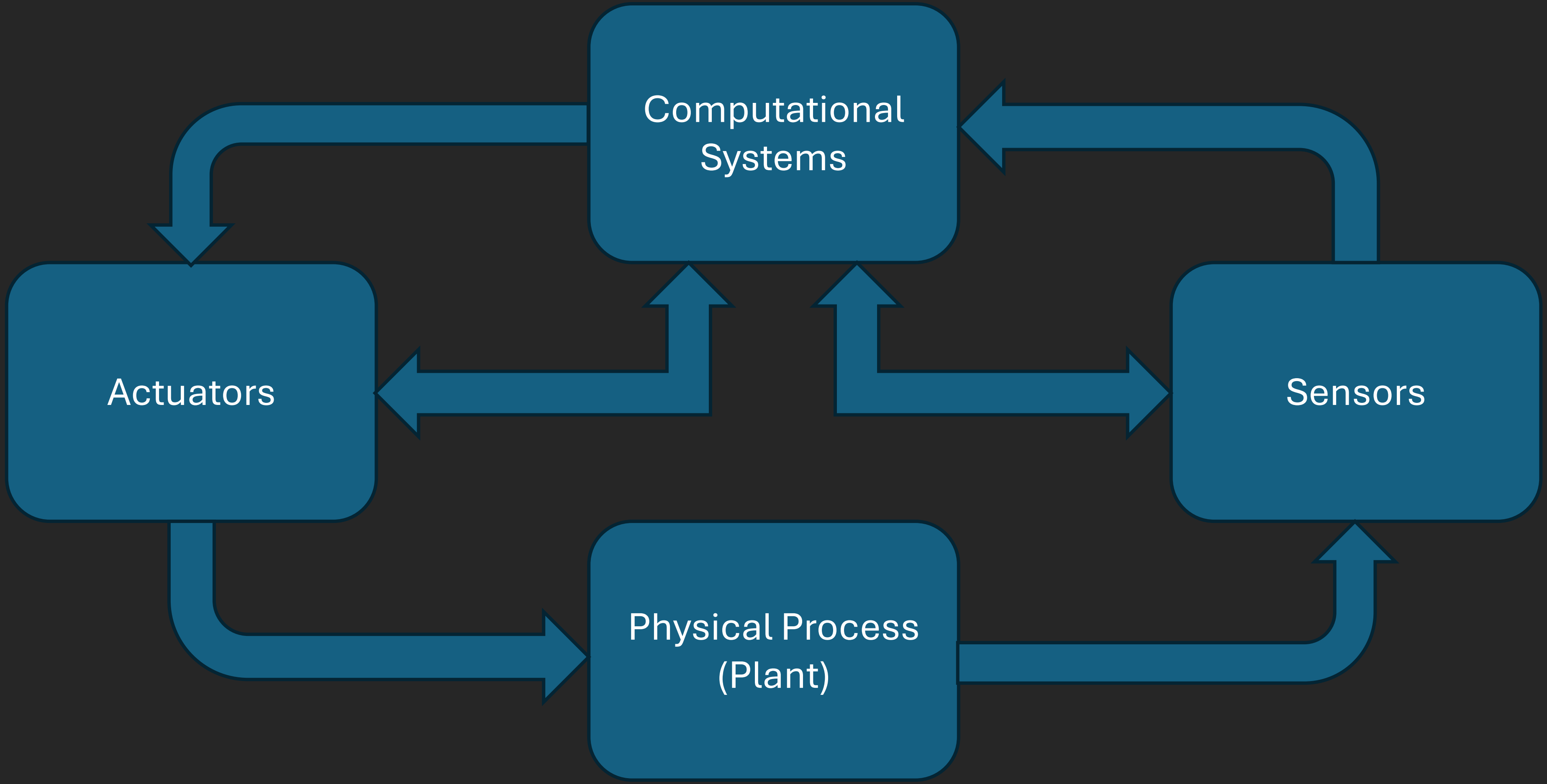
Cyber-Physical Systems

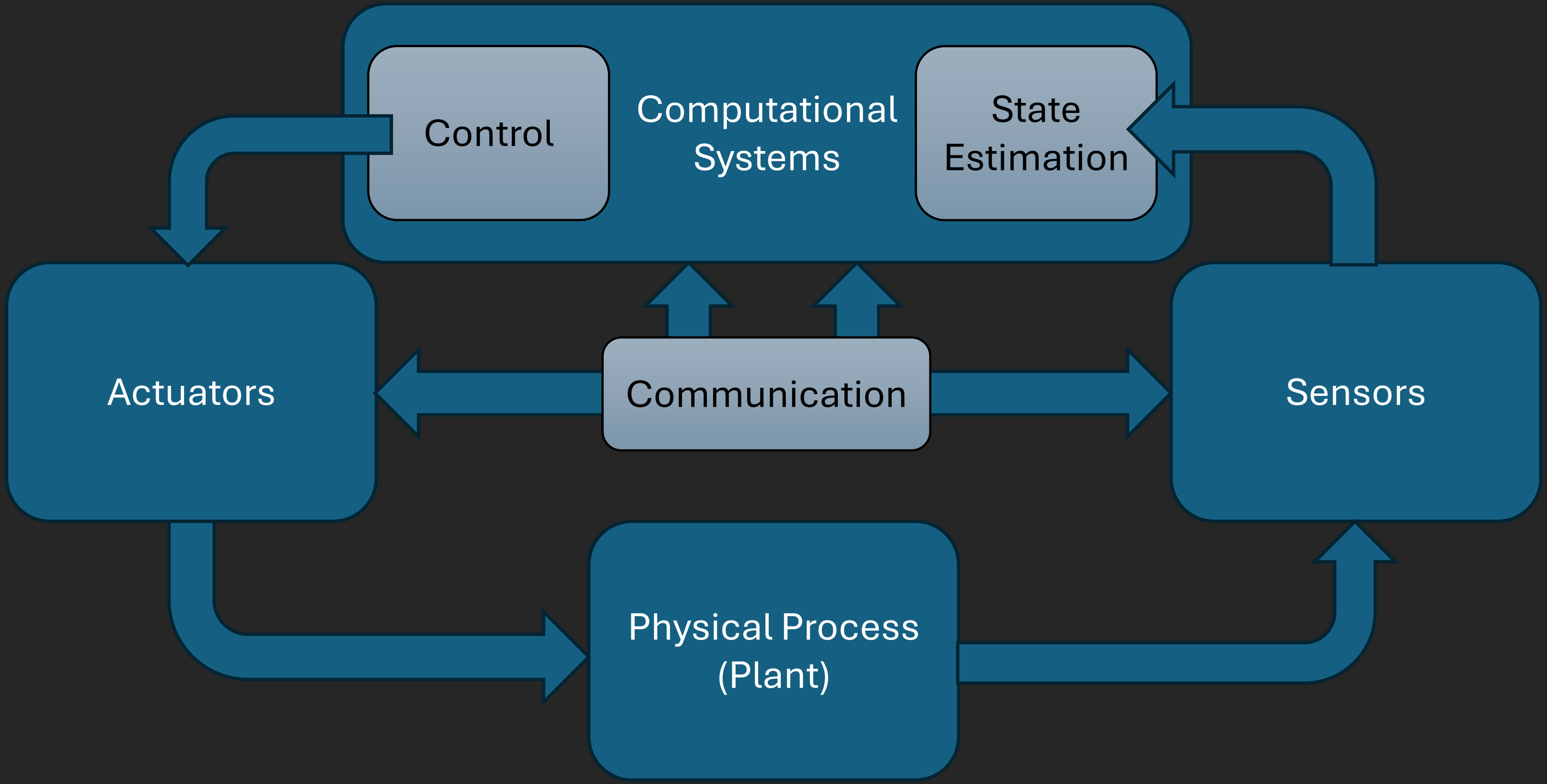
Dr. Jonathan Jaramillo

Feedbacl Control

What are Cyber-Physical Systems?



What are Cyber-Physical Systems?

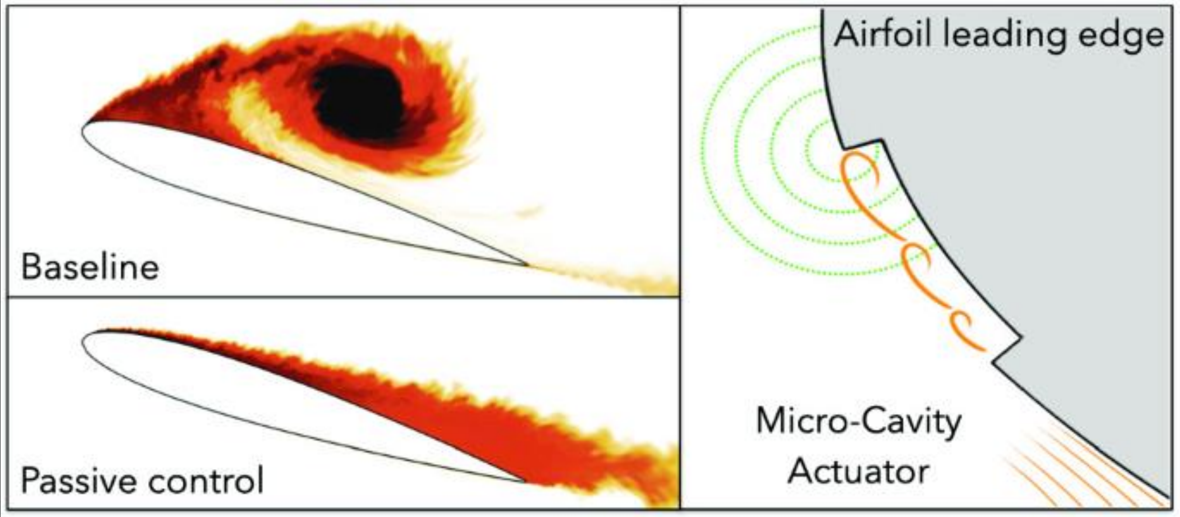


Feedback Control

- Feedback control is a method of automatically regulating a system by continuously measuring its output, comparing it to a desired reference, and adjusting inputs to minimize the error
- Feedback control in Cyber-physical Systems
  - Computational components
  - System Models
  - State Estimation

Non-Feedback Control

- Timers
- Rotational Stability – Spinning frisbee
- Passive drag elements (arrows and planes)
- Boiling food

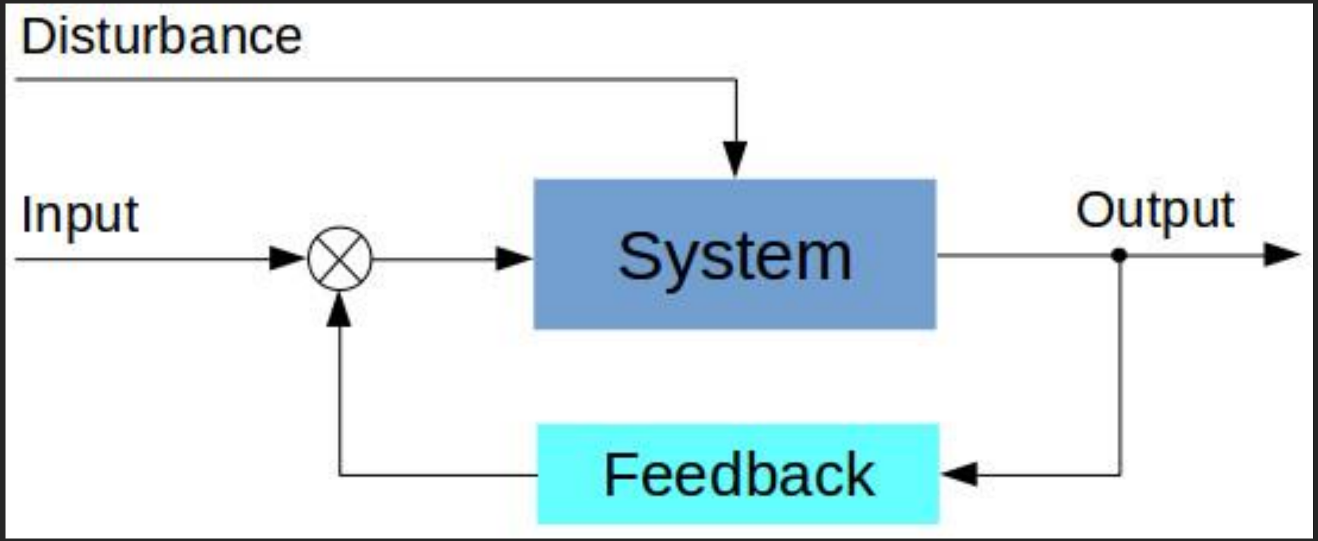


Feedback Control

- Thermostat
- Cruise control
- Autofocus
- Computer fan speed

Concepts of Feedback Control

- State-space control
- PID Control
- Other Control Concepts
  - Adaptive Control
  - Model Predictive Control
  - Distributed Control



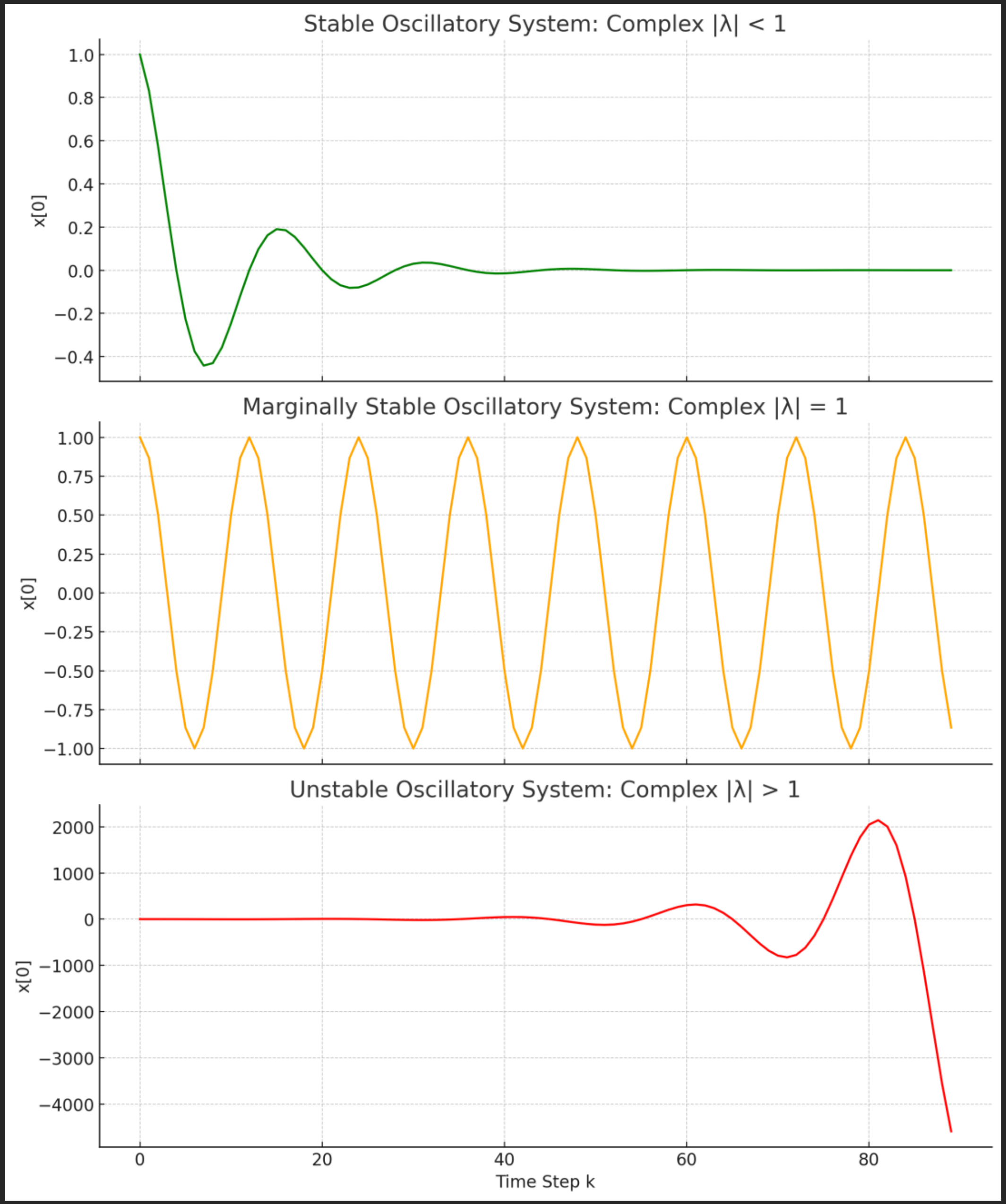
State Space Control

State Space Control

- Canonical State Space Form (Assume full Observability)
  - $x_{k+1} = Ax_k + Bu_{\mathrm{k}}$
  - $y_{k+1} = x_{k+1}$
- With no input system dynamics are governed by A for all time
  - $x_{k+1} = Ax_k$
- Stability analysis
  - Eigan values of A are given by $\det(\lambda I - A) = 0$
  - Unstable if $|\lambda_i| > 1$ for all λ_i
  - Marginally Stable if at least one $|\lambda_i| = 1$ and all other $|\lambda_i| \leq 1$
  - Stable if all $|\lambda_i| < 1$

State Space Control

- $x_{k+1} = Ax_k$
- Stability analysis
  - $|\lambda_i| > 1$ for all λ_i
  - At least one $|\lambda_i| = 1$, all other $|\lambda_i| \leq 1$
  - $|\lambda_i| < 1$
- If stable, system will converge to dominate eigen value



Can We Change the Eigan Values?

- The system dynamics can't change without altering the system.
- Create a control law
  - $x_{k+1} = Ax_k + Bu_{\mathrm{k}}$
  - $u_{\mathrm{k}} = -Kx_k$
- $x_{k+1} = Ax_k - BKx_{\mathrm{k}} = (A - BK)x_{\mathrm{k}}$

- Now system dynamics are governed by $A - BK$ and its eigen values

Control in State Space Methods

- Pole Placement
  - Choosing K such that the system has the exact dynamics you want
- Linear Quadratic Regulator
  - Choses K to minimize a loss function
  - $J = \int_0^\infty x(t)^T\, Qx(t) + u(t)^T Ru(t)dt$
  - Q – penalizes deviation in the state
  - R – penalizes control effort

State Space Control

- Advantages
  - Tools for fine tuning exact system performance exist
  - Simple, predicable output
  - Works for multidimensional control
- Disadvantages
  - Requires exact knowledge of system
  - Errors in system dynamics could result in unstable outputs
  - Susceptible to noise
  - Not robust

Model Predative Control

LQR State Space Control

- Linear Quadratic Regulator
  - Choses K to minimize a loss function
  - $J = \int_0^\infty x(t)^T\, Qx(t) + u(t)^T Ru(t)dt$
  - Q – penalizes deviation in the state
  - R – penalizes control effort
- Time horizon is infinite
- Once K is calculated, it is applied for all time

Discrete Finite Time Horizon

- $x[k+1] = f(x[k], u[k])$
- Look at a finite amount of time into the future
- $u^* = arg \min_{u[0 \ldots N-1]} \sum_{i=0}^{N-1} (x[i] - x_{ref}[i])^T Q (x[i] - x_{ref}[i]) + u[i]^T R u[i])$
- $u^* = arg \min_{u[0 \ldots N-1]} \sum_{i=0}^{N-1} x[i]_{error}^T Q x_{error} + u[i]^T R u[i])$

Choosing Q & R

- $$Q = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & q_n \end{bmatrix}$$

- $$R = \begin{bmatrix} r_1 & 0 & \cdots & 0 \\ 0 & r_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_n \end{bmatrix}$$

Choosing Q & R

- Usually, Q & R are diagonal because state variables are independent
- When to use non-diagonal matrices
  - Coupled state variables
  - Coupled control variables
- Example of coupled state variables
  - Angular speed and linear speed
  - $x = \begin{bmatrix} x \\ y \\ \theta \\ v \\ \dot{\theta} \end{bmatrix}, Q = \begin{bmatrix} q_x & 0 & 0 & 0 & 0 \\ 0 & q_y & 0 & 0 & 0 \\ 0 & 0 & q_\theta & q_{v\theta} & 0 \\ 0 & 0 & q_{v\theta} & q_v & q_{v\dot{\theta}} \\ 0 & 0 & 0 & q_{v\dot{\theta}} & q_{\dot{\theta}} \end{bmatrix}$

Model Predictive Control

- Advantages
  - Handles multi-input, multi-output systems
  - Handles non-linear systems
  - Receding horizon approach and accurate system prediction
- Disadvantages
  - *Very* computationally expensive
  - Errors in system dynamics could result in unstable outputs
  - Tuning complexity
  - Implementation complexity

Model Predictive Control

PID Control

Proportional Control

- Calculate error
  - $e(t) = x_{ref}(t) - x(t)$
- Calculate control value
  - $u(t) = K_P e(t)$

- What happens when there is a force resisting the system?

- Proportional control will always have steady state error

Proportional-Integral Control

- Calculate error
  - $e(t) = x_{ref}(t) - x(t)$
- Calculate control value
  - $u(t) = K_P e(t) + \mathrm{K_I} \int_0^t e(\tau) d\tau$

- Removes steady state error!

- Introduces overshoot.

Proportional-Integral-Derivative Control

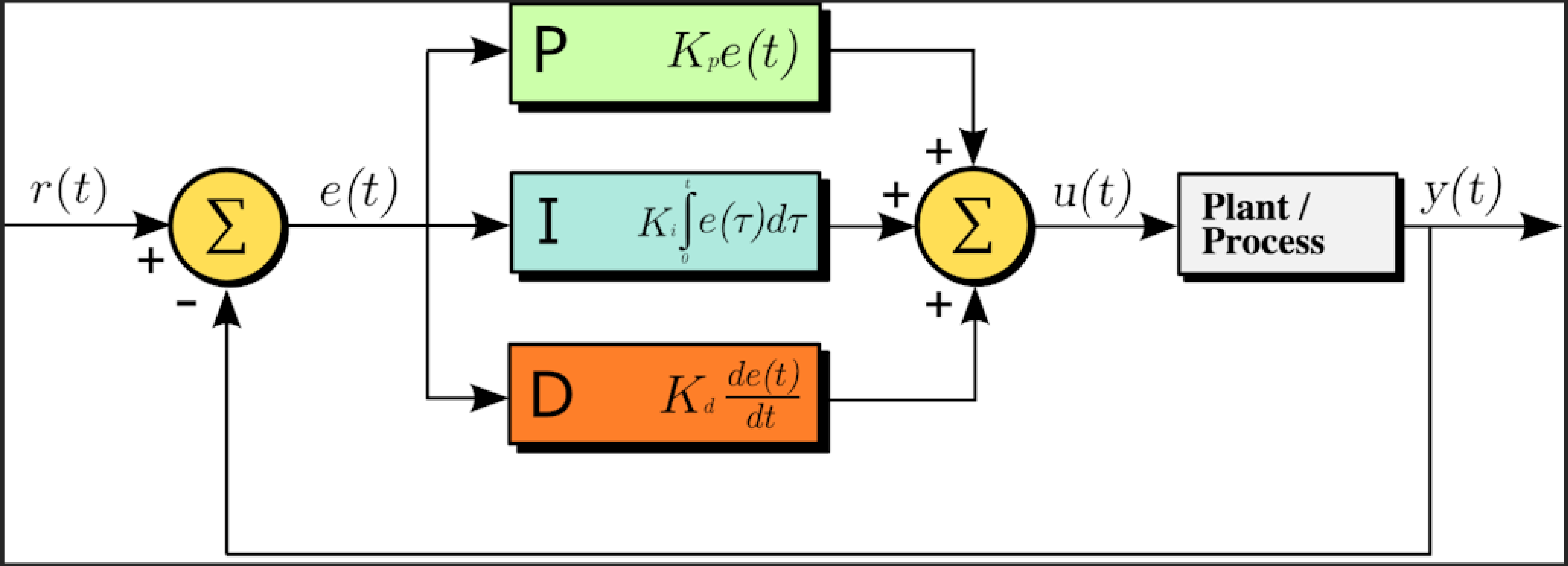
- Calculate error
  - $e(t) = x_{ref}(t) - x(t)$
- Calculate control value
  - $u(t) = K_P e(t) + \mathrm{K_I} \int_0^t e(\tau) d\tau + \mathrm{K_D} \frac{d}{dt} e(t)$

- Can reduce overshoot

- Can produce instability

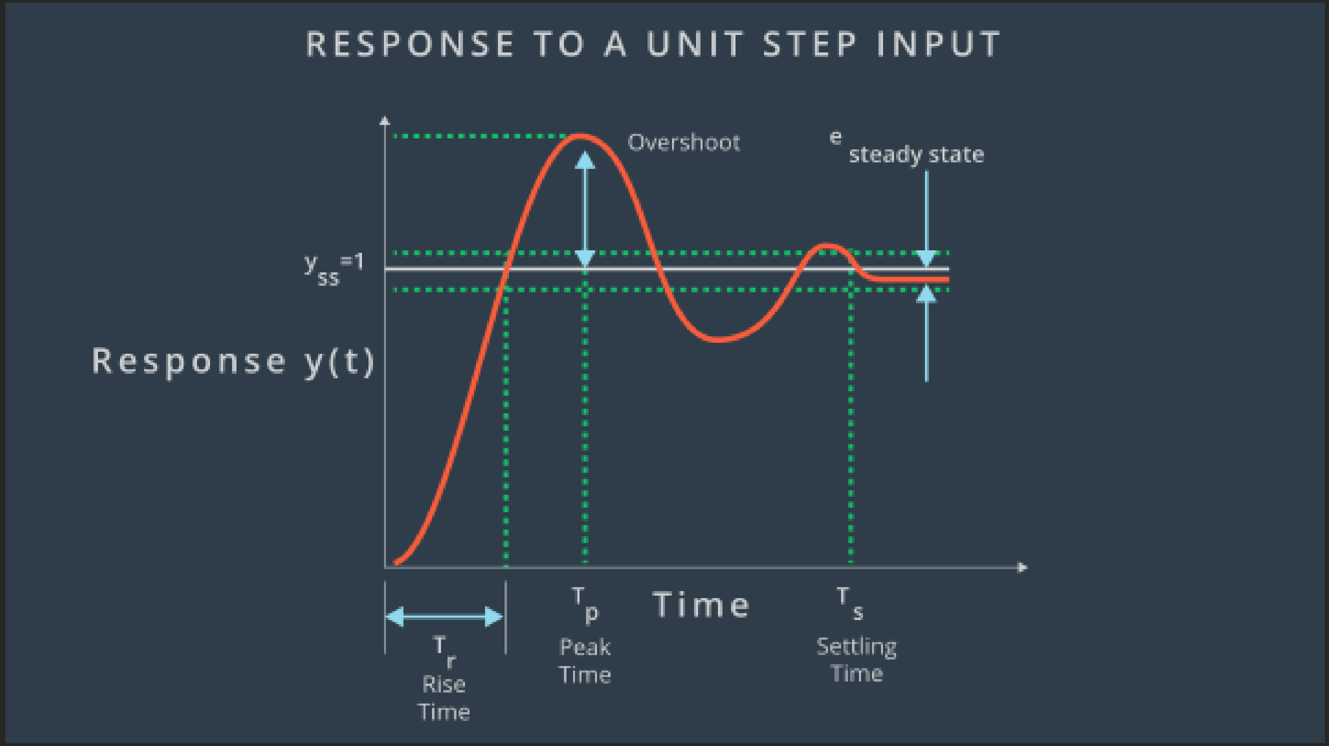
PID Control in Discrete Time

- $u[k] = K_P + K_I \sum_{i=0}^{k} e[\,\mathrm{i}]\mathrm{T_s} + \mathrm{K_D}\, \frac{e[k]-e[k-1]}{T_S}$



Characterizing a PID Controller

- Rise Time
- Settling Time
- Overshoot
- Steady State Error

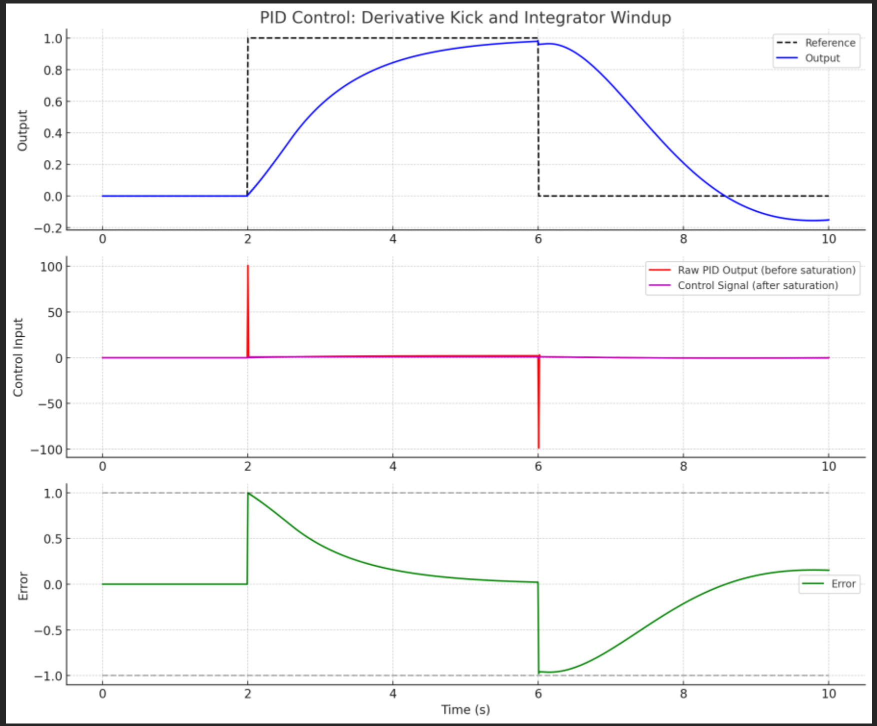


PID Parameters

- K_P
  - Faster response, more reactive
  - Overshoot and oscillation
- K_I
  - Eliminates steady-state error
  - Sluggish, integrator windup
- K_D
  - Reduces overshoot, adds dampening
  - Sensitive to noise

Potential Problems with PID

- Constant steady-state error
  - Integrator windup
- Derivative Kick
  - High frequency noise
  - Step change in reference signal
- Output Clamping
  - Calculate $u(t)$ is greater than actuator can generate



Derivative Kick

- $\frac{de}{dt} = \frac{d}{dt}\left(x_{ref}(t) - x(t)\right)$
- $\frac{de}{dt} = \frac{dx_{ref}(t)}{dt} - \frac{dx(t)}{dt}$

- $\frac{dx_{ref}(t)}{dt}$ will spike if there is a change in reference signal

- $\frac{de}{dt} = -\frac{dx(t)}{dt}$

Manually Tuning PID Controllers

- Start with $K_I = 0$ and $K_D = 0$
- Increase K_P until system begins to oscillate
- Add K_D to reduce overshoot and dampen oscillations
- Add K_I to eliminate steady-state error
- Repeat

Ziegler-Nichols Method

- Set $K_I = 0$ and $K_D = 0$
- Increase K_P until system starts to oscillate
  - This gain is K_w and period of oscillation is T_w

Controller	K_P	K_I	K_D
P	$0.5K_w$	-	-
PI	$0.45K_w$	$1.2K_P/T_w$	-
PID	$0.6K_w$	$2K_P/T_w$	$K_pT_w/8$

PID Control

- Advantages
  - Simple – doesn't require detailed system modeling or design
  - Computationally efficient
  - Effective for many single input-single output systems
- Disadvantages
  - Limited in multi-variable systems
  - Poor predictive capability
  - Sensitive to tuning and nonlinearities

Adaptive Control

Adaptive Control

- Challenges in implementing feedback control systems
  - Unknown system dynamics
  - Changing system dynamics
- Examples
  - Vehicles carry different payloads
  - Electrical grid demand changes from season to season
  - User of a system might change

Adaptive Control

- $x_{k+1} = A_k x_k + B_k u_{\mathrm{k}}$

- Adaptive control
  - As you control your system, save the last N data point
  - Perform linear regression to estimate A_k and B_k
  - Use A_k and B_k to calculate new control law

Adaptive control

- Self Tuning Regulator
  - Estimates A_k and B_k in real time
  - Recalculates control law (pole-place, MPC, PID) based on new A_k and B_k
- Gain Scheduling (Quasi-Adaptive)
  - Precomputes different laws (pole-place, MPC, PID)
  - Switches or interpolates between them based on estimated A_k and B_k

Adaptive Control

Adaptive Control