Chapter Outline

Controller Design and Control Allocation

Chapter 5
Filtering, State Estimation,
and Low-Level Control

5.1 Filtering and Low-Level Control

5.2 Sensor Calibration and Estimation

5.3 Controller Design and Allocation

5.4 Control Performance and Tuning

5.5 PX4 Architecture and Interfaces

5.6 Control Development and Testing

5.7 Other Vehicles and Advanced Cases

5.8 Exercises and Classroom Discussion

4 Units in This Section

5.3.1 Proportional-Integral-Derivative (PID) Controllers

Textbook: Sec. 5.1.3.1

5.3.2 Saturated PID Controllers

Textbook: Sec. 5.1.3.1

5.3.3 Low-Level Control Architecture and Controller Types

Textbook: Secs. 5.1.2.4, 5.1.3.1

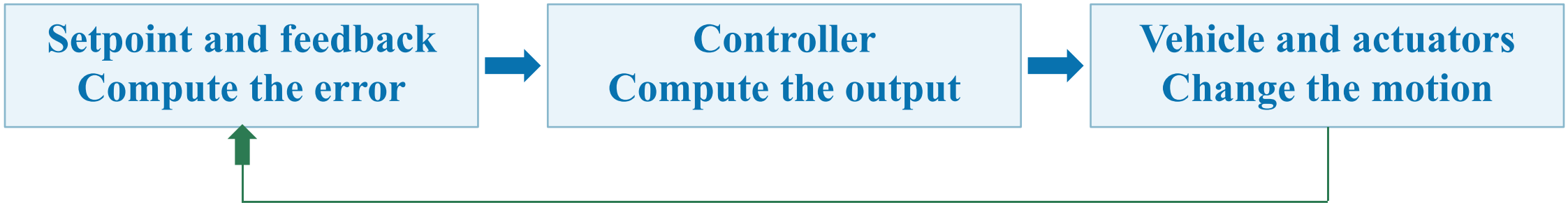
5.3.4 Actuator Control Allocation

Textbook: Sec. 5.1.5

5.3.1 PID Controllers: Setpoints and Error

Setpoint and Feedback Determine the Control Output

PID uses the error and its history. Assume accurate feedback first to examine the control law.



$$e = r - \hat{y}, \qquad u = \mathcal{C}(e)$$

r is the setpoint, $\hat{y}$ the state estimate, e the error, and u the control output. Its units depend on the interface.

5.3.1 PID Controllers: PID Terms

Three Terms Use Error Information from Different Times

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,\mathrm{d}\tau + K_d \frac{\mathrm{d}e(t)}{\mathrm{d}t} \qquad \text{Eq. 5.1}$$

Term	Information Used	Effect and Tradeoff
Proportional (P)	Current error	Immediate correction. Excessive gain may cause oscillation.
Integral (I)	Error accumulated over time	Compensates for constant bias. Requires anti-windup.
Derivative (D)	Rate of change of error	Adds damping or trend information. Amplifies noise.

Estimate the current rate from signals. Convert all terms to common output units before summing.

5.3.1 PID Controllers: Gain Units

Error and Output Units Determine Gain Units

Term	Input Units	Gain Units for a Force Output
Proportional	e: m	K_p: N/m
Integral	$\int e\,dt$: m s	K_i: N/(m s)
Derivative	$\dot{e}$: m/s	K_d: N s/m

With attitude error in rad and output in N·m, the gain units change accordingly.
Normalized examples illustrate the calculation. Tune real-vehicle gains using the actual error and output units.

5.3.1 PID Controllers: Discrete PID

Every Integration Step Includes the Sample Time

$$J_k = J_{k-1} + e_k \Delta t, \quad D_k = \frac{e_k - e_{k-1}}{\Delta t}, \quad u_k = K_p e_k + K_i J_k + K_d D_k$$

J is the raw error integral (in s here). $I = K_i J$ is the weighted integral contribution (in output units).
Normalized teaching example: $e_k = 0.20$, $e_{k-1} = 0.24$, $\Delta t = 0.1$ s, $J_{k-1} = 0.08$ s.
Gains: $K_p = 2$, $K_i = 0.5\ \text{s}^{-1}$, $K_d = 0.1$ s.

$$J_k = 0.08 + 0.20 \times 0.1 = 0.10\ \text{s}, \quad D_k = (0.20 - 0.24)/0.1 = -0.40\ \text{s}^{-1}$$

$$u_k = 2 \times 0.20 + 0.5 \times 0.10 + 0.1 \times (-0.40) = 0.40 + 0.05 - 0.04 = 0.41$$

5.3.1 PID Controllers: Derivative Implementation

Differencing Amplifies Noise in the Control Output

Normalized stationary readings: $0.010 \rightarrow -0.010$. $\Delta t = 0.01$ s, $K_d = 0.1$ s.
Use derivative on measurement, $u_D = -K_d \dot{y}$. The true rate is zero.

$$\dot{y}_k = \frac{-0.010 - 0.010}{0.01} = -2.0\ \mathrm{s}^{-1} \qquad u_D = -0.1 \times (-2) = +0.20$$

Implementation	Input and Result	Conditions
Filtered derivative	Low-pass filter the difference	Reduces noise, adds phase lag
Derivative on measurement	Differentiate feedback only: $-\dot{y}$	Setpoint steps bypass D. No derivative kick.
Angular-rate feedback	Use gyro data for the attitude channel	Body rates generally differ from Euler-angle rates

5.3.1 PID Controllers: Response Comparison

How Do Four Controllers Respond on the Same Plant?

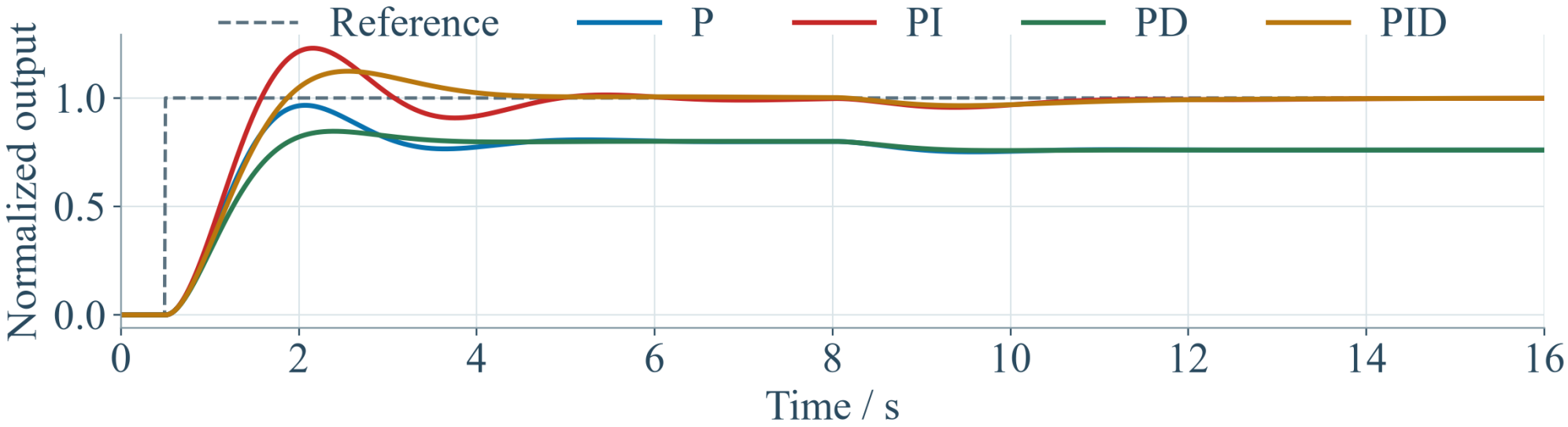


Figure: P, PI, PD, and PID responses of a separate second-order teaching plant

Plant $\ddot{y} + 2\dot{y} + y = u + d$ (normalized, zero initial conditions).

0.5 s: setpoint 0→1; 8 s: input disturbance $d = -0.2$. Fourth-order Runge–Kutta (RK4), step 0.01 s.

Derivative on measurement: $u_D = -K_d\dot{y}$

Gains (K_p, K_i, K_d): P(4,0,0), PI(4,2,0), PD(4,0,1), PID(4,2,1) (K_i: s^{-1}, K_d: s).

5.3.1 PID Controllers: Control Inputs and Results

Read the Control Input Alongside the Tracking Response

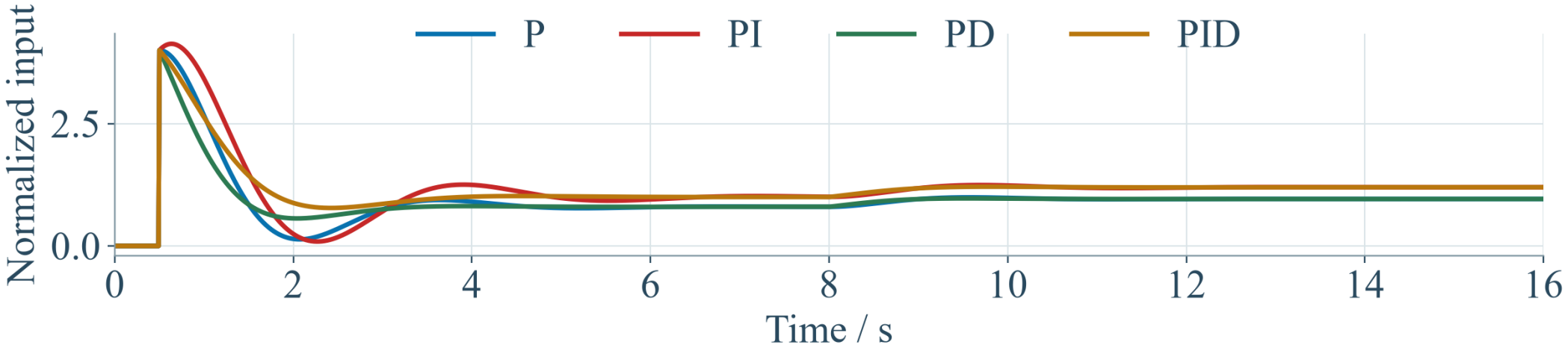


Figure: Control inputs from the same calculation, with no output limits

Method	P	PI	PD	PID
Response at 16 s	0.7600	0.9986	0.7600	0.9991
Peak response	0.9663	1.2296	0.8467	1.1235

P/PD retain ≈0.24 steady-state error. PI/PID reach ≈1 using I. PID peaks lower here. Input peak ≈4.

5.3.1 PID Controllers: Term Contributions and Motion

How P, I, and D Combine at Each Instant

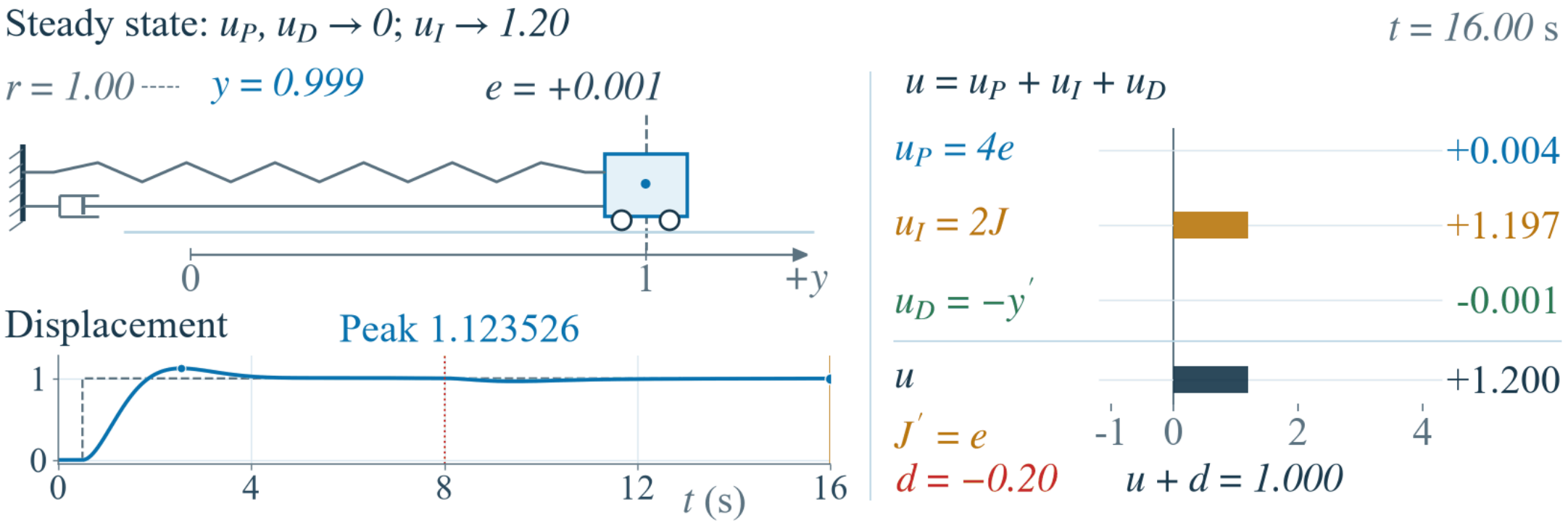


Figure: Simultaneous P/I/D contributions, same second-order plant (normalized, no limits)

P follows error. D brakes rapid motion. I sustains compensation after a disturbance.

Classroom Exercise

Compute One Discrete PID Step with New Data

Problem: Compute one normalized output using the discrete PID recursion (error-difference form).
Data: $e_k = 0.30$, $e_{k-1} = 0.20$, $\Delta t = 0.05$ s, $J_{k-1} = 0.10$ s.
Gains: $K_p = 1.5$, $K_i = 0.8\ \text{s}^{-1}$, $K_d = 0.05$ s.
(1) Find J_k, D_k, and u_k. (2) With a constant setpoint, does using derivative on measurement change the D term?

Suggested Answers

$$J_k = 0.10 + 0.30 \times 0.05 = 0.115\,\text{s}, \quad D_k = \frac{0.30 - 0.20}{0.05} = 2.0\,\text{s}^{-1}$$

$$u_k = 1.5 \times 0.30 + 0.8 \times 0.115 + 0.05 \times 2.0 = 0.45 + 0.092 + 0.10 = 0.642$$

(2) No. For a constant setpoint, $\dot{e} = -\dot{y}$, so both forms give the same D term.
Common errors: omitting Δt in integration gives 0.87. Omitting division by Δt in differencing gives 0.547.

Chapter Outline

Controller Design and Control Allocation

Chapter 5
Filtering, State Estimation,
and Low-Level Control

5.1 Filtering and Low-Level Control

5.2 Sensor Calibration and Estimation

5.3 Controller Design and Allocation

5.4 Control Performance and Tuning

5.5 PX4 Architecture and Interfaces

5.6 Control Development and Testing

5.7 Other Vehicles and Advanced Cases

5.8 Exercises and Classroom Discussion

4 Units in This Section

5.3.1 Proportional-Integral-Derivative (PID) Controllers

Textbook: Sec. 5.1.3.1

5.3.2 Saturated PID Controllers

Textbook: Sec. 5.1.3.1

5.3.3 Low-Level Control Architecture and Controller Types

Textbook: Secs. 5.1.2.4, 5.1.3.1

5.3.4 Actuator Control Allocation

Textbook: Sec. 5.1.5

5.3.2 Saturated PID Controllers: Output Limits

Control Requests and Actuator Actions Are Separate

PID request
Raw control output

→

Saturation function
Limit the range

→

Actual control output
Applied to the plant

$$u = \mathrm{sat}_{[0,1]}(u_{\mathrm{raw}}) = \min(1, \max(0, u_{\mathrm{raw}}))$$

Request	Actual Output	Unrealized Portion
1.40	1.00	0.40
−0.20	0.00	−0.20

5.3.2 Saturated PID Controllers: Integral Windup

The Integral Can Grow for Unreachable Setpoints

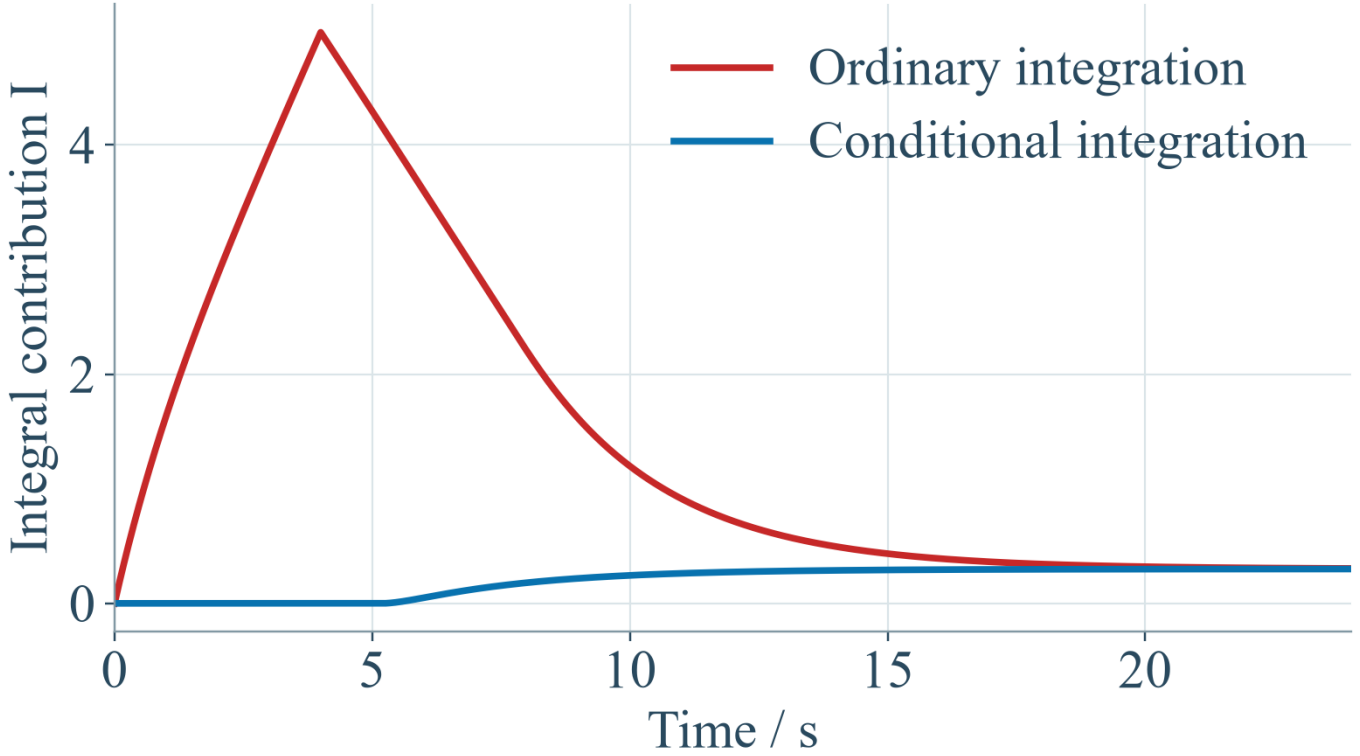


Figure: I for ordinary and conditional integration

Model: $\dot{y} = -y + u$
Limits: $0 \leq u \leq 1$
Setpoint: 2, then 0.3 at 4 s.
$K_p = 2, K_i = 1\ \text{s}^{-1}$
Ordinary integration: $I = K_i J$.
Conditional integration pauses increments.
Both outputs are 0.982 at 4 s.
Integral contribution I:
Ordinary: 4.982. Conditional: 0.

5.3.2 Saturated PID Controllers: Conditional Integration

Pause Further Windup and Allow Unwinding

Here $K_i > 0$ and I is the integral contribution. Positive error increases I. Negative error decreases I.

Current State	Error Direction	Integral Action
Upper-limit saturation	Positive error	Pause further windup
Lower-limit saturation	Negative error	Pause further windup
Saturated, opposing error	Reduces saturation	Allow unwinding
Unsaturated	Any	Integrate normally

$K_p = 2$, $K_i = 1\ \text{s}^{-1}$, $K_d = 0$, $\Delta t = 0.1$ s, $I_{k-1} = 0.40$, $e_k = 0.50$, limits $[0,1]$.

$$I_{\text{trial}} = 0.40 + 1 \times 0.50 \times 0.1 = 0.45, \quad u_{\text{trial}} = 2 \times 0.50 + 0.45 = 1.45$$

Trial $u_{\text{trial}} = 1.45 > 1$ with $e_k > 0$: retain $I_k = 0.40$. Opposing error may unwind the integral.

5.3.2 Saturated PID Controllers: Back-Calculation

Output Mismatch Feeds Back to the Integral

$$\dot{I} = K_i e + K_{\text{aw}}(u - u_{\text{raw}})$$

I is the integral contribution. Back-calculation corrects its state. Without extra correction, $I = K_i J$ (J: raw error integral).

I has output units. K_{aw} has units s^{-1}. $K_p = 2$, $K_i = 1\ \text{s}^{-1}$, $K_{\text{aw}} = 2\ \text{s}^{-1}$.

$e = 0.5$, $I_{k-1} = 0.40$, $\Delta t = 0.1$ s. Requested: $u_{\text{raw}} = 1.4$. Actual: $u = 1.0$.

$$\dot{I} = 1 \times 0.5 + 2(1.0 - 1.4) = -0.3\ \text{s}^{-1}$$

$$I_k = 0.40 + (-0.3) \times 0.1 = 0.37$$

5.3.2 Saturated PID Controllers: Recovery

Compare Recovery Under the Same Physical Limits

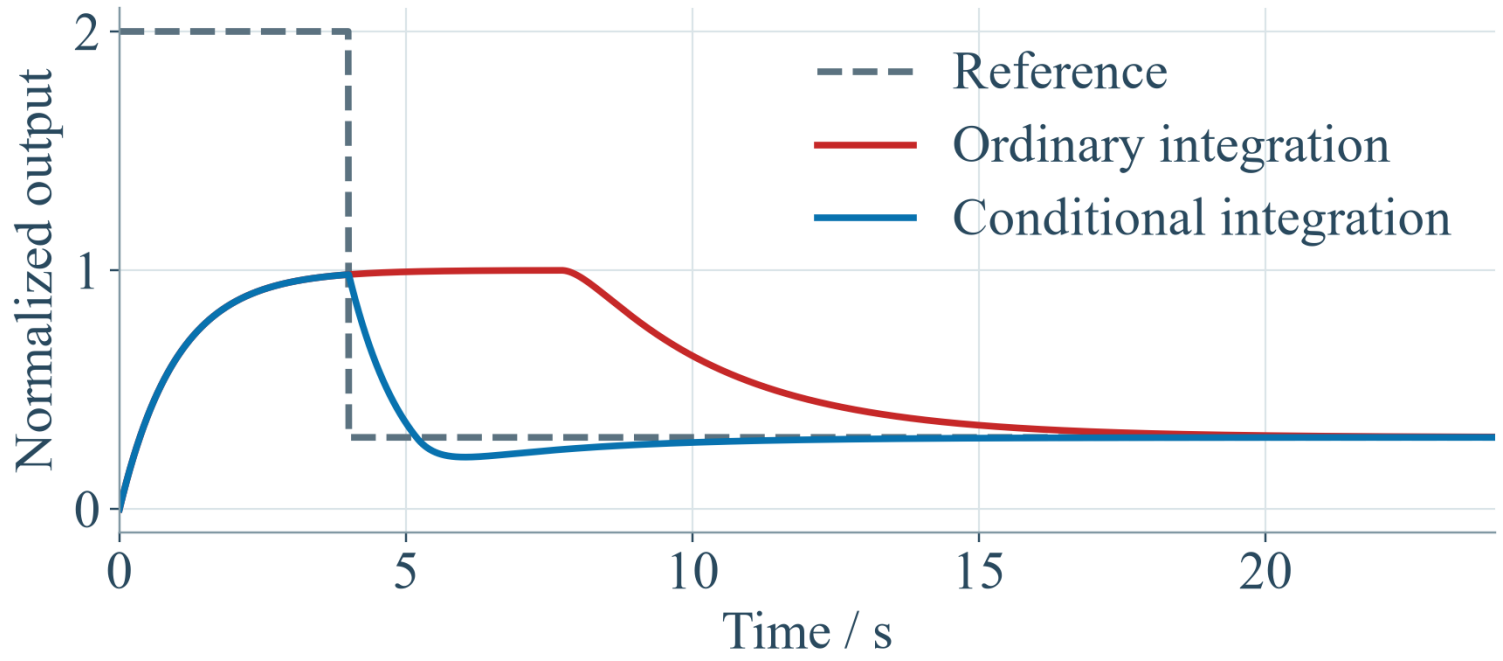


Figure: Recovery under identical limits

At 4 s: setpoint 2→0.3.
Enter and remain within $|y - 0.3| \leq 0.03$.
Recovery time stamp:
Ordinary 16.39 s; conditional 9.10 s.
Elapsed from 4 s:
Ordinary 12.39 s; conditional 5.10 s.
Same model, gains, limits and initial state; only integration differs.

A persistent request beyond actuator capability changes how the integrator recovers.

5.3.2 Saturated PID Controllers: Velocity Control Example

Separate Velocity PID: Integral Strategy and Recovery

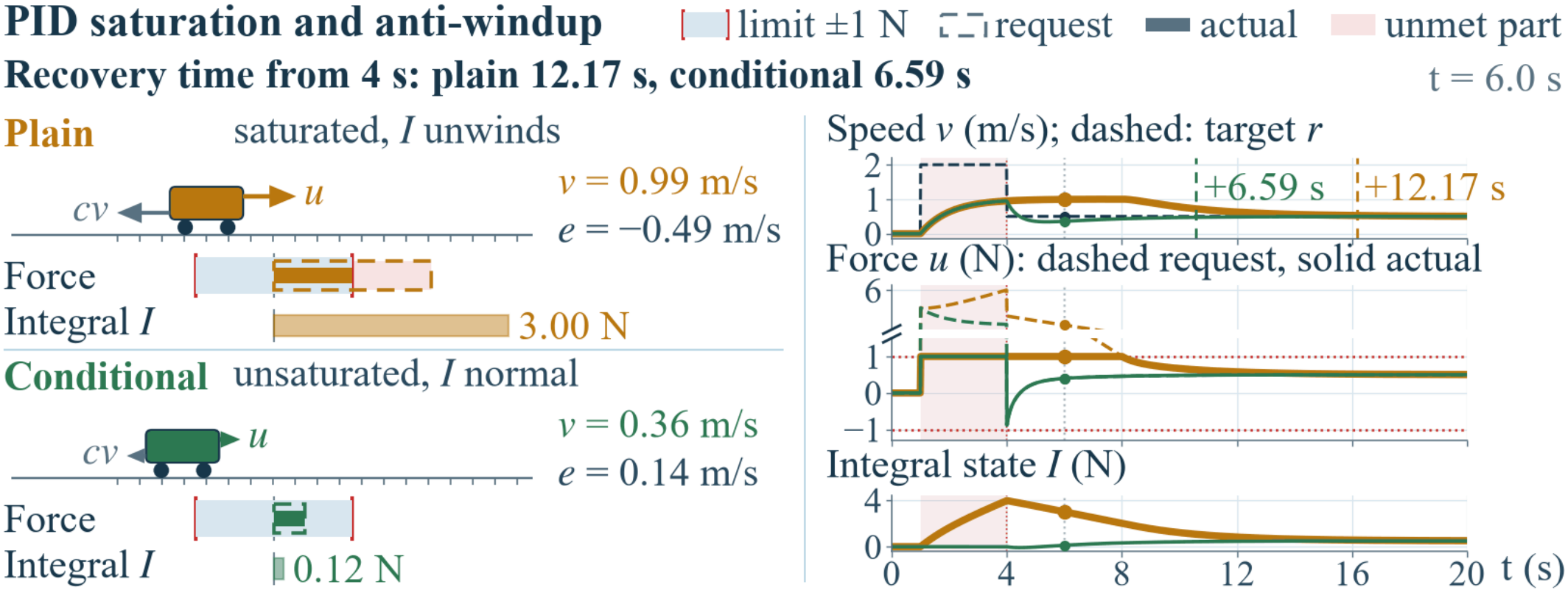


Figure: PID teaching simulation. Linear drag cv, c=1 N·s/m. At 4 s: 2→0.5 m/s setpoint.

5.3.2 Saturated PID Controllers: Velocity Example Parameters

Same Motion at 1–4 s, Integral History Shapes Recovery

Plant: $m\dot{v} = u - cv$, $m = 1$ kg, linear drag $c = 1$ N·s/m (cv in the animation).
PID: $K_p = 2$ N·s/m, $K_i = 1$ N/m, $K_d = 0.1$ N·s²/m. Measured-velocity D: 0.1 s low-pass filter.
Limits: Force $|u| \leq 1$ N. Maximum steady speed: 1 m/s.
Criterion: Recovery means $|e| \leq 0.025$ m/s holds from then onward.
Setpoint: 0 to 2 m/s at 1 s (unreachable), then 0.5 m/s at 4 s.

1–4 s: Both actual forces are 1 N, so motion is identical. Ordinary integration reaches $I \approx 3.95$ N.
Conditional integration keeps $I = 0$.
After 4 s, ordinary integration still requests about 3.05 N, saturates at the upper limit, and keeps pushing until about 8 s.
After 4 s, conditional integration requests about −0.91 N and brakes immediately. Speed bottoms near 0.34 m/s, then rises.
Result: Elapsed recovery from 4 s is about 12.17 s with ordinary integration and 6.59 s with conditional integration.

Classroom Exercise

Should the Integral Update During Saturation?

Problem: PI control with conditional integration. Use the rules on the "Conditional Integration" page.
Parameters: $K_p = 1.5$, $K_i = 2\ \text{s}^{-1}$, $\Delta t = 0.05$ s, limits $[0,1]$.
(1) $I_{k-1} = 0.30$, $e_k = 0.60$. (2) $I_{k-1} = 1.40$, $e_k = -0.20$.
Use trial values to decide whether to update the integral this step. Find I_k and the actual output u.

Suggested Answers

$$(1)\ I_{\text{trial}} = 0.30 + 2 \times 0.60 \times 0.05 = 0.36, \quad u_{\text{trial}} = 1.5 \times 0.60 + 0.36 = 1.26 > 1$$

Upper-limit saturation with $e_k > 0$: pause, so $I_k = 0.30$ and $u = 1$.

$$(2)\ I_{\text{trial}} = 1.40 + 2 \times (-0.20) \times 0.05 = 1.38, \quad u_{\text{trial}} = -0.30 + 1.38 = 1.08 > 1$$

Still at the upper limit, but $e_k < 0$ reduces saturation. Allow the update: $I_k = 1.38$, $u = 1$.
Common error: freezing whenever saturated incorrectly gives 1.40 for (2).

Chapter Outline

Controller Design and Control Allocation

Chapter 5
Filtering, State Estimation,
and Low-Level Control

5.1 Filtering and Low-Level Control

5.2 Sensor Calibration and Estimation

5.3 Controller Design and Allocation

5.4 Control Performance and Tuning

5.5 PX4 Architecture and Interfaces

5.6 Control Development and Testing

5.7 Other Vehicles and Advanced Cases

5.8 Exercises and Classroom Discussion

4 Units in This Section

5.3.1 Proportional-Integral-Derivative (PID) Controllers

Textbook: Sec. 5.1.3.1

5.3.2 Saturated PID Controllers

Textbook: Sec. 5.1.3.1

5.3.3 Low-Level Control Architecture and Controller Types

Textbook: Secs. 5.1.2.4, 5.1.3.1

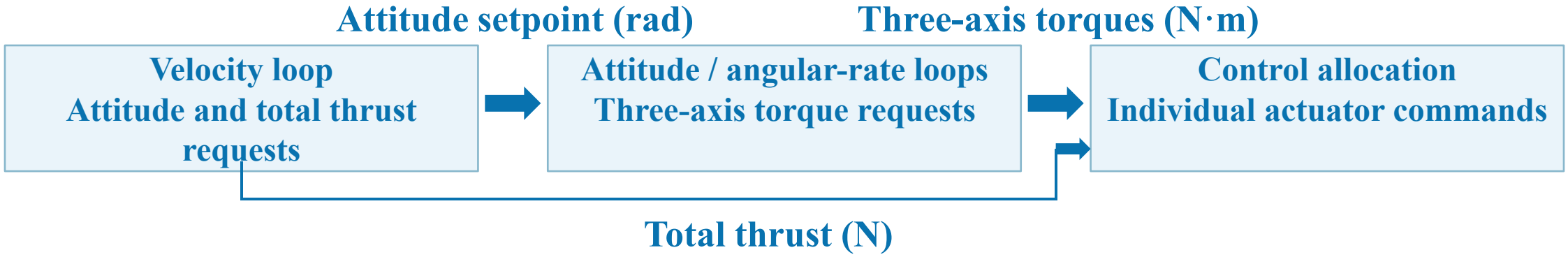
5.3.4 Actuator Control Allocation

Textbook: Sec. 5.1.5

5.3.3 Low-Level Control Architecture and Controller Types: System Framework

An Allocator Must Realize the Controller Requests

Textbook Fig. 5.2 (deck 5.1): full system. Here: intermediate control signals and units.

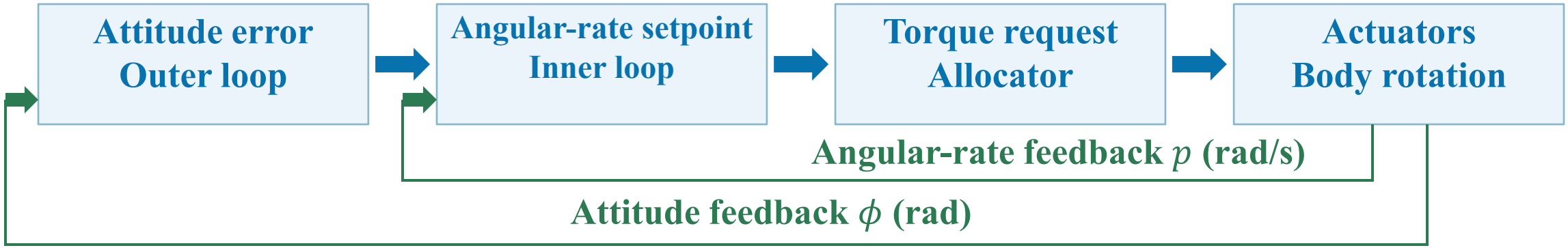


Module	Input	Output
Velocity loop	Velocity setpoint / feedback, m/s	Desired attitude, rad. Total thrust, N.
Attitude/rate loops	Attitude, rad. Angular rate, rad/s.	Desired three-axis torques, N·m
Allocator	Total thrust and three-axis torques	Individual motor speeds or other actuator quantities

5.3.3 Low-Level Control Architecture and Controller Types: Attitude Cascade

The Attitude Loop Sets Rate, the Inner Loop Sets Torque

Four controller types: PID, saturated PID, model-based, and intelligent. Build the cascade first, then identify which layer the last two replace.



$$p_{\mathrm{sp}} = K_{\phi}(\phi_{\mathrm{sp}} - \phi), \qquad \tau_x = K_{\omega}(p_{\mathrm{sp}} - p)$$

Feedback: ϕ (rad) returns to the outer loop; p (rad/s) to the inner loop.
Subscript sp denotes a setpoint. 3D attitude often uses quaternion error; inner-loop bandwidth must exceed outer-loop bandwidth.

5.3.3 Low-Level Control Architecture and Controller Types: Finite Inner-Loop Bandwidth

A Slower Rate Loop Increases Attitude Overshoot Here

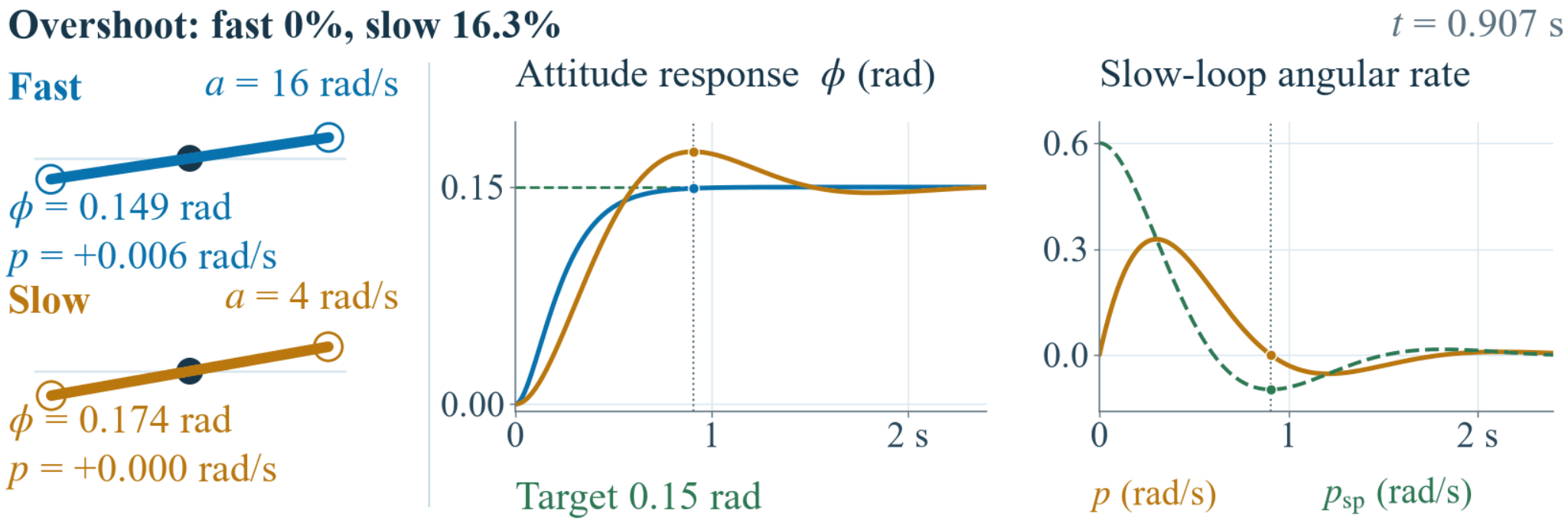


Figure: Fast/slow angular-rate loops, same attitude outer loop (simulation; no delay or limits)

Pure inertia: $\dot{\phi} = p, \quad J\dot{p} = \tau_x$. $J = 0.02$ kg·m², $K_\phi = 4$ s⁻¹, $\phi_{sp} = 0.15$ rad. Only bandwidth $a = K_\omega / J$ changes: 16 / 4 rad/s. At the setpoint, the slow case has $p > 0$ and keeps rotating into overshoot.

5.3.3 Low-Level Control Architecture and Controller Types: Position Cascade

Tilted Thrust Produces Horizontal Acceleration

The cascade converts position error 0.8 m into acceleration setpoint $a_x = 1.2$ m/s² through two gains.

Position error
Position loop
→
Velocity setpoint
Velocity loop
→
Acceleration setpoint
Attitude and thrust
→
Attitude inner loop
Actuators

Vertical velocity and acceleration are both zero here. Gravity and thrust are the only vertical forces.

$$T\sin\beta = ma_x, \quad T\cos\beta = mg, \quad \beta = \operatorname{atan2}(a_x, g)$$

2D convention: altitude positive upward; β tilts toward horizontal acceleration.
For $m = 1$ kg, $a_x = 1.2$ m/s² and $g = 9.8$ m/s²: $\beta = 6.98°$ and $T = 9.873$ N.
For textbook north-east-down (NED), recompute signs from desired direction and body axes.

5.3.3 Low-Level Control Architecture and Controller Types: Cascade Calculation

One Position Correction Produces 1.2 m/s²

Horizontal position setpoint: 1.0 m. Current position: 0.2 m. Velocity: 0.2 m/s. Outer loop: $K_x = 1\ \text{s}^{-1}$. Inner loop: $K_v = 2\ \text{s}^{-1}$.

$$v_{\text{sp}} = K_x(x_{\text{sp}} - x) = 1(1.0 - 0.2) = 0.8\ \text{m/s}$$

$$a_{\text{sp}} = K_v(v_{\text{sp}} - v) = 2(0.8 - 0.2) = 1.2\ \text{m/s}^2$$

The outer loop converts position error to velocity. The inner loop converts velocity error to acceleration. Teaching plant: $\dot{x} = v$, $\dot{v} = a$. Assume ideal acceleration tracking for now. See the full cascade response for the closed-loop result.

5.3.3 Low-Level Control Architecture and Controller Types: Cascade Correction

Above the Speed Setpoint, Tilt Reverses for Braking

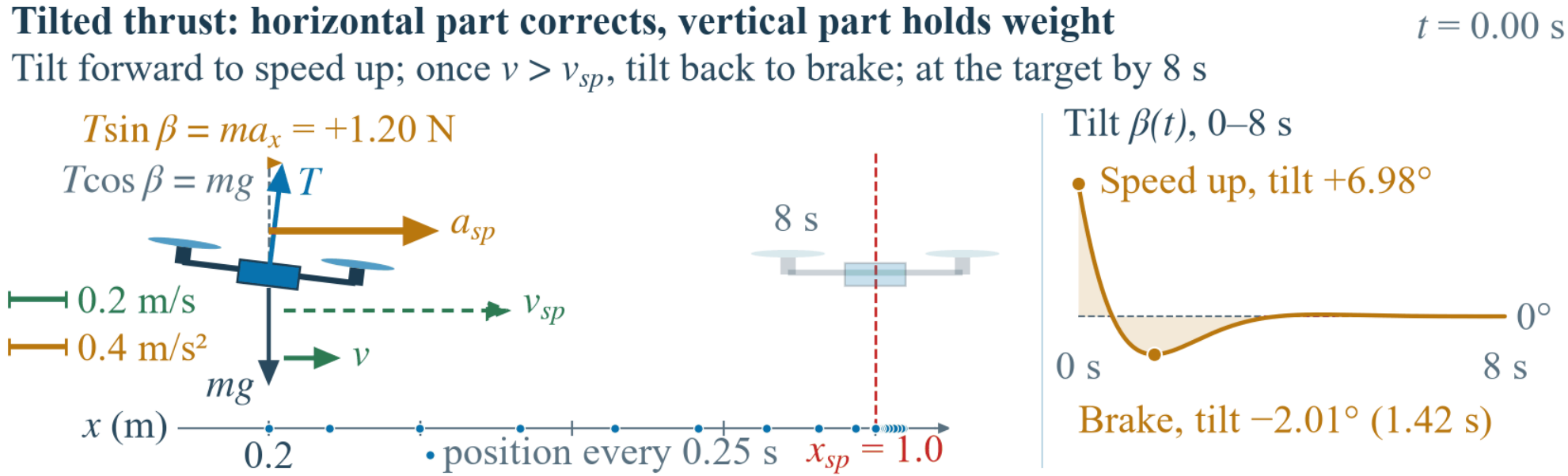


Figure: Position–velocity cascade convergence (teaching simulation)

Four paused steps: error 0.8 m → v_{sp}=0.8 m/s → a_{sp}=1.2 m/s² → forward tilt 6.98°, thrust 9.873 N.

Replay: $v < v_{sp}$ gives forward a_{sp} and accelerating forward tilt. When $v > v_{sp}$, a_{sp} reverses; about 2° backward tilt brakes.

5.3.3 Low-Level Control Architecture and Controller Types: Full Cascade Response

Position Converges to the Setpoint with the Same Parameters

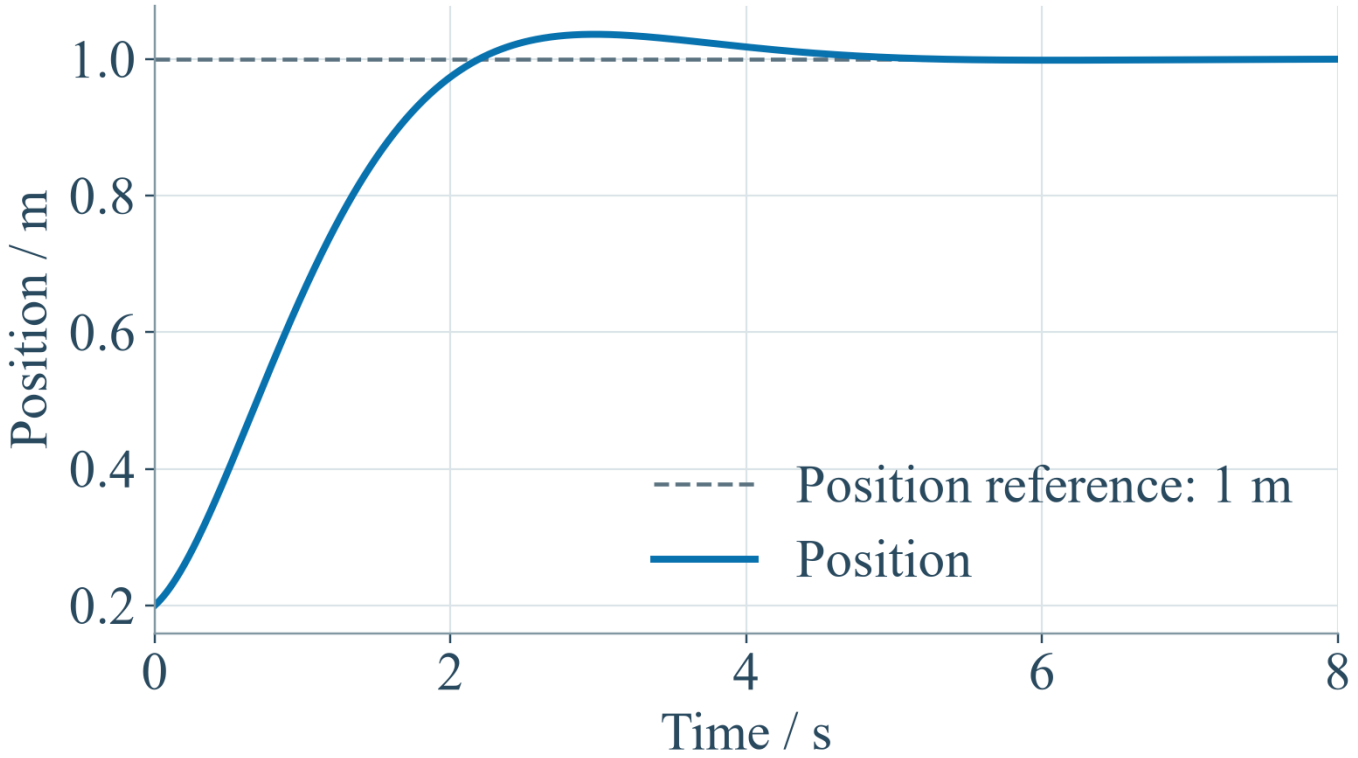


Figure: Teaching calculation of the position–velocity cascade

Separate horizontal-motion calculation.
Setpoint: 1 m. Initial position: 0.2 m.
Initial speed: 0.2 m/s. Step: 0.01 s.
$K_x = 1\ \mathrm{s}^{-1}$, $K_v = 2\ \mathrm{s}^{-1}$
Position at 8 s: 0.9999 m
Peak: 1.0363 m
Initial acceleration: 1.2 m/s²

5.3.3 Low-Level Control Architecture and Controller Types: Model-Based Controllers

Pole Placement and LQR: Design Objectives

Method	Required Model and Design Choices	Result
Pole placement	Linear model. Specify eigenvalues of A_{cl} (closed-loop poles).	Feedback gains that shape decay rate and oscillation
Linear quadratic regulator (LQR)	Linear model, quadratic state / input weights	Optimizes state and input cost for the given problem

$$u = -Kx, \quad A_{\mathrm{cl}} = A - BK$$

x: state. A, B: state and input matrices. K: feedback gain. Either method can replace PID at one cascade layer. Gain meaning depends on the model, states, and units. See “Two State-Feedback Design Methods” in slide deck 5.4 for solutions and examples.

5.3.3 Low-Level Control Architecture and Controller Types: Sliding Mode and Predictive Control

Models Support Robust Control and Constrained Optimization

Method	Mechanism	Practical Conditions
Sliding mode control	A sliding surface defines a chosen state relation. Switching first approaches that surface, then converges along it to the target, rejecting matched disturbances.	Matched: disturbance uses the control-input channel. Chattering: rapid switching causes high-frequency fluctuations. Also check discrete sampling.
Model predictive control (MPC)	k: apply | k+1 | k+2 | … | k+N → Shift the horizon Predict/optimize N steps; apply first only. Constraints, solve time, feasibility.	

Build the model → **Set objectives and constraints** → **Design the control law** → **Validate under the same conditions**

Models: physics or data (e.g., least squares). Test in software-in-the-loop (SITL; textbook SIL) and hardware-in-the-loop (HITL; textbook HIL), then fine-tune on hardware.

5.3.3 Low-Level Control Architecture and Controller Types: Intelligent Controllers

Learning and Search Enter at Different Points

Method	Role in the Chain	What Is Learned or Searched
Fuzzy control	Control law	Expert rules and defuzzified output
Neural networks	Model or control law	Nonlinear mappings in data
Genetic algorithms	Often offline parameter design	Candidate parameters or structures, and the evaluation function
Reinforcement learning	Policy trained through interaction	Reward-guided action selection

Online control, offline search, and model learning serve different roles. Performance depends on training coverage, constraints, and closed-loop validation.

Chapter Outline

Controller Design and Control Allocation

Chapter 5
Filtering, State Estimation,
and Low-Level Control

5.1 Filtering and Low-Level Control

5.2 Sensor Calibration and Estimation

5.3 Controller Design and Allocation

5.4 Control Performance and Tuning

5.5 PX4 Architecture and Interfaces

5.6 Control Development and Testing

5.7 Other Vehicles and Advanced Cases

5.8 Exercises and Classroom Discussion

4 Units in This Section

5.3.1 Proportional-Integral-Derivative (PID) Controllers

Textbook: Sec. 5.1.3.1

5.3.2 Saturated PID Controllers

Textbook: Sec. 5.1.3.1

5.3.3 Low-Level Control Architecture and Controller Types

Textbook: Secs. 5.1.2.4, 5.1.3.1

5.3.4 Actuator Control Allocation

Textbook: Sec. 5.1.5

5.3.4 Actuator Control Allocation: Allocation Model

Each Motor Is One Column of Thrust and Three-Axis Torque

$$\mathbf{u} = [f, \tau_x, \tau_y, \tau_z]^\top = \mathbf{M}_{n_r}\mathbf{P}, \quad \mathbf{P} = [\varpi_1^2, \ldots, \varpi_{n_r}^2]^\top \qquad \text{Eq. 5.21}$$

n_r: rotor count. $P_i = \varpi_i^2$: squared rotor speed, in (rad/s)². f: total thrust (N).

M maps squared rotor speeds to total thrust and three-axis torques. P is not normalized throttle.

Matrix Row	Physical Quantity	Meaning of Each Entry
Total thrust	N	This rotor's thrust coefficient
Roll / pitch torque	N·m	Thrust coefficient × signed moment arm
Yaw torque	N·m	Reaction-torque contribution, set by rotation direction

5.3.4 Actuator Control Allocation: Numbering and Signs

Body Coordinates Determine the Quadcopter Column Order

Motor	Position $[x_i, y_i]$, m	Roll / Pitch / Yaw Signs
1 Front right	$[+0.1, +0.1]$	$-/+/+$
2 Rear right	$[-0.1, +0.1]$	$-/-/-$
3 Rear left	$[-0.1, -0.1]$	$+/-/+$
4 Front left	$[+0.1, -0.1]$	$+/+/-$

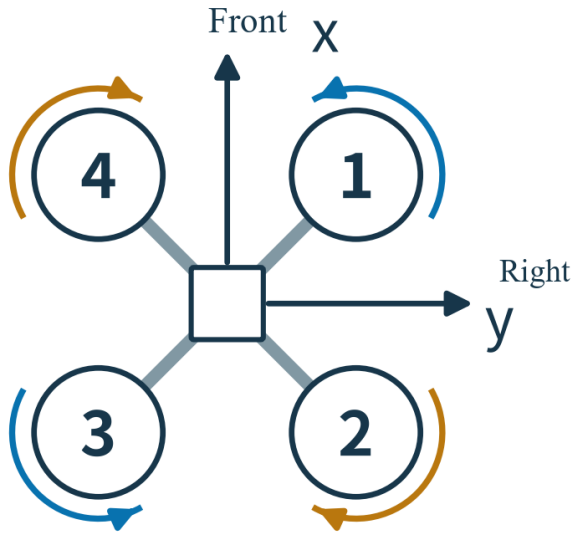


Figure: Top View and Rotor Spin

Top view. Body axes: forward, right, down. Motors are numbered clockwise.
Positive thrust: body −z. Motors 1/3 give positive yaw reaction torque; 2/4 spin oppositely.

5.3.4 Actuator Control Allocation: Solvability and Redundancy

Actuator Count and Full Matrix Rank Both Matter

Speed Channels	Rank and Solution	Design Meaning
Fewer than 4	$\text{rank}(M) \leq n_r < 4$	Cannot independently control all four quantities
4, full rank	Invertible square matrix, unique solution	Geometry and reaction-torque effectiveness must be valid
More than 4, full row rank	Generally multiple exact solutions	Allocate by norm, constraints, or other costs

Control effectiveness is the force or torque per unit actuator quantity. Tilt, servos, and tail rotors also provide effectiveness.
Even at full rank, the request may exceed actuator limits.

5.3.4 Actuator Control Allocation: Pseudoinverse

Meaning and Conditions of the Minimum-Norm Solution

$$\mathbf{P} = \mathbf{M}^{\dagger}\mathbf{u} = \mathbf{M}^{\top}(\mathbf{M}\mathbf{M}^{\top})^{-1}\mathbf{u}$$

Eq. 5.22

This closed form requires $\mathrm{rank}(M) = 4$ so that $MM^{\top}$ is invertible.
Without actuator constraints, it selects the exact solution with minimum $\|P\|_2$.
Energy optimization also needs motor, propeller, and electronic speed controller (ESC) power models. This norm measures only squared-speed magnitude.

$$\mathbf{P} = \mathbf{M}^{\dagger}\mathbf{u} + (\mathbf{I} - \mathbf{M}^{\dagger}\mathbf{M})\boldsymbol{\eta}$$

$\eta \in \mathbb{R}^{n_r}$ is a freely chosen vector. I is the $n_r \times n_r$ identity matrix.
Since $M(I - M^{\dagger}M)\eta = 0$, the null-space term leaves the requested thrust and three-axis torques unchanged.

5.3.4 Actuator Control Allocation: Dimensional Allocation

Realizing 10 N Thrust and 0.2 N·m Roll Torque

Teaching coefficients: $c_T = 10^{-5}$ N/(rad/s)², $c_M = 2 \times 10^{-7}$ N·m/(rad/s)². Lateral and longitudinal moment arms are both 0.1 m.

$$\mathbf{M} = 10^{-5}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -0.1 & -0.1 & 0.1 & 0.1 \\ 0.1 & -0.1 & -0.1 & 0.1 \\ 0.02 & -0.02 & 0.02 & -0.02 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} 10 \\ 0.2 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} 200000 \\ 200000 \\ 300000 \\ 300000 \end{bmatrix} (\mathrm{rad/s})^2, \quad \mathbf{MP} = \begin{bmatrix} 10 \\ 0.2 \\ 0 \\ 0 \end{bmatrix}$$

The sum of squared speeds is 10^6. The roll difference $\times$ 0.1×10^{-5} gives 0.2 N·m.

5.3.4 Actuator Control Allocation: Rotor Speed to Throttle

After the Square Root, Invert the Steady-State Motor Model

$$\varpi_i = C_R \sigma_i + \varpi_{\text{ss}} \qquad \text{Eq. 5.24}$$

$$\sigma_{i,\text{raw}} = \frac{\sqrt{P_i^*} - \varpi_{\text{ss}}}{C_R} \qquad \text{Eq. 5.25}$$

$P_i^\star$ is the requested squared speed, in (rad/s)². Denote the achieved value after saturation by $P_{i,\text{act}}$. Here $C_R = 800$ rad/s and $\varpi_{\text{ss}} = 100$ rad/s. Take square roots only for nonnegative requests.

$$\varpi_{1,2} = 447.214 \text{ rad/s}, \quad \sigma_{1,2} = 0.4340; \quad \varpi_{3,4} = 547.723 \text{ rad/s}, \quad \sigma_{3,4} = 0.5597$$

5.3.4 Actuator Control Allocation: Two Constraint Layers

Both Squared Speed and Throttle Must Respect Physical Limits

$$\mathrm{sat}_{[l,u]}(x) = \min(u, \max(l, x))$$

Eq. 5.23

$$\sigma_i = \mathrm{sat}_{[0,1]}\left(\frac{\sqrt{\mathrm{sat}_{[0,P_{\max}]}((\mathbf{M}^{\dagger}\mathbf{u})_i)} - \varpi_{\mathrm{ss}}}{C_R}\right)$$

Eq. 5.26

Solve for squared speed → **Zero and upper bounds** → **Take square root, solve for throttle** → **Clip throttle to range**

This page uses $\sigma_i \in [0,1]$ and steady-state model $\varpi_i = C_R\sigma_i + \varpi_{\mathrm{ss}}$. Use the calibrated mapping for the actual ESC.

Subtract ϖ_{ss} outside the square root. After clipping, recompute achieved thrust, three-axis torques, and residuals.

5.3.4 Actuator Control Allocation: Normalizing Physical Commands

Reference Maxima Set the Channel Scales

$$\boldsymbol{\delta} = \begin{bmatrix} \mathrm{sat}_{[0,1]}(f/f_{\max}) \\ \mathrm{sat}_{[-1,1]}(\tau_x/\tau_{x,\max}) \\ \mathrm{sat}_{[-1,1]}(\tau_y/\tau_{y,\max}) \\ \mathrm{sat}_{[-1,1]}(\tau_z/\tau_{z,\max}) \end{bmatrix} \qquad \text{Eq. 5.27}$$

$$f = 10\ \mathrm{N},\ f_{\max} = 20\ \mathrm{N} \Rightarrow \delta_T = 0.5; \quad \tau_x = 0.2\ \mathrm{N\,m},\ \tau_{x,\max} = 2\ \mathrm{N\,m} \Rightarrow \delta_\phi = 0.1$$

Reference values must match the interface: calibrated capability or a tuning scale.
Thrust: 0 to 1. Torque channels: −1 to 1.
Even with all channels in range, the combined motor commands may exceed limits.

5.3.4 Actuator Control Allocation: General Channel Mixing

Control Effectiveness and Channel Weights Set the Coefficients

$$\sigma_{i,\text{raw}} = K_T\delta_T + K_\phi(-d_i \sin\varphi_i)\delta_\phi$$

$$+K_\theta(d_i \cos\varphi_i)\delta_\theta + K_\psi((-1)^{i+1}k_m)\delta_\psi \qquad \text{Eq. 5.28}$$

Quantity	Meaning	Contribution
d_i, φ_i	Motor mounting radius and azimuth	Signed moment arms for roll and pitch
$k_m = c_M/c_T$	Reaction-torque / thrust coefficient ratio	Yaw contribution
$K_T, K_\phi, K_\theta, K_\psi$	Channel contribution weights	Combine geometry with interface scales

5.3.4 Actuator Control Allocation: Normalized Quadcopter Allocation

One Roll Request Produces Four Motor Commands

$\boldsymbol{\delta}$: normalized total-thrust and three-axis torque request (Eq. 5.27). k: combined tuning coefficients.

$$\begin{bmatrix}\sigma_{1,\mathrm{raw}}\\ \sigma_{2,\mathrm{raw}}\\ \sigma_{3,\mathrm{raw}}\\ \sigma_{4,\mathrm{raw}}\end{bmatrix}=\begin{bmatrix}1 & -1 & 1 & 1\\ 1 & -1 & -1 & -1\\ 1 & 1 & -1 & 1\\ 1 & 1 & 1 & -1\end{bmatrix}\begin{bmatrix}k_T\delta_T\\ k_\phi\delta_\phi\\ k_\theta\delta_\theta\\ k_\psi\delta_\psi\end{bmatrix} \qquad \text{Eq. 5.29}$$

$$k_T=k_\phi=k_\theta=k_\psi=1,\quad \boldsymbol{\delta}=\begin{bmatrix}0.5\\ 0.1\\ 0\\ 0\end{bmatrix}\Rightarrow \boldsymbol{\sigma}_{\mathrm{raw}}=\begin{bmatrix}0.4\\ 0.4\\ 0.6\\ 0.6\end{bmatrix}$$

Normalized teaching mixer: positive roll lowers right 1/2, raises left 3/4. The 0.1 is not in N·m.

5.3.4 Actuator Control Allocation: Clipping Residual

After Clipping, Achieved Roll Falls Short by 0.2

$$\boldsymbol{\delta} = \begin{bmatrix} 0.5 \\ 0.7 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \boldsymbol{\sigma}_{\text{raw}} = \begin{bmatrix} -0.2 \\ -0.2 \\ 1.2 \\ 1.2 \end{bmatrix} \Rightarrow \boldsymbol{\sigma} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$$\boldsymbol{\delta}_{\text{achieved}} = \mathbf{A}^{-1}\boldsymbol{\sigma} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{r} = \boldsymbol{\delta} - \boldsymbol{\delta}_{\text{achieved}} = \begin{bmatrix} 0 \\ 0.2 \\ 0 \\ 0 \end{bmatrix}$$

A is the simplified mixer matrix in Eq. 5.29, with all channel coefficients set to 1 here. The total-thrust request and achieved value are both 0.5. Roll is requested at 0.7, but only 0.5 is achieved after clipping. These achieved values belong to the normalized teaching model.

5.3.4 Actuator Control Allocation: Clipping and Residuals in Motion

The Request Keeps Rising, but Achieved Roll Stops at 0.5

Roll request 0.70 → clipped $\sigma = [0, 0, 1, 1]$ → achieved 0.50, residual 0.20

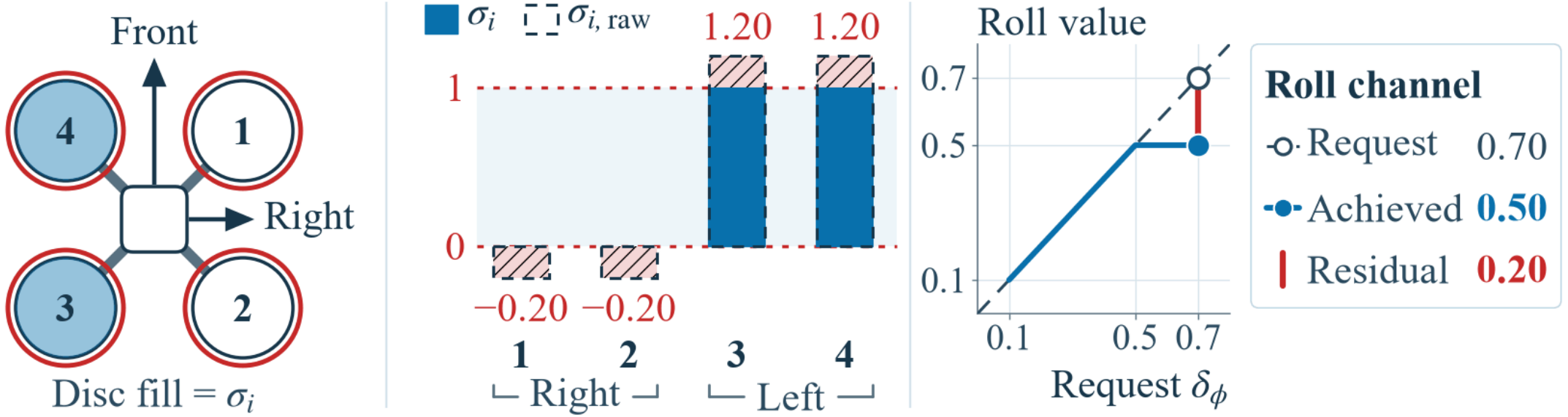


Figure: Eq. 5.29 normalized mixer, clipping and mapped-back residuals (teaching example)

For roll requests above 0.5, $\sigma_{i,\text{raw}}$ exceeds 0–1 and clips to $\boldsymbol{\sigma} = [0,0,1,1]^{\mathsf{T}}$.

Mapping back with $\mathbf{A}^{-1}\boldsymbol{\sigma}$ gives roll 0.5; request 0.7 leaves residual 0.2 (normalized).

5.3.4 Actuator Control Allocation: Pulse-Width Modulation (PWM)

Pulse-Width Mapping Also Needs a Device Range Convention

$$\sigma_{i,\mathrm{pwm}} = \begin{cases} 1000 + 1000\sigma_i, & \text{motor } \sigma_i \in [0,1] \\ 1500 + 500\sigma_i, & \text{servo } \sigma_i \in [-1,1] \end{cases} \qquad \text{Eq. 5.30}$$

$$\sigma_{\mathrm{motor}} = 0.6 \Rightarrow 1600\ \mu s, \qquad \sigma_{\mathrm{servo}} = -0.2 \Rightarrow 1400\ \mu s$$

Pulse-width modulation (PWM) encodes commands by time at logic high. The 1000–2000 μs range here is a teaching convention.
Servo center is 1500 μs. Motor and servo normalized ranges differ.
Actual range, direction, update rate, and arming requirements follow the device configuration.

5.3.4 Actuator Control Allocation: Digital Interfaces

A Protocol Name Does Not Define the Command Quantity

Interface	Quantity Transmitted	Usage Conditions
PWM	Pulse width defined by the device	Match range, direction, and center
DShot	Digital ESC commands: throttle and special commands	Optional speed telemetry: feedback. Closed loop: depends on the ESC.
DroneCAN	RawCommand: normalized raw value RPMCommand: speed target (rpm)	Check the message type and ESC implementation first

Before choosing an interface, check the message type, its command quantity (throttle, speed, or normalized value), and whether the ESC implements that message.

5.3.4 Actuator Control Allocation: The Complete Closed Loop

From Angle Setpoint to Torque, Motor Output, and Response

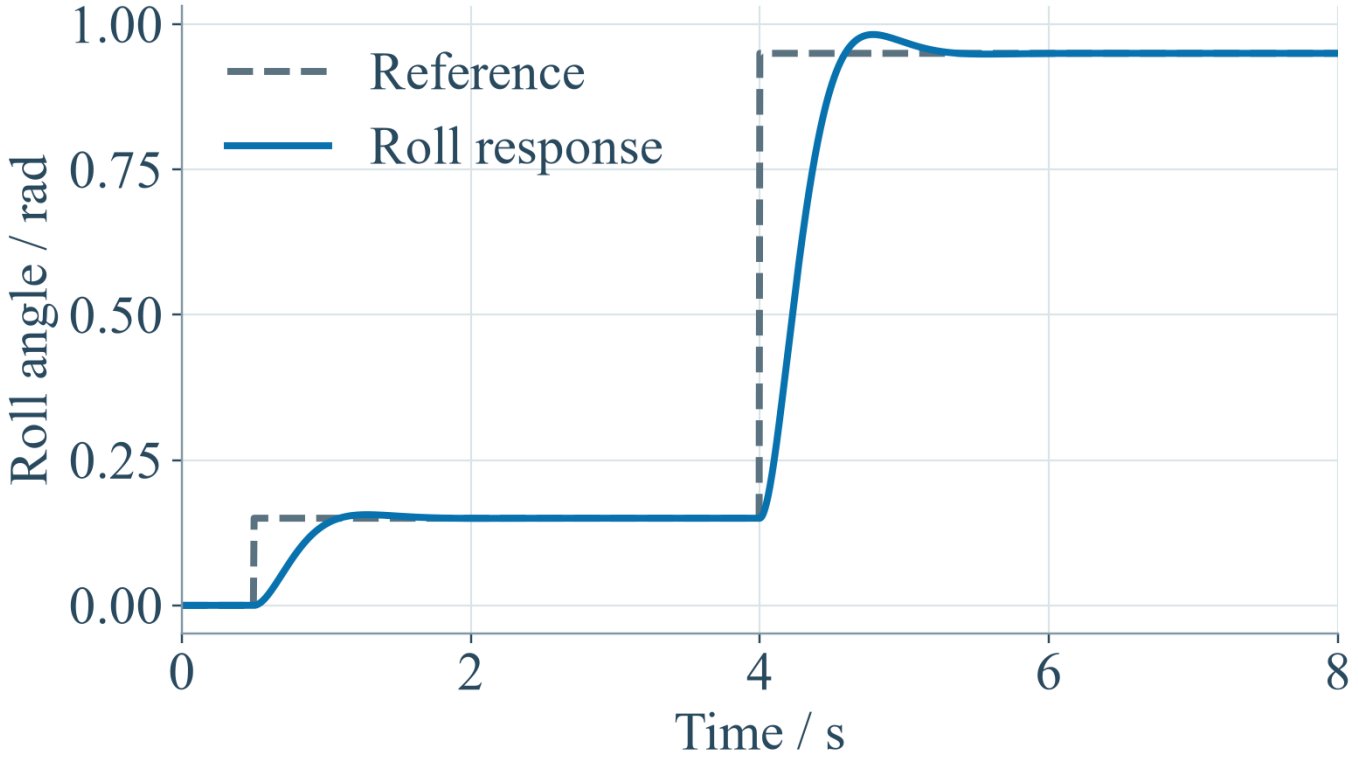


Figure: Teaching calculation of a single-axis roll closed loop

Single-axis rotation. Altitude is not evaluated.

$\dot{\phi} = p, \dot{p} = \tau_x / J_x$

$J_x = 0.02$ kg·m²

Attitude gain: 4 s^{-1}.

Angular-rate gain: 0.16 N·m·s/rad.

Squared-speed limits:

$P_i \in [10^4, 3.6 \times 10^5]$

Units: (rad/s)².

Angle at 8 s: 0.9500 rad.

5.3.4 Actuator Control Allocation: Achieved Output During Saturation

Torque Request and Output Briefly Separate at a Setpoint Step

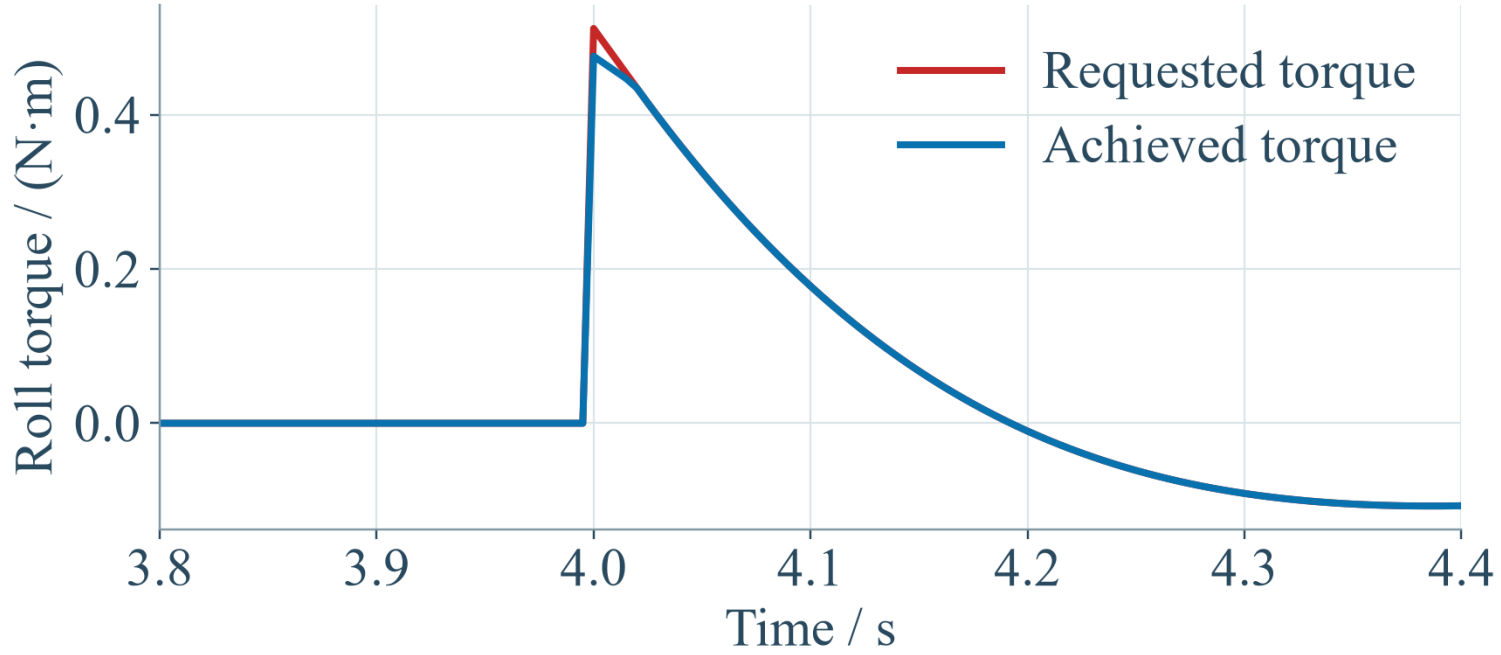


Figure: Roll torque before and after allocation

Same calculation, enlarged near 4 s.
Peak request: 0.512 N·m
Peak achieved: 0.476 N·m
Maximum residual: 0.036 N·m
Minimum total thrust: 9.640 N
Step: 0.005 s. Clipping occurs at 4 samples.

Actual thrust and torque after allocation determine how much of the control intent is realized.

5.3.4 Actuator Control Allocation: Example Entry and Requirements

Actuator Control Examples: Entry, Conditions, and Inputs

Directory: [安装目录]/RflySimAPIs/5.RflySimFlyCtrl/
0.ApiExps/16.CtrlsSingalsAPI/ (read Readme.pdf first)

Exp1_HIL16CtrlsPWM.slx: 16-dimensional PWM commands, range 1000–2000.
Exp2_HIL16CtrlsNorm.slx: 16-dimensional normalized actuator commands.
Exp3_TorqueThrustCtrls4D.slx: normalized three-axis torque and thrust.

Requirements: hardware-in-the-loop (HITL), a flight controller, transmitter, and receiver. Procedure: “Actuator Control Example” in slide deck 5.5.

Software: MATLAB 2022b or later. Build target: px4_fmu-v6x_default.

Firmware: Readme recommends PX4 1.12.3. To confirm the version in use, see “Control Outputs Across Versions” in slide deck 5.5.

Exp3 inputs: all three torques and thrust are normalized. Thrust 1D greater than 0 means upward. Normalize requests in N and N·m according to the interface. See “Actuator Control API” in slide deck 5.5 for input semantics.

5.3.4 Actuator Control Allocation: Where Allocation Occurs in the Examples

The Three Models Place Allocation in Simulink or PX4

Exp1 and Exp2: The model outputs 16-dimensional actuator commands (PWM 1000–2000 or normalized values), sent via actuator_outputs_rfly to CopterSim inPWMs, bypassing the PX4 mixer.
Exp3: The model supplies only normalized three-axis torque and thrust. In v1.12.3, these enter actuator_controls_0, and the PX4 mixer computes individual motor commands.
Connection to this unit: For allocation matrices and clipping implemented in Simulink, connect at the Exp1 or Exp2 layer. To supply only torque and thrust, use Exp3 and let the PX4 mixer allocate.

Expected behavior: All three models use the same procedure. Readme states that Exp2 and Exp3 should behave like Exp1: CH5 at its highest position gives Simulink control. Returning CH5 to its lowest position restores PX4 control.

Full procedure, criteria, and common failures: “Actuator Control Example” in slide deck 5.5. CH5 at its highest position automatically arms the flight controller. Operate only in hardware-in-the-loop simulation.

Classroom Exercise

Does the Original Torque Still Hold After Clipping?

Exercise: Eq. 5.29 mixer, coefficients 1. Request total thrust 0.8, roll 0.4, pitch / yaw 0. Clip motor commands to 0–1. Find the achieved normalized thrust/torque vector and its residual.

Required: Write σ_{raw}, the clipped σ, the back-calculated δ_{achieved}, and the residual. Then decide whether clipping affects only roll. Compare with the “Clipping Residual” example.

Suggested Answers

Clipping Changes Both Total Thrust and Roll

Problem recap: Simplified matrix in Eq. 5.29, coefficients 1. Total thrust 0.8, roll 0.4. Clip motor commands to 0–1.

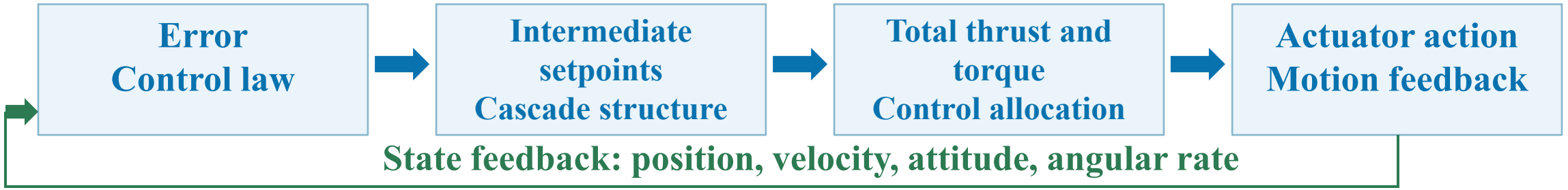
$$\boldsymbol{\sigma}_{\text{raw}} = [0.4,0.4,1.2,1.2]^{\mathsf{T}} \Rightarrow \boldsymbol{\sigma} = [0.4,0.4,1.0,1.0]^{\mathsf{T}}$$

$$\boldsymbol{\delta}_{\text{achieved}} = [0.7,0.3,0,0]^{\mathsf{T}}, \quad \boldsymbol{\delta} - \boldsymbol{\delta}_{\text{achieved}} = [0.1,0.1,0,0]^{\mathsf{T}}$$

Clipping changes both total thrust and roll. Request, allocate, limit, map back, and compare.

Section Summary

Each Control Layer Performs Its Own Calculation



Stage	Key Checks	Common Errors
Control law	Error, time, and output units Integral handling during saturation	D amplifies noise. I omits $\times\Delta t$. I keeps accumulating in saturation.
Cascade structure	Setpoint/feedback quantity by layer Specify the layer being replaced.	Treating position error directly as throttle
Allocator	Column order, propulsion model, and constraints	Mixing normalization conventions, ignoring clipping residuals