SNE SIMULATION NOTES EUROPE

Journal on Developments and Trends in Modelling and Simulation

EUROSIM Scientific Membership Journal

Vol. 33 No.3, September 2023 ISSN Online 2306-0271 DOI 10.11128/sne.33.3.1065
ISSN Print 2305-9974

ARGESIM

SIMULATION in Produktion und Logistik

13. – 15. SEPTEMBER 2023 | TECHNISCHE UNIVERSITÄT ILMENAU | GERMANY

Als größte europäische Simulationstagung für Produktion und Logistik präsentiert die ASIM Fachtagung alle zwei Jahre zukunftsweisende Trends und aktuelle Entwicklungen, wissenschaftliche Arbeiten sowie interessante Anwendungen in der Industrie. Der thematische Fokus der nächsten Fachtagung lautet Nachhaltigkeit in Produktion und Logistik. Damit greift die Tagung eine wichtige gesellschaftliche Thematik auf und fokussiert gleichzeitig auf aktuelle Forschungsthemen der Simulationswelt.

www.asim-fachtagung-spl.de/

I3M 2023
International Multidisciplinary
Modeling & Simulation Multiconference

18-20 September 2023
Athens, Greece

www.msc-les.org/i3m2023

Every year, the I3M Multiconference renovates the opportunity to bring together researchers, scientists and practitioners, from the Mediterranean Area, Latin & North Americas, Europe, Asia, Africa and Australia, who are concerned with Modeling and Simulation in Industry and Academia. As a result of the joint effort of many scientific excellences, I3M is an opportunity to meet and discuss and, as usually happens, constructive debate makes people open up, brings food for thoughts as well as new ways to explore encouraging multi-disciplinary cooperation and collaborations.

Winter Simulation Conference 2023
Simulation for Resilient Systems

December 10-13, 2023, San Antonio, TX
www.wintersim.org

The Winter Simulation Conference 2023 highlights the vital role that simulation plays in designing, planning, and operating resilient systems under uncertainty. In an increasingly inter-connected world, it is more critical than ever to ensure that systems quickly recover from and adapt to major disruptions. With its uncertainty modeling and explainable analytics capabilities, simulation is one of the key technologies that lie at the heart of building resilient systems. We invite papers that emphasize the latest advances in simulation theory and applications showcasing the integrated use of simulation with technologies ranging from the Internet of Things and statistics to AI/ML and optimization. We particularly encourage applications of simulation to improve resiliency in a wide range of domains, including but not limited to aviation, disaster response, education, energy, finance, healthcare, infrastructure, manufacturing, national security, space systems, and supply chains.

ASIM SST 2024 – Symposium Simulation Technique
Munich, September 2024

EUROSIM Congress 2026
July 2026, Italy www.eurosim.info

Editorial

Dear Readers, SNE 33(3), the third SNE issue in 2023, continues the SNE traditions: contributions from a very broad area of modelling and simulation, the contributions range from overview via applications to software, and the contributions come from submissions, from post-conference publication selections, and – new – are Proceedings Publications of conferences.
In July 2023, DBSS, the Dutch and Benelux simulation society, organized the 11th EUROSIM Congress in Amsterdam. Instead of classic proceedings, the contributions to this congress will be published in SPRINGER Communications in Computer and Information Science, and in special issues of simulation-oriented journals, among them also 'Simulation Notes Europe'. SNE is glad to publish the first EUROSIM 2023 Proceedings Contribution in this issue, 'Integration of Reinforcement Learning and Discrete Event Simulation using the Concept of Experimental Frame' by T. Pawletta and J. Bartelt. Further EUROSIM 2023 contributions will be published in the next SNE issues, partly in special issues. This issue continues with three post-conference publications from the ASIM Symposium Simulation Technique ASIM SST 2022 (July 2022, Vienna). There, Richter et al. present a model-based development of an automated demolition excavator', R. Büchi has elaborated parameter tables for PID controllers for time-delayed systems optimized with a learning method, and P. Junglas and L. Schmedes report on an application, the discrete event-based modeling of conveyors for dry bulk material. Furthermore this SNE continues also with the SNE Software Notes: Hofmeijer et al. introduce 'ERS - Enterprise Resource Simulator: a New Simulation Platform'.
And we are glad that we can welcome the new EUROSIM president, Agostino Bruzzone from Liophant Society, who will organize the 12th EUROSIM Congress in 2026 in Italy. In his inauguration speech he underlined SNE's benefits for EUROSIM and suggested widening and enhancement for SNE.
The cover of this issue presents the third digital marbling graphics by Graham Horton, type 'Bouquet'.
I would like to thank all authors for their contributions for this issue, also and many thanks to the SNE Editorial Office for layout, typesetting, preparations for printing, electronic publishing, and much more.

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

Contents SNE 33(3)

e-SNE 33(3), DOI 10.11128/sne.33.3.1065

www.sne-journal.org/sne-volumes/volume-33

SNE Basic e-Version with Open Access

SNE Full e-Version for Members of publication-active EUROSIM Societies: ASIM, CEA-SMSG, CSSS, DBSS, KA-SIM, LIOPHANT, NSSM, PTSK,SIMS, SLOSIM, UKSIM

print-SNE for members on demand (printer INTU TU Wien)

Content:

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: INTU (TU Wien), Wiedner Hauptstrasse 8-10, 1040, Vienna, Austria – *www.intu.at*

ARGESIM Publisher: 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*

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 (www.sne-journal.org).

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. **SNE** follows the recent developments and trends of modelling and simulation in new and/or joining 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.

SNE is the scientific membership journal of **EUROSIM**, the *Federation of European Simulation Societies and Simulation Groups* (www.eurosim.info), also providing Postconference publication for events of the member societies. **SNE**, primarily an electronic journal **e-SNE** (ISSN 2306-0271), follows an open access strategy, with free download in basic version (B/W, low resolution graphics). Members of most **EUROSIM** societies are entitled to download **e-SNE** in an elaborate full version (colour, high resolution graphics), and to access additional sources of benchmark publications, model sources, etc. (via group login of the society), **print-SNE** (ISSN 2305-9974) is available for specific groups of **EUROSIM** societies.

SNE is published by **ARGESIM** (www.argesim.org) on mandate of **EUROSIM** and **ASIM** (www.asim-gi.org), the German simulation society. **SNE** is DOI indexed with prefix 10.11128.

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, or from the Editor-in-Chief

www.sne-journal.org

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

SNE Editorial Board

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

Integration of Reinforcement Learning and Discrete Event Simulation Using the Concept of Experimental Frame

Thorsten Pawletta, Jan Bartelt

Research Group Computational Engineering and Automation (CEA), Wismar University of Applied Sciences, Philipp-Müller-Str. 14, 23966 Wismar, Germany; {thorsten.pawletta, jan.bartelt}@hs-wismar.de

SNE 33(3), 2023, 101-109, DOI: 10.11128/sne.33.tn.10651
Received (EUROSIM 2023): 2023-02-01; Acc.Conf.: 2023-03-30
Received SNE: 2023-08-15; Accepted: 2023-08-31
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Reinforcement Learning (RL) is an optimization method from the field of Machine Learning. It is characterized by two interacting entities referred to as the agent and the environment. The goal of RL is to learn how an agent should act to achieve a maximum cumulative reward in the long-term. A Discrete Event Simulation Model (DESM) maps the temporal behavior of a dynamic system. The execution of a DESM is done via a simulator.
The concept of an Experimental Frame (EF) defines the general structure used to separate the DESM into the dynamic system, called the Model Under Study (MUS), and its application context. This supports the diverse use of a MUS in different experimental contexts. This paper explores the generalized integration of discrete event simulation and RL using the concept of EF. The introduced approach is illustrated by a case study that has been implemented using MATLAB/Simulink and the SimEvents blockset.

Introduction

In modeling and simulation (M&S) theory [19], a model describes the dynamic behavior of a real or virtual system. A discrete event model is characterized by a finite number of states over a continuous time base. The execution of the model, i.e. the calculation of trajectories, is performed using a simulator. In the versatile use of a model, it should be developed independently from the context of use. The reference to a concrete experiment can be mapped by way of an Experimental Frame (EF).

An EF specifies the conditions under which a system is observed or a model experimented with [19, 17]. The model used is called the Model under Study (MUS). Depending on the experiments to be performed using a MUS, the corresponding EFs must be specified. Depending on the EF, the same model can be used in a parameter study, sensitivity analysis, optimization, etc. A context-specific EF and the MUS form the simulation model (SM) to be executed by the simulator. The concept of EF can be applied to all dynamic system models and their simulators but this paper focuses on discrete event simulation models (DESM).

The execution of a goal-directed experiment using a DESM and a simulator requires an *Experiment Control* (EC) [20]. The EC defines the goals and constraints of an experiment and structures the experiment process. Inspired by Breitenecker's [1] approach to structuring *simulation-based experiments* (SBE), Pawletta et al. [12] and Schmidt [13] concretized the concept of EC by introducing a *Simulation Method* (SimMeth) and *Experiment Method* (ExpMeth). The SimMeth controls the execution of the simulation runs and ExpMeth consists of arbitrary numerical methods. ExpMeth are used for the pre- and post-processing or to control the SimMeth, such as in simulation-based optimization experiments [3, 13].

Reinforcement Learning (RL) [15] in combination with a dynamic system simulation can be considered a specific SBE. According to Gosavi [8], RL is a simulation-based optimization of Markov Decision Processes (MDPs). In terms of RL, the MDP is modeled as an environment and the agent acts as a controller for the MDP. The agent influences the environment by actions, while the environment performs state transitions and responds with the new states and reward values for each transition. The optimization goal is to learn how the agent should act to achieve a maximum cumulative reward in the long-term.

In contrast to a discrete event dynamic system, an MDP is a discrete time process where the time base is only used for the sequential ordering of states. In addition, not all states of the MUS are usually of interest to the RL. Accordingly, the states of the MUS must be converted into MDP-compliant states. Due to the methodological differences, the combination of the two methods, RL and discrete event simulation, often lead in practice to implementations that are difficult to maintain and MUS that are not generally usable.

After discussing related work in section one, section two presents a general concept of SBE using the EF, and the basics of RL. In section three, an EF-based approach to integrate the RL method and DESM for performing SBE is presented. In addition, the approach is demonstrated using a case study.

1 Related Work

The combination of discrete event simulation and RL is part of numerous works in the field of M&S. The works can be roughly divided into four categories: (i) mathematically-oriented basics, (ii) application-specific solutions, (iii) extensions of the simulation environments, and (iv) generic M&S oriented approaches. The first category focuses on the mathematical principles for combining the two methods without addressing software implementations such as in the work of Gosavi [8].

Work of the second category often uses simplified models that are mapped as RL-compliant MDP environments [14, 10]. The models are not a general-purpose MUS. Other works use their own or proprietary simulation environments to represent the MUS and implement the RL-specific part in Python, often using AI libraries such as TensorFlow [6, 5]. In the case of Feldkamp et al. [6], the coupling is done using a client-server approach with the RL part acting as the server and the simulation environment as the client. The necessary state transformations for the coupling of the two methods are implemented on the RL side in Python.

The coupling of different software systems requires advanced programming skills. To simplify the application of RL techniques, the manufacturers of simulation environments have started to integrate RL-specific elements into their software systems such as described in the work Mahdavi and Tyler [11], Greasley [9], and The MathWorks [16] (third category).

The basic principle is to provide configurable RL agent objects that have a typical RL input/output interface (action as output, observation and reward as inputs). They can be used as parts to build a simulation model. In addition, RL-specific methods are provided that support, for example, the training of an agent. These supplements facilitate the integration of the two methods for users. Nevertheless, the method integration remains a challenge. Aspects such as a clear and maintainable structuring of the simulation model in a multi-purpose MUS, the reward calculation, the conversion of MUS system states or outputs into RL observations and vice versa RL actions into MUS inputs etc. have to be solved by the user.

Work of the fourth category deal with approaches to solving the challenges outlined above. Capocchi and Santucci [2] describe a structuring approach for integrating RL and DESM based on the Discrete Event System Specification (DEVS) [19]. They show the specification of agents with DEVS and how DEVS-based agents can communicate with an environment specified in DEVS. The focus is on the specification of agents. Choo et al. [4] analyzed the necessary transformations in the communication between an RL agent and an environment implemented as a DESM. Both agent and environment form the DESM. To structure the communication between both parts of the DESM, they introduce specific components called a decoder and encoder.

On the basis of Choo et al. [4], the concept of simulation-based experiments in the work of Pawletta et al. [12] and Schmidt [13], and the concept of EF, in the following, refers to a more advanced approach for the integration of RL and discrete event simulation is developed.

2 Basics of SBE, RL and EF

2.1 SBE and Concept of Experimental Frame

Schmidt [13] divides simulation-based experiments (SBEs) into three classes. In the following, only the first two classes are considered. The execution of one or more simulation runs by a SimMeth constitutes a *simple SBE* if the SimMeth is invoked directly by the user or an EC. The SimMeth sets the input values for the DESM, the execution parameters for the simulator, and controls the simulation runs. As previously mentioned, the EC specifies the experiment goals and experiment process. Besides the SimMeth, the EC can invoke further methods for pre- and post-processing.

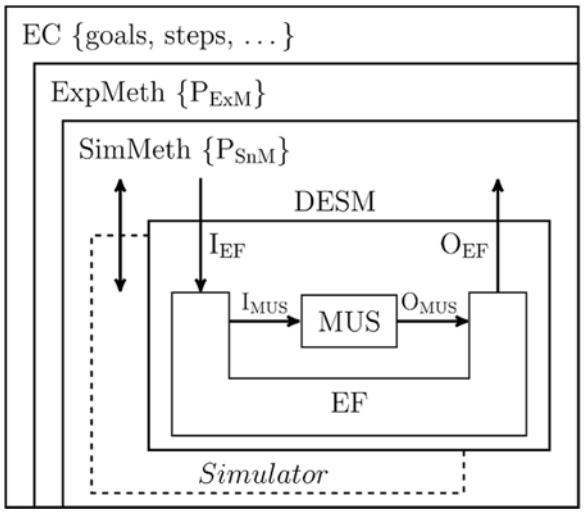

Figure 1: Basic structure of a complex SBE.

In a *complex SBE*, the SimMeth is controlled by an ExpMeth, for example, by a numerical optimization method. Figure 1 shows the basic structure of a complex SBE. Both the SimMeth and ExpMeth define the process parameters (P_{ExM}, P_{SnM}).

An SBE uses a DESM in a specific context. It defines certain experimental goals, conditions, and parameters. The concept of the EF separates the MUS from a specific context of use to improve the reusability of the MUS. Formally, Zeigler [18] defines the function of an EF using a 7-tuple.

$$EF = < T, I, C, O, \Omega_I, \Omega_C, SU >$$

Here T represents the time base, I and O the set of input and output variables of the MUS (equivalent to I_{MUS} and O_{MUS} in Figure 1), C the set of run control variables, Ω_I the set of admissible input segments, Ω_C the set of admissible control segments, and SU the set of summary mappings.

Set Ω_I refers to the input variables of the MUS and to the input/output relationships in the EF. Set Ω_C defines the experimental constraints which is a subset of C×T.

The experiment objectives are mapped to the variables, which are called *interest variables*. The set SU defines the determination of the interest variables based on the MUS output segments. The interest variables are the typical output variables of the EF (O_{EF}).

The implementation of an EF is done using three types of component, called *generator (Gen)*, *acceptor (Acc)* and *transducer (Trans)* [18, 19], as illustrated in Figure 2.

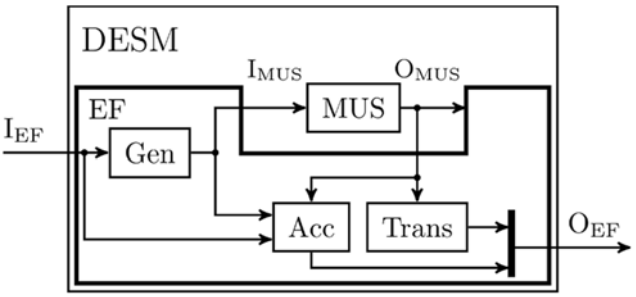

Figure 2: Basic structure of a DESM with MUS and EF. An EF does not necessarily have to contain all three components and the coupling relationships are not fixed.

Gen initializes the configurable parameters of the MUS and calculates the input segments for the MUS which can also be inputs of the Trans or Acc.

The Acc defines the admissible control segments and monitors their compliance. The output of the Acc is run control information. The Trans calculates the SU.

2.2 Reinforcement Learning

According to Sutton and Barto [15], RL focuses on the sequential decision-making by an agent that interacts with a real or virtual environment. The agent is trained by its interactions with the environment. The goal of RL is to learn a behavioral strategy π: $S \rightarrow A$ for the agent that assigns an action $a \in A$ to each state $s \in S$ of the environment. Thus, the agent can act as a controller for the environment. Using RL, a distinction is made between the training and deployment of an agent, although the agent can continue learning during deployment. The basic RL framework is shown in Figure 3.

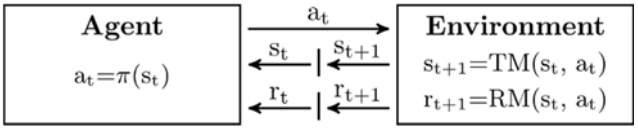

Figure 3: Basic RL framework.

In model-free RL, the agent only knows the allowed action set A at the start of training. The states $s \in S$ of the environment are unknown to the agent.

When an action $a_t \in A$ takes effect, the environment determines its next state s_{t+1} as well as a reward value r_{t+1} using a state transition model $TM: S \times A \rightarrow S$ and reward model $RM: S \times A \rightarrow R$.

The next state and the reward value are sent back to the agent. The index t marks a sequence of states in the sense of a MDP.

Through iterative interactions with the environment, the agent obtains information about possible states of the environment and the benefits of actions, gradually improving its behavioral strategy π. The goal of learning is to maximize the sum of the rewards until a goal is reached.

A variety of different learning strategies have been developed for RL agents such as Q-learning (QL), Deep Q Networks (DQN) etc.

We briefly consider Q-learning that uses formula (1) to learn a strategy π using a table function called the Q-matrix. A matrix element $Q(s,a)$ represents the estimated benefit of an action a_t when it is performed in the state s_t of the environment. The updated $Q(s_t,a_t)$ value of the current state/action tuple (s_t,a_t) is calculated from the previous $Q(s_t,a_t)$ value, the currently received reward r_{t+1}, and the maximum Q-value $(\max_a Q(s_{t+1},a))$ of all possible actions in the currently received next state s_{t+1}. The variables α and γ are process parameters, called hyperparameters.

$$Q(s_t a_t) \leftarrow Q(s_t a_t) + \alpha\,[r_{t+1} + \gamma \cdot \max_a Q(s_{t+1},a) - Q(s_t,a_t)] \qquad (1)$$

The training takes place in episodes. Episodes are independent of each other. Each episode starts in an initial state s_0 of the environment and ends when a target state s_{target} or abort state s_{abort} is reached. At the beginning of the training, the agent selects an action $a \in A$ randomly. This is called *exploration*. As the learning process progresses, the agent increasingly uses the knowledge it has acquired to select an action which is called *exploitation*. The ratio ε of exploration to exploitation is adjusted over the course of the training. After the completion of a defined number of training episodes, the behavioral strategy $a = \pi(s)$ is derived from the training data.

3 Integration of Reinforcement Learning and Discrete Event Simulations

3.1 Experimental Frame for Reinforcement Learning in the Training Phase

The basic structure of a DESM with an EF for RL in the training phase is shown in Figure 4. Although the approach is not limited to DESM, we will only focus on it.

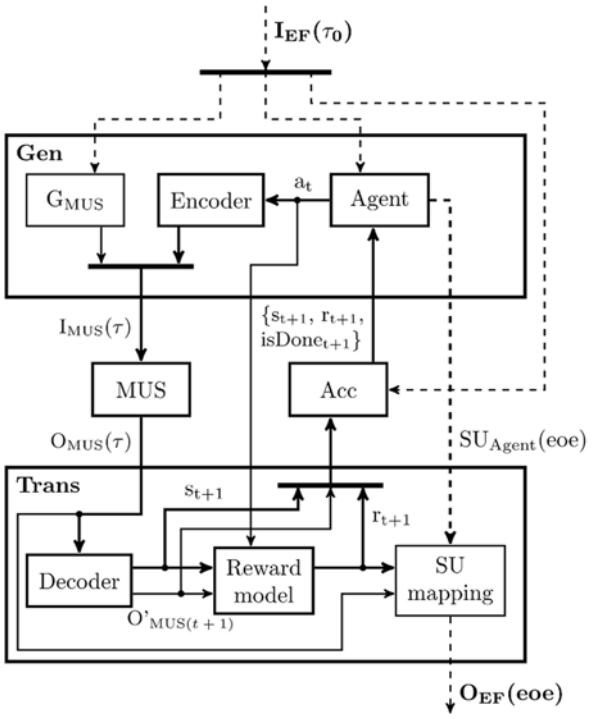

Figure 4: Basic structure of a DESM with MUS and EF for the training phase of a RL experiment.

The DESM consists of the three EF-components Gen, Trans, and Acc as well as the MUS. I_{EF} and O_{EF} represent the input/output interface of the DESM. The variables τ and t represent different time bases where τ is the continuous time base of the dynamic MUS and t the discrete time base for ordering the sequential states of the RL method.

The input variables I_{EF} are initialized by the EC or SimMeth (see Fig 1) at the simulation start time τ_0. The SimMeth executes single or multiple simulation runs.

Each simulation run corresponds to one episode of the RL method. The results of an episode get back via O_{EF} at the *end of an episode (eoe)*.

The Gen component is composed of three subsystems. *Gen.*G_{MUS} is a classical generator that initializes the parameters of the MUS at the beginning of an episode (τ_0) and calculates input segments Ω_I for the MUS inputs $I_{MUS}(\tau)$ over the course of an episode.

The RL agent is also mapped as a subsystem of the Gen (*Gen.Agent*) because the generated actions $a_t = \pi(s_t)$ are inputs of the MUS. The agent's hyperparameters are initialized at the beginning of an episode using the input interface $I_{EF}(\tau_0)$. The typical initialization parameters of component *Gen.Agent* are action set *A*, exploration rate ε and, in Q-learning, the current configuration of the *Q-matrix*.

Action a_t must often be transformed into MUS compatible inputs $i(\tau) \in I_{MUS}(\tau)$. For this data transformation, the encoder approach introduced by Choo et al. [4] is used. The subcomponent *Gen.Encoder* defines an application-specific transformation $i(\tau)=h(a_t)$. A typical example of such a data transformation is the generation of multiple input segments for the MUS based on a single action value. In addition to the inputs' new state s_{t+1} and the reward value r_{t+1} of the agent, a third input $isDone_{t+1}$ is defined.

The *isDone* information is a Boolean value that signals the end or cancellation of an episode to the agent. At the end of an episode*(eoe)*, the agent creates summary mapping $SU_{Agent}(eoe)$ that contains values such as the *number of steps in the episode*, the *total reward of an episode*, or the strategy learned so far (e.g. the *Q-matrix*). The $SU_{Agent}(eoe)$ is passed to the *Trans* component to create an overall summary mapping of the episode.

The *Trans* component is also composed of three subsystems. *Trans.Decoder* is a data transformation component. First, it defines the calculation of the *interest values* ($O'_{MUS(t+1)}$) from the current outputs of the MUS ($O_{MUS}(\tau)$) related to the time base of the RL, i.e.

$$O'_{MUS(t+1)}=f(O_{MUS}(\tau))$$

An example of such a transformation would be the calculation of the maximum queue length based on the previous queue occupancy. Second, it defines the transformation of the interest values $O'_{MUS(t+1)}$ to a state s_{t+1} in the RL space, that is

$$s_{t+1}=g(O'_{MUS(t+1)}).$$

Choo et al. [4] characterized this transformation form as

(i) *State Exhaustiveness* and
(ii) *State Mutual Exclusiveness.*

Here, (i) means that all interest values of the MUS are mapped into one state for the RL and (ii) that for each particular interest value of the MUS, there is only one corresponding state in the RL space.

The RM according to Section 2.2 is mapped in the component *Trans.Rewardmodel* because the reward value is an interest value based on the output variables of the MUS. The reward value characterizes a state tran-sition $s_t \rightarrow s_{t+1}$ in the RL space and is calculated by

$r_{t+1}=RM(s_t, a_t)$ or $r_{t+1}=RM(s_t, s_{t+1})$.

Defining the RM is sometimes a difficult problem. Our own experiments showed that the reward calculation can often be defined more efficiently based on the $O'_{MUS(t+1)}$ values, i.e.

$r_{t+1}=RM(O'_{MUS(t+1)}, a_t)$, or only $r_{t+1}=RM(O'_{MUS(t+1)})$.

The third subcomponent *Trans.SUmapping* implements the overall SU of an episode and passes it at the end of an episode to the EF output O_{EF}. In addition to the summary mapping of the agent (SU_{Agent}), the overall SU may also include the trajectories of the MUS outputs, a cumulative reward record etc.

In accordance with the concept of the EF in Section 2.1, the *Acc* component checks the compliance with the restrictions and termination conditions for the episode based on defined run control information.

The run control variables can be initialized at the start of an episode via the EF input $I_{EF}(\tau_0)$. Typical run conditions to be monitored include (i) the simulation interval $[\tau_0, \tau_{final}]$ of the MUS and thus the maximum duration of an episode and (ii) the detection of illegal system states or the reaching of a target state based on the $O'_{MUS(t+1)}$ or RL-related s_{t+1} values.

Accordingly, the *Acc* checks the newly calculated states s_{t+1} of the RL space as well as the reward values r_{t+1} before sending them to the *Gen.Agent* component. Furthermore, the *Acc* sets the Boolean *isDone* value which signals the continuation or the end of an episode according to the *Gen.Agent*.

3.2 Reinforcement Learning as a Simulation-based Experiment

According to the classification of SBE in Section 2.1, the RL is a complex SBE and has the general structure shown in Figure 1.

The goal of the experiment is

(i) to learn the best possible behavioral strategy π of an agent,
(ii) to extract the best strategy from the training data, and
(iii) to deploy the strategy.

When deploying, we have to distinguish whether a strategy is used with or without the further learning of the agent. The EC has to define these experimental steps. The steps involved in training and deploying the strategy require an ExpMeth that controls a SimMeth.

The ExpMeth *training* contains the following basic steps:

(1) Set the RL process parameters P_{ExM}, such as the learning rate, exploration rate, maximum number of episodes, Q-matrix etc.
(2) Set the simulation execution parameters P_{SnM} for the SimMeth, such as the simulator to be used, the simulation time interval etc.

(3) Set the DESM parameters for the EF components and the MUS and prepare the DESM for executing using a SimMeth.
(4) Initialize the statistical variables, such as those used to record the total reward per episode etc,
(5) Compute the defined number of episodes, i.e. call the SimMeth into a loop to execute the DESM, update the statistical variables after each episode, and check whether to abort the training or continue with another episode.
(6) Determine and save the best policy π, and plot essential learning results.

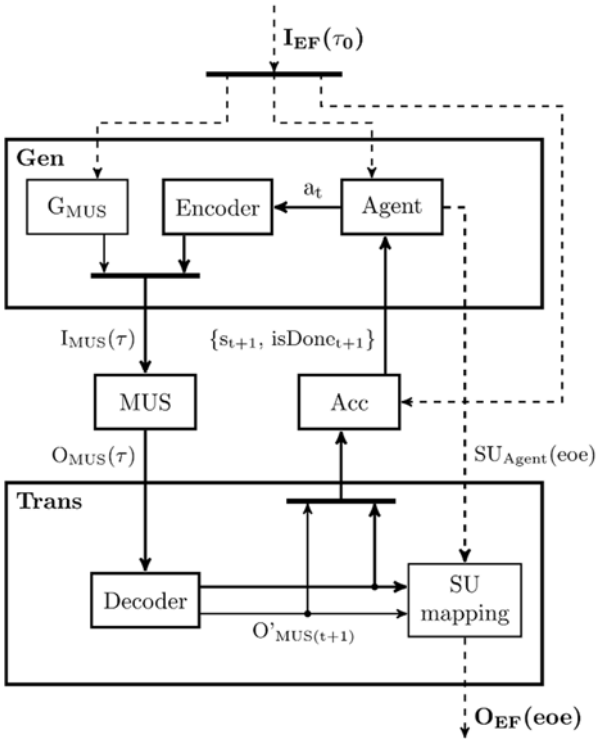

Figure 5: Basic structure of a DESM with MUS and EF for an experiment deployment without training.

An ExpMeth *deployment_with_training* has to implement nearly the same steps as the method *training* described before. The DESM with MUS and EF corresponds to the structure in Figure 4.
In contrast, the procedure and EF for an experimental step *deployment_without_training* is simplified. No explicit ExpMeth is required.

In the EC, the experiment parameters are defined and SimMeth is called on directly according to the number of simulation runs to be executed. Figure 5 shows the reduced structure of the DESM.

3.3 Case Study

The presented approach for integrating dynamic system simulations and RL was investigated on the basis of various case studies using MATLAB/Simulink and SimEvents.
Since the MathWorks' RL toolbox (release R2022a) supports integration with DESM – implemented using the SimEvents blockset – only via workarounds, we used a self-programmed Q-learning agent. In the following, we discuss some basic aspects of a case study without going into implementation details. The full implement-tation is available on Github [7].

The MUS is a simple server line consisting of an entity generator, a convertible operating unit, and two downstream servers connected in parallel with separate input queues as shown in Figure 6.

The operating unit can process two types of entity (*jobType=1|2*). A separate processing time can be defined for each entity type (*procT1, procT2*). A retooling time (*retoolingT*) is necessary when the entity type is changed in the operation unit. The calculation of the entity type and retooling time dependent processing time is done in the simulation runtime using two Simulink functions (not shown in Figure 6).

After processing, branching into one of the two FiFo queues of the downstream servers takes place depending on the entity type. The downstream servers have different processing times (*saleT1, saleT2*).

The definition of the different time values is determined by a value vector *param=[procT1, procT2, retoolingT, saleT1, saleT2]* at input port3 at the simulation start time τ_0. Entities are generated via input events (*msgGenJob*) at input port1. The entity type (*jobType*) to be generated follows on from the value at input port2.

After an entity has been processed in the operating unit, the MUS generates an output event (*y_msgFinish*) at output port1. Furthermore, the current tool setting (*sSetting*) of the operating unit, the current queue lengths (*y_#jobsQ1*, *y_#jobsQ2*), and the number of completed entities on the downstream servers (*y_#jobs1sold*, *y_#jobs2sold*) are output as data from port2 to port6.

Hence, input set I_{MUS} and output set O_{MUS} are defined by:

I_{MUS} = {msgGenJob(τ), type(τ), param(τ_0)}
O_{MUS} = {y_msgFinish(τ), y_sSetting(τ), y_#jobsQ1(τ), y_#jobsQ2(τ), y_#jobs1sold (τ), y_#jobs2sold(τ)}

Obviously, the MUS models the dynamic system behavior independent of a concrete experiment. The goal of the RL experiment is to learn the best possible injection strategy of the two entity types into the MUS so then the queues have the most balanced stock of both types available for the downstream servers.

After training is completed, the best strategy $\pi: S \rightarrow A$ should be extracted from the training data so then, subsequently, the agent can act as a controller of the MUS.

The EC is implemented as a MATLAB script. It defines the parameter sets P_{ExM} and P_{SnM}, such as:

(1) *action set={1, 2}*, coding the two entity types
(2) learning rate $\alpha=0.8$
(3) *sim. time interval=[*$\tau_0=0$, $\tau_{final}=480$*]* per episode
(4) The *number of episodes=20000*, etc.

and calls the ExpMeth *training*. This calls the SimMeth into a loop to execute the DESM for one episode.

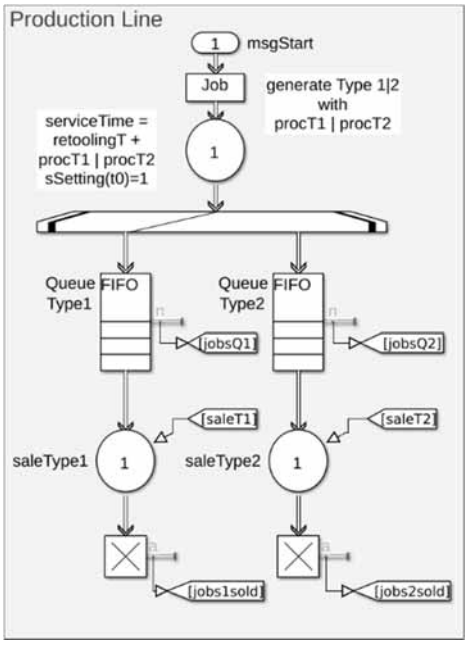

Figure 6: Structure of the MUS in SimEvents.

The MATLAB built-in function *sim* is used as SimMeth. The data transfer between the different methods and the DESM is mainly done via data workspaces.

Figure 7 shows the top-level structure of the DESM for the training phase. It contains all components according to the general approach shown in Figure 4. The MUS named *Prodline* provides the input-output interface described above with $I_{MUS}(\tau)$ and $O_{MUS}(\tau)$. The I_{EF} and O_{EF} of the EF are not visible on the top-level structure of the DESM.

This interface is realized via workspace variables. The encapsulation of the Gen and Trans subcomponents has been omitted. *Parameters* is a G_{MUS}. It generates the constant input segments for the MUS input vector *param* for initializing the MUS parameters. The initialization of parameters for the *Agent*, such as the *Q-matrix*, and for the *Acc* at the beginning of each episode is encoded directly in these components. Analogously, the components *Agent* and *SU.Mapping* output the O_{EF} at the end of an episode.

At simulation start time τ_0, an episode is started by the Agent sending an event *msgGenJob* and setting an action $a_t=\{1 \mid 2\}$ at the *action* port. In this case, the outputs of the Agent are compatible with the inputs of the MUS in value and timestamp with respect to the global simulation clock. Hence, the Agent's outputs are only forwarded by the *Encoder* to the MUS *ProdLine* that generates a new entity with *jobType=action value*.

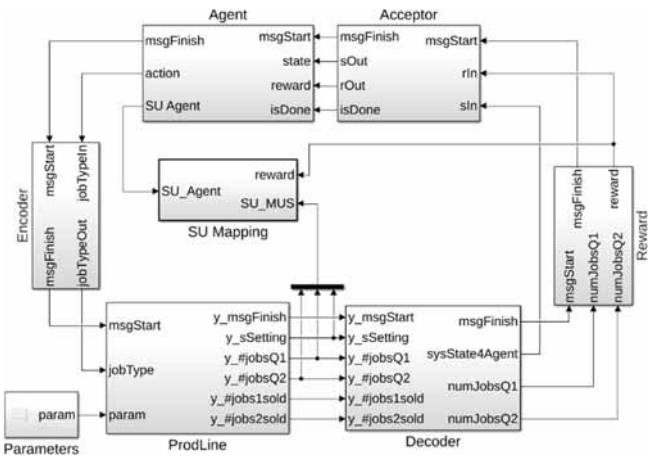

Figure 7: Top-level structure of the DESM with MUS (named *ProdLine*) and EF for the training phase of an RL experiment.

When an entity has completed on the operation unit and been forwarded to one of the two queues, an output event *y_msgFinish(τ)* is sent to activate the *Decoder* and the output data of the MUS is updated. Study-specific *output data(τ)* is passed to the *SU_Mapping* for trajectory recording.

The *Decoder* selects the information relevant to the RL from the MUS *output data(τ)* and calculates the new state s_{t+1} of the RL space (output port *sysState4Agent*). To limit the RL space, the state set of the two queues is limited to a maximum length ($qlength_{max}$) to be considered (2), (3).

The new state s_{t+1} is calculated based on the current tool setting (*sSetting*) of the operating unit and the two limited queue lengths (4).

$$qlQ1 = \max(y_{\#jobsQ1}, qlength_{max}) \quad (2)$$

$$qlQ2 = \max(y_{\#jobsQ2}, qlength_{max}) \quad (3)$$

$$s_{t+1} = (sSetting - 1) \cdot (qlength_{max} + 1)^2 + \\ + qlQ1 \cdot (qlength_{max} + 1) + qlQ2 + 1 \quad (4)$$

An experiment, with *sSetting={1,2}* and $qlength_{max}=30$, results in set of RL states *S={1,2,3,…,1921,1922}*.

After decoding, the reward calculation is activated by an event (*msgFinish*). Contrary to the general approach, the reward is not calculated using the RL-related state $s \in S$ but on the basis of MUS-related variables $O'_{MUS}(t+1)$. In terms of content, both approaches are identical but the second one resulted in a much better structured reward computation (5).

$$r_{t+1} = \begin{Bmatrix} 100 \mid qlQ1 \geq 10 \; and \; qlQ2 \geq 10 \\ qlQ2^2 \mid qlQ1 \geq 10 \; and \; qlQ2 < 10 \\ qlQ1^2 \mid qlQ1 < 10 \; and \; qlQ2 \geq 10 \\ \frac{qlQ2^2 \cdot qlQ1^2}{100} \mid else \end{Bmatrix} \quad (5)$$

After the reward calculation, the *Acceptor* is activated by an event (*msgFinish*). No constraints are defined for s_{t+1} and r_{t+1}, so they are only passed to the *Agent* (*rIn* to *rOut* and *sIn* to *sOut*).

The *Acceptor* defines only a control segment for the simulation time interval $[\tau_0, \tau_{final}]$, which defines the length of an episode. Moreover, restrictions can be defined depending on the queue lengths, for example, thus the premature termination of an episode. At the termination of an episode, the *Acceptor* schedules an internal event with an infinitesimal time advance. The time delay is necessary for data updates in the *Agent* and *SU_Mapping* at the end of an episode.

The *Acceptor* activates the *Agent* via an event (*msgFinish*) and signals using the Boolean variable *isDone* whether the end of an episode has been reached or not.

The *Agent* executes its learning rules and, depending on the *isDone* value, calculates a new *action* value or performs a final data update.

When training the Q-learning agent, the total reward, i.e. the sum of the rewards of an episode, converged on its final value after about 5000 episodes.

The time trajectories of the queue lengths computed using the learned policy π: $S \rightarrow A$ shown in Figure 8 prove that the agent can act as a controller of the MUS.

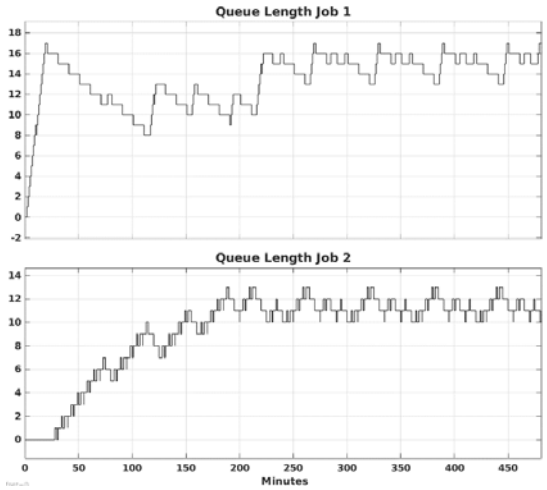

Figure 8: Time trajectories of the queue lengths computed using the learned policy.

4 Conclusions

The integration of dynamic system simulation and RL methods has a high application potential for both M&S and AI applications.

On the basis of the concept of EF and the general structure of complex SBE, it has been shown how a clear methodological separation can be made so then the dynamic system models, simulation methods, simulators and AI methods can be developed independently and reused in different contexts. The methodological considerations have been practically underpinned using a case study.

In the next steps, the methodological approach will be formalized using the Discrete Event System Specification (DEVS). Furthermore, the specification and automation of simulation-based RL experiments based on an extended System Entity Structure and Model Base (SES/MB) framework will be investigated.

References

[1] Breitenecker F. Models, methods and experiments - a new structure for simulation systems. *Mathematics and Computers in Simulation*; 1992. 34(3). 231–260.

[2] Capocchi L, Santucci JF Discrete event modeling and simulation for reinforcement learning system design. *Information*; 2022. 13(3). 1-13

[3] Carson Y, Maria A. Simulation optimization: methods and applications. In: *Proceedings of the 1997 Winter Simulation Conference*; 1997. 118-126.

[4] Choo B, Graham C, Stephen A, Dadgostari F, Beling PA. Reinforcement learning from simulated environments: an encoder decoder framework. In: *Proceedings of the SCS SpringSim'20 (Virtual) Conference;* 2020. 12 pages.

[5] Ehn G, Werner H. Reinforcement learning for planning of a simulated production line. *Master thesis*, Lund University; 2018.

[6] Feldkamp N, Bergmann S, Strassburger S. Simulation-based deep reinforcement learning for modular production systems. In: *Proceedings of the 2020 Winter Simulation Conference*; 2020. 1596-1607.

[7] FG CEA Integration of RL and Discrete Event Simulation: A Case study using MATLAB/ Simulink/ SimEvents. Wismar Univ. of Applied Sciences; Wismar. https://github.com/cea-wismar.

[8] Gosavi A. Solving markov decision processes via simulation. In Fu, M.C. (ed.), *Handbook of Simulation Optimization.* Springer; 2015. 341-374.

[9] Greasley A. Implementing Reinforcement Learning in Simio discrete-event simulation software. In *Proceedings of the SCS SummerSim'20 (Virtual) Conference*; 2020. 11 pages.

[10] Leng J, Jin C, Vogl A, Lui H. Deep reinforcement learning for color-batching resequencing problem. *Journal of Manufacturing Systems*, Elsevier, 56(2020); 2020. 175-187.

[11] Mahdavi A, Wolfe-Adam T. Artificial Intelligence and Simulation in Business [White Paper]; 2020. (reading date 10/20/2022).

[12] Pawletta T. Specification and execution of simulation models and experiments. *MS Workshop 'One simulation model is not enough'*, Univ. of Rostock, Dep. of Computer Science; 2019. https://www.cea-wismar.de/pawel/Forschung/Poster_Slides/2019-04-23-Presi_FG-CEA_UnivRo-WS_reducedSize.pdf.

[13] Schmidt A. Variant management in modeling and simulation using the SES/MB framework [Dissertation]. In: *Advances in Simulation.* Bd. 30. TU Publisher Vienna (in German); 2019.

[14] Shuhui Q, Wang J, Shivani G. Learning adaptive dispatching rules for a manufacturing process system by using reinforcement learning approach. In: *2016 IEEE - 21st International Conference on Emerging Technologies and Factory Automation (ETFA);* 2016. 1-8.

[15] Sutton RS, Barto AG. *Reinforcement learning: an introduction – 2nd edition*. MIT Press; 2018.

[16] The MathWorks. Reinforcement Learning Toolbox. *https://mathworks.com/products/reinforcement-learning.html*, ©1994-2022 The MathWorks, Inc.; 2022.

[17] Traore MK, Muzy A. Capturing the dual relationship between simulation models and their context. *Simulation Modeling Practice and Theory, Elsevier*, 14(2006); 2006. 126-142.

[18] Zeigler BP. *Multifacetted Modelling and Discrete Event Simulation.* Academic Press; 1984.

[19] Zeigler BP, et al. *Theory of modeling and simulation – 3rd edition.* Elsevier, Academic Press; 2018.

ASIM Books – ASIM Book Series – ASIM Buchreihen

Monograhps / Proceedings

Proceedings Langbeiträge ASIM Workshop 2023 STS/GMMS/EDU - ASIM Fachgruppenworkshop 2023
Univ. Magdeburg, März 2023; C. Krull; W. Commerell, U. Durak, A. Körner, T. Pawletta (Hrsg.)
ARGESIM Report 21; ASIM Mitteilung 185; ISBN ebook 978-3-903347-61-8, DOI 10.11128/arep.21, ARGESIM Verlag, Wien, 2023

Kurzbeiträge & Abstract-Beiträge ASIM Workshop 2023 STS/GMMS/EDU - ASIM Fachgruppenworkshop 2023
Univ. Magdeburg, März 2023; C. Krull; W. Commerell, U. Durak, A. Körner, T. Pawletta (Hrsg.)
ARGESIM Report 22; ASIM Mitteilung 186; ISBN ebook 978-3-903347-62-5, DOI 10.11128/arep.22, ARGESIM Verlag, Wien, 2023

Proceedings Langbeiträge ASIM SST 2022 -26. ASIM Symposium Simulationstechnik, TU Wien, Juli 2022
F. Breitenecker, C. Deatcu, U. Durak, A. Körner, T. Pawletta (Hrsg.), ARGESIM Report 20; ASIM Mitteilung AM 181
ISBN ebook 978-3-901608-97-1, DOI 10.11128/arep.20, ARGESIM Verlag Wien, 2022

Proceedings Kurzbeiträge ASIM SST 2022 -26. ASIM Symposium Simulationstechnik, TU Wien, Juli 2022
F. Breitenecker, C. Deatcu, U. Durak, A. Körner, T. Pawletta (Hrsg.), ARGESIM Report 19; ASIM Mitteilung AM 179
ISBN ebook 978-3-901608-96-4, DOI 10.11128/arep.19, ISBN print 978-3-901608-73-5, ARGESIM Verlag Wien, 2022

Simulation in Production and Logistics 2021 – 19. ASIM Fachtagung Simulation in Produktion und Logistik
Online Tagung, Sept. 2021, J. Franke, P. Schuderer (Hrsg.), Cuvillier Verlag, Göttingen, 2021,
ISBN print 978-3-73697-479-1; ISBN ebook 978-3-73696-479-2; ASIM Mitteilung AM177

Proceedings ASIM SST 2020 – 25. ASIM Symposium Simulationstechnik, Online-Tagung
14.-15.10.2020; C. Deatcu, D. Lückerath, O. Ullrich, U. Durak (Hrsg.), ARGESIM Verlag Wien, 2020;
ISBN ebook: 978-3-901608-93-3; DOI 10.11128/arep.59; ARGESIM Report 59; ASIM Mitteilung AM 174

Book Series Fortschrittsberichte Simulation – Advances in Simulation

Cooperative and Multirate Simulation: Analysis, Classification and New Hierarchical Approaches. I. Hafner, FBS 39
ISBN ebook978-3-903347-39-7, DOI 10.11128/fbs.39, ARGESIM Publ. Vienna,2022

Die Bedeutung der Risikoanalyse für den Rechtsschutz bei automatisierten Verwaltungsstrafverfahren. T. Preiß, FBS 38
ISBN ebook 978-3-903347-38-0, DOI 10.11128/fbs.38, ARGESIM Publ. Vienna,2020

Methods for Hybrid Modeling and Simulation-Based Optimization in Energy-Aware Production Planning. B. Heinzl, FBS 37
ISBN ebook 978-3-903347-37-3, DOI 10.11128/fbs.37, ARGESIM Publ. Vienna,2020;

Konforme Abbildungen zur Simulation von Modellen mit verteilten Parametern. Martin Holzinger, FBS 36
ISBN ebook 978-3-903347-36-6, DOI 10.11128/fbs.36, ARGESIM Publ. Vienna, 2020

Fractional Diffusion by Random Walks on Hierarchical and Fractal Topological Structures. G. Schneckenreither, FBS 35
ISBN ebook 978-3-903347-35-9, DOI 10.11128/fbs.35, ARGESIM Publ. Vienna, 2020

A Framework Including Artificial Neural Networks in Modelling Hybrid Dynamical Systems. Stefanie Winkler, FBS 34
ISBN ebook 978-3-903347-34-2, DOI 10.11128/fbs.34, ARGESIM Publ. Vienna, 2020

Modelling Synthesis of Lattice Gas Cellular Automata and Random Walk and Application to Gluing of Bulk Material. C. Rößler, FBS 33
ISBN ebook 978-3-903347-33-5, DOI 10.11128/fbs.33, ARGESIM Publ. Vienna, 2020

Combined Models of Pulse Wave and ECG Analysis for Risk Prediction in End-stage Renal Desease Patients. S. Hagmair, FBS 32
ISBN ebook 978-3-903347-32-8, DOI 10.11128/fbs.32, ARGESIM Publ. Vienna, 2020

Mathematical Models for Pulse Wave Analysis Considering Ventriculo-arterial Coupling in Systolic Heart Failure. S. Parragh, FBS 31
ISBN ebook 978-3-903347-31-1, DOI 10.11128/fbs.31, ARGESIM Publ. Vienna, 2020

Variantenmanagement in der Modellbildung und Simulation unter Verwendung des SES/MB Frameworks. A. Schmidt, FBS 30;
ISBN ebook 978-3-903347-30-4, DOI 10.11128/fbs.30, ARGESIM Verlag, Wien 2019

Classification of Microscopic Models with Respect to Aggregated System Behaviour. Martin Bicher, FBS 29;
ISBN ebook 978-3-903347-29-8, DOI 10.11128/fbs.29, ARGESIM Publ. Vienna, 2017

Model Based Methods for Early Diagnosis of Cardiovascular Diseases. Martin Bachler, FBS 28;
ISBN ebook 978-3-903347-28-1, DOI 10.11128/fbs.28, ARGESIM Publ. Vienna, 2017

A Mathematical Characterisation of State Events in Hybrid Modelling. Andreas Körner, FBS 27;
ISBN ebook 978-3-903347-27-4, DOI 10.11128/fbs.27, ARGESIM Publ. Vienna, 2016

Comparative Modelling and Simulation: A Concept for Modular Modelling and Hybrid Simulation of Complex Systems. FBS 26**,** N.Popper, FBS 26; ISBN ebook 978-3-903347-26-7, DOI 10.11128/fbs.26, ARGESIM Publ. Vienna, 2016

Rapid Control Prototyping komplexer und flexibler Robotersteuerungen auf Basis des SBE-Ansatzes. Gunnar Maletzki, FBS 25;
ISBN ebook 978-3-903347-25-0, DOI 10.11128/fbs.25, ARGESIM Publ. Vienna, 2019

A Comparative Analysis of System Dynamics and Agent-Based Modelling for Health Care Reimbursement Systems. P. Einzinger, FBS 24; ISBN ebook 978-3-903347-24-3, DOI 10.11128/fbs.24, ARGESIM Publ. Vienna, 2016

Agentenbasierte Simulation von Personenströmen mit unterschiedlichen Charakteristiken. Martin Bruckner, FBS 23;
ISBN ebook Online 978-3-903347-23-6, DOI 10.11128/fbs.23, ARGESIM Verlag Wien, 2016

Deployment of Mathematical Simulation Models for Space Management. Stefan Emrich, FBS 22;
ISBN ebook 978-3-903347-22-9, DOI 10.11128/fbs.22, ARGESIM Publisher Vienna, 2016

Lattice Boltzmann Modeling and Simulation of Incompressible Flows in Distensible Tubes for Applications in Hemodynamics. X. Descovich, FBS 21; ISBN ebook 978-3-903347-21-2, DOI 10.11128/fbs.21, ARGESIM, 2016

Mathematical Modeling for New Insights into Epidemics by Herd Immunity and Serotype Shift. Florian Miksch, FBS 20;
ISBN ebook 978-3-903347-20-5, DOI 10.11128/fbs.20, ARGESIM Publ. Vienna, 2016

Integration of Agent Based Modelling in DEVS for Utilisation Analysis: The MoreSpace Project at TU Vienna. S. Tauböck, FBS 19
ISBN ebook 978-3-903347-19-9, DOI 10.11128/fbs.19, ARGESIM Publ., 2016

Model-based Development of an Automated and Remotely-Controlled Demolition Excavator

Christian Richter, Masoud Shakoorianfard, Volker Waurich, Frank Will

Professur für Baumaschinen, TU Dresden, Münchner Platz 3, 01069 Dresden, Germany
{christian.richter1, masoud.shakoorianfard, volker.waurich, frank.will}@tu-dresden.de

SNE 33(3), 2023, 111-116, DOI: 10.11128/sne.33.tn.10652
Selected ASIM SST 2022 Postconf. Publication: 2023-08-15;
Received Revised Improved: 2023-10-18; Accepted: 2023-10-22
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. This paper describes the application of a 3D-simulation model to develop the control system of a demolition excavator. In order to simplify the system development and provide means for early prototype validation, a simulation-based development workflow is presented. A 3D model of the excavator, a demolishable wall, a virtual LiDAR sensor as well as CAN and Ethernet communication interfaces are used to support the development and testing of different software components as well as a test setup for automation strategies.

Introduction

In the case of nearby settlements and infrastructure, the demolition of natural draft cooling towers is restricted to non-explosive demolition techniques. One innovative approach is the usage of a modified excavator that sits on the wall structure and demolishes the wall piece by piece with a pulverizer.Figure 1 shows the excavator during operation. The excavator drives around the wall to demolish a 2 m high ring segment during each turn and reduce the height of the cooling tower up to a certain height. After this, the cooling tower can be collapsed in a controlled fashion by weakening the structure.

Figure 1: Demolition of the cooling tower in Mühlheim-Kährlich with the demolition excavator.

The excavator is currently operated from a platform on top of the cooling tower with a handheld remote control by sight, as depicted in Figure 2. Former attempts of controlling the excavator by video streams have been neglected because of image latency and poor depth perception. Due to these inefficient and dangerous operation conditions, an improved approach for a tele-remote workplace must be designed and implemented. In order to further facilitate the operation, suitable sub-tasks (e.g., the demolition of a wall segment) must be automated. The authors developed a simulation model of the machine and the process in order to support the product development even in the early stages.

This chapter gives an introduction to the basic machine setup and the corresponding simulation model. Chapter 1 describes the virtual LiDAR sensor and the point cloud processing.

Chapter 2 presents the abilities to use the simulation model to validate CAN-communication and to act as a Hardware-in-the-Loop (HiL)-simulation. Chapter 3 outlines the automation strategy and how this can be tested with the simulation model. Chapter 4 describes the developed approach for digital twin-based teleoperation. Chapter 5 closes the paper with a conclusion.

Figure 2: Conventional operation of the excavator with a handheld remote control from a platform on top of the cooling tower.

1 Machine and Simulation Setup

1.1 Equipment and Sensor Setup

The excavator consists of a frame structure including two driven wheel flanges that drive the whole machine on the edge of the cooling tower. On the frame, the working attachment of a conventional excavator is installed. During demolition work, the frame is clamped on the wall by two runners on both sides. The excavator has an adjustable boom and has four cylinders to adjust the height and range of the tool. The working attachment can be rotated by a slewing cylinder. The demolition tool is a pulverizer e.g., a massive gripper that cuts out a segment of the reinforced concrete. The tool itself can be rotated, opened, and closed.

In order to sense the working attachment, the main boom, the adjustable boom, the arm, and the tool coupler are equipped with 1D inclination sensors in order to compute the joint angles. The length of the slewing cylinder is sensed with a draw-wire sensor.

An encoder inside the rotary feedthrough of the pulverizer captures the rotation angle. Pressure sensors determine the opening and closing state of the tool. The pitch and roll angle of the machine is captured by a 2D-inclination sensor. With this sensor setup, a complete kinematic representation is possible by calculating the forward kinematics.

A 3D-LiDAR sensor is used to capture the working process, i.e. the wall during demolition. A LiDAR is a laser-based distance sensor that is able to record a 3D point cloud of the surrounding. The LiDAR is attached to the frame and faces the inner face of the nearby wall that is in reach of the excavator. Figure 3 illustrates the sensors to sense the working attachment and the wall surface.

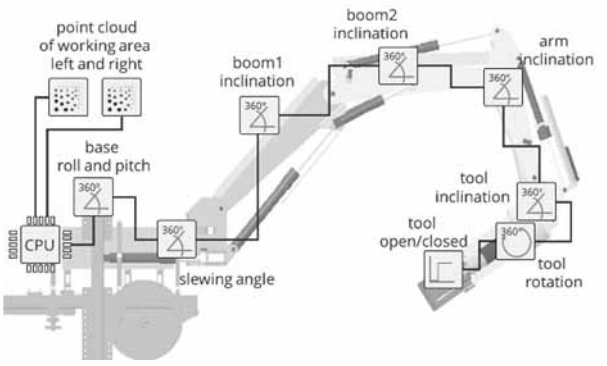

Figure 3: Sensor setup of the demolishion excavator.

1.2 Simulation Setup

At the beginning of the project, the excavator was modeled as a CAD-Model. The excavator working attachment itself had to be reverse-engineered whereas the manufacturer has provided a CAD-Model of the frame. Based on the geometric information, a simulation model has been developed that serves the following purposes:

- Kinematic simulation to test reachability
- 3D-collision detection
- Simulation of a destructible wall volume
- Emulation of LiDAR sensors to retrieve virtual point cloud data of the machine and the variable wall geometry
- Software and hardware interfaces to enable HiL, SiL, and Co-simulation
- An appealing 3D visualization to reuse the model for a tele-remote operation
- Realtime Feedback
- Scriptable simulations for automated testing

As a simulation framework, the Unity Engine [1] was chosen since it provides an efficient 3D environment with a built-in physics engine, which is optimized for real-time applications. Compared to domain-specific simulation environments, it is relatively easy to implement special-purpose features such as specific virtual sensors or variable 3D geometries.

The serial kinematic of the machine is modeled as a kinematic tree that is visualized with reduced CAD models, as can be seen in Figure 4. The inputs to drive the model are the angles of the revolute joints that connect the rigid bodies of the working attachment. The visualization of hydraulic cylinders and rods moves accordingly. A kinetic multi-body simulation or hydraulic system simulation is not implemented as this is unnecessary for the application. A Co-Simulation to e.g., a Modelica-based model via an FMU or interprocess communication would be possible as done by the authors in previous projects [2].

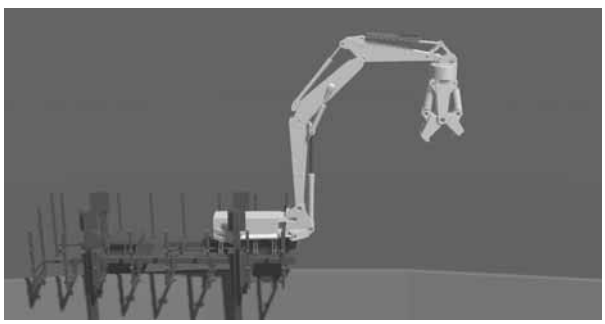

Figure 4: Visualization of the kinematic excavator model.

2 Virtual LiDAR and Wall Sensing

To detect the wall segment in front of the excavator, a Livox Mid-70 LiDAR is used [4]. In general, a LiDAR is an array of laser-based distance measurement sensors that either rotate around an axis (rotary LiDAR) or are arranged in a matrix (flash LiDAR).

The used LiDAR sensor is somehow unique in its scanning pattern. It features a non-repetitive scanning technology in the shape of a hypotrochoid (i.e., a small circle rolls inside a bigger one). Therefore, the coverage inside the field of view (FOV) increases over time.

In the simulated sensor model, the FOV, rotation speed, number of rays per shot, ratio of the circle radiuses are adjustable. In addition to the simulated LiDAR, a point cloud accumulator is implemented, that aggregates the measured point clouds over time.

Figure 5 depicts the scanning patterns of the rotating hypotrochoid in the simulation and the accumulated point clouds over time.

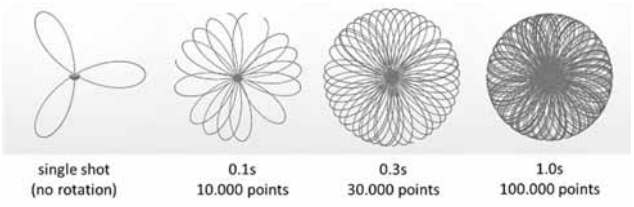

Figure 5: Scanning Pattern and accumulated point cloud over time.

The simulated sensor is able to provide the same point rate of 100.000 points/s as the real sensor on an i7-3930K @3.20GHz.

The simulation of the LiDAR utilizes the built-in physics engine in the Unity software. Each distance measurement is done by calling the built-in function:

```
bool UnityEngine.Physics.Raycast(
Vector3 origin,
Vector3 direction,
out RaycastHit hit,
float maxDistance);
```

A Raycast outputs a RaycastHit object if the ray hits a so-called GameObject that has an instance of a collider-class. Therefore, every object that must be detected by the LiDAR needs to provide this ability.

There are different collider classes available, both for geometric primitives (cubes, spheres, to name but two) and mesh-based colliders. For the detection of the deforming wall, an adaptive mesh-collider is needed.

For the detection of the working attachment of the excavator, geometric primitives as bounding boxes of the complex geometry are sufficient. The base machine and the cooling tower do not feature a Collider object because these areas will be cropped out of the point clouds.

Moreover, considering the interaction between the wall and the tool as the most important part of the simulation, it is required to differentiate the points that corresponded with the wall and tool from the other objects in the scene. This step, which is called Segmentation, is done by taking advantage of the kinematic information provided by sensors and accordingly moving colliders of the working attachment of the excavator, resulting in removing the points that corresponded with the machine objects except for the tool. Figure 6 shows the simulated point cloud on the wall segment and the tool.

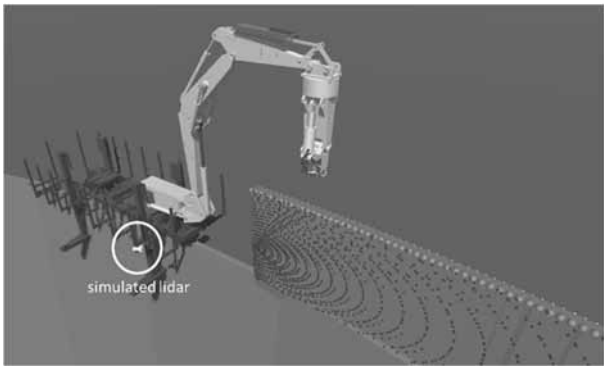

Figure 6: Simulated LiDAR with a wall segment that implements a mesh collider.

In order to simulate the demolition process of the wall, parts of the wall have to be removed. Therefore, a procedural and time-varying mesh of the wall segment is implemented. The same mesh is used, both for visualization and to serve as a mesh collider for the simulated LiDAR.

The procedural wall model is initialized with a fixed height and depth at a specified position. At the position of the excavator tool, a cubic collider is attached, that has the size of the pulverizer chamber. If a wall vertex is hit by the tool collider, the top vertices of the mesh are shifted vertically to the bounding box of the collider. Figure 7 illustrates the deformation of the wall mesh by shifting the top vertices.

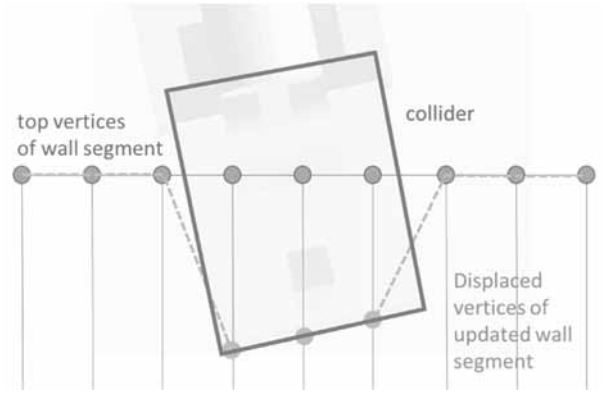

Figure 7: Wall deformation by the excavator tool.

The result of a deformed wall segment reflecting the raycasts by the emulated LiDAR can be seen in Figure 8.

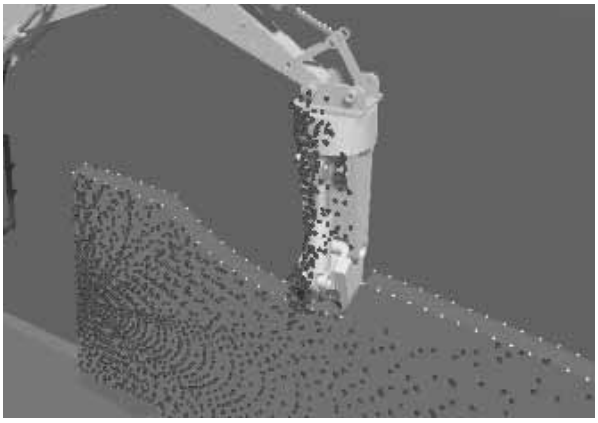

Figure 8: Deformed wall segment with updated mesh collider to reflect the raycasts of the LiDAR.

The resulting point cloud data of the emulated LiDAR sensor is provided as a simple array of Cartesian coordinates referenced in the sensor coordinate system to be used for interprocess communication via shared memory. Additionally, a network interface is provided as well, featuring the basic parts of the official Livox-communication protocol, resulting in the possibility to establish an emulated network stream to communicate with the commonly used tools officially provided by the manufacturer, including the official Livox-SDK, livox-ros-driver and livox-ros2-driver ROS packages, and the Livox Viewer software.

The point cloud data provided by the simulation must be processed as if it is provided by the real sensor installed on the machine. To do so, an industrial PC (IPC) and the ROS2 framework are utilized as the processing unit.

This unit provides the required data for the visualization unit in the tele-remote-operation mode and implements the overall automation strategy in automation mode, both with the help of the processed data.

In order to have a more realistic data communication in the simulation of the process, the simulation PC and the IPC are networked, and the point cloud data is sent and received over the implemented emulated network stream. An instance of the simulation in Unity, ROS2, and Livox Viewer can be seen in Figure 9 , which shows that the point cloud data is successfully sent from the Unity (simulation PC), received by ROS2 (IPC), and optionally visualized by Livox Viewer (IPC) which is the official software provided by the manufacturer to use with real sensors.

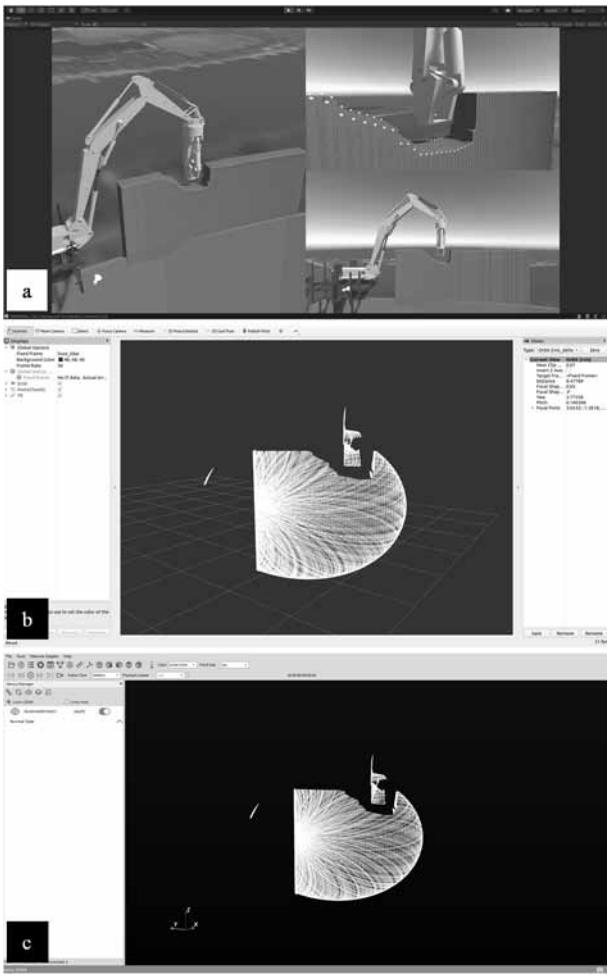

Figure 9: An instance of the simulation environment in: a) Unity software on the simulation PC, b) ROS2 framework on IPC, c) Livox Viewer on IPC. The point cloud data is accumulated for 3 seconds.

3 CAN-Communciation

The real machine is equipped with two CAN bus systems. The first CAN bus features the sensor data of the angle sensors. All the sensors attached to the working equipment are CANOpen based [5].

With the help of a USB-to-CAN interface and the corresponding API, the simulation model feeds its sensor data with a real CAN bus. A subset of the CANOpen-slave functionality is implemented in order to provide the same CAN-bus protocol as the real sensors in the expected operation mode.

The second CAN bus serves as an interface between the ECU to control the hydraulic system and the IPC that processes the point-cloud data and implements the overall automation strategy. A custom CAN protocol has been developed to send the data of the desired trajectory from the IPC to the ECU and to transmit status and control messages between the automation authority and the control ECU.

In order to test the CAN communication on the sensor bus as well as to test and modify the custom CAN protocol, the simulation model which implements the emulated CAN interface is a helpful tool.

The emulation of the CAN communication which is fed by a comprehensive system simulation provides several advantages:

- Software development and interface specification can be done without the need of the real system.
- The simulation can be distributed several times whereas the real machine exists only once.
- Simulation-based but realistic data is a feasible way to provide a vast amount of test-cases for test-driven development.
- A simulator can facilitate the reconstruction of dangerous or complex situations.

4 Tele-Remote Operation

The machine is supposed to be controlled remotely. In previous applications, the machine was controlled with a handheld device with a direct line of sight. A camera-based solution to control the excavator from a control cabin has been rejected by the operators because of the high delay of the video streaming which makes the tele-operation inefficient.

Besides the latency issue, camera-based tele-remote control solutions have several other drawbacks:

- The 2D image of a video hinders the depth perception of the operation which is important particularly when moving the working attachment of an excavator
- The camera position is fixed and does not allow a moving point of view.
- Optical cameras only work in good light conditions and in the absence of fog and dust.
- Compression and Decompression of digital video signals introduces additional delay and requires dedicated hardware
- Data throughput increases with resolution and number of cameras

In order to provide an efficient solution for tele-remote operation, a digital twin approach is implemented. All sensor signals are used to visualize a 3D model of the machine. The amount of data to represent all degrees of freedom of the excavator is comparatively low (9 × Int16 at 50Hz). As the raw point cloud data stream requires a performant wireless link, a post-processing algorithm is implemented on the machine's IPC. This reduces the throughput to transmit the point data drastically (300 × Int16 at 10Hz).

The basic idea to reduce the point cloud data is to extract only the relevant information to control the demolishing process, i.e. the top edge of the remaining wall. Different filter algorithms are applied to the point cloud to compute an array of points in a discrete spacing to indicate the top-right edge of the wall. Figure 10 shows the result of the edge detection algorithm based on the simulated LiDAR data.

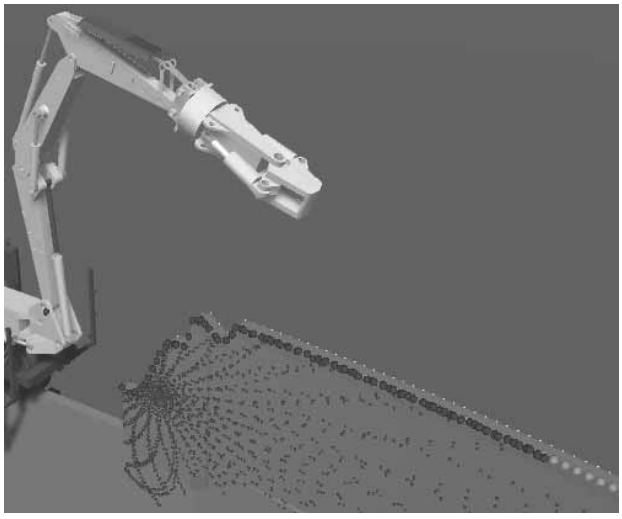

Figure 10: Edge detection for the simulated wall under demolition. The red spheres indicate the detected points of the top-right wall edge.

To provide a real-time tele-remote visualization, the 3D visualization of the simulator is reused to visualize the machine and point cloud data. Based on the detected points on the top-right edge of the wall, a mesh is created that resembles the real wall. Considering the fixed wall thickness and using radial extrapolation of the given points, a textured volume can be constructed to visualize the wall segment within reach of the excavator.

This approach to visualize this particular machine-process interaction has several advantages:

- The overall amount of data to visualize all relevant information is relatively low compared to multiple high-definition video streams. This reduces delay.
- The virtual camera that is used to render the scene can be moved by the user during operation or several virtual cameras can be used to visualize the scene from different locations.
- Additional information can be computed and displayed in the scene, of which position coordinates of the end effector, limits of the working area, recommendations to align the tool efficiently, to name but a few, are of great importance.
- To enhance depth perception of the scene, the visualization could be rendered for a VR-Headset. As this is not feasible for long operations, a head tracking-based camera movement has been tested to align the view with the head movement of the operator and to create motion parallax.

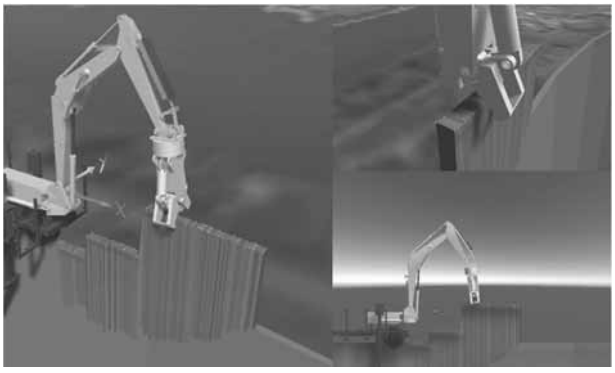

Figure 11: Visualization of real machine data and real sensor data for tele-remote operation with 3 different camera positions.
The left bottom image shows a side view with an orthographic projection.

5 Conclusion

This paper describes the components of a 3D-simulation environment for a demolition excavator, which can also serve as a visualization tool for a real-time tele-remote operation. The simulation comprises a 3D kinematic model for the excavator, an emulated Livox LiDAR, CAN bus-based interfaces for Hardware-in-the-loop applications, and a deformable wall segment that resembles a demolishable wall. The simulation is a helpful tool to develop and test the communication protocol, the software for automation and machine control as well as the point cloud processing. The majority of the simulation environment has been reused to serve as a real-time visualization tool for a tele-remote operation that has several advantages compared to a conventional camera-based view of the process.

In the ongoing project, a partly automation of the demolition process has to be developed. Therefore, a trajectory planner and a process strategy have to be implemented and tested. As the real demolition of a wall is a laborious and costly procedure, the simulation-based development of the process routine is an invaluable method.

Acknowledgement

This Project is supported by the Federal Ministry for Economic Affairs and Energy (BMWi) on the basis of a decision by the German Bundestag.

References

[1] Unity Technologies, 14 10 2021. [Online]. Available: www.unity.com

[2] Waurich V, Weber J. Interactive FMU-Based Visualization for an Early Design Experience. In *Proceedings of the 12th international Modelica Conference*, Prague, 2017.

[3] Livox Technology, [Online]. Available: https://www.livoxtech.com/de/mid-70. [Access am 15 10 2021]

[4] CiA, „www.can-cia.org,“ [Online]. Available: https://www.can-cia.org/canopen/. [Access 26 10 2021]

[5] V. Waurich V. Simulation der automatisierten Schaufelbefüllung eines Radladers mit Modelica. In *Tagungsband ASim Workshop STS/GMMS*, Heilbronn, 2018

Parameter Tables for PID Controllers for Time- delayed Systems Optimized with a Learning Method

Roland Büchi

Zurich University of Applied Sciences, School of Engineering, Technikumstrasse 9, CH 8401 Winterthur, Switzerland; *roland.buechi@zhaw.ch*

SNE 33(3), 2023, 117-123, DOI: 10.11128/sne.33.tn.10653
Selected ASIM SST 2022 Postconf. Publication: 2023-08-15;
Received Revised Improved: 2023-10-20; Accepted: 2023-10-22
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. This publication provides useful parameter sets in tabular form for PID controllers for various rise and dead times of step responses of asymptotically stable control systems, which minimize the common quality criteria in the time domain, integral of absolute error IAE, integral of time-multiplied absolute value of error ITAE and integral of squared error ISE. Since the determination of the parameter sets is very computationally intensive, an approach from the field of artificial intelligence was chosen.
The application of the parameter sets found is verified with examples. The parameter sets also take into account the controller output limitations that are relevant in practice and can basically be used for all PID controllers of controlled systems with a time delay.

1 Introduction and Related Work

PID controllers are still by far the most frequently used controller structures for single-in, single-out (SISO) systems. The control of controlled systems with dead time is challenging. The parameters found with heuristic methods lead to aysmptotically stable systems. The most famous of them are those from Ziegler Nichols, Latzel, or Chien Hrones and Reswick. However, all parameters found with these methods still have to be readjusted in the practical system so that sensible transient behavior results.

The time-delayed systems are very common in practice. They require special demands, because their control is challenging. In practice, however, they are very common, especially in process engineering or in thermal systems, since the sensor often can not be placed directly next to the actuator.

There are different approaches known for finding PID controller parameters from step responses of time-delayed systems. All of them result in stable control systems. Especially, as the dead time increases, it becomes difficult to find suitable PID parameter sets. There are some heuristic methods for this, which can be used in the time and frequency domain. However, these parameters must be further optimized afterwards. The first approach was the parameter set from Ziegler Nichols [1]. There are also several others existing, for example Chien, Hrones and Reswick [2].

For further optimization, there are several methods used, also some from the field of artificial intelligence, for example particle swarm optimization, PSO [3]. In this paper, the method hill climbing [4] is used. It is another stochastic method for optimizing controllers, but it is related to PSO. For optimizing controller parameters, there also other approaches known [5]–[9], [13]–[15].

In order to be able to handle time-delayed systems in terms of simulation at all, the turning point tangent method is often used. A PTn system is identified with *n* PT1 (1st order) elements connected in series, which have identical time constants. They are dealt with in the literature [10], [11], [12]. The PID parameter tables, which are described and used in the next chapters, however, refer to these PTn systems with identical time constants. Such systems are very common and can be found in all engineering disciplines. The series connection of such PTn systems according to formula (1) leads to step responses which are delayed. In particular, the dead times can be approximated with linear models in this way.

Here, K_s is the static gain, n is the system order and T_1 is the time constant of the *n* identical PT1 elements.

$$\frac{K_s}{(s \cdot T_1 + 1)^n} \tag{1}$$

2 Identification of PTn

The turning point tangent should be used here as a reference for identification. In many cases, one can simply measure the delay time T_u and the rise time T_g according to Figure 1 by placing a tangent at the point of inflection. From this one can identify the number n of PT1 elements connected in series and their identical time constants T_1. The measurement of the step response of a controlled system can then be dealt with using Table 1, which is well known from literature [10].

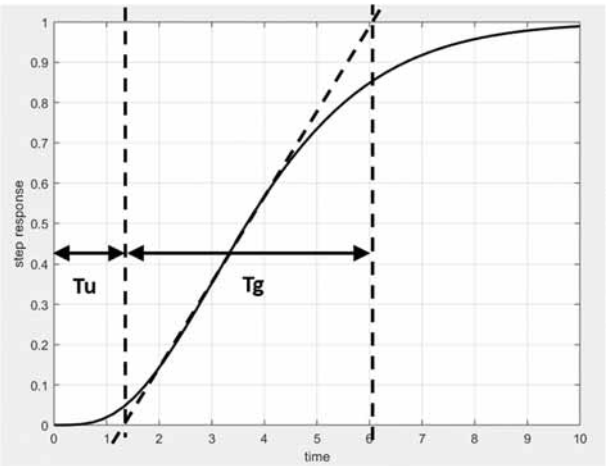

Figure 1: Step response of a PTn- Systems, turning point tangent and subdivision into T_u and T_g.

n, PTn	Tg/T1	Tu/T1	Tg/Tu
2, PT2	2.72	0.28	9.65
3, PT3	3.69	0.81	4.59
4, PT4	4.46	1.43	3.13
5, PT5	5.12	2.10	2.44
6, PT6	5.70	2.81	2.03

Table 1: Calculation of T_g, T_u, T_1 and PTn

The parameters in Table 1 can be calculated using formulas (2) to (4), for different system order n.

$$\frac{T_g}{T_1} = \frac{(n-1)!}{(n-1)^{n-1}} \cdot e^{n-1} \quad (2)$$

$$\frac{T_u}{T_1} = n - 1 - \frac{(n-1)!}{(n-1)^{n-1}} \cdot [e^{n-1} - \sum_{m=0}^{n-1} \frac{(n-1)^m}{m!}] \quad (3)$$

$$\frac{T_g}{T_u} = \frac{\frac{(n-1)!}{(n-1)^{n-1}} \cdot e^{n-1}}{n - 1 - \frac{(n-1)!}{(n-1)^{n-1}} \cdot [e^{n-1} - \sum_{m=0}^{n-1} \frac{(n-1)^m}{m!}]} \quad (4)$$

3 ITAE, IAE and ISE Criterions

The block diagram of the controlled system is shown in Figure 2. The parameters found for the PTn system are K_s, $T_1$and n.

Among others, the criteria IAE, ITAE and ISE are used for optimizing, which describe the error area of a step res-ponse of the controlled system. These error areas are shown in Figure 3.

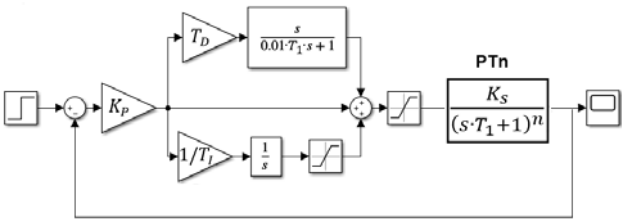

Figure 2: Block diagram of a PTn- systems, which is controlled by a PID controller.

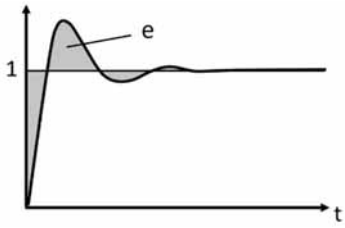

Figure 3: Error area in the transient response of the closed-loop system according to Figure 2, for calculating the IAE, ITAE and ISE criterions.

IAE means integral of absolute error, ITAE means integral of time-multiplied absolute value of error and ISE means integral of squared error. It can be seen from this, that the IAE criterion calculates the amount of the error area.

The ITAE criterion is an extension of the IAE criterion and also takes time into account. Thus, the error area is weighted more heavily as time progresses. The two criterions IAE and ITAE are also called the L1 criterion.

The ISE criterion does not calculate the error area, but its square. This means, that there it is no need to calculate the absolute values of the error area, since negative signs cancel each other out when squared. The ISE criterion is also called the L2 criterion.

$$IAE: \int_0^\infty |e(t) - e(\infty)|\, dt \tag{5}$$

$$IITAE: \int_0^\infty |e(t) - e(\infty)| \cdot t \cdot dt \tag{6}$$

$$ISE: \int_0^\infty [e(t) - e(\infty)]^2\, dt \tag{7}$$

4 Calculation of PID Parameters with the Hill Climbing Method

Now, since in Chapter 1 the time-delayed systems were approximated with the turning point tangent method as PTn elements, the quality criteria for step responses calculated for all orders *n* and also all PID controller parameters K_p, T_i and T_d.

The parameters which correspond to the minimum of the criteria can then be displayed as table values. The problem is that this has to be done for a multidimensional space (order *n*, K_p , T_i ,T_d, quality criteria). So it would take far too long with the computing power available today. Therefore, 'hill climbing', a method from the field of artificial intelligence was chosen [4].

With this method, heuristic functions are added to some of the parameters, in this case the parameters of the PID controller, and then it is calculated whether the quality criterions IAE, ITAE and ISE have become smaller. If there is, the new parameters will be used as a reference. If not, the old ones stay. In this way and after many iterations, the final values of the parameters remain at local minima of the quality criteria.

The method requires much less computing time than a complete calculation in multi-dimensional space, for example with nested loops of all parameters. With the 5 parameters, order *n*, K_p , T_i ,T_d, quality criteria, and this would have the time complexity $f(n) = O(x^5)$

However, since it only finds local minima, several different random tuples of starting values for the control parameters are used. Since many of the results of the converged parameters for the minimal quality criteria then agree with one another, it can be assumed with reasonably good certainty that the parameters found are actually PID parameters K_p , T_i and T_d, which either correspond to the absolute minima of the criterions, or which come very close to these at least.

The search for optimal parameters in multi-dimensional space, as with this specific problem in control engineering, is also one of the good arguments for using an artificial intelligence method here as well. Often, complete calculations cannot be carried out in the entire parameter space due to the computing power available. Since only part of the parameter space is calculated with such methods, the computing time is significantly reduced and the results are parameter sets for excellent transient behavior.

5 PID Controller Parameters after Minimization of the Quality Criterions IAE, ITAE and ISE

This chapter is the essence of the publication. The table below describes the PID controller parameters calculated with Matlab/Simulink and the 'hill climbing' method according to the minimized quality criteria IAE, ITAE and ISE. The block diagram in Figure 2 serves as the basis for this.

It is particularly noteworthy that the static gain K_s and in particular the time constant T_1 of the *n* identical PT1 elements are included in the table. This makes them usable and scalable for all PTn systems. The values up to $n = 6$ are shown here.

The controller output limitation is implemented on the one hand after the controller and on the other hand also after the integrator (anti windup) and is assumed to be ± 2, ±3, ± 5, ±10. In the calculations the anti windup is never active, but it is inserted anyway, because in practice it can happen for various reasons that the controlled system does not reach the desired value in the static end value.

The controller output limitation is calculated as (maximum possible controller output - controller output before the step) divided by (controller output for the stationary end value - controller output before the step). In many cases, the controller output before the step is equal to 0, thus the controller output limitation is calculated as maximum possible controller output divided by controller output for the stationary end value. In the table, the maximum parameter value is limited to 10.

PT1	+/- 2	+/- 3	+/- 5	+/- 10
IAE	Kp·Ks = 10 Ti = 3.1·T1 Td = 0 (PI)	Kp·Ks = 10 Ti = 2·T1 Td = 0 (PI)	Kp·Ks = 10 Ti = 1.3·T1 Td = 0 (PI)	Kp·Ks = 10 Ti = 1·T1 Td = 0 (PI)
ITAE	Kp·Ks = 9.3 Ti = 2.9·T1 Td = 0 (PI)	Kp·Ks = 9.5 Ti = 1.9·T1 Td = 0 (PI)	Kp·Ks = 9.1 Ti = 1.2·T1 Td = 0 (PI)	Kp·Ks = 10 Ti = 1·T1 Td = 0 (PI)
ISE	Kp·Ks = 10 Ti = 2.7·T1 Td = 0 (PI)	Kp·Ks = 10 Ti = 1.6·T1 Td = 0 (PI)	Kp·Ks = 9.8 Ti = 1.5·T1 Td = 0 (PI)	Kp·Ks = 10 Ti = 0.2·T1 Td = 0 (PI)

PT2	+/- 2	+/- 3	+/- 5	+/- 10
Tg/Tu: 9.65				
IAE	Kp·Ks = 10 Ti = 9.6·T1 Td = 0.3·T1	Kp·Ks = 10 Ti = 7.3·T1 Td = 0.3·T1	Kp·Ks = 10 Ti = 5.6·T1 Td = 0.3·T1	Kp·Ks = 10 Ti = 3.7·T1 Td = 0.2·T1
ITAE	Kp·Ks = 10 Ti = 9.6·T1 Td = 0.3·T1	Kp·Ks = 10 Ti = 7.3·T1 Td = 0.3·T1	Kp·Ks = 9.6 Ti = 5.4·T1 Td = 0.3·T1	Kp·Ks = 9.8 Ti = 4.7·T1 Td = 0.3·T1
ISE	Kp·Ks = 10 Ti = 9.7·T1 Td = 0.2·T1	Kp·Ks = 10 Ti = 7.3·T1 Td = 0.2·T1	Kp·Ks = 10 Ti = 5.1·T1 Td = 0.2·T1	Kp·Ks = 10 Ti = 4.6·T1 Td = 0.1·T1

PT3	+/- 2	+/- 3	+/- 5	+/- 10
Tg/Tu: 4.59				
IAE	Kp·Ks = 5.4 Ti = 9.4·T1 Td = 0.7·T1	Kp·Ks = 7 Ti = 10·T1 Td = 0.7·T1	Kp·Ks = 8.4 Ti = 9.8·T1 Td = 0.7·T1	Kp·Ks = 10 Ti = 9.7·T1 Td = 0.7·T1
ITAE	Kp·Ks = 5.4 Ti = 9.4·T1 Td = 0.7·T1	Kp·Ks = 7 Ti = 10·T1 Td = 0.7·T1	Kp·Ks = 8.2 Ti = 9.6·T1 Td = 0.7·T1	Kp·Ks = 10 Ti = 9.7·T1 Td = 0.7·T1
ISE	Kp·Ks = 6.1 Ti = 10·T1 Td = 0.6·T1	Kp·Ks = 8.1 Ti = 9.8·T1 Td = 0.6·T1	Kp·Ks = 10 Ti = 10·T1 Td = 0.6·T1	Kp·Ks = 10 Ti = 7.8·T1 Td = 0.6·T1

PT4	+/- 2	+/- 3	+/- 5	+/- 10
Tg/Tu: 3.13				
IAE	Kp·Ks = 2 Ti = 5.2·T1 Td = 1.1·T1	Kp·Ks = 2.9 Ti = 6.5·T1 Td = 1.2·T1	Kp·Ks = 3.3 Ti = 7.1·T1 Td = 1.3·T1	Kp·Ks = 3.3 Ti = 6.9·T1 Td = 1.3·T1
ITAE	Kp·Ks = 1.9 Ti = 5·T1 Td = 1.1·T1	Kp·Ks = 2.4 Ti = 5.9·T1 Td = 1.2·T1	Kp·Ks = 2.3 Ti = 5.7·T1 Td = 1.2·T1	Kp·Ks = 2.1 Ti = 5·T1 Td = 1.1·T1
ISE	Kp·Ks = 2.8 Ti = 6.6·T1 Td = 1.2·T1	Kp·Ks = 3.6 Ti = 7·T1 Td = 1.2·T1	Kp·Ks = 4.9 Ti = 7.1·T1 Td = 1.4·T1	Kp·Ks = 5.2 Ti = 7·T1 Td = 1.4·T1

PT5	+/- 2	+/- 3	+/- 5	+/- 10
Tg/Tu: 2.44				
IAE	Kp·Ks = 1.7 Ti = 5.8·T1 Td = 1.6·T1	Kp·Ks = 1.8 Ti = 5.9·T1 Td = 1.6·T1	Kp·Ks = 1.8 Ti = 5.8·T1 Td = 1.6·T1	Kp·Ks = 1.7 Ti = 5.5·T1 Td = 1.6·T1
ITAE	Kp·Ks = 1.4 Ti = 5.3·T1 Td = 1.4·T1	Kp·Ks = 1.4 Ti = 5.2·T1 Td = 1.4·T1	Kp·Ks = 1.4 Ti = 5.2·T1 Td = 1.4·T1	Kp·Ks = 1.4 Ti = 5.0·T1 Td = 1.4·T1
ISE	Kp·Ks = 1.9 Ti = 5.9·T1 Td = 1.7·T1	Kp·Ks = 2.6 Ti = 6.5·T1 Td = 1.8·T1	Kp·Ks = 2.5 Ti = 6.3·T1 Td = 1.8·T1	Kp·Ks = 2.5 Ti = 6.1·T1 Td = 1.8·T1

PT6	+/- 2	+/- 3	+/- 5	+/- 10
Tg/Tu: 2.03				
IAE	Kp·Ks = 1.3 Ti = 5.9·T1 Td = 1.9·T1	Kp·Ks = 1.3 Ti = 5.8·T1 Td = 1.9·T1	Kp·Ks = 1.3 Ti = 5.8·T1 Td = 1.9·T1	Kp·Ks = 1.3 Ti = 5.6·T1 Td = 1.9·T1
ITAE	Kp·Ks = 1.1 Ti = 5.5·T1 Td =1.7·T1	Kp·Ks = 1.1 Ti = 5.5·T1 Td = 1.7·T1	Kp·Ks = 1.1 Ti = 5.4·T1 Td = 1.7·T1	Kp·Ks = 1.1 Ti = 5.3·T1 Td = 1.7·T1
ISE	Kp·Ks = 1.8 Ti = 6.8·T1 Td = 2.1·T1	Kp·Ks = 1.8 Ti = 6.5·T1 Td = 2.1·T1	Kp·Ks = 1.8 Ti = 6.5·T1 Td = 2.1·T1	Kp·Ks = 1.8 Ti = 6.3·T1 Td = 2.1·T1

Table 2: Table values of the PID parameters for the minimum IAE, ITAE and ISE criterions of controlled PTn or time delayed systems.

It is noteworthy that the table scales with T_1 and K_s. The results are therefore very widely applicable.

6 Applications of Table Values

6.1 Control of a PT3 System

In the first application example, a didactic example is used to show the general usability of the parameter table. The response of a time-delayed system to a unit jump shows a static end value of 1, a delay time T_u of 0.81 seconds and a rise time T_g of 3.69 seconds.

This results in $T_g/T_u = 4.59$ and this results in a PT3 behavior with $K_s = 1$ and $T_1 = 1$ second.

For the ITAE criterion, the table values of the PID parameters for the PT3 system are read off. Since $K_s = 1$ and T_1= 1s, the table values are multiplied by 1 and therefore correspond to those for the controller parameters K_p, T_i, and T_d.

The simulation of the step responses of the closed loop system according to Figure 2 is shown in Figure 4. It shows a very nice transient response.

The different dynamics or rise times can be explained with the different controller output limitations. This also shows very well that these must be included into the controller design.

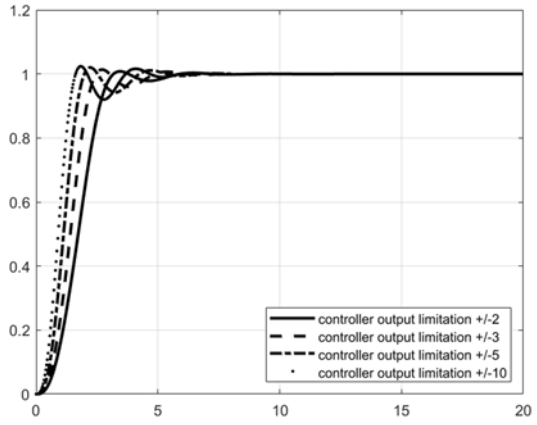

Figure 4: Step response of the closed loop system according to figure 2 for a PT3 system with the PID table values for the ITAE criterion.

6.2 Control of a PT2 System, Comparison with Ziegler Nichols and Chien, Hrones and Reswick

In the following, the controller parameters found are compared with those of Ziegler-Nichols and Chien, Hrones and Reswick, using an example of second order. The used system has an order n = 2 and a time constant T_1 of 8 s. As a comparison to practical systems this could be a thermal system, where the heating coil and the temperature sensor are not exactly located at the same place.

$$\frac{K_s}{(s \cdot T_1 + 1)^n} = \frac{1}{(s \cdot 8 + 1)^2} \qquad (8)$$

The controller output signal for the stationary end value of the controlled system is 1 and the controller output signal is limited to ± 2, which results in a controller output limitation factor ± 2. Using the Table 1, it results for PT2 a T_g = 21.76 seconds and T_u = 2.24 seconds

According to the Ziegler-Nichols step response method, controlled systems with dead time and a PT1 are treated. In this case, T_u is assumed to be the dead time and T_g as the time constant.

This results in the controller parameters:

$$Kp = \frac{1.2 \cdot Tg}{Ks \cdot Tu} = 11.65, \quad Ti = 2 \cdot Tu = 4.48s$$

$$Td = 0.5 \cdot Tu = 1.12s$$

According to Chien, Hrones and Reswick with the parameters for 'aperiodic', the result is:

$$Kp = \frac{0.6 \cdot Tg}{Ks \cdot Tu} = 5.83, \qquad Ti = 1.0 \cdot Tg = 21.76s$$

$$Td = 0.5 \cdot Tu = 1.12s$$

The method calculated above with the parameters according to the minimal ITAE criterion provides a T_1 of 8s and n = 2 according to Table 1, i.e. a PT2 behavior. This results in the following parameters from Table 2:

$$Kp = \frac{10}{Ks} = 10, \quad Ti = 9.6 \cdot T1 = 76.8s,$$

$$Td = 0.3 \cdot T1 = 2.4s$$

The simulation according to the block diagram according to Figure 2 (PT2 with PID) shows the results according to Figure 5 for the three parameter sets.

The rise time is similar for all three parameter sets, because all systems run into the controller output limitation in this phase. It shows very nicely that the calculated values with the minimum ITAE criterion according to Table 2 show an excellent transient behavior.

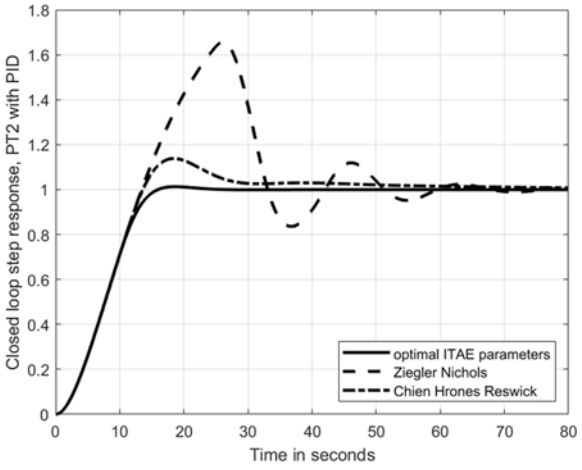

Figure 5: Comparison of the step responses with the control parameters for a PT2 plant, with a control output limitation factor ±2.

6.3 Control of a General PT4 System, Comparison with $K_S \neq 1$ and $T_1 \neq 1$

In the next example, a general problem is dealt with in order to also show the scalability of the presented parameter table. There was measured the response to a unit step and it is shown in Figure 6.

With the application of the turning point tangent method according to Figure 1 and Table 1, the step response leads to the following transfer function, with values K_s = 0.4 and T_1 = 0.5 s :

$$G = \frac{0.4}{(s \cdot 0.5 + 1)^4} \quad (9)$$

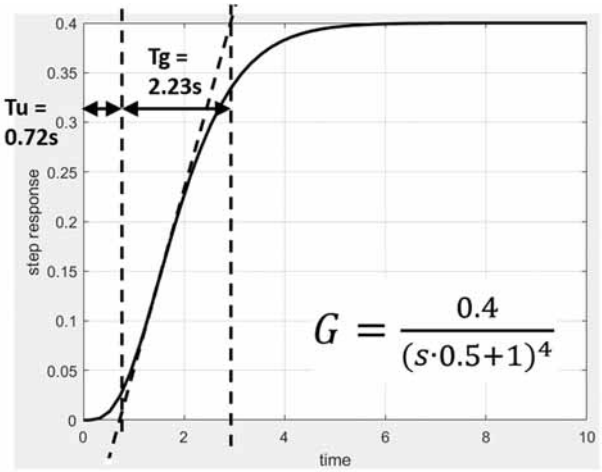

Figure 6: Step response of a PT4 system.

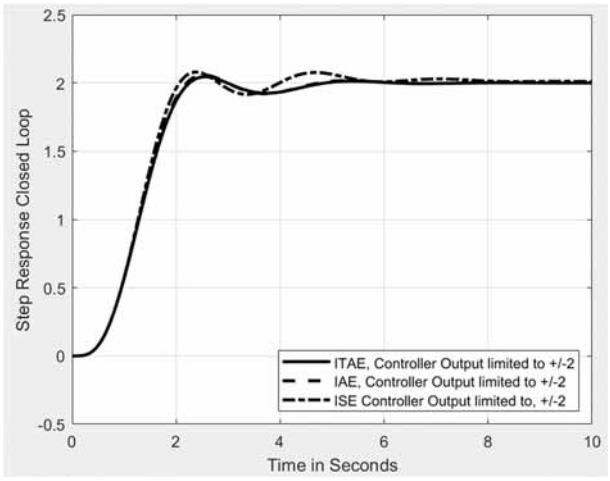

Figure 7: Response of a closed loop, PID with PT4 for a step from 0 to 2

One would like to design the system with a PID controller according to Figure 2 and execute a setpoint jump from 0 to 2. Since the static gain K_s = 0.4, the controller output for the stationary end value is then 2/ K_s, or 2/0.4 = 5. Assuming that the controller output is limited to ± 10, the result is a controller output limitation factor ± 10/5 = ± 2. The controller parameters are to be calculated for the IAE criterion as an example.

$$Kp = \frac{2}{Ks} = 5 \quad Ti = 5.2 \cdot T1 = 2.6s$$

$$Td = 1.1 \cdot T1 = 0.55s$$

For minimizing the criteria IAE, ITAE and ISE, this results in the closed loop behavior according to figure 7 for a setpoint jump from 0 to 2.

7 Discussion and Outlook

Good transient behavior can be seen for all parameter sets in the table. Compared with heuristic methods, these parameters are hard-calculated values that minimize the quality criteria. It is also up to the discussion what would happen if one would perform different steps and therefore had to choose the parameters according to different factors of the controller output limitation. The parameters are very similar, however, and values for jumps should be selected which are most likely to occur in the specific system. Even for the general step by any value, the parameters still give very good transient behavior.

An exciting finding emerges from the discussion of the question of how the parameters K_p, T_i and T_d develop for changing ratios T_g/T_u (i.e. rise time in relation to the delay time).

The smaller the ratio, the greater the delay time in relation to the rise time, the smaller K_p on the one hand and the greater Td on the other hand.

The effect of a small K_p means that the system can only be regulated slowly. In the literature [10] this is also described in such a way that the controllability for systems with longer dead times is reduced. If one would also plot the controller output, one would see that this is also only relatively small. Therefore, it is of no use in these systems if an additional regulator reserve is made available through amplifiers because this cannot be used at all due to the time delay of the system.

If you follow the development of the value of K_p in the tables, then with smaller ratios T_g/T_u (or larger orders n of the PTn systems and thus larger dead times) and smaller K_p, greater system dynamics are no longer achieved. On the one hand, the controlled system shows a nice transient response according to the minimized quality criteria and, in particular, also reaches the setpoint in the stationary end value, which is often sufficient in practice. When looking at the differential component of the controller T_d, it becomes apparent that a differential component for optimizing the quality criteria is missing when regulating frequently occurring PT1 elements, i.e. a pure PI controller is already optimal.

With an increasing system order, i.e. a decreasing ratio T_g/T_u or a larger delay, the required D component (T_d) becomes larger and larger.

It turns out that the PTn systems that occur very frequently in practice can be regulated very well with the table values available according to the minimized IAE, ITAE and ISE criteria. In practice, you can often do without a simulation and only measure the step response of the system. Then the order n and the associated parameters for the PID controller can be read from the table, also using K_s and T_1 and implement the controller directly on the system.

References

[1] Ziegler JB, Nichols NB. Optimum settings for automatic controllers. ASME Transactions, v64 (1942), pp. 759-768

[2] Chien KL, Hrones JA, Reswick JB. On the Automatic Control of Generalized Passive Systems. In: Transactions of the American Society of Mechanical Engineers., Bd. 74, Cambridge (Mass.), USA, Feb. 1952, S. 175–185

[3] Qi Z, Qian S, Hui Z. Tuning of digital PID controllers using particle swarm optimization algorithm for a CAN-based DC motor subject to stochastic delays. IEEE Transactions on Industrial Electronics 67.7 (2019): 5637-5646.

[4] Russel SJ, Norvig P. Artificial Intelligence: A Modern Approach. (2nd edition), Upper Saddle River, New Jersey: Prentice Hall, pp 111- 114, ISBN 0-13-790395-2, 2003

[5] Joseph EA, Olaiya OO. Cohen-Coon PID Tuning Method, A Better Option to Ziegler Nichols- PID Tuning Method. Computer Engineering and Intelligent Systems, ISSN 2222-1719, Vol. 9, No. 5, 2018

[6] Ozana S, Docekal T. PID Controller Design Based on Global Optimization Technique with Additional Constraints. Journal of Electrical Engineering, Vol. 67, No3, pp 160 – 168, 2016

[7] Hussain KM, et al. Comparison of PID Controller Tuning Methods with Genetic Algorithm for FOPTD System. Int. Journal of Engineering Research and Applications, ISSN: 2248-9622, Vol. 4, Issue 2, pp. 308-314, February 2014

[8] Büchi R. State space control, LQR and observer: step by step introduction with Matlab examples. Norderstedt Books on Demand, 2010.

[9] da Silva LR, Flesch RC, Normey-Rico JE. Controlling industrial dead-time systems: When to use a PID or an advanced controller. ISA transactions. 2020 Apr 1;99:339-50.

[10] Unbehauen H. Regelungstechnik. Braunschweig: Vieweg; 1992.

[11] Zacher S, Reuter M. Regelungstechnik für Ingenieure. 15.Auflage, Springer Vieweg Verlag, 2017

[12] Schwarze G. Bestimmung der regelunsgtechnischen Kennwerte von P-Gliedern aus der Übergangsfunktion ohne Wendetangentenkonstruktion, In: - messen-steuern-regeln Heft 5, S. 447-449, 1962

[13] Martins FG. Tuning PID Controllers using the ITAE Criterion. Int. Journal Engineering, Edition. Vol. 21, No. 5. Pp. 867-873, 2005

[14] Büchi R. Optimal ITAE criterion PID parameters for PTn plants found with a machine learning approach. 2021 9th International Conference on Control, Mechatronics and Automation (ICCMA). IEEE, 2021

[15] Silva GJ, Datta A, Bhattacharyya SP. PID Controllers for Time-Delay Systems. Boston. ISBN 0-8176-4266- 8.2005.

ERS - Enterprise Resource Simulator: a New Simulation Platform

Mark Oostveen, Michel Hofmeijer*, Fred Jansma

ERS development team, Incontrol Enterprise Dynamics, Jaap Bijzerweg 21 A, 3446 CR Woerden, The Netherlands; *Michel.Hofmeijer@incontrolsim.com

SNE 33(3), 2023, 125-132, DOI: 10.11128/sne.33.sw.10654
Received: 2023-10-18
Accepted: 2023-10-22
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. ERS is a new simulation platform that allows you to develop and run simulations fully utilizing modern hardware. ERS supports multi-formalism in a simulation model and utilizes a new mechanism to leverage new techniques to scale models without fundamental size limits. The ERS Platform provides development tools alongside the simulation engine. ERS aims to integrate with other tools and platforms. ERS allows the development of tailor-made applications and libraries based on the engine. Those libraries are, in principle, interoperable unless specified. This allows experts in the field to create plug & play applications and libraries to share inside the ERS ecosystem, including in specialized fields like Material handling, logistics, crowd management, chemical materials, etc.

Introduction

Enterprise Resource Simulator (ERS) is the new simulation platform developed by InControl (Enterprise-Dynamics). ERS provides an environment to develop, maintain, and run a significant variation of custom state-of-the-art simulation applications. ERS provides these applications with a new powerful engine that allows them to simulate what they need without worrying about the most technical aspects of building a simulation application. Using ERS, users can build applications with their expertise while using InControl's simulation expertise. By building your own application on top of the ers-core, you can have a large degree of freedom in how your application simulates and runs.

ERS does not just allow the applications to be built with the current state-of-the-art simulation capabilities but advances that state-of-the-art based on the demands of industry and science alike. ERS does this by enabling a skilled developer to create large-scale applications that can run with the proper computational resources. The new platform does this by enabling users to efficiently split Models so that computational resources can be used to their full potential.

ERS provides access to state-of-the-art simultaneous computation and multi-formalism in one Model. The broad possibilities of ERS allow the developer to build applications that can intuitively simulate the user's problems without worrying if it fits the formalism chosen by the platform.

This paper will explain how ERS works and how using ERS can benefit the developer of simulation applications and the user. We will explain the use case of ERS. Also, we will explain how ERS works from a technical point of view and why we choose the design of ERS. Later in the paper, we will explain how this setup allows us to develop applications that can have Models with multiple formalisms within one Model. Lastly, we will explain why the technical setup will lead to good performance and scalability.

1 Technical Overview

In this chapter, we will discuss the technical architecture of ERS. While ERS as a platform has many features and abilities, the most important for this paper is the core simulation abilities of ERS. ERS does not define the logic of the Model but does still run the Model. ERS provides specific built-in tools that enable ERS to run complex Models efficiently.

1.1 Model Structure

Before we discuss how ERS works, we need to introduce four core concepts of the architecture of ERS. The first of these concepts is the Model.

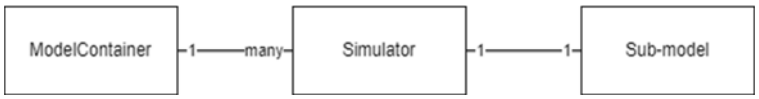

Figure 1: View of relationships between concepts, specifically between Sub-Model, Simulator, and Model.

The **Model** is the environment that ensures that all parts of the simulation-model are synced, it handles the run and the communication within the simulation-model. If two Models are loaded into ERS at the same time, they function completely separately. The Model is essential for implementing multi-formalism through a Lookahead-Table.

All data needs to be inside the Model or have a connection explicitly defined in the Model. One default connection mandatorily created is with the Shared Space. The **Shared Space** is unique per Model which can contain predefined objects accessible in all Sub-Models. The Shared Space can also contain assets that the Sub-Models share. This can be, for example, 3D models, shared functions, or shared static data. The only limitation is that objects in the shared space should not be changed during run time. The limitation on changing objects in the Shared Space at runtime is to enable ERS to parallelize execution automatically. A Model can be multi-formalistic and can be very complex. The smaller constituent parts of the Model are called Sub-Models. The Model is also where the initial random number generator is responsible for creating a valid state in the Sub-Models, ensuring determinism.

A **Sub-Model** is a largely independent uni-formalistic part of the Model. What would be considered a complete Model in most software applications would be considered a Sub-Model in ERS. Sub-Models are extremely flexible, almost anything could be in a Sub-Model, and their primary purpose is to use the efficient computation possibilities and multi-formalism built into ERS. Sub-Models implement their random number generator to ensure determinism in the Model. Sub-Models can contain entire physical environments, or they could simply contain a single algorithm. The decision to create a Sub-Model should primarily depend on how separate the new Sub-Model will be from the overall Model.

The **Simulator** is the object that runs a Sub-Model and manages the sync-events and time within its Sub-Model. Each Simulator has one unique Sub-Model. Each Simulator is uni-formalistic and communicates with the Model to sync the simulation-model.

The Simulator has its own internal time system, which regulates the uni-formalistic Sub-Model.

For a Simulator to be used in a Model, it is tuned to a specific formalism to handle time correctly. The Simulator can run largely independently and only needs to stop when it syncs with the rest of the Model. This makes multi-threading more efficient, as when the Sub-Models are mostly separate, they can work on separate threads efficiently, reducing overhead, and conserving data bandwidth.

1.2 Jobsystem, Lookahead-Table and Syncing

In ERS, the Models can be extensive, and many processes can be scheduled and executed simultaneously. To manage all these different tasks, possibly at the same time, we have implemented a JobSystem. This system schedules all the tasks that follow the user's logic so that the computation resources are kept sufficiently busy. The system can be divided for ease of understanding into two parts: jobs within a Sub-Model and jobs that do not belong to one Sub-Model.

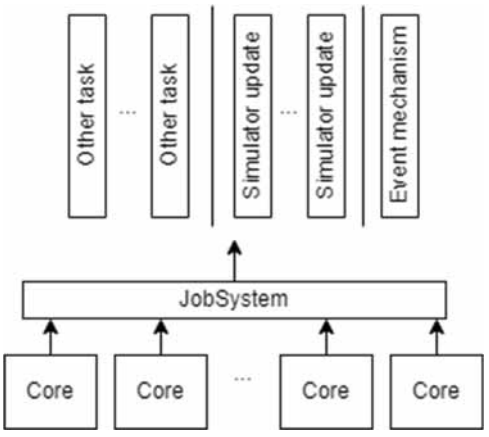

Figure 2: The JobSystem, where multiple threads are working together to execute five active jobs of the JobSystem simultaneously.

The JobSystem generally aligns with other simulation software when working in a single Sub-Model.

When scheduling jobs generated within a single Sub-Model, the JobSystem follows the order defined by the user and the formalism of that Sub-Model. The basic properties of Sub-Model logic are sequential, meaning that the order of jobs in time is absolute.

However, as soon as jobs must be scheduled that involve multiple Sub-Models, this becomes more difficult. We must solve the problem of data that is needed by multiple threads that aren't necessarily at the same point in (simulation) time because we must be certain that all previously scheduled jobs must be completed and cannot change the data and structures. There are multiple ways of handling this problem.

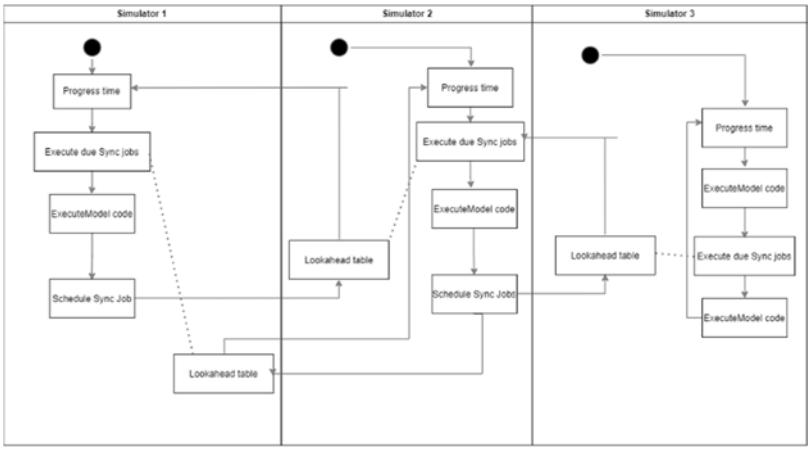

Figure 3: sync-events in the relation between Simulators.

The first possible solution to synchronization would be optimistic synchronization (Jafer, 2010), which allows scheduling jobs that need Sub-Models that are not necessarily aligned in time and saving the Sub-Models before we execute the code. If it then turns out this was not possible, we restore the Models to their pre-calculation state. However, because the size of the Models in ERS can be enormous, making frequent backups is very memory expensive.

Instead, we use conservative synchronization, where we only schedule the job when we know that both Sub-Models are aligned. Of course, this means we may have idle processor time since we might have to wait on the slowest Model to realign in time. However, it can be shown that conservative synchronization does outperform serial computation (Nicol, 1993) even in the worst case. There are also examples of conservative synchronization outperforming optimistic synchronization in every metric (Jafer, 2010). One of the reasons for the good performance of conservative synchronization is that we can use the resources, not used for the calculation of possible future states, to do background tasks. In light of that evidence and the belief that we have found a way to minimize the waste of computational resources, we have chosen conservative synchronization.

To solve the issue of processor time being wasted, we use a Lookahead-Table in combination with the JobSystem that determines the order of synchronization and at what time the synchronization job takes place for the Sub-Models in their local time.

The Lookahead-Table is a record of when each Simulator needs to be synced with each other Simulator. Each Simulator has its own Lookahead-Table. The Simulator generates the sync-events by processing each action in a Sub-Model (Figure 3). The Lookahead-Table lets the Simulators run independently until they have to sync. The user has to define a table that identifies these moments. The sync can be run during the simulation by copying the sync data to prevent modifications during the continued execution of the Model (Figure 4).

Allowing application developers to determine the sync-event time schedule makes it possible to have order-of-events-violating Models without compromising causality. The events will always obey relative causality if a sync-event is scheduled between events. The Lookahead-Table enables the application developer to determine how strict causality is enforced without leading to issues in the results.

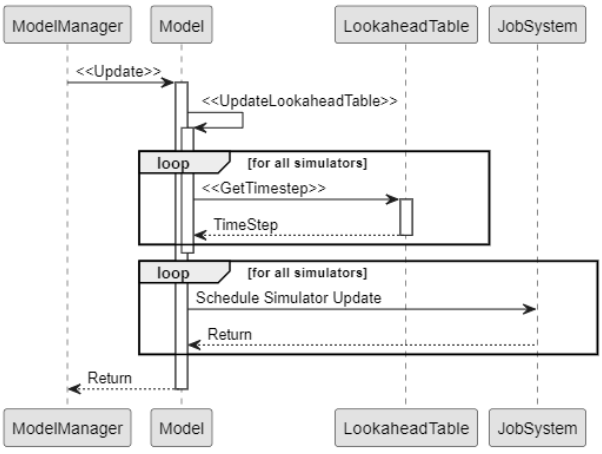

Figure 4: Update function of a Model calculating a new destination time for each Simulator, and then starting a job to update in parallel if it wasn't already.

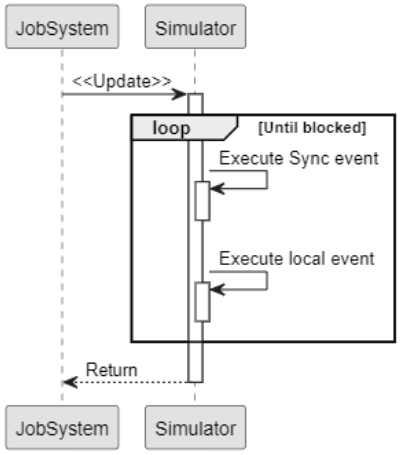

Figure 5: Update function of the Simulator, which keeps executing events until causal rules block it.

The Lookahead-Table is different between simulation runs with the same parameters, but this doesn't impact the simulation results. With the efficient splitting of the Model into Sub-Models, there should not be a constant stream of data going from one Sub-Model to another since this will make overhead much larger and reduce the benefits of multi-threading.

Another key observation is that the Lookahead-Table cannot predict if a conditional data exchange is necessary (unless explicitly given this possibility). So all possible data exchanges need to be included in the Lookahead-Table and can cause threads to wait on each other. However, ERS allows advanced users to interact with the Lookahead-Table directly by scheduling sync-events and modifying promises made to the Lookahead-Table by the Sub-Model's content.

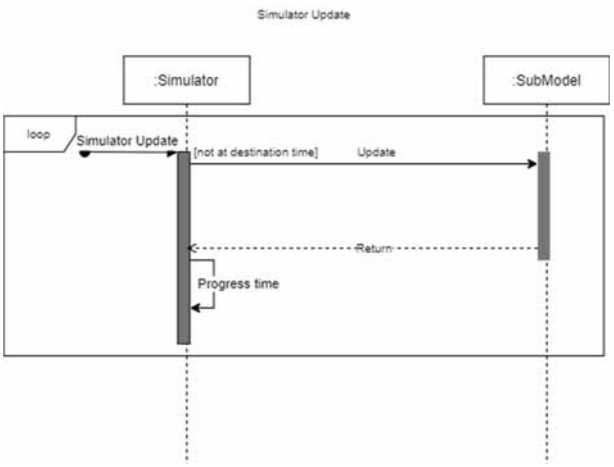

Figure 6: Update loop of a Simulator in relation with a Sub-Model.

1.3 Event Mechanism

In ERS, we found that some terminology did not suit our needs perfectly. Because of this, we will define additional terminology below.

Model-orchestrator is the time mechanism influenced by the Lookahead-Tables to orchestrate sync-events between Simulators.

Our new mechanism uses a pessimistic local hybrid-DE-orchestrator as in (Gomes, 2018) sub-section 5.1 without rollbacks on each Sub-Model. The Model uses conservative synchronization using sync-events that run in parallel time-flows and conditionally converge, which are orchestrated by a Model-orchestrator that can be distributed. The pessimistic nature of the orchestrator is more complex than traditional orchestrators. In essence, some events are executed in an optimistic fashion relative to each other, but they cannot interfere with the states relevant to each other. The "worst case" behavior is purely pessimistic but can be avoided in almost all situations.

events	Sub-Model a	Sub-Model b
Local	X_a	X_b
Non-local	Y_a	Y_b

We define the relative temporal restrictions between events in two ways. First, we categorize them as either pessimistic or type 1 optimistic based on whether an earlier event can be executed after a later event. If an earlier event can be executed later, the restriction is type 1 optimistic. The restriction is pessimistic if the later event waits until the earlier event is finished. If two events happen at the same time, there are slightly different concerns. When two events happen at the same time, the consideration is whether one event can start without the other and if the events can progress independently.

For this classification, we consider sync-events as two events, one in each Sub-Model. In this setting, the relation is type 2 optimistic when one event can start without having to wait until the other event starts and runs independently, and the restriction is conservative if both Sub-Models have to wait on each other and run together. We follow a somewhat different naming convention as syncing is more typically categorized as conservative or (type 2) optimistic. In ERS, this relation is unique to sync-events, as those are the only events that can be required to be executed simultaneously.

It is important to note that the conditions are not equivalent. Pessimistic restrictions imply that a previous event must be finished, while conservative restrictions imply that the two events have to start at the same time and run together.

These restrictions are important to allow whether we can calculate the events separately, increasing the efficiency of the calculations.

Local Sub-Model Events: are events entirely inside a Sub-Model.

Model-Global-Events: are events involving multiple Sub-Models.

Pessimistic relationship: requires that events are executed in order. Otherwise, causality is broken. We do not have to specify whether it is also Conservative because two events of this type cannot be executed at the same time.

Optimistic relationship: does not require that the event is executed in the same order because events do not impact one another in any way. This relation is both type 1 and 2 optimistic since these events are fully independent of each other.

Pessimistic & conservative relationship: does require events to be executed in order and can have a shared state.

In the following table, we categorize each type of event pair as either pessimistic or optimistic. For the pairs (Y_a, Y_b) and (Y_b, Y_a) we also included whether they are conservative or optimistic. Note that In ERS, simultaneity cannot occur for two events in the same Sub-Model, so we have not categorized those events. Between Sub-Models, simultaneity can occur, but not coordinated by the Model directly, save for sync-events. So while we categorize these pairs of events as optimistic, they never interact, so there is no danger in doing so.

In the table, we show the relative temporal relations between different events inside a Model (as defined above).

Relative temporal restrictions	X_a	X_b	Y_a	Y_b
X_a	Pessimistic	Optimistic	Pessimistic	Optimistic
X_b	Optimistic	Pessimistic	Optimistic	Pessimistic
Y_a	Pessimistic	Optimistic	Pessimistic	Pessimistic & Conservative
Y_b	Optimistic	Pessimistic	Pessimistic & Conservative	Pessimistic

2 Multi-formalism

In an ERS-based application, an application-builder is no longer restricted by the formalism dictated by the software he uses; instead, the application-builder can choose the optimal formalism for the problem.

The freedom of choice for different formalisms is achieved due to the new architecture of ERS, where the engine can handle simulations running in multiple time signatures simultaneously. This allows the user to Model in several formalisms and allows different Sub-Models to have different formalisms. These Sub-Models can interact and form a larger whole, allowing even the most complex systems to be simulated.

The strength of this way of modeling is how previously separate disciplines can be united in one Model. This allows multiple teams to work in one integrated Model. This can best be demonstrated in example case 1.

3 Example Case

Consider the case of a large international airport. The managers of this airport want to know if their airport can handle an expected increase in customers and flights. The increased number of passengers can give issues to two separate systems: terminal operations and baggage handling. Of course, these could affect each other, but only at specific points.

If the passengers are delayed checking in their baggage, they will arrive later at the security lanes. Conversely, if the security lanes become too long, this might delay flights, which will change the timing of the baggage system.

Crowd dynamics, as is needed for the security lanes, would typically be done in an agent-based simulation, while the baggage system will typically be modeled in a discrete-event based simulation.

Currently, these simulations would be made in different applications that best handle their specific situations. Resulting in two Models that would be run statically in relation to each other. In ERS, these two Models could be one Model which could correctly identify the effect the two Sub-Models would have on each other.

This would not require reading in arrival lists, and the Models can also dynamically be linked with each other and, depending on the number of syncs needed, would run in less than double the time of running the slowest algorithm. However, because it only must run once and not iteratively, it will result in much faster runs. In addition, it will result in more reliable results because the entire Model can be created in a single application running on the ERS platform, and it will allow faster debugging and working in the Model because the connections between the various parts are more intuitive.

4 Splitting Models Efficiently

One of the core concerns of any program is that it runs fast enough for its given purpose. For simulation software, this means that the program needs to scale well and be able to run within a reasonable time frame, given the proper computational resources. In ERS, the application-builder can significantly increase the application's efficiency since the application-builder can divide mostly separate processes into separate Sub-Models. These Sub-Models then can do most of their computations separately because of the JobSystem and the Lookahead-Table. The separation of these Sub-Models can speed up computations if the Sub-Models don't need to communicate too often, resulting in less overhead. The independence of these Sub-Models also allows the user to define Sub-Models so that the calculations can be distributed over multiple threads.

5 Additional Challenges

In literature, specifically in (Gomes, 2018), and (Taylor, 2019) some challenges that have not yet been explicitly discussed are identified. This part of the paper will discuss these challenges and their application to ERS.

Latency: It is recognized that latency is challenging for synchronizing multiple computers to work on 1 task. However, in most places where ERS will be used, e.g., data centers or local networks, latency will naturally be minimized due to the scaling when splitting Sub-Models. This latency problem grows smaller with the number of Sub-Models. The Model can be split into other Sub-Models. We do not need to replicate the entire Model consensus across all computers, only Lookahead-Tables for Sub-Models that influence the Sub-Model running on a computer.

Modular Composition—Algebraic Constraints: the authors identify the need for some (continuous) Models to enforce algebraic constraints at all times on several Sub-Models, making them depend on each other. This dependency can cause (near infinite) feedback loops. This is unavoidable because ERS is a platform, so we do not restrict the relations that can be defined between Sub-Models. However, the worst case does not happen as an infinite loop is not possible in ERS in that way, so it will, at some point, resolve. In general, these kinds of errors cannot be prevented by a simulation platform because it is caused by inter-Sub-Model relations, which we cannot regulate if we want to give application builders sufficient freedom. In general, this is a concern, but this is not applicable to the ERS engine.

Algebraic Loops: algebraic loops are loops created by the indirect dependence of variables on themselves. They are very similar to *Modular Composition—Algebraic Constraints,* and we accept them as possible problems because we do not limit the ability of application builders to make connections between Sub-Models.

Consistent Initialization of Simulator state: in some Simulators, the input data has to obey certain conditions to be valid. This can be seen as a sub-problem of the problem with Modular composition-algebraic constraints, in the sense that this constraint is only necessary at the start of the simulation. The argument is the same for the overall problem. At the same time, it is a problem; it is not a problem that a simulation platform can solve and instead should be handled by the application developer or the model builder.

Compositional Convergence—Error Control: in many simulation Models, there is the desire to estimate the errors related to the underlying theoretical solution. In ERS, we do not calculate this error since this is too specific to be built into a platform. Instead, this can be best handled by an application builder.

Compositional Stability: Similar to the last point, many simulations might also want to estimate the stability of the error compared to the theoretical solution. However, this problem is too specific to be handled at a platform level and should instead be handled on an application or user level.

***Compositional continuity*:** for continuous Simulators that are connected to non-continuous Simulators, it can be an issue to retain the continuity in the connection. In ERS, we allow almost arbitrarily small-time steps (up to a single Planck time).

This combats this issue as far as possible on a platform level. Special measures can be taken on an application or Model level, but a platform should not enforce these.

Real-Time Constraints, Noise, and Delay: For continuous time simulation, whether it is completely internal or part emulation, it is important to be able to support the right frequency of updating (micro-step). ERS takes three measures to support the right frequency.

First of all, as mentioned earlier, the platform does not enforce a step size limit that can be physically restrictive as time steps can be as small as a single Planck time. Secondly, ERS allows the total Model to be split into many Sub-Models so that an application can take advantage of simultaneous calculations as much as possible. Lastly, ERS can support many different types of simulations simultaneously, removing the need to model in several applications and thus eliminating the issue of bad communication, as long as no emulation is included.

However, even with these measures, implementing the right frequency is not always possible, and dealing with this remaining issue will have to be handled on an application-to-application basis.

Discontinuity Identification: In communication with continuous simulations, it is beneficial to identify discontinuities. However, the core cause of the discontinuity lies in the continuous Sub-Model or the communication between Sub-Models.

In either case, the application developer is responsible, so it should be solved on an application or model level and not on a platform level.

Discontinuity Handling: Once a discontinuity is identified, it would be beneficial to handle that discontinuity in a particular way so that the simulation can continue with reasonable accuracy.

However, because the cause of the discontinuity is in the Model, it is best to handle this on an application level and not on a platform level, because different types of discontinuities might be handled differently.

Algebraic Loops, Legitimacy, and Zeno Behavior: as we discussed earlier with algebraic loops, a more general question can be asked about whether we should detect potentially infinite event chains that either keep the simulation at the same time or represent an increasingly minor progression of time in such a way that the simulation never reaches the designated endpoint. In general, we cannot detect this behavior in ERS since it is Sub-Model specific and cannot be identified with certainty without knowledge of the internal working of the Sub-Model.

However, In ERS, loops cannot extend indefinitely since the engine will not allow sub-Planck time increments (that by definition will not be physically consequential) nor infinite scheduling on a single point in time. This means that this behavior will always end in ERS – but this will take a very long time to materialize as ERS is designed for microscopic time-scale simulation.

Stability Under X: a concern for co-simulation (Gomes, 2018), in general, is that the entire Model might not be stable, even if all the Sub-Models are stable. This is specific to a Simulator and should be handled by the application or on the model level.

Some issues might cause instability, such as noise caused by the communication between continuous time and discrete time Models. However, the instability of the whole system is still inherently caused by the design of the Sub-Models, and so this issue should be checked and corrected for by applications for which this behavior could occur.

Theory of discrete event Approximated States: With multi-formalistic co-simulation becoming commercially viable, there is a need to develop a theoretical framework for the quality of communication between continuous-time and discrete-time simulation Models. We acknowledge this, but this goes beyond the scope of this paper. InControl will engage with the science community to start the development of such a theoretical framework.

Standards for Hybrid Co-Simulation: Besides theory, a new standard for co-simulation should be established so that new Models can be developed on a solid foundation. This is, in a sense, what ERS as a platform does since it offers a way of building a wide variety of co-simulation scenarios on one simulation platform.

This means that all applications build to solve these scenarios can build on the solid foundation that ERS has established.

Semantic Adaptation and Model Composition: A central question in co-simulation is what information needs to be included in the wrapper of a simulation. In our case, this would be the Sub-Model or the Simulator. In (Gomes, 2018) it is argued that this should be specific to the Model, in contrast to how this is handled in ERS. This paradox is solved by considering that the paper includes data transfer as part of the wrapper, while in ERS, this is part of the sync event, which an application builder can alter.

Thus ERS can have a single implementation of the Sub-Model and Simulator concepts without running into problems identified in (Gomes, 2018) .

Predictive Step Sizes and Event Location: If the core time concept of a simulation engine is based on discrete-time (like in ERS), there is a question of how large the time steps should be. In ERS, application builders primarily regulate this since they can schedule their own events. Meaning that the application builder can decide the precision required.

While the precision might be sufficient, this approach might still have efficiency concerns. For example, if very high precision is used, this high precision can lead to a large number of sync-events that do not transfer data. These sync-events are unavoidable in ERS on a platform level because we use conservative synchronization, so we do not use the Lookahead-Table to resolve these sync-events. This efficiency problem is of limited severity because sync-events that transfer no information only use a tiny amount of computation resources.

In addition, the application builder can mostly prevent unnecessary sync-events, so efficiency can be high with the right implementation. Also, each Simulator allows a specific step size to incorporate high precision in sub-subsections of the entire simulation Model.

6 Conclusion

ERS is a new simulation platform for application builders who want more freedom, power, and possibilities than other simulation packages can offer. It allows for applications that model reality closely, even if reality is complex and does not follow the constraints that any specific formalism requires.

ERS works by splitting a complete Model into Sub-Models whereby each Sub-Model uses its formalism and can communicate and exchange data with other Sub-Models through sync-events. The Model causality is maintained by Lookahead-Tables, which create and maintain a time consensus that determines the latest point to which a Sub-Model can independently run. Sub-Models run concurrently or even remotely, allowing ERS to scale well.

ERS allows application-builders to reach their full potential and connect all relevant systems with reasonable run times in one Model. It does this by allowing the modelers and application builders to build a platform that can support massive models and use third-party libraries in the languages they know best.

ERS can support the needs of the application the modeler wants.

References

[1] Gomes C. T. Co-simulation: a survey. ACM Computing Surveys (CSUR). 2018 May; 51(3): 1-33.

[2] Jafer S, Wainer S. A. Conservative vs. Optimistic Parallel Simulation of DEVS and Cell-DEVS: A Comparative Study. Proceedings of the 2010 Summer Computer Simulation Conference. 2010 July; (pp. 342-349.).

[3] Nicol, D. M. The cost of conservative synchronization in parallel discrete event simulations. 1993 April; J. ACM, 2(40), 304–333. doi:https://doi.org/10.1145/151261.151266

[4] Taylor, S. J. Distributed simulation: State-of-the-art and potential for operational research. European Journal of Operational Research, 2019 October; 237(1), 1-19.

Discrete Event-based Modeling of Conveyors for Dry Bulk Material

Peter Junglas[1*], Lukas Schmedes[2]

[1] PHWT-Institut, PHWT Vechta/Diepholz, Thüringer Straße 3, 49356 Diepholz, Germany;
[*] *peter@peter-junglas.de*

[2] GRIMME Landmaschinenfabrik GmbH & Co. KG, Hunteburger Straße 32, 49401 Damme, Germany

SNE 33(3), 2023, 133-140, DOI: 10.11128/sne.33.tn.10655
Selected ASIM SST 2022 Postconf. Publication: 2023-08-01;
Received Rev. Improved: 2023-09-21; Accepted: 2023-09-25
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. The simulation of transport processes, though inherently continuous, is often done in a discrete-event simulation environment. In the case of conveyors for dry bulk material, this can lead to modeling difficulties, especially regarding the coupling of two conveyors with different velocities. We will present a modeling approach solving such problems, describe an implementation in SimEvents and present results of systematic tests.

Introduction

In the simulation of production and logistic processes, the modeling of materials handling is of paramount importance. Though the detailed description of a transport process uses continuous functions of time, such as positions or mass flows, in the context of a complex simulation it is often simplified and modeled using only discrete events. But this reduction of complexity can lead to problems, because:

> It is often important to model such entity transfer accurately since studies have shown that delays and inefficiencies in operations might be caused more by the need just to move things around rather than in actually doing the work. [1, p.345]

A simple conveyor belt moving discrete unit loads with constant velocity generally can be modeled easily enough. But building an adequate model for the transport of dry bulk material with a wide range of granularity and changing velocities of one belt or between belts is much more difficult.

Modifying the belt velocity is a standard method to adapt to a varying input mass flow. This can be used to utilise the full capacity of the conveyor at a lower speed, which will often decrease the power consumption [2], or to speed up in order to shorten the transportation time. On the other hand, when transporting damageable goods like apples or potatoes, one could try to slow down the conveyor to guarantee a high quality of the goods.

In the following we will describe a discrete-event model of a conveyor for dry bulk material, which has a control input to change the velocity. A special focus will be on the coupling of conveyors running with different velocities, since this leads to modeling problems in a discrete environment. Finally the model will be implemented in SimEvents from Mathworks [3] and tested systematically.

The acceleration or deceleration of a highly loaded conveyor creates considerable tension in the belt, leading to local stretching or even breaking of the belt [4]. In this study we will neglect this effect and treat the belt as a rigid body with the same velocity everywhere.

1 Modeling of a Single Conveyor

A discrete-event model of a conveyor has to implement the delays of the incoming entities given by the conveyor length l and the velocity v. In addition it has to store the positions of all entities at the current (discrete) time in order to cope with entities of varying size l_E or with a time-dependent velocity (cf. Figure 1).

Such a component exists in many commercial discrete simulation environments such as Arena [1], SimEvents [3] or PlantSimulation [5]. The length of the entities can either be defined as a fixed parameter or as an entity-specific attribute. The various programs have different additional features like a minimal distance between entities, accumulation of entities in case

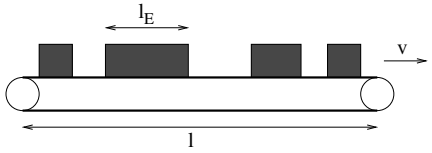

Figure 1: Simple conveyor model.

of blocking or incorporating acceleration phases at the beginning – all of which will not be considered in the following. Another difference is the exact procedure, how an entity enters (or leaves) the conveyor. For concreteness we will define that a (right-going) entity starts at position $x = 0$ with its left edge coinciding with the left edge of the conveyor, and leaves at position $x = l$, when its left edge coincides with the right edge of the conveyor.

For the transportation of dry bulk material the situation is a bit more complicated, because there are no easily identifiable entities and the transport process is inherently continuous: Due to a usually non-uniform production process and the inhomogeneity of the material the conveyor is filled with an incoming mass flow $\dot{m}_{in}(t)$, which leads to a line load $\lambda := \frac{\partial m}{\partial x}$ given by

$$\lambda(t,x) = \frac{1}{v}\dot{m}_{in}(t - \frac{x}{v}) \tag{1}$$

for a constant velocity v (cf. Figure 2).

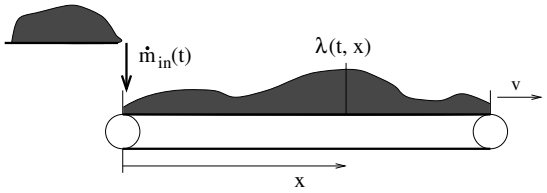

Figure 2: Conveyor model for bulk dry material.

The granularity of dry bulk material varies widely in practical applications, ranging from almost microscopic particles (powder) over small or medium sized particles (rice, apples) to large lumps (ore). The modeling methods used vary accordingly. Two extreme methods are described and compared in [6]: the microscopic description, where the movement of each particle is described separately, and a macroscopic representation, using a mass density and differential equations describing the conservation laws.

Both methods are computationally intensive, therefore in [7] a "mesoscopic" approach has been utilised, which is well suited for medium sized granularity and can be easily incorporated into standard discrete modeling environments. Basically, the continuous line load is replaced by discrete entities E_i with mass m_i given by

$$m_i = \int_{(i-1)\Delta t}^{i\Delta t} \dot{m}_{in}(t)\,dt, \quad i \in \mathbb{N}^+$$

for an arbitrary fixed time interval Δt. Using (1) one gets for constant velocity v:

$$\begin{aligned} m_i &= v\int_{(i-1)\Delta t}^{i\Delta t} \lambda(i\Delta t, v(i\Delta t - t))\,dt \\ &= \int_0^{v\Delta t} \lambda(i\Delta t, x)\,dx \end{aligned}$$

This shows that the conveyor is divided into compartments of equal length $l_E = v\Delta t$. For simplicity, one often chooses $l_E = l/N$ for $N \in \mathbb{N}^+$, so that entity E_i enters the conveyor at time $i\Delta t$, i. e. when its compartment is filled, and leaves at $(N+i)\Delta t$.

An alternative approach could be to use compartments of equal mass instead of equal length. This would lead to a more complex timing of events, which makes the interpretation of results more involved. Furthermore, some simulation environments (e. g. Arena) use conveyor components with a fixed cell size, which would make the implementation of this approach quite ugly.

2 Coupling of Conveyors with Different Velocities

We will now analyse how one can combine two single conveyors with lengths l_1, l_2 and velocities v_1, v_2. A mathematical description of the continuous process with additional input and output reservoirs at the end of each conveyor has been given in [8]. Considering only the case of constant (but different) velocities, one has:

$$\begin{aligned} \lambda_1(t,x) &= \frac{1}{v_1}\dot{m}_{in,1}(t - \frac{x}{v_1}) \\ \lambda_2(t,x) &= \frac{1}{v_2}\dot{m}_{in,2}(t - \frac{x}{v_2}) \end{aligned}$$

Connecting the conveyors directly, the output of conveyor 1 is the input of conveyor 2, therefore:

$$\begin{aligned} & \dot{m}_{in,2}(t) &=& \; v_1 \lambda_1(t, l_1) \\ \Rightarrow \qquad & \lambda_2(t,x) &=& \; \frac{v_1}{v_2} \lambda_1(t - \frac{x}{v_2}, l_1) \end{aligned}$$

For a continuous model one simply changes the line load λ_2 by the factor v_1/v_2. This happens automatically in real life, when the bulk material gets thinned out or condensed on the second conveyor – as long as the capacity of the second belt is not exceeded.

But our discrete model runs into problems with the timing of the entities: Conveyor 1 sends entities at time intervals $\Delta t_1 = l_{E,1}/v_1$ to the entrance of conveyor 2, which in turn delivers entities at its output at generally different time intervals $\Delta t_2 = l_{E,2}/v_2$. Therefore one cannot maintain the idea of an entity defined as a fixed set of particles with given mass. Instead one identifies an entity with the content of a given compartment on a conveyor. Such a compartment is created and filled at the entrance of a conveyor, and emptied and destroyed at its exit.

The remaining task is now to compute the mass of such a newly created entity. According to the ratio

$$k := \frac{\Delta t_2}{\Delta t_1} = \frac{l_{E,2}}{l_{E,1}} \cdot \frac{v_1}{v_2}$$

one needs different strategies, how to cope with this problem. Such strategies should fulfil two requirements:

- mass conservation, i. e. incoming and outgoing masses should balance on a short time scale,
- homogeneity, i. e. the output mass distribution should closely follow the input mass values. For a constant incoming distribution this means, that the outgoing values shouldn't vary much.

If k is integer, one simply adds up the masses of k incoming entities to create an outgoing one, whereas if $1/k$ is integer, one distributes the mass of one incoming entity among $1/k$ outgoing entities. In all other cases one has to account for the unbalanced timing of input and output entities. Since the problem appears only at the connection of the two conveyors, we can concentrate on the second conveyor with its incoming values $m_{in,i}$ at times $i\Delta t_1$ and the corresponding outgoing values $m_{out,j}$ at times $j\Delta t_2$, disregarding the delay time of the second conveyor.

If $k > 1$ one can apply a simple collection strategy using a virtual bin, which accumulates all incoming masses into m_{acc}. A new output entity then empties the bin and gets the total accumulated mass. Table 1 shows how this works in an example with equal entity lengths $l_{E,1} = l_{E,2} = 1\,\mathrm{m}$, velocities $v_1 = 2.5\,\mathrm{m/s}$, $v_2 = 1\,\mathrm{m/s}$ and constant incoming masses $m_i = 1\,\mathrm{kg}$.

i	j	t	m_{in}	m_{acc}	m_{out}
1	-	0.4	1	1	-
2	-	0.8	1	2	-
-	1	1.0	-	0	2
3	-	1.2	1	1	-
4	-	1.6	1	2	-
5	2	2.0	1	0	3
6	-	2.4	1	1	-

Table 1: Times and masses for example 1 ($k = 2.5$).

For $k < 1$ one has to use a partition strategy instead. The following strategy "A" defines the mass of a partition as

$$m_p = k\, m_{in}$$

each time a new entity enters, and sets outgoing entities accordingly. This simple scheme leads to a problem due to the timing, as can be seen in Table 2, which uses $v_1 = 1\,\mathrm{m/s}$, $v_2 = 1/0.35\,\mathrm{m/s}$: At $t = 1.75$ there is not enough mass available for the outgoing entity E_5. But this can be cured easily by setting

$$m_{out} = \min(m_p, m_{acc})$$

Unfortunately strategy A has a serious drawback, as Table 3 shows using $v_1 = 1\,\mathrm{m/s}$, $v_2 = 1/0.8\,\mathrm{m/s}$: Though the mean ratio of input entities to output entities is 0.8, for a while the actual ratio is 1. Therefore the accumulated mass m_{acc}, which is just the difference between total input and total output mass, grows.

i	j	t	m_{in}	m_p	m_{acc}	m_{out}
1	-	1.0	1	0.35	1	-
-	3	1.05	-		0.65	0.35
-	4	1.4	-		0.3	0.35
-	5	1.75	-		0	0.30
2	-	2.0	1	0.35	1	-

Table 2: Times and masses for example 2 ($k = 0.35$).

This is in conflict with the primary goal of mass conservation on a short time scale. Even worse: When the following input entities are empty (i. e. $m_{in} = 0$), m_p is set to 0 and the accumulated mass stays in the internal bin.

i	j	t	m_{in}	m_p	m_{acc}	m_{out}
1	-	1.0	1	0.8	1	-
-	2	1.6	-		0.2	0.8
2	-	2.0	1	0.8	1.2	-
-	3	2.4	-		0.4	0.8
3	-	3.0	1	0.8	1.4	-
-	4	3.2	-		0.6	0.8

Table 3: Times and masses for example 3 ($k = 0.8$), strategy A.

Strategy "B" tries to solve this problem by changing the value of the partition mass to

$$m_p = k\, m_{acc},$$

where the value is only computed, when an input entity arrives. This will distribute the surplus value of m_{acc} among the next outgoing entities, thereby reducing the total mass imbalance, as can be seen in Table 4 for the values of example 3.

i	j	t	m_{in}	m_p	m_{acc}	m_{out}
1	-	1.0	1	0.800	1.000	-
-	2	1.6	-		0.200	0.800
2	-	2.0	1	0.960	1.200	-
-	3	2.4	-		0.240	0.960
3	-	3.0	1	0.992	1.240	-
-	4	3.2	-		0.248	0.992

Table 4: Times and masses for example 3 ($k = 0.8$), strategy B.

We will finally provide a mathematical description of the distribution strategy B to clarify possible open points and to guide the implementation. Starting point are the two positive time intervals Δt_1, $\Delta t_2 = k\,\Delta t_1$ between arrival or departure of entities at the virtual connecting bin and the positive end time t_{end} of the simulation. We now define the sets

$$\begin{aligned} T_{in} &= \{i\,\Delta t_1 \,|\, i \in \mathbb{N}^+\} \cap [0, t_{end}] \\ T_{out} &= \{j\,\Delta t_2 \,|\, j \in \mathbb{N}^+\} \cap [0, t_{end}] \end{aligned}$$

The function $m_{in}(t)$ is given for $t \in T_{in}$ (by a production process) and constant elsewhere.

The functions m_{acc}, m_p and m_{out} will be defined on $T_{in} \cup T_{out}$, they are constant elsewhere. For simplicity we denote

$$f(t^-) := f(t-\varepsilon) \quad (\varepsilon > 0 \text{ sufficiently small}),$$

where "sufficiently small" means "smaller than the size of any open time interval from $T_{in} \cup T_{out}$". We now start with

$$m_{acc}(0) = 0\,kg$$

and define:

$$m_p(t) = \begin{cases} k\,(m_{acc}(t^-) + m_{in}(t)) & | t \in T_{in} \\ \text{const.} & | \text{otherwise} \end{cases}$$

$$m_{out}(t) = \begin{cases} \min(m_p(t), m_{acc}(t^-)) & | t \in T_{out} \setminus T_{in} \\ \min(m_p(t), m_{acc}(t^-) + m_{in}(t)) & | t \in T_{in} \cap T_{out} \\ \text{const.} & | \text{otherwise} \end{cases}$$

$$m_{acc}(t) = \begin{cases} m_{acc}(t^-) + m_{in}(t) & | t \in T_{in} \setminus T_{out} \\ m_{acc}(t^-) - m_{out}(t) & | t \in T_{out} \setminus T_{in} \\ m_{acc}(t^-) + m_{in}(t) - m_{out}(t) & | t \in T_{in} \cap T_{out} \\ \text{const.} & | \text{otherwise} \end{cases}$$

One easily checks that these definitions reproduce the collection strategy and partition strategy B. Strategy A is a bit simpler and can be easily defined in a similar way.

3 Implementation in SimEvents

SimEvents [9] is a blockset for the Simulink environment from Mathworks [10] that enables discrete event modeling. It uses the transaction-based approach, which describes entities that are handled by fixed components. It contains the usual components like an entity generator, a server, a queue and several routing blocks. As stated above, a basic `Conveyor System` component is available that transports discrete entities of given length. Many components include so-called "action"-functions, which are called at the entry or exit of an entity, and can be defined using Simulink function blocks.

The conveyor for dry bulk material (cf. Figure 3) is defined as a component with an input and output port for the entities, inputs for the incoming and outgoing

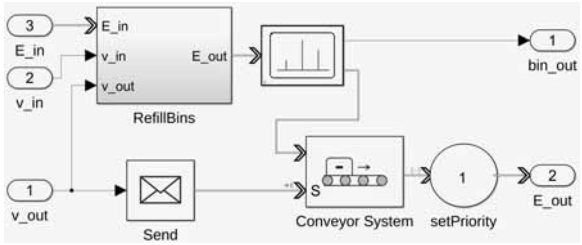

Figure 3: Conveyor model.

velocities and an additional output to display the entities leaving the internal bin. The length l_E of the compartments and the total length l are provided as parameters. Incoming and outgoing entities have attributes describing their length and their mass. The block uses the predefined `C onveyor System` and a component `RefillBins` that handles the adaptation of the different velocities.

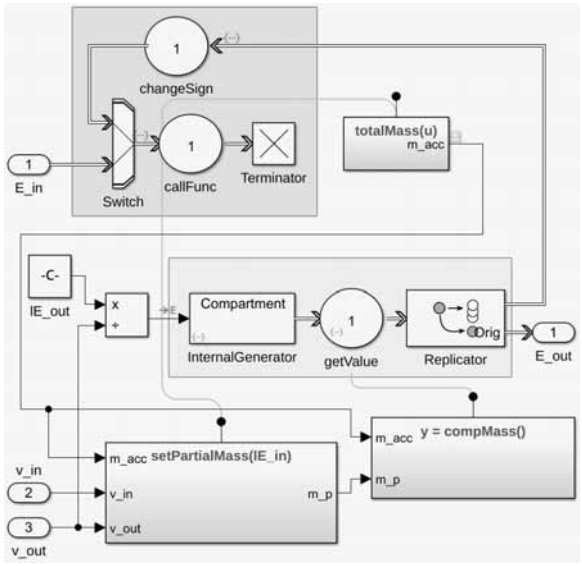

Figure 4: Implementation of the `RefillBins` component.

The central block `RefillBins` (cf. Figure 4) implements the formulae described in section 2 that define strategy B. Incoming entities are routed through a server, which calls the Simulink functions `totalMass` to compute m_{acc} and `setPartialMass` to compute m_p, and destroyed afterwards. An internal generator creates new entities at times in T_{out} and sends them to a server that sets the mass attribute to m_{out}, which is computed with the function `compMass`. Before an entity leaves the block, a copy is created and sent back to the input server, so that its mass can be subtracted from m_{acc}. The alternative strategy A can be implemented easily in an analogous way.

As usual for transaction-based modeling, one has to make sure that concurrent events are handled in the correct order to make things work. If $t \in T_{in} \cap T_{out}$, this means that the incoming entity has to be processed before the internally created one to guarantee the correct computation of m_p and m_{out}. For this purpose entities enter the conveyor with a high priority (low value), while the internal generator creates entities with low priority, which is raised, when an entity leaves the conveyor.

A more subtle timing problem has lead to the inclusion of the server `getValue` behind the internal generator: In principle the call of the function `compMass` could have been done immediately inside the generator. But then the order of the mass computation and the processing of a concurrent incoming entity are not defined! The priority only affects events and messages, not internal function calls.

4 Test Results

To compare the performance of the strategies A and B in detail, a set of tests have been carried out that concentrate on two key figures: the mean value over time $\overline{m_{acc}}$ of the internally accumulated mass, which shows the short-time mass conservation, and the standard deviation σ_{out} of the output mass, which measures the homogeneity of the outgoing mass distribution.

All tests use constant entity lengths $l_{E,1} = l_{E,2} = 1\,\mathrm{m}$ and outgoing velocity $v_2 = 1\,\mathrm{m/s}$. The input velocity is given as $v_1 = k\,v_2$, where different values of k and different input mass distributions will be analysed. All results are compiled in Table 5 and are referenced by their number in the following.

The first group $(1-9)$ consists of tests with constant input mass $m_i = 1\,\mathrm{kg}$ and varying k. For k or $1/k$ integer, optimal procedures have been given in section 2, which lead to a constant output mass, i. e. $\sigma = 0$. For these cases the average value of m_{acc} can easily be computed to be

$$\overline{m_{acc}} = \frac{n-1}{2n} \quad | \, n \equiv 1/k \in \mathbb{N}^+$$

$$\overline{m_{acc}} = \frac{n-1}{2} \quad | \, n \equiv k \in \mathbb{N}^+$$

Test results 1 – 4 show that both implementations reproduce these values, minor differences are due to a short initial period.

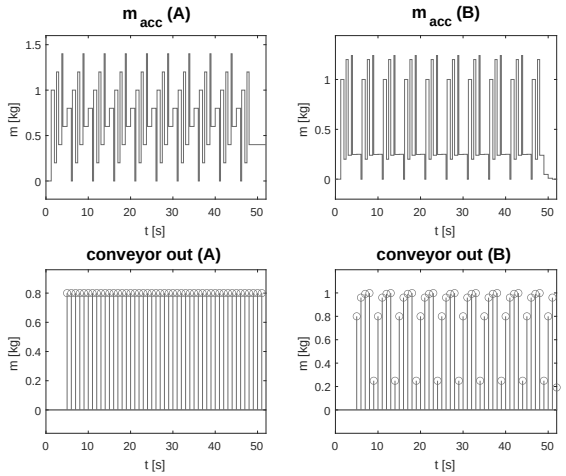

Figure 5: Test result 6, constant input, k = 0.8.

Results for other values of k are shown in tests 5 – 9, among them the examples from section 2. The plots in Figure 5 display the function $m_{acc}(t)$ and the conveyor output over time for both strategies, they reproduce the results for $k = 0.8$ from Tables 3 and 4. The figures from Table 5, no. 6, show that the mass balance of strategy B is better by a factor of 1.5 than that of strategy A.

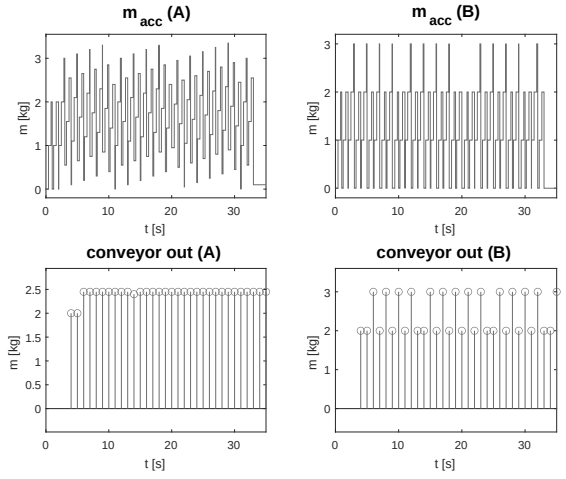

Figure 6: Test result 8, constant input, k = 2.45.

Very instructive is the comparison of tests 8 and 9: Changing k from 2.5 to 2.45 leads to a much worse local mass balance, especially for strategy A.

The reason for this behaviour can be seen in Figure 6: The accumulated mass rises slowly over fast cycles of 2 s, but is reset with a longer period of 20 s. A look at Table 1 shows that for $k = 2.5$ a much shorter period of 5 s appears, so that m_{acc} can't grow as much. The length of the period is given by the representation $k = p/q$ with coprime natural numbers p, q:

$$k = \frac{p}{q} = \frac{\Delta t_2}{\Delta t_1} \quad \Rightarrow \quad q\Delta t_2 = p\Delta t_1,$$

where $\Delta t_2 = 1\,\mathrm{s}$ in all our tests. A rational k with a large denominator therefore leads to a long period, which can possibly produce a long time accumulation and a bad mass balance. The problem is less severe for strategy B, since it gets rid of short time accumulations as fast as possible.

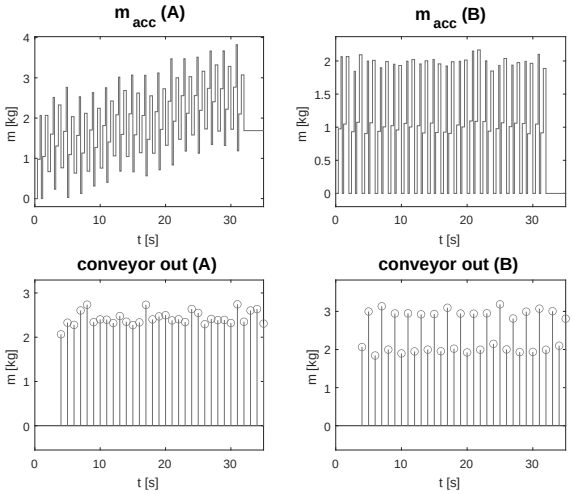

Figure 7: Test result 14, uniform input, k = 2.5.

In real applications the input flow is usually not constant, at least it fluctuates due to the granularity of the material. To model this, the next tests (10 – 14) take mass input values $m_i(t)$ using a uniform distribution on the interval [0.9, 1.1] kg, which has a standard deviation of $\sigma_{in} = 0.058$ kg.

The corresponding results in Table 5 are generally similar to the previous ones, but the standard deviations seem to be too small, they are sometimes smaller than σ_{in}. This is due to two effects: Firstly, the mean value of the output mass is not 1, but k, and σ_{out} has to be scaled accordingly. Secondly, the internal accumulation process smoothes the incoming values, thereby reducing the standard deviation.

A striking result is the mass balance of strategy A in test 14 (k = 2.5), which is much larger than expected. Figure 7 shows that the accumulation of rest masses, which was limited before due to the periodic behaviour, now grows apparently unbounded.

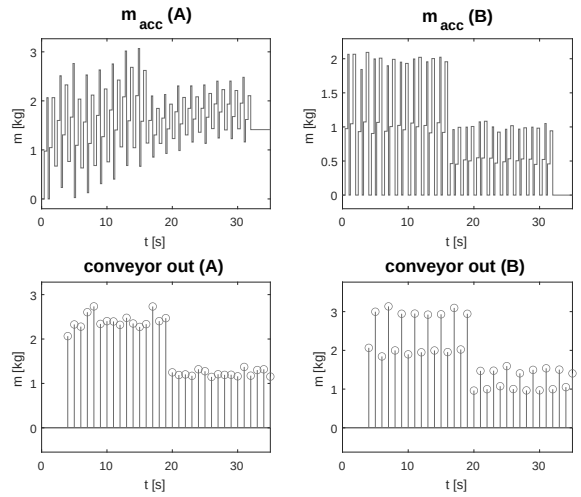

Figure 8: Test result 19, falling input, k = 2.5.

For the last tests a macroscopic change has been added to the small scale fluctuations: In tests 15 – 19 the mean value of the input mass is reduced by a factor two in the middle of the measurement, in tests 20 – 24 it is doubled.

If one takes the changing mass scale into account, the results are similar to the last ones. It is interesting to note, what happens to the previous problematic case $k = 2.5$: As can be seen in Figure 8 the accumulation of "residual mass" continues, even if the mean value drops. The same happens, if the mean value rises (cf. Table 5, no. 24).

5 Conclusion

We have presented a simple discrete event-based model of a conveyor system for the transport of dry bulk material, which allows for the coupling of conveyors with different velocities. Two strategies have been compared to cope with the timing problems, where strategy B is much better, if the short time mass balance is of highest concern, while strategy A provides a better homogeneity of the outgoing masses.

No.	k	$\overline{m_{acc}}(A)$ [kg]	$\overline{m_{acc}}(B)$ [kg]	$\sigma_{out}(A)$ [kg]	$\sigma_{out}(B)$ [kg]
1	0.33	0.3300	0.3300	0.0000	0.0000
2	0.14	0.4243	0.4243	0.0000	0.0000
3	3.00	1.0000	1.0000	0.0000	0.0000
4	7.00	3.0000	3.0000	0.0000	0.0000
5	0.35	0.5315	0.4460	0.0145	0.0394
6	0.80	0.6644	0.4555	0.0000	0.2930
7	1.20	0.8429	0.5000	0.0672	0.3966
8	2.45	1.5315	1.1300	0.1106	0.5040
9	2.50	1.1714	0.9143	0.0884	0.5080
10	0.35	0.5331	0.4440	0.0241	0.0386
11	0.80	0.6825	0.4499	0.0455	0.2933
12	1.20	0.8351	0.4933	0.0867	0.3948
13	2.45	1.3826	1.1141	0.1690	0.4864
14	2.50	1.8598	0.9123	0.1552	0.5136
15	0.35	0.4111	0.3237	0.0875	0.0929
16	0.80	0.6621	0.3391	0.2048	0.3075
17	1.20	0.6865	0.3629	0.2910	0.4185
18	2.45	1.1876	0.8426	0.5983	0.6979
19	2.50	1.5818	0.6838	0.6132	0.7500
20	0.35	0.8212	0.6846	0.1764	0.1809
21	0.80	0.9939	0.6715	0.4006	0.5983
22	1.20	1.3305	0.7239	0.6296	0.9080
23	2.45	2.0568	1.6572	1.2454	1.4854
24	2.50	2.4205	1.3695	1.2898	1.5003

Table 5: Test results comparing mass conservation and homogeneity of both strategies.

While discretisation of continuous systems is important to reduce computation times drastically, it creates problems of its own. To solve them, a precise mathematical description is of uttermost importance, not only to precisely define the model, but also to guide and thereby simplify the implementation process. A typical implementation problem, which had to be solved here, was to ensure the correct ordering of concurrent events. While using priorities is a standard way to cope with it, one had to dig deeply into internal features of SimEvents to come up with a final solution. Since such details vary between different simulation environments [11], a precise (mathematical!) definition of the exact behaviour of SimEvents would have been helpful.

Though the model has shown its principle validity in a series of tests, the real proof of its usefulness would be seen in the integration with a controller.

The coupling with a continuous controller should work in principle, but for several reasons – practical as well as theoretical –, a discrete controller with a finite set of velocities would be more adequate in a lot of applications [12, 13].

Whether the simple strategies proposed here are useful in such a context, or whether one needs more complex strategies, which are adapted to the controller algorithm, is a question for future research.

Acknowledgement

The authors like to thank GRIMME Landmaschinenfabrik GmbH & Co. KG for the permission to publish results of internal research.

References

[1] Kelton WD, Sadowski R, N. Zupick. *Simulation with Arena*. New York: McGraw-Hill, 6th ed. 2015.

[2] Hiltermann J, Lodewijks G, Schott DL, Rijsenbrij J, Dekkers J, Pang Y. A methodology to predict power savings of troughed belt conveyors by speed control. *Particulate science and technology*. 2011;29(1):14–27.

[3] The MathWorks. *SimEvents: Model and simulate discrete-event systems*. `www.mathworks.com/products/simevents.html`.

[4] He D, Pang Y, Lodewijks G. Belt conveyor dynamics in transient operation for speed control. *International Journal of Civil, Environmental, Structural, Construction and Architectural Engineering*. 2016; 10(7):828–833.

[5] Bangsow S. *Tecnomatix Plant Simulation - Modeling and Programming by Means of Examples*. Cham: Springer, 2nd ed. 2020.

[6] Göttlich S, Hoher S, Schindler P, Schleper V, Verl A. Modeling, simulation and validation of material flow on conveyor belts. *Applied mathematical modelling*. 2014; 38(13):3295–3313.

[7] Fioroni MM, Franzese LAG, Zanin CE, Furia J, de Toledo Perfetti L, Leonardo D, da Silva NL. Simulation of continuous behavior using discrete tools: Ore conveyor transport. In: *2007 Winter Simulation Conference*. IEEE. 2007; pp. 1655–1662.

[8] Pihnastyi O, Kozhevnikov G, Khodusov V. Conveyor model with input and output accumulating bunker. In: *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*. IEEE. 2020; pp. 253–258.

[9] Li W, Mani R, Mosterman PJ. Extensible discrete-event simulation framework in SimEvents. In: *Proc. 2016 Winter Simulation Conference*. Arlington: IEEE. 2016; pp. 943–954.

[10] The MathWorks. *Simulink: Simulation and Model-Based Design*. `https://www.mathworks.com/products/simulink/`.

[11] Schriber TJ, Brunner DT, Smith JS. How discrete-event simulation software works and why it matters. In: *Proceedings of the 2012 Winter Simulation Conference (WSC)*. IEEE. 2012; pp. 1–15.

[12] Pang Y, Lodewijks G, Schott DL. Fuzzy Controlled Energy Saving Solution for Large-Scale Belt Conveying Systems. In: *Energy, Environment and Sustainable Development*, vol. 260 of *Applied Mechanics and Materials*. Trans Tech Publications Ltd. 2013; pp. 59–64.

[13] Reutov AA. Simulation of load traffic and steeped speed control of conveyor. In: *IOP Conference Series: Earth and Environmental Science*, vol. 87. IOP Publishing. 2017; p. 082041.

EUROSIM Data and Quick Info

ASIM SPL 2023 — **20th ASIM Dedicated Conference on Simulation in Production and Logistics**
September 13-15, 2023, Ilmenau, Germany www.asim-fachtagung-spl.de

I3M 2023 **International Multidisciplinary Modeling & Simulation Multiconference**
September 18-20, 2023, Athens, Greece www.msc-les.org/i3m2023

SIMS 2023 – 64. Int. Conference
September 26-27, 2023, Västerås
www.scansims.org

Winter Simulation Conference 2023,
December 10-13, 2023,
San Antonio, TX, USA www.wintersim.org

EUROSIM CONGRESS 2026
July 2026, Italy www.eurosim.info

ASIM SST 2024 – Symposium Simulation Technique
Munich, September 2024

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

EUROSIM members may be national simulation societies and regional or international societies and groups dealing with modelling and simulation.

Full Members are ASIM, CEA-SMSG, CSSS, DBSS, KA-SIM, LIOPHANT, LSS, PTSK, NSSM, SIMS, SLOSIM, UKSIM. Observer Members are ALBSIM and ROMSIM. Former Members (societies in re-organisation) are: CROS-SIM, FRANCOSIM, HSS, ISCS.

EUROSIM is governed by a B**oard** consisting of one representative of each member society, president, past president, and **SNE** representative.

President	Agostino Bruzzone (LIOPHANT) *agostino@itim.unige.it*
Past President	M. Mujica Mota (DBSS), *m.mujica.mota@hva.nl*
Secretary	Marina Massei (LIOPHANT), *massei@itim.unige.it*
Treasurer	Felix Breitenecker (ASIM) *felix.breitenecker@tuwien.ac.at*
Webmaster	Irmgard Husinsky (ASIM), *irmgard.husinsky@tuwien.ac.at*

SNE – Simulation Notes Europe is EUROSIM's membership journal with peer reviewed scientific contributions about all areas of modelling and simulation, including new trends as big data, cyber-physical systems, etc.

The EUROSIM societies distribute **e-SNE** in full version to their members as official membership journal. The basic version of **e-SNE** is available with open access. Publishers are **EUROSIM**, **ARGESIM** and **ASIM**,

www.sne-journal.org
office@sne-journal.org

SNE-Editor: Felix Breitenecker (ASIM)
felix.breitenecker@eic@sne-journal.org

EUROSIM Congress and Conferences

Each year a major **EUROSIM** event takes place, as the **EUROSIM CONGRESS** organised by a member society, **SIMS EUROSIM Conference**, and **MATHMOD Vienna Conference (ASIM)**.

On occasion of the **EUROSIM Congress 2023**, the *11th EUROSIM Congress* in Amsterdam, July, 2023, a new **EUROSIM** president has been elected: we welcome Agostino Bruzzone, well known simulationist, as new president. His society **LIOPHANT** will organize the next EUROSIM Congress in 2026 in Italy.

Furthermore, **EUROSIM Societies** organize local conferences, and **EUROSIM** co-operates with the organizers of **I3M Conference** and **WinterSim Conference Series.**

EUROSIM Member Societies

ASIM
German Simulation Society
Arbeitsgemeinschaft Simulation

ASIM is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria.

President	Felix Breitenecker, *felix.breitenecker@tuwien.ac.at*
Vice President	Sigrid Wenzel, *s.wenzel@uni-kassel.de* Thorsten Pawletta, *thorsten.pawletta@hs-wismar.de* Andreas Körner, *andreas.koerner@tuwien.ac.at*

ASIM is organising / co-organising the following international conferences: **ASIM SPL** Int. Conference 'Simulation in Production and Logistics' (biannual), **ASIM SST** 'Symposium Simulation Technique' (biannual), **MATHMOD** Int. Vienna Conference on Mathematical Modelling (triennial). Furthermore, **ASIM** is co-sponsor of *WSC* - Winter Simulation Conference and of the *I3M* and conference series.

ASIM Working Committees

GMMS: Methods in Modelling and Simulation
U. Durak, umut.durak@dlr.de
SUG: Simulation in Environmental Systems
J. Wittmann, wittmann@informatik.uni-hamburg.de
STS: Simulation of Technical Systems
W. Commerell, commerell@hs-ulm.de
SPL: Simulation in Production and Logistics
S. Wenzel, s.wenzel@uni-kassel.de
EDU: Simulation and Education
A. Körner, andreas.koerner@tuwien.ac.at
Working Group Big Data: Data-driven Simulation in Life Sciences, N. Popper, niki.popper@dwh.at
Other Working Groups: Simulation in Business Administration, in Traffic Systems, for Standardisation, etc.

Contact Information

www.asim-gi.org

info@asim-gi.org, admin@asim-gi.org

ASIM – Inst. of Analysis and Scientific Computing, TU Wien, Wiedner Hauptstraße 8-10, 1040 Vienna, Austria

CEA-SMSG – Spanish Modelling and Simulation Group

CEA is the Spanish Society on Automation and Control. The association is divided into national thematic groups, one of which is centered on Modeling, Simulation and Optimization (**CEA-SMSG**).

President	José L. Pitarch, *jlpitarch@isa.upv.es*
Vice President	Juan Ignacio Latorre, *juanignacio.latorre@unavarra.es*

Contact Information

www.ceautomatica.es/grupos/

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

CSSS – Czech and Slovak Simulation Society

CSSS is the Simulation Society with members from the two countries: Czech Republic and Slovakia. The CSSS history goes back to 1964.

President	Michal Štepanovský *michal.stepanovsky@fit.cvut.cz*
Vice President	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*

Contact Information

cssim.cz

michal.stepanovsky@fit.cvut.cz

CSSS – Český a Slovenský spolek pro simulaci systémů, Novotného lávka 200/5, 11000 Praha 1, Česká republika

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.

President	M. Mujica Mota, *m.mujica.mota@hva.nl*
Vice President	A. Heemink, *a.w.heemink@its.tudelft.nl*
Secretary	P. M. Scala, *paolo.scala@fedex.com*

Contact Information

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

KA-SIM Kosovo Simulation Society

The Kosova Association for Modeling and Simulation (**KA-SIM**) is closely connected to the University for Business and Technology (UBT) in Kosovo.

President	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Vice President	Muzafer Shala, *info@ka-sim.com*

Contact Information

www.ubt-uni.net

ehajrizi@ubt-uni.net

Dr. Edmond Hajrizi
Univ. for Business and Technology (UBT)
Lagjja Kalabria p.n., 10000 Prishtina, Kosovo

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.

President	A.G. Bruzzone, *agostino@itim.unige.it*
Director	E. Bocca, *enrico.bocca@liophant.org*

Contact Information

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

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.

President	Artis Teilans, *Artis.Teilans@rta.lv*
Vice President	Oksana Kuznecova, *Oksana.Kuznecova@rtu.lv*

Contact Information

www.itl.rtu.lv/imb/

Artis.Teilans@rta.lv, Egils.Ginters@rtu.lv

LSS, Dept. of Modelling and Simulation, Riga Technical University, Kalku street 1, Riga, LV-1658, Latvia

NSSM – National Society for Simulation Modelling (Russia)

NSSM – The National Society for Simulation Modelling (Национальное Общество Имитационного Моделирования – НОИМ) was officially registered in Russia in 2011.

President	R. M. Yusupov, *yusupov@iias.spb.su*
Chairman	A. Plotnikov, *plotnikov@sstc.spb.ru*

Contact Information

www.simulation.su

yusupov@iias.spb.su

NSSM / R. M. Yusupov, St. Petersburg Institute of Informatics and Automation RAS, 199178, St. Petersburg, 14th line, h. 39

PTSK – Polish Society for Computer Simulation

PTSK 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.

President	Tadeusz Nowicki, *Tadeusz.Nowicki@wat.edu.pl*
Vice President	Leon Bobrowski, *leon@ibib.waw.pl*

Contact Information

www.ptsk.pl

leon@ibib.waw.pl

PSCS, 00-908 Warszawa 49, ul. Gen. Witolda Urbanowicza 2, pok. 222

SIMS – Scandinavian Simulation Society

SIMS is the Scandinavian Simulation Society with members from the five Nordic countries Denmark, Finland, Norway, Sweden and Iceland. The **SIMS** history goes back to 1959.

President	Tiina Komulainen, *tiina.komulainen@oslomet.no*
Vice President	Erik Dahlquist, *erik.dahlquist@mdh.se*

Contact Information

www.scansims.org

vadime@wolfram.com

Vadim Engelson, Wolfram MathCore AB, Teknikringen 1E, 58330, Linköping, Sweden

SLOSIM – Slovenian Society for Simulation and Modelling

The Slovenian Society for Simulation and Modelling was established in 1994. It promotes modelling and simulation approaches to problem solving in industrial and in academic environments by establishing communication and cooperation among corresponding teams.

President	Goran Andonovski, *goran.andonovski@fe.uni-lj.si*
Vice President	Božidar Šarler, *bozidar.sarler@fs.uni-lj.si*

Contact Information

www.slosim.si

slosim@fe.uni-lj.si, vito.logar@fe.uni-lj.si

SLOSIM, Fakulteta za elektrotehniko, Tržaška 25, SI-1000, Ljubljana, Slovenija

UKSIM - United Kingdom Simulation Society

The UK Modelling & Simulation Society (**UKSim**) is the national UK society for all aspects of modelling and simulation, including continuous, discrete event, software and hardware.

President	David Al-Dabass, *david.al-dabass@ntu.ac.uk*
Secretary	T. Bashford, *tim.bashford@uwtsd.ac.uk*

Contact Information

uksim.info

david.al-dabass@ntu.ac.uk

UKSIM / Prof. David Al-Dabass, Computing & Informatics, Nottingham Trent University, Clifton lane, Nottingham, NG11 8NS, United Kingdom

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.

Contact Information

florin_h2004@yahoo.com
ROMSIM / Florin Hartescu, National Institute for Research in Informatics, Averescu Av. 8 – 10, 011455 Bucharest, Romania

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.

Contact Information

kozeta.sevrani@unitir.edu.al

Albanian Simulation Goup, attn. Kozeta Sevrani, University of Tirana, Faculty of Economy , rr. Elbasanit, Tirana 355, Albania

Former Societies / Societies in Re-organisation

- **CROSSIM** – *Croatian Society for Simulation Modelling*
  Contact: Tarzan Legović, *Tarzan.Legovic@irb.hr*
- **FrancoSim** – Société Francophone de Simulation
- **HSS** – Hungarian Simulation Society
  Contact: A. Gábor, *andrasi.gabor@uni-bge.hu*
- **ISCS** – Italian Society for Computer Simulation

The following societies have been formally terminated:

- **MIMOS** –Italian Modeling & Simulation Association; terminated end of 2020.

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 and 978-3-903347-xx-y, root DOI 10.11128/z…zz.zz. Publication is open for **ASIM** and for **EUROSIM Member Societies**.

SNE – Simulation Notes Europe

The scientific journal **SNE – Simulation Notes Europe** 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** puts special emphasis on the overall view in simulation, and on comparative investigations.
Furthermore, **SNE** welcomes contributions on education in/for/with simulation.

SNE is also the forum for the **ARGESIM Benchmarks** on *Modelling Approaches and Simulation Implementations* publishing benchmarks definitions, solutions, reports and studies – including model sources via web.

SNE Editorial Office /ARGESIM

→**www.sne-journal.org**

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

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

SNE, primarily an electronic journal, follows an open access strategy, with free download in a basic version (B/W, low resolution graphics). **SNE** is the official membership journal of **EUROSIM**, the *Federation of European Simulation Societies*. Members of (most) **EUROSIM Societies** are entitled to download the full version of e-**SNE** (colour, high-resolution graphics), and to access additional sources of benchmark publications, model sources, etc. (group login for the 'publication-active' societies; please contact your society). Furthermore, **SNE** offers **EUROSIM Societies** a publication forum for post-conference publication of the society's international conferences, and the possibility to compile thematic or event-based **SNE Special Issues**.

Simulationists are invited to submit contributions of any type – *Technical Note*, *Short Note*, *Project Note*, *Educational Note*, *Benchmark Note*, etc. via **SNE**'s website:

→ **www.sne-journal.org**,

As the largest European simulation conference for production and logistics, every two years the ASIM Dedicated Conference presents forward-looking trends and current developments, scientific papers and interesting applications in the industry.

The thematic focus of the next conference is sustainability in production and logistics. Thus, the conference will address an important social issue and at the same time focus on current research topics in the simulation world.

Keynotes:

Sustainable transformation of industrial production - illustrated by the example of climate-friendly steel production
Prof. Dr. Thomas S. Spengler, Lehrstuhl für Produktion und Logistik; TU Braunschweig

Digital Twins – A Journey from Particle Physics at CERN to Industry 4.0 Manufacturing in Singapore
Dr. Peter Lendermann, D-SIMLAB Technologies Ptd Ltd, Singapore

Reports from research and teaching, development and industrial use are deliberately placed on an equal footing. Workshops and tutorials, an evening dialog event as well as the accompanying company exhibition with software vendors and service providers of the simulation industry create, in addition to the scientific program, manifold opportunities for in-depth discussions and for getting to know current topics and offers. Presentations are available in German and English.

We look forward to welcoming you to Ilmenau in September 2023.

www.asim-fachtagung-spl.de/asim2023/en/