Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

How to Simulate Networked Control Systems with Variable Time Delays? ⋆

Martin Steinberger * Markus Tranninger * Martin Horn *,**
Karl Henrik Johansson ***

* *Institute of Automation and Control, Graz University of Technology, 8010 Graz, Austria, (e-mail: martin.steinberger@tugraz.at, markus.tranninger@tugraz.at, martin.horn@tugraz.at)*
** *Christian Doppler Laboratory for Model Based Control of Complex Test Bed Systems, Institute of Automation and Control, Graz University of Technology, Graz, Austria.*
*** *Division of Decision and Control Systems, Royal Institute of Technology, 100 44 Stockholm, Sweden (e-mail: kallej@kth.se)*

Abstract: Techniques to accurately simulate networked control systems with bounded time-varying delays are proposed. They link delays in transmission channels to individual packets and can take into account packet disordering that may occur in wireless multi-hop networks. Examples underpin the properties of the proposed approach and show why other simulation and modeling approaches may fail.

Keywords: networked control systems, time-varying delays, delay modeling, network modeling, simulation with variable delays, packet disordering, large delay case, true time

1. INTRODUCTION

In Networked Control Systems (NCS) controllers are connected to sensors and actuators via wired or wireless communication channels. This leads to additional challenges in the controller design due to, e.g., packet dropouts and time-varying packet delays in the transmission channels, see Park et al. (2018), Zhang et al. (2017), Heemels and van de Wouw (2010) and references therein.

The literature is very rich in terms of different approaches to tackle these issues. Ludwiger et al. (2018) and Ludwiger et al. (2019) proposed robust controllers for networked systems that exploit a buffering mechanism. In Repele et al. (2014), an adaptive buffering approach is reported to reduce the conservatism of classical buffering approaches.

A wide range of control design techniques consider the effects of time-varying delays. These include, for example, (i) robust networked predictive control (Mu et al. (2005)), (ii) predictor-based schemes (Liu et al. (2012)), (iii) LMI-based designs (Cloosterman et al. (2010), Liu and Fridman (2012), Liu (2010)), (iv) adaptive sliding mode control (Xia et al. (2010)) and (v) adaptive Smith predictor approaches (Batista and Jota (2018)).

All these design approaches assume that the time-varying delay is lower and upper bounded. Network simulation tools like TrueTime [1] (Cervin et al. (2003)) and OMNeT++ [2] may not directly be used for validation purposes, since they do not provide the possibility to define arbitrary transmission delays. They model the transmission channels in some detail, e.g., based on physical principles (like location of the nodes, channel fading effects) and the implemented communication protocols.

The contributions of the present paper are to illustrate that existing NCS simulation tools might give inconsistent results for loops with bounded time-varying delays, and then to propose two new approaches of how such simulations could be performed using specific packet delay models. Our results can be extended to simple models of networks with packet dropouts, but does not cover more detailed network protocols or physical-layer models.

The first proposed technique is limited to delays equal to an integer multiple of the sampling time, while the second technique is also applicable for simulating arbitrary delays. Both techniques extend basic Matlab/Simulink [3] and TrueTime blocks to accurately simulate packetized transmission channels with time-varying delays. Simulation examples provide insights to the properties of the different approaches and compare the outcomes to simulation results based on models from Li and Gao (2011), Gao and Chen (2007) and Heemels and van de Wouw (2010).

⋆ This work was supported by the LEAD project "Dependable Internet of Things in Adverse Environments" funded by Graz University of Technology. The financial support by the Christian Doppler Research Association, the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

[1] http://www.control.lth.se/research/tools-and-software/truetime/ (accessed on 27.10.2019)
[2] https://omnetpp.org/intro/ (accessed on 27.10.2019)
[3] www.mathworks.com (accessed on 27.10.2019)

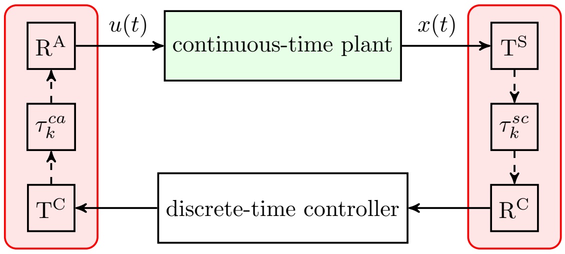

Fig. 1. Networked control system with a plant that is connected to a controller via two transmission channels (transmitter T, variable time delay τ_k, receiver R).

2. PROBLEM STATEMENT

Consider a networked control system consisting of a continuous-time plant and a discrete-time controller that are connected via two transmission channels as shown in Fig. 1. The plant is modeled as a linear time-invariant system

$$\frac{dx}{dt} = \tilde{A}x(t) + \tilde{B}u(t) \tag{1}$$

with states $x(t) \in \mathbb{R}^n$, inputs $u(t) \in \mathbb{R}^m$ and known and constant matrices $\tilde{A} \in \mathbb{R}^{n\times n}$ and $\tilde{B} \in \mathbb{R}^{n\times m}$. All states $x(t)$ are sampled with a constant sampling time h at the transmitter on the plant (sensor) side (T^{S} in Fig. 1). The transmission towards the receiver on the controller side (R^{C}) is subject to a time-varying delay τ_k^{sc}.

The discrete-time linear state feedback controller acts in an event-triggered fashion, i.e. it calculates the control signal as soon as a new packet arrives, and sends it via the transmitter on the controller side (T^{C}) through a channel with time-varying delay τ_k^{ca} to the receiver at the plant (actuator) side (R^{A}). A zero-order-hold mechanism is employed at the actuator side to obtain the continuous control signal $u(t)$, see Fig. 1.

Assumption 1. Both variable time delays τ_k^{sc} and τ_k^{ca} are bounded such that

$$0 \le \tau_{\min}^{sc} \le \tau_k^{sc} \le \tau_{\max}^{sc}\ , \quad 0 \le \tau_{\min}^{ca} \le \tau_k^{ca} \le \tau_{\max}^{ca}\ . \tag{2}$$

The upper bounds $\tau_{\max}^{sc}$ and $\tau_{\max}^{ca}$ may be greater than the sampling time h.

The goal is to evaluate different possible strategies to achieve accurate simulation results under the presence of time-varying delays and to and compare them to results based on mathematical descriptions of closed loop networked systems from literature. In addition it should be investigated how Matlab/Simulink and TrueTime (Cervin et al. (2003)) standard blocks could be extended to reproduce correct packet transmissions.

3. SIMULATION OF CHANNELS WITH VARIABLE TIME DELAYS

First, only one transmission channel with variable time delay τ_k is considered to evaluate different simulation approaches. In the test scenario, a ramp signal $v(t)$ is sampled with sampling time $h = 1$ and sent over the channel with corresponding time delays τ_k as depicted in the top plot of Fig. 2. The signal $v(t)$ and its sampled version v_k are shown in blue in Fig. 2.

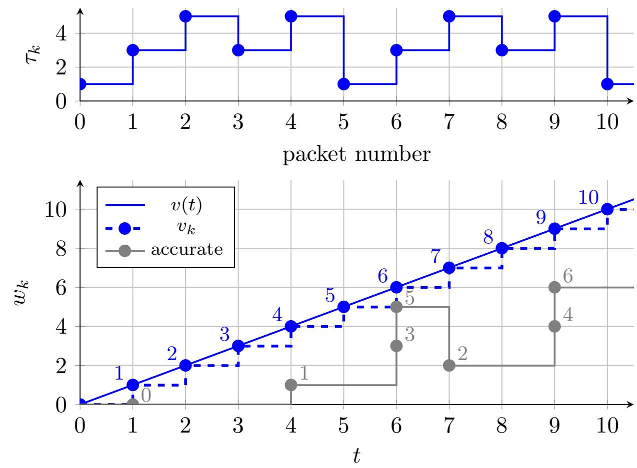

Fig. 2. Evaluation example of simulation methods for a transmission channel with variable time delay τ_k (top plot): continuous-time signal to be transmitted $v(t)$, transmitted packets v_k (blue) and accurately received packets w_k (gray).

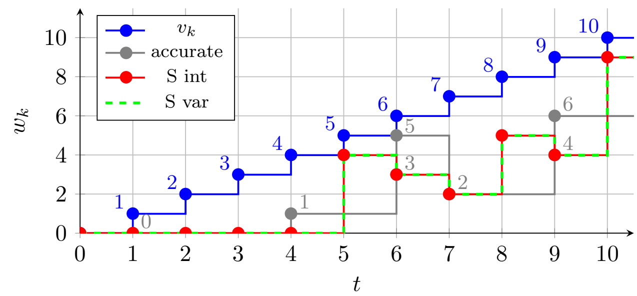

Fig. 3. Comparison of the accurate result (gray) with results from simulations using: Simulink integer delay (S int), Simulink variable time delay (S var).

Packet 0 should be delayed by 1, packet 1 by 3, packet 2 by 5, and so on. The correct, i.e. the accurately simulated, arriving packets w_k at the receiver are plotted in gray in Fig. 2. Note that several packets could arrive at the same time (e.g., at $t = 6$) yielding by definition a value of w_k according to the most recent packet (e.g. packet 5 at $t = 6$). Also note that packet disordering (packets 1, 5, 2, 6 in Fig. 2) is possible at the receiver. This may occur, e.g., in multi-hop networks where different routing paths between the transmitter and the receiver are possible.

Different approaches that are expected to lead to the accurate results as in Fig. 2 are compared subsequently. For simplicity reasons, the time-varying delay is assumed to be a multiple of the sampling time.

3.1 Simulations with Simulink Standard Blocks

A natural way to simulate time-varying delays is to exploit the Simulink built-in blocks *Variable Integer Delay* (S int) or *Variable Time Delay* (S var). The block *Variable Transport Delay* must not be used in the context of NCS because it relates to physical processes where a medium is transported with a specified speed as pointed out in Zhang and Yeddanapudi (2012).

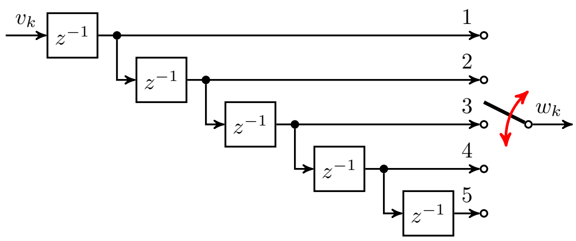

Fig. 4. Functional principle of the *Simulink Integer delay* block.

Fig. 3 reveals the fact that both blocks should not be used to simulate a packet based time-varying communication channel. The resulting received information strongly deviates from the accurate behavior because the specified delay is related to sampling instances and not (as desired) to individual packets in the networked control loop.

This can be clearly seen for the considered example using the *Variable Integer Delay* block. It is internally structured as shown in Fig. 4, i.e. a tapped delay line in combination with a switching mechanism is utilized to realize a variable time delay. In the example presented in Fig. 3, all delay blocks in Fig. 4 are initialized with zero. The input sequence v_k increases by one in each sampling step h and is transferred to the five delay blocks with internal states $x_1, x_2, \ldots, x_5$, as shown in Table 1.

Table 1. Results for the simulation example plotted in Fig. 3 for the case with Simulink integer delay (S int) block.

t	0	1	2	3	4	5	6	7	8	9
x_1	$\boxed{0}$	0	1	2	3	$\boxed{4}$	5	6	7	8
x_2	0	0	0	1	2	3	4	5	6	7
x_3	0	$\boxed{0}$	0	$\boxed{0}$	1	2	$\boxed{3}$	4	$\boxed{5}$	6
x_4	0	0	0	0	0	1	2	3	4	5
x_5	0	0	$\boxed{0}$	0	$\boxed{0}$	0	1	$\boxed{2}$	3	$\boxed{4}$
τ_k	1	3	5	3	5	1	3	5	3	5

The internal switch selects port 1 to 5 depending on the specified delay τ_k for the actual sampling step, cf. top plot of Fig. 2. The resulting sequence w_k is visualized in Table 1 by using red boxes. These values coincide with the presented signals in Fig. 3 (S int). Non-integer values for the desired delays are truncated in the this Simulink implementation.

Hence, the desired delay should not directly be used as an input signal to the *Simulink Integer Delay* block. Instead the proposed Algorithm 1, which extends the Simulink built-in block, can be used to link the packets to the desired packet delays to achieve accurate simulation results, as shown in Fig. 5 (S int ext).

In some applications it may be advantageous to skip old packets in the case that newer packets are available, see black signal in Fig. 5 (S int ext skip).

3.2 Simulations using TrueTime

An alternative way to simulate a time-varying channel is to use the toolbox TrueTime (Cervin et al. (2003)). It was developed to enable a co-simulation of controller tasks in real-time kernels, continuous plant dynamics and network transmissions including task scheduling. A big variety of different (wireless) network configurations and protocols can be chosen within the simulation toolbox.

Algorithm: Input generation for Simulink Var. Delay Block

Initialize packet buffer (length: $\bar{d}+1$);
while *simulation is running* **do**
  Shift elements in buffer by 1;
  Subtract h from all delays in the packet buffer;
  Get actual packet delay (in multiples of h);
  Write actual packet delay at first position in buffer;
  if *skip old packets* **then**
    Replace negative delays in buffer by 0;
  end
  Choose most recent packet with delay 0 (from buffer);
end

Algorithm 1. Pseudo code for the delay input signal generation for the *Simulink Integer Delay* block.

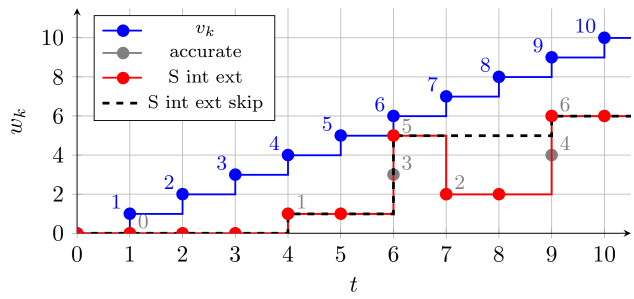

Fig. 5. Comparison of the accurate result (gray) with results from simulations using: Extended Simulink integer delay (S int ext), extended Simulink integer delay with mechanism to skip old packets (S int ext skip).

Fig. 6 shows the packets that are sent (blue) and received (green) when a basic TrueTime simulation (TT) is used. Since no additional network traffic is simulated in this example, the green packets immediately appear at the receiver. Hence, an extension of TrueTime to specify individual packet delays is necessary.

The link between specified delays and the individual packets can be established by the following proposed extension for TrueTime. It introduces an additional block (Network Delay Node) between transmitter (Send Node) and receiver (Receive Node).

This node schedules the arriving packets in the considered transmission channel. Packets "on the fly" are stored in an internal list with size $\bar{d} - \underline{d} + 1$, where integers

$$\underline{d} = \left\lfloor \frac{\tau_{min}}{h} \right\rfloor \qquad \text{and} \qquad \bar{d} = \left\lceil \frac{\tau_{max}}{h} \right\rceil \tag{3}$$

depend on the sampling time h as well as on the minimal delay τ_{min} and maximal delay τ_{max} respectively. Algorithm 2 is implemented in the Network Delay Node to schedule the packets that should be sent with individual specified delays τ_k. In addition, a delay job is used that sleeps until the packet related to the first entry in the list is sent to the receiver. Then, the first element is removed and a new delay job corresponding to the next packet in the list is started.

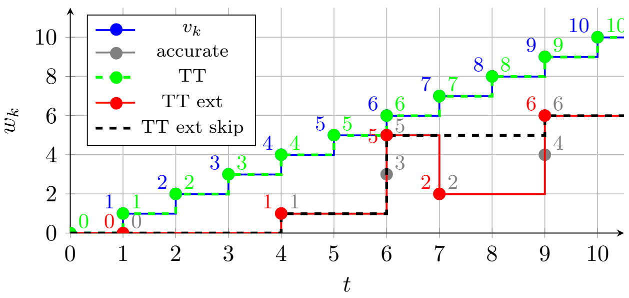

Fig. 6. Comparison of the accurate result (gray) with results from simulations using: basic TrueTime (TT), TrueTime extended (TT ext), TrueTime extended with mechanism to skip old packets (TT ext skip).

Algorithm: Network Delay Node
Initialize packet buffer (length: $\bar{d} - \underline{d} + 1$);
while *simulation is running* **do**
  Get actual packet (to be scheduled for time t^*) from transmitter;
  Read actual delay job (next packet that has already been scheduled to be received next) from list;
  if *no packet has already been scheduled for* t^* **then**
    // *not scheduled*
    if *time* t^* *less than next planned packet arrival at receiver* **then**
      Cancel running delay job (next packet to be received);
      Include actual packet at first position in list;
      Move other packets backward in the list;
      Start new delay job for the actual packet;
    else
      Include actual packet in list;
      Sort list based on the planned arrival times;
    end
  else
    // *already scheduled*
    Replace packet in list (that was scheduled for time t^*) with the actual packet;
  end
end

Algorithm 2. Pseudo code for the network delay node that schedules the packet arrivals in the transmission channel.

Fig. 6 shows the results using this proposed extended TrueTime (TT ext) simulation approach. It perfectly reproduces the accurate (gray) behavior of the packetized transmission channel.

A slight modification of the code of the receiving node allows to skip old packets in the case that newer packets are available, see black signal in Fig. 6 (TT ext skip).

One additional advantage of the presented approach is that it is also capable to deal with real-world scenarios where variable packet delays are not multiples of the sampling time.

4. COMPARISON WITH NCS MODELS FOR VARIABLE TIME DELAYS

In this section, the approaches for the simulation of time-varying delays are compared for a networked feedback loop as in Fig. 1 using a linear time-invariant state feedback controller in the form

$$u_k = L\, x_{k-\tau_k} \tag{4}$$

which makes use of the delayed plant state $x_{k-\tau_k}$. For this static controller, both time-varying delays can be combined to one delay $\tau_k = \tau_k^{sc} + \tau_k^{ca}$ as, e.g., indicated in Heemels and van de Wouw (2010). Additionally, two different ways to model the closed loop are evaluated with respect to the simulation approaches shown in the previous section.

4.1 Model with Time-Delay

A first possibility to model the closed loop is to directly combine the discretized plant (1) with controller (4) yielding

$$x_{k+1} = Ax_k + BL\, x_{k-\tau_k} \tag{5}$$

with

$$A = e^{\tilde{A}h} \qquad \text{and} \qquad B = \int_0^h e^{\tilde{A}\eta} \tilde{B}\, d\eta\ , \tag{6}$$

as, e.g. presented in Li and Gao (2011) and Gao and Chen (2007). This model uses the actual states x_k and states $x_{k-\tau_k}$, which are shifted in time. Note that only delays that are multiple of the sampling times can be considered.

4.2 Lifted Model

An alternative way to model the considered NCS is presented in Heemels and van de Wouw (2010). The use of an extended (lifted) state vector

$$\xi_k = \begin{bmatrix} x_k^T \ u_{k-1}^T \ u_{k-2}^T \cdots u_{k-\bar{d}}^T \end{bmatrix}^T \in \mathbb{R}^{n+m\bar{d}}\ , \tag{7}$$

allows to formulate a closed loop model such that

$$\xi_{k+1} = \mathcal{A}\xi_k + \mathcal{B}u_k \tag{8}$$

with matrices

$$\mathcal{A} = \begin{bmatrix} e^{\tilde{A}h} & M_{\bar{d}-1} & M_{\bar{d}-2} & \cdots & M_2 & M_1 & M_0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & I_m & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & I_m & \ddots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & I_m & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & I_m & 0 \end{bmatrix}, \tag{9a}$$

$$\mathcal{B} = \begin{bmatrix} M_{\bar{d}}^T & I_m & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}^T . \tag{9b}$$

Matrices

$$M_j(\tau_k) = \begin{cases} \displaystyle\int_{h-t_{j+1}^k}^{h-t_j^k} e^{\tilde{A}\eta}\tilde{B}\, d\eta & \text{for} \quad 0 \le j \le \bar{d} - \underline{d} \\ 0 & \text{for} \quad \bar{d} - \underline{d} < j \le \bar{d} \end{cases} \tag{10}$$

depend on the actual arrival times t_j^k of the individual packets and integers $\underline{d}$, $\bar{d}$ as defined in (3). Because the packet delays τ_k are known in simulation, the arrival times t_j^k can be calculated as shown in Heemels and van de Wouw (2010). Hence matrices $\mathcal{A} \in \mathbb{R}^{(n+m\bar{d})\times(n+m\bar{d})}$ and $\mathcal{B} \in \mathbb{R}^{(n+m\bar{d})\times m}$ are time-varying and known. Time delays that are not multiples of the sampling time can be included easily in this approach.

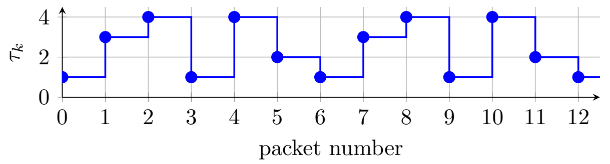

Fig. 7. Repeating sequence of variable time delays for the closed loop simulation.

4.3 Simulation Example

The plant dynamics is given by a double integrator,

$$\tilde{A} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad \tilde{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} . \tag{11}$$

Following the approach of Cloosterman et al. (2010), a state controller

$$u_k = L\, x_k = [L\ 0\ \cdots\ 0]\, \xi_k \tag{12}$$

is designed that stabilizes the feedback loop under the presence of a bounded but time-varying delay, where $h = 1$, $\underline{d} = 0$, $\overline{d} = 4$ and controller design parameter $\gamma = 0.001$ are used to get L in (12) by means of the design approach presented in Cloosterman et al. (2010). The initial states of the plant and the initial conditions of the network are chosen equal to $x(t = 0) = [2\ \ -1]^T$ and zero respectively. The packet delays are defined by a repeating sequence that is shown in Fig. 7.

Fig. 8 depicts the simulation results for the proposed extended TrueTime simulation where, in addition, old packets are skipped (TT ext skip). The plots show the control signal as well as the transmitted (tx) and received (rx) states. It can be seen, that the lifted model yields identical results when compared to the continuous plant states at the sampling instants.

In contrast, deviations from these accurate (gray) signals can be seen in Fig. 9 where the extended TrueTime simulation without the dropping mechanism for old packets is used. This becomes evident by inspecting the transmitted signals (TT ext, tx).

Fig. 10 reinforces the statements from the previous section that the Simulink built-in block *Variable integer delay* is not suitable for solving the considered task. Interestingly, the model with time-delay (5) yields the same results as indicated in Fig. 10. This is due to the fact that the desired delays (see Fig. 7) should be related to a specific packet, but are actually associated with the iteration index instead of the individual packets.

5. CONCLUSION AND OUTLOOK

This paper evaluates different approaches to simulate variable time delays in networked control loops. It is pointed out that simulation of packetized transmissions with individual time-varying delays for each packet should be done with care.

An extended TrueTime approach is proposed to provide a well founded strategy to accurately simulate NCS for bounded varying delays. It is not limited to delays that are multiples of the sampling time and can cope with packet disordering in the transmission channels. An extension of

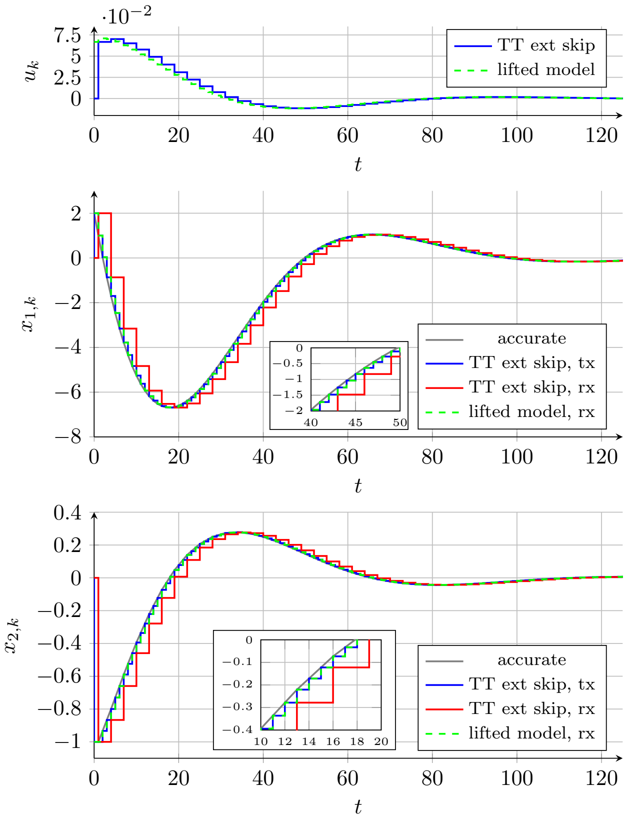

Fig. 8. Closed loop simulation: TrueTime extended with mechanism to skip old packets (TT ext skip). Comparison of the accurate continuous-time signal, the transmitted (tx) and received (rx) signal, as well as the lifted states corresponding to (8).

the *Simulink Integer Delay* block is presented that enables accurate simulations for the case where the delays are multiples of the sampling time.

Two different possibilities to model the closed loop NCS are compared for the case using a linear time-invariant plant and a linear state feedback controller that acts in an event-driven fashion. For the model with time-delay, the relation between packets and delays (to be simulated) is lost. In contrast, the lifted model reproduces the expected results for the case that old packets are discarded whenever newer packets are available. This is possible due to the calculation of the arrival times t_j^k.

The presented simulation approach can easily be extended to incorporate packet dropouts in transmission channels of networked systems with variable time delays.

REFERENCES

Batista, A.P. and Jota, P.G. (2018). Performance improvement of an NCS closed over the internet with an adaptive Smith Predictor. *Control Engineering Practice*, 71, 34–43.

Cervin, A., Henriksson, D., Lincoln, B., Eker, J., and Arzen, K.. (2003). How does control timing affect performance? Analysis and simulation of timing using Jitterbug and TrueTime. *IEEE Control Systems Magazine*, 23(3), 16–30.

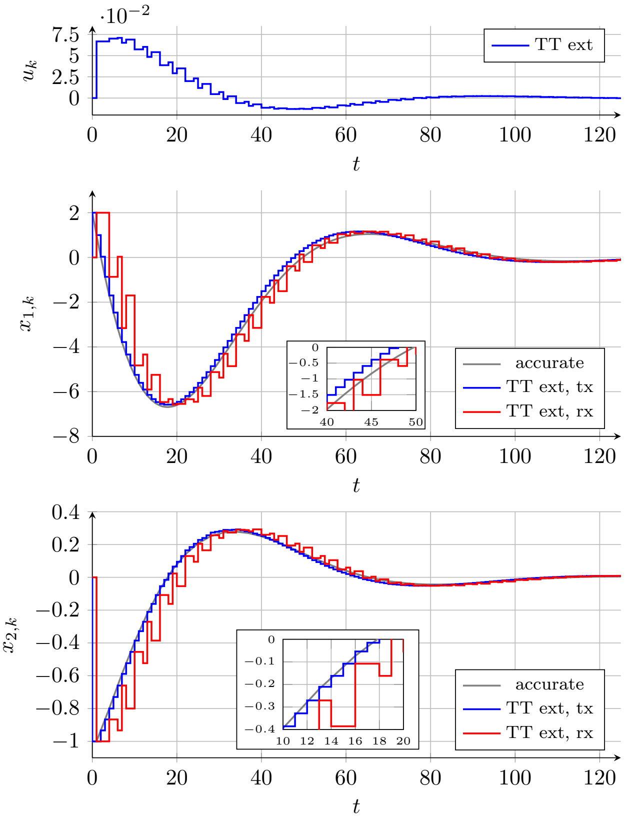

Fig. 9. Closed loop simulation: TrueTime extended (TT ext). Comparison of the accurate continuous-time signal, the transmitted (tx) and received (rx) signal.

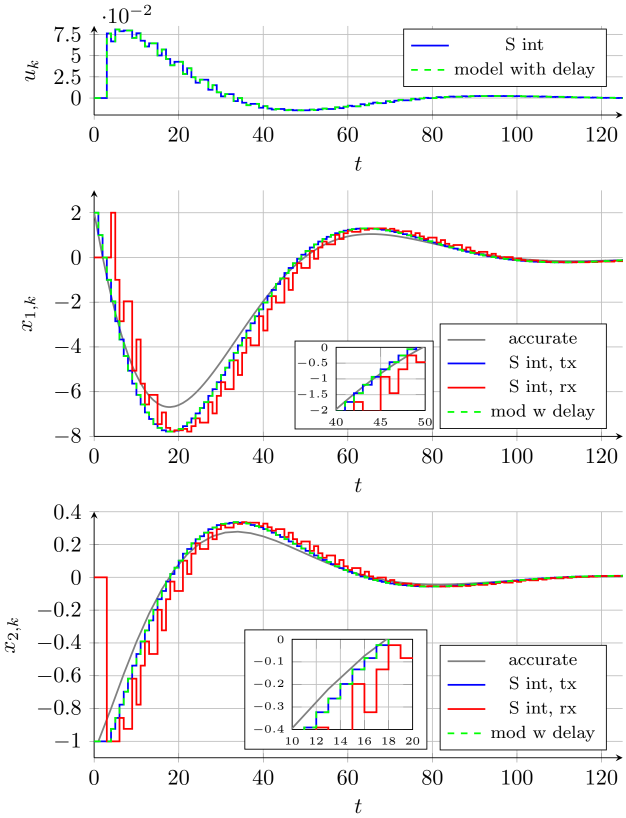

Fig. 10. Closed loop simulation: Simulink integer delay (S int). Comparison of the accurate continuous-time signal, the transmitted (tx) and received (rx) signal as well as the states corresponding to (5).

Cloosterman, M., Hetel, L., van de Wouw, N., Heemels, W., Daafouz, J., and Nijmeijer, H. (2010). Controller synthesis for networked control systems. *Automatica*, 46(10), 1584 – 1594.

Gao, H. and Chen, T. (2007). New results on stability of discrete-time systems with time-varying state delay. *IEEE Transactions on Automatic Control*, 52(2), 328–334.

Heemels, W.P.M.H. and van de Wouw, N. (2010). Stability and stabilization of networked control systems. In Bemporad A., Heemels M., Johansson M. (ed.), *Networked Control Systems*, volume 406 of *Lecture Notes in Control and Information Sciences*. Springer, London.

Li, X. and Gao, H. (2011). A new model transformation of discrete-time systems with time-varying delay and its application to stability analysis. *IEEE Transactions on Automatic Control*, 56(9), 2172–2178.

Liu, G.P. (2010). Predictive controller design of networked systems with communication delays and data loss. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 57(6), 481–485.

Liu, K., Fridman, E., and Hetel, L. (2012). Network-based control via a novel analysis of hybrid systems with time-varying delays. In *51st IEEE Conference on Decision and Control (CDC)*, 3886–3891.

Liu, K. and Fridman, E. (2012). Networked-based stabilization via discontinuous lyapunov functionals. *International Journal of Robust and Nonlinear Control*, 22(4), 420–436.

Ludwiger, J., Steinberger, M., and Horn, M. (2019). Spatially distributed networked sliding mode control. *IEEE Control Systems Letters*, 3(4), 972–977.

Ludwiger, J., Steinberger, M., Horn, M., Kubin, G., and Ferrara, A. (2018). Discrete time sliding mode control strategies for buffered networked systems. In *57th Annual Conference on Decision and Control (CDC)*, 6735–6740.

Mu, J., Liu, G., and Rees, D. (2005). Design of robust networked predictive control systems. In *ACC: Proceedings of the 2005 American Control Conference, Vols 1-7*, 638–643.

Park, P., Ergen, S.C., Fischione, C., Lu, C., and Johansson, K.H. (2018). Wireless Network Design for Control Systems: A Survey. *IEEE Communications Surveys and Tutorials*, 20(2), 978–1013.

Repele, L., Muradore, R., Quaglia, D., and Fiorini, P. (2014). Improving Performance of Networked Control Systems by Using Adaptive Buffering. *IEEE Transactions on Industrial Electronics*, 61(9), 4847–4856.

Xia, Y., Zhu, Z., Li, C., Yang, H., and Zhu, Q. (2010). Robust adaptive sliding mode control for uncertain discrete-time systems with time delay. *Journal of the Franklin Institute - Engineering and Applied Mathematics*, 347(1), 339–357.

Zhang, D., Shi, P., Wang, Q.G., and Yu, L. (2017). Analysis and synthesis of networked control systems: A survey of recent advances and challenges. *ISA Transactions*, 66, 376–392.

Zhang, F. and Yeddanapudi, M. (2012). Modeling and simulation of time-varying delays. In *Proceedings of the 2012 Symposium on Theory of Modeling and Simulation*.