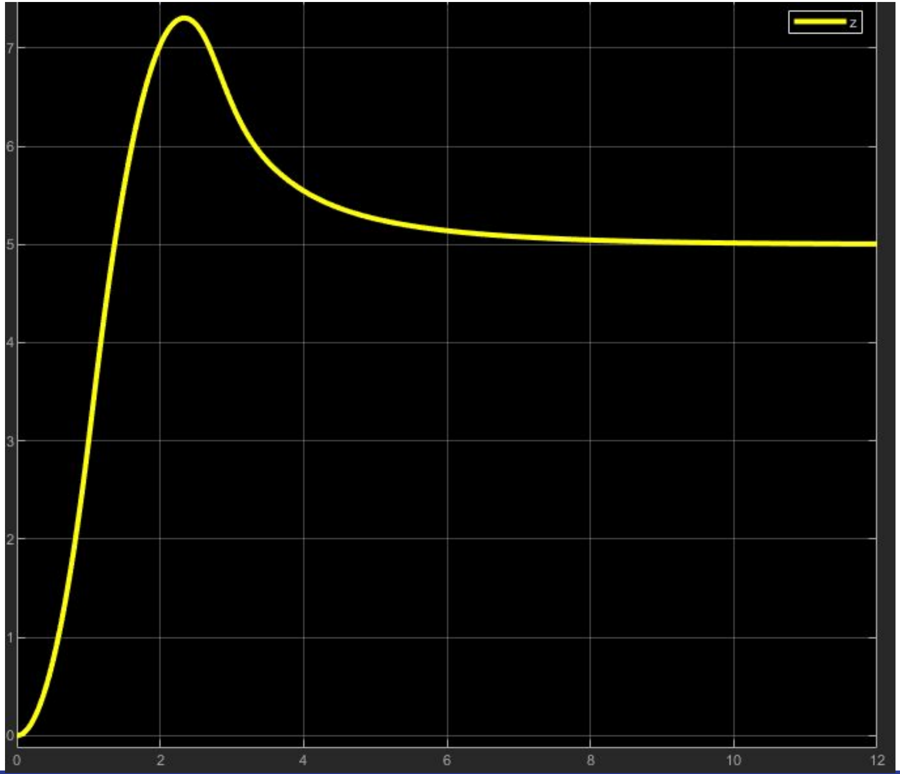
PI Controller - Example



- The Plant is the drone, the Input is the vertical velocity (vz), and the Output is the vertical position (z).
- The drone starts at z = 0 and must go to z = 5.
- A PI controller (K_P = 5, K_I = 3) will command the velocity.



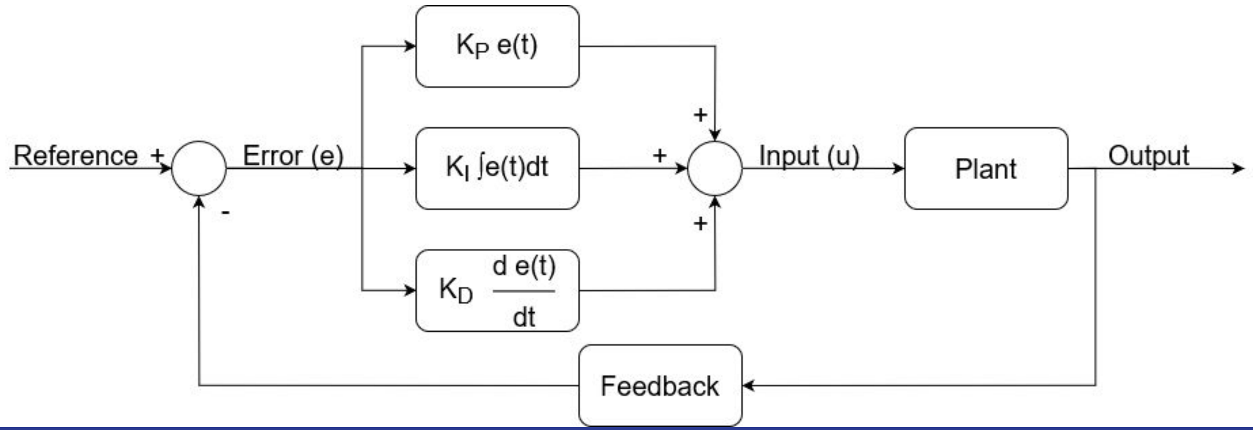
PI Controller



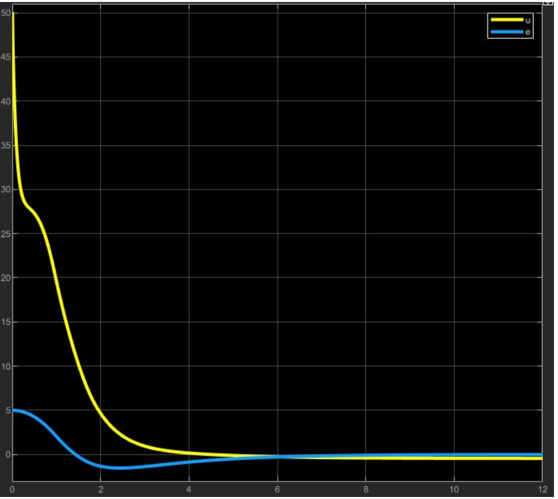
- It surpasses the desired value by more than 2 meters before stabilizing.
- We say that it has a high overshoot.
- This overshoot can be a problem.

PID Controller

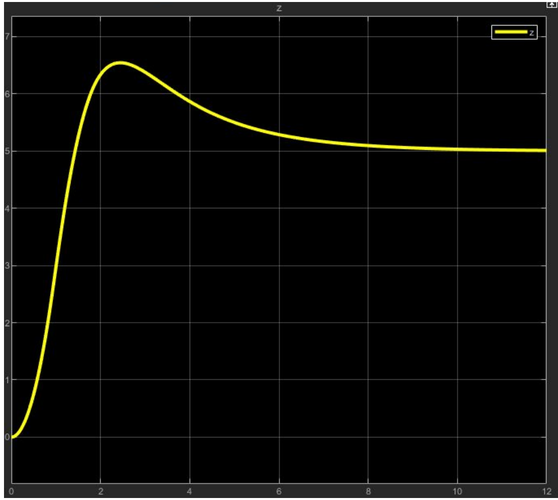
- Have a Derivative component to generate the Input.
- That way, the tendency for future errors is used on the calculus, allowing a performance improvement.
- $u(t) = K_P\, e(t) + K_I \int e(t)dt + K_D\, d\, e(t)/dt$.



PID Controller - Example



- The Plant is the drone, the Input is the vertical velocity (vz), and the Output is the vertical position (z).
- The drone starts at z = 0 and must go to z = 5.
- A PID controller (K_P = 5, K_I = 3, K_D = 1) will command the velocity.



MATLAB and Simulink Time