

A Testbed for Evaluating Video Streaming Services in LTE

M. Solera*, M. Toril, I. Palomo, G. Gomez, J. Poncela*

*Dpto. Ingenieria de Comunicaciones, Universidad de Malaga, Campus de Teatinos s/n,
29071 Malaga, Spain*

Abstract

With the deployment of the first commercial Long Term Evolution (LTE) networks, mobile operators need to understand how Quality of Service (QoS) network indicators and codec parameters affect subjective quality in video streaming services as perceived by customers. In this paper, the development of a testbed for evaluating the Quality of Experience (QoE) of 3D video streaming service over LTE is described. The proposed system consists of three elements: a streaming server, an Internet Protocol (IP)-level mobile network emulator, based on NetEm tool, and a streaming client. The main contribution of this testbed is the modification of NetEm code to model the impact of time correlation between packet arrivals on the packet delay in a video stream. In the testbed, different network conditions are configured by setting network emulator parameters based on the results obtained by a system-level LTE simulator. Results show how average network load and user position inside a cell have a strong impact on the QoS and QoE perceived by the end video user.

Keywords: Network emulation, Quality of Experience, 3D video, LTE, Streaming

2010 MSC: 00-01, 99-00

*Corresponding author

Email address: msolera@ic.uma.es ()

1. Introduction

The introduction of tablets and smartphones has favored the launch of new multimedia mobile services. Mobile video is forecast to reach 70% of all mobile data traffic by 2021 [1]. Unfortunately, current mobile networks were not
5 conceived for the distribution of such demand of video traffic. In this context, the recent development of 4G commercial mobile networks based on Long Term Evolution (LTE) standard opens new opportunities to offer real-time video streaming services over these networks. Nonetheless, their consolidation requires efficient mechanisms to encode/decode, compress/decompress and trans-
10 mit/receive this data.

At the same time, rising user expectations are conditioning the way operators manage their networks. Until very recently, mobile network operators have used Quality of Service (QoS) indicators, such as packet loss rate, packet delay or packet jitter, to evaluate network performance. Although QoS measurements
15 are related to user satisfaction, the exact relationship is hard to establish. Thus, network operators are changing their network management processes, currently focused on network performance, to a more modern approach, focused on user opinion. In a global market where most cellular operators offer similar services, Quality of Experience (QoE) is a key driver for differentiation amongst opera-
20 tors [2]. For this reason, vendors have identified end-to-end service management and provision as a crucial technology for the development of mobile broadband networks [3].

As service provision gets more complicated, defining reliable QoE models that link objective network-level measurements to end-user perceived quality
25 becomes a challenging task. In the case of video streaming service, such a complexity comes from the introduction of data encryption (e.g., Transport Layer Security- TLS, Secure Sockets Layer- SSL [4]), adaptive coding (e.g., MPEG-DASH [5]) or new protocols (e.g., Quick UDP Internet Connections, QUIC [6]). To deal with this complexity, researchers adopt three different approaches to
30 evaluate end-to-end network performance: 1) simulation, where a simplified

model of the real environment is used to estimate user QoE from QoS measurements by means of utility functions [7]; 2) emulation, which reproduces network behavior within a controlled environment based on the values of some general network performance parameters; and 3) field trials, where the impact
35 of network parameters on the real users is checked in a live scenario.

The ITU-T G.1070 [8] and ITU-T J.144 [9] standards propose QoE models for video telephony and IPTV. The Broadband Forum has also defined similar models for video, audio and data [10]. More sophisticated QoE models also consider aspects such as service availability and accessibility [11]. In video
40 streaming, it is difficult to establish a direct relationship between QoS and QoE, as it depends on factors such as video encoder, content resolution, type of scene or screen size. Under these conditions, the best approach to characterize the relationship between QoS and QoE is conducting experiments with users in laboratory environments. The standard ITU-R BT.500 [12] describes the
45 methodology for the subjective assessment of the quality of television pictures. ITU-R BT.710 and ITU-T P.910 adapt the above procedures for high-definition displays and multimedia. With these methods, the authors in [13] analyze the impact of bit rate and packet loss rate on the Mean Opinion Score (MOS) for MPEG-2 and MPEG-4 videos. This study is extended in [14] [15] [16] to 3D
50 video sequences in wired and wireless networks, where other aspects has to be considered, such as depth perception and visual fatigue. Other works have focused on the distribution of conventional 2D videos over mobile networks [17]. A survey of research activities in QoE modeling, monitoring and control in wireless networks is presented in [18].

55 In a preliminary work [19], the authors presented a basic system testbed to emulate 3D video distribution in real time over IP. With that basic testbed, it was possible to check with real users the impact of QoS parameters defined at a network level (e.g., packet delay, packet loss rate) and parameters configurable at application level (e.g., encoder, content type) on the QoE measured in terms
60 of Mean Opinion Score (MOS). Its main novelty, compared to similar tools described in the literature (e.g., [20] [21]) was the combination of 3D video traffic

with a mobile network emulator. As a network emulator, the authors chose NetEm [22], which is widely used to produce packet delays, losses, reordering and duplication. As will be shown later, NetEm has severe limitations to model network behavior when the input traffic consists of packet bursts, typical in video streaming traffic. In that preceding paper, a simple configuration of the emulator was tested, based on measurements taken from a live LTE network.

The present work extends and improves the proposal in [19] by taking a closer look on the configuration of the emulation platform. The main contributions here are:

1. a new version of the network emulator code that better models the characteristics of the packet delay when the packet arrival process is bursty, as in video streaming service; for this purpose, the emulator source code is redesigned to include a queuing system to model time correlation between packet arrival, and
2. a new methodology for adjusting the parameters in a mobile network emulator based on the results obtained with a mobile network simulator; such a methodology combines the three above-mentioned evaluation approaches by configuring network emulator parameters from LTE simulations or field measurements taken from a live LTE network.

The rest of the paper is organized as follows. Section 2 outlines the basic network emulation platform for video streaming service over LTE, based on the classical version of NetEm. Section 3 presents the modified version of NetEm that reflects the impact of time correlation between packet arrivals on packet delay statistics. Section 4 explains the LTE network simulator used to configure emulator parameters. Section 5 presents the experimental methodology and the experiments to validate the emulation system testbed. Finally, the main conclusions are presented in Section 6.

2. Basic VideoStreaming-over-LTE Emulation System

90 This section outlines the preliminary version of the emulation platform described in [19]. That basic platform consists of a streaming server, an IP-level mobile network emulator, based on the classical version of NetEm, and a streaming client. The framework is suitable for all kinds of videostreaming services (e.g., standard/high/ultra high definition, 2D/3D, buffered/live streaming,...).
95 In this work, the analysis is focused on 3D live streaming services, as an example of videostreaming application. For clarity, a review of the most common 3D video format is first presented. Then, the general protocol stack of the video streaming service is introduced. Later, the technical specification of the elements in the emulation platform is described. Finally, the problems of the
100 standard version of Netem are discussed.

2.1. 3D Video Formats

The simplest representation format of 3D video is Conventional Stereo Video (CSV), consisting of two independent, yet synchronized, video flows corresponding to a hypothetical viewer's left and right eyes. Both flows can be distributed
105 in parallel using conventional systems, which is known as Simulcast strategy [23]. Such an approach requires no image processing in the spatial domain to provide 2D video with sense of depth, although it doubles the data rate of conventional video flow. Therefore, the use of compression methods becomes necessary to circumvent the limited capabilities of mobile networks and handsets. For this
110 purpose, the most common 3D video formats with reduced data rate are Side-by-Side (SbS) and Multi-View Coding (MVC).

SbS is an example of frame-compatible format, in which the left and right views are sub-sampled and merged into a single frame at the encoder side. Decimation is often done in the spatial domain, either horizontally or vertically,
115 leading to the Side-by-Side (SbS) or Top-and-Bottom (TaB) formats [23]. At the decoder side, de-multiplexing and interpolation is carried out to reconstruct the two views. To facilitate the adoption of these formats, the H.264/AVC standard

introduces a new Supplemental Enhancement Information (SEI) message to signal the frame packing arrangement and the view order [24].

120 Alternatively, MVC format makes use of differential encoding for both views to eliminate redundant information [16]. The decoding process requires several additions to the H.264/MPEG-4 AVC standard [23]. The high-level syntax defined by H.264/MPEG-4 AVC is enriched with a multiview extension of the sequence parameter set (SPS) providing view identification and view dependen-
125 cies. To reduce complexity, MVC does not allow the prediction of a picture in one view at a given time using a picture from another view at a different time. Likewise, MVC encoders can be configured to different levels, each defining constraints on the generated bitstream.

The High Efficiency Video Coding (HEVC) standard [25] has recently been
130 extended to support multiview video (MV-HEVC) and depth-based 3D video formats (3D-HEVC). MV-HEVC [26] allows efficient coding of multiple camera views without changing block-level processing by using inter-view prediction. The more advanced 3D-HEVC [27] provides increased coding efficiency by jointly coding texture and depth information at the expense of introducing new block-
135 level processing modules.

2.2. Video Streaming Distribution Service

For video distribution over IP networks, several combinations of application and transport protocols can be used [28]. The options in the application layer are Hyper-Text Transfer Protocol (HTTP) and Real Time Protocol (RTP),
140 while the most common options in the transport layer are the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). The HTTP-TCP combination including some modifications relative to the standards is often the preferred option, as TCP has built-in congestion control and reliable in-order packet delivery, and, unlike UDP, is firewall friendly. This option is,
145 e.g., the one chosen for YouTube media streaming service, whose mechanism is known as progressive download. In contrast, RTP-UDP is preferred for interactive/conversational type of services due to its lower delays and more efficient

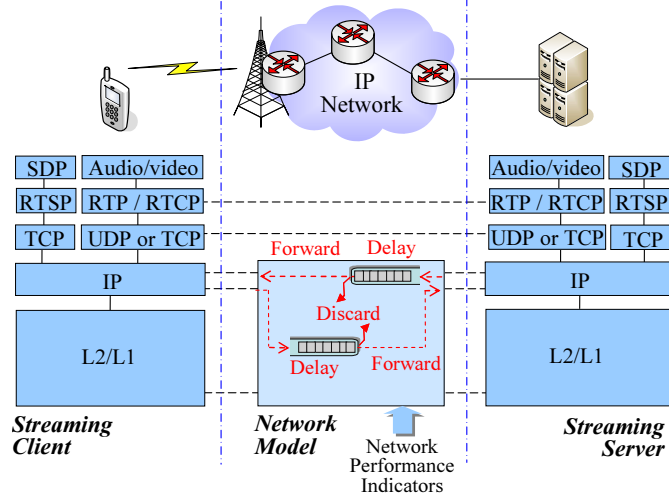


Figure 1: Protocol architecture of the system.

use of bandwidth. For instance, videoconferencing systems, such as Skype, are based on the latter option.

150 The proposed protocol architecture in the emulation platform is shown in Fig. 1. Relations between the client, the network emulator and the server are presented in that figure. The client and the server can run both protocol architectures (i.e., HTTP/TCP or RTP/UDP). For simplicity, in this work, only video streaming service over RTP/UDP is considered hereafter. The network
 155 emulator includes packet losses and delay extracted from the results of a mobile network simulator.

The streaming service consists in sending data continuously through a permanent client-server connection established by means of a Real-Time Streaming Protocol (RTSP). After logging in, video and audio data are normally sent to
 160 two RTP/RTCP ports (data and control) over UDP, although the interleaving mechanism, which entrusts data transport to RTSP, is also available.

2.3. Technical specification of platform components

Fig. 2 shows the pieces of equipment and applications used in the emulation platform. The former consist of the server, the network emulator and two kinds

165 of clients.

The streaming server is a personal computer (PC) with an i5-750 processor at 2.67 GHz and 4 GB of RAM with Ubuntu 14.04 LTS 64-bit operating system, and kernel 3.11.10.4. In this PC, VLC media player v2.1.4 Rincewind [29] is installed as a streaming server software.

170 A first client is a PC with the same hardware characteristics as the server and the same operative system. In this case, VLC media player v2.1.4 Rincewind [29] is installed as a streaming client. VLC is the most widespread media player, since it is free, easy to use, quick, reliable and versatile, and supports a wide variety of codecs. Moreover, it allows recording the decoded sequence on the
175 client side for subsequent processing and analysis by comparing the original and played (i.e., impaired) video sequences.

A second client is an LG Optimus 3D-P920 mobile device with Android 2.2.2 operative system. This terminal has an autostereoscopic display for visualization of 3D video content without the need for 3D glasses. Thus, is possible to check
180 with real observers the impact of network conditions on subjective image quality, depth quality and naturalness. For this purpose VLC media player for Android v0.1.3 is installed in the terminal. Measurements can be done either in real time through the device's Wi-Fi connection or deferred by reproducing the files stored in the PC client with the impaired video sequences after going through
185 the network emulator.

The network emulator is installed on a PC with an i5-3470 processor at 3.2 GHz with 8 GB of RAM and Ubuntu 14.04 LTS 64-bit operating system. This computer has two network cards for connecting the server and the client through it. The connection between the streaming server and the network emulator, and between the network emulator and the client side, is done by a cross
190 twisted pair cable for a completely isolated network. Routing tables are configured by the Linux command `ip route`. The software used as a network emulator is NetEm [22].

To analyze network traffic, Wireshark v1.6.7-1 is selected. Wireshark [30] is
195 the protocol analyzer most widely used by the scientific community to analyze

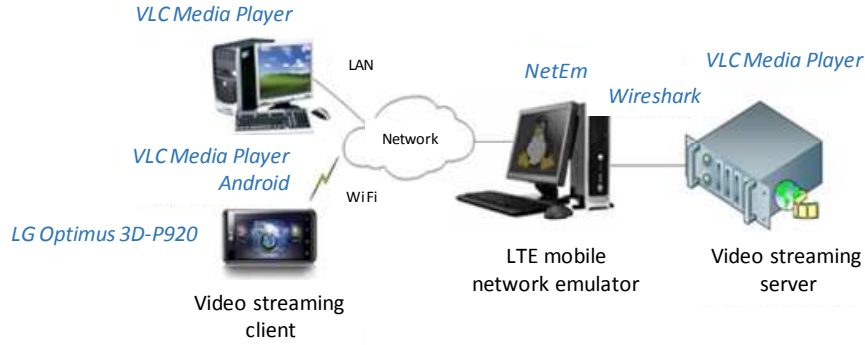


Figure 2: Components in the emulation platform.

and diagnose problems in communication networks. It provides similar features to those provided by tcpdump, yet adding a graphic interface and multiple management and recording filtering options that make it user-friendly. Both server and client run Wireshark to derive transmission performance statistics at the source and destination.

As a network emulator, the platform includes NetEm [22]. This software is an open-source network emulator widely used in evaluation systems for the development of network protocols and applications. It reproduces network dynamics by controlling packet transmission parameters such as delay, jitter and losses, duplicating or eliminating packets. It is explained later how these parameters are set based on simulations carried out with a LTE network simulator.

2.4. The network emulator: NetEm

NetEm is a network emulator that controls delays, losses, duplication and reordering of IP packets [31]. It is included in the Linux kernel since version 2.6. NetEm comprises two parts: a small kernel module for a queueing discipline and a command-line utility for configuration purposes, tc, that is part of Linux's iproute2 package. The latter can be automated using shell scripts for convenience.

Internally, NetEm has two queues: one is a private holding queue, and the other is a nested queue discipline (typically a FIFO). The enqueue interface

takes in packets, timestamps them with a send time, and places them in the holding queue. A timer moves packets from the holding queue to the nested discipline for transmission. The dequeue interface gets packets from the nested queue.

220 In NetEm, packet loss is implemented by randomly discarding a percentage of packets before queuing. The exact value is controlled by a parameter that can be introduced in the command line defining the percentage of packets to be dropped. Burst packet loss emulation is performed by means of one of the available loss models (correlation, four-state model, Gilbert-Eliot model).
225 Packet duplication is carried out by randomly cloning packets before they are queued. It is controlled by specifying the percentage of duplicated packets. Packet reordering is modeled by adding a delay in selected packets, for which a percentage and a packet distance is specified.

In NetEm, delay parameters and their variations (jitter) are described by
230 mean value (μ), standard deviation (σ), and, optionally, a correlation value (ρ). By default, NetEm uses a uniform jitter distribution that can be changed for another predefined distribution function, such as Pareto, Paretonormal, Normal or custom distributions that can be created from experimental or simulation data. The latter option is chosen in this work to configure the network emulator
235 with simulation results.

Although NetEm is an extremely flexible tool, several works have shown that it has important limitations. In [32], it was shown that NetEm is able to generate constant delay higher than 50 ms and uncorrelated packet loss accurately. However, the introduction of jitter on top of the constant base delay is
240 not obvious, as the resulting standard deviation is lower than expected. More importantly, it is shown in [33] that, when NetEm is forced to maintain the packet stream integrity to avoid packet reordering (e.g., to model a first-in first-out system queue), the packet delay distribution changes its shape and main statistics (i.e., average and standard deviation). Such an undesirable behavior
245 makes NetEm inappropriate for modeling real communication systems.

To illustrate the above-mentioned limitations, an experiment is carried out

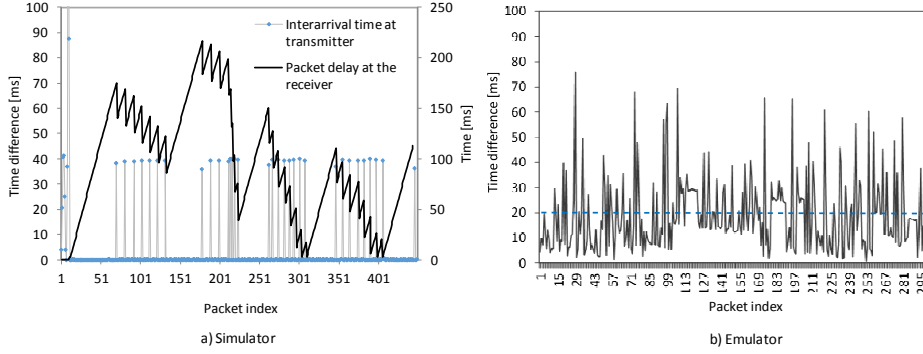


Figure 3: Packet delay in the LTE network simulator and the emulator.

to compare packet delay statistics obtained with NetEm against those obtained from a LTE simulator. The experiment consists of simulating and emulating the same 3D video streaming session, where a particular video sequence is downloaded. Fig. 3 a) shows the outcome of simulation and Fig. 3 b) shows the result of the emulation. For simplicity, it is first assumed in the simulations that the channel delay is constant, equal to 2 ms. Once the behavior of this ideal case is understood, the channel delay in the emulation is randomized.

Fig. 3 a) shows IP-level transmission statistics in the LTE simulator. The x-axis represents the packet index, ranging from 1 (first packet) to 500 (last packet). The grey line with circles represents inter-arrival times (i.e., time difference between successive arrivals) at the transmitter side on the primary axis, while the solid bold line represents the packet delay at the receiver side on the secondary axis.

In the same figure, the analysis of inter-arrival times shows that, after the first 10 packets, which are devoted to the setup of the RTP session, a periodic pattern is observed. A deeper analysis shows that this pattern is originated by the packet arrival process. When downloading a video sequence, many packets arrive simultaneously at the transmitter buffer at the beginning of each frame of the sequence. The 40 ms value observed on the primary y-axis corresponds to the frame period (i.e., the inverse of the video frame rate, 25 Hz). A closer analysis of packet indices on the x-axis reveals that the size of bursts also fol-

lows a periodic pattern, where a large burst of 35-40 packets is followed by 7 smaller bursts of 8-10 packets. This is just a consequence of the Group of Picture (GOP) structure used to code the video sequence, which, in the example,
270 consists of 1 intra-coded frame followed by 7 predictive code frames (i.e., GOP is IPPPPPPP). Such a GOP is an input parameter of the video encoder.

Recall that, in the simulations, it has been assumed that the channel delay is constant. As a result, in Fig. 3 a), a sawtooth profile is observed for the
275 packet delay, showing that transmission delay progressively increases within a frame. This is due to the fact that the last packet of the frame has to wait for previous packets of the same frame to be transmitted. The time gap between frames (i.e., the frame period) helps to reduce packet delay by 40 ms, and hence the sudden drop at the beginning of each frame. However, if the aggregated
280 delay for the last packet of a frame is larger than 40 ms, the first packet of the next frame has to wait. In the example, the transmitter buffer is filled in the larger I frame, and slowly emptied during the subsequent smaller P frames, and hence the triangular shape. Again note that, in Fig. 3 a), the transmission delay is constant, which is the reason for the constant increasing slope. Such an
285 assumption was made to simplify the analysis.

Fig. 3 b) shows packet delay at the receiver side when NetEm is used to emulate the same video streaming session. For this purpose, NetEm is configured without packet integrity (i.e., packet reordering is possible if the delay of a packet is less than the previous packet). The delay in NetEm is configured
290 to a random value with average 20 ms. For clarity, interarrival times have been removed from the figure, as they are the same of Fig. 3 a). It can be noted that no pattern is observed in the delay values. On the contrary, in Fig. 3 b), packets with large delays are followed by packets with small delays within the same frame. This is just a consequence of the absence of a queue that builds
295 up the delay experienced by packets of the same frame. Not shown here is the fact that enforcing packet integrity does not solve the problem [33].

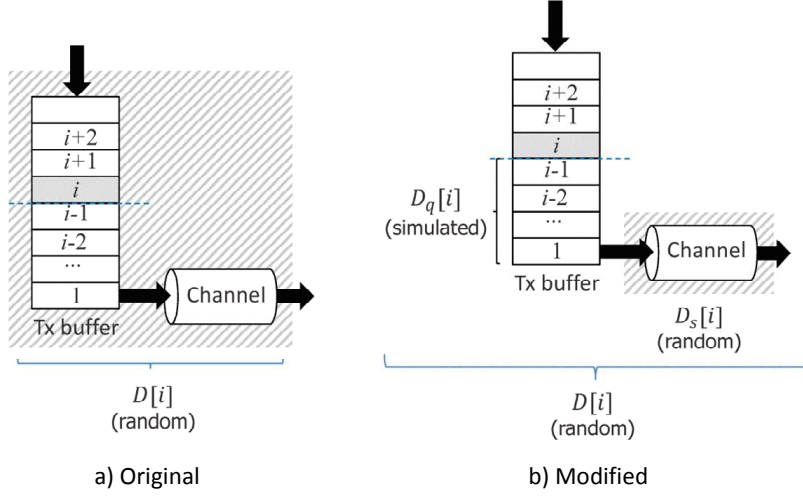


Figure 4: Modeling of packet delay in the original and modified versions of NetEm.

3. Modified Network Emulation Code

In this work, the NetEm source code is modified to model the accumulation of delay that occurs when a bunch of packets arrive simultaneously at the beginning of each video frame.

Fig. 4 compares the different approaches used to model packet delay in the original and modified versions of the network emulator. In both approaches, the aim is to model the packet delay introduced by the whole distribution network. In both subfigures, the network is represented as a queueing system, consisting of a server (i.e., the transmission channel) and a queue (i.e., the transmitter buffer). As in any queue, the system time is the sum of the waiting time and the service time. In this case, the system time for the packet i is the total packet delay, $D[i]$, the waiting time is the buffer time, $D_q[i]$, and the service time is the channel transmission delay, $D_s[i]$.

In the original NetEm, shown in Fig. 4 a), the packet delay for packet i , $D[i]$, is modeled as a whole, by generating a single random value per packet from a predefined statistical distribution (in this work, taken from the LTE simulator). Depending on the value of $D[i]$, packets might be reordered if no action is taken.

In the modified version of NetEm, shown in Fig. 4 b), the packet delay
 315 for packet i , $D[i]$, is divided in two different components, which are modeled
 separately: the waiting time, $D_q[i]$, and the service time, $D_s[i]$. The service
 time, $D_s[i]$, is still generated randomly from a probability distribution built with
 channel delay statistics taken from the LTE simulator. However, the waiting
 time, $D_q[i]$, is obtained by explicitly simulating the queue, unlike in the original
 320 version of NetEm. With these changes, the new NetEm code is able to model the
 increase of packet delay due to the time correlated arrivals, while maintaining
 the statistical properties of transmission delay typical of wireless systems. The
 simulation of the queue does not affect relations with other components of the
 unmodified NetEm software.

325 The proposed changes are introduced by modifying the function that handles
 packet queueing (*netem_enqueue* function) in NetEm. First, the service time,
 D_s , is introduced as a custom delay distribution table that models the proba-
 bility distribution of the Empirical Cumulative Distribution Function (ECDF)
 obtained from the LTE simulator. This ECDF is one of the arguments of the
 330 *tabledist* function. Then, when the packet has to be served, if the packet does
 not find the queue empty, the waiting time is added to the service time. Thus,
 the stream packet integrity (i.e., packet order) is maintained, as no packet can
 be transmitted before the previous one (i.e., the transmission instant of a packet
 is always later than that of the preceding packet in arrival order).

335 4. LTE Radio Interface Simulation

Any new feature introduced in mobile networks is first validated through
 simulations. Depending on the purpose of the feature, simulations can be per-
 formed on the physical layer by a *link-level* simulator or on the network layer
 by a *system-level* simulator. While link-level simulations are focused on the
 340 performance of multi-antenna techniques and modulation and coding schemes,
 system-level simulations are focused on network-related issues, such as schedul-
 ing, mobility handling or interference management.

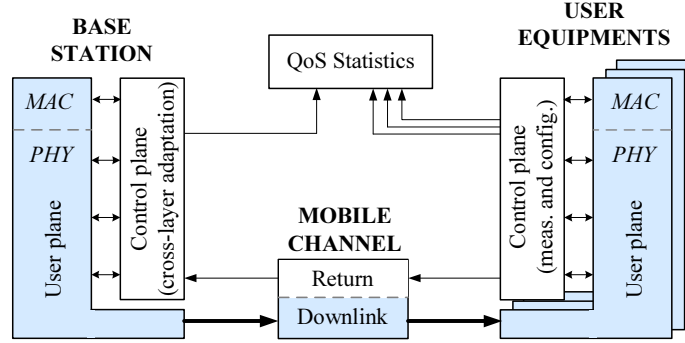


Figure 5: Simplified LTE Simulator Architecture(for 1 eNode B and several UEs) in the downlink [34]

In this work, a simulation campaign was performed with a dynamic system-level LTE simulator [34] to configure the network emulator with realistic values. The goal of simulations is to estimate the performance of packet data transmission in the radio access network with different radio conditions.

Fig. 5 shows a simplified architecture of the LTE simulator (further details can be found in [34]). The physical layer implementation of this LTE simulator is compliant with the 3GPP-LTE specifications. Adaptive Modulation and Coding (AMC) can be used to keep instantaneous Block Error Rate (BLER) below a given target value. An Outer Loop Link Adaptation (OLLA) scheme is also implemented. Hybrid Automatic Repeat reQuest (HARQ) is used as high-rate forward error-correcting coding and ARQ error-control. Variable coding rate is achieved by applying different puncturing patterns depending on the current H-ARQ incremental redundancy (IR) version. The turbo decoder uses a Maximum a Posteriori (MAP) decoding algorithm with a maximum of 8 decoding iterations. At the MAC layer, several cross-layer scheduling algorithms are available to allocate time-frequency resources to the different users. The scheduler decisions are based on the per-user reported PHY measurements as Channel Quality Indicator (CQI), Rank Indicator (RI), pending HARQ retransmissions, and Precoding Matrix Indicator (PMI). Other considerations such as the terminal speed or Quality of Service (QoS) requirements (e.g., priority handling)

can be also taken into account in the scheduling process. The simulator also includes a module responsible for collecting Quality of Service (QoS) statistics. WM-SIM results can be saved into files that are fully compatible with Matlab format (MAT files) making it possible to take advantage of its potential for off-line post-processing. Results available by default are closely related to Quality of Service (QoS) functionality (e.g., BER, BLER, delay, throughout).

The scenario simulated in this work consists of 53 cells with omnidirectional antennas, located in the center of each cell and distributed on a regular basis in space. Several 3D video streaming users are distributed per cell. The main simulation parameters are described in Table 1. For simplicity, only the downlink is simulated, which is the main bottleneck in the download of video sequences from a server.

The main output generated by the simulator is a trace file with the arrival time and total delay (including queuing and transmission delay) of IP packets in the downlink for a user in certain conditions. Four different network conditions are defined, which reflect the experience of a user located in the center or the edge of the cell, under low or high network load. User position determines the signal quality (i.e., the signal-to-interference-plus-noise ratio, SINR) experienced by the user, while network load determines the number of radio resources available to the user. The user position is achieved by setting the user in the center or cell edge. For each user position, two network load cases are simulated, corresponding to a Physical Resources Block (PRB) utilization ratio of 25% (low load) and 75% (high load). In the experiments, the network load is controlled by modifying the number of simultaneous active video users per cell.

Fig. 6 presents the ECDF of the total packet delay, D , in the four use cases considered: low load with user in the center and in the border and high load with user in the center and in the border of the cell; Table 2 details some relevant packet transmission statistics for each case. It is observed that the median of the packet delay ranges from 8 ms for the best case (user in cell center and low load) to 200 ms for the worst case (user in edge cell and high load).

For each use case, the trace with the arrival time and total delay of each

Table 1: LTE network simulator parameters.

Carrier Frequency	2.5 GHz
System Bandwidth	10 MHz (50 Physical Resource Blocks)
No. of cells	53
Cell Radius	1.5 km
Propagation Model	Hata
	Slow fading lognormal, std. deviation 6dB
Effective Isotropically	
Radiated Power (EIRP)	43 dBm
Noise Power Density	-174 dBm/Hz
Scheduler	Proportional Fair
MIMO transmission scheme	2x2 Beamforming
Channel Model	Extended Pedestrian A (EPA), 3 km/h
Channel Decoding Method	Soft-Output Viterbi Algorithm
Target BLER	10%
Service Model	Real 3D Video Sequence (Bullinger)
	H.264-MPEG4 AVC SbS format
	432x240 pels 4:2:0
	Joint Video Team (JVT) JM v18.5 codec [35]
	Avg. bitrate: 3 Mbps
	Avg. packet size: 1340 Bytes
	Avg. packets per second: 280 pps
Network load	25 or 75%
Simulation Time	40s (≈ 11200 packets)
Time Resolution	1ms (Time Transmission Interval - TTI)

packet from the simulations is processed to obtain the queueing and transmission
 395 delay of each packet. Specifically, the transmission delay of packet i , $D_s(i)$, is
 estimated from the total delay, $D(i)$, as

$$D_s(i) = D(i) - D_q(i) = D(i) - \max(T(i-1) - A(i), 0), \quad (1)$$

where $D_q(i)$ is the queueing delay of packet i , calculated from the difference
 between the instant of the previous packet transmission, $T(i-1)$, and the arrival
 time of the packet, $A(i)$. Note that $D_s(i)$ depends on both packet size and radio
 400 link conditions.

Table 2 shows the average values of the different delay components in each
 use case. It is observed that most of the delay is due to the queueing delay in all
 cases. A detailed analysis shows that this behavior is due to the simultaneous
 arrival of packets at the beginning of each frame in the video sequence.

405 Fig. 7 shows the ECDF of the transmission delay, D_s , for each of the four
 use cases. It is observed that the median ranges from 1.17 ms for the best case
 (user in cell center and low load) to 6.5 ms for the worst case (user in cell edge
 and high load).

The ECDFs shown in Fig. 7 are used to generate four tables with 4096 ran-
 410 dom delay values. These values, once converted to tick units (1 tick = 64 μ s)
 and ordered from lowest to highest, are introduced in a table of 512 * 8 compo-
 nents that NetEm uses to generate the transmission delay, D_s , in each case use
 [36]. NetEm code has been modified so that standard input parameters, such
 as mean and standard deviation of delay, are overridden by the ECDF table.

415 5. Validation of Emulation Platform

In this section, different experiments are carried out to validate the changes
 introduced in NetEm.

5.1. Analysis setup

Fig. 8 summarizes the experimental methodology. First, the network emu-
 420 lator is configured with the ECDF of the transmission delay for the considered

Table 2: Results of simulation tests.

Network Load	Low		High	
User Position	Center	Edge	Center	Edge
Average total delay, D [ms]	14.96	48.80	30.90	4542.9
Average queueing delay, D_q [ms]	12.48	45.77	28.70	4258.8
Average transmission delay, D_s [ms]	1.20	3.03	2.48	284.1
Average Jitter [ms]	3.50	5.18	3.81	2.79

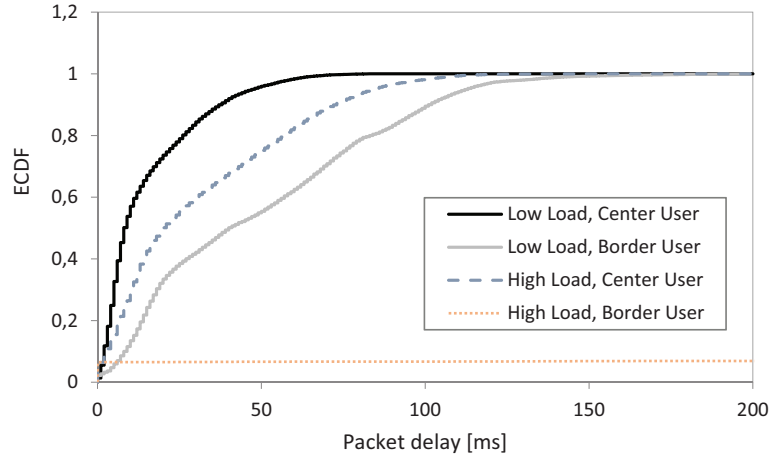


Figure 6: Distribution of packet delay, D , for different use cases.

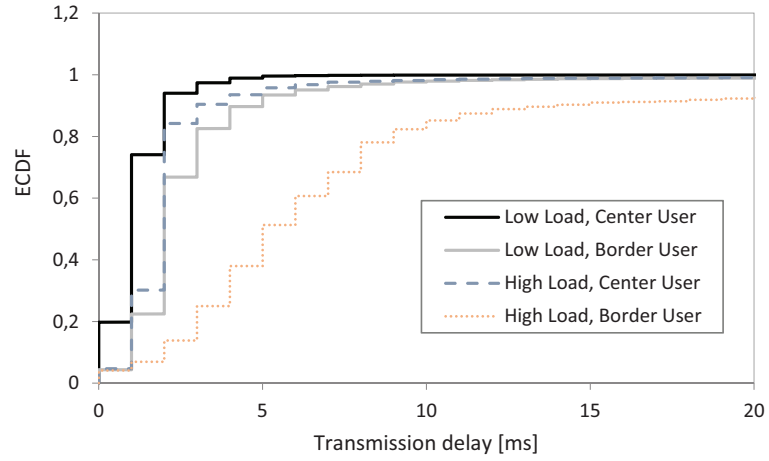


Figure 7: Distribution of transmission delay, D_s , for different use cases.

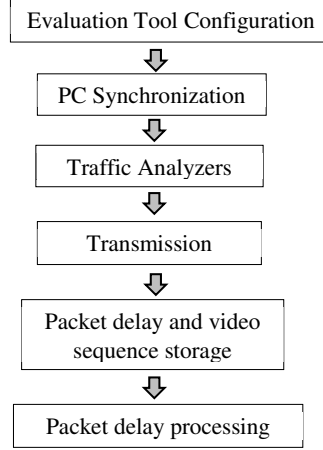


Figure 8: Experimental methodology.

use case (user position and network load), obtained from the system-level LTE simulator. Second, the VLC client is configured by selecting the 3D video sequence tested, the streaming protocols and the receiver buffer size. Third, the Network Time Protocol (NTP) is used to synchronize the three computers in the platform (client, server and network emulator) to a reference time source. Fourth, Wireshark is launched at the server and client sides to capture traffic traces. Fifth, the video transmission starts from VLC Streaming Server to VLC media player. Sixth, packet traces at the server and client are saved, as well as the recovered video at the client. Finally, delay components are computed from packet delay obtained from traces.

Table 4 describes some relevant parameters in the tests, such as the selected video sequences as well as NetEm and VLC settings.

Four stereoscopic sequences are selected, which are widely used in other studies related to 3D video [16]. They are chosen according to recommendations in [37], which suggest the inclusion of sequences with animated images, fast movements in the image and real images with people. In addition, video content should avoid discomfort and visual fatigue. A brief description of each sequences is shown in Table 3.

Table 3: Sequences characteristics.

Name	Format	Frames/s	Content
Flower	480 * 270	25	Flower in slow motion
Moabit	512 * 384	25	Urban road with traffic
Rope	432 * 240	25	Animated hopping rabbit
Bullinger	432 * 240	25	Static man talking

Table 4: Configuration setup.

Sequence	Bullinger, 432x240 pixels, 4:2:0
Video Codec	H.264/AVC SbS, 3086 kbps
Duration [s]	40
NetEm Parameters	[ECDF for each use case]
VLC Parameters	Receiver buffer size: 50 ms

The client buffer size is set to 50ms, which is suitable for conversational video services, for which end-to-end delay should not be higher than 150 ms [38].

Fig. 9 exhibits a frame of the Bullinger video sequence [39], with the left and right views. It consists of a single shot of a man talking to the audience without any camera movement. The 3D scene has low object movement, low structural complexity and low depth complexity, which might be representative of a videoconference. The streaming protocol architecture based on RTP/UDP is selected by entering the appropriate URL (rtsp://server.example.org:8080/Bullinger.mp4) in VLC.

To emulate the LTE scenario, NetEm is configured according to the above-described four scenarios (high/low load and user in cell center/edge). For this purpose, NetEm is configured with the specific table that models the transmission delay from those shown in Fig. 7.

Wireshark is used to collect traffic data. In principle, its RTP Analysis feature should be able to measure mean delay, jitter and packet loss in an RTP data flow. However, tests have shown that Wireshark displays no jitter or mean delay results appropriately, since it is unable to interpret RTP flows correctly.



Figure 9: Image from Bullinger video [35].

Alternatively the delay is read within Wireshark's ExpertInfo. Detailed delay statistics for the RTP traffic are calculated on a per-packet basis by comparing the RTP packet header timestamp, showing the transmission time, and the reception time, stored by Wireshark on the receiver side. More precisely, the
460 delay difference between two packets i and j , $D(i, j)$, experienced by an RTP flow is calculated as in RFC 3550 [40]:

$$D(i, j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i), \quad (2)$$

where S_i and S_j are the RTP timestamp of packets i and j , and R_i and R_j are the arrival times of packets i and j in timestamp RTP units, and the jitter between arrivals, $J(i)$, as

$$J(i) = J(i - 1) + (|D(i - 1, i)| - J(i - 1))/16, \quad (3)$$

465 where $J(i - 1)$ is packet $i - 1$'s jitter, and $D(i - 1, i)$ is the difference in transmission and reception time of packets i and $i - 1$ in order of arrival (not necessarily consecutive).

Finally, end user quality is measured by presenting the received video sequences on the screen of the VLC player of the client. To get an objective
470 measure of the impact of delay fluctuations on video quality, the VLC functionality that allows recording the decoded sequence is used. These saved sequences are then processed with *ffmpeg* [41] to assess the quality of the sequence in terms of the number of missing and duplicate pictures.

5.2. Results

475 This section shows the results of the experiments carried out to validate the changes introduced in NetEm. For this purpose, an example is first given to qualitatively show the potential benefit of changes by visual inspection. Then, detailed measurements are presented to prove that the new network emulator behaves very similar to the LTE simulator. For brevity, only the results obtained
480 for one video sequence (i.e., Bullinger) are shown.

Fig. 10 shows an example of packet delay trace obtained by connecting video streaming client and server through different versions of NetEm, i.e., the classical vs the modified one. For a fair comparison, both traces correspond to the same video sequence and both emulators are configured with the same delay
485 parameters (average and standard deviation). The grey curve corresponds to the original NetEm, while the blue curve corresponds to the modified NetEm. It is observed that the trace obtained with the classical NetEm fails to model the increase of delay caused by consecutive packets arriving simultaneously at the beginning of a frame, as was already shown in Fig. 3 b). In contrast, it is
490 observed that the trace obtained with the modified NetEm reflects the build up of delay observed in the LTE network simulators, shown in Fig. 3 a). From the shape of the curve, it is easy to deduce the GOP structure (IPPPPPPP) and the frame period (40 ms). Note that, in this case, the transmission delay is not fixed, but fluctuates, and hence the changes in the slope of the curve. Such changes
495 are mainly due to the variable channel capacity. From these results, it can be concluded that the proposed changes have a potential benefit on emulation.

To measure emulation accuracy, a quantitative analysis is presented based on the comparison of delay statistics obtained with the network emulator and the LTE simulator. The comparison includes the four use cases (i.e., cell center/edge
500 user, low/high network load). The tests reported here only consider a single video sequence, since similar results are obtained with other sequences.

Fig. 11 presents the ECDFs of end-to-end packet delay obtained by the network emulator (modified NetEm, dashed lines) compared to the one from the LTE simulator (solid lines). In the figure, the x-axis has been enlarged

505 to see the differences between the curves of lower delay. Note that, in the case of high load and cell-edge user, the queuing system is unstable, as arrival packet rate is greater than output packet rate, causing an increase of the delay. Therefore, the curve is displaced to the right and is not visible. From Fig. 11, it can be concluded that the curves of the simulator and the emulator fit almost
510 perfectly in the cases of low load-cell center user and high load-cell center user center, while the adjustment is less accurate, but still acceptable, in the case of low load-cell edge user.

Table 5 summarizes the main packet transmission statistics. Again, it is seen that the values obtained by the emulator and the simulator are really close. In
515 the case of low load and cell-center user, the relative error in the mean delay is lower than 0.03%. In the case of high load and cell-center user, the relative error is less than 4%, while, for low load and cell-edge user, is 10%. These results are consistent with those observed in Fig. 11. In the case of high load and cell-edge user, where the system is unstable, the deviation in the mean delay is
520 considerable.

In Table 5, it is also noted that jitter values are in the order of the transmission delay (once the packet is the first one in the queue) in all cases. As expected, the difference of the total delay between two consecutive packets of the same burst (video frame) approaches the mean transmission delay, as can
525 be deduced from the Fig. 7. By comparing the values of the simulator and the emulator, it follows that the relative error in jitter is below 5% in all cases when the queue system is stable.

Finally, Table 6 presents several performance indicators associated with the video decoding process. These are obtained by processing the recorded sequence
530 with *ffmpeg*. In the table, it is seen that the number of duplicate or lost pictures in the three cases where the system is stable is less than those obtained when the system is unstable. However, no important conclusion can be drawn from the values in the table, as the number of lost/duplicate pictures is very small in any case, which makes them dependent on the specific realization of the random
535 variables in the experiment. Similar results are obtained for the other sequences

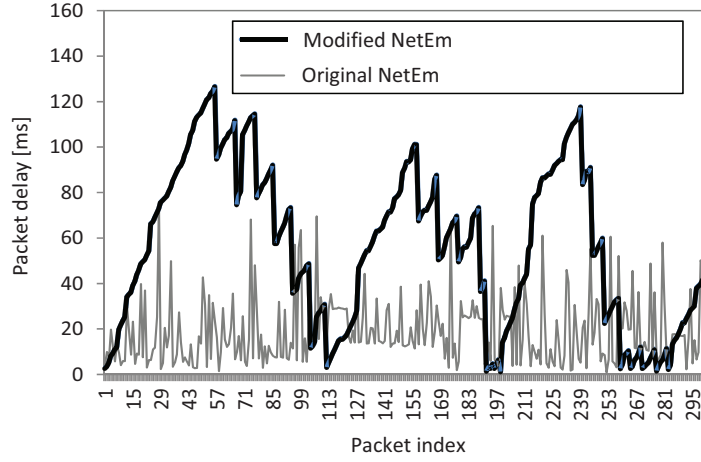


Figure 10: Packet delay trace with different versions of NetEm.

Table 5: Packet delay and jitter for the four scenarios.

Network Load	Low		High	
User Position	Center	Edge	Center	Edge
Average total delay (LTE sim.) [ms]	15.16	48.91	31.77	4857
Average total delay (NetEm) [ms]	14.19	43.71	33.06	21278
Average jitter (LTE sim.) [ms]	2.09	4.42	3.86	9.86
Average jitter (NetEm) [ms]	2.00	4.53	3.77	6.43

(i.e., Flower, Moabit and Rope).

Subjective tests show almost no difference between the three cases where the system is stable. In contrast, in the case of high network load and cell edge user, the player stops after a few frames.

540 6. Conclusions

As the complexity of mobile networks increases, emulating these networks precisely is key to understand the impact of the underlying processes on the end-user experience. This work has presented an emulation platform for the evaluation of video streaming services over LTE. The applications integrated

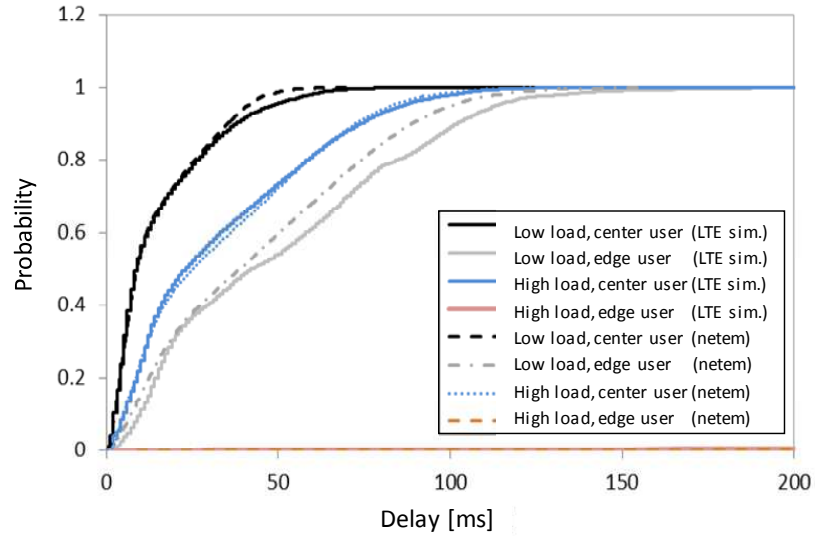


Figure 11: Packet delay distribution in the network emulator and LTE simulator.

Table 6: Video transmission performance (from *ffmpeg*).

Network Load	Low		High	
User Position	Center	Edge	Center	Edge
Duplicated frames	16	12	46	585
Lost frames	3	0	0	6

545 in the platform are VLC as a video streaming server and client, NetEm as a network emulator, and Wireshark as a monitoring tool.

To capture the impact of the mobile network on packet delay, NetEm source code has been modified. Unlike the original version, where the packet delay is introduced as a whole, in the proposed code, the delay is separated in two
550 components, namely the queuing delay and the transmission delay. The latter delay component is obtained from packet delay distributions computed by a system-level LTE simulator configured to model realistic cellular scenarios.

Results show that the evaluation tool models fairly accurately the mean packet delay and jitter under different radio conditions. More specifically, the
555 relative error of the average packet delay and jitter, compared to the values from the LTE simulator, is less than 10% and 5%, respectively, when the system is stable (i.e., the average data transmission rate exceeds the average encoding video rate).

It is planned to extend the evaluation tool to include new video encoding
560 formats and objective video quality measurements (e.g., PSNR, MS-SSIM). In addition, to extent the QoE analysis over 3D video a measurement campaign will be developed to check the subjective quality on a mobile terminal with autostereoscopic display.

Acknowledgments

565 This work was funded by the Spanish Ministry of Economy and Competitiveness (grants TEC2015-69982-R and TEC2013-44442-P). In addition, the authors wish to acknowledge the collaboration of Almudena Sanchez Martinez and Francisco Blaquez Casado in carrying out the experiments.

References

570 [1] Ericsson Mobility Report, Ericsson, 2015.

URL <http://www.ericsson.com/res/docs/2015/mobility-report/ericsson-mobility-report-1>

- [2] A. Banerjee, Revolutionizing CEM with subscriber-centric network operations and QoE strategy, White paper, Heavy Reading, 2014.
- [3] Top 10 Technologies for Mobile Broadband Traffic Management, Disruptive Analysis, 2010.
- [4] E. Rescorla, SSL and TLS: Designing and Building Secure Systems, Addison-Wesley Pub Co., 2001.
- [5] I. Sodagar, The MPEG-DASH standard for multimedia streaming over the internet, IEEE MultiMedia 18 (4) (2011) 62–67.
- [6] QUIC protocol official website, last accessed: 03-Apr-2015.
URL <https://www.chromium.org/quic>
- [7] M. Fiedler, T. Hossfeld, P. Tran-Gia, A generic quantitative relationship between quality of experience and quality of service, IEEE Network 24 (2) (2010) 36–41.
- [8] Opinion model for video-telephony applications, ITU-T Recommendation G.1070, 2012.
- [9] Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference, ITU-T Recommendation J.144, 2004.
- [10] T. Rahrer, R. Fiandra, S. Wright, Triple-Play Services Quality of Experience (QoE) Requirements, DSL Forum TR-126, 2006.
- [11] B. Erman, E. P. Matthews, Analysis and realization of IPTV service quality, Bell Labs Technical Journal 12 (4) (2008) 195–212.
URL <http://dx.doi.org/10.1002/bltj.20276>
- [12] Methodology for the Subjective Assessment of the Quality of Television Pictures, ITU-R Rec. BT. 500-11, 2002.

- [13] M. H. Pinson, S. Wolf, G. Cermak, HDTV subjective quality of H.264 vs. MPEG-2, with and without packet loss, *IEEE Transactions on Broadcasting* 56 (1) (2010) 86–91.
- 600 [14] I. Kulik, T. A. Trinh, *Energy-Aware Communications: 17th International Workshop, EUNICE 2011, Dresden, Germany, September 5-7, 2011. Proceedings*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, Ch. Investigation of Quality of Experience for 3D Streams in Gigabit Passive Optical Network, pp. 157–168.
- 605 [15] Z. Liu, Y. Qiao, B. Lee, E. Fallon, K. AK, C. Zhang, S. Zhang, Experimental evaluation of H.264/multiview video coding over IP networks, *Irish Signal and Systems Conference (ISSC)*, Trinity College Dublin, 2011.
- [16] C. Jean, A. Melo, A. Douglas, A. Abelém, E. Cerqueira, A. Neto, A. Riker, M. Curado, QoE-based packet drop control for 3D-video streaming over
610 wireless networks, in: *Proceedings of the 7th Latin American Networking Conference, LANC '12*, ACM, New York, NY, USA, 2012, pp. 59–66.
- [17] S. Singh, O. Oyman, A. Papathanassiou, D. Chatterjee, J. G. Andrews, Video capacity and QoE enhancements over LTE, in: *2012 IEEE International Conference on Communications (ICC)*, 2012, pp. 7071–7076.
- 615 [18] S. Barakovic, L. Skorin-Kapov, Survey and challenges of QoE management issues in wireless networks, *Journal of Computer Networks and Communications*, 2013.
- [19] I. Palomo, M. Solera, M. Toril, J. Poncela-Gonzalez, G. Gomez, Maqueta de sistema para la evaluacin del servicio de video streaming 3D sobre LTE,
620 in: *URSI 2014, Valencia (Spain)*, 2014.
- [20] G. Gomez, J. Poncela-Gonzalez, M. C. Aguayo-Torres, J. T. Entrambasaguas, *Future Multimedia Networking: Second International Workshop, FMN 2009, Coimbra, Portugal, June 22-23, 2009. Proceedings*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, Ch. A Real-Time End-to-End

- 625 Testbed for Evaluating the Performance of Multimedia Services, pp. 212–217.
- [21] S. P. Uppu, QoE of Video Streaming over LTE Network, MS Thesis, 2013.
- [22] Linux foundation website, last accessed Mar 2016.
URL <http://www.linuxfoundation.org/collaborate/workgroups/networking/netem>
- 630 [23] A. Vetro, A. M. Tourapis, K. Muller, T. Chen, 3D-TV content storage and transmission, IEEE Transactions on Broadcasting 57 (2) (2011) 384–394.
- [24] ITU-T and ISO/IEC JTC 1, Advanced video coding for generic audiovisual services, ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), 2010.
- 635 [25] G. Tech, Y. Chen, K. Mller, J. R. Ohm, A. Vetro, Y. K. Wang, Overview of the multiview and 3d extensions of high efficiency video coding, IEEE Transactions on Circuits and Systems for Video Technology 26 (1) (2016) 35–49. doi:10.1109/TCSVT.2015.2477935.
- [26] G. Tech, K. Wegner, Y. Chen, M. Hannuksela, J. Boyce, MV-HEVC Draft
640 Text 9, ISO/IEC JTC1/SC29/WG11, MPEG/m34571, JCT3V-I1002, Sapporo, Japan, Jul 2014.
- [27] G. Tech, K. Wegner, Y. Chen, S. Yea, 3D-HEVC Draft Text 7, ISO/IEC JTC1/SC29/WG11, MPEG/m36127, JCT3V-K1001, Geneva, Switzerland, Feb 2015.
- 645 [28] C. G. Gurler, B. Gorkemli, G. Saygili, A. M. Tekalp, Flexible transport of 3-D video over networks, Proceedings of the IEEE 99 (4) (2011) 694–707.
- [29] Videolan website, last accessed Mar 2016.
URL <http://www.videolan.org/vlc/>
- [30] Wireshark website, last accessed Mar 2016.
650 URL <https://www.wireshark.org/>

- [31] S. Hemminger, Network emulation with netem, Open Source Development Lab, 2005.
- [32] A. Jurgelionis, J. P. Laulajainen, M. Hirvonen, A. I. Wang, An empirical study of netem network emulation functionalities, in: Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on, 2011, pp. 1–6.
- [33] J. Sliwinski, A. Beben, P. Krawiec, Empath: Tool to emulate packet transfer characteristics in IP network, in: Proceedings of the Second International Conference on Traffic Monitoring and Analysis, TMA’10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 46–58.
- [34] G. Gomez, D. Morales-Jimenez, J. J. Sanchez-Sanchez, J. T. Entrambasaguas, A next generation wireless simulator based on MIMO-OFDM: LTE case study, EURASIP Journal on Wireless Communication and Networking 2010 (2010) 15:1–15:12.
- [35] 3D Bullinger video.
URL <http://sp.cs.tut.fi/mobile3dtv/video-plus-depth/>
- [36] How to create NetEm distribution tables.
URL <http://piao-tech.blogspot.com.es/2009/10/how-to-create-netem-distribution-table>
- [37] M. H. Pinson, M. Barkowsky, P. Le Callet, Selecting scenes for 2d and 3d subjective video quality tests, EURASIP Journal on Image and Video Processing 2013 (1) (2013) 50.
doi:10.1186/1687-5281-2013-50.
URL <http://dx.doi.org/10.1186/1687-5281-2013-50>
- [38] 3GPP TS 23.203, v13.6.0, Policy and charging control architecture, 2015.
- [39] HHI Fraunhofer Institute, Developer group of JM 18.5 coding/decoding tool.
URL <http://iphone.hhi.de/suehring/tml/download/>

[40] IETF, RFC 3550, RTP: A Transport Protocol for Real-Time Applications, 2003.

[41] ffmpeg website, last accessed Mar 2016.

680 URL <https://www.ffmpeg.org/>