ESRF Double Crystal Monochromator - Dynamical Multi-Body Model

Dehaeze Thomas

June 2, 2022

Contents

In this document, a Simscape (.e.g. multi-body) model of the ESRF Double Crystal Monochromator (DCM) is presented and used to develop and optimize the control strategy.

It is structured as follow:

- Section 1: the system dynamics is identified in the absence of control.
- Section 2: an open-loop noise budget is performed.
- Section 3: it is studied whether if the strain gauges fixed to the piezoelectric actuators can be used to actively damp the plant.
- Section 4: piezoelectric force sensors are added in series with the piezoelectric actuators and are used to actively damp the plant using the Integral Force Feedback (IFF) control strategy.
- Section 6: the High Authority Control - Low Authority Control (HAC-LAC) strategy is tested on the Simscape model.

1 Open Loop System Identification

1.1 Identification

Let's considered the system $\boldsymbol{G}(s)$ with:

- 3 inputs: force applied to the 3 fast jacks
- 3 outputs: measured displacement by the 3 interferometers pointing at the ring second crystal

It is schematically shown in Figure 1.1.

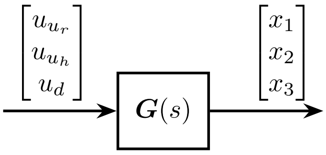

$\begin{bmatrix} u_{u_r} \\ u_{u_h} \\ u_d \end{bmatrix}$
$\boldsymbol{G}(s)$
$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$

Figure 1.1: Dynamical system with inputs and outputs

The system is identified from the Simscape model.

Matlab

```
%% Input/Output definition
clear io; io_i = 1;

%% Inputs
% Control Input {3x1} [N]
io(io_i) = linio([mdl, '/control_system'], 1, 'openinput');  io_i = io_i + 1;

%% Outputs
% Interferometers {3x1} [m]
io(io_i) = linio([mdl, '/DCM'], 1, 'openoutput'); io_i = io_i + 1;
```

Matlab

```
%% Extraction of the dynamics
G = linearize(mdl, io);
```

Matlab

```
size(G)
```

Results

```
size(G)
State-space model with 3 outputs, 3 inputs, and 24 states.
```

1.2 Plant in the frame of the fastjacks

Matlab
```matlab
load('dcm_kinematics.mat');
```

Using the forward and inverse kinematics, we can computed the dynamics from piezo forces to axial motion of the 3 fastjacks (see Figure 1.2).

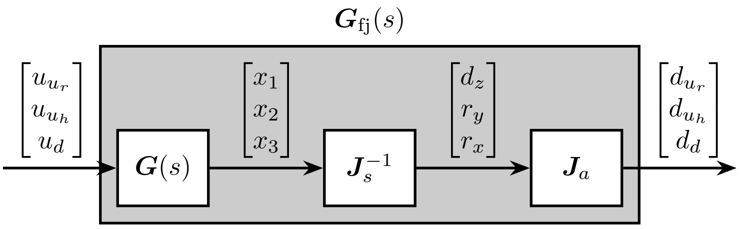

$\boldsymbol{G}_{\text{fj}}(s)$
$\begin{bmatrix} u_{u_r} \\ u_{u_h} \\ u_d \end{bmatrix}$
$\boldsymbol{G}(s)$
$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$
$\boldsymbol{J}_s^{-1}$
$\begin{bmatrix} d_z \\ r_y \\ r_x \end{bmatrix}$
$\boldsymbol{J}_a$
$\begin{bmatrix} d_{u_r} \\ d_{u_h} \\ d_d \end{bmatrix}$

Figure 1.2: Use of Jacobian matrices to obtain the system in the frame of the fastjacks

Matlab
```matlab
%% Compute the system in the frame of the fastjacks
G_pz = J_a_h*inv(J_2h_s)*G;
```

The DC gain of the new system shows that the system is well decoupled at low frequency.

Matlab
```matlab
dcgain(G_pz)
```

Table 1.1: DC gain of the plant in the frame of the fast jacks $\boldsymbol{G}_{\text{fj}}$

4.4407e-09	2.7656e-12	1.0132e-12
2.7656e-12	4.4407e-09	1.0132e-12
1.0109e-12	1.0109e-12	4.4424e-09

The bode plot of $\boldsymbol{G}_{\text{fj}}(s)$ is shown in Figure 1.3.

Matlab
```matlab
G_pz = diag(1./diag(dcgain(G_pz)))*G_pz;
```

Important

Computing the system in the frame of the fastjack gives good decoupling at low frequency (until the first resonance of the system).

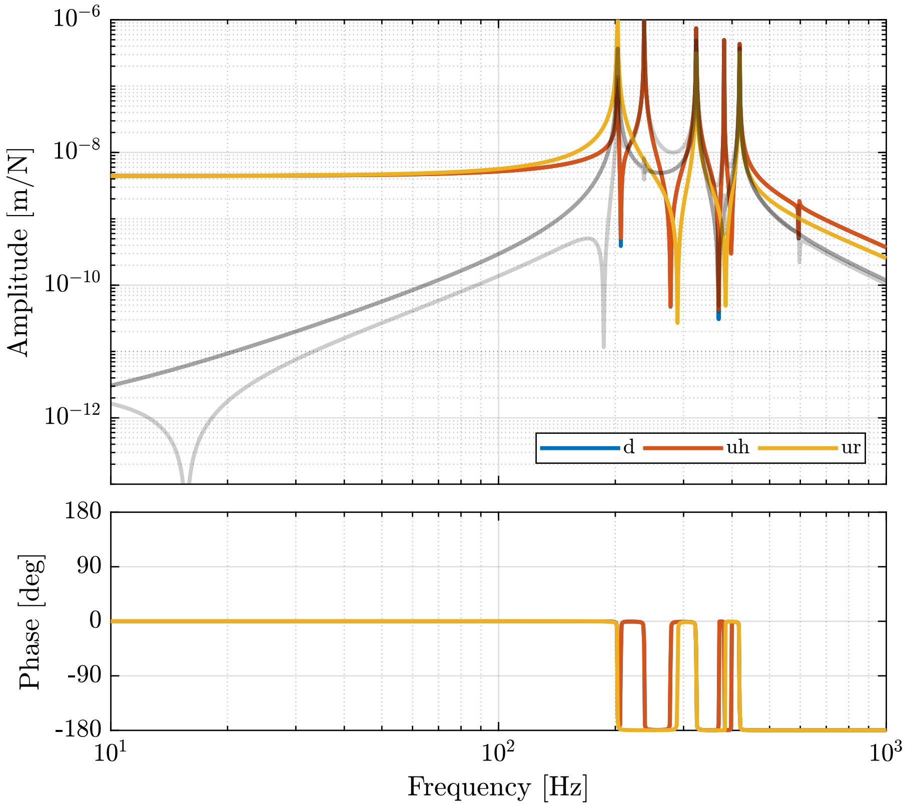
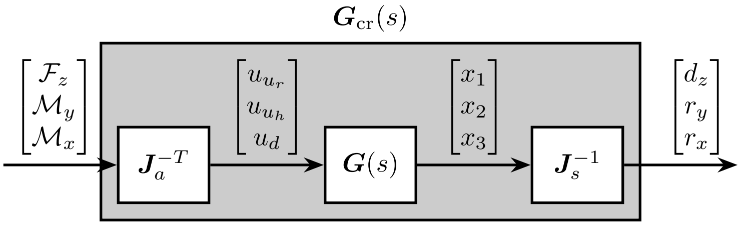


Figure 1.3: Bode plot of the diagonal and off-diagonal elements of the plant in the frame of the fast jacks



$\boldsymbol{G}_{\text{cr}}(s)$
$\begin{bmatrix} \mathcal{F}_z \\ \mathcal{M}_y \\ \mathcal{M}_x \end{bmatrix}$
$\boldsymbol{J}_a^{-T}$
$\begin{bmatrix} u_{u_r} \\ u_{u_h} \\ u_d \end{bmatrix}$
$\boldsymbol{G}(s)$
$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$
$\boldsymbol{J}_s^{-1}$
$\begin{bmatrix} d_z \\ r_y \\ r_x \end{bmatrix}$

Figure 1.4: Use of Jacobian matrices to obtain the system in the frame of the crystal

1.3 Plant in the frame of the crystal

Matlab

```
G_mr = inv(J_s_r)*G*inv(J_a_r');
```

Matlab

```
dcgain(G_mr)
```

```
1.9978e-09    3.9657e-09    7.7944e-09
3.9656e-09    8.4979e-08   -1.5135e-17
7.7944e-09   -3.9252e-17     1.834e-07
```

This results in a coupled system. The main reason is that, as we map forces to the center of the ring crystal and not at the center of mass/stiffness of the moving part, vertical forces will induce rotation and torques will induce vertical motion.

2 Open-Loop Noise Budgeting

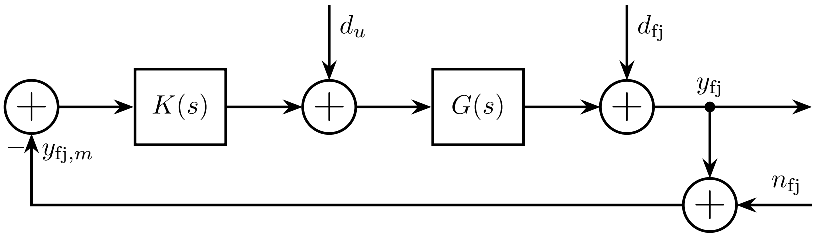


Figure 2.1: Schematic representation of the control loop in the frame of one fast jack

2.1 Power Spectral Density of signals

Interferometer noise:

Matlab

```
Wn = 6e-11*(1 + s/2/pi/200)/(1 + s/2/pi/60); % m/sqrt(Hz)
```

Results

```
Measurement noise: 0.79 [nm,rms]
```

DAC noise (amplified by the PI voltage amplifier, and converted to newtons):

Matlab

```
Wdac = tf(3e-8); % V/sqrt(Hz)
Wu = Wdac*22.5*10; % N/sqrt(Hz)
```

Results

```
DAC noise: 0.95 [uV,rms]
```

Disturbances:

Matlab

```
Wd = 5e-7/(1 + s/2/pi); % m/sqrt(Hz)
```

Results

```
Disturbance motion: 0.61 [um,rms]
```

Matlab

```matlab
%% Save ASD of noise and disturbances
save('mat/asd_noises_disturbances.mat', 'Wn', 'Wu', 'Wd');
```

2.2 Open Loop disturbance and measurement noise

The comparison of the amplitude spectral density of the measurement noise and of the jack parasitic motion is performed in Figure 2.2. It confirms that the sensor noise is low enough to measure the motion errors of the crystal.

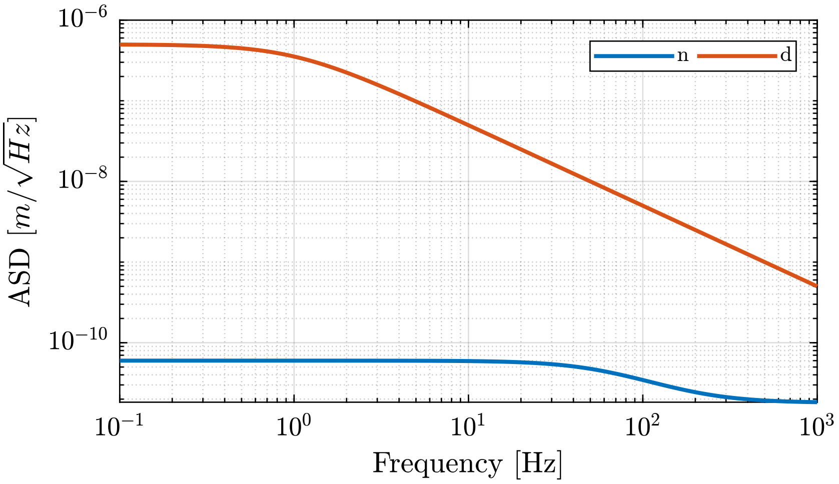


Figure 2.2: Open Loop noise budgeting

3 Active Damping Plant (Strain gauges)

In this section, we wish to see whether if strain gauges fixed to the piezoelectric actuator can be used for active damping.

3.1 Identification

```matlab
%% Input/Output definition
clear io; io_i = 1;

%% Inputs
% Control Input {3x1} [N]
io(io_i) = linio([mdl, '/control_system'], 1, 'openinput');  io_i = io_i + 1;

%% Outputs
% Strain Gauges {3x1} [m]
io(io_i) = linio([mdl, '/DCM'], 2, 'openoutput'); io_i = io_i + 1;
```

```matlab
%% Extraction of the dynamics
G_sg = linearize(mdl, io);
```

```matlab
dcgain(G_sg)
```

```
4.4443e-09   1.0339e-13    3.774e-14
1.0339e-13   4.4443e-09    3.774e-14
3.7792e-14   3.7792e-14   4.4444e-09
```

Important

As the distance between the poles and zeros in Figure 4.1 is very small, little damping can be actively added using the strain gauges. This will be confirmed using a Root Locus plot.

3.2 Relative Active Damping

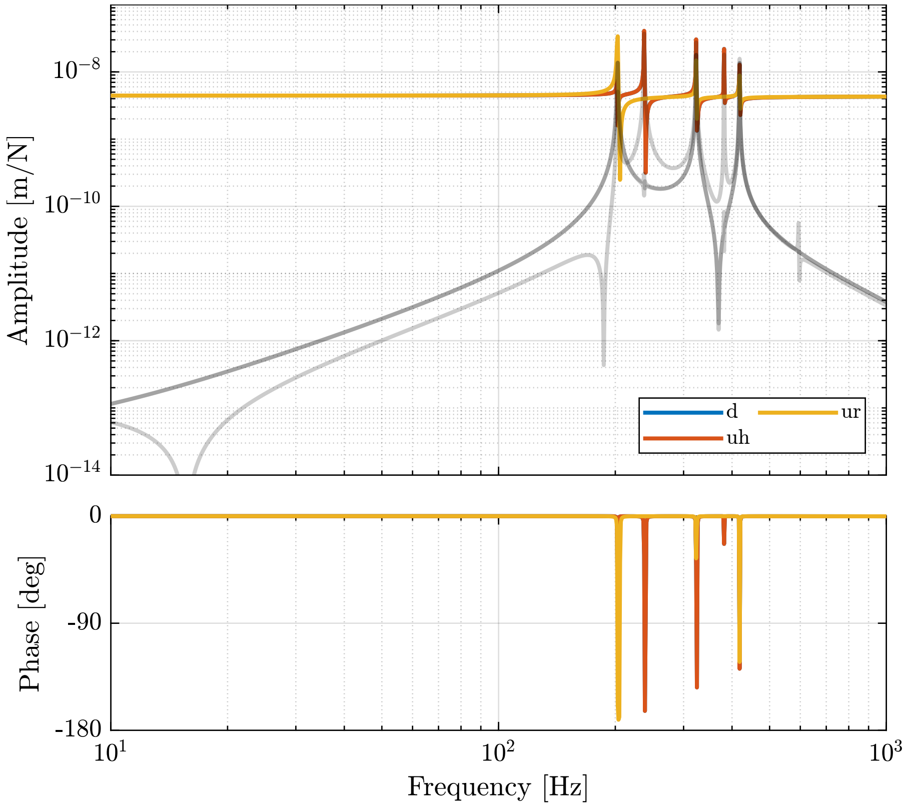


Figure 3.1: Bode Plot of the transfer functions from piezoelectric forces to strain gauges measuremed displacements

```
Krad_g1 = eye(3)*s/(s^2/(2*pi*500)^2 + 2*s/(2*pi*500) + 1);
```

As can be seen in Figure 3.2, very little damping can be added using relative damping strategy using strain gauges.

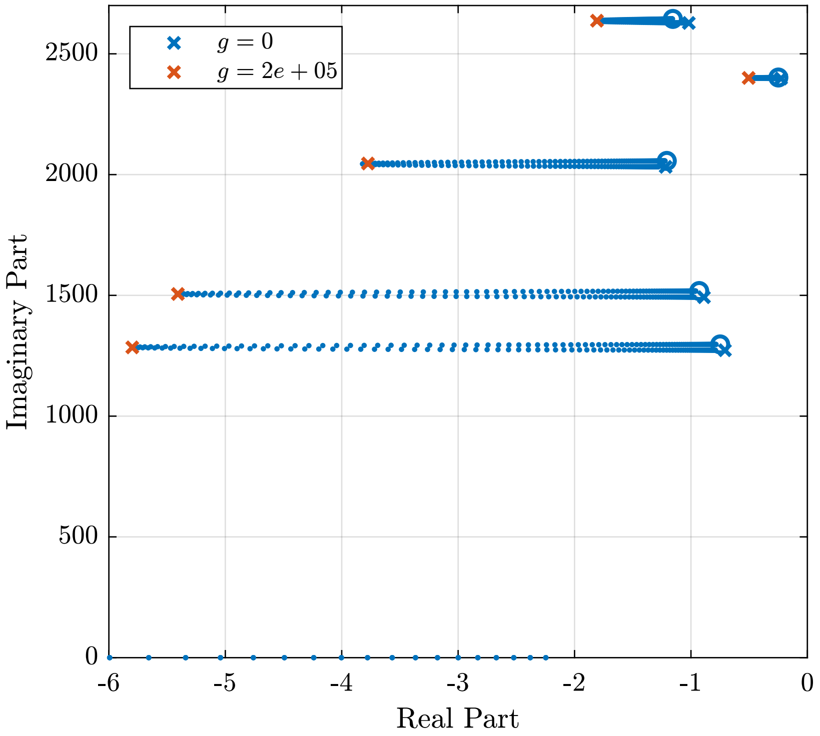


Figure 3.2: Root Locus for the relative damping control

```
Krad = -g*Krad_g1;
```

3.3 Damped Plant

The controller is implemented on Simscape, and the damped plant is identified.

```
%% Input/Output definition
clear io; io_i = 1;

%% Inputs
% Control Input {3x1} [N]
io(io_i) = linio([mdl, '/control_system'], 1, 'input');  io_i = io_i + 1;

%% Outputs
% Force Sensor {3x1} [m]
io(io_i) = linio([mdl, '/DCM'], 1, 'openoutput'); io_i = io_i + 1;
```

Matlab

```
%% DCM Kinematics
load('dcm_kinematics.mat');
```

Matlab

```
%% Identification of the Open Loop plant
controller.type = 0; % Open Loop
G_ol = J_a_r*inv(J_s_r)*linearize(mdl, io);
G_ol.InputName  = {'u_ur',  'u_uh',  'u_d'};
G_ol.OutputName = {'d_ur',  'd_uh',  'd_d'};
```

Matlab

```
%% Identification of the damped plant with Relative Active Damping
controller.type = 2; % RAD
G_dp = J_a_r*inv(J_s_r)*linearize(mdl, io);
G_dp.InputName  = {'u_ur',  'u_uh',  'u_d'};
G_dp.OutputName = {'d_ur',  'd_uh',  'd_d'};
```

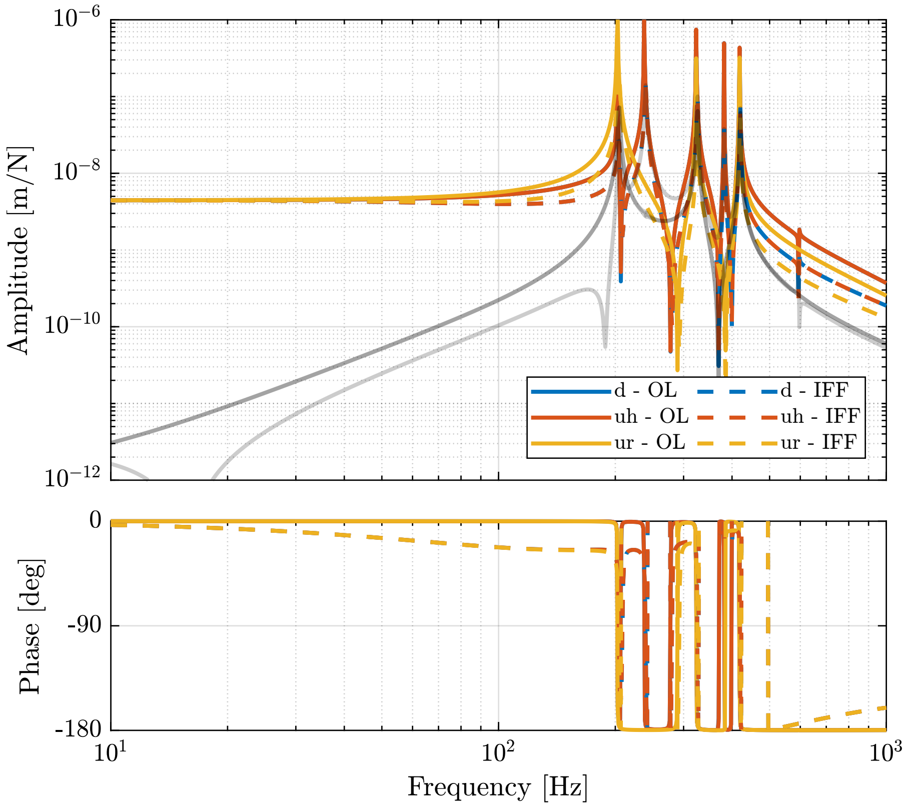


Figure 3.3: Bode plot of both the open-loop plant and the damped plant using relative active damping

4 Active Damping Plant (Force Sensors)

Force sensors are added above the piezoelectric actuators. They can consists of a simple piezoelectric ceramic stack. See for instance [1].

4.1 Identification

Matlab

```
%% Input/Output definition
clear io; io_i = 1;

%% Inputs
% Control Input {3x1} [N]
io(io_i) = linio([mdl, '/control_system'], 1, 'openinput');  io_i = io_i + 1;

%% Outputs
% Force Sensor {3x1} [m]
io(io_i) = linio([mdl, '/DCM'], 3, 'openoutput'); io_i = io_i + 1;
```

Matlab

```
%% Extraction of the dynamics
G_fs = linearize(mdl, io);
```

The Bode plot of the identified dynamics is shown in Figure 4.1. At high frequency, the diagonal terms are constants while the off-diagonal terms have some roll-off.

4.2 Controller - Root Locus

We want to have integral action around the resonances of the system, but we do not want to integrate at low frequency. Therefore, we can use a low pass filter.

Matlab

```
%% Integral Force Feedback Controller
Kiff_g1 = eye(3)*1/(1 + s/2/pi/20);
```

Matlab

```
%% Integral Force Feedback Controller with optimal gain
Kiff = g*Kiff_g1;
```

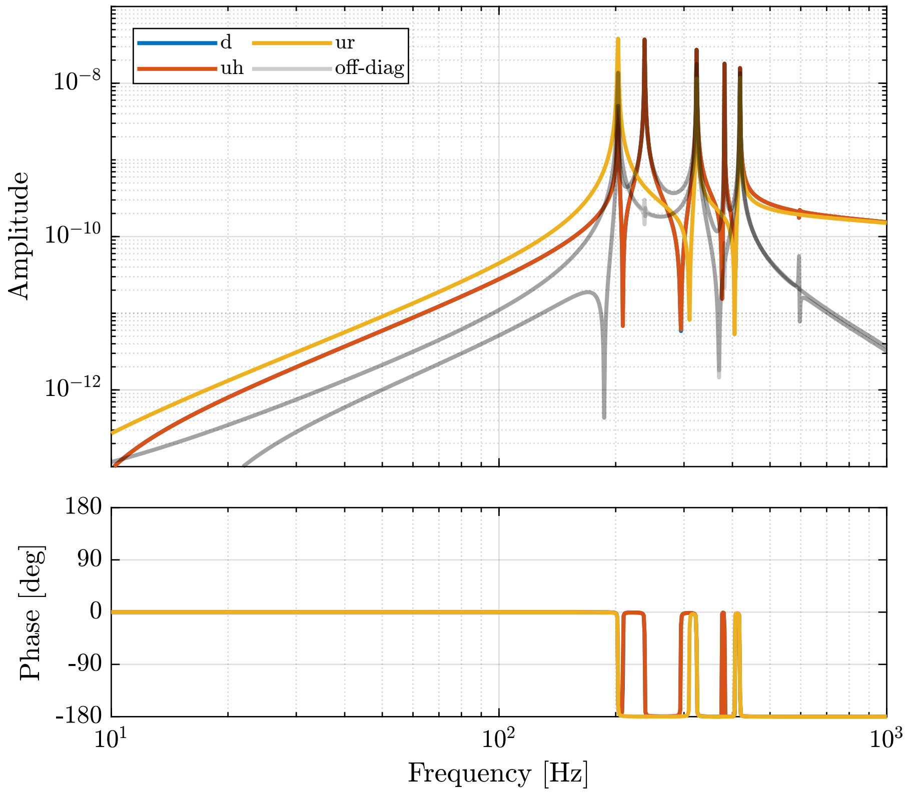


Figure 4.1: Bode plot of IFF Plant

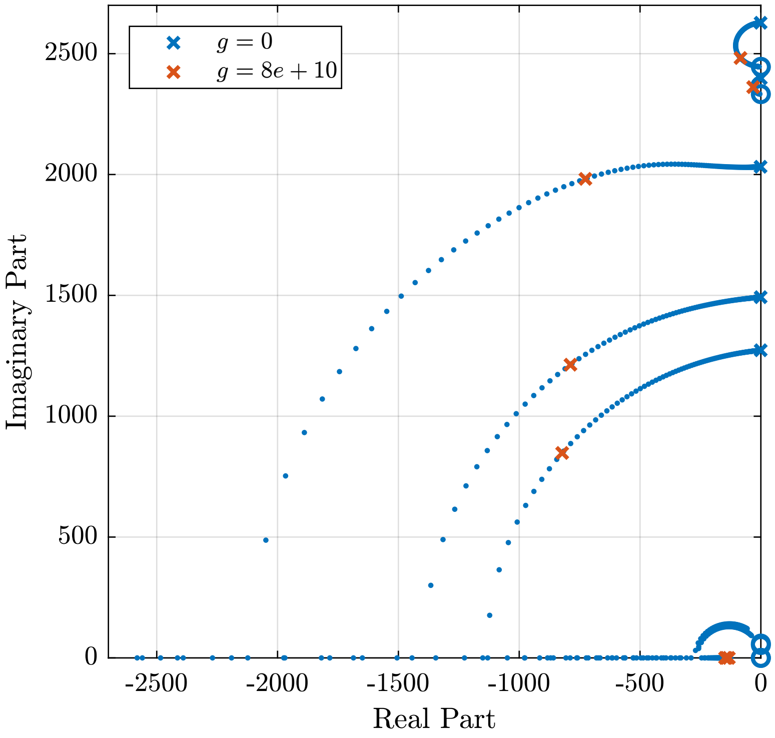


Figure 4.2: Root Locus plot for the IFF Control strategy

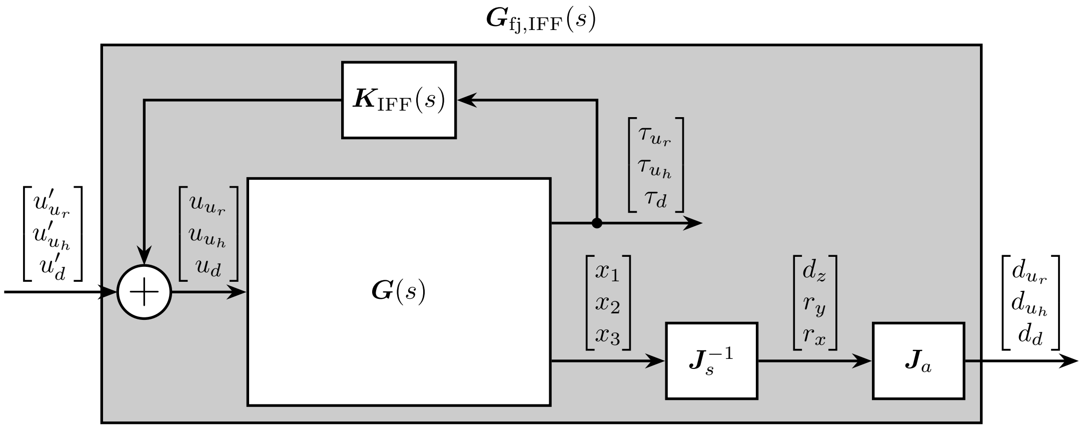
Matlab

```matlab
%% Save the IFF controller
save('mat/Kiff.mat', 'Kiff');
```

4.3 Damped Plant

Both the Open Loop dynamics (see Figure 1.2) and the dynamics with IFF (see Figure 4.3) are identified.

We are here interested in the dynamics from $\boldsymbol{u}' = [u'_{u_r},\ u'_{u_h},\ u'_d]$ (input of the damped plant) to $\boldsymbol{d}_{\text{fj}} = [d_{u_r},\ d_{u_h},\ d_d]$ (motion of the crystal expressed in the frame of the fast-jacks). This is schematically represented in Figure 4.3.



$\boldsymbol{G}_{\text{fj,IFF}}(s)$
$\boldsymbol{K}_{\text{IFF}}(s)$
$\begin{bmatrix} \tau_{u_r} \\ \tau_{u_h} \\ \tau_d \end{bmatrix}$
$\begin{bmatrix} u'_{u_r} \\ u'_{u_h} \\ u'_d \end{bmatrix}$
$\begin{bmatrix} u_{u_r} \\ u_{u_h} \\ u_d \end{bmatrix}$
$\boldsymbol{G}(s)$
$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$
$\begin{bmatrix} d_z \\ r_y \\ r_x \end{bmatrix}$
$\begin{bmatrix} d_{u_r} \\ d_{u_h} \\ d_d \end{bmatrix}$
$\boldsymbol{J}_s^{-1}$
$\boldsymbol{J}_a$

Figure 4.3: Use of Jacobian matrices to obtain the system in the frame of the fastjacks

The dynamics from $\boldsymbol{u}$ to $\boldsymbol{d}_{\text{fj}}$ (open-loop dynamics) and from $\boldsymbol{u}'$ to $\boldsymbol{d}_{\text{fs}}$ are compared in Figure 4.4. It is clear that the Integral Force Feedback control strategy is very effective in damping the resonances of the plant.

Important

The Integral Force Feedback control strategy is very effective in damping the modes present in the plant.

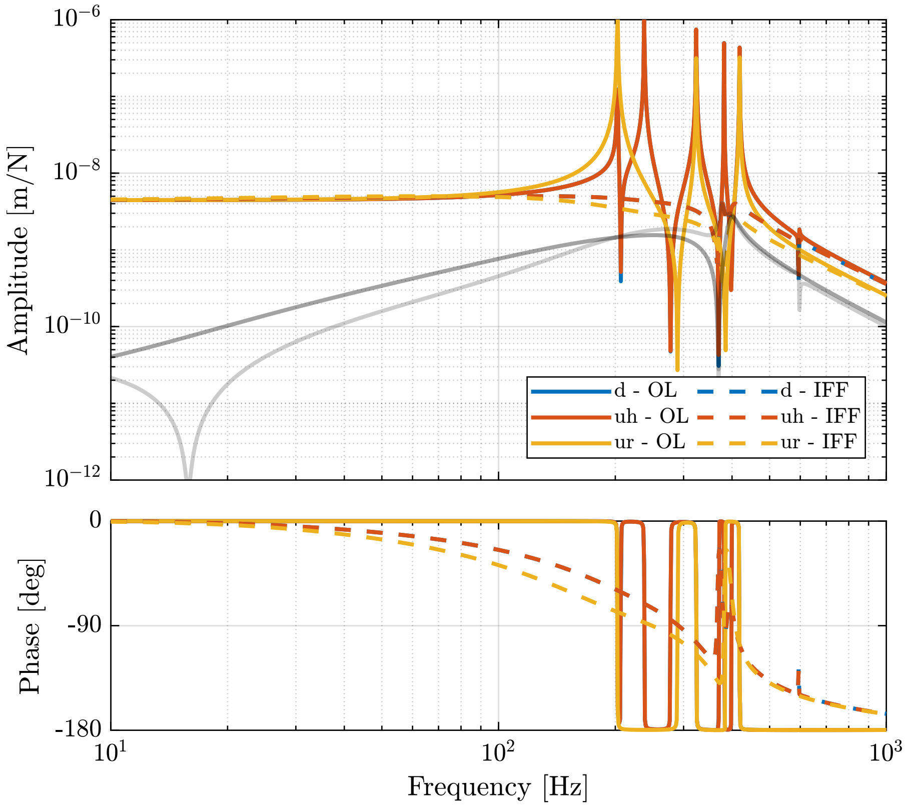
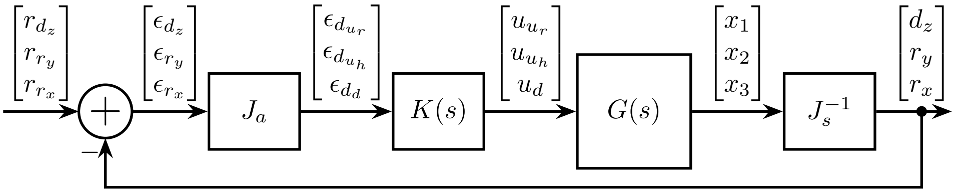


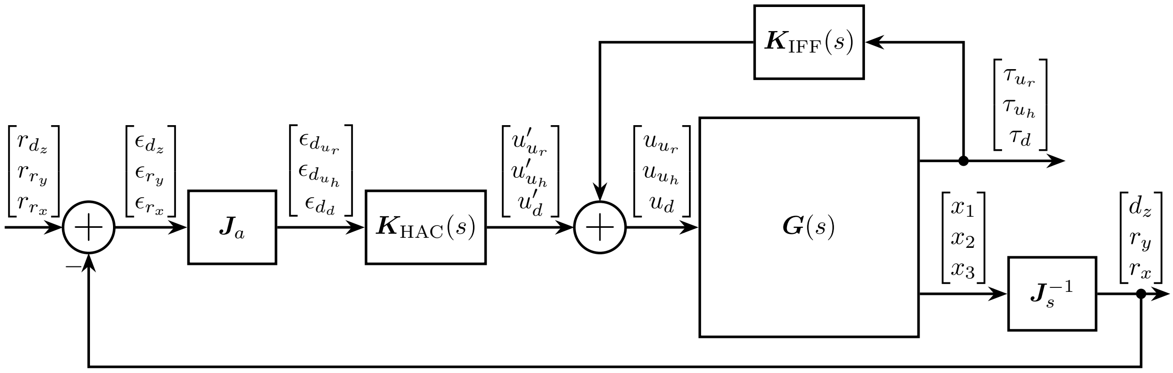
Figure 4.4: Bode plot of both the open-loop plant and the damped plant using IFF

5 Feedback Control


$\begin{bmatrix} r_{d_z} \\ r_{r_y} \\ r_{r_x} \end{bmatrix}$
$-$
$\begin{bmatrix} \epsilon_{d_z} \\ \epsilon_{r_y} \\ \epsilon_{r_x} \end{bmatrix}$
J_a
$\begin{bmatrix} \epsilon_{d_{u_r}} \\ \epsilon_{d_{u_h}} \\ \epsilon_{d_d} \end{bmatrix}$
$K(s)$
$\begin{bmatrix} u_{u_r} \\ u_{u_h} \\ u_d \end{bmatrix}$
$G(s)$
$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$
J_s^{-1}
$\begin{bmatrix} d_z \\ r_y \\ r_x \end{bmatrix}$

6 HAC-LAC (IFF) architecture

The HAC-LAC architecture is shown in Figure 6.1.



$\boldsymbol{K}_{\text{IFF}}(s)$
$\begin{bmatrix} r_{d_z} \\ r_{r_y} \\ r_{r_x} \end{bmatrix}$
$-$
$\begin{bmatrix} \epsilon_{d_z} \\ \epsilon_{r_y} \\ \epsilon_{r_x} \end{bmatrix}$
$\boldsymbol{J}_a$
$\begin{bmatrix} \epsilon_{d_{u_r}} \\ \epsilon_{d_{u_h}} \\ \epsilon_{d_d} \end{bmatrix}$
$\boldsymbol{K}_{\text{HAC}}(s)$
$\begin{bmatrix} u_{u_r}^\prime \\ u_{u_h}^\prime \\ u_d^\prime \end{bmatrix}$
$\begin{bmatrix} u_{u_r} \\ u_{u_h} \\ u_d \end{bmatrix}$
$\boldsymbol{G}(s)$
$\begin{bmatrix} \tau_{u_r} \\ \tau_{u_h} \\ \tau_d \end{bmatrix}$
$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$
$\boldsymbol{J}_s^{-1}$
$\begin{bmatrix} d_z \\ r_y \\ r_x \end{bmatrix}$

Figure 6.1: HAC-LAC architecture

6.1 System Identification

Let's identify the damped plant.

6.2 High Authority Controller

Let's design a controller with a bandwidth of 100Hz. As the plant is well decoupled and well approximated by a constant at low frequency, the high authority controller can easily be designed with SISO loop shaping.

```
Matlab
%% Controller design
wc = 2*pi*100; % Wanted crossover frequency [rad/s]
a = 2; % Lead parameter

Khac = diag(1./diag(abs(evalfr(G_dp, 1j*wc)))) * ... % Diagonal controller
        wc/s * ... % Integrator
        1/(sqrt(a))*(1 + s/(wc/sqrt(a)))/(1 + s/(wc*sqrt(a))) * ... % Lead
        1/(s^2/(4*wc)^2 + 2*s/(4*wc) + 1); % Low pass filter
```

```
Matlab
%% Save the HAC controller
save('mat/Khac_iff.mat', 'Khac');
```

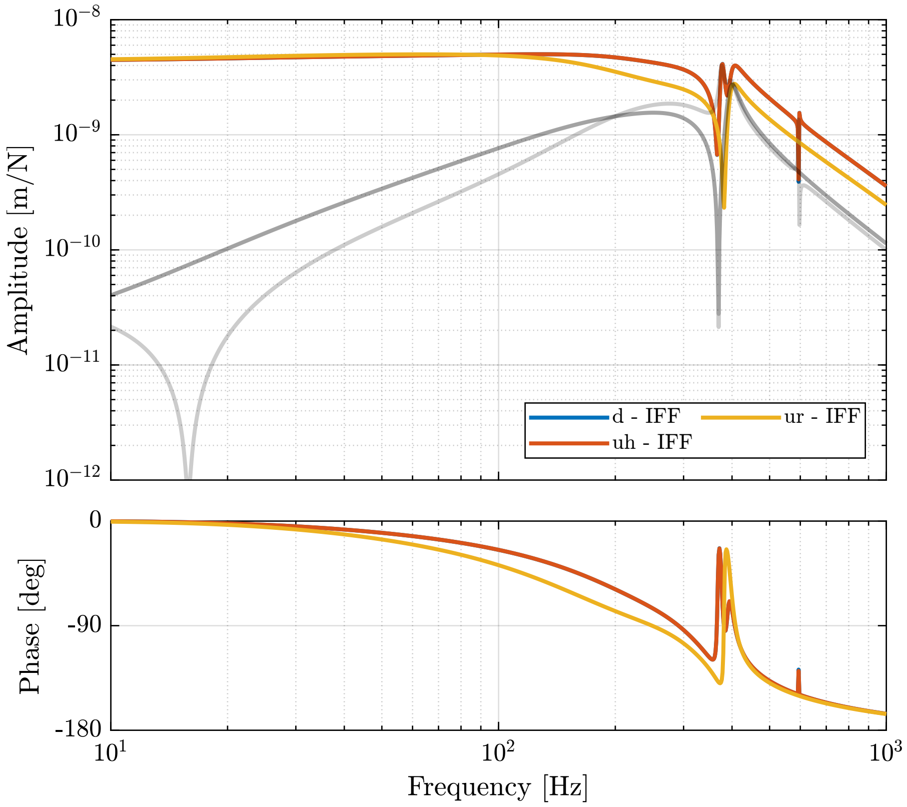


Figure 6.2: Bode Plot of the plant for the High Authority Controller (transfer function from $\boldsymbol{u}'$ to $\boldsymbol{\epsilon}_d$)

Matlab

```
%% Loop Gain
L_hac_lac = G_dp * Khac;
```

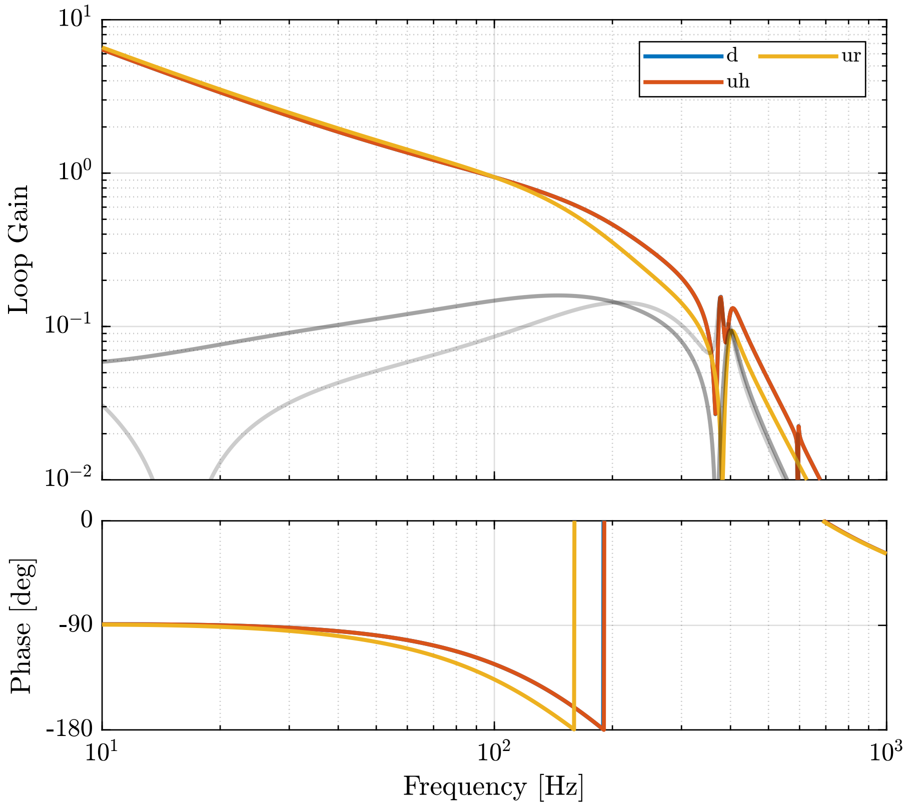


Figure 6.3: Bode Plot of the Loop gain for the High Authority Controller

As shown in the Root Locus plot in Figure 6.4, the closed loop system should be stable.

6.3 Performances

In order to estimate the performances of the HAC-IFF control strategy, the transfer function from motion errors of the stepper motors to the motion error of the crystal is identified both in open loop and with the HAC-IFF strategy.

It is first verified that the closed-loop system is stable:

Matlab

```
isstable(T_hl)
```

Results

```
1
```

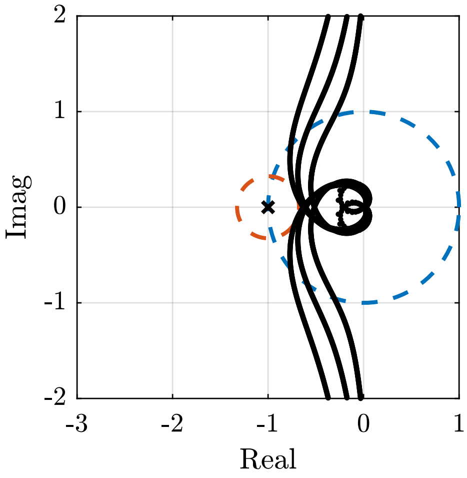


Figure 6.4: Root Locus for the High Authority Controller

And both transmissibilities are compared in Figure 6.5.

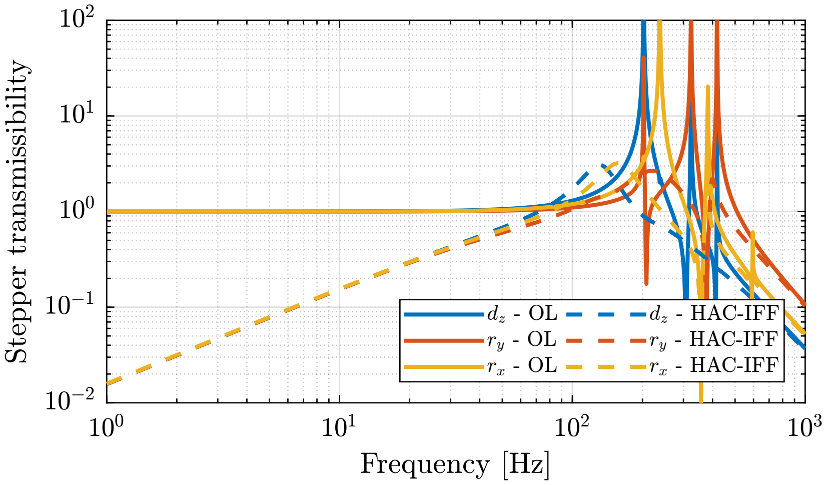


Figure 6.5: Comparison of the transmissibility of errors from vibrations of the stepper motor between the open-loop case and the hac-iff case.

Important

The HAC-IFF control strategy can effectively reduce the transmissibility of the motion errors of the stepper motors. This reduction is effective inside the bandwidth of the controller.

6.4 Close Loop noise budget

Let's compute the amplitude spectral density of the jack motion errors due to the sensor noise, the actuator noise and disturbances.

Matlab

```
%% Computation of ASD of contribution of inputs to the closed-loop motion
% Error due to disturbances
asd_d = abs(squeeze(freqresp(Wd*(1/(1 + G_dp(1,1)*Khac(1,1))), f, 'Hz')));
% Error due to actuator noise
asd_u = abs(squeeze(freqresp(Wu*(G_dp(1,1)/(1 + G_dp(1,1)*Khac(1,1))), f, 'Hz')));
% Error due to sensor noise
asd_n = abs(squeeze(freqresp(Wn*(G_dp(1,1)*Khac(1,1)/(1 + G_dp(1,1)*Khac(1,1))), f, 'Hz')));
```

The closed-loop ASD is then:

Matlab

```
%% ASD of the closed-loop motion
asd_cl = sqrt(asd_d.^2 + asd_u.^2 + asd_n.^2);
```

The obtained ASD are shown in Figure 6.6.

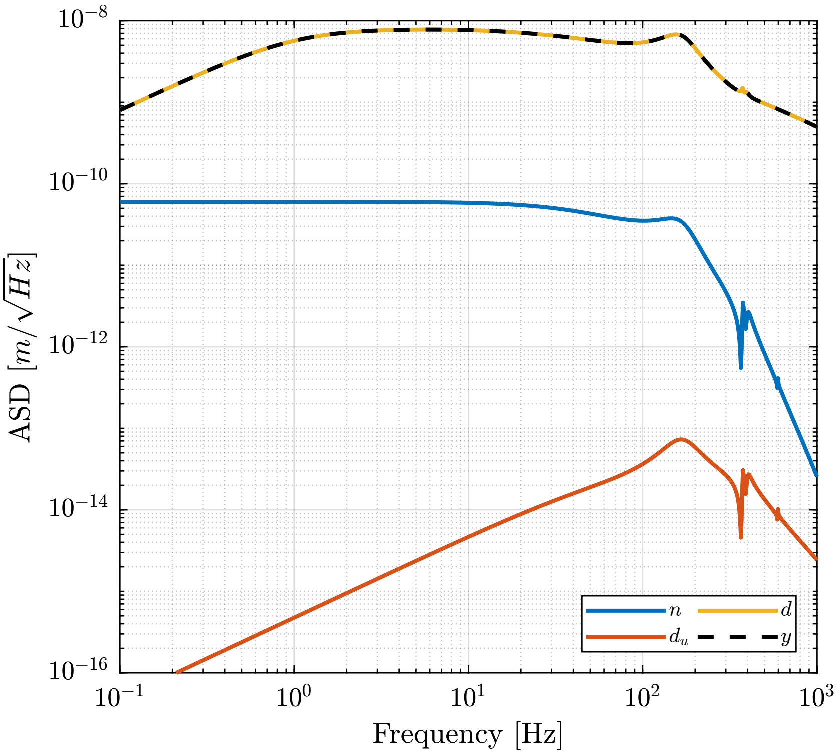


Figure 6.6: Closed Loop noise budget

Let’s compare the open-loop and close-loop cases (Figure 6.7).

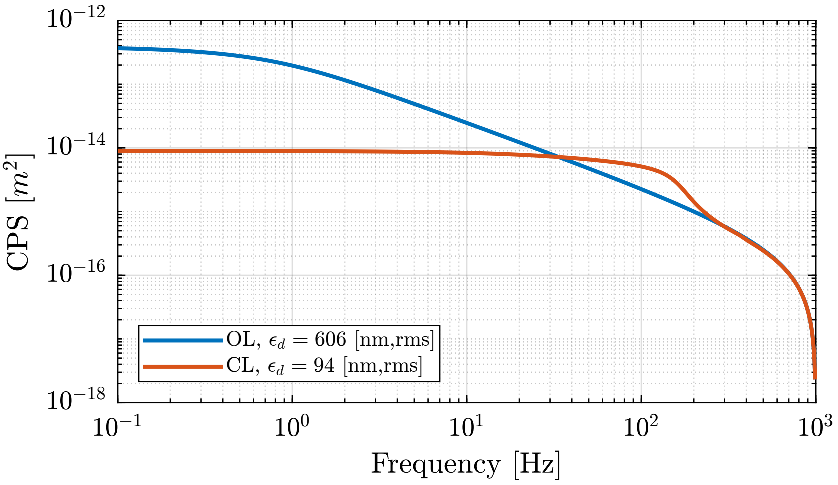


Figure 6.7: Cumulative Power Spectrum of the open-loop and closed-loop motion error along one fast-jack

Bibliography

[1] Andrew J Fleming and Kam K Leang. “Integrated Strain and Force Feedback for High-Performance Control of Piezoelectric Actuators”. In: *Sensors and Actuators A: Physical* 161.1-2 (2010), pp. 256–265.