A Nonlinear State Space Model of Network Transmissions in a Network Control System

Octavian Stefan,* Alexandru Codrean,*
Toma-Leonida Dragomir*

** Automation and Applied Informatics Department, "Politehnica" University of Timisoara, Timisoara, Romania (e-mails:{octavian.stefan, alexandru.codrean}@aut.upt.ro, tldragomir@yahoo.com).*

Abstract: The latest advances in digital communication network technologies make network control systems suitable for a broad range of telecontrol applications. In the design and analysis of a network control system it is important to capture the dynamics of network transmissions under different scenarios. In this context, the current study proposes a new mathematical model for network transmissions with an additional packet handling strategy at the receiver level. The model takes into account time-varying delays and packet loss that occur during network transmissions. Additionally, the model is described through an algorithm used in simulations. The simulation results support the validity of the proposed model.

Keywords: network control system, network transmission model, time-varying delay, packet loss.

1. INTRODUCTION

The increase in data processing power and the decrease of equipment costs determined an extraordinary expansion of computer networks in the last years. Low cost, high fault tolerance and reliability of today's networks opened a new perspective in the field of control structure design. The idea of using computer networks for control systems is receiving more and more attention from the scientific community (Zampieri et al. (2008), Hespanha et al. (2007), Colom (2002)).

Although it has a lot of advantages, a network control system has also some shortcomings, induced by the network component, like time varying delays and data loss that need to be overcome before it can be largely implemented in industrial applications.

Extensive research has been done in order to overcome the problems introduced by the network in a network control system and some solutions have been proposed (Tipsuwan et al. (2003)). But, before such solutions can be implemented and used in real life applications they have to be tested in a controlled environment. The simulation of system behaviour under predefined conditions is a method largely used by engineers for testing a design pattern. In order to test a network control system through simulations both in the design stage and in the analysis stage, in the context of real network complexity, a practical approach would be to model the network system from the signal processing point of view. In other words, of interest is the way in which an input signal is deformed when transmitted through a real network.

The origin of this study is based on the observed lack, in specialized literature, of models that emulate, in terms of input-output signal alterations, the behaviour of a real network spanning from local to wide intercontinental areal. The paper highlights the particularities of network transmissions from the control point of view (Section 2), presents a mathematical model for the description of the signal transmission process through a network and an algorithm to implement the model (Section 3). Next, some simulation results are described as a case study (Section 4) and finally some conclusions are drawn (Section 5).

2. NETWORK TRANSMISSIONS

By far, the most used networks today are those based on the TCP/IP model, making them a good candidate for usage in network control systems. Control signals are affected differently when transferred through a TCP/IP network depending on the network hardware components and also on the software protocols used for data transmissions. Data can be transmitted over the network using connectionless or connection oriented protocols (Tanenbaum et al. (2010)). Both options can be used for data transmissions in a network control system, but most of the time, because of the imposed sample time restrictions, connectionless protocols are the only valid choice. In this context, the paper will only refer to TCP/IP networks using UDP as a transport protocol, regardless of topology and areal coverage.

Network transmissions can be modelled from a systemic point of view as an input-output signal transformation. In case of TCP/IP networks, when using UDP as a transport protocol, because the dynamics of network transmissions involves time varying delays and sudden jumps due to packet loss, the signals that pass through the network can be delayed and deformed.

Hence, the network can be considered as a separable discreet-time pair of transfer elements in the control system – Fig. 1 – composed of a forward communication path (input u_c and output u_d) and a feedback communication path (input y_d and

output y_c). Control signals are transferred through the network using data packets. Although the adopted connectionless transmission is usually faster, it has the disadvantage of being unreliable: data packet integrity and availability is not guaranteed. These lead to a possible information loss additional to the network time varying delay and to the necessity to adopt, at receiver level, a handling strategy for the arrived packets.

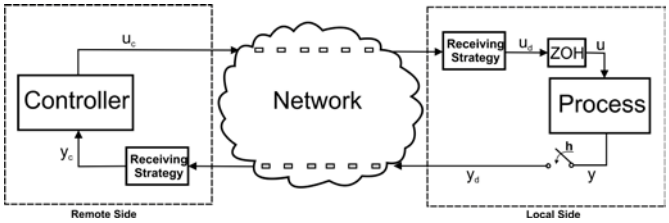

Fig. 1. Structure of a Network Control System

The conceptual and mathematical models associated with the network, from the point of view of how it interferes with the control system's dynamics, are determined by the description level of the signal transmission process and receiving processing operations. In this context, a conceptual model is further considered for the receiving strategy.

A constant sampling time operating regime with a sampling period h is assumed, so that, in both directions, the signal transmissions are synchronized with the beginning of every sampling period. Also, it is considered that, from the control point of view, when transmitting data through the network, one packet at every sample rate, three irregular situations are identified and handled as follows (Stefan et al. (2010)):

IS1: No data packets arrive during the current sampling period at the receiving end of a network path. In this case, the last valid information received is used as output of the network.

IS2: Two or more packets arrive during the current sampling period at the receiving end of a network path. In this case, only the latest information will be used as output of the network.

IS3: A data packet arrives during the current sampling period at the receiving end of a network path after a subsequent packet arrived and it had been already used. In such case, the packet is discarded and the latest valid information is used as output of the network block.

The handling manner flowchart of these irregular situations is synthesized by the logic diagram in Fig. 2. The algorithm associated to the diagram is executed at the beginning of each sampling period.

The development of a model for the network element which includes all the aspects mentioned above regarding data transmission would be required in most cases for the analysis and synthesis of a network control system. The modelling approaches found in the literature, such as discrete state space model of constant delay (Åström et al. (1997)), discrete state space model with time varying delay and packet loss (Li et al. (2011)), only partially address the three characteristics of data transmission mentioned (time varying delay, packet loss and irregular situations).

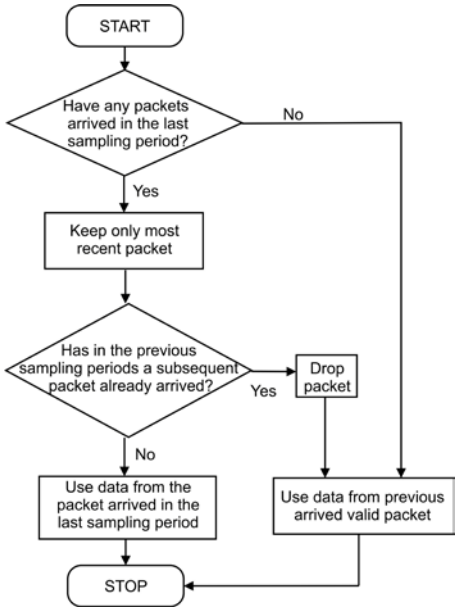

Fig. 2. Logic diagram for handling the irregular situations

Modelling such network dynamics can pose problems due to delay variation constraints imposed by the control system and due to the packet loss phenomena. Current simulation environments, like Matlab/Simulink or Maple, provide in their standard libraries only ideal time-varying delay blocks which do not permit additional constraints, like the ones specific to network transmissions. An interesting implementation of a real time network is presented in (Cervin et al. (2003)), a Matlab toolbox, but it is limited only to local area networks and to time delay induced by layer 1 and 2 protocols.

In this context, the next chapter presents a model that can be easily implemented in Matlab/Simulink and is able to emulate input-output signals dependencies of real network transmissions. The model takes into account all the above possible situations, simulating in a simplified manner the behaviour of a real life network.

3. NETWORK TRANSMISSIONS MODEL

Let u be the input data signal for a communication channel of the network system, y the output signal of the same channel, h the sampling period (the network system being considered as a discrete system) and k represents the index of discrete sample times. The time delay of a packet needed to pass through a communication channel, including packet processing time at network nodes, varies from a sampling instant to another sampling instant, therefore it is a time varying delay τ_k. Because in the considered discrete time processing systems data is used only at sample instants kh, τ_k is a multiple of h. Further on, the notation $y[k]$, $u[k]$, $\tau[k]$ may be used with the same meaning as $y(kh)$, $u(kh)$, τ_k.

In order to develop a model for network transmissions, a simple example will be chosen (Fig. 3), which includes all three types of situations that occur during data transmissions (varying delay, packet loss and irregular situations). Fig. 3 expresses the time moments at which data packets are generated by the sender and when they arrive at the receiver. During the time window of more consecutive sample moments beginning with *k-2*, besides each packet's delay, a packet is lost at moment *k+1* and the following irregular situations appear: IS1 at {*k-1*, *k+1*, *k+2*}, IS2 at {*k*, *k+6*} and IS3 at *k+4*. Based on the handling strategy from Section 2 the input sequence *u* should be transformed by the network system into the output sequence *y* as shown in Fig. 4 (receiver's operational level).

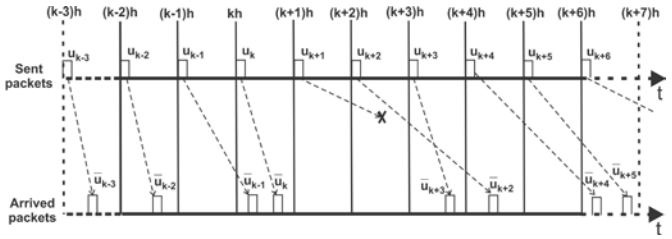

Fig. 3. Network transmission example – packet arrival sequence

In order to motivate the need of a new model to describe the input-output dependency $u \rightarrow y$, first, the classic time-varying delay model, already implemented in Matlab/Simulink library (*** (2011)), will be considered:

$$y(kh) = u(kh - \tau_k) \tag{1}$$

Using model (1) to generate the output sequence *y*[k] for the example from Fig. 3 the following result is obtained (absolute time):

$$\begin{cases} \tau_{k-2} = h \rightarrow & y[k-2] = u[k-3], \\ \tau_{k-1} = 2h \rightarrow & y[k-1] = u[k-3], \\ \tau_k = h \rightarrow & y[k] = u[k-1], \\ \tau_{k+1} = \infty \rightarrow & y[k+1] = u[-\infty], \\ \tau_{k+2} = 3h \rightarrow & y[k+2] = u[k-1], \\ \tau_{k+3} = h \rightarrow & y[k+3] = u[k+2], \\ \tau_{k+4} = 3h \rightarrow & y[k+4] = u[k+1], \\ \tau_{k+5} = 2h \rightarrow & y[k+5] = u[k+3]. \end{cases} \tag{2}$$

On the operational level (2) becomes (the time variable refers to the current sampling instance):

$$\begin{cases} y[t] = u[t-1], & for\ t = k-2 \\ y[t] = u[t-2], & for\ t = k-1 \\ y[t] = u[t-1], & for\ t = k \\ y[t] = u[-\infty], & for\ t = k+1 \\ y[t] = u[t-3], & for\ t = k+2 \\ y[t] = u[t-1], & for\ t = k+3 \\ y[t] = u[t-3], & for\ t = k+4 \\ y[t] = u[t-2], & for\ t = k+5 \end{cases} \tag{3}$$

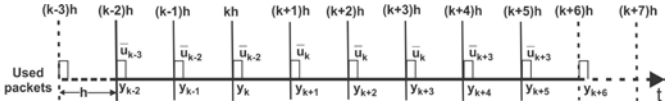

Fig. 4. Network transmission example – proposed packet handling strategy

These results are illustrated in Fig. 5. It can be noticed that these results are different from the ones shown in Fig. 4.

It can also be noticed that packet loss situations, as well as the situations when the packets are not used, cannot be captured by model (1). Such packets don't appear on the right side of the equations (3). It should be observed that by permitting the delay value to increase or decrease, the irregular situations cannot be addressed, so the principle of using the latest available packet at each sample time is not respected.

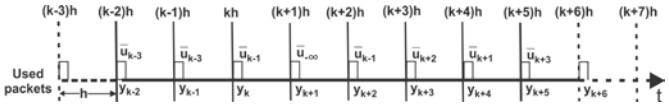

Fig. 5. Network transmission example – packet handling using (1)

As a different approach, considering that a discrete time delay is a finite dimensional system, a state space model could be addressed. Fig. 6 defines the Network Transmission Block (NTB), with its inputs and outputs. Here, *u* is the input data signal for the NTB and *y* is the output data signal. τ and *p* are two additional input signals referring to the time varying delay and to the packet loss flag. These two input signals can be generated according to certain patterns (e.g. probability distributions) or can be obtained directly from measurements on a real TCP/IP network. At a given sample instant *k*, the value of τ[k] is a multiple of the sampling period {1, 2, 3, ..., τ_{max}}. The delay of a lost packet is adopted as τ_{max}.

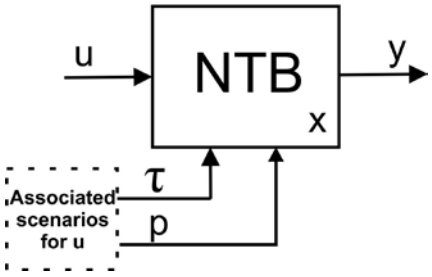

Fig. 6. Network Transmission Block

A number of *n* states of the NTB is represented through the vector variable $\boldsymbol{x}$, where *n* is given by the maximum varying delay allowed $n = \tau_{max}$. Data packets that arrive with superior delays will be considered as lost packets. The minimum time delay is limited to a sample period. Thus, a first component of the state vector has the form:

$$\mathbf{x}[k] = \begin{bmatrix} x_1[k] \\ \vdots \\ x_n[k] \end{bmatrix} \tag{4}$$

First, in order to build the model, the input is redefined as a *n*-dimensional vector:

$$\bar{u}[k]=\begin{bmatrix}0\\ \vdots\\ 0\\ u[k]\\ 0\\ \vdots\\ 0\end{bmatrix}\begin{matrix}\left.\begin{matrix}\\ \\ \\ \\ \end{matrix}\right\}\tau[k]\\ \left.\begin{matrix}\\ \\ \\ \end{matrix}\right\}n-\tau[k]\end{matrix} \tag{5}$$

where $\bar{u}[k]$ now actually depends on the u[k] and τ[k] signals. Defining $\bar{u}$ limits the possibility to express the value of zero for the input signal u[k], so another value has to be adopted as a convention for it. At any sample time the signal p[k] can take one of two values: 1 if the data packet has arrived or 0 if the packet is lost.

Based on the input defined in (5), the state vector $\boldsymbol{x}$ at the next sample time $k+1$ is obtained as:

$$\begin{aligned}\mathbf{x}[k+1]=&\left(p[k]\cdot\mathbf{f_n}(\tau[k])+(1-p[k])\cdot\mathbf{I}_n\right)\cdot\mathbf{I}_{sn}\cdot\mathbf{x}[k]\\&+p[k]\cdot\bar{\mathbf{u}}[k]\end{aligned} \tag{6}$$

with the function $\mathbf{f_n}:\mathrm{N}\to\mathrm{N}^{nxn}$ defined as:

$$\mathbf{f_n}(\tau[k])=diag(\sigma(\tau[k]-2),..,\sigma(\tau[k]-n-1)) \tag{7}$$

Here $\boldsymbol{I}_n$ is the unitary matrix, while the unitary step function σ and the $n\times n$ matrix $\boldsymbol{I}_{sn}$ are defined by:

$$\sigma(x)=\begin{cases}0, x<0\\ 1, x\geq 0\end{cases},\ \mathbf{I}_{sn}=\begin{bmatrix}0&1&0&\cdots&\cdots&0\\ \vdots&\ddots&\ddots&\ddots&&\vdots\\ \vdots&&\ddots&\ddots&\ddots&\vdots\\ \vdots&&&\ddots&\ddots&0\\ \vdots&&&&\ddots&1\\ 0&\cdots&\cdots&\cdots&\cdots&0\end{bmatrix} \tag{8}$$

The equations (6)-(8) calculate the state vector for the next sampling period. First, $\boldsymbol{I}_{sn}$ shifts the state vector upwards with one position (the value from x_1 is eliminated because it has been already used, while zero is inserted at position x_n). Second, $\boldsymbol{f}_n$ sets to zero all the state values at positions beyond τ[k]-*1* (the values are no longer needed because newer data has arrived). If a packet is lost (p[k]=0) the state vector is modified only through the shift operation ($\boldsymbol{I}_{sn}$). Third, the new value of the input signal u modifies the state vector through $\bar{u}$.

Next, an additional state x_a is defined as:

$$\begin{aligned}x_a[k+1]=&\bar{u}_1[k]+\sigma(\tau[k]-2)\cdot x_2[k]\\&+(1-g(x_2[k]))\cdot(1-g(\bar{u}_1[k]))\cdot x_a[k]\end{aligned} \tag{9}$$

where:

$$g(x)=\begin{cases}0, x=0\\ 1, x\neq 0\end{cases} \tag{10}$$

The x_a state is used in order to maintain at the output the value from the last valid arrived packet. If both $\bar{u}_1$ and x_2 are zero then the same value of the state x_a is maintained for the next sample period by function g. If $\bar{u}_1$ is nonzero, which implies τ[k]=*1*, the value of x_2 is invalidated through the σ function.

The output y at each sample time is given by the value of the state x_a:

$$y[k]=x_a[k] \tag{11}$$

Equations (4)-(11) can be lumped together into a state space model by defining a global state vector

$$\bar{\mathbf{x}}[k]=\begin{bmatrix}\mathbf{x}^T[k] & x_a[k]\end{bmatrix}^T \tag{12}$$

the matrix

$$\mathbf{A}[k]=p[k]\cdot\left(\mathbf{f}_n(\tau[k])-\mathbf{I}_n\right)\cdot\mathbf{I}_{sn}+\mathbf{I}_{sn} \tag{13}$$

a function

$$\tilde{g}[k]=(1-g(x_2[k]))\cdot(1-g(\bar{u}_1[k])) \tag{14}$$

and the additional $1\times n$ vectors:

$$\begin{aligned}\mathbf{v}^T&=\begin{bmatrix}0 & \sigma(\tau[k]-2) & 0 & \cdots & 0\end{bmatrix},\\ \mathbf{w}^T&=\begin{bmatrix}1 & 0 & 0 & \cdots & 0\end{bmatrix},\\ \mathbf{c}^T&=\begin{bmatrix}0 & 0 & \cdots & 0 & 1\end{bmatrix}.\end{aligned} \tag{15}$$

The following state space model is obtained:

$$\begin{cases}\bar{\mathbf{x}}[k+1]=\begin{bmatrix}\mathbf{A}[k] & \mathbf{0}_{n\times 1}\\ \mathbf{v}^T & \tilde{g}[k]\end{bmatrix}\cdot\bar{\mathbf{x}}[k]+\begin{bmatrix}\mathbf{I}_n\\ \mathbf{w}^T\end{bmatrix}\cdot p[k]\cdot\bar{u}[k]\\ y[k]=\mathbf{c}^T\bar{\mathbf{x}}[k]\end{cases} \tag{16}$$

Next, using the proposed model (16) to generate the output sequence y[k] for the example from Fig. 3 the following results are obtained:

$$\begin{cases}\tau_{k-2}=h\ \to & y[k-2]=u[k-3],\\ \tau_{k-1}=2h\to & y[k-1]=u[k-2],\\ \tau_k=h\ \to & y[k]\ =u[k-2],\\ \tau_{k+1}=\tau_{\max}\to & y[k+1]\ =\ u[k],\\ \tau_{k+2}=3h\to & y[k+2]=u[k],\\ \tau_{k+3}=h\to & y[k+3]=u[k],\\ \tau_{k+4}=3h\to & y[k+4]=u[k+3],\\ \tau_{k+5}=2h\to & y[k+5]=u[k+3].\end{cases} \tag{17}$$

The results from (17) are the same with those presented in Fig. 4 and they are in agreement with the proposed packet handling methodology described in Section 2.

Finally, as seen in the example above, model (16) manages to completely characterize network transmissions. The model behaves as a buffer which is continuously shifted, filled and emptied at each sample time, according to the time varying delay, the handling of the irregular situations and the packet loss flag. The shift corresponds to the passing of a new sample instance, the filling corresponds to a new data packet arrival which will be released according to the scheduled delay, and the emptying corresponds to packets that arrive too late and are dropped because newer data has already arrived (irregular situation).

From the point of view of the implementation, the model (16) for the NTB can be described through the following algorithm:

Initialization:

- *define maximum allowed delay* τ_{max}
- *initialize buffer x*
- *initialize* y_0

Repeat at each sample time k:

Step1: read inputs u_k, τ_k *and* p_k

Step2: if $\tau_k > \tau_{max}$ *or* $p_k=0$ *jump at Step5*

Step3: rewrite value at position τ_k+1 *in buffer x with* u_k

Step4: delete all buffer values from the positions to the right of position τ_k+1

Step5: if the value from the first position of buffer x is non zero write it to output y_k, *else* y_k *becomes* y_{k-1}

Step6: shift buffer x with one position to the left

The following section will present simulation results obtained with an implementation of this algorithm in Matlab/Simulink.

4. SIMULATIONS

The system from Fig. 6 will be considered as case study for testing the proposed algorithm. The NTB is considered to be a discrete time system with sampling period *h* of 1 ms. Three scenarios will be considered.

First, the scenario corresponding to the example from Fig. 3 will be addressed. The first sample instance *(k-3)h* of the considered time window is adopted as 0.010 s. The values for the input *u* are the ones from Fig. 7. Time delay τ varies as in Fig. 8, where at moment 0.014 s τ gets infinitely large due to a packet loss ($p=0$). The output *y* obtained with the model (1) implemented as the standard time varying delay Simulink block is illustrated in Fig. 9. It can be observed that the time variance pattern of the input signal is significantly altered. As opposed to the result from Fig. 9, by using the proposed algorithm in this paper, the output signal manages to reproduce the input signal pattern (Fig. 10). It can be noticed that the simulation results are in compliance with the ones presented in Section 3.

In the following two scenarios the inputs τ and *p* are generated as uniformly distributed pseudorandom numbers in a given range.

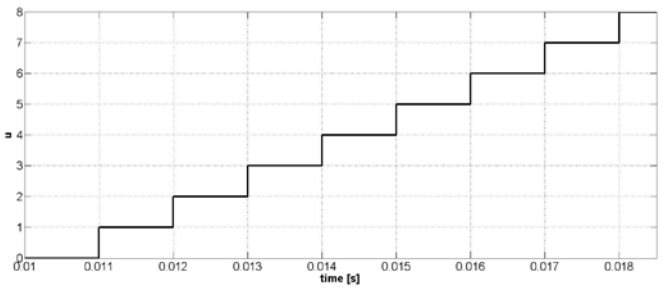

Fig. 7. Input signal u

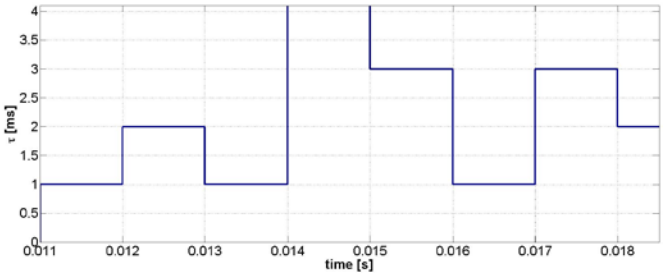

Fig. 8. Time varying delay signal τ

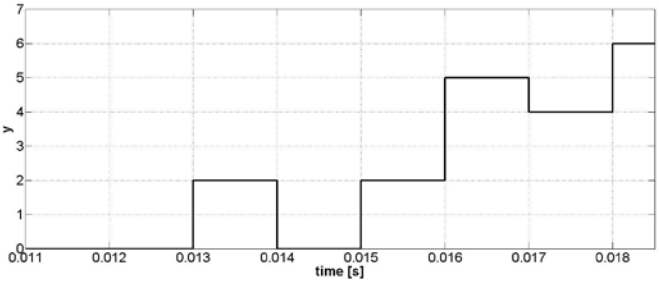

Fig. 9. Output signal y for time varying delay model (1)

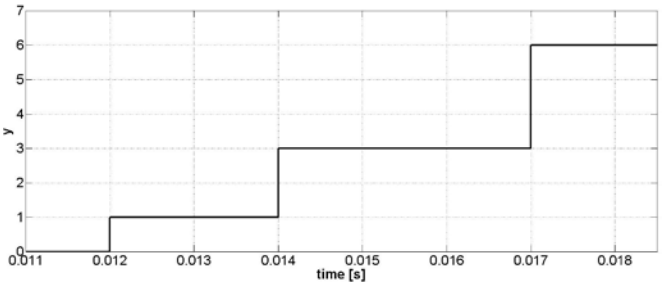

Fig. 10. Output signal y for the proposed NTB model (16)

The second scenario corresponds to a discrete ramp input signal *u*. Fig. 11 shows the results obtained on a two second window, while Fig. 12 shows a zoomed version on a 0.2 second window. On a large time scale (2 s), it appears that the output *y* maintains a constant time shift in respect to the input *u* due to the mean value of the delay. However, by taking a look at a smaller time scale (0.2 s), it can be observed that actually the signal is significantly deformed because of packets which arrive too late or which are lost.

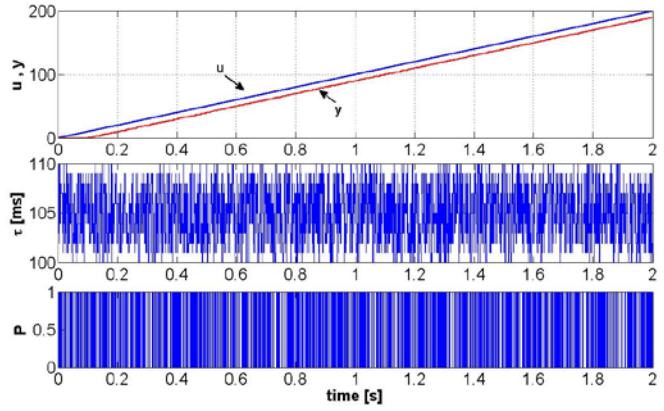

Fig. 11. Simulations results for the NTB with a discrete ramp input on a 2 seconds window

For the third scenario a discrete sinusoidal input of 10 rad/s was considered – Fig. 13. Due to the significantly increased standard deviation of the delay, signal deformations can be observed here even on a large time scale of 2 s.

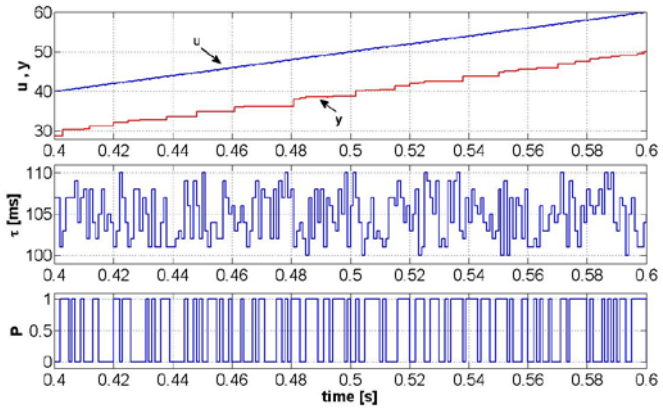

Fig. 12. Simulations results for the NTB with a discrete ramp input on a 0.2 seconds window

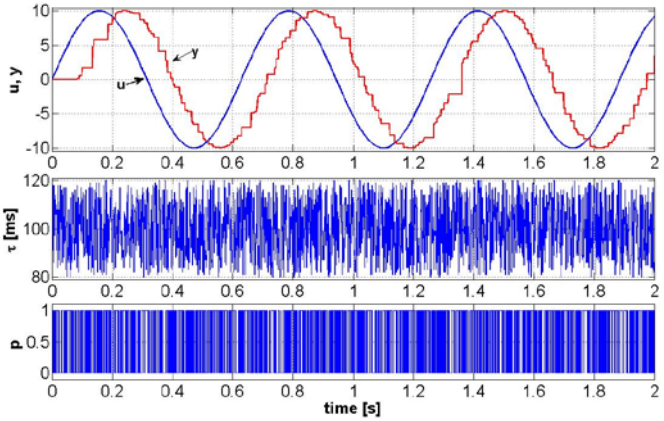

Fig. 13. Simulations results for the NTB with a discrete sinusoidal input on a 2 seconds window

Ultimately, it can be concluded that NTB implementation manages to reproduce patterns which, from the point of view of signal transmissions, are very close to real TCP/IP network transmissions complemented with an additional handling strategy.

5. CONCLUSIONS

The network transmission model presented in this paper has been developed in order to fill in the gap in literature of a model that correctly describes the network element from a Network Control System. The study assumed that the sampling instances of the network nodes are synchronized.

The obtained mathematical model is a nonlinear discrete time varying model, which has been brought to a state space form and has also been described through an algorithm.

The state space model can be integrated in different types of control system models, and can be used for estimating the performance and analysing the stability of the systems. The stability analysis of Network Control Systems has received considerable interest in the scientific community (e.g. Zhang et al. (2001), Hespanha et al. (2007), Li et al. (2011),), where models such as the one presented in this paper would be needed.

The algorithm can be easily implemented as a Simulink block which takes as input the signal that has to be transmitted and the transmission scenarios defined through sequences of values for the time delay and packet loss flag. The Simulink block would then be helpful for simulating Network Control Systems under different network scenarios.

As future work, two main directions may be addressed. On one hand, the nonlinear system described through the proposed model should be brought to the form of a jump linear system, for which several analysis tools have been developed (Hespanha et al. (2007)). On the other hand, a straightforward extension would imply the development of a stochastic model that would correlate different network parameters with the time delay and packet loss flag values.

REFERENCES

Åström, K. J., Wittenmark, B. (1997). *Computer-Controlled Systems*. Prentice Hall, pp. 38-41.

Cervin, A., Henriksson, D., Lincoln, B., Eker, J., Årzén, K.-E. (2003). How Does Control Timing Affect Performance? Analysis and Simulation of Timing Using Jitterbug and TrueTime. *IEEE Control Systems Magazine*, vol. 23, no. 3, pp. 16-30.

Colom, P.M. (2002). Analysis and Design of Real-Time Control Systems with Varying Control Timing Constraints. *Ph.D. Thesis*. Polytechnic University of Catalonia, Barcelona.

Hespanha, J. P., Naghshtabrizi, P., and Xu, Y. (2007). A Survey of Recent Results in Networked Control Systems. *Proc. IEEE*, vol. 95, no. 1, pp. 138–162.

Li, H., Sun, Z., Chow, M.-Y., and Sun, F. (2011). Gain-Scheduling-Based State Feedback Integral Control for Networked Control Systems. *IEEE Trans. Ind. Electron.*, vol. 58, no. 6, pp. 2465–2472.

Stefan, O., Dragomir, T.L., Codrean, A., Silea I. (2010). Issues of Identifying, Estimating and Using Delay Times in Telecontrol Systems Based on TCP/IP Networks. *Proceedings of the 2nd IFAC Symposium on Telematics Applications*, Timisoara, Romania, pp. 143-148.

Tanenbaum, A. S., Wetherall, D.J. (2010). *Computer Network*, 5th Edition. Prentice Hall.

Tipsuwan, Y., Chow, M.-Y. (2003). Control methodologies in networked control systems. *Control Engineering Practice*, vol. 11, pp. 1099–1111.

Zampieri, S. (2008). Trends in Networked Control Systems, *Proceedings of the 17th IFAC World Congress*, Seoul, Korea, pp. 2886-2894.

Zhang, W., Branicky, M.S., Phillips, S.M. (2001). Stability of Networked Control Systems. *IEEE Control Systems Magazine*, vol. 21, no. 1, pp. 84-99.

*** (2011). Simulink 7 User's Guide. The MathWorks Inc.