ERROR MODELS SIMULATIONS IN TRANSMISSION CHANNELS USING NETWORK SIMULATOR ENVIRONMENT

Ján VALISKA, Branislav HRUŠOVSKÝ, Stanislav MARCHEVSKÝ , Slavomír PILLÁR
Department of Electronic and Multimedia Communications, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic, e-mail: jan.valiska@tuke.sk, branislav.hrusovsky@tuke.sk, stanislav.marchevsky@tuke.sk

ABSTRACT

This paper discusses transmission channels evaluation, error models and their subsequent simulations for the transmission of video content. To create a network topology and error channel models, NS-2 environment is used. In this paper, we simulated two basic error channel models, namely Gilbert-Elliot and Binary Symmetric channel model. Network traffic in topology is created by NT-MyEvalvid2 toolkit, which also serves to evaluate the data transferred and their subsequent visualization. In paper, we compare the measure of packet loss when using these error models with different input parameters and their impact on the quality of video transmission parameter evaluated by PSNR.

Keywords: *NS-2, MyEvalvid-NT, channel model, multicast, unicast*

1. INTRODUCTION

NS-2 is an open-source event-driven simulator designed specifically for research in computer communication networks. Since its inception in 1989, NS2 has continuously gained tremendous interest from industry, academia, and government. Having been under constant investigation and enhancement for years, NS-2 now contains modules for numerous network components such as routing, transport layer protocol, application, etc. To investigate network performance, researchers can simply use an easy-to-use scripting language to configure a network, and observe results generated by NS-2. Undoubtedly, NS-2 has become the most widely used open source network simulator, and one of the most widely used network simulators [6].

Generally, network simulators facilitate the study of network architectures such as IP QoS implementations. NS-2 is one of them. It is widely used for IP network studies. One of the areas that can be enhanced in NS-2 are the post-processing capabilities [15]. Network simulation scripts in NS-2 are used to create the network scenarios and upon the completion of the simulation, trace files that capture events occurring in the network are produced, as is shown in Fig. 1. The trace files would capture information that could be used in performance study, e.g. the amount of packets transferred from source to destination, the delay in packets, packet loss etc. However, the trace file is just a block of ASCII data in a file and quite cumbersome to access using some form of post-processing technique [16].

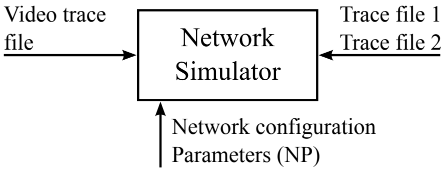

Fig. 1 Network Simulator

2. SOFTWARE ARCHITECTURE

NS-2 is an object oriented simulator written in C++ that uses OTcl (an object oriented version of Tcl) as the command and configuration interface. Thus, the entire software hierarchy is written in C++, with OTcl used as a front end. C++ code serves as a backbone for the whole simulation process. NS-2 use two languages because simulator has two different kinds of things it needs to do. On the one hand, detailed simulations of protocols require a systems programming language which can efficiently manipulate bytes, packet headers, and implement algorithms that run over large data sets [14]. For these tasks, run-time speed is important and turn-around time (run simulation, find bug, fix bug, recompile, re-run) is less important. On the other hand, a large part of network research involves slightly varying parameters or configurations, or quickly exploring a number of scenarios. In these cases, iteration time (change the model and re-run) is more important. Since configuration runs once (at the beginning of the simulation), runtime of this part of the task is less important [2]. One drawback of combining two languages is that debugging becomes more complicated than with one language alone [10].

NS-2 gluess C++ and OTcl together by providing support for split objects, that are objects that appear in both languages. When the user creates a new simulator object through the interpreter, the object is first instantiated in the *interpreted* hierarchy (OTcl). Then a corresponding object in the *compiled* hierarchy (C++) is created. Attributes of these objects can be bound together. When one is changed, the other is automatically updated. Likewise, compiled methods can be called from OTcl and vice versa [10].

The basic simulator objects that NS-2 includes are *Nodes, Links, Agents and Applications.* Nodes and Links define the network topology. Agents represent endpoints where network-layer packets are constructed or consumed and are commonly used to implement protocols at various levels. Applications are sources that send and receive data. The objects are connected to each other in a layered fashion as shows Fig. 2 [10].

ISSN 1335-8243 (print) © 2012 FEI TUKE
www.aei.tuke.sk

ISSN 1338-3957 (online)
www.versita.com/aei

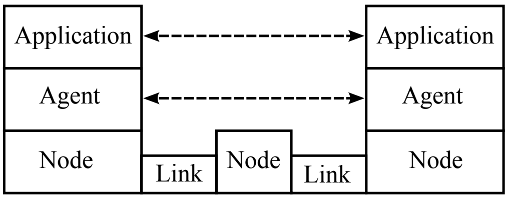

Fig. 2 The basic NS-2 objects and their interconnections

When a agent is *attached* to a node, it is given a unique address consisting of the node id and a port number within that node. Each node can have multiple agents attached to it. Multiplexing/demultiplexing between those are handled automatically within the Node object. Before the simulation starts, agents must be *connected* to each other in pairs [12]. This will not trigger a protocol level connection. It will only set the source and destination address of the packets to match the two endpoints. This is similar to the connection used in UDP.

There are two major types of agents in NS-2: one-way and two-way agents. The one-way agent consists of one sender and one corresponding receiver. Data can only be sent in one direction and often connection establishment/teardown is not performed. A one-way agent simulates the basic protocol behavior, in most cases how data is transferred, without the added complexity of a full protocol implementation [13]. The result is a simple implementation with reduced development time. If the service of the one-way agent is not enough for the simulation scenario, two-way agent can be used instead. The two-way agent is both a sender and receiver, supports bi-directional data transfers, and could be very well a real-world counterpart. Currently, NS-2 offers support for various versions of TCP and for TRFC [10].

3. PACKET LOSS MODELING

Communication channel can be classified into two categories: memoryless channels and channels with memory [8]. Space or satellite channels are a few examples of memoryless channels. On these channels, a received signal at a given time interval depends only on the transmitted signal in that time interval, but not on previous signal transmissions. As a result, errors appear randomly on memoryless channels, and therefore these channels are often referred as random-error channels. Many real communication channels, however, have memory. Errors on these channels tend to occur in clusters or bursts [4]. This bursty behaviour is common in mobile communication channels, telephone channels, and magnetic or optical recording systems. In simulation section, we focused on two simple channel models, the binary symmetric channel (BSC) model representing a memoryless channel, and the Gilbert-Elliott channel model representing a channel with memory.

In order to evaluate quality of transmission, lets have random variable X. If packet is not lost, then $X = 0$, otherwise $X = k$ for k lost packets. After that, we can build loss model with infinite number of states (m is infinite value see Fig. 3). Such model gives us opportunity to model packet loss probabilities in dependence on burst lengths (several consecutively lost packets). For each additional lost packet, which adds to the length of a loss burst, a state transition takes place. If packet is correctly received, then the state returns to $X = 0$ [17].

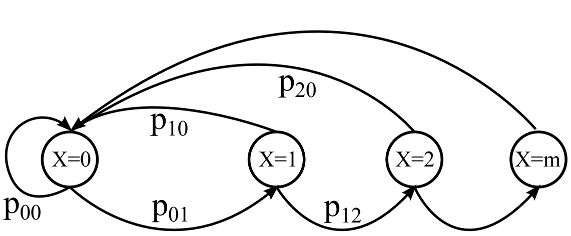

Fig. 3 Loss model with infinite number of states ($m \to \infty$)

State probability for system $k > 0$ with is $P(X \geq k)$. For finite number of received packets a, state probabilities of the system with $k > 0$ can be approximated with cumulative loss rate [17].

$$p_{L,cum}(k) = \sum_{n=k}^{\infty} p_{L,n} \tag{1}$$

Cumulative loss rate for $k = 0$, thus for no loss case, can be computed using the following equation:

$$\begin{aligned} p_{L,cum}(k=0) &= 1 - \sum_{n=k}^{\infty} p_{L,cum}(k) = 1 - \sum_{k=1}^{\infty} \frac{\sum_{n=k}^{\infty} o_n}{a} = \\ &= 1 - \sum_{k=1}^{\infty} \frac{k o_k}{a} = 1 - p_L \end{aligned} \tag{2}$$

where $o_k(o_n)$ is occurrence of loss with length $k(n)$.

3.1. Loss model with limited number of states

In order to model packet losses during video or audio transmissions, it is sufficient to use models with limited number of states due to strict requirements, which have to be fulfilled, i.e. it is useless to transmit multimedia through a network with long consecutive packet losses.

Three the most commonly used models with limited number of states are: the k-th order Markov chain model, the 2-state Markov chain model and the Bernoulli loss model.

3.11. $K - th$ order Markov chain model

$K - th$ order Markov chain model has similar performance measures as model with infinite number of states, however state probability for finite state m is added and also probability for transition from m to m is added. Thus for cumulative loss holds the following term [17]

$$p_{L,cum}(k) = \sum_{n=k}^{\infty} \frac{o_k}{a} \tag{3}$$

For burst loss $k = m$ holds:

$$p_{L,m}(k) = \sum_{n=m}^{\infty} \frac{(n-m+1)o_n}{a-m+1} \tag{4}$$

and for conditional loss $k = m$ holds:

$$p_{L,m}(k) = \sum_{n=m}^{\infty} \frac{(n-m+)o_n}{d-m} \tag{5}$$

where: d - number of dropped packets:

$$d = \sum_{k=1}^{\infty} k o_k \tag{6}$$

K-th order Markov chain model is also known as Extended Gilbert model [5].

3.12. Gilbert model

Packet loss measurements on the Internet have shown that the probability of loss episodes of length k decreases approximately geometrically with increase of k [9]. Thus it is possible to use simpler packet loss model, e.g. Gilbert model.

Special case of $k-th$ order Markov chain model is Gilbert model with , see Fig. 4.

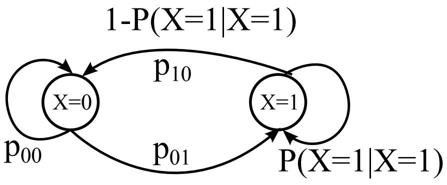

Fig. 4 Gilbert model

In this model, 0 represents state with no packet loss and on the other hand 1 represents the state of packet being lost.

The matrix for transition probabilities and for state probabilities can be expressed in form:

$$\begin{bmatrix} 1-p_{01} & p_{10} \\ p_{01} & 1-p_{10} \end{bmatrix} \begin{bmatrix} P(X=0) \\ P(X=1) \end{bmatrix} = \begin{bmatrix} P(X=0) \\ P(X=1) \end{bmatrix} \tag{7}$$

For unconditional probability $P(X = 1)$ holds the following equation:

$$P(X=1) = \frac{p_{01}}{p_{01}+p_{10}} \tag{8}$$

If previous packet is lost, then for conditional probability of having loss holds:

$$P(X=1|X=1) = 1p_{10} \tag{9}$$

Gilbert model memorizes only the previous state, thus the probability, that the next packet will be lost is dependent only on the previous state.

Transition probabilities p_{01} and p_{10} can be expressed with the following equations:

$$p_{01} = P(X=1|X=0) = \sum_{k=1}^{\infty} \frac{o_k}{a} \tag{10}$$

$$1-p_{10} = P(X=1|X=1) = \frac{\sum_{k=1}^{\infty}(k-1)o_k}{d-1} \tag{11}$$

The probability of having a lost episode with length k [9]

$$p_k = (a-p_{10})^{k-1} p_{10} \tag{12}$$

3.13. Bernoulli model

The simplest way, how to model packet losses, is using Bernoulli model. In this model, the probability, that the packet is lost or correctly received is independent on all other values.

Bernoulli model can be characterized by a single parameter r, which describes the probability of packet being lost [12].

$$\hat{r} = \frac{n_i}{n} \tag{13}$$

where: n_i - number of times, when packet loss has occurred, n - total number of packets.

3.2. The binary symmetric channel model

The binary symmetric channel model (BSC) is a simple model for discrete memoryless channels (DMC). A discrete memoryless channel of M-ary inputs and Q-ary outputs is defined by the transition probabilities $P(j/i)$,where $0 \leq i \leq (M1)$ a $0 \leq j \leq (Q1)$.

In the BSC model, $M = 2$ and $Q = 2$. The model is completely defined by a channel error rate or crossover probability ε as shown in Fig. 5 [8].

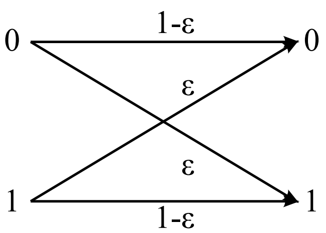

Fig. 5 The binary symmetric channel model

4. PERFORMANCE PARAMETER

The performance parameters that can be obtained through the NS2 trace analyzer are as follows, which are the main parameters of interest for IP QoS [16]:

- Throughput
- Packet Loss
- End to End Delay
- Jitter

4.1. Throughput

Throughput is the rate at which a network sends or receives data. It is a good channel capacity of network connections and rated in terms bits per second (bit/s).

$$T_p = \frac{P_a}{P_f} \tag{14}$$

where P_a - received packets, P_f - amount of forwarded packets over certain time interval.

4.2. Packet loss

Packet loss is where network traffic fails to reach its destination in a timely manner. Most commonly packets get dropped before the destination can be reached.

$$P_d = P_s - P_a \tag{15}$$

where P_s - amount of sent packets, P_a - amount of received packets.

4.3. Jitter

Jitter is a fluctuation of end-to-end delay from one packet to the next packet of connection flow.

$$J = |D_{j+1} - D_j| \tag{16}$$

where D_{i+1} - delay of $i_{th}+1$ packet, D_i - delay of i_{th} packet.

4.4. End to end delay

End-to-end delay refers to the time taken for a packet to be transmitted across a network from source to destination.

$$D = |T_d - T_s| \tag{17}$$

where T_d - packet receive time at the destination, T_s - packet send time at source node.

5. EVALVID - VIDEO QUALITY EVALUATION TOOL-SET

EvalVid is a framework and tool-set for evaluation of the quality of video transmitted over a real or simulated communication network. It is targeted for researchers who want to evaluate their network designs or setups in terms of user perceived video quality [7].

Besides measuring QoS parameters of the underlying network, like loss rates, delays, and jitter, standard video quality metrics like PSNR and SSIM and a subjective video quality evaluation metric of the received video are provided. Currently, H.264 MPEG-4 and H.263 are supported. AAC support is also included, though the perceptual quality evaluation has to be done by external tools implementing dedicated metrics like PESQ or PEAQ [7].

5.1. MyEvalvid-NT

A novel realistic simulation tool-set known as MyEvalvid-NT for evaluating video delivery over simulated networks was created. This tool-set integrates Evalvid and NS2 [3]. The tool-set can adopt traffic trace file from the output of encoding raw YUV video or from Internet traffic traces.

5.11. System characteristics

Fig. 6 shows the system architecture of MyEvalvid-NT [3]. In this framework, we can use two different input sources: 1, publicly available traffic trace [11] and 2, the trace from encoding raw YUV video. Fig. 6 shows the QoS assessment framework for video traffic enabled by the new tool-set that combines EvalVid and NS2. As shown in Fig. 6, three connecting interfaces, namely MyTrafficTrace2, MyUDP, and MyEvalvid Sink2, are implemented in NS2 environment [3].

These three interfaces are described as follows:

MyTrafficTrace2: It is employed to extract the frame type and the frame size of the video trace file. Then it fragments the video frames into smaller segments and sends these segments to the lower UDP layer at the appropriate time according to the user settings specified in the simulation script-file.

MyUDP: Essentially, MyUDP is an extension of the UDP agent in NS2. This new agent allows users to specify the output file name of the sender trace file and it records the timestamp of each transmitted packet, the packet id, and the packet payload size.

MyEvalvid Sink2: is the receiving agent for the fragmented video frame packets sent by MyUDP. When receiving a video packet, this agent also records the timestamp, packet ID, and payload size of each received packet in the user specified file [3].

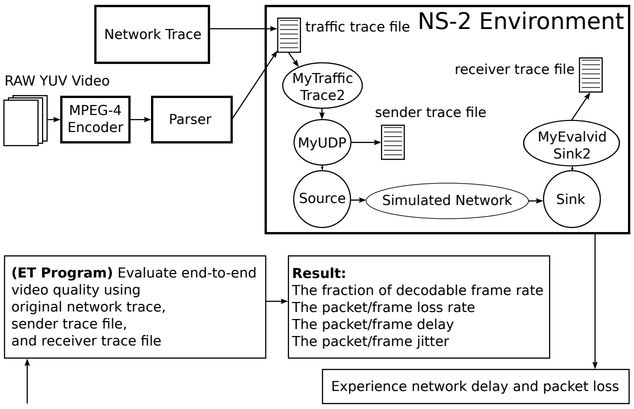
Network Trace
traffic trace file
NS-2 Environment
RAW YUV Video
MyTraffic Trace2
receiver trace file
MPEG-4 Encoder
Parser
sender trace file
MyUDP
MyEvalvid Sink2
Source
Simulated Network
Sink
(ET Program) Evaluate end-to-end video quality using original network trace, sender trace file, and receiver trace file
Result: The fraction of decodable frame rate The packet/frame loss rate The packet/frame delay The packet/frame jitter
Experience network delay and packet loss

Fig. 6 The system architecture of MyEvalvid-NT

6. SIMULATIONS

In General, PSNR is the metric used to measure the perceived quality at the end user. However, PSNR takes a long time to compare each pixel in the original frame and distorted frame to get PSNR value. Alternatively we can use decodable frame rate (Q) [3] to measure video quality at the receiver, which takes less time compared to PSNR. Therefore, in this paper, we use decodable frame rate as one of our main video evaluation metric. Through simulations, we also reveal that decodable frame rate can evaluate the perceived quality well by an end user.

6.1. Used topology

The simulation topology used in this paper is shown in Fig. 7. The video server transmits video streams over the Internet and wireless links to the video receiver. The maximum transmission packet size is 1024 bytes. The link between the wireless access point and the video receiver is IEEE 802.1 lb 11Mbps. For simplicity, we assume that the link between the video server and the wireless access point has a 10Mbps bandwidth and 10 ms latency [3].

In this simulation, we make three assumptions as follows:

1. No packet loss occurs in the wired segment of the video delivered path.
2. One packet loss will result in frame undecodable
3. No concealment strategy is used when decoding

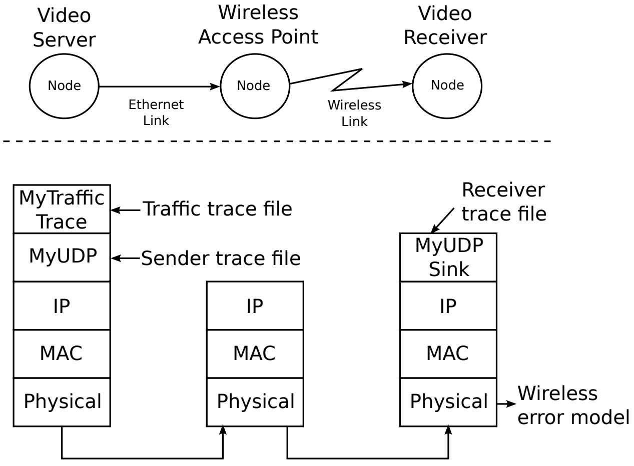

Fig. 7 Simulation topology

6.2. Simulation results

The several simulations were done due to evaluation of used scenario. The results were processed using graphs and tables. In each simulation, the same values (10 Mb/s) of bit rate and maximal delay (10 ms) were used.

6.21. Simulation results using BSC multicast channel model

The first simulated model was BSC channel model in multicast wireless network [1]. In order to portray the graphs clearly, we simulated three different values of error rate: 1%, 10% and 50%. As is shown in Fig. 8, the first two values of error rate achieved approximately the same PSNR values. In some cases only, they were different due to more dynamic moments in video sequence. At the highest value of error rate (50%), the PSNR value was less than 20 dB.

On the basis of further simulations, we compared the values Q and PSNR, that are listed in Table.1. As is possible to see, the lower value of Q, the lower PSNR value and the lower quality of received video sequence. In case of Q = 92,83%, the PSNR value was equal to 32.7 dB, whereas in case of Q = 7.88%, the PSNR value was equal to 12.3 dB.

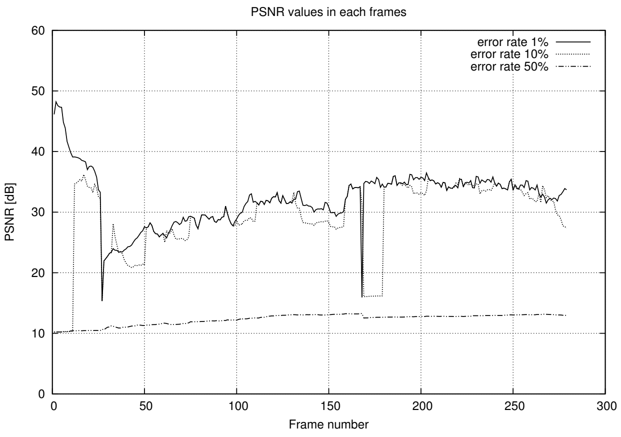

Fig. 8 PSNR values for BSC multicast channel model

Table 1 PSNR and Q values for BSC multicast channel model

Error rate	PSNR [dB]	Q
0.01	32.07	0.9283
0.05	30.81	0.8566
0.1	29.16	0.7562
0.3	19.96	0.2974
0.5	12.301	0.0788

Using video sequences obtained in simulations, we observed the losses and errors of each video sequences in yuv-viewer tool. We compared them with original ones. Fig. 9 and Fig. 10 show an example of that case. We compared the frame no.278 of used original video sequence (see Fig. 9 right) with damaged video (see Fig. 9 left) at 0.1 error rate.

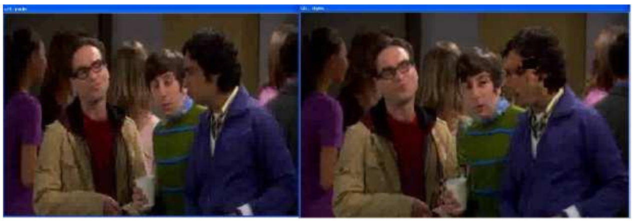

Fig. 9 Frame no.278 of the chosen original and damaged video sequence

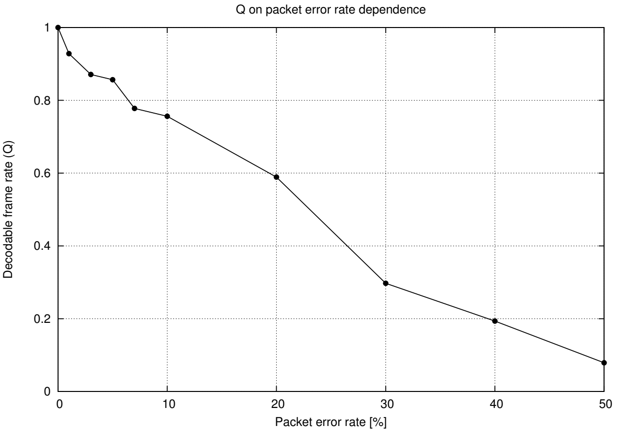

Fig. 10 The packet error rate in % and Q values comparison for BSC multicast channel model

6.22. Simulation results using BSC unicast channel model

The second simulated model was BSC channel model in unicast wireless network [1]. Again, in order to portray the graphs clearly, we simulated three different values of error rate. Since there was no difference in results between error rate values from 1% to 20%, we portrayed the graph for three values different from those in previous channel model, namely 1%, 30% and 80%. As is shown in Fig. 11, the first two values of error rate achieved approximately the same PSNR values. They were different only in some cases during more dynamic moments in video sequence. At the highest value of error rate (80%), the PSNR value was less than 20 dB, only from frame no. 120 to frame no. 160, PSNR exceeded 20 dB value.

Again, we compared the values of Q and PSNR, that are present in Table.2. As we already know, the lower value of Q, the lower PSNR value and the lower quality of received video sequence. In case of Q = 95,69%, the PSNR value was equal to 32.49 dB, what we consider to be a quite good video quality. Fig. 12 shows dependence of mentioned Q and error rate values using BSC unicast wireless channel model.

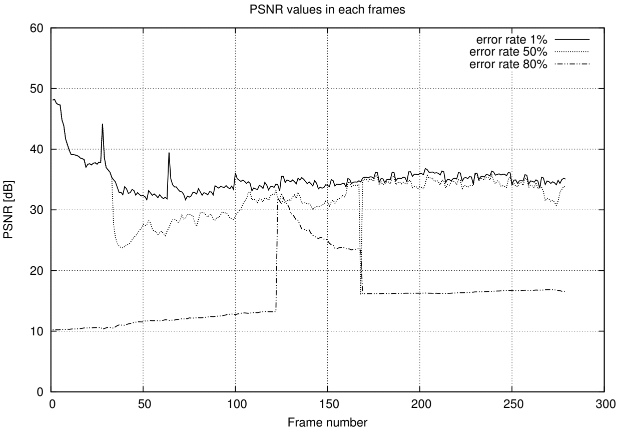

Fig. 11 PSNR values for BSC unicast channel model

Table 2 PSNR and Q values for BSC unicast channel model

Error rate	PSNR [dB]	Q
0.01	34.97	1
0.1	34.97	1
0.3	32.49	0.9569
0.5	28.81	0.6774
0.8	15.98	0.1218

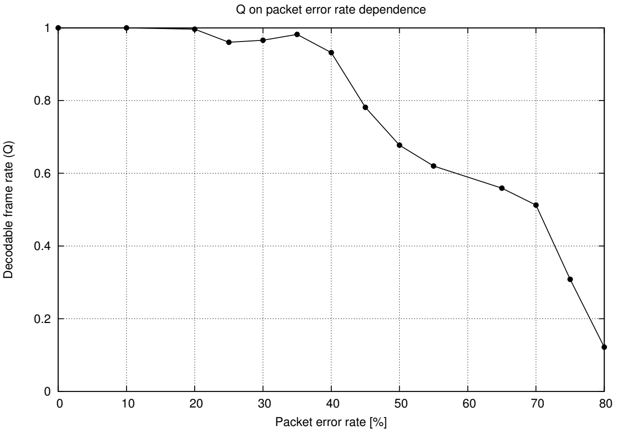

Fig. 12 The packet error rate in % and Q values comparison for BSC unicast channel model

6.23. Simulation results using Gilbert-Eliott multicast channel model

The last simulated model was BSC channel model in unicast wireless network [1]. We simulated three different values of error rate: 1%, 10% and 50%, as it was in first simulated channel model As is shown in Fig. 13, the first two values of error rate achieved approximately the same PSNR values. They were different only in some cases during more dynamic moments in video sequence. At the highest value of error rate (50%), the PSNR value was less than 20 dB, only from frame no. 120 to frame no. 160, PSNR exceeded 25 dB value.

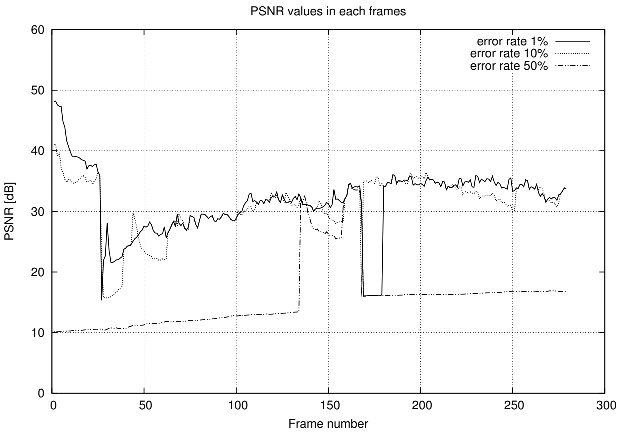

Fig. 13 PSNR values for Gilbert-Eliott multicast channel model

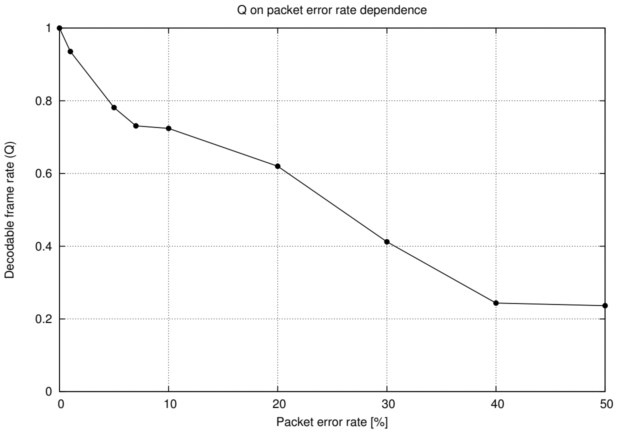

Fig. 14 The packet error rate in % and *Q* values comparison for Gilbert-Eliott multicast channel model

Table 3 PSNR and Q values for Gilbert-Eliott multicast channel model

Error rate	PSNR [dB]	Q
0.01	31.40	0.9354
0.05	31.28	0.7813
0.1	30.86	0.7240
0.3	25.42	0.4121
0.5	15.76	0.2365

As shows Table 3, Q and PSNR values are changing regularly, but at the level of error rate 30%, both values changed markedly, what we consider to be a worse video quality. Fig. 14 shows relation between Q and error rate values using Gilbert-Eliott multicast channel model.

7. CONCLUSIONS

In our paper, we focused on testing of new simulation tool-set MyEvalvid-NT, that connects Evalvid and NS-2 simulators. The benefit of such simulations is the possibility to test a real behaviour of wireless or others networks and various variants offered by this tool-set. We simulated two channel models, namely Binary Symmetric and Gilbert-Eliott models, in order to evaluate their characteristics, in both unicast and multicast transmission cases. From the simulation results, we learn that the video quality of the lower packet error rate is superior to the higher packet error rate and corresponds to the results of decodable frame rate (Q). In addition, because PSNR is a well-known performance metric of evaluating video delivery quality, we also revealed that the Q metric reflects the behavior of PSNR well [3]. Therefore, we can evaluate the video delivered quality more quickly through the tool-set when we verify designs in multimedia communication.

In brief, the used scenario would be a good choice for researchers who want to verify their designs in multimedia communication, such as network protocols or QoS control schemes.

ACKNOWLEDGEMENT

The work presented in this paper was supported by Grant of Ministry of Education and Academy of Science of Slovak Republic VEGA under Grand No.1/0828/12 ”Progresive Digital Signal Processing Methods for Wireless Reconfigurable Sensor Networks with 3D Vision” (50%). This publication is the result of the project implementation Centre of Information and Communication Technologies for Knowledge Systems (project number: 26220120030) supported by the Research & Development Operational Programme funded by the ERDF (50%).

REFERENCES

[1] ADAMČÍN, P.: Maskovanie chýb pri prenose videotokov bezdrôtovým prostredím v dôsledku preťaženia siete v programovom prostredí NS-2, Diploma thesis, Technical University of Košice, May 2011.

[2] ALTMAN, E. – JIMNEZ, T.: NS Simulator for Beginners, *Lecture notes, 2003-2004, Univ. De Los Andes, Mrida, Venezuela and ESSI, Sophia- Antipolis, France*, December 4, 2003.

[3] CHIA-YU, Y. – CHIH-HENG, K. – CE-KUEN, S. – CHILAMKURTI, N.: MyEvalvid-NT-A Simulation Tool-set for Video Transmission and Quality Evaluation, in: *TENCON 2006, 2006 IEEE Region 10 Conference, Hong-Kong*, 14-17 November 2006, pp.1–4.

[4] HUDEC, R. – BOJMÍR, M. – BENČO, M.: Video concealment method using the detection of damaged watermarks in frequency domain, *RTT 2010* september 2010, Velké Losiny, Czech Republic, ISBN 978-80-248-2261-7, (AEC), (Thomson Reuters)

[5] HYUNGKEUN, L. – VARSHNEY, P. K.: Gap-based modeling of packet losses over the Internet, *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems, MASCOTS 2002*, Fort Worth (Texas, USA), p. 507–510.

[6] ISSARIYAKUL, T. – HOSSAIN, E.: Introduction to Network Simulator NS-2, *Springer Science+Business Media*, LLC, 233 Spring Street, New York, USA, 2009.

[7] KLAUE, J. : Evalvid
http://www.tkn.tu-berlin.de/menue/research/evalvid/

[8] LAI HON, C.: Efficient Coding/Decoding Strategies for Channel with Memory, *Diploma thesis, University of British Columbia, Vancouver, Canada*, November 1992.

[9] MARKOVSKI, V. – XUE, F. – TRAJKOVIC, L.: Simulation and analysis of packet loss in video transfers using User Datagram Protocol, *The Journal of Supercomputing, Kluwer Academic Publishers, Hingham (Massachusetts,USA)*, 2001, Vol. 20, pp. 175–196. ISSN 0920-8542.

[10] MATTSSON, N.: A DCCP Module for Ns-2, *masters thesis, Lulea University of Technology*, 2004.

[11] MPEG-4 and H.263 Video Traces for Network, Performance Evaluation:
http://www.tkn.tu-berlin.de/research/trace/trace.html

[12] PIEDA, P. – ETHRIDGE, J. – BAINES, M. – SHALLWANI, F.: A Network Simulator Differentiated Services Implementation, *Open IP, Nortel Networks*, July 26, 2000.

[13] POLEC, J. – KARLUBÍKOVÁ, T. – ONDRUŠOVÁ, S. – KOTULIAKOVÁ, K.: An Error Evaluation Apparatus for Object Contour. *In: EUROCON 2009 : International IEEE Conference devoted to the 150-Anniversary of Alexander S.Popoov.* Saint Petersburg, Russia, May 18–23, 2009, Piscataway: IEEE, 2009, pp. 1354–1359, ISBN 978-1-4244-3861-7.

[14] POLEC, J. – ONDRUŠOVÁ, S. – KOTULIAKOVÁ, K. – HELEŠOVÁ, N. – KARLUBÍKOVÁ, T.: Hierarchical Transform Coding Using NURBS Approximation. *In: Proceedings Elmar-2008 : 50th International Symposium ELMAR-2008*, Zadar, Croatia, 10-12 September 2008. - Zadar : Croatian Society Electronics in Marine, 2008, Vol. 1, pp. 65–68, ISBN 978-953-7044-09-1.

[15] POLEC, J. – POHANČENÍK, M. – ONDRUŠOVÁ, S. – KOTULIAKOVÁ, K. – KARLUBÍKOVÁ, T.: Error Concealment for Classified Texture Images. *In: EUROCON 2009: International IEEE Conference devoted to the 150-Anniversary of Alexander S.Popoov. Saint Petersburg*, Russia, May 18–23, 2009, Piscataway: IEEE, 2009, pp. 1348-1353, ISBN 978-1-4244-3861-7.

[16] SALLEH, A. U. – ISHAK, Z. – DIN, N. – JAMALUDIN, Z.: Trace Analyzer for NS-2, *4th Student Conference on Research and Development (SCOReD 2006)*, Shah Alam, Selangor, MALAYSIA, 27-28 June, 2006.

[17] SANNECK, H. – CARLE, G. – KOODLI, R.: A framework model for packet loss metrics based on loss runlengths, *Proceedings of the SPIE/ACM SIGMM Multimedia Computing and Networking Conference 2000 (MMCN 2000)*, San Jose (California, USA), 2000, pp. 177–187.

Received January 10, 2012, accepted May 24, 2012

BIOGRAPHIES

Ján Valiska was born on 14.10.1986 in Michalovce. He received his M.Sc. degree in Technical University of Košice. He graduated from Department of Electronics and Multimedia Communications, from Faculty of Electrical Engineering and Informatics in May 2011. He is a PhD. student in Department of Electronics and Multimedia Communications at the same university since september 2011. His current research interests include image processing, object tracking, digital television systems and satellite communications.

Branislav Hrušovský was born on 23.10.1984 in Košice. He received his M.Sc. degree in Technical University of Košice. He graduated from Department of Electronics and Multimedia Communications, from Faculty of Electrical Engineering and Informatics in May 2009. He is a PhD. student in Department of Electronics and Multimedia Communications at the same university since september 2009. His current research interests include error concealment of coded videodata and satellite communications.

Stanislav Marchevský received the M.Sc. in electrical engineering at the Faculty of Electrical Engineering, Czech Technical University in Prague, in 1976 and PhD. degree in radioelectronics at the Technical University of Košice in 1985. Currently he is a Professor of Electronics and Multimedia Communications Department of Faculty of Electrical Engineering and Informatics of Technical University of Košice. He teaching interests include switching theory, digital television technology, and satellite communications. His research interests include image nonlinear filtering, neural networks, genetic algorithms, and multiuser detection, space-time communication, diversity communication over fading channel, and power and bandwidth efficient multiuser communications.

Slavomír Pillár was born in 1975 in Bardejov, Slovakia. He received the Ing. (MSc) degree in Radioelectronics at the Faculty of Electrical Engineering, Technical University of Koice, Slovakia, in 2001. Currently, he works as an electrotechnical engineer in Teleservis Ltd., Košice. Hi is interested in satellite communications and other satellites services.