Laboratory Notebook

Servo Controller Tuning Project

R. Adeyemi
Mechatronics Lab, Bench 4

Notebook opened: 3 March

Monday, 3 March

Entry 1

Set up the test rig: a 12V DC gear motor with a quadrature encoder (500 counts/rev) driving a small flywheel, controlled through a PID loop running on the bench microcontroller at 1 kHz. Goal for this notebook: tune the PID gains for a step-response settling time under 200 ms with no more than 10% overshoot.

Baseline run with $K_p = 1.0$, $K_i = 0$, $K_d = 0$: motor reaches target speed but oscillates continuously, never settles. Expected — proportional-only control on a system with this much rotational inertia always rings.

Table 1: Step response, proportional-only gains

K_p	Overshoot	Settling time	Notes
0.5	61%	did not settle	sustained oscillation
1.0	74%	did not settle	sustained oscillation
2.0	88%	did not settle	oscillation grows

Higher K_p makes it worse, not better — as expected, need derivative action to damp this.

Tuesday, 4 March

Entry 2

Added derivative term. $K_p = 1.0$, $K_d = 0.05$: oscillation damps out over about 6 cycles, settles at 340 ms with 22% overshoot. Better, but overshoot is still too high and settling time is over the 200 ms target.

Tried increasing K_d further to see if that alone gets us there:

Table 2: Step response, $K_p = 1.0$ fixed, varying K_d

K_d	Overshoot	Settling time
0.05	22%	340 ms
0.10	9%	265 ms
0.15	4%	310 ms

$K_d = 0.10$ hits the overshoot target ($9\% < 10\%$) but settling time is still 65 ms over budget. Pushing K_d further reduces overshoot but slows the response down instead — classic overdamping. Need integral action to remove the small steady-state droop I'm seeing at $K_d = 0.10$ before pushing K_p up to speed the response, otherwise a higher K_p will just reopen the overshoot problem K_d just closed.

Wednesday, 5 March

Entry 3

Added a small integral term to kill steady-state error, then raised K_p to speed up the response, re-tuning K_d each time to hold overshoot near the 10% ceiling:

$K_p = 2.0$, $K_i = 0.02$, $K_d = 0.15$ is the best point found: 10% overshoot (right at the ceiling) and 178 ms settling, comfortably inside the 200 ms target. $K_p = 2.5$ pushes settling time down further but blows through the overshoot budget, so not usable.

Chosen gains: $K_p = 2.0$, $K_i = 0.02$, $K_d = 0.15$.

Thursday, 6 March

Entry 4

Table 3: Full PID sweep

K_p	K_i	K_d	Overshoot	Settling time	Steady-state error
1.0	0.02	0.10	8%	258 ms	<0.5%
1.5	0.02	0.12	9%	205 ms	<0.5%
2.0	0.02	0.15	10%	178 ms	<0.5%
2.5	0.02	0.18	15%	165 ms	<0.5%

Repeated the chosen-gain run 10 times to check reproducibility before signing this off. Settling time varied 172–186 ms, overshoot 9–11% across runs — consistent with the single-run result and within acceptable trial-to-trial noise for this rig. One run at 11% overshoot is a hair over the nominal 10% ceiling but well within the noise band observed here, not treated as a new data point requiring re-tuning.

Next session: repeat this sweep at a higher flywheel load (doubled moment of inertia) to check how much the chosen gains need to change for a different mechanical load, since the production application will not always be driving today's flywheel.