SNE SIMULATION NOTES EUROPE

SNE Promotion Issue
MATHMOD 2022 & ASIM 2022

Journal on Developments and Trends in Modelling and Simulation

EUROSIM Scientific Membership Journal

Vol. 31 No.4, Dec. 2021
ISSN Online 2306-0271
ISSN Print 2305-9974
DOI 10.11128/sne.31.4.1058
ISBN Print 978-3-903311-18-3

TU Verlag
an der Technischen Universität Wien

ARGESIM

SNE - Aims and Scope

Simulation Notes Europe (SNE) provides an international, high-quality forum for presentation of new ideas and approaches in simulation - from modelling to experiment analysis, from implementation to verification, from validation to identification, from numerics to visualisation - in context of the simulation process.

SNE seeks to serve scientists, researchers, developers and users of the simulation process across a variety of theoretical and applied fields in pursuit of novel ideas in simulation and to enable the exchange of experience and knowledge through descriptions of specific applications. **SNE** follows the recent developments and trends of modelling and simulation in new and/or joining application areas, as complex systems and big data. **SNE** puts special emphasis on the overall view in simulation, and on comparative investigations, as benchmarks and comparisons in methodology and application. For this purpose, **SNE** documents the **ARGESIM Benchmarks** on *Modelling Approaches and Simulation Implementations* with publication of definitions, solutions and discussions. SNE welcomes also contributions in education in/for/with simulation.

A *News Section* in **SNE** provides information for **EUROSIM** Simulation Societies and Simulation Groups.

SNE, primarily an electronic journal, follows an open access strategy, with free download in basic layout. **SNE** is the official membership journal of **EUROSIM**, the *Federation of European Simulation Societies and Simulation Groups* – www.eurosim.info. Members of **EUROSIM** societies are entitled to download **SNE** in an elaborate and extended layout, and to access additional sources of benchmark publications, model sources, etc. **Print SNE** is available for specific groups of **EUROSIM** societies, and starting with Volume 27 (2017) as print-on-demand from TU Verlag, TU Wien. **SNE** is DOI indexed by CrossRef, identified by DOI prefix 10.11128, assigned to the **SNE** publisher **ARGESIM** (*www.argesim.org*).

Author's Info. Individual submissions of scientific papers are welcome, as well as post-conference publications of contributions from conferences of **EUROSIM** societies. **SNE** welcomes special issues, either dedicated to special areas and/or new developments, or on occasion of events as conferences and workshops with special emphasis.

Authors are invited to submit contributions which have not been published and have not being considered for publication elsewhere to the **SNE** Editorial Office.

SNE distinguishes different types of contributions (*Notes*), i.e.

- **TN** Technical Note, 6–10 p.
- **SN** Short Note, max. 5 p.
- **SW** Software Note , 4–6 p.
- **BN** Benchmark Note, 2–10 p.
- **ON** Overview Note – only upon invitation, up to 14 p.
- **EN** Education Note, 6–8 p.
- **PN** Project Note 6–8 p.
- **STN** Student Note, 4-6 p., on supervisor's recommendation
- **EBN** Educational Benchmark Note, 4–10 p.

Further info and templates (doc, tex) at **SNE**'s website.

www.sne-journal.org

SNE Editorial Board

SNE - Simulation Notes Europe is advised and supervised by an international scientific editorial board. This (increasing) board is taking care on peer reviewing of submission to SNE:

Editorial

***Dear Readers**, This SNE issue SNE 31(4), the SNE Promotion Issue 'MATHMOD 2022 & ASIM 2022' is a novelty, it is a 'special' special issue. SNE 31(4) aims for promotion of the conferences MATHMOD 2022 and ASIM 2022, and for promotion of SNE itself (informing about aims, contributions types, broadness of spectrum, etc.) – for more details see the SNE Promotion Issue Editorial at next page. SNE is EUROSIM's scientific journal, and the conferences MATHMOD 2022 and ASIM 2022 are co-sponsored by EUROSIM – the Federation of European Simulation Societies. Also EUROSIM and SNE are affected by the pandemics. Almost all member societies had to cancel their conferences, or to switch from personal mode to virtual mode. In the first year of the pandemic, participants accepted the virtual conferences as a novelty, as charming, as alternative, and almost as splendid. In the second year, the virtual conferences lost their splendidness, and the number of participants was decreasing, sometimes dramatically, and the simulation community is longing for conferences with personal attendance. For SNE, an almost contrary development was arising: we receive more post-conference publications for conferences of the member societies, so that SNE Volume 31, 2021 published 20% contributions more, and the forecast for 2022 has the same tendency. It seems that publication in SNE has become a support and promotion for the EUROSIM societies. A promotion as with this SNE Promotion Issue, and we are glad to do this support. The EUROSIM societies are also changing, and unfortunately we must inform that the simulation community has lost a prominent proponent: András Jávor, former President of HSS, the Hungarian Simulation Society, passed away in spring 2021 – we remember him in the Societies' News Section.*

Let us hope for a Good Year 2022, and stay safe and healthy, and let us meet again personally at a EUROSIM conference !

Felix Breitenecker, SNE Editor-in-Chief, eic@sne-journal.org; felix.breitenecker@tuwien.ac.at

Contents SNE 31(4)

Promotion Issue *MATHMOD & ASIM 2022*

Online SNE 31(4), DOI 10.11128/sne.31.4.1058

ARGESIM Publisher, Vienna, www.argesim.org

Print SNE 31(4) 978-3-903311-18-3

TU Verlag Vienna, Print-on-Demand, www.tuverlag.at

SNE Contact & Info

SNE Online ISSN 2306-0271, SNE Print ISSN 2305-9974

→ ***www.sne-journal.org***

✉ ***office@sne-journal.org, eic@sne-journal.org***

✉ **SNE Editorial Office**
Johannes Tanzler (Layout, Organisation),
Irmgard Husinsky (Web, Electronic Publishing),
Felix Breitenecker (Organisation, Author Mentoring)
ARGESIM/Math. Modelling & Simulation Group,
Inst. of Analysis and Scientific Computing, TU Wien
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria

SNE Simulation Notes Europe

WEB: → *www.sne-journal.org*, DOI prefix 10.11128/sne

Scope: Developments and trends in modelling and simulation in various areas and in application and theory; comparative studies and benchmarks (documentation of ARGESIM Benchmarks on modelling approaches and simulation implementations); modelling and simulation in and for education, simulation-based e-learning; society information and membership information for EUROSIM members (Federation of European Simulation Societies and Groups).

Editor-in-Chief: Felix Breitenecker, TU Wien, Math. Modelling Group
✉ *Felix.Breitenecker@tuwien.ac.at,* ✉ *eic@sne-journal.org*

Print SNE and Print-on-Demand: TU-Verlag, Wiedner Hauptstrasse 8-10, 1040, Vienna, Austria – *www.tuverlag.at*

Publisher: ARGESIM Arbeitsgemeinschaft Simulation News
c/o Math. Modelling and Simulation Group, TU Wien / 101,
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria;
www.argesim.org, ✉ *info@argesim.org*
on behalf of ASIM *www.asim-gi.org* and
EUROSIM → *www.eurosim.info*

Editorial SNE 31(4) – Promotion Issue *MATHMOD 2022 & ASIM 2022*

SNE 31(4), the **SNE Promotion Issue** *MATHMOD 2022 & ASIM 2022*, is a 'special' SNE Special Issue. It aims for promotion of the conferences **MATHMOD 2022** and **ASIM 2022**, and for promotion of **SNE** itself.

Both conferences are EUROSIM conferences, with special offers for members of **EUROSIM** societies. **MATHMOD** Vienna is a triennial conference series on Mathematical Modelling, usually held in February at TU Vienna. The last **MATHMOD** conference took place February 2018, next **MATHMOD** should take place February 2021, and had because of Corona pandemics to be shifted to February 2022. The organizers, TU Vienna, IFAC, and ASIM, the German simulation society had in mind to organize a conference 'as it was', with personal participation.

ASIM, also upset about the inflation of virtual conferences, took the opportunity to shift the planned 26th ASIM Symposium Simulation Technique just before **MATHMOD 2022**, now called **ASIM 2022** (in February), and to invite simulationists to Vienna for two conferences. And it was planned to have a *SNE Special Issue* for these conferences, with post-conference publications.

In October 2021 the corona situation became worse, and the organizers were faced with a decision: organizing both conferences as virtual conference, or shifting again. In the first year of the pandemic, participants accepted the virtual conferences as a novelty, as charming, as alternative, and almost as splendid. In the second year, the virtual conferences lost their splendidness, and the number of participants was decreasing. An enquiry among **ASIM** members and participants of previous **MATHMOD** participants resulted in a unique answer: no more virtual conferences. As consequence, the organizing teams shifted the conferences to July 2022 with possibility for late submission:

ASIM 2022, July 25-27, 2022, TU Vienna
Submissions until April 14, 2002
www.asim-gi.org/asim2022

MATHMOD 2022, July 27-29, 2022, TU Vienna
Late Submission until 1.2.2022 / 15.3.2022
www.mathmod.at

And we decided to make the post-conference special issue a pre-conference promotion issue, electronically published with changed calls for the conferences, and with **SNE** print versions as part of the conference handouts, and with an exciting mixture of contributions as content (by rescheduling late 2021 contributions and 2022 **SNE** contributions and by help of the authors – many thanks !).

The mixture of contributions in this issue reflect not only the general broadness of modelling and simulation, it demonstrates also the classification of **SNE** contributions as *Technical Note*, *Short Note*, *Software Note*, *Overview Note*, *Benchmark Note*, *Educational Note*, *Project Note*, and *Student Note*, expressing different aims for publications.

SNE 31(4) starts with two contrasting primary contributions. D. Zimmer and co-authors present recent results in Modelica Development in 'Robust Simulation of Stream-Dominated Thermo-Fluid Systems: from Directed to Non-Directed Flows' – a classic SNE *Technical Note*, and also a *Software Note*. As contrast, I. Hafner and N. Popper contribute with the *Overview Note* 'An Overview of the State of the Art in Co-Simulation and Related Methods', also a very rich source with six pages references on the subject. Co-simulation is also a topic in the contribution 'Systems Engineering as the Basis for Design Collaboration' by E. Russwurm et al., which presents a system for collaboration during development in simulation environments – a *Technical Note* with emphasis on methodology.

Next, 'ROCS: A Realtime Optimization and Control Simulator' by A. Britzelmeier et al., introduce the features of a simulator – a typical *Software Note.* A classic simulation application is 'Simulating and Evaluating Different Boarding Strategies on the Example of the Airbus A320' by *J. Wunderlich,* a *Technical Note.* SNE reports also on ongoing development in *Short Notes*, as with the contribution 'Modeling the Spread of Tree Pests after Aerial Pest Control with the Means of a Geo-Information System' by C. Krugmann and J. Wittmann.

H.-T. Mammen and co-authors present the technical application 'System Simulation as Part of Systems Engineering for Headlamp and Pedal Systems based on the Modeling Language Modelica' – a *Technical Note*; also the next contribution 'Simulation of a Discharge Electrode Needle for Particle Charging in an Electrostatic Precipitator' by S. Beckers et al.

The issue finishes with a special contribution type, a *Benchmark Note*, with educational tendency, so also an *Educational Note*, and as the contribution originates from a student project, also a *Project Note* and a *Student Note* , classified as now as *Educational Benchmark Note*: ARGESIM Benchmark C7 'Constrained Pendulum' – Solution in MATLAB Environment and Extensions with Linear Approach, Symbolic Approach, Sensitivity, and Integration into TU Vienna's MMT E-Learning Environment by P. Setinek. The title refers to the **ARGESIM Benchmarks**, co-organised by **EUROSIM**, a series of benchmarks for *Modelling Approaches and Simulation Implementations.*

We hope that we can interest potential authors by this broad spectrum of publication types, and we hope to meet you personally at **MATHMOD 2022** and/or at **ASIM 2022.**

Sincerely, the **SNE 31(4)** Promotion Issue Editors

A. Körner, F. Breitenecker (members SNE Board and Organizing Team MATHMOD2022 and ASIM2022)

Robust Simulation of Stream-Dominated Thermo-Fluid Systems: From Directed to Non-Directed Flows

Dirk Zimmer*, Niels Weber, Michael Meißner

DLR German Aerospace Center, Institute of System Dynamics and Control, Münchener Strasse 20, 82234 Oberpfaffenhofen, Germany; *dirk.zimmer@dlr.de

SNE 31(4), 2021, 177-184, DOI: 10.11128/sne.31.tn.10581
Received: 2021-11-15 (selected EUROSIM 2019 Postconf. Pub., extended version); Accepted: 2021-11-20
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna, ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. A new robust and efficient formulation for stream-dominated thermal fluid systems has been developed and published as open-source library. This methodology has been predominantly designed for and applied on systems with known flow direction. Since it is not directly evident how it is transferred to systems with unknown flow direction, this paper details the implementation in the corresponding library. As an example, a reversible heat-pump is presented.

Introduction

What are stream dominated systems? Let us suppose we have a component where a fluid is flowing from a set of inlets to a set of outlets. A pipe section with one inlet and one outlet is a simple representation of such a component. When the mass flow rate is high, the fluid within the component will be quickly replaced by the fluid stream of the inlet(s).

In such a case, it is often a reasonable idealization to assume that the thermodynamic state of the outlet Θ_{out} is algebraically coupled to the thermodynamic state of the inlet Θ_{in} (it may also depend on mass flowrate $\dot{m}$ and internal states $\mathbf{x}$ and inputs $\mathbf{u}$). If so, we denote the formulation of equations for such a component as stream-dominated. Correspondingly, a system or subsystem may be denoted as stream-dominated when it is (primarily) composed out of such components.

$$\Theta_{out} = f(\Theta_{in}, \dot{m}, \mathbf{x}, \mathbf{u}) \tag{1}$$

The reason is evident: with an algebraic coupling, any change in the thermodynamic state of the inlet has an immediate effect to the state of the outlet. This is never true for an actual physical system but as an idealization it may be upheld if the stream of the fluid is dominating over the capacity.

Stream dominance is a very useful idealization and hence frequently used. The corresponding algebraic equations enable to describe even complex thermodynamic processes very efficiently using very few time-dependent state variables or even none at all.

Because of this efficiency, various tools (often denoted as 1D tools) support the object-oriented modeling and simulation of thermal fluid systems. Examples are the optimal control of power plants, the simulation of building physics and the environmental control systems (ECS) for cars or aircraft; the very last one being the authors' application domain. Figure 1 shows a picture of a three wheel bootstrap cycle, a classic construction as part of the environmental control system of many civil aircraft. The example has been created using a proprietary library written in the open object-oriented language Modelica [1].

The authors recently published the DLR ThermofluidStream Library [2], an open-source implementation of the stream-dominated approach. This library contains a special sub-package that implements the concepts for non-directed flows described in this paper. The reader is invited to study the code of this implementation as additional content to this paper.

1 On Stream-dominated Systems

Although, stream-dominance may lead to a purely algebraic system that can be efficiently solved in theory, it is often difficult or inefficient to solve in practice. The system of Figure 1 is a good example.

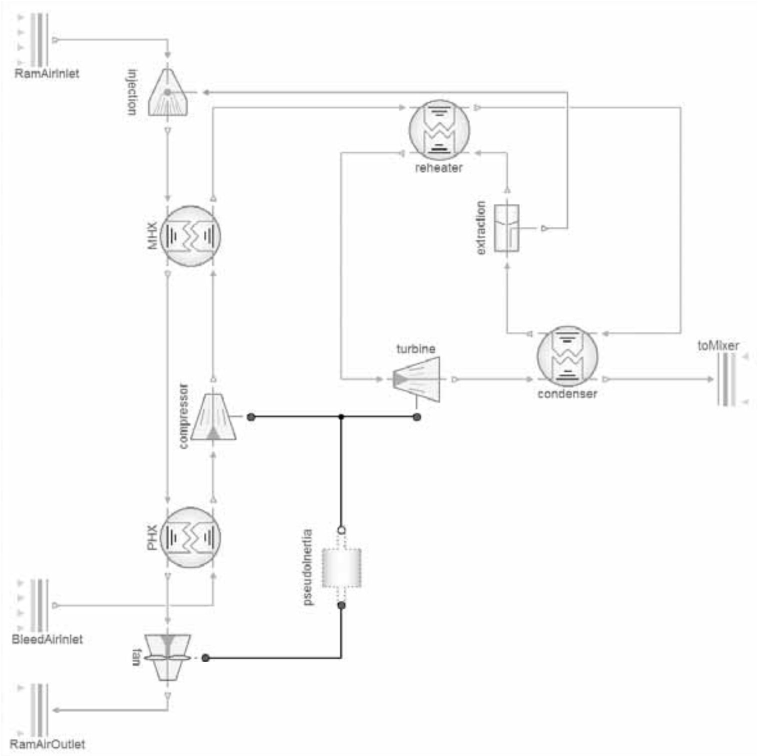


Figure 1: Modelica diagram of a three-wheel-bootstrap cycle. The hot and dense bleed-air is cooled against the outside ram air. The energy of the expansion is used to increase cooling efficiency. Furthermore the air is dehumidified. The system is built extremely compact and the stream of air dominates. The direction of the stream is known a priori.

When modeled purely by algebraic equations using the object-oriented language Modelica, a non-linear system of more than 200 equations results that needs to be solved iteratively by a numerical method [3]. Simulation tools such as Dymola [4] may automatically reduce the dimension to 40 but yet alone finding the area of convergence remains a serious problem.

The high-degree of non-linearity hence poses a serious robustness problem for the object-oriented modeling of stream-dominated systems. Attempts to solve this by more advanced numerical solvers (such as homothopy methods [5]) had been so far of limited success.

Fortunately, a recent advance led to a more robust formulation of stream-dominated systems. The idea is outlined and tested in [6] and further implemented and elaborated in [7] and [8]. Here, we quickly repeat the core idea which centers around the decomposition of the pressure p into the inertial pressure r and the steady-mass flow pressure $\hat{p}$

$$p = \hat{p} + r \tag{2}$$

For a mass-flow $\dot{m}$ that is constant along the stream direction ds, the difference in inertial pressure Δr is purely defined by the geometry of the flow and independent of the thermodynamic state:

$$-\Delta r = \frac{d\dot{m}}{dt} \int \frac{ds}{A_s} \tag{3}$$

where A_s represents the flow cross section at position s. Hence changes in mass flow rate $\dot{m}$ can be determined by a linear system of equations, once the gradients in steady mass flow pressure $\hat{p}$ between individual streams and at the boundaries are known. The steady-mass flow pressure $\hat{p}$ is an unusual term, not present in text-books on the matter. It is simply defined as the complement of r to p. For a steady mass flow (meaning $d\dot{m}/dt = 0$), $\hat{p} = p$ and hence its name. Fortunately, for many applications, it is feasible to express the thermodynamic state using $\hat{p}$

When doing so, the equations for a stream-dominated system can be set up in a very favorable form. The highly non-linear computations of the thermodynamic state can be arranged in an explicit order

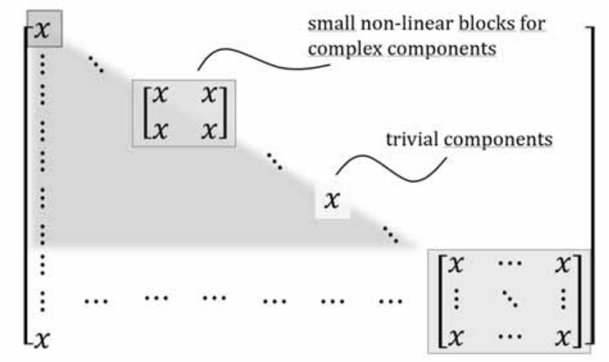


Figure 2: The structure of the resulting equation system. The blue part forms a non-linear LU system that can be computed downstream (barring deliberate small exceptions on component level). It computes all $\hat{p}$ and the thermodynamic state. The green system is linear and computes r and $d\dot{m}/dt$.

going downstream from sources to sinks. The changes in the individual mass flow rates are then computed solving a system of linear equations. This scheme is illustrated by the corresponding BLT-form in Figure 2. Again see [6] and [7] for more details.

When upholding these rules and realizing this scheme in an object-oriented modeling framework such as Modelica, it is reflected in the design of the component interface (denoted as connector in Modelica). The thermodynamic state is represented as a signal going in direction of the stream from source to sink. The inertial pressure r and the mass flow rate $\dot{m}$ form a pair of effort and flow. Here is the corresponding code of an inlet and an outlet in Modelica:

```
connector Inlet
 replaceable package Medium = Modelica.
     Media.Interfaces.PartialMedium;
 SI.Pressure r;
 flow SI.MassFlowRate m_flow;
 input Medium.ThermodynamicState
     state_hat;
end Inlet;

connector Outlet
 replaceable package Medium = Modelica.
     Media.Interfaces.PartialMedium;
 SI.Pressure r;
 flow SI.MassFlowRate m_flow;
 output Medium.ThermodynamicState
     state_hat;
end Outlet;
```

The definition of the thermodynamic state state_hat can differ for different media but commonly consists of pressure $\hat{p}$, specific enthalpy $\hat{h}$ as well as mass fractions for media with more then one component. In any case the state is expressed by quantities for the steady mass flow as described above.

This scheme has been applied with great success for the modeling and simulation of thermofluid systems with directed flow such as modern aircraft ECS. Using this scheme, robustness of the models could be drastically improved [6] since no large non-linear equation system needs to be solved iteratively anymore. With respect to performance, the approach is also interesting. Especially for real-time simulation of such systems, first investigations reveal promising results [7].

In order to be concise, we cannot review the complete built-up of the equation system here (please be referred to [6] and [7]) but for our analysis in Section 2, we need to focus on two items:

- How the linear equations for r and $\dot{m}$ are interlinked with the non-linear computation for $\hat{p}$.
- The structural prerequisite to ensure the LU-form of the non-linear part.

On the first point: the non-linear computation of $\hat{p}$ leads to differences in pressure that are compensated by the inertial pressure r in the following way:

- For each inlet boundary of the stream: $r = 0$ and hence $\hat{p} = p_{\text{inlet}}$ (4)
- For each outlet boundary of the stream: $\hat{p} + r = p_{\text{outlet}}$ (5)
- For each split of a mass flow $\dot{m}_0$ into $\dot{m}_1 \dots \dot{m}_\text{n}$:
  $$\hat{p}_1 = \hat{p}_2 = \ldots = \hat{p}_\text{n} = \hat{p}_0 \quad (6)$$
  and
  $$r_1 = r_2 = \ldots = r_\text{n} = r_0 \quad (7)$$
- For each junction of mass flows $\dot{m}_1 \dots \dot{m}_\text{n}$ into $\dot{m}_0$:
  $$\hat{p}_0 = g_{\text{mix}}(\hat{p}_1, \ldots, \hat{p}_\text{n}) \quad (8)$$
  and
  $$\hat{p}_1 + r_1 = \hat{p}_2 + r_2 = \ldots = \hat{p}_\text{n} + r_\text{n} = \hat{p}_0 + r_0 \quad (9)$$

Where g_{mix} represents the weighted average of the steady mass flow pressures $\hat{p}_1, \ldots, \hat{p}_\text{n}$ where the corresponding volume flow rates V_i form the weigths. To be well-natured, the function g_{mix} shall also be regularized against zero and negative mass (and volume) flow rates. Here is one possible implementation using a small ε for regularization:

$$g_{\mathrm{mix}} = \frac{\sum_{\mathrm{i}} (|V_{\mathrm{i}}| + \varepsilon) \hat{p}_{\mathrm{i}}}{\sum_{i} (|V_{\mathrm{i}}| + \varepsilon)} \qquad (10)$$

To ensure the LU-form of the non-linear part, the directed stream must form an acyclic graph. If there are cycles (such as in a vapor cycle) then the cycle must be torn apart by volume elements. The inlet of a volume element then acts as an outlet boundary of the stream and the outlet of the volume element acts as inlet boundary of the stream. This works because the internal state of a volume prevents a direct algebraic equation between volume inlet and outlet (at the expense of time-dependent states).

2 Non-directed Fluid Streams

So far, this approach has only been applied to 1D fluid streams of known flow direction; hence directed flow. For many aircraft system, such an approach suffices completely.

Some other systems however, like the aircraft bleed(-air) system can also be modeled as 1D system but the flow direction is a priori unknown. Bleed air may flow from the engine to the central hub during normal operation but in the other direction for engine startup. Such systems hence have non-directed flows. This paper presents an extension of the above scheme.

When extending stream-dominated systems from directed to non-directed flows, one inherently is confronted with a fundamental problem: the underlying assumption of stream dominance will inevitably be violated.

Any transient from a strong positive mass flow to a strong negative mass flow passes through zero mass flow. No matter how one quantitatively defines stream dominance, at zero mass flow, this assumption cannot be upheld anymore and the algebraic coupling between "outlets" and "inlets" loses all of its validity. And indeed, one shall not apply a stream-dominated modeling approach if the splish-splash behavior of a fluid at low mass-flow rates is of any interest.

However, for many applications, it is of no interest and the transient just leads from one stream dominated operation point to another stream dominated operation point. Hence, validity at zero-mass flow is not needed. It suffices when the model is robust and well-natured so that the transient does not break or corrupt the simulation. This is the first challenge.

The second challenge is of structural nature. In a system of directed flow, the fluid stream is represented by a signal flow for the thermodynamic state (see the inputs/outputs in the Modelica connector). Since the direction of the fluid flow is given, also the direction of the signal flow is predetermined.

A change in fluid flow direction hence also implies a change in signal flow direction. The algebraic equation systems must hence be structured in such a way that it supports both flow directions. A well proven solution for this is to double the signal flow as it is also applied for the Modelica Standard Fluid library [9, 10, 11].

Following this approach, each two port element (such as a pipe) has a signal flow in both directions. Depending on the actual mass flow rate, one of these signals is chosen as relevant whereas the other signal represents a dummy signal.

Consequently, the interface of a component for non-directed flows has now two signals: one for the thermodynamic state when the flow direction is out of the component and one for the thermodynamic state when the flow direction is into the component. There still remains the pair of effort and flow, formed by the inertial pressure and mass flow rate.

```
connector FluidPort
  replaceable package Medium =  Modelica.
       Media.Interfaces.PartialMedium;
  SI.Pressure r;
  flow SI.MassFlowRate m_flow;
  inpput Medium.ThermodynamicState
      state_hat_in;
  output Medium.ThermodynamicState
      state_hat_out;
end Inlet;
```

2.1 Cycle-free graphs for the signal flow of the thermodynamic state

In graph-theoretical terms, we thus convert a non-directed graph into a directed graph so that each non-directed edge is being replaced by two directed edges in opposite directions. However, we have to do this cleverly since as stated in section 2, the signal flow must be loop free in order to ensure that no non-linear equation system occurs and that the lower-triangular form can be maintained. This means that a loop-free non-directed graph must remain loop-free after conversion.

In order to understand the problem, let us first study what not to do. Figure 3 shows a straight forward, naïve conversion from an undirected graph to a directed graph: a cycle of two edges is being created between junctions A and B.

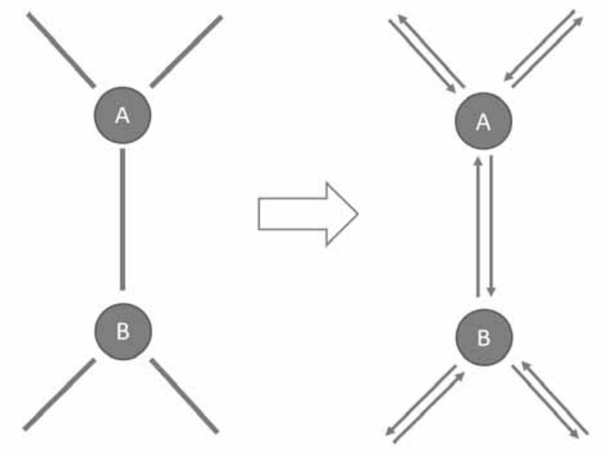


Figure 3: Naïve conversion from an undirected graph to a directed graph for expressing potential signal flows for the thermodynamic state.

The cycle in Figure 3 is however an artefact of conversion and not a loop of the actual system of fluid streams. A flow cannot flow from B to A while it flows from A to B. We have to incorporate this structural knowledge into the conversion and we do so by splitting up the junctions A and B and creating a separate virtual junction for each potential outflow. In this way, we can explicitly state that any outflow must be structurally independent from the flow in its directly opposing direction. Figure 4 depicts how to do this:

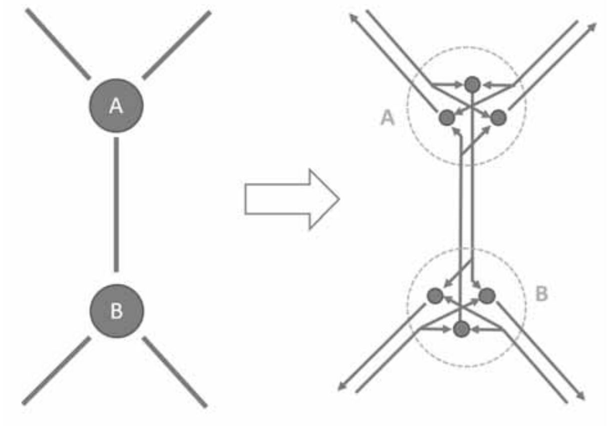


Figure 4: This conversion avoids the creation of cycles by going from a cycle-free undirected graph to a cycle-free directed graph but still expresses all potential information flows for the thermodynamic state. It however contains more nodes and edges.

The computation of the thermodynamic state based on $\hat{p}_{\text{in}}$ (hypothetical) downstream direction hence follows the structure of such a directed graph. Evidently this computation is always performed for both directions although only one direction can actually be relevant. Hence care must be taken that formulas used for the downstream direction are robust against mass flows in opposite direction (they do not need to be valid but should be well-natured).

2.2 Regularization scheme for the inertial pressure at boundaries

The linear equations for the inertial pressure r need to be reformulated as well. As described in Section 1, the boundary equations (4) and (5) for r differed from an inlet to an outlet. For a non-directed system, it is not predetermined what is an outlet and what is an inlet. Hence, the equation has to be unified for a general boundary and made dependent on the flow direction expressed by the sign of the mass flow rate $\dot{m}$. A straight forward implementation would be:

$$r = \text{if } \dot{m} > 0 \text{ then } p_{\text{Boundary}} - \hat{p} \text{ else } 0 \qquad (11)$$

However, such a hard switch is not feasible since the partial derivative $\partial r / \partial \dot{m}$ shall be bounded in order to enable numerical stability of explicit ODE solvers and a sufficient area of convergence for implicit ODE solvers. Hence a regularization scheme needs to be applied that expresses a continuous transition between the two flow directions. ε is used to express the size of this transition region in terms of mass flow rate. We then use the regstep-function

$$y = \text{regstep}(x, y1, y2, \varepsilon) \qquad (12)$$

as depicted in Figure 5 to softly interpolate between y_1 for positive x, and y_2 for negative.

Given this function we can reformulate the boundary equation for r in regularized form:

$$r = \text{regstep}(\dot{m}, p_{\text{Boundary}} - \hat{p}, 0, \varepsilon) \qquad (13)$$

Please note that there is no physical basis for the applied regularization. This means that when the simulation computes in the zone of regularization ($-\varepsilon < \dot{m} < \varepsilon$) the validity of the model may be (at least partially) lost. As stated before, this is a remedy solution for short term transients going through zero mass flow.

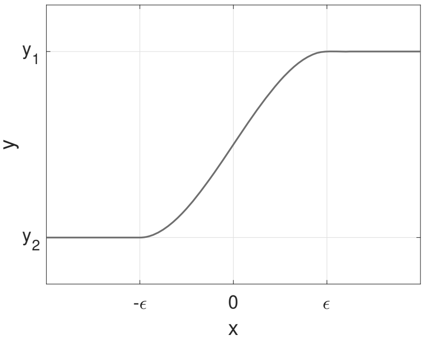


Figure 5: regstep-function used to smoothly interpolate between two values y_1 and y_2, depending on the sign of x.

2.3 Regularization scheme for the inertial pressure at junctions

A corresponding regularization scheme is also needed for the inertial pressures at the junctions. As in Section 1, the main point is to uphold the equivalence of pressure $p = \hat{p} + r$ for all flows at a junction. Because the flow direction is a priori unknown, there is no outflow indicated by the index 0 anymore (as in equation 9) and all flows (whether inflowing or outflowing) are indexed from 1 to n.

$$\hat{p}_1 + r_1 = \hat{p}_2 + r_2 = \ldots = \hat{p}_\mathrm{n} + r_\mathrm{n} \tag{14}$$

Due to the doubling of the signal flow, there is now a pair of steady mass flow pressure $(\hat{p}_{(\mathrm{i,in})}, \hat{p}_{(\mathrm{i,out})})$ for each inertial pressure r_i. The representative steady mass-flow pressure $\hat{p}_\mathrm{i}$ must hence be chosen according to the actual direction of the corresponding mass flow $\dot{m}_\mathrm{i}$. For the same reasons as in Section 2.2, this shall be done in a regularized form:

$$\hat{p}_\mathrm{i} = \mathrm{regstep}(\dot{m}, \hat{p}_{(\mathrm{i,in})}, \hat{p}_{(\mathrm{i,out})}, \varepsilon) \tag{15}$$

With this regularization in place, the linear equations for the inertial pressure can now be formulated without a priori knowledge of the flow direction. Please note that the non-linearity involving the regstep-function is irrelevant because the mass flow rate $\dot{m}$ always forms a state of the system and hence can assumed to be known. It is not part of the equation system, only its time derivative is.

3 Implementation in Modelica and Use Case Application

As we are now familiar with the theoretical background on how to set up the equations for non-directed thermo-fluid systems, we want to see how this can be applied in practice on an example system. Remembering that we are interested in stream-dominated systems in both flow directions, a reversible heat pump forms a suitable example system. In such heat pumps, the flow direction through the heat exchangers can be reversed and the validity at the zero-transition from positive to negative mass-flow is not of main interest. Reversible heat pumps are used for example in the thermal management of modern electric cars or for residential air conditioning. The latter will serve as our use case application.

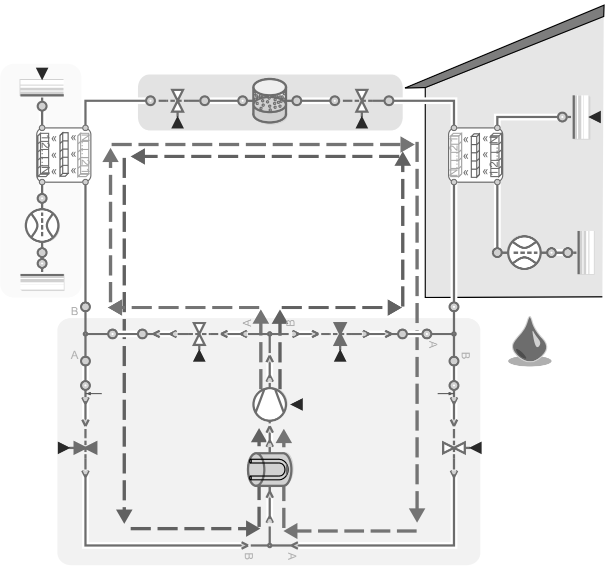


Figure 6: Example of a reversible heatpump that contains two non-directed heat exchangers.

Figure 6 gives an overview of the system architecture. The system consists solely of components from the freely available DLR ThermoFluidStream Library, which was recently published by the authors [2]. The main components of the system are known from a standard vapor cycle, as it can be found in every refrigerator. It consists of a compressor, condenser, expansion device and an evaporator. When the flow direction of the refrigerant is changed, the heat exchangers can act as evaporator or condenser according to the current flow direction. The system can be operated in two different cycle modes - heating (red arrows) or cooling (blue ar-

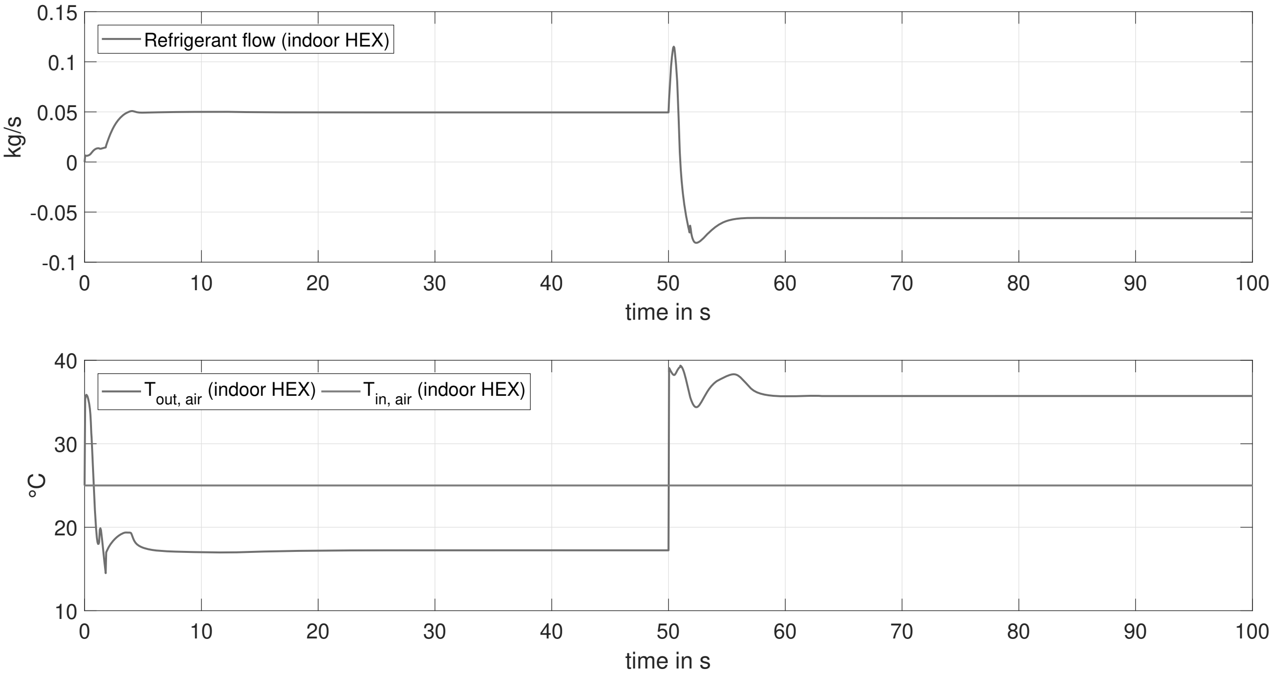


Figure 7: Simulation result showing the transient of flow reversal

rows). In cooling mode, the indoor unit acts as an evaporator and the outdoor unit acts as a condenser. Thus the heat is absorbed from the inside air and rejected to the outside. In heating mode, the cycle is reversed which makes the indoor unit the condenser and the outdoor unit the evaporator. Hence the heat is absorbed from the outside and rejected to the inside.

It is recommended to use components for non-directed flows only when the flow direction is really not known a priori. This is only the case for the heat exchangers, the phase-separator and the expansion devices (magenta). The two expansion valves are required to control the superheating temperature after the evaporator in both operating modes. In practice, the change of flow direction is carried out by a so-called reversing valve. In our simulation model, the flow direction is controlled by a system of valves and non-directed junctions (yellow). The flow direction through the compressor and the accumulator does not change during switching, which is also the case in real systems hence those components can be kept directed.

To conclude this section, let us have a look at an exemplary simulation result. For the sake of simplicity and to spare the model of an additional controller, we set the compressor speed to a fixed value. With the expansion valves, the superheating temperature is controlled to $5K$ and the temperature at the air side of the indoor unit is set to $25\,^{\circ}C$.

Looking at the results in Figure 7 we can observe that we are able to switch from cooling to heating mode at $t = 50s$ during simulation quite drastically. The plot shows a sudden change of the temperature at one end of the heat exchanger. This highlights the stream dominated modeling approach where the replacement of the fluid is neglected on purpose and the model reacts hence quicker than a real device would do.

From a computational point of view though, the transition from positive to negative mass flow through the indoor unit does not cause any problems. Nevertheless the system needs some time to reach the steady state during startup and after switching. This can mainly be referred to the controller to reach the setpoint for the superheating temperature and the time constants for temperature adaption in the heat exchangers. After all, this example shows that the underlying methodology can robustly be applied for system architectures with undirected flow components.

4 Conclusions

In previous papers, a new scheme for the robust computation of directed thermofluid systems has been presented. It proved to be extremely useful for our modelling and simulation activities for aircraft systems. Development time for system models could be drastically reduced and hard real-time simulation of complete systems became feasible. Yet, it was unclear whether this computational scheme (and the way to set up the equations) can be conveyed to (or even combined with) non-directed use cases.

This paper demonstrates that this is possible. It is not trivial and requires to take into account the structure of information flow (for the thermodynamic state) at the junctions and to apply a regularization scheme for low mass-flows and situations of flow reversal.

Once implemented in the equations of Modelica components, the proposed solution becomes easy to apply for the end-user. The only major concern for the end-user is that cycles of fluid streams shall be torn apart by volume elements. The open-source library [2] provides a corresponding implementation in Modelica and also includes the presented example.

Independent from the concrete implementation, two words of warning seem appropriate.

The first warning is a reminder that although non-directed flows are supported, stream dominance is still required for validity. Flow reversal shall occur only briefly during transient and should not be of main interest. This warning however also applies to similar popular modeling approaches as the Modelica Standard Fluid library [9, 10] that also implemented similar regularization schemes and relies on similar assumptions although this is unfortunately not prominently mentioned (honi soit qui mal y pense).

The second warning addresses a common misconception: one may think that because components for non-directed systems are more general than unidirectional components, it would be smart only to work with such components. However, models for directed systems yield far fewer cycles than their non-directed counterparts for the same reasons random directed graphs have fewer cycles than random undirected graphs. For instance, a bypass is not a cycle as long as the flow direction is known. Hence, we recommend to apply non-directed components only when necessary and to combine them with directed components where appropriate. Knowing the flow direction a priori is a piece of information too valuable to be thrown away.

References

[1] www.modelica.org

[2] Zimmer, D and Meißner, M and Weber, N: The DLR Thermofluid Stream Library *url: github.com/DLR-SR/ThermofluidStream* (2021).

[3] Zimmer D, Using Artificial States in Modeling Dynamic Systems: Turning Malpractice into Good Practice. In: *Proceedings of the 5th International Workshop on Equation-Based Object-Oriented Languages and Tools (EOOLT)* , Nottingham, United Kingdom (2013).

[4] Brück, Dag, Elmqvist, Hilding, Olsson, Hans: Dymola for Multi-engineering Modeling and Simulation. In: *Proc. of the 2nd International Modelica Conference*, Oberpfaffenhofen, Germany (2002).

[5] Casella F and Sielemann M und Savoldelli L (2011), Steady-state initialization of object-oriented thermo-fluid models for homotopy methods. In: *Proceedings of 8th International Modelica Conference. 8th International Modelica Conference*, 20.-22. Mar 2011, Dresden.

[6] Zimmer D. (2020), Robust Object-Oriented Formulation of Directed Thermofluid Stream Networks. In: *Mathematical and Computer Modelling of Dynamic Systems*, Vol 26, Issue 3.

[7] Otter M, et al.: Thermodynamic Property and Fluid Modeling with Modern Programming Language Constructs. In: *Proceedings of 8th International Modelica Conference. 13th International Modelica Conference*, Mar 04-06, 2019, Regensburg, Germany (2019).

[8] Zimmer D: Towards Hard Real-Time Simulation of Complex Fluid Networks. In: *Proceedings of the 13th International Modelica Conference* March 4-6, 2019, Regensburg, Germany (2019).

[9] Casella F et al.: The Modelica Fluid and Media library for modeling of incompressible and compressible thermo-fluid pipe networks, *Proceedings of the 13th International Modelica Conference* March 4-6, 2019

[10] Franke, Rüdiger, et. al.: Standardization of Thermo-Fluid Modeling in Modelica.Fluid. *Proceedings 7th Modelica Conference*, Como, Italy, (2009).

[11] Franke, Rüdiger, et. al.: Stream Connectors – An Extension of Modelica for Device-Oriented Modeling of Convective Transport Phenomena. *Proceedings 7th Modelica Conference*, Como, Italy (2009).

An Overview of the State of the Art in Co-Simulation and Related Methods

Irene Hafner[1*], Niki Popper[1,2]

[1]dwh GmbH, Neustiftgasse 57-59, 1070 Vienna, Austria; *irene.hafner@dwh.at

[2]TU Wien, Institute of Information Systems Engineering, Favoritenstraße 9-11, 1040 Vienna, Austria

SNE 31(4), 2021, 185-200, DOI: 10.11128/sne.31.on.10582
Received: 2021-09-10; Revised: 2021-11-03;
Accepted: 2021-11-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna,
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. In this paper, we present an overview of existing and ongoing developments in multirate and co-simulation. These are structured into thematical sections, providing general information on the state of the art while additionally allowing an impression of the progress in developments.
The first sections cover research on co-simulation of ODE and DAE systems, including different coupling methods along with comparisons and stability studies. This is followed by a description of standards for co-simulation, specific developments such as frameworks and investigations on hybrid co-simulation or partitioned methods. In conclusion, general strategies for the development and validation of coupling methods and summarized information on methods and challenges are given.

Introduction

One of the major challenges in research on multirate and co-simulation is the delineation of literature belonging to this area: the term “co-simulation” itself has surfaced shortly before the millennium and although it has become more commonly known since, some authors conduct co-simulation as understood in Definition 1, but do not use this precise term[1].

Definition 1 (Co-Simulation [1])**.** Co-simulation is the coupling of two or more simulations which differ in at least one of the following aspects:

- simulation tool
- solver algorithm
- step size.

While raising no claim for all-encompassing consideration, the following overview includes some publications without usage of the term co-simulation that nevertheless provides the basis of scientific research this area. Chronologically arranged details on the mentioned literature allowing a perspective on the “history of co-simulation" may be found in [2].

1 Beginnings in Classical Co-simulation: Coupling of ODEs

The first investigations on multirate and co-simulation have started on ODE systems, motivated by differing stiffness properties and time constants in system parts [3, 4, 5] or aiming at faster computation by parallelization [6]. The approaches vary between solutions with the same, yet adaptive step size [3], an adaptive approach with order control [5] and the introduction of waveform iteration [7]. Propositions regarding consistency depending on the used extrapolation order are found in [4, 5, 8]. What these investigations have in common is that the considered ODE IVP can be divided into two (or more) partial systems as depicted in (1):

$$\dot{\boldsymbol{x}}_1 = \boldsymbol{f}_1(t,\boldsymbol{x}_1,\boldsymbol{x}_2), \quad \boldsymbol{x}_1(t_0) = \boldsymbol{x}_{1,0} \tag{1a}$$

$$\dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(t,\boldsymbol{x}_1,\boldsymbol{x}_2), \quad \boldsymbol{x}_2(t_0) = \boldsymbol{x}_{2,0} \tag{1b}$$

2 Coupling Methods for DAEs

Owned in particular to applications in mechanical systems, research on co-simulation has soon extended to differential-algebraic equation systems. These can be

[1] Throughout this paper, the terminology from [1, 2] is applied. If needed, the reader is invited to look up the definition of specific terms (such as strong coupling, multirate simulation, hybrid simulation, etc.) there.

represented either as systems of ODEs which are coupled by algebraic constraints (see f.i. [9]), or systems of DAEs coupled by output-input dependencies (as in [10]). This means that the algebraic part can be restricted to the coupling equations (given for two subsystems in (2))

$$\begin{aligned} &\dot{\boldsymbol{x}}_1 = \boldsymbol{f}_1(t,\boldsymbol{x}_1,\boldsymbol{u}_1), \quad \boldsymbol{x}_1(t_0) = \boldsymbol{x}_{1,0} \\ &\dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(t,\boldsymbol{x}_2,\boldsymbol{u}_2), \quad \boldsymbol{x}_2(t_0) = \boldsymbol{x}_{2,0} \\ &0 = \boldsymbol{g}(t,\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{u}_1,\boldsymbol{u}_2) \\ &\text{with time varying inputs } \boldsymbol{u}_i, i = 1,2 \end{aligned} \tag{2}$$

or be part of every subsystem, shown for two subsystems in (3).

$$\begin{aligned} &\dot{\boldsymbol{x}}_1 = \boldsymbol{f}_1(t,\boldsymbol{x}_1,\boldsymbol{x}_2), \quad \boldsymbol{x}_1(t_0) = \boldsymbol{x}_{1,0} \\ &\boldsymbol{y}_1 = \boldsymbol{g}_1(t,\boldsymbol{x}_1,\boldsymbol{u}_1) \\ &\boldsymbol{u}_1 = \boldsymbol{L}_1\boldsymbol{y}_2 \\ &\dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(t,\boldsymbol{x}_1,\boldsymbol{x}_2), \quad \boldsymbol{x}_2(t_0) = \boldsymbol{x}_{2,0} \\ &\boldsymbol{y}_2 = \boldsymbol{g}_2(t,\boldsymbol{x}_2,\boldsymbol{u}_2) \\ &\boldsymbol{u}_2 = \boldsymbol{L}_2\boldsymbol{y}_1 \\ &\text{with the elements of } \boldsymbol{L}_i \text{ being equal to} \\ &\text{zero or one.} \end{aligned} \tag{3}$$

Specific coupling methods range from methods to regularize high-index DAEs [9, 11, 12] to automatic algorithms for the calculation of calling sequence [13] and linking of models [14]. The following sections cover certain kinds of coupling methods, therein iterative approaches (Section 2.1), master algorithms with different choices of macro step size (Section 2.2) and methods specialized in the decomposition and coupling of mechanical systems (Section 2.3). This arrangement may not be seen as classification (which may be found in [1]) but simply as means to provide a better overview owned to the multitude of referenced publications. In Section 2.4, works comparing two or more coupling approaches with respect to stability, accuracy or performance are presented. Investigations on stability and error estimates for co-simulation are found in Section 2.5.

2.1 Iterative methods

Iterative coupling methods, *waveform relaxation* (WR) in particular seem to have been introduced by [15] for DAEs, while the first mention including convergence theorems for certain methods applied to ODEs is found in [7]. In general, WR starts with a Gauß-Seidl or Jacobi type master step that is then repeated until desired tolerances are met. Thereby, a contractivity condition has to be fulfilled to guarantee stability and convergence [15, 16].

Throughout the years, dynamic iteration occurs time and again in different variations and improvements: The iterative approach presented by [17] utilizes reduced order models, [16] and [18] introduce preconditioning to counter instabilities while [19] present an iterative algorithm showing similarities to the sliding mode control method (cf. [12]) and the algorithm of [20, 21] uses interface Jacobians for stabilization. [22] extends the application on PDAEs and discusses emerging stability issues. In [23], a possibility for step size control is presented in addition to convergence criteria for coupled DAE systems in general.

2.2 Choice of macro steps

In this section, methods employing a dynamic choice of the macro step are presented. Most of these are adaptive algorithms where the macro step size, at which all subsystem simulators communicate, is chosen according to varying estimates. [24] and [25] realize automatic adaption of macro step sizes via a predictor-corrector method while [26] takes into account eigen frequencies of the overall and/or partial systems instead of local error estimates. [27] include an iterative approach with *increasing* macro size which is reduced again if a maximum of iterations is reached.

[28], on the other hand, present an algorithm without common macro steps, where the subsystems are solved sequentially with their individual step size, determining after every step the slowest und thus next system to be executed. A similar approach without synchronized time steps is applied by [29].

[30] point out challenges in macro step size control such as slow-down by small step sizes and error calculations, accuracy loss in case of large steps or, specific to co-simulation, the unknown influences between macro and micro steps.

2.3 Decomposition and coupling of mechanical systems

Coupling techniques for mechanical systems described as DAEs are assembled in this section. Due to the specific structure of these systems' description, several investigations on their decomposition ([31, 32, 33, 34]) and further, gluing in the form of different force-force (also called "T-T"), force-displacement (T-X), and displacement-displacement (X-X) coupling approaches (cf. [1, 35]) are made. On the one hand, the corresponding literature can be distinguished depending on these gluing strategies: [36] present an X-X strategy, [37] and [38] a T-T method and [39] a T-X method, while the works of [35] compare all three and consider systems coupled by applied forces/torques ([40, 35]) and also systems coupled by reaction forces/torques [41, 42, 43]. In addition, they apply different stabilization techniques: by additional Lagrange multipliers ([40]), consideration of derivatives or integrals of coupling conditions ([44]), or Baumgarte stabilization ([43]). Iterative methods are found in [37, 38, 34]; semi-implicit (i.e. predictor-corrector) approaches are considered f.i. by [41, 25]. Furthermore, [39] apply automatic partitioning and parallel computing.

2.4 Comparisons

In this section, comparisons regarding performance, accuracy, stability or suitability among different co-simulation methods or versus a monolithic approach are presented. [45, 46, 47] present and compare different strong coupling schemes to simulate fluid-structure interaction. Comparisons of loose with strong coupling schemes for the application in building energy systems are performed in [48, 49], concluding that selecting one of these methods comes down to a choice between performance and independent time steps or accuracy. Regarding the possibility of modularity in multiphysics system simulation, [25] comes to the conclusion that classical co-simulation is advantageous compared to coupling of dynamic with static subsystems. [50] even implements a framework with the aim of comparing protocols for data exchange and different coupling methods.

Rather than algorithms themselves, different implementations of co-simulation masters are compared by [51]. [29] compare non-iterative slowest first and fastest first approaches with inter- and extrapolation polynomials of varying degrees with a – maybe for some researchers frustrating yet crucial – conclusion that the choice of the best coupling algorithm has to be exercised individually for every given problem.

2.5 Stability and error estimates

To quantify the worth of coupling methods, these have to be investigated for the numerical effects they have on separately nicely working integration algorithms.

Already in 1984, [5] use error estimates for the truncation error to adapt the macro step size. In general, the order of the global error of the coupled method is bounded by the error of the subsystem solvers and the extrapolation method [8, 52, 53, 54]. In general, consistency is maintained when consistent methods are co-simulated, but maybe of lower order [55].

Further error estimates, based on Richardson extrapolation, can be found in [56] and [57]; an investigation on relative consistency by calculating the defect in [58]. [59] quantify the convergence rate of co-simulation with more than two participating subsystems.

While in the area of partitioned methods, investigations on stability have been published since the 1980s (see Section 5), they gain currency only since the year 2000 for classical loose coupling schemes. As the field of numerics of differential equations and differential algebraic equations itself comes as a vast area of research, the combination of different methods out of this area is even harder to investigate from a general point of view. That generalized stability analysis is difficult to accomplish is also pointed out by [60]: "A detailed stability analysis for modular time-integration methods is technically very complicated since it has to take into account several types of stiff coupling terms and different extrapolation and interpolation methods". Hence, many studies on stability of coupling methods are done on systems with certain limitations, such as constant extrapolation [61]. To obtain higher accuracy, however, a higher extrapolation order may be preferred, which can increase stability issues. These can be met by methods for stabilization such as iteration [10, 16], asynchronous algorithms [28, 29] or weighting algorithms [62]. For the latter, detailed error estimates regarding extrapolations of different order are given in [57]. An interesting outcome is that the reliability of estimates can depend on the kind of DAE coupling (T-T, X-T, or X-X, see Section 2.3).

Some promising stabilization techniques, such as the bilinear delay line by [63] or the introduction of filters by [10], require alteration of the models themselves, which is often not possible with complex problems of the integrate-and-collaborate kind.

The approach found in [64] stands out as they aim to increase stability by energy conservation between co-simulated systems, thereby using power bonds to calculate energy residuals. Stabilization of strongly coupled systems is addressed by [65].

[10] deduce that zero-stability cannot be guaranteed for loose coupling co-simulation in case algebraic loops occur and [61] shows that for sequential algorithms, the order in which the subsystems are executed is crucial for the stability properties of their co-simulation. Even in case of general convergence, the sequence of subsystem execution can influence the *order* of convergence [66].

3 Standards for Co-simulation

The variety of co-simulation methods and tools to be coupled with their origin from different fields of application has led to the desire of unification, which is aimed by the specification of standards. Still, these are constantly revised by the developers and also extended by other researchers to meet specific requirements. The two most popular standards which are also frequently found in the literature, the *High Level Architecture* and the *Functional Mockup Interface*, are presented here along with the DEV&DESS formalism. The latter – whether it may or may not be regarded as standard for co-simulation (cf. Section 2.12 in [2]) – constitutes an important approach that therefore also occurs occasionally, be it directly utilized or adapted, in the literature presented in this chapter.

3.1 High Level Architecture

The High Level Architecture (HLA) has been specified by the US Department of Defense to address the need for reuse and interoperability of simulations within the department. It provides an architecture defining functional elements, interfaces and design rules for simulation applications and a common framework for the definition of specific system architectures [67]. The HLA is software and programming language independent.

Its key functional components are *federates*, the *runtime infrastructure* (RTI) and the *runtime interface*. Federates can range from computer simulations to manned simulators and even interfaces to live players: the representation of a federate is not restricted as long as it allows the interaction with other objects through data exchanges via services from the RTI. The RTI is a distributed operating system offering these services for interaction and federation management. The runtime interface specification defines a standardized manner of interaction between the federates and the RTI independently from the implementation. Monitoring of simulation activities and interfaces to live participants such as control systems are also supported.

Formally, the HLA is defined by the following three components: *object model template*, *interface specification*, and the *HLA rules*. Different timing services by the HLA are described in [68].

It is made clear in [67] that while the HLA provides the minimum essential tools for interoperability, it is itself insufficient to guarantee interoperability.

3.2 Functional Mockup Interface

The Functional Mockup Interface (FMI) is a standard for model exchange and co-simulation initiated by the project MODELISAR and now maintained and developed by the Modelica Association. In a nutshell, the FMI defines the manner in which Functional Mockup Units (FMUs) have to be built so they can be imported by tools serving as master orchestrator and the functionalities and interfaces for the latter. When an FMU *for model exchange* is exported, the tool where the respective model has been implemented translates it into a dynamic system model in C-code with inputs and outputs. The models can contain events as well as differential, algebraic or discrete equations. In the FMI *for co-simulation*, not only the model but also the solution algorithm is included in the exported code.

Master algorithms can then define points in time where participating FMUs exchange data and control this data exchange. In addition to the C-code file, an FMU contains an XML file with the definition of input and output variables and other model information. Further C-functions for the setup of co-simulation minions or execution of model equations and optional data such as icons or documentation are also included in the zip-file (extension ".fmu") which finally constitutes a complete FMU. In the current version of the standard (FMI 2.0, see [69, 70]), the interfaces for model exchange and co-simulation are unified. Additional features such as getting and setting an FMU state (thus potentially enabling rollback) are introduced, but not mandatory for tools that support the FMI. Input and output dependencies of variables and their derivatives (important for algebraic loop detection) or Jacobian information (potentially needed for implicit integration methods or linearization) can also be included in an FMU.

The great potential and renown but also drawbacks and possibilities for improvements are assessed in an empirical survey [71]. Some of the main difficulties are accounted for by the optional features of the FMI, many of which are not supported by most (in particular open source) tools that often do not even properly define which features they support and which they do not. This hampers the implementation of coupling methods requiring specific functionalities such as simulator rollback, information on derivatives, or input-output dependencies. Another problem regarding discrete event or hybrid co-simulation is the requirement of time passing between two synchronization references, which means that simultaneous events cannot be handled by several exchanges of data at the same time step. This has led to extensions to the FMI standard f.i. by [72], who propose an extension by a procedure returning an upper bound for the FMU's acceptable step size, thus allowing adaptive master steps without requiring rollback, or [73], who aims to encode different formalisms such as state machines, discrete event, and synchronous dataflow as FMUs. These extensions naturally are not supported by all tools currently supporting the FMI 2.0 itself[2].

Different, but similar formalizations for FMU execution are presented in [74, 72, 75]. In all these, the authors argue that the validity of a master algorithm depends on the input-output dependencies inside FMUs, an information often not available as it is not required in the FMU description according to the standard alone. [74] present an algorithm based on topological ordering of a graph constructed according to input/output dependencies and further information such as feed-through and reactivity.

3.3 DEVS-based formalisms

The Discrete Event System Specification (DEVS) is a formalism to describe hierarchically structured Discrete Event systems based on systems theory. Similarly, the Differential Equation System Specification (DESS) allows the description of ODE systems. Both have been introduced by [76] and combined for the description of hybrid systems to the DEV&DESS (Discrete Event System & Differential Equation System Specification) formalism. On the deepest level of hierarchy, an *atomic DEVS* can be described as a set of inputs, outputs, states, internal and external transition functions, an output function and a time advance function. Instead of transition functions and the time advance function, an *atomic DESS* contains a *rate of change function* corresponding to the right side of an ODE. In contrast to DESS of Moore type, where the output function has only states in its argument, for Mealy type DESS the output function may depend directly on the inputs as well. In an *atomic DEV&DESS*, both are combined, resulting in discrete and continuous inputs, outputs, states, transition and output functions, a rate of change function and, in addition, a state event condition function. Two or more DEVS (or DESS, DEV&DESS respectively) can be combined into a *coupled DEVS* (or DESS or DEV&DESS), enhancing clarity and supporting modularity. The problem of concurrent events can be tackled by parallel DEVS (P-DEVS), where concurrency is resolved locally in every DEVS. Hybrid P-DEVS (introduced by [77]) are designed to represent discrete and continuous systems as parallel DEV&DESS. Since the DEVS constitutes a formalism, it is software independent. Specific implementations are found in [78, 77, 79, 80].

4 Specific Applications and Developments

This section covers on the one hand specifically implemented frameworks for co-simulation (Section 4.1) and on the other hand developments for a particular model description (hybrid systems in Section 4.2, FEM in Section 4.3) or application (Section 4.4).

4.1 Frameworks

The introduction of frameworks has become more and more popular to allow easy "plug-and-play" co-simulation. However, many frameworks have again been designed motivated by a specific problem or area of application, such as building simulation [81], automotive systems [82], traffic [83], multi-domain physical [84] or cyber-physical systems [85] and are limited to the co-simulation of certain tools, leaving gaps aimed to be filled by further developments. What is more is that these seemingly simple "enablers" of co-simulation bear the risk that systems are not properly checked for stability properties but rashly coupled, which can be amended by notwithstanding mindful consideration and inspection of every user.

[2] A complete list of tools supporting the FMI 2.0 can be found on https://fmi-standard.org/tools/

Many recent, independent developments respect the FMI standard [86, 87, 88, 89]. [68] even utilizes the HLA as well as the FMI. An implementation of a framework extending the FMI to allow hybrid co-simulation is found in [90, 91], see also Section 4.2, where these can be found along further frameworks that are specifically tailored to support hybrid co-simulation.

Other frameworks implement multi-threading with FMUs by deployment on a cluster (as in [86]) or on multiple-core machines [87], by which supra-linear speedup can be achieved.

4.2 Hybrid (co-)simulation

Hybrid systems – in the sense of combined continuous time (CT) and discrete event (DE) systems – have been an ever-present challenge of special interest within modeling and simulation. Only recently, co-simulation has emerged as a possible solution approach that brings along advantages but also approach-specific complications. Although several investigations considered in this section are not focused on *co*-simulation, the peculiarities as well as methods for hybrid simulation frequently apply regardless whether the combination of DE with CT approaches is realized via co-simulation or integrated models.

This pertains for instance to [92], who presents an overview of phenomena in hybrid simulation reported in the literature: event handling, run-time equation processing, discontinuous state changes, event iteration, chattering, and comparing Dirac pulses. These, of course, are equally important issues in hybrid co-simulation. Solutions for event respectively zero crossing detection are addressed by [93] and [94], event ordering by [95], chattering avoidance by [96] and [93], zeno-behavior by [93], and debugging in hybrid simulations by [97].

[98] describes events update schemata and presents a generic methodology for developing hybrid co-simulation tools. For a similar purpose, formalisms have been introduced, f.i. by [99], who proposes the Heterogeneous Flow System Specification (HFSS), or [94], who formalize the FMI, taking input-output dependencies and abstraction of functions into regard to create a non-cyclic graph of the overall system. [79] present a DEVS wrapper for hybrid co-simulation of FMUs implemented in MECSYCO using the DEV&DESS standard.

The work of [75] shall be emphasized at this point, as they define a range of requirements for hybrid co-simulation standards along with test components and acceptance criteria.

In many specific approaches, one part is controlled respectively set back by the other: methods with the DE simulation as master are found in [100], CT simulation is taken as master by [101], and [98] employs both of these options. [102] present parallel approaches with potential rollback in both parts. [68] uses an iterative approach and [103] apply step size control. Comparisons of different hybrid simulation approaches can be found in [104, 105, 106] and [107], who compare platforms rather than approaches per se.

Prominent applications are various kinds of controlled systems. These seem predestined as hybrid systems due to their common representation by a continuous time system with a discrete control [108, 79, 101]. Specific applications range from power systems [102, 105], networked control systems [104, 107], voltage distribution control ([109], tanks with controller [101, 79], room temperature control [108] and manufacturing systems [110] to cyberphysical systems in general [91].

Especially developed frameworks are FIDE by [90], SAHISim by [68] (see also Section 4.1), CODIS by [98], an adaption of the Crescenco tool to combine Overture and 20-sim by [100], and a systematic approach for multi-level simulation by [110]. [102] consider the EPOCHS and GECO framework in their review of simulation methods of both communication and power systems. [111] propose a conceptual structuration of co-simulation frameworks consisting of the following five generic layers: conceptual (generic structure), semantic (interaction), syntactic (formalization), dynamic (execution, synchronization), and technical (implementation details, evaluation).

Recent developments in particular are utilizing the FMI in their solution approaches for hybrid systems [109, 68, 108, 75, 90, 79, 95, 103]. As the FMI by itself proves insufficient to satisfy requirements for hybrid co-simulation (see f.i. [112], [108] and cf. [75]), proposals for extensions to the FMI standard are given in [94, 91].

4.3 Coupled simulation of FEM models

Co-simulation of FEM models amongst themselves is covered by ([113, 114]). Others couple FEM with other models, such as multibody ([115]), BEM ([116]) or circuit models ([117]). The developments focus on specific kinds of applications such as fluid-structure interaction ([118, 116]), electro-thermal systems ([119, 117]) or vehicle dynamics ([115]). Most approaches in this area of application are either plainly sequential ([117, 118, 116]) or iterative ones ([119, 115, 113]).

4.4 Application-specific research

Many publications describe very specific applications that do not necessarily offer potential to aid general developments. Nevertheless, examples such as coupling methods for flow simulation [120], the development of a Functional Mockup Unit (FMU) in EnergyPlus for loose coupling co-simulation of Jacobi type with an EnergyPlus master [121], coupling MATLAB/Simulink with GENSYS using TCP/IP and S-functions for the simulation of rail traction vehicles [122], co-simulation in real-time hardware applications [123], or a survey on the state of the art in process-machine interactions focusing on metal-working processes [124] convey an idea of the vast field of areas profiting from the concept of co-simulation.

5 Partitioned Multirate Schemes

The introduction of partitioned multirate schemes is motivated by dividing stiff systems of ordinary differential equations into an active and latent part depending on the time constants of the respective subsystems: The active parts need to be integrated with a small step size, the latent parts with a comparatively large step size which is also used as macro time step. Stiffness is thus isolated in the latent parts which can therefore be integrated with an implicit algorithm while the active subsystems can be solved with an explicit solver [125, 126], incorporated in one partitioned solver algorithm. This way, computational effort can be reduced up to 90% [127].

According to [128], applying a multirate scheme is sensible if

- the systems (1) are weakly coupled, meaning $\left\|\frac{\partial f_1}{\partial \boldsymbol{x}_2}\right\| \ll \left\|\frac{\partial f_1}{\partial \boldsymbol{x}_1}\right\|$ and $\left\|\frac{\partial f_2}{\partial \boldsymbol{x}_1}\right\| \ll \left\|\frac{\partial f_2}{\partial \boldsymbol{x}_2}\right\|$
- "the activity levels are widely separated", meaning the micro steps are much smaller than the macro steps
- the activity is concentrated on a small part, meaning there are much less subcircuits in the active system

While the division into a system like (1) is mostly done “by hand” or even assumed to be given initially [127, 129], several approaches include automatic partitioning of the system (depending on step size comparisons, asymptotic behavior, precision of extrapolated values or error estimates), which is sometimes renewed after every macro step [125, 130, 131].

The regarded multirate schemes range from one-step (f.i. [132], who are co-simulating partitioned electrical networks with a w-multirate method or [133], who present an adaptive multirate strategy with a two-stage second-order Rosenbrock method) to multi-step methods [134] and variants [135] including slowest first [136, 125, 129], fastest first [131] and compound methods [126, 129]. Adaptive approaches have been developed by [133] and [137], who control the step size of both micro and macro steps.

Detailed investigations on stability properties of different multirate schemes are found in [138, 134, 129, 139]. [133] conduct error estimates for Rosenbrock methods depending on integration and interpolation orders which are utilized in the adaption of the step size. [140] conduct error analysis for the BDF compound-fast multirate method presented in [129].

While hierarchical structures have up to now been mostly neglected in classical integrate-and-collaborate co-simulation (cf. Section 6.1 from [2]), they have long been introduced for partitioned schemes: [134] already consider three different step sizes. [136] do not restrict the number of levels as long as these show “triangular” dependencies – i.e., equations can be ordered so that System 1 does not depend on values from any other subsystems, System 2 may only depend on values from Systems 1 and 2 and so on. [128] develops an approach suitable for an arbitrary number of activity levels that does not actually restrict dependencies but acknowledges that the partitioning only makes sense for weakly coupled systems, meaning relatively small magnitudes of dependencies (measured by the derivative of the right hand side by the respective state variables, see below).

An apt summary of limitations of multirate methods has been formulated by [29]:

> (...) if the mechatronic system is modelled according to the weakly coupled strategy, these multirate integration methods cannot be applied directly due to their particular features:

1. They introduce modifications in the integration schemes, something that is not possible in commercial off-the-shelf modeling and simulation tools used for weakly coupled co-simulation. For example, the aforementioned block diagram simulators and multibody system simulation packages offer their own set of integration schemes that cannot be modified.

2. They assume that the coarse and refined time-grids are equidistant and synchronized, which means that the large stepsize H is a multiple of the small stepsize h. This condition cannot be guaranteed in weakly coupled co-simulations if one or more subsystems are integrated with a variable time-step integrator, since the stepsize control algorithms of the different commercial simulation environments cannot be synchronized.

3. They mitigate the unstable behavior caused by the explicit extrapolation of some equation terms by introducing implicit schemes, which involve some kind of iterative process. Again, off-the-shelf simulation tools such as block diagram simulators do not allow this kind of iteration with other simulation tools.

6 General Information

In this section, general strategies for coupling methods ([34]), validation and verification of co-simulation ([55]) and results from a survey by [141] on the state of the art in co-simulation, including challenges in discrete event, continuous time and hybrid co-simulation, are summarized.

[34] present guidelines for an effective gluing algorithm, aiming to "execute coupled system simulation without sacrificing the integrity of subsystem modeling and solution and to maintain the efficacy of the overall results." They state that such an algorithm has to be

- *Sticky*: The inter-connection relations between subdomains should be well satisfied, i.e. coupling between subdomains should be resolved and captured.
- *Green*: It should not contaminate subdomain solution strategy. The integrity of the individual model and solution methods should be maintained. Minimum modification of the original solution scheme is desired.
- *Inexpensive*: The overhead should be minimized.
- *Pretty*: The results should be pretty; that is, the overall solution should be numerically correct within the bounds of the desired accuracy. [34]

Chapter 6 of [55] is dedicated to validation and verification of co-simulation. In general, validation is about whether the conceptual model describes the regarded system accurately, verification about the correct implementation and simulation of the conceptual model. [55] has verified her co-simulation by: static verification (structural properties of the code) and dynamical verification (exact synchronization and data transfer tested by varying of time constants).

Validation for coupled simulation is tricky as for different simulation tools ofttimes only different validation approaches exist and comparison with mono-simulation might not be expedient as modeling and simulation of the same system in only one (and hence different for at least one subsystem) simulator could yield different results due to the differences in the simulation tools. [55] apply a method based on inter-model comparison, using only one simulator for mono- and co-simulation.

[141] provide a survey on state-of-the-art techniques for co-simulation, starting by the introduction of a formalization similar to DEVS ([76]). As challenges specific to DE co-simulation, [141] name causality (especially for parallel execution with the possibility of rollback), determinism and confluence (the same results for all possible interleavings of executions), dynamic structure (varying dependencies), and distribution. Fulfillment of algebraic constraints and algebraic loops (closed-loop feed-through in input-output dependencies), which are of special interest for coupled DAE systems, are named as typical challenges in CT co-simulation next to consistent initialization, compositional convergence (error control), compositional stability, compositional continuity (discontinuities in input trajectories due to extrapolation), and real-time constraints. Formalization of hybrid (CT/DE) co-simulation is considered a non-trivial task and thus not given specifically. However, the idea is explained and specific challenges are given, the latter being semantic adaptation (the choice of wrappers depends on the

co-simulation scenario); predictive step sizes (fixed step sizes will miss events, adaptive approaches require detailed information on the subsystems); event location (related to step size prediction, requires information for prediction or rollback functionality); discontinuity identification; discontinuity handling (re-initializing might cause others and not terminate, energy conservation has to be respected); algebraic loops, legitimacy (infinite events at the same time step), and zeno behavior (infinite, consecutive events in ever decreasing intervals but in a bounded time frame, hard to detect in hybrid co-simulation); stability (issues of different origin; further analysis required); theory of DE approximated states (error bounds for the DE part) and establishing a standard for hybrid co-simulation. A taxonomy of a broad selection of literature on co-simulation has revealed the following most observed non-functional requirements: accuracy, protection of intellectual property and performance. Extensibility is among the least observed. Within framework requirements, least observed are dynamic structure co-simulation, interactive visualization, multi-rate, algebraic coupling, and partial/full strong coupling support. In general, they find that there is a lack of research in methods which are both DE and CT based and in leveraging features from simulation units.

7 Conclusion

This paper has given insights on various developments in the area of multirate and co-simulation, therein common methods, standards and frameworks. While there are broad areas of application and research, most investigations and developments are specialized on a certain kind of underlying equation system and may demand restrictions on the manner of coupling. This is not altogether surprising, as special problems come with specialized demands on their solution, which leads us to the most important conclusion to be drawn from this survey: that the choice for the one or the other method cannot be made globally but depends on the underlying system, the status of model development, know-how and interdisciplinarity of the team of developers.

This holds true for selecting special coupling algorithms – see f.i. [43], who show that depending on the system, even higher order extrapolation or higher macro step sizes can yield more stable results – as well as determining whether or not to approach a problem via co-simulation at all: For instance, the disadvantage mentioned in [80] that integration of hybrid aspects on the semantic level is not possible with their chosen co-simulation approach (in comparison to a DEV&DESS-based solution) could for some use cases be seen as advantage, as co-simulation does not require detailed insight and understanding of the partial models' description but *allows* them to be developed independently by experts in the corresponding domain or field. With regard to the additional capabilities or intrusions into subsystem simulators which would be required for rollbacks in co-simulation, this is a minor requirement of insight in comparison to the renewed formalization of every participating model.

In addition, we can observe that, while sensible for the reasons given above, restriction of investigations to systems fulfilling certain requirements holds a few risks: There exist several software tools allowing the more or less easy coupling of certain simulators. Unfortunately, these are often used without further investigation on the consequences regarding numerical stability – such as, for example, testing the system and used algorithms for the requirements necessary to guarantee stability. This, among others, holds true for hierarchical or nested co-simulation, which is allowed by some tools and even, although scarcely, performed, but has only recently been investigated regarding consistency and stability [142, 2].

The restriction to cases with special requirements also leaves a lot of unexploited methods for further investigations. Likewise does the pressing topic of hybrid co-simulation, for which promising developments are in progress in the research groups around the authors of [75, 91, 143]. We conclude with the observation in the words of [29] that "it is not possible to find an optimal general purpose co-simulation method", which leaves co-simulation as ever present topic of interest with plenty of open research questions to be addressed in the future.

References

[1] Hafner I, Popper N. On the Terminology and Structuring of Co-simulation Methods. In: *Proceedings of the 8th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, EOOLT '17. Weßling, Germany: ACM. 2017; pp. 67–76.

[2] Hafner I. Cooperative and Multirate Simulation: Analysis, Classification and New Hierarchical

Approaches. phd thesis, TU Wien, Vienna, Austria. 2021. (submitted for publication).

[3] Hofer E. A Partially Implicit Method for Large Stiff Systems of ODEs with Only Few Equations Introducing Small Time-Constants. *SIAM Journal on Numerical Analysis*. 1976;13(5):645–663.

[4] Andrus J. Numerical Solution of Systems of Ordinary Differential Equations Separated into Subsystems. *SIAM Journal on Numerical Analysis*. 1979; 16(4):605–611.

[5] Gear CW, Wells DR. Multirate linear multistep methods. *BIT Numerical Mathematics*. 1984; 24(4):484–502.

[6] Jackson KR. A survey of Parallel Numerical Methods for Initial Value Problems for Ordinary Differential Equations. *IEEE Transactions on Magnetics*. 1991; 27(5):3792–3797.

[7] White J, Odeh F, Sangiovanni-Vincentelli A, Ruehli A. Waveform Relaxation: Theory and Practice. *Technical Report M85/65*, EECS Department, University of California, Berkeley. 1985.

[8] Knorr S. Multirate-Verfahren in der Co-Simulation gekoppelter dynamischer Systeme mit Anwendung in der Fahrzeugdynamik. Diploma Thesis, Universität Ulm, Ulm, Germany. 2002. Last accessed 1 Oct 2021.

[9] Gu B, Gordon BW, Asada HH. Co-simulation of coupled dynamic subsystems: a differential-algebraic approach using singularly perturbed sliding manifolds. In: *American Control Conference, 2000. Proceedings of the 2000*, vol. 2. 2000; pp. 757–761 vol.2.

[10] Kübler R, Schiehlen W. Two Methods of Simulator Coupling. *Mathematical and Computer Modelling of Dynamical Systems*. 2000;6(2):93–113.

[11] Gu B, Asada HH. Co-Simulation of Algebraically Coupled Dynamic Subsystems Without Disclosure of Proprietary Subsystem Models. *Journal of Dynamic Systems, Measurement, and Control*. 2004; 126(1):1–13.

[12] Gu B. Co-simulation of algebraically coupled dynamic subsystems. Thesis, Massachusetts Institute of Technology. 2001.

[13] Glumac S, Kovacic Z. Calling Sequence Calculation for Sequential Co-simulation Master. In: *Proceedings of the 2018 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS '18. New York, NY, USA: ACM. 2018; pp. 157–160. Event-place: Rome, Italy.

[14] Stecken J, Lenkenhoff K, Kuhlenkötter B. Classification method for an automated linking of models in the co-simulation of production systems. *Procedia CIRP*. 2019;81:104–109.

[15] Lelarasmee E, Ruehli A, Sangiovanni-Vincentelli A. The Waveform Relaxation Method for Time-Domain Analysis of Large Scale Integrated Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 1982;1(3):131–145.

[16] Arnold M, Günther M. Preconditioned Dynamic Iteration for Coupled Differential-Algebraic Systems. *BIT Numerical Mathematics*. 2001;41(1):1–25.

[17] Rathinam M, Petzold L. Dynamic Iteration Using Reduced Order Models: A Method for Simulation of Large Scale Modular Systems. *SIAM Journal on Numerical Analysis*. 2002;40(4):1446–1474.

[18] Ebert F. Convergence of relaxation methods for coupled systems of ODEs and DAEs. 2004;Accessed on 11-08-2021.
URL
`https://opus4.kobv.de/opus4-matheon/frontdoor/index/index/docId/177`

[19] Tomulik P, Fraczek J. Simulation of multibody systems with the use of coupling techniques: a case study. *Multibody System Dynamics*. 2011; 25(2):145–165.

[20] Sicklinger S, Belsky V, Engelmann B, Elmqvist H, Olsson H, Wüchner R, Bletzinger KU. Interface Jacobian-based Co-Simulation. *International Journal for Numerical Methods in Engineering*. 2014; 98(6):418–444.

[21] Sicklinger S, Lerch C, Wüchner R, Bletzinger KU. Fully coupled co-simulation of a wind turbine emergency brake maneuver. *Journal of Wind Engineering and Industrial Aerodynamics*. 2015; 144:134–145.

[22] Schöps S. Multiscale Modeling and Multirate Time-Integration of Field/Circuit Coupled Problems. Ph.D. thesis, Universität Wuppertal, Wuppertal. 2011.

[23] Ebert F. On Partitioned Simulation of Electrical Circuits using Dynamic Iteration Methods. Doctoral thesis, Technische Universität Berlin, Fakultät II - Mathematik und Naturwissenschaften, Berlin. 2008.

[24] Busch M. *Zur effizienten Kopplung von Simulationsprogrammen*. Kassel University Press. 2012.

[25] Schmoll R. Co-Simulation und Solverkopplung. Buch, Kassel Univ. Press. 2015.

[26] Völker L. *Untersuchung des Kommunikationsintervalls bei der gekoppelten Simulation*. No. Bd. 6 in Karlsruher Schriftenreihe

Fahrzeugsystemtechnik. Karlsruhe: KIT Scientific Publ. 2011.

[27] Benedikt M, Stippel H, Watzenig D. An Adaptive Coupling Methodology for Fast Time-Domain Distributed Heterogeneous Co-Simulation. In: *SAE 2010 World Congress & Exhibition*. 2010; .

[28] Liang S, Zhang H, Wang H. Combinative Algorithms for the Multidisciplinary Collaborative Simulation of Complex Mechatronic Products Based on Major Step and Convergent Integration Step. *Chinese Journal of Mechanical Engineering*. 2011;24(03):355.

[29] González F, Naya Mn, Luaces A, González M. On the effect of multirate co-simulation techniques in the efficiency and accuracy of multibody system dynamics. *Multibody System Dynamics*. 2011; 25(4):461–483.

[30] Schierz T, Arnold M. MODELISAR: Innovative numerische Methoden bei der Kopplung von multidisziplinären Simulationsprogrammen. In: *Tagungsband ASIM-Konferenz STS/GMMS 2011*. ZHAW Winterthur, Schweiz: Shaker. 2011; .

[31] Jia Z, Leimkuhler B. A parallel multiple time-scale reversible integrator for dynamics simulation. *Future Generation Computer Systems*. 2003;19(3):415–424.

[32] Featherstone R. A Divide-and-Conquer Articulated-Body Algorithm for Parallel O(log(n)) Calculation of Rigid-Body Dynamics. Part 1: Basic Algorithm. *The International Journal of Robotics Research*. 1999;18(9):867–875.

[33] Featherstone R. A Divide-and-Conquer Articulated-Body Algorithm for Parallel O(log(n)) Calculation of Rigid-Body Dynamics. Part 2: Trees, Loops, and Accuracy. *The International Journal of Robotics Research*. 1999;18(9):876–892.

[34] Tseng FC, Hulbert G. A Gluing Algorithm for Network-Distributed Multibody Dynamics Simulation. *Multibody System Dynamics*. 2001;6(4):377–396.

[35] Schweizer B, Li P, Lu D. Explicit and Implicit Cosimulation Methods: Stability and Convergence Analysis for Different Solver Coupling Approaches. *Journal of Computational and Nonlinear Dynamics*. 2015;10(5):051007.

[36] Tseng FC, Ma ZD, Hulbert GM. Efficient numerical solution of constrained multibody dynamics systems. *Computer Methods in Applied Mechanics and Engineering*. 2003;192(3–4):439–472.

[37] Wang J, Ma ZD, Hulbert GM. A Gluing Algorithm for Distributed Simulation of Multibody Systems. *Nonlinear Dynamics*. 2003;34(1):159–188.

[38] Wang J, Ma ZD, Hulbert GM. A Distributed Mechanical System Simulation Platform Based on a “Gluing Algorithm”. *Journal of Computing and Information Science in Engineering*. 2005;5(1):71.

[39] Rustin C, Verlinden O, Bombled Q. A Cosimulation T-T Procedure Gluing Subsystems in Multibody Dynamics Simulations. ASME. 2009; pp. 83–92.

[40] Schweizer B, Lu D. Stabilized index-2 co-simulation approach for solver coupling with algebraic constraints. *Multibody System Dynamics*. 2014; 34(2):129–161.

[41] Schweizer B, Lu D. Semi-implicit co-simulation approach for solver coupling. *Archive of Applied Mechanics*. 2014;84(12):1739–1769.

[42] Schweizer B, Lu D. Predictor/corrector co-simulation approaches for solver coupling with algebraic constraints. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*. 2015;95(9):911–938.

[43] Schweizer B, Li P, Lu D. Implicit co-simulation methods: Stability and convergence analysis for solver coupling approaches with algebraic constraints. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*. 2016;96(8):986–1012.

[44] Schweizer B, Li P, Lu D, Meyer T. Stabilized implicit co-simulation methods: solver coupling based on constitutive laws. *Archive of Applied Mechanics*. 2015; 85(11):1559–1594.

[45] Matthies HG, Steindorf J. Partitioned but strongly coupled iteration schemes for nonlinear fluid–structure interaction. *Computers & Structures*. 2002; 80(27–30):1991–1999.

[46] Matthies HG, Steindorf J. Strong Coupling Methods. In: *Analysis and Simulation of Multifield Problems*, edited by Wendland PW, Efendiev PM, no. 12 in Lecture Notes in Applied and Computational Mechanics, pp. 13–36. Springer Berlin Heidelberg. 2003;.

[47] Matthies HG, Niekamp R, Steindorf J. Algorithms for strong coupling procedures. *Computer Methods in Applied Mechanics and Engineering*. 2006; 195(17–18):2028–2049.

[48] Trčka M, Wetter M, Hensen J. Comparison of co-simulation approaches for building and HVAC/R system simulation. In: *Proc. of the 10th IBPSA Conference*. Beijing, China. 2007; pp. 1418–1425.

[49] Trčka M, Hensen JL, Wetter M. Co-simulation of innovative integrated HVAC systems in buildings.

Journal of Building Performance Simulation. 2009; 2(3):209–230.

[50] Pühringer C. Analysis of Coupling Strategies and Protocols for Co-Simulation. Diplomarbeit, TU Wien, Wien. 2017.

[51] González F, González M, Mikkola A. Efficient coupling of multibody software with numerical computing environments and block diagram simulators. *Multibody System Dynamics*. 2010; 24(3):237–253.

[52] Arnold M. Multi-Rate Time Integration for Large Scale Multibody System Models. In: *IUTAM Symposium on Multiscale Problems in Multibody System Contacts*, edited by Eberhard P, no. 1 in IUTAM Bookseries, pp. 1–10. Springer Netherlands. 2007;.

[53] Arnold M, Hante S, Köbis MA. Error analysis for co-simulation with force-displacement coupling. *PAMM*. 2014;14(1):43–44.

[54] Schmoll R, Schweizer B. Convergence Study of Explicit Co-Simulation Approaches with Respect to Subsystem Solver Settings. *PAMM*. 2012;12(1):81–82.

[55] Trčka M. Cosimulation for Performance Prediction of Innovative Integrated Mechanical Energy Systems in Buildings. Ph.D. thesis, Technische Universiteit Eindhoven, Eindhoven. 2008.

[56] Zhang H, Liang S, Song S, Wang H. Truncation error calculation based on Richardson extrapolation for variable-step collaborative simulation. *Science China Information Sciences*. 2011;54(6):1238–1250.

[57] Arnold M, Clauss C, Schierz T. Error Analysis and Error Estimates for Co-Simulation in FMI for Model Exchange and Co-Simulation V2.0. *Archive of Mechanical Engineering*. 2013;LX(1).

[58] Glumac S, Kovacic Z. Relative Consistency and Robust Stability Measures for Sequential Co-simulation. 2019; pp. 197–206.

[59] Bartel A, Brunk M, Schöps S. On the convergence rate of dynamic iteration for coupled problems with multiple subsystems. *Journal of Computational and Applied Mathematics*. 2014;262:14–24.

[60] Arnold M, Burgermeister B, Führer C, Hippmann G, Rill G. Numerical methods in vehicle system dynamics: state of the art and current developments. *Vehicle System Dynamics*. 2011;49(7):1159–1207.

[61] Arnold M. Stability of Sequential Modular Time Integration Methods for Coupled Multibody System Models. *Journal of Computational and Nonlinear Dynamics*. 2010;5(3):031003.

[62] Schierz T, Arnold M. Stabilized overlapping modular time integration of coupled differential-algebraic equations. *Applied Numerical Mathematics*. 2012; 62(10):1491–1502.

[63] Larsson J, Krus P. Stability Analysis of Coupled Simulation. vol. Dynamic Systems and Control, Volumes 1 and 2 of *ASME International Mechanical Engineering Congress and Exposition*. 2003; pp. 861–868.

[64] Sadjina S, Pedersen E. Energy Conservation and Coupling Error Reduction in Non-Iterative Co-Simulations. *arXiv:160605168 [cs]*. 2016;ArXiv: 1606.05168.

[65] Viel A. Implementing stabilized co-simulation of strongly coupled systems using the Functional Mock-up Interface 2.0. 2014; pp. 213–223.

[66] Bartel A, Brunk M, Günther M, Schöps S. Dynamic Iteration for Coupled Problems of Electric Circuits and Distributed Devices. *SIAM Journal on Scientific Computing*. 2013;35(2):B315–B335.

[67] Dahmann JS, Fujimoto RM, Weatherly RM. The Department of Defense High Level Architecture. In: *Proceedings of the 29th conference on Winter simulation - WSC '97*. Atlanta, Georgia, United States: ACM Press. 1997; pp. 142–149.

[68] Awais MU. Distributed hybrid co-simulation. Ph.D. thesis, TU Wien, Vienna, Austria. 2015.

[69] Blockwitz T, Otter M, Akesson J, Arnold M, Clauss C, Elmqvist H, Friedrich M, Junghanns A, Mauss J, Neumerkel D, Olsson H, Viel A. Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models. 2012; pp. 173–184.

[70] Modelica Association. Functional Mock-up Interface for Model Exchange and Co-Simulation. `https://svn.modelica.org/fmi/branches/public/specifications/v2.0/FMI_for_ModelExchange_and_CoSimulation_v2.0.pdf`. 2014. Accessed on 2017-10-19.

[71] Schweiger G, Gomes C, Engel G, Hafner I, Schoegg JP, Posch A, Nouidui T. Functional Mock-up Interface: An empirical survey identifies research challenges and current barriers. 2019; pp. 138–146.

[72] Broman D, Brooks C, Greenberg L, Lee EA, Masin M, Tripakis S, Wetter M. Determinate Composition of FMUs for Co-simulation. In: *Proceedings of the Eleventh ACM International Conference on Embedded Software*, EMSOFT '13. Piscataway, NJ, USA: IEEE Press. 2013; pp. 2:1–2:12. Event-place: Montreal, Quebec, Canada.

[73] Tripakis S. Bridging the semantic gap between heterogeneous modeling formalisms and FMI. In: *2015 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS)*. Samos, Greece: IEEE. 2015; pp. 60–69.

[74] Gomes C, Thule C, Lúcio L, Vangheluwe H, Larsen PG. Generation of Co-simulation Algorithms Subject to Simulator Contracts. Oslo, Norway. 2019; p. 15.

[75] Broman D, Greenberg L, Lee EA, Masin M, Tripakis S, Wetter M. Requirements for Hybrid Cosimulation Standards. In: *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, HSCC '15. New York, NY, USA: ACM. 2015; pp. 179–188.

[76] Zeigler BP, Praehofer H, Kim TG. *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*. San Diego: Academic Press, 2nd ed. 2000.

[77] Preyser FJ. An approach to develop a user friendly way of implementing DEV&DESS models in powerDEVS. Thesis. 2015.

[78] Deatcu C, Pawletta T. A Qualitative Comparison of Two Hybrid DEVS Approaches. *SNE Simulation Notes Europe*. 2012;22(1):15–24.

[79] Camus B, Galtier V, Caujolle M. Hybrid Co-simulation of FMUs using DEV DESS in MECSYCO. In: *2016 Symposium on Theory of Modeling and Simulation (TMS-DEVS)*. 2016; pp. 1–8.

[80] Heinzl B, Raich P, Preyser F, Kastner W. Simulation-based Assessment of Energy Efficiency in Industry: Comparison of Hybrid Simulation Approaches. *IFAC-PapersOnLine*. 2018; 51(2):689–694.

[81] Wetter M. Co-simulation of building energy and control systems with the Building Controls Virtual Test Bed. *Journal of Building Performance Simulation*. 2011;4(3):185–203.

[82] Zhang Z, Eyisi E, Koutsoukos X, Porter J, Karsai G, Sztipanovits J. Co-simulation framework for design of time-triggered cyber physical systems. *Simulation Modelling Practice and Theory*. 2014;43:16–33.

[83] Ferreira PAF, Esteves EF, Rossetti RJF, Oliveira EC. A Cooperative Simulation Framework for Traffic and Transportation Engineering. In: *Cooperative Design, Visualization, and Engineering*, Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. 2008; pp. 89–97.

[84] Friedrich M. Parallel Co-Simulation for Mechatronic Systems. Ph.D. thesis, Technische Universität München. 2011.

[85] Karsai G, Sztipanovits J. Model-Integrated Development of Cyber-Physical Systems. In: *Software Technologies for Embedded and Ubiquitous Systems*, edited by Brinkschulte U, Givargis T, Russo S, Lecture Notes in Computer Science. Springer Berlin Heidelberg. 2008; pp. 46–54.

[86] Galtier V, Vialle S, Dad C, Tavella JP, Lam-Yee-Mui JP, Plessis G. FMI-based distributed multi-simulation with DACCOSIM. In: *Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium*, DEVS '15. San Diego, CA, USA: Society for Computer Simulation International. 2015; pp. 39–46.

[87] Ben Khaled A, Duval L, Ben Gaid M, Simon D. Context-based polynomial extrapolation and slackened synchronization for fast multi-core simulation using FMI. Lund, Sweden: inköping University Electronic Press,. 2014; pp. 225–234.

[88] Wang K, Siebers PO, Robinson D. Towards Generalized Co-simulation of Urban Energy Systems. *Procedia Engineering*. 2017;198:366–374.

[89] Thule C, Palmieri M, Gomes C, Lausdahl K, Macedo HD, Battle N, Larsen PG. Towards Reuse of Synchronization Algorithms in Co-simulation Frameworks. In: *Software Engineering and Formal Methods*, edited by Camara J, Steffen M. Cham: Springer International Publishing. 2020; pp. 50–66.

[90] Cremona F, Lohstroh M, Tripakis S, Brooks C, Lee EA. FIDE: An FMI Integrated Development Environment. In: *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16. New York, NY, USA: ACM. 2016; pp. 1759–1766. Event-place: Pisa, Italy.

[91] Cremona F, Lohstroh M, Broman D, Lee EA, Masin M, Tripakis S. Hybrid co-simulation: it's about time. *Software & Systems Modeling*. 2019;18(3):1655–1679.

[92] Mosterman PJ. An Overview of Hybrid Simulation Phenomena and Their Support by Simulation Packages. In: *Hybrid Systems: Computation and Control*, edited by Goos G, Hartmanis J, van Leeuwen J, Vaandrager FW, van Schuppen JH, vol. 1569, pp. 165–177. Berlin, Heidelberg: Springer Berlin Heidelberg. 1999;.

[93] Zhang F, Yeddanapudi M, Mosterman PJ. Zero-Crossing Location and Detection Algorithms For Hybrid System Simulation. *IFAC Proceedings Volumes*. 2008;41(2):7967–7972.

[94] Cremona F, Lohstroh M, Broman D, Natale MD, Lee EA, Tripakis S. Step revision in hybrid Co-simulation with FMI. In: *2016 ACM/IEEE International*

Conference on Formal Methods and Models for System Design (MEMOCODE). 2016; pp. 173–183.

[95] Thule C, Gomes C, Deantoni J, Larsen PG, Brauer J, Vangheluwe H. Towards the Verification of Hybrid Co-simulation Algorithms. In: *Software Technologies: Applications and Foundations*, edited by Mazzara M, Ober I, Salaün G, vol. 11176, pp. 5–20. Cham: Springer International Publishing. 2018;.

[96] Barros FJ. Chattering Avoidance in Hybrid Simulation Models: A Modular Approach Based on the HyFlow Formalism. In: *Proceedings of the Symposium on Theory of Modeling & Simulation*, TMS/DEVS '17. San Diego, CA, USA: Society for Computer Simulation International. 2017; pp. 15:1–15:12.

[97] Van Mierlo S, Gomes C, Vangheluwe H. Explicit Modelling and Synthesis of Debuggers for Hybrid Simulation Languages. In: *Proceedings of the Symposium on Theory of Modeling & Simulation*, TMS/DEVS '17. San Diego, CA, USA: Society for Computer Simulation International. 2017; pp. 4:1–4:12.

[98] Gheorghe L. Continuous/Discrete Co-Simulation Interfaces from Formalization to Implementation. phd, École Polytechnique de Montréal. 2009.

[99] Barros FJ. Semantics of Dynamic Structure Event-based Systems. In: *Proceedings of the Second International Conference on Distributed Event-based Systems*, DEBS '08. New York, NY, USA: ACM. 2008; pp. 245–252.

[100] Fitzgerald J, Larsen PG, Verhoef M, eds. *Collaborative Design for Embedded Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg. 2014. DOI: 10.1007/978-3-642-54118-6.

[101] Tudoret S, Nadjm-Tehrani S, Benveniste A, Strömberg JE. Co-Simulation of Hybrid Systems: Signal-Simulink. In: *Formal Techniques in Real-Time and Fault-Tolerant Systems*, edited by Joseph M. Berlin, Heidelberg: Springer Berlin Heidelberg. 2000; pp. 134–151.

[102] Tong H, Ni M, Yu W, Li Y. Reviews and perspectives of hybrid system simulation for power and communication. In: *The 4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent*. 2014; pp. 302–306.

[103] Farkas R, Bergmann G, Horváth k. Adaptive Step Size Control for Hybrid CT Simulation without Rollback. In: *Proceedings of the 13th International Modelica Conference*, vol. 157 of *Linköping Electronic Conference Proceedings*. Regensburg, Germany: Linköping University Electronic Press. 2019; p. 10.

[104] Quaglia D, Muradore R, Bragantini R, Fiorini P. A SystemC/Matlab co-simulation tool for networked control systems. *Simulation Modelling Practice and Theory*. 2012;23:71–86.

[105] Palensky P, Widl E, Elsheikh A. Simulating Cyber-Physical Energy Systems: Challenges, Tools and Methods. *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)*. 2014; 44:318 – 326.

[106] Heinzl B. Hybrid Modeling of Production Systems: Co-simulation and DEVS-based Approach. Diplomarbeit, TU Wien, Wien. 2016.

[107] Li W, Zhang X, Li H. Co-simulation platforms for co-design of networked control systems: An overview. *Control Engineering Practice*. 2014;23:44–56.

[108] Widl E, Judex F, Eder K, Palensky P. FMI-based co-simulation of hybrid closed-loop control system models. In: *2015 International Conference on Complex Systems Engineering (ICCSE)*. 2015; pp. 1–6.

[109] Savicks V, Butler M, Colley J. Co-simulating event-B and Continuous Models via FMI. In: *Proceedings of the 2014 Summer Simulation Multiconference*, SummerSim '14. Society for Computer Simulation International. 2014; pp. 37:1–37:8. Event-place: Monterey, California.

[110] Thiede S, Schönemann M, Kurle D, Herrmann C. Multi-level simulation in manufacturing companies: The water-energy nexus case. *Journal of Cleaner Production*. 2016;139:1118–1127.

[111] Nguyen V, Besanger Y, Tran Q, Nguyen T. On Conceptual Structuration and Coupling Methods of Co-Simulation Frameworks in Cyber-Physical Energy System Validation. *Energies*. 2017;10(12):1977.

[112] Schweiger G, Gomes C, Engel G, Hafner I, Schoeggl J, Posch A, Nouidui T. An empirical survey on co-simulation: Promising standards, challenges and research needs. *Simulation Modelling Practice and Theory*. 2019;95:148–163.

[113] Ibrahimbegović A, Markovič D. Strong coupling methods in multi-phase and multi-scale modeling of inelastic behavior of heterogeneous structures. *Computer Methods in Applied Mechanics and Engineering*. 2003;192(28–30):3089–3107l.

[114] Esgandari M, Olatunbosun O. Implicit–explicit co-simulation of brake noise. *Finite Elements in Analysis and Design*. 2015;99:16–23.

[115] Mousseau CW, Laursen TA, Lidberg M, Taylor RL. Vehicle dynamics simulations with coupled multibody and finite element models. *Finite Elements in Analysis and Design*. 1999;31(4):295–315.

[116] Felippa CA, Park K, Farhat C. Partitioned analysis of coupled mechanical systems. *Computer Methods in Applied Mechanics and Engineering*. 2001; 190(24-25):3247–3270.

[117] Wünsche S, Clauss C, Schwarz P, Winkler F. Electro-thermal circuit simulation using simulator coupling. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. 1997;5(3):277–282.

[118] Farhat C, Lesoinne M. Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. *Computer Methods in Applied Mechanics and Engineering*. 2000;182(3-4):499–515.

[119] van Petegem W, Geeraerts B, Sansen W, Graindourze B. Electrothermal simulation and design of integrated circuits. *IEEE Journal of Solid-State Circuits*. 1994; 29(2):143–146.

[120] Steinebach G, Rademacher S, Rentrop P, Schulz M. Mechanisms of coupling in river flow simulation systems. *Journal of Computational and Applied Mathematics*. 2004;168(1-2):459–470.

[121] Nouidui T, Wetter M, Zuo W. Functional mock-up unit for co-simulation import in EnergyPlus. *Journal of Building Performance Simulation*. 2014;7(3):192–202.

[122] Spiryagin M, Simson S, Cole C, Persson I. Co-simulation of a mechatronic system using Gensys and Simulink. *Vehicle System Dynamics*. 2012; 50(3):495–507.

[123] Stettinger G, Benedikt M, Horn M, Zehetner J. Modellbasierte Echtzeit-Co-Simulation: Überblick und praktische Anwendungsbeispiele. *e & i Elektrotechnik und Informationstechnik*. 2015; 132(4-5):207–213.

[124] Brecher C, Esser M, Witt S. Interaction of manufacturing process and machine tool. *CIRP Annals*. 2009;58(2):588–607.

[125] Günther M, Rentrop P. Partitioning and Multirate Strategies in Latent Electric Circuits. In: *Mathematical Modelling and Simulation of Electrical Circuits and Semiconductor Devices*, edited by Bank RE, Gajewski H, Bulirsch R, Merten K, ISNM International Series of Numerical Mathematics. Basel: Birkhäuser. 1994; pp. 33–60.

[126] Günther M, Kværnø A, Rentrop P. Multirate Partitioned Runge-Kutta Methods. *BIT Numerical Mathematics*. 2001;41(3):504–514.

[127] Rice JR. Split Runge-Kutta method for simultaneous equations. *Journal of Research of the National Bureau of Standards Section B Mathematics and Mathematical Physics*. 1960;64B(3):151.

[128] Striebel M. Hierarchical Mixed Multirating for Distributed Integration of DAE Network Equations in Chip Design. Ph.D. thesis, Bergische Universität Wuppertal. 2006.

[129] Verhoeven A, Guennouni AE, ter Maten EJW, Mattheij RMM. *A General Compound Multirate Method for Circuit Simulation Problems*, pp. 143–149. Berlin, Heidelberg: Springer Berlin Heidelberg. 2006;.

[130] Engstler C, Lubich C. Multirate extrapolation methods for differential equations with different time scales. *Computing*. 1997;58(2):173–185.

[131] Kvaernø¸ A, Rentrop P. Low Order Multirate Runge-Kutta Methods in Electric Circuit Simulation. 1999.

[132] Bartel A, Günther M. A multirate W-method for electrical networks in state-space formulation. *Journal of Computational and Applied Mathematics*. 2002; 147(2):411 – 425.

[133] Savcenco V, Hundsdorfer W, Verwer JG. A multirate time stepping strategy for stiff ordinary differential equations. *BIT Numerical Mathematics*. 2007; 47(1):137–155.

[134] Skelboe S, Andersen PU. Stability Properties of Backward Euler Multirate Formulas. *SIAM Journal on Scientific and Statistical Computing*. 1989; 10(5):1000–1009.

[135] Biesiadecki JJ, Skeel RD. Dangers of Multiple Time Step Methods. *Journal of Computational Physics*. 1993;109(2):318–328.

[136] Esposito J, Kumar V. Efficient dynamic simulation of robotic systems with hierarchy. In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, vol. 3. Seoul, South Korea: IEEE. 2001; pp. 2818–2823.

[137] Verhoeven A, Tasić B, Beelen TGJ, ter Maten EJW, Mattheij RMM. BDF Compound-Fast Multirate Transient Analysis with Adaptive Stepsize Control. *Journal of Numerical Analysis, Industrial and Applied Mathematics*. 2008;3(3-4):275–297.

[138] Gomm W. Stability analysis of explicit multirate methods. *Mathematics and Computers in Simulation*. 1981;23(1):34–50.

[139] Verhoeven A, Maten EJWT, Mattheij RMM, Tasić B. Stability analysis of the BDF Slowest-first multirate methods. *International Journal of Computer Mathematics*. 2007;84(6):895–923.

[140] Verhoeven A, Beelen TGJ, Guennouni AE, ter Maten EJW, Mattheij RMM, Tasić B. Error analysis of BDF Compound-fast multirate method for

differential-algebraic equations. *CASA-Report*. 2006; 06(10).

[141] Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H. Co-simulation: State of the art. *arXiv:170200686 [cs]*. 2017;ArXiv: 1702.00686.

[142] Hafner I, Popper N. Investigation on Stability Properties of Hierarchical Co-Simulation. In: *Proceedings ASIM SST 2020, 25. Symposium Simulationstechnik*, vol. 59 of *ARGESIM Report*. Online-Tagung: ARGESIM Verlag. 2020; pp. 41–48.

[143] Gomes C, Karalis P, Navarro-López EM, Vangheluwe H. Approximated Stability Analysis of Bi-modal Hybrid Co-simulation Scenarios. In: *Software Engineering and Formal Methods*, edited by Cerone A, Roveri M, vol. 10729, pp. 345–360. Cham: Springer International Publishing. 2018;.

Systems Engineering as the Basis for Design Collaboration

Eva Russwurm[*], Florian Faltus, Joerg Franke

Chair of Factory Automation and Production Systems, Friedrich-Alexander Universität Erlangen-Nürnberg,
Egerlandstraße 7-9, 91058 Erlangen, Germany; **eva.russwurm@faps.fau.de*

SNE 31(4), 2021, 201-208, DOI: 10.11128/sne.31.tn.10583
Received: 2021-09-10 (Selected ASIM WS 2021 Postconf. Pub., English version); Revised: 2021-11-19; Accepted: 2021-11-28
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Systems engineering (SE) approach is undergoing constant change and is already being used in many companies as part of product development. This approach, which offers many advantages in collaboration, efficiency and product quality, as well as cost, can meet the challenges of Industry 4.0. Various tools are needed to implement SE. These include a PLM system for collaboration during development, as well as various simulation environments that are brought together to form a co-simulation. One way to enable communication for this is to use the OPC UA communication standard. The application of the SE approach will be illustrated using two application examples, namely a holistic simulation of a production plant with energy management and a teaching course.

Introduction

The automation of production systems is characterized by high technical complexity and strong interdisciplinarity. The planning of an industrial plant requires the coordination and integration of various specialist disciplines such as mechanical, process and electrical engineering or software with regard to the procedure and work results.

In this context, automation as a connector ensures the correct interaction of various disciplines. The Industry 4.0 approach and the associated increase in IT penetration in the manufacturing industry are significantly increasing the relevance of digitalization in production.

This is already evident during product development, because engineers from different disciplines already work together as part of systems engineering. This requires a data platform that ensures data consistency and enables access, as well as the possibility of exchange through all disciplines.

This article first gives an overview of the methodology of continous engineering and classifies the development methodology according to the V-model. It will then be shown how the individual disciplines can participate synchronously in a simulation and thus how virtual commissioning can succeed.

Finally, the application of integrated engineering will be presented based on two use cases; on the one hand, the methodical approach at university with students as a basis for future collaboration in the company and, on the other hand, the development of a co-simulation for an automated production plant in an industrial company.

1 Continuous Engineering

In modern mechanical engineering, there are hardly any purely mechanical products anymore. The share of electronics and software or machine control in the product development process (PDP) is not only continuously increasing, see Figurure 1, but is also becoming more and more complex.

Whereas in the 1980s an industrial plant consisted mainly of mechanics and mechatronics was more of a supporting accessory, today it is the software that decisively determines the functionalities of a plant. The reasons for this arise from customer requirements such as process flexibilization and networking of all plants in operation to enable simple operation and monitoring of production [1].

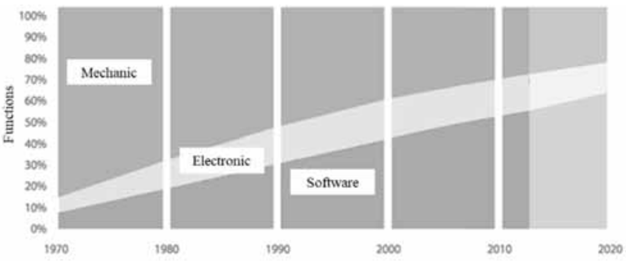


Figure 1: Proportion of individual disciplines in the PEP [1].

Therefore, successful product development already requires successful interaction between various fields of expertise: mechanical engineering, electrical engineering and computer science, especially software engineering. This complexity can no longer be covered by a small number of people, so development teams have to be more interdisciplinary and larger than they were in the past. This inevitably means that the groups also have to work together in different locations in different time zones.

1.1 Definition of Mechatronic Systems

The term "mechatronics" is an artificial word composed of mechanics and electronics and refers to "the synergetic interaction of the disciplines of mechanical engineering, electrical engineering and information technology in the design and manufacture of industrial products, as well as in process design" [2]. The interdisciplinary interaction of a high number of coupled elements, interfaces and interactions of the mechatronic modules with each other makes the application of a holistic, cross-disciplinary approach to system definition indispensable. This also includes communication and cooperation between the individual disciplines [3].

For this reason, new development methods based on the fundamental idea of systems engineering (SE) are increasingly being used for mechatronic systems.

1.2 Strategy of Continous Engineering

The three guiding principles of simultaneous engineering are parallelization, standardization and integration. Parallelization means optimizing the timing of sub-processes that are independent of each other and can be processed independently. In standardization, the primary goal is to avoid duplication and repetition of work. This is achieved primarily through a specified uniform design of modules, components, phases and interfaces between projects and departments. The goal of integration is to turn interfaces into seams. They allow all product information to be brought together.

1.3 Multidisciplinary Approach and Parallelization of Development Activities

This multidisciplinary structure of the model is based on an iterative procedure. Integration progress and synchronization points are continuously checked and guarantee successful product development due to early error prevention.

Furthermore, parallelization instead of sequential product development can save valuable time, which has a positive effect on the required time target (time-to-market). In order to meet this target, certain methods, such as reuse within the SE, must be applied [4].

Consequently, the use of SE to support the upcoming transformation to customized products should be considered as a possible solution approach.

2 Product Development along the V-model

2.1 Requirement and Solution Specification

Another concept, which is mainly used in the requirement engineering, i.e. the development of requirements from abstract specifications of the customer, is the method of RFLP [5]. In German-language literature also named AFLP, it describes the subdivision into requirements, functions, logic and physical model. By these four ranges the goal of the concept, i.e. the draft of a uniform structure, which can be applied to all disciplines, can be pursued.

This interdisciplinary combination results in an initially discipline-independent description of the system and takes on an extraordinary role especially in the early development phases of multidisciplinary products.

The physical model describes the elaboration of the system architecture by adding physical properties using discipline-specific methods and IT applications [6].

2.2 Mechanical, Electrical and Software Development

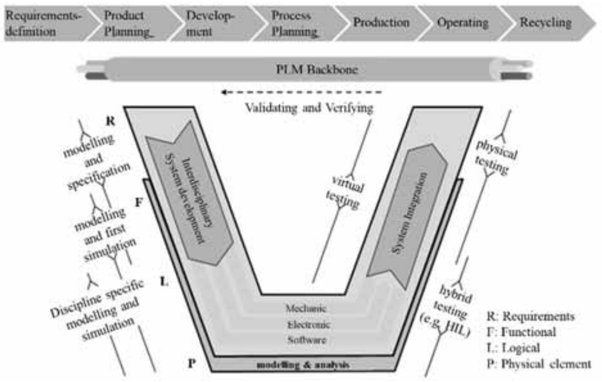


Figure 2: MVPE process model based on VDI 2206 [3].

In systems engineering process, mechanical design follows the solution specification. Simultaneously with it, the electrical design and the software development are proceeded, as can be seen in Figure 2. After completion of the design phase, to which it belongs, realization follows, as well as virtual and actual commissioning [3].

In order to enable a continuous flow of information and a coherent product model across disciplines, the respective discipline naturally has interfaces with other disciplines (mechanical, electrical and software). For this purpose, specifically new approaches of product development have to be defined [7]. The interfaces are predominantly mapped by the PLM system.

2.3 Further Development to Model-based Systems Engineering

One possible approach is Model-Based Systems Engineering (MBSE), which describes the transition from document-based to model-based systems engineering and combines MVPE with systems engineering. It is a "formalized application of modeling to support the incorporation of system requirements, design, analysis, verification, and validation from the concept phase through the development phase to later life cycle phases" [8]. In this case, the draft engineering is of particular importance, since it is during this phase that the system model is created, which contains all the product lifecycle requirements of the product.

Completely digital product models (plant models) are already created during the product development process, enabling physical products to be linked to the associated virtual models from product development. This enables the networking of real production facilities with the digital images created during product planning, as required by Industry 4.0 [10], which helps to merge the virtual and real world.

Within the framework of MBSE, a holistic and consistent data model is created for each product over the entire product lifecycle, which enables enormous productivity, efficiency and quality increases as real and digital processes overlap. For example, in addition to the product, manufacturing is also planned digitally and mapped and validated through early simulations, resulting in up to 50% reduction in time-to-market for new products [11]. The end-to-end data model enables flexibility in the production process through consistency, which allows products to be individualized.

2.4 Integration of MBSE into the Stage-Gate Process and the Quality Gate Model

The VDMA Quality Gate model corresponds to a stage-gate process based on Cooper's model [9] . It divides innovation and product development processes into different stages, to which similar activities are grouped. The individual stages end with a quality control (gate), which can only be crossed if the defined requirements are filfilled.

The decisive factor is the assessment of the management. One advantage of this method can be found in the structuring of long processes, which become thereby controllable and steerable. The regular reconciliation of the project team with management also serves to inform all stakeholders and involves interdisciplinary company departments (marketing, sales, etc.) early in the process. Overall, stage-gate processes lead to higher quality and more innovations [9], [12].

The procedure has many parallels to the V-Modell of the VDI standard 2206 and both models comprise almost the same steps. The description of the phase contents, deliverables and gate criteria turn the general V-Modell into a controllable process. However, the work here is not yet model-based, because the greatest challenge is the realization of consistency in the PDP by means of consistent system models.

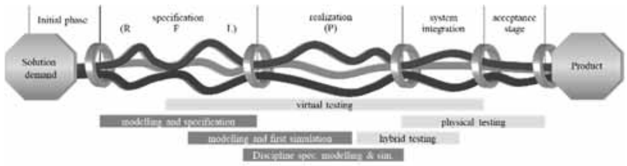


Figure 1: Model-based quality gate approach.

Figure 3 illustrates the modified quality gate approach. As in the VDMA quality gate model, the turquoise, yellow and red bands represent the classic disciplines of mechanics, electronics and software, which continue to be processed as parallel strands [13], [14].

During further development, the requirements and solution specification phases are combined into a single specification phase so that a holistic system model consisting of system requirements, functions and logic (R, F, L) can be created and released. The creation of test specifications also forms part of this phase. The performance of quantitative simulations also supports the objective selection of solution alternatives.

The subsequent realization phase includes discipline-specific modeling and simulation, which is combined in the integration phase at the latest and supplemented by physical tests. Final tests are performed in the acceptance phase.

During the PEP, when working with evolving models, simulation is of great importance as a supporting module. Furthermore, the first simulation studies, which, as shown in Figure **1**ure 3, already start in the specification phase, form the basis for the digital twin of the product or the production plant [15].

3 Co-simulation for Synchronization of the Disciplines

"Simulation is the recreation of a system with its dynamic processes in a model capable of experimentation in order to arrive at knowledge that can be applied to reality" [16, p. 3].

3.1 Production System Simulation

Nowadays, it is necessary to design an automated production system not only in terms of strength in mechanics (e.g. by means of finite element simulation), but to consider the entire mechatronic model and to take into account interactions between modules and models, because it is a complex system. This complexity increases with the number of elements as well as with the links between the elements in terms of tolerance or dynamics.

In order to make the complexity manageable, the system is divided into several - partly hierarchical - levels. In each level, relevant questions can be answered by the simulation.

The first step is to consider, evaluate and, if necessary, improve the issues and solutions in the individual levels. This is performed independently from simulations of other levels.

In context of MBSE, this means that individual components of the plant can be considered separately, but at the same time that all individual disciplines can be considered independently of each other. The fact that the second approach will not result in meaningful solutions is proven at the latest when considering mechatronic models, because mechanical, electrical and information technology (software) tasks must be solved in an integrated manner.

Overall, simulation has proven to be a profitable method to highlight discussion points, but also to support complex decision-making processes [17]. Thus, simulation technology offers great opportunities for improvement in planning and operation of production systems [18].

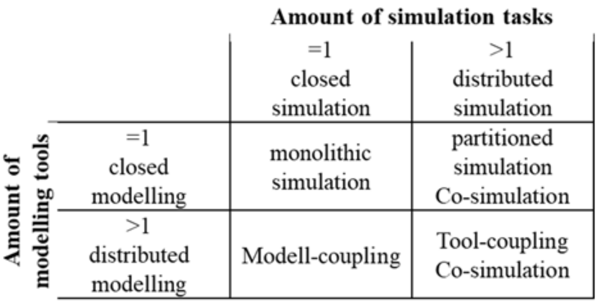
3.2 Multiphysical Simulation Programs

There are already simulation programs from various manufacturers that are predestined for solving mechatronic requirements. Examples include the Mechatronics Concept Designer (MCD) from Siemens PLM, iPhysics from machineering or virtuos from isg. These systems not only offer a simulation environment, but also interfaces for virtual commissioning. [19].

These are tools for mechanical and electromechanical simulation (computer-aided engineering), manufacturing, tool and fixture design, quality inspection, and mechatronic concept development [20]. In this context, geometries modeled in mechanical design can be extended to a simulation model which, in addition to multi-body physics-based simulation, also includes aspects of automation technology and thus visualizes the physical behavior of different solution concepts. Based on the MBSE approach and a cross-domain solution concept, the simulation tools promote early interdisciplinary collaboration between mechanics, electronics and software development. This is reflected in particular by cost savings and accelerated product development times.

3.3 Coupling of Simulation Programs

The aforementioned increasing degree of complexity of production systems makes it necessary to link additional simulation environments. To this end, distributed simulation has gained in importance in recent years. Initially, this involves a simulation model that is divided into different models in the sense of different levels. Data flows exist between the individual submodels via databases, which ensure their consistency. Digital product data in the form of work plans, calculations and CAD models are available in various database systems and can be used and modified by all those involved in the development process. The resulting product or product data models, which function both as interdisciplinary information carriers and as a link between the individual product development areas, such as planning and design, form the basis of distributed simulation.



		Amount of simulation tasks	
		=1 closed simulation	>1 distributed simulation
Amount of modelling tools	=1 closed modelling	monolithic simulation	partitioned simulation Co-simulation
	>1 distributed modelling	Modell-coupling	Tool-coupling Co-simulation

Table 1: Matrix of simulation architectures -[21].

Distributed simulation is also referred to as co-simulation [21], and can consist of a partitioned simulation or a tool coupling, as shown in Table 1.

The aforementioned discipline-specific simulation solutions developed in the context of the digital factory are very powerful in terms of mapping accuracy and calculation performance. However, these are designed for the virtual mapping of selected processes.

For a comprehensive modeling of the entire process, machine and plant behavior with all occurring interactions, high-resolution submodels of various simulation domains must be integrated into the digital image of the production plant and coupled in an overall simulation [22] .

In the context of multidisciplinary modeling of NC machine tools, a central problem in unifying models from different simulation disciplines into an overall multidisciplinary model is said to challenging [23] . This is because, in principle, the relevant data of the individual models must be provided via a neutral uniform data interface by means of a uniform data format. Manufacturers of simulation software meanwhile confirm this thesis by integrating neutral data interfaces [24] .

Furthermore, a simulation tool is needed as a basis that passes on the data of all individual simulation tools. For this purpose, there are different approaches and tests in various contexts, but no system that has become established. In the use cases shown here, an approach for a co-simulation for the holistic simulation of a production plant operated with direct current is to be demonstrated.

4 IT Infrastructure

A functional and far-reaching infrastructure is required for the implementation of MBSE and the associated simulation. PLM systems are a common platform for the sensible implementation of SE in order to support and map model-based work.

For the mapping of co-simulations, additional data exchange formats are required that are as real-time-capable as possible.

4.1 PLM System as the Basis for MBSE

Product Lifecycle Management (PLM) describes a "holistic approach to enterprise-wide management and control of all product data and processes throughout the lifecycle along the extended logistics chain" [25].

As part of the change in development methodology, the engineer's tasks have changed from creative development activity to organizational and communication tasks (planning, procurement). Corresponding IT tools for more efficient management of the tasks are necessary [26] in order to master digitization.

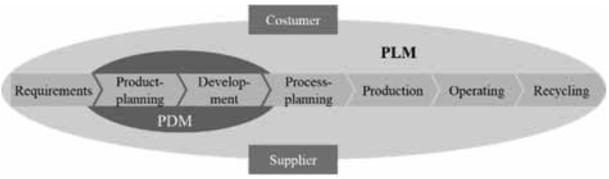


Figure 4: Distinction between PLM and PDM according to [27].

These challenges can only be overcome through the use of product data management (PDM) and PLM software [27]. The former focus on the product development phase, while the PLM approach expands the concept by considering the entire life cycle from initial idea to recycling [28]. Accordingly, PLM represents a concept rather than an IT system.

The PLM software is a component of the concept, serves to integrate IT tools into a development environment and supports the interaction of the tools in terms of models, systems, processes and procedures [3]. Holistic PLM concepts result from strategy, processes and the IT solution [32].

4.2 OPC UA as Communication Standard

In the field of automation, the goal of developing more effectively, more cost-effectively and more time-efficiently and, if possible in the sense of "first-time-right", to commission plants directly at the customer's site ready for production, is at the top of the list.

Just as the need for standardization and modularization in the software landscape is becoming louder, there is an increased demand for a standardized interface between several components at the communication level.

Regarding this, OPC has emerged. OPC stands for "Open Platform Communications" and is a communication protocol that is used primarily in the context of Industrie 4.0 and that enables standardized access to devices, machines and other systems in the industrial environment. It forms the interface between control systems and the control level, enabling uniform data exchange regardless of the manufacturer. The most current specification is "Unified Architecture" or UA for short. OPC UA consists of a server and the client, with the OPC UA server forming the basis. The logical counterpart to the OPC UA server is the client. By connecting to the server, the data provided by the server can be read out [29], [30].

With the help of the described OPC UA standard, the co-simulation for an automated production plant was realized, which is supplied with its own industrial-level DC circuit with a voltage of 650 V DC.

Here, in addition to the multiphysical simulation of the process flow, the simulation of the power supply network plays a significant role.

5 Virtual Commissioning as System Integration

Virtual commissioning is a tool of the digital factory. It is assigned to the "planning of production facilities" phase and thus takes place before the "assembly and commissioning of production facilities" phase [31]. It describes the control commissioning on a virtual machine model, which represents the mechanical, electrical, pneumatic and hydraulic functionalities of an automated, mechatronic plant [33].

In the three types of virtual commissioning (model-in-the-loop (Mil), software-in-the-loop (Sil) and hardware-in-the-loop (Hil)), testing is always performed on the model. The aim is therefore to model the model as realistically as possible. The terms "Model", "Software" and "Hardware" in this context refer to the form of the control that is used, the control within the model, control by a simulated Programmable Logic Controller (PLC) (via the program PLCSim Advanced) or the control as a hardware component.

At any early stage, simulation can be used according to model-in-the-loop. Here, the sequence control is inserted directly into the physical model and tested within the model. This is more of a process-accompanying simulation, since individual steps are successively taken over by the control system.

At this point, however, the physical plant model is already used, which is also used in the other two forms of virtual commissioning. In a further step, the control program can first be tested with a simulated control on the model (Sil). Compared to Hil, this has the advantage that the entire periphery (input and output modules) does not yet have to be defined and modeled. Because, if the real control is used, it is necessary to simulate the whole periphery and the Profibus connection as well as the whole plant by the physical model.

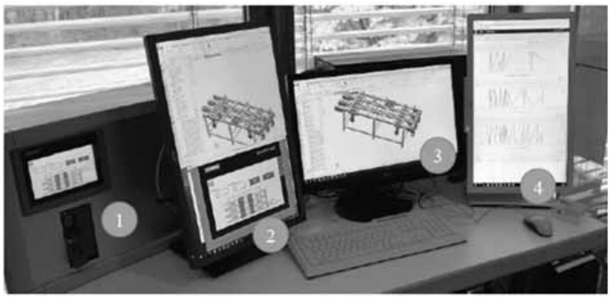


Figure 5: Illustration of Hil using a real controller with a human-machine interface (HMI) and a simulation unit (1) and the SIl with a simulated controller, simulated HMI (2) and the multi-physical model (3) in the MCD, as well as exemplary cloud-based visualization of production data**.**

6 Use Cases

The demonstrated methodology of MBSE and a model environment for a co-simulation were implemented on the basis of two use cases in the field of teaching and research. The goals of the two cases are consistently different, but both are intended to contribute to the fact that product development can be carried out more cost-effectively, better and more efficiently in the future.

6.1 Co-simulation for the Holistic Simulation of an Automated Production Plant

The continuing, or rather successively increasing, scarcity of resources is now a worldwide problem, which is why the energy turnaround has been initiated. In the future, renewable resources such as photovoltaic systems or wind power plants are to be used in the context of manufacturing plants, and storage technologies (accumulators, capacitors) are to be integrated into the plants. Smart grids are being created in the manufacturing halls to meet the energy needs of manufacturing and to make optimal use of the distribution of resources. The energy supply must therefore be regulated via a further control system.

However, the physical behavior of the accumulator or PV system cannot be represented in the common multiphysics simulation environments, nor can the extraction of energy from the utility grid. For this purpose, further simulations have to be used. In this case, MATLAB Simulink was used to simulate the accumulator. For the simulation of the process flow of the manufacturing plant, MCD was used. In the present simulation study, both environments were connected by means ofan OPC UA server. Furthermore, each tool has an OPC UA client. This is used to exchange state variables in discrete time steps and to test the energy management. For example, if an energy-intensive work step begins in the process simulation, the accumulator discharges to the maximum in its own simulation, as shown in Figure 6.

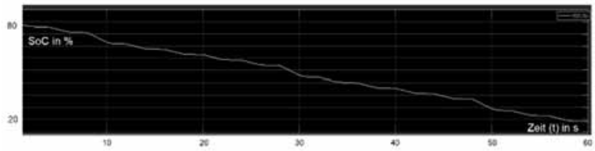


Figure 6: Course of the state of charge over the discharge process. (in MATLAB Simulink).

If this is completely discharged, the high-energy process is stopped and only restarted when the accumulator has been charged by the PV system.

This application primarily demonstrates the feasibility of co-simulation using OPC UA. However, all models are stored in the PLM system, because the structure of the product development process was carried out along the SE.

6.2 Internship for Students as a Basis for Future Cooperation in Companies

Another use case in regarding this topic was implemented in teaching. Every semester, students work in an interdisciplinary team on a development project based on a conveyor system on the chair's own I4.0 demonstrator.

In a fictitious development team, various roles are assigned whose task it is either to create and control the mechanical, electronic or information technology design, or to take care of the simulation accompanying the development, or to monitor the creation of the model already in the specification phase.

The business game should help to understand and comprehend the described basics of SE, because from the recording of the requirements from the customers' declarations to the commissioning of the conveyor belt, the students go through all the steps of the product development process, with all the necessary iterations.

By applying the methodology of SE already in the course of studies, a contribution is to be made to advance the introduction and application in industry.

7 Summary and Outlook

In summary, the approach of model-based development with the aid of PLM systems can meet the requirements placed on manufacturers of mechatronic systems by Industry 4.0. Simulation plays an important role here. Simulation in a multi-physics tool is not sufficient, so co-simulations must be used. One approach to this is to use the OPC UA standard. This approach can be extended so that it can be used to perform virtual commissioning. In order to establish this approach, it will be taught to students as close to practice as possible.

References

[1] Armin Barnitzke. *Machine software must become more modular: Industry survey*. In: Automationspraxis - die anwenderorientierte Fachzeitschrift für Führungskräfte in der Industrie; 2014 December: 12/2014. p. 1-3.

[2] Harashima F, Tomizuka M, Fukuda T. *Mechatronics - "What Is It, Why, and How?" An editorial.* In: IEEE/ASME Transactions on Mechatronics 1996; 1: No. 1. p. 1-4.

[3] 2206. 2004. *guideline VDI 2206.* development methodology for mechatronic systems.

[4] Renault O, Expert P IM. *Reuse/Variability Management and System Engineering.* In: Poster Workshop of the Complex Systems Design & Management Conference CSD&M 2014; 2014.

[5] Kleiner S, Husung S, Schulze S O, Tschirner C, Kaffenberger R. *Model Based Systems Engineering: Principles, Application, Examples, Experience and Benefits from a Practical Perspective.* In: Systems Engineering Day 2016. p. 13-22.

[6] Eigner M, Roubanov D, Zafirov R. *Model-based virtual product development.* Springer; 2014.

[7] Fay Alexander. *Continuous engineering of control systems* (Conference on Continuous Plant Design 2013). Nuremberg, 20.03.2013

[8] Technical Operations International Council on Systems Engineering (INCOSE). *Systems Engineering Vision (2025)*. URL https://www.incose.org/docs/default-source/aboutse/se-vision-2025.pdf?sfvrsn=b69eb4c6_4 - Review date 02/22/2021.

[9] Cooper R G. *Stage-gate systems: A new tool for managing new products.* In: Business Horizons 1990; 33: No. 3. p. 44-54.

[10] Spath, D. *Produktionsarbeit der Zukunft - Industrie 4.0: Studie.* Stuttgart: Fraunhofer-Verl.; 2013.

[11] Russwurm S. *Software: The Future of Industry*. In: Sendler, U (Ed.): Industrie 4.0. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. p. 21-36.

[12] Szinovatz A, Müller C. *Managing complexity in the innovation process From the stage-gate model to the survival-of-the-fittest model.* In: Schoeneberg, K-P (Ed.): complexity management in companies. Wiesbaden: Springer Fachmedien Wiesbaden; 2014. p. 93-112.

[13] Geisberger E, Schmidt R. *Final report of the project "ProMiS" - Project management for interdisciplinary system developments: From the topic area "Software in technical products - application of methods and procedures for engineering-based software development in production"* within the framework of the BMBF research project "Research for the production of tomorrow" ; [Guide for the application of project management and system specification with a practical handbook on CD-ROM] ; [Software VDMA] ; sponsored by the Federal Ministry of Education and Research. Frankfurt am Main, c 2004 (Software vf177400).

[14] Augustin C. *Guide to requirements analysis: software* from the Methods and Procedures series. Frankfurt am Main: VDMA-Verl.

[15] Sauer O, Schleipen M, Ammermann C. *Digital Factory Operation.* In: Zülch, G; Stock, P (Eds.): Integration aspects of simulation: technology, organization and personnel. Karlsruhe: KIT Scientific; 2010. p. 559-566.

[16] *VDI Guideline 3633; Sheet 1.* 1993. VDI Guideline 3633, Sheet 1

[17] Feldmann K. *Simulation-based planning systems for organization and production: model construction, simulation experiments, examples of use.* Berlin [u.a.]: Springer; 2000.

[18] Reinhart G, Feldmann K, Heitmann K. *Simulation - key technology of the future.* In: status and perspectives. Munich: Utz, Wiss 1997.

[19] *Lechler T, Fischer E, Metzner M, Mayr A, Franke J. Virtual Commissioning - Scientific review and exploratory use cases in advanced production systems.* In: Procedia CIRP 2019; 81. p. 1125-1130.

[20] Siemens Industry Software GmbH & Co. KG. : *NX : Transforming the entire product creation process through an integrated software solution for design, simulation and manufacturing*. 2012

[21] Günther F C: *Contribution to co-simulation in the overall system development of the motor vehicle.* Munich, TU Munich. 2017

[22] Scheifele C: *Platform for real-time co-simulation for virtual commissioning.* Stuttgart, University of Stuttgart, Institute for Control Engineering of Machine Tools and Manufacturing Units: Dissertation. 2019

[23] Pritschow G, Berkemer T, Bürger T, Croon N, Korajda B, Röck S. *The simulated machine tool.* In: Heisel, U (ed.): Stuttgart impulses: shaping the future - setting the tone ; FtK 2003, Fertigungstechnisches Kolloquium Stuttgart, October 13-15, 2003 ; conference proceedings ; [written version of the lectures. Stuttgart: Ges. für Fertigungstechnik ; 2003. p. 219-246.

[24] Pritschow G, Röck S. *"Hardware in the Loop" Simulation of Machine Tools.* In: CIRP Annals 2004; 53: no. 1. p. 295-298.

[25] Schuh G. Innovationsmanagement: *Handbuch Produktion und Management 3.* 2., vollst. neu edit. und erw. ed. Berlin, Heidelberg: Springer; 2012.

[26] Eigner M, Stelzer R. *Product Lifecycle Management: A Guide to Product Development and Life Cycle Management.* 2nd, newly edited ed. Dordrecht: Springer; 2013.

[27] Eigner M, Roubanov D, Zafirov R. *Model-based virtual product development.* Berlin: Springer Vieweg; 2014.

[28] Sendler U. *The PLM compendium: reference book of product lifecycle management.* Springer Science & Business Media; 2009.

[29] Lange J, Iwanitz F, Burke T J. *OPC: From Data Access to Unified Architecture*. 5th, revised edition. Berlin, Offenbach: VDE Verlag GmbH; 2014.

[30] Mahnke W, Leitner S-H, Damm M. *OPC Unified Architecture.* 1st ed. Berlin: Springer; 2009.

[31] *VDI 4499.* 2008. VDI 4499 Sheet 1 Digital Factory - Basics

[32] WZL RWTHAachen. *PLM.* URL http://www.plm-info.de - Review date 08.06.2015

[33] Wenk M. *Virtual Commissioning of Production Plants - Effort-Benefits, Implementation Strategies, Future Developments.* In: VDE-Verlag (Ed.) 2008 - Electrical Automation - Systems and Components. p. 533.

ROCS: A Realtime Optimization and Control Simulator

Andreas Britzelmeier*, Matthias Gerdts, Omid Moslehi Rad, Sonali Rani, Thomas Rottmann

Institute of Applied Mathematics and Scientific Computing, Universität der Bundeswehr München, Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany; *andreas.britzelmeier@unibw.de

SNE 31(4), 2021, 209-216, DOI: 10.11128/sne.31.tn.10584
Received: 2021-04-10 (selected ASIM SST 2020 Postconf. Pub.); Revised: 2021-08-17; Accepted: 2021-09-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna, ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. The Realtime Optimization and Control Simulator (ROCS) is a software package written with Qt. It is conceived to be a versatile tool to develop, investigate, and visualize control and trajectory optimization tasks for automated vehicles, aircrafts, and robots in multimodal scenarios. It is also conceived as a platform which allows to combine real driving data with virtual simulation using a vehicle in the loop.

Introduction

The task of simulating and modeling physical systems has always been an important instrument of scientific and industrial research and development. It allows to study and to evaluate the behavior of proposed models and to compare the outcome with the performance of the actual system under consideration. Therein, we can distinguish two major types of simulation tasks, that is, with and without real time requirements and visualization. Often computations are undertaken and afterwards visualized through graphs or non-immersive types of representation. However, with advancements in hardware technology and the increase in digitization in many systems, like cars, planes, and mobile robots, the requirement for virtual environmental simulation is drastically increasing. Another aspect which justifies immersive simulation tools comes along with automation of systems and the desire to create a digital twin. Developers are obliged to prove the reliability and safety of newly developed systems, as well as that the expected increase in utility is guaranteed. However, building prototypes often is very expensive as well as intensive testing to generate reproducible results. Therefore, the need of alternative methods to analyze the behavior and interaction of systems is imminent. Prominent examples of powerful simulation tools in the automotive industry are Virtual Testdrive (VTD) [17, 15], and SILAB, [11]. Both are widely used in the automobile industry, compare [1]. Apart from the automobile industry, also in aerospace engineering and flight training simulative tools have a long history and are widely used. The Flight Simulator from Microsoft, see [12], as well as X-Plane, see [18], both of which are certified by the Federal Aviation Administration, are used in pilot training and research. Hence, immersive simulation tools are not just mere tools of real-time visualization but necessary tools to develop the technology of the future. An indispensable advantage of such tools comes to play whenever new technologies which require human cooperation or interaction is necessary. Then these simulative tools provide a safe environment to test acceptance and reliability, compare [8, 13]. Despite the different types of simulation tools already available, most of them are limited to a specific use case, may it be cars or planes.

In addition, there are very popular game engines, e.g, Unity [16], which are applicable to both, gaming and simulation applications. Such game engines have numerous advantages, e.g. fast and agile development, huge asset stores, optimized graphics, physics and audio engines. However, these platforms come with some specific limitations that can hinder scientific simulation purposes. Few of those complications are licensing and costs for activation of desired features, difficulty in organizing its complex directory hierarchy, non-public source code, making it difficult to track or debug issues, increase in consumption of hardware resources due to complex environment, and finally it is convenient to use only with C Sharp as the primary scripting language. Moreover, downward compatibility issues owing to new versions may arise and can make it difficult to maintain a long term project. Finally, the addition of one's own particular models, controllers, or optimizers can be cumbersome or even impossible.

These potential drawbacks motivated us to build a research and development tool called Real-time Optimization and Control Simulator (ROCS). The idea was to build a versatile research tool which allows for in time visualization and testing of our online optimization algorithms and feedback controllers for automated agents in multi-model scenarios. A further goal was to include control interfaces to real systems. ROCS is build as a modular tool which allows for simple extension by further optimizers and controllers, and the integration of sensor data. Further it is able to visualize scenarios and conduct experimental validation with a variety of vehicles like cars, industrial or mobile robots, and flying platforms like drones, planes or quadrocopters. Owing to the modularity of the tool it is comparatively simple to add models for every required type of vehicle. Different modes of simulation are implemented. One can either provide precomputed data, use feedback controllers in combination with model simulation or employ an optimizer to perform online path planning tasks. ROCS already provides a set of vehicular controllers as well as different models for cars and integrators.

The outline of the paper is as follows. In Section 1 we discuss the overall conceptual design of ROCS. Simulation aspects and two selected vehicle models in ROCS are discussed in Section 2. Section 3 addresses the controller design, while Section 4 presents the 3D simulation environment. Some simulation results are presented in Section 5. Finally, a summary and an outlook with future developments conclude the paper.

1 Design

Realtime Optimization and Control Simulator (ROCS) is designed in a modular way. We decided to implement it in C++ with Qt as it is a programming language widely used in industry and academia and facilitates integration of algorithms and modules. In addition it provides convenient 3D visualization capabilities and the slots and signal mechanism is very well suited for the realtime control purposes.

The main components of ROCS are depicted in Figure 1. The core class objects are a vehicle class, a control class, an input/output class, and a visualization class. The vehicle class contains all vehicle relevant parameters, numerical integrators for motion prediction and simulation, interfaces to controllers and graphical objects describing the shape. The control class contains a collection of tracking controllers and optimization-based path planning tools as detailed in Section 3. The visualizaton class serves to display the simulation and control outputs in a 3D view or in chart plots. It is also possible to store the simulation results or measurements in a file. Likewise it is possible to use ROCS in an offline mode in oder to visualize external data from a data file. The central control unit is the User Control Interface (UCI) described in Section 1.1. An automatic world generator class is part of the concept, but not fully realized up to now.

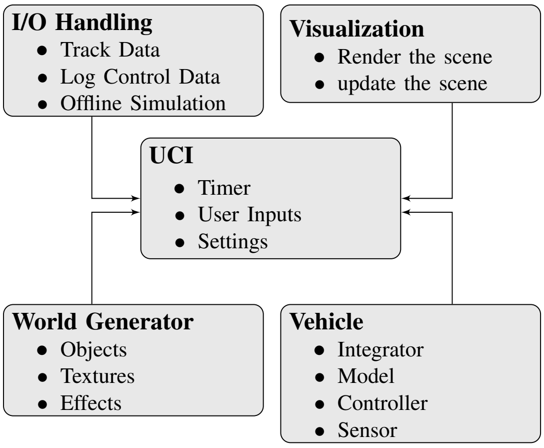
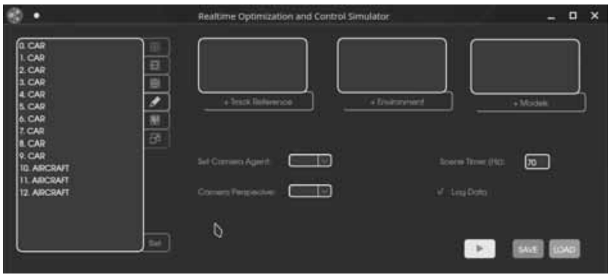


Figure 1: Information flow between simulation models and controllers.

1.1 The user control interface (UCI)

The central control panel of ROCS is the User Control Interface (UCI) in Figure 2. This panel allows to load reference paths, environments, and models. Moreover it provides an overview on the number and type of agents within the simulation. The properties of the agents can be edited through additional dialogs, compare Figures 4, 5. The UCI furthermore allows to select a camera perspective, to switch on or off a data logging mode, and it permits to adjust the scene timer for 3D vizualization. Finally it offers options for saving and loading in order to conveniently store or re-store complex scenarios and settings.


Realtime Optimization and Control Simulator

0. CAR
1. CAR
2. CAR
3. CAR
4. CAR
5. CAR
6. CAR
7. CAR
8. CAR
9. CAR
10. AIRCRAFT
11. AIRCRAFT
12. AIRCRAFT

Set Camera Agent:
Camera Perspective:
Scene Timer (Hz): 70
✓ Log Data

Figure 2: Central user control interface.

1.2 Handling of multiple agents

Due to the object oriented programming style the vehicle class can be sub-classed to differentiate among vehicle types. Furthermore, multiple objects of one vehicle can be created inheriting the same properties and functions, controlling their visualization. On top, each object is stored in a list such that each vehicle appearing in the scene can also be customized. Customization includes changing the objects model, linking to different controllers or integrators or changing the pipeline of data acquisition.

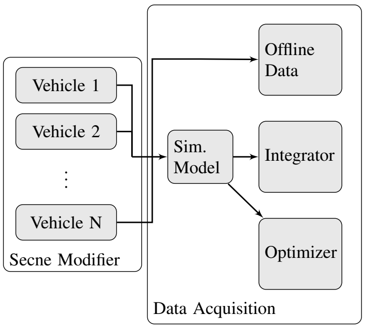


Figure 3: Customizable linking of different data structures for Simulation.

Data can be acquired three ways. The first is to load offline created data from text files which provide data required for simulation. The second method consists in online computation of simulative data through integration and feedback controllers. The computed data is then pipelined by a signal to a slot in the scene modifier class. Thereby, each individual vehicle object and the corresponding controller run in a separate thread and do no interfere with the update of the scene or other operations of the tool. Threads are managed in a synchrnous and threadsafe way. The last option includes the data generation by an optimization-based path planner, which repeatedly solves optimal control problems within a model-predictive control loop. The output data can also be directed to the scene modifier by addressing the same slot from the optimizer class. Hence, we have a uniform connection through signal and slots which can be used to adjoin further modules as well.

In summary, ROCS is centered around the feedback control loops for the agents. These control processes run at a specified frequency in their own threads and are decoupled from the visualization, which is able to run at its own frequency and merely accesses simulation data generated by the control loops. Both frequencies can be synchronized in which case visualization and control work in realtime, if the hardware permits it.

2 Simulation of Multi-Agent Systems

ROCS allows to investigate heterogeneous multi-agent systems consisting of, e.g. cars, robots, or aircrafts. These agents or vehicles, respectively, can be derived from a basic vehicle class, which inherits core functionalities for any type of agent. The derived objects allow to set particular features of the individual agents. The individual agents can be added to the simulation through a dialog window, compare Figure 4.

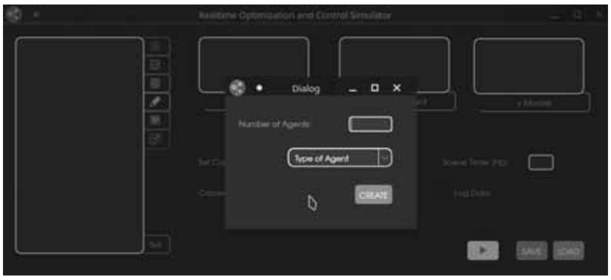


Figure 4: Dialog for adding agents.

The individual properties, models, and parameters of the agents can be adjusted and selected in an editing dialog, compare Figure 5.

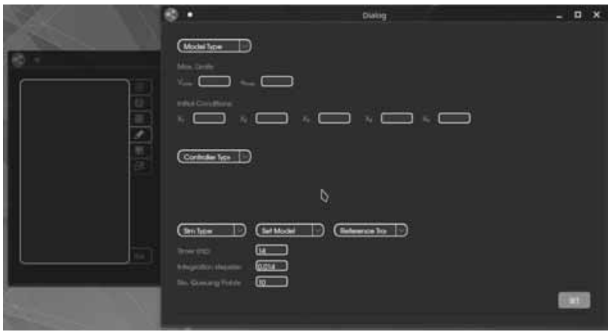


Figure 5: Dialog for editing agents.

Once all agents have been configured (including dynamics, initial states, controller types) and added to the scenario, it remains to simulate the whole multi-agent system. To this end let $N \in \mathbb{N}$ agents be given. We assume that each agent can be controlled and for each agent $i \in \{1,\ldots,N\}$ we denote the control input at time t by $u_i(t)$ and the state at time t by $x_i(t)$. The motion of the i-th agent is modelled mathematically by an initial value problem of type

$$x_i'(t) = f_i(t, x_i(t), u_i(t)), \quad x_i(t_{i,0}) = x_{i,0}, \quad (1)$$

with initial time $t_{i,0}$ for $i = 1,\dots,N$. The agents can be controlled either in open-loop, i.e., by providing the control input $u_i = u_i(t)$ as a given function of time as in (1), or in closed-loop by providing a feedback law $u_i = \mu_i(t,x)$, where $x = (x_1,\dots,x_N)^\top$ is the combined state of all agents. This leads to the closed-loop system

$$x_i'(t) = f(t,x_i(t),\mu_i(t,x(t))), \quad x_i(t_{i,0}) = x_{i,0}, \tag{2}$$

for $i = 1,\dots,N$. In both cases the overall dynamic system will be solved numerically by a Runge-Kutta method. ROCS uses standard solvers with fixed step-sizes (Euler method, Heun's method, classic 4-th order Runge-Kutta method) and variable step-sizes (DOPRI5(4)), see [10].

The outcome of the simulation can be stored in a data logging file or in a chart window, compare Figure 6.

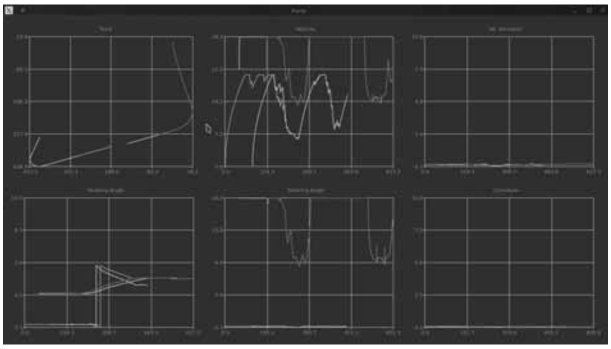


Figure 6: Chart window for detailed view of sensors, states, and controls.

The design of the feedback laws μ_i for the agents $i = 1,\dots,N$, will be outlined in Section 3. Currently, we only have individual controllers implemented, which do not take into account the behavior of the other agents. In the future we will add controllers and optimization strategies for interacting systems as outlined in [4]. This will require to set up an agent-to-agent or agent-to-cloud communication procedure, in which, e.g., position data or driving intentions are exchanged.

2.1 Vehicle models

At the current state of development, due to the focus on autonomous driving, two vehicle models, a single track model and a kinematic model of a two wheel driven mobile robot have been implemented. The equations of motion of the single track model read as,

$$x' = v\cos(\psi - \beta), \tag{3}$$

$$y' = v\sin(\psi - \beta), \tag{4}$$

$$\begin{aligned} v' =& \frac{1}{m}\left[(F_{uh} - F_{Lx})\cos\beta + F_{uv}\cos(\delta + \beta)\right. \\ &\left. - (F_{sh} - F_{Ly})\sin\beta - F_{sv}\sin(\delta + \beta)\right], \end{aligned} \tag{5}$$

$$\begin{aligned} \beta' =& w_z - \frac{1}{mv}\left[(F_{uh} - F_{Lx})\sin\beta + F_{uv}\sin(\delta + \beta)\right. \\ &\left. - (F_{sh} - F_{Ly})\cos\beta - F_{sv}\cos(\delta + \beta)\right], \end{aligned} \tag{6}$$

$$\psi' = w_z, \tag{7}$$

$$w_z' = \frac{1}{I_{zz}}\left[F_{sh}\ell_v\cos\delta - F_{sh}\ell_h - F_{Ly}e_{SP} + F_{uv}\ell_v\sin\delta\right], \tag{8}$$

$$\delta' = \frac{\delta_c - \delta}{T_c}. \tag{9}$$

Herein, x and y are the spacial coordinates and v denotes the velocity. The side slip angle is given by β, the yaw angle is ψ and the steering angle δ. The single track model is already a quite detailed model of a car, which is frequently used in the automotive industry for the investigation of the lateral motion of cars. The model includes various forces acting on the vehicle body. That is, the lateral tyre forces F_{sh}, F_{sv}, longitudinal forces F_{uv}, F_{uh} as well as air resistance in longitudinal F_{Lx} and lateral F_{Ly} direction. Further we have the vehicle mass m and the distance from the centre of gravity to the drag mount point e_{SP}. The distance from the centre of gravity to the front and rear wheel are described by ℓ_v and ℓ_h respectively. The control input to the model are the commanded steering angle δ_c and a combined acceleration and decelleration force, which enters the above force terms. Details of the model can be found in, e.g., [6, 7]. The constant $T_c > 0$ is used to model a delay in the adjustment of the steering angle towards the commanded steering angle.

Another model in ROCS describes a mobile robot with two driven wheels on the left and the right, respectively. Its equations of motion read as follows:

$$x' = \frac{v_L + v_R}{2}\cos\psi, \tag{10}$$

$$y' = \frac{v_L + v_R}{2}\sin\psi, \tag{11}$$

$$\psi' = \frac{v_R - v_L}{B}, \tag{12}$$

$$v_L' = \frac{v_L^c - v_L}{T_c}, \tag{13}$$

$$v_R' = \frac{v_R^c - v_R}{T_c}. \tag{14}$$

Herein, x and y denote the center of gravity of the robot, ψ the yaw angle, v_R and v_L the velocity of the right and left wheels, respectively, and B is the width of the robot. The robot is controlled by the commanded velocities v_R^c and v_L^c of the right and left wheels. The constant $T_c > 0$ is used to model a delay in the adjustment of the velocities towards the commanded velocities.

These two models are included in order to illustrate that heterogeneous agents can be considered. Further vehicle models and models for mobile robots can be found in [5]. We like to point out that further models can be integrated into ROCS in a straightforward way. This is an important feature for our research purposes.

3 Control and Path Planning

The realtime feature is implemented through timers for the control loop of each vehicle, i.e., the control loop runs at a user-defined rate and triggers the import of sensor data and the update of controls. The computed controls are then applied to the vehicle, either for simulation purposes or to control a real vehicle. As for the control we distinguish between path tracking control and path planning control. The former aims to track a predefined (spline) path while the latter generates a path and a trajectory using mathematical vehicle models and online optimization methods in combination with model-predictive control, see, e.g., [7]. In both, path tracking and path planning, the aim is to realize the feedback law μ_i in (2). Currently, a dynamic inversion controller and a linear model-predictive controller are used for path tracking, see [5, 3] for details. These controllers can be applied to both models in Section 2.1. The model-predictive control concept is applicable to path planning as well, compare [7]. Since model-predictive control is a powerful and versatile control paradigm, especially for multi-agent systems, we outline in brief the working principle. Further details can be found in the monographs [14, 9].

To this end we consider dynamics in discrete time $t_n = t_0 + nh$, $n \in \mathbb{N}$, where $h > 0$ is the stepsize given by the control timer in ROCS. For notational convenience we restrict the discussion to $N = 1$ agent with state x, control u, and dynamics (1). Discretization of the latter using a suitable Runge-Kutta method leads to a discrete time system. A typical path tracking task requires to solve a linear-quadratic optimization problem of the following type at each t_n with measured state x_n at t_n:
Minimize the tracking error

$$\frac{1}{2}\sum_{k=n}^{n+M-1} \|x(t_k) - x_{ref}(t_k)\|^2 + \|u(t_k) - u_{ref}(t_k)\|^2$$

subject to the constraints

$$\begin{aligned} x(t_{k+1}) &= A_k x(t_k) + B_k u(t_k) && (k = n,\ldots,n+M-1)\\ x(t_k) &\in X && (k = n,\ldots,n+M)\\ u(t_k) &\in U && (k = n,\ldots,n+M-1)\\ x(t_n) &= x_n \end{aligned}$$

Herein, the linear dynamics are obtained by linearization at the reference path (x_{ref}, u_{ref}). The number $M \in \mathbb{N}$ denotes the preview horizon, which has to be chosen appropriately. The sets X and U define state and control constraints. Likewise a typical path planning task consists in solving a nonlinear optimization problem of the following type at t_n with measured state x_n at t_n:
Minimize the objective

$$\varphi(x(t_{n+M})) + \sum_{k=n}^{n+M-1} \ell(x(t_k), u(t_k))$$

subject to the constraints

$$\begin{aligned} x(t_{k+1}) &= F(x(t_k), u(t_k)) && (k = n,\ldots,n+M-1)\\ x(t_k) &\in X && (k = n,\ldots,n+M)\\ u(t_k) &\in U && (k = n,\ldots,n+M-1)\\ x(t_n) &= x_n \end{aligned}$$

Herein, φ and ℓ are suitable functions modelling the control objective, e.g. driving fastly or economically. Now, the standard model-predictive control (MPC) concept requires to solve one of the above optimization problems repeatedly on a shifted time horizon. Figure 7 shows the outcome of a path planning task using the single track model in Section 2.1 for a track on the campus of the Universität der Bundeswehr München. Obstacles can be avoided as well, see [3].

3.1 Sample vehicle controller

We outline path tracking controllers for the vehicle models in Section 2.1. For the implementation of the controllers we use slightly modified models in terms of a curvilinear coordinate system:

$$s'(t) = \frac{v(t)\cos(\psi(t) - \psi_m(t))}{1 - r(t)\kappa_m(s(t))}, \tag{15}$$

$$r'(t) = v(t)\sin(\psi(t) - \psi_m(t)), \tag{16}$$

$$\psi'(t) = v(t)\kappa(t), \tag{17}$$

$$\kappa'(t) = u(t), \tag{18}$$

$$\psi_m'(t) = v(t)\kappa_m(s(t)). \tag{19}$$

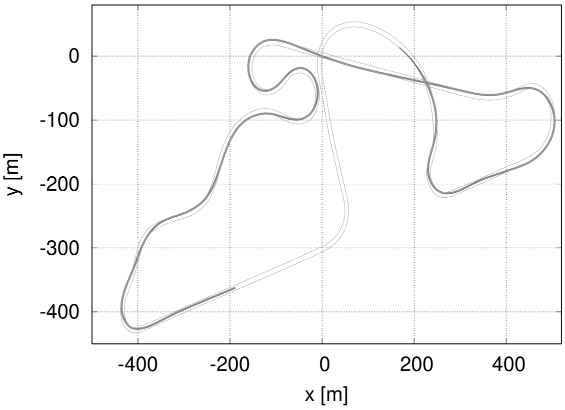


Figure 7: Nonlinear MPC result of a path planning task.

where s is the arc length along a given reference spline curve and r is the lateral offset from the reference spline. The actual heading is given by $\psi(t)$ and the corresponding reference heading is given by $\psi_m(t)$. The curvature of the driven path is denoted by κ and the curvature of the reference path is κ_m.

Both controllers are based on a simple kinematic model, Eqs. (15) to (19) and are designed to control the curvature deviation to track a given reference path. Herein, the controller class provides a control input to the vehicle models through a signal and slot connection. This allows for an easy extension with additional controllers, since the user only needs to provide an output signal. Then, the output is transformed for the respective model and eventually can be integrated employing one of the integrators, provided by the integration class, see Figure 8.

The aforementioned transformations are model dependent. The single track model could be controlled through the steering angle δ, the commanded steering angle δ_c or the steering angle rate $\delta' = \omega_\delta$ respectively. Hence, we require a relation between the output of the controller, i.e., κ, and the control variables. For the commanded steering angle and the steering angle velocity, respectively, these relations are given by

$$\delta_c = \arctan\left(\ell\kappa\right), \omega_\delta = \ell\kappa' \cdot \cos^2\left(\delta\right). \qquad (20)$$

Herein, $\delta' = \omega_\delta = \frac{\delta_c - \delta}{T_c}$ with constant $T_c > 0$. The two wheeled robot is steered through the velocities of the left and right wheel. Exploiting physical relations yields,

$$v_L^c = v_d - \frac{1}{2}B \cdot v \cdot \kappa, \quad v_R^c = v_d + \frac{1}{2}B \cdot v \cdot \kappa, \qquad (21)$$

with B the width of the robot, v_d the desired longitudinal velocity, and $v = (v_L + v_R)/2$ the current velocity. Both controllers are discussed in detail in [4] and [3].

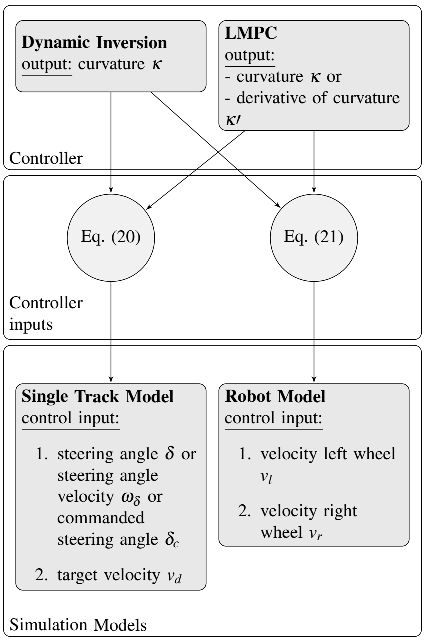

Dynamic Inversion
output: curvature κ
LMPC
output:
- curvature κ or
- derivative of curvature $\kappa\prime$
Controller
Eq. (20)
Eq. (21)
Controller inputs
Single Track Model
control input:
1. steering angle δ or steering angle velocity ω_δ or commanded steering angle δ_c
2. target velocity v_d
Robot Model
control input:
1. velocity left wheel v_l
2. velocity right wheel v_r
Simulation Models

Figure 8: Information flow between simulation models and controllers.

4 Visualization

For the visualization of the control and simulation results we utilize the Qt Framework, which provides an OpenGL high level interface and allows for performant rendering in C++ applications. To visualize certain objects the data structure is based on a scene graph defined by a system of entities, where the scene graph is a tree structure made of these entities and other components. The entities to be rendered can be assigned through object files containing 3D models of, e.g., a car, an air plane, a robot, or buildings. Therefore we implemented an overloaded class of QSceneLoader addressing our requirements and managing the entities, as well as interfacing the rendering canvas. Herein, the rendering is solely a data driven process. Prebuild camera entities are provided by Qt providing viewpoints through which the scene is rendered. Multiple cameras are implemented in ROCS to capture different perspectives, e.g., the ego person's view in 9, the third person view, see Figure 10, where the camera follows in a fixed distance behind the object and a birds view in Figure 11.

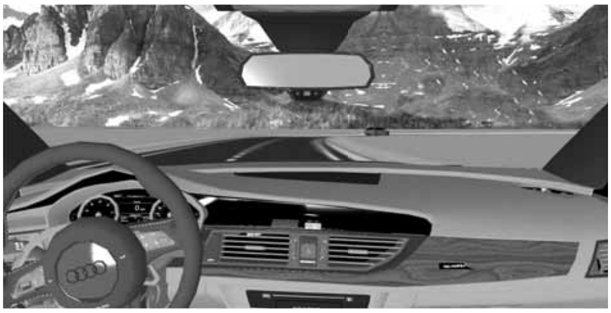


Figure 9: Ego person's view of a scene.

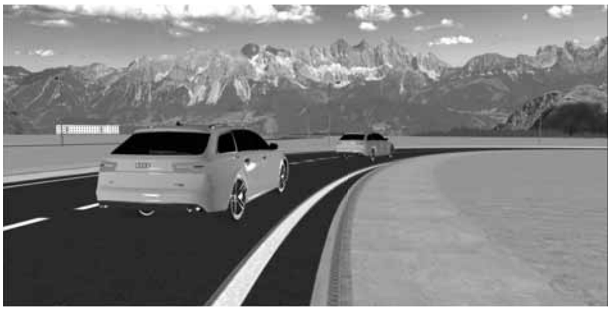


Figure 10: Third person's view of a scene.

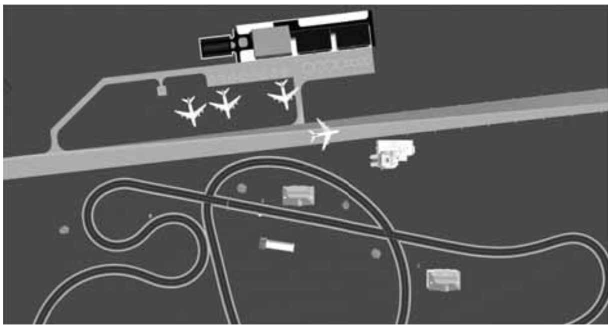


Figure 11: Bird's view of a scene.

5 Evaluation and Results

Figure 12 shows selected car data stored by the data logger function of ROCS. The virtual RAM used by ROCS for this simulation amounts to 2.37 GB and the RAM to 3.67 GB for a total of 12 simultaneously controlled cars. The GPU load amounts to 35.2 %. The computations were performed on a system with 16 GB of memory, Intel i7-8700 processor with 3.2 GHz (6 cores, 12 threads) and a Nvidia Geforce FTX 1060GB (6 GB RAM) graphic card.

The results show the output of the single track model with a linear model-predictive path tracking controller for the track depicted in Figure 7. This controller is able to track a given geometric reference path even for comparatively high velocities with a maximum deviation of 0.15 m (see r in Figure 12).

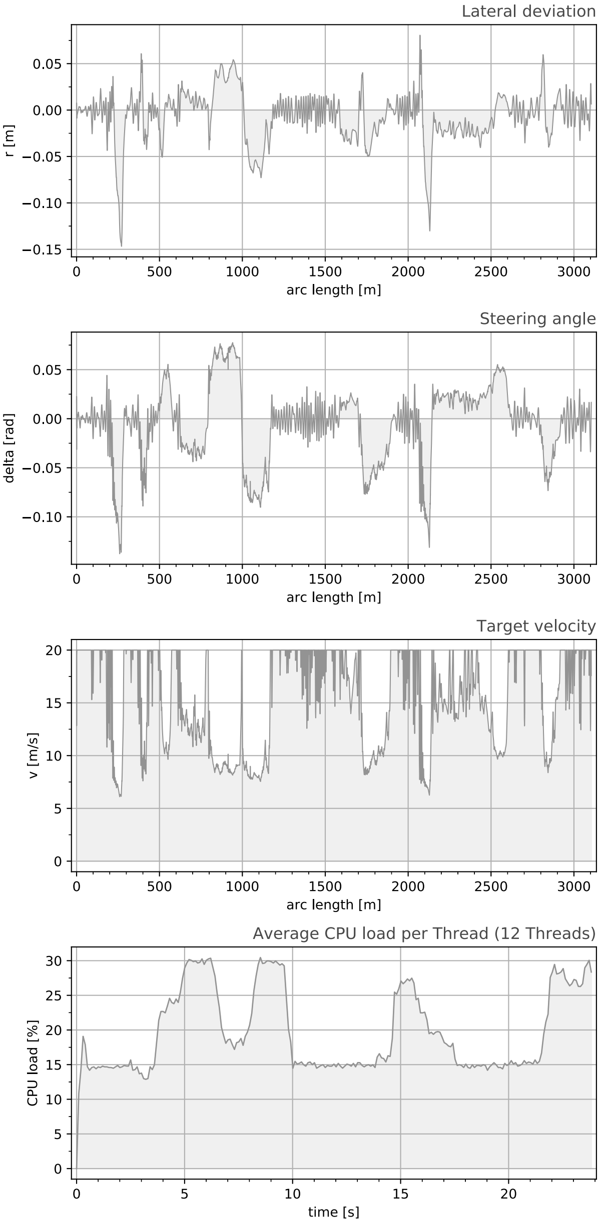


Figure 12: Data logger output (from top to bottom): lateral deviation r, steering angle δ, target velocity, and average CPU load per thread (12 threads, 12 vehicles).

6 Current Developments and Future Extensions

The development of ROCS is on-going and vehicle models from different disciplines (mobile robots, flight systems, space systems) of different complexity with appropriate controllers and path planning tools will be added step-by-step. The interfaces of ROCS will allow to directly import real sensor measurements of vehicles and to generate data to control a vehicle. This option allows to run simulation and real vehicle motion in parallel in order to overlap the two motions with the aim to design accurate digital twins. At the same time it allows to simulate a virtual world for a research vehicle at the Universität der Bundeswehr called Vehicle-in-the-loop [1, 2]. This research platform is based on a real car (Audi A6 Avant) and uses virtual environments to couple real driving experience and virtual scenarios. This concept is ideal for testing potentially dangerous scenarios in a safe way and we aim to integrate ROCS into the vehicle in the loop (VIL) for visualization, but also as an automatic control tool.

Acknowledgement

Copyright and ownership of ROCS and its derivatives solely resides with its founders Andreas Britzelmeier and Matthias Gerdts.

References

[1] Berg G, Karl I, Färber B. Vehicle in the loop - validierung der virtuellen welt. In Nichtred. Ms.-dr., editor, *Der Fahrer im 21. Jahrhundert: Fahrer, Fahrerunterstützung und Bedienbarkeit*, volume 6. Verein Deutscher Ingenieure, VDI-Verl., November 2011.

[2] Berg G, Nitsch V, Färber B. *Vehicle in the loop*. In: Winner H., Hakuli S., Lotz F., Singer C. (eds), Handbook of Driver Assistance Systems, Springer, 2015; pp. 199–210.

[3] Britzelmeier A, Gerdts M. *A Nonsmooth Newton Method for Linear Model-Predictive Control in Tracking Tasks for a Mobile Robot With Obstacle Avoidance*. in IEEE Control Systems Letters, 2020; Vol. 4(4), pp. 886–891, doi: 10.1109/LCSYS.2020.2996959.

[4] Britzelmeier A, Gerdts M, Rottmann T. *Control of interacting vehicles using model-predictive control, generalized Nash equilibrium problems, and dynamic inversion*. 2020 IFAC World Congress, 2020.

[5] Burger M, Gerdts M. *DAE aspects in vehicle dynamics and mobile robotics*. Applications of differential-algebraic equations: examples and benchmarks, Differential-Algebraic Equations Forum, Springer, 2019; pp. 37–80.

[6] Gerdts M. *Solving mixed-integer optimal control problems by branch&bound: a case study from automobile test-driving with gear shift*, Optimal Control Applications and Methods, Vol. 26, pp. 1–18, 2005.

[7] Gerdts M, Karrenberg S, Müller-Beßler B, Stock G. *Generating locally optimal trajectories for an automatically driven car*. Optimization and Engineering, 2009; Vol. 10, pp. 439–461.

[8] Graichen M, Graichen L, Rottmann T, Nitsch V. Using the projection-based vehicle in the loop for the investigation of in-vehicle information systems: First insights. In *Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems - Volume 1: VEHITS,*, pages 231–237. INSTICC, SciTePress, 2018.

[9] Grüne L, Pannek J. *Nonlinear Model Predictive Control – Theory and Algorithms*. 2nd Edition, Springer, 2017

[10] Hairer E, Norsett SP, Wanner G. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer Series in Computational Mathematics, 2nd Ed., Vol. 8, Berlin-Heidelberg-New York, 1993.

[11] Krueger HP, Grein M, Kaussner A, Mark C. SILAB – A Task Oriented Driving Simulation. *North America*, page 9, 2005.

[12] Microsoft Corporation. Microsoft flight simulator. www.flightsimulator.com, 04 2020.

[13] Nitsch V, Färber B, Rüger F. Automatic evasion seen from the opposing traffic - an investigation with the vehicle in the loop. In *IEEE 18th International Conference on Intelligent Transportation Systems*, 2015.

[14] Rawlings JB, Mayne DQ, Diehl M. *Model Predictive Control: Theory, Computation, and Design*. 2nd Edition, Nob Hill Publishing, Madison, 2018.

[15] Roth E, Dirndorfer T, Knoll A, von Neumann-Cosel K, Ganslmeier T, Kern A, Fischer MO. *Analysis and validation of perception sensor models in an integrated vehicle and environment simulation.* Proceedings of the 22nd Enhanced Safety of Vehicles Conference, 2011.

[16] Unity 3D. Unity website. unity.com, 04/2020.

[17] VIRES Simulationstechnologie GmbH. Virtual test drive. vires.com/vtd-vires-virtual-test-drive, 04/2020.

[18] Laminar Research. X-plane 11. www.x-plane.com, 04/2020.

Simulating and Evaluating Different Boarding Strategies on the Example of the Airbus A320

Jürgen Wunderlich*

Faculty of Computer Science, Landshut University of Applied Sciences, Am Lurzenhof 1, 84036 Landshut, Germany; *juergen.wunderlich@haw-landshut.de

SNE 31(4), 2021, 217-222, DOI: 10.11128/sne.31.tn.10585
Received: 2021-03-15 (Selected ASIM SST 2020 Postconf. Pub., English version); Revised: 2021-10-30; Accepted: 2021-11-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Based on an Airbus A320 simulation model, this paper compares the boarding times of random boarding with the most frequently applied boarding strategy back-to-front boarding as well as the alternatives outside-in boarding and back-to-front combined with outside-in boarding. The study shows that, on the one hand, outside-in boarding can reduce boarding times by more than 12%, but on the other hand, this also requires a high degree of discipline. As boarding is part of the turnaround process, shorter boarding times mean faster take offs and allow an airport to handle a higher number of planes.

Introduction

Boarding strategies are the subject of ongoing discussions. As a result, travellers, airlines, and airports regularly develop ideas for improvement.

These ideas were rarely implemented pre-pandemic. However, current challenges posed by infection control and the results of the following simulation study may change this attitude.

1 Motivation

In times of COVID-19, many airlines have adjusted the boarding process to ensure that the minimum distance can be maintained [3]. This is an excellent opportunity to think about boarding strategies different from the most common one [2], which is to let passengers board the aircraft in groups, starting with the back rows after first-class and other priority passengers have boarded.

Additionally, boarding is part of the turnaround process. This term denotes the handling of an aircraft between landing and takeoff and should be as efficient as possible, which becomes all the more important as air traffic increases again. The reason is that certain activities, such as safety instructions by flight attendants, cannot begin until boarding is completed. In this respect, several criteria and their interactions must be taken into account, for which simulation is a suitable approach.

2 Objective

The final goal of this study is to improve the flow and efficiency of the boarding process by selecting the most appropriate boarding method. The basic idea here is to avoid congestions in the aisle as much as possible by varying the sequence of boarding passengers, which should lead to an acceleration of boarding, hence to a reduction of the boarding time required and, consequently, to a shorter turnaround process time.

Since the turnaround process is similar for most aircraft types and only the sequence and duration of individual sub-processes differ, the present study will continue to be structured in such a way that the core concepts can easily be transferred to other aircraft types. For this purpose, a realistic reference system is first defined on the basis of which the simulation model is finally created and the advantages and disadvantages of each boarding strategy investigated are evaluated.

3 Reference System

With more than 15,000 aircraft sold, the A320 family is Airbus' greatest success [5]. Therefore, this aircraft type is chosen for this simulation. By default, its seats are divided into business and economy class and arranged in a configuration of 154 seats with 30 rows of seats, 2 of which are not intended for passengers. Of these, 28 seats in the first 7 rows are in Business Class and 126 seats in rows 8-30 are in Economy Class [4]. For passengers, a distinction is made between business travelers and tourists, or whether they carry hand luggage to be stowed in the overhead lockers.

Furthermore, the reference system and subsequently the simulation also are based on the following eight assumptions:

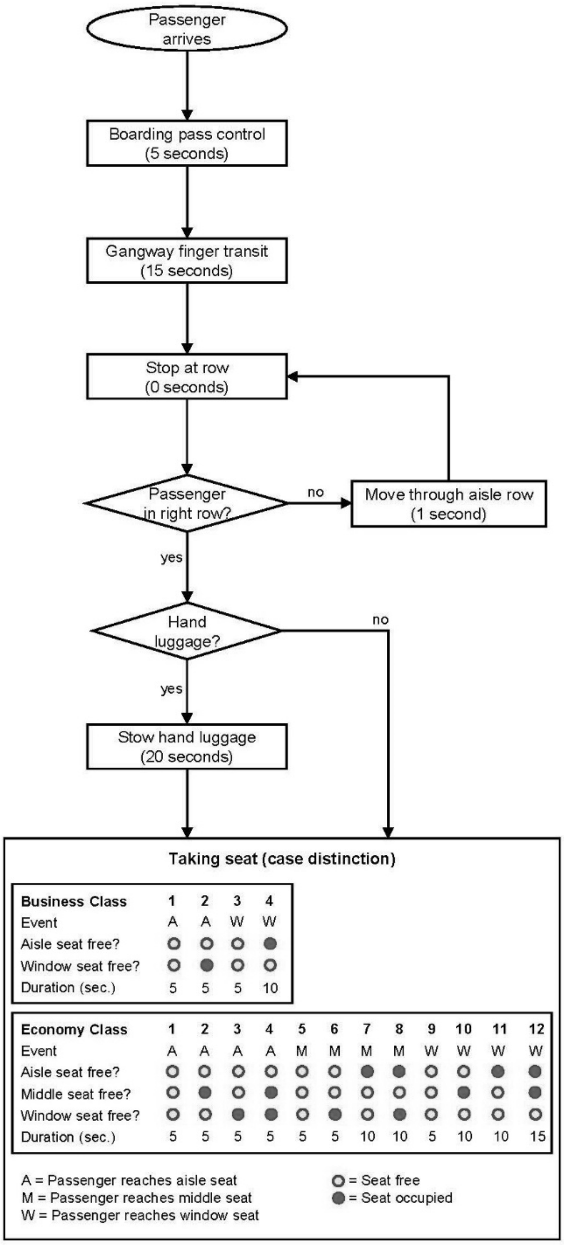
- boarding begins when the passengers are called, at which time both the checked-in passengers and the aircraft are already at the gate
- each passenger already has a fixed seat, i.e. there is no free choice of seats
- the boarding pass control is carried out by the airport staff and ensures compliance with the order of boarding
- passengers enter the aircraft in the order of boarding through the front aircraft door via a passenger boarding bridge
- a single aisle is defined by the A320 aircraft type, which means that there is only one aisle available for moving through the aircraft
- on the plane passengers behave well and do not pass each other
- passengers have only one or no pieces of hand luggage, which already have the prescribed dimensions
- there is sufficient storage space for the hand luggage of each passenger, so that entering passengers can stow it without capacity-related time delays

The simplest boarding method used is random boarding, which is practiced, for example, by the airlines Lufthansa and Eurowings at Munich Airport. With the random boarding method, all passengers have a reserved seat, but can board the aircraft in random order without any further instructions. Only the individual booking classes (Zone 1 for Business and Zone 2 for Economy Class) are boarded one after the other.

This method is strikingly simple. An additional advantage is the distributed utilization of the aircraft aisle. Passengers do not crowd the same rows or the same overhead lockers all at once. With this boarding method traffic jams also occur - for example, because people have to rise from their seats repeatedly to let others pass - but these are at least spread over the entire aisle of the aircraft. A disadvantage of this method is that the order of boarding cannot be influenced.

The boarding process itself begins with boarding pass control. Passengers then pass through the buffer sleeve before reaching the first row of aisles. There they check to see if their seat is in it. If this is the case, they first stow their carry-on luggage - if any. This is followed by taking their seats.

The time required for this depends on the exact location of the seat and on how many seats in front of the assigned seat are already occupied. As long as a passenger has not reached his or her destination row, he or she continues to walk row by row until his or her seat is found. The exact process is shown in Figure 1. In the case of a business class passenger, only the time taken to occupy a seat changes or is shorter compared to an economy class passenger, as there are only two seats on each side in business class.


Passenger arrives
Boarding pass control (5 seconds)
Gangway finger transit (15 seconds)
Stop at row (0 seconds)
Passenger in right row?
no
Move through aisle row (1 second)
yes
Hand luggage?
no
yes
Stow hand luggage (20 seconds)
Taking seat (case distinction)

Business Class	1	2	3	4
Event	A	A	W	W
Aisle seat free?	○	○	○	●
Window seat free?	○	●	○	○
Duration (sec.)	5	5	5	10

Economy Class	1	2	3	4	5	6	7	8	9	10	11	12
Event	A	A	A	A	M	M	M	M	W	W	W	W
Aisle seat free?	○	○	○	○	○	○	●	●	○	○	●	●
Middle seat free?	○	●	○	●	○	○	○	○	○	●	○	●
Window seat free?	○	○	●	●	○	●	○	●	○	○	○	○
Duration (sec.)	5	5	5	5	5	5	10	10	5	10	10	15

A = Passenger reaches aisle seat
M = Passenger reaches middle seat
W = Passenger reaches window seat

○ = Seat free
● = Seat occupied

Figure 1: Flowchart of the boarding process from the perspective of a passenger.

The times for the individual process steps were determined in cooperation with Martin Bertling, a process planner at Munich Airport. They are based on the official documentation for airport planners "AIRBUS A320 Aircraft Characteristics Airport And Maintenance Planning, Chapter 5-2-0" (Feb. 2018) [1]. To this, still missing data was added based on the doctoral dissertation "Analyse der Verzögerungen beim Boarding von Flugzeugen und Untersuchung möglicher Optimierungsansätze" by Holger Stefan Appel (2014) [2].

Key figure	Time data / share	Source
average boarding time	18 minutes	Process planner Munich Airport, official document for airport planners
Boarding pass control	5 seconds	Process planner Munich Airport
Entry time into the aircraft without queueing	15 seconds	Process planner Munich Airport
Taking seat (see fig. 1)	no person: 5 sec. one person: 10 sec. two persons: 15 sec.	Process planner Munich Airport
Hand luggage share	Business travelers: 95% Tourists: 90%	Process planner Munich Airport, PhD thesis Holger Appel
Hand luggage time	20 seconds	Process planner Munich Airport, PhD thesis Holger Appel

Table 1: Data source for the boarding process.

4 Simulation Model

The starting point for the simulation in Plant Simulation is the passenger list. This list determines the order of the passengers arriving at the boarding pass control and contains the time of entry, the assigned seat as well as the number of hand luggage items for each passenger, whereas the order of the passengers and the number or presence of hand luggage items are based on random numbers.

The aircraft type A320 is divided into 28 rows of seats that can be occupied, which are represented in the model as separate application modules. Each application module comprises either four (Business Class) or six (Economy Class) seats, which are replicated in the form of single stations. Both the occupancy of the seats and the passengers in the aisle or in a row of seats are illustrated with the aid of animations. Each passenger is inserted into the simulation model as a moveable unit. The arrival of the first passenger at boarding pass control marks the starting point of the simulation. The simulation is finished as soon as the last passenger has taken his or her seat.

The simulation model was validated against the total lead time for a boarding process. The average lead time for random boarding resulting from multiple simulation runs was 17 minutes and 43 seconds. This means that the deviation from the 18 minutes given by the process planner at Munich Airport as the average boarding time is only 1.57%, which means that the model can be considered valid and thus forms a solid basis for the experiments.

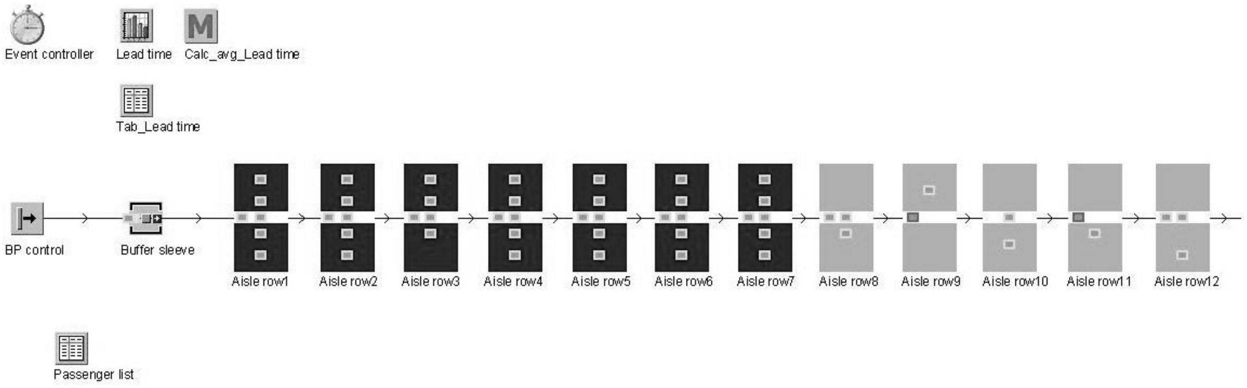


Figure 2: Simulating random boarding (excerpt).

5 Experiments

As alternatives to random boarding, the boarding strategies back-to-front boarding, outside-in boarding, and the combination of back-to-front and outside-in boarding are mentioned in the literature. Therefore, after a brief explanation, a simulation-based investigation of these three methods is conducted.

The basic idea of ***back-to-front boarding*** is to let passengers board from the back to the front [2]. This is to prevent the rear section of the aisle from remaining temporarily unused at the start of boarding because of the first passengers blocking the aisle in the front section in order to stow their hand luggage. This means that the passengers in the back enter the aircraft first (with the exception of Business Class), so that each passenger can reach his or her seat with as few interruptions as possible. The simulation resulted in an average boarding time of 17 minutes and 55 seconds for back-to-front boarding, which means it is twelve seconds slower compared to random boarding.

With the ***outside-in boarding*** method, the plane is boarded from the outside to the inside, i.e. first the window seats, then the middle seats and finally the aisle seats [2]. It does not matter in which row the passengers are seated. However, business class passengers also enjoy higher priority and thus are boarded first. In the simulation, an average lead time of 15 minutes and 34 seconds was achieved for this boarding method. Compared to the initial value, this result yields an improvement of two minutes and nine seconds, or 12.1 percent.

Just as with the back-to-front boarding method, with the ***combination of back-to-front and outside-in boarding***, the passengers enter the aircraft from the back to the front. At the same time, according to the principle of outside-in boarding, first the window seats, then the middle seats and finally the aisle seats are occupied [2]. Here, too, business class passengers are the first to board and economy class passengers follow. In the simulation runs, the average lead time for this boarding method was exactly 16 minutes.

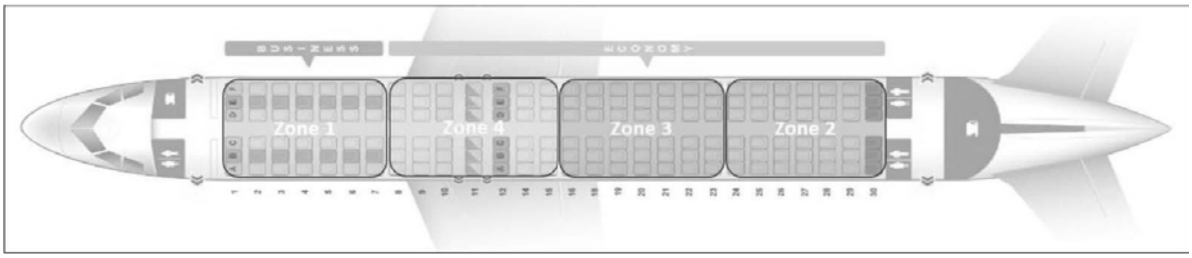


Figure 3: Back-to-front boarding.

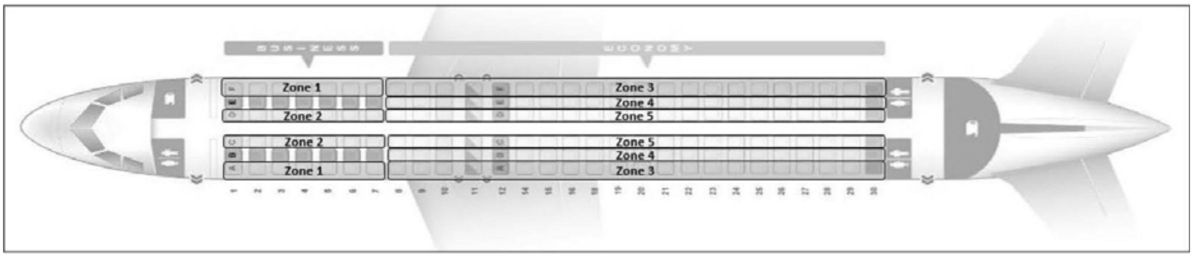


Figure 4: Outside-in boarding.

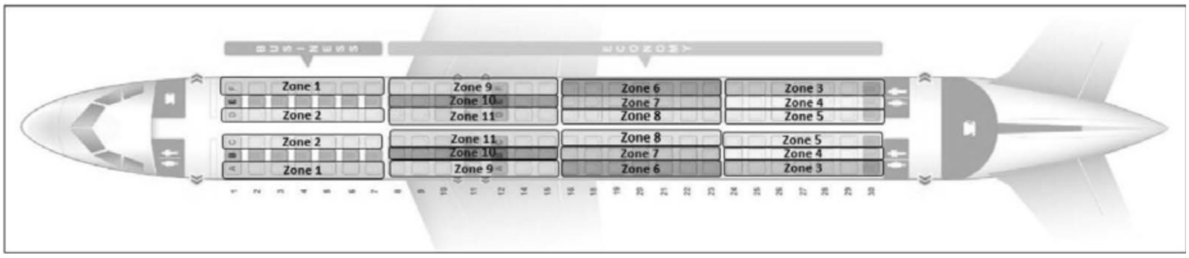


Figure 5: Combination of back-to-front and outside-in boarding.

This value still means an improvement of one minute and 43 seconds compared to random boarding, but a deterioration of 26 seconds compared to pure outside-in boarding.

Overall, ***back-to-front boarding*** is the most inefficient. The main problem is that passengers spend much time waiting in the aisle because a significant number of passengers try to occupy several rows simultaneously. On the one hand, it causes disruption in the row of seats when a passenger who is already seated must get up again because an arriving passenger’s seat is at the window or in the middle. On the other hand, only the first passengers arriving at their row of seats can stow away their hand luggage – blocking the way for all following passangers. As a result, the queue is shifted from the gangway into the aircraft. The advantage of this method, like random boarding, seems to be that it is easy to understand, as the aircraft is divided into few areas only where passengers board at the same time.

The advantage of ***outside-in boarding*** is that it prevents the aircraft aisle from being congested with passengers blocking the way. In contrast to back-to-front boarding, the distribution regarding the utilization of the aisle is improved (similar to random boarding). In fact, once seated, passengers do not have to stand up again and thus do not block the aisle again. The benefits of boarding methods can be seen in the improved lead time. A decisive disadvantage mentioned in the literature is that the seats in a row are not boarded together. This means that travel groups or families have to separate for a short period of time when boarding the aircraft. Accordingly, acceptance of this boarding method is low, since most passengers assiciate a certain level of comfort with flying. As a result, many airlines discarded this boarding method after a short test phase. A possible solution would be to give families, for example, priority when boarding - similar to business class passengers.

With the ***combination of the back-to-front and the outside-in boarding*** **method**, an attempt is made to combine the advantages of these two methods and to offset their disadvantages. The boarding of the aircraft is done from the back to the front, so that the passengers in the aircraft aisle are as evenly spread as possible. At the same time, however, the aircraft is also boarded from the outside to the inside in order to avoid disturbances within the rows, i.e. row interference. In the present simulation, however, these advantages did not appear at an aircraft utilization of 100% (but only in further tests with a utilization of 90% and less).

In addition to this, this method has some practical disadvantages. On the one hand, the correct arrangement of passengers before boarding the aircraft is a challenge, and on the other hand, as already described under the outside-in boarding method, families, or travel groups in general, also have to separate at this point.

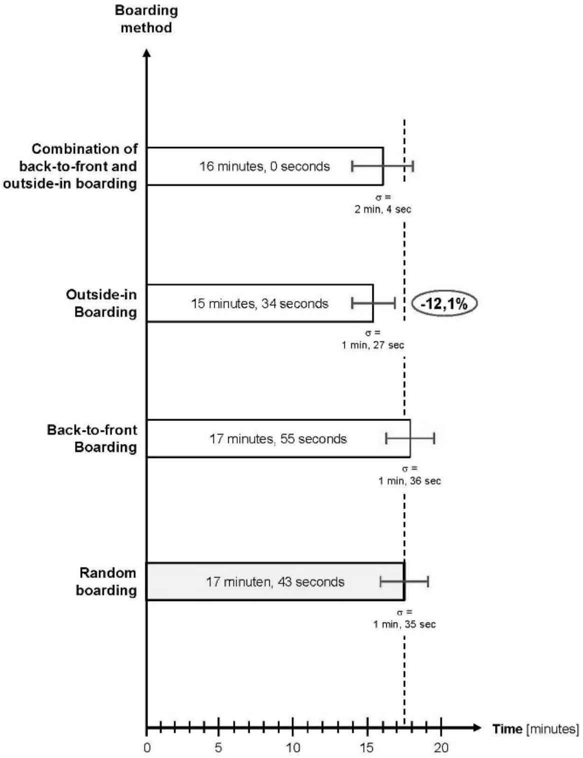


Figure 6: Average boarding duration and standard deviation of the individual boarding methods after 35 simulation runs each.

Should airlines consider changing their current boarding strategy based on the results obtained thus far and with the current situation, the present simulation model offers a good starting point for additional or possible future extensions to approximate the simulated boarding process even closer to reality and hence to determine the optimal boarding strategy for specific environmental or utilization situations. Approaches to possible extensions are, for example, the inclusion of additional boarding methods (e.g. open seating, reverse pyramid, by half block or alternating rows), differentiation according to the number and type of hand luggage in cabin suitcase, hand- or backpack, and jacket including capacity restrictions of the luggage compartments as well as the consideration of different personal walking speeds and an error rate for late passengers.

6 Conclusion

Even with a simple simulation model, it was possible to compare visually the different boarding strategies. What was noticeable was that, at least at a load factor of 100%, the simpler outside-in boarding strategy proved to be more advantageous than the combination of outside-in and back-to-front boarding. The advantage was not only that the average boarding time was 26 seconds shorter, but above all a significantly lower standard deviation of one minute and 27 seconds compared to the standard deviation of two minutes and four seconds for the boarding combination. This increases planning security considerably, which in turn appears to be especially important when the load factor is 100%, which the airlines are aiming for on the few planes currently in use.

In the course of implementation in practice, as a minimum requirement visual aids are needed to increase passengers' understanding of the boarding strategy applied to ensure compliance. Furthermore, it is important to remember that the first prerequisite for a successful realization is that all passengers arrive at the gate at a fixed time at the latest. Since this can almost never be 100% guaranteed, the implementation of even more sophisticated boarding strategies than those presented in this article or discussed in the academic world will fail.

A representative of Lufthansa even considers boarding to be too complex a process to be able to optimize it with mathematical algorithmic methods alone. Experts who can model and simulate should be brought together with psychologists who can understand and explain group phenomena. In addition to this, cross-cultural differences may be relevant and, for example, cause passengers from group-oriented cultures to be particularly considerate and disciplined during boarding, whereas other cultures impress with punctuality and arrive at the gate on time. In this respect, before refining the simulation model, this begs the question of what effort is involved and what benefits are actually reaped in terms of transferability into practice, unless the prerequisites of the model are already deemed to be too restrictive. Since the crowding in airplanes known from the pre-pandemic era must be avoided during the pandemic, also for reasons of infection control, the new situation now at least offers a chance to experiment with comparatively easy-to-understand out-side-in-boarding.

Note for Publication. This article was originally published in German in ARGESIM Report 59 (ISBN 978-3-901608-93-3). I would like to thank Professor Claudia Wunderlich for editing this English language version.

References

[1] Airbus. *A320 – Aircraft Characteristics Airport and Maintenance Planning;* 2018.

[2] Appel H. *Analyse der Verzögerungen beim Boarding von Flugzeugen und Untersuchung möglicher Optimierungsansätze* [Dissertation]. Rheinisch-Westfälische Technische Hochschule Aachen; 2014

[3] Islam T, Sadeghi Lahijani M, Srinivasan A, Namilae S, Mubayi A, Scotch M. *From bad to worse: airline boarding changes in response to COVID-19.* Royal Society Open Science 8: 201019; 2021; doi: 10.1098/rsos.201019

[4] Seatguru. https://www.seatguru.com/airlines/Lufthansa/Lufthansa_Airbus_A320-200_NEK.php; (accessed on 10/30/2021)

[5] Slotnick D. *The amazing story of how the Airbus A320 became the Boeing 737's greatest rival.* Business Insider April 1, 2020; https://www.businessinsider.com/airbus-a320-history-boeing-737-rival-2018-9; (accessed on 10/30/2021)

Modeling the Spread of Tree Pests after Aerial Pest Control with the Means of a Geo-Information System

Colja Krugmann, Jochen Wittmann*

HTW Berlin University of Applied Sciences, Environmental Informatics, Wilhelminenhofstraße 75A,
12459 Berlin, Germany, **wittmann@htw-berlin.de*

SNE 31(4), 2021, 223-226, DOI: 10.11128/sne.31.sn.10586
Received: 2021-03-15 (Selected ASIM SST 2020 Postconf. Pub., English version); Revised: 2021-10-15; Accepted: 2021-11-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. As part of a student project for the module Environmental and Geoinformation Systems, this article takes up the discussion about aerial pest control in forests. A model based solely on publicly available data is created to identify the area in which the aerial application of insecticides is permitted. It further estimates the renewed spread of forest pests after the control measure using a simple dynamic model. The background of this model is the fact that the insecticide is applied in compliance with protective zones around populated areas, as well as surface waters and forest edges. In those untreated areas of the forest, the pests survive and will spread anew. Assuming a certain duration for the propagation cycle of the pests as well as an average radius, the dynamic spread of the pest population over several propagation cycles can be simulated and displayed using simple onboard methods of a geoinformation system. The difficulty in determining these zones is the fact that the geodata of German forests is frequently intersected by small roads that are irrelevant to the aerial application of the insecticide. As a feasibility study, this paper proves that such a model can provide useful information. For the case of an actual aerial application of an insecticide, it provides a list of essential parameters for a practically oriented simulation.

Motivation

In 2019, an aerial application of Syngenta's "Karate Forst flüssig", a liquid agent against forest pests, primarily the nun moth (Lymantria monacha), was considered in Brandenburg. A large number of environmentalists and local residents opposed these plans (rbb24.de, 2019). Pine trees account for 70% of Brandenburg's forests.

This monoculture in combination with the drought of the recent years increased the susceptibility of the trees to pest organisms such as the pinetree lappet (Dendrolimus pini), pine looper moth (Bupalus piniaria), pine sawfly (Diprion pini), pine beauty moth (Panolis flammea) and nun moth (Lymantria monacha). Forest monitoring predicted a high risk of clearcutting under favorable conditions for the pests, such as warm and dry weather.

Pine trees can survive a onetime defoliation of about 90% with little loss, but in case of further or repeated damage, the forest is expected to die. As forests are a significant factor in the carbon dioxide (CO_2) balance, a complete defoliation would transform them from a CO_2 sink into a CO_2 source.

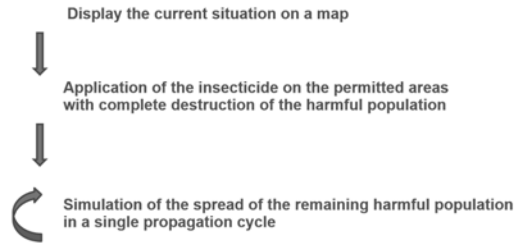
The Landeskompetenzentrum Forst Eberswalde (LFE) planned to encounter this situation with aerial pest control using Syngenta's Karate Forst flüssig. This insecticide is a contact poison that is applied to the canopy of the infested trees in highly diluted form in order to kill the nun moth in its caterpillar stage (Waldschutzmaßnahmen gegen Nonnenraupen | Landesbetrieb Forst Brandenburg, n.d.). Conservationists strongly doubted the usefulness and environmental compatibility of this approach as the natural enemies of the pests would also be affected (rbb24.de, 2019). Though according to the LFE, the concentration of the insecticide can be set just high enough to kill only the nun moth caterpillars, leaving their natural enemies and for example cockchafers or other ground beetles unaffected (Waldschutzmaßnahmen gegen Nonnenraupen | Landesbetrieb Forst Brandenburg, n.d.).

At this point, the present feasibility study intervenes in the discussion. It examines whether statements about the effects of aerial pest control on the development of pest infestations can be made by means of a model developed based exclusively on publicly available data. In the model special consideration is given to the importance of the safety distances that must be maintained around settlements, water bodies, etc. when applying the insecticide. These protected zones thus constitute refuge areas for the pest populations that can not only recover but also spread anew.

1 Model Idea and Preparation for a GIS Analysis

The idea of the present work is first of all to use only publicly available maps as basic data for the usecase showing the areas permitted and suitable for the aerial application of the insecticide. It also attempts to estimate the distribution of forest pests after the control measure within the framework of the development of a dynamic model.

The basic assumption for the modeling is that the protected zones that are excluded from the application of the pesticide provide refuge areas for the pests and therefore serve as a basis for a subsequent reinfestation. Assuming a certain duration for the reproduction cycle of the pests as well as an average distribution radius, the dynamic spread of the pest population over several reproduction cycles can be simulated and represented using only standard methods for spatial analysis provided by a geographic information system (Figure 1).


Display the current situation on a map

Application of the insecticide on the permitted areas with complete destruction of the harmful population

Simulation of the spread of the remaining harmful population in a single propagation cycle

Figure 1: Model idea.

The aim of this modeling is to highlight the influence and the importance of the protected zones and to open them to a quantitative investigation. A topographically highly structured landscape that is strongly interspersed with large and numerous areas protected from the application of the insecticide will show very different effects on the assumed reinfestation process than a landscape with large structures with accordingly less protected areas.

To carry out this analysis in a geoinformation system, the following subtasks arise for the modelling:

1. finding and processing suitable publicly accessible geodata
2. determination of the parameter values necessary for modelling
   a. With regard to the application instructions of the insecticide
      i. Safety distance to settlements
      ii. Safety distance to bodies of water
      iii. Safety distance to forest edges
   b. With regard to the biological data of the pest
      i. Duration of a reproduction cycle
      ii. Propagation range in one reproduction cycle

2 Analysis Workflow

2.1 Step 0: Creating a Base Map and Basic Geodata

The limitation of this feasibility study to publicly available maps represents a certain challenge, which on the one hand requires additional work steps, but on the other hand underlines the independence of this approach from already existing, but only limited and/or conditionally available data.

A region in Brandenburg between Bad Belzig and Werder (Havel) was chosen as the sample area. The underlying geodata for the modeling is as following:

- A cutout from openstreetmap was selected as base (Geofabrik Download Server, 2019).
- The watercourse data as shape files originate from Geoportal Brandenburg (Geoportal Brandenburg, 2019).
- The road network data for Brandenburg was downloaded as GML-file from Inspire Brandenburg via API. (LGB (Landesvermessung und Geobasisdaten Brandenburg), 2019).
- The forest data originate from the Landeskompetenzzentrum Brandenburg and were available as a GML file (Forstgrunddaten - Flaechen - Waldbedeckung - Land Brandenburg INSPIRE, 2018)
- Ideally, residential and agricultural areas are available as shape files to apply the exact safety distances to each. In this example they are represented by the blank spaces within the forest areas.

2.2 Step 1: Creating Contiguous Forest Areas

The geodata of the forest areas are dissected by smaller or dirt roads, thus representing fictitious edges of the forests that would lead to protected zones in the modeling. With respect to an insecticide application, forest roads do not count as a boundary of a forest area and must therefore be ignored for this project.

To achieve this a buffer is placed around the forest areas to cover the respective forest roads. The resulting overlapping forest polygons are then merged by applying a dissolve. A subsequent negative buffer of the same width is applied to the resulting areas to reduce them to their original and actual size. (Sketch of the procedure Figure 2, base map after this processing step in Figure 3).

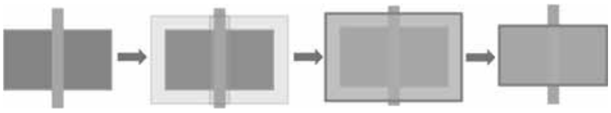


Figure 2: Step 1: Creating contiguous forest areas.

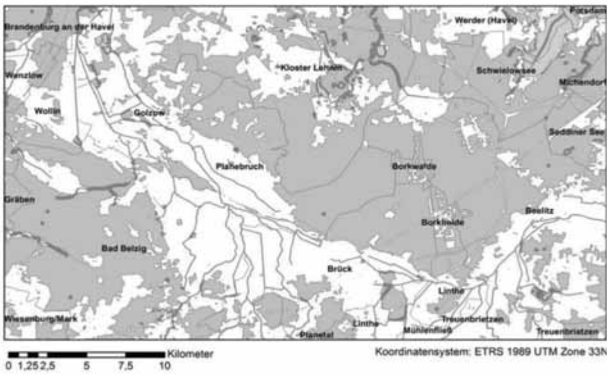


Figure 3: Base map after step 1.

2.3 Step 2: Creating Uniform Water Surfaces and their Protected Zones

Geodata for lakes are available as polygons, whereas rivers and canals are represented by lines. To obtain one layer containing all waterbodies as continuous polygons, the lines representing rivers and creeks need to be converted into polygons. They can subsequently be combined with the lake areas.

For this purpose, a buffer was created around the rivers as a line feature. Afterwards lakes and rivers can be merged to one layer "water bodies", which is relevant for the determination of the protected zones. All water features must be buffered again by the width of the protected zone to create polygons that represent the water bodies including their safety distance (Figure 4).

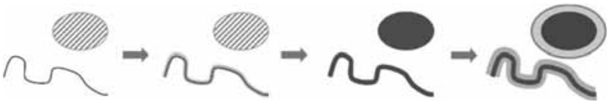


Figure 4: Step 2: Creating uniform water surfaces with safety distance.

2.4 Step 3: Creating the Safety Distance from Major Roads and Merging of all Protected Zones

The aerial application of pesticides over larger roads, such as federal and district roads or federal motorways, is not permitted. As for rivers, geodata for roads are represented as line features. To determine the protected zones along these roads, they are buffered according to an averaged road width and the prescribed safety distance. The resulting protected zones around roads are then merged with those around water bodies. (Figure 5).

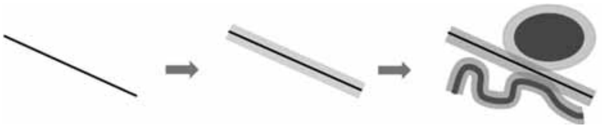


Figure 5: Step 3: Safety distance from major roads and merging of protected zones.

2.5 Step 4: Determining all Forest Areas left Untreated by the Aerial Pest Control

To determine all forest areas that must not be treated with the insecticide, the treatable forest area must be identified. This is accomplished by applying a negative buffer of the prescribed safety distance to forest edges to forest polygons resulting from step 1. This area is then intersected with the polygons representing the protected zones of water bodies and major roads from steps 2 and step 3. The result of this intersection is now subtracted from the buffered forest area by a symmetrical difference, to obtain all the areas of the forest eligible for an aerial application of the insecticide.

Another symmetrical difference of this area and the entire forest area results in a polygon representing all forest areas excluded from the aerial application (Figure 6).

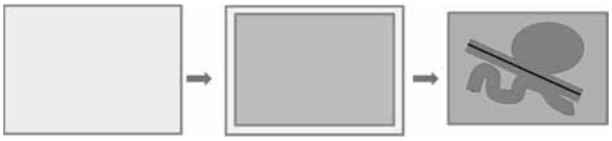


Figure 6: Step 4: Determination of untreated forest areas.

2.6 Step 5: Spread of the Pest

It is assumed that the pests survive completely in the protected areas. From there they can then spread again to the neighboring and previously treated areas. This spreading process is modeled a buffer around the untreated areas. The buffer width corresponds to the radius of the spread of the pest in one propagation cycle. This assumed spread extends beyond the forest boundaries. Accordingly, the resulting polygons are intersected with the forest areas resulting from step 1. In addition, there may be an overlap of parts of the dispersal zone, which are removed by a dissolve (Figure 7). The result is a polygon representing all infested areas including the protected zones from step 4.

This procedure can be run repeatedly based on these polygons to simulate the expansion of the pest with each propagation cycle.

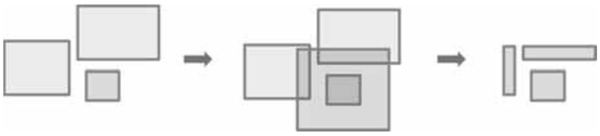


Figure 7: Step 5: Spread of the pest.

3 Results

3.1 Feasibility Study for the Sample Region

The analysis described in the previous section were applied to the sample region from Figure 3 by using the Model Builder in Esri's ArcMap 10.5. The values for the various parameters mentioned in the individual steps were plausibly estimated or extracted from manuals for the aerial application of pesticides. The exact values are deliberately not mentioned in this paper, since the model is only a prototype aimed at the analysis workflow. The results can serve as a plausibility test. The parameterization will be discussed separately in the following subsection.

As a result of the modeling and the simulation, three qualitative conclusions can be drawn:

1. The described step-by-step analysis workflow can be mapped completely and transparently with the standard methods of ArcMap. It is available as a freely parameterizable new tool.

2. It is possible to implement a simple dynamic simulation by iterating the propagation method explained in step 5.
3. The procedure described by the workflow leads to plausible results, as shown in Figure 8.

As shown in the result map after three defensively parameterized propagation cycles large parts of the forest area are already newly colonized by the pest (marked orange to yellow, the protected zones are marked red in Figure 8). The model emphasizes the dependency of the degree of pest infestation on the landscape's structure.

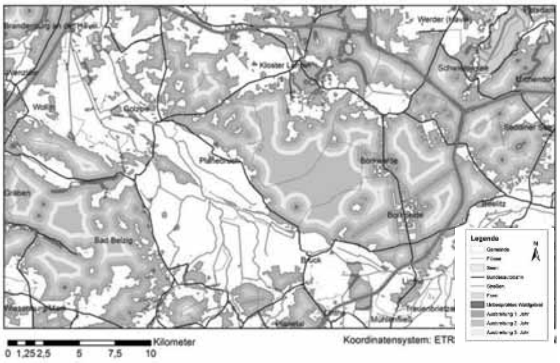


Figure 8: Simulated state of pest infestation for the example region after three propagation cycles.

3.2 Necessary Parameter Values for an Accurate Model Study

As already mentioned, the model presented is based on publicly available data. In several instances it is dependent on parameter values that could only be plausibly estimated in the feasibility study (e.g. the width of roads and rivers). For a reliable quantitative investigation, the parameters of the model must be filled with valid values. However, it should be noted that an aim of this specific modeling was to obtain results using a small number of parameters, that are relatively easy to determine.

Below is the complete list of these model parameters that must be provided for a concrete application:

1. Geospatial data:
   a. Base map (the feasibility study deliberately works exclusively with publicly available material; better suited thematic base maps may be available).
   b. Width of small watercourses
   c. Width of roads
2. Application instructions of the insecticide:
   a. Minimum distance to forest edges
   b. Minimum distance to surface water
   c. Minimum distance to residential areas
3. Data on the biology of the pest:
   a. Duration of a reproductive cycle
   b. Range of spread in one propagation cycle.

It goes without saying that many other influencing factors are conceivable and useful for a more detailed modeling. Two examples for influencing factors on the risk and spread of a pest infestation that were ignored in this model are the weather and the forest's composition as its main goal was to demonstrate its feasibility.

4 Conclusion

The present work is intended to show that even a very simple modelling approach can generate meaningful results allowing for a determination of untreated areas, as well as for an estimation of the effects of an intervention in pest infestations by aerial application insecticide. With a manageable amount of parameter values required and little expenditure, both algorithmically and in terms of software technology, this model can help objectifying a discussion about the application of insecticides by investigating different scenarios obtained through parameter variations.

On the methodological level, the presented model shows that the frequently discussed coupling and/or integration of geoinformation and simulation systems can succeed pragmatically and successfully for individual projects with a narrow requirement profile on the one hand and a very pragmatic modeling approach on the other.

References

[1] Geofabrik.de. 2019. Geofabrik Download Server. [online] Available at: <https://download.geofabrik. de/europe.html> [Accessed 7 July 2019].

[2] Landesbetrieb Forst Brandenburg. n.d. Waldschutzmaßnahmen gegen Nonnenraupen | Landesbetrieb Forst Brandenburg. [online] Available at: <https://forst. brandenburg.de/lfb/de/lfe/waldschutzinformationen/waldschutz-massnahmen-gegen-nonnenraupen/> [Accessed 3 April 2020].

[3] Brandenburg-forst.de. 2018. Forstgrunddaten - Flaechen - Waldbedeckung - Land Brandenburg INSPIRE. [online] Available at: <http://www. brandenburg-forst.de/inspire/dls/ifgk_wld/> [Accessed 23 April 2018].

[4] LGB (Landesvermessung und Geobasisdaten Brandenburg). 2019. Geoportal Brandenburt. [online] Available at: <https://geoportal. brandenburg.de/ startseite/> [Accessed 14 May 2019].

[5] Majunke, C., Möller, K. and Funke, M., 2004. Die Nonne. Waldschutzmerkblatt 52. [online] Potsdam: Landesforstanstalt Eberswalde. Available at: <https://opus4.kobv.de/opus4-slbp/files/5068/ nonne.pdf> [Accessed 1 May 2019].

[6] OpenStreetMap. 2019. OpenStreetMap. [online] Available at: <https://www.openstreetmap.org> [Accessed 28 April 2019].

[7] rbb24.de, 2019. Mit „Karate Forst“ gegen Raupen in Brandenburgs Wäldern. [online] Available at: <https://www.rbb24.de/panorama/beitrag/2019/04/brandenburg-beelitz-mit-karate-forst-fluessig-gegen-insekten.html> [Accessed 3 April 2020].

System Simulation as Part of Systems Engineering for Headlamp and Pedal Systems based on the Modeling Language *Modelica*

Heinz-Theo Mammen*, Phillip Limbach, Thorsten Maschkio

HELLA GmbH & Co. KGaA, Rixbecker Straße 75, 59552 Lippstadt; *Heinz-Theo.Mammen@hella.com

SNE 31(4), 2021, 227-232, DOI: 10.11128/sne.31.tn.10587
Received: 2021-01-10 (Selected ASIM SST 2020 Postconf. Pub., English version); Revised: 2021-09-17; Accepted: 2021-11-26
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Based on two application examples, i.e. a DC-actuator and a brake pedal system, the analysis of this article demonstrates, which synergies can be achieved by applying system simulation. The first example, focussing on the simulation of the temperature behaviour of a DC-actuator to identy critical operating conditions, reveals an effort reduction by a factor of 4 in comparison to measurements. The second example, a brake pedal system, shows an approach of contact modeling between rigid bodies in Modelica, in order to analyse the kinematic movement of the pedal and its force-path characteristics. In this case, the original effort of model creation has been reduced by a factor of 10 s.

Beside the increase of efficiency, the respective de-sign concepts could be optimized concerning the friction and temperature behaviour according to the specifications.

Introduction

Due to rising complexity of automotive systems and subsystems, system simulation is becoming more and more important to continuously analyse and verify the interdisciplinary system behaviour along the product development process.

Therefore, system models for various subsystems and components have been developed and validated at HELLA in recent years. In this article two examples, i.e. a DC actuator of a mechatronic headlamp module and a brake pedal system, are presented, aiming at the development of applicable models for an efficient and effective system development and optimization.

Considering temperature behaviour, the DC-actuator model includes:

- the control unit,
- the electrical adjustment unit (DC-motor)
- the gear.

The brake pedal system consists of:

- the joint system,
- the position detection unit
- the reset mechanism.

The basic model structure has been automatically derived from the CAD design including its kinematics via direct coupling and extended by modelling the friction behaviour, foot force, return springs and contact surfaces.

For the development of the models, including models for contact surfaces, the modeling language *Modelica* and the simulator *Dymola* [1] [2] have been applied.

1 Model of the DC-Actuator

The increasing utilization of mechatronic components in the automotive industry can be exemplarily shown on headlamps. While previously simple light sources in combination with reflectors provided a static illumination of the street, nowadays complex mechatronic headlamp systems provide a variety of functions (such as dynamic cornering light, automatic levelling, and many more) depending on the driving conditions. Since such mechatronic systems are getting more and more complex, a holistic development approach is necessary, which can be supported by accompanying modeling. Regarding the example of the DC-actuator, the development and application of a system model aims at increasing system understanding and efficiency at the same time, in particular for the analysis of critical operating conditions, such as blocking of the motor.

The DC-actuator is a subsystem of the mechatronic system headlamp and consists of the control unit, the electrical adjustment unit (motor), the gear as well as the drive shaft.

FiguFigure 1 exemplarily shows a DC-actuator of a mechatronic headlamp module.

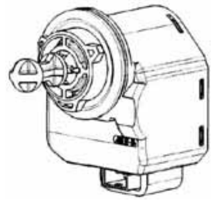


Figure 1: DC-actuator of a mechatronic headlamp modul.

In the following sections, these individual model elements are described.

1.1 Control Unit of the DC-Actuator

The schematic diagram of the control unit of the DC-actuator is shown in Figure 2 and essentially consists of two half bridges and a logic module to control them.

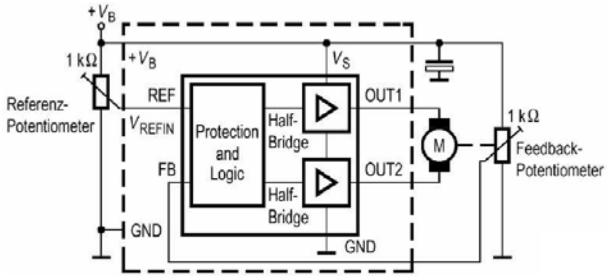


Figure 2: Schematic diagram of the DC-actuator.

The corresponding model of the control unit is shown in **Figu**Figure 3. It is modeled in a hierarchical manner, i.e. the half bridges shown in Figure 2 are part of the IC TLE 4209G (see Figure 3).

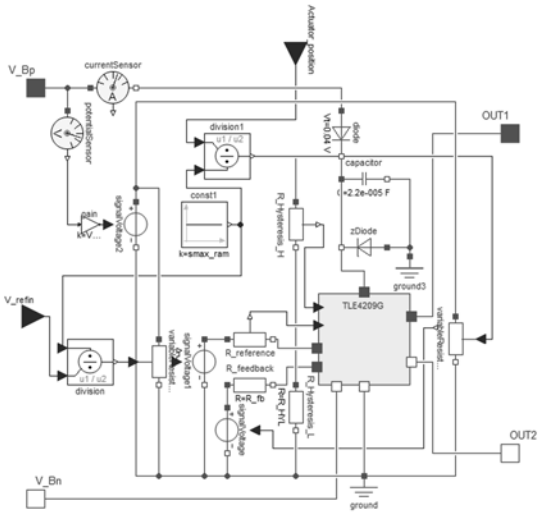


Figure 3: Control unit model of the DC-actuator.

1.2 Motor Model of the DC-Actuator

The motor model of the DC-actuator represents the electrical, mechanical and thermal behaviour. Regarding the latter one, on the one hand, the environment temperature and on the other hand the self heating during operation are considered. The Dymola/Modelica motor model is shown in Figure 4.

The accurate prediction of the thermal behaviour requires the description of the commutation characteristics. The motor consists of three windings, which are powered by a commutator. In case of blocking (malfunction), the motor must withstand a continuous current feed over a long period of time. Dependent on the type of blocking, the temperature may quickly increase to over 200°C due to self heating, so that the critical motor temperature is exceeded, which in turn leads to failure of the actuator.

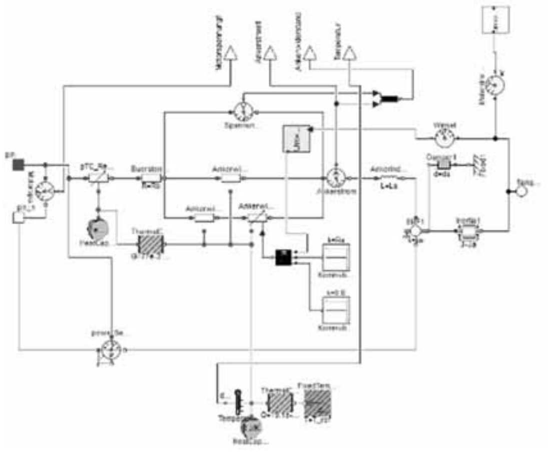


Figure 4: Motor model of the DC-actuator.

The analysis aims at the prediction of the temperature behaviour of different motors in case of a blocking. A distinction is made between two blocking cases, i.e. 1/3 and 2/3 blocking. In case of the 1/3 blocking, both brushes are each on one vane of the commutator. In case of the 2/3 blocking, one of the two brushes is between two vanes, bridging them. This results in two states of armature resistance in case of blocking as shown in Figure 5.

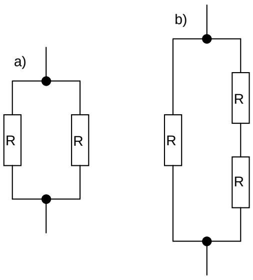


Figure 5: a) 2/3 blocking;
b) 1/3 blocking.

For a 2/3 blocking, the total armature resistance amount to R_{tot}=1/2 R and for 1/3 blocking to R_{tot}=2/3 R. Both cases can be adjusted separately in the model, in order to evaluate the related temperature behaviour in case of blocking explicitely.

1.3 Gear Model of the DC-Actuator

The gear model describes the behaviour of a worm gear, as well as the stops limiting both directions of the actuator. In addition, the friction characteristics such as the Coulomb and Stribeck forces are modeled. Summarizing, this model includes parameters for the gear ratio, both stops and friction. The gear model is shown in Figure 6.

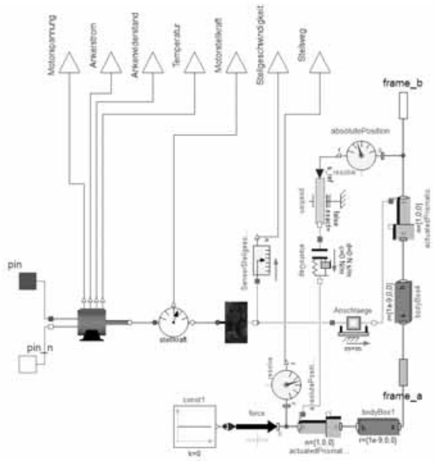


Figure 6: Gear model of the DC-actuator (including DC-motor).

1.4 Model Validation

The validation of the model plays a major role to ensure that the model is applicable without restrictions for all investigations along the development process.

Parameterization of the model can be done via specification sheets or measurements. With regard to motors, specification sheets are often not sufficient to define all necessary parameters. Hence, measurements need to be performed, which are usually extensive in terms of costs and time. In this case, relevant motor parameters have been evaluated from specification sheet data, while temperature characteristics have been determined by means of measurements in a climate chamber. These measurements have been performed on three operating conditions, i.e. rotary motion, 1/3 blocking and 2/3 blocking. Following from that, the necessary model parameters have been determined and implemented in the model.

As a next step, the model is validated for several test circuits by means of comparison of simulation results and measurements, respectively. An exemplary test circuit is shown in Figure 7.

Figure 8 exemplarily shows the comparison of the predicted and measured armature current of the DC-motor.

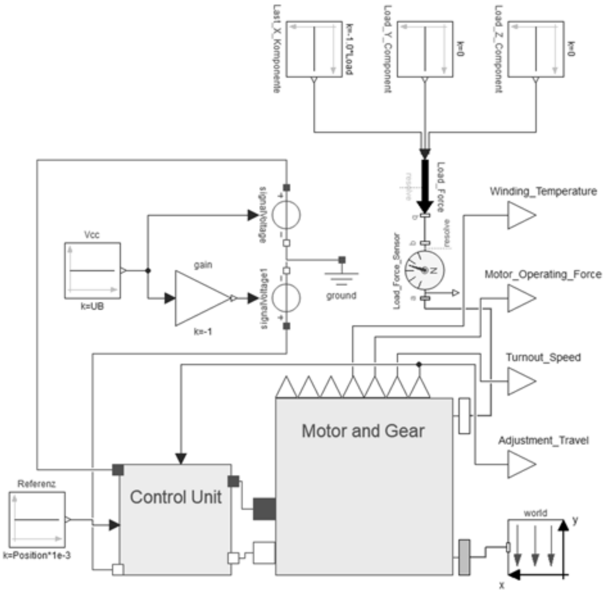


Figure 7: Exemplary test circuit for model validation.

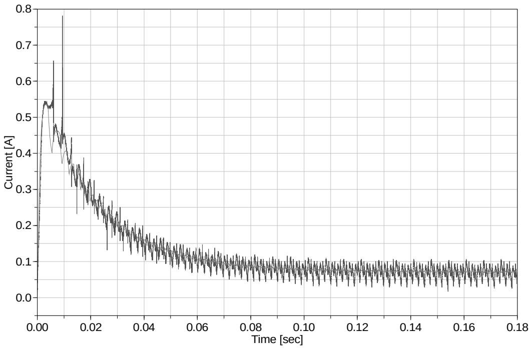


Figure 8: Armature current of the DC-motor model – Comparison between measurement (blue) and simulation (red).

The trend of the measured armature current of the DC-motor is satisfactorily reproduced by the simulation, which has been also confirmed for additional setups. Hence, the model is valid to be utilized in the context of product development.

Analysing the effort for prototype hardware development versus simulation reveals a ratio of 4:1, i.e. the effort for hardware development is four times higher than for simulation. This leads to the conclusion that by applying simulation methods, the development effort in the respective concept phases can be decreased significantly.

2 Model of a Brake Pedal System

Three different types of pedals can be distinguished in a vehicle, i.e. the brake pedal, the clutch pedal as well as the accelerator pedal.

Since these pedal systems are no longer mechanically connected to the systems to be operated (e.g. connection of the gas pedal to the motor via rope system), todays pedal systems are restricted to various requirements. Hence, beside the pedal arm, these pedal systems consist of friction elements, sensors and springs, in order to imitate closely the requested haptic feedback. Due to the rising complexity of interacting components, a system model has been developed aiming at a virtual prediction of the force-path behaviour, which includes the demand for an efficient contact modeling. An exemplary pedal system is shown in Figure 9.

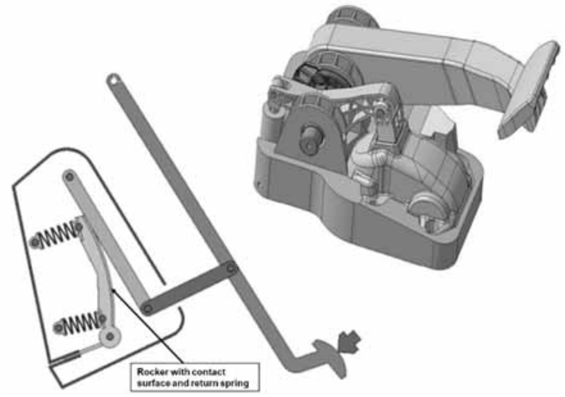


Figure 9: Example of a brake pedal system.

This brake pedal essentially consists of a four-bar linkage, a rocker including return spring, a cam shape defining the contact surface and a friction element. The contact surface forms the interface between the four-bar linkage and the rocker. The geometric shape of the contact surface contributes to the force-path behaviour, so that an appropriate shape design is necessary to satisfy the customer requirements. Most of the above-mentioned pedal elements can be modeled by using Modelica standard elements, while modeling the contact surface is a particular challenge. Hence, this is described in more detail. One possible contact modeling approach in Dymola is available via the Idealized Contact Library [3].

2.1 Description of a Contact Surface

In a system simulation model, bodies are usually assumed to be rigid. This assumption is valid if the kinematics are in the focus of investigation.

In this case it is sufficient to describe the body by the location of its center of gravity as well as its mass and inertia moments within the center of gravity. The body expansion is not considered. However, regarding contact phenomena, both, modeling elastic bodies as well as defining the surface are mandatory.

For the description of simple geometries, such as rectangles, cylinders or spheres, the *Idealized Contact Library* provides one “surface” block, respectively, that describes surface dimensions as well as its orientation in the coordinate system. The latest release additionally provides also blocks for the description of ellipsoids and other convex bodies.

The “surface” block represents a thin surface with no mass, which can be connected to a rigid body via a “frame” interface.

The calculation of the contact force is performed in a “contact” block. Necessary information of the contact surface is transfered via a “contact” interface. Beside defining the body-fixed coordinate system of the surface, geometric information of the contact surface and (in the latest release) the surface type are transferred.

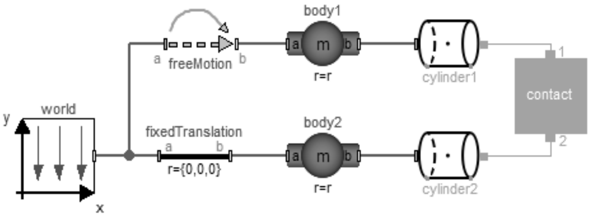


Figure 10: Schematics of a simple contact model with two cylinders.

Figure 10 shows an exemplary model for two cylinders. The blocks “cylinder1” and “cylinder2” describe the surfaces, connected via a “contact” block (orange). More complex geometries can be assembled by a parallel connection of individual contact surfaces. In this case, each contact pair must be connected via a “contact” block [3].

2.2 Surface Modeling via Analytical Functions

The approach to describe the cam shape via analytical functions results from the necessity of a variable curvature, which must be tangential and constant in curvature in the region of interest.

Third degree polynomial function. According to this approach, the geometry is described by one or more polynomial functions, which satisfy the necessary geometrical conditions at their intersection points.

A third degree polynomial function is defined by

$$f(z) = a_3(z - z_0)^3 + a_2(z - z_0)^2 + a_1(z - z_0) + a_0 \tag{1}$$

and is clearly defined by determining the coefficients a_i as well as the parameter z_0. If the coordinates of two points on the function and the first and second derivatives are known, the unknown variables can be determined by

$$a_0 = f(z_1) - a_1(z - z_0) - a_2(z - z_0)^2 - a_3(z - z_0)^3 \tag{2}$$

$$a_1 = \frac{f(z_2) - a_0 - a_2(z - z_0)^2 - a_3(z - z_0)^3}{(z - z_0)} \tag{3}$$

$$a_2 = \frac{f'(z_1) - a_1 - 3a_3(z - z_0)^2}{2(z - z_0)} \tag{4}$$

$$a_3 = \frac{f''(z_1) - 2a_2}{6(z - z_0)} \tag{5}$$

The parameter z_0 is defined as zero. If there is an intersection to another polynomial function, which is tangential and constant in curvature in the intersection point, the results of the first and second derivatives in this point are equal and thereby known, while only the coordinates of two points on the second function need to be specified. The model to describe a polynomial function is integrated in the Modelica model.

The contact surface model and the model components for determining the contact point on the ellipsoidal surface are partially shown inFigure 11.

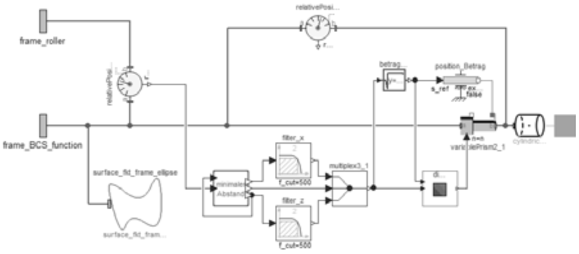


Figure 11: Schematics of the model for calculation of the contact point on the ellipsoidal surface.

To calculate the contact point on the cam shape, the Newton method is implemented.

If the polynomial function is defined within a limited range only, calculation of the minimal distance is performed only in this range by defining thresholds. This is realized by setting the position of the contact point equal to its respective threshold value in case the threshold is exceeded.

One single polynomial function is not sufficient to describe the cam shape and to achieve the demanded pedal behaviour along the entire path.

Hence, the model is assembled by several functions and successively approximated to the demanded behaviour.

Realization of the cam shape. First, the cam shape is described by a single polynomial function and adapted to the demanded behaviour of pedal force at the start of the displacement in accordance with the specified preload and implemented spring parameters of the rocker.

If the force deviates from the reference values at higher pedal angles, the position of the contact point for the respective pedal angle is determined and defined as intersection point for the successive function. Next, the second point of the successive function is specified, in order to achieve the demanded behaviour also for the second section. This is continued successively, until the cam shape is completely described.

The defined surface shape can be directly considered for the design of the geometry in CATIA [4]. This leads to a significant reduction of design steps, prototype development and measurements. In comparison to a decoupled approach, the effort of model creation has been reduced by a factor of 10.

2.3 Validation of the System Model

The overall system model basically consists of 4 main categories, as shown in Figure 12.

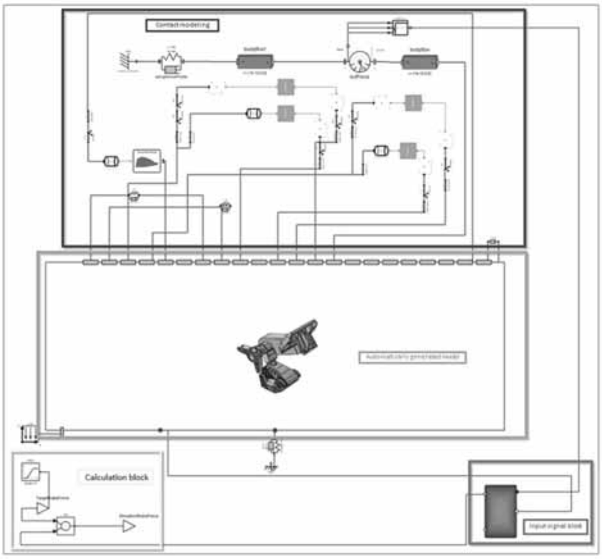


Figure 12: System model of the pedal system.

These are the pedal geometry, which is automatically translated from the CATIA design, a component for modeling the restoring force and the contact surface between the four-bar linkage and the rocker, as well as components for modeling the foot force and friction within the pedal.

The simulation result of this model is shown in Figure 13 in terms of the pedal force as a function of the pedal angle in comparison to the requested characteristics.

In case of return, the simulation result slightly deviates from the desired behaviour. However, the general trend can be reproduced. By increasing the friction value (which is not verified by measurements), the spread of the hysteresis can be increased.

By adjusting the cam shape, the behaviour can be accurately predicted. For a pedal angle of 32° the return force is smaller than the demanded minimum. Going towards a smaller pedal angle, the desired behaviour can be satisfactorily achieved.

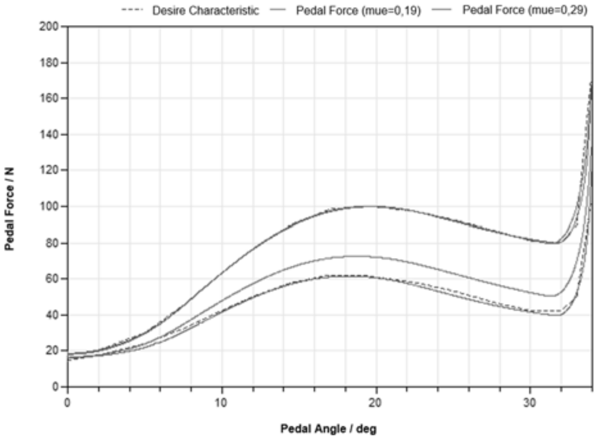


Figure 13: Pedal force as a function of the pedal angle for varying friction values.

3 Summary

The application of system simulation delivers an effective possibility to verify the system behaviour along all phases of product development, leading to a reduction of cost and time (e.g. by reduction of prototypes), and moreover, to gain more information about the system, which is not accessible in prototype measurements (e.g. force measurements in encapsulated systems).

The examples presented in this paper clearly demonstrate that the system behaviour including critical operating conditions can be predicted and analyzed via system simulation, leading to a more comprehensive understanding about the system. This is also supported by animations, which simplify interpretation and communication between different stakeholders.

Moreover, such an approach can support on detecting and avoiding failures, which often have tremendous consequences, in early phases of development. In addition, the simulation approach offers the possibility to exactly reproduce scenarios and results, so that the impact of changes can be analyzed and different variants and concepts can be compared more accurately.

Literature

[1] Fritzson P. Object-Oriented Modeling and Simulation with Modelica 2.1. John Wilex & Sons, New York 2004.

[2] DYMOLA: Multi-Engineering Modeling and Simulation.
www.dynasim.se, www.dymola.com.

[3] Oestersötebier F, Wang P, Trächtler A.
A Modelica Contact Library for Idealized Simulation of Independently Defined Contact Surfaces. Paderborn.

[4] Dassault Systemes: 3DExperience Platform
https://www.3ds.com/de/ueber-3ds/3dexperience-plattform/

Simulation of a Discharge Electrode Needle for Particle Charging in an Electrostatic Precipitator

Sebastian Beckers[1*], Julian Pawlik[2], Hikmet Eren[1], Adam Sanaf[1], Jürgen Kiel[1]

[1]FMDauto-Institut, Düsseldorf University of Applied Sciences, Münsterstr. 156, 40476 Düsseldorf, Germany;
*sebastian.beckers@hs-duesseldorf.de

[2]getAir GmbH, Krefelder Str. 670, 41066 Mönchengladbach, Germany

SNE 31(4), 2021, 233-238, DOI: 10.11128/sne.31.tn.10588
Received: 2021-03-15 (Selected ASIM SST 2020 Postconference Publication); Revised: 2021-08-19; Accepted: 2021-09-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. This paper describes a model approach for the simulation of a discharge electrode (DE) needle to charge particles using positive ions in an electrostatic precipitator. This includes the simulation of the electrostatic field, the space charge field of the ions and the flow field at the DE needle. The interactions of the fields, e.g. the reaction of the space charge on the electrostatic field or the electric wind are also considered in the model. To simplify and accelerate the simulation, a radial symmetry around the DE needle is partly assumed. The results of the simulation are validated by comparing the experimentally determined current-voltage characteristic with the simulated one, which show a satisfying correlation. Therefore, this model can be used as a basis for future particle flight simulation and further investigations.

Introduction

In residential applications, two-stage electrostatic precipitators (ESPs) are mainly used to separate harmful particles from the air. Particles entering the filter are first charged in the ioniser by an ion field based on a corona discharge and then separated in a subsequent filter stage by an electrostatic field (Coulomb's law) on the electrodes of the collector.

Although this filtering process is very efficient, it has the major disadvantage that it generates ozone during operation [1] [2].

Ozone can be harmful to human health when inhaled, therefore the WHO (Air Quality Guidelines Global Update 2005) sets a limit value of 50 ppb (parts per billion) for an average exposure of eight hours.

A very effective method to minimize the ozone concentration is to reduce the corona plasma region at the discharge electrode (DE) within the ioniser [3] [4], where the ozone production process takes place. Consequently, the development of DEs is geared towards ever smaller dimensions [5]. The shape and arrangement of these DEs can be very different for each application, which makes a generally valid analytical mathematical description difficult and therefore requires numerical modelling.

Experimental studies on particle separation and ozone generation have shown good results with particle charging by a DE needle [6]. Therefore, the modelling of this approach is described in the following.

1 Experimental Setup

The experimental setup used in this study consists of a stainless-steel DE needle with a radius of curvature of 55 μm at the tip and a round grid arranged at a distance of 50 mm as a ground electrode with a diameter of 85 mm, as shown in Figure 1. The DE needle is centered by a holder on the rotation axis and protrudes 4 mm from it.

Furthermore, the DE needle is raised to a positive voltage potential by a high-voltage source of the company FUG (HCP35-20000) and the grid is connected to an electrical grounding. By using this configuration, it is possible to set and measure voltages as well as currents. Thus, the voltage-current characteric of the DE needle can be analysed.

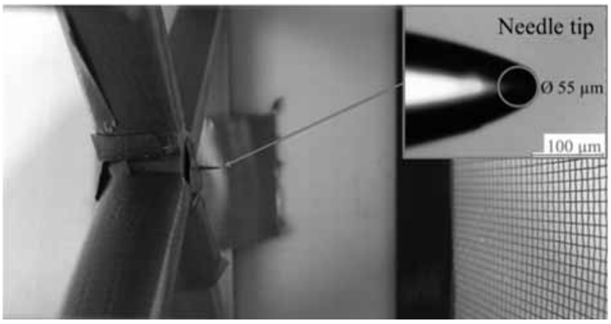


Figure 1: Test setup with the stainless-steel discharge electrode needle in the middle.

2 Model

2.1 Model Approach

In order to implement the simulation of a DE needle, not only the electrostatic field, but also the flow and space charge field must be modelled.

Furthermore, the interactions of the different fields are considered in the model. For example, the reaction of the space charge density to the electrostatic field as well as the electric wind as an impact of the electrohydrodynamic (EHD) effect are taken into consideration. Figure 2 gives an overview of the model approach.

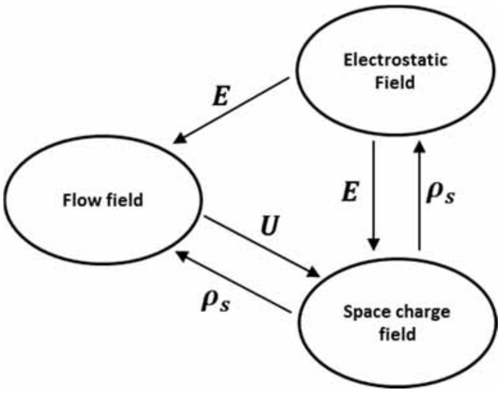


Figure 2: Overview of the simulated fields and their interactions.

To simplify the simulation, the geometry of the DE needle is modelled as a simple composition of a truncated cone with an outer diameter of 0.6 mm and a length of 4 mm and a semi-sphere with a diameter of 55 µm as needle tip, see Figure 3.

The holder of the DE needle is also simplified as a cylinder with a diameter of 11.4 mm and a length of 10 mm, as is the measuring chamber with a diameter of 85 mm and a length of 64 mm. The geometry of the grid at the exit of the measuring chamber is neglected.

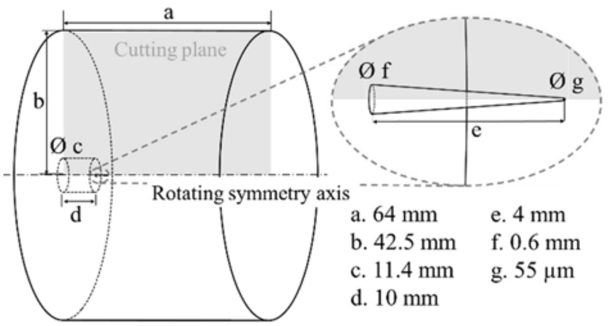


Figure 3: Simplified geometric model of the experiment.

In addition, a radial symmetry around the DE needle is assumed for the simulation of the space charge field and the flow field respectively, as shown in Figure 4.

The micromechanisms of the corona plasma region are not simulated but the resulting convection current of the space charges are. As a further simplification of the procedure, the corona plasma region is placed on the DE needle tip surface. A positive corona and thus a positive convection current (positive ions) are assumed.

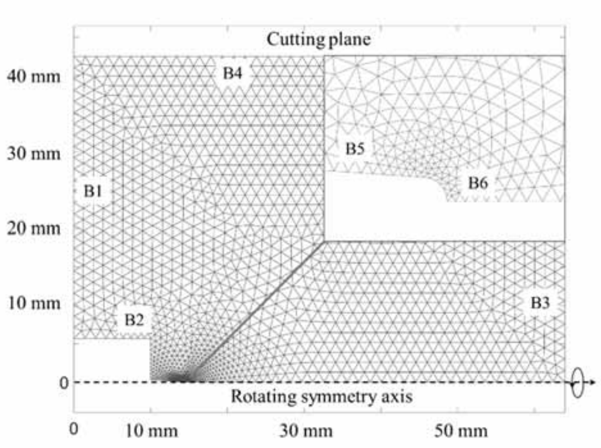


Figure 4: Geometry and meshing of the cutting plane with the boundary numbers.

2.2 Electrostatic Field

The electrostatic field can be described mathematically using the following equation of Poisson [7]:

$$\Delta V = -\frac{\rho_s}{\varepsilon_0 \varepsilon_r} \tag{1}$$

where V is the voltage potential (V), ρ_s the space charge density $(A\,s\,m^{-3})$, ε_0 the permittivity of the vacuum $(8.85 \cdot 10^{-12} A\,s\,V^{-1} m^{-1})$ and ε_r the relative permittivity (approx. 1 for air).

Due to the very fine DE electrode tip, which is strongly curved in contrast to the grid ground electrode, a very inhomogeneous electrostatic field is created.

In order to cope with this and take all effects into account, it is simulated in three dimensions. For the simulation itself, as well as the meshing, the Partial Differential Equation Toolbox (PDE-Tool) in MATLAB® is used. The total amount of tetrahedral cells used in the simulation mesh is 48811. The tool's integrated solver calculates the solution using an FEM algorithm, assuming the following boundary conditions, where the locations can be obtained from Figure (4).

Boundary	Description	Value
B5, B6	Potential at DE needle	$V_{SE} = V_0 + \Delta V$
B3	Potential at grid	$V_{grid} = 0$ (ground)

Table 1: Boundary conditions of the electrostatic field.

The voltage potential at the DE needle V_{SE} is composed of the breakdown voltage V_0 which corresponds to the initial voltage of the corona discharge, and a correction value ΔV which is described in detail in Chpt. (2.5).

2.3 Space Charge Field

The simulation of the space charge field is based on the formula of White [8], which describes the current density $\boldsymbol{J}$ $(A\,m^{-2})$ considering the convection and diffusion charge transport components.

$$\nabla \cdot \boldsymbol{J} = 0 \tag{2}$$

$$\boldsymbol{J} = (b_i \boldsymbol{E} + \boldsymbol{U})\rho_s - D_i \nabla \rho_s \tag{3}$$

The convection part of Eq. (3) shows the coupling to the electrostatic field $\boldsymbol{E}$ $(V\,m^{-1})$ and to the velocity field $\boldsymbol{U}$ $(m\,s^{-1})$. The quantities ρ_s and b_i represent the space charge density $(A\,s\,m^{-3})$ and the ion mobility $(m\,V^{-1}s^{-1})$ respectively. The latter is assumed as a constant with the value $b_i = 1.85 \cdot 10^{-4}\,m\,V^{-1}s^{-1}$ [9].

The diffusion part of Eq. (3) consists of the local gradient of the space charge $\nabla\rho_s$ $(A\,s\,m^{-4})$, and the ionic diffusion coefficient D_i $(m^2\,s^{-1})$, which can be estimated using the following formula [10]:

$$D_i = (b_i k T)/e \tag{4}$$

where k $(k = 1.38 \cdot 10^{-23}\,J\,K^{-1})$ is the Boltzmann's constant, e the elementary charge $(1.6 \cdot 10^{-19}\,As)$ and T the temperature (K).

Since the geometry can be assumed to be approximately rotationally symmetrical, the simulation area for modelling the space charge density can be reduced to a two-dimensional cutting plane, see Figure 3 and Figure 4.

As with the electrostatic field, the automatic mesher of the PDE-Tool is used for the grid generation of the two-dimensional solution area. The two-dimensional grid used has 4512 triangular cells.

The solution of Eq. (2) which describes the space charge transport is achieved by using the Finite Volume Method (FVM) in MATLAB®. In order to accomplish that, the solution area (Ω) is divided into many subareas (Ω_i) (finite volumes) and the current density at the interfaces is balanced:

$$\int_{\Omega_i} \nabla \cdot \left((b_i \boldsymbol{E} + \boldsymbol{U})\rho_s - D_i \nabla \rho_s \right) d\Omega_i = 0 \tag{5}$$

Due to the Gaussian integral theorem and the assumption that the values on the cell-face are uniform over the entire face, Eq. (5) can be brought into a discrete form [11]:

$$\sum_f \left[[(b_i \boldsymbol{E} + \boldsymbol{U}) \cdot \boldsymbol{n}]_f \rho_{sf} - \left(D_i \frac{\partial \rho_s}{\partial n} \right)_f \right] A_f = 0 \tag{6}$$

where the index f represents the face, $\boldsymbol{n}$ the normal vector and A_f the area of the face.

The convection term of Eq. (6) is calculated according to Long [12] using the second order Upwind Difference Method (2nd UDM). In this method, a Taylor series approach is used to project the respective space charge density onto the center of the intersection face (F), see also Figure 5.

$$\sum_f [(b_i \boldsymbol{E} + \boldsymbol{U}) \cdot \boldsymbol{n}]_f \, \rho_{sF,f} \, A_f \tag{7}$$

The projected space charge density $\rho_{sF,f}$ can be calculated using the 2nd UDM with the following case distinction:

$$\boldsymbol{\rho_{sF,f}} = \begin{Bmatrix} \rho_{s_C} + \nabla\rho_{s_C} \cdot \boldsymbol{d}_{CF} \; if \; \left((b_i \boldsymbol{E} + \boldsymbol{U}) \cdot \boldsymbol{n}\right)_{F,f} > 0 \\ \rho_{s_N} + \nabla\rho_{s_N} \cdot \boldsymbol{d}_{NF} \; if \; \left((b_i \boldsymbol{E} + \boldsymbol{U}) \cdot \boldsymbol{n}\right)_{F,f} < 0 \end{Bmatrix} \tag{8}$$

where $\nabla\rho_{s_C}$ is the local gradient of space charge densities of the cell and $\nabla\rho_{s_N}$ is the one of the neighbouring cell. In this equation, the vectors $\boldsymbol{d}_{CF}$ and $\boldsymbol{d}_{NF}$ represent the distance vectors between the centers of the particular cell (C and N) and the center point of the intersection face (F).

The diffusion term in Eq. (6) is implicitly calculated in this study using the space charge field.

$$\sum_f - \left(D_i \frac{\partial \rho_s}{\partial n} \right)_f A_f \tag{9}$$

Following the approach of Long [12], the gradient of space charge density in the diffusion term is determined by projected substitute points for the space charge density of the cell $\rho_{sC'}$ and the neighboring cell $\rho_{sN'}$ as well as the projected substitute point on the intersection face $\rho_{sF'}$. These three substitute points are determined by the following equations:

$$\rho_{sC'} = \rho_{sC} + \nabla\rho_{sC} \cdot \boldsymbol{d}_{CC'} \tag{10}$$

$$\rho_{sN'} = \rho_{sN} + \nabla\rho_{sN} \cdot \boldsymbol{d}_{NN'} \tag{11}$$

and

$$\rho_{sFC} = \rho_{sC} + \nabla\rho_{sC} \cdot \boldsymbol{d}_{CF} \tag{12}$$

$$\rho_{sFN} = \rho_{sN} + \nabla\rho_{sN} \cdot \boldsymbol{d}_{NF} \tag{13}$$

$$\rho_{sF} = \frac{\rho_{sFC} + \rho_{sFN}}{2} \tag{14}$$

where $\nabla\rho_s$ is the local gradient of the respective cell and $\boldsymbol{d}$ is the respective difference vector between the corresponding points in the indices. ρ_{sFC} and ρ_{sFN} represent space charge density values projected from the centers of the cell (C) and the neighbouring cell (N) to the center point of the intersection face (F).

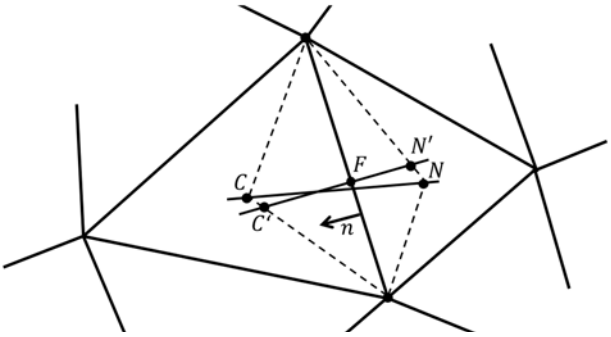


Figure 5: Demonstration of the gradient calculation between adjacent mesh cells.

Based on these three substitute points of the space charge density (Eq. (12) to Eq. (14)), the mean slope can be determined by linear interpolation. The mean slope then corresponds to the gradient at the intersection face of both cells.

The convection and diffusion term of space charge transport shown in Eq. (7) and Eq. (9) can then be expressed in a simple form:

$$\sum_f a_{conv_f}\, \rho_{s,f} = \sum_f \left(b_{conv_f} - b_{diff_f}\right) \tag{15}$$

where

$$a_{conv_f} = [(b_i\boldsymbol{E} + \boldsymbol{U})\cdot\boldsymbol{n}]_f A_f \tag{16}$$

$$b_{conv_f} = -a_{conv_f}\left(\nabla\rho_{s_i}\cdot\boldsymbol{d}_{ij}\right)_f \tag{17}$$

$$b_{diff_f} = -\left(D_i\frac{\partial\rho_s}{\partial n}\right)_f A_f \tag{18}$$

The quantities $\nabla\rho_{s_i}$ and $\boldsymbol{d}_{ij}$ refer to the case distinction of the 2nd UDM in Eq. (8).

If Eq. (15) is applied to all cells in the solution area it yields a linear system of equations in the form:

$$\boldsymbol{A}\cdot\boldsymbol{\rho_s} = \boldsymbol{B} \tag{19}$$

which is then solved using the method of least squares (*lqslin* function) in MATLAB®.

The boundary conditions used for the simulation of the space charge field are given below.

Boundary	Description	Value
B6	Current density input	$\boldsymbol{J}\cdot\boldsymbol{n} = \frac{I_0}{A_{out}}$
B1, B3	Current density output	$\boldsymbol{J}\cdot\boldsymbol{n} = [(b_i\boldsymbol{E} + \boldsymbol{U})\cdot\boldsymbol{n}\,]\rho_s$
B2, B4, B5	Wall	$\boldsymbol{J}\cdot\boldsymbol{n} = 0$

Table 2: Boundary conditions of the space charge field.

The current value I_0 in the boundary condition of the input current density represents an input parameter of the model and must be distributed over the entire outlet surface A_{out} of the DE needle tip.

For the boundary condition of the output current density, only the convection component is taken into account, due to the assumption that the change of space charge density near the surface of the output is neglectable.

2.4 Flow Field

The flow field in an electrostatic precipitator which can be modelled according to [13 - 15] by the time-averaged Navier-Stokes equation for incompressible fluids with the standard k-ϵ turbulence model [16]:

$$\nabla\cdot\boldsymbol{U} = 0 \tag{20}$$

$$\rho_F(\boldsymbol{U}\cdot\nabla)\boldsymbol{U} - (\mu + \mu_T)\Delta\boldsymbol{U} = -\nabla\mathrm{p} + \rho_F\mathbf{g} + \boldsymbol{F_{EHD}} \tag{21}$$

where ρ_F is the fluid densitiy ($kg\ m^{-3}$), μ the laminar viscosity ($kg\ m^{-1}\ s^{-1}$), μ_T the turbulent viscosity of the κ-ϵ turbulence model ($kg\ m^{-1}\ s^{-1}$), p the fluid pressure (Pa) and $\mathbf{g}$ the body accelerations acting on the continuum ($m\ s^{-2}$). $\boldsymbol{F_{EHD}}$ represents the electrical body force term of the EHD-effect ($N\ m^{-3}$), which appears in form of electric wind in the flow field and is determined as follows:

$$\boldsymbol{F_{EHD}} = \boldsymbol{E}\, \rho_s \tag{22}$$

The flow field is simulated with the flow simulation software OpenFOAM® based on the Finite Volume Method. A program interface between MATLAB® and OpenFOAM® was developed to exchange input and output parameters in form of geometry and mesh data, boundary and start conditions, material and substance values as well as field data.

Geometry and mesh data are created in MATLAB® by the PDE Tool's automatic mesher and the finished mesh is transferred to the OpenFOAM® software. As a simplification, a two-dimensional geometry with a radial symmetry is assumed, see Figure 4. The number of triangular cells of the mesh is also 4512.

For the implementation of the EHD effect the *simpleFoam* solver was modified. Flow simulations of stationary and incompressible Newtonian and turbulent fluids can be performed with the *simpleFoam* solver (OpenFOAM® User Guide), in which the standard k-ϵ model was used as turbulence model. The modified solver considers the influence of the electric wind as a body source term in the Navier-Stokes equation based on the current fields of $\boldsymbol{E}$ and ρ_S, also see Eq. (21) and Eq. (22). As $\boldsymbol{F_{EHD}}$ is a spatial volume force in the flow field, it must be projected onto a two-dimensional geometry.

The resulting simulated flow field is then returned to other MATLAB® models via the programmed interface.

The following boundary conditions are used for the flow and pressure field of the model:

Boundary	Description	Value
B1	inlet flow	$\boldsymbol{U} = \boldsymbol{U}_{in}$ $\nabla p = 0$
B3	outlet flow	$\nabla \boldsymbol{U} = 0$ $p = p_0$
B2, B4, B5	Wall	$\boldsymbol{U} = 0$ (no slip) $\nabla p = 0$

Table 3: Boundary conditions of the flow and pressure field.

In Tab. (3), U_{in} corresponds to the inlet flow ($m\ s^{-1}$) and p_0 to the ambient pressure (Pa).

2.5 Calculation Sequence

The calculation sequence shown in Figure 6 starts with an input current I_0 and an input start voltage V_0 of the DE needle. This input voltage can be determined experimentally or estimated by using empirical formulas (e.g. according to Peek [8]). Next, the three model fields are calculated until convergence is achieved. After convergence, a correction value ΔV is determined for the voltage potential of the DE needle via the resulting electrostatic field. Afterwards, the potential V_0 is adjusted accordingly with $V_0 = V_0 + \Delta V$ and the calculation of the fields is started again.

The calculation sequence ends as soon as the correction value ΔV runs towards zero and no voltage potential change can be observed anymore.

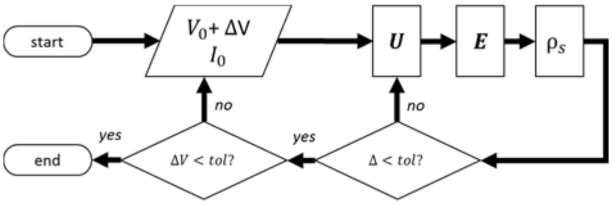


Figure 6: Schematic representation of the calculation sequence.

3 Model Validation

The model is validated by comparing the experimentally determined and simulated voltage-current characteristics of the DE needle, which are shown in Figure 7.

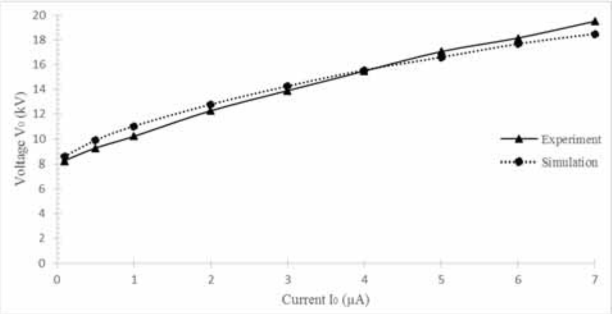


Figure 7: Voltage-current characteristics of the DE needle.

The breakdown voltage of the DE needle was approximately $V_0 = 8\ kV$ in the experiment. This value was used as the starting value for the simulation according to the calculation sequence.

As can be seen in Figure 7, the curve progression of the simulation largely complies well with the experimentally determined curve progression, whereby the simulation slightly exceeds the voltage potential below 4 µA and slightly falls below it above 4 µA.

These deviations can probably be explained by inaccuracies in geometric modelling (e.g. the shape of the DE needle) and by the model simplifications of rotational symmetry that were applied.

4 Conclusion and Outlook

Due to the various simplifications in geometry and symmetry assumptions, an efficient DE needle model could be developed, which provides fast and sufficiently good results with regard to validation.

Based on this, the particle flight can then be modelled using the Discrete Element Method (DEM) to analyse the particle separation behaviour by the DE needle in the electrostatic precipitator.

If the accuracy of the model is to be improved, a three-dimensional model approach to the space charge density as well as the flow field should be used. It would also be advisable to use a more precise geometric model. However, these improvements would be accompanied by an increased computing time.

Based on the model presented, the modelling of ozone production at the DE needle would also be an interesting topic for future studies.

References

[1] Caste GSP. *Electrostatic Precipitation in Electrified Media and Positive Corona Ozone Generation in the Design of High Efficiency Air Cleaners* [dissertation]. [Faculty of Engineering Science, CAN]. University of Western Ontario; 1969.

[2] Boelter KJ, Davidson JH. Ozone Generation by Indoor, Electrostatic Air Cleaners. *Aerosol Sci. and Technology.* 1997; 27(6), 689-708. doi: 10.1080/0278682970 8965505.

[3] Chen J, Davidson JH. Ozone production in the positive DC corona discharge: Model and comparison to experiments. *Plasma Chem. and Plasma Proc.* 2002; 22 (4), 495–522. doi: 10.1023/A:10231315412208

[4] Chen J, Davidson JH. Ozone production in the negative DC corona: the dependence of discharge polarity. *Plasma Chem. and Plasma Proc.* 2003; 23 (3), 501–518. doi: 10.1023/A:1022468803203

[5] Bo Z, Yu K, Lu G, Mao S, Chen J, Fan F-G. Nanoscale discharge electrode for minimizing ozone emission from indoor corona devices. *Env. science & technology.* 2010; 44 (16), 6337–6342. doi: 10.1021 /es903917f.

[6] Hak-Joon K, Myungjoon K, Bangwoo H, Chang GW, Ayyoub Z, Noureddine Z, Yong-Jin K. Fine particle removal by a two-stage electrostatic precipitator with multiple ion-injection-type prechargers. *J. of Aerosol Sciene.* 2019; 130, 61–75. doi: 10.1016/j.jaerosci.2019.01.004.

[7] Leuchtmann P. *Einführung in die elektrische Feldtheorie.* 1. Auflage. Freising: Pearson Studium; 2007. 602 p.

[8] White HJ. *Entstaubung industr. Gase mit Elektrofiltern.* Leipzig: D. Verlag für Grundstoffindustrie; 1969. 336 p.

[9] McDonald JR, Smith WB, Spencer, Herbert W, Sparks, Leslie E. A mathematical model for calculating electrical conditions in wire-duct electrostatic precipitation devices. *J. of Appl Phys.* 1977; 48 (6), 2231–2243. doi: 10.1063/1.324034.

[10] Abdel-Salam M, Nakano M, Mizuno A. Corona-induced pressures, potentials, fields and currents in electrostatic precipitator configurations. *J. Phys. D: Appl. Phys.* 2007; 40 (7), 1919-1926. doi: 10.1088/0022-3727/40/7/014.

[11] Patankar SV, Minkowycz WJ, Sparrow EM. *Series in computational methods in mechanics and thermal sciences.* New York: McGraw-Hill Book Company; 1980, 197 p.

[12] Long Z, Yao Q, Song Q, Li S. A second-order accurate finite volume method for the computation of electrical conditions inside a wire-plate electrostatic precipitator on unstructured meshes. *Journal of Electrostatics.* 2009; 67 (4), 597–604. doi: 10.1016/j.elstat.2008.12.006.

[13] Chun YN, Chang J-S, Berezin AA, Mizeraczyk J. Numerical modeling of near corona wire electro-hydrodynamic flow in a wire-plate electrostatic precipitator. *IEEE Trans. Dielect. Electr. Insul.* 2007; 14 (1), 119–124. doi: 10.1109/TDEI.2007.302879.

[14] Long Z, Yao Q. Evaluation of various particle charging models for simulating particle dynamics in electrostatic precipitators. *J. of Aer. Sci.* 2010: 41 (7), 702–718. doi: 10.1016/j.jaerosci.2010.04.005.

[15] Chang JS, Dekowski J, Podlinski J, Brocilo D, Urashima K, Mizeraczyk J. Electrohydrodynamic gas flow regime map in a wire-plate electrostatic precipitator. *Fourtieth IAS Ann..Meet. C. Record of the 2005 Ind. Appl. Conf.* 2005; 4, 2597-2600. doi: 10.1109/IAS.2005.1518826.

[16] Launder BE, Spalding DB. The numerical computation of turbulent flows. *C. Meth.in Appl. Me. and En.* 1974; 3(2), 269–289. doi: 10.1016/0045-7825(74)90029-2.

ARGESIM Benchmark C7 'Constrained Pendulum' - Solution in MATLAB Environment and Extensions with Linear Approach, Symbolic Approach, Sensitivity, and Integration into TU Vienna's MMT E-Learning Environment

Marko Grujic[2], Jakob Haupt[2], Ypti Hossain[2], Lorenz Klimon[2], Paul Setinek[1], Felix Breitenecker[1*]

[1]Institute of Analysis and Scientific Computing, [2]Inst. of Mechanics and Mechatronics, TU Wien,
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria; **felix.breitenecker@tuwien.ac.at*

SNE 31(4), 2021, 239-254, DOI: 10.11128/sne.31.bne07.10589
Received: 2020-12-10; Revised: 2021-07-05;
Revised: 2021-09-10; Accepted: 2021-09-15
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. The ARGESIM Benchmark 'C7 Constrained Pendulum' is based on the dynamics of a pendulum which hits a pin: hit and release of the pin is a state event, which has to be managed properly. This Educational Benchmark Note, a detailed Benchmark Study, presents four issues for this benchmark. First, the study describes classical approaches, implementation and results for the requested benchmark tasks in MATLAB, Simulink and Stateflow, putting emphasis on the quality of event finding. Second, the study investigates in detail the possibilities of the linear pendulum model for event management: ODE approach, state space approach with exponential matrix, approach with analytical solution, and approach with symbolic computation. Third, the study sketches sensitivity analysis for the model, and fourth, the study presents the implementation of the model into TU Vienna's MMT E-Learning Server for education in modelling and simulation (MMT – Mathematics – Modelling – Tools).

Introduction - Modelling

ARGESIM Benchmark 'C7 Constrained Pendulum' is based on the dynamics of a pendulum which hits a pin: hit and release of the pin is a state event, which has to be managed properly ([1]). At *Hit* and *Release*, the pendulum changes its pivot point (Figure 1), so that the dynamics is composed of the movement of a 'long' pendulum and of a 'short' pendulum. Both movements are described by the classical nonlinear pendulum equation:

$$m \cdot l \cdot \ddot{\varphi} = -m \cdot g \cdot \sin\varphi(t) - d \cdot l \cdot \dot{\varphi}(t)$$

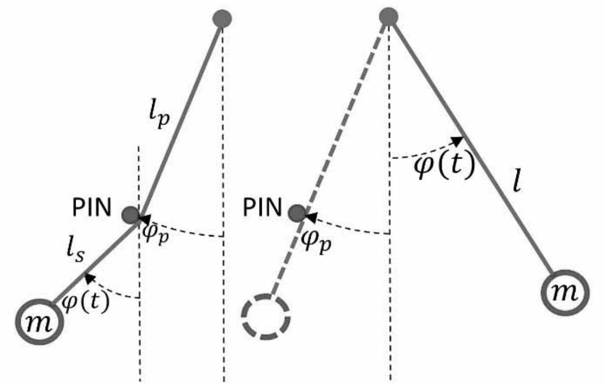


Figure 1: Sketch of the constrained pendulum.

For small angles, also the linear pendulum model is sufficient accurate. The classical linearization around the operating point $\varphi_S = 0$ is independent from angular velocity $\dot{\varphi}_S$ as the model works with linear damping:

$$m \cdot l \cdot \ddot{\varphi} = -m \cdot g \cdot \varphi - d \cdot l \cdot \dot{\varphi}(t)$$

The parameters pendulum length l, short pendulum length l_s, damping factor d, point mass m, angular pin position φ_{pin}, pin distance from pivot l_p, and initial values characterize the system.

The system is a so-called structural dynamic system ([2]), as caused by state events (*Hit* or *Release*) the dynamics change – in this case only a parameter, the pendulum length changes, and the equations remain unchanged.

The events *Hit* and *Release* obey a simple *Event Function* $e(t)$, whose zeros t_e determine the time instants of the events:

$$e(t) = \varphi(t) - \varphi_{pin} = 0 \;\rightarrow\; e(\varphi(t)) = \varphi - \varphi_{pin} \overset{!}{\underset{\pm}{=}} 0$$

Here the first equation is the mathematical description, the second the algorithmic: a zero search algorithm with either positive, negative, or both-sided crossing of zero.

For a dynamic system $\dot{\vec{x}}(t) = \vec{f}(\vec{x},\vec{p},\vec{u},t)$ with event function $e(\vec{x},\vec{p},t)$, event handling generally requires the following steps within an ODE solver's integration step from t_i to t_{i+1}

- *Event Detection* by sign of event function:
  $$sign(e(t_i)) \neq sign(e(t_{i+1}))$$
- *Event Localisation* and stop of ODE solving by zero search of $e(\vec{x}(t)) = 0$ at $[t_i,t_{i+1}] \longrightarrow t_e$
- *Event Action* at t_e
- *Re-Initialisation* and re-start of ODE solving

Event actions may be simple to complex:

- *Output Event*: no event action, only time output
- *Parameter Change Event*: $p \rightarrow p^*$
- *Input Change Event*: synchronisation of input jumps with stepsize
- *State Change Event*: $x(t_e) \rightarrow x^*(t_e)$
- *Derivative Change Event*: $f(x,t) \rightarrow f^*(x,t)$
- *Model Change Event*: $\dot{\vec{x}} = \vec{f}(\vec{x}) \rightarrow \dot{\vec{z}} = \vec{g}(\vec{z})$

The constrained pendulum system with events *Hit* and *Release* involves *Parameter Change Events* and *State Change Events*. At event $e(t_e) = \varphi_{pin} - \varphi(t_e) = 0$,

- the pendulum length changes: $l \rightarrow l_s$ or $l_s \rightarrow l$,
- and due to conservation of momentum, the angular velocity changes discontinuously:
  $\dot{\varphi}(t_e) \rightarrow \dot{\varphi}\frac{l}{l_s}(t_e)$ or $\dot{\varphi}(t_e) \rightarrow \dot{\varphi}\frac{l_s}{l}(t_e)$.

Indeed it is strange, that the angular velocity, a state variable, changes discontinuously – this cannot happen in reality, it is result of simplification in modelling. This drawback can be eliminated by a simple transformation of the state, using instead of the angular velocity $\dot{\varphi}(t)$ the tangential velocity $v(t) = l \cdot \dot{\varphi}(t)$, which does not change in case of *Hit* or *Release*:

$$\dot{\varphi} = \frac{1}{l} \cdot v \qquad \dot{v} = g \cdot \sin\varphi - \frac{d}{m} \cdot v$$

1 MATLAB Model Approaches

MATLAB's ODE-solvers generally need a state space description of the model with coupled first-order differential equations, best choice for the constrained pendulum is

$$x_1 = \varphi \quad x_2 = v$$

$$\dot{x}_1 = \frac{1}{l} \cdot x_2 \quad \dot{x}_2 = -g \cdot \sin(x_1) - \frac{d}{m} \cdot x_2,$$

resulting in nonlinear state space description:

$$\dot{\vec{x}}(t) = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{l} \cdot x_2 \\ -g \cdot \sin(x_1) - \frac{d}{m} \cdot x_2 \end{pmatrix} = \vec{f}(\vec{x},t)$$

The classically linearized model – needed later - is

$$\dot{y}_1 = \frac{y_2}{l} \qquad \dot{y}_2 = -g \cdot \mathrm{y}_1 - \frac{d}{m} \cdot y_2,$$

and reformulated as LTI state space system:

$$\dot{\vec{x}} = A \cdot \vec{x} + B \cdot \vec{u} \qquad \vec{y} = C \cdot \vec{x} + D \cdot \vec{u}$$

$$A = \begin{pmatrix} 1 & \frac{1}{l} \\ -g & -\frac{d}{m} \end{pmatrix} \; B = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \; C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \; D = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

1.1 TASK A: MATLAB Nonlinear Model with Event Handling

The first task of the benchmark is to solve the pendulum problem with an ODE-solver and to find pin touch and release with events functions.

MATLAB's ODE solvers provide event detection, but no event action handling. For events, additionally to the model derivative function $\vec{f}(\vec{x},t)$ the event function $e(\vec{x},t)$ can be provided.

This solution works with the classical Runge-Kutta ODE45 solver, with stepsize control. Before calling the solver, options define accuracy for step size control - 'Rel-Tol', 1e-4, – and event specification - 'Event', @hitrelease. The solver call needs as inputs the derivative function - @pend_func - and simulation interval, initial values, and the reference to further options:

```
options=odeset('RelTol',1e-5,'Event',@hitrelease)
ode45(@pend_func, [tstart, tend], xstart, options)
```

The ODE solver can detect an event, and he can localize an event by iteration within the integration interval $[t_i,t_{i+1}] \longrightarrow t_e$ (using the *Regula Falsi* method, a combination of bisection method and secant method), resulting in a reduced integration interval $[t_i, t_e = t_{i+1}]$. There is no possibility to force *Event Actions* at event time t_e (except *Output Events*). Now the solver either re-starts the integration at $[t_e = t_{i+1}, t_{i+2}]$ and continues, or he terminates the ODE solving at t_e state with state $\vec{x}(t_e)$.

The second option, the termination at the event, is basis of the implementation for the implementation of the constrained pendulum model: a loop switches between solving the 'long' pendulum model and the 'short' pendulum model, each terminated by the *Hit* or *Release* events.

The implementation itself is quite straightforward with a while-loop, which stops if the time reaches the defined simulation end time (10 sec).

Inside the loop an *if-elseif-else* clause decides whether the long or the short pendulum system is used and appended the overall solution. The decision logic works for arbitrary initial values and pin positions, but becomes more complex for a possible special case: in case the *Hit* or *Release* event is around at pin position (within a certain numerical accuracy, the tangential velocity at event time must decide about further model selection. The following code snippet shows details of this implementation, which is a classical hybrid decomposition of the constrained pendulum model into a controlled sequence of ‘long’ pendulum model and ‘short’ pendulum model.

```
if y_start(1) > phi_p % calculating with long pendulum
    sol = ode45(dydt1, [tstart, tend], y_start, options);
    t = [t, sol.x]; y = [y, sol.y]; t_events = [t_events, sol.xe];
elseif y_start(1) < phi_p    % calculating with short pendulum
          sol = ode45(dydt2, [tstart, tend], y_start, options);
          t = [t, sol.x]; y = [y, sol.y]; t_events = [t_events, sol.xe];
else
if y_start(1) > phi_p % calculating with long pendulum
    sol = ode45(dydt1, [tstart, tend], y_start, options);
    t = [t, sol.x]; y = [y, sol.y]; t_events = [t_events, sol.xe];
elseif y_start(1) < phi_p    % calculating with short pendulum
      sol = ode45(dydt2, [tstart, tend], y_start, options);
      t = [t, sol.x]; y = [y, sol.y]; t_events = [t_events, sol.xe];
else
if y_start(2) < 0    % calculate with short pendulum
    sol = ode45(dydt2, [tstart, tend], y_start, options);
    t = [t, sol.x]; y = [y, sol.y]; t_events = [t_events, sol.xe];
else    % calculate with long pendulum
    sol = ode45(dydt1, [tstart, tend], y_start, options);
    t = [t, sol.x]; y = [y, sol.y]; t_events = [t_events, sol.xe];
end; end
```

The model derivative functions can be defined as inline function by

```
dydt1 = @(t,y)[y(2)/l; -g*sin(y(1))-d/m*y(2)];
dydt2 = @(t,y)[y(2)/ls; -g*sin(y(1))-d/m*y(2)];
```

The algorithmic event function has as parameters the event function ‘value’ itself, the stopping flag ‘is terminal=1’ to stop ODE solving at the event, and ‘direction=0’ to detect *Hit* and *Release*:

```
function [value,isterminal,direction] = hitrelease(~,y)
    value = y(1)-phi_p;
    isterminal = 1; direction = 0;
end
```

Figure 2 shows the results for the ‘standard’ initial values $\varphi_0 = \pi/6$, $\dot{\varphi}_0 = 0$, $\varphi_{pin} = -\pi/12$. Event times are:

0.7035 1.1518 2.5904 2.9905 4.5427 4.8675 6.6487 6.7204

Obviously the fourth contact (7th event)results in a very short window for the ‘short’ pendulum, and may cause ‘event vanished’ for too big stepsizes.

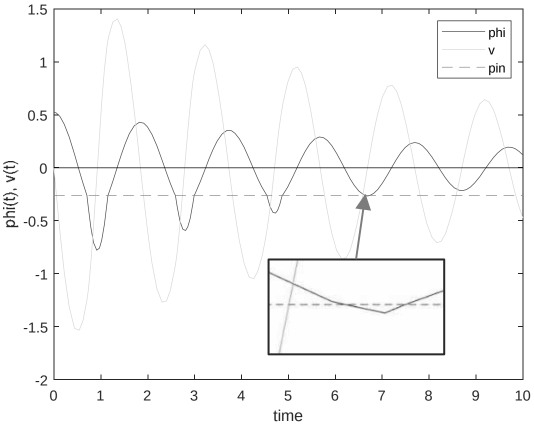


Figure 2: ODE45 solutions for $\varphi(t)$ and $v(t)$ for ‘standard’ initial values with detail for last two events rel. tolerance 1e-4, max. stepsize 0.15.

Important for the accuracy of event finding is the stepsize control of the ODE solver. ODE45 estimates the local error by the difference of a 4th order step and a 5th from t_i to $t_{i+1} = t_i + h$: exceeding the given relative tolerance, the stepsize decreases to h^-, $t_{i+1} = t_i + h^-$, a too big undercut increases the stepsize to h^+, $t_{i+1} = t_i + h^+$.

After the choice of a proper stepsize the event finding starts - with an accuracy depending on ODE solver accuracy and general accuracy eps. A small stepsize brackets a small interval for fast event finding, but may result in slow ODE solving. A too big stepsize may cause problems: events may vanish, as in this case with the forth pin contact: here the bracketed interval for event finding may be too large, so that both events are within the window and will therefore not be detected.

1.2 TASK B: MATLAB Linear Model – ODE Solver with Event Handling

Task is to compare the nonlinear model with the linear model. For the linear model also the event finding features of the ODE solver can be used, so that the implementation simply replaces the nonlinear model from Task A with the linear one:

For graphical comparison, both linear and nonlinear solutions are plotted into one graphic window. Figure 3 displays both results for the ‘standard’ initial values, showing only slight differences in the event times. Event times are summarizes in Table 3, Section 5, for better comparison.

Of course the implementation works also for the ‘original’ smaller initial values foreseen for this task, $\varphi_0 = \pi/12$, $\dot{\varphi}_0 = 0$, $\varphi_{pin} = -\pi/24$, resulting in even smaller differences of nonlinear/linear event times.

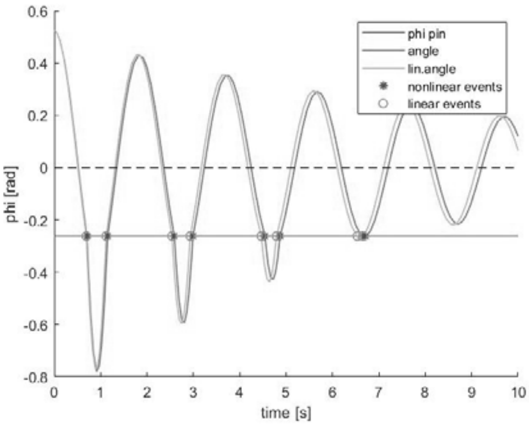


Figure 3: ODE45 solution for linear and nonlinear system in MATLAB with event finding (rel.tol 1e-5).

It is to be noted, that simulation of nonlinear and linear system results in different time bases, because of differences in the step size control. For a numerical comparison, e.g. difference of the angles φ(t) and $\varphi_{lin}(t)$, the time bases must be interpolated after the simulation. One could force the ODE solvers to a given (output) time base, but then problems with the event times occur.

For a real precise comparison of the full time courses, both models must run in parallel with state vector $(\varphi, v, \varphi_L, v_L)^T = (x_1, x_2, x_3, x_4)^T$, with an extended event control of the linear and of the nonlinear system using a vector event function:

$$\vec{e}(\varphi(t), \varphi_L(t)) = (\ \varphi(t) - \varphi_{pin} \quad \varphi_L(t) - \varphi_{pin})^T$$

The model becomes a joint model implemented as

```
dydt1 = @(t,y)[y(2)/lnlakt; -g*sin(y(1))-d/m*y(2)];
dydt2 = @(t,y)[y(2)/lnlakt; -g*sin(y(1))-d/m*y(2)];
dydt3 = @(t,y)[y(4)/llakt; -g*y(3)-d/m*y(4)];
dydt2 = @(t,y)[y(2)/llakt; -g*y(3)-d/m*y(4)];
```

Now the loop, switching, and concatenating of the sequence of models becomes more complex: in each event the next actual length can be any combination, as events are linear long – linear short & nonlinear long – nonlinear long, ... The algorithmic event function must work with two event entries:

```
 function [value,isterminal,direction] = hitrelease(~,x)
 value(1) = x(1)-phi_p;  value(2) = x(3)-phi_p
  isterminal(1) = 1; direction(1) = 0;
  isterminal(2) = 1; direction(2) = 0;
end
```

This procedure seems complicated, but it is the general event handling strategy used in Simulink, and therefore useful to study.

1.3 MATLAB Nonlinear Model without Event Handling

The loop, switching, and concatenation of 'long' pendulum and 'short' pendulum is indeed laborious – why not to change the length directly in the algorithmic pendulum function, depending on angle position ?

This quick and 'dirty' approach has 'strange' results. The model function for both models, using MATLAB's effective abbreviations for *if-then-else* clauses, becomes

```
function dxdt = pend_noev_fun(~,x)
   lakt = (x(1) >= phi_p)*l +(x(1) <= phi_p)*ls
   dxdt(1) = x(2)/lakt;
   dxdt(2) = -g*sin(x(1))-d/m*x(2); end;
sol = ode45(@pend_noev_func, [tstart, tend], xstart, options),
```

and the simulation call consist only of one call of the ODE solver. The results are astonishing close to the simulation with events handling, shown in Table 1 (event times for the 'standard' initial values), with unexpected results.

Phase Start	Event Times		
	Event Finder	No Event Finder	
	rtol 1e-4	rtol 1e-5	rtol 1e-4
Long 1	0.0	0.0	0.0
Short 1	0.703459556	0.703459559	0.702954406
Long 2	1.151778788	1.151778616	1.157402743
Short 2	2.590418102	2.590358975	2.583773000
Long 3	2.990527098	2.990509554	2.998855259
Short 3	4.542743634	4.542667578	4.535188672
Long 4	4.867485452	4.867455379	4.874065441
Short 4	6.648742768	6.648572636	
Long 5	6.720351405	6.7204086952	

Table 1: Event times with and without event detection – with vanishing events and unexpected event sequence.

What results are to be expected? Generally, without event finder, the ODE solver recognizes the necessary change of the length at the next integration time instant, i.e. at t_{i+1} definitively too late - it should have happened at unknown $t_e, t_i < t_e < t_{i+1}$

Figure 4 explains the situation, showing both solutions around an event time, taking into account the different stepsizes of ODE solver with event handling t_i^e, t_{k+1}^e and without event handling $t_{k-1}^n, t_k^n, t_{k+1}^n$. Obviously the solver without event finding chooses for the same given tolerances shorter stepsizes around the event.

The reason is a numerical problem: the jump of the length makes the ODE function discontinuous, and the ODE solver tries to keep the tolerances, decreasing the stepsize – in vain: he ends up with $t_{k+1}^n = t_e^n$, violating the tolerances (hidden warnings).

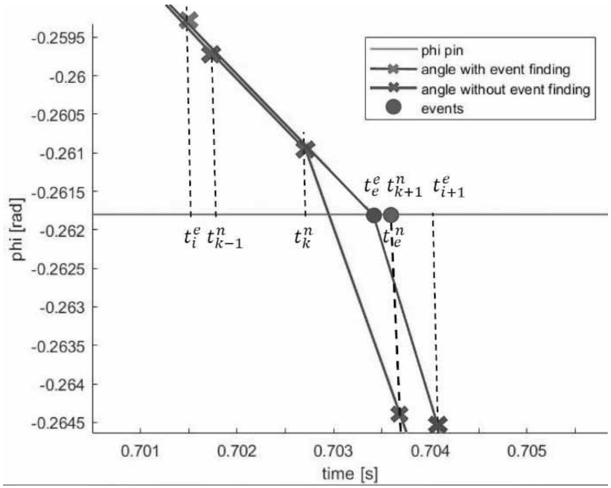


Figure 4: Operation of ODE solver with and without event detection, expected event sequence.

But interestingly the results seem plausible, because the comparisons of time instants

$$t_i^e < t_{k-1}^n < t_k^n < t_e^e < t_{k+1}^n = t_e^n < t_{i+1}^e$$

shows the expected behaviour, 'correct' event before 'faked' event: $t_e^e < t_{k+1}^n = t_e^n$.

Table 1 – event times (t_e^e) with and without (t_e^n) event detection and different ODE tolerances – shows expected numerical values, but only for some event times (denoted in green). Some other event times t_e^n without event detection take place before the correct event ($t_e^n < t_e^e$, denoted in red). This unexpected result is caused by the 'failing' stepsize control, which for higher tolerances takes 'too small' stepsizes, so that the 'correct' event lies after the 'faked' event (Figure 5):

$$t_i^e < t_{k-1}^n < t_k^n < t_{k+1}^n = t_e^n < t_e^e < t_{k+2}^n < t_{i+1}^e$$

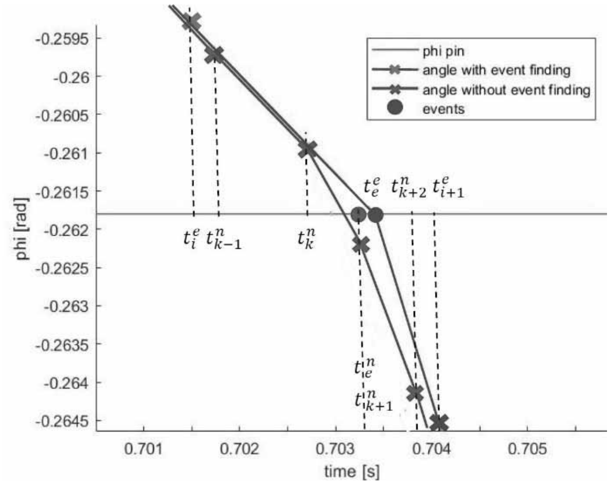


Figure 5: Operation of ODE solver with and without event detection, unexpected event sequence.

Which event time is now the correct one $-t_e^e$, or t_e^n? Indeed the 'exact' event time t_e^e is not exact, it is a numerical approximation. Curiously the quick and 'dirty' implementation with the discontinously changing length can give a better result $t_e^n < t_e^e$, misapplying the failing stepsize control as 'pseudo-event-finder'. But Table 1 shows for low tolerances definitely wrong results, with vanishing events for this strategy. But this strategy must be used, if no event detection is available (as in case of EXCEL, [3]), but only with extreme care. As consequence, event finding is necessary, but it has to be 'synchronised' carefully with tolerance parameters of the ODE solver.

1.4 MATLAB Linear Model with LTI Solving

The linear model is appropriate for small angles, and for time analysis an ODE solver is not the best approach (only approximating the time course). The linear pendulum is an LTI system, and therefore the linear theory with the exponential matrix provides a powerful tool, which is exact with respect to the algorithmic error:

The classically linearized model with reformulation as LTI state space system is

$$\dot{y}_1 = \frac{y_2}{l} \qquad \dot{y}_2 = -g \cdot \mathrm{y}_1 - \frac{d}{m} \cdot y_2,$$

$$\dot{\vec{x}} = A \cdot \vec{x} + B \cdot \vec{u}\,, \quad A = \begin{pmatrix} 1 & \frac{1}{l} \\ -g & -\frac{d}{m} \end{pmatrix} \; B = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Linear theory derives a solution using the exponential matrix

$$\dot{\vec{x}}(t) = e^{A\cdot t} \cdot \vec{x}_0 + \int_0^t e^{A\cdot(t-\tau)} \cdot B \cdot u(\tau) d\tau$$

The properties of the exponential matrix allow to calculate a solution recursively on a time grid by

$$\vec{x}(t_{k+1}) = \vec{x}_{k+1} = e^{A\cdot h} \cdot \vec{x}_k, \qquad h = t_{k+1} - t_k$$

MATLAB offers with the LSIM solver an integrated tool for solving LTI systems by

```
sol=lsim( A, B, C ,D, timegrid )
```

using the state update with the exponential matrix, but without event finding capabilities. So it is now the task to combine the linear exact method with event finding.

Trying to use LSIM, one possibility would be to simulate only one single time step per iteration. After each one-timestep simulation with LSIM, the resulting angle gets checked for crossing the pin angle, before it gets written into a consistent result vector or the event gets estimated with one Newton-algorithm step.

However, this method is very inefficient. Each call of the LSIM forces a new calculation of the exponential matrix for every time step.

Another possibility is to simulate with LSIM longer time periods in a while-loop, and run through the solution vector to check for the event. If the angle crosses the pin angle within the solution vector a Newton-algorithm step gives the estimated event time and only the part of the solution vector until the event gets used – and the while loop continues. This method however can easily get a bit confusing or chunky to implement.

The best method – and presented here - is indeed to calculate only one timestep and check for the event per loop iteration, but not by means of LSIM, but by direct use of the recursion with the exponential matrix - this makes a clearer implementation and reduces unnecessary evaluations and recalculations of the exponential matrix. The event finding is a heuristic Newton implementation: it performs only one iteration, but with exact derivative calculation: $\dot{\varphi}(t^*) = v(t^*)/l$ is generically given by the ODE.

The implementation with while-loop and decision logic for choice of the next pendulum length is similar to the ODE approach with event finder; additionally the event finding is done by the Newton heuristics:

```
% calculate the exponential matrices
   A_expm = expm(A*tstep);
   A_red_expm = expm(A_red*tstep);

 while sol.t(end) < tend          % rewrite initial conditions
 y_start = sol.y(:,end);t_start = sol.t(end);
  if y_start(1) < phi_p    % calculate with short pendulum
  [t,y_new] = expsolve(y_start, t_start, A_red_expm); l_ind = ls;
 elseif y_start(1) > phi_p     % calculate with long pendulum
  [t,y_new] = expsolve(y_start, t_start, A_expm); l_ind = l;
 else    % consider velocity direction
  if y_start(2) < 0     % calculate with short pendulum
  [t,y_new] = expsolve(y_start, t_start, A_red_expm); l_ind = ls;
 else     % calculate with long pendulum
  [t,y_new] = expsolve(y_start, t_start, A_expm); l_ind = l;
end; end

% detect event and perform Newton approximation
 if (sol.y(1,end)-phi_p)*(y_new(1)-phi_p) < 0
   t_event = sol.t(end) + (phi_p - sol.y(1,end))/…
                (sol.y(2,end)/l_ind);
   y_event = [phi_p; (y_new(2)-sol.y(2,end))/(y_new(1)-
                sol.y(1,end))*(phi_p-sol.y(1,end)) + sol.y(2,end)];
   sol.y = [sol.y, y_event]; sol.t = [sol.t, t_event];
   event_times = [event_times, t_event];
 else       % no event happening
   sol.y = [sol.y, y_new]; sol.t = [sol.t, t];
end; end

 function [t,y] = expsolve(y_start, t_start, exp_matrix)
  y = [exp_matrix*y_start];
  t = t_start+tstep; end
```

The results for the ‘standard’ initial values are very close to those of the results from the ODE solution of the linear model, time events are:

0.6920 1.1205 2.5409 2.9318 4.4659 4.7909 6.5325 6.6528

It is to be noted, that the LTI approach with the exponential matrix is a numerical exact method – so these event times may be more reliable as results with the ODE solver. Also the exponential matrix can be computed numerical exact via eigenvalues, etc. MATLAB operates with a very sophisticated environment for calculation of the experimental matrix – see ‘Nineteen dubious ways to compute the matrix exponential’ – [4].

1.5 MATLAB Analytical Model Approach

The linear model is indeed appropriate for small angles, and for time analysis the LTI algorithm with the exponential matrix is very useful in applications. But the pendulum system is a small one, so another approach can make use of the analytical solution, in combination with an appropriate event finding algorithm. This task requires symbolical and numerical computations, and the following investigations deal with three approaches.

Analytic-Numeric Approach

This approach makes directly use of the known analytical solution, a closed formula to be evaluated at arbitrary time instants:

$$\varphi_{sym}(t) = \varphi_{sym}(t, l, \varphi_0, v_0) =$$
$$= e^{-\omega_0 * D * t}[c_1 \cos(\omega_0 \cdot \delta \cdot t) + c_2 \sin(\omega_0 . \delta \cdot t)]$$
$$c_1 = \varphi_0, \omega_0 = \sqrt{\frac{g}{l}}, D = \frac{d \cdot \omega_0}{2m}, \delta = \sqrt{1 - D^2}, c_2 = \frac{v_0}{l \cdot \omega_0 \cdot \delta}$$

and with related tangential velocity $v_{sym}(t, l, \varphi_0, v_0)$.

The analytical (symbolic) solution depends on the pendulum length, and on the initial values, which change in case of event *Hit* or *Release*: $\varphi_{sym}(t, l_e, \varphi_{t_e}, v_{t_e})$.

Again the event function $e(t) = \varphi(t) - \varphi_{pin}$ is used, but now inserting the analytical symbolic solution valid since the previous event $t_{e,p}$ with actual length $l_{e,p}$ chosen at previous event:

$$e(t) = \varphi_{sym}\left(t, l_{e,p}, \varphi_{t_{e,p}}, v_{t_{e,p}}\right) - \varphi_{pin}$$

Starting now with an appropriate guess $t_{e,n}^{[0]}$ for the next event time, a Newton iteration recursively tries to determine the zero of the event function:

$$t_{e,n}^{[k+1]} = t_{e,n}^{[k]} - \frac{e\left(t_{e,n}^{[k]}\right)}{\dot{e}\left(t_{e,n}^{[k]}\right)} =$$
$$= \frac{l_{e,p} \cdot \left(\varphi_{sym}\left(t_{e,n}^{[k]}, l_{e,p}, \varphi_{t_{e,p}}, v_{t_{e,p}}\right) - \varphi_{pin}\right)}{v\left(t_{e,n}^{[k]}, l_{e,p}, \varphi_{t_{e,p}}, v_{t_{e,p}}\right)}$$

Again the necessary derivative is generically given by the tangential velocity, and the resulting MATLAB implementation is simpler than the iteration formula. Again a while-loop performs the iteration, and interestingly four iterations are sufficient to result in event times as accurate as calculated by other methods:

```
while iterations<4
  newton_time=before-((part_sol_phi-phi_p)/part_sol_v);
  part_sol_phi = exp(-alpha_red*newton_time)*…
         ((C1*cos(w_red*newton_time))+…
         (C2_red*sin(w_red*newton_time)));
  part_sol_v = exp(-alpha_red*newton_time)*…
          ((((w_red*C2_red)-(alpha_red*C1))*…
          cos(w_red*newton_time))-…
         (sin(w_red*newton_time)*((w_red*C1)+…
         (alpha_red*C2_red))));
  before=newton_time;
  iterations=iterations+1;
end
```

The iteration loop runs in a while-loop switching between ‘long’ and ‘short’ pendulum: a simple binary counter decides which pendulum length is to be used.

Analytic-Symbolic Approach

This approach again makes directly use of the known analytical solution, a closed formula to be evaluated at arbitrary time instants:

$$\varphi_{sym}(t) = \varphi_{sym}(t, l, \varphi_0, v_0),\ \ v_{sym}(t) = v_{sym}(t, l, \varphi_0, v_0)$$

Task is to determine the events, i.e. the zeros of the event function by means of the event function valid since the previous event $t_{e,p}$ with actual length $l_{e,p}$, chosen at previous event:

$$e(t) = \varphi_{sym}\left(t, l_{e,p}, \varphi_{t_{e,p}}, v_{t_{e,p}}\right) - \varphi_{pin}$$

But now the symbolic solution is inserted directly, so that a nonlinear equation for the next event time $t_{e,n}$ arises:

$$e(t_{e,n}) = \varphi_{sym}\left(t_{e,n}, l_{e,p}, \varphi_{t_{e,p}}, v_{t_{e,p}}\right) - \varphi_{pin} = 0$$

$$e^{-\omega_0 * D * t}\left[c_1 \cos(\omega_0 \cdot \delta \cdot t_{e,n}) + \right.$$

$$\left. + c_2 \sin(\omega_0 \cdot \delta \cdot t_{e,n})\right] - \varphi_{pin} = 0$$

$$c_1 = \varphi_0, \omega_0 = \sqrt{\frac{g}{l}}, D = \frac{d \cdot \omega_0}{2m}, \delta = \sqrt{1 - D^2}, c_2 = \frac{v_0}{l \cdot \omega_0 \cdot \delta}$$

It is laborious to solve this equation with respect to $t_{e,n}$ ‘manually’, but MATLAB provides with the *Symbolic Toolbox* an adequate tool. Defining the event time $t_{e,n}$ as symbolic variable, and the error function as symbolic equation, MATLAB’s vpasolve tool indeed masters this task. After solution, the symbolic event time can be numerically evaluated. The implementation is quite short, and results in almost equivalent results with other approaches – see Table 3, Section 5.

```
syms t  equa
C1=phi0;  C2=(v0+(alpha*phi0*l))/(w*l);  %constants
equa=exp(-alpha*t)*((C1*cos(w*t))…
        +(C2*sin(w*t)))==phi_p;      %equation for phi=phi_p
te_sym = vpasolve( equa , t);  te = double(te_sym)
```

Full Symbolic Approach

For this approach the *Symbolic Toolbox* also sets up the analytical solutions $\varphi_{sym}(t)$ and $v_{sym}(t)$ by solving the ODEs analytically. Therefore, the state variables must be implemented as symbolic functions, as well as the differential equations. The following implementation documents the symbolic automatized operations:

```
syms phi(t) v(t)   %work with symbolic variables
%differential equations with symbolic values
  eqns = [diff(phi,t) == v/l, diff(v,t) == -g*phi-d*v/m];
  eqns_red=[diff(phi,t) == v/ls, diff(v,t) == -g*phi-d*v/m];
  cond=[phi(0)==solution_phi(end),v(0)==solution_v(end)];
%solve differential equations
  if n==0   structure=dsolve(eqns,cond); n=1;
  else structure=dsolve(eqns_red,cond);  n=0; end
```

The ODEs are solved with the dsolve tool which returns a symbolic time-dependent solution. As in the *Analytic-Symbolic Approach*, the symbolic vpasolve tool allows to determine the event times, using now the symbolic ODE solutions, and not the ‘manually’ derived solutions – very comfortable. With a *for-loop* the symbolic ODE solutions are evaluated till the event time with a defined time step and transformed to a numerical value.

But because every evaluation step includes a transformation from symbolic to numeric value, this approach takes much longer time in MATLAB.

Comparison of Analytic Approaches

All analytic approaches provide the ‘exact’ same results for the event times (within rounding tolerances):

0.6920 1.1205 2.5409 2.9318 4.4658 4.7908 6.5321 6.6530

But the calculation time duration differs significantly:

- the *Analytic-Numeric Approach* has the shortest calculation time (0.1-0.25 seconds),
- the *Analytic-Symbolic Approach* takes 10 times longer (2-5 seconds),
- and the *Full Symbolic Approach* has the longest time (15-17 seconds).

For the used ‘standard’ initial values $\varphi_0 = \pi/6$, $\varphi_{pin} = -\pi/12$, the event times are close to the event times of the nonlinear system. For smaller initial values $\varphi_0 = \pi/12$, $\varphi_{pin} = -\pi/24$ they get very close (Task B).

An interesting alternative for getting generally closer to the nonlinear behaviour would be the use linear affine systems with different linearization points.

1.6 TASK C: Boundary Value Problem

The benchmark defines as third task the solution of a boundary value problem: which initial angular velocity $\dot{\varphi}_0$ with initial angle $\varphi_0 = \pi/6$ and pin angle $\varphi_{pin} = -\pi/12$ is necessary to reach after one *Hit* event exactly $\varphi_{target} = -\pi/2$ (includes $\dot{\varphi}_{target} = 0$) ?

The classic approach is an iteration of $\dot{\varphi}_0$ or $\dot{v}_0$, resp., each with a simulation run with 'sufficient' time length. A charming easier alternative is to view the problem from the target values: starting with the (short) pendulum at angle $\varphi_{target} = -\pi/2$ and velocity $\dot{\varphi}_{target} = 0$ backwards in time, the pendulum will reach after one *Release* event as long pendulum at a certain time $t_e^{[\pi/6]}$ the angle $\varphi\left(t_e^{[\pi/6]}\right) = \pi/6$ with wanted velocity $\dot{\varphi}(t_e^{[\pi/6]}$).

The dynamics backwards in time can be derived by a time transformation $\tau = -t$ yielding $d\tau = -dt$ and

$$\frac{d}{d\tau}\varphi(\tau) = -\frac{1}{l}\cdot v(\tau), \frac{d}{d\tau}v(\tau) = -g\cdot\sin\varphi(\tau) + \frac{d}{m}\cdot v(\tau)$$

As consequence, the boundary value problem changes to a simulation backwards in time, with *Release* event and with a *Target* event at $t_e^{[\pi/6]}$. The event function becomes a vector event function:

$$\vec{e}\big(\varphi(t)\big) = \quad (\varphi(t) - \varphi_{pin} \quad \varphi(t) - \pi/6)^T$$

The implementation for the nonlinear model with event handling can be re-used, changing only the signs in the model function and adding the *Target* event:

```
function [value,isterminal,direction] = hitrelease(~,y)
 value  = [y(1)-phi_p    y(1)-pi/6];
 isterminal = [1  0]; direction = [0 -1]; end
```

Note, that the *Target* event does not terminate the simulation, it stores only the state target values (Figure 6, events *Release* (green cross) and *Target* (red cross).

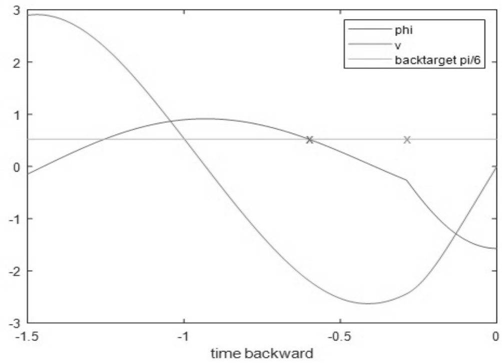


Figure 6: Simulation backwards in time for determining wanted angular velocity $\dot{\varphi}_0$.

The numerical results are quite accurate:

$$t_e^{[\pi/6]} = 0.601299, \dot{\varphi}(t_e^{[\pi/6]}) = v\left(t_e^{[\pi/6]}\right) = -2.184514$$

2 Simulink Model with and without Event Handling

Simulink is MATLAB's graphical modelling environment based on directed signal flow and classic block-oriented modelling. Basis is the *Integrator Block* – denoted by $\int or \frac{1}{s}$, which 'integrates' the input, the ODE. In principle, Simulink makes use of the integral notation of ODEs, in case of the pendulum of

$$\varphi = \int\left(\frac{1}{l}\cdot v\right) \quad v = \int\left(-g\cdot sin\varphi - \frac{d}{m}\cdot v\right)$$

The Simulink model of the constrained pendulum is based on the pendulum ODEs using two integrator blocks.

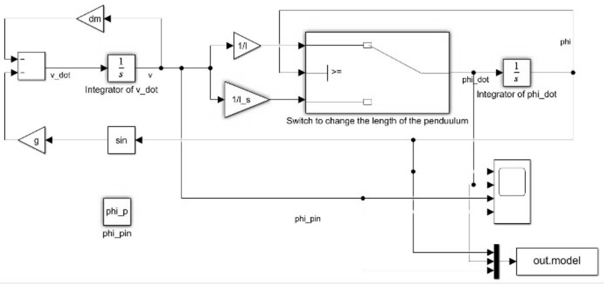


Figure 7: Simulink model for the constrained pendulum with switch block for changing length.

Due to continuous differentiability of the states φ and v, only the length of the pendulum must be changed, depending on the angle φ. There, Simulink offers a S*witch* block, which compares whether the current angle φ is greater than pin angle φ_p (the threshold is defined in the switch block), and which therefore models the event function (Figure 7).

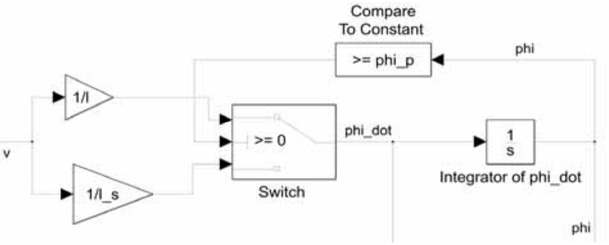
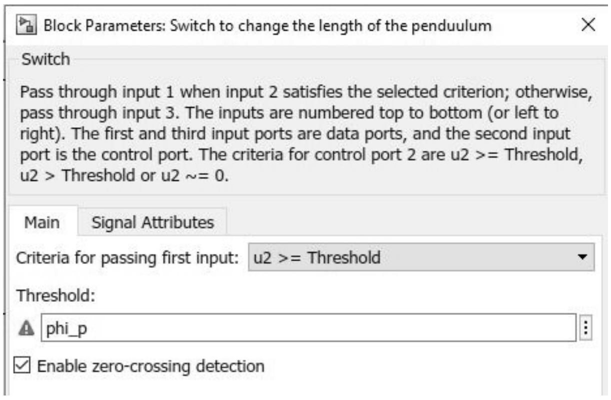


Figure 8: Simulink model for the constrained pendulum with compare block and switch block for changing length; detail.

A more detailed approach for the event function is the use of a *Compare to Constant* block in addition to a switch block. Since the output of this block is Boolean, the threshold of the consecutive switch is set to 0 (Figure 8). However, both switching options deliver the same results.

Simulink offers state event handling by means of the *Zero Crossing* options in many blocks (23 blocks !), as e.g. in the *Hit Crossing* block (for general event functions), and in *Compare* and *Switch* blocks (for simple event functions).

It is possible to activate and deactivate the zero-crossing detection (Figure 9, switch block), so simulation can run with and without event detection.



Block Parameters: Switch to change the length of the penduulum

Switch

Pass through input 1 when input 2 satisfies the selected criterion; otherwise, pass through input 3. The inputs are numbered top to bottom (or left to right). The first and third input ports are data ports, and the second input port is the control port. The criteria for control port 2 are u2 >= Threshold, u2 > Threshold or u2 ~= 0.

Main | Signal Attributes

Criteria for passing first input: u2 >= Threshold

Threshold: phi_p

☑ Enable zero-crossing detection

Figure 9: Configuration menu for switch block with threshold definition and zero crossing enabling and disabling.

Simulink parses the graphic model and compiles it into state space model $\dot{\vec{x}}(t) = \vec{f}(\vec{x}, t)$. For simulation it makes use of the MATLAB ODE solver suite, quite similar to the use in MATLAB, but with extended possibilities for events, triggered or enabled/disabled submodels, and some other special tasks.

Indeed Simulink does state event handling, and offers – in contrary to MATLAB – also possibilities to implement *Event Actions*. Generally, for this purpose there are two possibilities:

- *Parameter Change Events* and *State Change Events* can be directly described in one model by switches and re-initialisation of integrator blocks, triggered by blocks capable of zero crossing detection.
- *Derivative Change Events* and *Model Change Events* need another technique: original and changed derivatives or original and changed models resp., are both put into different Simulink submodels, which can be enabled or disabled. The 'root'model handles via events the switching between the submodels by enabling or disabling them.

The above submodel approach for structural-dynamic systems is the so-called *Monolithic State Space Approach* ([2]), the alternative to the *Hybrid Decomposition Approach*, used also in MATLAB (see Section 1.1).

The term monolithic refers to the fact that the state space is a maximal one and not consecutively split into smaller state spaces: during simulation, in disabled submodels the states are 'frozen', and re-activated, as soon as the submodel is enabled.

In this Simulink model for the constrained pendulum the 'one model' approach is chosen, a generic simple approach with state vector $(\varphi(t)\ v(t))^T$ and switching length. The more general alternative would work with a monolithic overall state

$$\left(\varphi_l(t)\ v_l(t)\ \varphi_{l_s}(t)\ v_{l_s}(t)\right)^T$$

– used in the Stateflow modelling approach, Section 3.

State Event Detection – Zero Crossing

Simulink allows to enable or disable event handling. Simulink's event detection is more sophisticates than the MATLAB algorithm. Again for event detection the *Regula Falsi* method, a combination of bisection method and secant method, is used, with a 'hard' accuracy limit – in case of the *non-adaptive* strategy; recent Simulink versions offer an *adaptive* strategy, which instead of the 'hard' limit works with an appropriate threshold around zero, stopping the detection algorithm.

The investigations here refer to enabled (non-adaptive) zero crossing and disabled zero crossing (no event detection), and show as with MATLAB astonishing but different results. Key parameters for localisation and accuracy are again the parameters for the ODE45 solver, the tolerances and the maximal stepsize.

Generally, the results are very close to the MATLAB results – time courses as well as event times – with and without event handling.

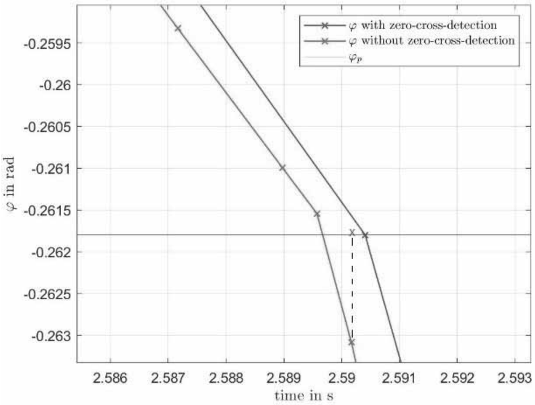


Figure 10: Solution with and without zero crossing near event: blue cross – detected event time, red cross – 'accepted' late event time.

There are small, but crucial influences of the algorithmic parameters, yielding astonishing results. Figure 10 displays in detail the ODE45 calculations around an event time, with enabled/disabled zero crossing:

- Without event detection: ODE45 varies the step-size, and close to reaching the angle φ_{pin},
  the solver reduces the step size because the state forecast with switched length exceeds the given tolerance;
  after some tries (more solution points before event), the solver ends up with an internal error warning and accepts the 'next' time instant as event time $t_e^n = t_{n+1}$ (theoretically after the event).
- With event detection: the ODE45 solver approaches the event with bigger stepsizes. After detection of the event, event localisation starts and results in 'exact' event time t_e^e (theoretically before the time instant calculated without zero crossing algorithm: $t_e^e < t_e^n$).
- Comparison: with fixed stepsizes, the event time sequence must obey $t_e^e < t_e^n$;
  with step size control, without zero crossing algorithm, the stepsize control decreases the step-sizes, so that the 'accepted' event time $t_e^n = t_{n+1}$ may be before (!) the exact time: $t_e^n < t_e^e$;
  for further details, see discussion in Section 1.4).

Indeed the stepsize control based on tolerances and step-size limits yields this astonishing results. From another viewpoint, stepsize control could be seen as competitive event handling, searching for a stepsize which tolerance reaching event. Some observations:

- Using the ODE45 solver with maximal stepsize set to automatic, only three hits of the pin are found. The amount of touches found depends on the chosen maximum stepsize and the tolerances.
- For the ODE45 with enabled zero-cross-detection with a maximum step size smaller than 0.154 and a relative tolerance of 10^{-4} the pendulum hits the pin four times (Figure 11).
- In contrast to this, for the ODE45 without zero-cross-detection, since it is not as accurate, the maximum step size needs to be smaller than 0.145 to observe four hits.

Further results, especially a comparison between the different approaches and resulting ecent times, see Section 5, Table 3.

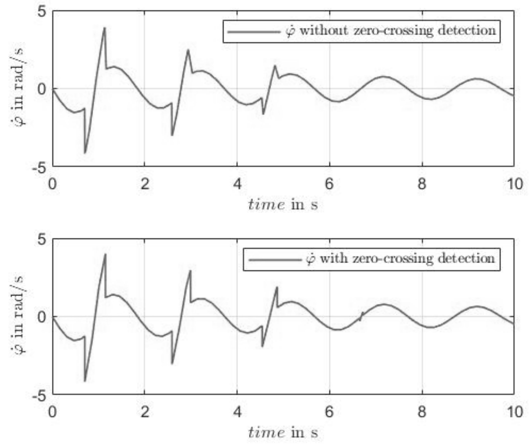


Figure 11: Angular velocity with disabled and enabled zero crossing, solver option maximum stepsize set to 0.154, rel. tolerance 10^{-4}; disabled zero crossing lets 7th and 8th event vanish.

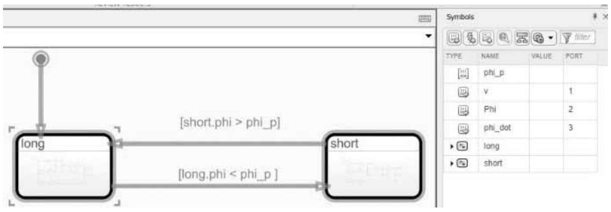
3 Simulink Stateflow Model with and without Event Handling

Stateflow is a Simulink extension offering control schemes of signals and submodels by automata. Recent versions of Stateflow allow not only logic states in the automata, but also hybrid continuous states.

Use of Stateflow for the constrained pendulum model could on the one side simply replace the switch block of the Simulink implementation in Section 2 by one state chart 'actual' length' alternatively switched via feedback with switch of length. On the 'advanced' side, Stateflow allows to implement the constrained pendulum system as structural-dynamic system by the monolithic state space approach ([2]) using indeed the maximal state space

$$\left(\varphi_l(t)\ v_l(t)\ \varphi_{l_s}(t)\ v_{l_s}(t)\right)^T$$

The implementation is based on two almost identical Simulink submodels (Figure 13, a) and b)) with the pendulum system. Stateflow (Figure 12) switches between these two models, one with length l and one with length l_s.



TYPE	NAME	VALUE	PORT
	phi_p		
	v		1
	Phi		2
	phi_dot		3
	long		
	short		

Figure 12: Overall Stateflow model with two hybrid states 'long' and 'short'.

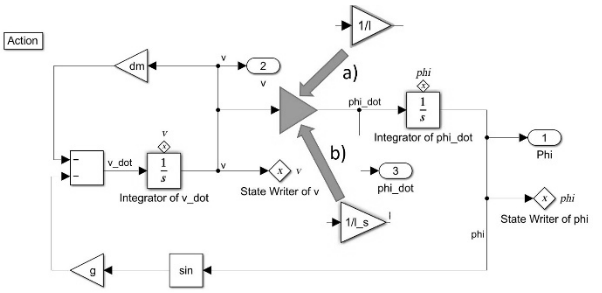


Figure 13: Model of the long pendulum (a) in the hybrid state 'long', and of short pendulum (b) in hybrid state 'short'. StateWriter blocks transfer the system states φ and v between the hybrid states.

The Stateflow model (Figure 12) includes the hybrid states 'long' and 'short'. The arrows in between mark the switch between the hybrid states: the conditions, when to switch (event function zero crossing) are given above the arrows in the squared brackets. The two hybrid states consist of Simulink submodels with the continuous dynamics (Figure 13, a) and b)).

The transition condition is given by the change of sign in the event function. The transition condition from the long pendulum to the short one is $\varphi_{long} < \varphi_{pin}$. As soon as the inequation is satisfied, the computation is done in the model of the short pendulum, and the system states of the long pendulum are frozen. To switch from the short pendulum to the long pendulum model, the inequation $\varphi_{short} > \varphi_{pin}$ needs to be fulfilled.

To start with the correct initial values after changing the model, a *StateWriter* block is used transferring the values at event time. Although the default model is the long pendulum (defined by the root arrow to state 'long', the simulation still works if the initial angle $\varphi_0 < \varphi_{pin}$. This is based by the order of work steps in Simulink Stateflow: if a new hybrid state, in this case a dynamic model, is entered, Simulink checks if a transition condition is fulfilled, in which case the transition is done before the calculation in this hybrid state starts.

An important tool in Stateflow is the *Symbols Panel*, (Figure 12, at right). In this panel the variables used in the Stateflow model can be defined as 'constant data', 'parameter data', 'local data', etc. In case of the constrained pendulum 'phi_p' is defined as a 'parameter data' and 'v', 'phi' and 'phi_dot' are defined as 'output data'.

As the Stateflow implementation clearly makes use of the same zero crossing as in the Simulink implementation, the results – time courses and event times – are very close. Further results, especially the comparison between the different approaches can be found in Table 3, Section 5.

4 System Sensitivity

Time domain analysis is the primary tool for analysis of dynamic systems $\dot{\vec{x}}(t) = f(\vec{x}, t)$. But as systems depend also on parameters $\vec{p}$, so systems and solutions are also functions of the parameters: $\dot{\vec{x}}(t, \vec{p}) = f(\vec{x}, \vec{p}, t), x(t, \vec{p})$.

Sensitivity Analysis is a method, which qualifies and quantifies the change of the solutions (or key measures of the solution) with respect to change of the parameters – and the two method groups are analytical methods and stochastic methods.

4.1 Parameter Sensitivity Functions

The so-called *Parameter Sensitivity Functions* with the *Parameter Sensitivity Equations* are a classical tool for analysing the dynamics of an ODE system $\dot{\vec{x}}(t) = f(\vec{x}, \vec{p}, t)$ with respect to change of parameters $\vec{p}$.

The sensitivity functions are generally the partial derivatives of the states $x_i(t)$ with respect to the parameters p_k, obeying the sensitivity equations, ODEs coupled with the system equations and derived by valid change of the derivation sequence:

$$x_{i,p_k}(t) = \frac{\partial}{\partial p_k} x_i(t) \quad \dot{x}_{i,p_k}(t) = \frac{\partial}{t} \frac{\partial}{\partial p_k} x_i(t)$$

$$\frac{\partial}{t} \frac{\partial}{\partial p_k} x_i(t) = \frac{\partial}{\partial p_k} \frac{\partial}{t} x_i(t) = \frac{\partial}{\partial p_k} \dot{x}_i(t) = \frac{\partial}{\partial p_k} f_i(\vec{x}, t)$$

$$\dot{x}_{i,p_k}(t) = \frac{\partial}{\partial p_k} f_i(\vec{x}, t) \quad x_{i,p_k}(0) = 0$$

While the sensitivity function $x_{i,p_k}(t)$ is a general measure for the change of state $x_i(t)$ with respect to parameter p_k, the *Normalized Sensitivity Function* $\lambda_{i,p_k}(t)$ measures quantitatively the change of the state $x_i(t)$ due to a 1% relative change of the parameter p_k:

$$\lambda_{i,p_k}(t) = x_{i,p_k}(t) \cdot \frac{p_k}{100}$$

The nonlinear pendulum with two states and four parameters deduces eight sensitivity functions and ODE sensitivity equations $\varphi_d, v_d, \varphi_l, v_l, \varphi_m, v_m, \varphi_g, v_g$ with e.g.:

$$\dot{\varphi} = \frac{1}{l} \cdot v \quad \dot{v} = -g \cdot sin\varphi - \frac{d}{m} \cdot v$$

$$\dot{\varphi}_d = \frac{1}{l} \cdot v_d \quad \dot{v_d} = -g \cdot cos\varphi \cdot \varphi_d - \frac{d}{m} \cdot v_d - \frac{1}{m} \cdot v$$

$$\dot{\varphi}_l = \frac{1}{l} \cdot v_l - \frac{1}{l^2} \cdot v \quad \dot{v}_l = -g \cdot cos\varphi \cdot \varphi_l - \frac{d}{m} \cdot v_l$$

Figure 14 shows all sensitivity functions for a longer time horizon, showing interesting oscillatory behaviour especially for the length sensitivity φ_l.

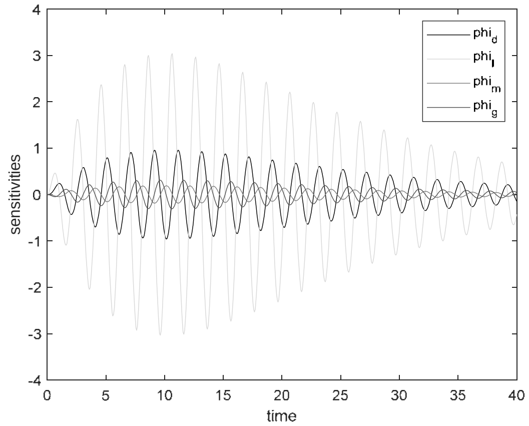


Figure 14: Sensitivity functions as ODE solutions of the sensitivity equations with additional oscillation.

The MATLAB implementation of the sensitivity system is straightforward:

```
   x0 = [ phi0 v0 0 0 0 0 0 0 0 0 ]';
   [t, x] = ode45(@(t,x) pend_sens(t,x,rl,rl2,dm,rm,dmm,g),…
            [0, tend], x0);
 function dxdt = pend_sens(t,x,rl,rl2,dm,rm,dmm,g)
 dxdt=zeros(10,1);
 dxdt = [ rl*x(2);          -g*sin(x(1))      - dm*x(2);          ...
      rl*x(4);          -g*cos(x(1))*x(3) - dm*x(4)  - rm*x(2)  ;...
      rl*x(6)-rl2*x(2);  -g*cos(x(1))*x(5) - dm*x(6);          ...
      rl*x(8);          -g*cos(x(1))*x(7) - dm*x(8)  + dmm*x(2) ;...
      rl*x(10);          -g*cos(x(1))*x(9) - dm*x(10) - sin(x(1))];
 end
```

Of interest for the constrained pendulum are shorter periods of the oscillations in between the events *Hit* and *Release*. Figure 15 shows the normalized sensitivity functions for length and damping: a length change is dominating:

$$\lambda_{\varphi,d}(t) = \varphi_d(t) \cdot \frac{d}{100} \quad \lambda_{\varphi,l}(t) = \varphi_d(t) \cdot \frac{l}{100}$$

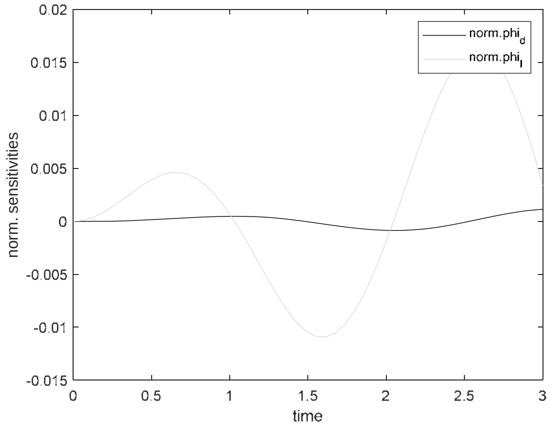


Figure 15: Normalized sensitivity functions for damping (smooth) and length (dominating).

Of course the sensitivity functions can be continued after an event, starting with nonzero initial values (final values of previous segment).

Already the small pendulum system shows that derivation of the sensitivity functions is voluminous and error-prone. Symbolic computation in combination with appropriate function handling automatizes the derivation and simulation of sensitivity equations, e.g. with the MATLAB Symbolic Toolbox:

- Definition of the symbolic system ODE function with symbolic states and parameters
- Definition of symbolic sensitivity states, symbolic derivation of the sensitivity ODE functions
- Composing system ODE functions and sensitivity ODE functions to complete symbolic function set
- Transformation of symbolic sensitivity ODEs to numerical ODE (vector) function for ODE simulation

Nevertheless, the sensitivity analysis with sensitivity functions and sensitivity ODE system is based on continuous dependency of the parameters – which is not the case for the parameter pin position φ_{pin}.

4.2 Sensitivity by Monte-Carlo Method

Partly simpler is another method for sensitivity analysis, the *Monte-Carlo Method* (used also for other tasks, as simulation itself). Generally, Monte-Carlo technique works with multiple random disturbances of inputs, calculating the multiple system responses, and finally calculating mean and standard deviation of the responses.

For the constrained pendulum this technique could be used for the pendulum parameters d, l, m, g and time courses $\varphi(t;d)$ and $\varphi(t;l)$ as system response – without benefit compared to sensitivity functions. But the technique offers itself for analyzing the sensitivity of the event times $t_{e,k}$ with respect to the pin angle φ_{pin}.

Starting with a sufficient big sample of disturbed pin angles $\varphi_{pin}^{[rand_i]}$, the resulting *Hit* and *Release* event times

$$t_{e,k}(\varphi_{pin}^{[rand_i]})$$

for k=1,…, n_e (#events), i=1,…,n_s (#samples) are calculated, and then statistically evaluated with mean value and standard deviation:

$$t_{e,k}^{mean} = \frac{1}{n_s}\sum_i t_{e,k}\left(\varphi_{pin}^{[rand_i]}\right), k = 1, \dots n_e,$$

$$t_{e,k}^{std} = \sqrt{\frac{1}{n_s}\sum_i \left(t_{e,k}\left(\varphi_{pin}^{[rand_i]}\right) - t_{e,k}^{mean}\right)^2}$$

Sensitivity investigations will consider now the event times $t_{e,k}^{e}$ for the nonlinear model with event detection, the event times $t_{e,k}^{n}$ for the nonlinear model without event detection, and the event time differences

$$\Delta_{e,k}^{e-n} = t_{e,k}^{e} - t_{e,k}^{n}$$

An implementation in MATLAB is very easy, because calculation of mean value and standard deviation are basic tasks. Simulation is performed in a loop with sample size n_s, collecting a matrix T_e^e (t_events_e_mc) with n_e event times for each event time and for each sample:

$$T_e^e = \begin{pmatrix} t_{e,1}^{e,1} & t_{e,2}^{e,1} & \cdots & t_{e,n_e}^{e,1} \\ t_{e,1}^{e,2} & t_{e,2}^{e,2} & \cdots & t_{e,n_e}^{e,2} \\ \vdots & \vdots & \ddots & \vdots \\ t_{e,1}^{e,n_s} & t_{e,2}^{e,n_s} & \cdots & t_{e,n_e}^{e,n_s} \end{pmatrix}$$

and equivalently the matrix T_e^n (t_events_n_mc) for event times without event detection, and Δ_e^{e-n} (t_events_dif_mc) for the time differences, with results

$t_{e,k}^{e,mean}$ (mean_t_events_e), $t_{e,k}^{e,std}$ (std_t_events_e)

$t_{e,k}^{n,mean}$ (mean_t_events_n), $t_{e,k}^{n,std}$ (std_t_events_n),

$\Delta_{e,k}^{e-n,mean}$ (mean_t_events_dif), $\Delta_{e,k}^{e-n,std}$ (std_t_events_dif).

In contrary to the symbolic vector and matrix notation the implementation is very simple:

```
mean_t_events_e   = mean(t_events_e_mc)
std_t_events_e    = std(t_events_e_mc)
mean_t_events_n   = mean(t_events_n_mc)
std_t_events_n    = std(t_events_n_mc)
mean_t_events_dif = mean(t_events_dif_mc)
std_t_events_dif  = std(t_events_dif_mc)
```

Event No.	t_e^e	$t_e^{e,mean}$	$t_{e,k}^{n,mean}$	$\Delta_e^{e-n,mean}$
		$t_e^{e,std}$	$t_{e,k}^{n,std}$	$\Delta_{e,k}^{e-n,std}$
1	0.7034	0.7034	0.7039	0.0007
		0.0022	0.0024	0.0009
2	1.1520	1.1519	1.1522	0.0005
		0.0004	0.0007	0.0006
3	2.5901	2.5903	2.5912	0.0009
		0.0057	0.0059	0.0011
4	2.9905	2.9907	2.9917	0.0010
		0.0029	0.0031	0.0012
5	4.5425	4.5426	4.5423	0.0015
		0.0110	0.0109	0.0019
6	4.8672	4.8676	4.8675	0.0017
		0.0053	0.0056	0.0021

Table 2: Monte-Carlo sensitivity analysis for event times with and without event detection. nonlinear model.

Table 2 summarizes the results for a Monte-Carlo study with a 5% uniformly distributed change in pin position, with a sample of $n_s = 500$ tolerance of 1e-4. The results indicate that event times are not very sensitive with respect to small changes in the pin position, and that deviations are in the same range as the deviations between event times with and without state detection (the step size control in case of no event detection really seems to compensate the missing event detection).

5 Comparison of Event Detection

In case of nonlinear dynamics, the quest for the 'exact' event time t_e^{exact} cannot really be determined – all ODE solutions with (t_e^e) and without (t_e^n) event algorithms are only approximations.

Results with event detection in MATLAB, Simulink, and Stateflow are reliable and very close, if the ODE45 parameters (tolerances, maximal stepsize) are chosen properly (Table 3, case A, C, D). Because the last 'short' pendulum phase is very short (~0.045 sec), all ODE approaches must limit the stepsize.

The quick and 'dirty' approaches without event algorithm calculate astonishing results for the event times t_e^n: partly they occur before the 'exact' ones t_e^e – contrary to expectation (Table 3, Simulink case A vs. B, MATLAB case D vs E & F). Responsible is the stepsize control: the jump in length causes smaller stepsizes to keep the tolerances – in vain: step size control ends up with $t_{k+1}^n = t_e^n$ violating the tolerances (hidden warnings). The step size control seems to replace the event algorithm, although it is mathematically wrong (discontinuity). Is the earlier event time t_e^n more exact than the later t_e^e ? – no, because different solver parameters let also t_e^e happen earlier. Furthermore, the 'dubious' stepsize control lets events t_e^n vanish, which can be corrected by new solver tuning (Table 3, Simulink case B, MATLAB case E vs. F).

All linear model solutions are close to the nonlinear ones. For algorithmic event detection, linear models can make use of an ODE solver – with similar results for t_e^e and t_e^n (Table 3, case G vs. H).

Usually, linear systems are solved 'exactly' by the exponential matrix, in a recursive loop with fixed steps. Event detection can be implemented by a 'cheap' Newton step –- successful, effective and more reliable t_e^e event times with ODE (Table 3, case E vs. G). Accuracy can be improved by smaller steps around the events. Mathematicians like the analytic solution, where event detection requires solution of nonlinear equations, either by a partly symbolic Newton algorithm, or direct symbolically, all with same 'most' exact results t_e^e and extremely close to solution with the exponential matrix (Table 3, case I vs J).

Event time	A Simulink Nonlinear Model with Event Detection rtol 1e-4	B Simulink Nonlinear Model without Event Detection rtol 1e-4	C Stateflow Nonlinear Model with Event Detection rtol 1e-4	D MATLAB Nonlinear Model with Event Detection rtol 1e-4	E MATLAB Nonlinear Model without Event Detection rtol 1e-4	F MATLAB Nonlinear Model without Event Detection rtol 1e-5	G MATLAB Linear Model ODE Solver with Event Detection rtol 1e-4	H MATLAB Linear Model ODE Solver without Event Detection rtol 1e-4	I MATLAB Linear Model Solution Exponential Matrix & Onestep Newton	J MATLAB Linear Model Analytic Solution all Approches
1.	0.703454	0.70228	0.703459	0.703459	0.703327	0.703427	0.692023	0.686477	0.692018	0.692023
2.	1.151559	1.159532	1.151763	1.151771	1.160078	1.151778	1.120535	1.130859	1.120547	1.120545
3.	2.590362	2.585031	2.590407	2.590416	2.588979	2.590338	2.540851	2.510915	2.540911	2.540860
4.	2.990219	2.986587	2.990503	2.990514	2.998792	2.990403	2.931783	2.932346	2.931825	2.931800
5.	4.542716	4.54376	4.542743	4.542752	4.543987	4.542705	4.465752	4.447178	4.465854	4.465756
6.	4.867135	4.865345	4.867457	4.867471	4.874602	4.867492	4.790766	4.782015	4.790856	4.790785
7.	6.649999	-	6.648841	6.648860	-	6.648731	6.532110	-	6.532465	6.532060
8.	6.719074	-	6.720245	6.720253	-	6.720995	6.652945	-	6.652801	6.652993

Table 3: Event times t_e^e and t_e^n for all presented approaches – 'standard' initial values.

6 Integration of Approaches into MMT E-Learning Server

The *MMT E-learning System* – MMT stands for *Mathematics, Modelling and Tools* – is a tool used at the Institute of Analysis and Scientific Computing and at TU Vienna for education in modelling and simulation (and also by other institutes dealing with education in modelling and simulation).

The MMT server, developed since 2006 ([5]), plays a major role in lectures for modelling and simulation and courses in applied mathematics. The case studies and examples for modelling and simulation deal with different kinds of modelling, like ODEs, cellular automata or agent-based models, and distinct applications.

The MMT System is a web application with

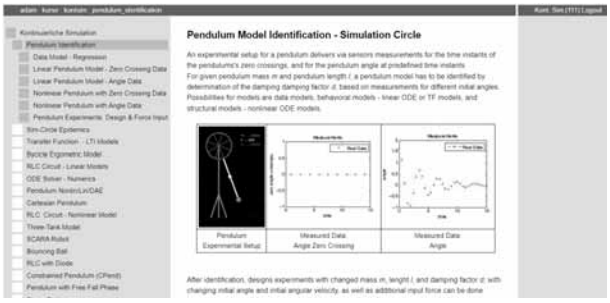
- a frontend presenting interactive examples and case studies for modelling and simulation, as well as related lecture notes,
- with MATLAB running as simulation engine,
- and with a backend content management system for preparing examples, case studies etc. and lecture note content.

Figure 16: Entrance Webpage of the MMT Server.

When a student enters the website, he is welcomed by Adam Ries (1492 – 1559; a German mathematician), with login (Figure 16).

After login, all courses the student takes are listed on the left hand side. When the student chooses a course and in this course an example, the web page offers experimentation features (Figure 17). On the left hand side a navigation lets select different examples, and further course topics (here 'Pendulum Identification').

Each course topic includes appropriate examples (Figure 18, selected 'Nonlinear Pendulum with Zero Crossing Data'). On the right hand side there is a link to the source code of the example ('view m-file'; MATLAB m-files). Furthermore there can be other files linked, which can be downloaded like lecture notes, tables, etc.

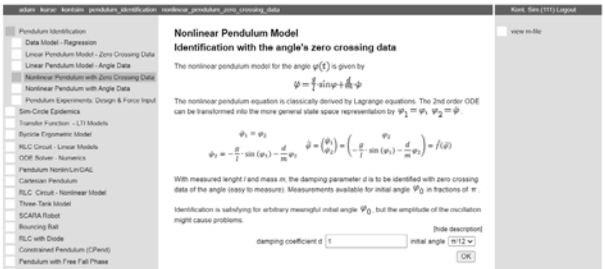

adam kurse kontsim pendulum_identification
Kont. Sim (111) Logout
Kontinuierliche Simulation
Pendulum Identification
Data Model - Regression
Linear Pendulum Model - Zero Crossing Data
Linear Pendulum Model - Angle Data
Nonlinear Pendulum with Zero Crossing Data
Nonlinear Pendulum with Angle Data
Pendulum Experiments: Design & Force Input
Sim-Circle Epidemics
Transfer Function - LTI Models
Bycicle Ergometric Model
RLC Circuit - Linear Models
ODE Solver - Numerics
Pendulum Nonlin/Lin/DAE
Cartesian Pendulum
RLC Circuit - Nonlinear Model
Three Tank Model
SCARA Robot
Bouncing Ball
RLC with Diode
Constrained Pendulum (CPend)
Pendulum with Free Fall Phase
Real Data
time
Pendulum
Experimental Setup
Measured Data
Angle Zero Crossing
Measured Data
Angle

Pendulum Model Identification - Simulation Circle

An experimental setup for a pendulum delivers via sensors measurements for the time instants of the pendulums's zero crossings, and for the pendulum angle at predefined time instants.
For given pendulum mass *m* and pendulum length *l*, a pendulum model has to be identified by determination of the damping damping factor *d*, based on measurements for different initial angles.
Possibilities for models are data models, behavioral models - linear ODE or TF models, and structural models - nonlinear ODE models.

After identification, designs experiments with changed mass *m*, lenght *l*, and damping factor *d*, with changing initial angle and initial angular velocity, as well as additional input force can be done

Figure 17: MMT Course Entrance.


adam kurse kontsim pendulum_identification nonlinear_pendulum_zero_crossing_data
Kont. Sim (111) Logout
view m-file
Pendulum Identification
Data Model - Regression
Linear Pendulum Model - Zero Crossing Data
Linear Pendulum Model - Angle Data
Nonlinear Pendulum with Zero Crossing Data
Nonlinear Pendulum with Angle Data
Pendulum Experiments: Design & Force Input
Sim-Circle Epidemics
Transfer Function - LTI Models
Bycicle Ergometric Model
RLC Circuit - Linear Models
ODE Solver - Numerics
Pendulum Nonlin/Lin/DAE
Cartesian Pendulum
RLC Circuit - Nonlinear Model
Three Tank Model
SCARA Robot
Bouncing Ball
RLC with Diode
Constrained Pendulum (CPend)
Pendulum with Free Fall Phase

Nonlinear Pendulum Model
Identification with the angle's zero crossing data

The nonlinear pendulum model for the angle $\varphi(t)$ is given by

$$\ddot{\varphi}=\frac{g}{l}\cdot\sin\varphi+\frac{d}{m}\cdot\dot{\varphi}$$

The nonlinear pendulum equation is classically derived by Lagrange equations. The 2nd order ODE can be transformed into the more general state space representation by $\varphi_1=\varphi$, $\varphi_2=\dot{\varphi}$.

$$\dot{\varphi}_1=\varphi_2 \qquad \dot{\vec{\varphi}}=\begin{pmatrix}\dot{\varphi}_1\\ \dot{\varphi}_2\end{pmatrix}=\begin{pmatrix}\varphi_2\\ -\frac{g}{l}\cdot\sin(\varphi_1)-\frac{d}{m}\varphi_2\end{pmatrix}=\vec{f}(\vec{\varphi})$$
$$\dot{\varphi}_2=-\frac{g}{l}\cdot\sin(\varphi_1)-\frac{d}{m}\varphi_2$$

With measured lenght *l* and mass *m*, the damping parameter *d* is to be identified with zero crossing data of the angle (easy to measure). Measurements available for initial angle φ_0 in fractions of π.

Identification is satisfying for arbitrary meaningful initial angle φ_0, but the amplitude of the oscillation might cause problems.

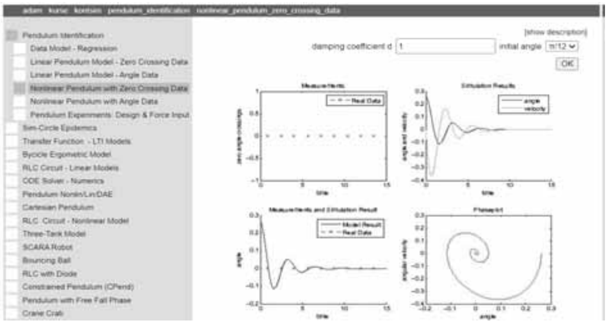
[hide description]

damping coefficient d: 1 initial angle: π/12

OK

Figure 18: MMT Course Example.

The availability of the source code is an important feature of the MMT system: students can use the code for further development. In the centre information on the selected example is shown, and parameters for experiments can be chosen. With a click on the "OK" button, the server computes the results with the chosen parameters (Figure 19).


adam kurse kontsim pendulum_identification nonlinear_pendulum_zero_crossing_data
Pendulum Identification
Data Model - Regression
Linear Pendulum Model - Zero Crossing Data
Linear Pendulum Model - Angle Data
Nonlinear Pendulum with Zero Crossing Data
Nonlinear Pendulum with Angle Data
Pendulum Experiments: Design & Force Input
Sim-Circle Epidemics
Transfer Function - LTI Models
Bycicle Ergometric Model
RLC Circuit - Linear Models
ODE Solver - Numerics
Pendulum Nonlin/Lin/DAE
Cartesian Pendulum
RLC Circuit - Nonlinear Model
Three-Tank Model
SCARA Robot
Bouncing Ball
RLC with Diode
Constrained Pendulum (CPend)
Pendulum with Free Fall Phase
Crane Crab
[show description]
OK
Measurements
Real Data
time
Simulation Results
angle
velocity
angle and velocity
time
Measurements and Simulation Result
Model Result
Real Data
angle
time
Phaseplot
angular velocity
angle

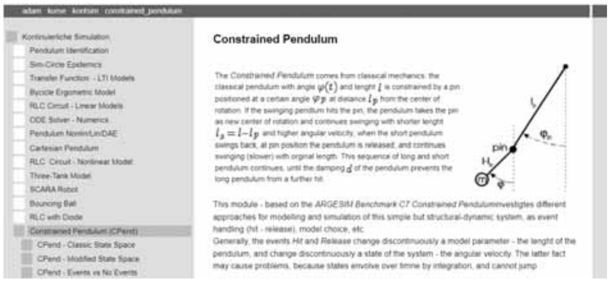
damping coefficient d: 1 initial angle: π/12

Figure 19: MMT Course Example – Results.

There are various aspects for choosing examples and case studies for the MMT server: modelling topics, applications, methods, etc. The *ARGESIM Benchmarks for Modelling Approaches and Simulation Implementations* are a challenging mixture of modelling approaches and application, and have therefore become also a basis for education in modelling and simulation ([6]), using the various solutions published in SNE. Consequently the benchmarks are also interesting topics as case studies for the MMT server, and the MMT development team has started to integrate the some benchmarks into the MMT server, taking a well-elaborated MATLAB approach with the defined tasks as examples, and extending them by further topics and tasks.

After C9 *Two-Tank Fuzzy Control*, C11 *SCARA Robot*, C12 *Collision of Spheres*, C13 *Crane with Control*, C15 *Kidney Clearance Identification*, and C17 *SIR-type Epidemics*, now C7 *Constrained Pendulum* is integrated.

The C7 integration into the MMT server (Figure 20, MMT introduction page for Constrained Pendulum) extends the benchmark tasks by topics presented in this contribution: waiving event detection, event handling methods, linear system cases, linear analytic and symbolic solutions, sensitivity analysis, and Monte-Carlo sensitivity. Additionally, the team prepares as further topic the approximation of the nonlinear model by a sequence of linear affine models with adaptive linearization points.

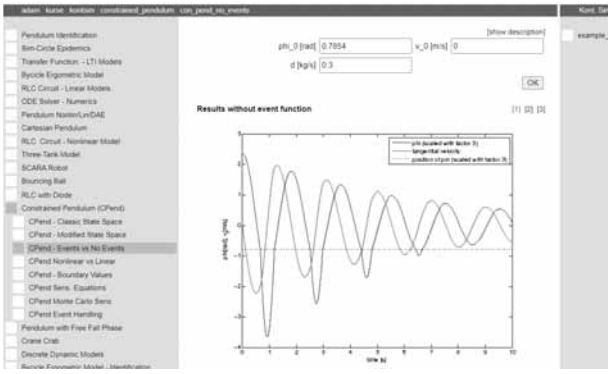

adam kurse kontsim constrained_pendulum
Kontinuierliche Simulation
Pendulum Identification
Sim-Circle Epidemics
Transfer Function - LTI Models
Bycicle Ergometric Model
RLC Circuit - Linear Models
ODE Solver - Numerics
Pendulum Nonlin/Lin/DAE
Cartesian Pendulum
RLC Circuit - Nonlinear Model
Three-Tank Model
SCARA Robot
Bouncing Ball
RLC with Diode
Constrained Pendulum (CPend)
CPend - Classic State Space
CPend - Modified State Space
CPend - Events vs No Events
l_s
φ_p
pin
l_p
m
φ

Constrained Pendulum

The *Constrained Pendulum* comes from classical mechanics: the classical pendulum with angle $\varphi(t)$ and lenght l is constrained by a pin positioned at a certain angle φ_p at distance l_p from the center of rotation. If the swinging pendulum hits the pin, the pendulum takes the pin as new center of rotation and continues swinging with shorter lenght $l_s=l-l_p$ and higher angular velocity; when the short pendulum swings back, at pin position the pendulum is released, and continues swinging (slower) with orginal length. This sequence of long and short pendulum continues, until the damping d of the pendulum prevents the long pendulum from a further hit.

This module - based on the *ARGESIM Benchmark C7 Constrained Pendulum*investigtes different approaches for modelling and simulation of this simple but structural-dynamic system, as event handling (hit - release), model choice, etc.
Generally, the events *Hit* and *Release* change discontinuously a model parameter - the lenght of the pendulum, and change discontinuously a state of the system - the angular velocity. The latter fact may cause problems, because states envolve over timne by integration, and cannot jump

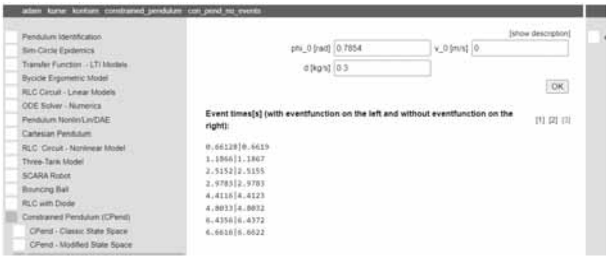
Figure 20: MMT Case Study 'Constrained Pendulum'.


adam kurse kontsim constrained_pendulum con_pend_no_events
Pendulum Identification
Sim-Circle Epidemics
Transfer Function - LTI Models
Bycicle Ergometric Model
RLC Circuit - Linear Models
ODE Solver - Numerics
Pendulum Nonlin/Lin/DAE
Cartesian Pendulum
RLC Circuit - Nonlinear Model
Three-Tank Model
SCARA Robot
Bouncing Ball
RLC with Diode
Constrained Pendulum (CPend)
CPend - Classic State Space
CPend - Modified State Space
CPend - Events vs No Events
CPend Nonlinear vs Linear
CPend - Boundary Values
CPend Sens. Equations
CPend Monte Carlo Sens
CPend Event Handling
Pendulum with Free Fall Phase
Crane Crab
Discrete Dynamic Models
example_
[show description]
OK
[1] [2] [3]
phi (scaled with factor 3)
tangential velocity
position of pin (scaled with factor 3)
time [s]

phi_0 [rad]: 0.7854 v_0 [m/s]: 0
d [kg/s]: 0.3

Results without event function

Figure 21: MMT 'Constrained Pendulum' – Results 1.


adam kurse kontsim constrained_pendulum con_pend_no_events
Pendulum Identification
Sim-Circle Epidemics
Transfer Function - LTI Models
Bycicle Ergometric Model
RLC Circuit - Linear Models
ODE Solver - Numerics
Pendulum Nonlin/Lin/DAE
Cartesian Pendulum
RLC Circuit - Nonlinear Model
Three-Tank Model
SCARA Robot
Bouncing Ball
RLC with Diode
Constrained Pendulum (CPend)
CPend - Classic State Space
CPend - Modified State Space
[show description]
OK
[1] [2] [3]

phi_0 [rad]: 0.7854 v_0 [m/s]: 0
d [kg/s]: 0.3

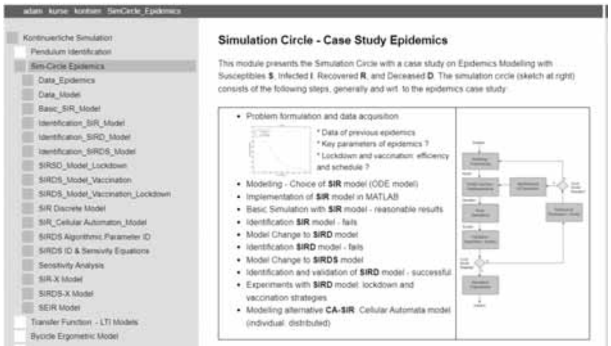
Event times[s] (with eventfunction on the left and without eventfunction on the right):

```
0.66128|0.6619
1.1866|1.1867
2.5152|2.5155
2.9783|2.9783
4.4116|4.4123
4.8033|4.8032
6.4356|6.4372
6.6616|6.6622
```

Figure 22: MMT 'Constrained Pendulum' – Results 2.

Choosing for instance the experiment 'CPend – Events vs. No Events' offers to enter model parameters. Pressing 'OK', the MATLAB engine runs, and gives back various results, as e.g. the time courses (Figure 21), or the event times (Figure 22).

At present the MMT Server offers about 200 case studies, each with various detailed examples. In winter 2021 the MMT has also extended the already integrated C17 *SIR-type Epidemic* by model identification, and lockdown and vaccination strategies based on Corona epidemics data from Austria (Figure 23).

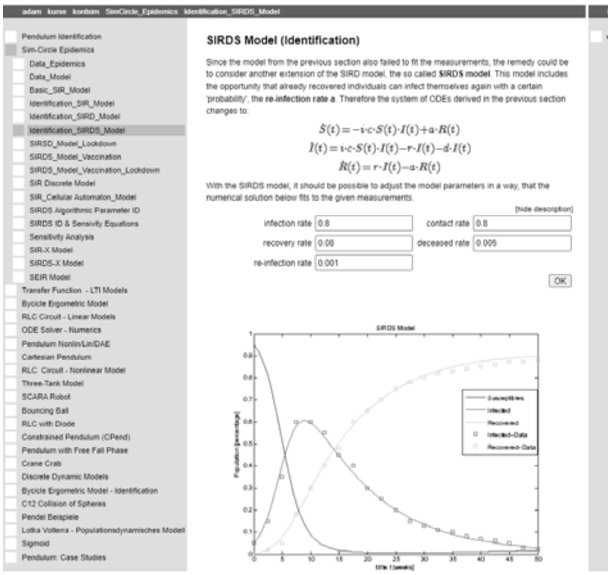

adam kurse kontsim SimCircle_Epidemics
Kontinuerliche Simulation
Pendulum Identification
Sim-Circle Epidemics
Data_Epidemics
Data_Model
Basic_SIR_Model
Identification_SIR_Model
Identification_SIRD_Model
Identification_SIRDS_Model
SIRSD_Model_Lockdown
SIRDS_Model_Vaccination
SIRDS_Model_Vaccination_Lockdown
SIR Discrete Model
SIR_Cellular Automaton_Model
SIRDS Algorithmic Parameter ID
SIRDS ID & Sensivity Equations
Sensitivity Analysis
SIR-X Model
SIRDS-X Model
SEIR Model
Transfer Function - LTI Models
Bycicle Ergometric Model

Simulation Circle - Case Study Epidemics

This module presents the Simulation Circle with a case study on Epidemics Modeling with Susceptibles **S**, Infected **I**, Recovered **R**, and Deceased **D**. The simulation circle (sketch at right) consists of the following steps, generally and wrt. to the epidemics case study:

- Problem formulation and data acquisition
  * Data of previous epidemics
  * Key parameters of epidemics ?
  * Lockdown and vaccination: efficiency and schedule ?
- Modeling - Choice of **SIR** model (ODE model)
- Implementation of **SIR** model in MATLAB
- Basic Simulation with **SIR** model - reasonable results
- Identification **SIR** model - fails
- Model Change to **SIRD** model
- Identification **SIRD** model - fails
- Model Change to **SIRDS** model
- Identification and validation of **SIRD** model - successful
- Experiments with **SIRD** model: lockdown and vaccination strategies
- Modeling alternative **CA-SIR**: Cellular Automata model (individual: distributed)

Figure 23: MMT SIR Case Study: Spread of Infection.



SIRDS Model (Identification)

Since the model from the previous section also failed to fit the measurements, the remedy could be to consider another extension of the SIRD model, the so called **SIRDS model**. This model includes the opportunity that already recovered individuals can infect themselves again with a certain 'probability', the **re-infection rate a**. Therefore the system of ODEs derived in the previous section changes to:

$$\dot{S}(t) = -i\cdot c\cdot S(t)\cdot I(t) + a\cdot R(t)$$
$$\dot{I}(t) = i\cdot c\cdot S(t)\cdot I(t) - r\cdot I(t) - d\cdot I(t)$$
$$\dot{R}(t) = r\cdot I(t) - a\cdot R(t)$$

With the SIRDS model, it should be possible to adjust the model parameters in a way, that the numerical solution below fits to the given measurements.

[hide description]

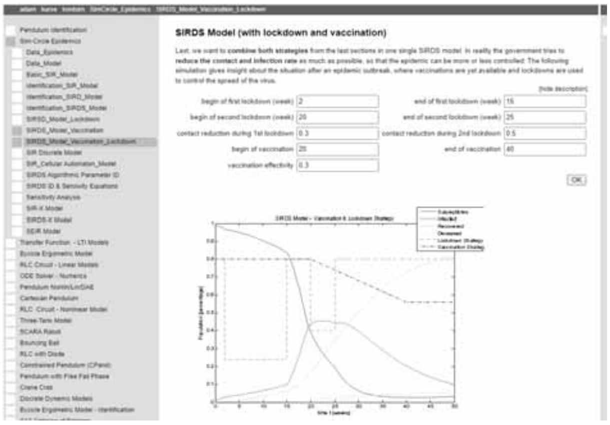
infection rate 0.8 contact rate 0.8

recovery rate 0.08 deceased rate 0.005

re-infection rate 0.001

OK

Figure 24: SIR Case Study – Identification.



SIRDS Model (with lockdown and vaccination)

Last, we want to **combine both strategies** from the last sections in one single SIRDS model. In reality the government tries to **reduce the contact and infection rate** as much as possible, so that the epidemic can be more or less controlled. The following simulation gives insight about the situation after an epidemic outbreak, where vaccinations are yet available and lockdowns are used to control the spread of the virus.

[hide description]

begin of first lockdown (week) 2 end of first lockdown (week) 15

begin of second lockdown (week) 20 end of second lockdown (week) 25

contact reduction during 1st lockdown 0.3 contact reduction during 2nd lockdown 0.5

begin of vaccination 20 end of vaccination 40

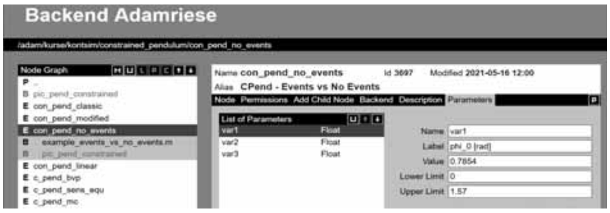
vaccination effectivity 0.3

OK

Figure 25: SIR Case Study – Lockdown and Vaccination.

In some simulation courses, students could investigate with the MMT Server strategies again pandemics (Figure 24 and Figure 25).

And last but not least, a view into the MMT backend with the model interface page for the constrained pendulum (Figure 26). Novices need about one week to learn how to work and develop in this backend.



/adam/kurse/kontsim/constrained_pendulum/con_pend_no_events

Name con_pend_no_events Id 3697 Modified 2021-05-16 12:00

Alias CPend - Events vs No Events

var1	Float
var2	Float
var3	Float

Name var1

Label phi_0 [rad]

Value 0.7854

Lower Limit 0

Upper Limit 1.57

Figure 26: MMT Case Study 'Constrained Pendulum' – Backend with parameter interface.

Acknowledgement. The development of new MMT case studies is also done by students themselves – they present their seminar work, their practical course, or parts of their bachelor work or diploma work also in the MMT server – to be used by other students. This contribution on the benchmark *Constrained Pendulum* with all extensions and the MMT integration is result of a student project work from the lecture 'Continuous Simulation', composed by four students from mechatronics, and by the supervising tutor and lecturer, who added theoretical topics and MMT preparation.

References

[1] Breitenecker F. Comparison 7: Constrained Pendulum, Definition. SNE 3(7), 1993, 29

[2] Körner A, Breitenecker F. State Events and Structural-dynamic Systems: Definition of ARGESIM Benchmark C21. Simulation Notes Europe SNE 26(2), 2016, 117 – 122. DOI: 10.11128/sne.26.bn21.10339

[3] Stockinger AE, Gütl E, Rath S, Strasser D, Bicher M, Körner A, Ecker H. Direct Implementation of ARGESIM Benchmark C7 'Constrained Pendulum' in MATLAB and EXCEL. SNE 29(2), 2019, 105-110. DOI: 10.11128/sne.29.bne07.10478

[4] Moler C, Van Loan C. Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later. SIAM Review 45(1):3-49. DOI: 10.1137/S00361445024180

[5] Winkler S, Körner A, Popper N. MMT – Mathematics, Modelling and Tools: An E-Learning Environment for Modelling and Simulation. SNE 21(2), 2011, 99-102. DOI: 10.11128/sne.21.en.10069

[6] Breitenecker F, Körner A, Ecker H, Popper N, Pawletta T. ARGESIM Benchmarks on Modelling Approaches and Simulation Implementations – Development, Classification and Basis for Simulation Education. SNE 29(1), 2019, 49-61. DOI: 10.11128/sne.29.bn.10468

SNE Simulation News

EUROSIM Data and Quick Info

VESS – Virtual EUROSIM Seminar

Virtual Simulation Presentations, since June 2020 www.eurosim2023.eu

MATHMOD Vienna 2022

July 27-29, 2022, Vienna, Austria www.mathmod.at

ASIM 2022 - 26.Symposium Simulation Technique

July 25-27, 2022, Vienna, Austria www.asim-gi.org/asim2022

EUROSIM CONGRESS 2023

Summer 2023, Amsterdam, The Netherlands www.eurosim2023.eu

Contents

Simulation Notes Europe SNE is the official membership journal of **EUROSIM** and distributed / available to members of the **EUROSIM Societies** as part of the membership benefits.

If you have any information, announcement, etc. you want to see published, please contact a member of the editorial board in your country or the editorial office. For scientific publications, please contact the EiC.

This *EUROSIM Data & Quick Info* compiles data from **EUROSIM** societies and groups: addresses, weblinks, and officers of societies with function and email, to be published regularly in **SNE** issues. This information is also published at **EUROSIM**'s website *www.eurosim.info*.

SNE Reports Editorial Board

SNE Editorial Office /ARGESIM

→ *www.sne-journal.org, www.eurosim.info*
✉ *office@sne-journal.org, eic@sne-journal.org*
✉ SNE Editorial Office
Johannes Tanzler (Layout, Organisation)
Irmgard Husinsky (Web, Electronic Publishing)
Felix Breitenecker EiC (Organisation, Authors)
ARGESIM/Math. Modelling & Simulation Group,
Inst. of Analysis and Scientific Computing, TU Wien
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria

EUROSIM
Federation of European Simulation Societies

General Information. **EUROSIM**, the Federation of European Simulation Societies, was set up in 1989. The purpose of **EUROSIM** is to provide a European forum for simulation societies and groups to promote modelling and simulation in industry, research, and development – by publication and conferences. → *www.eurosim.info*

Member Societies. **EUROSIM** members may be national simulation societies and regional or international societies and groups dealing with modelling and simulation. At present **EUROSIM** has *Full Members* and *Observer Members* (*), and *Member Candidates* (**).

ASIM	Arbeitsgemeinschaft Simulation *Austria, Germany, Switzerland*
CEA-SMSG	Spanish Modelling and Simulation Group; *Spain*
CSSS	Czech and Slovak Simulation Society *Czech Republic, Slovak Republic*
DBSS	Dutch Benelux Simulation Society *Belgium, Netherlands*
KA-SIM	Kosovo Simulation Society, *Kosovo*
LIOPHANT	LIOPHANT Simulation Club; *Italy & International*
LSS	Latvian Simulation Society; *Latvia*
PSCS	Polish Society for Computer Simulation; *Poland*
NSSM	Russian National Simulation Society *Russian Federation*
SIMS	Simulation Society of Scandinavia *Denmark, Finland, Norway, Sweden*
SLOSIM	Slovenian Simulation Society; *Slovenia*
UKSIM	United Kingdom Simulation Society *UK, Ireland*
ALBSIM	Albanian Simulation Society*; *Albania*
ROMSIM	Romanian Society for Modelling and Simulation*; *Romania*
Societies in Re-Organisation:	
CROSSIM	*Croatian Society f. Simulation Modeling;* Croatia
FRANCOSIM	*Société Francophone de Simulation* *Belgium, France*
HSS	*Hungarian Simulation Society;* Hungary
ISCS	*Italian Society for Computer Simulation, Italy*

EUROSIM Board / Officers. **EUROSIM** is governed by a board consisting of one representative of each member society, and president, past president, and **SNE** representative. The President is nominated by the society organising the next **EUROSIM** Congress. Secretary, and Treasurer are elected out of members of the board.

President	M. Mujica Mota (DBSS), *m.mujica.mota@hva.nl*
Past President	Emilio Jiménez (CAE-SMSG), *emilio.jimenez@unirioja.es*
Secretary	Niki Popper, *niki.popper@dwh.at*
Treasurer	Felix Breitenecker (ASIM) *felix.breitenecker@tuwien.ac.at*
Webmaster	Irmgard Husinsky, *irmgard.husinsky@tuwien.ac.at*
SNE Editor	F. Breitenecker, *eic@sne-journal.org*

SNE – Simulation Notes Europe. **SNE** is **EUROSIM**'s scientific journal with peer reviewed contributions as well as a membership journal for **EUROSIM** with information from the societies. **EUROSIM** societies distribute **SNE** (electronic or printed) to their members as official membership journal. **SNE** Publishers are **EUROSIM**, **ARGESIM** and **ASIM**.

SNE Editor-in-Chief	Felix Breitenecker *eic@sne-journal.org*

→ *www.sne-journal.org*, ✉ *office@sne-journal.org*

EUROSIM Congress and Conferences.

Each year a major **EUROSIM** event takes place, the **EUROSIM CONGRESS** organised by a member society, **SIMS EUROSIM Conference**, and **MATHMOD Vienna Conference (ASIM)**.

EUROSIM Congress 2019, the *10th EUROSIM Congress*, was organised by **CEA-SMSG**, the Spanish Simulation Society, in La Rioja, Logroño, Spain, July 1-5, 2019;

Due to Covid-19 virus in 2020 no **EUROSIM** events take place. To bridge this gap, **EUROSIM** is organising the series **VESS - Virtual EUROSIM Simulation Seminar** – seminars by simulation professionalists (2 hours via web), in preparation for upcoming **EUROSIM** events. → ***www.eurosim2023.eu***

Next main event is **MATHMOD Vienna**. This triennial **EUROSIM Conference** is mainly organized by **ASIM**, the German simulation society, and **ARGESIM,** with main cosponsor **IFAC**.

MATHMOD 2022, the *10th MATHMOD Vienna Conference on Mathematical Modelling* will take place in Vienna, July 27-29, 2022. → ***www.mathmod.at***

EUROSIM Congress 2023, the *11th EUROSIM Congress*, will be organised by **DBSS**, the Dutch Benelux simulation society, in Amsterdam, Summer 2023.
→ ***www.eurosim2023.eu***

Furthermore, **EUROSIM Societies** organize also local conferences, and **EUROSIM** co-operates with the organizers of the **I3M Conference Series.**
→ ***www.liophant.org/conferences/***

EUROSIM Member Societies

ASIM
German Simulation Society
Arbeitsgemeinschaft Simulation

ASIM (Arbeitsgemeinschaft Simulation) is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria. **ASIM** was founded in 1981 and has now about 400 individual members (including associated), and 90 institutional or industrial members.

→ *www.asim-gi.org* with members' area

✉ *info@asim-gi.org, admin@asim-gi.org*

🖂 ASIM – Inst. of Analysis and Scientific Computing
Vienna University of Technology (TU Wien)
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria

ASIM Officers	
President	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*
Vice presidents	Sigrid Wenzel, *s.wenzel@uni-kassel.de*
	T. Pawletta, *thorsten.pawletta@hs-wismar.de*
	A. Körner, *andreas.koerner@tuwien.ac.at*
Secretary	Ch. Deatcu, *christina.deatcu@hs-wismar.de*
	I. Husinsky, *Irmgard.husinsky@tuwien.ac.at*
Membership Affairs	S. Wenzel, *s.wenzel@uni-kassel.de*
	Ch. Deatcu, *christina.deatcu@hs-wismar.de*
	F. Breitenecker, *felix.breitenecker@tuwien.ac.at*
Repr. EUROSIM	F. Breitenecker, *felix.breitenecker@tuwien.ac.at*
	A. Körner, *andreas.koerner@tuwien.ac.at*
Internat. Affairs – GI Contact	O. Rose, *Oliver.Rose@tu-dresden.de*
	N. Popper, *niki.popper@dwh.at*
Editorial Board SNE	T. Pawletta, *thorsten.pawletta@hs-wismar.de*
	Ch. Deatcu, *christina.deatcu@hs-wismar.de*
Web EUROSIM	I. Husinsky, *Irmgard.husinsky@tuwien.ac.at*

Last data update A*pril 2020*

ASIM is organising / co-organising the following international conferences:

- **ASIM** Int. Conference 'Simulation in Production and Logistics' – biannual
- **ASIM** 'Symposium Simulation Technique' – biannual
- **MATHMOD** Int. Vienna Conference on Mathmatical Modelling – triennial

Furthermore, **ASIM** is co-sponsor of *WSC* - Winter Simulation Conference, of SCS conferences *SpringSim* and *SummerSim*, and of *I3M* and *Simutech* conference series.

ASIM Working Committees	
GMMS	Methods in Modelling and Simulation Th. Pawletta, *thorsten.pawletta@hs-wismar.de*
SUG	Simulation in Environmental Systems Jochen Wittmann, *wittmann@informatik.uni-hamburg.de*
STS	Simulation of Technical Systems Walter Commerell, *commerell@hs-ulm.de*
SPL	Simulation in Production and Logistics Sigrid Wenzel, *s.wenzel@uni-kassel.de*
EDU	Simulation in Education/Education in Simulation A. Körner, *andreas.koerner@tuwien.ac.at*
BIG DATA	Working Group Data-driven Simulation in Life Sciences; *niki.popper@dwh.at*
WORKING GROUPS	Simulation in Business Administration, in Traffic Systems, for Standardisation, etc.

CEA-SMSG – Spanish Modelling and Simulation Group

CEA is the Spanish Society on Automation and Control and it is the national member of IFAC (International Federation of Automatic Control) in Spain. Since 1968 **CEA-IFAC** looks after the development of the Automation in Spain, in its different issues: automatic control, robotics, *SIMULATION*, etc. The association is divided into national thematic groups, one of which is centered on Modeling, Simulation and Optimization, constituting the **CEA** Spanish Modeling and Simulation Group (**CEA-SMSG**). It looks after the development of the Modelling and Simulation (M&S) in Spain, working basically on all the issues concerning the use of M&S techniques as essential engineering tools for decision-making and optimization.

→ *http://www.ceautomatica.es/grupos/*

→ *emilio.jimenez@unirioja.es*
simulacion@cea-ifac.es

🖂 CEA-SMSG / Emilio Jiménez, Department of Electrical Engineering, University of La Rioja, San José de Calasanz 31, 26004 Logroño (La Rioja), SPAIN

CEA - SMSG Officers	
President	Emilio Jiménez, *emilio.jimenez@unirioja.es*
Vice president	Juan Ignacio Latorre, *juanignacio.latorre@unavarra.es*
Repr. EUROSIM	Emilio Jiménez, *emilio.jimenez@unirioja.es*
Edit. Board SNE	Juan Ignacio Latorre, *juanignacio.latorre@unavarra.es*
Web EUROSIM	Mercedes Perez *mercedes.perez@unirioja.es*

Last data update February 2018

CSSS – Czech and Slovak Simulation Society

CSSS -The *Czech and Slovak Simulation Society* has about 150 members working in Czech and Slovak national scientific and technical societies (*Czech Society for Applied Cybernetics and Informatics*, *Slovak Society for Applied Cybernetics and Informatics*). **CSSS** main objectives are: development of education and training in the field of modelling and simulation, organising professional workshops and conferences, disseminating information about modelling and simulation activities in Europe. Since 1992, **CSSS** is full member of **EUROSIM**.

→ *www.fit.vutbr.cz/CSSS*

📧 snorek@fel.cvut.cz

✉ CSSS / Miroslav Šnorek, CTU Prague
FEE, Dept. Computer Science and Engineering,
Karlovo nam. 13, 121 35 Praha 2, Czech Republic

CSSS Officers	
President	Miroslav Šnorek, *snorek@fel.cvut.cz*
Vice president	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Scientific Secr.	A. Kavička, *Antonin.Kavicka@upce.cz*
Repr. EUROSIM	Miroslav Šnorek, *snorek@fel.cvut.cz*
Edit. Board SNE	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Web EUROSIM	Petr Peringer, peringer@fit.vutbr.cz

Last data update December 2012

DBSS – Dutch Benelux Simulation Society

The *Dutch Benelux Simulation Society* (**DBSS**) was founded in July 1986 in order to create an organisation of simulation professionals within the Dutch language area. **DBSS** has actively promoted creation of similar organisations in other language areas. **DBSS** is a member of **EUROSIM** and works in close cooperation with its members and with affiliated societies.

→ *www.DutchBSS.org*

📧 *a.w.heemink@its.tudelft.nl*

✉ DBSS / A. W. Heemink
Delft University of Technology, ITS - twi,
Mekelweg 4, 2628 CD Delft, The Netherlands

DBSS Officers	
President	M. Mujica Mota, *m.mujica.mota@hva.nl*
Vice president	A. Heemink, *a.w.heemink@its.tudelft.nl*
Treasurer	A. Heemink, *a.w.heemink@its.tudelft.nl*
Secretary	P. M. Scala, p.m.scala@hva.nl
Repr. EUROSIM	M. Mujica Mota, *m.mujica.mota@hva.nl*
Edit. SNE/Web	M. Mujica Mota, *m.mujica.mota@hva.nl*

Last data update June 2016

LIOPHANT Simulation

Liophant Simulation is a non-profit association born in order to be a trait-d'union among simulation developers and users; Liophant is devoted to promote and diffuse the simulation techniques and methodologies; the Association promotes exchange of students, sabbatical years, organization of International Conferences, courses and internships focused on M&S applications.

→ *www.liophant.org*

📧 *info@liophant.org*

✉ LIOPHANT Simulation, c/o Agostino G. Bruzzone,
DIME, University of Genoa, Savona Campus
via Molinero 1, 17100 Savona (SV), Italy

LIOPHANT Officers	
President	A.G. Bruzzone, *agostino@itim.unige.it*
Director	E. Bocca, *enrico.bocca@liophant.org*
Secretary	A. Devoti, *devoti.a@iveco.com*
Treasurer	Marina Massei, *massei@itim.unige.it*
Repr. EUROSIM	A.G. Bruzzone, *agostino@itim.unige.it*
Deputy	F. Longo, *f.longo@unical.it*
Edit. Board SNE	F. Longo, *f.longo@unical.it*
Web EUROSIM	F. Longo, *f.longo@unical.it*

Last data update June 2016

LSS – Latvian Simulation Society

The Latvian Simulation Society (LSS) has been founded in 1990 as the first professional simulation organisation in the field of Modelling and simulation in the post-Soviet area. Its members represent the main simulation centres in Latvia, including both academic and industrial sectors.

→ *www.itl.rtu.lv/imb/*

📧 *Egils.Ginters@rtu.lv*

✉ Prof. Egils Ginters, Kirshu Str.13A, Cesis LV-4101,
Latvia

LSS Officers	
President	Yuri Merkuryev, *merkur@itl.rtu.lv*
Vice President	Egils Ginters, *egils.ginters@rtu.lv*
Secretary	Artis Teilans, *artis.teilans@rta.lv*
Repr. EUROSIM	Egils Ginters, egils.ginters@rtu.lv
Deputy	Artis Teilans, *artis.teilans@rta.lv*
Edit. Board SNE	Juri Tolujew, *Juri.Tolujew@iff.fraunhofer.de*
Web EUROSIM	Vitaly Bolshakov, *vitalijs.bolsakovs@rtu.lv*

Last data update November 2020

KA-SIM Kosovo Simulation Society

Kosova Association for Modeling and Simulation (KA-SIM, founded in 2009), is part of Kosova Association of Control, Automation and Systems Engineering (KA-CASE). KA-CASE was registered in 2006 as non Profit Organization and since 2009 is National Member of IFAC – International Federation of Automatic Control. KA-SIM joined EUROSIM as Observer Member in 2011. In 2016, KA-SIM became full member.

KA-SIM has about 50 members, and is organizing the international conference series International Conference in Business, Technology and Innovation, in November, in Durrhes, Albania, and IFAC Simulation Workshops in Pristina.

→ *www.ubt-uni.net/ka-case*

✉ *ehajrizi@ubt-uni.net*

⌧ MOD&SIM KA-CASE; Att. Dr. Edmond Hajrizi
Univ. for Business and Technology (UBT)
Lagjja Kalabria p.n., 10000 Prishtina, Kosovo

KA-SIM Officers	
President	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Vice president	Muzafer Shala, *info@ka-sim.com*
Secretary	Lulzim Beqiri, *info@ka-sim.com*
Treasurer	Selman Berisha, *info@ka-sim.com*
Repr. EuroSim	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Deputy	Muzafer Shala, *info@ka-sim.com*
Edit. Board SNE	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Web EuroSim	Betim Gashi, *info@ka-sim.com*

Last data update December 2016

NSSM – National Society for Simulation Modelling (Russia)

NSSM - The Russian National Simulation Society (Национальное Общество Имитационного Моделирования – НОИМ) was officially registered in Russian Federation on February 11, 2011. In February 2012 NSS has been accepted as an observer member of **EUROSIM**, and in 2015 NSSM has become full member.

→ *www.simulation.su*

✉ *yusupov@iias.spb.su*

⌧ NSSM / R. M. Yusupov,
St. Petersburg Institute of Informatics and Automation
RAS, 199178, St. Petersburg, 14th lin. V.O, 39

NSSM Officers	
President	R. M. Yusupov, *yusupov@iias.spb.su*
Chair Man. Board	A. Plotnikov, *plotnikov@sstc.spb.ru*
Secretary	M. Dolmatov, *dolmatov@simulation.su*
Repr. EuroSim	R.M. Yusupov, *yusupov@iias.spb.su* Y. Senichenkov, *senyb@dcn.icc.spbstu.ru*
Deputy	B. Sokolov, *sokol@iias.spb.su*
Edit. Board SNE	Y. Senichenkov, *senyb@mail.ru*, *senyb@dcn.icc.spbstu.ru*,

Last data update February 2018

PSCS – Polish Society for Computer Simulation

PSCS was founded in 1993 in Warsaw. PSCS is a scientific, non-profit association of members from universities, research institutes and industry in Poland with common interests in variety of methods of computer simulations and its applications. At present PSCS counts 257 members.

→ *www.eurosim.info*, www.ptsk.pl/

✉ *leon@ibib.waw.pl*

⌧ PSCS / Leon Bobrowski, c/o IBIB PAN,
ul. Trojdena 4 (p.416), 02-109 Warszawa, Poland

PSCS Officers	
President	Leon Bobrowski, *leon@ibib.waw.pl*
Vice president	Tadeusz Nowicki, *Tadeusz.Nowicki@wat.edu.pl*
Treasurer	Z. Sosnowski, *zenon@ii.pb.bialystok.pl*
Secretary	Zdzislaw Galkowski, *Zdzislaw.Galkowski@simr.pw.edu.pl*
Repr. EuroSim	Leon Bobrowski, *leon@ibib.waw.pl*
Deputy	Tadeusz Nowicki, *tadeusz.nowicki@wat.edu.pl*
Edit. Board SNE	Zenon Sosnowski, z.sosnowski@pb.ed.pl
Web EuroSim	Magdalena Topczewska *m.topczewska@pb.edu.pl*

Last data update December2013

SIMS – Scandinavian Simulation Society

SIMS is the *Scandinavian Simulation Society* with members from the five Nordic countries Denmark, Finland, Iceland, Norway and Sweden. The SIMS history goes back to 1959. SIMS practical matters are taken care of by the SIMS board consisting of two representatives from each Nordic country (Iceland one board member).

SIMS Structure. SIMS is organised as federation of regional societies. There are **FinSim** (Finnish Simulation Forum), **MoSis** (Society for Modelling and Simulation in Sweden), **DKSIM** (Dansk Simuleringsforening) and **NFA** (Norsk Forening for Automatisering).

→ *www.scansims.org*
🖃 *bernt.lie@usn.no*
✉ SIMS / Bernt Lie, Faculty of Technology, Univ.College of Southeast Norway, Department of Technology, Kjølnes ring 56, 3914 Porsgrunn, Norway

SIMS Officers	
President	Bernt Lie, *Bernt.Lie@usn.no*
Vice president	Erik Dahlquist, *erik.dahlquist@mdh.se*
Treasurer	Vadim Engelson, *vadime@mathcore.com*
Repr. EUROSIM	Esko Juuso, *esko.juuso@oulu.fi*
Edit. Board SNE	Esko Juuso, *esko.juuso@oulu.fi*
Web EUROSIM	Vadim Engelson, *vadime@mathcore.com*

Last data update February 2020

SLOSIM – Slovenian Society for Simulation and Modelling

SLOSIM - Slovenian Society for Simulation and Modelling was established in 1994 and became the full member of **EUROSIM** in 1996. Currently it has 90 members from both Slovenian universities, institutes, and industry. It promotes modelling and simulation approaches to problem solving in industrial as well as in academic environments by establishing communication and cooperation among corresponding teams.

→ www.slosim.si
🖃 *slosim@fe.uni-lj.si*
✉ SLOSIM / Vito Logar, Faculty of Electrical
Engineering, University of Ljubljana,
Tržaška 25, 1000 Ljubljana, Slovenia

SLOSIM Officers	
President	Vito Logar, *vito.logar@fe.uni-lj.si*
Vice president	Božidar Šarler, *bozidar.sarler@ung.si*
Secretary	Simon Tomažič, *simon.tomazic@fe.uni-lj.si*
Treasurer	Milan Simčič, *milan.simcic@fe.uni-lj.si*
Repr. EUROSIM	B. Zupančič, *borut.zupancic@fe.uni-lj.si*
Deputy	Vito Logar, *vito.logar@fe.uni-lj.si*
Edit. Board SNE	R. Karba, *rihard.karba@fe.uni-lj.si*
Web EUROSIM	Vito Logar, *vito.logar@fe.uni-lj.si*

Last data update December 2018

UKSIM - United Kingdom Simulation Society

The UK Simulation Society is very active in organizing conferences, meetings and workshops. UKSim holds its annual conference in the March-April period. In recent years the conference has always been held at Emmanuel College, Cambridge. The Asia Modelling and Simulation Section (AMSS) of UKSim holds 4-5 conferences per year including the EMS (European Modelling Symposium), an event mainly aimed at young researchers, organized each year by UKSim in different European cities. Membership of the UK Simulation Society is free to participants of any of our conferences and their co-authors.

→*uksim.info*
🖃 *david.al-dabass@ntu.ac.uk*

✉ **UKSIM / Prof. David Al-Dabass**
Computing & Informatics,
Nottingham Trent University
Clifton lane, Nottingham, NG11 8NS, United KingdomUKSIM Officers

President	David Al-Dabass, *david.al-dabass@ntu.ac.uk*
Secretary	T. Bashford, *tim.bashford@uwtsd.ac.uk*
Treasurer	D. Al-Dabass, *david.al-dabass@ntu.ac.uk*
Membership chair	G. Jenkins, *glenn.l.jenkins@smu.ac.uk*
Local/Venue chair	Richard Cant, *richard.cant@ntu.ac.uk*
Repr. EUROSIM	Dr Taha Osman, taha.osman@ntu.ac.uk
Deputy	T. Bashford, *tim.bashford@uwtsd.ac.uk*
Edit. Board SNE	D. Al-Dabass, *david.al-dabass@ntu.ac.uk*

Last data update March 2020

EUROSIM Observer Members

ROMSIM – Romanian Modelling and Simulation Society

ROMSIM has been founded in 1990 as a non-profit society, devoted to theoretical and applied aspects of modelling and simulation of systems.

→ *www.eurosim.info/societies/romsim/*

📧 *florin_h2004@yahoo.com*

✉ ROMSIM / Florin Hartescu,
National Institute for Research in Informatics, Averescu
Av. 8 – 10, 011455 Bucharest, Romania

ROMSIM Officers	
President	N. N.
Vice president	Florin Hartescu, *florin_h2004@yahoo.com* Marius Radulescu, *mradulescu.csmro@yahoo.com*
Repr. EuroSim	Marius Radulescu
Deputy	Florin Hartescu
Edit. Board SNE	Constanta Zoe Radulescu, *zoe@ici.ro*
Web EuroSim	Florin Hartescu

Last data update June 2019

ALBSIM – Albanian Simulation Society

The Albanian Simulation Society has been initiated at the Department of Statistics and Applied Informatics, Faculty of Economy at the University of Tirana, by Prof. Dr. Kozeta Sevrani. The society is involved in different international and local simulation projects, and is engaged in the organisation of the conference series ISTI - Information Systems and Technology. In July 2019 the society was accepted as EUROSIM Observer Member.

→ *www.eurosim.info/societies/albsim/*

📧 *kozeta.sevrani@unitir.edu.al*

✉ Albanian Simulation Goup, attn. Kozeta Sevrani
University of Tirana, Faculty of Economy
rr. Elbasanit, Tirana 355 Albania

Albanian Simulation Society- Officers	
Chairt	Kozeta Sevrani, *kozeta.sevrani@unitir.edu.al*
Repr. EuroSim	Kozeta Sevrani
Edit. Board SNE	Albana Gorishti, *albana.gorishti@unitir.edu.al* Majlinda Godolja, *majlinda.godolja@feut.edu.al*

Last data update July 2019

Societies in Re-organisation / Former Societies

The following societies are at present inactive or under re-organisation:

- CROSSIM – *Croatian Society for Simulation Modelling*
  Contact: Tarzan Legović, *Tarzan.Legovic@irb.hr*
- FrancoSim – Société Francophone de Simulation
- HSS – Hungarian Simulation Society
- ISCS – Italian Society for Computer Simulation

The following societies have been formally terminated:

- MIMOS –Italian Modeling & Simulation Association; terminated end of 2020.

HSS – Hungarian Simulation Society

There are plans to reactivate Hungarian Simulation Society. M. Mujica Mota EUROSIM President, is in contact with Andrási Gábor, Head of the Dean's office at the Faculty of International Management and Business of Budapest Business School University of Applied Sciences (BBS). We ask interested people to contact Mr. Gábor, *andrasi.gabor@uni-bge.hu*.

These are good news from HSS, but we must inform also about sad news. The simulation community has lost a prominent proponent: András Jávor, for many years President of the Hungarian Simulation Society HSS, passed away in spring 2021.

Obituary Andras Javor

At the age of 84, our dear friend, colleague and mentor, Professor András Jávor has died.

András Jávor graduated from the Technical University of Budapest and received his M.Sc.E.E. followed by his Ph.D. degree in computer science. He received his D.Sc. from the Hungarian Academy of Sciences. After his graduation, he worked at the KFKI Research Institute for Measurement and Computing Techniques leading the Department for Simulation. He was full professor at the Budapest University of Technology and Economics and the Szechenyi Istvan University, Győr. In the latter he was dean of the Faculty of Informatics and Electrical Engineering between 2000-2002. His major interests were discrete simulation methodologies; knowledge base and AI controlled dynamic simulation, simulation methodologies and their application in the simulation of highly complex systems in the fields of micro- and macroeconomy, development of regions, environmental problems, flexible manufacturing systems, traffic, logistics, etc. He led the development of several simulation systems of three3 EU projects.

The number of his publications – among them several books and papers published in international journals and proceedings of conferences – is above 179. Since 1995 until his death, he was the Director of the McLeod Institute of Simulation Sciences Hungarian Center..

Among his other responsibilities he was the chairman of IMACS/Hungary, the Hungarian Simulation Society and member of the Board of directors of EUROSIM. He was a member of the editorial boards of four international scientific journals.

Professor András Jávor remained active even after stopping teaching work and made numerous publications until his death. He was always an approachable, humble scientist, teaching younger colleagues not only professional knowledge but humanity as well. We will miss his specific, individual, friendly manners, cultural and unique professional knowledge, his humour and the extensive discussions with him.

The below picture shows András Jávor in discussion with Agostino Bruzzone, President of Italian Liophant Simulation Society, and Felix Breitenecker, President of the German simulation society ASIM and SNE editor-in-chief, on occasion of the I3M Simulation Conference 2018 in Budapest.

Obituary source: infota.org/en/javor-andras-1937-2021/

Association Simulation News

ARGESIM is a non-profit association generally aiming for dissemination of information on system simulation – from research via development to applications of system simulation. **ARGESIM** is closely co-operating with **EUROSIM**, the Federation of European Simulation Societies, and with **ASIM**, the German Simulation Society. **ARGESIM** is an 'outsourced' activity from the *Mathematical Modelling and Simulation Group* of TU Wien, there is also close co-operation with TU Wien (organisationally and personally).

→ *www.argesim.org*

→ *office@argesim.org*

✉→ ARGESIM/Math. Modelling & Simulation Group,
Inst. of Analysis and Scientific Computing, TU Wien
Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria
Attn. Prof. Dr. Felix Breitenecker

ARGESIM is following its aims and scope by the following activities and projects:

- Publication of the scientific journal **SNE** – *Simulation Notes Europe* (membership journal of **EUROSIM**, the *Federation of European Simulation Societies*) – *www.sne-journal.org*
- Organisation and Publication of the **ARGESIM Benchmarks** for *Modelling Approaches and Simulation Implementations*
- Publication of the series **ARGESIM Reports** for monographs in system simulation, and proceedings of simulation conferences and workshops
- Publication of the special series **FBS Simulation** – Advances in Simulation / Fortschrittsberichte Simulation - monographs in co-operation with ASIM, the German Simulation Society
- Support of the Conference Series **MATHMOD Vienna** (triennial, in co-operation with **EUROSIM**, **ASIM**, and **TU Wien**) – www.mathmod.at
- Administration of **ASIM** (German Simulation Society) and administrative support for **EUROSIM** *www.eurosim.info*
- Simulation activities for TU Wien

ARGESIM is a registered non-profit association and a registered publisher: **ARGESIM Publisher Vienna**, root ISBN 978-3-901608-xx-y, root DOI 10.11128/z…zz.zz. Publication is open for **ASIM** and for **EUROSIM Member Societies**.

Schedule for EUROSIM Conferences and Congress

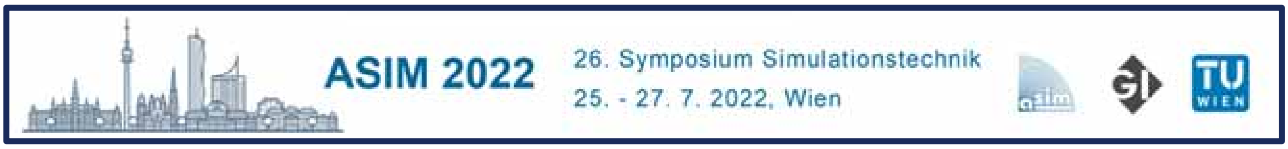
EUROSIM societies organise the following virtual and in-person events in 2022 and 2023:

DBSS
Dutch Benelux Simulation Society
www.eurosim2023.eu
VESS - VIRTUAL EUROSIM SIMULATION SEMINAR
free online seminar series

The **EUROSIM Board** and **DBSS** organise **VESS** – the **Virtual EUROSIM Seminar**, a series of online presentations discussing trends in modelling and simulation. These international online simulation seminars – monthly or bi-monthly – are open to everybody, via Zoom, lasting 60 minutes (45 minutes presentations, 15 minutes Q & A).
Information and informal registration via website **www.eurosim2023.eu**

MATHMOD 2022
Vienna
10th Vienna International Conference
on Mathematical Modelling
July 27 - 29, 2022, Vienna, Austria

MATHMOD organizers continue the conference series one year later, with **10th MATHMOD 2022**, July 27-29, 2022, as *in-person* event. **MATHMOD 2022**, one of **EUROSIM**'s main events, provides a forum for professionals, researchers, and experts in the field of theoretic and applied aspects of mathematical modelling for systems of dynamic nature.
The scope of the **MATHMOD 2022** conference covers theoretic and applied aspects of various types of mathematical modelling (equations of various types, automata, Petri nets, bond graphs, qualitative and fuzzy models) for systems of dynamic nature (deterministic, stochastic, continuous, discrete or hybrid) – info and details **www.mathmod.at**



ASIM - the German / Austrian / Swiss simulation society – is organising the **26th Symposium Simulation Technique** – **ASIM 2022** at TU Vienna, July 25-27, just before **MATHMOD 2022**. ASIM hopes for a German/English-based event as it used to be before – with personal contacts, and in synergy with **MATHMOD 2022**. – info **www.asim-gi.org/asim2022**

EUROSIM 2023, the **11th EUROSIM Congress**, will take place in Amsterdam, The Netherlands, Summer 2023. It will be organized by the Dutch Benelux Simulation Society (www.dutchbss.org) supported mainly by their corporate members like TU Delft, Amsterdam University of Applied Sciences, EUROCONTROL and IGAMT (www.igamt.eu).
Due to the growth of Simulation and its relationship with other analytical techniques like Big Data, AI, Machine Learning, Large Scale Simulation and others, the event will be structured, for the first time, in dedicated tracks focused on different areas and applications of Simulation ranging from aviation to health care and humanitarian activities.
Please follow the news and activities towards the **EUROSIM 2023** at **www.eurosim2023.eu**

New Schedule for Vienna **ASIM 2022** and **MATHMOD 2022**

Due to the critical pandemic situation in Europe it is not possible to hold these combined conferences as on-site events in February 2022. Since the simulation community strongly desires in-person events, it was decided to postpone **ASIM 2022** and **MATHMOD 2022** to **July 2022**.

ASIM 2022

26. Symposium Simulationstechnik
25. - 27. 7. 2022, Wien

The scope of the *ASIM Symposium Simulationstechnik* – also including the *workshop of the working groups GMMS and STS* – covers basics, methods, and tools of modeling and simulation as well as all areas of application (from engineering sciences to computer science, production and logistics, bio-, environmental and geosciences, climate and ecosystem, up to training and education in modeling and simulation.

Conference languages are German and English.

Submission of Full Contributions and Short Contributions is now possible until April 1, 2022.

Website: **www.asim-gi.org/asim2022**

Contact: *asim2022@asim-gi.org*

MATHMOD 2022
Vienna

10th Vienna International Conference on Mathematical Modelling

July 27 - 29, 2022, Vienna, Austria

The scope of *MATHMOD 2022* covers theoretic and applied aspects of various types of mathematical modelling (e.g., equations of various types, automata, Petri nets, bond graphs, qualitative and fuzzy models, machine learning)for systems of dynamic nature (deterministic, stochastic, continuous, discrete or hybrid with respect to time).

The reviewing of already submitted papers is already finished, and authors will be informed about acceptance until January 14, 2022. But it is possible to submit late papers: late full contributions until February 1, 2022, and late discussion contributions papers until March 15, 2022. See www.mathmod.at for details.

Website: **www.mathmod.at**

Contact: *mathmod@acin.tuwien.ac.at*

www.tuverlag.at
ISBN 978-3-903311-18-3