

W. Schiehlen, M. Arnold (eds.)

Co-Simulation for Mechatronic Systems

Materials of a Workshop that was held at the
University of Stuttgart on October 11, 2001

IB 532-2001-10

Freigabe:

Der Bearbeiter:

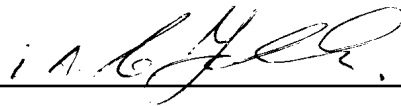
Unterschrift:

Prof. Dr.-Ing. W. Schiehlen



PD Dr. M. Arnold

Der Leiter der OE:



Prof. W. Kortüm

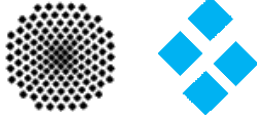
Dieser Bericht enthält

110

Blatt davon

Bilder

Diagramme



2nd Workshop on Co-Simulation Stuttgart, October 11, 2001



Preface

In mechatronic systems the mechanical, electronic and hydraulic components are interacting closely. The increasing integration of related industrial design processes is essentially based on the efficient computer simulation of mechatronic systems. One of the most favourable approaches for this purpose is co-simulation, i. e. the coupling of well established existing simulation tools from different fields such as multibody dynamics, hydraulics and control system design.

This report summarizes the materials of a workshop on co-simulation that was held at the University of Stuttgart (Germany) on October 11, 2001. It was the second workshop in a biennial series which was initiated in May 1999 by a group of interested colleagues at the “Herbertov Workshop” of the International Association of Vehicle System Dynamics (IAVSD). Only half a year later the “1st Workshop on Co-Simulation” was held at DLR Braunschweig.

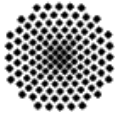
In the Stuttgart workshop the interest of most participants was again in the dynamical simulation of vehicles and vehicle components. But co-simulation is successfully used in other fields of application as well. This was nicely illustrated by two contributions on electric circuit simulation.

In all there were 18 participants from industry, universities and research institutes. Seven contributed papers and a software demonstration were presented at the workshop. Furthermore the timetable left ample space for fruitful discussions on co-simulation or — more generally — on different approaches to the dynamical simulation of heterogenous engineering systems.

Traditionally, rather elementary numerical strategies have been used in co-simulation: fixed communication stepsizes, substitution of algebraic constraints by penalty terms and low order data approximation for the data exchange between subsystems. All these restrictions may be overcome by an iterative simulator coupling method with adaptive stepsize control that was developed at Stuttgart University.

Co-simulation is often used as a convenient approach to the coupled simulation of heterogenous systems. It may be applied as well to solve homogenous problems that are too large to be handled by standard methods. At the Fraunhofer Institute in Dresden (IIS/EAS) a blockoriented splitting method has been developed that makes large-scale problems from circuit simulation solvable and improves already for medium-scale problems the numerical efficiency since different subsystems may be handled by different numerical methods.

In automotive and railway applications co-simulation is used for real-time simulations. In the framework of the ODECOMS project real-time simulations of complete vehicle models have been considered at CEIT San Sebastián. The model of a Peugeot 806 van includes detailed descriptions of car body, suspensions, tyres and the driveline. The kinematic and kinetostatic behaviour of the suspensions is approximated by cubic splines to avoid closed loops and to get real-time performance.



2nd Workshop on Co-Simulation Stuttgart, October 11, 2001



At Dresden University of Technology hydraulic and multibody system models have been coupled to analyse railway vehicles with a hydraulically driven active tilting system. In the simulations Simulink is used to couple submodels that are exported from the commercial packages DSHplus and SIMPACK. With the Real-Time Workshop this coupled Simulink model is transferred to real-time hardware. It has been applied to test hydraulic actuators on a hardware-in-the-loop test rig.

The enhanced modelling and simulation package 20-sim is developed at Controllab Products BV and the University of Twente. Two years ago, at the Braunschweig workshop, the capabilities of 20-sim in the field of co-simulation were illustrated by an impressive demonstration on two coupled PC's. At the Stuttgart workshop the 20-sim demonstration focussed on the interfaces between Matlab and 20-sim.

Practical experience with industrial applications is an essential prerequisite to improve the numerical techniques and software tools for the dynamical simulation of heterogenous engineering systems. Various strategies for the interdisciplinary modelling and simulation were discussed in the framework of the EUMECH project. A complete vehicle model with mechanical, hydraulic and control elements was developed by INTEC GmbH, Wessling, and MLaP Paderborn. As a benchmark problem the braking maneuver of a car with anti-skid system was considered in detail.

The rapidly increasing interest in the simulation of coupled problems has its origin in the increasing integration of engineering systems and in the increasing miniaturization of components like microsensors or electric circuits. The heat evolution in highly integrated circuits may be considered as a typical example. In the approach of Karlsruhe University the electric network equations and the heat equation are considered simultaneously resulting in a partial differential-algebraic equation (PDAE) that may be solved numerically.

Similar to the first presentation at this workshop also the last one was devoted to numerical methods for co-simulation. It is well known that coupling of subsystems by constraints or by stiff forces may cause numerical instability in standard co-simulation methods. Several methods have been proposed in the literature to fix this instability phenomenon. At DLR Oberpfaffenhofen an overrelaxation method has been used for a stable and efficient coupled simulation of multibody systems and large elastic structures.

The materials of all eight contributions illustrate the wide spectrum of topics that was discussed on this workshop. Main objectives were the engineering aspects of co-simulation, industrial applications and the theoretical background. At the end it was decided to continue the biennial series with a "3rd Workshop on Co-Simulation" to be held in autumn 2003.



Table of contents

List of participants	3
W. Schiehlen, Ch. Scholz	
Step size control of simulator coupling for multibody systems	5
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/scholz.pdf)	
C. Clauß, P. Schwarz	
Coupled simulation in blockoriented network analysis	19
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/clauss.pdf)	
A. Suescun, J. González, S. Ausejo, J.T. Celigüeta	
Co-simulation of complete vehicle models with real-time performance	43
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/suescun.pdf)	
S. Dronka	
Modelling and simulation of coupled hydraulic and multibody subsystems	57
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/dronka.pdf)	
P. Breedveld, F. Groen	
MATLAB – 20-sim interaction	69
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/breedveld.pdf)	
F. Kohlschmied	
Simulation of an anti-skid system using several modelling and simulation tools	73
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/kohlschmied.pdf)	
A. Bartel, M. Günther	
Co-modelling of electric networks and heat evolution	87
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/bartel.pdf)	
M. Arnold	
The stabilization of time integration methods for co-simulation	97
(http://www.ae.op.dlr.de/workshop/CoSim01/pdf/arnold.pdf)	



List of participants

- PD Dr. M. Arnold (DLR Oberpfaffenhofen, Germany)
martin.arnold@dlr.de
- Dipl.-Math. A. Bartel (University of Karlsruhe, Germany)
bartel@iwrmm.math.uni-karlsruhe.de
- Prof. dr. ir. P. Breedveld (University of Twente, The Netherlands)
p.c.breedveld@el.utwente.nl
- Dr.-Ing. C. Clauß (FhG IIS EAS Dresden, Germany)
clauss@eas.iis.fhg.de
- Dipl.-Ing. S. Dronka (Dresden University of Technology, Germany)
dronka@rcs.urz.tu-dresden.de
- PD Dr. M. Günther (University of Karlsruhe, Germany)
guenther@iwrmm.math.uni-karlsruhe.de
- Dipl.-Ing. A. Heckmann (DLR Oberpfaffenhofen, Germany)
andreas.heckmann@dlr.de
- Dr. K.-D. Hilf (DaimlerChrysler, Germany)
klaus-dieter.hilf@daimlerchrysler.com
- Dipl.-Ing. F. Kohlschmied (INTEC GmbH Weßling, Germany)
frank.kohlschmied@simpack.de
- Dipl.-Ing. W. Neuwald (Robert Bosch GmbH, Germany)
wilfried.neuwald@bosch.com
- Dipl.-Ing. G. Preschany (Dr.-Ing. h. c. F. Porsche AG, Germany)
guenther.preschany@porsche.de
- Dr.-Ing. J. Rauh (DaimlerChrysler, Germany)
jochen.rauh@daimlerchrysler.com
- Dr.-Ing. U. Rein (DaimlerChrysler, Germany)
udo.rein@daimlerchrysler.com
- Prof. Dr.-Ing. W. Schiehlen (University of Stuttgart, Germany)
wos@mechb.uni-stuttgart.de
- Dipl.-Ing. C. Scholz (University of Stuttgart, Germany)
cs@mechb.uni-stuttgart.de
- Dr.-Ing. habil. P. Schwarz (FhG IIS EAS Dresden, Germany)
peter.schwarz@eas.iis.fhg.de
- Dr.-Ing. A. Suescun (CEIT San Sebastián, Spain)
asuescun@ceit.es
- Prof. Ing. M. Valášek, DrSc. (CTU Prague, Czech Republic)
valasek@fsik.cvut.cz

Step size control of simulator coupling for multibody systems

Werner Schiehlen, Christian Scholz

{wos,cs}@mechb.uni-stuttgart.de

**Institute B of Mechanics, University of Stuttgart
Pfaffenwaldring 9, D – 70569 Stuttgart**

Simulation of complex engineering systems requires modelling and computation of components from different engineering fields, e. g. mechanics, control and electronics. Each component can be modelled and computed with its domain-specific tool. But the nondomain-specific parts of the model can not be sufficiently treated by these tools resulting in unsatisfying simulations. It has been shown that approaches of simulator coupling open a systematic and accurate way to combine simulation tools and get satisfying results. Then, the global system has to be decomposed into subsystems due to the different engineering disciplines using engineering intuition to treat it efficiently by a team of engineers.

With the iterative simulator coupling method stabilizing the modular simulation the computation of the global system is realized by a time discrete linker and scheduler which combines the inputs and outputs of the corresponding subsystems and establishes communication between them. In this approach the communication between the coupled simulation tools is executed at fixed time steps.

However, if the coupled modules are characterized by large differences in their eigendynamic, an increase of the numerical efficiency by an automatic communication step size control is achieved. Therefore, two methods of steps size control, Richardson extrapolation and embedded formula, are discussed. It is important that the communication step size control does not interfere with the coupled simulation tools because of the subsystems black-box description within the modular simulation.

It will be shown that such an automatic communication step size control allows to minimize the quantity of communications between the coupled subsystems as well as to use well-known integration methods with step size control for each independent module. A better efficiency by a faster computation can be expected if the dynamic characteristics of the subsystems show large differences.

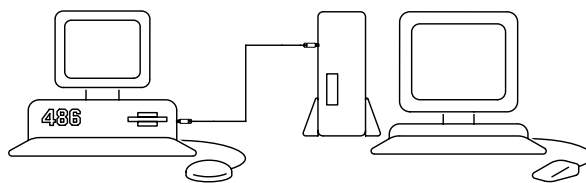


CO-SIMULATION FOR MECHATRONIC SYSTEMS, STUTTART, GERMANY, OCTOBER 11, 2001

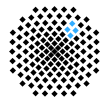


Step Size Control of Simulator Coupling for Multibody Systems

Werner Schiehlen and Christian Scholz



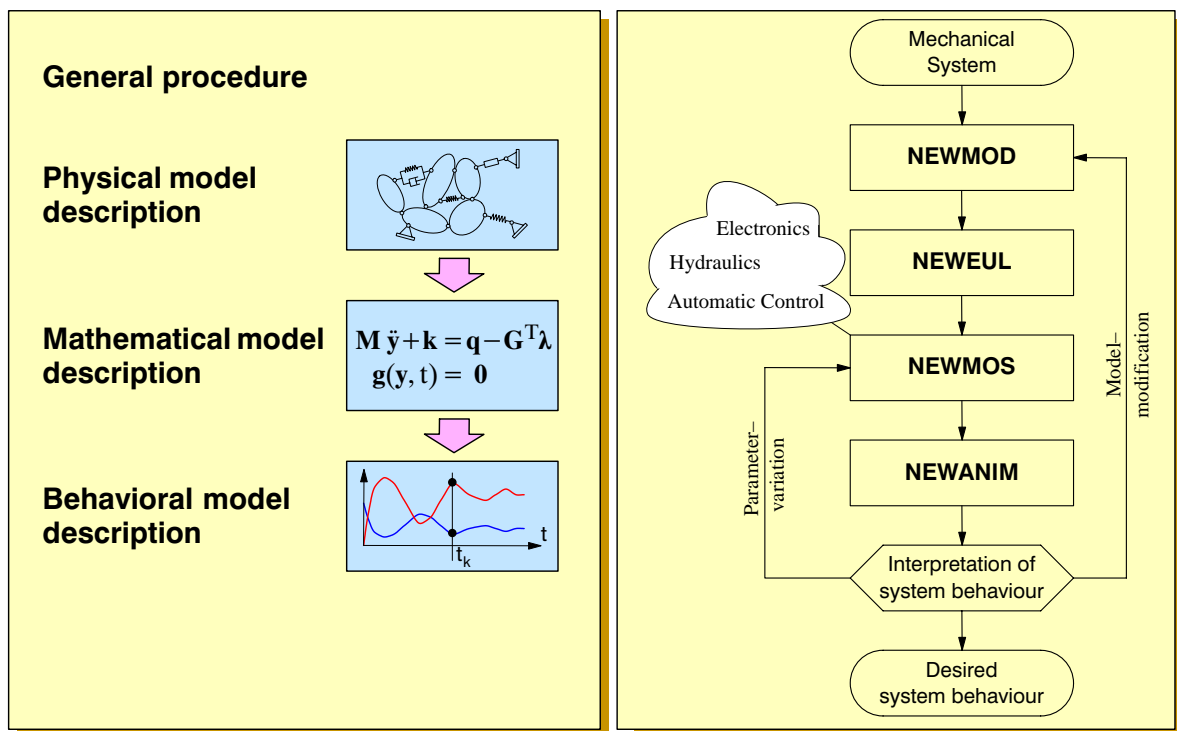
- Introduction
- Dynamic Analysis
- Communication Step Size Control
- Example
- Conclusion



Institute B of Mechanics
Prof. Dr.-Ing. W. Schiehlen
University of Stuttgart

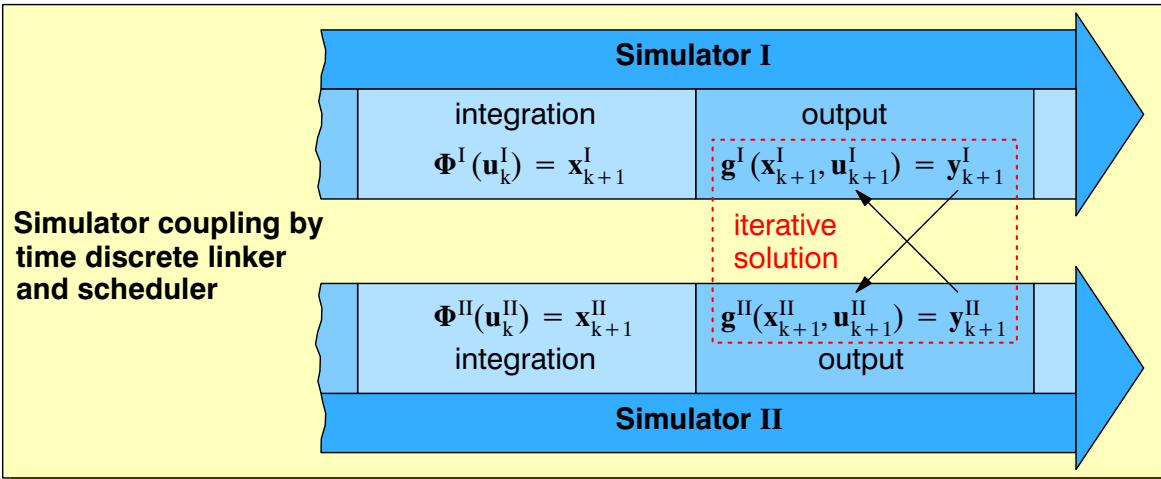


Introduction





Introduction



Stabilization of modular simulation

(see Kübler, Fortschritt–Berichte VDI, Reihe 20, 2000)

- nonlinear algebraic equations
- black–box evaluation of outputs
- quasi–Newton–method (Broyden)



Introduction

Objectives

- Efficiency by coupled simulations
- Consideration of root events
- Virtual Reality combined with mechanical simulation

Advantages

- Communication step size control :
 - application of proved integration methods with step size control
 - minimization of communication quantities
 - reduced computation time using modules with different dynamic properties
- Interaction with animation :
 - user situated inside application by 3D–projection
 - direct system analysis by parameter variations and force feedback

Open questions

- Approach for simulation and animation tools combination ?
- Approach for automatic communication step size control ?
- Software for implementation ?

CoSim2001, CS --> 5 / 21

Dynamic Analysis

Modular modelling of subsystems

Global system-structure

CoSim2001, CS --> 6 / 21

Dynamic Analysis

Modular Simulation

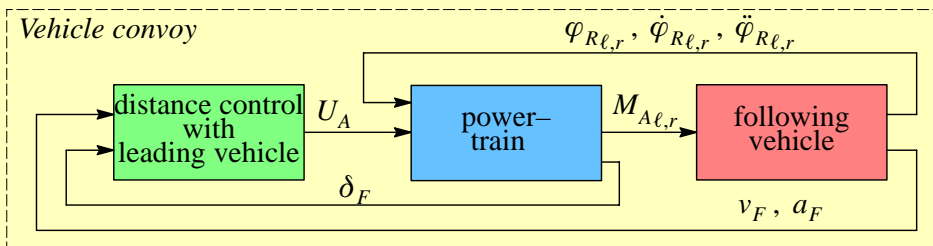
	subsystem 1	subsystem 2	subsystem 3
computation tool	MATLAB/SIMULINK	SIMPACK	NEWEUL/NEWMOS
operating system	Windows NT	UNIX/SOLARIS	LINUX
model	<i>state space</i>	<i>mbs</i>	<i>mbs, state space</i>
integration code	Shampine-Gordon Extrapol. 4. order	Shampine-Gordon Extrapol. 4. order	Dormand-Prince 5 Extrapol. 4. order

NEWMOS
Parallel Virtual Machine
Broyden
H = 1 ms



Dynamic Analysis

Example



Dynamic Analysis

Animation

- Position vectors
 - Rotation matrices
 - Geometrical informations
 - Color-, lighting-, texture-, perspective- informations
- } from simulation

- ☞ No interaction between viewer and system
- ☞ No change of parameters
- ☞ No feedback-effects on viewer

Demand

- Manipulation of mechanical properties
- Manual change of body constraints
- Force feedback
- Real-time representation

CoSim2001, CS --> 9 / 21

Dynamic Analysis

Virtual Reality

COVER PVM NEWMOS

Examples

CoSim2001, CS --> 10 / 21

Communication Step Size Control

Modular simulation

- efficiency problems of coupled simulators
- dynamic properties change of interacting coupled simulation

☞ Automatic communication step size control of coupled simulators

Approach

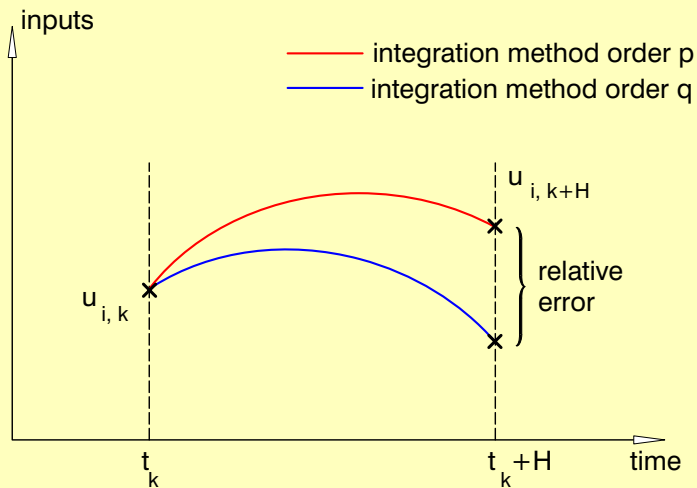
- Embedded formula
- Richardson extrapolation



Communication Step Size Control

Embedded formula

global time step H
of subsystem i



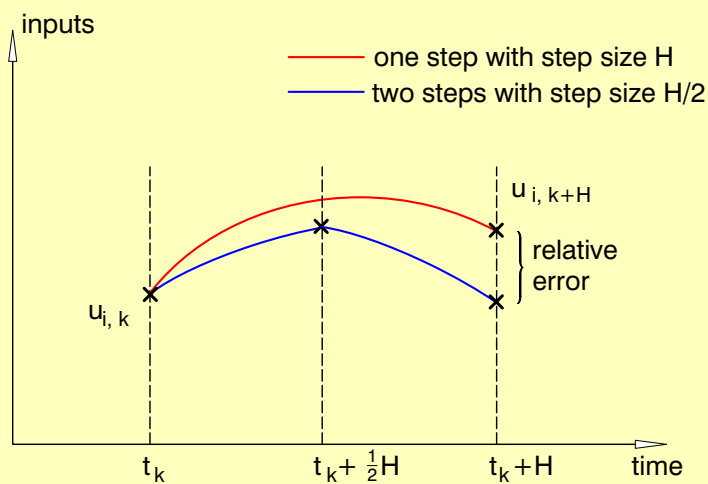
☞ Interference in coupled simulation tools necessary



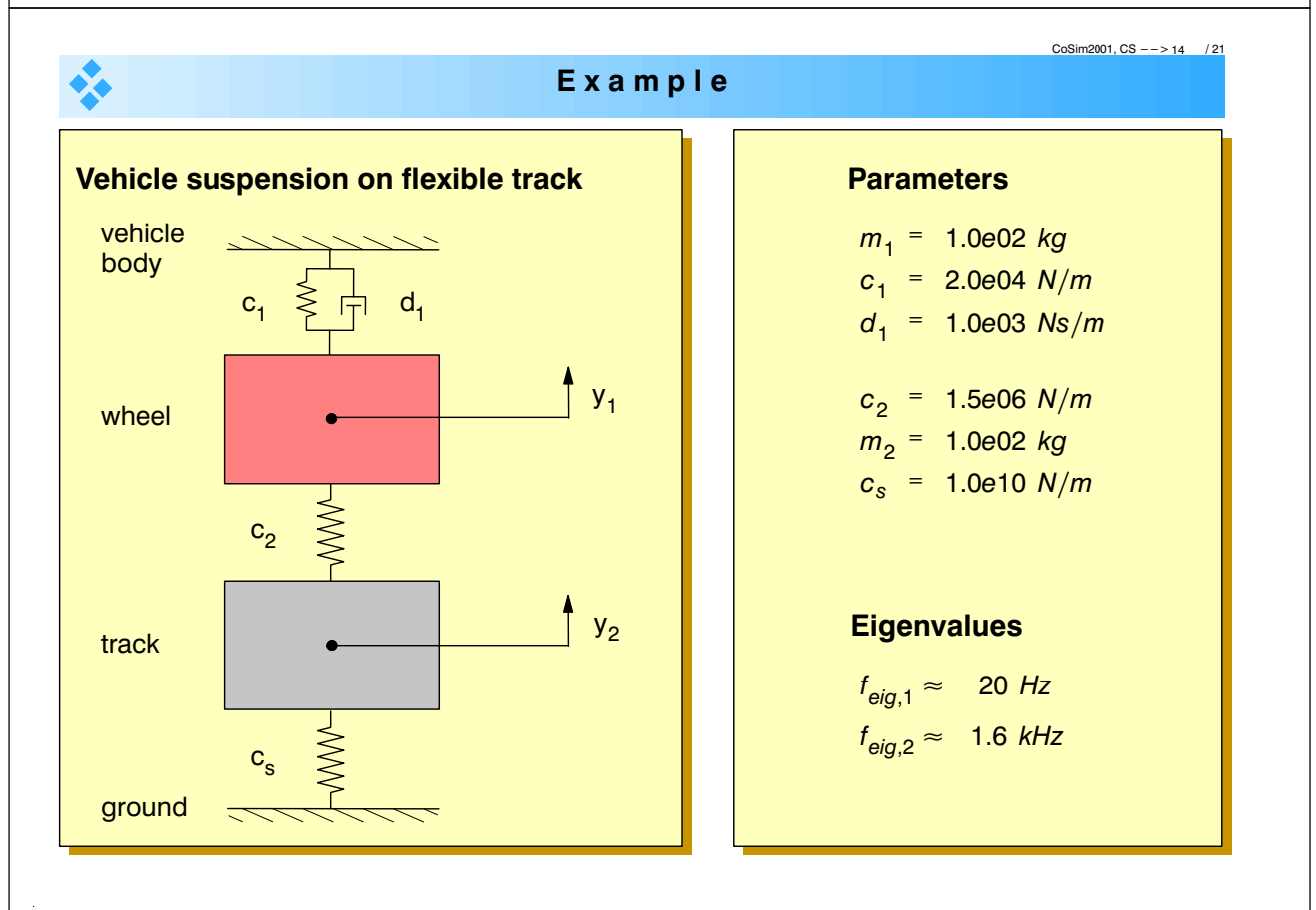
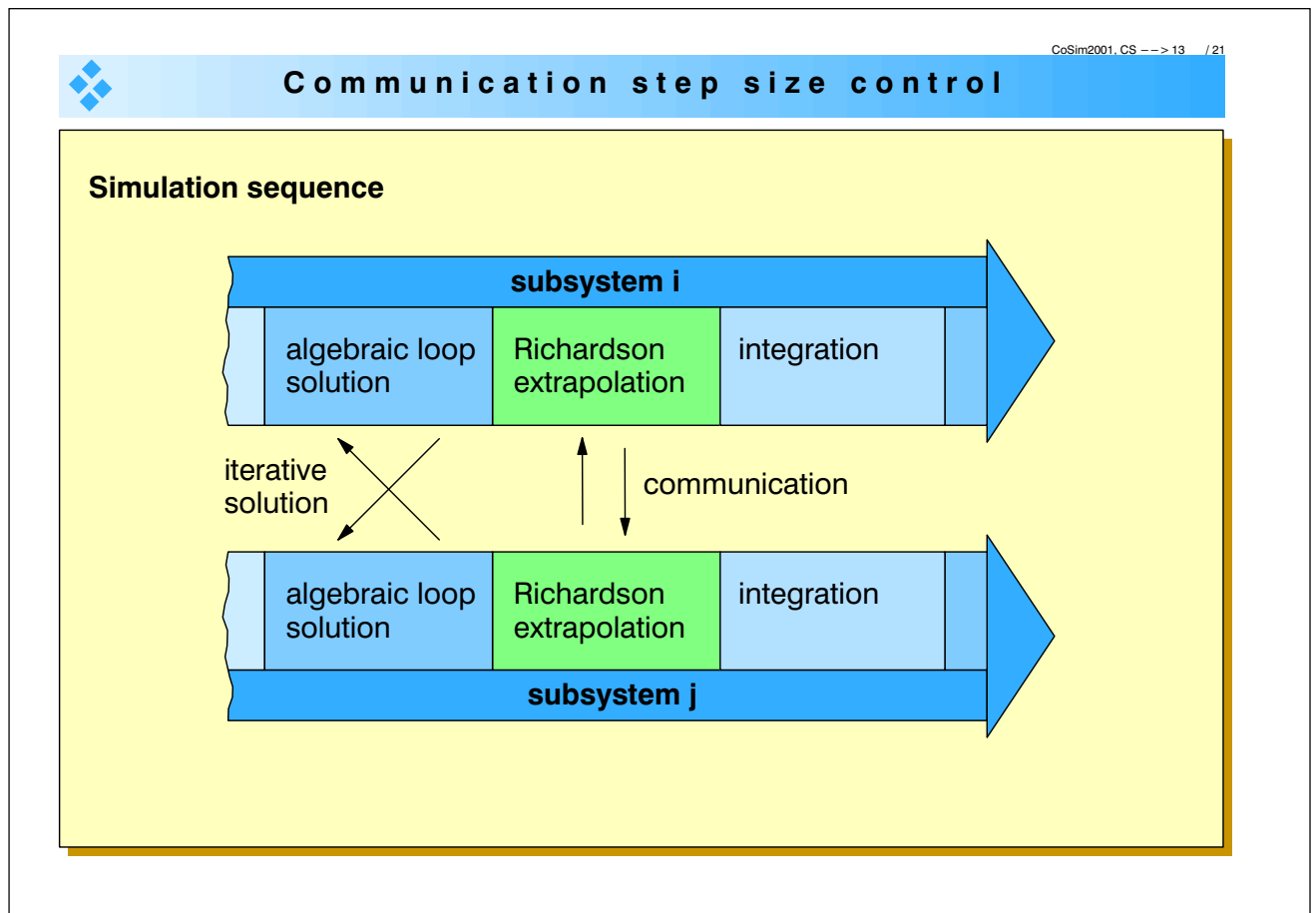
Communication Step Size Control

Richardson extrapolation

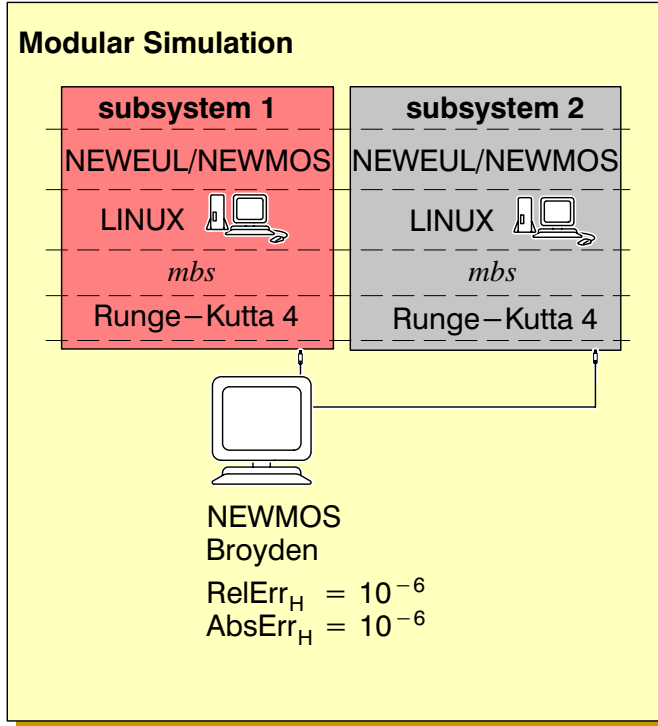
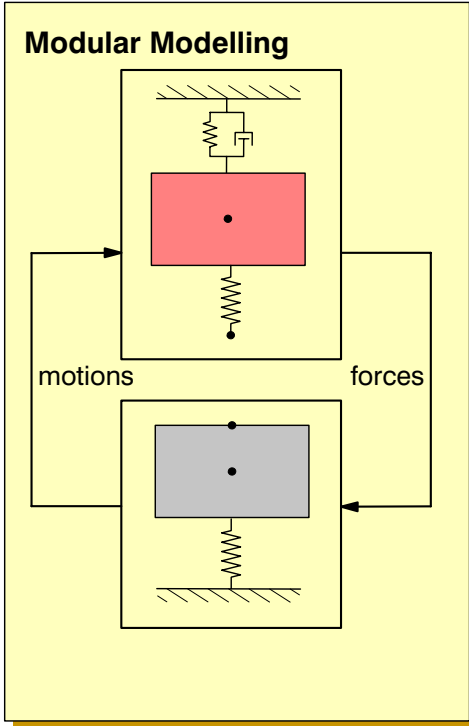
global time step H
of subsystem i



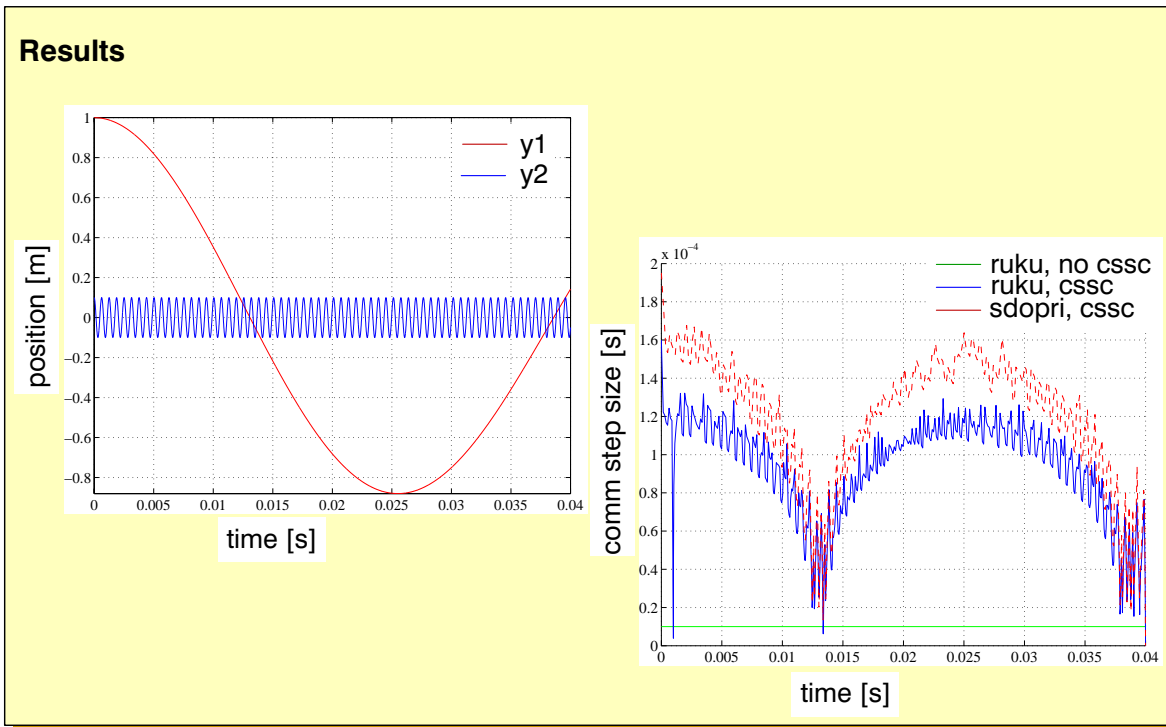
☞ Multiple computation and exchange of out- and inputs



Example



Example



CoSim2001, CS --> 17 / 21

Example

Double pendulum with elastic contact

Parameters

$m_1 = 1.0 \text{ kg}$
 $j_1 = 1.0 \text{ kgm}^2$
 $l_1 = 1.0 \text{ m}$

 $m_2 = 1.0 \text{ kg}$
 $j_2 = 1.0 \text{ kgm}^2$
 $l_2 = 1.0 \text{ m}$

 $c_c = 800 \text{ N/rad}$
 $d_c = 3000 \text{ Ns/rad}$

CoSim2001, CS --> 18 / 21

Example

Modular Modelling

Modular Simulation

subsystem 1	subsystem 2
NEWEUL/NEWMOS	NEWEUL/NEWMOS
LINUX	LINUX
<i>mbs</i>	<i>mbs</i>
LSODAR	Runge-Kutta 4

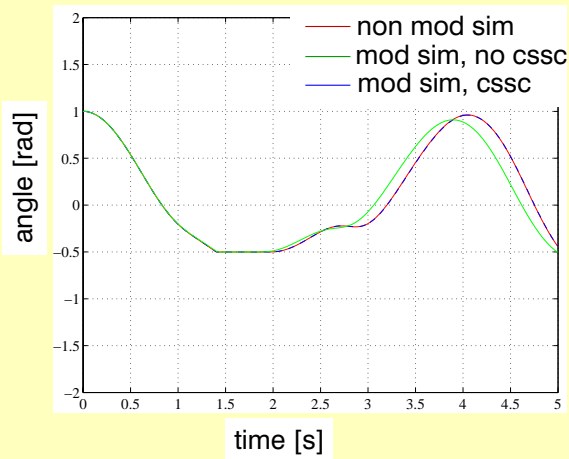
NEWMOS
 Broyden
 $RelErr_H = 10^{-7}$
 $AbsErr_H = 10^{-7}$



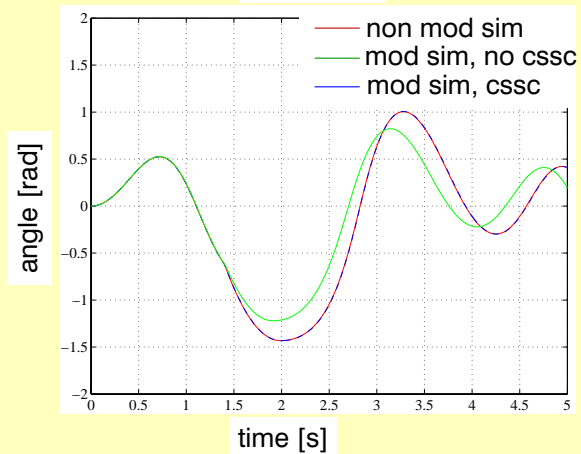
Example

Results

pendulum 1



pendulum 2



Conclusion

Task

- Improve efficiency of modular simulations
- Consider special events in any subsystem
- Combine mechanical simulation with Virtual Reality

Approach

- Modular description and simulation of multibody systems
- Richardson extrapolation for communication step size control
- Exchange informations of simulation and animation by PVM

Example

- Modular modelling and simulation of stiff suspension track system
- Modular modelling and simulation of double pendulum with one side constraint both with NEWMOS (LINUX)

Perspective

- Implementation testing by simulation tool NEWMOS and SIMULINK



Conclusion

CoSim2001_CS --> 21 / 21

Literature

Hairer, E.; Norsett, S.P.; Wanner, G.: *Solving Ordinary Differential Equations I*. Berlin: Springer, 1993.

Kübler, R.: *Modulare Modellierung und Simulation mechatronischer Systeme*. Fortschritt-Berichte VDI, Reihe 20, Nr. 327. Düsseldorf: VDI, 2000.

Rükgauer, A.: *Modulare Simulation mechatronischer Systeme mit Anwendung in der Fahrzeugdynamik*. Fortschritt-Berichte VDI, Reihe 20, Nr. 248. Düsseldorf: VDI, 1997.

Strehmel, K.; Weiner, R.: *Numerik gewöhnlicher Differentialgleichungen*. Stuttgart: Teubner, 1995.

Coupled simulation in blockoriented network analysis

C. Clauß, P. Schwarz

{christoph.clauss,peter.schwarz}@eas.iis.fhg.de

FhG IIS/EAS Dresden

Zeunerstr. 38, D – 01069 Dresden, Germany

The simulation of large electronic circuits on transistor level is both time and memory consuming. However, extremely large electronic circuits are mostly composed by subcircuits. Therefore, a hierarchical simulation algorithm basing on the subcircuit structure offers the opportunity to handle larger circuits.

In mathematical terms the problem to be solved is a partitioned ODE/DAE like this

$$\begin{aligned} 0 &= H(x_1, x_2, \dots, x_n, y) \\ 0 &= F_1(x_1, y) \\ &\dots \\ 0 &= F_n(x_n, y), \end{aligned}$$

if H, F include differentiation with respect to time, x, y are time dependent variables which are to be computed.

Important solution methods are relaxation methods or Newton type methods. If the F_i -equations (subcircuits) are able to be solved for x_i , after one step of the H -equation the F_i -equations can be solved simultaneously. Moreover, each F_i -equation can be solved using its own tolerances and stepsize, which is more economical than the usage of the smallest stepsize to each equation. Furthermore, each F_i -equation can be solved by its own simulation tool.

Via the H -equation the F_i -equations are connected together. Depending on the simplicity of the H -equation and on the solution methods used different methods of simulator coupling can be derived. In the paper the blockoriented network analysis is discussed, which uses the Jacobians of the F_i -equation for the solution of the H -equation. Therefore, it is a two-stage Newton method which offers better properties of convergence than relaxation methods. The method becomes practicable because the number of the pin variables y is usually low compared with the number of internal variables x_i .

Basing on the experiences of the blockoriented network analysis method a coupling modul is suggested which offers different steps of functionality:

- statistical analysis
- supervision of convergence property
- calculation of the Jacobian, eigenvalue check
- Newton’s method

The module and its usefulness should be discussed in view of the coupling of two and more than two simulators.

Coupled Simulation - in Blockoriented Network Analysis

Peter Schwarz
Christoph Clauß

Fraunhofer-Institut für Integrierte Schaltungen
Außenstelle Entwurfsautomatisierung EAS Dresden

Copyright © 2001 Fraunhofer-Gesellschaft

P. Schwarz

ASIM 2001- 15, Symposium Simulationstechnik, Paderborn 11-14. Sept. 2001



Contents

Introduction

Newton's method for structured equations

Simple example network

Blockoriented network analysis

Example

Summary

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

2



Introduction



Newton's method for structured equations

Simple example network

Blockoriented network analysis

Example

Summary

$$F(x) = 0$$

Newton's Method

1° $j = 0$, guess x^0

2° solve linear system:

$$\partial F * (x^{j+1} - x^j) = -F(x^j)$$

$$\text{---> } x^{j+1}$$

3° if convergent, then stop,
else $j := j+1$, goto 2°

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

5

Structured Equations

$$n > 1$$

$$H(x_1, \dots, x_n, y) = 0$$

$$F_i(x_i, y) = 0 \quad i = 1(1)n$$

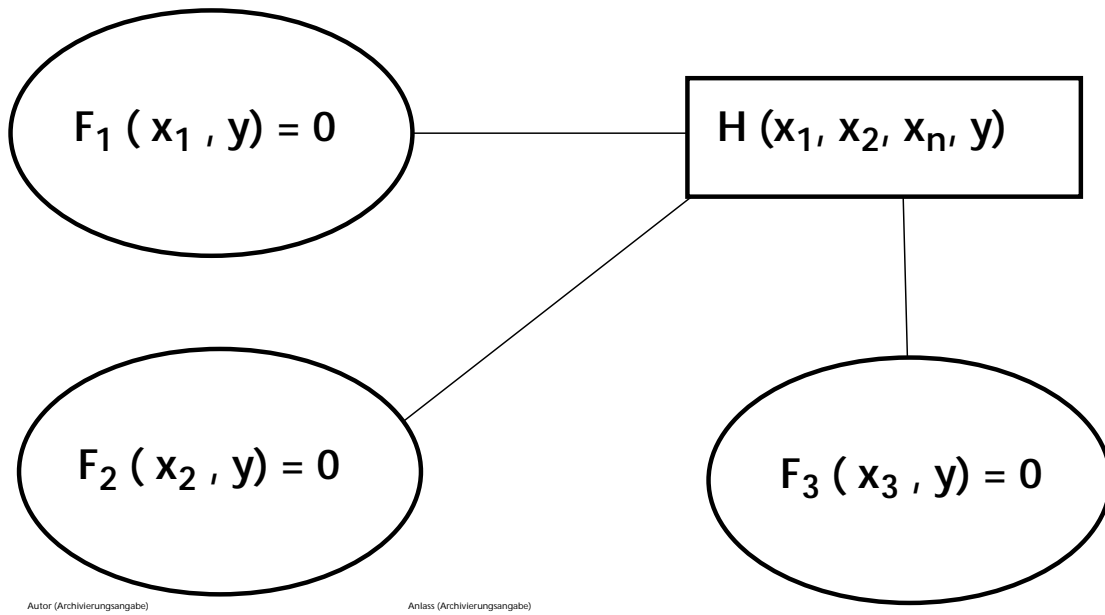
Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

6

Structured Equations



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

7



Fraunhofer
Institut
Integrierte Schaltungen

Newton's method for structured equations

1° $j = 0$, guess x_1^0, \dots, x_n^0, y^0

2° solve linear system:

$$\sum_{i=1}^n \partial_i H^*(x_i^{j+1} - x_i^j) + \partial_{n+1} H^*(y^{j+1} - y^j) = -H(x_1^j, \dots, x_n^j, y^j)$$

for $i=1(1)n$:

$$\partial_1 F_i^*(x_i^{j+1} - x_i^j) + \partial_2 F_i^*(y^{j+1} - y^j) = -F_i(x_i^j, y^j)$$

3° if convergent, then stop, else $j := j+1$, goto 2°

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

8



Fraunhofer
Institut
Integrierte Schaltungen

Newton's method for structured equations

1° $j = 0$, guess x_1^0, \dots, x_n^0, y^0

2° solve linear system: $\rightarrow x_i^{j+1}, y^{j+1}$

$$\sum_{i=1}^n \partial_i H^*(x_i^{j+1} - x_i^j) + \partial_{n+1} H^*(y^{j+1} - y^j) = -H(x_1^j, \dots, x_n^j, y^j)$$

for $i=1(1)n$:

$$\partial_1 F_i^*(x_i^{j+1} - x_i^j) + \partial_2 F_i^*(y^{j+1} - y^j) = -F_i(x_i^j, y^j)$$

3° if convergent, then stop, else $j := j+1$, goto 2°

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

9



Fraunhofer Institut
Integrierte Schaltungen

Newton's method for structured equations

1° $j = 0$, guess x_1^0, \dots, x_n^0, y^0

2° solve linear system:

$$\sum_{i=1}^n \partial_i H^*(x_i^{j+1} - x_i^j) + \partial_{n+1} H^*(y^{j+1} - y^j) = -H(x_1^j, \dots, x_n^j, y^j)$$

for $i=1(1)n$:

$$\partial_1 F_i^*(x_i^{j+1} - x_i^j) + \partial_2 F_i^*(y^{j+1} - y^j) = -F_i(x_i^j, y^j)$$

regular ?

3° if convergent, then stop, else $j := j+1$, goto 2°

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

10



Fraunhofer Institut
Integrierte Schaltungen

Newton's method for structured equations

1° $j = 0$, guess x_1^0, \dots, x_n^0, y^0

2° solve linear system

$$\begin{aligned}
 & - \sum_{i=1}^n \partial_i H^* (\partial_1 F_i)^{-1} * (F_i(x_i^j, y^j) + \partial_2 F_i * (y^{j+1} - y^j)) * (y^{j+1} - y^j) \\
 & + \partial_{n+1} H^* (y^{j+1} - y^j) = - H(x_1^j, \dots, x_n^j, y^j)
 \end{aligned}$$

for $i=1(1)n$:

$$\partial_1 F_i * (x_i^{j+1} - x_i^j) + \partial_2 F_i * (y^{j+1} - y^j) = - F_i(x_i^j, y^j)$$

3° if convergent, then stop, else $j := j+1$, goto 2°



Modified Newton's method for structured equations

1° $j = 0$, guess y^0

2° solve $F_i(x_i^0, y^0) = 0, i = 1(1)n$ ----> x_i^0

3° solve linear system

$$\begin{aligned}
 & - \sum_{i=1}^n \partial_i H^* (\partial_1 F_i)^{-1} * (F_i(x_i^j, y^j) + \partial_2 F_i * (y^{j+1} - y^j)) * (y^{j+1} - y^j) \\
 & + \partial_{n+1} H^* (y^{j+1} - y^j) = - H(x_1^j, \dots, x_n^j, y^j) \\
 & \text{----> } y^{j+1}
 \end{aligned}$$

4° for $i=1(1)n$ solve nonlinear system:

$$F_i(x_i^{j+1}, y^{j+1}) = 0 \quad \text{----> } x_i^{j+1}$$

5° if convergent, then stop, else $j := j+1$, goto 2°



Modified Newton's method for structured equations

1^o $j = 0$, guess y^0

2^o solve $F_i(x_i^0, y^0) = 0, i = 1(1)n$ ----> x_i^0

3^o solve linear system

$$-\sum_{i=1}^n \partial_i H^* (\partial_1 F_i)^{-1} * \cancel{F_i(x_i^j, y^j)} + \partial_2 F_i^* (y^{j+1} - y^j) * (y^{j+1} - y^j) + \partial_{n+1} H^* (y^{j+1} - y^j) = -H(x_1^j, \dots, x_n^j, y^j)$$

----> y^{j+1}

4^o for $i=1(1)n$ solve nonlinear system:

$$F_i(x_i^{j+1}, y^{j+1}) = 0$$

----> x_i^{j+1}

5^o if convergent, then stop, else $j := j+1$, goto 3^o

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

13



Fraunhofer
Institut
Integrierte Schaltungen

Modified Newton's method - after reordering

1^o $j = 0$, guess y^0

2^o for $i=1(1)n$ solve nonlinear system:

$$F_i(x_i^j, y^j) = 0$$

----> x_i^{j+1}

3^o solve linear system:

$$\left(\sum_{i=1}^n \partial_i H^* (\partial_1 F_i)^{-1} * \partial_2 F_i - \partial_{n+1} H\right) * (y^{j+1} - y^j) = H(x_1^j, \dots, x_n^j, y^j)$$

----> y^{j+1}

4^o if convergent, then stop, else $j := j+1$, goto 2^o

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

14



Fraunhofer
Institut
Integrierte Schaltungen

Introduction

Newton's method for structured equations



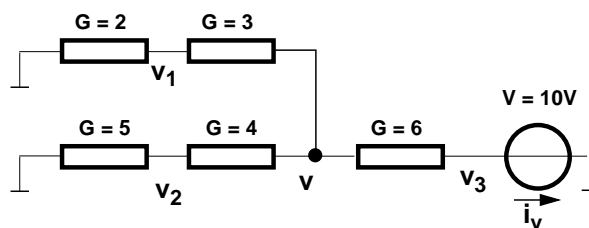
Simple example network

Blockoriented network analysis

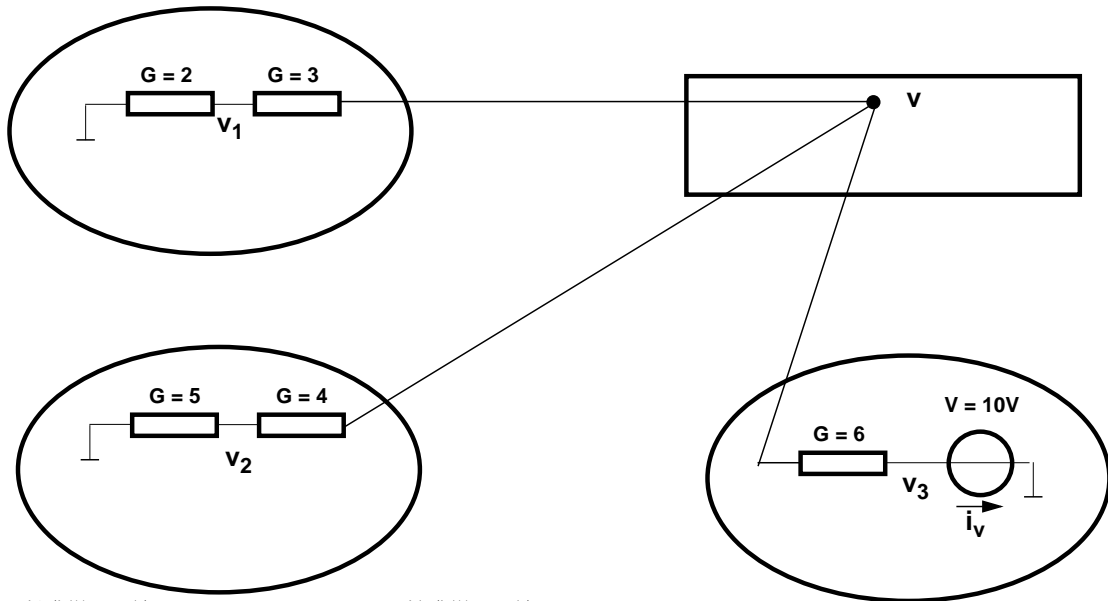
Example

Summary

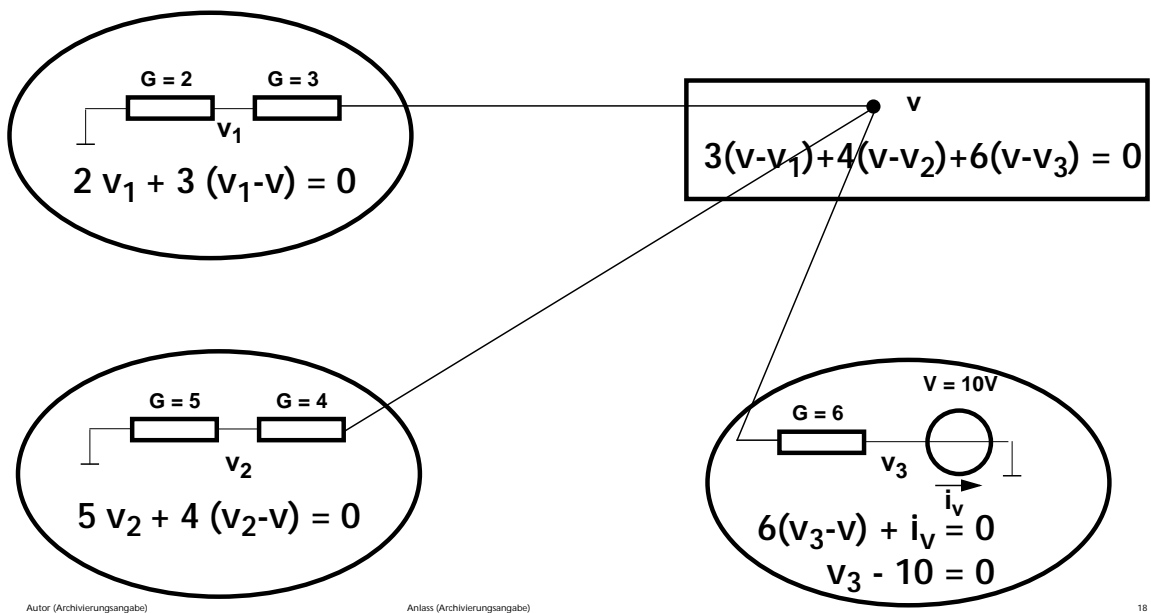
Simple example network



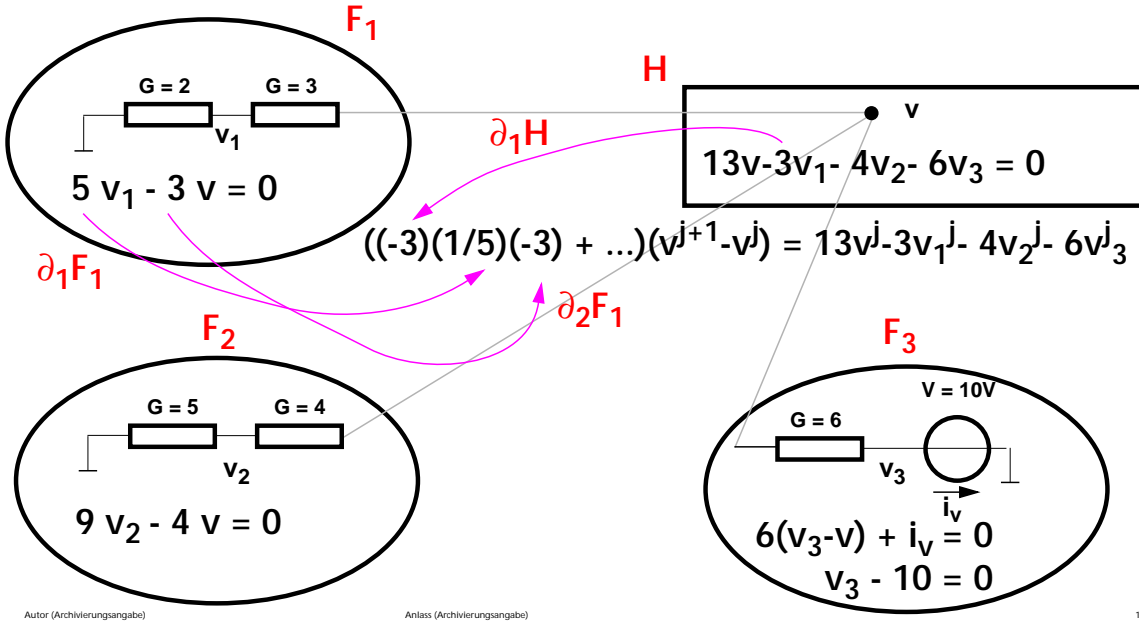
Simple network example - partitioned



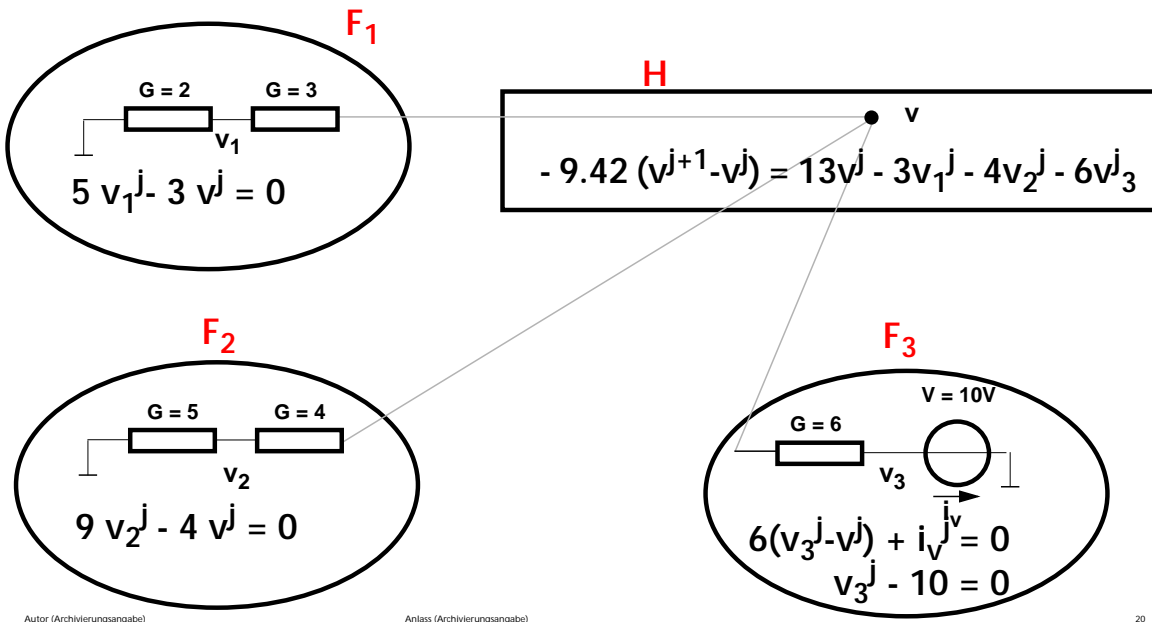
Simple network example - equations



Simple network example - Modified Newton equations



Simple network example - Modified Newton equations



Iteration: 1^o j = 0, guess $v^0 = 0$

F₁

$5 v_1^0 - 3 v^0 = 0$

H

$v = 0$

$- 9.42 (v^1 - v^0) = 13v^0 - 3v_1^0 - 4v_2^0 - 6v_3^0$

F₂

$9 v_2^0 - 4 v^0 = 0$

F₃

$6(v_3^0 - v^0) + i_V^0 = 0$

$v_3^0 - 10 = 0$

Copyright © 2001 Fraunhofer-Gesellschaft
 Autor (Archivierungsangabe) Anlass (Archivierungsangabe) 21

Iteration: 2^o for i=1(1)n solve nonlinear system: $F_i = 0$

F₁

$5 v_1^0 - 3 v^0 = 0$

H

$v = 0$

$- 9.42 (v^1 - v^0) = 13v^0 - 3v_1^0 - 4v_2^0 - 6v_3^0$

F₂

$9 v_2^0 - 4 v^0 = 0$

F₃

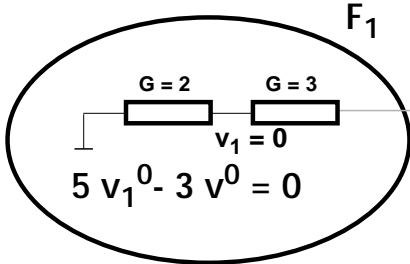
$6(v_3^0 - v^0) + i_V^0 = 0$

$v_3^0 - 10 = 0$

Copyright © 2001 Fraunhofer-Gesellschaft
 Autor (Archivierungsangabe) Anlass (Archivierungsangabe) 22

Iteration:

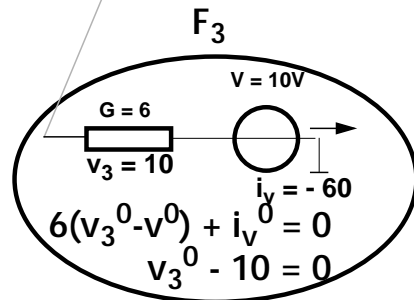
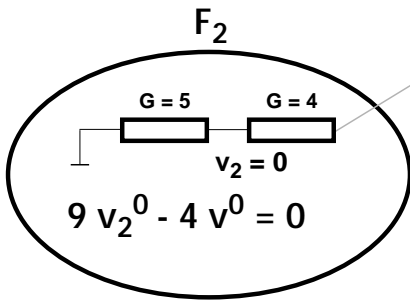
3^o solve linear system 4^o j = 1



H

$v = 6.37$

$- 9.42 (v^1 - v^0) = 13v^0 - 3v_1^0 - 4v_2^0 - 6v_3^0$



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

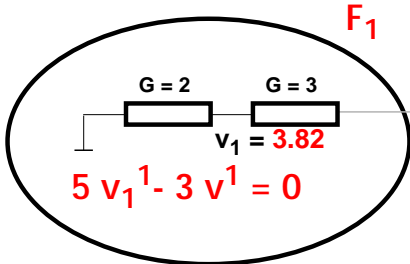
23



Fraunhofer Institut Integrierte Schaltungen

Iteration:

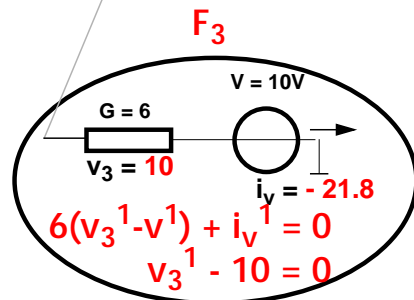
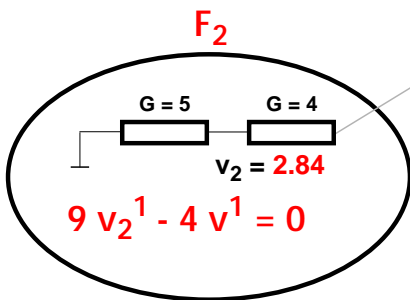
2^o for i=1(1)n solve nonlinear system: F_i = 0



H

$v = 6.37$

$- 9.42 (v^2 - v^1) = 13v^1 - 3v_1^1 - 4v_2^1 - 6v_3^1$



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

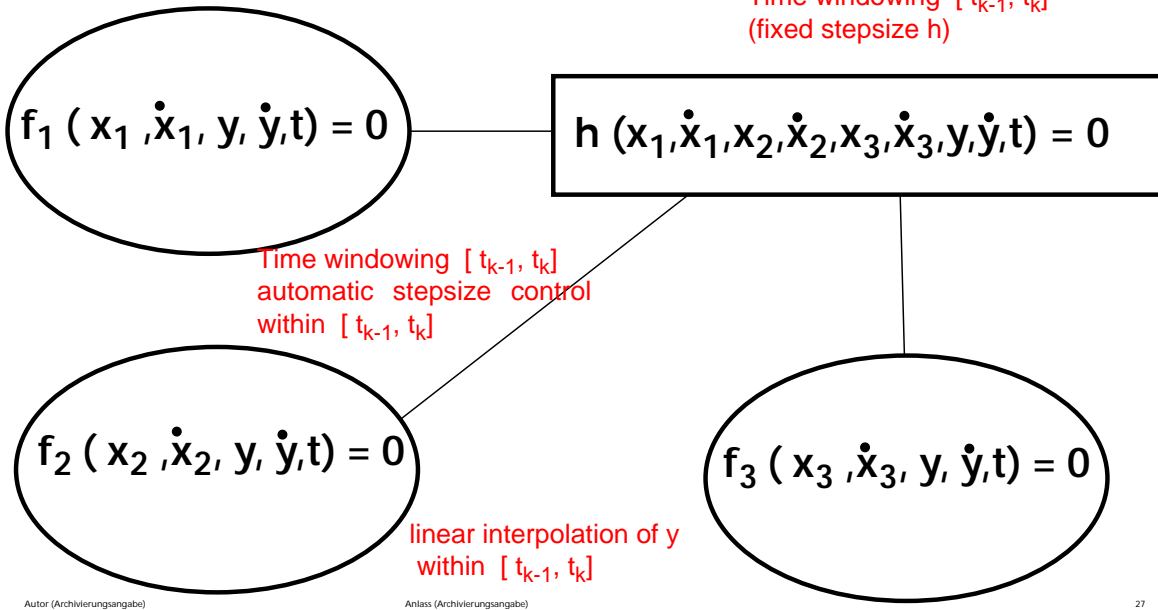
Anlass (Archivierungsangabe)

24



Fraunhofer Institut Integrierte Schaltungen

Blockoriented network analysis



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

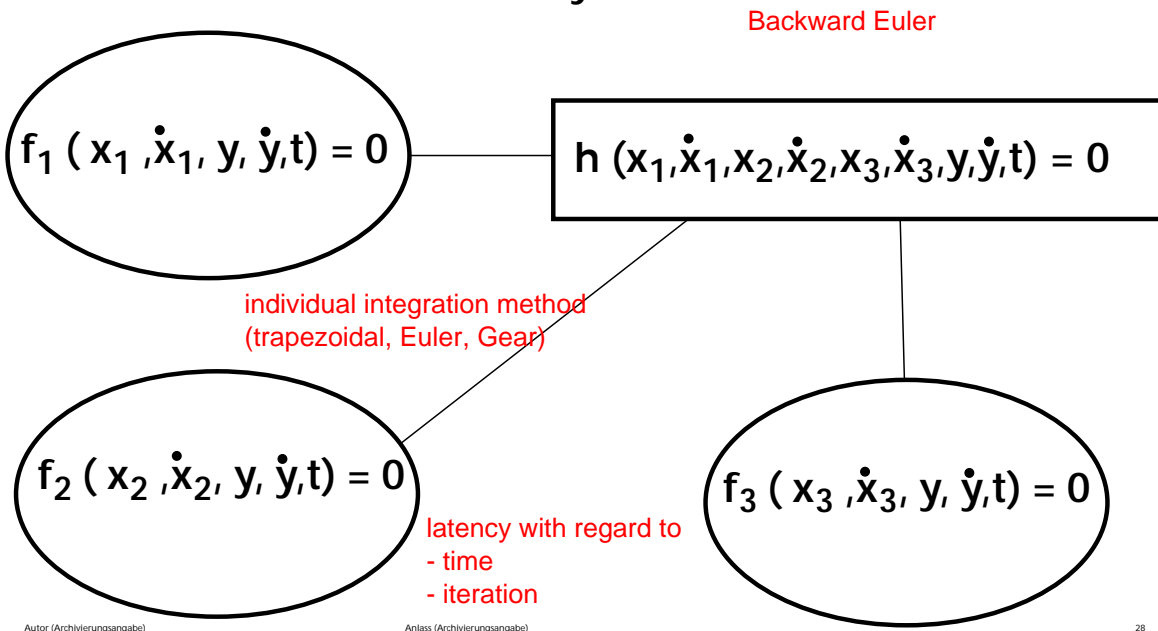
Anlass (Archivierungsangabe)

27



Fraunhofer Institut Integrierte Schaltungen

Blockoriented network analysis



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

28



Fraunhofer Institut Integrierte Schaltungen

Blockoriented network analysis within $[t_{k-1}, t_k]$

1^o $j = 0$, guess $y^0(t_k)$

$$2^o \ y^j(t) = \alpha y(t_{k-1}) + (1 - \alpha) y^j(t_k), \quad \alpha = (t - t_{k-1}) / (t_k - t_{k-1})$$

for $i=1(1)n$ solve: $f_i(x_i^j(t), \dot{x}_i^j(t), y^j(t), \dot{y}^j(t), t) = 0 \rightarrow x_i^j(t)$

$\rightarrow x_i^j(t_k), \ d x_i^j(t) / d y^j(t) | t_k$

3^o : calculate $y^{j+1}(t_k)$ from

$$\left(\sum_{i=1}^n \partial_i \tilde{h} * (d x_i^j(t) / d y^j(t) | t_k) - \partial_{n+1} \tilde{h} \right) * (y^{j+1}(t_k) - y^j(t_k)) = h(x_1^j(t_k), \dots, x_n^j(t_k), y^j(t_k), t_k) \rightarrow y^{j+1}(t_k)$$

4^o if convergent, then stop, else $j := j+1$, goto 2^o

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

29

Blockoriented network analysis - Jacobian calculation

$$f_1(x_1, \dot{x}_1, y, \dot{y}, t) = 0$$

$$h(x_1, \dot{x}_1, x_2, \dot{x}_2, x_3, \dot{x}_3, y, \dot{y}, t) = 0$$

$$(\partial_1 F)^{-1} * \partial_2 F :$$

$$(d f(x(t), \dot{x}(t), y(t), \dot{y}(t), t) / dx(t))^{-1} * d f(x(t), \dot{x}(t), y(t), \dot{y}(t), t) / dy(t) | t_k$$

$$(d \tilde{f}(x(t), y(t), t) / dx(t))^{-1} * d \tilde{f}(x(t), y(t), t) / dy(t) | t_k$$

$$(d \tilde{f}(x(t_k), y(t_k), t_k) / dx(t_k))^{-1} * d \tilde{f}(x(t_k), y(t_k), t_k) / dy(t_k)$$

internal available Jacobian

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

30

Introduction

Newton's method for structured equations

Simple example network

Blockoriented network analysis



Example

Summary

Example

Memory circuit examples

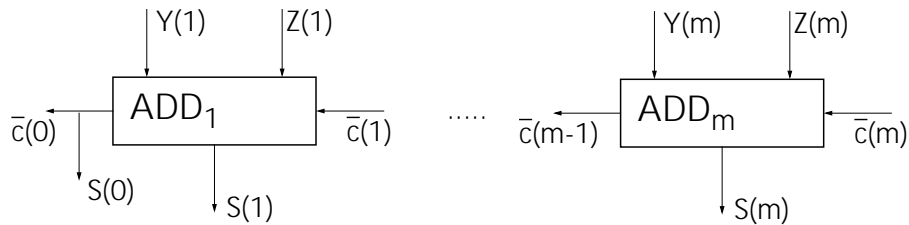
Nodes	Components	CPU/min without partitioning	CPU/min blockoriented method
113	375	80	120
414	1584	286	90
480	1000	350	142

small circuits - overhead

large circuits - speed up

Example

Adder circuit chain



25 MOS transistors

5 single adders - one subcircuit

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

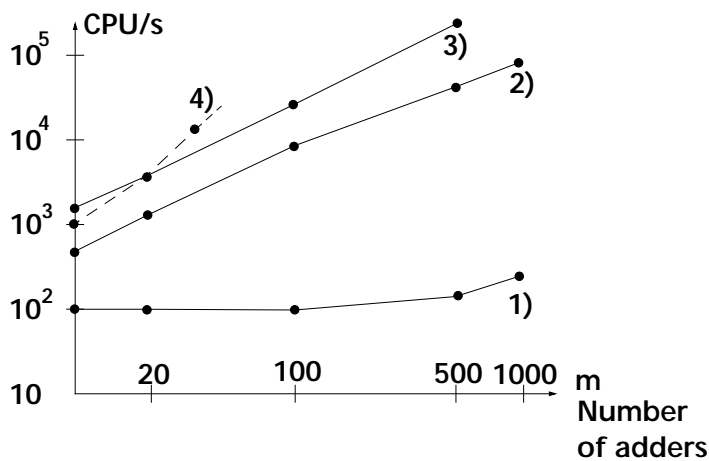
33



Fraunhofer Institut Integrierte Schaltungen

Example

Adder circuit chain



1) Z = 0000 ... 00
Y = 0000 ... 01

2) Z = 1010 ... 10
Y = 0101 ... 01

3) Z = 1111 ... 11
Y = 0000 ... 01

4) without partitioning

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

34



Fraunhofer Institut Integrierte Schaltungen

Example

Iterations

typical: 2 - 3 main iterations

more iterations:

- very small stepsize during subcircuit integration
- nonlinear behaviour of subcircuits
- not correct balanced tolerances

Partitioning: optimal size of subcircuits

Introduction

Newton's method for structured equations

Simple example network

Blockoriented network analysis

Example

→ Summary

Summary

- Modified Newton's method for structured equations
- Application to hierarchical structured electrical networks
 - > Blockoriented Network Analysis
- Experience:
 - useful for large circuits with latency
 - feedbacks cause iterations
 - subcircuit simulators:
 - repetition of simulation of time intervals
 - calculation of output and Jacobians
 - optimal tuning of tolerances necessary
 - optimal partitioning necessary

Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

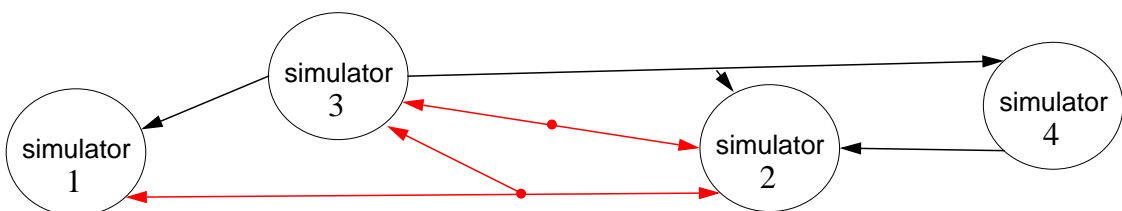
Anlass (Archivierungsangabe)

37

Summary

Conservative equations

Variable	difference variable (e.g. node voltage)
Equation	sum of flows = zero (Kirchhoff's current law)



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

38

Summary

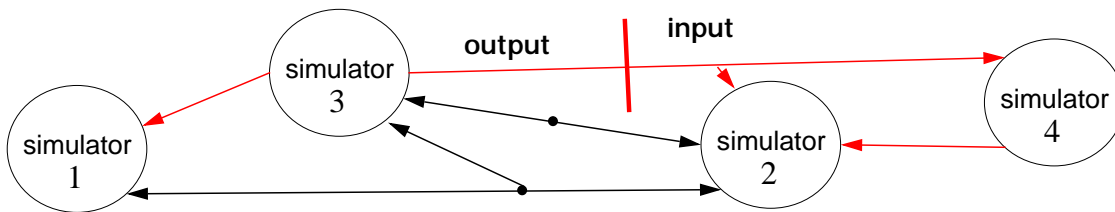
Nonconservative equations

Variable

input

Equation

output - input = zero



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

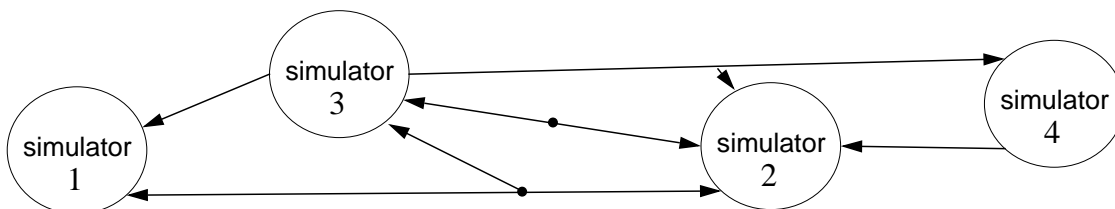
39



Fraunhofer
Institut
Integrierte Schaltungen

Summary

Suggestion: Master simulator



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

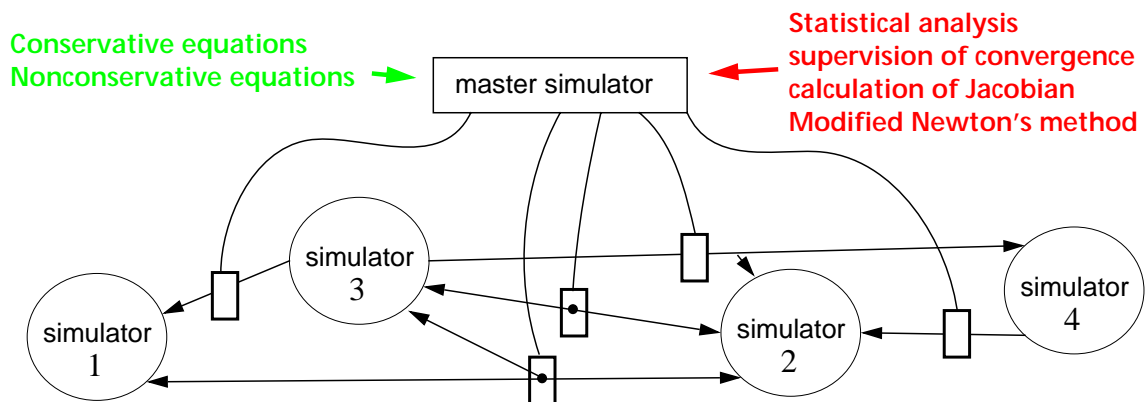
40



Fraunhofer
Institut
Integrierte Schaltungen

Summary

Suggestion: Master simulator



Copyright © 2001 Fraunhofer-Gesellschaft

Autor (Archivierungsangabe)

Anlass (Archivierungsangabe)

41



Fraunhofer
Institut
Integrierte Schaltungen

References

- [1] Clauß, C.: Blockorientierte Netzwerkanalyse - ein Ansatz zur Simulation strukturierter elektrischer Netzwerke. GME/ITG-Diskussionssitzung. 10./11. Oktober 1991, Paderborn.
- [2] Clauß, C.; Reitz, S.; Schwarz, P.: Simulation mechanisch-elektrischer Wechselwirkungen am Beispiel eines sensorischen Mikrosystems. SIM'2000 Dresdner Tagung Simulation im Maschinenbau, Dresden, 24./25. Februar 2000, 183-196
- [3] Elst, G.; Poenisch, G.: On Two-Level Newton Methods for the Analysis of Large Scale nonlinear networks. Proc. ECCTD'85, Prag, 1985, 165-169
- [4] Schulte, S.: Modulare und hierarchische Simulation gekoppelter Probleme. Fortschrittberichte VDI, Reihe 20: Rechnerunterstützte Verfahren, Nr. 271, VDI Verlag GmbH, Düsseldorf, 1998

Co-simulation of complete vehicle models with real-time performance

A. Suescun, J. González, S. Ausejo, J.T. Celigüeta

asuescun@ceit.es

**CEIT – Centro de Estudios e Investigaciones Técnicas de Gipuzkoa
M. Lardizábal, 15, 20018 San Sebastián, Spain**

This presentation describes the use of co-simulation techniques for the simulation of complete vehicle models, reaching real time performance. This work is a continuation of the work presented during the 1st workshop on co-simulation, in October 1999, and has been carried out mainly in the frame of the ODECOMS project.

The vehicle model used corresponds to a Peugeot 806 van. It includes a detailed description of the car body and the suspensions, as well as models of the tyres, and the driveline (clutch, gear box, ...). A model of the ABS digital controller and a simplified model of the hydraulic braking system are also included.

The models of the suspensions are considered in two different ways: first is a simplified version of the suspensions, by using the surface response method, where the exact kinematic and kinetostatic behaviour are substituted by polynomial approximations of the cubic spline type. The second method uses an exact representation of the suspension kinematics, including the closed loops of the McPherson-type front suspension.

In both cases the motion equation of the car body and the 4 suspensions have been generated symbolically, by means of a software tool, called SAMBS, that allows for the description of mechanisms using a high level language based on the use of C++ objects. With this description the software produces automatically the motion equations in the form of ODEs.

The simulation of the complete model has been carried out by using an in-house developed software package for co-simulation. This software allows for the co-simulation of complex systems made up of interconnected blocks. Each block represents a dynamic system, the deals with its own motion differential equations, by means of an integrator of ODE or DAE self-contained in the block. The software provides the connections between the inputs and outputs of the different blocks, the time simulation loop and the input-output management.

For comparative purposes, some simulations have been also carried out by using the Matlab/Simulink software, by implementing a co-simulation within it.

The presentation contains a detailed description of the models, the co-simulation environment and results of different manoeuvres.



Co-simulation of complete vehicle models with real-time performance

A. Suescun*, J. González-Luna, S. Ausejo, J.T. Celigüeta

International Workshop

"Co-Simulation for Mechatronic Systems"

Stuttgart (Germany), October 11, 2001

***Contact: Dr. Angel SUESCUN**
E-mail: asuescun@ceit.es
Address: Pº. de Manuel Lardizabal, 15
20018 San Sebastian (SPAIN)
Phone: +34 943 212 800
Fax: +34 943 213 076
URL: <http://www.ceit.es>

CEIT
Centro de Estudios e Investigaciones Técnicas de Gipuzkoa
Dept. of Applied Mechanics
Simulation Area

Introduction

- ⊕ Continuation of the work presented in 1st. Workshop in 1999 at DLR Braunschweig (Germany)
- ⊕ Based on the latest results obtained in ODECOMS project:
 - EC funded project (Brite Euram BE-97.4123)
- ⊕ Goal: to achieve **real time performance** when simulating **accurately** complex vehicle models (chassis, suspension, driveline, brake system, tyres)
 - Symbolic generation of motion equations
 - Co-simulation strategy
 - Local simplifications



Outline

◆ Overview of ODECOMS

- Strategy
 - ◆ Software SAMBS (Symbolic Analysis for MBS)
 - ◆ Simplifications
 - ◆ Co-simulation
- Results

◆ Beyond ODECOMS

- Further simplifications
- No simplifications
- Accuracy

◆ Recall of conclusions



Dept. of Applied Mechanics
Simulation Area

Intl. Workshop "Co-Simulation for Mechatronic Systems"
Stuttgart (Germany), October 11, 2001

-3-

ODECOMS Overview

◆ ODECOMS

- Open Design Environment for Controlled Mechatronic Systems
- Brite-Euram Project BE 97-4123
- Start Date: February 1998
- End Date: February 2001 (Duration: 36 months)

◆ Consortium

- PSA Peugeot Citroën, Robert Bosch, Centro Ricerche FIAT
- ETAS, Imagine, CEIT



Dept. of Applied Mechanics
Simulation Area

Intl. Workshop "Co-Simulation for Mechatronic Systems"
Stuttgart (Germany), October 11, 2001

-4-

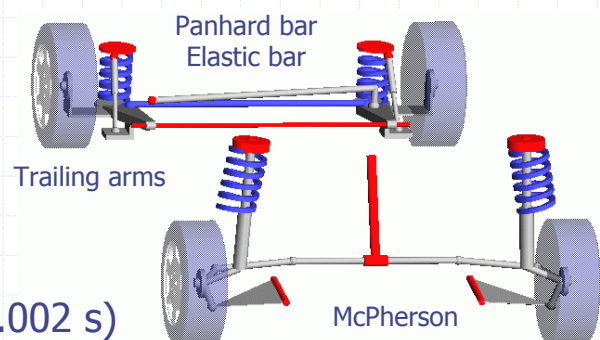
ODECOMS Main objective

- ⊕ Enhance ASCET capabilities for simulating mechatronic systems with real-time performance for control design
 - To import complex RT mechanical models
 - To import complex RT hydraulic models
 - Setup of a mechatronic model with RT performance and HIL
- ⊕ CEIT approach to improve efficiency of mechanical models:
 - Symbolic generation of motion equations (SAMBS)
 - Local simplifications
 - Co-simulation strategy (solver embedding)
- ⊕ Case study: P806 + ABS

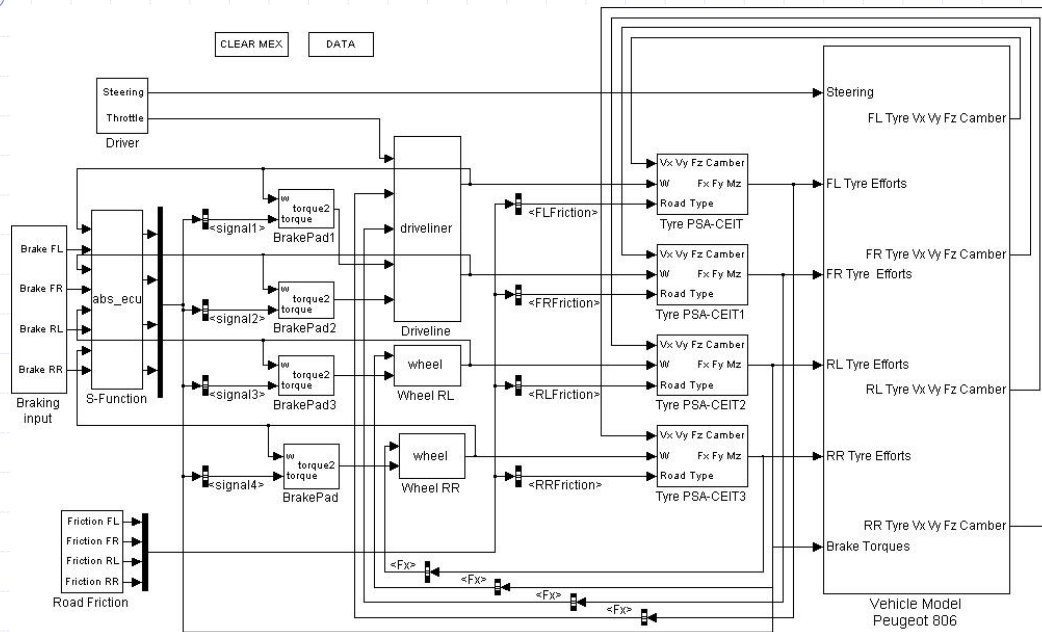


Case-study: Peugeot 806 + ABS

- ⊕ P806/Evasion/Ulysse/Z
- ⊕ Data provided by CRF and PSA
- ⊕ DOF: 10 (chassis) + 4 (wheels)
- ⊕ Steering as an input
- ⊕ Driveline
 - Engine, Gear box
 - Rigid shafts, Differential
- ⊕ Tyres (Pacejka)
- ⊕ Brake pads
- ⊕ ABS digital controller (0.002 s)



Vehicle model: block diagram



Simulation details

- ⊕ Two test environments:
 - Simulink R12 (MATLAB 6.0 / SIMULINK 4.0)
 - In-house standalone co-simulation software (C++)
- ⊕ Fixed step integrators and time-steps (same for communication and integration tasks)
 - Euler and Runge-Kutta 4 (RK4)
 - $h = 0.001$ s, $h = 0.002$ s
- ⊕ Various PC platforms
 - Pentium III – 600 Mhz
 - AMD K7 – 1,2 GHz
- ⊕ Manoeuvre: braking in straight line



SAMBS: Symbolic Analysis of MBS

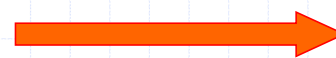
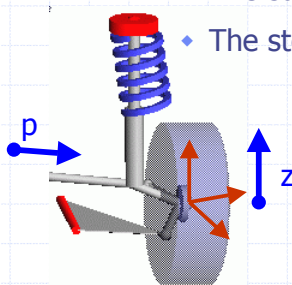
- ⊕ Equations of mechanical systems in symbolic form:
 - Developed in ODECOMS, based on previous software
- ⊕ Library of C++ classes for MBS symbolic modelling:
 - Physical components (bodies, joints, springs...)
 - mathematics (derivative, rotational matrix, constraint eq.)
 - Flexibility:
 - ◆ Formalism independent (Lagrange, Penalty formulation)
 - ◆ Parametric modelling, set of coordinates
 - ◆ Custom modelling approach (i.e. simplifications)
- ⊕ Automatic generation of symbolic code
 - SIMULINK S-fun (C code), VHDL-AMS, C code



Local Simplifications: Level 1

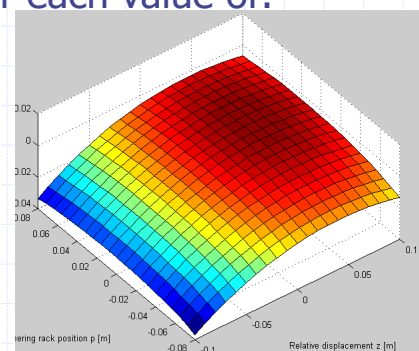
- ⊕ Substitute the real suspension by a **complex joint** connecting the chassis and the wheel
- ⊕ Define the joint behaviour by piecewise polynomial expressions (cubic splines) giving the position and orientation of the wheel spindle, for each value of:

- ◆ The suspension vertical deflection (z)
- ◆ The steering rack position (p)



$$x(z, p), y(z, p)$$

$$\alpha(z, p), \beta(z, p), \gamma(z, p)$$



Local Simplifications: Level 1 (cont.)

- ⊕ Not a linearisation:
 - It retains non-linear behaviour of the suspensions
 - It retains non-linear behaviour of the chassis
- ⊕ Small set of equations
 - Mass matrix is 10x10
- ⊕ It can be enhanced to include elastic deformations due to rubber bushings
 - Calculate the variations in the spindle orientation due to the elastic deformation: elasto-kinematic problem
 - Most relevant: increment in yaw angle (toe in-toe out)



Co-simulation

- ⊕ Solver embedding. Each dynamic block is able to integrate itself one time step.
 - Integrators RK4 and Euler
- ⊕ Vehicle model is broken down in several blocks
- ⊕ In-house co-simulation software
 - Stand-alone
 - No GUI
 - VERY simple co-simulation loop
 - ◆ NO step size control
 - ◆ SAME step size for communication and integration tasks
 - Accepts SIMULINK S-functions (C code)
 - Take care of blocks with direct feed-through inputs



Results of ODECOMS (Level 1)

- ⊕ Good results in today standard PCs => REAL TIME
- ⊕ Comparison SIMULINK vs. In-house co-simulation soft.
 - Noticeable overhead (integrating with Euler)
 - Benefits of co-simulation (integrating with RK4)

% of Real-Time	Simplif Level	Euler (h = 0.002)		RK4 (h = 0.002)	
		SIMULINK	In-house	SIMULINK	In-house
Pentium III 600 MHz	L1	30.0	12.6	102.2	35.0
AMD K7 1.2 GHz	L1	9.5	6.8	35.2	21.4

Tabular suspension
SPLINES

Benefit \approx 28 - 58%
Overheads

Benefit \approx 39 - 66%
Co-simulation + Overheads



Beyond ODECOMS

- ⊕ Question 1: Could we produce more efficient models?
 - We want to use poorer performance RT hardware
 - We want to use more complex hydraulic/electronic models
 - Strategy
 - ◆ Level 2 of simplifications
 - ◆ Again symbolically generated by SAMBS
 - ◆ Working again with a co-simulation schema
- ⊕ Question 2: Do we still need simplifications?
 - Strategy:
 - ◆ Model without simplifications
 - ◆ Again symbolically generated by SAMBS
 - ◆ Working again with a co-simulation schema



Q1 - Further simplifications: Level 2

- ⊕ Further simplification of the tabular suspensions
- ⊕ Tabular data and cubic splines (step 1) substituted by:
 - $x(z, p)$: planes
 - Rest: paraboloids
- ⊕ Suspension springs & dampers are function of the vertical component of the displacement/velocity, not the actual displacement/velocity
- ⊕ Eliminates the time required for table look-up and spline evaluation



CPU Times with Level 2

- ⊕ Better results than Level 1

Benefit \cong 15 - 25%
SIMULINK

Benefit \cong 29 - 39%
In-house

% of Real-Time	Simplif Level	Euler (h = 0.001)		Euler (h = 0.002)	
		SIMULINK	In-house	SIMULINK	In-house
Pentium III 600 MHz	L1	60.0	25.2	30.0	12.6
	L2	46.7	15.7	24.6	7.9
AMD K7 1.2 GHz	L1	18.4	13.2	9.5	6.8
	L2	14.7	9.3	8.0	4.5
		RK4 (h = 0.001)		RK4 (h = 0.002)	
Pentium III 600 MHz	L1	211.0	78.2	102.2	39.2
	L2	158.7	48.7	81.3	24.2
AMD K7 1.2 GHz	L1	69.0	42.4	35.0	21.4
	L2	52.5	29.2	26.5	14.5



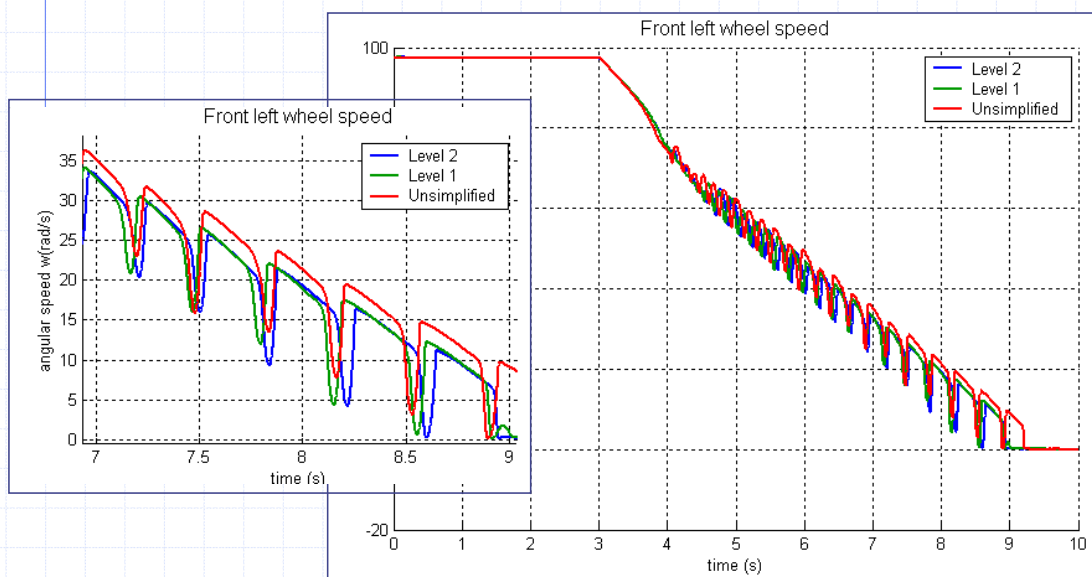
Q2 – Unsimplified Vehicle model

- ⊕ Exact kinematics of front and rear suspensions
 - Closed loops
 - Global formulation, Euler Angles and Penalty Method
 - ◆ Inefficient (mass matrix is 18x18)
 - Model symbolically generated with SAMBS
- ⊕ NO real-time performance

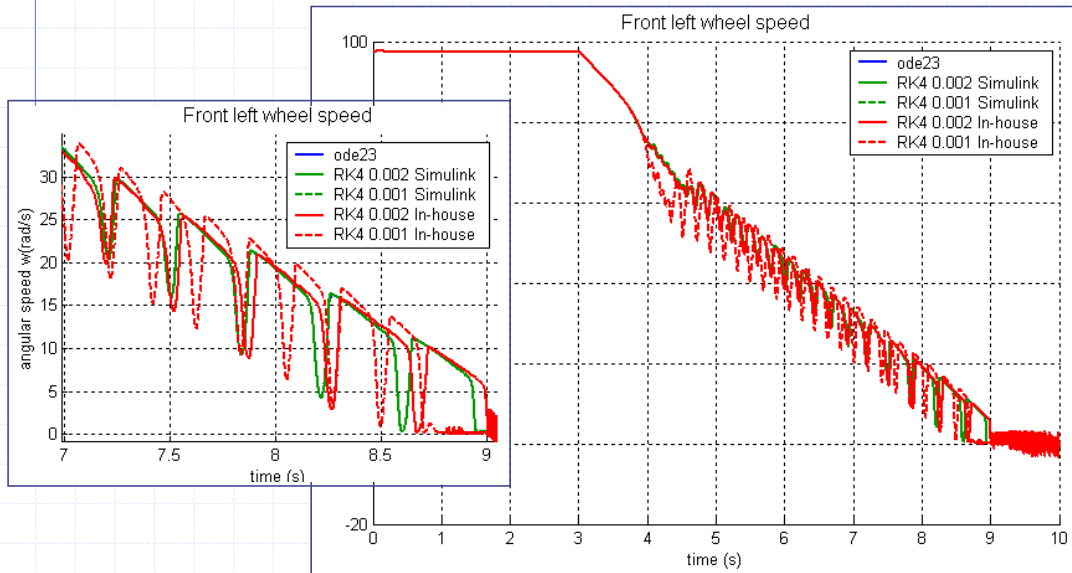
% of Real-Time	Simplif Level	RK4 (h = 0.002)		Euler (h = 0.001)	
		SIMULINK	In-house	SIMULINK	In-house
AMD K7 1.2 GHz	Un-sim	373.2	330.5	190.3	171.0
	L1	35.0	21.4	18.4	13.2
	L2	26.5	14.5	14.7	9.3



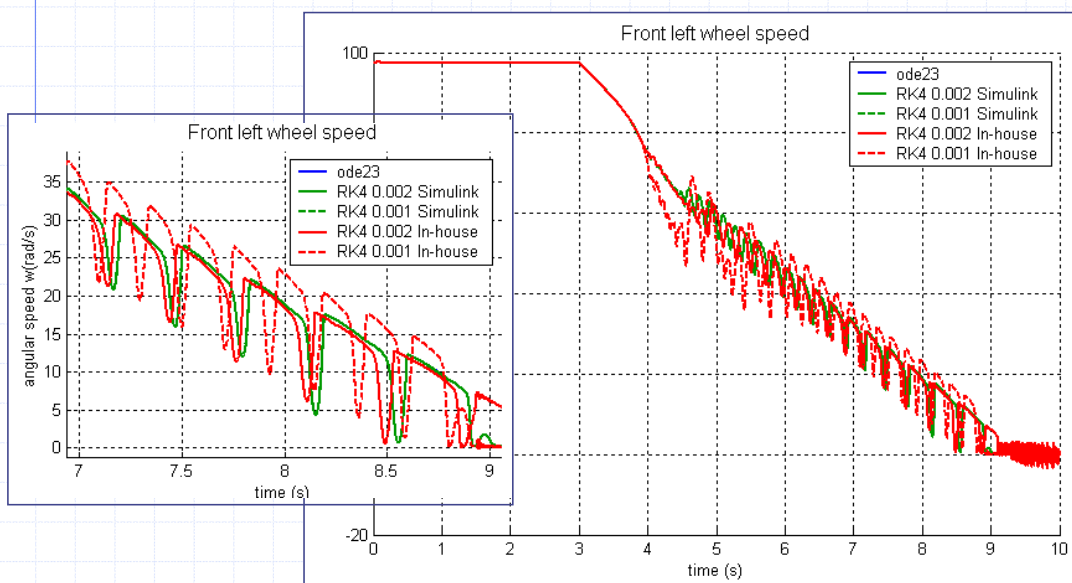
Unsimplified, L1 and L2 with ODE23



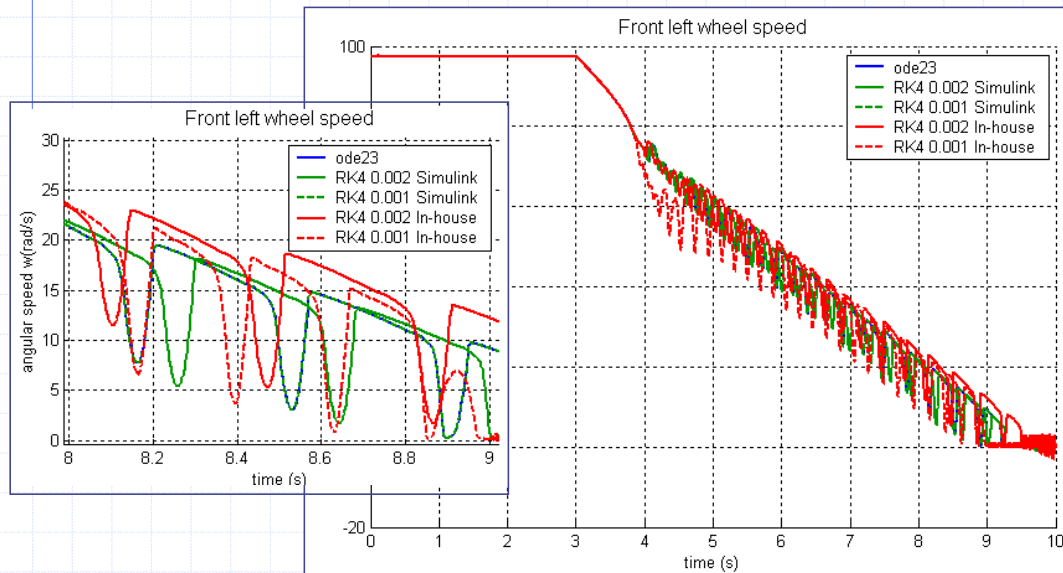
Different integrations of Level 2



Different integrations of Level 1



Different integrations of the unsimplified



Conclusions

- ⊕ Co-simulation contributes to more efficient simulations
 - It produces less accurate results, but still good ones
 - Trial and error to determine the most suitable integrator and time step that guaranty stability, accurate results and RT
- ⊕ Level 1 and Level 2 simplifications produce quite accurate results
- ⊕ Simplifications are still needed for RT purposes. The unsimplified model is not efficient due to:
 - Dynamic formulation chosen
 - Almost double size when solving $Mq'' = Q$ (10x10 vs. 18x18)



Future work

- ⊕ Analytical approach for determining the most suitable integration time-step for a given integrator in co-simulation:
 - Two simple linear blocks (one DOF or two DOF) interconnected
 - Definition of limits of stability
- ⊕ Think a more elaborated co-simulation loop
 - Current is very simple
 - Considering a different integration time-step for each block
 - Distinguish between integration time-step and communication time-step



Modelling and simulation of coupled hydraulic and multibody subsystems

Sven Dronka

dronka@rcs.urz.tu-dresden.de

**Dresden University of Technology
Faculty of Transportation Sciences
D – 01062 Dresden, Germany**

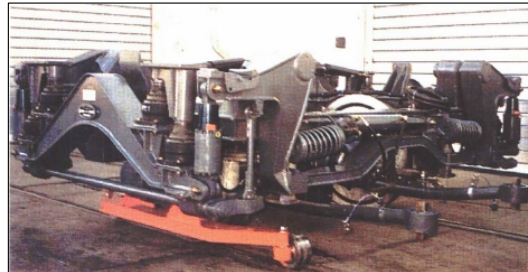
For the simulation of a railway vehicle with hydraulically driven active tilting system it is necessary to model a hydraulic subsystem representing the actuators and furthermore a multibody subsystem representing the mechanical structure of the railway vehicle. To simulate the whole system, it is necessary to couple the subsystems.

In this presentation, a method for the modelling and simulation of such railway vehicle is described, which can be applied up to the level of real-time simulations. For modelling, two commercial simulation tools were applied: SIMPACK for the modelling of the multibody subsystem and DSHplus for the modelling of the hydraulic subsystem. Both tools offer the possibility of model export. The exported models can be imported into Simulink (by using its s-function interface) for the (non real-time) simulation of the coupled subsystems.

By the use of the real-time workshop, the simulink model (with the coupled subsystems) can be transferred to a real-time hardware to simulate the model under real-time conditions. The real-time simulation is then used in a hardware-in-the-loop test rig, where hydraulic actuators can be tested.



Modelling and Simulation of coupled hydraulic and multibody subsystems (with extension to realtime application)



Prof. S. Liebig, S. Dronka
Technische Universität Dresden
Institut für Theoretische Grundlagen der
Fahrzeugtechnik

Prof. S. Helduser, M. Stüwing
Technische Universität Dresden
Institut für Fluidtechnik

The presented results are generated by a project with the title „Development tools for railway carriages with hydraulic components“. The project was funded by the German Federation of Industrial Cooperative Research Associations „Otto von Guericke“ (AiF-Nr. 12074 B/1), on behalf of The Federal Ministry of Economics and Technology (BMW), in the Section Fluid Power of the German Engineering Federation (VDMA).

11 October 2001

Co-Simulation-Workshop in Stuttgart



Motivation

**Adding active elements
into structures
(mechatronic systems)**

+

**Application of
simulation tools in
development process**

=

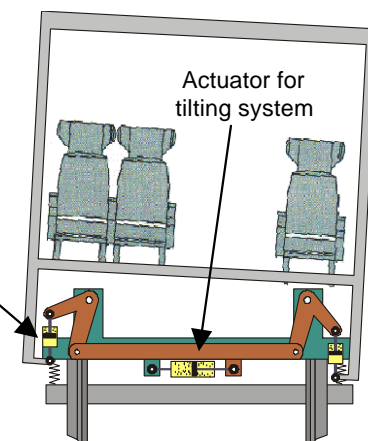
**Need for
multi-domain
simulation tools**

Example: Tilting train

Hydraulic components for

- Active tilting system
- Active secondary suspension
- Active axle steering system

Hydro-
pneumatic
suspension

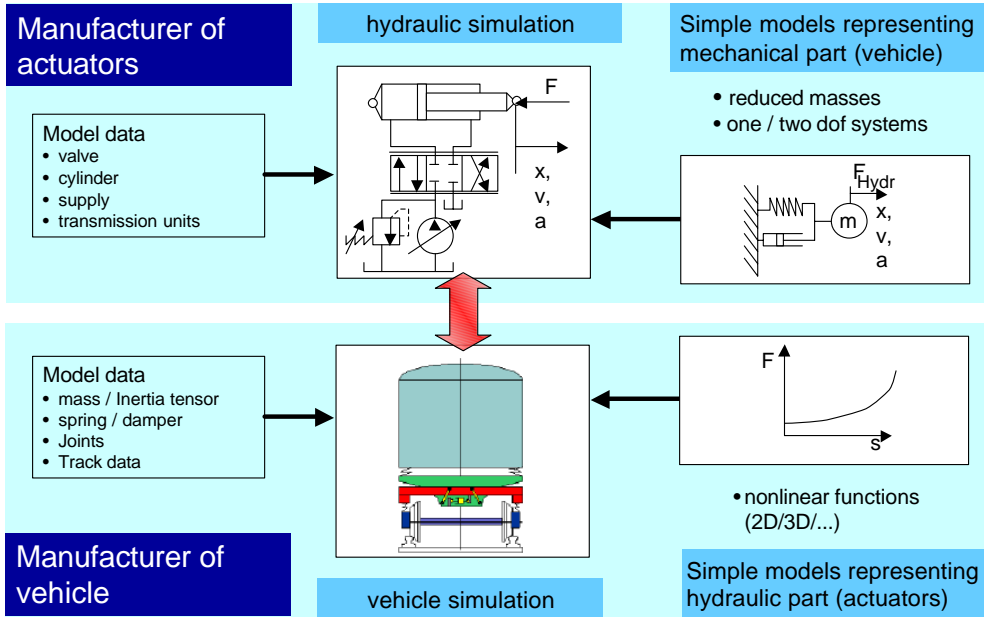


11. October 2001

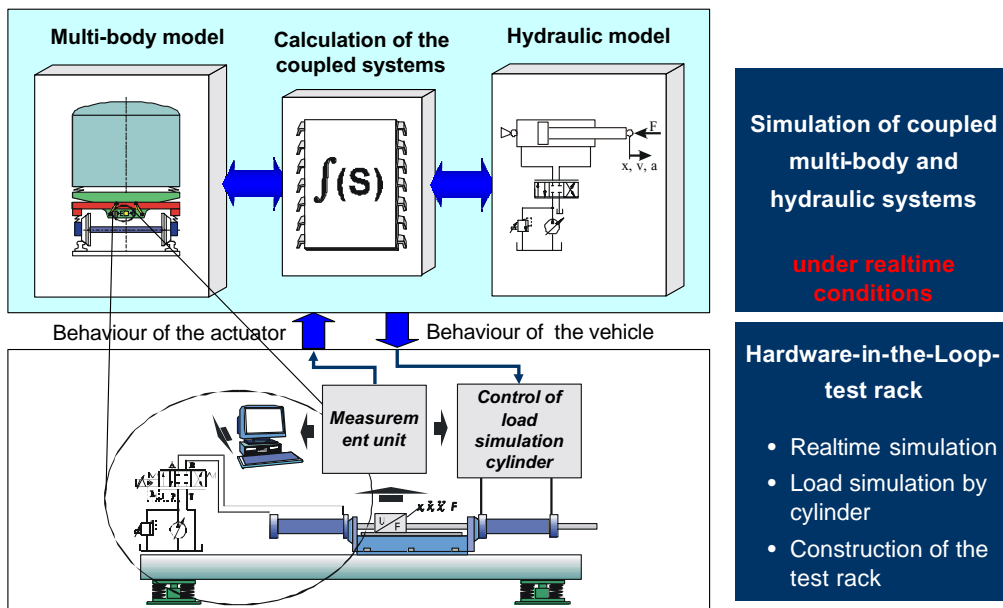
Co-Simulation-Workshop in Stuttgart



Situation with the manufacturers



Main topics of the project





Formulation of demands for the solution

Modelling of the subsystems

- Application of specialized tools
- Application of commercially available standard tools
- Tools must offer the possibility of model export

Coupling of the subsystems

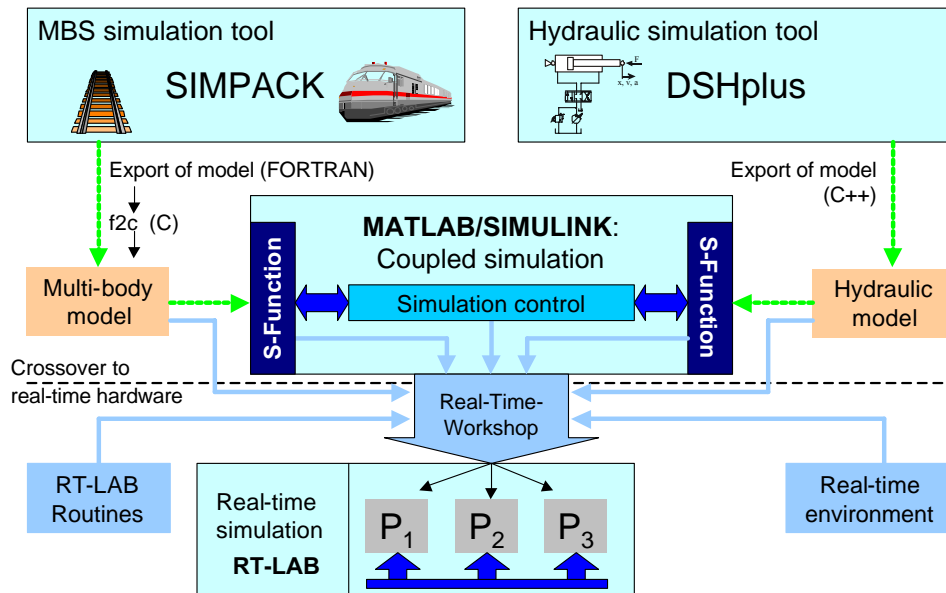
- Use of standard interfaces

Crossover to the realtime hardware

- Software support for this step with a minimum programming effort



Software concept





Potential of the solution I



Application of general software tools for modelling

SIMPACK:
- any* mechanical 3D-
structure can be modelled

DSHplus:
- any* hydraulic or pneumatic
system can be modelled

*) corresponding to the possibilities of the modelling tool



Application of RTW for the crossover to real-time hardware

Integrated tool chain from
(offline-)simulation up to the level
of real-time simulation

There are several real-time
systems available for using with
RTW



Potential of the solution II



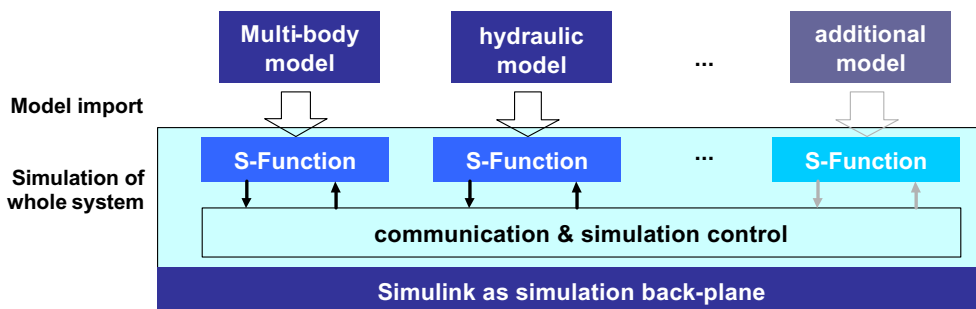
Use of standard interface (s-function) for integration
of models into simulink

Substitution of modelling
tools¹⁾

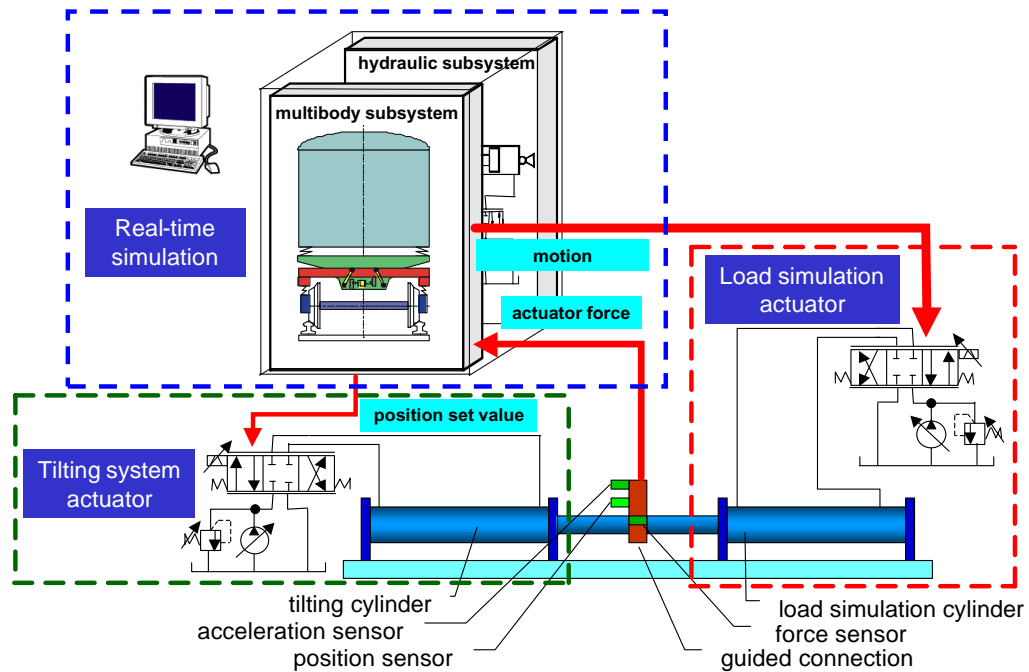
Addition of subsystems of
other engineering disciplines²⁾

1) tool with the possibility of model export (MKS: NEWEUL instead of SIMPACK, Hydraulic: AMESim instead of DSHplus)

2) models with continuous states, which can be described by ODE or DAE



Survey of the HIL test bed

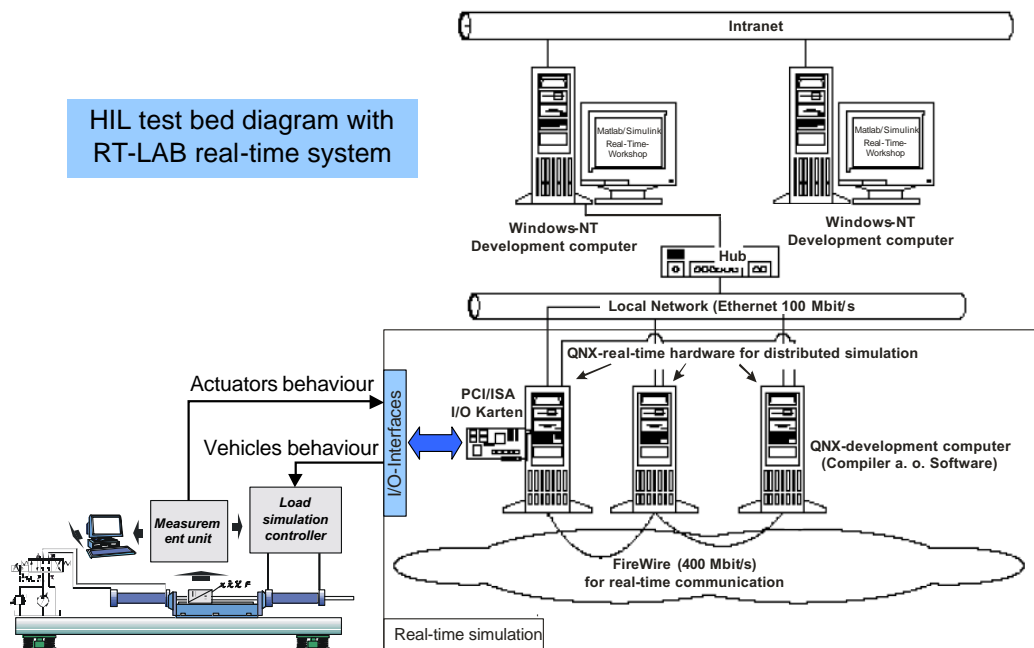


11. October 2001

Co-Simulation-Workshop in Stuttgart

Survey of the real-time system RT-LAB

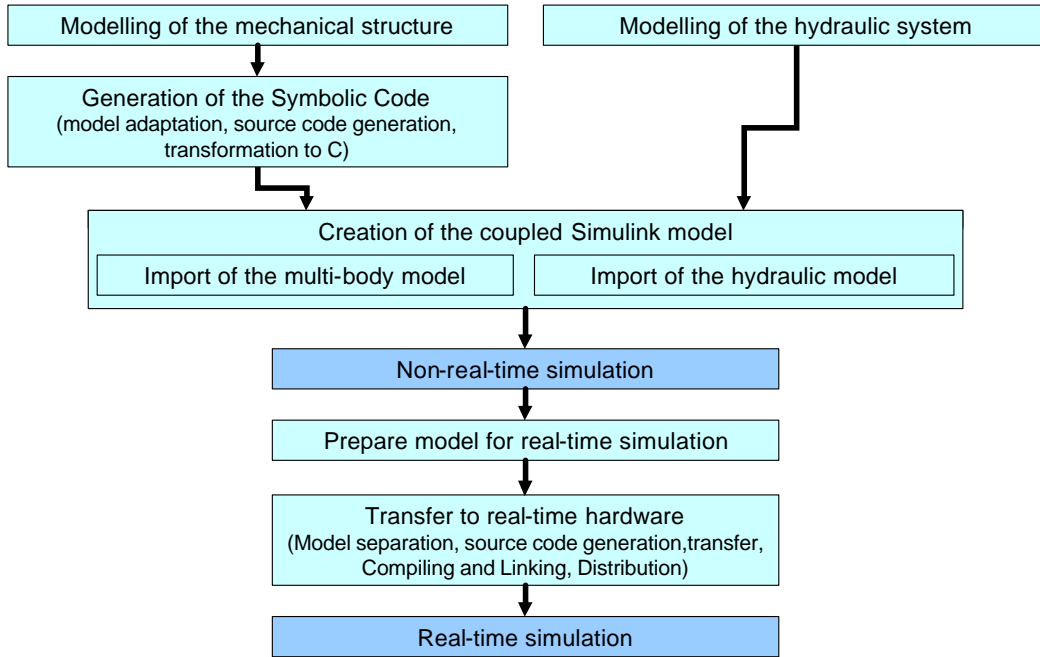
HIL test bed diagram with RT-LAB real-time system



11. October 2001

Co-Simulation-Workshop in Stuttgart

Application of the tools

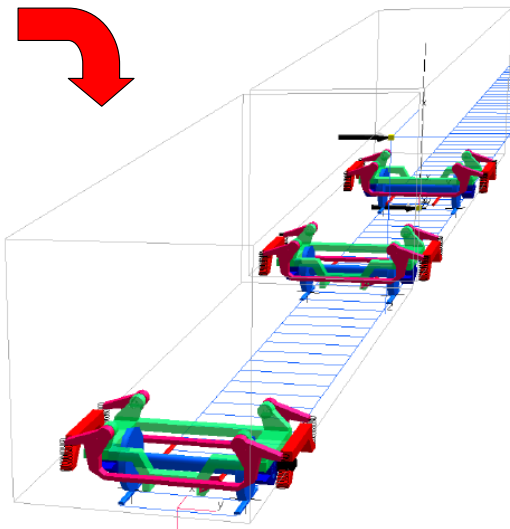
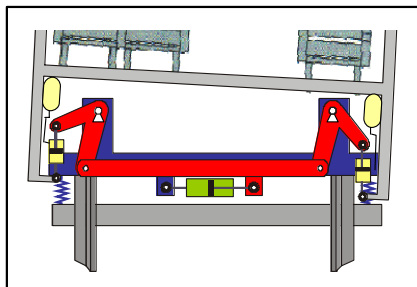


11. October 2001

Co-Simulation-Workshop in Stuttgart



Modelling of the mechanical structure (SIMPACK)

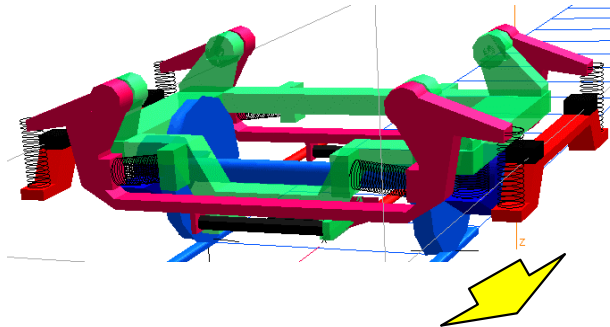


11. October 2001

Co-Simulation-Workshop in Stuttgart



Reduction of the mbs model

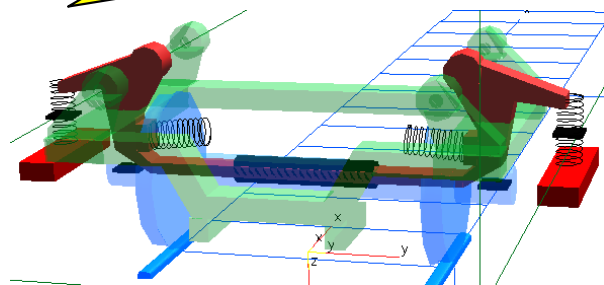


Original model

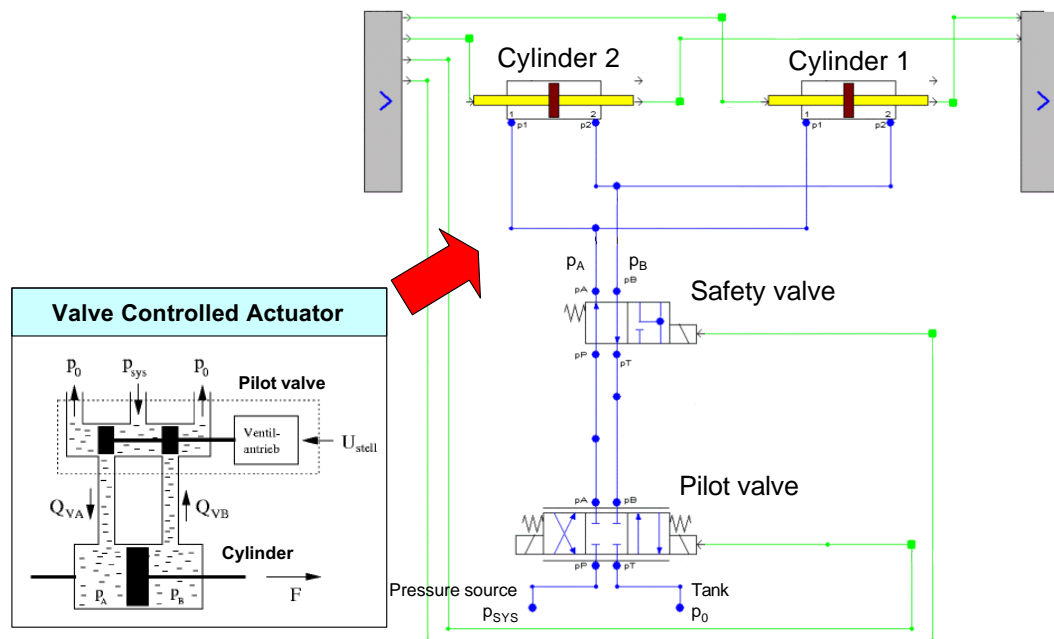
- 74 states
- 5 constraints
- 42 force elements

Reduced model

- 30 states
- no constraints
- 15 force elements
- Closing of the kinematic loop with spring
- No wheel-rail elements

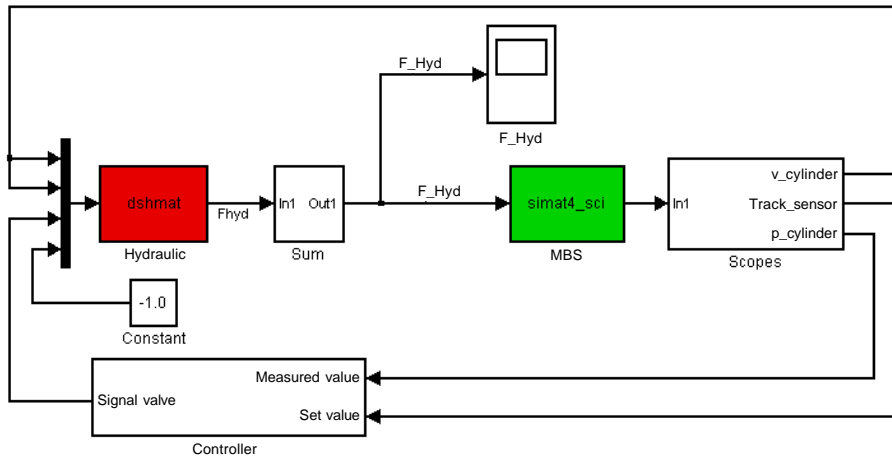


Modelling of the hydraulic system (DSHplus)





Design of the coupled model (Simulink)



Design of the coupled model (Simulink)

DSHplus Modellimport

DSHplus model data

Model file: RE2000M6b.cpp

Parameter file: RE2000P6b.txt

Input Function file: RE2000P6b.txt

Path for target files: s:\gekosim\models\simulink\gekosim\

Configuration of DSHplus model import actions

Step 1: Copy all source files to target directory

Setup the Search Path for all necessary files:

Step 2: Make SFunction in target directory

Compile options: Compile with verbose output, Compile for debugging

Simulation options: Scale states during simulation

Message area

Step 1:

Step 2:

SIMPACT Modellimport

SIMPACT model import

MATLAB mex compiler settings

Source path for model files: s:\gekosim\models\SIMPACT-SymCode\RE2000M6b\

Source path for simat files: s:\gekosim\src\simat

Target path for simat files: s:\gekosim\models\simulink\gekosim\

Model import options: C-code model (f2c), FORTRAN-code model

SIMPACT (exported) model settings

Name of (exported) model file: RES_mit_NT_Rueckstellf

Additional settings

F2C settings:

SIMPACT settings:

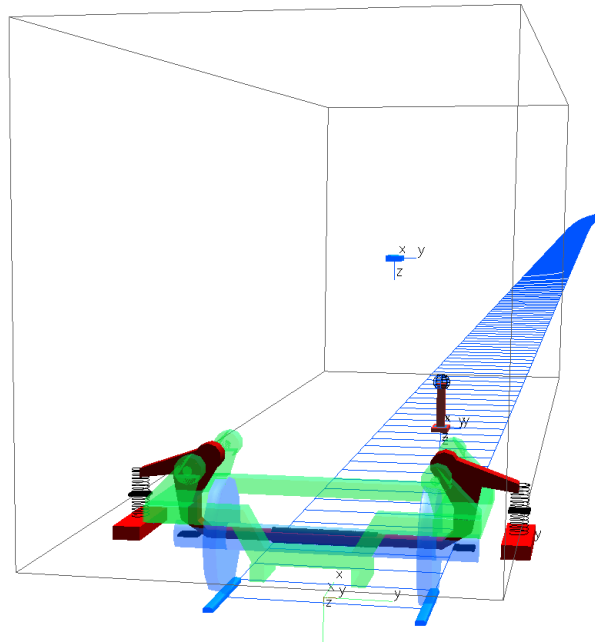
Investigation with tilting train at curve ride

• Passing an S-curve

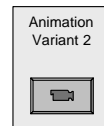
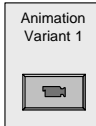
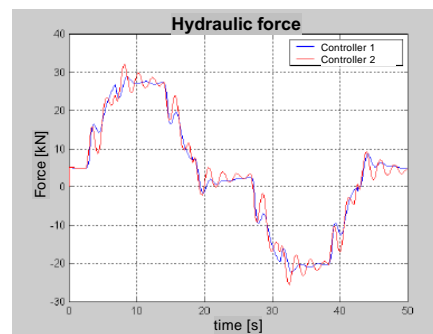
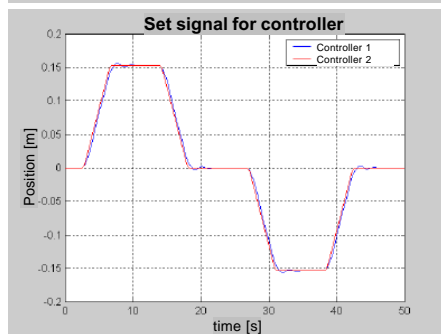
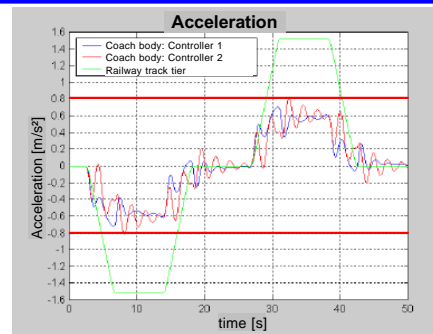
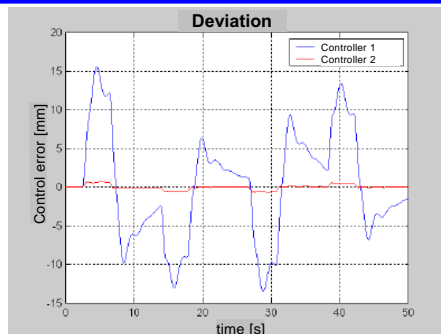
curve radius: 450 m
 Design speed: 90 km/h
 Driving speed: 120 km/h
 Transverse acceleration
 on railway track tier:
 at 90 km/h: 0,44 m/s²
 at 120 km/h: 1,52 m/s²

• 2 controller variants

- Control of the actuators position
- V1: PI-Controller + feed forward
- V2: PI-Controller



Selected results





Summary and outlook

Realization of a tool framework for the coupled simulation of multi-body and hydraulic subsystems

- Features of the SIMPACK s-function
 - Track functionality (Repetition)
 - Write results of simulation in SIMPACK and MATLAB format
 - Read and write states from/to file
 - Features of the DSHplus s-function
 - Read and write states from/to file
 - Application for real-time simulation
- Off-line-simulation of the coupled models
 - Real-time simulation of multi-body model
 - Real-time simulation of hydraulic model
 - **Real-time simulation of coupled models**

Further job / extension possibility

- Realization of a real co-simulation (solution of multi-body models with constraints)
- Enlargement of the functionality (f.e. Read parameters of the multi-body model from file)
- Tuning of multi-body and hydraulic models



References

Links concerning the used hard- and software

- Homepage of SIMPACK: <http://www.simpack.de/>
- Homepage of DSHplus: <http://www.fluidon.com/>
- Homepage of MATLAB / Simulink: <http://www.mathworks.de/>
- Homepage of RT-LAB: <http://www.opal-rt.ca/>

References

- Liebig, S., Quarz, V., Dronka, S.: Schienenfahrzeuge mit aktiven Elementen, Proceedings of the conference 17. *Verkehrswissenschaftliche Tage, Tagungssektion 3: Entwicklungstendenzen in der Fahrzeugtechnik*, 04./05. June 1998, Dresden, Germany
- Liebig, S., Quarz, V., Dronka, S.: Simulation von Schienenfahrzeugen mit MKS-Software, Proceedings of the conference *SIM 2000: Simulation im Maschinenbau*, P. 713-736, February 24/25, 2000, Dresden, Germany
- Liebig, S., Helduser, S., Stüwing, M., Dronka, S.: Die Modellierung und Simulation gekoppelter mechanischer und hydraulischer Systeme, Proceedings of the conference *18th Dresden Conference on Traffic and Transportation Sciences, Session 2: Vehicle Mechatronics*, September 17/18, 2001, Dresden, Germany

‘MATLAB™ - 20-sim interaction’

Short description of the 20-sim demonstration given during the **co-simulation** meeting at the University of Stuttgart, Stuttgart-Vaihingen, Oct. 11, 2001.

Peter Breedveld

Cornelis J. Drebber Institute for Mechatronics & Control Laboratory
Electrical Engineering Department, University of Twente, P.O. Box 217, 7500 AE Enschede,
Netherlands, ph.: +31 53 489 2792, fax: +31 53 489 2223, e-mail: p.c.breedveld@el.utwente.nl

Frank Groen

Control Lab Products B.V., Drienerlolaan 5, EL-RT, 7522 NB Enschede, Netherlands, ph.: +31 53 489 3096, fax: +31 53 489 2223, e-mail: info@20sim.com, <http://www.20sim.com>

Information and a demo version of the 20-sim software can be found at www.20sim.com, MATLAB™ is a trademark of the The MathWorks Inc.

Interaction between MATLAB™ and 20-sim can take place in different ways (not all of them were actually demonstrated):

1) ‘On-line’ interaction:

20-sim has the instructions (‘SIDOPS functions’) **toMatlab**(‘*MATLAB command line*’), **doMatlab**(‘*MATLAB command line*’), **fromMatlab**(‘*MATLAB command line*’) in order to communicate with and control MATLAB™ before, during and after simulation, as illustrated by the following simple example:

Example

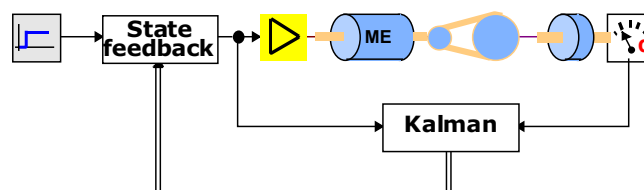
The following lines are input code for the 20sim equation editor (‘!’ is the 20-sim comment symbol):
//this example shows how variables can be transferred to and from MATLAB™ and how 20-sim can
//instruct MATLAB™ to start and execute instructions before, during and after simulation

```

variables                                //Declaration of variables in the 20-sim part of the model.
    real x,y;
initialequations                         //At the start of the simulation:
    doMatlab ('a = [];');                   //create an empty array a,
    doMatlab ('b=0;');                     //create a variable b.
equations                                 //During simulation:
    x = sin (time);                         //calculate x,
    toMatlab (x);                           //send it to MATLAB™
    doMatlab ('a = [ a x ];');              //and add it to array a.
    doMatlab ('b = x + 2;');                //In MATLAB™ add 2 to x,
    fromMatlab (y,'b');                     //read b.
finalequations                            //At the end of the simulation:
    doMatlab (' plot (a);');                //plot the resulting array in MATLAB™.
  
```

This on-line interaction was demonstrated by the 20-sim model of a servo system shown below by:

- calculation of the optimal gain matrix by the MATLAB™ LQR command in the ‘initial equation’ section of the **state feedback** submodel
- calculation of the optimal estimates of the states of a process by the MATLAB™ LQE command in the ‘initial equation’ section of the **Kalman** filter submodel.



2) 'Off-line' interactions:

LINEARIZATION

20-sim has a 'Linearize model command' that allows you to create a linear model in a specified operating point. The results may be edited and looked at in different forms (state-space, zero pole gain, transfer function, bode plots, step response, Nyquist diagram, Nichols chart, pole-zero plot, but it may also be exported to a separate 20-sim or MATLAB™ model. Furthermore, a linear (ABCD) MATLAB™ model by be the input of the linear system editor. Apart from exporting the result to MATLAB™ directly, 20-sim can generate MATLAB™ command lines that can be copied into the MATLAB™ command window for immediate execution, as shown in the example below:

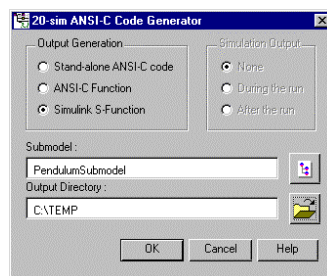
Example

The following lines are input code for the MATLAB™ Command Window ('%' is the MATLAB™ comment symbol):

```
% 20-sim Linear System Editor.
%
% This linear model is in MATLAB™ format
% and can be copied and pasted directly.
% linear system matrices
A = [0.0 1.0; -9.8066499999984, 0.0];
B = [0.0; 1.0];
C = [1.0 0.0];
D = [0.0];
% preformatted commands for MATLAB™
% generate state-space description of linear model
sys = ss (A, B, C, D);
% show bode plot of state-space system
bode (sys);
% show transfer function of state-space system
reduced = minreal (sys);
tf (reduced)
```

S-function generation

In the professional version of 20-sim, ANSI-C code may be generated by automatic conversion of a 20-sim main or model into ANSI C-code, as long as the model does not contain specific functions like discrete or event functions, time delays, etc. The application can be used to generate MATLAB™/SIMULINK™ **S-functions**, to generate standalone executables or to generate input/output functions for use in other C and C++ programs. The ANSI-C code Generator can be opened from the Simulator (Tools menu, Generate C-Code command).



DATA EXPORT

20-sim provides several ways to export 20-sim simulation data to MATLAB™ for further analysis.

Suppose you have saved the simulation setup in the experiment file 'MotorExperiment.exp'. In the same directory where this experiment file is stored, a subdirectory matlab is created in which you may find the following files:

mr.dat	output data of a multiple run or parameter sweep (ASCII format).
mulrun.m	MATLAB™ m-file to generate a plot out of mr.dat.
opt.dat	output data of an optimization run (ASCII format).
optim.m	MATLAB™ m-file to generate a plot out of opt.dat.
mc.dat	output data of a Monte Carlo run (ASCII format).
monte.m	MATLAB™ m-file to generate a plot out of mc.dat.

Multiple Run / Parameter Sweeps

After performing multiple runs and parameter sweeps the results may be inspected in the ‘multiple run results window’. If selected, the final value of the result variable is shown in this window. Final value means: the value of that variable, at the end of each run. If the MATLAB™ button of the ‘multiple run results window’ is activated, the resulting end values will be stored in the data file `mr.dat` (ASCII-format). A script file `mulrun.m` is also created to automatically enter this data file into MATLAB™ and to automatically perform the required commands in the MATLAB™ command window (the user does not have to enter anything himself in MATLAB™). The order in which the parameters are stored in the data file can best be described by pseudo code. Assuming n parameters (p_1 to p_n) with a minimum (min) and maximum (max) value and s steps, data will be stored according to:

```

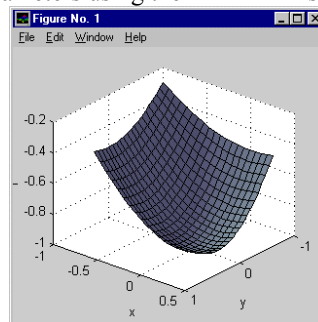
for (pn = min to pn = max ; step = (max-min+1)/s)
  for (p(n-1) = min to p(n-1) = max ; step = (max-min+1)/s)
    ...
    for (p2 = min to p2 = max ; step = (max-min+1)/s)
      {
        for (p1 = min to p1 = max ; step = (max-min+1)/s)
          {
            print final value;
            print tab
          }
        print newline
      }
    }
  }
}

```

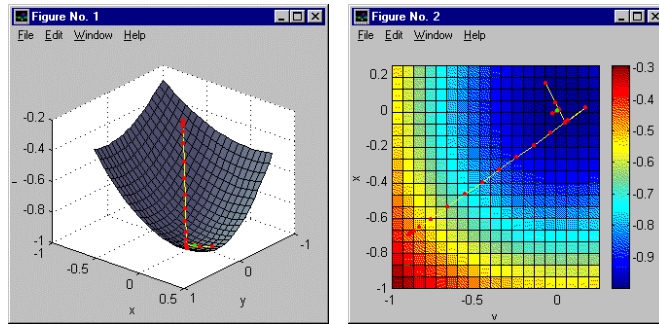
For three parameters ($1 \leq p_1 \leq 3$, $1 \leq p_2 \leq 3$, $1 \leq p_3 \leq 3$) and 2 steps this means that the final values will be stored according to the following layout:

(p1=1, p2=1, p3=1)	(p1=2, p2=1, p3=1)	(p1=3, p2=1, p3=1)
(p1=1, p2=2, p3=1)	(p1=2, p2=2, p3=1)	(p1=3, p2=2, p3=1)
(p1=1, p2=3, p3=1)	(p1=2, p2=3, p3=1)	(p1=3, p2=3, p3=1)
(p1=1, p2=1, p3=2)	(p1=2, p2=1, p3=2)	(p1=3, p2=1, p3=2)
(p1=1, p2=2, p3=2)	(p1=2, p2=2, p3=2)	(p1=3, p2=2, p3=2)
(p1=1, p2=3, p3=2)	(p1=2, p2=3, p3=2)	(p1=3, p2=3, p3=2)
(p1=1, p2=1, p3=3)	(p1=2, p2=1, p3=3)	(p1=3, p2=1, p3=3)
(p1=1, p2=2, p3=3)	(p1=2, p2=2, p3=3)	(p1=3, p2=2, p3=3)
(p1=1, p2=3, p3=3)	(p1=2, p2=3, p3=3)	(p1=3, p2=3, p3=3)

If only one or two parameters / initial conditions are defined to be open for variation, the final values may be plot as function of these parameters using the MATLAB™ script `mulrun.m`. (cf. the figure below).

**Optimization**

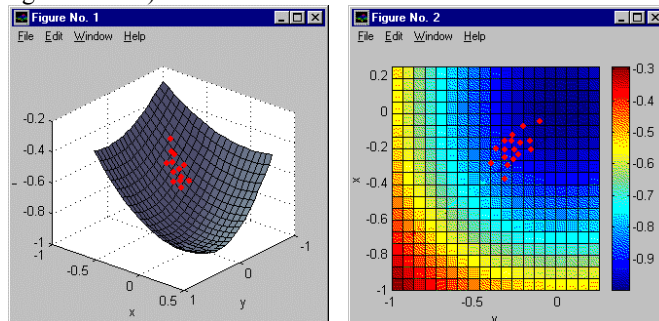
After performing a 20-sim optimization the results may be inspected in the multiple-run-results window. The final value of the result variable is shown in this window. Final value means: the value of that variable, at the end of each run. With the ‘create trajectory button’ in the multiple-run-results window the resulting end values can be stored in the data file `opt.dat` (ASCII-format). A script file `optim.m` is also created that may be used to enter this data file automatically into MATLAB™ and to automatically perform the required commands in the MATLAB™ command window. The data file contains a number of lines. Each line contains the data of one run: the parameters values and the final value. If only one or two parameters are defined to be open for variation, the final values may be plot as function of these parameters. Interesting plots may be obtained, when a parameter sweep with the same parameters is performed first. Then the optimization trajectory is shown on top of the ‘parameter sweep’ plot (cf. the figures below).



Monte Carlo Analysis

The results of a Monte Carlo analysis in 20-sim may be inspected in the ‘multiple-run-results window’. The final value of the result variable is shown in this window. Final value means: the value of that variable, at the end of each run. Activation of the MATLAB™ button of the ‘multiple-run-results window’ stores the resulting end values in a data file mc.dat (ASCII-format). A script file monte.m is also created that may be used to enter this data file automatically into MATLAB™ and to automatically perform the required commands in the MATLAB™ command window. The data file contains a number of lines. Each line contains the data of one run: the parameters values and the final value.


If only one or two parameters are defined to be open for variation, the final values may be plot as function of these parameters. Interesting plots may be obtained, when a parameter sweep with the same parameters is performed first. In that case the Monte Carlo results are shown on top of the ‘parameter sweep’ plot (cf. the figures below).




Simulation run

During a regular simulation run simulation data can be stored on file using the 20-sim ‘data file’ command. This command opens an editor in which the data to be stored on file can be specified. Data stored in text format can easily be imported into MATLAB™.

Data export to the MATLAB™ data space

A direct link to the MATLAB™ data space is possible in the 20-sim parameters editor, the 20-sim variables chooser and the 20-sim linear system editor (cf. the linearization section). Using the  button, data can be sent to MATLAB™.

DATA IMPORT FROM MATLAB™ DATA SPACE

A direct link from the MATLAB™ data space is also possible in 20-sim parameters editor, the 20-sim variables chooser and the 20-sim linear system editor (cf. the linearization section). Using the  button, data can be imported from MATLAB™. As mentioned when linearization was discussed, also a linear ABCD model may be imported into the linear system editor either to prevent time consuming interaction during simulation and/or to let models created in MATLAB™ benefit from the superior speed of the 20-sim simulator.

ACKNOWLEDGEMENTS

Job van Amerongen created the model of the servo system and with Kalman filter and state feedback that uses on-line MATLAB™ interaction.

Simulation of an anti-skid system using several modelling and simulation tools

Frank Kohlschmied

frank.kohlschmied@simpack.de

INTEC GmbH

Argelsrieder Feld 13, D – 82234 Wessling, Germany

In the framework of the EUMECH Project, where various possibilities of interdisciplinary modelling and simulation were discussed, a vehicle model was generated using mechanical, hydraulic and control elements. This was carried out in co-operation between MLaP, Paderborn, and INTEC, Wessling.

The parts of the various disciplines were modelled independently from each other in different modelling tools and put together for a simulation of the complete system in one simulation tool. The focus of this project was to describe the possibilities of data exchange between several domains and disciplines by way of example, and to document problems that were encountered. The simulation performed concerned the braking of a car using an anti-skid system. The car model, including detailed description of the wheel bearings and the behaviour of the tyre, was created at INTEC using the multi-body simulation program SIMPACK.

The braking hydraulics and the logics of the anti-skid system were modelled by MLaP, using the simulation package CAMEL. The complete scenario was simulated in MATLAB/SIMULINK using the interfaces of CAMEL and SIMPACK to MATLAB/SIMULINK.

International Workshop
"Co-Simulation for Mechatronic Systems"
Stuttgart (Germany), October 11, 2001

Simulation of an anti-skid system using several modelling and simulation tools

Frank Kohlschmied
Ingenieurgesellschaft für neue Technologien (Intec) GmbH, Weßling
www.simpack.de



International Workshop
"Co-Simulation for Mechatronic Systems"
Stuttgart (Germany), October 11, 2001



Contents

- 1. Intec and SIMPACK**
- 2. Overview on possible and realized Co-Simulation Interfaces with SIMPACK**
- 3. Multi-disciplinary simulation of an anti-skid system**
(as sub-project of the Eumech project)

Focus on :

- strategies for the coupling of systems originating from different sources
- application to mechatronic system design and vehicle simulation
- industrial experience and demands

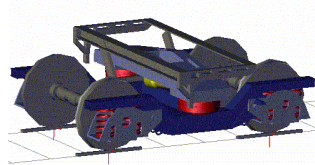
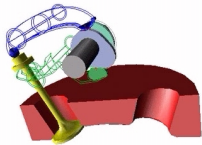
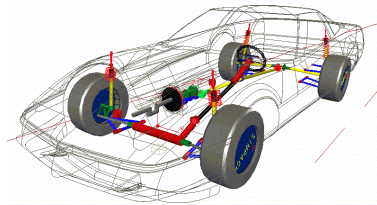
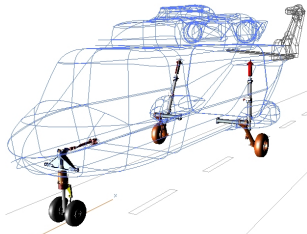


International Workshop
 „Co-Simulation for Mechatronic Systems“
 Stuttgart (Germany), October 11, 2001



SIMPACK

Analysis and Design of General Mechanical Systems



What Are You Intending
 What Are You Intending To Do ?

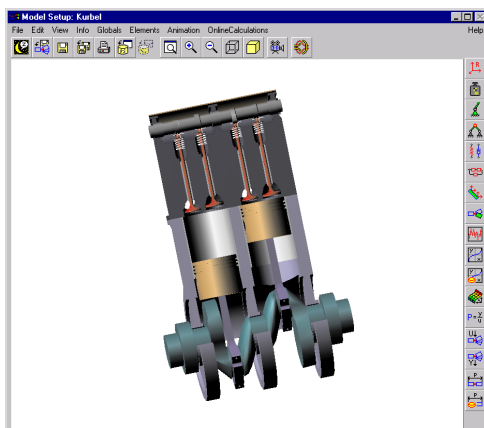


International Workshop
 „Co-Simulation for Mechatronic Systems“
 Stuttgart (Germany), October 11, 2001



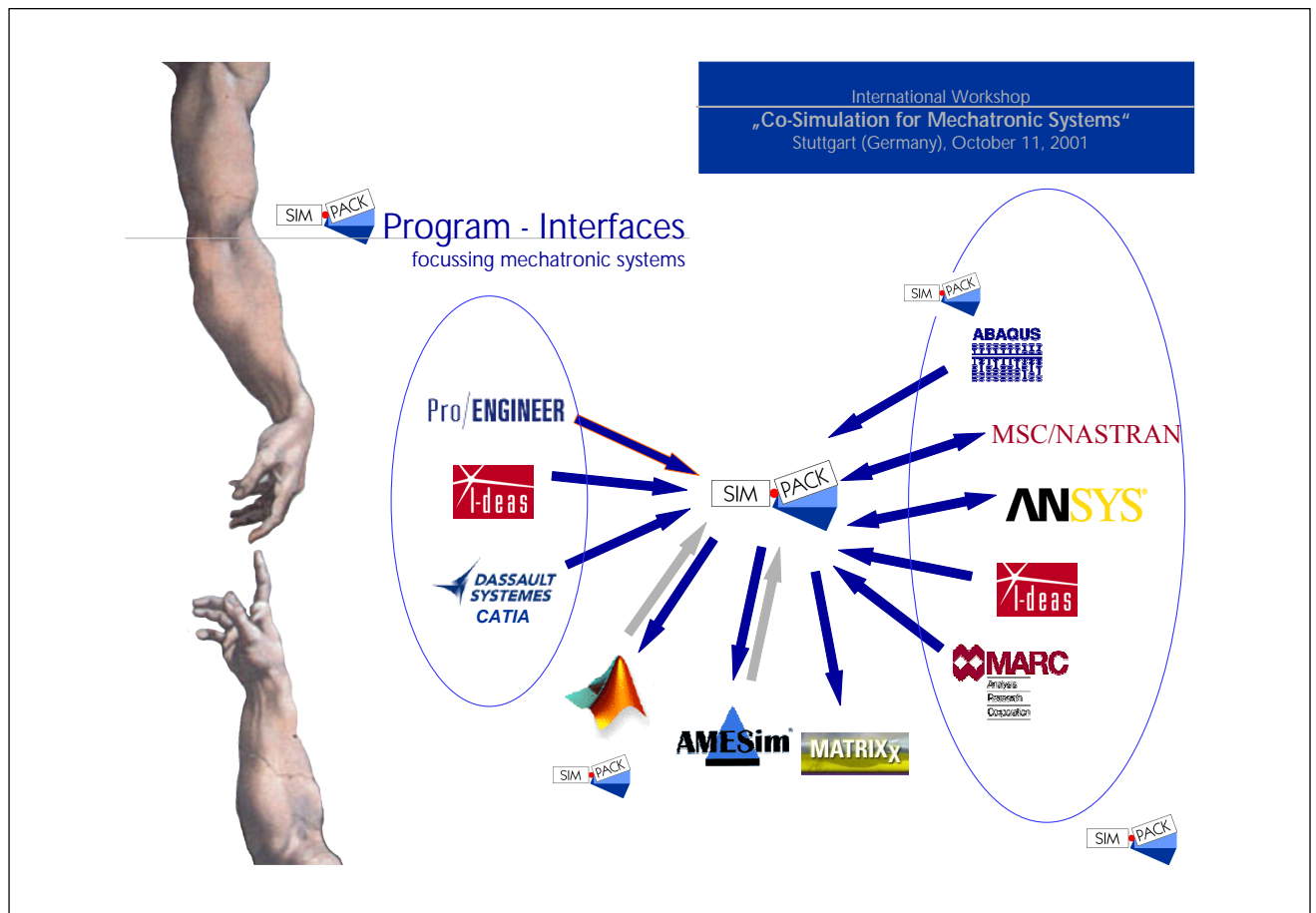
SIMPACK

Multi-Body Simulation Concepts



- ▶ *Pre Processor* „close to physical reality“
- ▶ Large Library with *High Level Modelling Elements*
- ▶ Big Variety of *Solver Options*: linear, non-linear, combined, symbolic, etc.
- ▶ *Powerful Solver*: fast, stable and reliable
- ▶ Advanced Concept for Vehicle Models in SIMPACK Wheel/Rail





International Workshop
 "Co-Simulation for Mechatronic Systems"
 Stuttgart (Germany), October 11, 2001

intec

The SIMPACK Company

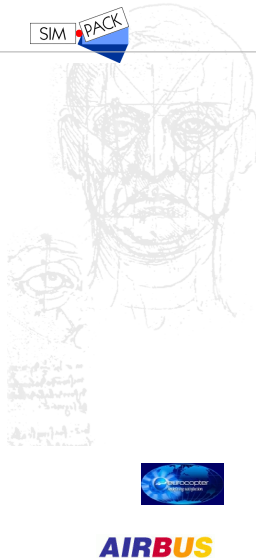
- ▶ SIMPACK Software Development (Libraries, GUI, Graphics, etc.)
- ▶ Software Sales
- ▶ Software Training
- ▶ SIMPACK Accademy
- ▶ Hotline, User Meetings, SIMPACK-News
- ▶ Setting up Models, SIMPACK User Routines, Concept Computations, Real Time Models, Complete Projects

International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

SIMPACK

The Main Customers



ADtranz

ALSTOM

ANSALDOBREDA

Nederlandse Spoorwegen

JR

BOMBARDIER
TRANSPORTATION

Deutsche Bahn DB

SUMITOMO
METALS

SGP

SIEMENS



LAND-ROVER

BOSCH



WABCO



reynard

MAN

DAF

mazda



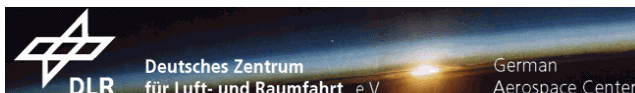
International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

SIMPACK

Partner in Development: DLR

- ▶ Development of SIMPACK Basics: Simulation Engine, Solver, Mathematical Methods
- ▶ General Research oriented SIMPACK Development
- ▶ Research Projects with SIMPACK in the Fields: Aircraft Landing Gears, Railway Dynamics, Road Vehicle Handling and Comfort
- ▶ SIMPACK Test platform: More than 40 Installations at DLR





Co-Simulation with SIMPACK

general programming interfaces

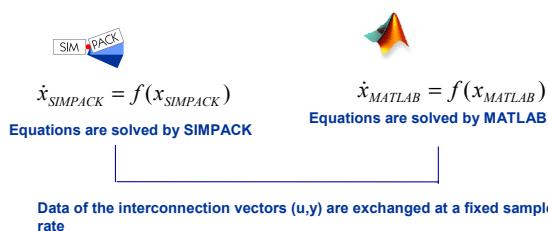
- ▶ **SIMPACK USER Routines**
 allows to link arbitrary external functionality to the SIMPACK algorithm based on FORTRAN or C-Code, e.g. force laws, controllers etc
- ▶ **Symbolic Code Interface**
 export of FORTRAN Code containing the differential equations of any SIMPACK model. This code can be linked to an external DAE or ODE integrator
- ▶ **IPC Co-Simulation Interface**
 pre-defined Cosimulation interface which allows a Co-Simulation of a SIMPACK model with an external system via socket communication



Co-Simulation with SIMPACK

realized Co-Simulation Interfaces with SIMPACK

- ▶ **One process Coupling**
 master and slave process are linked together in one executable. Interface values are exchanged as subroutine parameters
 - ➔ **Example 1: Interface to MATLAB (NT)**
 SIMPACK model reading routines, equation generator and integrator are linked to MATLAB as S-Function.



- MATLAB/SIMULINK is only able to solve ODEs.
- Thus, a system consisting of constraint elements (DAEs) cannot be solved by MATLAB
- Solution: the mechanical system is solved by SIMPACK while MATLAB solves, for example, the control loop.
- Integrator stepsize is limited to sample time





Co-Simulation with SIMPACK

realized Co-Simulation Interfaces with SIMPACK

▶ One process Coupling (II)

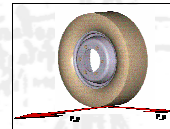
➔ Example 2: RmodK Tyre-model (VW Gedas)

the tyre model is linked to SIMPACK as discrete user force element. In a vehicle simulation using this tyre model the equations of the vehicle model is solved by SIMPACK the behaviour of the tyre is solved within the force element. The communication of the interacting values (like tyre forces, wheel position) is carried out by cosimulation

An application area of this szenario is handling (< 5Hz) and konfort simulation (< 20 Hz)

- + only one prozess has to be executed
- + optimum communication time

- compatibility problems may occur with a new release of one of the partners
- if identical function names are used in master/ slave or slave1/slave2 routines a coupling may be not successfull, or the cosimulation can only be carried out with one slave pro



Co-Simulation with SIMPACK

realized Co-Simulation Interfaces with SIMPACK

▶ Two process Coupling

two independent prozesses have to be started one by one. The communication is carried out via an 'exchange media'. The interconnecting values are stored and read from there. Communications for following 'exchange medias' were realized:

- Shared memory: the communicating prozesses, both on one computer, acces the same memory area. To avoid conflicts of the two prozesses accessing the memory at the same time, an access controlling has to be installed. Thcis can be done using semaphores
- Unix Sockets: both prozesses on an UNIX computer communiante via a socket which is created during the initialisation prozess. An extra access controlling is not necessary using Unix Sockets. Here both systems have to be installed on one computer
- Inet Sockets: with these Sockets all performances of UNIX Sockets are possible. Apart from that a communication via the Internet or an internal network is possible

- + coupling of the systems is rather independent to a version update of one or both partners
- + integration of several subsystems with equal function names is possible
- + Inet Sockets: communication via the Internet and a distribution of hardware requirements is possible

- for any simulation two prozesses have to be started
- possibly higher calculation time



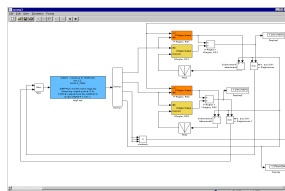


Co-Simulation with SIMPACK

realized Co-Simulation Interfaces with SIMPACK



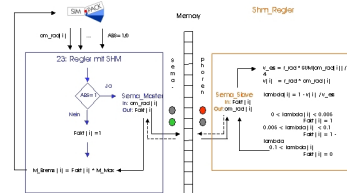
▶ Two process Coupling (II)



▶ **Example 1 Unix Socket:** Co-Simulation with MATLAB / MATRIXX (UNIX) the one process coupling as performed on NT did not work according to library conflicts

▶ **Example 2 Inet Socket:** Coupling to the hydraulic simulation program AMESIM and to the overhead system simulation program PROSIM from Deutsche Bahn AG

▶ **Example 3 Shared Memory:** quasi anti-skid controller of a SIMPACK vehicle model



Co-Simulation with SIMPACK

planned Co-Simulation Interfaces with SIMPACK

▶ Co-Simulation with MATLAB based on SIMPACK Symbolic Code

- SIMPACK DAE Integrator and the Symbolic Code of an arbitrary SIMPACK model are linked to MATLAB. The interconnecting vectors U, Y are exchanged at fixed sample rates.
- with that DAEs from a mechanic system can be solved in MATLAB without any SIMPACK installation

▶ Adaptation of the known MATLAB interface on Inet Sockets

- MATLAB and SIMPACK can be installed on two different computers. The Co-Simulation is carried out by a communication via the internet or an internal network
- enables the distribution of necessary hardware performance on two computers



International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

Co-Simulation with SIMPACK

some industrial applications of a CoSimulation with SIMPACK

- ▶ **Deutsche Bahn Ag, Munich**
simulation of the interaction of the overhead system and the brush contact.
Here the brush contact device was modelled in SIMPACK the overhead system in PROSA and the contact controller was modelled in MATRIX
- ▶ **CEIT, San Sebastian, Mechatronic Train Project**
here investigations were performed about possible applications of steering controllers for railway vehicles. The vehicles were modelled in SIMPACK the controller in MATLAB
- ▶ **Fairchild - Dornier, Oberpfaffenhofen**
Semiaktive landing gear for Dornier airplanes. Co-Simulation is carried out between SIMPACK and MATLAB
- ▶ **IWB, University of Technologie, Munich**
controlling of a CNC-steered machine-tool with SIMPACK and MATLAB
- ▶ **Adtranz, Winterthur**
anti-slipdriving controller for locomotives with SIMPACK and MATLAB
- ▶ many more



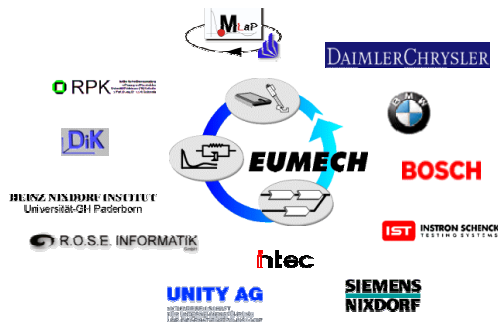
International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

EUMECH Project

Partners and focus

For the simulation of mechatronic systems several technical disciplines, such as mechanics, control systems, hydraulics etc have to be taken into account. To consider several physical aspects and their influence on each other within one simulation environment becomes more and more important



The focus of the Eumech Project (EC Mechatronic), where several partners from industry, universities and software houses participated, was to create an environment for the development of mechatronic systems.

Within the Eumch project several sub-projects were carried out, dealing with the field of application of the participating partners



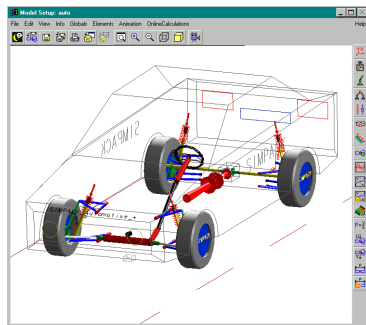


Multi-disciplinary simulation of an anti-skid system

Within the framework of the Eumech project, where the development of an simulation environment for mechatronic systems was discussed a vehicle model was created in cooperation of Intec, Weßling and the Mechatronic Laboratorium, Paderborn (MLaP). The model consisted of several technical disciplines, such as mechanics, hydraulics and control

Focus

The Focus of this sub-project was to investigate possibilities of data exchange between differnt systems and domains to revel and document connected problems. The situation, a simulation of a mechatronic system where several technical disciplines of different sources had to be put together is rather realistic.



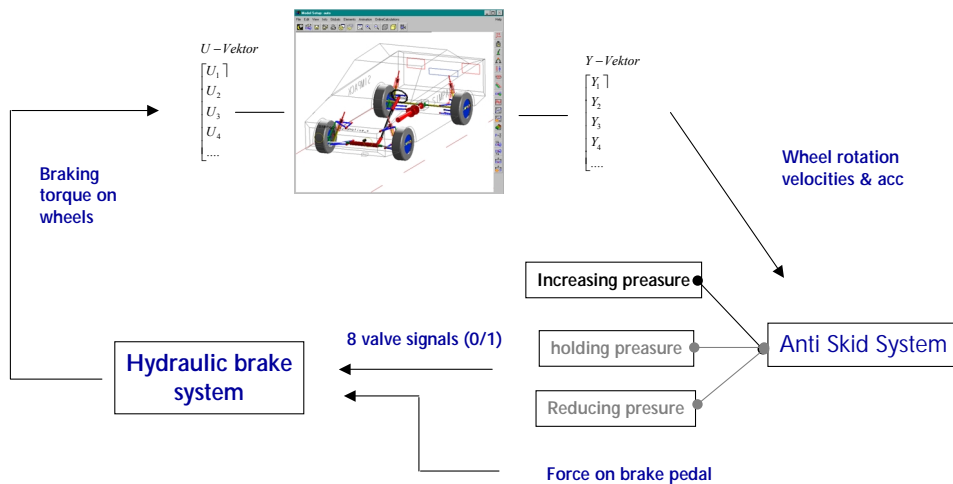
Simulation Szenario

Subject was the simulation of an anti skid system of a vehicle model. The effects of the anti skid system on the behaviour of the car were investigated, in casse of different friction coefficients of either side of the car (mue-split).



Multi-disciplinary simulation of an anti-skid system

Model descption



International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

Multi-disciplinary simulation of an anti-skid system

Model description

• Vehicle Model

The vehicle model was created at Intec with SIMPACK. The elements of the vehicle model were taken from the standard SIMPACK Automotive+ data base. It consisted of a double wishbone front- and a five link rear axis. For the tyre model the Pajeca Similarity method was used. Further more steering device and a simplified powertrain was modelled. For the anti skid system the wheel rotation velocities and accelerations were exported. The brake torques were imported from the hydraulic system.

• Anti skid logic

the anti skid logic and the hydraulics of the brake system was modelled at the MLaP with CAMeL. CAMeL is a simulation package to model and simulate control systems, hydraulics and mechanics. It was developed at the University of Paderborn and is distributed by iXtronics, Paderborn. For the logics of the anti skid the principle of Bosch's ABS 5 as used. Here three discrete positions for the valves of the brake oil cylinders are determined depending on wheel rotation velocities and accelerations. The three states are; increasing pressure, keeping pressure and reducing pressure. Altogether 8 discrete signals (0/1) for the valves of the brake cylinder were given to the hydraulic part.

• Hydraulic brake system

Here the oil cylinder for each wheel, the main cylinder and the valves were modelled with CAMeL hydraulic elements. Depending on the discrete states of the valves and a given force on the brake pedal the braking torque on each wheel was calculated and given to the mechanic system

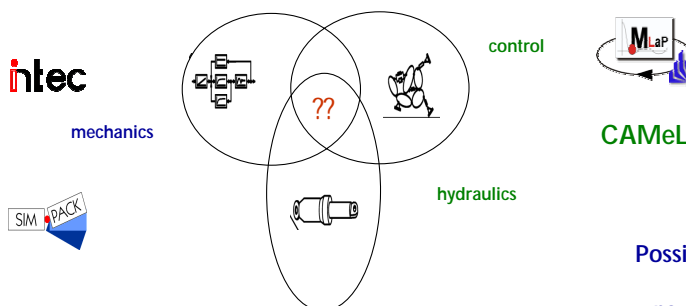


International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

Multi-disciplinary simulation of an anti-skid system

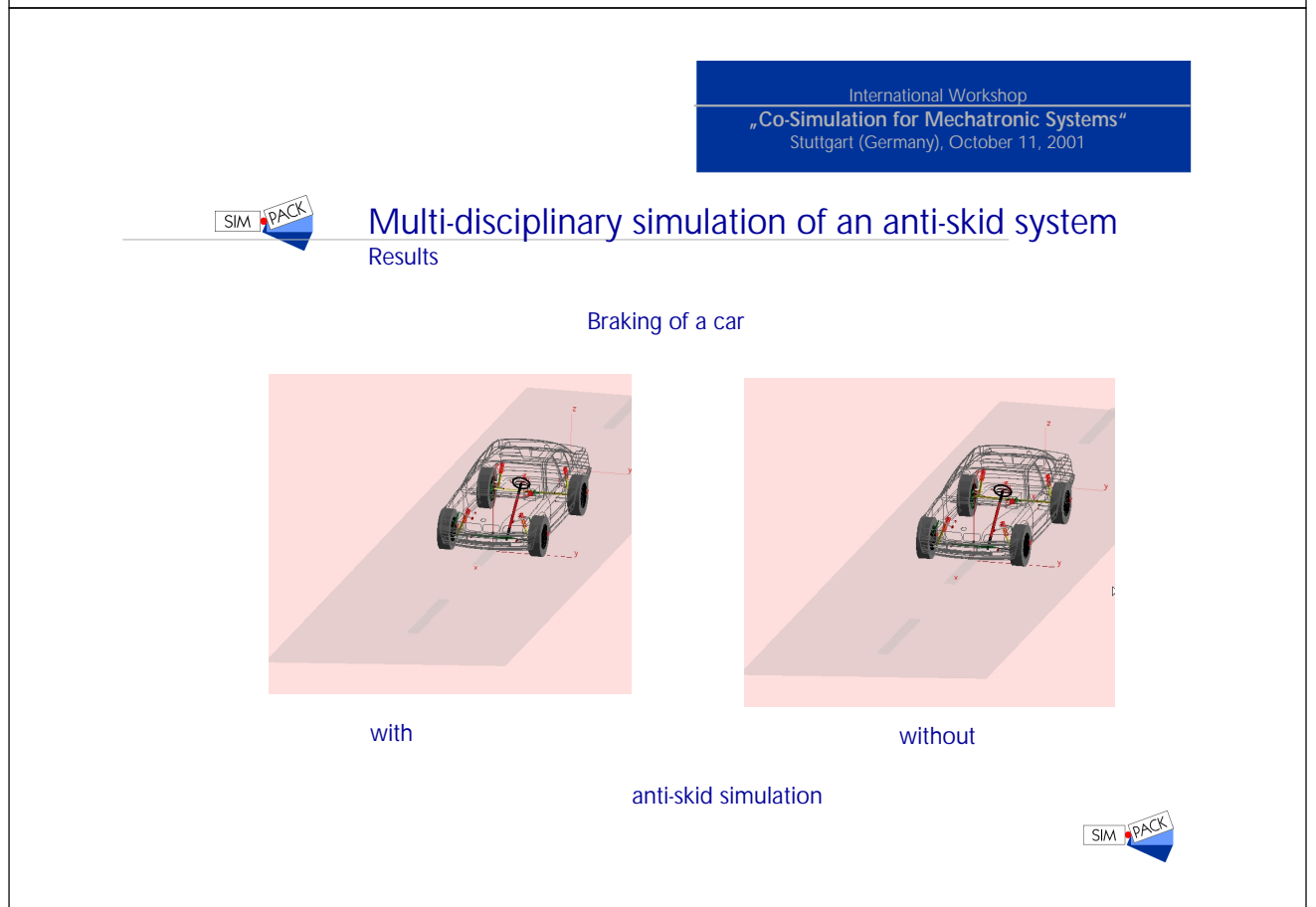
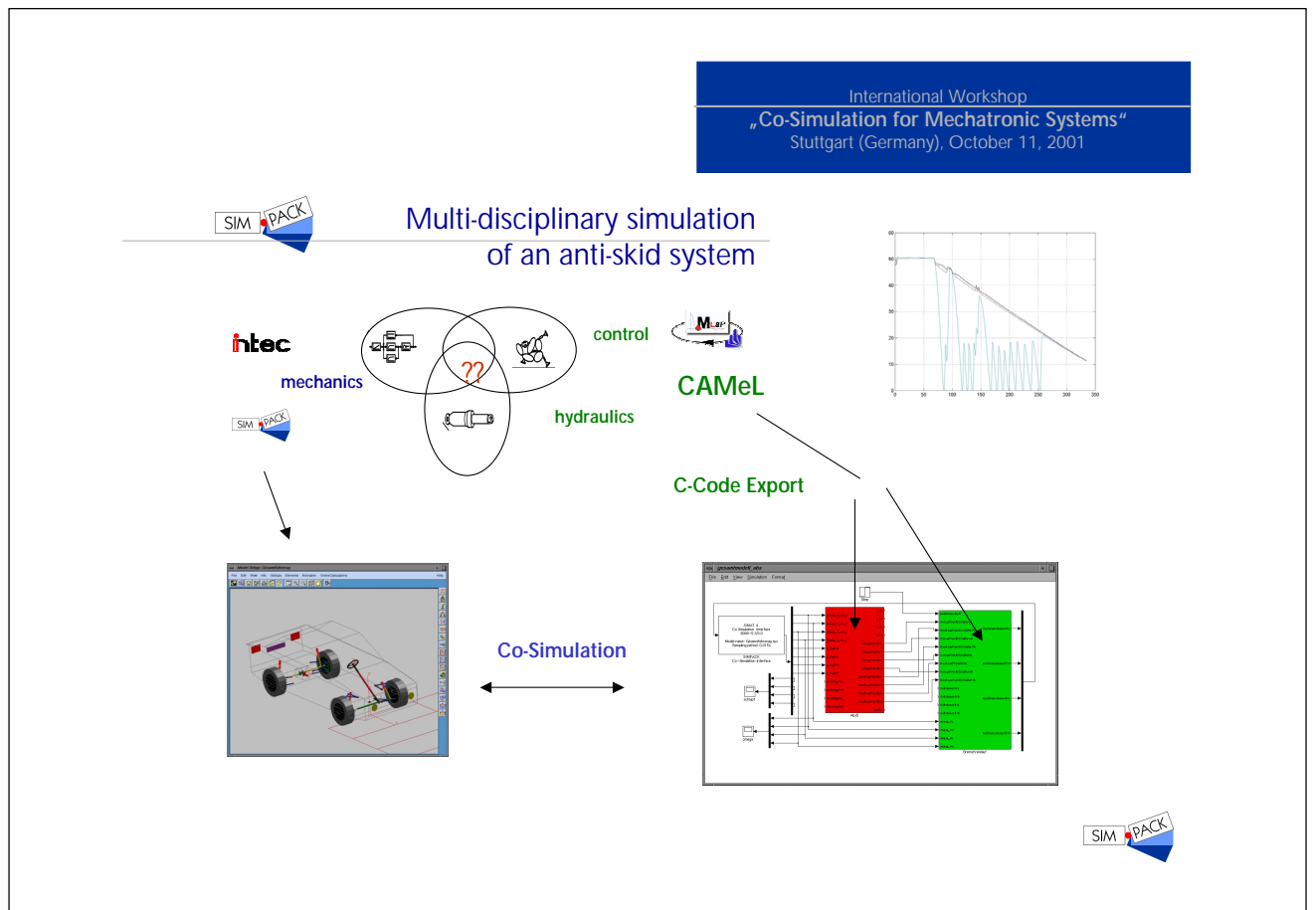
Model integration



Possible coupling methods:

- no direct interface
- SIMPACK Symbolic Code Export and link to CAMeL
-> no DAE solver in CAMeL
- CAMeL C-Code Export and link to SIMPACK (User Routine)
-> possible interface, needs further programming effort
- CAMeL connection to SIMPACKs IPC Co-Simulation Interface
-> possible; needs further programming effort, no advantages to upper solution





International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

Multi-disciplinary simulation of an anti-skid system

occured problems

- ▶ **Specialization yields to less flexibility**
Not all aspects of a mechatronic system can be modelled in one system. Highly specialised tools focus on the description of the physics in one special domain. Functionality of not-focussed domains have to be modelled either with less complexity or in another specialized tool
- ▶ **Need of communication devices**
In order to put submodels of different domains and systems together to one complete-simulation, the modelling and simulation tools need to have the capability of code in- and export or communication interfaces to other systems
- ▶ **Compatibility of solution methods**
The solution of the domain specific equation systems is based on specialized methods and konzept which are not necessarily compatible to those of other domains, e.g. DAE and ODE systems in mechanic and control simulation
- ▶ **Need of synchronization of sample times**
For the example as described above the Co-Simulation proved to be an appropriate means to solve the problems as listed above. Further problems occure if several Co-Simulations or additional discrete systems should be taken into account



International Workshop

„Co-Simulation for Mechatronic Systems“
Stuttgart (Germany), October 11, 2001

Conclusions

demands on future applications

- ▶ **Co-Simulation of several independent solver systems of differnt domains, with independent sample rates**
e.g. vehicle model, hydraulics, control, driver model
- ▶ **Realtime environment using Co-Simulation, Software-in-the-Loop (SIL), Hardware-in-the-Loop (HIL)**
- ▶ **Stabilization of initial steps for a Co-simulation of different domains**
e.g. Co-Simulation of mechanics and hydraulics
- ▶ **Dynamic step size control, to reduze calculation time**



Co-modelling of electric networks and heat evolution

Andreas Bartel, Michael Günther

{bartel, guenther}@iwrmm.math.uni-karlsruhe.de

**Institute of Scientific Computing and Mathematical Modelling
University of Karlsruhe, Engesser Str. 6, D – 78128 Karlsruhe, Germany**

Since the dimensions of chip technologies shrink and clock rates of CPUs increase, the semiconductor industry predicts power losses up to 40 W/cm^2 in the near future. Therefore temperature evolution needs to be simulated, too, in order to guarantee functionality of these devices. Starting from a benchmark network, which includes basic effects of heat production, heat conduction and temperature dependence, we discuss the modelling and resulting structures. Co-modelling yields a coupled system of the parabolic heat equation and differential algebraic equations, which describe the electric network. Finally, we present some strategies for co-simulation. This simulation type is not only desirable for its use of existing software, but also for exploiting the multirate behaviour of the coupled system.

Co-Simulation Workshop Stuttgart 2001

Co-modelling of electric networks and heat evolution

Andreas Bartel,
Michael Günther
IWRMM Karlsruhe

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Outline

(1)

1. Motivation
2. Starting from a benchmark
(coupled PDE and DAE)
3. General modelling
4. Semi-discretisation
5. Outlook



Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 1: Motivation

(2)

High Integrated Circuit – in near future:

- even larger number of devices
- growing package density

↪ power loss per cm² : 40W

i.e., coupled system: electric network & heat evolution

properties:

- DAEs and PDEs (parabolic)
- largely separated time scales: micro/nano seconds vs. seconds

simplification:

- 3D heat distribution ↪ 1D (and 0D) world

industrial solution (up to now): heat network, i.e., 0D approximation, corresponds to semi-discretisation of underlying PDE

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)



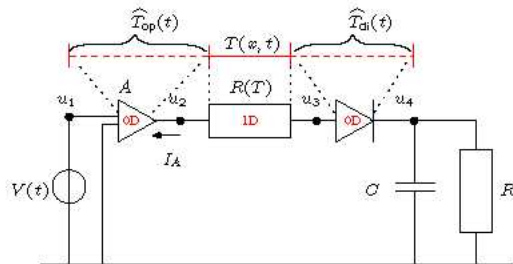
INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 2: Heat-Circuit Benchmark

(3)

• Network Equations:

$$\begin{aligned}
 I_A + \frac{u_2 - u_3}{R(T)} &= 0 \\
 -\frac{u_2 - u_3}{R(T)} + I_D(u_3 - u_4) &= 0 \\
 -I_D(u_3 - u_4) + C\dot{u}_4 + \frac{u_4}{R_L} &= 0 \\
 u_1 &= V(t) \\
 u_2 - Au_1 &= 0 \\
 & (+ \text{consistent initial values}) \quad \text{with} \quad I_D(\hat{T}_{di}) = \hat{I}_D(\hat{T}_{di})(\exp((u_3 - u_4)/v_T) - 1)
 \end{aligned}$$



• Heat Evolution:

1D Heat Conduction: $\rho_a c \frac{\partial T}{\partial t} = \frac{\partial}{\partial x} \lambda \frac{\partial T}{\partial x} + \frac{(u_2 - u_3)^2}{R^2 a_0^2} \rho(x, T) - \frac{\gamma S}{V} \cdot (T - T_{env})$

0D-Elements: $M_0 \hat{T}_{op}(t) = \lambda T_w(0, t) + cool_0(T(0, t)) + \kappa \cdot \boxed{(u_2 - u_1) \cdot I_A}$
 $M_1 \hat{T}_{di}(t) = -\lambda T_w(1, t) + cool_1(T(1, t))$

Initial: $T(x, 0) = T_0(t)$, Identification/BC: $\hat{T}_{op}(t) = T(0, t)$, $\hat{T}_{di} = T(1, t)$

Resistance: $R(T) = \int_0^l \rho(x, T)/a_0 dx$

Andreas Bartel / October 11, 2001

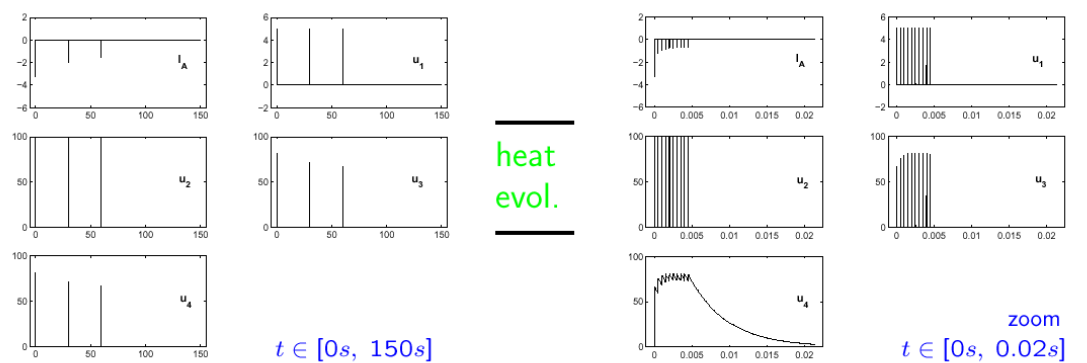
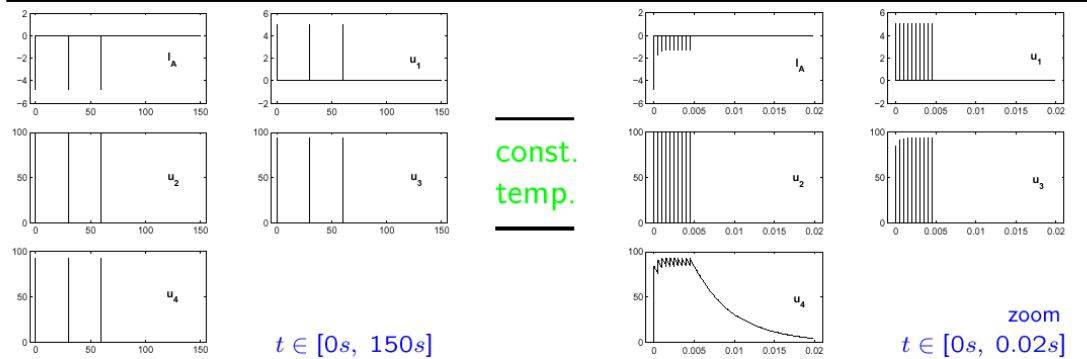
UNIVERSITÄT KARLSRUHE
(TH)



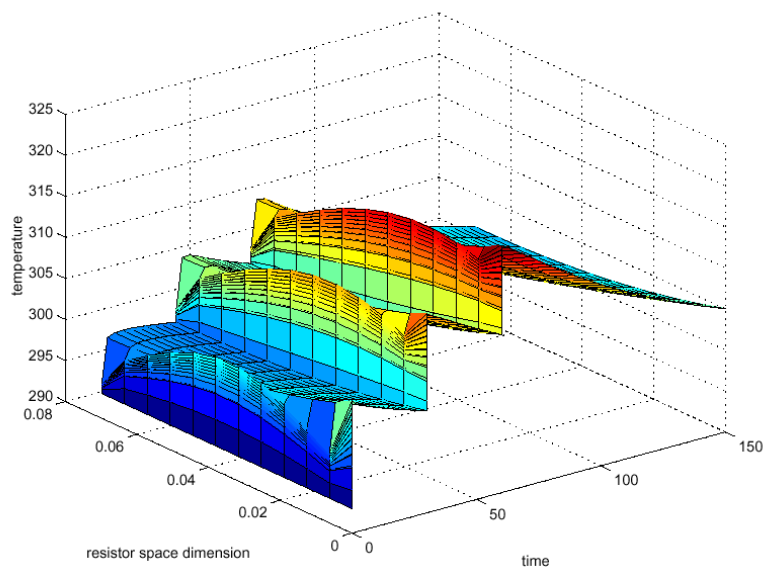
INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 2: Simulation

(4)



Heat evolution (von-Neumann-type BC)



Ch 2: Energy coupling

(5)

↪ physical coupling: via power

- circuit's power loss results in heat
- temperature distribution (1D/0D) specifies device parameters

Advantages:

- allows energy conservation
- co-simulation with 'large' communication step size (?)

Realisation: currents of thermal branches: j

↪ branch power: $E(t) = \text{diag}(j) A_{tp}^T u$

- transmission line / resistor: self-heating

$$+ \frac{(u_2 - u_3)^2}{R^2 a_0^2} \rho(x, T) \quad \rightsquigarrow \quad + \frac{E_{tr}}{R a_0^2} \rho(x, T)$$

- lumped elements: energy loss (transition coefficient κ)

$$+ \kappa |(u_2 - u_1) \cdot I_A| \quad \rightsquigarrow \quad + \kappa E_{lp}$$

Andreas Bartel / October 11, 2001

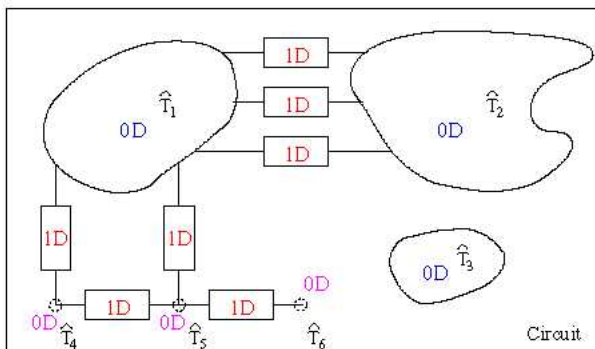
UNIVERSITÄT KARLSRUHE
(TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 3: Multiline

(6)



- Thermal branches: A_{tp}

1D elements: resistors, transmission lines, ...
0D elements: diodes, amplifiers, ...

$$A_{tp, tr} = B-D$$

$$T(x, t)$$

$$\hat{T}_{br}(t)$$

- form connected 0D units (k): 0D element-units (k_{lp}) + 1D-1D coupling nodes (k_{tr})

$$\hat{T} = (\hat{T}_{lp}, \hat{T}_{tr})$$

- form projectors: P projects thermal branches onto connected unit

$$\text{heat capacity per unit: } \hat{M} = PMP^T$$

$$\text{surface/volume ratio per unit: } \hat{F} = PFP^T$$

S projects network nodes onto connected unit:

$$B_{fac} = SB, D_{fac} = SD$$

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 3: Equations (7)

Circuit

$$\begin{aligned} 0 &= \mathbf{A}_C \dot{\mathbf{q}}_C(\mathbf{u}(t), t) + \mathbf{A}_G \mathbf{r}(\mathbf{u}(t), t, \hat{\mathbf{T}}_{br}(t), \mathbf{R}) + \mathbf{A}_I \mathbf{i}(t) + \mathbf{A}_V \mathbf{j}_V(t) \\ 0 &= \mathbf{A}_V^T \mathbf{u}(t) - \mathbf{v}(t) \end{aligned} \quad (\mathbf{A} = (\mathbf{A}_C, \mathbf{A}_G, \mathbf{A}_I, \mathbf{A}_V))$$

Coupling Interface

$$\begin{pmatrix} \mathbf{E}_{tr}(t) \\ \mathbf{E}_{lp}(t) \end{pmatrix} = \mathbf{E}(t) = \text{diag}(\mathbf{j}) \mathbf{A}_{tp}^T \mathbf{u} \quad (\mathbf{j} = \mathbf{J}_E(\mathbf{u}, \mathbf{j}_V)), \quad \hat{\mathbf{T}}_{br} = \mathbf{P}^T \hat{\mathbf{T}}, \quad \mathbf{R}$$

Heat

$$\begin{aligned} c_i \hat{T}_i(x, t) &= \nabla_* (\lambda_i \nabla_* T_i(x, t)) - \gamma F_i (T_i(x, t) - T_{env}) + \rho_i(x, T_i) \cdot \frac{\mathbf{E}_{tr,i}(t)}{a_i^2 \cdot R_i(x, T_i)} \\ 0 &= \nabla_* R_i - \rho_i(x, T_i(x, t)) / a_i \quad i = 1, \dots, m_{tr} \\ \hat{\mathbf{M}} \hat{\mathbf{T}}(t) &= (\mathbf{B}_{fac} \mathbf{Q}_0 - \mathbf{D}_{fac} \mathbf{Q}_1) (\Delta \nabla_* \mathbf{T}(x, t)) - \gamma \hat{\mathbf{F}} (\hat{\mathbf{T}} - T_{env} \mathbf{1}_k) + \kappa \mathbf{P} \mathbf{E}_{lp}(t) \\ \mathbf{Q}_0 \mathbf{T} &= \mathbf{B}_{fac}^T \hat{\mathbf{T}}, \quad \mathbf{Q}_1 \mathbf{T} = \mathbf{D}_{fac}^T \hat{\mathbf{T}} \end{aligned}$$

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 3: Abstract equations (8)

assume: semi-explicit network

DAE	$\begin{aligned} \dot{\mathbf{y}}_1 &= \mathbf{f}_1(\mathbf{y}_1, \mathbf{y}_{2b}, \mathbf{z}_{1a}, \mathbf{z}_{2b}) \\ 0 &= \mathbf{h}_{1a}(\mathbf{y}_1, \mathbf{y}_{2b}, \mathbf{z}_{1a}, \mathbf{z}_{2b}) \\ 0 &= \mathbf{h}_{1b}(\mathbf{y}_1, \mathbf{z}_{1a}, \mathbf{z}_{1b}) \\ 0 &= \mathbf{h}_{1c}(\mathbf{y}_1, \mathbf{z}_{1a}, \mathbf{z}_{1c}) \end{aligned}$	$\begin{aligned} \mathbf{y}_1 &= \mathbf{u}_1 \\ \mathbf{z}_{1a}^T &= (\mathbf{u}_2^T \quad \mathbf{j}_V^T) \\ \mathbf{z}_{1b} &= \mathbf{E}_{tr} \\ \mathbf{z}_{1c} &= \mathbf{E}_{lp} \end{aligned}$	CIRCUIT
PDE time & space	$\begin{aligned} \dot{\mathbf{y}}_{2a} &= \mathbf{f}_{2a}(\Delta \mathbf{y}_{2a}, \mathbf{y}_{2a}, \mathbf{z}_{1b}, \mathbf{z}_{2b}) \\ 0 &= \mathbf{h}_{2c}(\mathbf{y}_{2a}, \nabla \mathbf{z}_{2b}) \end{aligned}$	$\begin{aligned} \mathbf{y}_{2a} &= \mathbf{T} \\ \mathbf{z}_{2b} &= \mathbf{R} \end{aligned}$	HEAT
“DAE”	$\begin{aligned} \dot{\mathbf{y}}_{2b} &= \mathbf{f}_{2b}(\nabla^B \mathbf{y}_{2a}, \mathbf{y}_{2b}, \mathbf{z}_{1c}) \\ 0 &= \mathbf{h}_{2a}(\nabla^B \mathbf{y}_{2a}) \\ 0 &= \mathbf{h}_{2b}(\mathbf{y}_{2a}^B, \mathbf{y}_{2b}, \mathbf{z}_{2a}) \end{aligned}$	$\begin{aligned} \mathbf{y}_{2b} &= \hat{\mathbf{T}}_{lp} \\ \mathbf{z}_{2a} &= \hat{\mathbf{T}}_{tr} \\ & \text{(boundary)} \end{aligned}$	

(^B boundary, only)

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

Ch 4: Semi-Discretisation

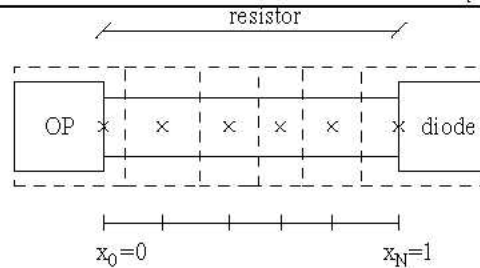
(9)

- grid I_h :

$$0 = x_0 < x_1 < \dots < x_N = 1$$

$$T_i = T(x_i, t)$$

$$h_i = x_{i+1} - x_i$$



- using finite volume technique:

$$(\rho_d c \cdot a_0 \frac{h_0}{2} + \rho_{Fe} c_{Fe} e_{op}^3) \hat{T}_{op} = a_0 \lambda \frac{T_1 - T_0}{h_0} + \frac{E_0}{R a_0^2} \rho(0, \hat{T}_{op}) \cdot a_0 \frac{h_0}{2} - \gamma \cdot (F \cdot a_0 \frac{h_0}{2} + 6 e_{op}^2) \cdot (\hat{T}_{op} - T_{env}) + \kappa E_{op}$$

$$\rho_d c T_i = \frac{2 \lambda}{h_i + h_{i-1}} \left(\frac{T_{i+1} - T_i}{h_i} - \frac{T_i - T_{i-1}}{h_{i-1}} \right) + \frac{E_0}{R a_0^2} \rho(x_i, T_i) - \gamma F \cdot (T_i - T_{env}) \quad (i = 1, \dots, N-1)$$

$$(\rho_d c \cdot a_0 \frac{h_{N-1}}{2} + \rho_{Si} c_{Si} e_{di}^3) \hat{T}_{di} = a_0 \lambda \frac{T_{N-1} - T_N}{h_{N-1}} + \frac{E_0}{R a_0^2} \rho(1, \hat{T}_{di}) \cdot a_0 \frac{h_{N-1}}{2} - \gamma \cdot (F \cdot a_0 \frac{h_{N-1}}{2} + 6 e_{di}^2) \cdot (\hat{T}_{di} - T_{env})$$

→ ODEs for all connected (0D) units (+ algebraic BC as identification: $\hat{T}_{op} = T_0, \hat{T}_{di} = T_N$)

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE (TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN UND MATHEMATISCHE MODELLBILDUNG

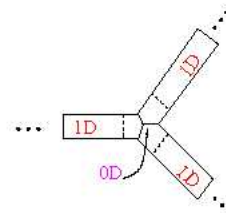
Ch 4: Semi-Discretisation (II) – overall

(10)

DAE — (1) using identifications:

$$\begin{cases} \dot{y}_{2a} = f_{2a}^h(\Delta^h(y_{2a}, z_{2a}), \boxed{z_{1b}}, z_{2b}) \\ 0 = h_{2a}^h(y_{2a}, z_{2a}, y_{2b}) \\ \dot{y}_{2b} = f_{2b}^h(y_{2a}, z_{2a}, y_{2b}, \boxed{z_{1c}}, z_{2b}) \\ 0 = z_{2b} - h_{2c}^h(y_{2a}, z_{2a}) \end{cases} \quad \left. \begin{array}{l} y_{2a} = (T_i)_{i=1}^{N-1} \\ z_{2a}^\top = (T_0^\top, T_N^\top) \\ y_{2b} = \hat{T} \\ z_{2b} = R \end{array} \right\} \begin{array}{l} \text{idea } w^\top = (y_{2a}^\top, z_{2a}^\top) \\ \mathbf{A} \dot{w} = f(\Delta^h w, \dots) \end{array}$$

- (2) using plugged in discretisation: no heat capacity for 1D-1D coupling



ODE — using minimal number of unknowns:

$$\begin{cases} \dot{y}_{2a} = f_{2a}^h(\Delta^h y_{2a}, y_{2a}, y_{2c}, \boxed{z_{1b}}, z_{2b}) \\ \dot{y}_{2c} = f_{2c}^h(y_{2a}, y_{2c}, \boxed{z_{1c}}, z_{2b}) \\ 0 = z_{2b} - h_{2c}^h(y_{2a}, y_{2c}) \end{cases} \quad \begin{array}{l} y_{2a} = (T_i)_{i=1}^{N-1} \\ y_{2c} = \hat{T} \\ z_{2b} = R \end{array}$$

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE (TH)



INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN UND MATHEMATISCHE MODELLBILDUNG

Ch 4: Semi-Discretisation (III) (11)

- Waveform Relaxation: **energy coupling**

$$\begin{pmatrix} \mathbf{z}_{1b} \\ \mathbf{z}_{1c} \end{pmatrix} \rightsquigarrow \dot{\mathbf{y}}_{1b}, \quad \text{i.e.,} \quad \mathbf{y}_{1b} = \int_0^t \mathbf{E}(\tau) d\tau$$

structure

$$\dot{\mathbf{y}}_{1a} = \mathbf{f}_{1a}(\mathbf{y}_{1a}, \mathbf{y}_2, \mathbf{z}_1)$$

$$\dot{\mathbf{y}}_{1b} = \mathbf{f}_{1b}(\mathbf{y}_{1a}, \mathbf{z}_1)$$

$$\mathbf{0} = \mathbf{h}_1(\mathbf{y}_{1a}, \mathbf{y}_2, \mathbf{z}_1)$$

$$\dot{\mathbf{y}}_2 = \mathbf{f}_2^h(\mathbf{y}_2, \mathbf{y}_{1b})$$

+ possibly algebraic equations

DAE–ODE (DAE) coupling
no special coupling equation

} simple case \rightsquigarrow solution for WR

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNGCh 5: Outlook (12)**What has been done:**

- modelling: first layout of power interface \rightarrow abstract coupled system

- Co-Simulation: (without energy coupling)

\rightsquigarrow fixed communication step size (H), small (!) \rightarrow 1 iteration suffices

\rightsquigarrow interface: $\mathbf{y}(t_0)$, $\mathbf{y}(t_0 + H)$ + linear interpolation

What is next:

- investigation of abstract structure
- comparison with Infineon's strategy (network approach)
- controlling mechanism for communication step size
- error estimator (e.g. on basis of energies and available Jacobians)

Andreas Bartel / October 11, 2001

UNIVERSITÄT KARLSRUHE
(TH)INSTITUT FÜR WISSENSCHAFTLICHES RECHNEN
UND MATHEMATISCHE MODELLBILDUNG

The stabilization of time integration methods for co-simulation

Martin Arnold

`martin.arnold@dlr.de`

**DLR German Aerospace Center, Vehicle System Dynamics Group
P.O. Box 1116, D – 82230 Wessling, Germany**

The increasing integration of mechanical, hydraulic and electric components is one of the major challenges in the dynamical simulation of modern technical systems. A very straightforward approach to the simulation of these coupled systems is based on the coupling of several standard simulation tools by co-simulation. The dynamical behaviour of the subsystems is simulated separately in different simulation tools. At discrete synchronization points data are exchanged between the tools.

From a numerical point of view co-simulation results in a modular time integration of the coupled system. For each subsystem a classical time integration method is used with stepsize and order being adapted to the solution behaviour of this individual subsystem. The data exchange between subsystems is based on the interpolation between synchronization points. Alternatively extrapolation has to be used if the data from other subsystems is not yet available. Interpolation and extrapolation introduce additional numerical errors in the time integration. These errors are kept small as long as the subsystems are coupled weakly. In general, however, the extrapolation may cause numerical instability.

Recently, this instability phenomenon has been analysed for systems that are coupled by constraints. Several strategies have been proposed to guarantee a stable error propagation: non-linear projection steps (Kübler and Schiehlen), linear projection steps (Tseng and Hulbert) and overrelaxation techniques (Arnold and Günther).

In the present paper the common ideas and the differences between these three closely related strategies are studied in detail. Furthermore the stable modular time integration is analysed for mechanical systems that are strongly coupled by stiff springs.

DLR German Aerospace Center

Vehicle System Dynamics Group

The stabilization of time integration methods for co-simulation

2nd International Workshop on Co-Simulation
University of Stuttgart, October 2001

Martin Arnold

DLR German Aerospace Center, Institute of Aeroelasticity, Vehicle System Dynamics Group
P.O. Box 1116, D – 82230 Wessling, Germany
<http://www.ae.op.dlr.de/~arnold>, **email:** martin.arnold@dlr.de



DLR German Aerospace Center

Vehicle System Dynamics Group

Outline

1. Constrained coupled systems
2. Multibody systems (MBS) and large elastic structures
 - Dynamical interaction vehicle / bridge
 - Dynamical interaction pantograph / catenary
3. Exponential instability of coupled time integration
 - Coupled time integration of 2 subsystems, stability analysis
 - Stabilization strategies, stabilization and projection
 - Overlapping modular time integration
4. Summary



Constrained coupled systems

$$\begin{cases} \dot{y}_i = \varphi_i(y_1, \dots, y_l, w), & (i=1, \dots, l) \\ 0 = \gamma(y_1, \dots, y_l, w) \end{cases}$$

- Differential variables y_i in l subsystems
- Algebraic variables w :
 - internal **algebraic** variables of the subsystems
 - **coupling** of the subsystems
- Index-1 system: $\partial\gamma/\partial w$ non-singular
- Analytical transformation of higher index systems

$$\left. \begin{cases} M\ddot{q} = f(q, \dot{q}) - B^T(q)\lambda \\ 0 = b(q) \end{cases} \right\} \Rightarrow \begin{cases} \dot{q} = v \\ \dot{v} = a \\ 0 = Ma - f(q, v) + B^T(q)\lambda \\ 0 = B(q)a + b_{qq}(q)(\dot{q}, \dot{q}) \end{cases}$$

Constraint has to be differentiated (2x)



Special case: Two coupled systems

$$\dot{y}_1 = f_1(y, z_1)$$

$$0 = h_1(y, z_1, u)$$

$$\dot{y}_2 = f_2(y, z_2)$$

$$0 = h_2(y, z_2, u)$$

$$0 = g(y, z)$$

Index-1 conditions

Jacobians $\frac{\partial h_i}{\partial z_i}$, $\begin{pmatrix} \frac{\partial h_i}{\partial z_i} & \frac{\partial h_i}{\partial u} \\ \frac{\partial g}{\partial z_i} & 0 \end{pmatrix}$, $\begin{pmatrix} \frac{\partial h}{\partial z} & \frac{\partial h}{\partial u} \\ \frac{\partial g}{\partial z} & 0 \end{pmatrix}$ non-singular

$$\dot{q}_1 = v_1, \quad \dot{v}_1 = a_1$$

$$0 = M_1 a_1 - f_1(q, v) + B_1^T \lambda$$

$$\dot{q}_2 = v_2, \quad \dot{v}_2 = a_2$$

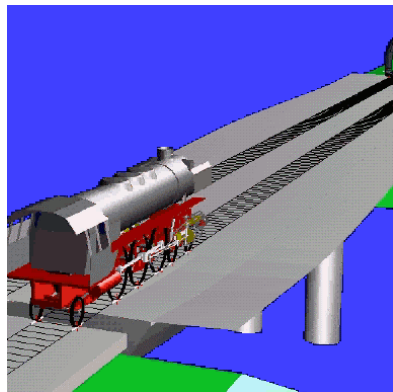
$$0 = M_2 a_2 - f_2(q, v) + B_2^T \lambda$$

$$0 = B_1 a_1 + B_2 a_2 + b_{qq}(q)(v, v)$$



Interaction of MBS and large elastic structures

Example 1: Vehicle / Bridge

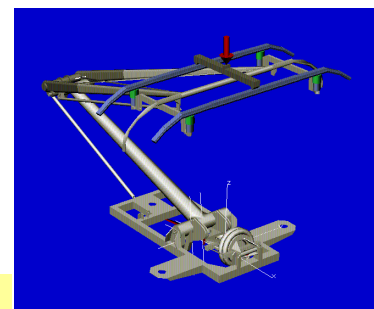
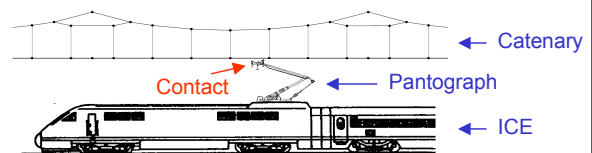


© S. Dietz (2001)

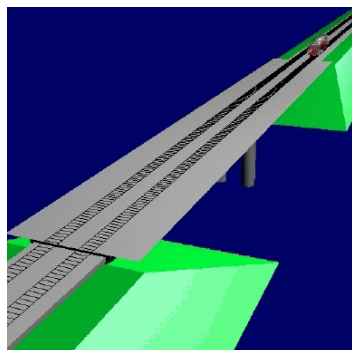
- Vehicle: Multibody system, nonlinear, ≈ 100 DOF
- Elastic structure: FE model, \approx linear, $\gg 5000$ DOF

Example 2: Catenary / Pantograph

© A. Veitl (1999)



Dynamical interaction vehicle / bridge

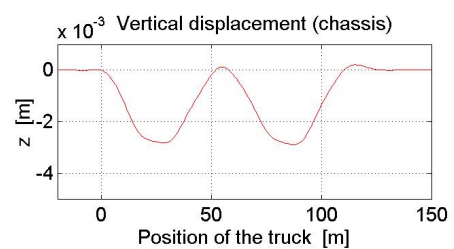
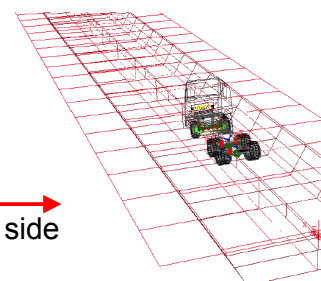


© S. Dietz (2001)

Coupling by

constraints
(wheel/rail contact)

right hand side
(tyre forces)



- Dynamical loads on bridge
- Damage of structures
- “Road-friendly truck”



Interaction vehicle / bridge: Two approaches

(A) Guideway operator (Duffek 1991, Duffek / Schupp 2001)

- Bridge modelled as beam structure
- Analytical computation of eigenmodes (Frýba), modal reduction
- Constant speed \Rightarrow (Semi-)analytical solution of equations of motion

(B) FEM + modal reduction (Dietz / Schupp / Hippmann / A. 2001)

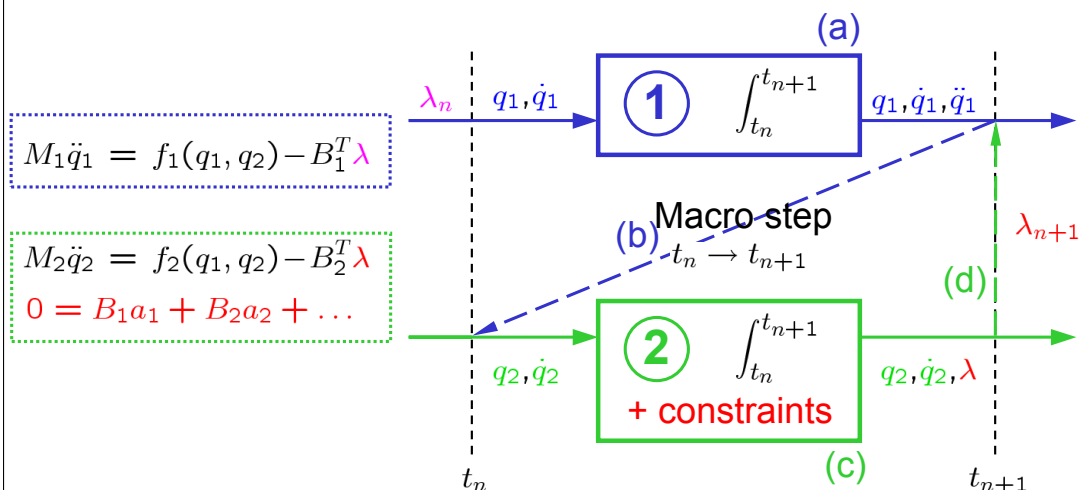
- Bridge: FEM model, modal reduction (ANSYS, NASTRAN, ...)
- Eigenmodes are given pointwise, cubic Hermite interpolation
- Linear approximation of loads in each macro step, analytical solution

Co-Simulation: macro step (Idea: Duffek 1991)

- Time integration BRIDGE for given loads
- Deformation of elastic structure defines "track irregularities"
- Time integration VEHICLE
- State of multibody system defines interaction forces



Co-Simulation vehicle / bridge: Algorithm



Subsystem 1: BRIDGE (elastic structure)
 Subsystem 2: VEHICLE (multibody system)
 Typical macro stepsize: 1.0 ms



DLR German Aerospace Center
Vehicle System Dynamics Group

Co-Simulation pantograph / catenary: Instability

Subsystem 1: CATENARY
(elastic structure)

Subsystem 2: PANTOGRAPH
(multibody system)

Macro stepsize H

Subsystem 1: PANTOGRAPH
(multibody system)

Subsystem 2: CATENARY
(elastic structure)

Macro stepsize H

Benchmark problem 1000 DOF, (A./Simeon 1999)

DLR German Aerospace Center
Vehicle System Dynamics Group

Stabilization by low level modifications

$$\begin{aligned}
 M_1 \ddot{q}_1 &= f_1(q_1, \dot{q}_1, q_2, \dot{q}_2) - B_1^T \lambda \\
 M_2 \ddot{q}_2 &= f_2(q_1, \dot{q}_1, q_2, \dot{q}_2) - B_2^T \lambda \\
 0 &= B_1 a_1 + B_2 a_2 + b_{qq}(q)(v, v)
 \end{aligned}$$

Stability constant $\alpha = \|[(B_2 M_2^{-1} B_2^T)^{-1} (B_1 M_1^{-1} B_1^T)](q_1, q_2)\|$

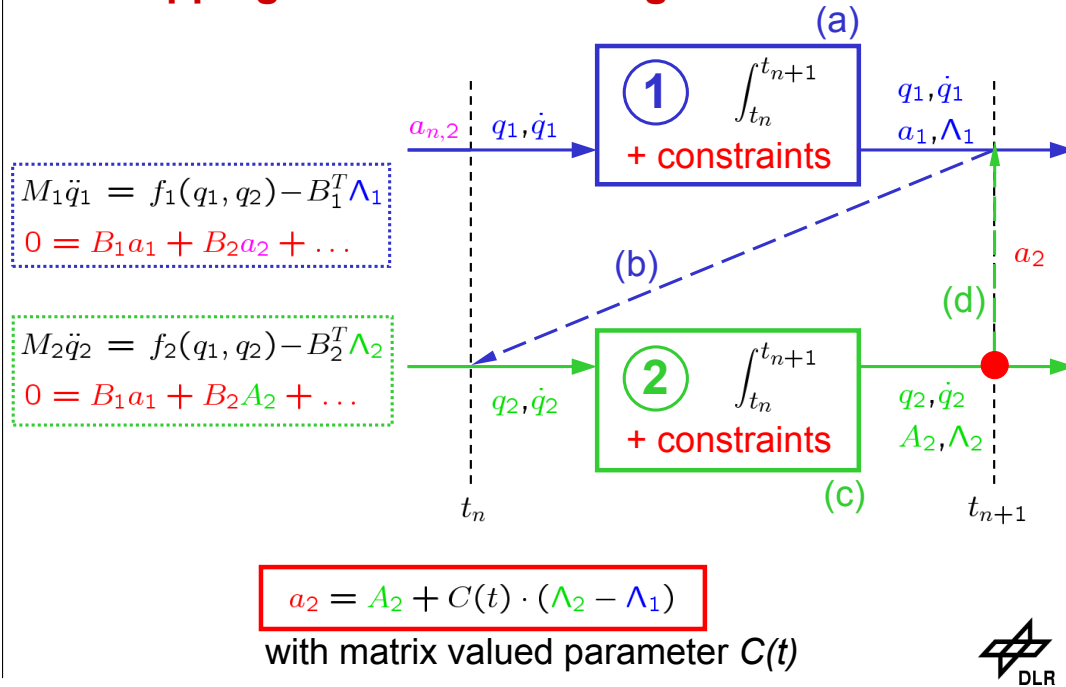
- Stable integration BRIDGE / VEHICLE $\alpha = 0.6$
- Unstable integration CATENARY / PANTOGRAPH $\alpha = 31.0$
- Stable integration PANTOGRAPH / CATENARY $\alpha = 0.03$

Contractivity condition $\alpha < 1$

- “Correct” order of subsystems (circuit simulation)
- Constraints attached to

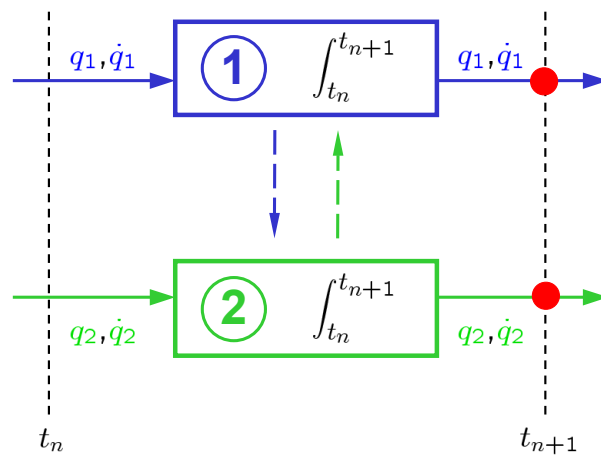
1 st subsystem	2 nd subsystem	Both subsystems

Overlapping modular time integration



Stable co-simulation of 2 coupled systems

- Kübler/Schiehlen (2000)
Nonlinear projection steps
Broyden's method
- Park (2000)
Staggered time integration
- Tseng/Hulbert (1999)
Linear projection steps
- A./Günther (2001)
Overrelaxation technique
Overlapping methods



Stabilization and projection

Index reduction

$$0 = b(q_1(t), q_2(t)) \Rightarrow 0 = \frac{d^2}{dt^2} b(q_1(t), q_2(t)) = B_1 a_1 + B_2 a_2 + \dots$$

Drift-off effect

$$\left\| \frac{d^2 b}{dt^2} \right\| = \mathcal{O}(\text{TOL}) \quad \text{BUT:} \quad \|b(q_1(t), q_2(t))\| = c_1 t^2 + c_2 t$$

Projection

(P) Compute $\Delta q_1, \Delta q_2$ such that $0 = b(q_1 + \Delta q_1, q_2 + \Delta q_2)$.

Update $q_1 := q_1 + \Delta q_1, q_2 := q_2 + \Delta q_2$.



Stabilization and projection (II)

(V) Compute $\Delta \dot{q}_1, \Delta \dot{q}_2$ with $0 = B_1 \cdot (\dot{q}_1 + \Delta \dot{q}_1) + B_2 \cdot (\dot{q}_2 + \Delta \dot{q}_2)$

$$\begin{pmatrix} M_1 & 0 & B_1^T \\ 0 & M_2 & B_2^T \\ B_1 & B_2 & 0 \end{pmatrix} \begin{pmatrix} \Delta \dot{q}_1 \\ \Delta \dot{q}_2 \\ \bar{\mu} \end{pmatrix} = - \begin{pmatrix} 0 \\ 0 \\ B_1 \dot{q}_1 + B_2 \dot{q}_2 \end{pmatrix}$$

Update $\dot{q}_1 := \dot{q}_1 + \Delta \dot{q}_1, \dot{q}_2 := \dot{q}_2 + \Delta \dot{q}_2$.

Stabilization

Example: standard Gauss-Seidel (no overlapping)

(L) Standard co-simulation gives approximation \wedge

Compute λ_{n+1} as solution of

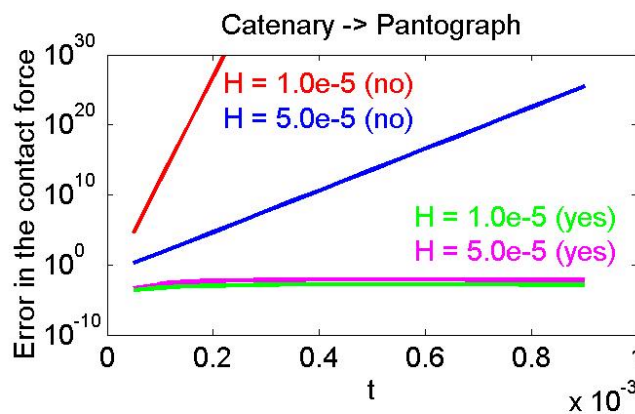
$$\begin{pmatrix} M_1 & 0 & B_1^T \\ 0 & M_2 & B_2^T \\ B_1 & B_2 & 0 \end{pmatrix} \begin{pmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \lambda_{n+1} \end{pmatrix} = \begin{pmatrix} B_1^T \lambda_n \\ B_2^T \wedge \\ 0 \end{pmatrix}$$



Stabilization and projection (III)

- Coupled conservative mechanical systems: special case
- Stabilization by linear projections / linear transformations
- Extension of standard DAE techniques, efficient implementation

Example



Stabilization: General approaches

Coupled systems

Finite number of subsystems, nonlinear constraints

$$\begin{cases} \dot{y}_i = \varphi_i(y_1, \dots, y_l, w), & (i=1, \dots, l) \\ 0 = \gamma(y_1, \dots, y_l, w) \end{cases}$$

Kübler / Schiehlen (2000)

Constraints in fixed point form $w = c(y, w)$

(a) Extrapolate algebraic components to get approximation w_n

(b) Time integration to solve $\dot{y}_i = \varphi_i(y_1, \dots, y_l, w_n), \quad (i=1, \dots, l)$

(c) Solve system of nonlinear equations

$$0 = \gamma(y, w_{n+1}) := w_{n+1} - c(y, w_{n+1})$$

by Newton's method with initial guess $w_{n+1}^{(0)} := c(y, w_n)$



Stabilization: General approaches (II)

$$\begin{cases} \dot{y}_i = \varphi_i(y_1, \dots, y_l, w), & (i=1, \dots, l) \\ 0 = \gamma(y_1, \dots, y_l, w) \end{cases}$$

Overlapping modular time integration (A. 2001)

- (a) Extrapolate algebraic components to get approximation w_n .
- (b) Assign each constraint to $r \geq 1$ subsystems.
- (c) Time integration of r differential-algebraic systems

$$\begin{cases} \dot{y}_i = \varphi_i(y_1, \dots, y_l, (I - P_i P_i^T)w_n + (P_i P_i^T)W_i), \\ 0 = P_i^T \gamma(y_1, \dots, y_l, (I - P_i P_i^T)w_n + (P_i P_i^T)W_i) \end{cases}$$

to get $y_i, P_i^T W_i$.

- (d) Define w_{n+1} as weighted linear combination of $w_n, P_1^T W_1, \dots, P_l^T W_l$.
Choose weight matrices such that stability is guaranteed.

**Stabilization: General approaches (III)**

Stability guaranteed with weight matrices

$$C_i(t) = \left(\frac{\partial \gamma}{\partial w}(y, w) \right)^{-1} P_i P_i^T D^{-1} P_i P_i^T \frac{\partial \gamma}{\partial w}(y, w) \quad \text{with} \quad D = \sum_{i=1}^l P_i P_i^T$$

Overlapping modular time integration with special weight matrices

=

One Newton step of Kübler's iterative method

Newton's method:

- + Stability guaranteed by one iteration step
- Time consuming evaluation of Jacobians

Broyden's method:

- + Re-evaluation of Jacobian avoided by update formula
- Stability guaranteed only after convergence of the iteration

Structure exploiting methods (e.g. “correct” order of subsystems)

- + Coarse approximations or even no Jacobians needed
- Depends strongly on special problem structure



Summary

Industrial applications

Dynamical interaction vehicle / bridge
Dynamical interaction pantograph / catenary

Stable time integration methods for co-simulation

Constrained coupled systems
Exponential instability of "standard" methods
Stabilization and projection for coupled mechanical problems
Overlapping modular time integration
Comparison of different approaches

Open problems, future work

Implementation in a general purpose software package
Stabilization for stiff coupled ordinary differential equations



References

- [1] M. Arnold. A pre-conditioned method for the dynamical simulation of coupled mechanical multibody systems. *Z. Angew. Math. Mech.*, Supplement 3 to Vol. 80:S817–S818, 2000.
- [2] M. Arnold and M. Günther. Preconditioned dynamic iteration for coupled differential-algebraic systems. *BIT Numerical Mathematics*, 41:1–25, 2001.
- [3] J.C. Chiou, K.C. Park, and C. Farhat. A natural partitioning scheme for parallel simulation of multibody systems. *Intern. Journal for Num. Math. in Engineering*, 36:945–967, 1993.
- [4] S. Dietz, M. Arnold, W. Duffek, G. Schupp, and G. Hippmann. Co-Simulation of MBS and FEM for the simulation of vehicles crossing a bridge, (in German). In M. Arnold, editor, *Working materials of the Workshop "Fahrzeug-Fahrweg-Wechselwirkung", March 5, 2001, Oberpfaffenhofen*, pages 20–28, DLR German Aerospace Center, Technical Report IB 532-01-04, March 2001.
- [5] R. Kübler. *Modulare Modellierung und Simulation mechatronischer Systeme*. Fortschritt-Berichte VDI Reihe 20, Nr. 327. VDI-Verlag GmbH, Düsseldorf, 2000.
- [6] R. Kübler and W. Schiehlen. Two methods of simulator coupling. *Mathematical and Computer Modelling of Dynamical Systems*, 6:93–113, 2000.
- [7] K.C. Park, J.C. Chiou, and J.D. Downer. Explicit-implicit staggered procedure for multibody dynamics analysis. *Journal of Guidance, Control and Dynamics*, 13:562–570, 1990.
- [8] F.C. Tseng and G.M. Hulbert. Network-distributed multibody dynamics simulation — gluing algorithm. In J.A.C. Ambrósio and W.O. Schiehlen, editors, *Advances in Computational Multibody Dynamics*, pages 521–540, IDMEC/IST Lisbon, Portugal, 1999.
- [9] O. Vaculin, W. Krüger, and M. Spieck. Coupling of multibody and control simulation tools for the design of mechatronic systems. In *Proceedings of DETC2001, ASME Design Engineering Technical Conferences DETC3001/VIB-21323*, Pittsburgh, 2001.
- [10] A. Veitl and M. Arnold. Coupled simulation of multibody systems and elastic structures. In J.A.C. Ambrósio and W.O. Schiehlen, editors, *Advances in Computational Multibody Dynamics*, pages 635–644, IDMEC/IST Lisbon, Portugal, 1999.
- [11] A. Veitl, T. Gordon, A. van de Sand, M. Howell, M. Valásek, O. Vaculin, and P. Steinbauer. Methodologies for coupling simulation models and codes in mechatronic system analysis and design. In *Proc. of the 16th IAVSD-Symposium on Dynamics of Vehicles on Roads and Tracks*, pages 231–243. Supplement to Vehicle System Dynamics, Vol. 33, Swets & Zeitlinger, 1999.



