Topic 12

University Mechanical Engineering

What this course is, and how to use it

This is the twelfth and last course in University Mechanical Engineering: **system dynamics** and **control** – how machines respond over time, and how feedback makes them do what we want. The project closes the cooling loop: a small dc motor turns a ball valve through a 50:1 gearbox, and a controller sets the valve from the loop temperature.

Scope limit, stated once: a study course, not a degree; there is no hardware. Motor, sensor and converter data are typical catalogue values, and every simulated response in this course is checked against an analytical result at a checkpoint (a steady state, an overshoot or a time constant).

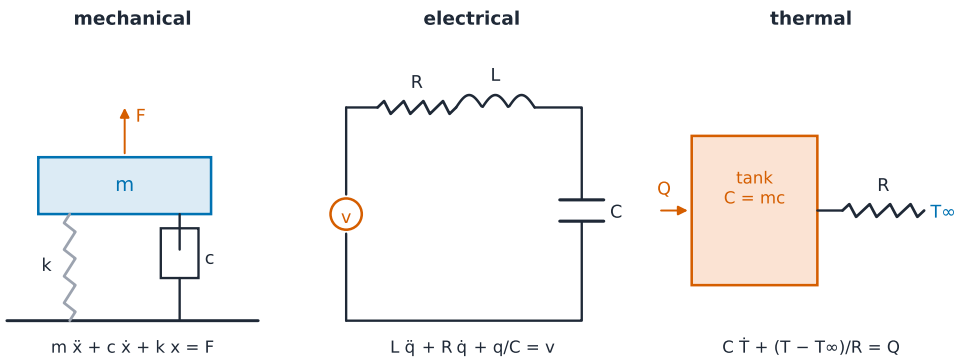
- Units 12.1–12.3 build models and read their responses.
- Units 12.4–12.6 close the loop, put it on a microcontroller and choose its sensors and actuators.
- Unit 12.7 designs and verifies the valve positioner against a spec.

The habit: **write the model's assumptions before its equations** – a model is only as good as the effects it leaves out, and you must know which ones they are.

Lumped-parameter modelling

Why it matters. The cooling loop has masses, springs, wires, motors and tanks of hot water. If each needs its own mathematics, the controller designer must learn five subjects; with **lumped parameters** they become one.

The idea, with a picture. Concentrate each effect in an ideal element: a mass, spring or damper; an inductor, capacitor or resistor; a thermal capacitance or resistance. Each domain has an effort (force, voltage, temperature) and a flow (velocity, current, heat flow), and the elements obey the same three laws: store energy in two ways, dissipate it in one. The equations then have the same form, so one method solves them all – the **state-space form** $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u$.



Mass-spring-damper, RLC circuit and heated tank: three domains, one equation form

The derivation (from equation to transfer function). Take the Laplace transform of a linear model with zero initial conditions: $m\ddot{x} + c\dot{x} + kx = F$ becomes $(ms^2 + cs + k)X(s) = F(s)$. The **transfer function** is the output over the input, $G(s) = X/F = 1/(ms^2 + cs + k)$. For the heated tank, $C\dot{T} + (T - T_\infty)/R = Q$ gives $G(s) = \Delta T/Q = R/(RCs + 1)$: a first-order lag with **time constant** $\tau = RC$.

> **Worked example 1 – the loop's reservoir.** 200 L of water: $C = mc = 200 \times 4180 =$ **836** kJ/K. It loses heat to the room through $UA = 100$ W/K, so $R =$ **0.01** K/W, and $\tau = RC =$ **8360** s = 2.3 h. A 1 kW heat input raises it by $QR =$ **10** K in the end – 63% of that after 2.3 h, 95% after 7 h. Assumptions, stated: the water is well mixed (one temperature), UA is constant, and the tank wall stores no heat.

> **Worked example 2 – the pump on its mounts as state space.** With $x_1 = x$, $x_2 = \dot{x}$ and Course 11's numbers ($m = 120$ kg, $c = 910$ N · s/m, $k = 300$ kN/m): $\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -2500 & -7.58 \end{bmatrix}$, $\mathbf{B} = \begin{bmatrix} 0 \\ 1/120 \end{bmatrix}$. The eigenvalues of $\mathbf{A}$ are $-3.79 \pm 49.86j$: the same $\zeta\omega_n$ and ω_d as the decay record. One matrix form now holds both the mount and, below, the motor.

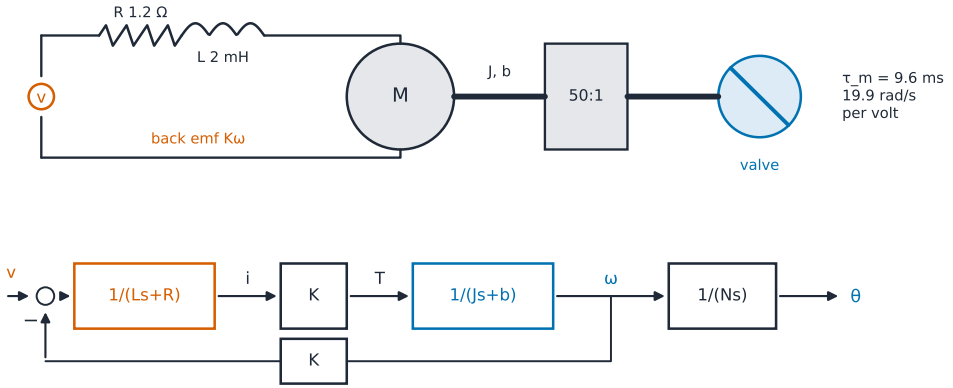
Common mistakes. Mixing up effort and flow between domains. Forgetting the assumption that makes a lump valid (a tank is one temperature only if it is mixed). Taking a Laplace transform of a nonlinear model.

Summary. Store, store, dissipate in every domain; $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u$; $G(s) =$ output over input with zero initial conditions; first order $R/(RCs + 1)$.

Electromechanical systems

Why it matters. The valve actuator is where electrical energy becomes motion. Its two time constants and its limits – voltage, current, speed – decide how fast the valve can move.

The idea, with a picture. A permanent-magnet **dc motor** has an armature circuit and a rotor, joined by two constants: torque $T = K_t i$ and **back emf** $e = K_e\omega$. In SI units $K_t = K_e = K$, because the electrical power ei equals the mechanical power $T\omega$ – the energy conversion is exact.



The valve actuator: dc motor circuit, rotor and gearbox, and the block diagram they make

The derivation (speed per volt). The armature gives $V = (Ls + R)I + K\Omega$; the rotor gives $(Js + b)\Omega = KI - T_{load}$. Eliminate I with $T_{load} = 0$: $\dfrac{\Omega}{V} = \dfrac{K}{(Js+b)(Ls+R)+K^2}$. If L is small, this is first order with gain $K/(bR + K^2)$ and **mechanical time constant** $\tau_m = JR/(bR + K^2) \approx JR/K^2$; the electrical one is $\tau_e = L/R$.

> **Worked example 1 – the valve motor.** $R = 1.2\,\Omega$, $L = 2$ mH, $K = 0.05$ N·m/A, $J = 2\times10^{-5}$ kg·m^2, $b = 10^{-5}$ N·m·s/rad. Gain $= 0.05/(1.2\times10^{-5}+0.0025) =$ **19.9** rad/s per volt, so 12 V gives 239 rad/s (2280 rpm). $\tau_m =$ **9.6** ms, $\tau_e =$ **1.7** ms. The exact poles are -135 and -466 s^{-1} (7.4 and 2.1 ms): τ_e is not quite negligible, but the sum is the same, and the 9.6 ms model is good below about 100 rad/s. Stall: $12/1.2 = 10$ A and 0.5 N · m.

> **Worked example 2 – through the gearbox to the valve.** With $N = 50$ the valve sees $19.9/50 =$ **0.398** rad/s per volt, so at the 12 V limit it turns at most at **4.78** rad/s: a 90° move takes at least **0.33** s. A load torque of 5 N · m on the valve stem (flow forces and seal friction) reaches the motor as $5/(50\times0.8) = 0.125$ N · m, which the motor balances with $0.125\times1.2/0.05 =$ **3.0** V. A 5 A current limit gives $0.25\times50\times0.8 = 10$ N · m at the valve: the model is valid only while the current stays below it.

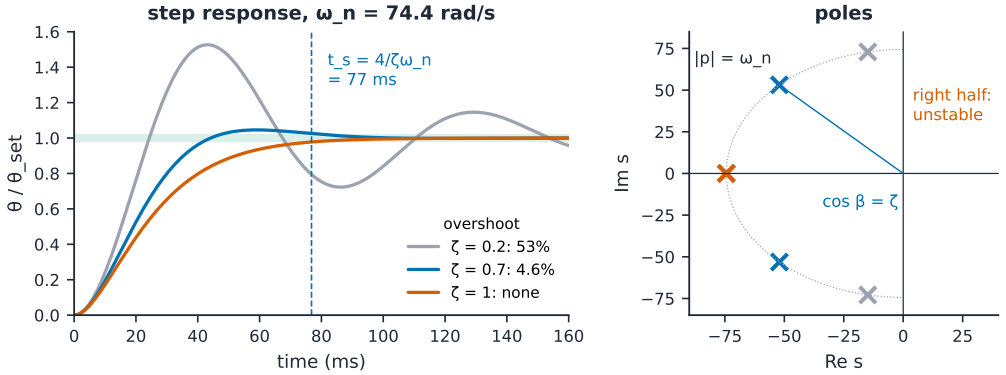
Common mistakes. Using different numbers for K_t and K_e in SI. Forgetting the back emf, which is what limits speed. Applying the linear model after the voltage or current has saturated.

Summary. $T = Ki$, $e = K\omega$; $\Omega/V = K/[(Js + b)(Ls + R) + K^2]$; $\tau_m \approx JR/K^2$, $\tau_e = L/R$; limits decide where the model stops.

Transfer functions and response

Why it matters. Before closing any loop we must be able to look at a transfer function and say how its output will move: how fast, how much overshoot, and whether it settles at all.

The idea, with a picture. The roots of the denominator are the **poles**, the roots of the numerator the **zeros**. A second-order system $\omega_n^2/(s^2 + 2\zeta\omega_n s + \omega_n^2)$ has poles at $-\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$: the distance from the origin is ω_n, and the angle from the negative real axis is $\cos^{-1}\zeta$. Poles in the left half-plane decay – the system is **stable**; one in the right half-plane grows.



Second-order step responses for three damping ratios, and where their poles sit

The derivation (overshoot from ζ). The step response is $1 - e^{-\zeta\omega_n t}[\cos\omega_d t + \frac{\zeta}{\sqrt{1-\zeta^2}}\sin\omega_d t]$. Its first peak is at $t_p = \pi/\omega_d$, where the **overshoot** is $M_p = e^{-\zeta\pi/\sqrt{1-\zeta^2}}$. It stays within 2% after about $t_s = 4/(\zeta\omega_n)$. A zero added in the left half-plane speeds the response and adds overshoot.

> **Worked example 1 – the position loop with P control.** The valve plant is $G = 0.398/[s(0.0096s + 1)]$. With proportional gain K_p the closed loop is $0.398K_p/(0.0096s^2 + s + 0.398K_p)$. For $\zeta = 0.7$: $0.398K_p = 1/(4\times0.49\times0.0096) = 53.1$, so $K_p =$ **134** V/rad, $\omega_n =$ **74.4** rad/s, overshoot **4.6**%, $t_s =$ **77** ms – for small steps, where the voltage stays below 12 V.

Worked example 2 – the frequency response. Put $s = j\omega$. The open loop $L = 53.1/[j\omega(0.0096j\omega + 1)]$ has $|L| = 1$ at the **crossover** $\omega_c = \mathbf{48.2}$ rad/s, where its phase is $-90° - \tan^{-1}(0.46) = -114.8°$: a **phase margin** of **65.2**°. The rule of thumb $\zeta \approx PM/100 = 0.65$ agrees with the 0.7 we designed for. Both poles of the closed loop are in the left half-plane for every $K_p > 0$: this loop cannot be made unstable by gain alone – but delay (unit 12.5) changes that.

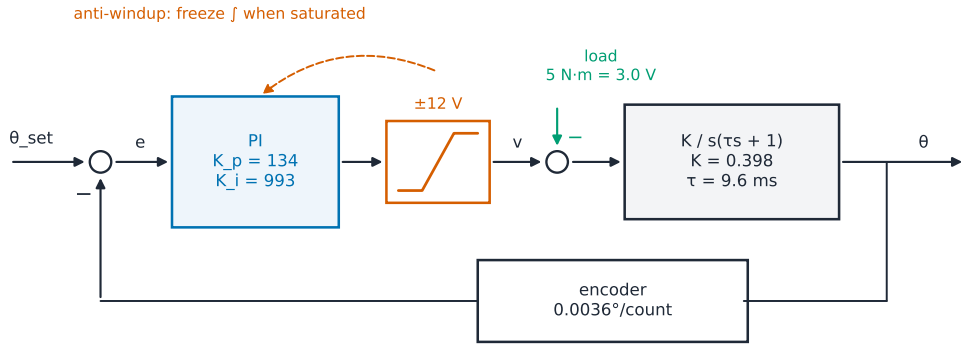
Common mistakes. Reading overshoot from ζ for a system with extra zeros or poles nearby. Using $t_s = 4/\zeta\omega_n$ when the actuator saturates. Calling a system with a pole at the origin "stable" (it is marginally stable).

Summary. Poles set the response: $|p| = \omega_n$, angle $\cos^{-1}\zeta$; $M_p = e^{-\zeta\pi/\sqrt{1-\zeta^2}}$, $t_s \approx 4/\zeta\omega_n$; left half-plane = stable; phase margin measures distance from instability.

Feedback and PID control

Why it matters. The valve must reach its angle within half a degree even when the water pushes on it. Feedback is what makes a cheap motor accurate – if it is designed with the actuator's limits in mind.

The idea, with a picture. Feedback compares the measured output with the set point and acts on the **error** e. A **PID controller** PID outputs $u = K_p e + K_i \int e\,dt + K_d \dot{e}$: proportional for speed, integral to remove steady error, derivative to add damping. With loop gain $L = CG$, a disturbance at the plant input is reduced by the factor $1/(1 + L)$.



The valve position loop: PI controller, voltage limit, plant, load torque and encoder feedback

The derivation (why the integral is needed). The load torque acts like a voltage $d = 3.0$ V subtracted at the motor. In steady state the valve is still, so the motor voltage must equal d. With P control that voltage comes from $K_p e$, so a steady error $e = d/K_p$ must remain. An integral term can supply d with $e = 0$, because $\int e\,dt$ keeps growing until the error is gone.

Worked example 1 – tuning against a spec. Spec: steady error $\leq 0.5°$ under a 5 N · m load. P alone: $e = 3.0/134 = 0.0225$ rad = **1.29**° – fails. Add integral action with its zero a decade below ω_n: $K_i = K_p\omega_n/10 = \mathbf{993}$ V/(rad · s). Steady error now 0; the phase margin falls by about 9°, to 56°. Derivative is not needed: the 0.7 damping already comes from K_p and the motor's own lag, and a derivative would amplify encoder noise.

Worked example 2 – saturation and windup. A 90° step asks $K_p \times 1.571 = \mathbf{210}$ V of a 12 V driver. The motor saturates and the valve moves at its **rate limit** , $0.398 \times (12 - 3.0) = 3.58$ rad/s under load, for about 0.4 s. Meanwhile the integrator keeps adding error it cannot act on – **integrator windup** – and when the valve arrives it carries on to **159**° before the stored integral is unwound. **Anti-windup** by clamping (stop integrating while the output is saturated and the error would push it further) removes the overshoot completely.

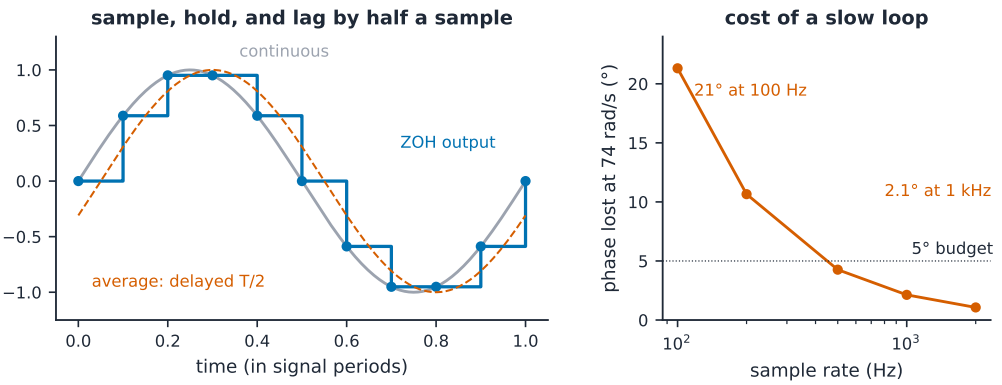
Common mistakes. Tuning with the linear model and testing only small steps. Adding derivative action on a noisy measurement. Using integral action without any windup protection.

Summary. $u = K_p e + K_i \int e + K_d \dot{e}$; P leaves error d/K_p under a constant disturbance, I removes it; saturation makes a linear design nonlinear; clamp the integrator.

Digital implementation

Why it matters. The controller runs on a microcontroller, not in continuous time. Sampling, a finite word length and computation time each cost phase margin – affordable only if they are counted.

The idea, with a picture. An **ADC** samples the signal every T seconds; the controller computes; a **zero-order hold** keeps the output constant until the next sample. That staircase is, on average, the signal delayed by $T/2$, which costs a phase of $\omega T/2$ at frequency ω. Sampling also folds anything above half the sample rate (the **Nyquist frequency**) back into the band – **aliasing** – so it must be filtered out before the ADC.



Sampling and the zero-order hold, and the phase a slow sample rate costs the loop

The derivation (the Tustin discretisation). Replace each s by $\frac{2}{T}\frac{z-1}{z+1}$, where z^{-1} means "one sample ago". For the integral term K_i/s this gives the trapezium rule: $I_k = I_{k-1} + \frac{T}{2}(e_k + e_{k-1})$. The PI controller becomes $u_k = K_p e_k + K_i I_k$ – a few multiplications per sample. The method is valid while the sample rate is well above the loop bandwidth (a common rule: 20 to 30 times).

> **Worked example 1 – choosing the sample rate.** At the loop's natural frequency 74.4 rad/s, the hold costs $74.4T/2$: **2.1**° at 1 kHz but **21**° at 100 Hz. At crossover the phase margin falls from 65.2° to 63.8° at 1 kHz and to 51.3° at 100 Hz. Choose 1 kHz: $74.4/(2\pi) = 11.8$ Hz of bandwidth sampled about 85 times per cycle. Simulated at 1 kHz, the 90° move settles exactly as in continuous time.

> **Worked example 2 – quantisation and latency.** The motor encoder has 500 lines, 2000 counts per revolution after quadrature; through the gearbox that is 100 000 counts per valve turn, **0.0036**° per count – far below the 0.5° spec. But speed estimated by counting at 1 kHz has a resolution of one count per millisecond, $2\pi/2000/0.001 =$ **3.14** rad/s at the motor – coarse, one more reason not to use derivative action. A 0.2 ms computation delay adds $48.2 \times 0.0002 = 0.6°$ of lag at crossover.

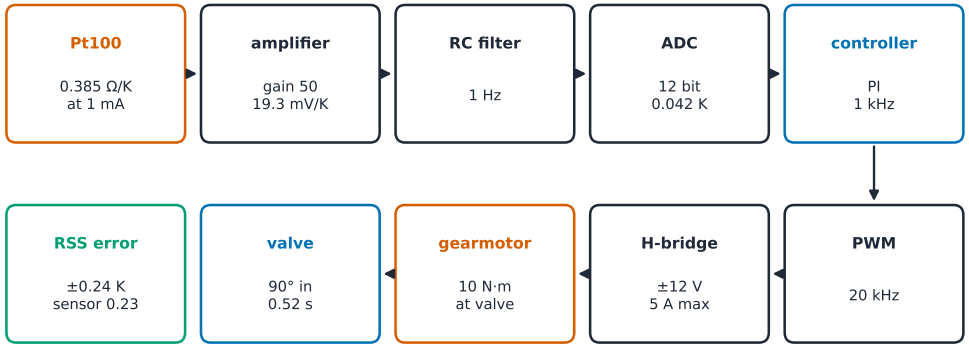
Common mistakes. Sampling just above twice the bandwidth. Forgetting the anti-alias filter. Ignoring computation delay, or differentiating a quantised position.

Summary. ZOH delays by $T/2$; lag $\omega T/2$; sample 20–30× the bandwidth; Tustin $s \to \frac{2}{T}\frac{z-1}{z+1}$; anti-alias before the ADC; count quantisation and latency in the margin.

Sensors and actuators

Why it matters. The valve loop keeps the water at a set temperature only as well as the thermometer reads it. A sensor chain with an unknown error makes the whole control design a guess.

The idea, with a picture. Choose the sensing principle for the **measurand** : a **platinum resistance thermometer** (Pt100) for 0–100 °C water, an encoder for angle. Then follow the **signal chain** – excitation, amplifier, filter, ADC – and write an **error budget** : each independent error, combined by root sum of squares. Actuators have their own limits: voltage, current, speed, power and **bandwidth** .



The loop-temperature measurement chain in, and the valve drive chain out

The derivation (scaling the chain). A Pt100 changes by 0.385 Ω/K. Driven at 1 mA it gives 0.385 mV/K; a gain of 50 makes 19.25 mV/K. A 12-bit ADC over 3.3 V has a step of $3.3/4096 = 0.806$ mV, so one step is $0.806/19.25 =$ **0.042** K, and its rounding error has a standard deviation of step$/\sqrt{12} = 0.012$ K. The 0–100 °C range uses 1.93 V of the 3.3 V input, leaving room for offset and drift.

> **Worked example 1 – the error budget at 40 °C.** Sensor (class A, $\pm(0.15 + 0.002T)$): $\pm$**0.23** K. ADC quantisation: 0.012 K. Amplifier gain error 0.1% of 40 K: 0.04 K. Self-heating (I^2R = 0.1 mW at 2 mW/K): 0.05 K. RSS: $\sqrt{0.23^2 + 0.012^2 + 0.04^2 + 0.05^2} = \pm$**0.24** K. The sensor dominates; a 16-bit ADC would change nothing. A 1 Hz RC filter ahead of a 10 Hz temperature sampler cuts 5 Hz noise to 20%.

> **Worked example 2 – the actuator's limits.** Voltage 12 V → valve speed ≤ 4.78 rad/s (90° in ≥ 0.33 s). Current limit 5 A → 10 N · m at the valve, twice the 5 N · m load. Power at full speed and load: 0.125 N · m × 239 rad/s ≈ 30 W at the motor shaft. Bandwidth: the motor's own lag (9.6 ms) is far faster than the 2.3-hour tank, so for the temperature loop the valve is almost instantaneous – while for the position loop it is the whole plant.

Common mistakes. Quoting ADC resolution as the accuracy. Adding errors that are independent instead of using RSS (or using RSS for errors that are not). Forgetting self-heating and lead resistance in resistance thermometers.

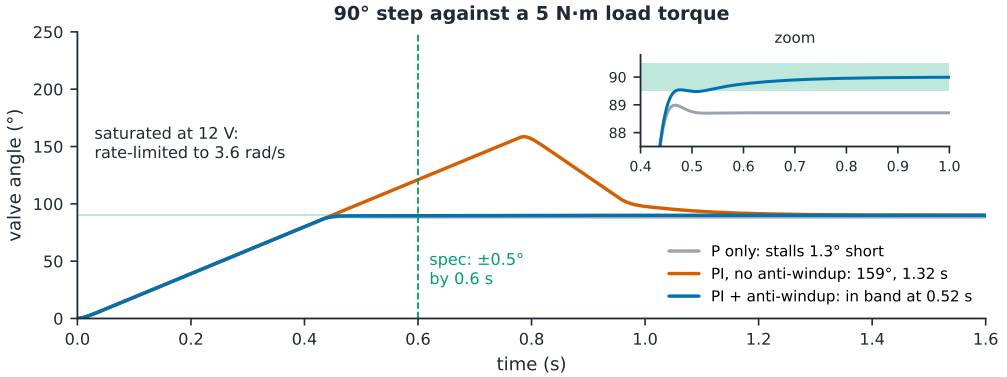
Summary. Pick the principle for the measurand; scale every stage; RSS the inde-

pendent errors; the biggest term decides; actuator limits: volts → speed, amps → torque.

Mechatronics project

Why it matters. Every unit so far meets in one small product: a valve positioner that must move 90° quickly, stop exactly and hold against the water. The project is to show, with traceable numbers, that it meets its spec – and where it does not.

The idea, with a picture. Write a spec with measurable metrics, simulate the full nonlinear model (saturation, load, sampling), compare candidate controllers against every metric, and check the simulation against analytical checkpoints before trusting it.



Simulated 90° valve moves under load: P, PI with windup, and PI with anti-windup

The derivation (checkpoints for the simulation). Three results are known without simulating: during saturation the valve moves at $12 \times 0.398 - 3.0 \times 0.398 = 3.58$ rad/s plus the start-up lag (it enters the last 10% at about 0.40 s); P control must stop $3.0/134 = 1.29°$ short; and a small step without load must show the 4.6% overshoot of $\zeta = 0.7$. The simulation reproduces all three – then its other numbers are worth reporting.

> **Worked example 1 – the spec table.** Spec: 90° within ±0.5° by 0.6 s, overshoot below 5%, under a 5 N · m load. P only: never inside the band (stops 1.29° short) – **fails**. PI without anti-windup: overshoots to **159°**, inside the band only at 1.32 s – **fails**. PI with clamping anti-windup: inside the band at **0.52 s**, no overshoot, zero steady error – **passes**, sampled at 1 kHz with the same result.

> **Worked example 2 – reporting limits honestly.** The passing design has weaknesses the report must state. Without load torque, a small 5° step overshoots by **18%**: the integral zero adds overshoot that the load normally hides – set-point weighting or a smaller K_i would fix it at the cost of slower load recovery ($K_i/2$ leaves 0.85° of error at 0.5 s). The 0.52 s result has only 0.08 s of margin against the spec, and it depends on the 12 V supply: at 11 V the valve reaches the band only at 0.58 s. Untested: encoder faults, valve sticking, and temperature effects on R (copper +0.4%/K).

Common mistakes. Reporting only the passing case. Trusting a simulation never checked against a known result. Writing "meets spec" without the margin.

Summary. Measurable spec → nonlinear simulation → analytical checkpoints → a table of every candidate against every metric → the limits, stated.