اصول علم ربات – اسلاید یازدهم

Fundamentals of Robotics – Slide 11

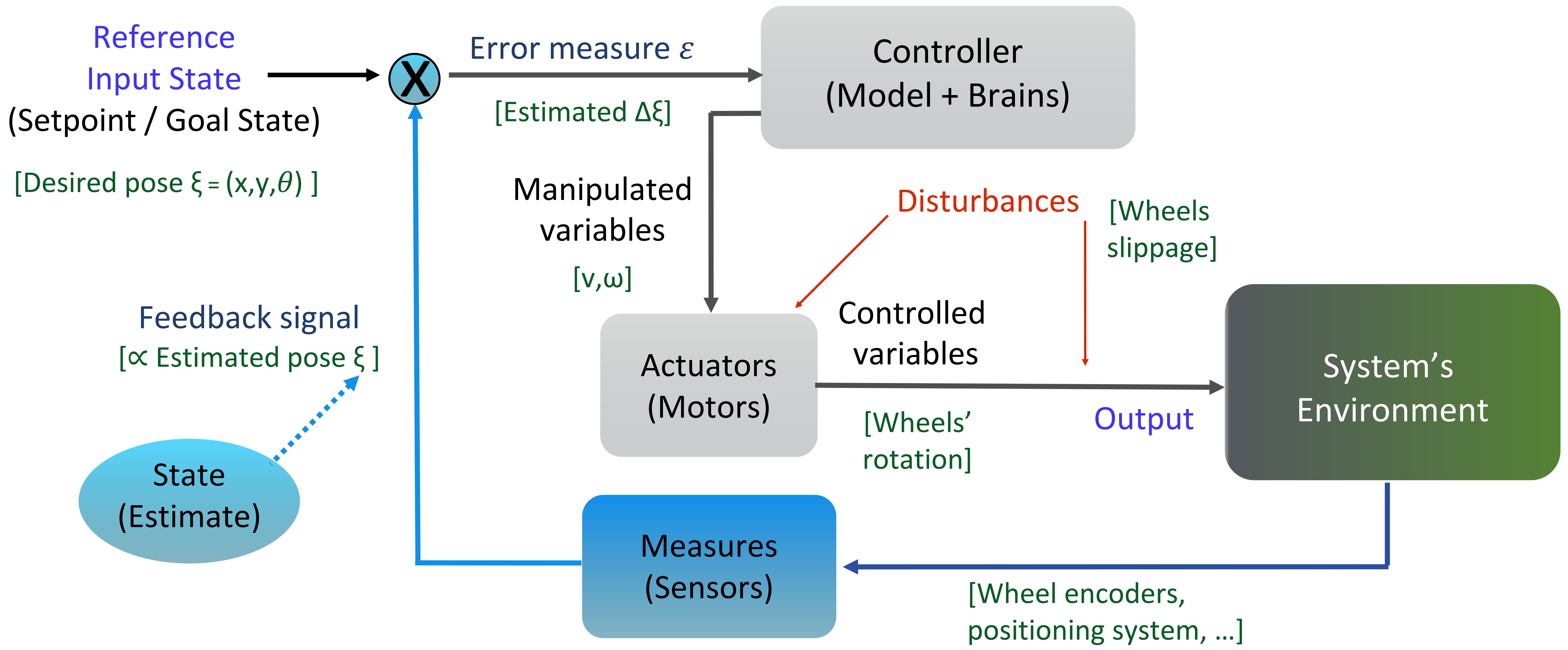
Control 1

دکتر مهدی جوانمردی

زمستان ۱۴۰۰ – بهار ۱۴۰۱

[slides adapted from Gianni Di Caro, @CMU with permission]

Closed-loop vs. Open-loop control



Closed-loop: status information is fed back to the controller to evaluate the difference between the desired setpoint (goal) and the actual output, and to implement corrective actions, if needed

Open-loop: feedback information is not used to implement corrective actions, the assumption is that, given the inputs, the desired results / goals will be achieved

Types of goal states

A robot is a *goal-driven* physical entity

Achievement goals (typical of AI):

States the system tries to reach and once reached, the job is done

Exit from a maze,
reach a specific location or pose,
complete a construction

Maintenance goals (typical of Control):

Require a continual active work/tracking

Keep balance for a bipedal robot,
keep following a wall,
keep tracking a moving target

External goal states

Get to the kitchen
Balance a pole
Find a treasure (!)

Internal goal states

Keep battery levels in some range
Avoid excessive torque on the effectors

Types of feedback-based controllers

The goal of any control system is to minimize the **Error**:
the difference between the current (as measured) state and the desired goal state

The adopted representation of the error

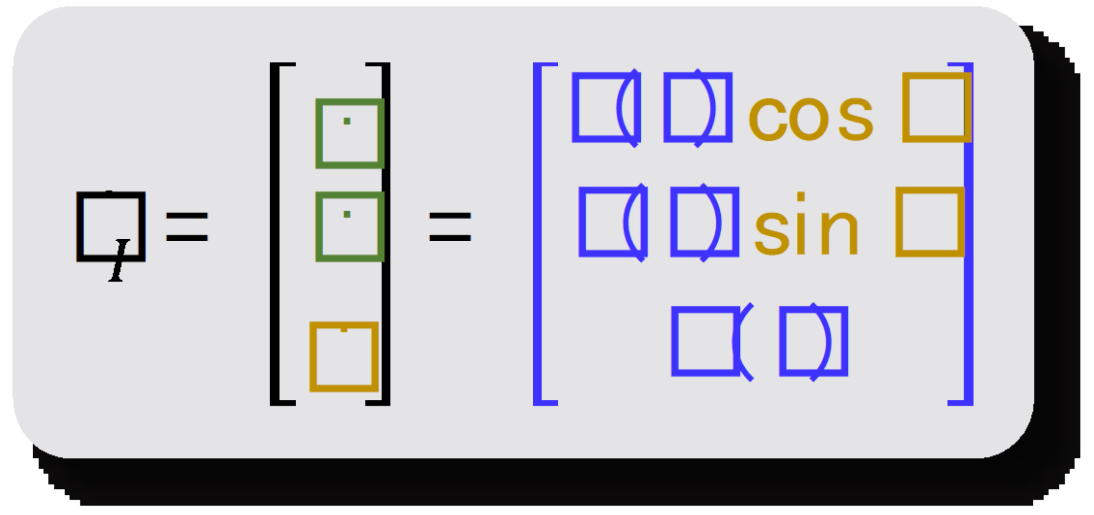
- has a magnitude
- has a direction
- has a scale, interval
- can be quantitative: continuous, discrete, binary
- can be ordinal: ranking, binary

The different ways the error is represented and is treated give raise to a number of different frameworks for control, we will focus on the case of quantitative errors in the context of PID controllers

Dynamical systems and controllers

- **Dynamical system**: a system whose state descriptor $\boldsymbol{x} \in \mathbb{R}^n$ changes *continuously* (almost always) according to a law

$$\frac{d\mathbf{x}}{dt} = \dot{\mathbf{x}} = \boldsymbol{F}(\mathbf{x})$$



- A **controller** is defined to change the coupled robot and environment system into a dynamical system showing the desired behavior

$\dot{\mathbf{x}} = F(\mathbf{x},\mathbf{u})$ Dynamical system coupled with controls

$\mathbf{y} = G(\mathbf{x})$ Observed values of the state

$\mathbf{u} = H_i(\mathbf{y})$ Controls, depending on observed values of the state

$$\dot{\mathbf{x}} = F(\mathbf{x},H_i(G(\mathbf{x})))$$

$$\dot{\mathbf{x}} = \Phi(\mathbf{x})$$

Dynamical systems and Types of controllers

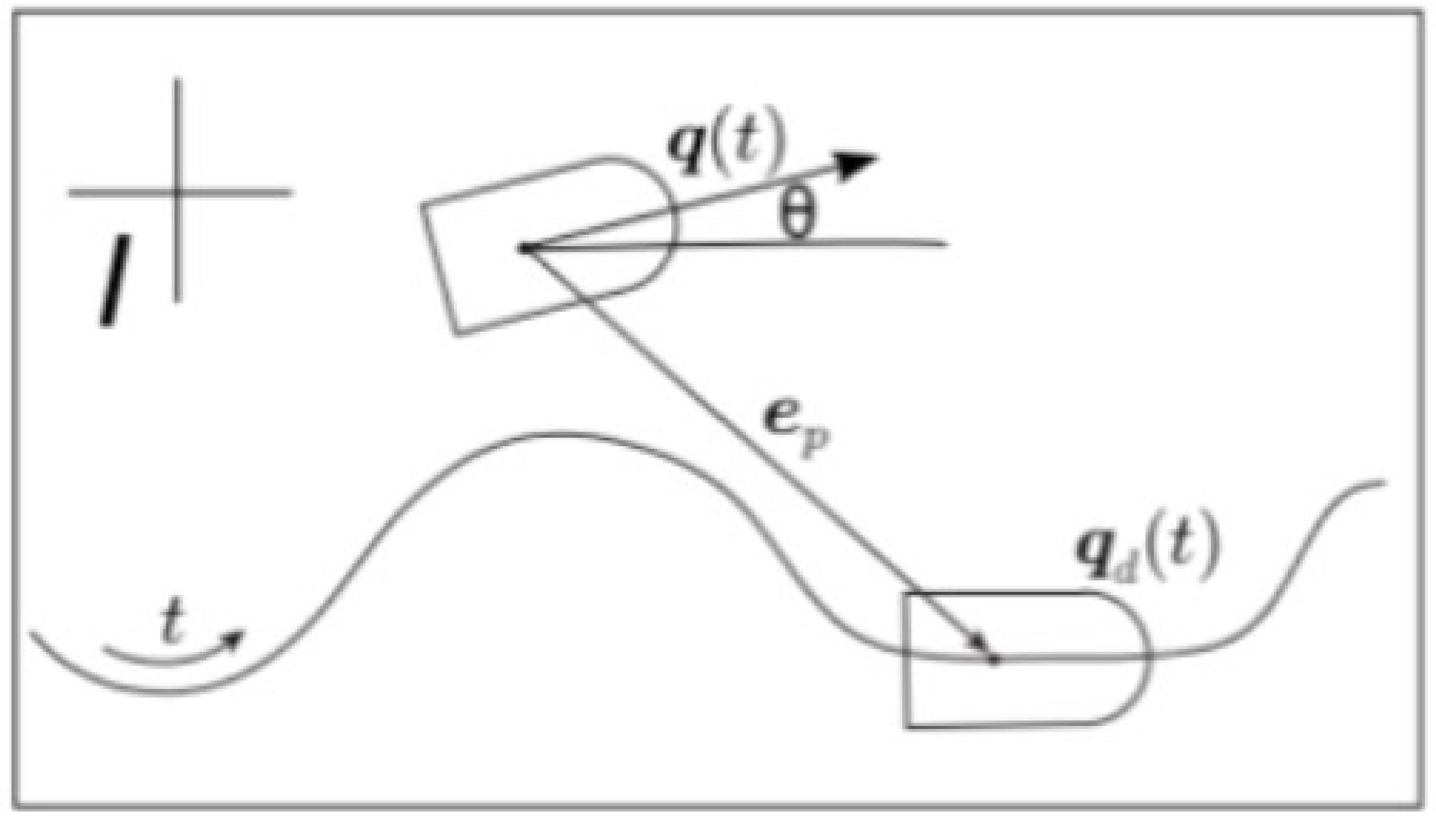
$$\dot{\mathbf{x}} = F(\mathbf{x},\mathbf{u})$$

$$\mathbf{y} = G(\mathbf{x})$$

$$\mathbf{u} = H_i(\mathbf{y})$$

- Open-loop control: No sensing
- Feedback control (closed-loop): Sense error, determine control response.
- Feedforward control (closed-loop): Sense disturbance, predict resulting error, respond to predicted error before it happens.
- Model-predictive control (closed-loop): Plan trajectory to reach goal; Take first step; Repeat.

Example: trajectory tracking



- The tracking error vector e is conveniently expressed in terms of its *projections on the rotated reference frame of the robot wrt to the inertial frame.* In this way the positional part of the error is the Cartesian component of the error expressed in a reference frame aligned with the current orientation of the robot:

$$e = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{bmatrix} = \begin{bmatrix} (x_d - x)\cos(\theta) + (y_d - y)\sin(\theta) \\ -(x_d - x)\sin(\theta) + (y_d - y)\cos(\theta) \\ \theta_d - \theta \end{bmatrix}$$

- Differentiating wrt time and using kinematic equations for expressing $x(t), y(t), \theta(t), x_d(t), y_d(t), \theta_d(t)$, the error dynamics becomes:

$$\dot{e}_1 = v_d \cos(e_3) - v + e_2\omega$$
$$\dot{e}_2 = v_d \sin(e_3) - e_1\omega$$
$$\dot{e}_3 = \omega_d - \omega$$

- Find $v(t), \omega(t)$ control laws that take the error steadily to zero over the entire trajectory

Controlling a simple system

- Consider a simple system: $\dot{x} = F(x,u)$
  - Scalar variables x and u, not vectors $\mathbf{x}$ and $\mathbf{u}$.
  - Assume x is observable: $y = G(x) = x$
  - Assume effect of motor command u: $\frac{\partial F}{\partial u} > 0$
- The setpoint x_{set} is the desired value.
  - The controller responds to error: $e = x - x_{set}$
- The goal is to set u to reach $e = 0$.
- Use action u to push back toward error $e = 0$
  - error e depends on state x (via sensors y)

- What does pushing back do?
  - Depends on the structure of the system
  - Velocity versus acceleration control

- How much should we push back?
  - What does the magnitude of u depend on?

Velocity or acceleration control?

- If error reflects $\mathbf{x}$, does $\mathbf{u}$ affect $\mathbf{x}'$ or $\mathbf{x}''$?

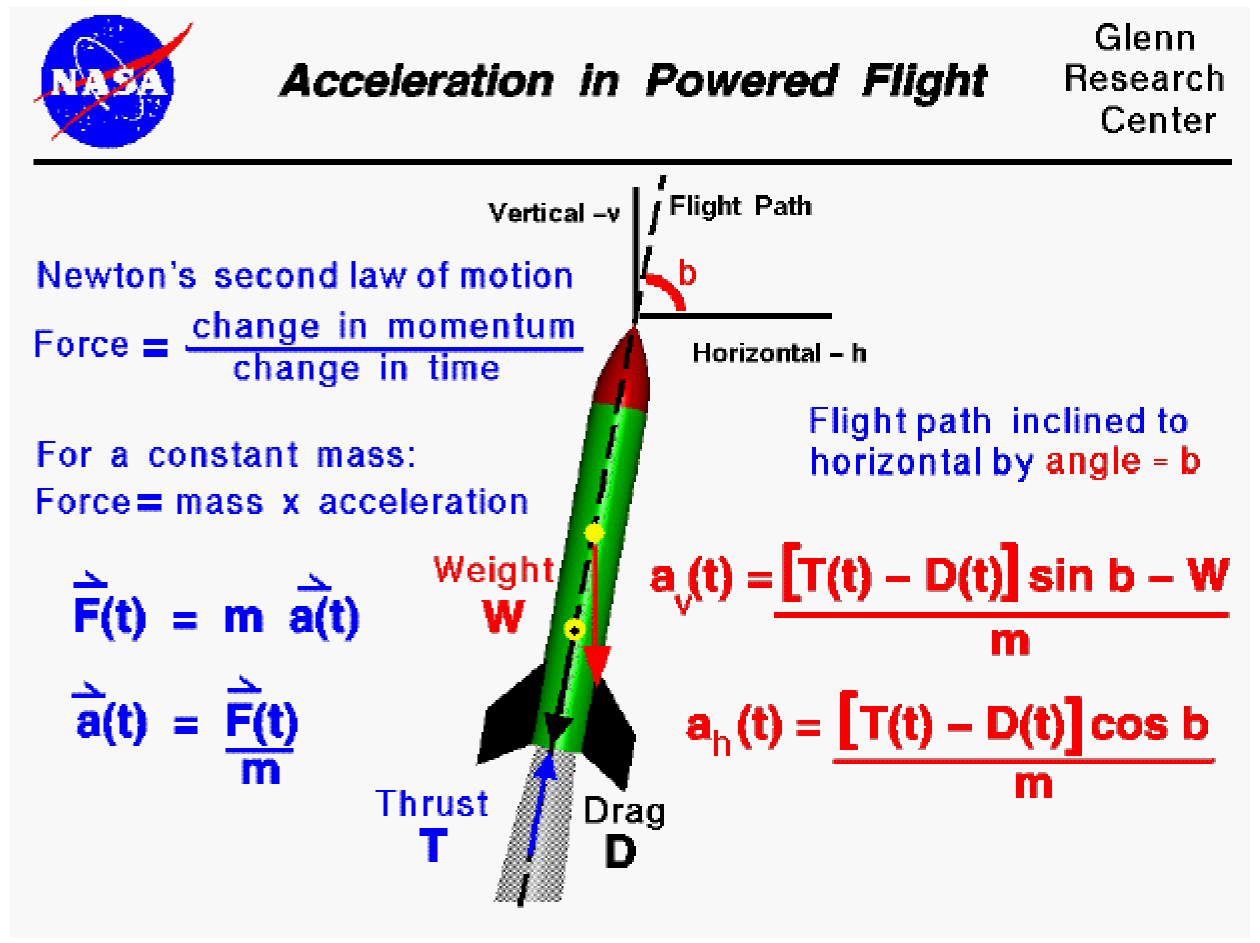
- Velocity control: $\mathbf{u} \to \mathbf{x}'$ (valve fills tank)
  - let $\mathbf{x} = (x)$

$$\dot{\mathbf{x}} = (\dot{x}) = F(\mathbf{x}, \mathbf{u}) = (u)$$

- Acceleration control: $\mathbf{u} \to \mathbf{x}''$ (rocket)
  - let $\mathbf{x} = (x\ v)^T$

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} = F(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} v \\ u \end{pmatrix}$$

$$\dot{v} = \ddot{x} = u$$



NASA
Acceleration in Powered Flight
Glenn Research Center
Vertical – v
Flight Path
b
Horizontal – h
Newton's second law of motion
Force = change in momentum / change in time
For a constant mass:
Force = mass x acceleration
Flight path inclined to horizontal by angle = b
$\vec{F}(t) = m\ \vec{a}(t)$
$\vec{a}(t) = \vec{F}(t) / m$
Weight W
$a_v(t) = \frac{[T(t) - D(t)] \sin b - W}{m}$
$a_h(t) = \frac{[T(t) - D(t)] \cos b}{m}$
Thrust T
Drag D

Bang-Bang control

- Push back, against the *direction* of the error
  - with constant action u
- Error is $e = x - x_{set}$

$$e < 0 \quad \Rightarrow \quad u := on \quad \Rightarrow \quad \dot{x} = F(x, on) > 0$$

$$e > 0 \quad \Rightarrow \quad u := off \quad \Rightarrow \quad \dot{x} = F(x, off) < 0$$

- For implementing heating up & cooling down, the on/off switch must be replaced by a fixed signal of opposite sign/effect (e.g., amount of electrical power G used to heat up / cool down)

It can be as basic an on / off switch for control:

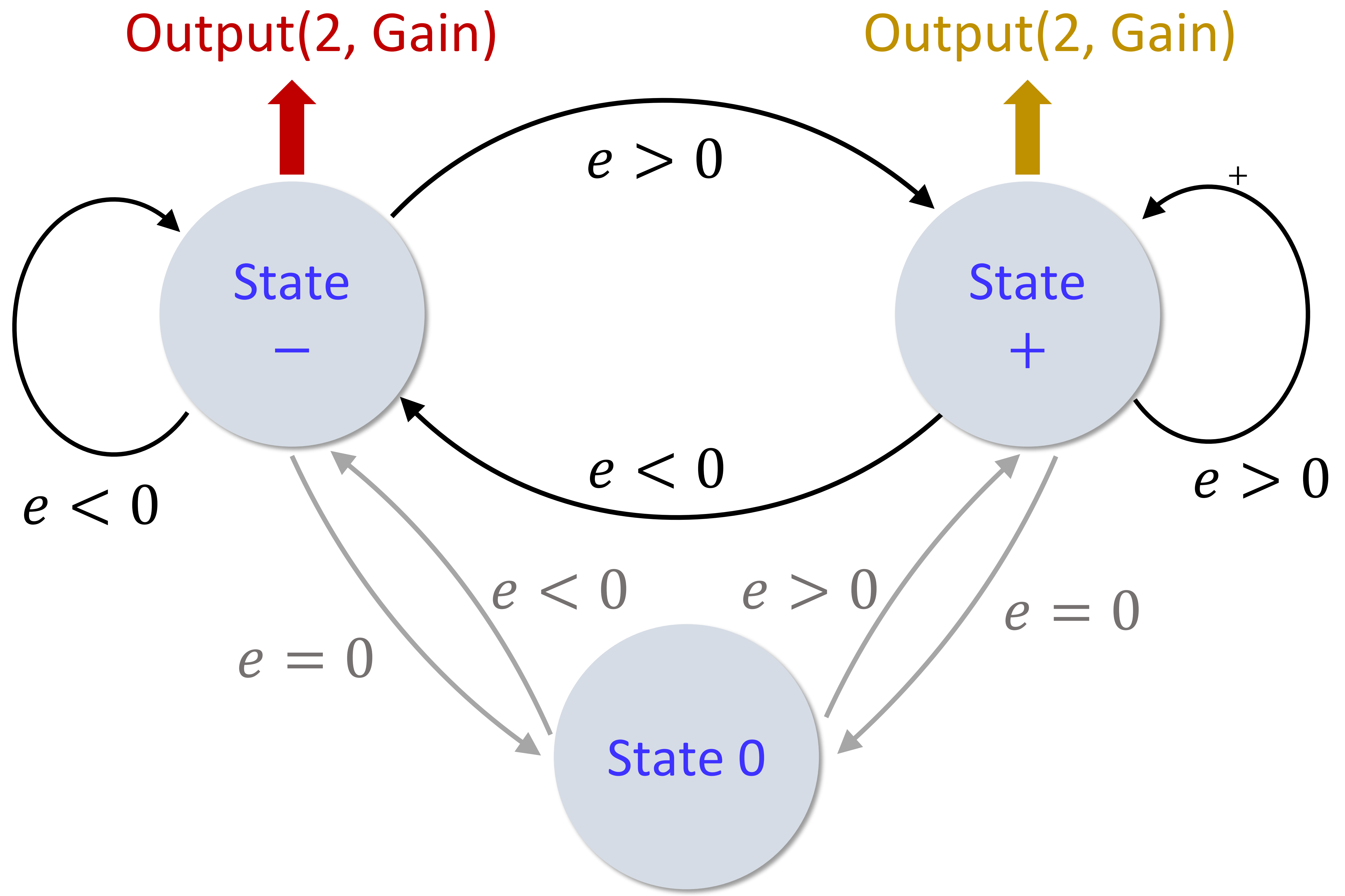
- Send a fixed control signal G when $e < 0$,
- Don't do any control actions when $e \geq 0$
- Or vice versa

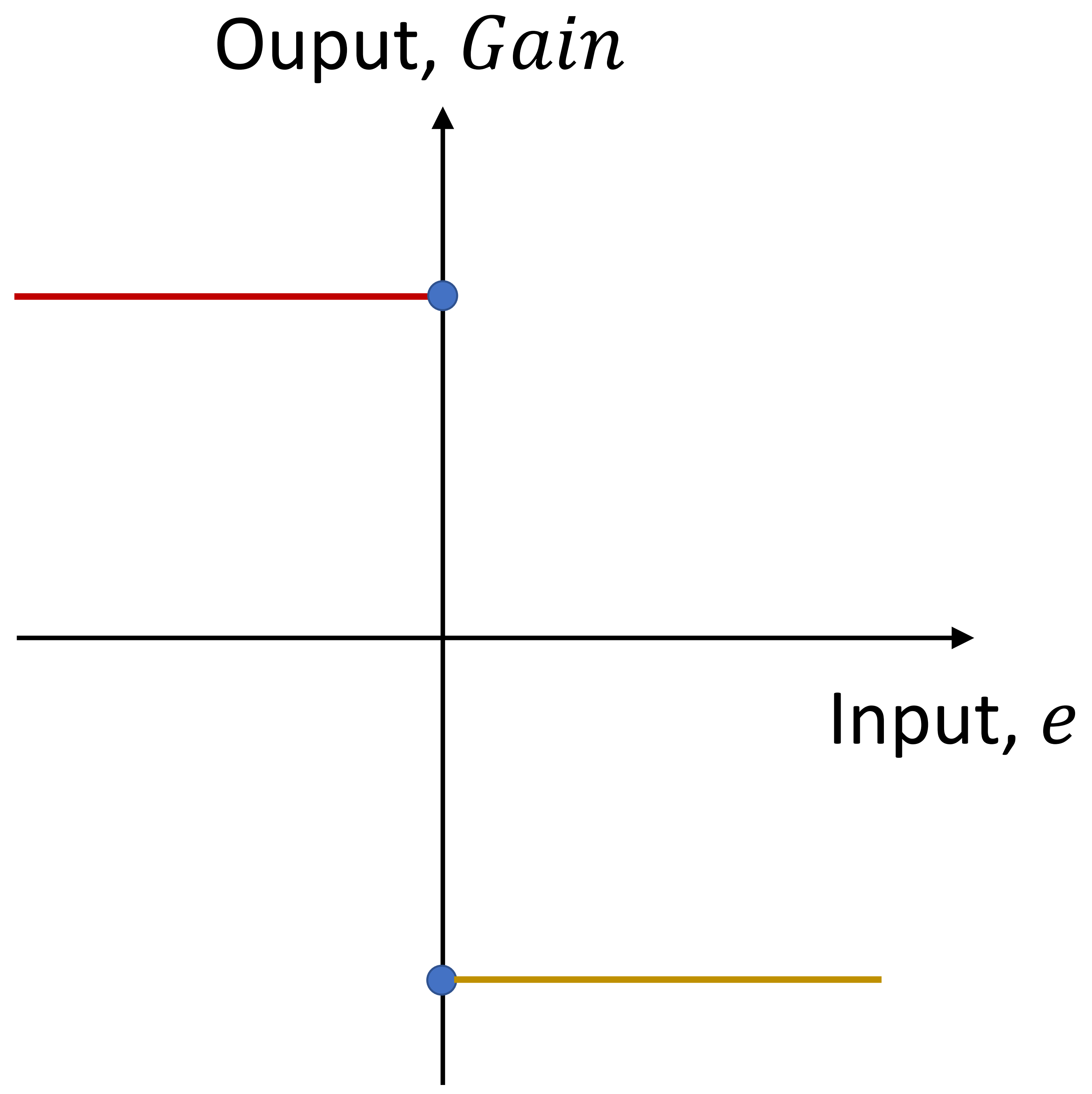
- E.g., a thermostat in wintertime: heat up when temperature is below the setpoint, do nothing when temperature is above the setpoint

Emko ESM-3710-N Bang-bang Temperature controller PTC -50 up to 130 °C 16 A relay (L x W x H) 65 x 76 x 35 mm

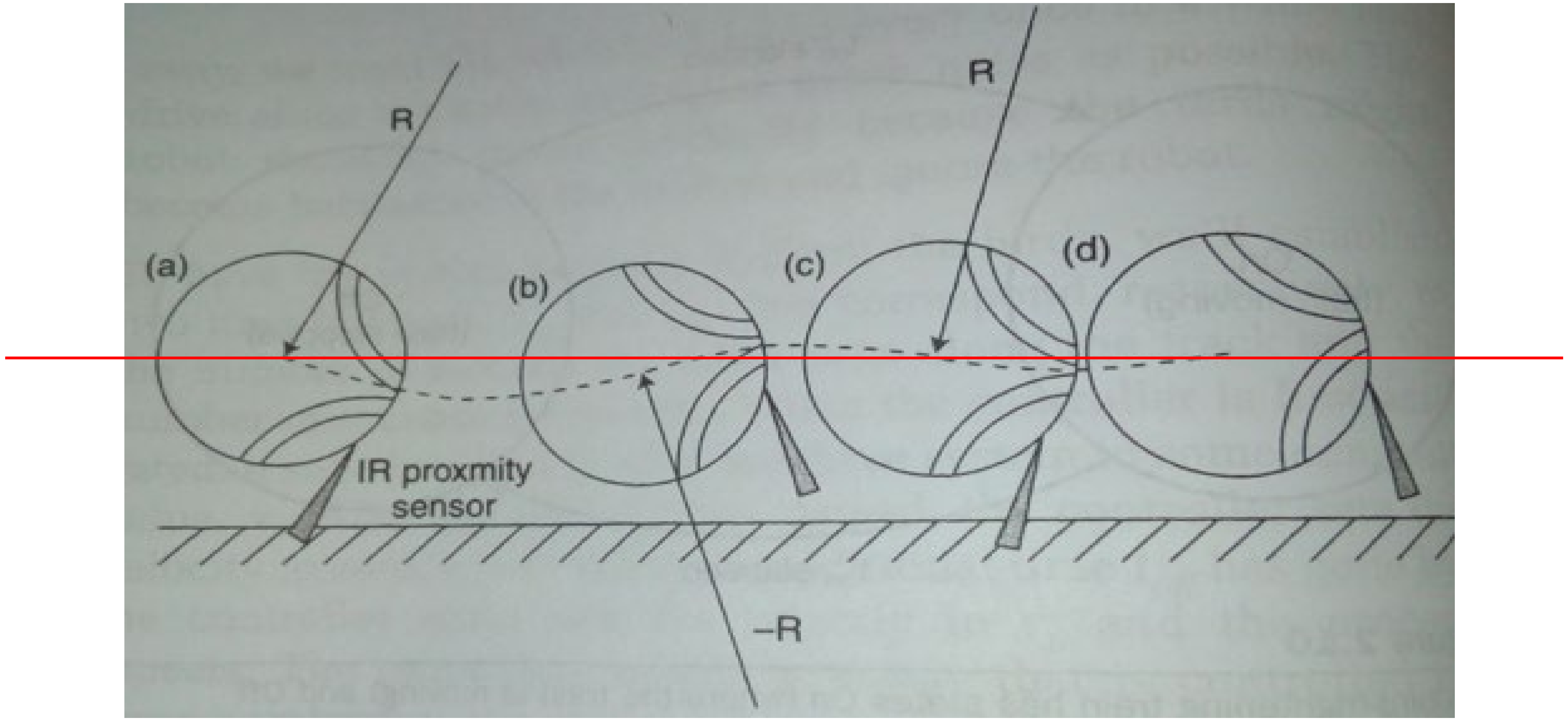
Bang-Bang control

- More in general, a bang-bang controller is equivalent to a two-state system or, more precisely, to a three-state system, where one of the states is neutral (no controls)
- State transitions happen when the error changes direction or passes from zero to non-zero and vice versa
- Action at each state depends on a fixed gain parameter G





Bang-Bang control for wall following



Setpoint: distance from wall

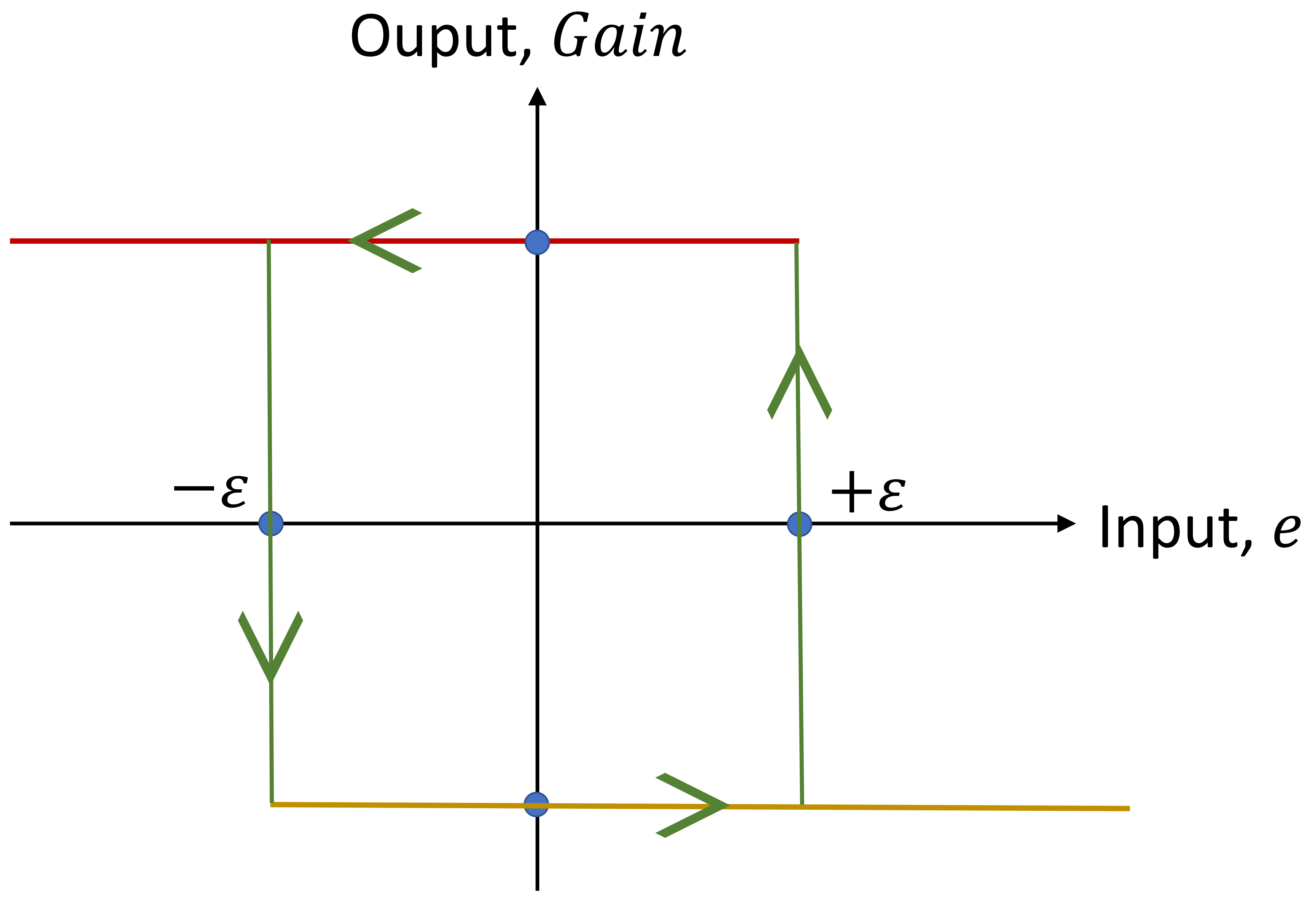
Bang-Bang control with Hysteresis

- To prevent chatter around $e = 0$,

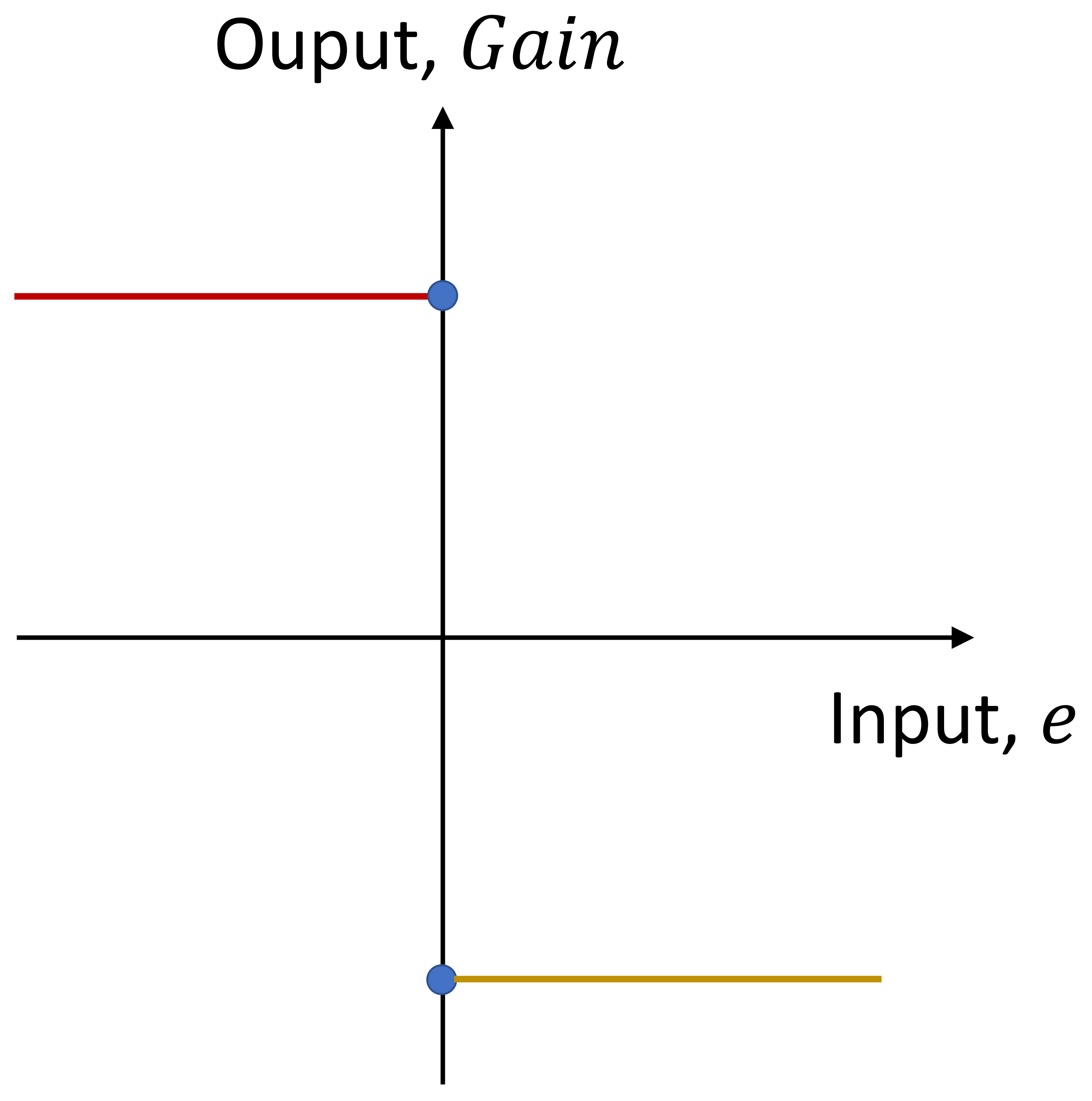
  $e < -\varepsilon \quad \Rightarrow \quad u := +\text{Gain (on)}$

  $e > +\varepsilon \quad \Rightarrow \quad u := -\text{Gain (off)}$

Bang–bang controls with hysteresis provide **optimal controls** in some cases, although they are often implemented just because of their simplicity or when binary behaviors are required

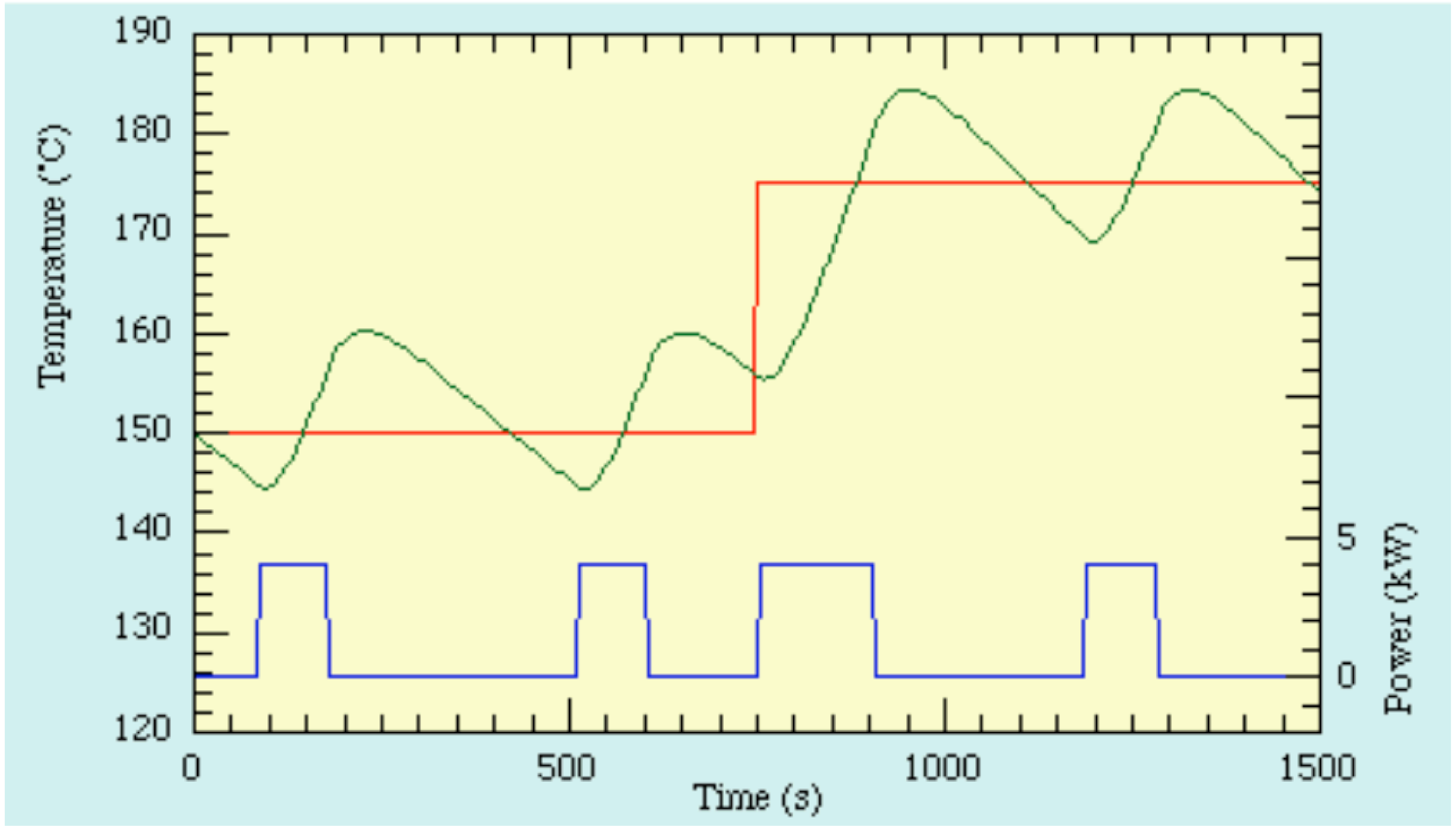


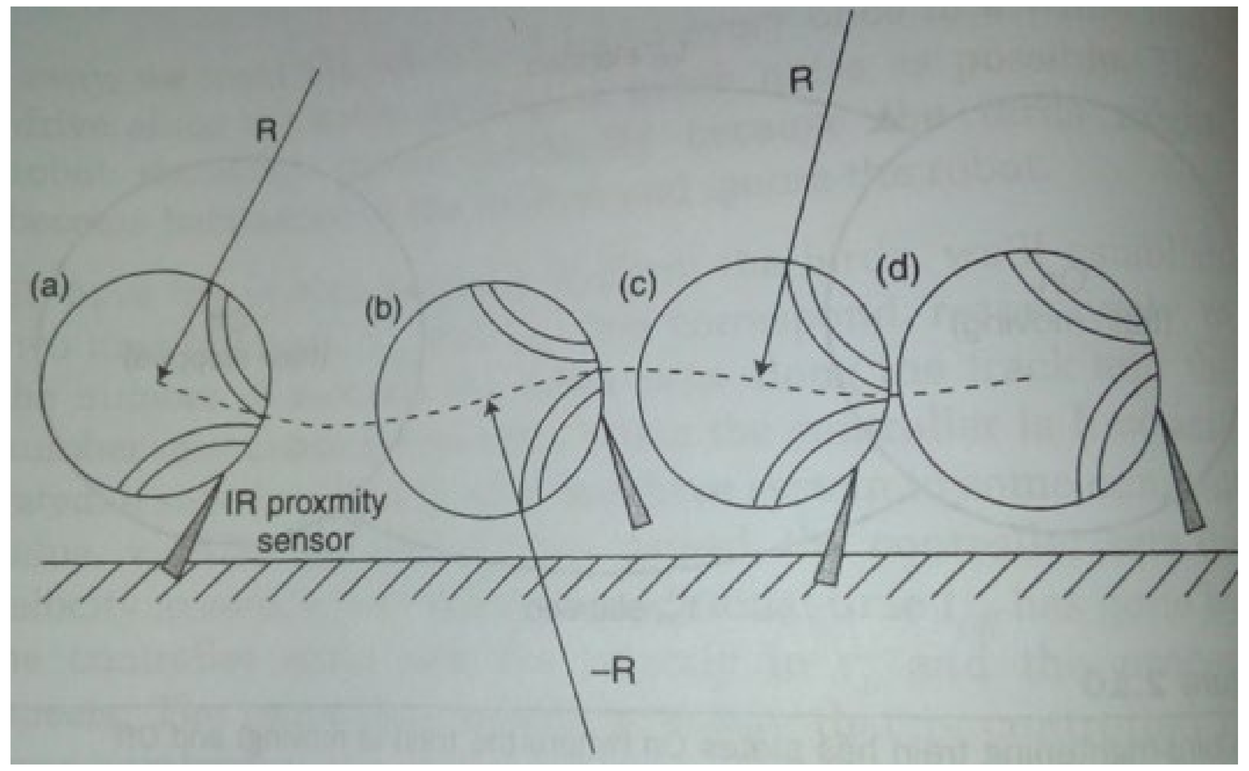
With hysteresis



With no hysteresis

Bang-Bang control at work: household thermostat





What to expect in wall-following?

Big oscillations
around the desired state!

– Optimal for reaching the setpoint
– Not very good for staying near it

Proportional control (P)

- Push back, *proportional* to the error.

$$u = -ke + u_b$$

  - set u_b so that $\dot{x} = F(x_{set}, u_b) = 0$

- For a linear system, we get exponential convergence.

$$x(t) = Ce^{-\alpha t} + x_{set}$$

- The controller gain k determines how quickly the system responds to error.

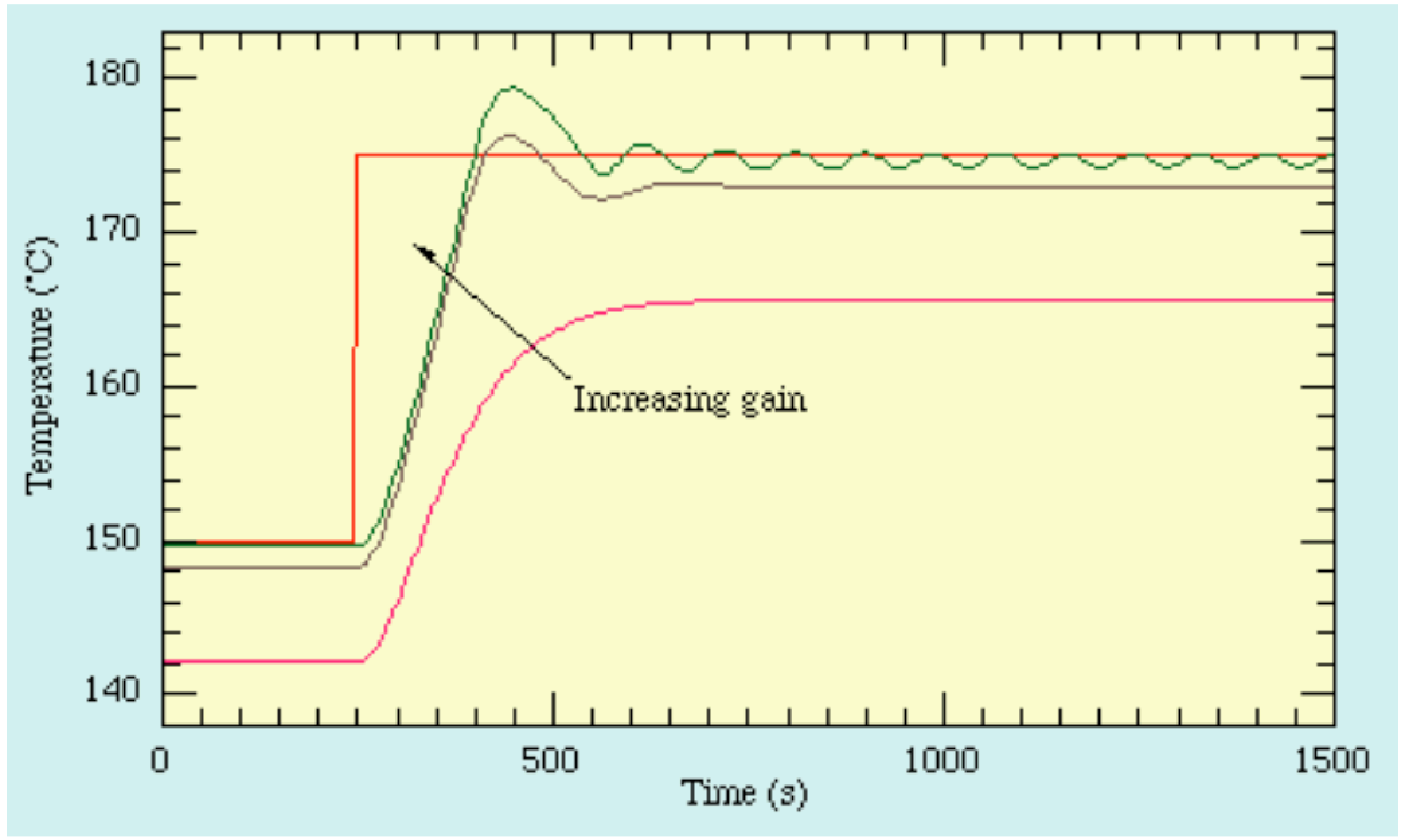
- You want to drive your car at velocity $v_{set.}$
- You issue the motor command $u = pos_{accel}$
- You observe velocity v_{obs}.

- Define a first-order controller:

$$u = -k(v_{obs} - v_{set}) + u_b$$

k is the controller Gain

Proportional control for the thermostat



- Increasing gain approaches setpoint faster
- Can leads to overshoot, and even instability
- Steady-state offset

Steady offset

- Suppose we have continuing disturbances:

$$\dot{x} = F(x,u) + d$$

And we don't know how to model such disturbances, i.e., how to include them in F

- The P-controller cannot stabilize at $e = 0$.
  - if u_b is defined so $F(x_{set},u_b) = 0$
  - then $F(x_{set},u_b) + d \neq 0$, so the system changes
- Must adapt u_b to different disturbances d.

Adaptive control

- Sometimes one controller isn't enough.
- We need controllers at different time scales.

$$u = -k_P e + u_b$$

$$\dot{u}_b = -k_I e \quad \text{where} \quad k_I << k_P$$

- This can eliminate steady-state offset.
  - Why?

– Because the slower controller adapts u_b.

Proportional-Integral (PI) controller

- The adaptive controller $\dot{\boldsymbol{u}}_b = -k_I$ means, by integrating to obtain a point value in time:

  - $\boldsymbol{u}_b = -k_I \int_{t_0}^{t} \boldsymbol{e}(t)dt + \boldsymbol{u}_b$

  - Additively keep memory of all the errors so far within a certain time window $[t_0, t]$, ideally $t_0 = 0$

$$\boldsymbol{u}(t) = -k_P \boldsymbol{e}(t) - k_I \int_{t_0}^{t} \boldsymbol{e}(t)dt + \boldsymbol{u}_b$$

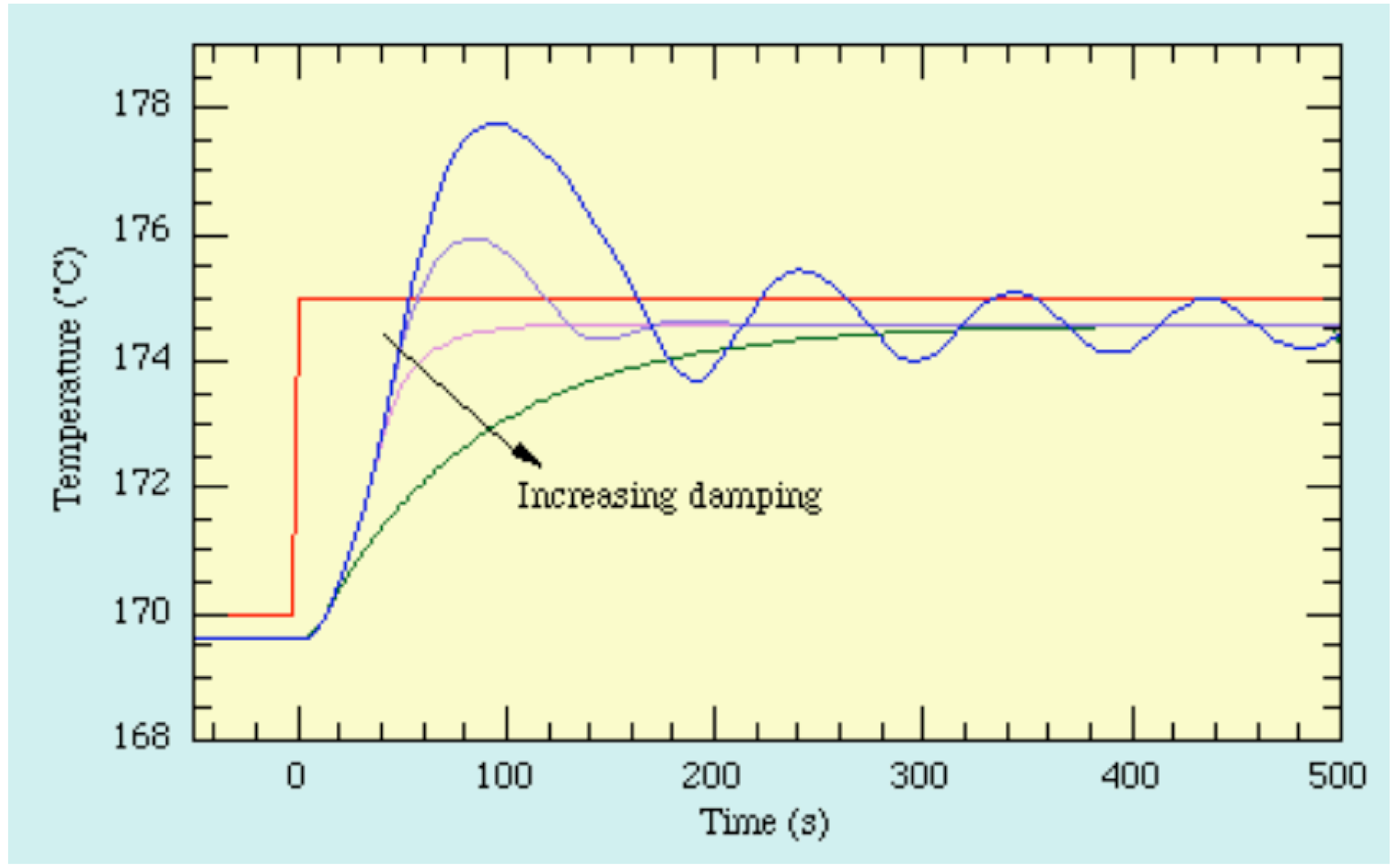
Proportional-Integral (PI) controller

Derivative control

- Damping friction is a force opposing motion, proportional to velocity.
- Try to prevent overshoot by damping controller response.

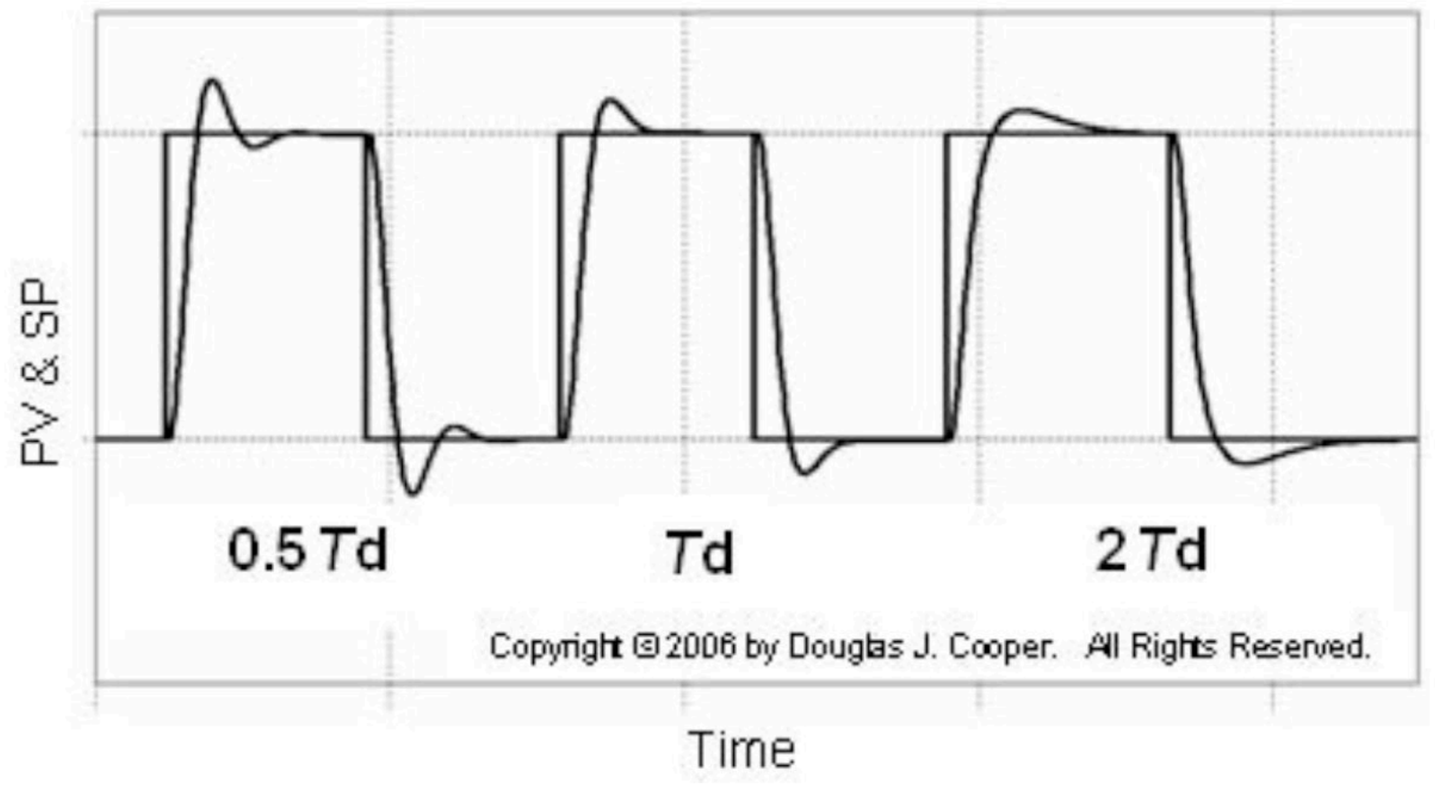
$$u = -k_P e - k_D \dot{e}$$

- Estimating a derivative from measurements is fragile, and amplifies noise.



– Damping fights oscillation and overshoot

– But it's vulnerable to noise



– Different amounts of damping (without noise)

Derivative $\dot{e}$ must be numerically estimated from relative local measures close in time → Subject to imprecisions and approximations, not very reliable in general

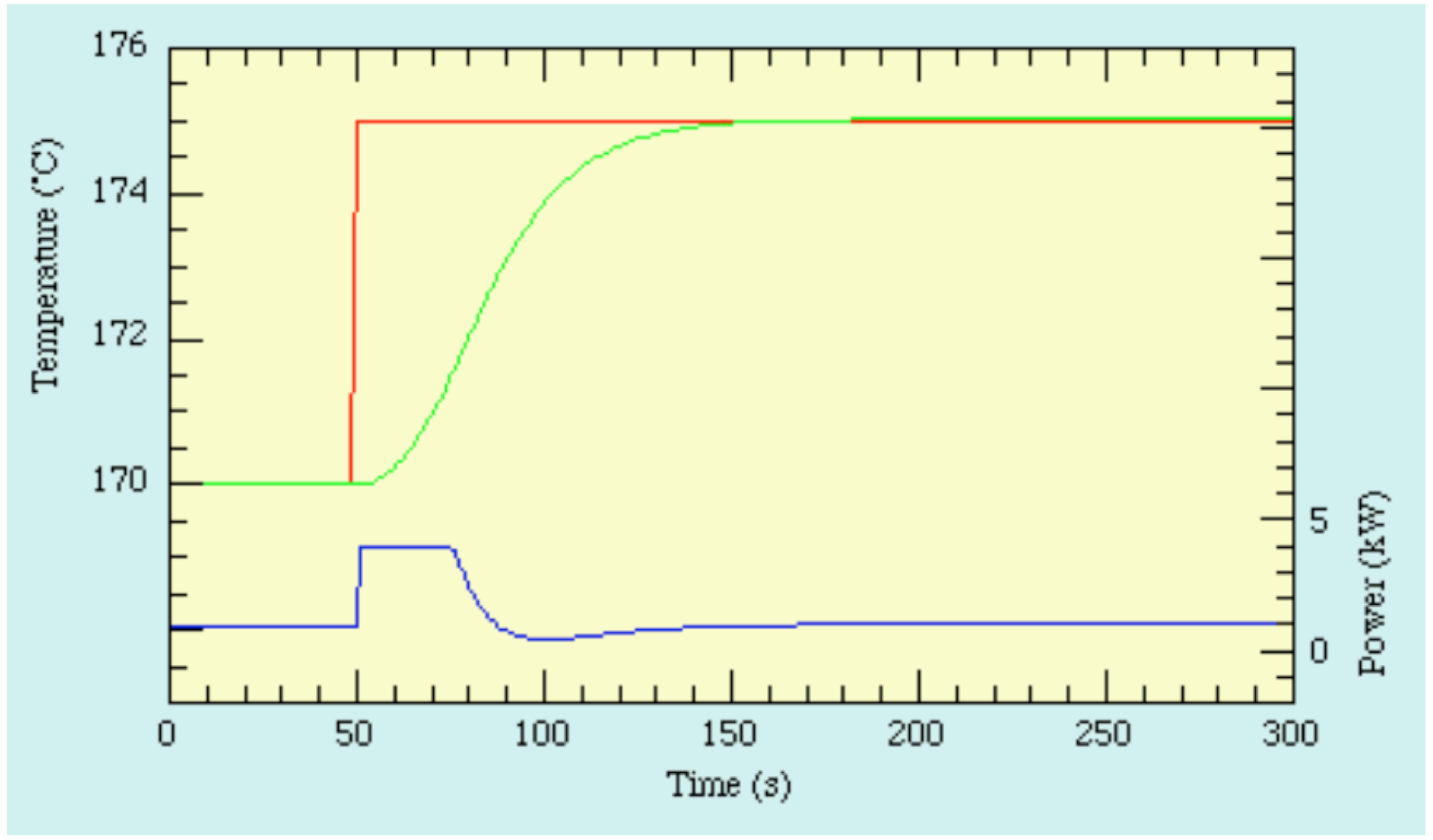
PID Control

- A weighted combination of Proportional, Integral, and Derivative terms.

$$\boxed{u(t) = -k_P\, e(t) - k_I \int_0^t e\, dt - k_D \dot{e}(t)}$$

- The PID controller is the workhorse of the control industry. Tuning is non-trivial.

To be continued …



– But, good behavior depends on good tuning!