Research Journal of Applied Sciences 8 (3): 199-207, 2013
ISSN: 1815-932X

NetLab: Simple Network Simulator for Educational Use: Case Study: Streaming Video over Band-Limited Channel

Mohamed E. Khedr, Heba G. Badie and Moustafa H. Aly
Arab Academy for Science, Technology and Maritime Transport,
Department of Electronics and Communications,
College of Engineering and Technology, Alexandria, Egypt

Abstract: This study presents a case study of video streaming over band-limited channel using a new network simulator called NetLab. Performance investigation and network requirements were carried out using this simulator by sending frames of video data over a shared band-limited wireless channel. The simulator investigates the deadline waiting time and packet loss parameters as an indication to the performance of the network. The simulator model is divided into three parts: the video source that provides the frames of video data to entities, the band-limited communication channel which buffers the incoming frames until they are served and the video end-user that buffers the received frames and forwards them to a viewer at a constant rate. This model was also developed to support more than one user using the same band-limited communication channel to send to more than one end-user. Video streaming is taken as one of the case studies carried out to proof the ease use of the proposed NetLab simulator in education and training of network models.

Key words: NetLab, investigation, buffers, video streaming, communication

INTRODUCTION

Modelling and simulation of network in general and wireless in specific has been the main field of research for many researches in the last years as it is essential to have an accurate evaluation of the performance of such networks (Wang and Poor, 2002). Simulation programs became extremely useable now a days. Those programs simulate many types of systems and validate the operation of such systems through the results of the simulation instead of building expensive prototypes to check that they are working. This makes the usage of simulation programs in simulating the desired systems cheaper, easier and more efficient (Forouzan, 2007; Fernekess *et al*., 2009). There are many network simulation programs available whether commercial or open source. For example, there is OPNET (Optimized Network Engineering Tools), ns-2 (network simulation) and GLOMOSIM (Global Mobile Information System Simulator).

OPNET is a leading provider of solutions for managing networks and applications. It provides a GUI (Graphical User Interface) for the topology design which allows for fast development of simulated networks and has a powerful performance data collection and display modules. The disadvantage of using OPNET is that it requires a high learning curve and also simulation requires a lot of processing power and can be very time consuming particularly for networks with a large number of transmitters and receivers.

The simulator ns-2 is another discrete event network one. It is popular in academia for its extensibility and plentiful online documentation. Ns-2 is popularly used in the simulation of routing and multicast protocols, among others and is heavily used in ad-hoc networking research. The key advantage of ns-2 is its rich libraries of protocols for nearly all network layers and for many routing algorithms. However, it is nearly as complicated as OPNET and requires a high learning curve.

Comparing MATLAB to OPNET and ns-2, it was found that MATLAB is both simple and widely accepted by the engineering community and is easily extensible and with very powerful data processing and representation modules. This point of view initiated the idea of using MATLAB as commercial and simple software in creating a new library to be used in simulating wired/wireless networks thus achieving the idea of presenting a simple, commercial, network simulation software.

This idea of developing a simulation tool was presented by Goes *et al*. (2005) where the JSDESLib (Java Simple Discrete-Event Simulation Library) is a simplified and easy to use Java-based library for the

Corresponding Author: Mohamed E. Khedr, Arab Academy for Science, Technology and Maritime Transport, Department of Electronics and Communications, College of Engineering and Technology, Alexandria, Egypt

development of discrete event simulation tools of parallel systems. The main objectives are: to propose, develop and implement JSDESLib: as a parallel event-oriented simulation library that simplifies the development of simulation tools with a next earliest event mechanism and communication management between entities.

Researchers developed a new simulator called Network laboratory NeTLaB based on existing MATLAB toolbox called the SIMEVENT toolbox. NeTLaB was developed with the target of easing the process of simulating networks as a discrete-event simulator and allowing the rapid prototyping of vast network types. NeTLaB was used earlier in building the medium access control of the IEEE 802.11 wireless LAN.

In this study, NeTLab is used to build scenarios of video streaming over a limited bandwidth communication channel model. It proposes a communication system that sends frames of video data over a wireless shared channel. By dropping frames that wait too long time to reach the end user, the model also illustrates how to use timeouts to implement point to point timing constraints and losses due to congestion. The simulator NeTLaB is developed to support multiple users using the same bandwidth-limited communication channel to send to more than one end-user and the performance of the system was experimented under different parameter variation.

SYSTEM MODEL

Abbas *et al.* (2011) presented a WiMAX simulator to evaluate the performance of the system. The key parts of the simulator are end to end communication path, traffic generation, Medium Access Control (MAC) and Physical (PHY) layers, resource allocation, frame construction and configuration options such as Adaptive Modulation and Coding (AMC).

The goal of the research by Segata and Cigno (2012) was to highlight the limitations of traditional physical channel models used in network simulators for wireless LANs with particular reference to VANETs. The development of a stochastic channel model which improves reliability of simulations was proposed but at the price of unacceptable computational (and model).

A proposed novel approach for video distribution over IEEE 802.16 networks for mobile Healthcare (M-Health) applications was presented by (Markarian *et al*., 2012). The technique incorporates resource distribution, scheduling and content-aware video streaming taking advantage of a flexible quality of service functionality offered by IEEE 802.16/WiMAX.

Fida *et al.* (2011) described a simulator-DTNRSIM to evaluate various routing protocols designed for delay-tolerant networks. The simulator is written in Java and is usable on Windows platform. The system model differs from abovementioned simulators in its ability to being simple, compact and based on MATLAB a well-known and trusted software. The main objectives of developing a discrete-event network simulator using MATLAB are simplifying the development of networks, developing an interaction mechanism and communication management between entities and finally developing a data collecting modules for verifying and analyzing the performance of networks.

NETLAB library consists of two parts. The first part is standard components and the second part is custom components. The standard components constitute the standard sources, sinks, queues and servers found in standard networks. On the other hand, custom components are generic components that can be altered and customized by the user according to his needs and requirements.

NETLAB also includes state diagrams, communication channels and enablers. The state diagrams are used to provide conditional transitions and functional calls while communication channels are used to provide the medium for entity transmission and synchronization between different modules. Finally, enablers are used to route entities in the model.

Each event represents an important action in a determined instant of time that changes the system state. An event must contain an identifier of the sender entity, the time instant of each event occurrence, the destination entities and data. NeTLab also includes state variables S = (s1, s2, ..., sn) that define the current state of the system it is simulating; a time-ordered event queue that contains pending, times stamped events and a global clock which indicates the current simulated time. An event's timestamp indicates the time at which the event would occur in the actual system the NeTLab is simulating. In a simulation of a network an event could represent the beginning of a transmission, the expiration of a timer or the movement of a node. NeTLab follows the standard model of simulation which is:

- Planning: in the planning phase, parameters that need to be investigated through simulation are defined as well as different modules such as queues, switches, path combiner, etc.
- Modelling phase: in this phase, the network model is constructed which is a representation of the real network to be simulated
- Model building: this includes abstraction of the system into a mathematical relationship with the problem formulation
- Model translation: preparation and debugging of the model for computer processing

Verification and validation are two important tasks that should be carried out for any simulation study. Verification is the process of finding out whether the model implements the assumptions correctly. Validation, on the other hand, refers to ensuring that the assumptions used in developing the model are reasonable in that if correctly implemented, the model would produce results close to these observed in real systems. The process of model validation consists of validating assumptions, input parameters and distributions and output values and conclusions. Validation can be performed by one of the following techniques: comparing the results of the simulation model with results historically produced by the real system operating under the same conditions; expert intuition; theoretical (analytic) results using queuing theory or other analytic methods; another simulation model and artificial intelligence and expert systems.

The case study presented in this study is divided into three parts, first the video source shown in Fig. 1 where the first block (from multimedia file) reads the video frame from multimedia file. The time based entity generator block which generates entities on which the video frame is attached by the set attribute block. The set attribute block also assigns the ID of each source to the generated entity then it goes through the schedule timeout block that schedules a timeout event for each arriving entity.

In other words, it limits the time that an entity spends on designated entity paths during the simulation. The entity is generated from the video source and goes into the second part of the model, the band-limited communication channel shown in Fig. 2. The channel starts with a path combiner that receives entities from all video sources as inputs to the channel and outputs the entities according to the input port precedence parameter. This parameter indicates how the block determines which entity input port to make available first whenever the output port changes from blocked to unblocked. The entity advances to the FIFO queue where the entities are stored in a first in first out sequence. It also has a port for the timed out entities and another for the number of entities in the queue. This number is used to plot the input buffer curve using the signal scope block. After that the entities move to the single server block that serves one entity for a period of time and then attempts to output the entity through the output port. It also has a port for the timed-out entities. Both timed out entities from FIFO queue block and single server block are inputs to another path combiner block and the output of the path combiner is input to an entity sink. The number of entities arrived at the entity sink is used to plot a timed-out entity curve using the signal scope block (Farahani, 2008).

The third part of the model is the video end user shown in Fig. 3 where the entity that departed from the single server block in the channel is input to a get attribute block in the end user. This block reads the value of the source ID. The entity then advances to the subsequent output switch block and the value of the source ID is used by this block as the value of the switching criterion parameter. This parameter indicates how the block determines which entity output port is selected for departure at any given time.

The selected output entity advances to the cancel timeout block which cancels the timeout tag assigned by the schedule timeout block at the video source. The entities are stored in the FIFO queue and the number of entities in queue is used to plot the reception buffer curve. Then, they proceed to the release gate block which permits the arrival of one pending entity only when a signal-based event or function call occurs. The gate opens when a change in the numerical value of a signal at port vc occurs. This change in the numerical value comes from the product of the flow control and the sample signal

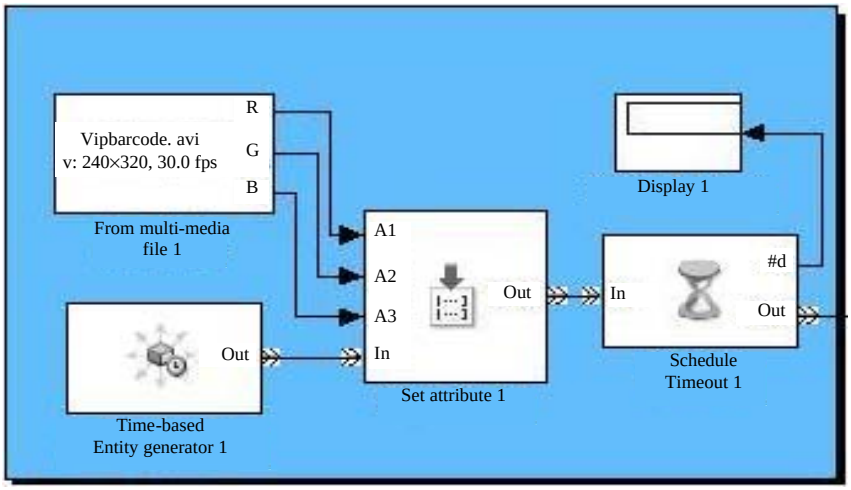

Fig. 1: Video source

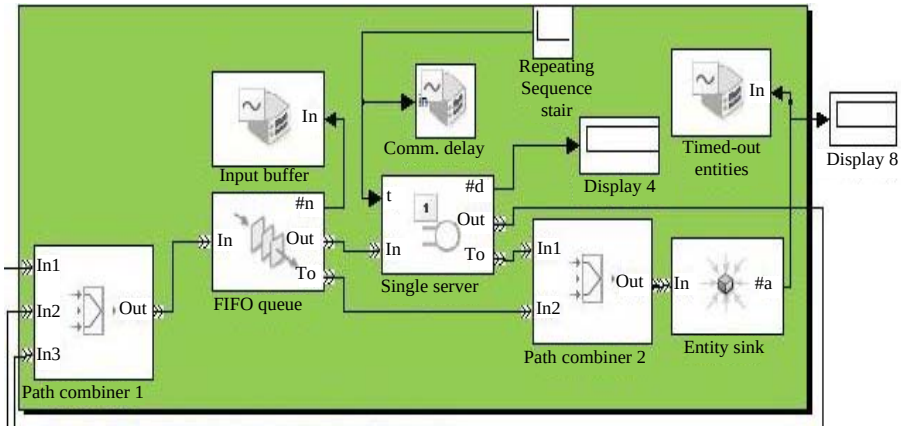

Fig. 2: Bandwidth limited communication channel

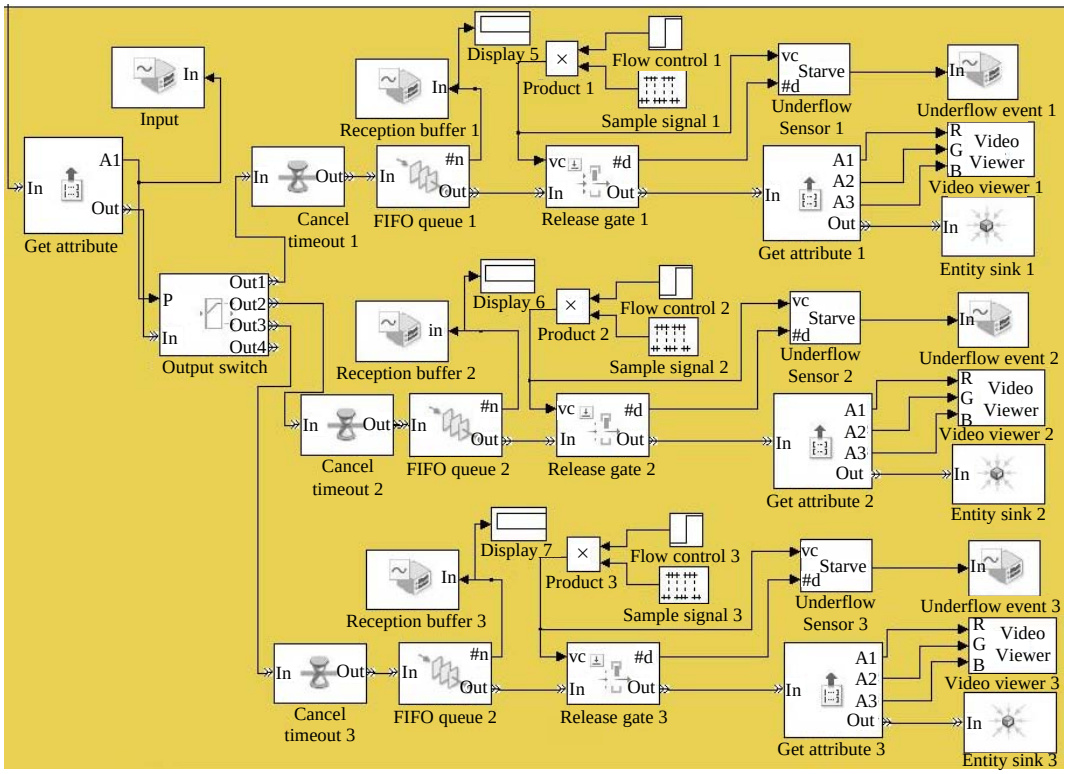

Fig. 3: Video end-user

block. The flow control is a step block that provides a step between two definable levels at a specified time. In another words if the simulation time is less than the step time parameter value, the block's output is the initial value parameter value. Also, for simulation time greater than or equal to the step time, the output is the final value parameter value. The sample signal is a pulse generator block that generates a square wave pulses at regular intervals. In the simulation model, the video viewer is initially blank and does not start to display the video until the simulation time is equal to 10 sec. The reason is that the block labelled flow control changes its output value from 0-1 at a simulation time equal to 10 sec which enables the periodic release of frames to the viewer. The entities proceed to the last part of the video end user which is the get attribute block where the R, G and B components of the video are output to the video viewer block which displays the video sent by the video source. A flowchart of the model is shown in Fig. 4.

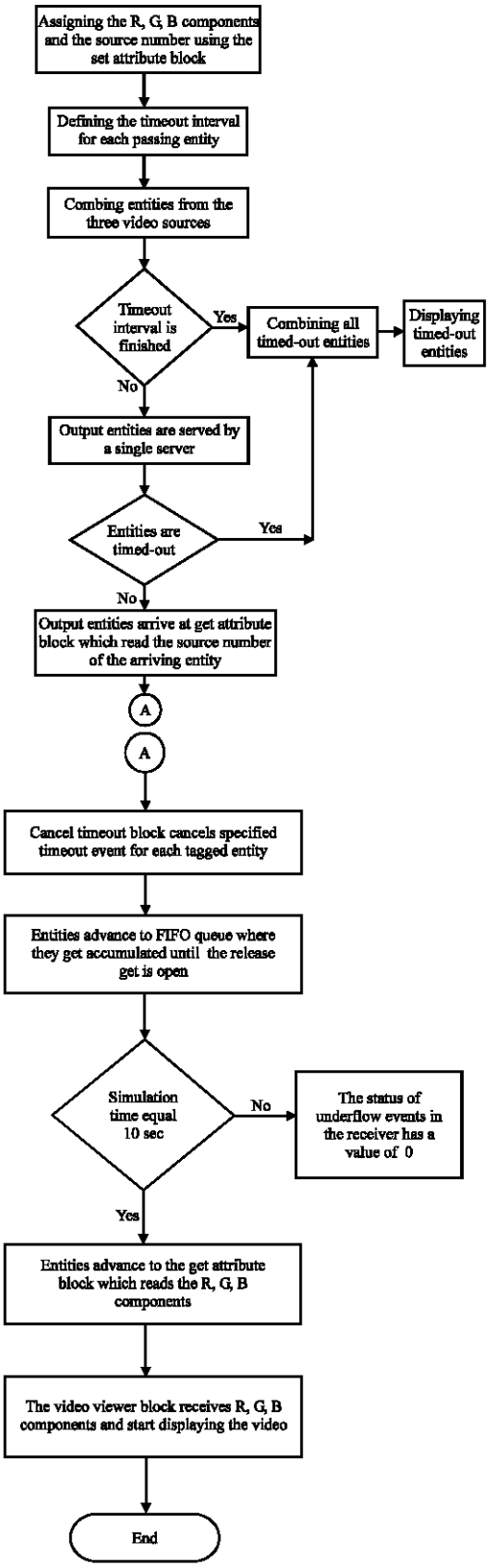

Fig. 4: Flowchart of the Video Streaming Model

SIMULATION AND RESULTS

The main issue affecting the performance of the model is the service time of a single server block which comes from the repeating sequence stair block in the simulation model.

1/30 sec communication delay per frame: The repeating sequence stair block outputs and repeats a discrete time sequence; the stair sequence can be specified with a vector of output values parameter. In this case, the vector of output values parameter is equal to 1/30 sec for sample time equal to 4 sec which means that for a duration of 4 sec, the received entities undergo a delay of 1/30 sec. The delay is changed for the next 4 sec and so on. But since in this case the repeated sequence is always equal to 1/30 then the entities undergo a delay of 1/30 for the entire simulation time. The integration time for generating entities is in a range from 1/50-1/30 sec so that entities are generated at random times between the required delay (1/30) and minimum delay (1/50) and all video sources have equal probabilities in sending their frames.

Input buffer curve: Every 1/30 sec, the FIFO queue of the channel receives 3 frames as researchers have three video sources in this simulation; one is served by the server and the two frames remain for the next 1/30 sec. Therefore, 2 frames will be queued every 1/30 sec. The number of frames in 1/10 sec will be equal to 6 frames as shown in Fig. 5. And thus in 10 sec it will be equal to 600 frames.

The total number of frames in 40 sec (total simulation time) will be 2400 frames when the integration time value equals 1/30 sec. When the integration time for generating entities is equal to 1/50 sec, the total number of frames in 40 sec will change. The number of frames in 1/10 sec is equal to 12 frames as shown in Fig. 6. So, in 10 sec it will be equal to 1200 frames. The total number of frames will be 4800 frames when the integration time value is equal to 1/50 sec. Since, the integration time parameter value falls in the range between 1/50-1/30 sec so the total number of frames in 40 sec will be equal to 3600 frames as shown in Fig. 7. This number falls in to the range between 2400 and 4800 frames.

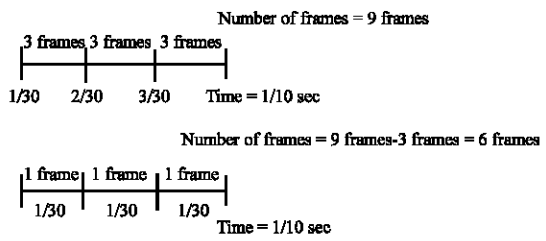

Fig. 5: Number of frames at integration time = 1/30 sec and communication delay = 1/30 sec

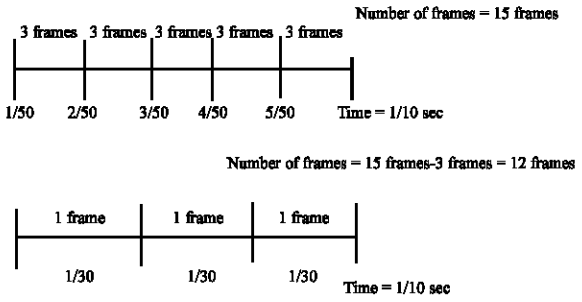

Fig. 6: Number of frames at integration time = 1/50 sec and communication delay = 1/30 sec

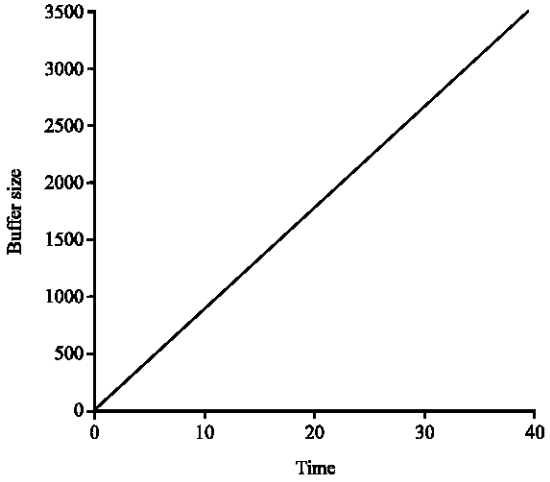

Fig. 7: Input buffer curve at a communication delay equals 1/30 sec

Reception buffer curve: Since, the server at the channel serves only one frame for 1/30 sec and 2 frames remain to the next 1/30 sec, the reception buffer will thus receive one frame at 1/30 sec and the other frames at multiples of 3/30 sec. This means that every 3/30 sec, it receives one frame. Therefore, the total number of frames in 10 sec is equal to 100 frames. Since, that the integration time parameter value falls in the range between 1/50-1/30 sec. It is found that there is a variation in the number of frames above and below 100 frames at the reception buffer.

Then, at time = 10 sec, the video starts to display from the frames accumulated in the reception FIFO queue. At the same time, there are still frames entering the reception FIFO queue. The number of frames entering the FIFO queue in one second is equal to 10 frames and the number of frames exiting in 1 sec is equal to 30 frames. So, subtracting the two numbers, it is found that the number of depleted frames from the reception FIFO queue in one second is equal to 20 frames.

Therefore, when the time equals 15 sec, the 100 frames would be depleted. Since, the integration time parameter value falls in the range between 1/50-1/30 sec, it is found that there is a variation in the time around the value of 15 sec.

Starting from time = 15 sec (after losing the accumulated 100 frames in display), the ongoing process of one frame entering the reception FIFO queue every 1/10 sec doesn't allow any frames to accumulate and the frames are displayed right away and that is why starting from time = 15 sec the reception buffer curve is equal to zero as shown in Fig. 8.

1/20 sec communication delay per frame: Repeating the same method of calculations when the communication delay per frame is equal to 1/20 sec, the following results are obtained.

Input buffer curve: The total number of frames will be 2800 frames when the integration time value is equal to 1/30 sec and the total number of frames will be 5200 frames when the integration time value is equal to 1/50 sec. Since, the integration time parameter value falls in the range between 1/50-1/30 sec so the total number of frames in 40 sec will be equal to 4000 frames and will be linear similar to Fig. 7. However, the number falls in to the range between 2800 and 5200 frames.

Reception buffer curve: The total number of frames in 10 sec is equal to 66.666 frames. Then, at time = 10 sec, the video starts to display the frames. It is found that the number of depleted frames from the reception FIFO queue in one second is equal to 23.334 frames. Therefore, by time equals 12.857 sec, the 66.66 frames would be depleted.

Starting from time = 12.857 sec (after losing the accumulated 66.666 frames in display), the ongoing process of one frame every 3/20 sec entering the reception FIFO queue doesn't allow any frames to accumulate and the frames are displayed right away. Therefore, the reception buffer curve is equal to zero as shown in Fig. 9.

1/40 sec communication delay per frame: Repeating the same method of calculations when the communication delay per frame is equal to 1/40 sec, the following results are obtained.

Input buffer curve: The total number of frames will be 2000 frames when the integration time value is equal to 1/30 sec and the total number of frames will be 4400 frames when the integration time value is equal to 1/50 sec. Since, the integration time parameter value falls in the range between 1/50-1/30 sec so the total number of frames

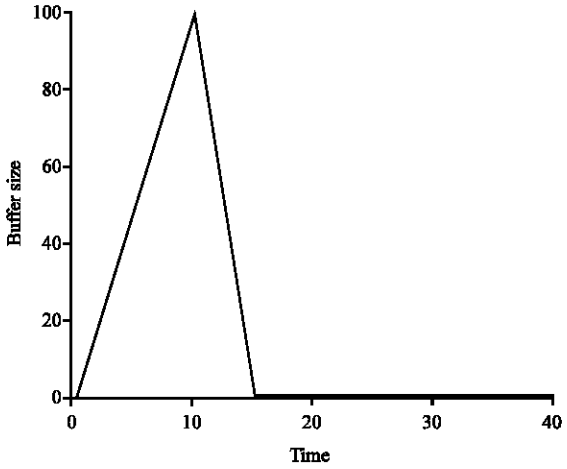

Fig. 8: Reception buffer curve at a communication delay equals 1/30 sec

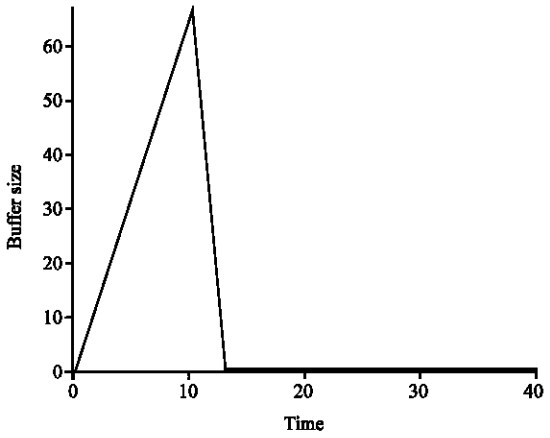

Fig. 9: Reception buffer curve at a communication delay equals 1/20 sec

in 40 sec will be equal to 3200 frames and will be linear similar to Fig. 6. However, the number falls in to the range between 2000 and 4400 frames.

Reception buffer curve: The total number of frames in 10 sec equal to 133.33 frames. Then, at time = 10 sec, the video starts to display the frames. It is found that the number of depleted frames from the reception FIFO queue in one second is equal to 16.67 frames. Therefore, by time equal to 17.998 sec, the 133.33 frames would be displayed. Starting from time = 17.998 sec (after losing the accumulated 133.33 frames in display), the ongoing process of one frame every 3/40 sec entering the reception FIFO queue doesn't allow any frames to accumulate and the frames are displayed right away. Therefore, reception buffer curve is equal to zero as shown in Fig. 10.

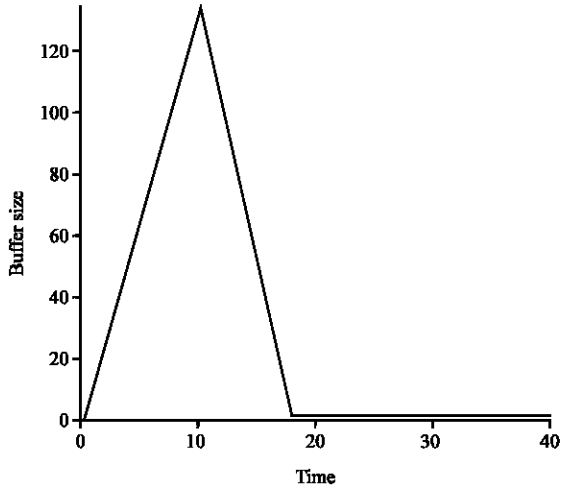

Fig. 10: Reception buffer curve at a communication delay equals 1/40 sec

1/30 sec communication delay per frame and variable timeout interval

Timeout interval = 1 sec: In this model, three frames are sent, one is served and two remains. Since, the timeout interval is equal to 1 sec thus dividing the 1 sec by the two remaining frames, each frame gets 0.5 sec. Therefore, after the 1 sec ends, there is 0.5 sec as the remaining time before the FIFO queue starts to timeout the frames. Since, that the integration time parameter value falls in the range between 1/50-1/30 sec. It is found that there is a variation in the time around the value of 1.5 sec.

Input buffer curve: Similar calculations are performed as in section A. The total number of frame in 1.5 sec will be 90 frames when the integration time value is equal to 1/30 sec. When the integration time for generating entities is equal to 1/50 sec, the total number of frames in 1.5 sec will change. Since, the integration time parameter value falls in the range between 1/50-1/30 sec, the total number of frames in 1.5 sec will be equal to 120 frames as shown in Fig. 11. This number falls in range between 90 and 180 frames as expected from theory. After time = 1.5 sec, the FIFO queue starts to timeout the frames. Therefore, the number of frames remains constant at 120 frames.

Reception buffer curve: Since, the reception buffer receives one frame at 1/30 sec and then receives another frame at 4/30 sec, this means that it receives one frame every 3/30 sec. Therefore, the total number of frames will be 15 frames in 1.5 sec. Then, after time = 1.5 sec, the frames start to timeout. So, it stays constant at 15 frames. Then, after time = 10 sec, the video starts to display one frame every 1/30 sec. So, it starts losing the accumulated frames. Therefore, by time equal to 10.5 sec, the 15 frames would be displayed as shown in Fig. 12.

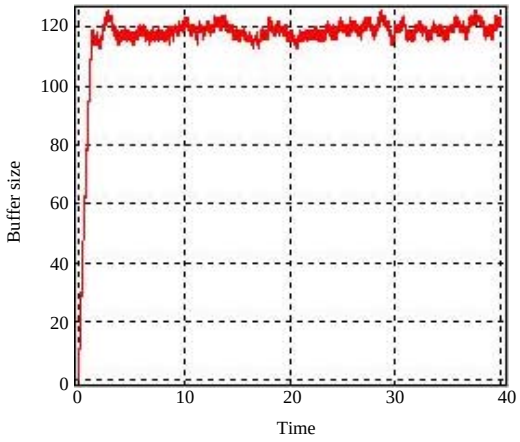

Fig. 11: Input buffer curve at a communication delay = 1/30 sec and timeout interval = 1 sec

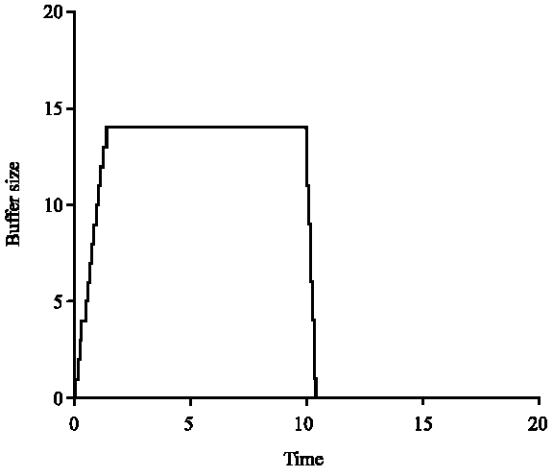

Fig. 12: Reception buffer curve at a communication delay = 1/30 sec and timeout interval = 1 sec

Variable timeout interval: The model is simulated again after changing the timeout interval value several times. The output values are recorded and are used to draw a general curve. The MATLAB curve fitting toolbox is used to draw this curve where the recorded values are input to the curve fitting tool and are used to generate the curve. The fitting option in the toolbox is used to fit the input data to the polynomial equation and to its degree too.

Reception buffer curve: The reception buffer curve consists of two curves; the blue curve and the red one. The red curve represents the relation between the timeout interval (X-axis) and the saturation time (Y-axis). The curve follows the 8th degree polynomial. The blue curve represents the relation between the timeout interval (X-axis) and the buffer size (Y-axis). The curve follows the 7th degree polynomial as shown in Fig. 13. It was found

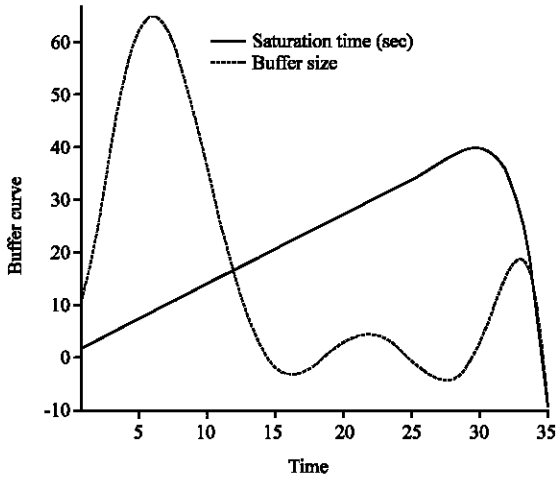

Fig. 13: Reception buffer curve at a communication delay = 1/30 sec and variable timeout interval

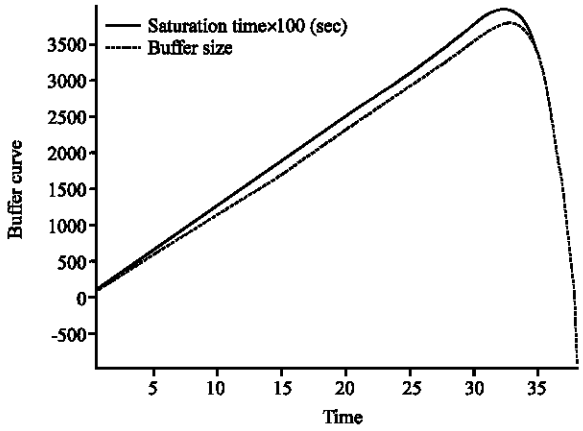

Fig. 14: Input buffer curve at a communication delay = 1/30 sec and variable timeout interval

that after the curve fitting process, the polynomial equation results contained some negative values, these values are neglected.

Input buffer curve: The blue curve represents the relation between the timeout interval (X-axis) and the saturation time (Y-axis) following the 8th degree polynomial. The red curve represents the relation between timeout interval (X-axis) and the buffer size (Y-axis) following the 8th degree polynomial. The saturation time range of numbers is smaller than the buffer size range of numbers. To overcome this issue, the saturation time values were multiplied by 100 as shown in Fig. 14. Similar to the reception buffer curves, the negative values of the polynomial equation are neglected.

Table 1: Number of input and reception buffer curve frames for various communication delays

Parameters	Communication delay = 1/40 sec	Communication delay = 1/30 sec	Communication delay = 1/30 sec
Input buffer curve	3200 frames in 40 sec	3600 frames in 40 sec	4000 frames in 40 sec
Reception buffer curve	133.33 frames in 10 sec then at time = 17.998 sec, 133.33 frames are displayed	100 frames in 10 sec then at time = 15 sec, 100 frames are displayed	66.66 frames in 10 sec then at time = 12.85 sec, 66.66 frames are displayed
Input buffer curve a timeout interval = 1 sec	125 frames in 1.5 sec then the curve stays constant	120 frames in 1.5 sec then the curve stays constant	124 frames in 1.5 sec then the curve stays constant
Reception buffer curve at timeout interval = 1 sec	20 frames in 1.5 sec then the curve stays constant then at 10.66 sec, 20 frames are displayed	15 frames in 1.5 sec then the curve stays constant then at 10.5 sec, 15 frames are displayed	9.99 frames in 1.5 sec then the curve stays constant then at 10.33 sec, 9.99 frames are displayed

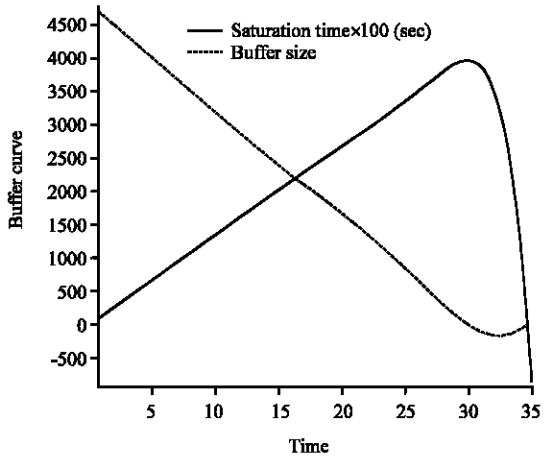

Fig. 15: Timed-out entities curve at a communication delay = 1/30 sec and variable timeout interval

Timed out entities curve: The red curve represents the relation between the timeout interval (X-axis) and the saturation time (Y-axis) following the 8th degree polynomial. The blue curve represents the relation between timeout interval (X-axis) and the buffer size (Y-axis) following the 7th degree polynomial as shown in Fig. 15.

CONCLUSION

In this study, the video streaming over a limited bandwidth communication channel model is explained. This model illustrates a communication system that sends frames of video data over a channel. By dropping frames that wait too long time to reach the end user, the model illustrates how to use timeouts to implement point to point timing constraints.

The simulation model was developed to support more than one user using the same bandwidth-limited communication channel to send to more than one end-user and the performance of the system was experimented under different parameter variation and the results are shown in Table 1.

REFERENCES

Abbas, N., H. Hajj and A. Borghol, 2011. A comprehensive WiMAX simulator. Proceedings of the 2011 IEEE Consumer Communications and Networking Conference, January 9-12, 2011, Las Vegas, NV., pp: 4-6.

Farahani, S., 2008. Zigbee Wireless Networks and Transceivers. Elsevier Ltd., UK., ISBN: 978-0-7506-8393-7.

Fernekess, A., A. Klein, B. Wegmann and K. Dietrich, 2009. Modular system-level simulator concept for OFDMA systems. IEEE Commun. Mag., 47: 150-156.

Fida, M., M. Ali and A.S. Arsalaan, 2011. DTN-RSim: An event-based routing simulator for DTN. Proceedings of the International Conference on Computer Networks and Information Technology, July 11-13, 2011, Abbottabad, pp: 99-104.

Forouzan, B.A., 2007. Data Communications and Networking. 4th Edn., The McGraw-Hill Co. Inc., London.

Goes, L.F.W., C.V. Pousa, M.B. Carvalho, L.E.S. Ramos and C.A.P.S. Martin, 2005. JSDESLib: A library for the development of discrete event simulation tools of parallel systems. Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, April 4-8, 2005, Minas Gerais, Brazil, pp: 6-6.

Markarian, G., L. Mihaylova, D.V. Tsitserov and A. Zvikhachevskaya, 2012. Video distribution techniques over WiMAX networks for m-Health applications. Inform. Technol. Biomed., 16: 24-30.

Segata, M. and R.L. Cigno, 2012. Simulation of 802.11 PHY/MAC: The quest for accuracy and efficiency. Proceedings of the 9th Annual Conference on Wireless On-demand Network Systems and Services, January 9-11, 2012, Courmayeur, pp: 99-106.

Wang, X. and H.V. Poor, 2002. Wireless Communication Systems: Advanced Techniques for Signal Reception. Prentice Hall, New York.