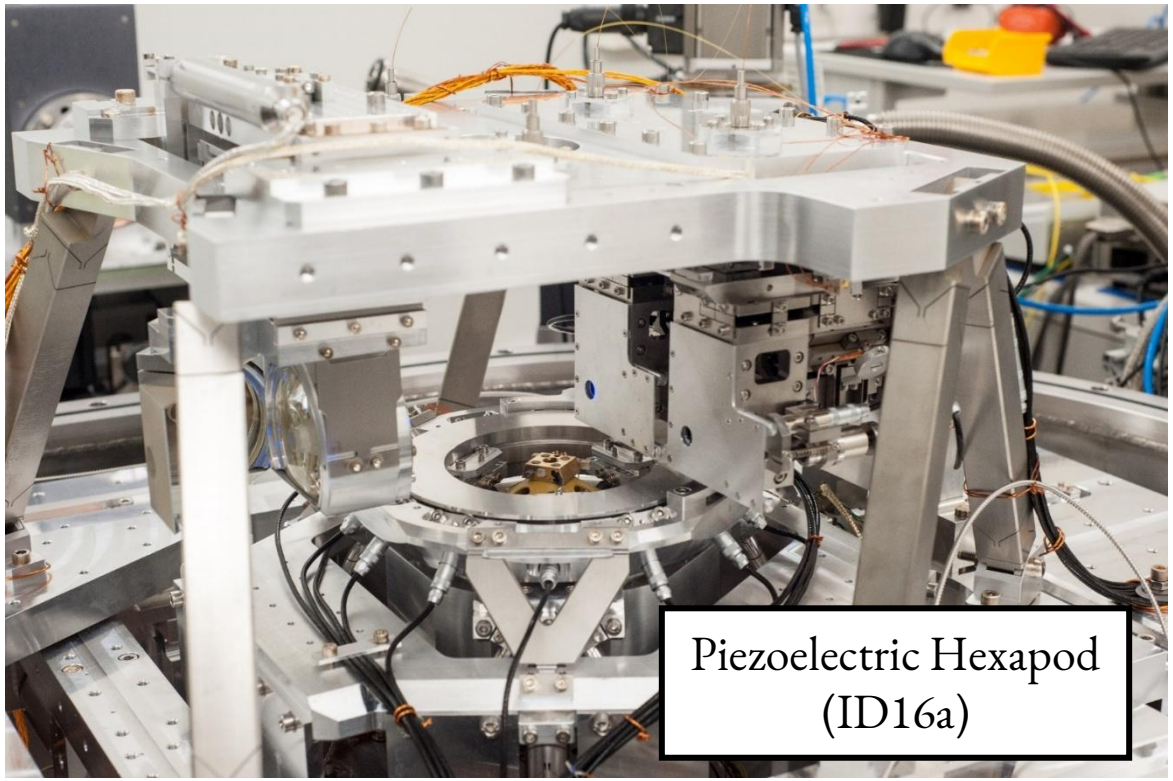
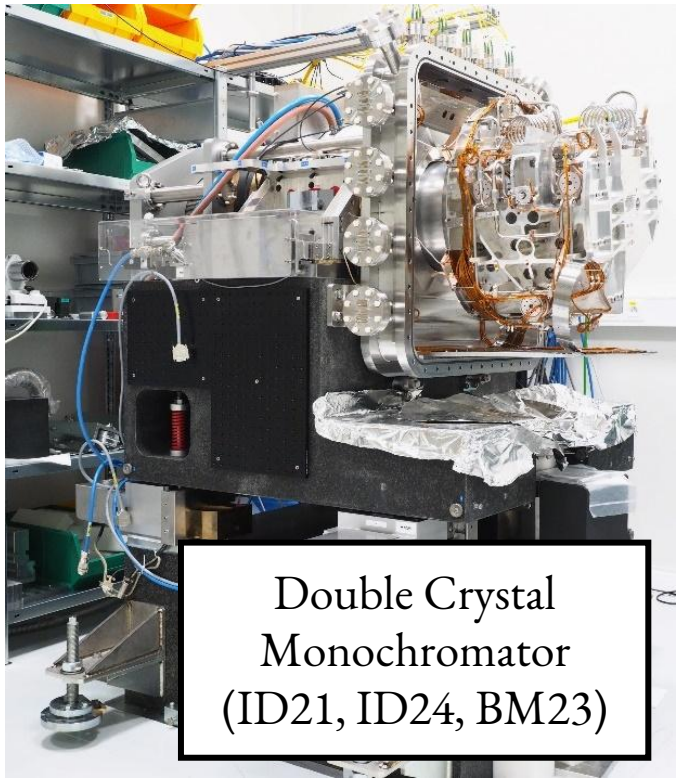
From Model to Motion: Matlab/Simulink for Mechatronics at the ESRF

Thomas Dehaeze

Nano Active Stabilization System (ID31)



Piezoelectric Hexapod (ID16a)



Double Crystal Monochromator (ID21, ID24, BM23)

Outline: Two Practical Examples

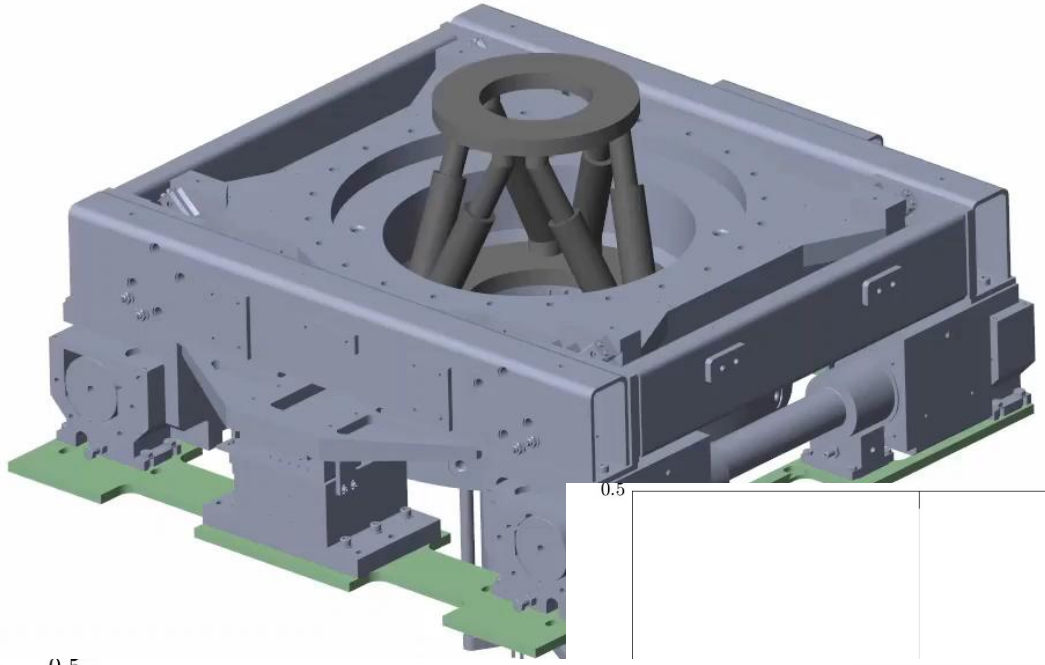
1/ Nano Active Stabilization System (10-15 minutes)

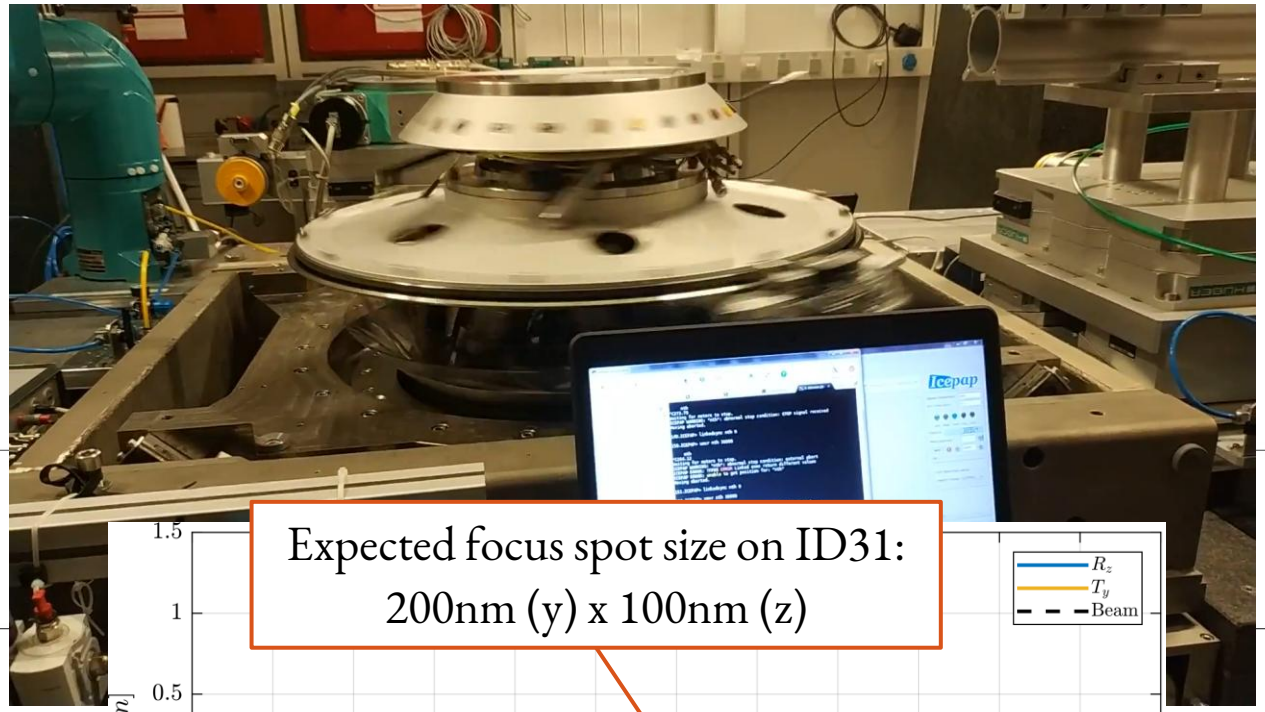
- Mechatronics Design Approach / Model Based Design
- Models and Simulations
- Real-Time control: Experimental Results

2/ ESRF XY(Z) Piezo Stage (10-15 minutes)

- Interfacing between Speedgoat and Python / Bliss
- Common Simulink / Python Library
- Experimental Results

ID31 Positioning Stage: The Micro Station

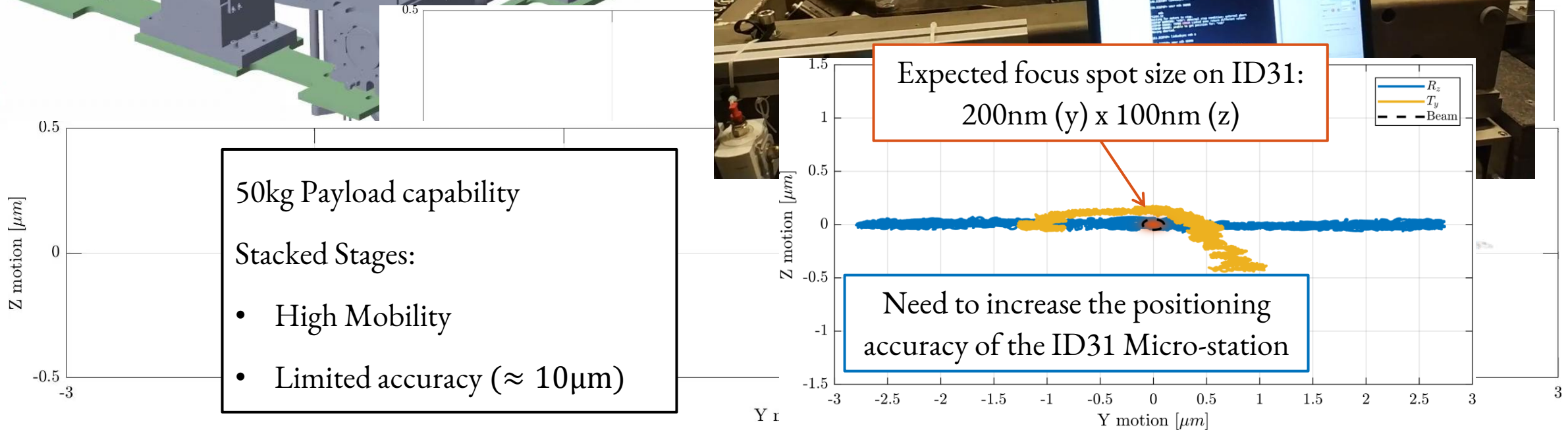




50kg Payload capability

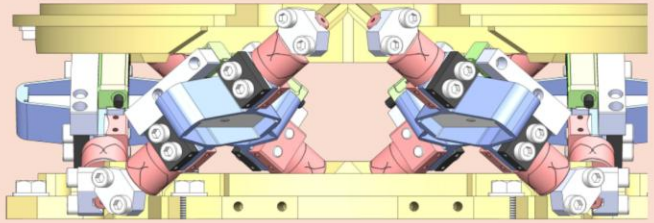
Stacked Stages:

- High Mobility
- Limited accuracy (≈ 10µm)

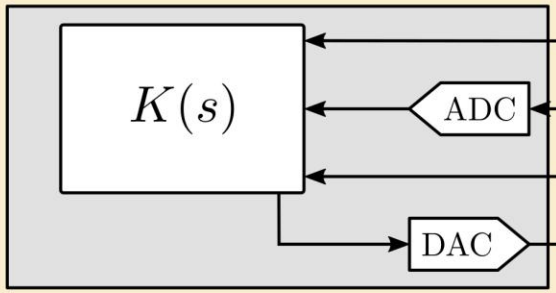


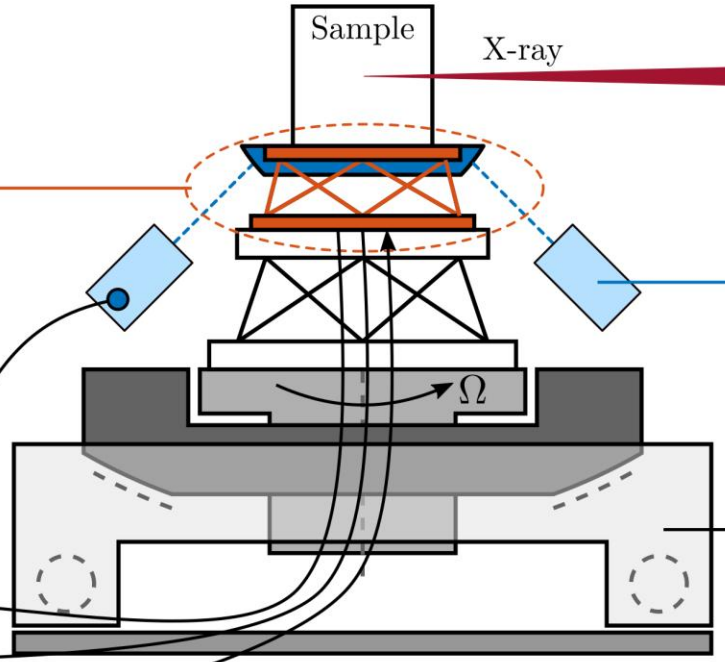
Nano Active Stabilization System (NASS) - Concept

Active Platform

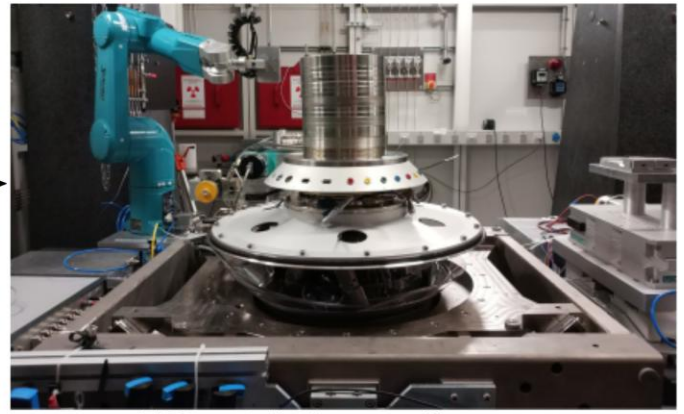


Control System and Instrumentation





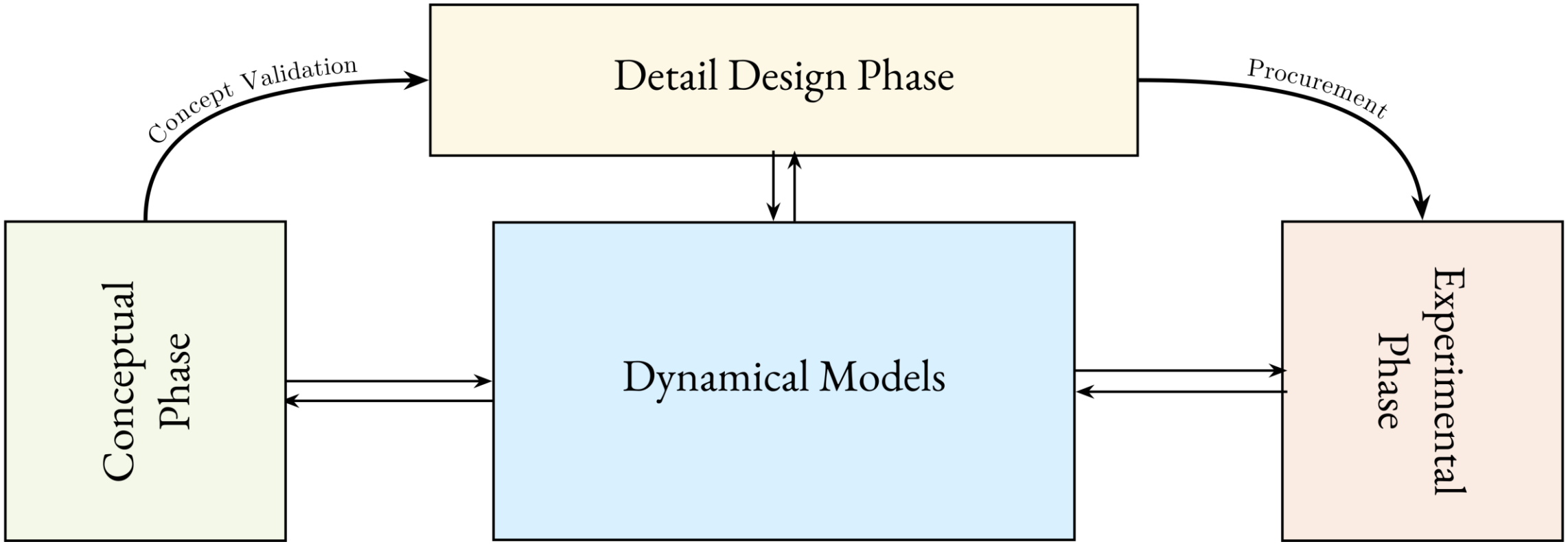
Online Metrology System



ID31 Micro-Station

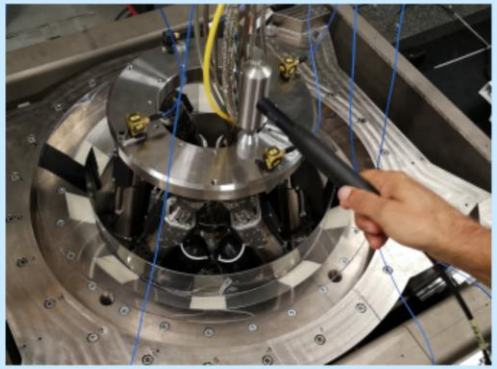
Goal: Keep the PoI of the sample on the beam (200nm x 100nm), as the micro-station is performing scans

Mechatronics Design Strategy

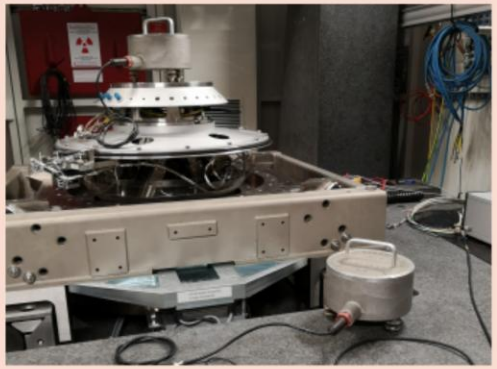


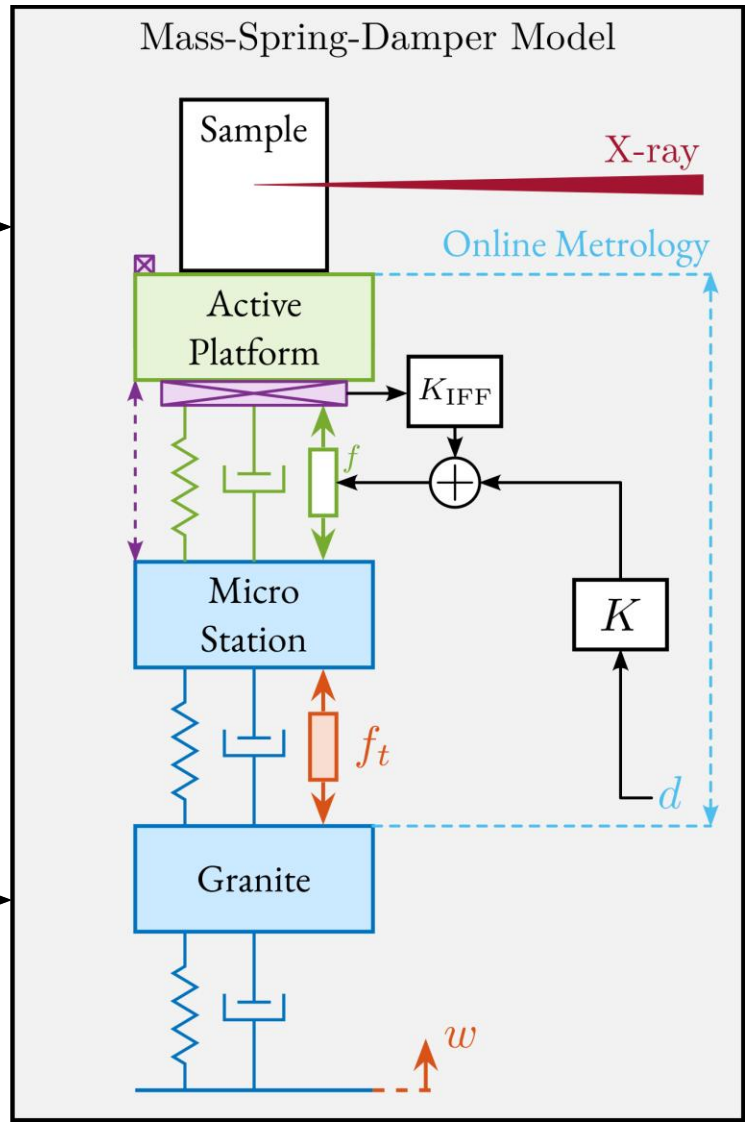
Uniaxial Model

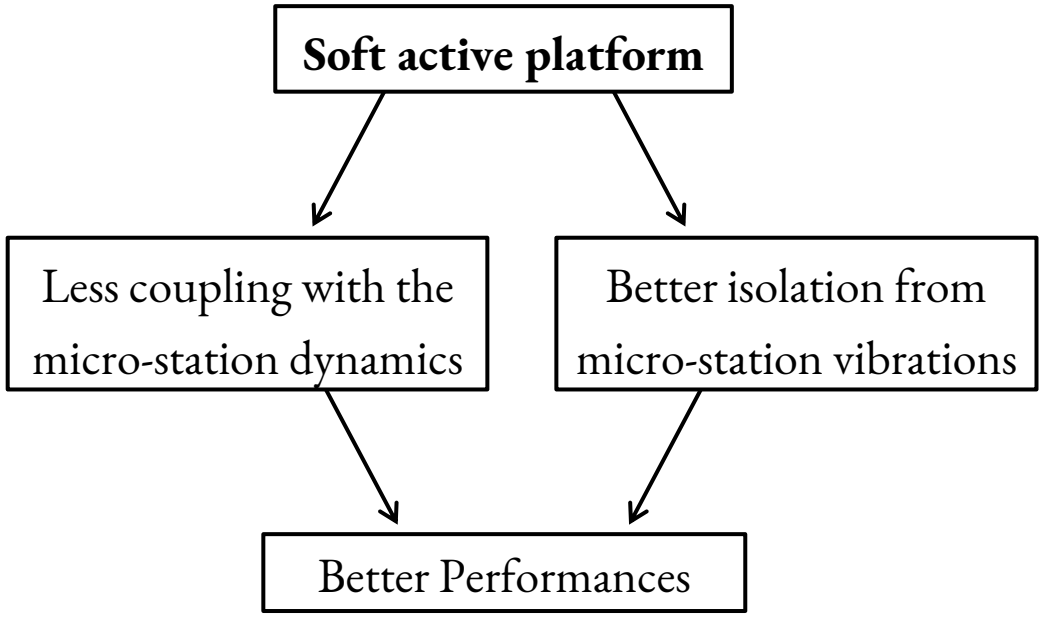
Dynamical Identification



Disturbance Measurements



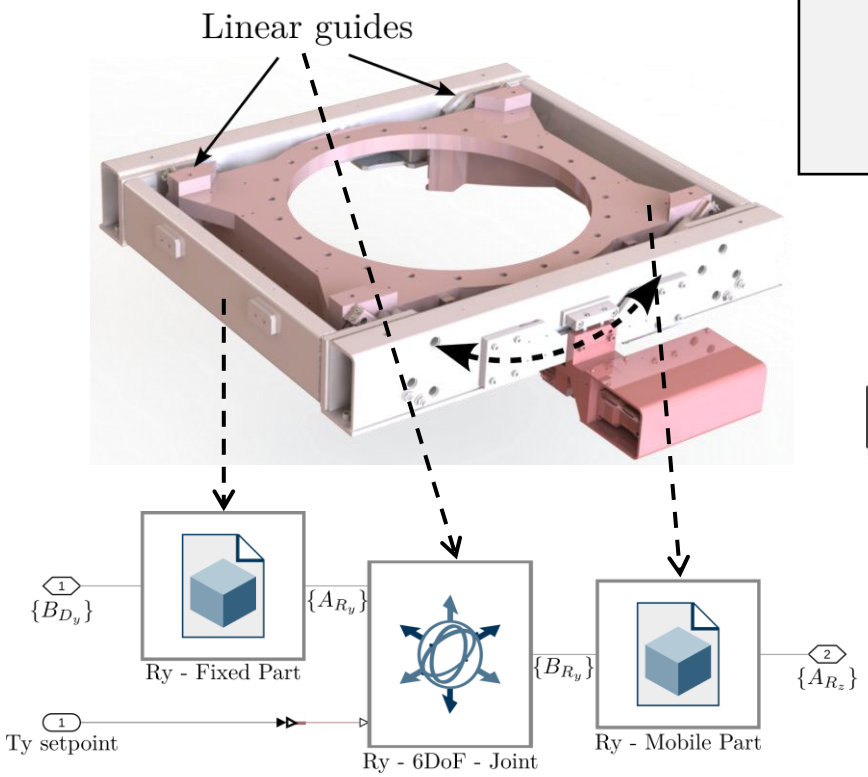


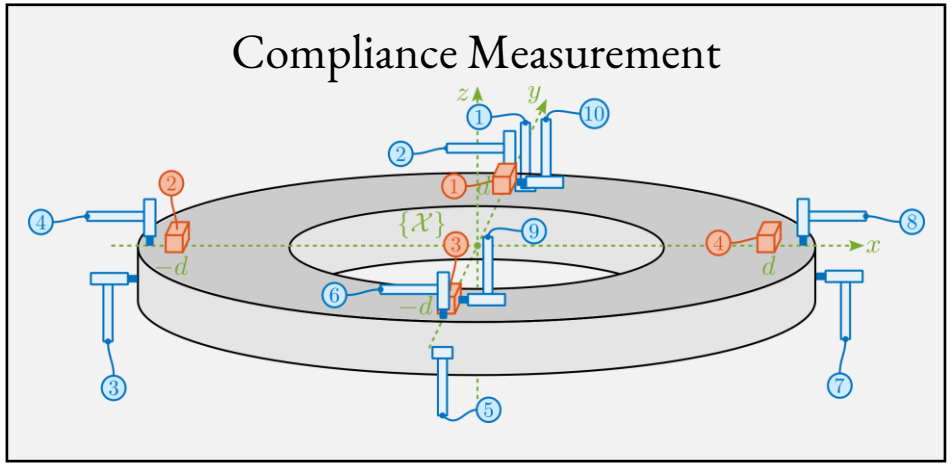


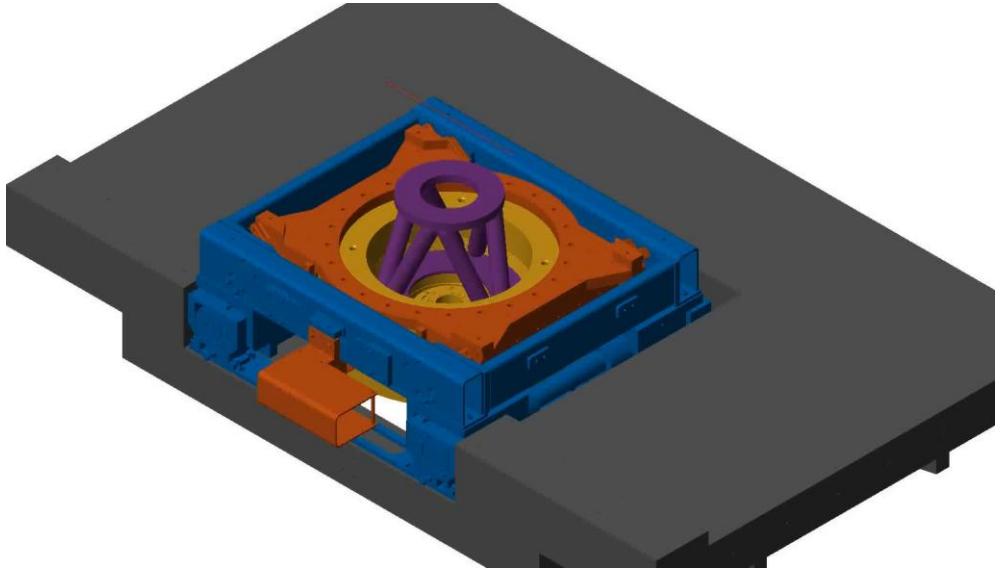
Micro-Station – Multi-Body Model

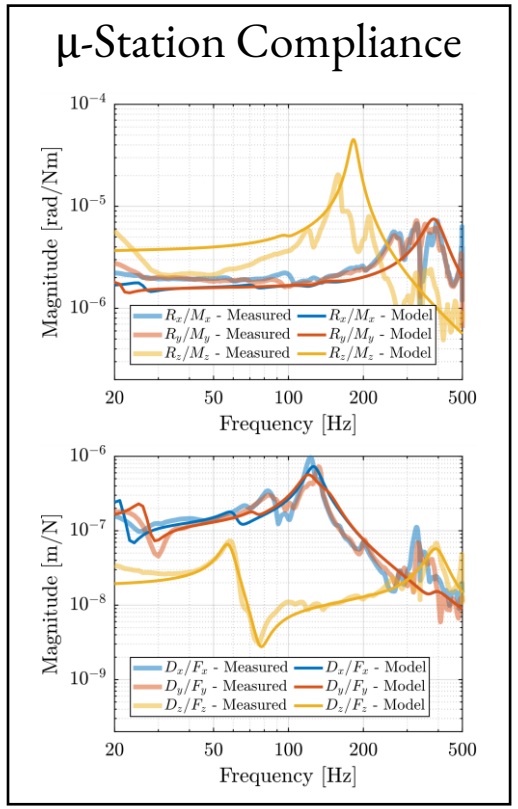
Multi-Body Model

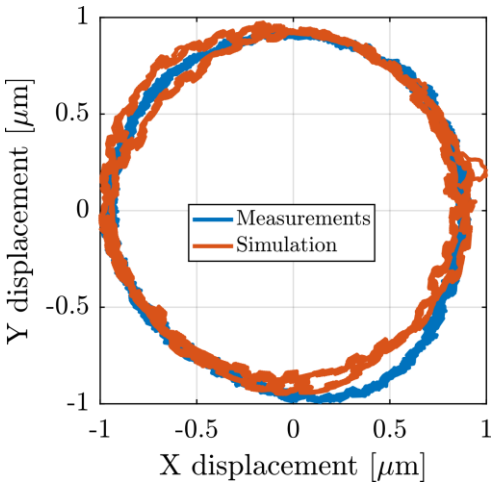
Solid bodies connected by springs and dampers







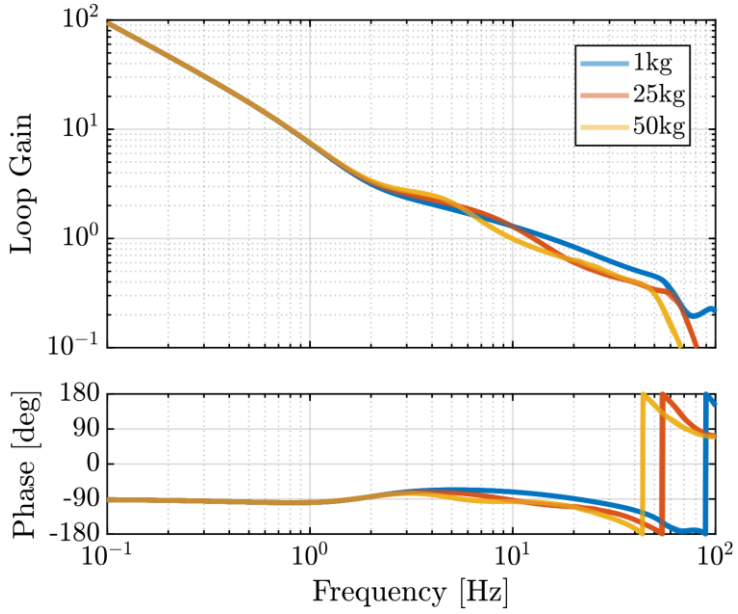


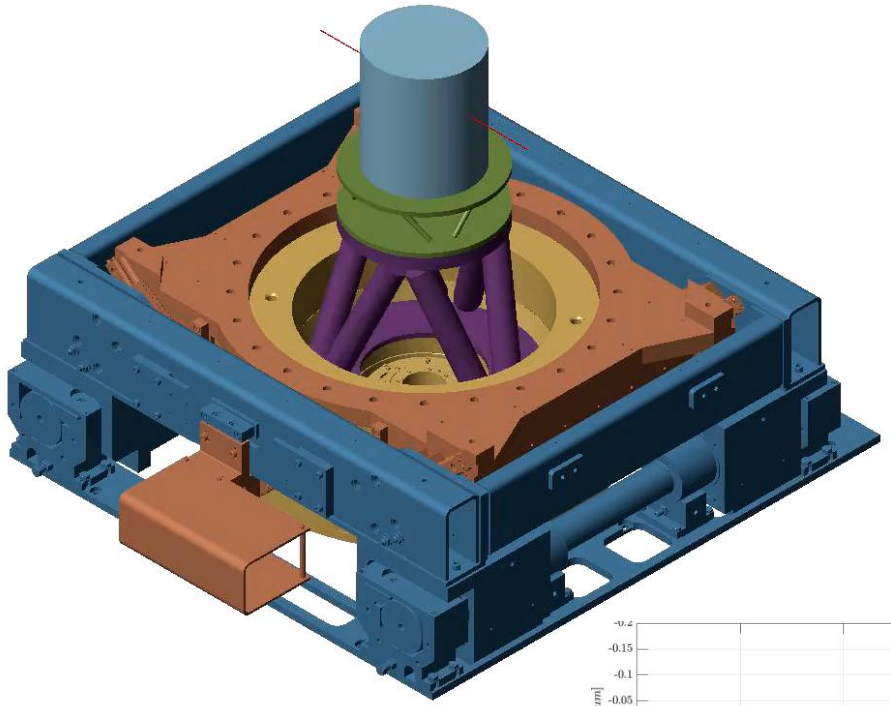


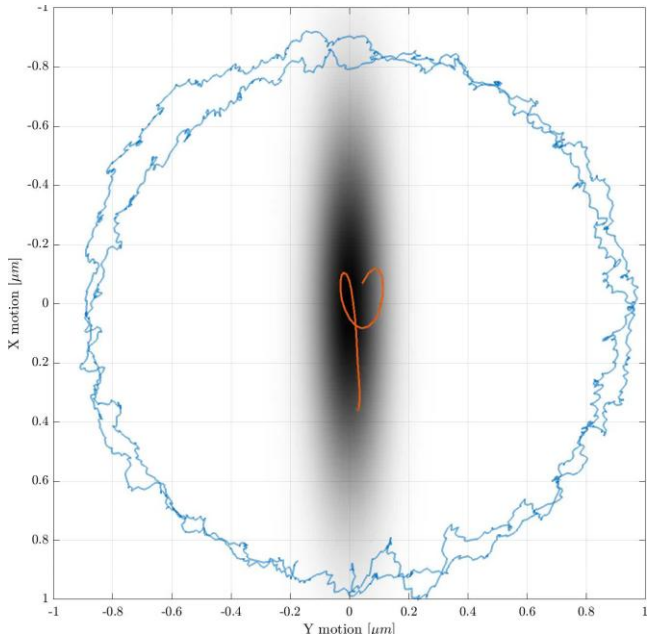
Validation of the Concept

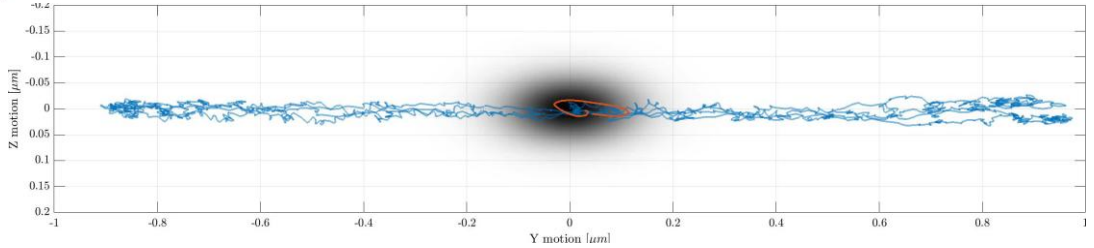
Integrator Lead LPF

$$K_{\mathrm{HAC}}(s) = g_0 \cdot \frac{\omega_c}{s} \cdot \frac{1}{\sqrt{\alpha}} \frac{1+\frac{s}{\omega_c/\sqrt{\alpha}}}{1+\frac{s}{\omega_c\sqrt{\alpha}}} \cdot \frac{1}{1+\frac{s}{\omega_0}}$$





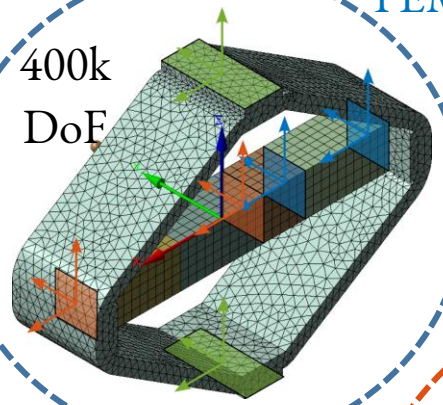




Component Optimization – Hybrid Modeling

FEM

400k DoF

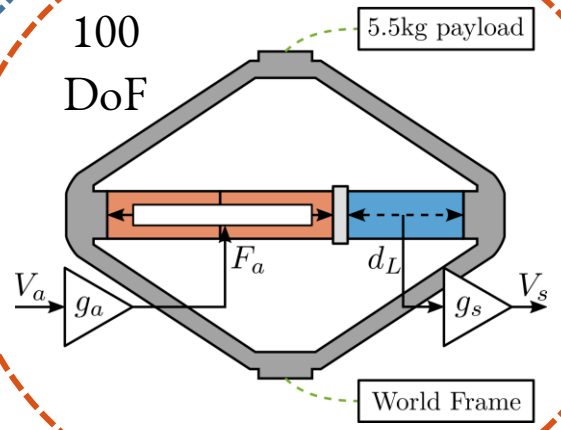


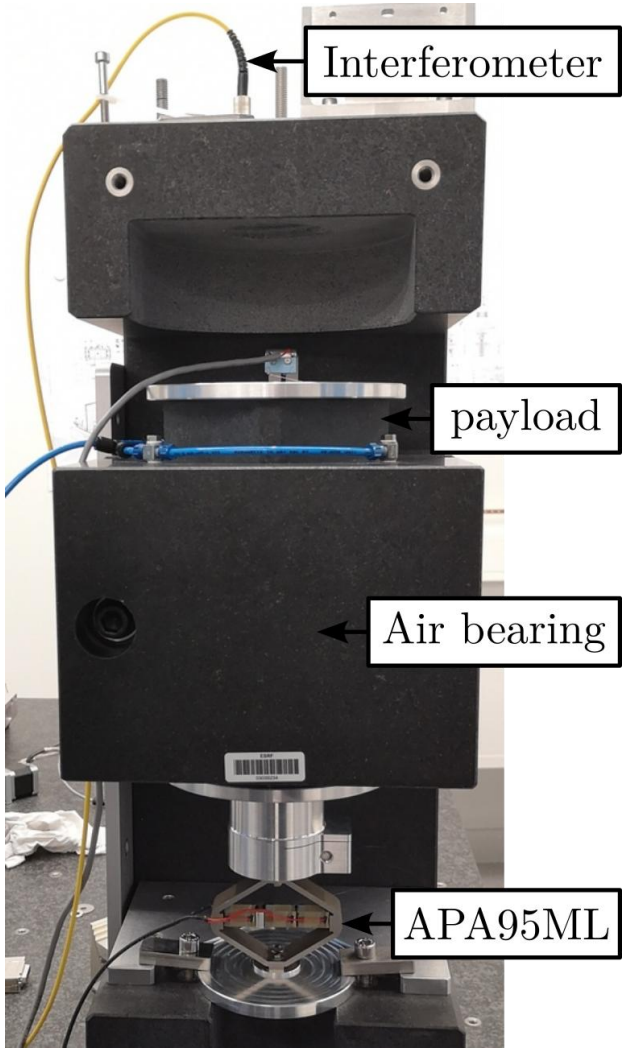
CMS

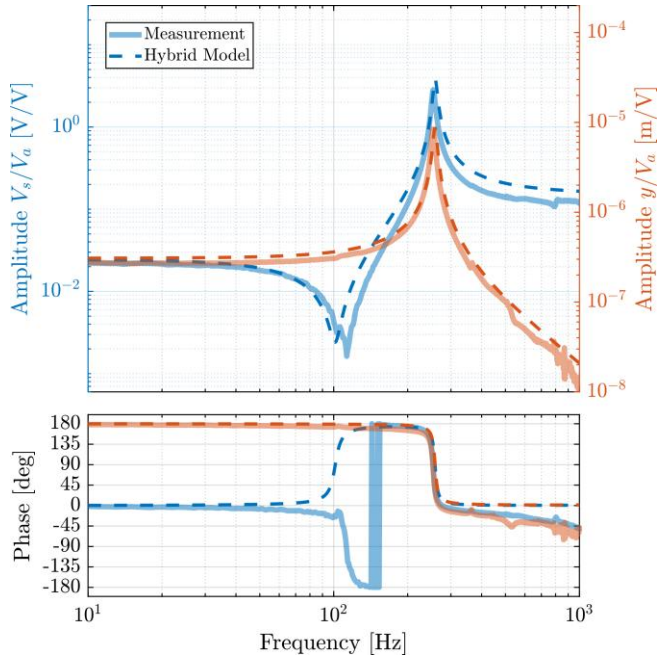
Reduced Order Flexible Body

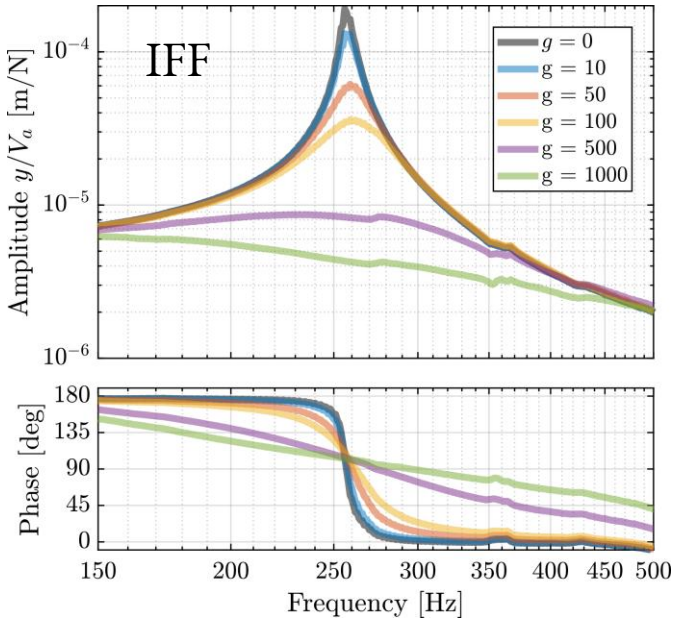
Multi-Body Model

100 DoF



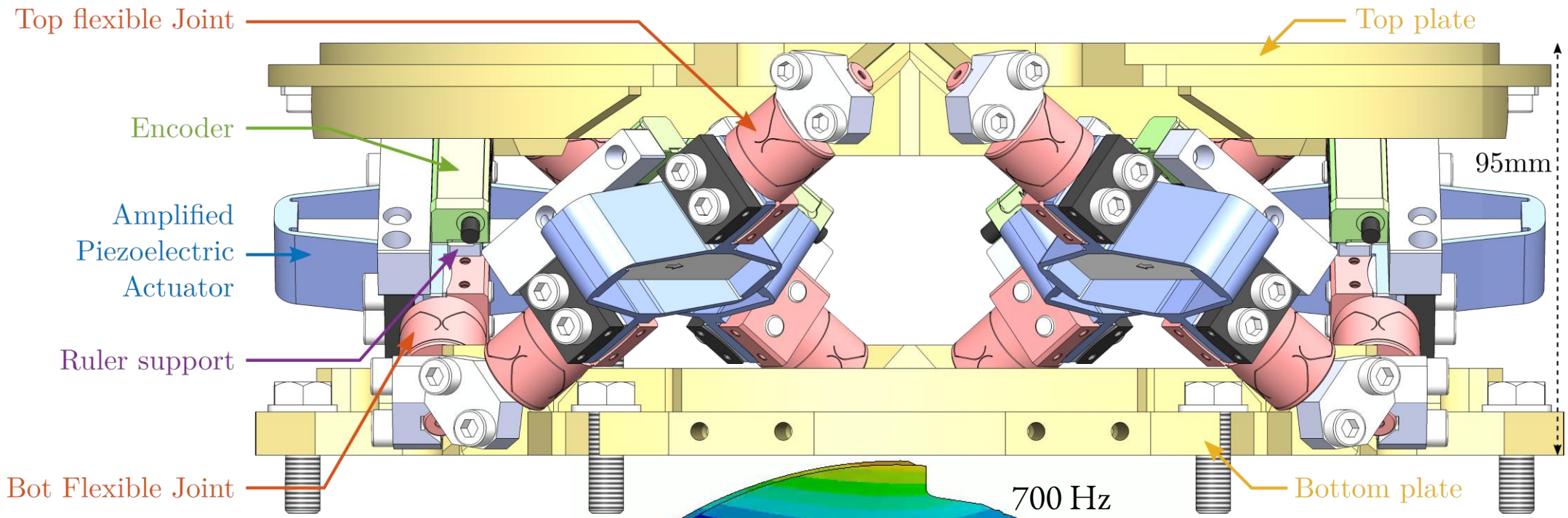


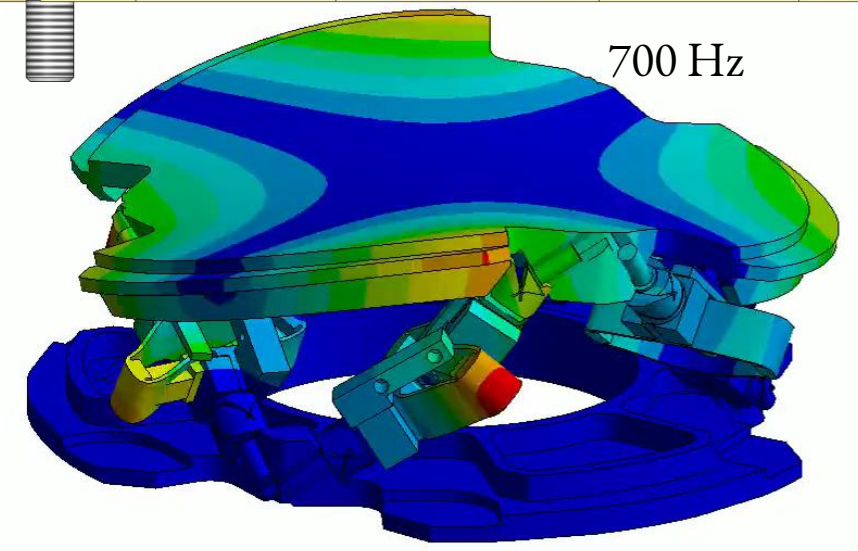




Courtesy: Philipp Brumund

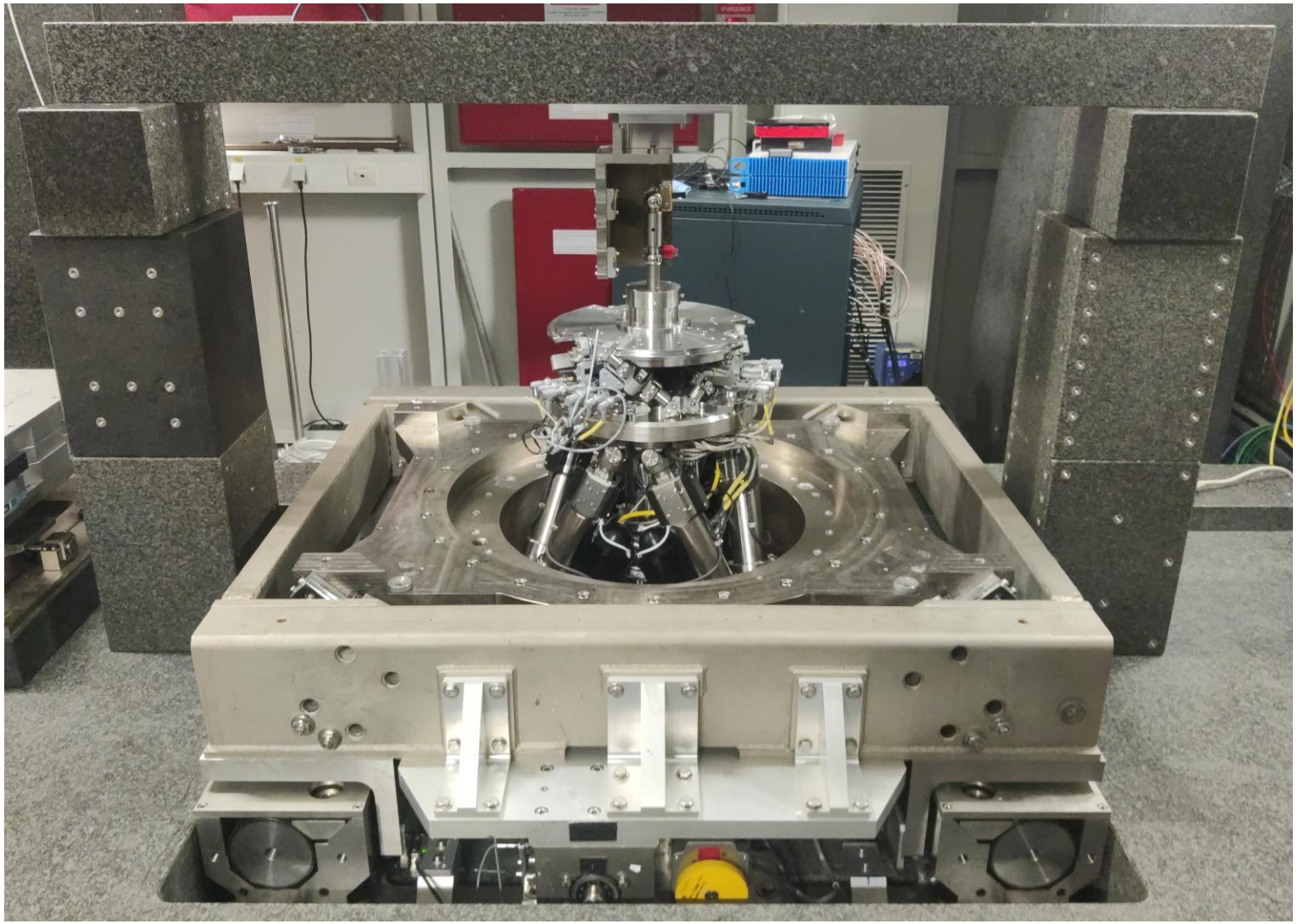
Obtained Design – The "Nano Hexapod"





Courtesy: Julien Bonnefoy
Damien Coulomb

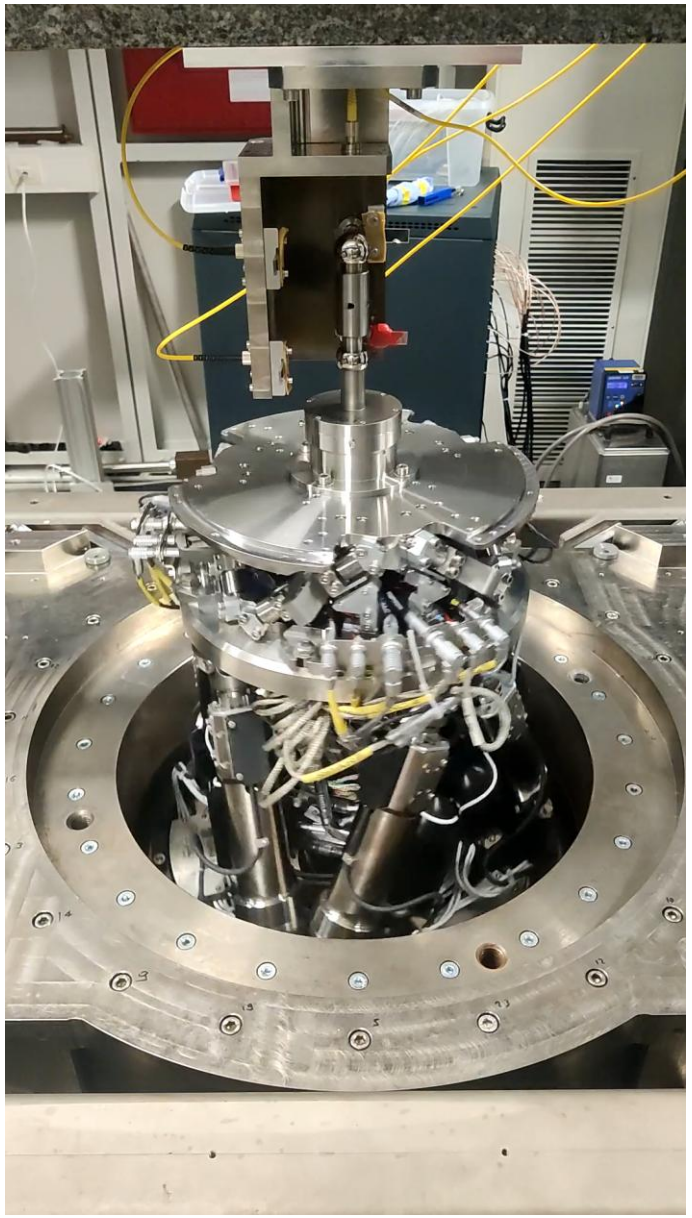
Nano Active Stabilization System – ID31

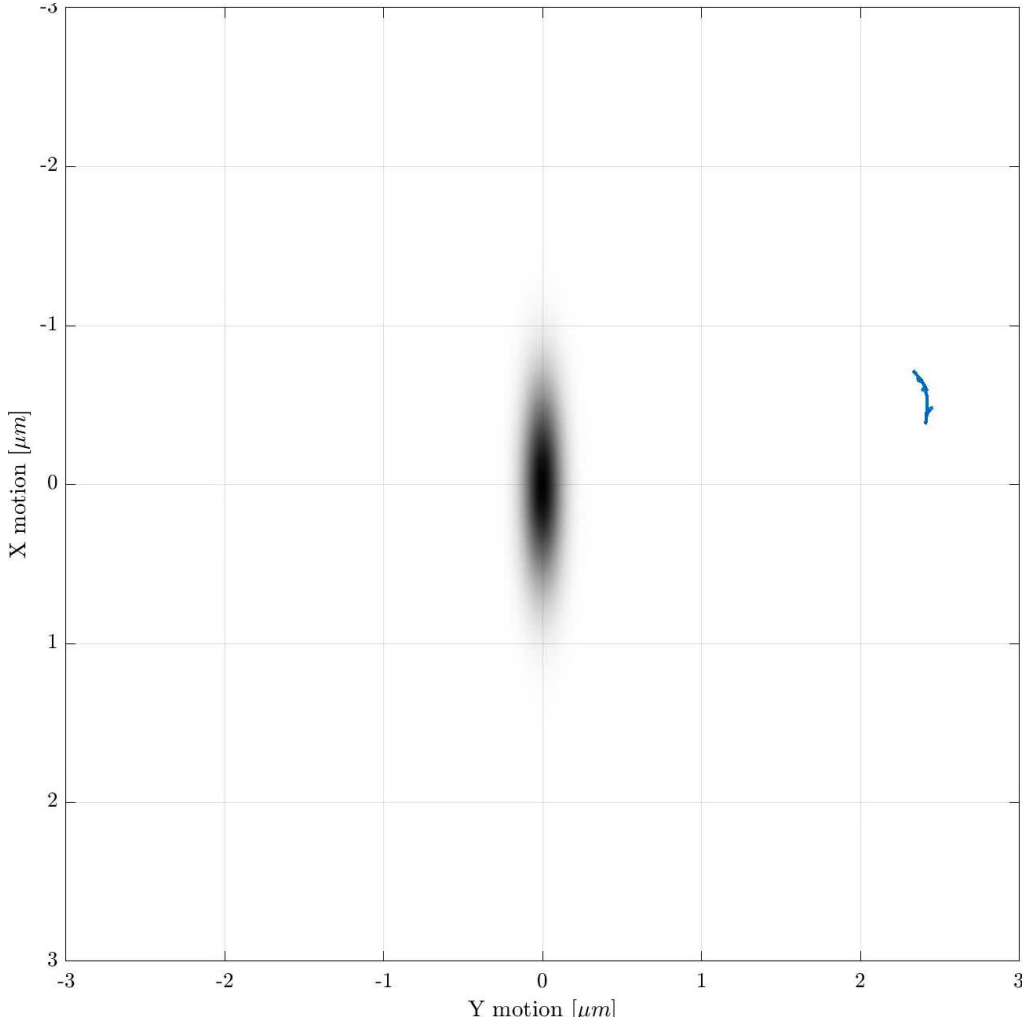


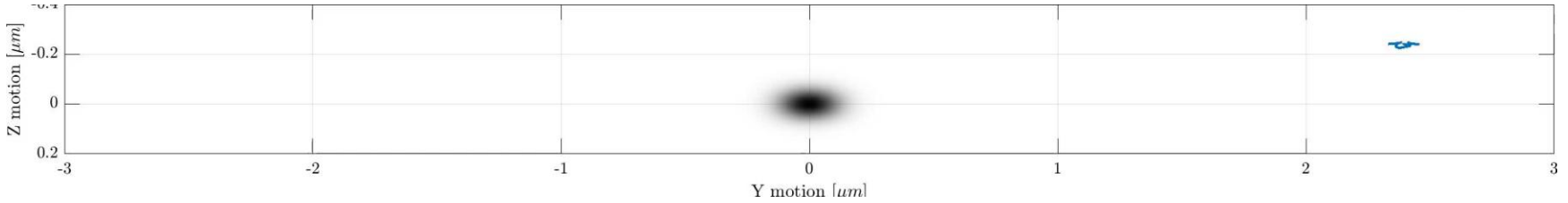
Speedgoat Performance Target Machine:

- 6 Analog Outputs
- 6 Analog Inputs
- 6 Quadrature Inputs
- 6 SSI Inputs
- Additional Digital I/O

Tomography Experiments







Outline

1/ Nano Active Stabilization System (10-15 minutes)

- Mechatronics Design Approach / Model Based Design
- Models and Simulations
- Real-Time control: Experimental Results

2/ ESRF XY(Z) Piezo Stage (10-15 minutes)

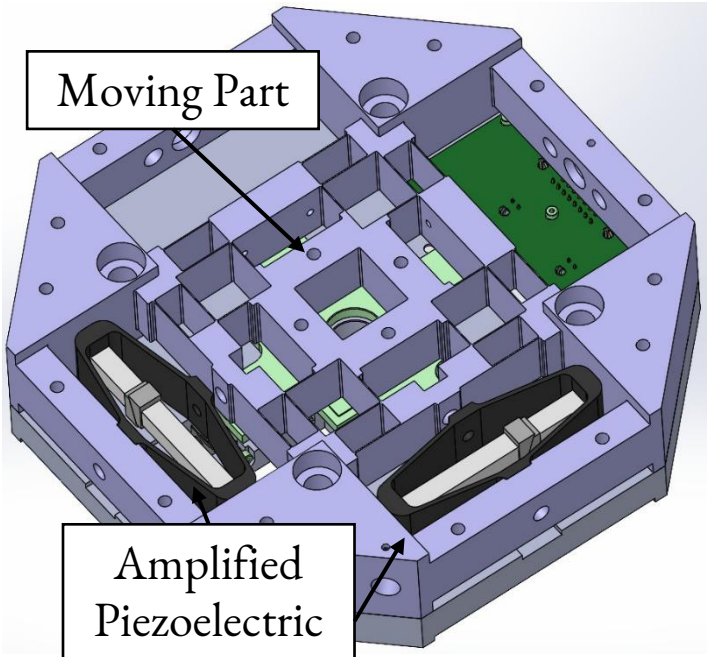
- Interfacing between Speedgoat and Python / Bliss
- Common Simulink / Python Library
- Experimental Results

XY(Z) Piezo Stage

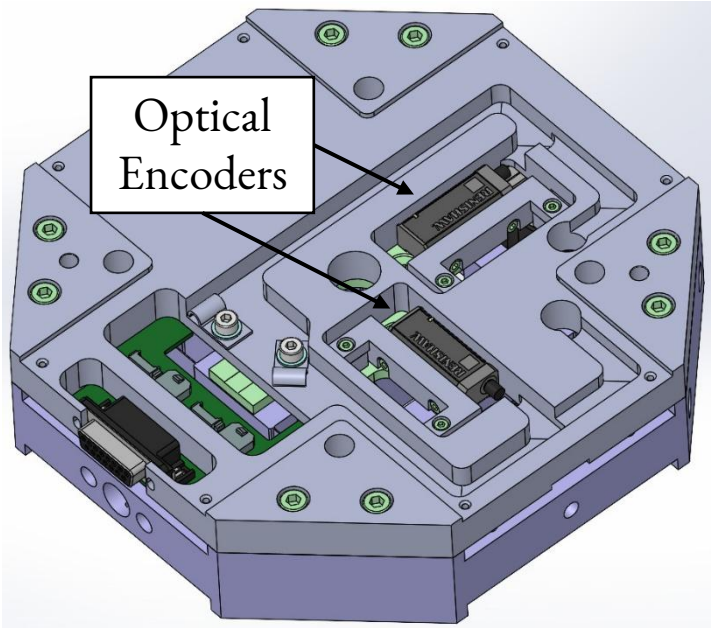
Specifications:

- Stroke XY : 300um
- Sample load : 5kg
- Aperture : 20x20mm
- Step by Step scans with 10 nm MIM
- Continuous scans

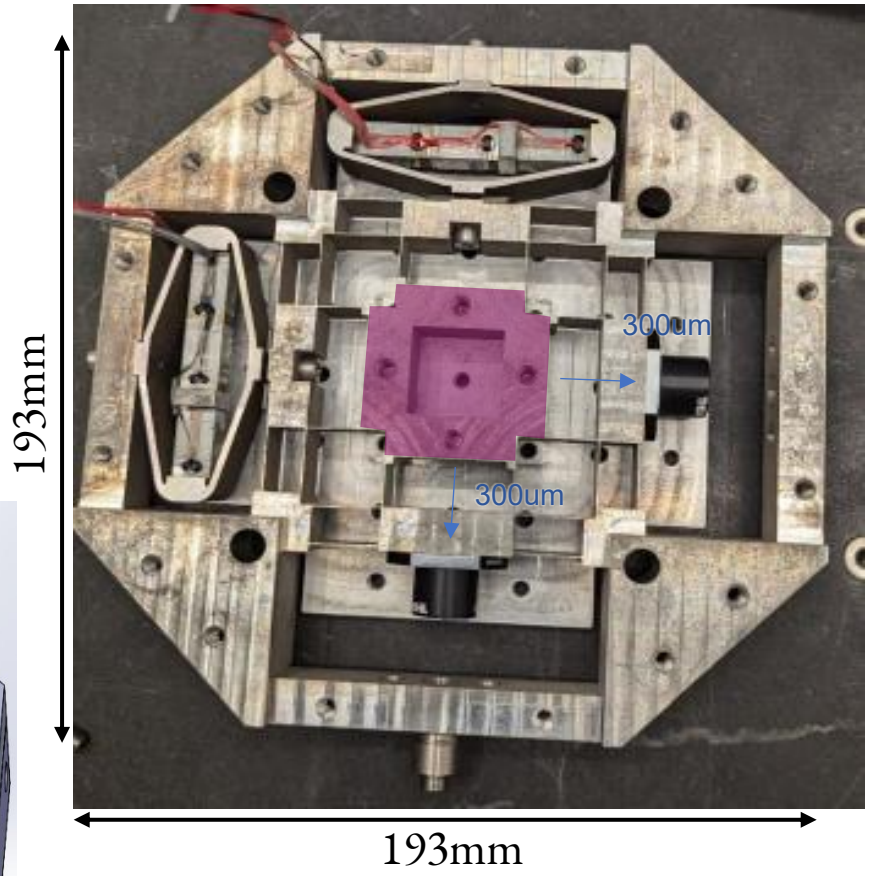
Top View



Bottom View

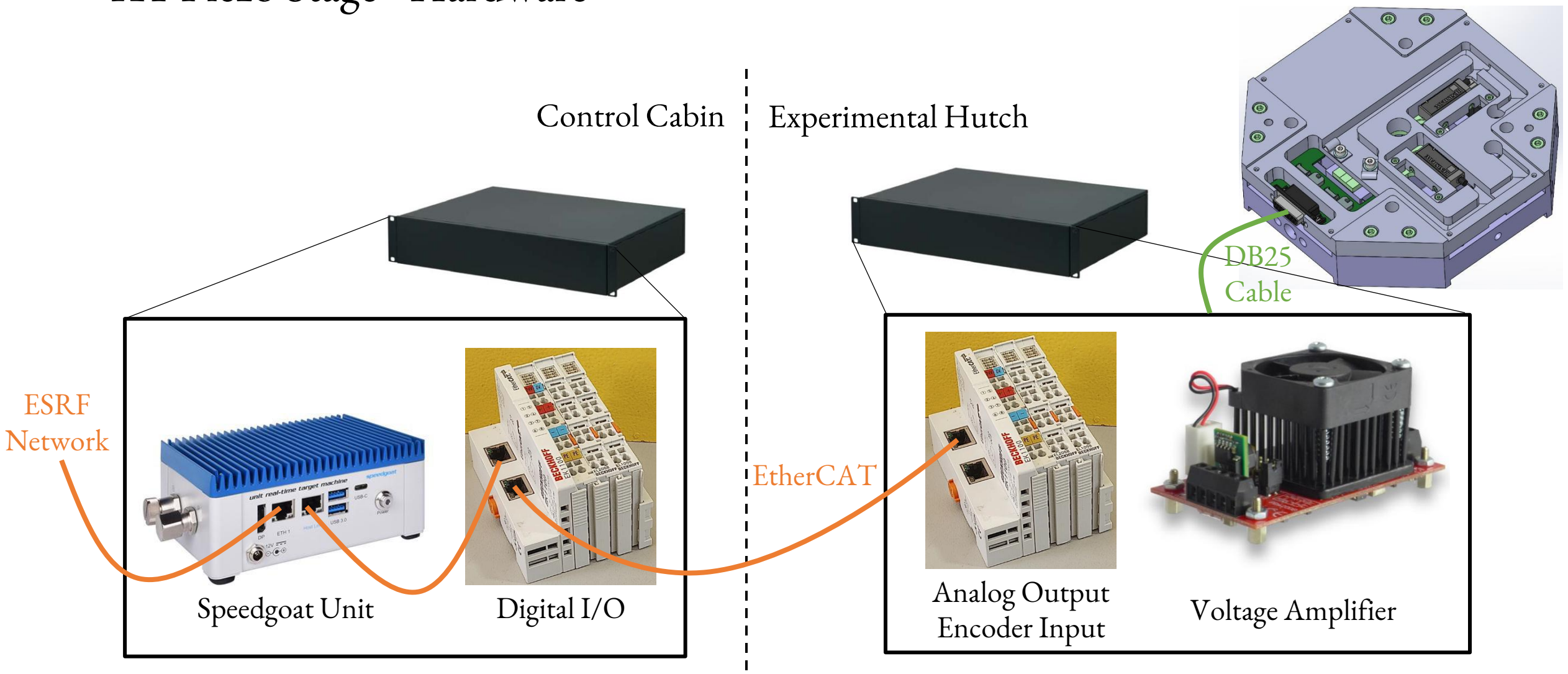


Picture of an early prototype



Courtesy: Anne-Lise Buisson

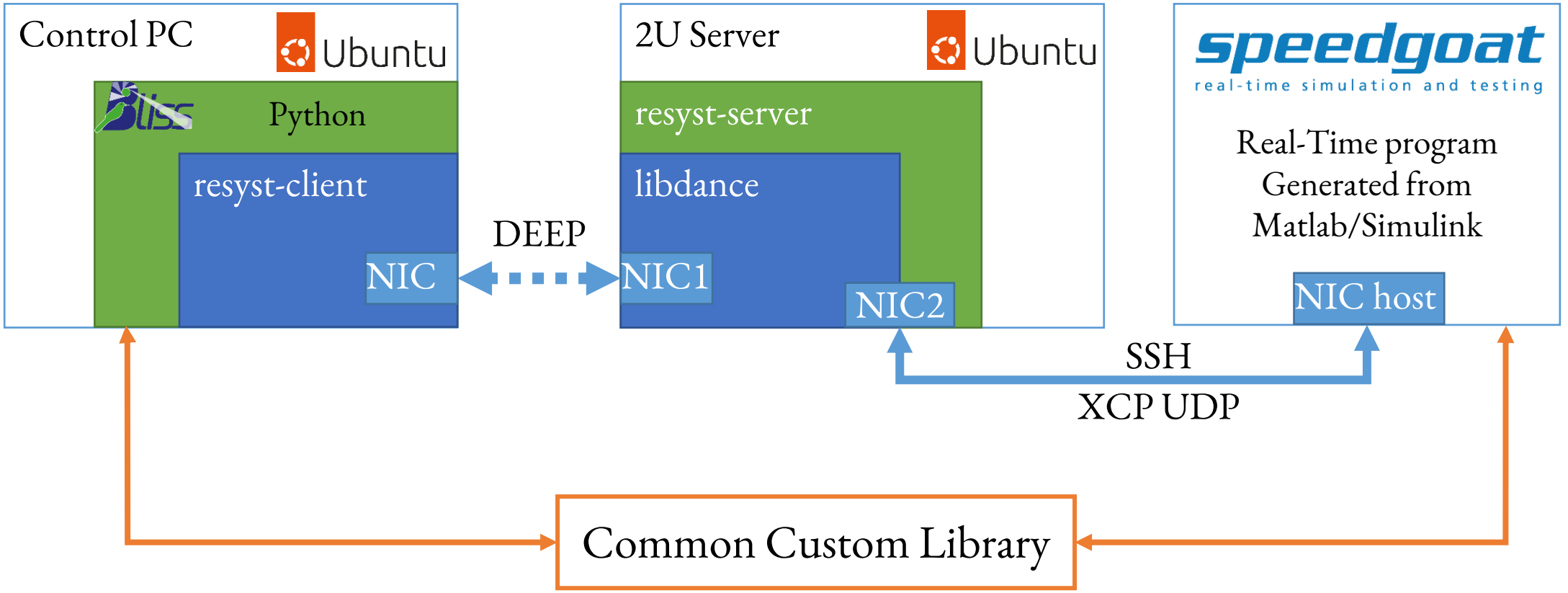
XY Piezo Stage - Hardware



Interface between Bliss (Python) and Speedgoat

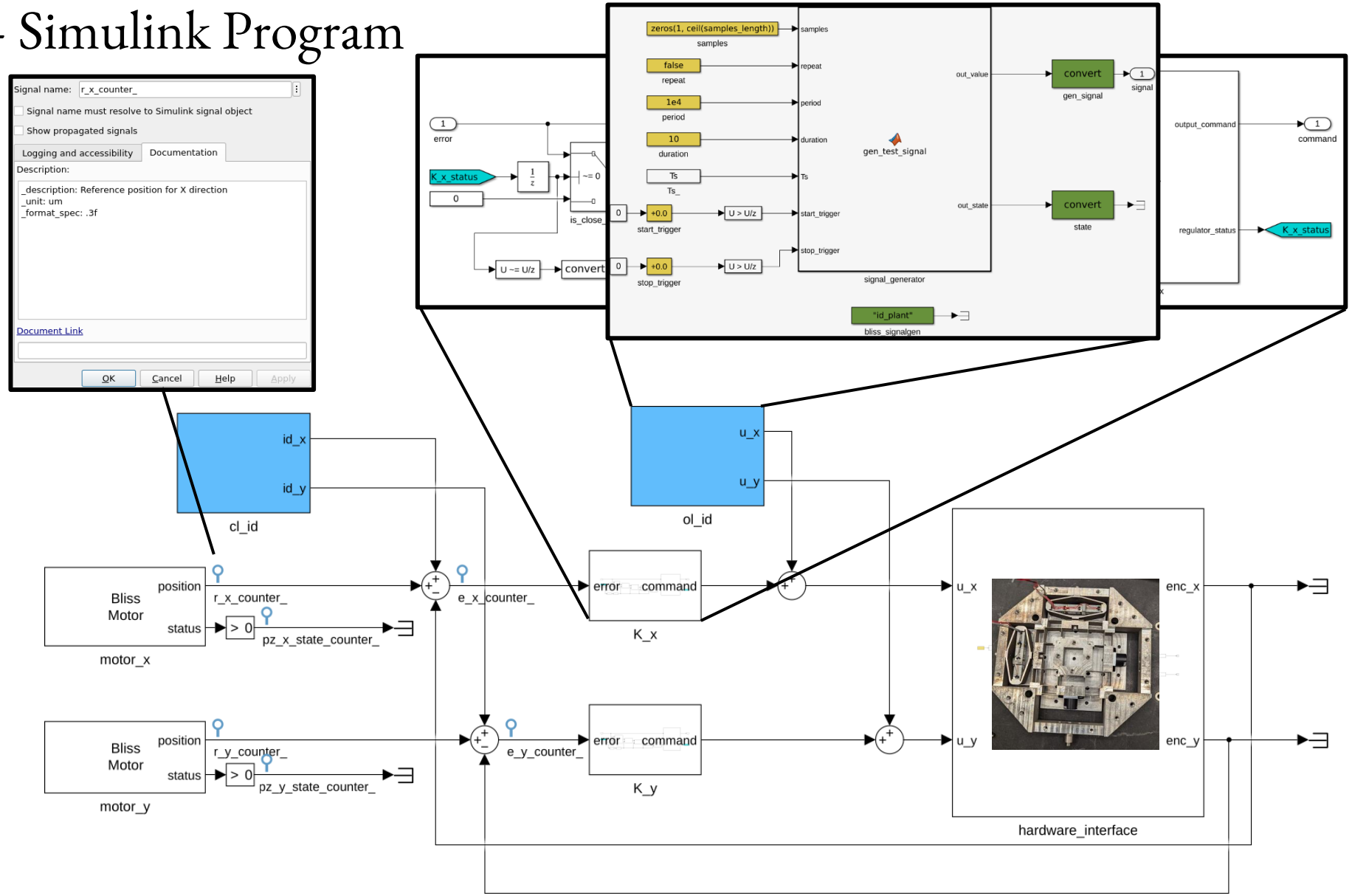
Bliss plugin based on **resyst-client**:
- Store/upload/run/stop programs
- Get and set parameters, acquire signals

Hybrid C/Python DAnCE **resyst-server**:
- Relay communication to Speedgoat,
- Implement UDP Rx in C.



Courtesy: Bruno Martin

XY Piezo Stage – Simulink Program

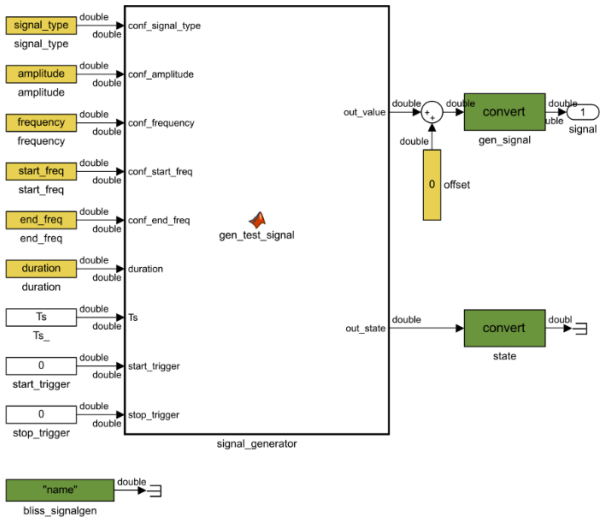


Simulink / Bliss – Common Library

Signal Generator Interface

Python

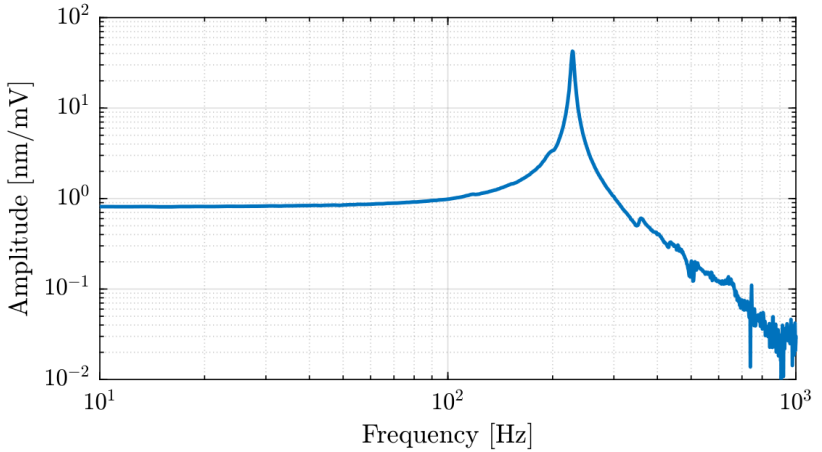
```
class SpeedgoatHdwGenerator:
    def __init__(self, speedgoat, unique_name):
        self._goat = speedgoat
        self._unique_name = unique_name
        self.name = self._speedgoat.parameter.get(
            f"{unique_name}/bliss_signalgen/String"
        )

    @property
    def amplitude(self):
        return self._goat.parameter.get(f"{self._unique_name}/amplitude")

    @amplitude.setter
    def amplitude(self, value):
        self.goat.parameter.set(f"{self._unique_name}/amplitude", value)

    def start(self):
        start_trigger = int(
            self.goat.parameter.get(f"{self._unique_name}/start_trigger")
        )
        self.goat.parameter.set(
            f"{self._unique_name}/start_trigger", start_trigger + 1
        )
```



System Identification Example

Python

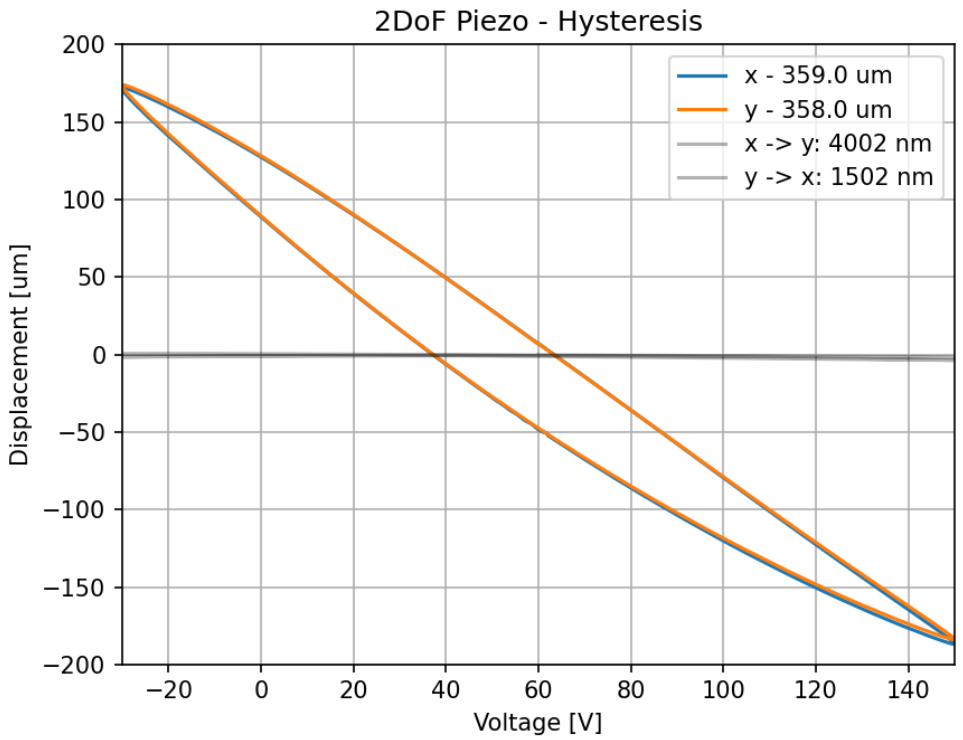
```
# Configure the signal generator
goat.generator.id_plant.type.Step
goat.generator.id_plant.amplitude = 1 # [V]
goat.generator.id_plant.duration = 10 # [s]

# Compute the transfer function
goat.utils.identification_tf( ...
    generator=goat.generator.id_plant,
    counter_in=goat.counter.du, # Voltage [V]
    counter_out=goat.counter.y) # Motion [m]
```



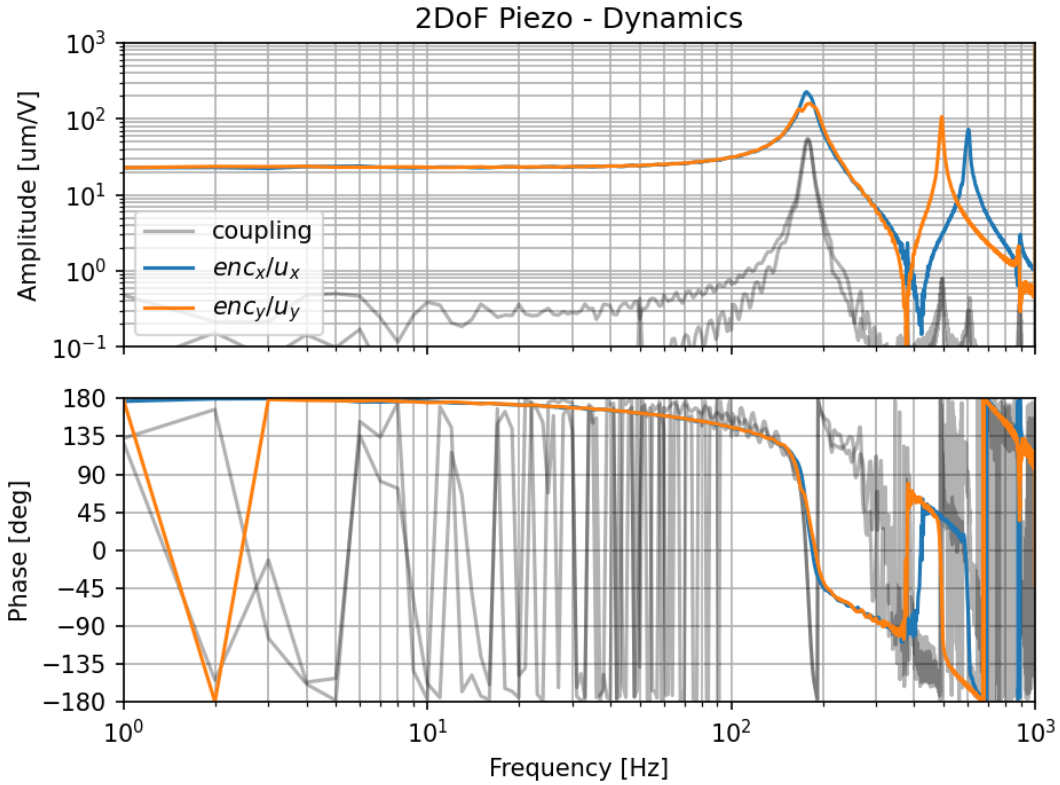
Using this common library allows for very fast system setup and high flexibility

XY Piezo Stage – Experimental Results – System Identification

1/ Verify everything working as expected

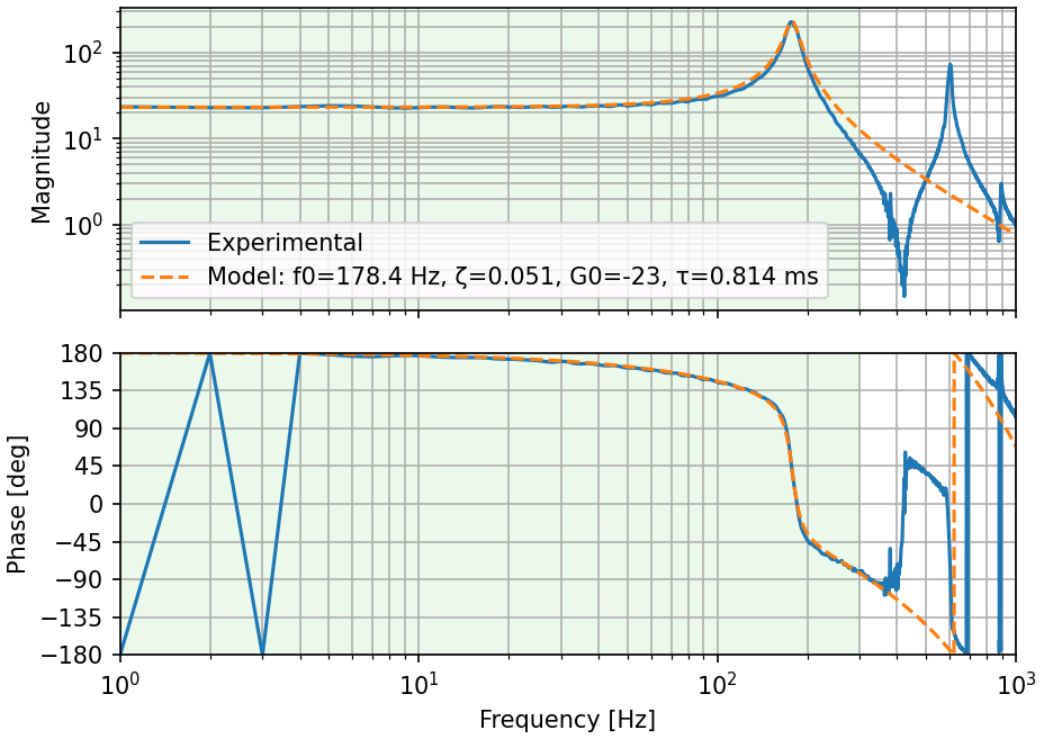


2/ Identify the system Dynamics

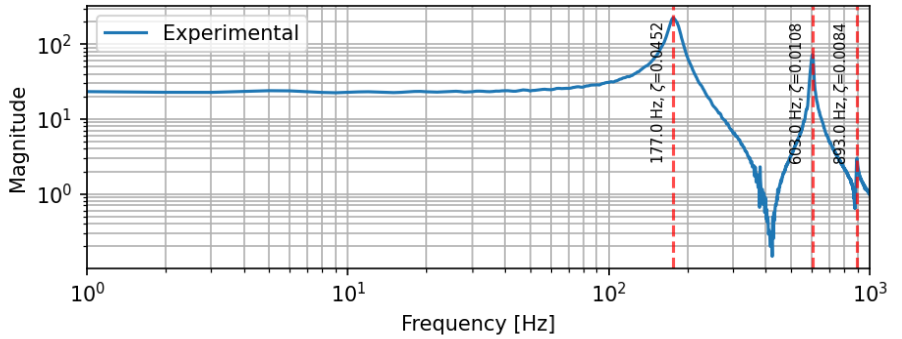


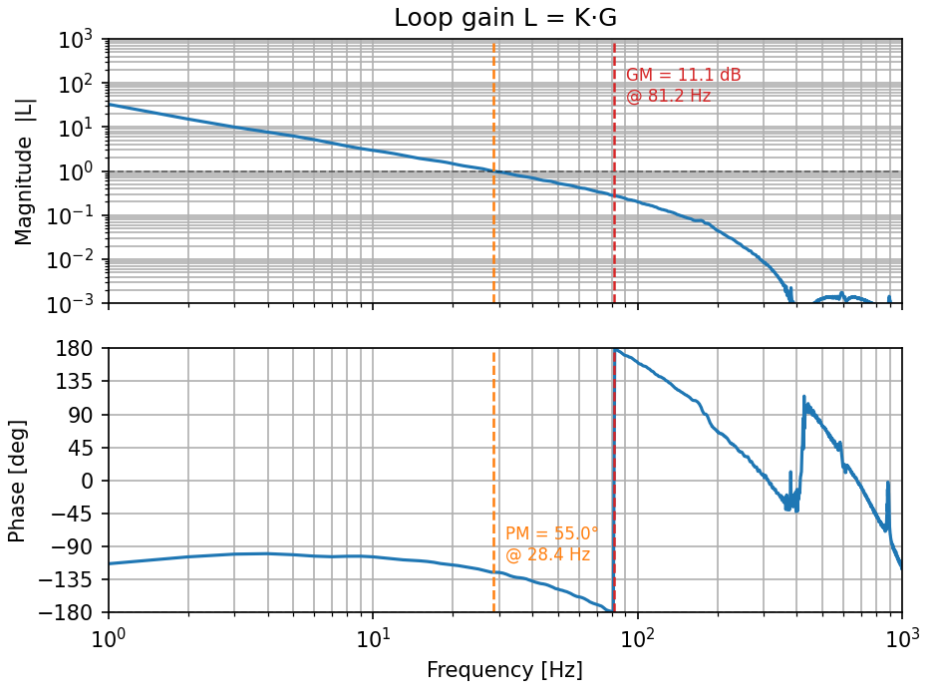
XY Piezo Stage – Experimental Results – Feedback Control

3/ Extract a plant model



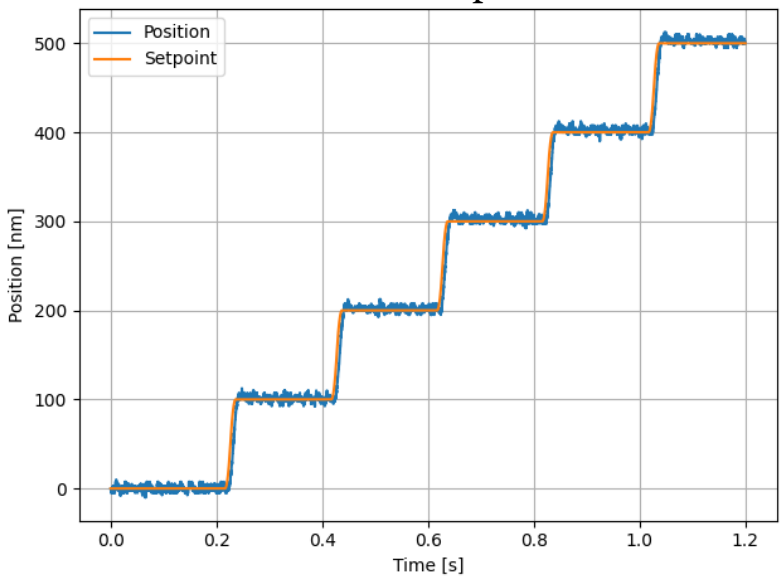
4/ Tune the controller (Custom Auto-Tune)



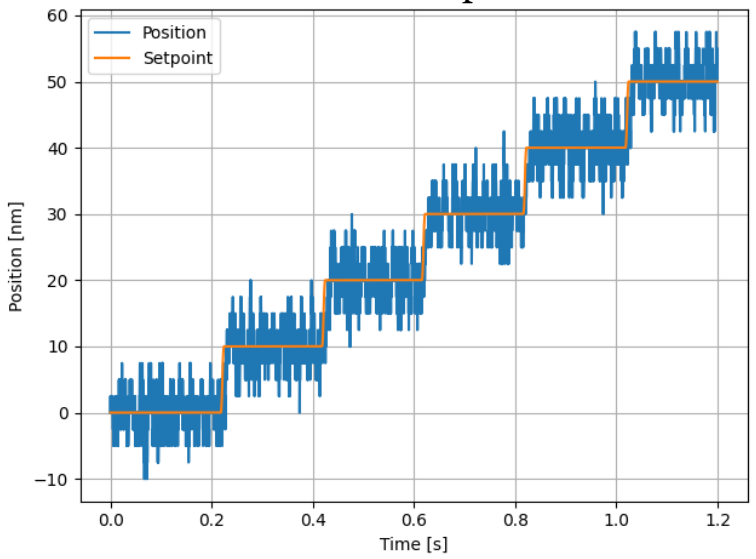


XY Piezo Stage – Experimental Results – Scans

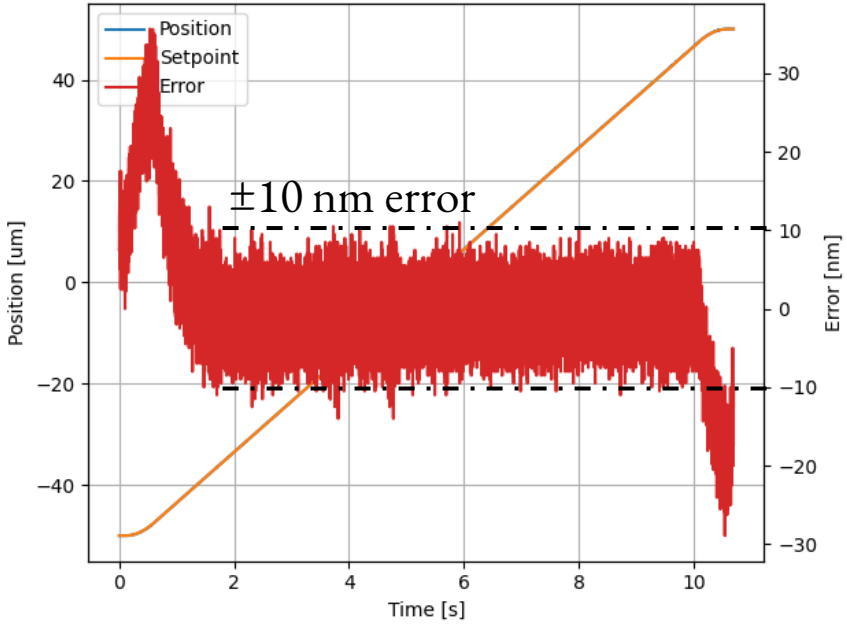
100nm Steps



10nm Steps



10 μm/s constant velocity scan



Conclusion – Key Takeaways

- Matlab/Simulink covers the full development cycle: design, simulation, and real-time implementation

- Simscape enables accurate system modeling (multi-body, hybrid with FEM)

- Shared Simulink/Bliss library → fast setup, high reusability across beamlines. Provides a flexible and robust real time control architecture

Outlook: real-time control and mechatronics design will play a growing role at the ESRF !