ME326 — Control Systems

Computer Exercises — Module 1 Report

Analysis of Feedback Control Systems

Authors

Ambrine Douhane | Victoria Chalain

Group B11

EPFL — Spring 2021

Introduction

This report presents the results of Module 1 of the ME326 Control Systems computer exercises. The objective of this module is to study the behaviour of a rotary flexible joint (Quanser) under proportional feedback control using MATLAB's Control System Toolbox.

The experimental setup consists of two coupled units: the Rotary Servo Base Unit, which drives the system via a DC motor and gearbox, and the Rotary Flexible Joint, which is a free arm attached to the base by two springs. The output of the system is defined as $y(t) = \theta(t) + \alpha(t)$, the total angle between the arm and the servo base.

The five exercises covered in this module are:

- Exercise 1.2 — Time- and frequency-domain analysis of the plant
- Exercise 1.3 — Closed-loop transfer functions with proportional control ($K = 1.2$)
- Exercise 1.4 — Computation of the ultimate gain K_u
- Exercise 1.5 — Closed-loop step response analysis at $K = 0.6 \cdot K_u$

Module 1.2 — Plant Model

System Parameters

The following numerical values were used in all exercises:

Parameter	Symbol	Value	Unit
Amplifier gain	Ka	2	—
Motor resistance	Rm	2.2	Ω
Torque constant	Km	0.00787	Nm/A
Gearbox ratio	KG	50	—
Motor inertia	Jmot	3.87×10^{-7}	kg·m²

Rotational stiffness	Ks	1.3	Nm/rad
Viscous damping	b	0.004	Nm/(rad/s)
Module inertia	Jmod	3.944×10^{-4}	kg·m²
Arm inertia	JBr	0.0037	kg·m²

The total equivalent inertia at the load side is computed as: $Jm = Jmod + KG^2 \cdot Jmot = 3.944\times10^{-4} + 50^2 \times 3.87\times10^{-7} = 1.3615\times10^{-3}\ kg\cdot m^2$.

Transfer Function G(s)

The plant transfer function G(s) = Y(s)/U(s) was derived from the equations of motion of the DC motor and the flexible joint module. The result is:

```
G(s) = Ks·KG·Km·Ka  /  [(JBr·s² + Ks)(Rm·Jm·s² + Rm·b·s + Km²·KG²·s)
+ Rm·Ks·JBr·s²]
```

This is implemented in MATLAB as:

```
Ka=2; Rm=2.2; Km=0.00787; KG=50; Jmot=3.87e-7;
Ks=1.3; b=0.004; JBr=0.0037; Jmod=3.944e-4;
Jm = Jmod + KG^2*Jmot;
s  = tf('s');
G  = (Ks*KG*Km*Ka) / ...
     ((JBr*s^2+Ks)*(Rm*Jm*s^2+Rm*b*s+Km^2*KG^2*s) + Rm*Ks*JBr*s^2);
G  = minreal(G);
```

Time- and Frequency-Domain Analysis

Using the MATLAB commands step(G), impulse(G), bode(G), pole(G), zero(G), damp(G) and zpk(G), the following qualitative observations were made:

- The plant has 4 poles and 0 finite zeros after minreal simplification.
- Two poles are very close to the imaginary axis (weakly damped), corresponding to the lightly damped flexible mode of the joint spring-arm system.
- The step response shows a sustained oscillation — consistent with a near-marginally stable flexible mode — before reaching a non-zero steady state.
- The Bode diagram reveals a resonance peak near $\omega \approx 18.7$ rad/s, corresponding to the natural frequency of the flexible joint.
- The DC gain (from zpk or dcgain) is finite and positive, confirming the system is type 0.

Module 1.3 — Analysis of the Feedback System

Closed-Loop Structure

A proportional controller Dc(s) = K is placed in a negative unity feedback loop around the plant G(s). With K = 1.2, the four closed-loop transfer functions are defined as follows:

Transfer Function	Expression	Description
S(s) = r → e	1 / (1 + KG)	Sensitivity function
U(s) = r → u	K / (1 + KG)	Control effort
T(s) = r → y	KG / (1 + KG)	Complementary sensitivity (tracking)
V(s) = w → y	G / (1 + KG)	Disturbance rejection

MATLAB Implementation

```
K = 1.2;
G = plant_tf();
s = tf('s');

S = feedback(1, K*G);          % r -> e  (sensitivity)
U = feedback(K, G);             % r -> u  (control effort)
T = feedback(K*G, 1);           % r -> y  (complementary sensitivity)
V = feedback(G, K);             % w -> y  (disturbance -> output)
```

Step Responses (K = 1.2)

The step responses of the four closed-loop transfer functions were plotted using step([S; U; T; V]):

- r → y (T): The output tracks the reference but with significant overshoot and oscillation, reflecting the weakly damped flexible mode. Steady-state error is present due to the type-0 open loop.
- r → u (U): The control signal shows a sharp initial peak before settling — acceptable for the proportional gain used.
- r → e (S): The tracking error decays but does not reach zero at steady state, confirming non-zero steady-state error for a step reference.
- w → y (V): The disturbance-to-output response shows the system partially attenuates input disturbances, but rejection is incomplete without integral action.

Stability Analysis (K = 1.2)

The closed-loop poles were computed using pole(T). All poles were found to have strictly negative real parts, confirming that the closed-loop system is stable for K = 1.2.

Module 1.4 — Computing the Ultimate Gain

Definition

The ultimate gain Ku is the minimum proportional gain K that renders the closed-loop system marginally stable, i.e., that places at least one closed-loop pole on the imaginary axis.

Method 1 — Root Locus

The root locus rlocus(G) was plotted to visualise the trajectories of the closed-loop poles as K varies from 0 to ∞. The value of K at which a locus branch crosses the imaginary axis gives Ku. From inspection of the root locus plot, the ultimate gain was identified as:

```
Ku = 3.63
```

Method 2 — Routh–Hurwitz Criterion (analytical verification)

The closed-loop characteristic polynomial was derived from 1 + K·G(s) = 0. Expanding the denominator of G(s) yields a degree-4 polynomial:

$$1 + K\cdot G(s) = \frac{[s^4 + 54.62s^3 + 1306s^2 + 1.919\times10^4\cdot s + K\cdot 9.229\times10^4]}{s^4 + 54.62s^3 + 1306s^2 + 1.919\times10^4\cdot s}$$

Constructing the Routh array with coefficients:

- $a_1 = 54.62$, $a_2 = 1306$, $a_3 = 1.919\times10^4$, $a_4 = K\cdot 9.229\times10^4$

The Routh array yields the following first-column entries (relevant rows shown):

$$b_1 = 954.7$$
$$b_2 = 9.229\times10^4 \cdot K$$
$$c_1 = -(5.05\times10^6\cdot K - 1.832\times10^9) / 954.7$$
$$c_2 = 0$$
$$d_1 = b_2 = 9.229\times10^4\cdot K$$

Stability conditions require all first-column entries to be positive:

- Condition (n): $c_1 > 0 \rightarrow K < 1.832\times10^9 / 5.05\times10^6 = 3.6277$
- Condition (2): $d_1 > 0 \rightarrow K > 0$

Therefore: **Ku = $1.832\times10^9 / 5.05\times10^6 \approx 3.63$**, confirming the root locus result.

Module 1.5 — Closed-Loop Step Response Analysis

Controller Gain Selection

The controller gain is set to **K = 0.6 · Ku = 0.6 × 3.63 = 2.178**. This is a standard conservative starting point, placing the system well within the stability margin while still achieving reasonable tracking performance.

MATLAB Implementation

```
Ku = ultimate_gain();
K  = 0.6 * Ku;

[~, ~, T, ~] = closed_loop_tfs(K);
```

```
INFO           = stepinfo(T);
rise_time      = INFO.RiseTime;
settling_time  = INFO.SettlingTime;
overshoot      = INFO.Overshoot;
closedloop_bw  = bandwidth(T);
```

Results

The following metrics were obtained from the step response of T(s) at K = 0.6·Ku:

Metric	Value
Rise Time	≈ 0.47 s
Settling Time	≈ 4.2 s
Overshoot	≈ 38 %
Closed-loop Bandwidth	≈ 3.1 rad/s

Discussion

The step response at K = 0.6·Ku exhibits the following characteristics:

- Rise time of ≈ 0.47 s indicates reasonably fast initial response.
- Settling time of ≈ 4.2 s reflects the slow decay of the lightly damped flexible mode oscillation.
- Overshoot of ≈ 38% is relatively large, a consequence of the weakly damped poles of the flexible joint that are brought close to the imaginary axis by the proportional gain.
- Closed-loop bandwidth of ≈ 3.1 rad/s is below the resonant frequency of the flexible mode (≈18.7 rad/s), which is beneficial for robustness.

These results confirm that a pure proportional controller is insufficient for high-performance control of this system: the flexible mode creates persistent oscillations that can only be attenuated with more advanced controllers (PID, cascade, or state-space), as addressed in Modules 2–5.

Conclusion

This module introduced key MATLAB tools for closed-loop analysis and applied them to the Quanser rotary flexible joint. The main findings are:

- The plant transfer function G(s) was derived analytically and implemented in MATLAB. The system has a lightly damped flexible mode at approximately 18.7 rad/s.
- For K = 1.2, the closed-loop system is stable. All four standard closed-loop transfer functions (S, U, T, V) were computed and their step responses analysed.
- The ultimate gain Ku was determined to be 3.63 via both the root locus and the Routh–Hurwitz criterion, which gave consistent results.
- At K = 0.6·Ku ≈ 2.178, the closed-loop system achieves a rise time of ≈0.47 s but suffers from significant overshoot (≈38%) and a long settling time (≈4.2 s) due to the flexible mode dynamics.

These limitations motivate the design of more sophisticated controllers in subsequent modules.

ME326 — Control Systems

Computer Exercises — Module 2 Report

PID Controller Design

Authors
Ambrine Douhane | Victoria Chalain
Group B11

EPFL — Spring 2021

Introduction

This report presents the results of Module 2 of the ME326 Control Systems computer exercises. The objective of this module is to design and compare PID controllers for the Quanser rotary flexible joint using three approaches:

- Module 2.1 — Ziegler–Nichols First Method (based on the open-loop step response)
- Module 2.2 — Ziegler–Nichols Second Method (based on the ultimate gain and period)
- Module 2.3 — Cascade Controller (inner PID loop + outer proportional loop)

All controllers are designed in MATLAB using the plant transfer function G(s) defined in Module 1. The tracking performance (r → y) and disturbance rejection (w → y) are compared across methods.

Module 2.1 — Ziegler–Nichols First Method

Method Overview

The first Ziegler–Nichols method estimates PID parameters from the open-loop step response of the plant, modelled as a first-order system with dead time. A tangent line is fitted at the inflection point of the step response to extract two parameters:

- L — dead time (x-intercept of the tangent line)
- R — slope of the tangent line at the inflection point

Parameter Extraction

From the step response of G(s) (Image 2 — slowly decaying step), a tangent was drawn at the inflection point. Two points on the tangent line were read:

Point	t (s)	y(t)
Point 1	4.557	21.589

Point 2	9.563	45.666

The slope R and dead time L were computed as follows:

```
R = (45.6655 - 21.5888) / (9.56335 - 4.55705);   % = 4.8093
% Tangent line: y(t) = 4.8093*t - 0.3273
% x-intercept (y=0): t = 0.3273 / 4.8093
L = 0.0681;                                          % dead time [s]
```

PID Controller Design

Using the ZN-1 tuning table for a PID controller:

Parameter	Formula	Value
Kp	1.2 / (R·L)	≈ 3.647
Ti	2·L	≈ 0.1362 s
Td	0.5·L	≈ 0.0341 s

```
Ti = 2*L;        % = 0.1362
Td = 0.5*L;      % = 0.0341
Kp = 1.2/(R*L); % = 3.647
K  = Kp * (1 + 1/(Ti*s) + Td*s);
```

Results

The closed-loop step response (r → y) with the ZN-1 PID controller shows:

- Zero steady-state error, as expected with integral action.
- Moderate overshoot (typical of ZN tuning, which targets approximately 25% overshoot).
- Fast initial rise due to the relatively high proportional gain Kp ≈ 3.647.
- The flexible mode oscillation is better damped compared to the proportional controller of Module 1, owing to the derivative and integral terms.

Module 2.2 — Ziegler–Nichols Second Method

Ultimate Period Determination

The second ZN method uses the ultimate gain Ku and the ultimate period Pu. From Module 1, Ku = 3.63. The ultimate frequency was read from the root locus (Image 3 — root locus plot):

Parameter	Value	Source
Ku	3.63	Root locus (Module 1)
ωu (ultimate frequency)	18.8 rad/s	Root locus imaginary-axis crossing
Pu = 2π / ωu	≈ 0.3342 s	Computed

From the root locus (Image 3), at K = 3.63 the closed-loop poles cross the imaginary axis at ±18.8j rad/s, confirming ωu = 18.8 rad/s.

```
f  = 18.8;
Pu = 2*pi/f;    % = 0.3342 s
```

PID Controller Design

Using the ZN-2 tuning table for a PID controller:

Parameter	Formula	Value
Kp	0.6·Ku	≈ 2.178
Ti	0.5·Pu	≈ 0.1671 s
Td	0.125·Pu	≈ 0.0418 s

```
Ku = ultimate_gain();    % = 3.63
Pu = 2*pi/18.8;          % = 0.3342 s
Kp = 0.6*Ku;             % = 2.178
Ti = 0.5*Pu;             % = 0.1671 s
Td = 0.125*Pu;           % = 0.0418 s
K  = Kp*(1 + 1/(Ti*s) + Td*s);
```

Results

The closed-loop step and disturbance responses with the ZN-2 PID controller show:

- Zero steady-state error for a step reference (integral action).
- Overshoot of approximately 25–35%, consistent with ZN-2 design targets.
- Slower settling compared to ZN-1 due to the lower Kp (2.178 vs 3.647), but more robust against the flexible mode.
- Good disturbance rejection: the output returns to zero after a step disturbance at the plant input, confirmed by the step(V) plot.

Module 2.3 — Cascade Controller

Architecture

The cascade controller uses two nested feedback loops:

- Inner loop: controls the angular velocity, using $G_1(s) = G(s)\cdot s$ as the plant. A PID controller Dc'(s) is designed for a bandwidth of 50 rad/s.
- Outer loop: controls the angular position using an integrator 1/s and a proportional controller Dc(s) = kP, designed for a bandwidth of 5 rad/s.

Since the inner loop is much faster than the outer loop, its dynamics can be neglected when designing the outer controller.

Second-Order Approximation of $G_1(s)$

$G_1(s) = G(s)\cdot s$ was approximated as a second-order system using the step response (Image 1 — step response of G_1):

$$G_1(s) \approx \gamma\cdot\omega n^2 / (s^2 + 2\zeta\cdot\omega n\cdot s + \omega n^2)$$

The parameters were identified from the step response using MATLAB's stepinfo:

Parameter	Formula	Value
γ (DC gain)	Read from step(G_1) steady-state	4.8055
ζ (damping ratio)	From overshoot: $\zeta = \sqrt{\ln(OS)^2 / (\pi^2 + \ln(OS)^2)}$	Computed from INFO_G1.Overshoot
ωn (natural frequency)	$\pi / (tp \cdot \sqrt{1 - \zeta^2})$	Computed from PeakTime

```
G1      = G * s;
INFO_G1 = stepinfo(G1);
tp      = INFO_G1.PeakTime;
gamma   = 4.8055;
OS      = INFO_G1.Overshoot / 100;
zeta    = sqrt(log(OS)^2 / (pi^2 + log(OS)^2));
omega_n = pi / (tp * sqrt(1 - zeta^2));
```

Inner Loop PID Controller (Dc')

Using the model reference method to achieve a closed-loop bandwidth of 50 rad/s ($\tau m = 1/50$ s):

Gain	Formula	Expression
Kp	$2\zeta / (\gamma\cdot\omega n\cdot\tau m)$	$2\cdot\zeta / (\gamma\cdot\omega n / 50)$
Ki	$1 / (\gamma\cdot\tau m)$	$50 / \gamma$
Kd	$1 / (\gamma\cdot\omega n^2\cdot\tau m)$	$50 / (\gamma\cdot\omega n^2)$

```
tau_m   = 1/50;
Kp      = 2*zeta / (gamma*omega_n*tau_m);
Ki      = 1 / (gamma*tau_m);
Kd      = 1 / (gamma*omega_n^2*tau_m);
Dc_prime = minreal(Kp + Ki/s + Kd*s);
```

Outer Loop Proportional Controller (Dc)

Since the inner loop is much faster, it is treated as unity gain for the outer loop design. The outer proportional gain was chosen to achieve a bandwidth of 5 rad/s:

```
Dc = 2.5;   % proportional gain for outer loop
```

This value was determined iteratively to achieve the target bandwidth of 5 rad/s on the outer closed-loop transfer function.

Closed-Loop Transfer Functions

The cascade closed-loop transfer functions T(s) = Y/R and V(s) = Y/W were computed as:

```
innerloop = zpk(minreal(feedback(G1*Dc_prime, 1)));
T = zpk(minreal(feedback(Dc*innerloop/s, 1)));     % Y/R
V = zpk(minreal(1/(1/G + Dc*Dc_prime + Dc_prime*s)));  % Y/W
```

Results

The cascade controller step responses (T and V) are shown in Images 1 and 2 respectively. Key observations:

- Tracking (r → y): The cascade controller achieves zero steady-state error with significantly less overshoot than the ZN-PID controllers. The response is smoother due to the well-tuned inner loop suppressing the flexible mode.
- Disturbance rejection (w → y): The cascade controller attenuates input disturbances effectively. The disturbance response (V) shown in Image 1 decays rapidly to zero.
- The inner loop bandwidth of 50 rad/s — well above the flexible mode at ~18.8 rad/s — ensures the flexible oscillations are damped before they affect the outer position loop.

Comparison of Controllers

Criterion	ZN-1 PID	ZN-2 PID	Cascade
Steady-state error	0 (integral)	0 (integral)	0 (integral)
Overshoot	~25–40%	~25–35%	Low (~5–15%)
Settling time	Moderate	Moderate–slow	Fast
Disturbance rej.	Good	Good	Excellent
Tuning complexity	Low	Low	High
Flexible mode	Partially damped	Partially damped	Well damped

The cascade controller clearly outperforms both ZN-PID controllers in terms of tracking and disturbance rejection. This is because the inner velocity loop effectively decouples the flexible mode dynamics from the outer position loop, making the system behave like a simpler second-order plant for the outer controller.

The ZN methods provide an adequate starting point but produce significant overshoot and are sensitive to the flexible mode resonance. ZN-2 is slightly more conservative than ZN-1 due to the lower Kp.

Conclusion

This module designed and compared three PID-based controllers for the Quanser rotary flexible joint:

- ZN-1 (L = 0.0681 s, R ≈ 4.809) yielded Kp ≈ 3.647 with fast rise time but significant overshoot.

- ZN-2 ($K_u = 3.63$, $\omega_u = 18.8$ rad/s, $P_u \approx 0.334$ s) yielded $K_p \approx 2.178$, more conservative but slightly slower.
- The cascade controller — with a PID inner loop (bandwidth 50 rad/s) and proportional outer loop ($D_c = 2.5$) — achieved the best overall performance: low overshoot, fast settling, and excellent disturbance rejection.

The cascade architecture is the preferred approach for this system due to its ability to independently address the flexible mode in the inner loop, leading to a well-behaved outer position control loop. This motivates the loop-shaping and state-space methods explored in subsequent modules.

ME326 — Control Systems

Computer Exercises — Module 3 Report

Loop Shaping Method

Authors
Ambrine Douhane | Victoria Chalain
Group B11

EPFL — Spring 2021

Introduction

This report presents the results of Module 3 of the ME326 Control Systems computer exercises. The objective of this module is to design controllers for the Quanser rotary flexible joint using the loop shaping method, which specifies controller behaviour directly in the frequency domain.

Two controllers are designed and analysed:

- Module 3.1 — Proportional controller achieving a crossover frequency of 5 rad/s
- Module 3.2 — Lead-lag compensator with integrator achieving a crossover frequency of 5 rad/s and a phase margin of 55°

The lead-lag controller is then compared against the cascade controller from Module 2 in terms of tracking and disturbance rejection (Module 3.3).

Background — Loop Shaping

The loop shaping method designs a controller Dc(s) such that the open-loop transfer function L(s) = Dc(s)·G(s) has a desired shape in the Bode diagram. The key design specifications are expressed as frequency-domain margins:

- Gain margin (GM): factor by which the gain can increase before instability. Expressed in dB: GM = $20 \cdot \log_{10}$(Gm).
- Phase margin (PM): additional phase lag (in degrees) at the crossover frequency before instability.
- Modulus margin (MM): inverse of the peak of the sensitivity function S(s) = 1/(1 + L(s)). Defined as MM = $1/\|S\|_\infty$. A larger MM indicates better robustness.

The crossover frequency ωc is the frequency at which |L(jωc)| = 1 (0 dB). Specifying ωc = 5 rad/s sets the closed-loop bandwidth.

Module 3.1 — Proportional Controller

Design Principle

A proportional controller K is designed to set the gain crossover frequency to ωc = 5 rad/s. The gain of the plant at ωc is measured from the Bode diagram, and K is set to the reciprocal so that |K·G(j·5)| = 1:

```
[mag, phase, ~] = bode(G, 5);   % evaluate G at omega = 5 rad/s
K = 1 / mag;                     % set gain so |K*G(j5)| = 1
```

Stability Margins

Once K is computed, the margins of the loop L(s) = K·G(s) are evaluated using MATLAB's margin command, and the modulus margin is derived from the peak of the sensitivity function:

```
[Gm, Pm, ~, ~] = margin(K*G);
gm = 20*log10(Gm);
pm = Pm;

S     = 1 / (1 + K*G);
% sup_s = peak of |S(jw)| observed from bode(S) = 3.29 dB = 1.4605
sup_s = 1.460494;
mm    = 1 / sup_s;
```

Results

Margin	Value	Interpretation
Gain Margin (GM)	≈ 9.3 dB	Gain can increase ~2.9× before instability
Phase Margin (PM)	≈ 15°	Limited — flexible mode reduces phase
Modulus Margin (MM)	≈ 0.685	1/1.4605 — moderate robustness
Peak sensitivity ‖S‖∞	3.29 dB = 1.461	Read from Bode of S(s)

The phase margin of ~15° is low, indicating limited robustness against perturbations near the crossover frequency. This is a direct consequence of the phase loss caused by the flexible mode resonance near 18.8 rad/s. The modulus margin MM ≈ 0.685 further confirms modest robustness.

A proportional controller alone cannot simultaneously achieve a reasonable crossover frequency and adequate phase margin for this system — motivating the lead-lag design in Module 3.2.

Module 3.2 — Lead-Lag Controller with Integrator

Design Specifications

The lead-lag controller must satisfy three simultaneous requirements:

- Reject step input disturbances → requires an integrator (1/s) in the controller
- Crossover frequency ωc = 5 rad/s

- Phase margin PM = 55°

Design Procedure

The controller takes the form:

```
Dc(s) = K_lag * (1 + alpha*tau*s) / (s * (1 + tau*s))
```

The integrator (1/s) ensures zero steady-state error for step disturbances. The lead-lag compensator $(1 + \alpha \cdot \tau \cdot s)/(1 + \tau \cdot s)$ provides the required phase boost at ωc.

The design steps are:

- Step 1: Evaluate G(s)/s at ωc = 5 rad/s to get the augmented plant magnitude and phase.
- Step 2: Compute the required phase boost $p = \tan(PM - (\angle G(j \cdot 5)/j - 180°))$.
- Step 3: Compute the gain condition $c = 1/|G(j \cdot 5)/j|^2$.
- Step 4: Solve the polynomial for α (the lead-lag zero/pole ratio).
- Step 5: Compute τ from the crossover frequency condition.

MATLAB Implementation

```
cf = 5;   pm_target = 55;

[mag, phase, ~] = bode(G/s, cf);   % augmented plant at omega_c

p = tan(deg2rad(pm_target - (phase - 180)));
c = 1 / (mag * mag);

% Solve quadratic for alpha
alpha = max(roots([(p^2-c+1), 2*p^2*c, p^2*c^2+c^2-c]));

% Compute tau from crossover condition
tau = sqrt((1-c) / (c - alpha^2)) / cf;

% Lead-lag controller with integrator
Dc = K_lag * (1 + alpha*tau*s) / (s * (1 + tau*s));
```

Margin Verification

The margins of the resulting open-loop $L(s) = Dc(s) \cdot G(s)$ were verified using margin(L):

Margin	Target	Obtained	Verified?
Crossover frequency	5 rad/s	≈ 5 rad/s	✓
Phase margin (PM)	55°	≈ 55°	✓
Modulus margin (MM)	—	≈ 0.638	—
Peak sensitivity $\|S\|_\infty$	—	3.9 dB = 1.567	—

The modulus margin MM = 1/1.567 ≈ 0.638. The peak sensitivity of 3.9 dB ($\|S\|_\infty$ = 1.567) is slightly higher than for the proportional controller (3.29 dB), reflecting the trade-off: the integrator slightly reduces robustness but is necessary to reject disturbances. Both values are acceptable.

```
% Modulus margin from sensitivity peak
% Observed from bode(S): max|S| = 3.9 dB = 1.566751
sup_s = 1.566751;
mm    = 1 / sup_s;   % = 0.638
```

Module 3.3 — Comparison: Lead-Lag vs Cascade Controller

Overview

The lead-lag controller (Module 3.2) is compared against the cascade controller (Module 2.3) in terms of:

- Tracking performance: step response of T(s) = Y/R
- Disturbance rejection: step response of V(s) = Y/W

Tracking Performance (r → y)

Both controllers achieve zero steady-state error for a step reference:

- Lead-lag: The integrator in Dc(s) ensures zero steady-state error. The response has moderate overshoot but good rise time due to the 5 rad/s crossover frequency. The phase margin of 55° ensures adequate damping.
- Cascade: Also zero steady-state error. The inner PID loop at 50 rad/s bandwidth suppresses the flexible mode, resulting in lower overshoot and faster settling than the lead-lag controller.

Disturbance Rejection (w → y)

Both controllers reject step disturbances at the plant input:

- Lead-lag: The integrator in Dc(s) guarantees zero steady-state output for a constant step disturbance. The transient response is governed by the open-loop crossover frequency of 5 rad/s.
- Cascade: The inner loop provides faster disturbance rejection since the inner PID acts directly on the velocity, attenuating disturbances before they propagate to the position output.

Summary Comparison

Criterion	Lead-Lag (3.2)	Cascade (2.3)
Steady-state error	0 (integrator)	0 (integrator)
Phase margin	55° (by design)	Not directly specified
Crossover frequency	5 rad/s (by design)	~5 rad/s (outer loop)

Overshoot	Moderate (~20–30%)	Low (~5–15%)
Settling time	Moderate	Faster
Disturbance rejection	Good	Excellent (inner loop)
Flexible mode	Partially attenuated	Well damped (inner loop)
Controller complexity	Low (1 TF)	Higher (2 nested loops)
Design transparency	High (frequency domain)	Moderate (2-step)

The cascade controller generally outperforms the lead-lag controller in transient performance because the inner loop explicitly addresses the flexible mode dynamics at 18.8 rad/s. However, the lead-lag controller has the advantage of being a single transfer function and being designed with clear, verifiable frequency-domain specifications (PM = 55°, ωc = 5 rad/s), making it more transparent and easier to certify in practice.

The loop shaping approach also reveals a fundamental limitation of this plant: the flexible mode at ~18.8 rad/s introduces significant phase loss near the crossover frequency, constraining the achievable phase margin for a given bandwidth. Controllers that do not explicitly account for this mode (like the proportional controller in 3.1) achieve only ~15° phase margin at ωc = 5 rad/s.

Conclusion

This module applied the loop shaping method to the Quanser rotary flexible joint. The main findings are:

- Proportional controller (3.1): A gain of K = 1/|G(j5)| achieves ωc = 5 rad/s but yields a poor phase margin (~15°) and modulus margin MM ≈ 0.685, due to the phase lag of the flexible mode.
- Lead-lag controller (3.2): By adding an integrator and a lead-lag compensator, the required specifications (ωc = 5 rad/s, PM = 55°) are met. The modulus margin MM ≈ 0.638 and peak sensitivity of 3.9 dB are acceptable. Step disturbances are rejected asymptotically.
- Comparison (3.3): The cascade controller from Module 2 achieves better transient performance (lower overshoot, faster settling) due to its inner velocity loop, but the lead-lag controller offers greater design transparency and simpler structure with guaranteed frequency-domain margins.

These results motivate the state-space and digital control approaches in Modules 4 and 5, where pole placement provides direct control over transient dynamics.

ME326 — Control Systems

Computer Exercises — Module 4 Report

State-Space Method

Authors
Ambrine Douhane | Victoria Chalain
Group B11

EPFL — Spring 2021

Introduction

This report presents the results of Module 4 of the ME326 Control Systems computer exercises. The objective of this module is to model and control the Quanser rotary flexible joint using the state-space approach, which allows direct placement of the closed-loop poles and explicit handling of unmeasured states through a state estimator.

The module is divided into three parts:

- Module 4.1 — Derivation of the state-space model (A, B, C, D) and verification of controllability and observability
- Module 4.2 — LQR state-feedback controller with state estimator (Luenberger observer) and feedforward gain for zero steady-state error
- Module 4.3 — Augmented state-space controller with integrator for zero steady-state error under step disturbances, using pole placement

All controllers use the state vector $x = [\dot\theta, \theta, \dot\alpha, \alpha]^T$, where θ is the chassis angle and α is the arm-to-chassis angle of the flexible joint.

Module 4.1 — State-Space Model

Derivation of the (A, B, C, D) Matrices

Starting from the equations of motion of the DC motor and the flexible joint (derived in the modelling section of the course), the state equations are written in the form $\dot x = Ax + Bu$, $y = Cx + Du$, with state vector $x = [\dot\theta, \theta, \dot\alpha, \alpha]^T$ and scalar input u (motor voltage) and scalar output $y = \theta + \alpha$.

The intermediate coefficients are:

```
Jm  = Jmod + KG^2 * Jmot

A11 = -(Km^2*KG^2 + b*Rm) / (Rm * Jm)
A14 =  Ks / Jm
```

```
A34 = -(Ks/JBr) - Ks/Jm
B11 =  (Km*KG*Ka) / (Rm * Jm)
```

The full state-space matrices are:

```
A = [A11,  0,  0, A14;
       1,  0,  0,   0;
    -A11,  0,  0, A34;
       0,  0,  1,   0];

B = [B11;  0; -B11; 0];
C = [0, 1, 0, 1];
D = [0];
```

The output y = θ + α is captured by C = [0, 1, 0, 1], consistent with the transfer function output defined in Module 1. The model is validated by comparing the step response and Bode diagram of the state-space model against the previously derived transfer function G(s).

Controllability and Observability

The controllability matrix Cc and observability matrix Oo are computed as:

```
Cc = [B,  A*B,  A^2*B,  A^3*B];     % 4×4 controllability matrix
Oo = [C; C*A; C*A^2; C*A^3];         % 4×4 observability matrix
```

Both matrices were found to have non-zero determinants, confirming:

- Controllability: all states can be driven to any desired value by the input u — the LQR design is therefore feasible.
- Observability: all states can be reconstructed from the output y = θ + α — the Luenberger observer design is feasible.

Property	Test	Result
Controllable?	det(Cc) ≠ 0	Yes — rank(Cc) = 4
Observable?	det(Oo) ≠ 0	Yes — rank(Oo) = 4

Module 4.2 — LQR Controller with State Estimator

LQR State-Feedback Design

The LQR method finds the optimal state-feedback gain K_ss that minimises the quadratic cost:

```
J = integral( x'*Q*x + u'*R*u ) dt
```

The weighting matrices are chosen as:

- Q = C^TC — penalises the output y = θ + α, giving more weight to states that affect the measured output

- R = 0.01 — penalises the control effort; a smaller R allows larger control signals for faster response

```
Q    = C' * C;
R    = 0.01;
K_ss = lqr(A, B, Q, R);
```

The LQR gain K_ss is a 1×4 row vector. The dominant closed-loop poles of (A − B·K_ss) were found to be approximately −30.02 ± 7.41j rad/s, giving a natural frequency of ≈ 31 rad/s and damping ratio ≈ 0.97 — well damped and fast.

State Estimator (Luenberger Observer)

Since not all states are directly measurable (only $y = \theta + \alpha$ is measured), a Luenberger observer reconstructs the full state vector $\hat{x}$ from y and u:

```
x^˙ = A*x^ + B*u + L*(y - C*x^)
```

The observer gain L is designed by pole placement. The observer poles must be faster than the dominant control poles (≈ −30 rad/s) by 10 rad/s, placing them at:

```
obs_pole = -abs(-30.0214 + 7.4142i) - 10;   % ≈ -41 rad/s
L = acker(A', C', ones(4,1)*obs_pole)';     % all 4 observer poles at
obs_pole
```

Placing all observer poles at the same location ensures fast state estimation convergence without requiring excessive computation for pole selection.

Feedforward Gain N̄

To achieve zero steady-state error for a step reference input, a feedforward gain N̄ is computed. It scales the reference signal r so that the DC gain of the closed-loop system from r to y equals unity:

```
A_cl = [A,   -B*K_ss;
        L*C,  A-B*K_ss-L*C];
B_cl = [B; B];
C_cl = [C, zeros(1,4)];
N_bar = 1 / ((C_cl / (-A_cl)) * B_cl);
```

Closed-Loop Transfer Functions

The full closed-loop state-space system (plant + observer) has 8 states (4 plant + 4 observer). The closed-loop matrices are:

```
A_cl = [A,        -B*K_ss;
        L*C,  A-B*K_ss-L*C];

B_cl = [N_bar*B;  N_bar*B];
C_cl = [C, zeros(1,4)];          % output y
```

```
T = ss(A_cl, B_cl, C_cl, 0);                   % r -> y
U = ss(A_cl, B_cl, [0,0,0,0,-K_ss], N_bar);   % r -> u
```

Results

The step responses of T(s) and U(s) show:

- Tracking (r → y): The output converges to unity with low overshoot and fast settling (~0.3 s), owing to the well-damped dominant poles at ≈ −30 ± 7.4j rad/s and the feedforward gain N̄.
- Control effort (r → u): The control signal shows an initial peak then settles rapidly. The peak magnitude remains bounded due to R = 0.01 penalising large inputs in the LQR cost.
- No steady-state error for a step reference, confirmed by the DC gain of T equalling 1.

Note: without an integrator, this controller does not reject step disturbances at the plant input — this is addressed in Module 4.3.

Module 4.3 — Augmented State-Space Controller with Integrator

Motivation

The LQR controller of Module 4.2 achieves zero steady-state error for step references via feedforward gain N̄, but cannot reject step disturbances applied to the plant input. To obtain zero steady-state error for both references and disturbances, the state-space model is augmented with an integrator of the tracking error.

Augmented State-Space Model

A new artificial state xI is introduced: the integral of the tracking error e = r − y. This augments the plant state vector from 4 to 5 states:

```
ẋI = r - y = r - C*x

=> A_bar = [A,          zeros(4,1);
            -C,              0    ];     % 5×5

   B_bar = [B;  0];                      % 5×1

A_bar = [A,  zeros(n,1);  -C,  0];
B_bar = [B;  0];
```

Pole Placement for Augmented Controller

The augmented controller gain [K0, K1] (where K0 is the 1×4 plant state gain and K1 is the 1×1 integrator gain) is computed by pole placement on the augmented system (A_bar, B_bar):

The desired closed-loop polynomial corresponds to ωn = 10 rad/s, ζ = 0.8:

```
s^2 + 2*0.8*10*s + 10^2  =  s^2 + 16s + 100
=> dominant poles: s = -8 ± 6j
```

Three additional fast poles are placed at −30 rad/s to ensure rapid convergence of the remaining modes:

```
desired_poles = [-8+6i, -8-6i, -30, -30, -30];
intermediaire = acker(A_bar, B_bar, desired_poles);

K0 = intermediaire(:, 1:4);   % state feedback gain
K1 = intermediaire(:, 5);     % integrator gain
```

State Estimator

The observer for the augmented system uses only the 4 original plant states (the integrator state xI does not need to be estimated). All 4 observer poles are placed at −30 rad/s:

```
L = acker(A', C', [-30, -30, -30, -30])';
```

Closed-Loop State Equations

The full closed-loop system (9 states: 4 plant + 4 observer + 1 integrator) is assembled as:

```
Acl = [A,     -B*K0,        -B*K1;
       L*C,   A-B*K0-L*C, -B*K1;
       -C,    zeros(1,4),      0];

Bcl = [0;0;0;0; 0;0;0;0; 1];    % 9×1, reference enters integrator
only

Css_y = [C, 0,0,0,0, 0];        % output y
Css_u = [0,0,0,0, -K0, -K1];    % control signal u

T = ss(Acl, Bcl, Css_y, D);
U = ss(Acl, Bcl, Css_u, D);
```

Results

The step responses for the augmented controller show:

- Tracking (r → y): The dominant poles at −8 ± 6j (ω_n = 10 rad/s, ζ = 0.8) give a well-damped, moderately fast response. Overshoot is limited by ζ = 0.8 to approximately 1.5%.
- Control effort (r → u): The control signal remains bounded. The integrator state initially accumulates to eliminate steady-state error, then settles as the error goes to zero.
- Zero steady-state error for both step references and step disturbances, due to the integrator in the augmented model.
- The fast poles at −30 rad/s ensure the estimator converges before the dominant transient, satisfying the separation principle.

Comparison: SS Controller vs SS with Integrator

Criterion	LQR (4.2)	Pole Placement + Integrator (4.3)
Steady-state error (ref)	0 (via $\bar{N}$)	0 (via integrator)
Steady-state error (dist)	Non-zero	0 (integrator rejects it)
Dominant poles	≈ −30 ± 7.4j rad/s	−8 ± 6j rad/s (ω_n=10, ζ=0.8)
Controller states	4 (plant) + 4 (observer)	4 + 4 + 1 (integrator)
Design method	LQR (optimal)	Pole placement (explicit)
Overshoot	Low (~0%)	Very low (~1.5%, ζ=0.8)
Speed	Fast (~0.3 s)	Moderate (~0.5 s)
Disturbance rejection	Incomplete	Asymptotic (zero SS error)

The LQR controller achieves faster transient response but lacks robustness to constant disturbances. The augmented controller sacrifices some speed (slower dominant poles at ω_n = 10 rad/s) in exchange for guaranteed disturbance rejection via the integrator.

Conclusion

This module applied the state-space approach to the Quanser rotary flexible joint. The key findings are:

- State-space model (4.1): The (A, B, C, D) matrices were derived from the equations of motion. Both the controllability matrix Cc and observability matrix Oo have full rank (det ≠ 0), confirming that LQR and observer design are feasible.
- LQR controller (4.2): With $Q = C^T C$ and R = 0.01, the LQR gain places dominant poles near −30 ± 7.4j rad/s. A Luenberger observer with all poles at ≈ −41 rad/s estimates the full state. The feedforward gain $\bar{N}$ ensures zero steady-state error for step references.
- Augmented controller (4.3): Augmenting the state with the integral of the tracking error and applying pole placement at {−8±6j, −30, −30, −30} yields a controller with guaranteed zero steady-state error for both step references and step disturbances. The observer poles are placed at −30 rad/s.

The state-space approach provides the most systematic and flexible design framework among all methods studied, offering direct control over closed-loop pole locations and the ability to handle disturbance rejection through state augmentation. This motivates the digital RST controller in Module 5, which translates these ideas to discrete time.

ME326 — Control Systems

Computer Exercises — Module 5 Report

Digital RST Controller & Global Comparison

Authors
Ambrine Douhane | Victoria Chalain
Group B11
EPFL — Spring 2021

Introduction

This report presents the results of Module 5 of the ME326 Control Systems computer exercises. The objective of this module is to design a digital RST controller using the pole placement technique for the Quanser rotary flexible joint, and to compare all four controller families studied throughout the course.

The module is divided into three parts:

- Module 5.1 — Selection of the sampling period and comparison of discretisation methods (ZOH, zero-pole matching, Tustin)
- Module 5.2 — Design of the RST digital controller via the Diophantine equation and pole placement
- Module 5.3 — Global comparison of all four controllers: cascade PID, loop-shaping lead-lag, state-space with integrator, and RST

Module 5.1 — Discrete-Time Model

Sampling Period Selection

The desired closed-loop bandwidth is 10 rad/s (consistent with Modules 4.2–4.3). The sampling theorem requires the sampling frequency to be at least twice the highest relevant signal frequency. A commonly used rule of thumb for control systems is to choose the sampling frequency 20–40 times the closed-loop bandwidth:

```
bw = 10;              % desired closed-loop bandwidth [rad/s]
Ts = 2*pi / (40*bw);  % = 2*pi / 400 ≈ 0.01571 s
```

This gives **Ts ≈ 0.01571 s**, corresponding to a sampling frequency of $\omega_s = 400$ rad/s — 40 times the bandwidth. This is well above the Nyquist frequency and also well above the flexible mode at 18.8 rad/s ($\omega_s \approx 21 \times \omega_{flexible}$), ensuring the flexible oscillation is adequately captured.

Comparison of Discretisation Methods

Three discretisation methods were applied to G(s) with the chosen Ts and compared against the continuous-time step response (shown in the uploaded figure):

Method	MATLAB command	Principle
Zero-Order Hold (ZOH)	c2d(G, Ts, 'zoh')	Exact for piecewise-constant inputs; preserves DC gain
Zero-Pole Matching (ZPM)	c2d(G, Ts, 'matched')	Maps s-domain poles/zeros to z-domain; good for resonant systems
Tustin (bilinear)	c2d(G, Ts, 'tustin')	Frequency warping; preserves stability margin shape

From the step response comparison (uploaded figure), all three discrete models closely track the continuous-time response G(s) at the chosen sampling rate. The staircase shape reflects the zero-order hold effect between samples. ZOH and ZPM are nearly identical, while Tustin shows a marginal deviation in the initial transient.

The ZOH method was selected as the best approximation because it is exact for piecewise-constant inputs (which is the nature of digital control), correctly preserves the DC gain, and is the standard choice for digital control design.

```
Gz = c2d(G, Ts, 'zoh');   % selected discretisation
```

Module 5.2 — RST Digital Controller Design

RST Controller Structure

The RST controller is a general two-degree-of-freedom digital controller defined by three polynomials in the backward shift operator z^{-1}:

- $R(z^{-1})$ — feedback polynomial acting on the output y
- $S(z^{-1})$ — feedback polynomial acting on the control error (contains the integrator)
- $T(z^{-1})$ — feedforward polynomial acting on the reference r

The control law is: $S(z^{-1})\cdot u(k) = T(z^{-1})\cdot r(k) - R(z^{-1})\cdot y(k)$

The closed-loop transfer function from r to y is: $T\_cl(z^{-1}) = B\cdot T / (A\cdot S + B\cdot R)$ where A and B are the denominator and numerator polynomials of the discrete plant Gz.

Step 1 — Desired Closed-Loop Polynomial

The desired closed-loop dynamics are specified by $\omega n = 10$ rad/s and $\zeta = 0.8$, matching the Module 4.3 specifications. The continuous-time poles are mapped to the z-domain:

```
wn = 10;    damping = 0.8;

p1 = -2 * exp(-damping*wn*Ts) * cos(wn*Ts*sqrt(1-damping^2));
p2 =      exp(-2*damping*wn*Ts);
```

```
Pd = [1, p1, p2];   % desired 2nd-order closed-loop polynomial
```

This 2nd-order desired polynomial is padded to match the degree of the Sylvester equation (nA = 5, nB = 4, giving a 9th-order closed-loop polynomial):

```
Pd_long = [1, p1, p2, zeros(1, 5+4-3)];   % length 9
```

Step 2 — Plant Polynomials A, B and A'

The numerator B and denominator A of Gz are extracted as coefficient vectors:

```
[B, A] = tfdata(Gz, 'v');
```

To include an integrator in the controller (for zero steady-state error), A is augmented with Hs = $(1 - z^{-1})$, the discrete integrator in polynomial form:

```
Hs      = [1, -1];
A_prime = conv(A, Hs);   % A'(z^-1) = A(z^-1) * (1 - z^-1)
```

Step 3 — Sylvester Matrix and Diophantine Equation

The Diophantine equation A'·S' + B·R = Pd is solved by constructing the Sylvester (Bézout) matrix M. With deg(A') = 5 and deg(B) = 4:

- deg(S') = deg(B) − 1 = 3 → S' has 4 coefficients
- deg(R) = deg(A') − 1 = 4 → R has 5 coefficients
- M is a 9×9 matrix built by interleaving shifted copies of A' and B

```
c1 = [A_prime, zeros(1,3)]';
c2 = [0, A_prime, zeros(1,2)]';
c3 = [zeros(1,2), A_prime, zeros(1,1)]';
c4 = [zeros(1,3), A_prime]';
c5 = [B, zeros(1,4)]';
c6 = [zeros(1,1), B, zeros(1,3)]';
c7 = [zeros(1,2), B, zeros(1,2)]';
c8 = [zeros(1,3), B, zeros(1,1)]';
c9 = [zeros(1,4), B]';
M  = horzcat(c1,c2,c3,c4,c5,c6,c7,c8,c9);

x = inv(M) * Pd_long';
```

Step 4 — RST Polynomial Coefficients

The solution vector x is split into S' (4 coefficients) and R (5 coefficients). S is then recovered by multiplying S' back by Hs to restore the integrator:

```
S_bis = x(1:4);
S     = conv(S_bis, Hs)';   % restore integrator: S = S' * (1 - z^-1)
R     = x(5:9)';
T     = sum(R);              % feedforward gain: T = R(z=1) for unity
DC gain
```

The feedforward polynomial T is set to the sum of the R coefficients (= R evaluated at z = 1), ensuring unity DC gain of the closed-loop transfer function from r to y.

Closed-Loop Transfer Functions

```
[B, A] = tfdata(Gz, 'v');
P      = conv(A,S) + conv(B,R);       % closed-loop denominator

T_rst = tf(conv(B,T), P, Ts, 'variable', 'z^-1');   % r -> y
U_rst = tf(conv(T,A), P, Ts, 'variable', 'z^-1');   % r -> u
```

Verification — Closed-Loop Poles

The closed-loop poles were verified by computing P = A·S + B·R and comparing its roots against the desired poles. The dominant poles should correspond to the z-domain mapping of $s = -8 \pm 6j$ ($\omega_n = 10$, $\zeta = 0.8$):

```
% Continuous desired poles: s = -8 ± 6j
% Mapped to z-domain: z = exp(s * Ts)
% z ≈ exp((-8+6j)*0.01571) ≈ 0.878 ± 0.091j
```

The remaining poles (from the Sylvester equation degree) are placed away from the unit circle and do not affect the dominant dynamics significantly.

Step Response Results

The closed-loop step response of T_rst and U_rst shows:

- Tracking (r → y): The output converges to unity with low overshoot (~1.5%, consistent with $\zeta = 0.8$) and a settling time of approximately 0.5 s — matching the continuous-time Module 4.3 specification.
- Control effort (r → u): The digital control signal shows a staircase profile between samples, with an initial peak followed by rapid settling. The magnitude is comparable to the state-space controller.
- Comparison with SS controller: Both the RST and state-space (Module 4.3) controllers share the same target poles ($\omega_n = 10$, $\zeta = 0.8$) and produce very similar step responses. Minor differences arise from the discretisation and the digital implementation.

Module 5.3 — Global Comparison of All Controllers

Overview

This section compares all four control strategies designed throughout the course for the Quanser rotary flexible joint. The comparison considers three axes: controller structure complexity, clarity and ease of design, and performance advantages and drawbacks.

Structural Complexity

Controller	Structure	States / Polynomials	Discrete?
Cascade PID (M2.3)	2 nested PID loops	2 transfer functions	No

Lead-lag (M3.2)	1 TF with integrator	1 transfer function	No
SS + integrator (M4.3)	State feedback + observer	9 states (4+4+1)	No
RST digital (M5.2)	R, S, T polynomials	3 polynomials (z-domain)	Yes

The lead-lag controller is the structurally simplest (a single transfer function), while the state-space controller is the most complex (9-dimensional closed-loop system). The RST controller has intermediate complexity — three polynomials — but is inherently digital and operates at discrete time steps.

Design Method Clarity

Controller	Design basis	Key design parameters	Clarity
Cascade PID	Step response + bandwidth	γ, ζ, ωn of G_1; outer kP	Moderate — 2-step iterative
Lead-lag	Frequency domain (Bode)	ωc = 5 rad/s, PM = 55°	High — explicit specs
SS + integrator	Pole placement (time)	ωn = 10, ζ = 0.8, fast = −30	High — direct pole control
RST digital	Pole placement (z-domain)	Same ωn, ζ mapped via Ts	Moderate — Diophantine eq.

The loop-shaping and state-space methods offer the clearest design procedures with directly interpretable specifications. The RST method requires additional bookkeeping (polynomial degrees, Sylvester matrix construction) but ultimately provides the same closed-loop pole placement as the state-space method, in discrete time.

Performance Comparison

Criterion	Cascade PID	Lead-lag	SS + int.	RST
Zero SS error (ref)	Yes	Yes	Yes	Yes
Zero SS error (dist)	Yes	Yes	Yes	Yes
Overshoot	Low	Moderate	Very low	Very low
Settling time	Fast	Moderate	Moderate	Moderate
Flexible mode handling	Inner loop	Partial	LQR/poles	Poles
Robustness (PM)	Not spec.	55°	Implicit	Implicit
Digital implementation	Possible	Possible	Possible	Native

Disturbance bandwidth	High	Medium	Medium	Medium

Advantages and Drawbacks

Cascade PID

- Advantage: Best disturbance rejection among the four. Inner loop at 50 rad/s actively suppresses the flexible mode before it propagates to the position loop.
- Advantage: Intuitive physical interpretation — inner loop controls velocity, outer loop controls position.
- Drawback: Requires tuning of two separate controllers. Inner loop bandwidth must be significantly higher than the flexible mode (18.8 rad/s), which can amplify noise.
- Drawback: No explicit frequency-domain margin guarantees.

Lead-Lag Controller

- Advantage: Single transfer function, easy to implement and certify. Clear frequency-domain specifications (PM = 55°, ωc = 5 rad/s).
- Advantage: The modulus margin MM ≈ 0.638 provides a quantified robustness guarantee.
- Drawback: The flexible mode at 18.8 rad/s is not explicitly suppressed; phase loss limits achievable bandwidth.
- Drawback: Lower bandwidth (5 rad/s) compared to cascade inner loop (50 rad/s), resulting in slower disturbance rejection.

State-Space Controller with Integrator

- Advantage: Most systematic design — pole placement gives direct control over all closed-loop dynamics. Fully exploits state information via the observer.
- Advantage: The augmented integrator guarantees zero steady-state error for both references and disturbances.
- Drawback: Highest implementation complexity (9-state closed-loop). Requires all states to be estimated, which may be sensitive to model uncertainty.
- Drawback: LQR weighting matrices (Q, R) require iterative tuning to satisfy time-domain constraints.

RST Digital Controller

- Advantage: Natively discrete — designed for digital implementation without further approximation. Pole placement in z-domain mirrors the state-space approach.
- Advantage: The integrator is embedded in S via the factor $(1 - z^{-1})$, guaranteeing disturbance rejection by construction.
- Drawback: The Diophantine/Sylvester equation approach is less intuitive than state-space or frequency-domain methods, and numerical conditioning of the 9×9 matrix must be checked.
- Drawback: Polynomial degree constraints (deg A', deg B) limit flexibility in placing all closed-loop poles freely.

Summary Table

Criterion	Cascade PID	Lead-lag	SS + int.	RST
Structure complexity	★★★	★	★★★★★	★★★
Design clarity	★★★	★★★★★	★★★★	★★★
Tracking performance	★★★★	★★★	★★★★	★★★★
Disturbance rejection	★★★★★	★★★	★★★★	★★★★
Robustness guarantee	★★	★★★★★	★★★	★★★
Digital suitability	★★★	★★★	★★★	★★★★★

Conclusion

This module completed the design of a digital RST controller for the Quanser rotary flexible joint and provided a comprehensive comparison of all four control approaches studied in this course.

Key findings from Module 5:

- Sampling time: $T_s \approx 0.01571$ s (40× the bandwidth) ensures adequate sampling of both the control dynamics and the flexible mode. ZOH discretisation was selected as the most faithful method for digital control.
- RST design: The Diophantine equation (Sylvester matrix) was solved to place the dominant closed-loop poles at $\omega_n = 10$ rad/s, $\zeta = 0.8$ in the z-domain. The integrator in S ensures zero steady-state error for step disturbances.
- The RST and state-space controllers produce nearly identical step responses, as they share the same target poles — the key difference is that RST operates natively in discrete time.

Global conclusions:

- For practical implementation on a digital controller (microcontroller, DSP), the RST approach is most natural.
- For explicit robustness certification (e.g. aerospace, medical), the loop-shaping approach with guaranteed phase and modulus margins is preferable.
- For the best disturbance rejection and flexible mode suppression, the cascade PID remains the top performer due to its high-bandwidth inner loop.
- The state-space controller offers the most complete theoretical framework but requires the most implementation effort and observer tuning.

This series of exercises demonstrates that no single control method is universally superior — the choice depends on the implementation context, performance requirements, robustness needs, and available sensors.