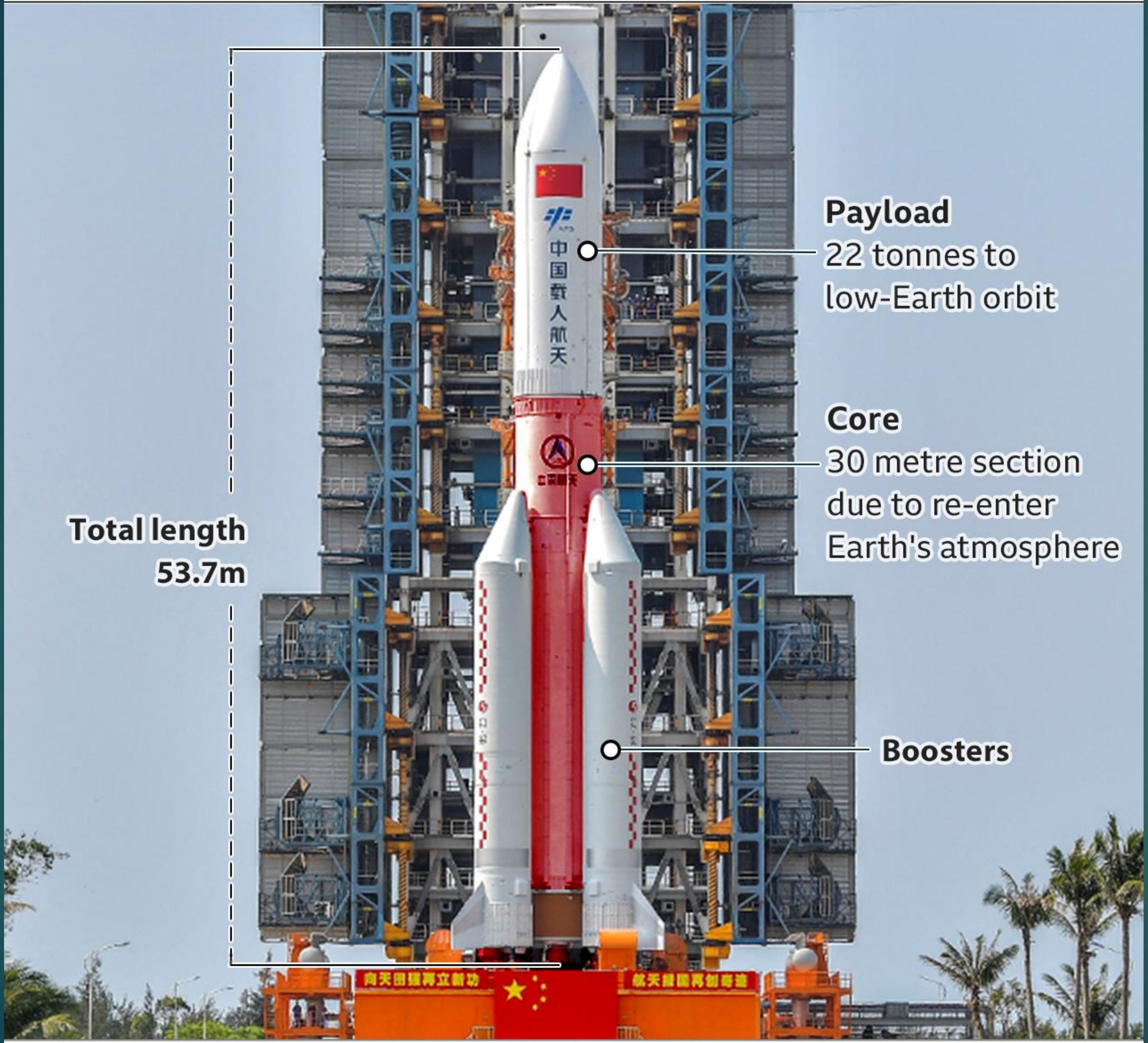
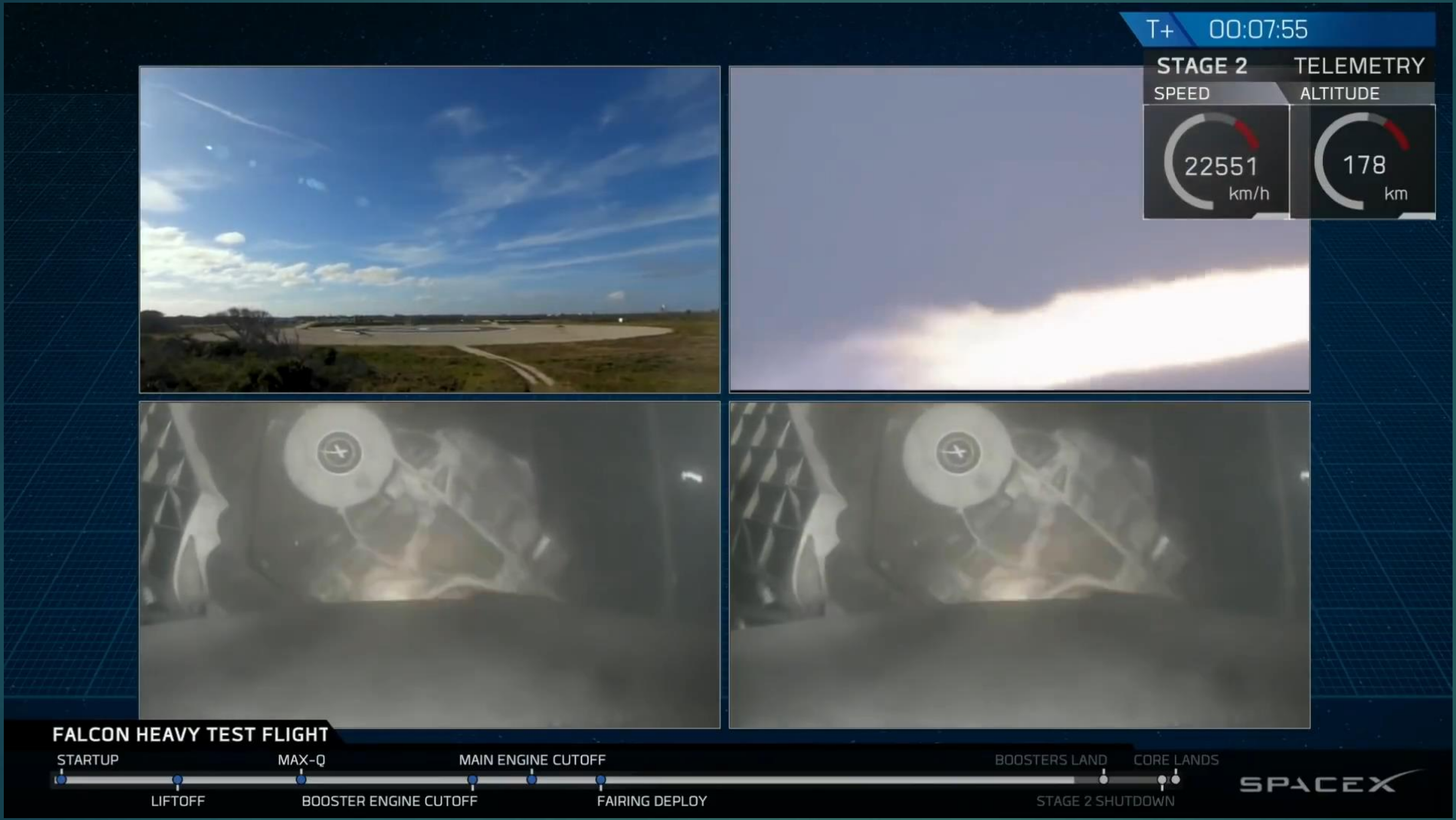
China's Long March 5b rocket



Source: Xinhua / China Daily

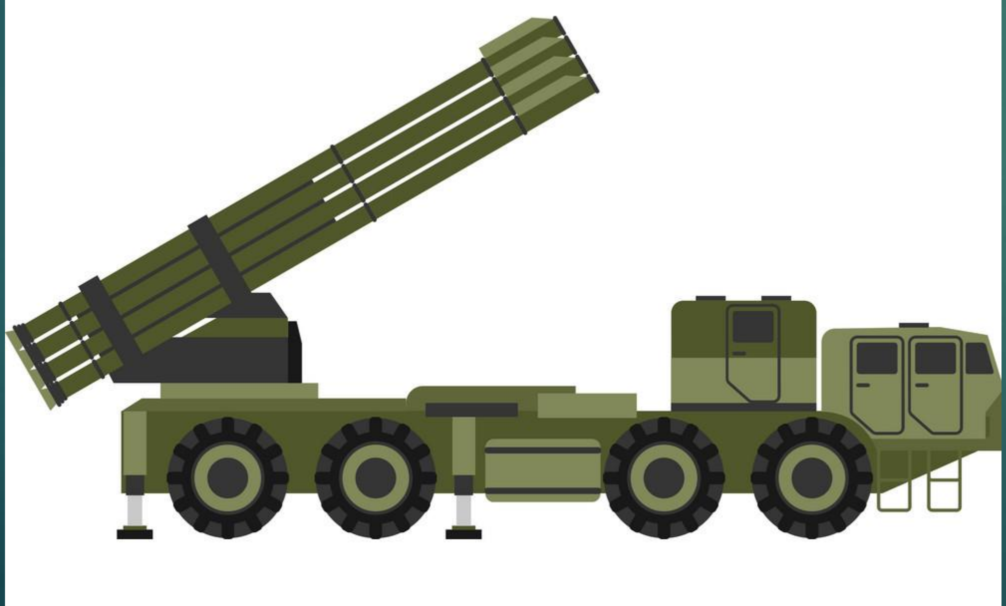

T+ 00:07:55
STAGE 2 TELEMETRY
SPEED
22551
km/h
ALTITUDE
178
km
FALCON HEAVY TEST FLIGHT
STARTUP
MAX-Q
MAIN ENGINE CUTOFF
BOOSTERS LAND
CORE LANDS
LIFTOFF
BOOSTER ENGINE CUTOFF
FAIRING DEPLOY
STAGE 2 SHUTDOWN
SPACEX

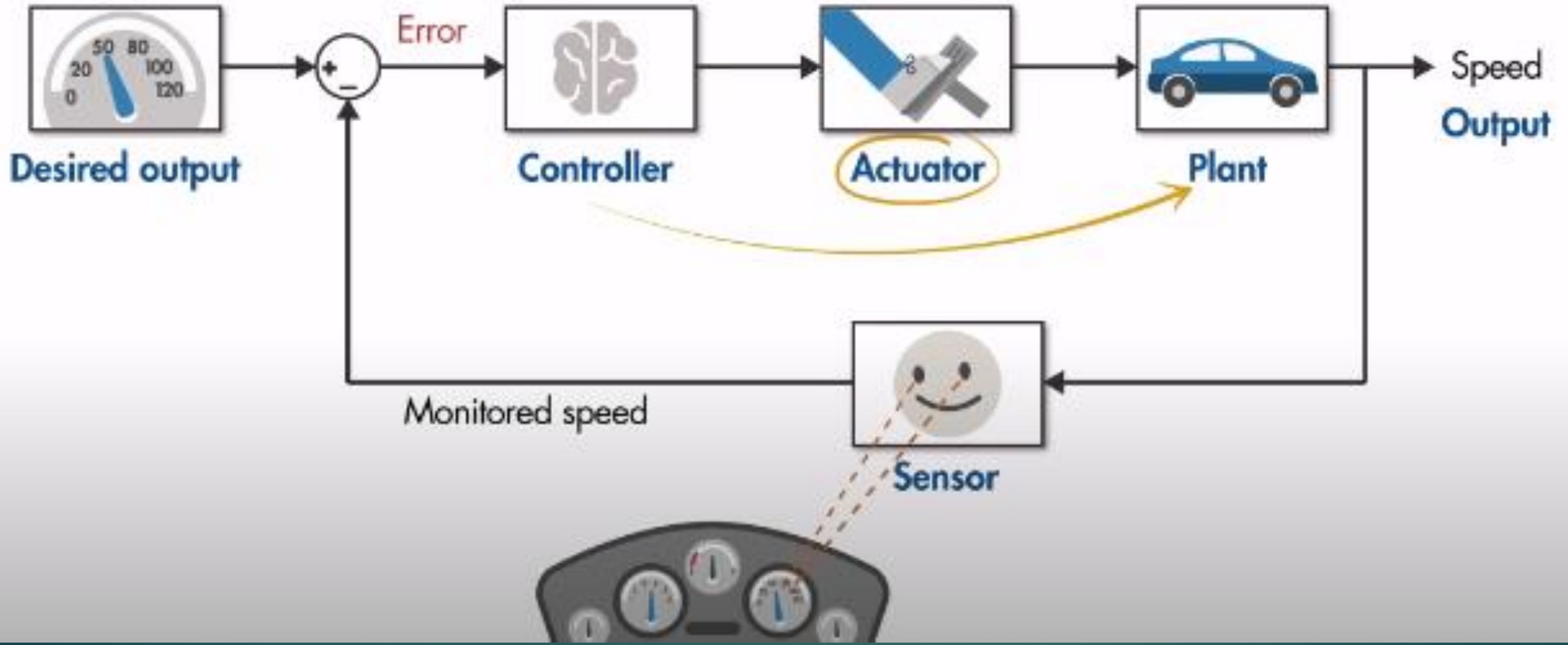
Control Systems

Types of control systems




PHILIPS
PHILIPS

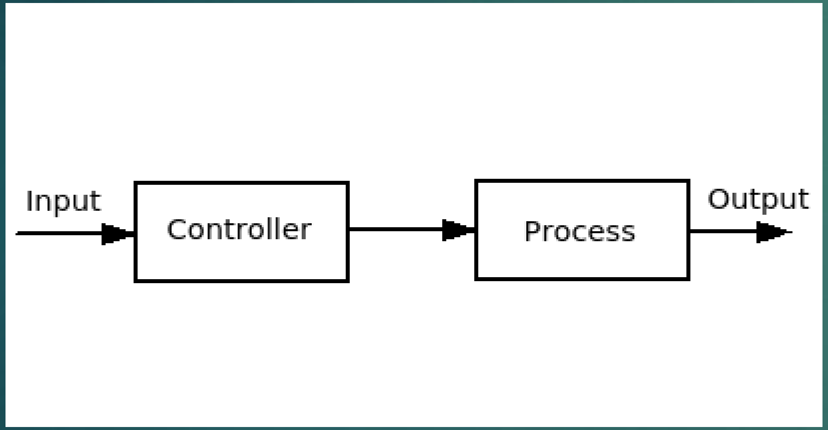




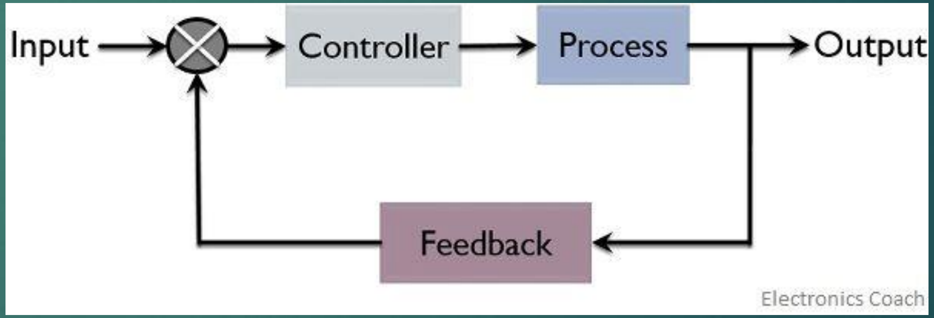
Error
Speed
Output
Desired output
Controller
Actuator
Plant
Monitored speed
Sensor

Block Diagram Comparison

Open loop



Closed loop



Controllers

Types of controllers

- continuous **controllers**
- discontinuous **controllers**.

Introduction to PID Control

- Let's take a step back… What is **control**?
- **Control** is just making a dynamic process behave in the way we want
- We need 3 things to do this:
  - A way to **influence** the process
  - A way to see how the process **behaves**
  - A way to **define** how we want it to behave

Defining behaviour

- We usually specify a **value** we want some output of the system to have
  - Usually called the **Setpoint (SP)**
- Can be the temperature of a room, the level in a tank, the flow rate in a pipe…
- The value can be fixed, or may change with time…

Influencing the process

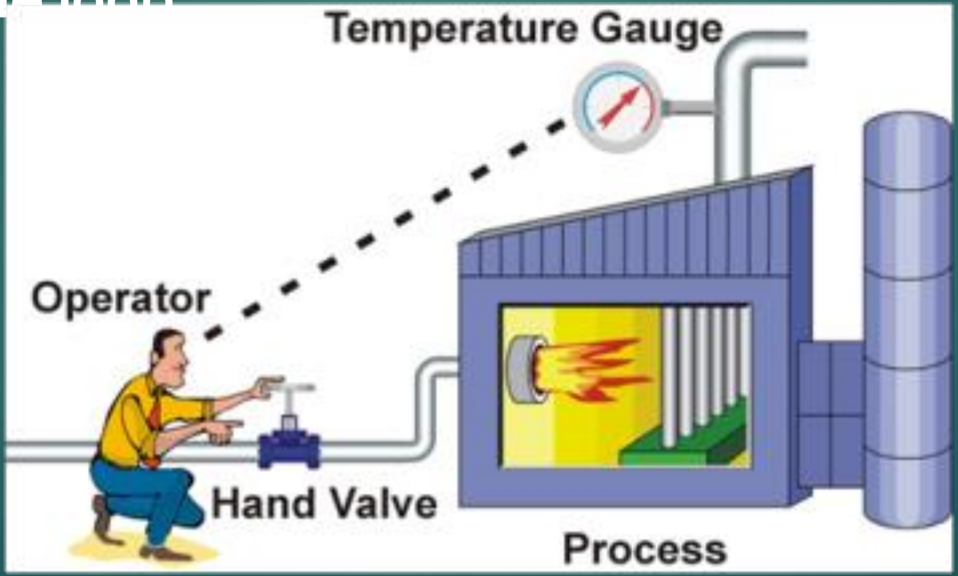
- We need some kind of **control input** which can create changes in the behavior of the process
- Can be a heater, a valve, a pump…
- Typically it is **not** the same physical quantity as what we are controlling

Observing behavior

- If we knew exactly how the process worked, we would know what the output would be for a given control input…
- Most of the time we don't know exactly, so we need to **measure** what the process does
  - Usually called **Measured Variable (MV)** or **Process Variable (PV)**

Feedback Control

- Now we have a measurement (**MV**), some value that we want it to be (**SP**), and some way to make changes to the process (**control input**)
- We can '**close the loop**'



The Controller as a System

- Now we can see that any controller can be thought of as a system that takes a **setpoint** and a **measured value** as inputs, and gives a **control** signal as an output

The Controller as a System

- The controller needs to convert two signals of one physical

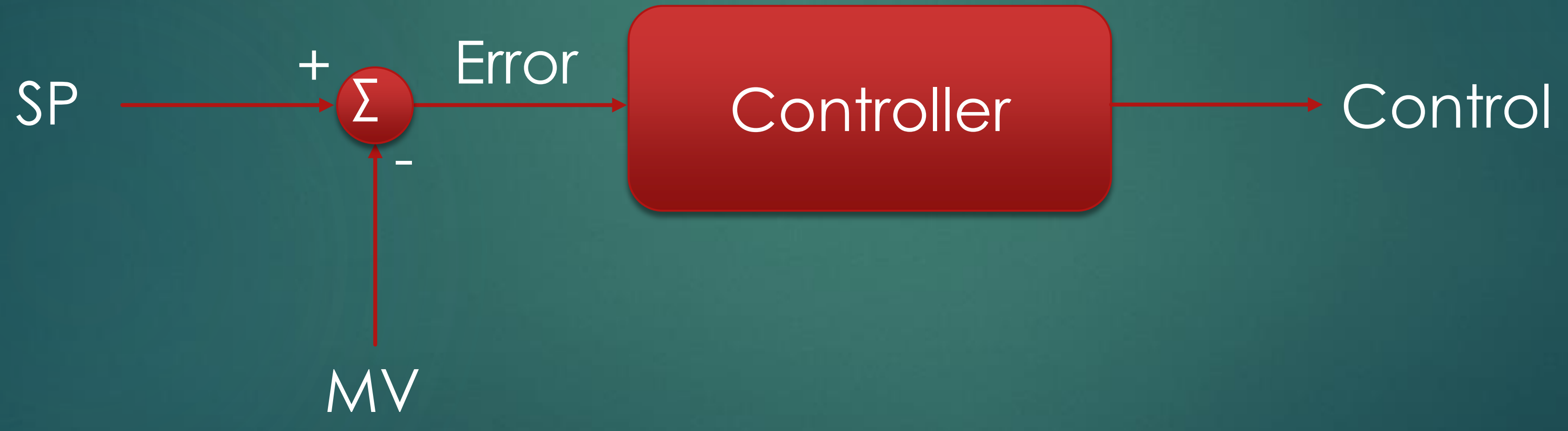
quantity (such as temperature) into one signal of another (such as valve position)

The Controller as a System

- We know that the process is a **dynamic** system:
  - Its outputs depend on **current** inputs as well as its **past** state
- For the controller to deal with this, it makes sense that it should be a **dynamic** system too

The Error Signal

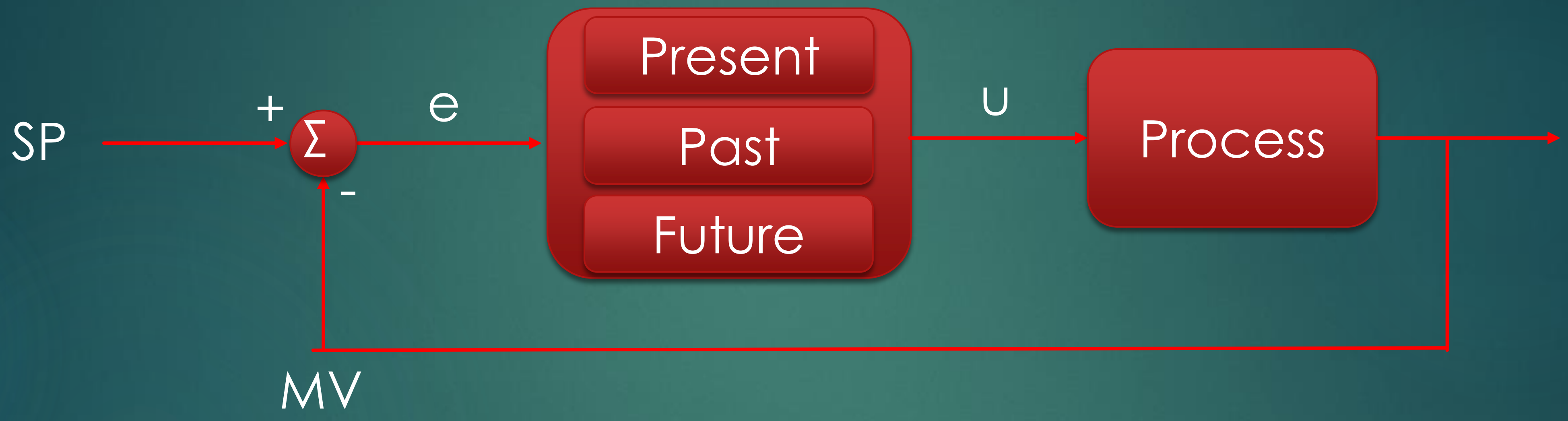
- Very often we can think of the controller acting on the **difference** between SP and MV:



A Dynamic Controller

- We said that since the process is dynamic (dependent on inputs made at different times), it makes sense that the controller should be too
- How do we usually think of time?
  - ‘Present’
  - ‘Past’
  - ‘Future’

Splitting the Controller

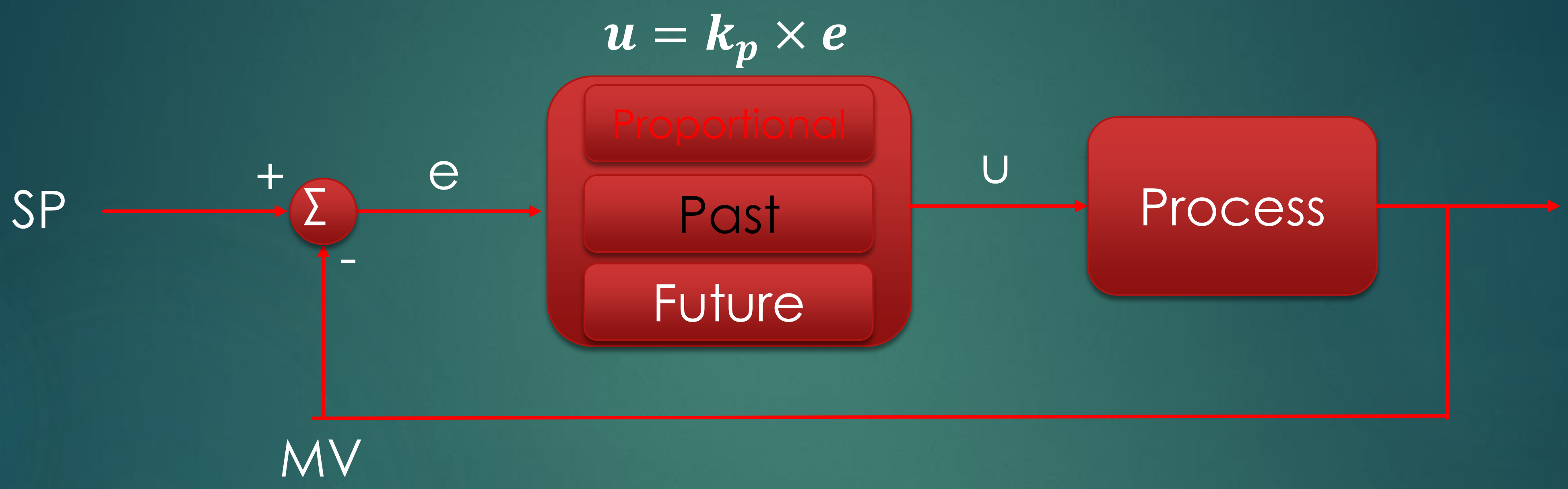


The 'Present'

- This part of the controller is only concerned with what the **error** is **now**
- Let's take a simple law: let the control signal be **proportional** to the error:

$$\boldsymbol{u} = \boldsymbol{k_p} \times \boldsymbol{e}$$

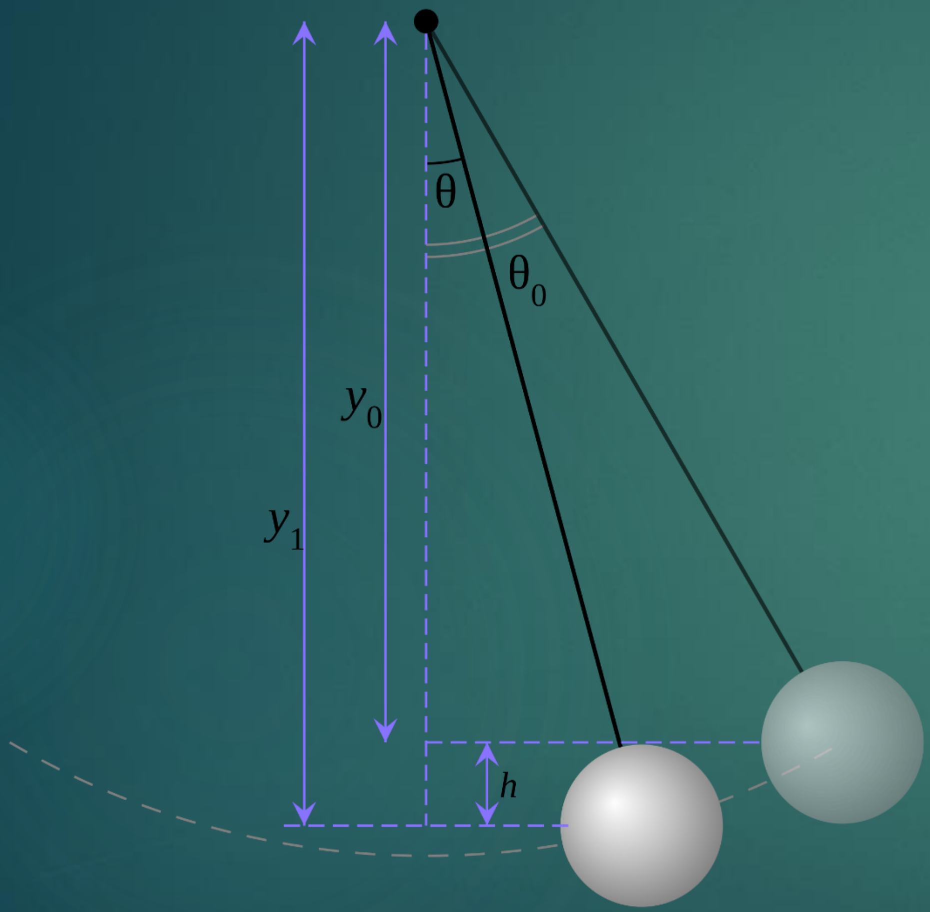
Proportional Control



Is Proportional Control enough?

- Intuitively it seems like it should be fine on its own: when the error is big, the control input is big to correct it. As the error reduced so does the control input.
- But there are problems…

Problem 1

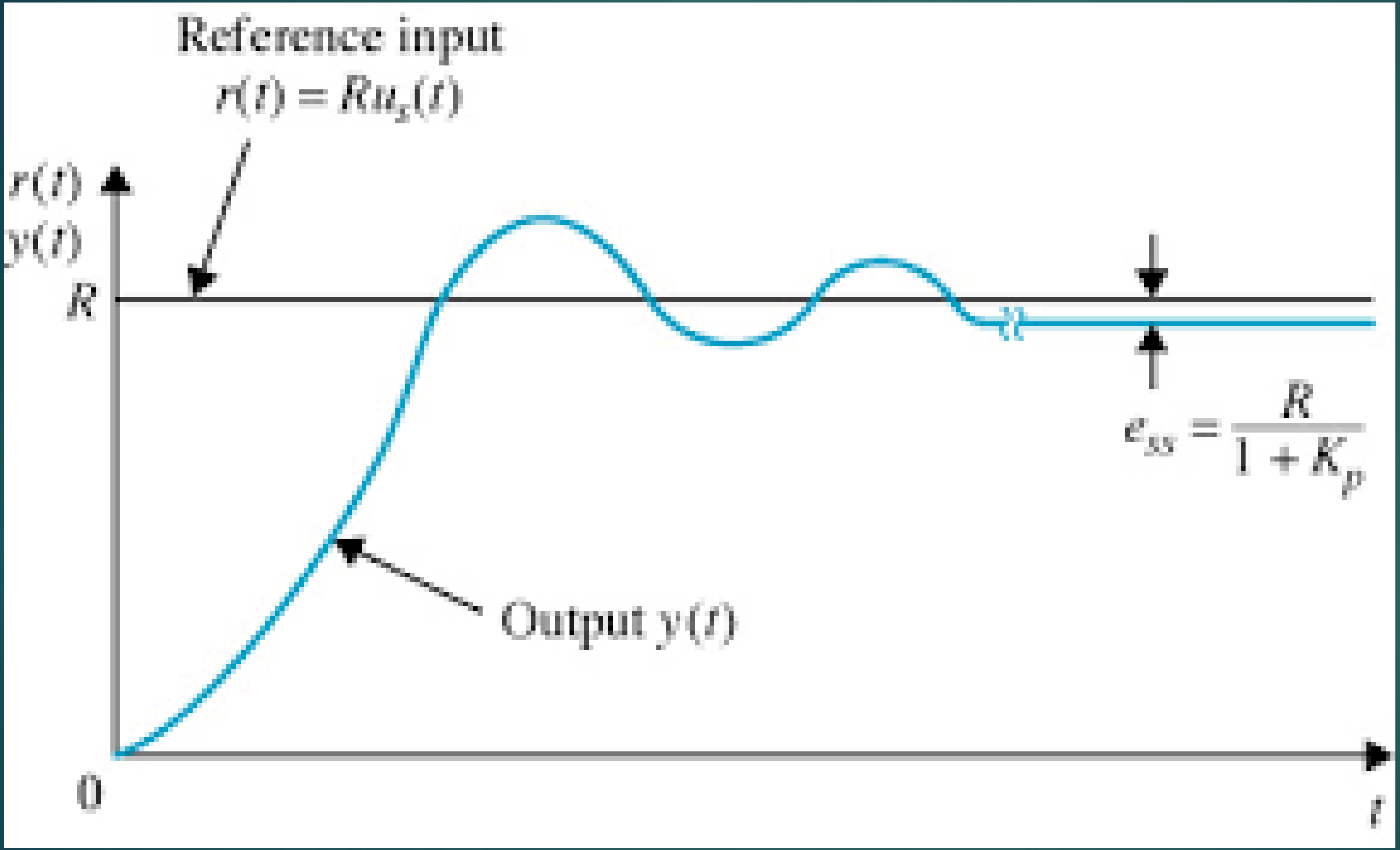


Think of a pendulum.

If the setpoint is hanging straight down, then gravity acts as a proportional controller for the position…

Pendulum will **oscillate**!

Problem 2



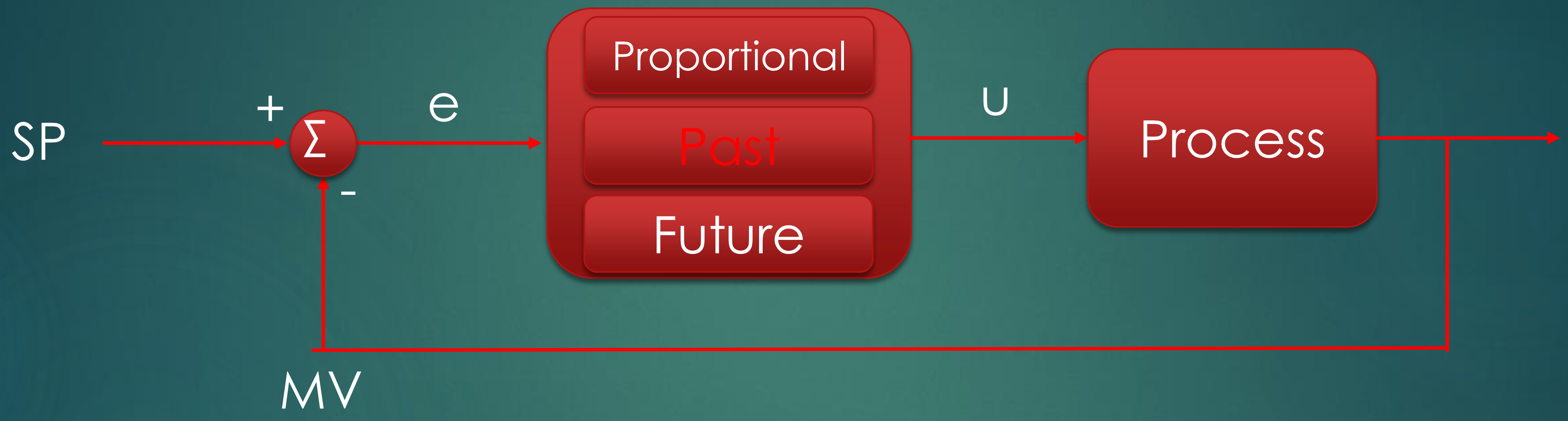
5/17/2021

P control problem summary

- Problem 1: oscillations
  - P control will give us oscillations in some processes, regardless of the value of the gain parameter.
- Problem 2: steady state error
  - P control cannot give us a nonzero value of the control at zero error for some types of process

Solving P control's problems

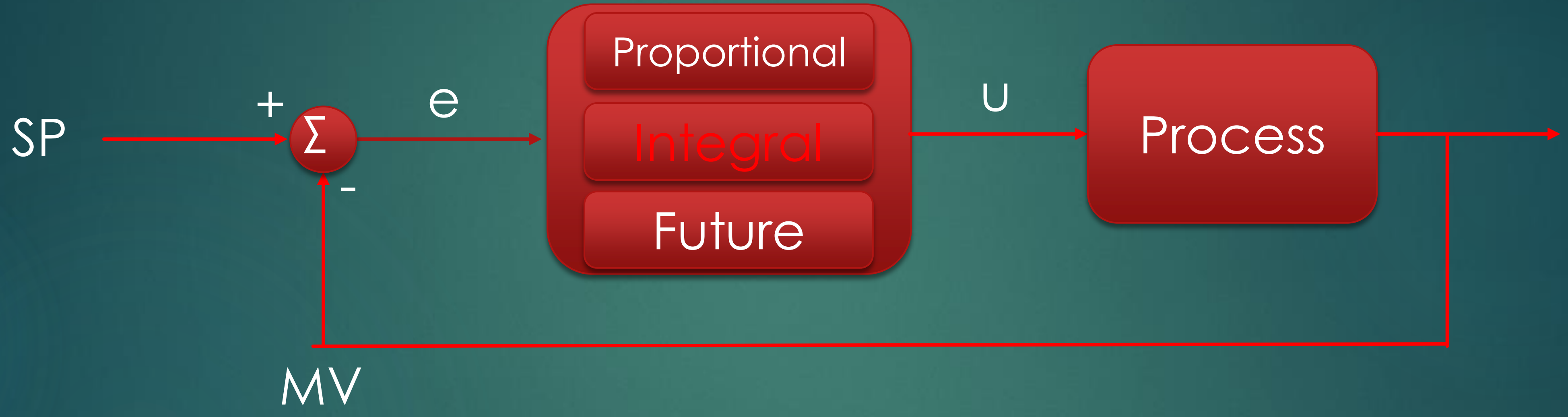
Solving steady state error: ‘the Past’



- Let’s look at the error in the **past**

Integral Action

$$u = k_i \int_0^t e(\tau)d\tau$$



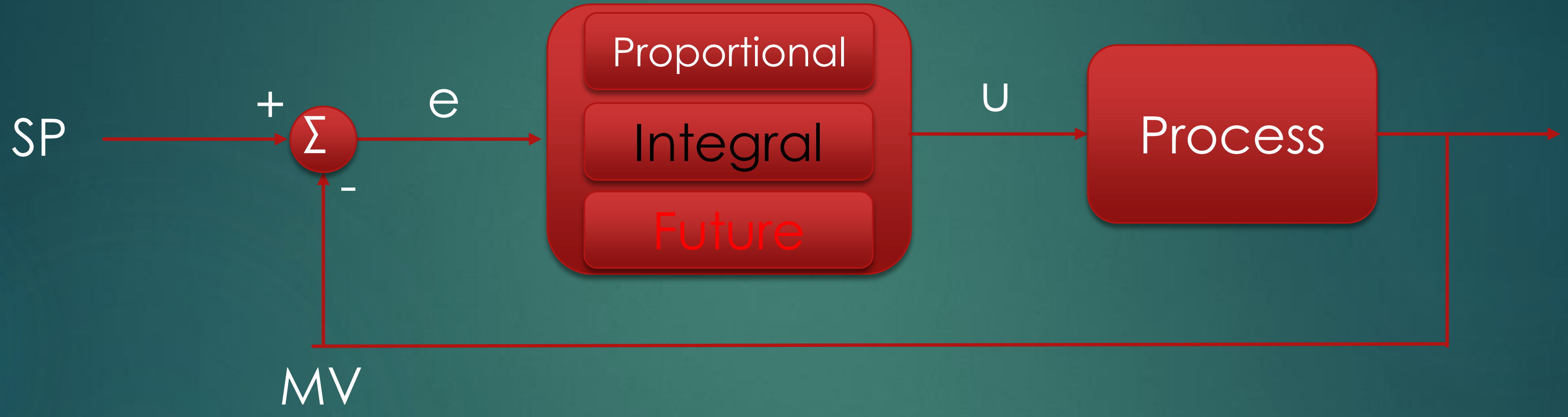
Proportional and Integral Controller

$$u = k_p e(t) + \frac{k_p}{T_i}\int_0^t e(\tau)d\tau$$

Solving P control's problems revisited

- We solved the steady state error by adding **integral action** (summing the past)
- How can we solve the oscillation problem?
- Let's look at the future!

Solving oscillations: ‘the Future’

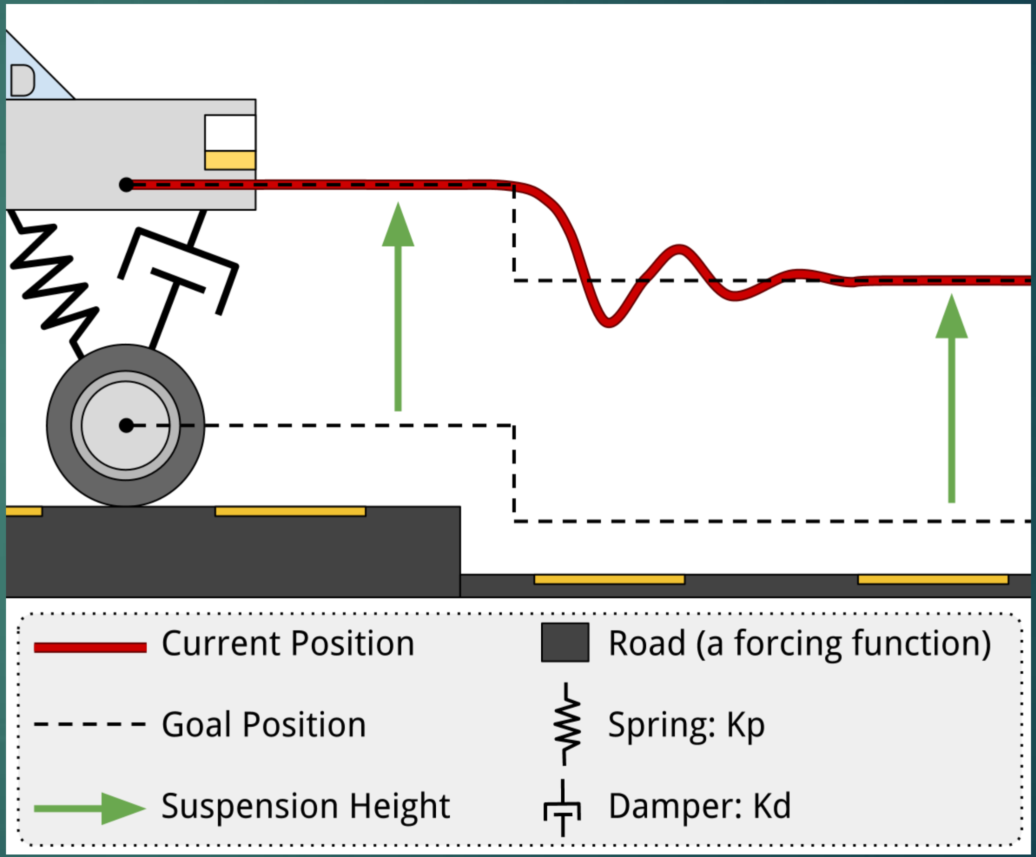


- Let’s look at the error in the **future**

Damping

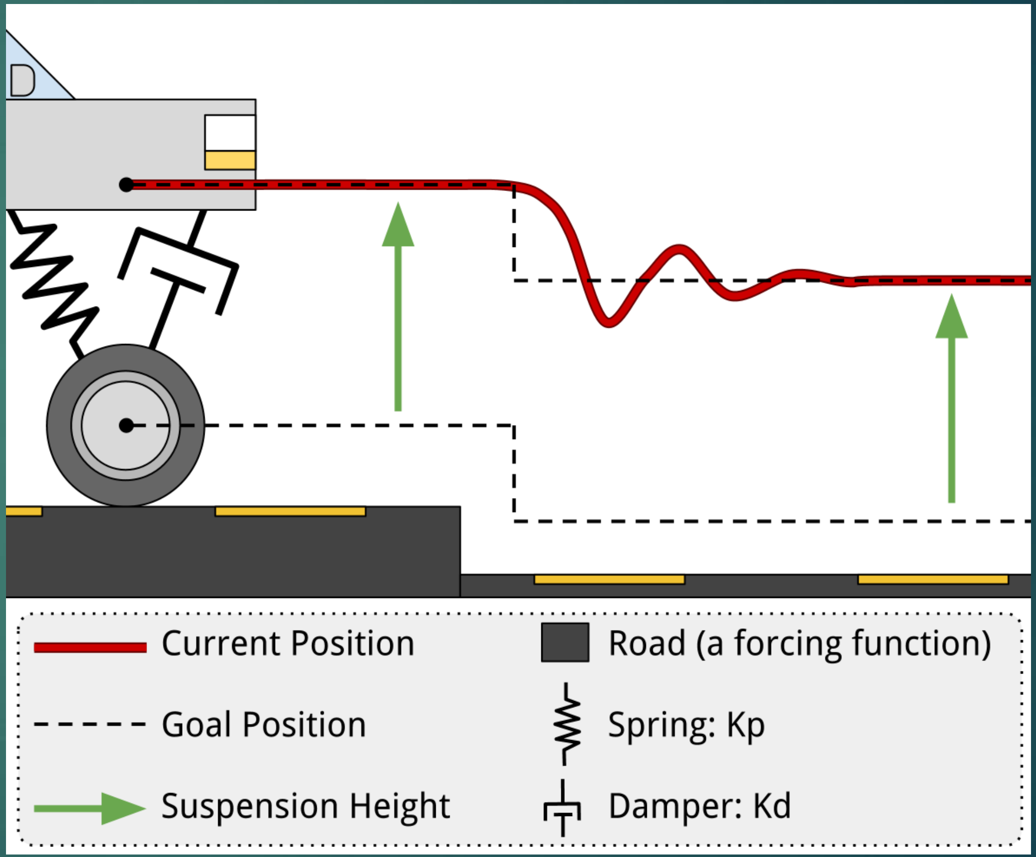
It can be easier to think of this as **damping**, something that resists velocity

Think of the wheel on your car…



Damping

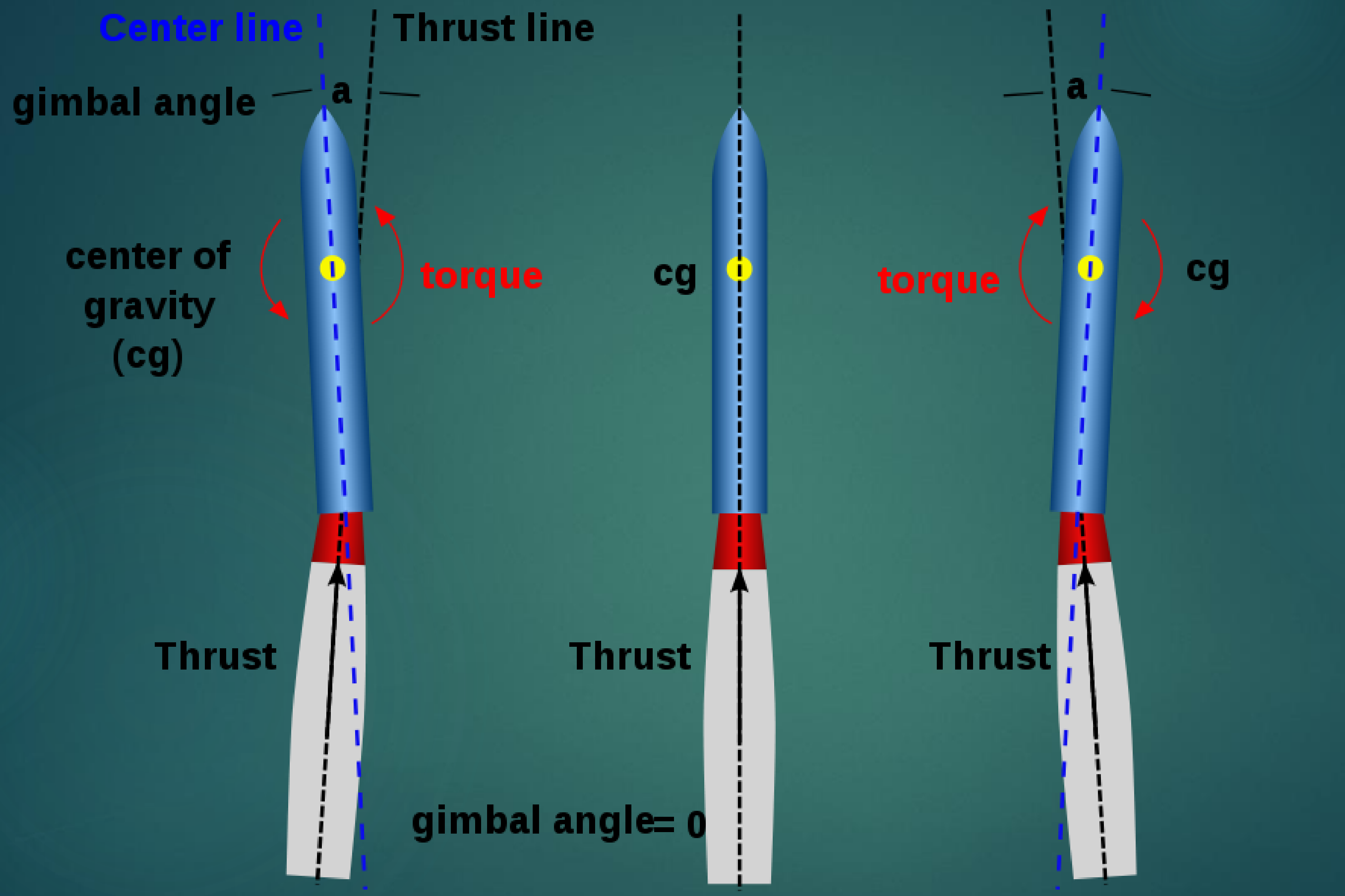
The spring is a proportional controller for the wheel position. The damper adds a **derivative action** by opposing the **velocity** of the wheel



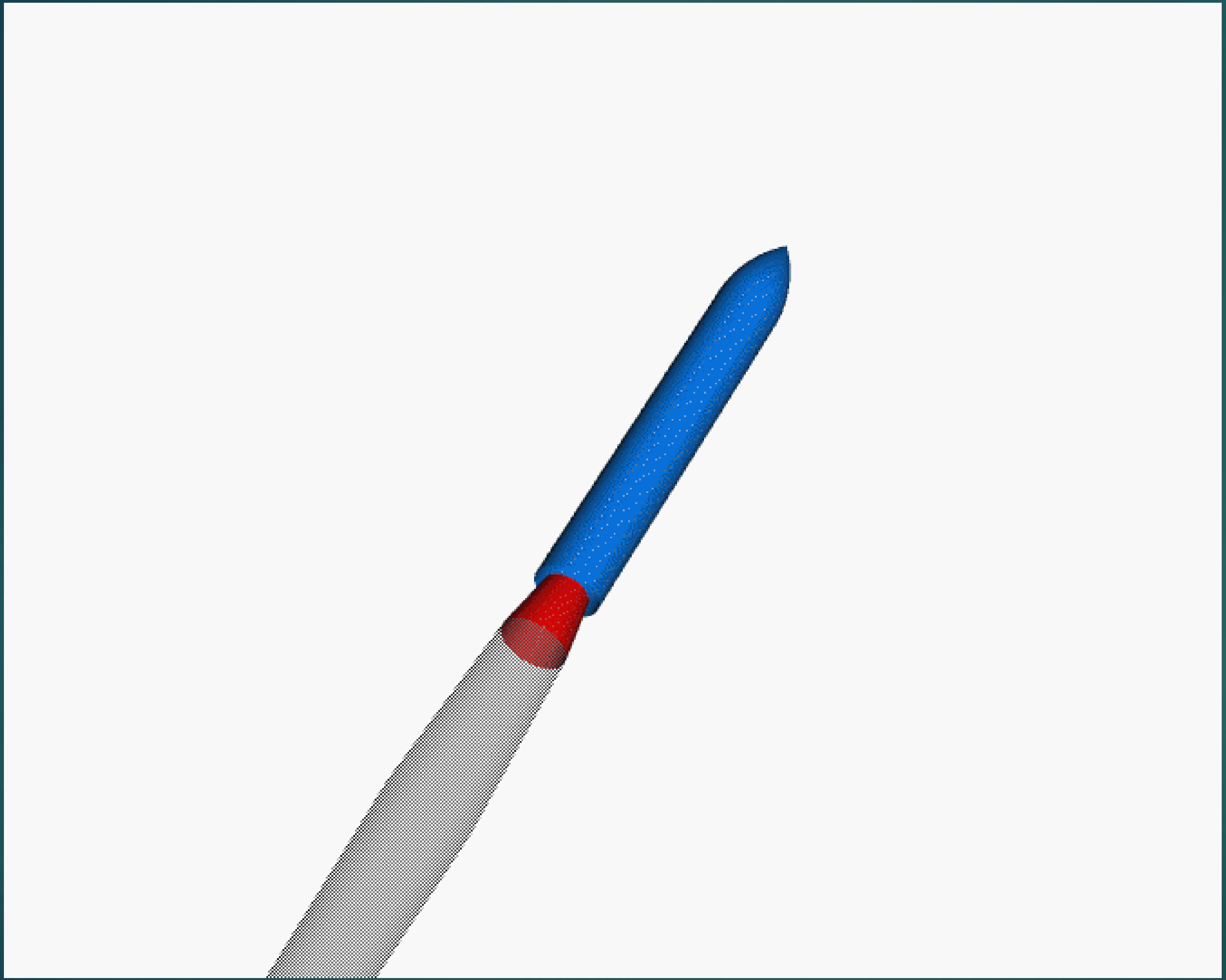
5/17/2021

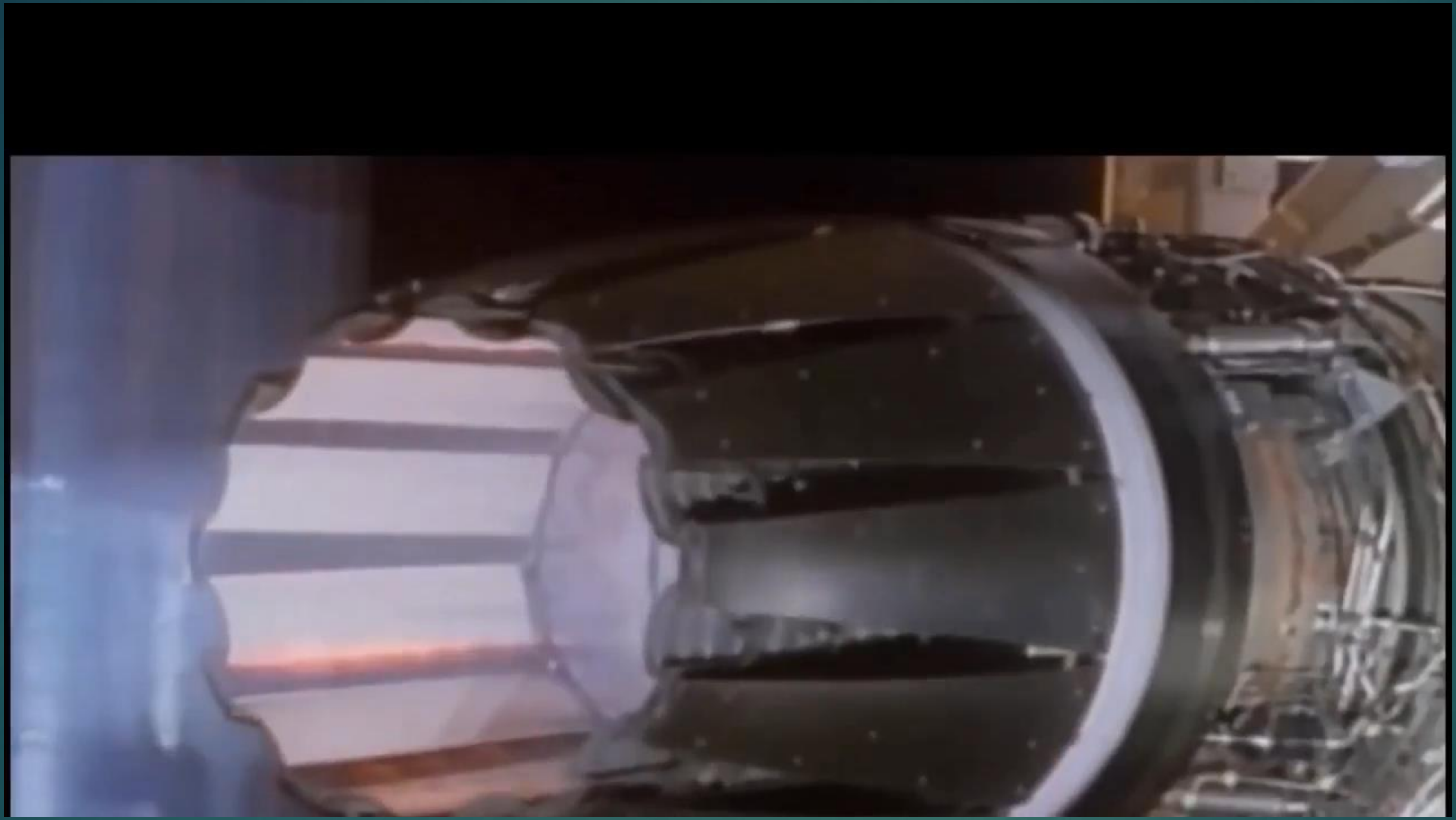
PID Controller!

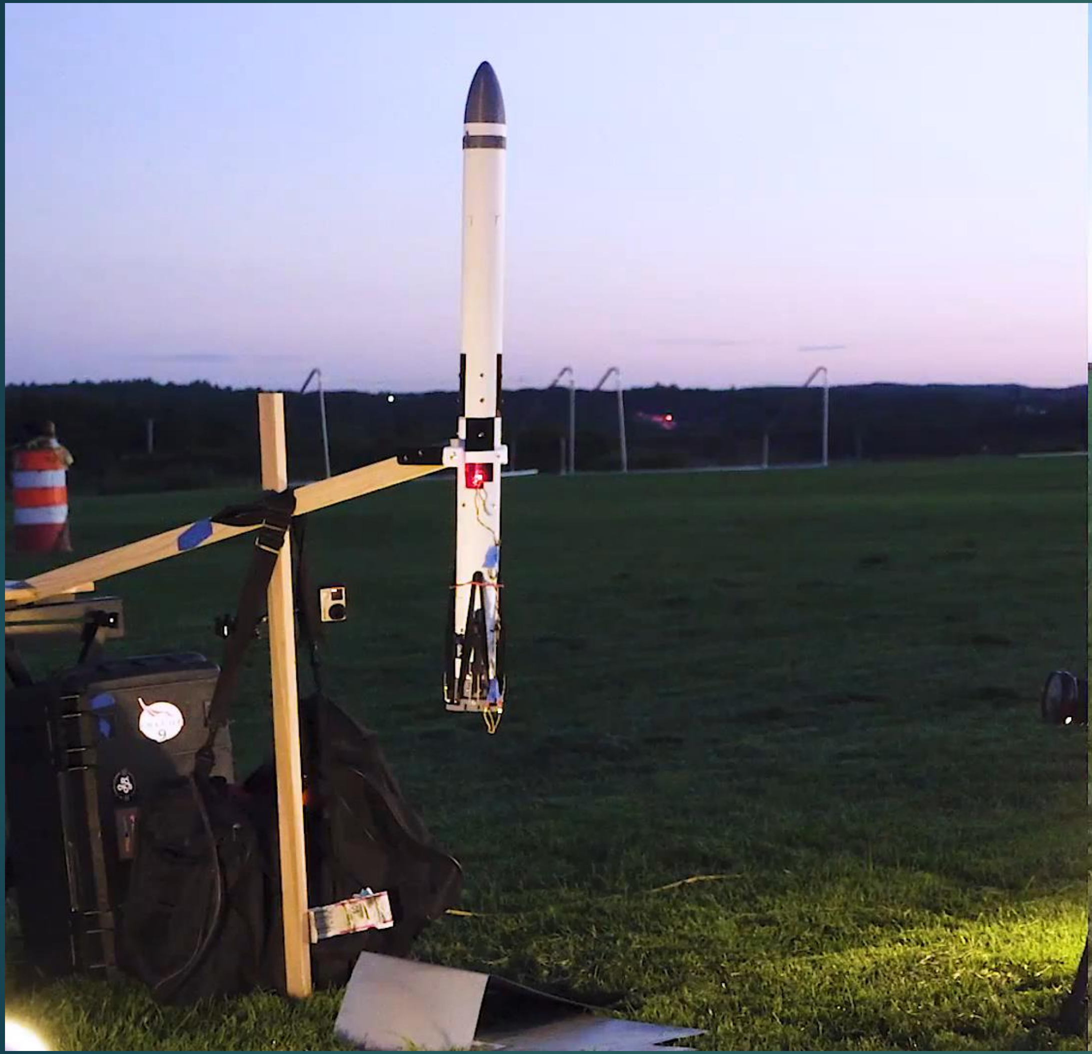
$$u = k_p e(t) + \frac{k_p}{T_i}\int_0^t e(\tau)d\tau + k_p T_d \frac{de(t)}{dt}$$

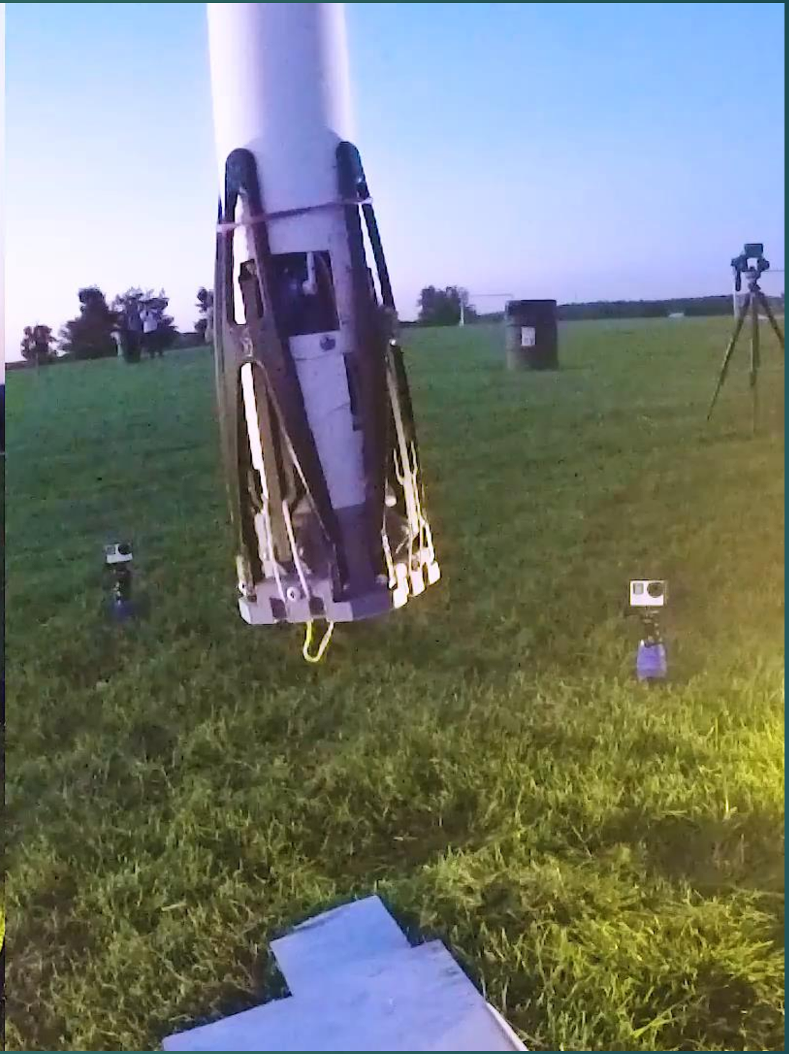


Center line
Thrust line
gimbal angle
a
center of
gravity
(cg)
torque
Thrust
cg
Thrust
gimbal angle = 0
a
torque
cg
Thrust









THE DESIGN PROCESS