

A photograph showing an astronaut in a white spacesuit working on a large, white, flexible robotic arm in space. The arm is part of a larger structure, possibly a space station or satellite. The background is the Earth's blue and white atmosphere against the blackness of space.

# Modeling and Model-Based Control Design/Simulation of Flexible Space Robots using MATLAB™/Simulink™

**Speaker: Valentin Pascu  
with: H. Garnier, A. Janot, J.-P. Noël**

**MATLAB EXPO France  
Paris – May 30, 2017**

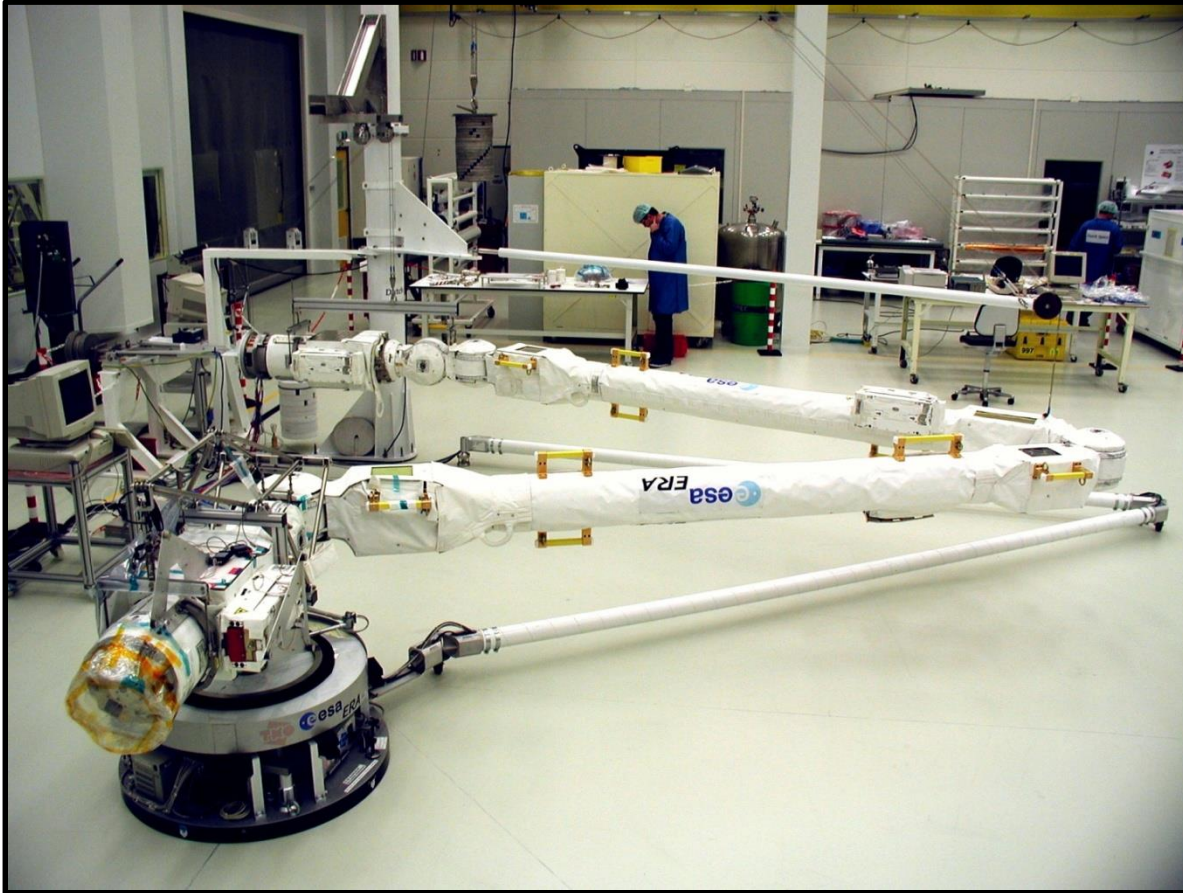
## Key Points

- A. **MATLAB** package: powerful simulation tool for showcasing R&D engineering challenges for complex mechanical and aerospace systems
- B. Robot position controls in two easy steps:
  - 1. feedback linearization using **MATLAB/Symbolic Math Toolbox™**
  - 2. tracking control design with **MATLAB/Control System Toolbox™** e.g. with the **PID Tuner App™**
- C. Rigid/flexible robot motion simulation/visualization: easy with **Simulink™** and with **Simscape Multibody™**
- D. Accessible, affordable simulations-based experimentation for data-driven modeling, plus some existing numerical tools (e.g. **MATLAB/System Identification Toolbox™**)

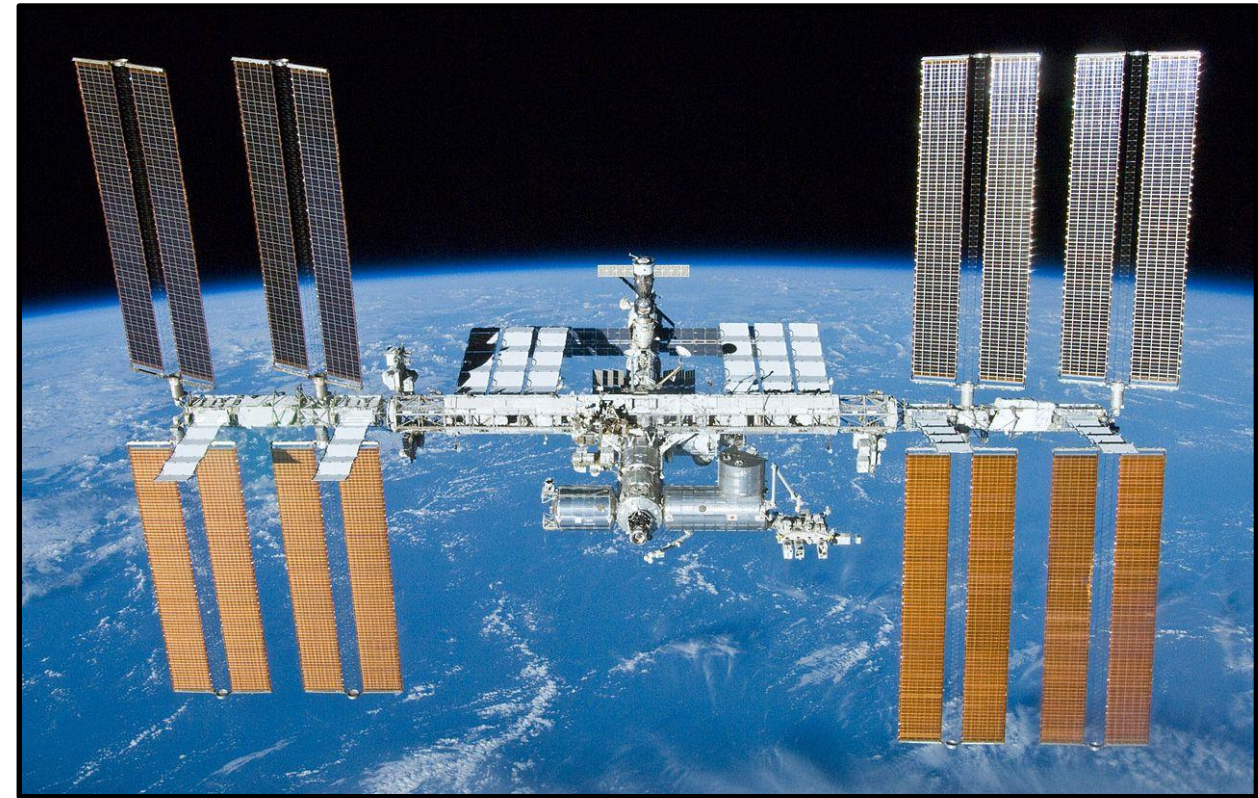
**Considerable reduction of time in assessing research-relevant problems!**



# Space Robot Manipulators and Large Satellites: What do they have in common?



The European Robotic Arm during ground testing at the European Space Agency in Noordwijk, The Netherlands



The International Space Station during orbital operation



# Space Robot Manipulator Controls: Multidisciplinary Research

System Identification



Hugues Garnier

System Identification  
for Robotics



Alexandre Janot

Control Engineering



Valentin Pascu

System Identification of  
Aerospace Structures



Jean-Philippe Noël



# The European Robotic Arm (ERA): Main Characteristics and Specifications



**Total length (unloaded): 11.3 m**

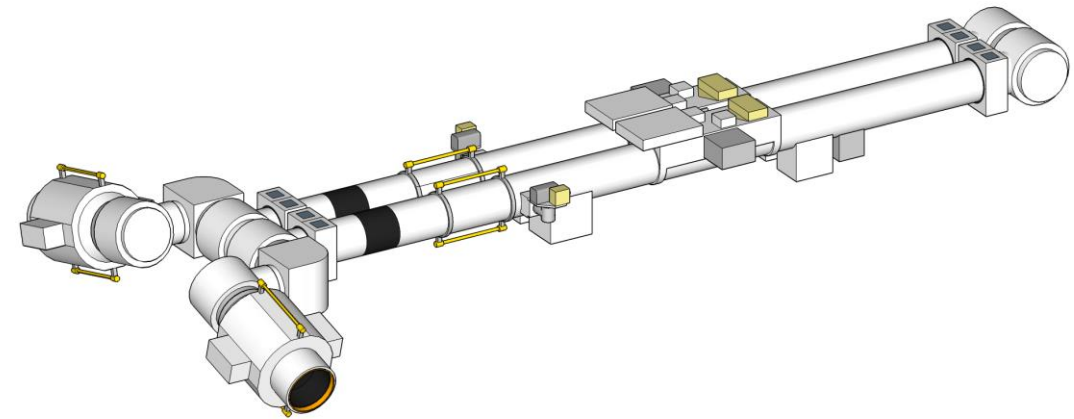
**Degrees of freedom: 7**

**Total mass (unloaded): 630 kg**

**Maximum load dimensions: 3x3x8.1 m**

**Maximum moveable mass: 8000 kg**

**Positioning accuracy (closed-loop): 5 mm**



**Most time-consuming space robotic manipulator design project to date!**



# Space Robot Manipulators: How do they work and what do they do?

Robot motion control implies a certain designer workflow:

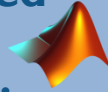
1. Desired position  $(x,y,z)$  of end-effector
2. Computed trajectory  $(\theta)$  for robot joints
3. Trajectory tracking with robot actuators  $(\tau)$



closed-loop control



model-based  
design  
and simulation



**R1. Models of robot dynamics**  
(attitude, structure, actuators)

**R2. Measurement priors**  
(location, SNR, power spectra)

experimental verification

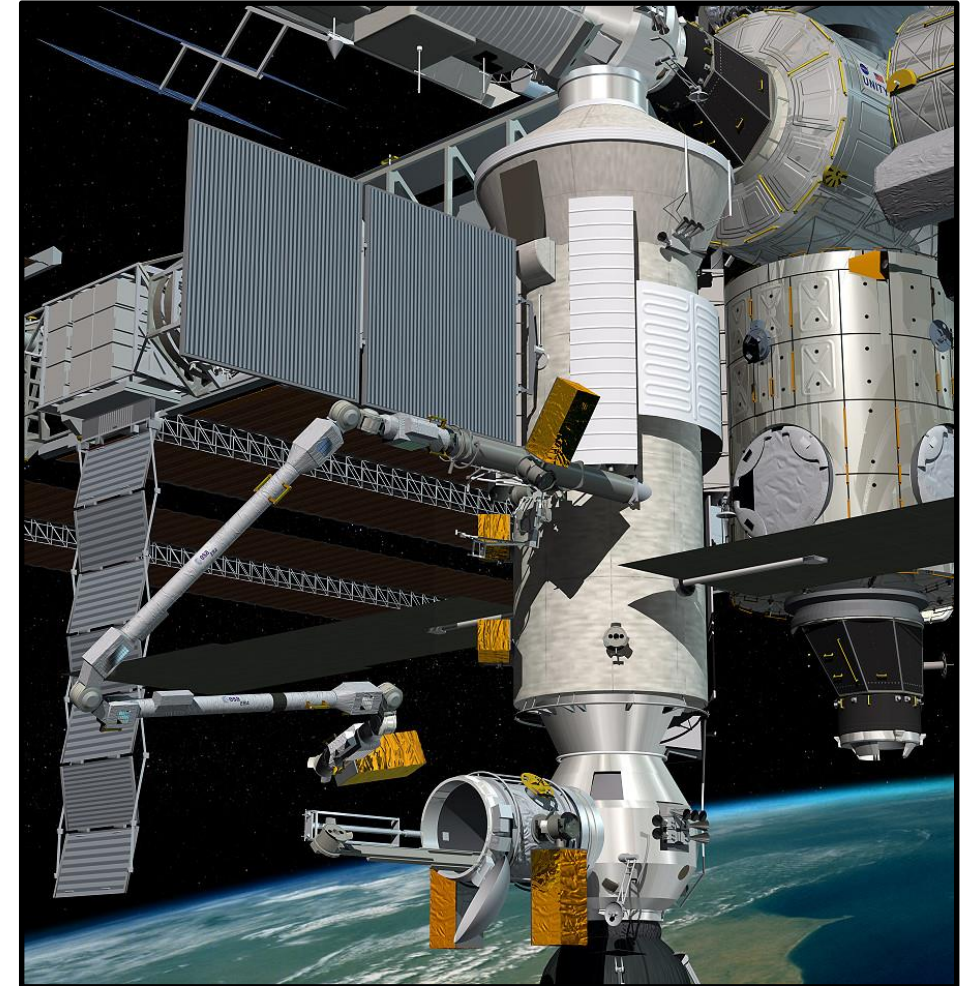


experimental validation



experimental modelling

**Reduce experimental effort through model-based analysis!**



Concept snapshot of the ERA during operation  
(courtesy of DLR)

# Feedback Linearization of Space Robot Dynamics: Basic Theory

Euler-Lagrange equation for *space robot dynamics*:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} = \tau$$

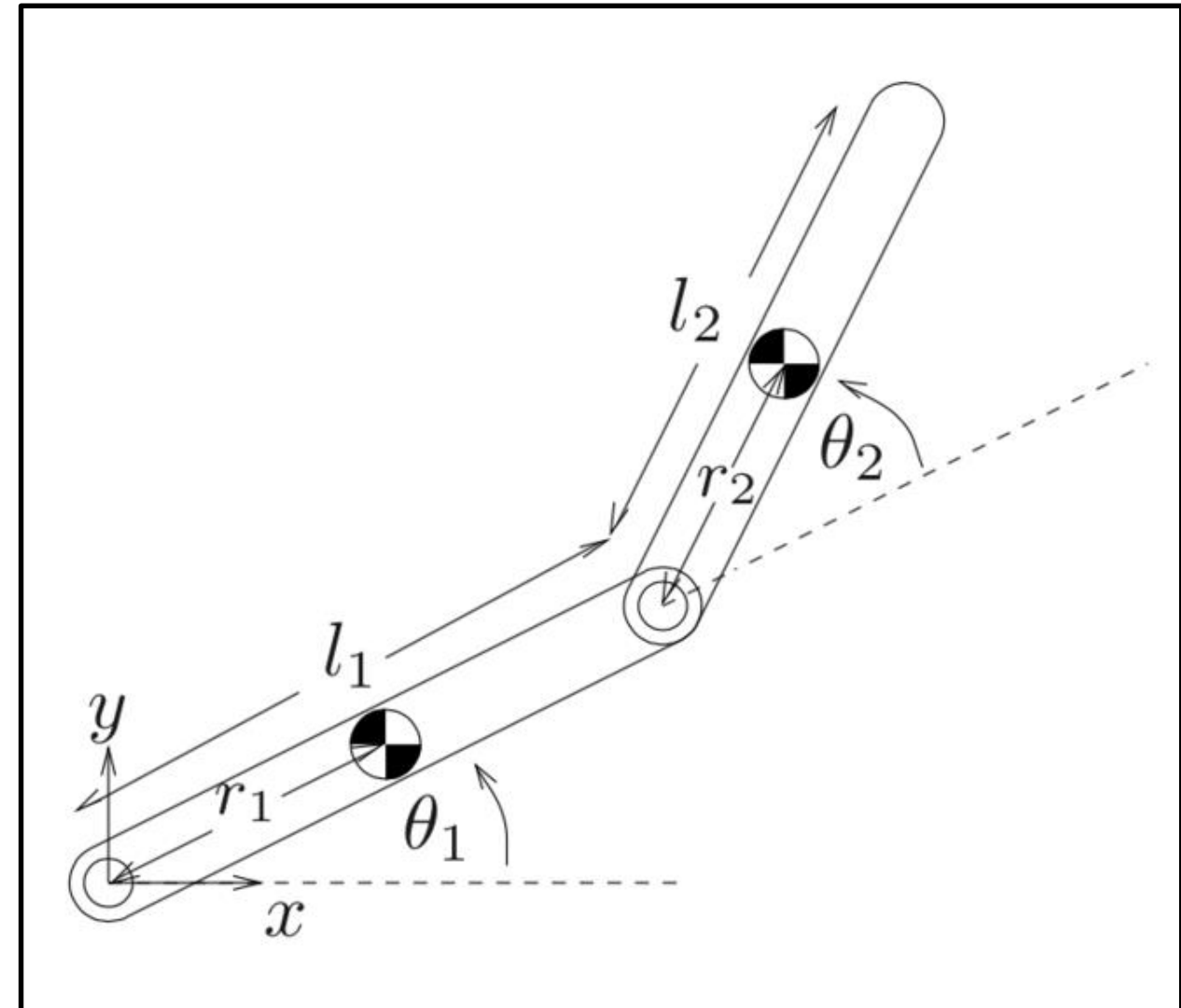
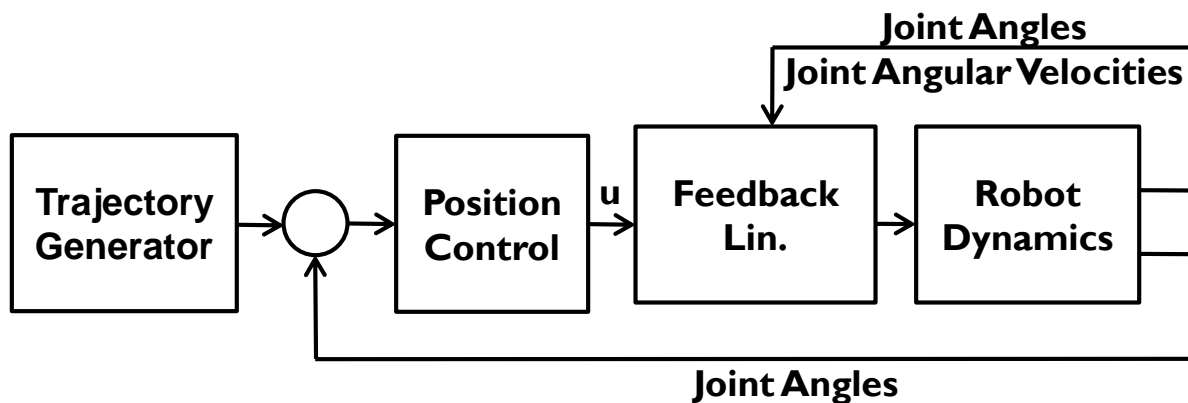
State-space representation (coupled, nonlinear):

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ -M^{-1}(\theta)C(\theta, \dot{\theta})\dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}(\theta) \end{bmatrix} \tau$$

Feedback linearization law:

$$\tau = M(\theta)u + C(\theta, \dot{\theta})\dot{\theta}$$

for inner-loop linearization/dynamic decoupling.



# Feedback Linearization of Robot Dynamics using Symbolic Calculations

The screenshot displays the MATLAB Editor interface with a script titled 'Demo1.m'. The script contains the following code:

```

13 % Symbolic variables for generalized coordinates and their derivatives
14 syms q1 q2 dq1 dq2 tau1 tau2;
15 assume(q1, 'real'); assume(q2, 'real'); assume(dq1, 'real'); assume(dq2, 'real'); assume(tau1, 'real'); assume(tau2, 'real')
16 % Symbolic variables for robot parameters
17 syms m1 l1 lc1 l1 m2 l2 lc2 I2
18 % Computation of robot dynamics in Euler-Lagrange form
19 Jvc1=[-lc1*sin(q1) 0; lc1*cos(q1) 0; 0 0];
20 Jvc2=[-l1*sin(q1)-lc2*sin(q1+q2) -lc2*sin(q1+q2); l1*cos(q1)+lc2*cos(q1+q2) lc2*cos(q1+q2); 0 0];
21 D=m1*transpose(Jvc1)*Jvc1+m2*transpose(Jvc2)*Jvc2+[I1+I2 I2; I2 I2];
22 % Symbolic computation of centrifugal and Coriolis forces matrix (C)
23 h=-m2*l1*lc2*sin(q2);
24 C=[h*dq2 h*dq2+h*dq1; -h*dq1 0];
25 % Computation of nonlinear robot dynamics in state-space equation form
26 f=[dq1; dq2; -inv(D)*C*[dq1; dq2]]; g=[0 0; 0 0; inv(D)];
27 % Representation of linearizing control law in required form
28 aux=C*[dq1; dq2];
29 syms u1 u2;
30 assume(u1, 'real'); assume(u2, 'real');
31 u=[u1; u2];
32 aux2=D*u;
33 % Computation of feedback interconnection
34 f cl=f+g*aux;

```

The script is annotated with three blue callouts:

- 1. definition of real symbolic variables**: Points to lines 14-15, where symbolic variables are defined and assumed to be real.
- 2. definition of robot dynamics**: Points to lines 19-26, where the Jacobian matrices, inertia matrix, and nonlinear dynamics are computed.
- 3. symbolic feedback linearization**: Points to lines 27-34, where the linearizing control law and feedback interconnection are defined.

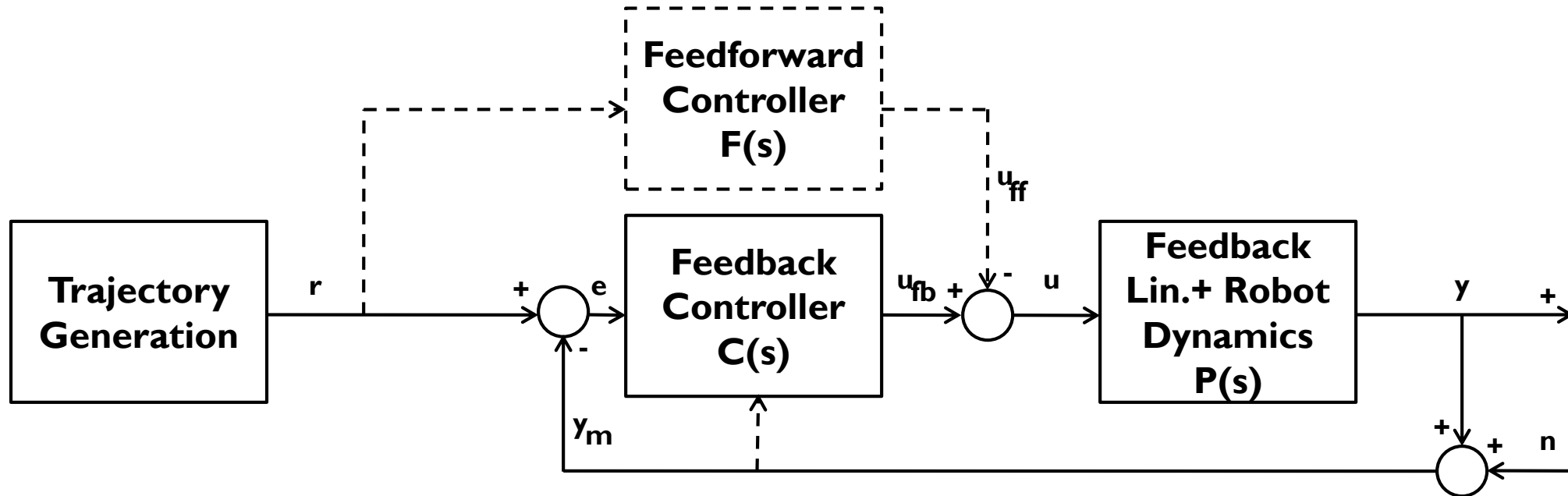
The workspace on the right shows the following variables:

- bending\_stiff\_y\_
- bending\_stiff\_y\_
- damping\_link1
- damping\_link2
- dq1
- dq2
- dtheta\_simscap
- I1
- I2
- Ix
- Ixy
- Ixz
- Iy
- Iyz
- Iz
- I
- L
- I1
- I2
- lc1
- lc2
- length\_element\_
- length\_element\_
- m
- M
- m1

The Command Window at the bottom shows the prompt `J>>`.



# Tracking Controls: Design and Fundamental Limitations



For  $F(s)=0$ : standard one degree-of-freedom control loop with the tracking error  $e$ :

$$e = S(s)r + T(s)n, \quad S(s) = \frac{I}{I + P(s)C(s)}, \quad T(s) = \frac{P(s)C(s)}{I + P(s)C(s)}$$

Desired: good reference tracking i.e.  $S(s) \ll 1$  and good noise rejection i.e.  $T(s) \ll 1$ . **But  $S(s) + T(s) = 1$ !**

Need for choosing a two degree-of-freedom control structure, using reference and output measurement.

# European Robotic Arm: Control Requirements and Design Assumptions

**Control task: reference tracking for load positioning (tight control)**

**Place load from home position e.g.  $(x,y)=(11.3\text{ m}, 0\text{ m})$  to mission position e.g.  $(x,y)=(4\text{ m}, -1.65\text{ m})$**

**Closed-loop tracking specs:**

- steady-state in max. 20 seconds (firm)
- no steady-state error, no overshoot (firm)
- motion decoupling between two links (firm)
- link I can move slower, if necessary

**Design assumptions:**

- reference trajectory available, given in joint space  $(0^\circ, 0^\circ)$  to  $(45^\circ, -135^\circ)$
- one single load with known mass and inertia
- motor torques directly commanded
- rigid body motion only (assumption not met later on)



# Interactive Decoupled Tracking Control Design using the PID Tuning GUI

The screenshot displays the MATLAB PID Tuner interface. The main window is titled "PID Tuner - Step Plot: Reference tracking". It features a "VIEW" tab and a "PID TUNER" section with the following settings:

- Plant: PDF2
- Type: PDF2
- Form: Standard
- Domain: Time
- Response Time (seconds): 1
- Transient Behavior: 0.5

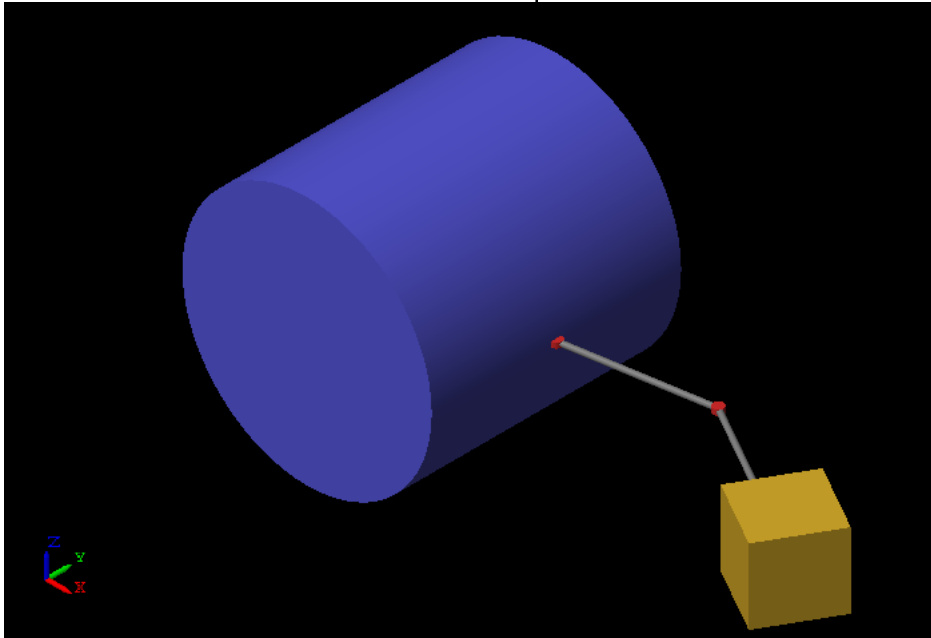
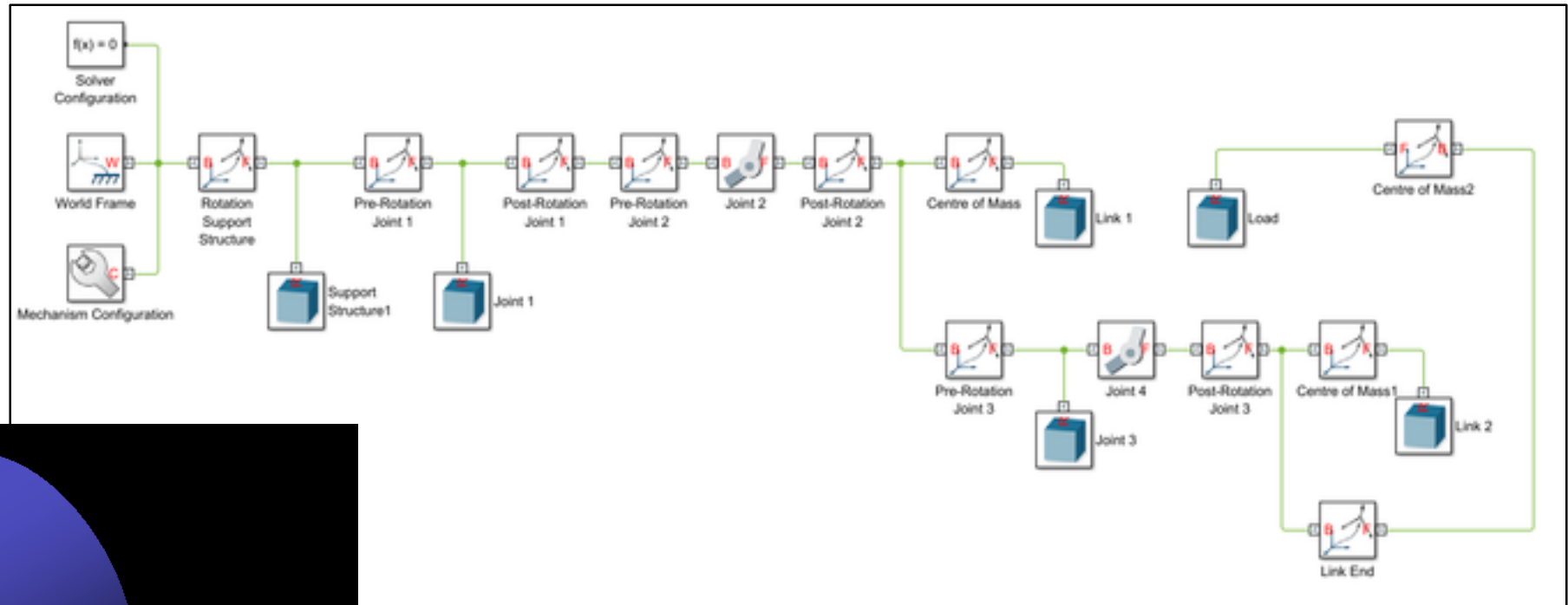
Three callouts highlight key features:

- 1. Control architecture**: Points to the "Type" and "Form" dropdowns.
- 2. Closed-loop specs**: Points to the "Response Time (seconds)" and "Transient Behavior" sliders.
- 3. Save design**: Points to the "Export" menu, which includes options like "Export plant or controller to MATLAB workspace" and "Save as Baseline".

The central plot, titled "Step Plot: Reference tracking", shows the "Tuned response" as a blue curve. The y-axis is labeled "Amplitude" (ranging from 0 to 1.0) and the x-axis is labeled "Time (seconds)" (ranging from 0 to 10). The curve starts at (0,0) and rises to a steady-state value of approximately 0.95. Below the plot, the controller parameters are listed:  $K_p = 0.7975$ ,  $T_d = 2.428$ ,  $N = 66.81$ ,  $b = 1$ ,  $c = 0.5221$ .

# Multibody Dynamics Visualization using Simscape Multibody™

Simulated robot motion can also be visualized in MATLAB™ with little extra work!



The control loop can be closed with previously-designed Simulink™-based controllers.

Multibody-based simulations can (in)validate previous steps!



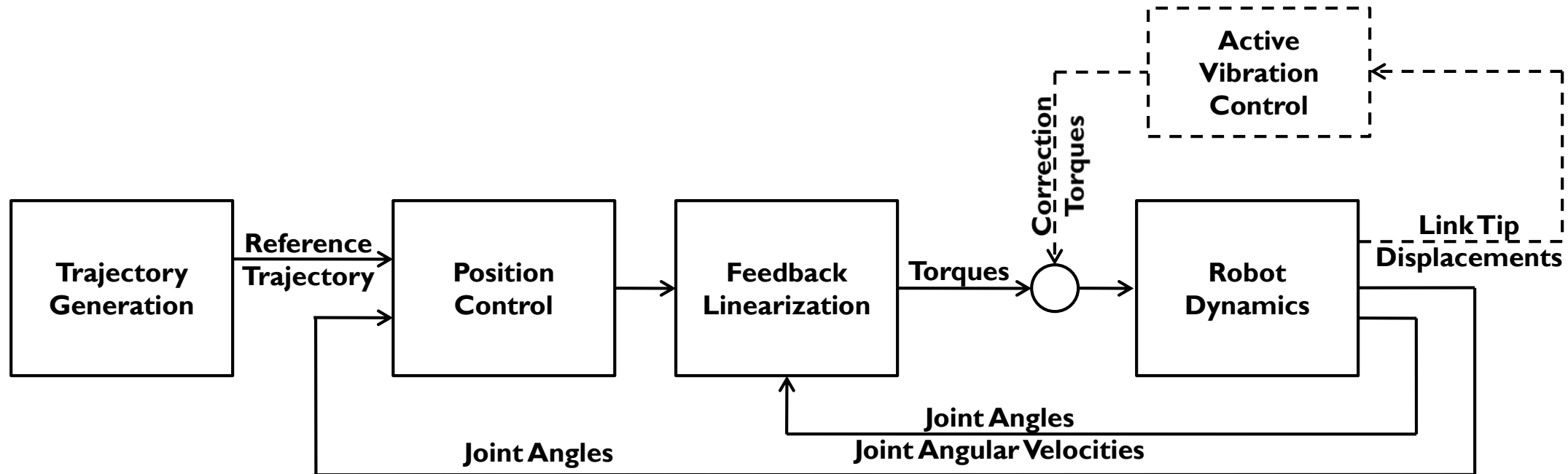
# Simulating Vibrations in Flexible Multibody Systems

**Mechanical vibrations: mathematically modeled with partial differential equations**

**For simulation and control design - approximate by ordinary differential equations:**

- empirically, using e.g. **lumped parameter modeling**
  - + intuitive, simple to implement in multibody modeling software e.g. **Simscape Multibody™**
  - limited accuracy even for fine grids, can be difficult to tune
- numerically, using e.g. **finite element analysis**
  - + accurate method, dedicated software e.g. **NASTRAN™, MATLAB/PDE Toolbox™**
  - computationally intensive, specifications not always trivial (e.g. meshing)

# System Identification for Active Vibration Controls



**Main idea:** design an additional control loop to damp the vibrations using correction torques.

**Model of the link flexibility dynamics necessary, best achieved from experimental data.**

**Main issues:**

1. choice of point of excitation, design of excitation (the experiment design problem)
2. choice/design of data-driven modeling approach (the identification method problem)
3. model assessment and uncertainty quantification (the model validation problem)

**In line with the control objective (desired closed-loop performance translates to model properties).**



## Concluding Remarks

- **Model-based analysis with MATLAB™ and Simulink™/Simscape™ greatly accelerates the research engineering process: extensive, versatile tools (1-2 man-months for ERA)**
- **Symbolic calculations possible: alternative to *pen and paper* derivations and allow avoidance of errors**
- **Simple, intuitive linear controller design and analysis of results using the available apps**
- **Fast prototyping for multibody dynamics (rigid/flexible) using Simulink™/Simscape™**
- **Algorithms for data-driven modeling available in the MATLAB/System Identification™ toolbox, regularly updated with validated novel algorithms**

## Related Works and Background Material

### Vibration suppression beyond flexible robots – an ubiquitous control challenge:

- improved aeroelastic response of aerospace structures (aircraft, wind turbines)
- improved drivetrain damping (automotive, wind turbines)
- fatigue reduction in large base-fixed structures (wind turbines, civil structures)

### Some background material for further reading:

- [1] M.W. Spong, S. Hutchinson and M.Vidyasagar – Robot Modeling and Control, Wiley, 2006.
- [2] H. Crujisen et. al – The European Robotic Arm: A High-Performance Mechanism Finally on its Way to Space, 42<sup>nd</sup> Aerospace Mechanics Symposium, NASA Goddard Space Flight Center, 2014.
- [3] S. Skogestad and I. Postlethwaite – Multivariable Feedback Control: Analysis and Design, Wiley, 2005.
- [4] J.-N. Juang – Identification and Control of Mechanical Systems, Cambridge University Press, 2001.



**Thank you for your attention!**

