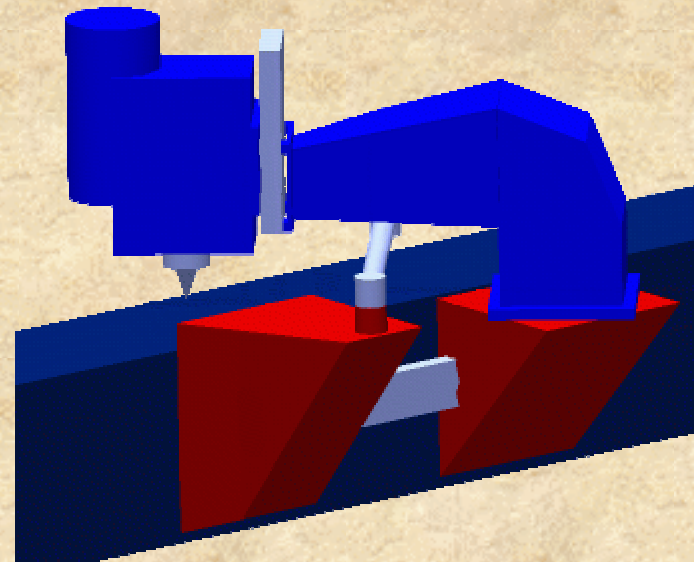


# **Modelling and Control of a Machine Tool Using Co-Simulation**

Alexandra Ratering  
Peter Eberhard



# Outline

- ❖ introduction
- ❖ the Lambda Kinematic
- ❖ modelling
  - mechanical and mathematical model
  - equations of motion
- ❖ flatness
  - introduction to the concept of flatness
  - flatness analysis of the machine tool
- ❖ controller design
  - linear cascaded P-PI control
  - flatness based control
- ❖ simulation
  - simulation environment
  - comparison of control concepts
- ❖ conclusions



### demand for **highly productive machine tools**

(high velocities, large scale movements, ...)

### development of

- new **machine types** (parallel and hybrid kinematics)
- new **technologies** (linear drives, lightweight structures, ...)

### problem

- assumption of **independently** controllable drives questionable
- **elastic properties** become important
- **nonlinearities** become important

### solution

**hierarchical controllers** combining position control with methods of active vibration damping



## the Lambda Kinematic

- processing of plate-like workpieces (esp. wood processing)
- scissors-like kinematic for movement in xy-plane
- movement along z-axis realized by a serial axle
- high dynamics: support velocities up to 2 m/s, max. acceleration 9 m/s<sup>2</sup>
- implemented motion control system: **two independent P-PI controllers** for each drive

built at the Institute of Machine Tools (IfW) at the University of Stuttgart

## Machine Tool



### problems with

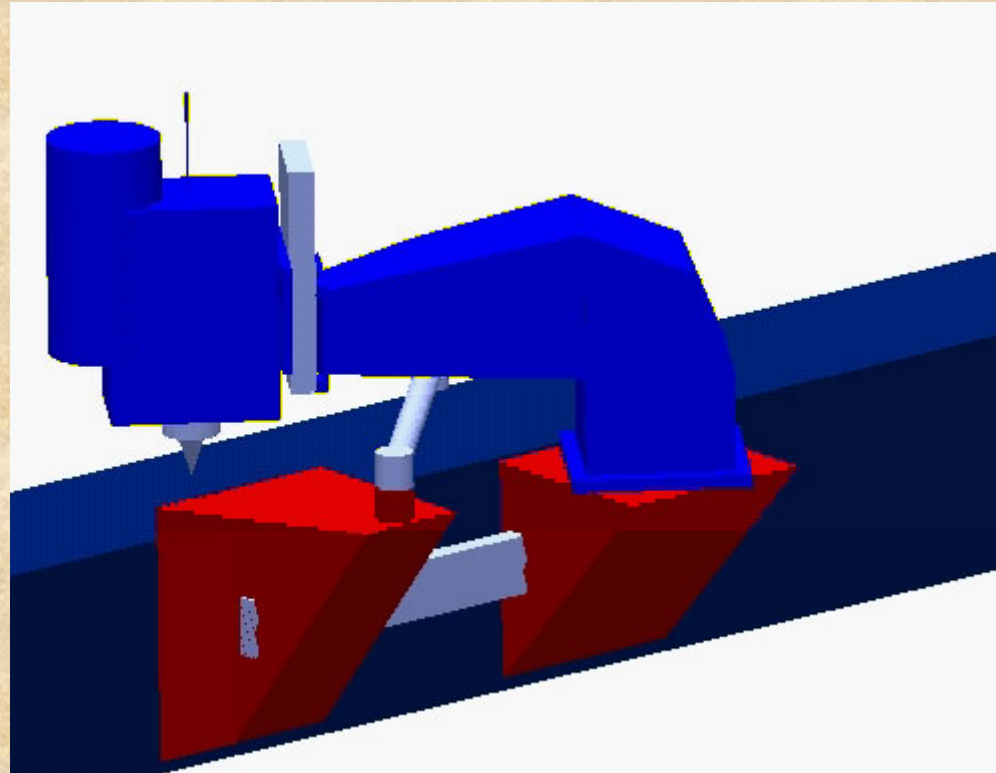
- overall performance
- vibrations during processing

⇒ **1<sup>st</sup> step: development of a better position controller**



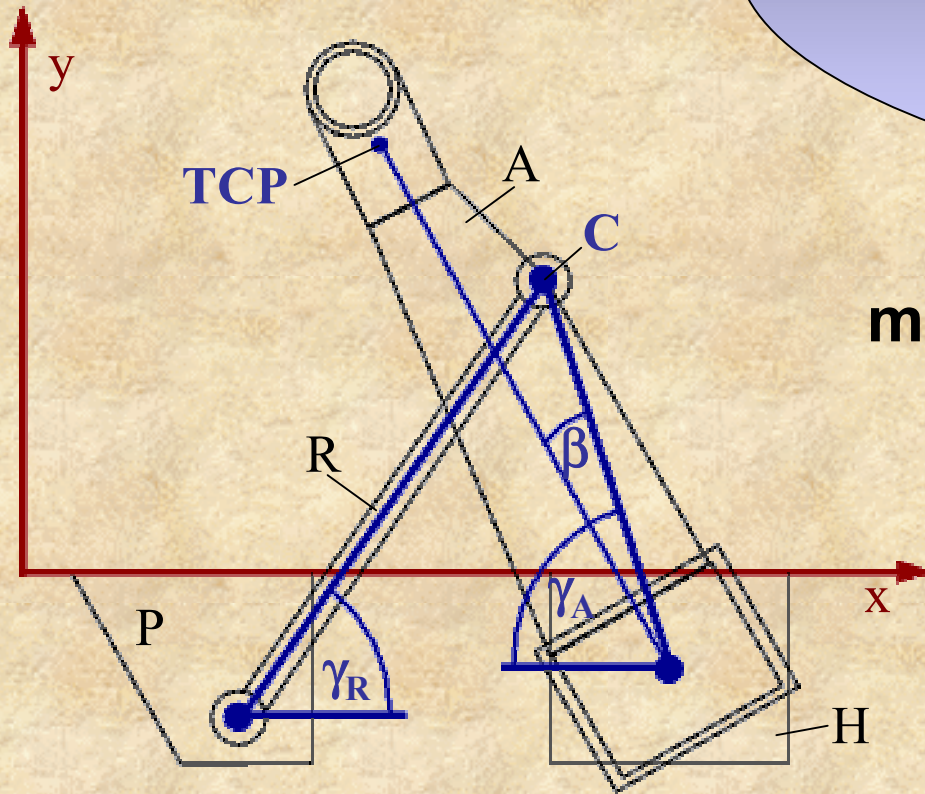
Institute B of Mechanics  
Prof. Dr.-Ing. Peter Eberhard  
University of Stuttgart

# Animation of Motion



Institute B of Mechanics  
Prof. Dr.-Ing. Peter Eberhard  
University of Stuttgart

# Mechanical Model



## multibody system model

- 4 rigid bodies P, H, R, A
- 2 degrees of freedom
- 1 kinematic loop: DAE system
- minimal coordinates  $\mathbf{q}_m = [x_P \ x_H]$
- joint coordinates  $\mathbf{q}_t = [x_P \ x_H \ \gamma_R \ \gamma_A]$

## procedure

- cut kinematic loop at joint C
- Newton-Euler equations for each branch of open tree structure
- algebraic constraint as loop closing condition



## Newton-Euler equations for the open tree structure

## Equations of Motion

$$\begin{array}{c}
 \text{generalized gyroscopic and centrifugal forces} \quad \text{generalized applied forces} \quad \text{Boolean matrix} \quad \text{motor forces} \\
 \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
 \text{mass matrix} \rightarrow \boxed{\mathbf{M}(\mathbf{q}_t) \cdot \ddot{\mathbf{q}}_t + \mathbf{k}(\mathbf{q}_t, \dot{\mathbf{q}}_t) = \mathbf{Q}^a(\mathbf{q}_t, \dot{\mathbf{q}}_t, t) + \mathbf{H} \cdot \mathbf{u} + \mathbf{Q}^r} \leftarrow \text{generalized constraint/ reaction forces} \\
 \mathbf{Q}^r = \mathbf{G}^T \cdot \boldsymbol{\lambda}
 \end{array}$$

### implicit constraint equations

$$\mathbf{g}(\mathbf{q}_t) = \begin{bmatrix} x_p - x_H + l_R \cos(\gamma_R) + l_A \cos(\gamma_A) \\ y_p - y_H + l_R \sin(\gamma_R) - l_A \sin(\gamma_A) \end{bmatrix} = \mathbf{0}$$

$$\dot{\mathbf{g}} \equiv \frac{\partial \mathbf{g}}{\partial \mathbf{q}_t} \cdot \dot{\mathbf{q}}_t = \mathbf{G}(\mathbf{q}_t) \cdot \dot{\mathbf{q}}_t = \mathbf{0}$$

### equations of motion in minimal coordinates

explicit constraint equations  $\mathbf{q}_t = \mathbf{q}_t(\mathbf{q}_m)$   
can be calculated analytically

$$\Rightarrow \begin{cases} \gamma_i = \gamma_i(\mathbf{q}_m), & i = R, A \\ \dot{\gamma}_i = \dot{\gamma}_i(\mathbf{q}_m, \dot{\mathbf{q}}_m), \\ \ddot{\gamma}_i = \ddot{\gamma}_i(\mathbf{q}_m, \dot{\mathbf{q}}_m, \ddot{\mathbf{q}}_m), \end{cases}$$

but resulting ODE-formulation is complicated and for flatness-based controller not necessary!



## definition

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}), \quad \mathbf{x}(0) = \mathbf{x}_0 \in \mathcal{R}^n, \mathbf{u} \in \mathcal{R}^m$$

is (differentially) **flat**, if there exists

$$\mathbf{y} = \Phi(\mathbf{x}, \mathbf{u}, \dot{\mathbf{u}}, \dots, \mathbf{u}^{(\alpha)}), \quad \dim(\mathbf{y}) = \dim(\mathbf{u})$$

for which (locally)  $\mathbf{x} = \Psi_1(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\beta)})$

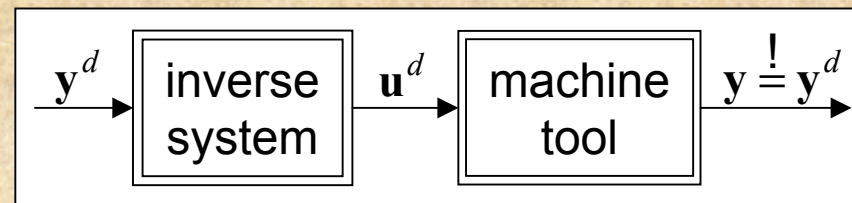
$$\mathbf{u} = \Psi_2(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\beta+1)})$$

$\mathbf{y}$  is called a **flat output** of the system

in other words:

all *states* and *inputs* of a system can be calculated as an *algebraic function* of the *flat output* and a final number of its *derivatives*

## open loop control



... sufficient for ideal system



# Flatness Analysis

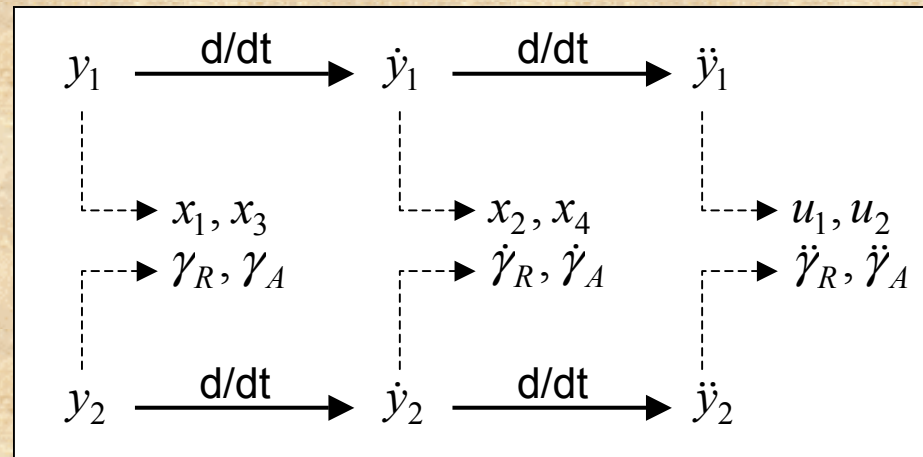
input:  $\mathbf{u} = [u_1 \ u_2]$

states:  $\mathbf{x} = [\mathbf{q}_m \ \dot{\mathbf{q}}_m] = [x_P \ \dot{x}_P \ x_H \ \dot{x}_H]$

flat output:  $\mathbf{y} = \mathbf{q}_m = [x_P \ x_H]$

$$\left. \begin{array}{l} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \\ \mathbf{y} = [x_P \ x_H] \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} \mathbf{x} = \Psi_1(\mathbf{y}, \dot{\mathbf{y}}) \\ \mathbf{u} = \Psi_2(\mathbf{y}, \dot{\mathbf{y}}, \ddot{\mathbf{y}}) \end{array} \right.$$

$\gamma_R, \gamma_A, \dot{\gamma}_R, \dot{\gamma}_A, \ddot{\gamma}_R, \ddot{\gamma}_A$   
... auxiliary variables



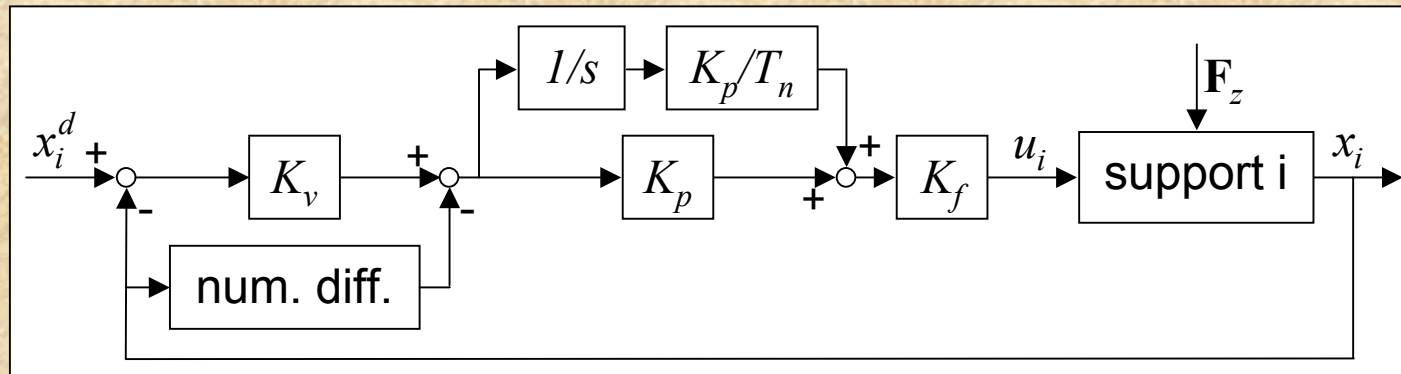
system is flat, hence controllable and observable

- singularities:
- 2 possible positions
  - not reachable due to construction



# Linear Controller

- preliminary: implemented P-PI controller (for a single support,  $i = P, H$ )



- equivalent to an incomplete PID controller
- tuned experimentally
- equal  $K_p, K_v$  and  $T_n$  values for both supports
- **system performance varies** over the workspace



# Flat Controller Design with Feedback Linearization

- static state feedback

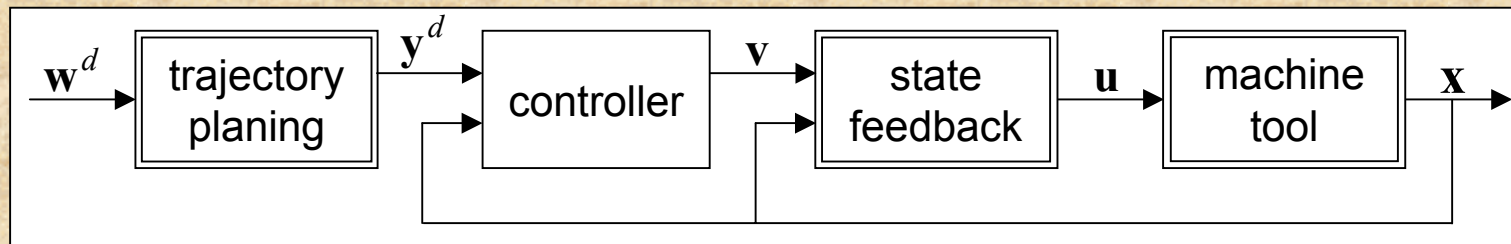
$$\mathbf{u} = \mathbf{u}(\mathbf{y}, \dot{\mathbf{y}}, \mathbf{v})$$

➤ linear system with new input  $\mathbf{v} = \ddot{\mathbf{y}} = \ddot{\mathbf{q}}_m$

- PD controller, PID controller

$$\mathbf{v} = \ddot{\mathbf{y}}^d + \mathbf{K}_1 \cdot (\dot{\mathbf{y}}^d - \dot{\mathbf{y}}) + \mathbf{K}_2 \cdot (\mathbf{y}^d - \mathbf{y}) + \mathbf{K}_3 \cdot \int (\mathbf{y}^d - \mathbf{y}) dt$$

- linear, asymptotic stable error dynamics  $\ddot{\mathbf{e}} + \mathbf{K}_1 \cdot \dot{\mathbf{e}} + \mathbf{K}_2 \cdot \mathbf{e} + \mathbf{K}_3 \cdot \int \mathbf{e} dt = \mathbf{0}$
- tuning e.g. with pole placement ...



- homogenous system performance over the whole workspace
- simple tuning of PID controller
- robustness?



# Flat Controller Design with Feedforward Linearization

- feedforward linearization (Hagenmeyer, Delaleau 2003)

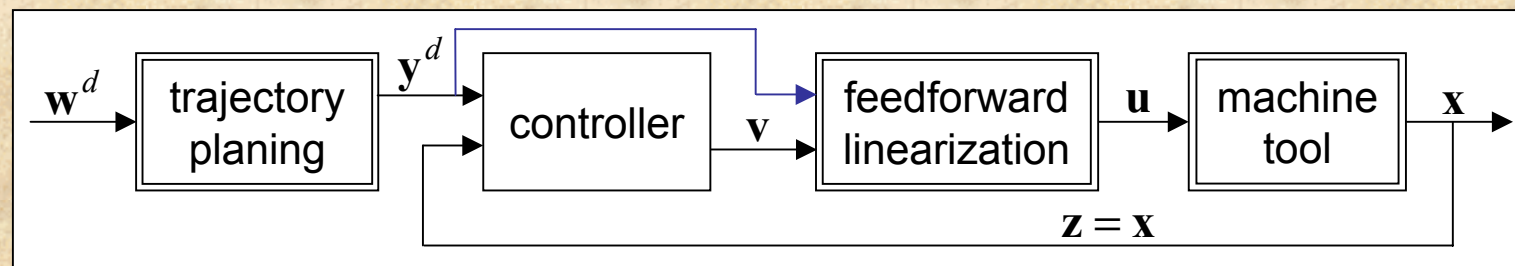
$$\mathbf{u} = \mathbf{u}(\mathbf{y}^d, \dot{\mathbf{y}}^d, \mathbf{v})$$

- exact linearization when being on the desired trajectory

- PD controller, PID controller

$$\mathbf{v} = \ddot{\mathbf{y}}^d + \mathbf{K}_1 \cdot (\dot{\mathbf{y}}^d - \dot{\mathbf{y}}) + \mathbf{K}_2 \cdot (\mathbf{y}^d - \mathbf{y}) + \mathbf{K}_3 \cdot \int (\mathbf{y}^d - \mathbf{y}) dt$$

- stabilizing in the vicinity of the desired trajectory



- better robustness expected against measurement noise, parameter uncertainties ...



# Why Co-Simulation?

## Co-Simulation



implementation of control and other non-mechanical model parts

- well established control design and simulation tools
- easy to adapt/change
- offline trajectory generation
- real time code generation
- ...

simulation of complex MBS

- nonlinear dynamics
- kinematic loops
- elastic bodies
- ...

ability to efficiently test MBS with different control concepts etc. and vice versa

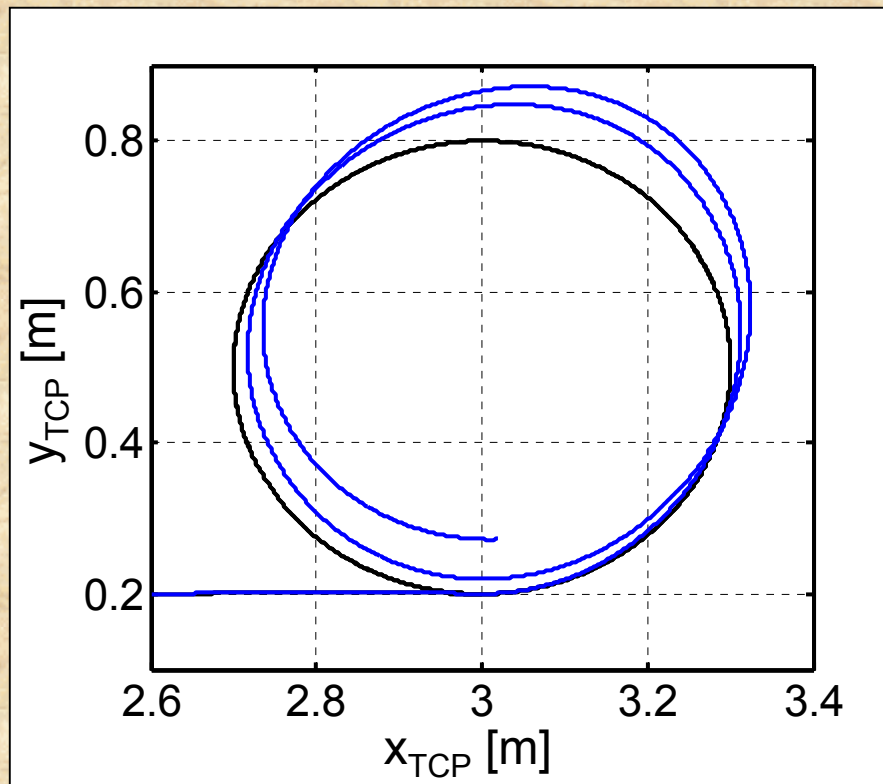




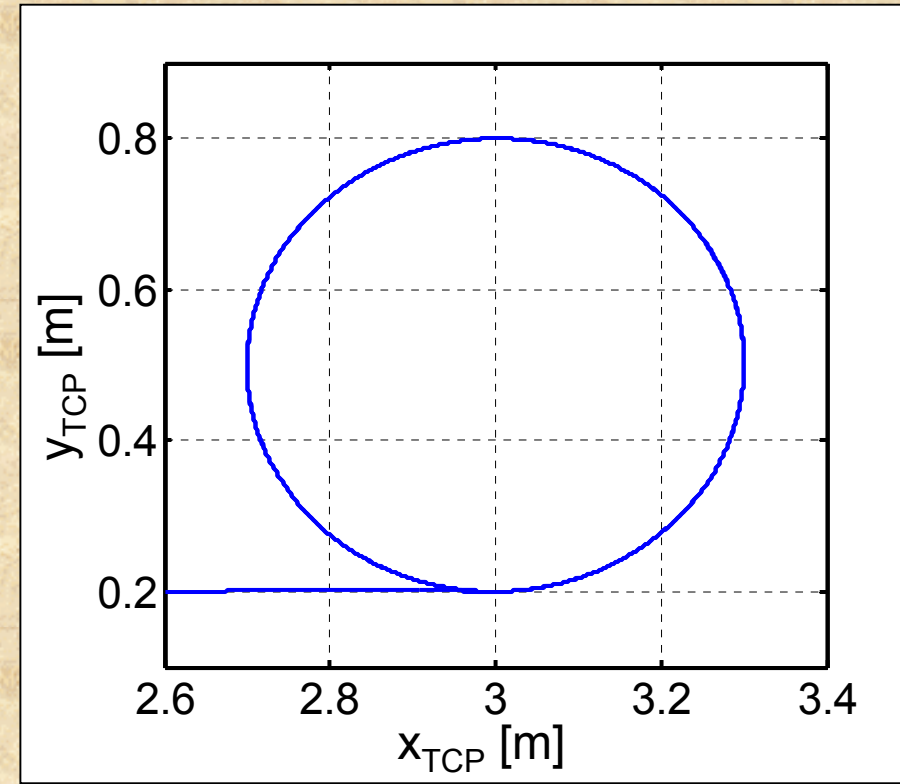
with 30% disturbance of the moment of inertia of the rod and the cantilever in the state feedback calculation

## Open- vs. Closed-Loop Control

open-loop



closed-loop

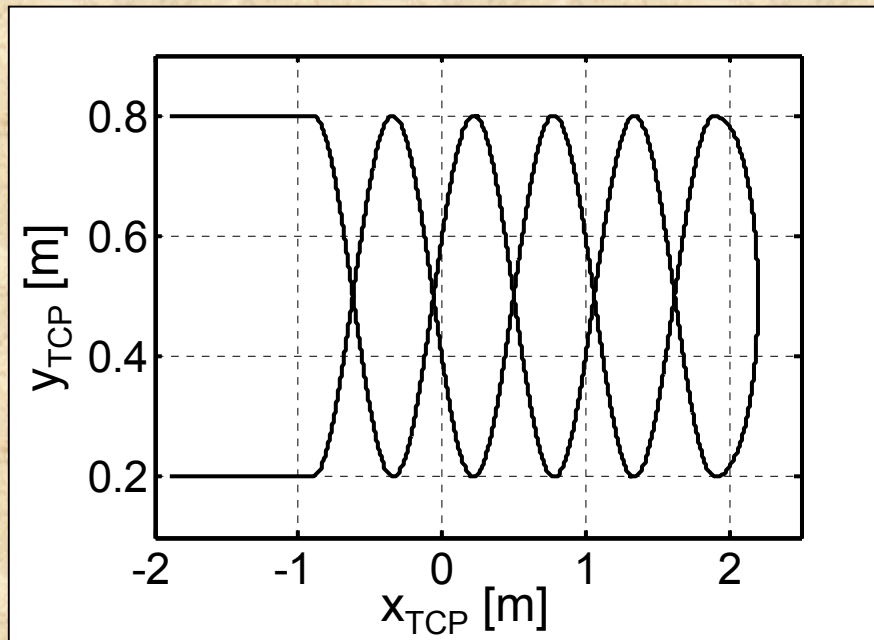


— : trajectory  
— : simulation result



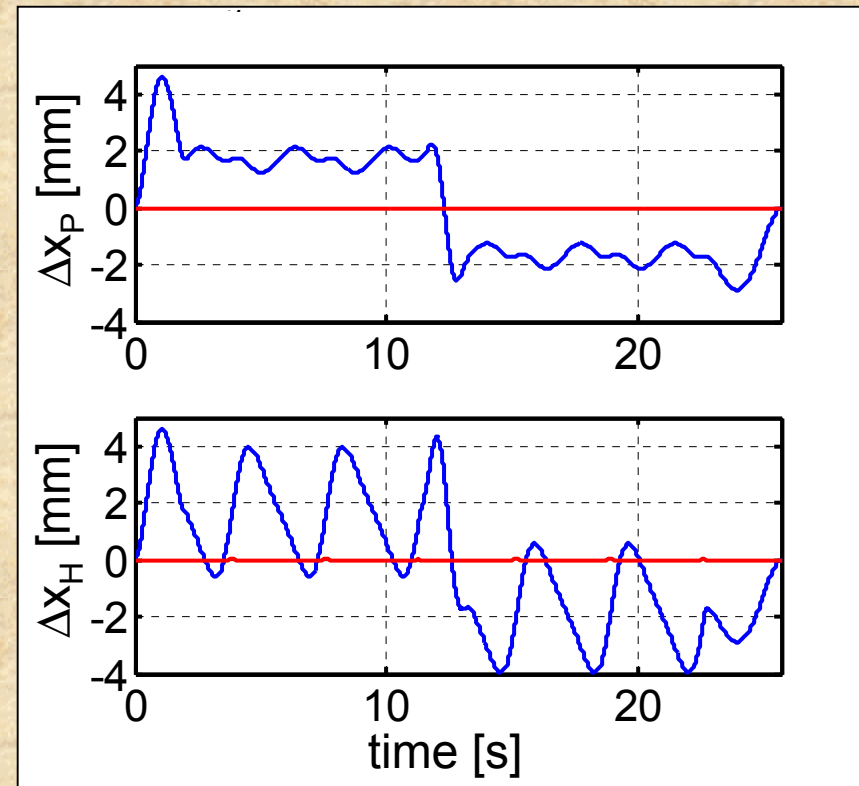
# Linear vs. Nonlinear Controller

trajectory



with 30% disturbance of the moment of inertia of the rod and the cantilever in the state feedback calculation

position error of the supports

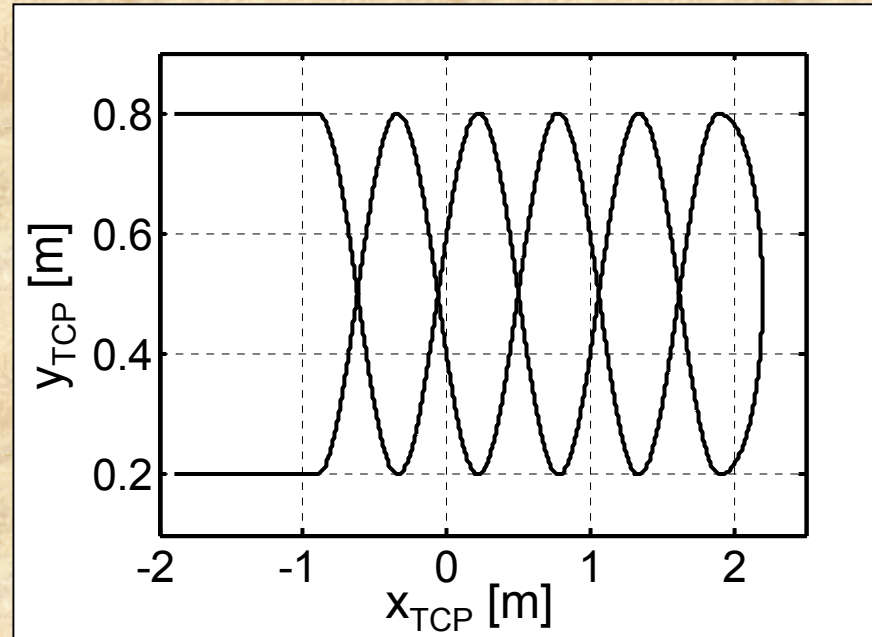


—: linear controller  
—: flatness-based controller



# Simulation with Process Forces

trajectory

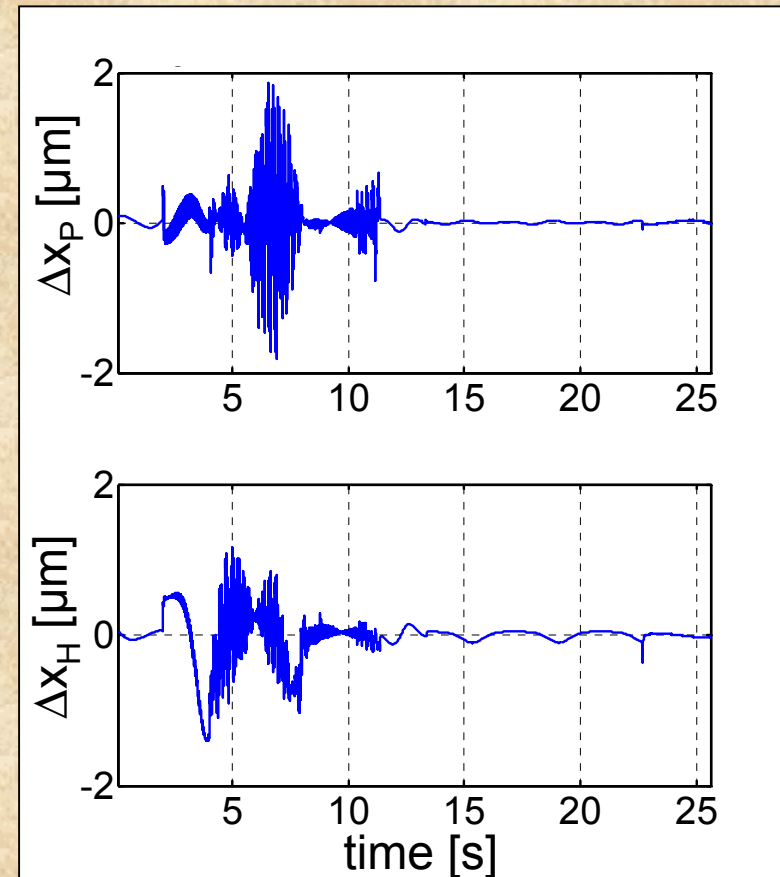


**simplified model of the milling process**

$$F_c = k_c (v_f) (\sin(\Omega z t) + 1) \text{ in feed direction}$$

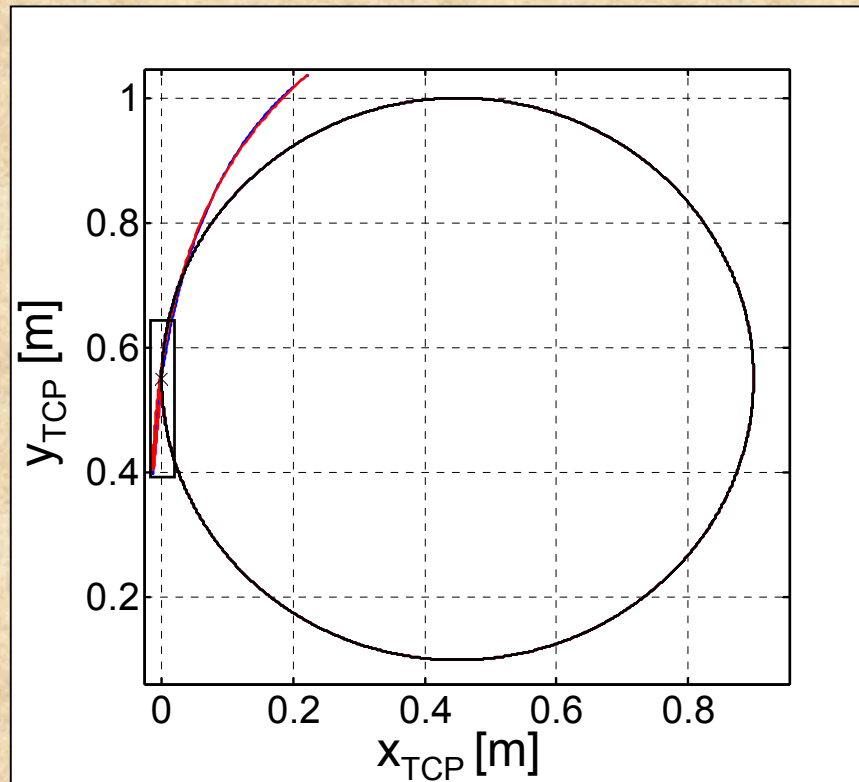
$$F_{cN} = k_{cN} (v_f) (\sin(\Omega z t) + 1) \text{ in z-direction}$$

position error of the supports



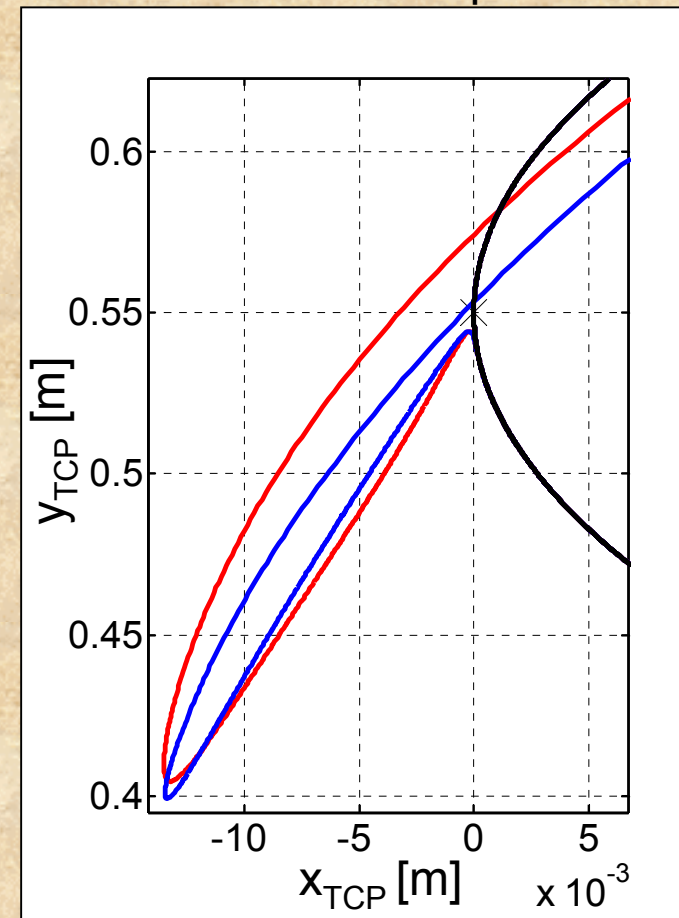
with 30% disturbance of the moment of inertia of the rod and the cantilever in the state feedback calculation workspace

## Feedforward and Feedback Linearization



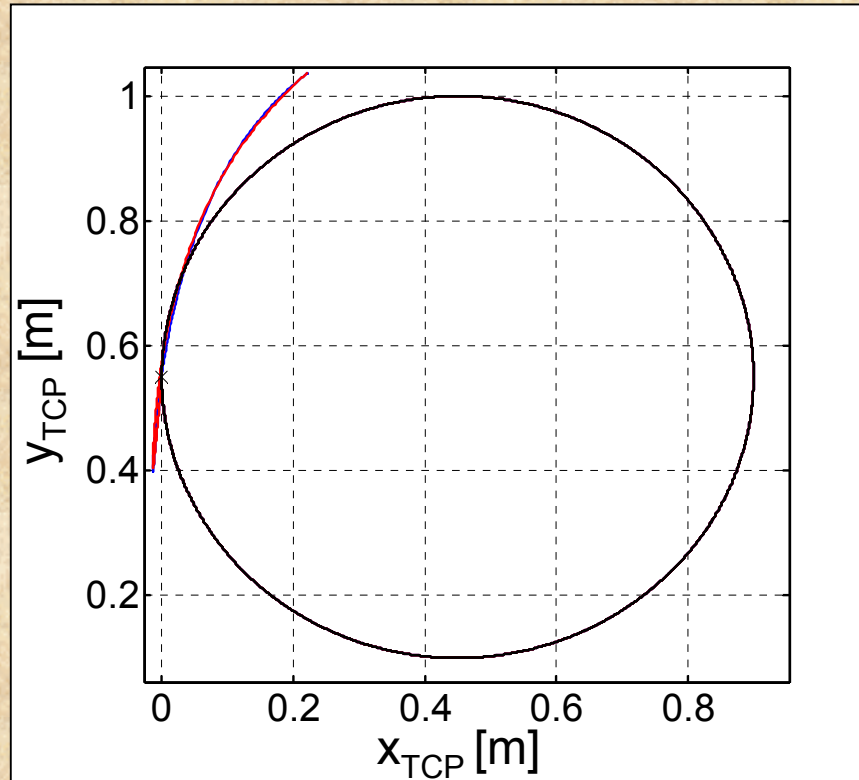
— : trajectory    — : feedback linearization  
— : feedforward linearization

section of workspace



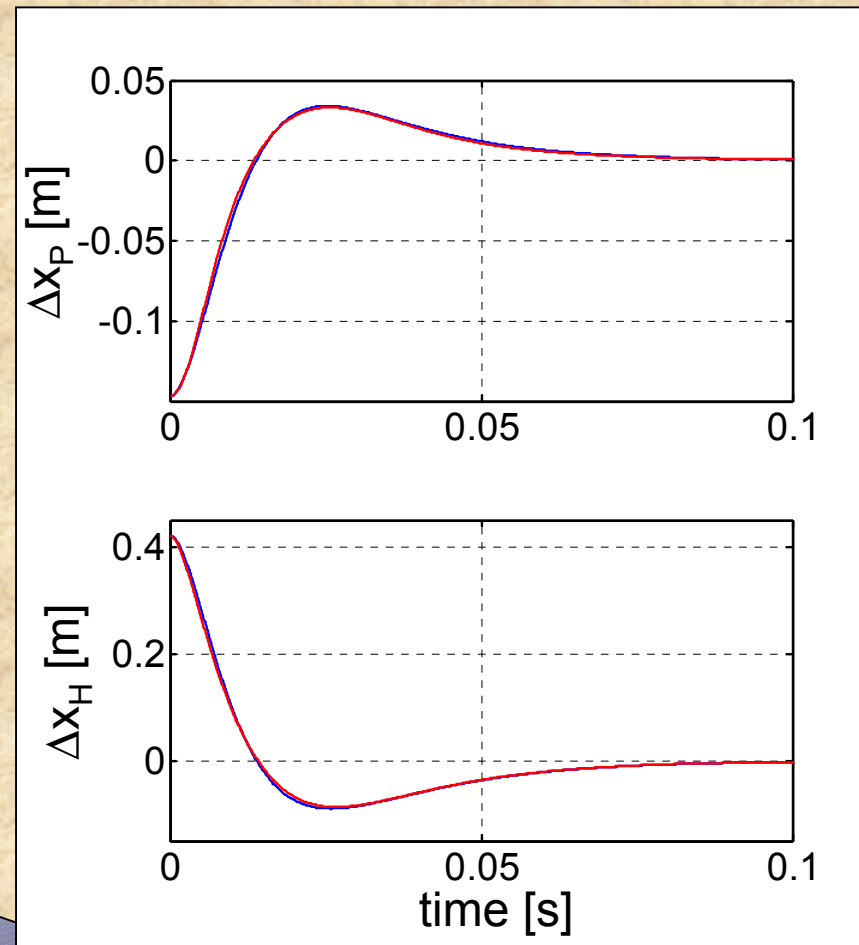
with 30% disturbance of the moment of inertia of the rod and the cantilever in the state feedback calculation workspace

## Feedforward and Feedback Linearization



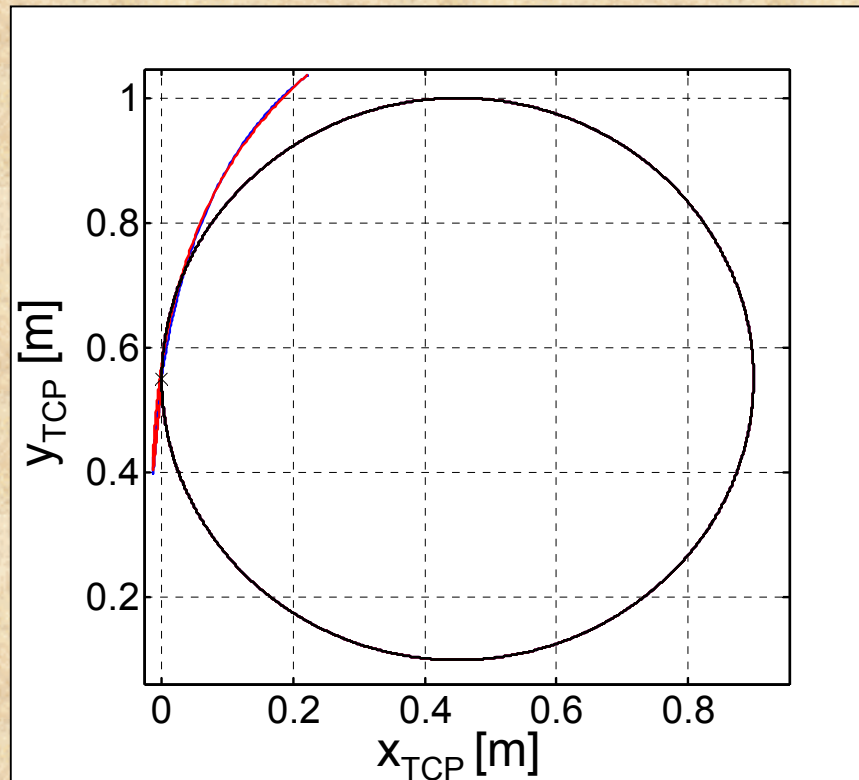
— : trajectory    — : feedback linearization  
— : feedforward linearization

large initial error of the supports



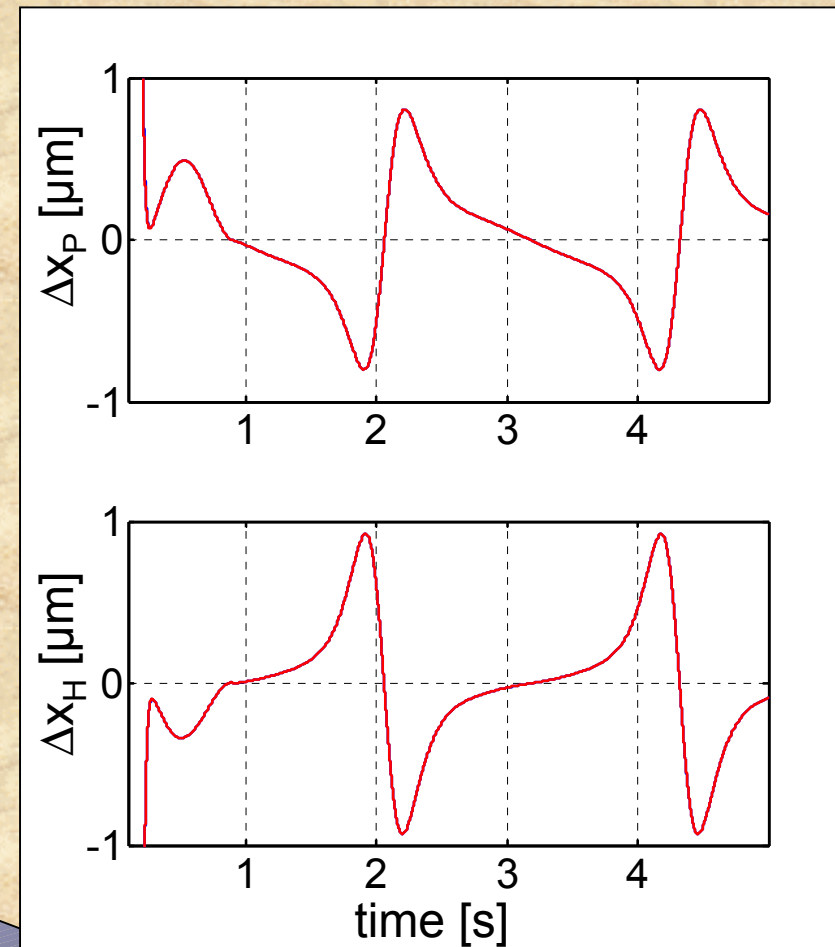
with 30% disturbance of the moment of inertia of the rod and the cantilever in the state feedback calculation workspace

## Feedforward and Feedback Linearization



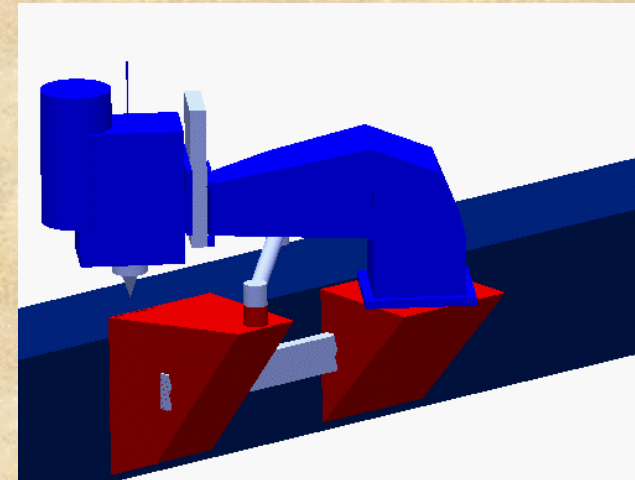
— : trajectory    — : feedback linearization  
— : feedforward linearization

position error of the supports



## conclusions

- flatness-based controller feasible
- physically insightful
- simple controller design
- improves performance significantly
- co-simulation with SIMPACK and MATLAB/SIMULINK very useful



## outlook

- model enhancement (elastic MBS, ...)
- controller enhancement (active vibration damping, hierarchical control)
- validation of model and simulation results at the real machine

