Testing the IQX Hypothesis for Exponential Interdependency between QoS and QoE of Voice Codecs iLBC and G.711

Tobias Hoßfeld*, David Hock*, Phuoc Tran-Gia*, Kurt Tutschku*†, Markus Fiedler‡
*University of Würzburg, Institute of Computer Science, Department of Distributed Systems, Würzburg, Germany
Email: {hossfeld,hock,trangia}@informatik.uni-wuerzburg.de
†now at NICT Network Architecture Group, Network Research Center Park Court Place, Tokyo, Japan
Email: tutschku@nict.go.jp
‡Blekinge Institute of Technology, Department of Telecommunication Systems, Karlskrona, Sweden
Email: markus.fiedler@bth.se

***Abstract*— Given the growing importance of quantitative relationships between user-perceived Quality of Experience (QoE) and network Quality of Service (QoS), this paper investigates the IQX hypothesis for two voice codecs, iLBC and G.711. This hypothesis expresses QoE as an exponential function of QoS degradation. The experiments are carried out in a controlled environment using the softphone SJPhone, the network emulator NIST Net, and a tool calculating the PESQ (Perceptual Evaluation of Speech Quality) from sent and received audio files. The IQX hypothesis is confirmed exactly for disturbances perceived on applications level, packet loss and packet reordering, which clearly correlate to the main sensitivities of the used softphone to packet-level disturbances such as loss, jitter and reordering. So, besides of providing a unified relationship between QoE and QoS, the IQX also proved to be capable of identifying the QoS parameters of relevance for QoE degradations. The study also points out interesting tracks for future work in terms of QoS degradations and related QoE evaluations.**

I. Introduction

User satisfaction with application and service performance in communication networks has attracted increased attention during the recent years. The interest in how the user perceives usability, reliability, quality and price-worthiness as a means of competition is increasing. The network and service providers need to be able to observe and react upon quality problems, at best before the customer perceives them.

The notion of QoE was introduced in several white papers, mostly in the context of multimedia delivery like IPTV. Besides of objective end-to-end QoS parameters, QoE focuses on subjective valuations of service delivery by the end users. The necessity of introducing QoE can be explained on the example of VoIP. A voice user is not interested in knowing performance measures like packet loss or received throughput, but mainly in the experienced speech quality and timeliness of the connection setup.

There is however still a lack of quantitative descriptions or exact definitions of QoE. One particular difficulty consists in matching subjective quality perception to objective, measurable QoS parameters. Subjective quality is amongst others expressed through *Mean Opinion Scores* (MOS) [1]. Links between MOS and QoS parameters exist predominately for packetised voice such as VoIP. Numerous studies have performed measurements to quantify the effects of individual impairments on the speech quality on a single MOS value for different codecs, for example G.729 [2], GSM-FR [3], iLBC used by Skype [4], or a comparison of some codecs [5]. Additionally, the E-model [6] and related extensions [7] assess the combined effects of different influence factors on the voice quality. In [8], the logarithmic function is selected as generic function for mapping the QoE, there denoted as user level QoS, from a single parameter because of the mathematical characteristics of the logarithmic function.

In this work in contrast, we motivate a fundamental relationship between the QoE and quality impairment factors such as packet loss or jitter. As an analytical solution of this relationship between QoE and QoS, we formulated the IQX hypothesis (exponential interdependency of QoE and QoS) in [9] which will be briefly reviewed in Section II.

After that, the hypothesis is tested for two different voice codecs, iLBC and G.711. Using a common VoIP application supporting both voice codecs, we conducted a set of measurements to describe the QoE in terms of mean opinion scores (MOS) depending on QoS parameters. In our test bed, we are able to control the network conditions and to inject for example packet loss or jitter. Packet traces are captured to measure the QoS parameters. The received audio signals are compared to the originally sent audio signals, the PESQ (Perceptual Evaluation of Speech Quality) is calculated and mapped to MOS as QoE indicator.

As stated before, the goal of this work to further quantify the relationship between QoE and QoS impairment factors. In particular, we investigate the impact of uncorrelated and correlated delay and jitter, packet reordering, random packet losses, and bursty losses. We test the IQX hypothesis, i.e. the exponential interdependency between QoS and QoE, and show that we can confirm the hypothesis when appropriate metrics to describe the QoS impact on application layer are chosen. Our methodology comprises measurements in a test bed with

controlled network conditions and optimal parameters of the mapping function are retrieved using nonlinear regression techniques.

This paper is organized as follows. In Sec. II, we briefly review the IQX hypothesis which is fundamental for this work. The applied methodology is illustrated in Sec. III which includes the setup of the test bed, the computation of the QoS and QoE parameters, and the derivation of the exponential mapping function. Furthermore, we take a close look at the used network emulator and check its functionality. The main contribution of our paper will be in Section IV, where we present our measurement studies and test the IQX hypothesis for several QoS parameters in different scenarios. Finally, we will conclude this paper with an outlook on future work.

II. The IQX Hypothesis

The QoE can be expressed as a function of n influence factors $I_j, 1 \leq j \leq n$:

$$QoE = \Phi(I_1, I_2, \cdots, I_n) \ . \tag{1}$$

However, in this contribution we focus on single influence factors indicating the QoS, like the packet loss ratio, in order to motivate the fundamental relationship between the QoE and an impairment factor corresponding to the QoS. The idea is to derive the function $QoE = f(QoS)$ with a single impairment factor $I = QoS$.

In general, the subjective sensibility of the QoE is the more sensitive, the higher this experienced quality is. If the QoE is very high, a small disruption will decrease strongly the QoE, also stated in [8]. On the other hand, if the QoE is already low, a further disturbance is not perceived in a significant way. This relationship can be motivated when we compare with restaurant quality of experience. If we dined in a five-star restaurant, a single spot on the clean white table cloth strongly disturbs the atmosphere. The same incident appears much less severe in a beer pub.

On this background, we assume that the change of QoE depends on the current level of QoE – the expectation level – given the same amount of change of the QoS value. Mathematically, this relationship can be expressed in the following way. The performance degradation of the QoE with respect to a certain QoS parameter, like packet loss, is $\frac{\partial QoE}{\partial QoS}$. Assuming a linear dependence on the QoE level, we arrive at the following differential equation:

$$\frac{\partial QoE}{\partial QoS} = -\tilde{\beta} \cdot (QoE - \gamma) \ . \tag{2}$$

The solution for this equation is easily found as an exponential function, which expresses the basic relation of the IQX hypothesis:

$$QoE = \alpha \cdot e^{-\beta \cdot QoS} + \gamma \ . \tag{3}$$

Note that in this context the IQX hypothesis is formulated with QoS as parameter reflecting the current objective service quality that grows with the impairment. The higher the value QoS, the lower the objective quality is. The higher the value QoE, the higher the subjective quality is. In Eq. 3, QoS is for example the packet loss ratio and QoE is described in terms of MOS. In any other cases, the algebraic signs have to adapted adequately in Eq. 3. In the limit, $QoS \to \infty$, QoE approaches γ from above.

III. Measurement Study for testing the IQX Hypothesis

For the investigation of the interdependency between QoS parameters and the QoE for voice calls, we emulated various network conditions, like packet loss or jitter. Therefore, a testbed was installed in the Routerlab laboratory of the University of Würzburg. The setup of the testbed is described in Sec. III-A. It allows a) to capture packet traces at the end hosts, which is required to compute QoS parameters and b) to capture the sent and the received audio signals required to obtain MOS as QoE parameter. The definition and computation of the applied QoS and QoE metrics in this work is given in Sec. III-B. The main goal is to quantify the relationship between QoS and QoE. In particular, we investigate whether this relationship can be expressed by a simple exponential function with appropriate parameters. The derivation of the best parameters for a certain measurement scenario is done via non-linear regression techniques by minimizing the residuals observed as difference between the exponential model and the measurement values. A short summary of the applied methods for this optimization problem can be found in Sec. III-C. The last paragraph of this section shows the results for the verification of the measurement setup. In particular, we take a closer look whether the standard network emulation tool NIST Net correctly generates the desired network conditions, i.e. packt loss, delay and jitter values, as well as autocorrelated packet streams.

A. Testbed Setup

The general measurement setup is the following. The voice user A sends audio data to voice user B via a network emulating machine using UDP and IP on transport and network layer, respectively. The audio data is an English spoken text without noise of length 51 seconds, sampled at a rate of 8 kHz, encoded with 16 bits per sample which is a standard audio file for evaluating VoIP and available at [10]. In the following, the used hardware and software of the measurement testbed are explained. Detailed information on the hardware of the used machines is given in Table I. An overview of the actual versions of the software and the used operating systems can be found in Table II.

1) Hardware Configuration: The measurement testbed is set up in a local area network without any connection to the Internet to avoid any noise or cross traffic. The testbed comprises two client machines for the voice communication, called *Katie* and *Salma*, and a dedicated machine, called *Demeter*, for emulating the network conditions. The LAN is realized with Ethernet and the voice client machines are connected via crossover-cables to the emulation machine. Demeter has an additional network interface that is used to control the measurements remotely. Fig. 1 illustrates the measurement

setup and the IP address configuration. Katie and Salma are located in different subnetworks and both use Demeter as routing gateway, hence, the complete traffic between Katie and Salma can be influenced by Demeter, e.g. by introducing additional delays or dropping IP packets.

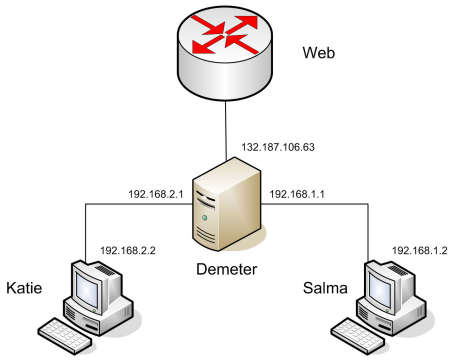

Fig. 1. Measurement Setup

TABLE I
OVERVIEW OF THE HARDWARE CONFIGURATION

Name	katie	demeter	salma
Role	Client	Router	Client
CPU	2 x Intel Pentium III		
	1.3 GHz	500 MHz	1.3 GHz
RAM	512 MB		
HDD	80 GB	16 GB	40 GB
NIC	3COM, 100 Mbps		
	1 x	3 x	1 x

TABLE II
OVERVIEW OF THE USED SOFTWARE VERSIONS

Name	Version
NIST net	2.0.12c
SJPhone	v.1.60.299, 09.24.05
Aumix	2.8
Play	(sox) 2.0-debian
Sound-recorder	0.06 (Oct 28 2005)
Tcpdump	3.9.5

2) OS and Software: For our experiments, we use the *SJPhone VoIP application* [11] for several reasons. First, SJPhone implements different voice codecs, among others, the iLBC and the G.711 voice codecs, in which we are interested in this study. The software allows to explicitly use a specified codec via the GUI or by adjusting a parameter file (in the Linux version). Second, SJPhone is open-source software that enables direct voice calls between any two hosts. Thus, the end hosts do not need to register at any SIP server in the Internet. The call initiator has to know the IP address of the machine to be called and then the call is directly established via SIP or H.323. The used session protocol suite can also be configured via the parameter file. In our measurements, we use direct SIP calls. Third, SJPhone can be controlled from the command line and configured via parameter files without using the GUI. This was a mandatory requirement to automate the measurement process. As a consequence, the measurements could be repeated many times to get statistically significant data while reducing the human efforts for conducting the measurements.

On the voice client machines, Knoppix Linux is used as operating system. During the course of this work it has been found out that conducting the measurement process with SJPhone running on Windows makes the voice client machines crash for some reasons. Additional software tools which are used in the context of this work are *aumix*, *play*, *sound-recorder*, and *tcpdump*. They are already included in the used Knoppix 5.1.1 distribution. At the sender side, *play* makes the audio file be played locally and *aumix* allows to redirect the sound output as input for SJPhone. On the receiver side, *sound-recorder* is used to capture the received audio signals and record them into a file which is later on compared with the sent audio file to obtain the QoE. *Tcpdump* is used to capture packet traces on OSI layer 2 at the sending and the receiving voice client machines in order to get statistics on QoS parameters.

The network emulation machine *demeter* runs SuSe Linux and hosts *NIST Net* that is a network emulation package running only on Linux. NIST Net allows a single Linux PC set up as a router in order to emulate a wide variety of network conditions. In particular, selected performance effects are applied to the IP packets of the out-going stream. Via command line, the network conditions of a single end-to-end path can be controlled, which is again required for the automated measurement process. The controllable network parameters of interest in this work are packet loss and delay. It is possible to generate random packet losses according to a given packet loss probability p_L. This means IP packets are randomly dropped with probability p_L. NIST Net additionally accepts an autocorrelation parameter ϱ_L for the loss, however, this parameter has no effect on the out-going stream, which is demonstrated in Sec. III-D. In order to control the delay between two nodes, the average delay μ_d, the standard deviation σ_d of the delay, and the autocorrelation ϱ_d can be passed to NIST Net, which uses a normal distribution with the related parameters go randomly generate delays. The verification of the correct emulation of these parameters is shown in Sec. III-D.

B. Computation of QoS and QoE Parameters

As result of the measurements we obtain the received audio file and tcpdump packet traces at the sender machine Katie and the receiver machine Salma. For each sent and received packet on both machines, we extract an unique ID, the size of the packet, and the local timestamp when the packet is sent or received, respectively. Note, that the clocks at Salma and Katie are not synchronized and might drift. However, this is not necessary for assessing the applied QoS parameters.

1) Packet loss: Let $s_{out} = \{p_{out,1}, p_{out,2}, \ldots, p_{out,n}\}$ be the set of packets that are sent from Katie to Salma. The packets $p_{out,i}$ are ordered in ascending order according to their sending timestamps $t_{s,p_{out,i}}$, i.e. $i < j \Rightarrow t_{s,p_{out,i}} \leq t_{s,p_{out,j}}$.

Analogously, let $s_{in} = \{p_{in,1}, p_{in,2}, \ldots, p_{in,m}\} \subseteq s_{out}$ be the set of packets that are received by Salma from Katie. The packets $p_{in,i}$ are ordered in ascending order according to the timestamps $t_{r,p_{in,i}}$ when the packets are received, i.e. $i < j \Rightarrow t_{r,p_{in,i}} \leq t_{r,p_{in,j}}$. The measured packet loss ratio simply follows as

$$\tilde{p}_L = 1 - \frac{|s_{in}|}{|s_{out}|} = 1 - \frac{m}{n} \ . \tag{4}$$

On average, the measured packet loss ratio $\tilde{p}_L$ should be equal to the preset packet loss probability p_L of the network emulator, i.e. $\tilde{p}_L = p_L$.

2) One-Way Delays: The one-way delay is basically defined as the time difference between the time $t_{s,p}$ when sending the first bit of a packet p at the sender side until the time $t_{r,p}$ receiving the last bit of the packet p at the receiver side [12]. The one-way delay d_p for a packet p follows as

$$d_p = t_{r,p} - t_{s,p} \quad \text{for } p \in s_{in} \subseteq s_{out} \ . \tag{5}$$

Note that in case of dropped packets the one-way delay is not defined. However, as the clocks at the sender and the receiver side do not need to be synchronized and the clocks might additionally drift, the estimation of one-way delays out of measurement data is a complex task. Binzenhöfer et. al propose in [13] a method to estimate accurate one-way delays based on packet captures at the sender and at the receiver side. Thus, the estimation method is applicable in the context of this work to get one-way delays. The proposed method overcomes unsynchronized clocks and linear clock drifts. Note that the one-way delays are only required in Sec. III-D to verify the emulated end-to-end one-way delays. However, we will additionally use them to show alternative metrics for jitter.

3) Jitter: The term jitter is used to express delay variations within a stream of received packets. In literature, there exist different definitions of how to assess the jitter. The most common ones are a) the standard deviation of the one-way delay $\omega = std_{OWD}$ and b) the inter-packet delay variation std_{IPDV} according to RFC 3393 [14]. The standard deviation of the round trip delay is also a common measure, however, it cannot be used in the context of VoIP, as the packets are neither acknowledged nor returned to the sender.

After computing the one-way delays d_p for all received packets $p \in s_{in}$, the standard deviation $\omega_{s_{in}}$ of the one-way delays simply follows as

$$\begin{aligned} \omega_{s_{in}} &= \text{STD}\left\{d_p | p \in s_{in}\right\} \\ &= \sqrt{\frac{1}{|s_{in}| - 1}\left(\sum_{p \in s_{in}} {d_p}^2 - \left(\sum_{p \in s_{in}} d_p\right)^2\right)} \ . \end{aligned} \tag{6}$$

A different common metric for expressing jitter uses the inter-packet delay variation IPDV as defined in [14]. The IPDV compares the one-way delays of a selected pair of packets within a stream. It is defined as the difference between the one-way delays d_p and d_q of the packets p and q. It holds $IPDV(p,q) = d_p - d_q = (t_{r,p} - t_{s,p}) - (t_{r,q} - t_{s,q}) = (t_{r,p} - t_{r,q}) - (t_{s,p} - t_{s,q}) = \Delta t_{r,p,q} - \Delta t_{s,p,q}$. Thus, the IPDV of two packets is the difference of the inter-packet delay in the outgoing stream of packets s_{out} and the inter-packet delay in the received stream s_{in}. As measure for the jitter of a packet stream, the standard deviation of the IPDV any two consecutively received packets is computed as follows:

$$IPDV_{s_{in}} = \text{STD}\left\{IPDV(p_{in,i}, p_{in,i+1}) | 1 \leq i < m\right\} \ . \tag{7}$$

4) Packet reordering: As a consequence of delay variations in a stream of packets, it might occur that packets are reordered. Depending on the actual implementation, an application might be able to handle jitter by using an appropriate jitter buffer, however, reordered packets might be more difficult to deal with on application layer and hence result into significant QoE degradations. This performance issue was revealed during the course of this work for the application under study. Therefore, we also investigate this phenomenon and its influence on the QoE, although in the Internet, it is assumed that the amount of reordered packets is not relevant.

There exist different metrics for quantifying packet reordering. In [15], a detailed introduction on the necessity of different packet reordering metrics is given and the computation of the metrics is proposed. In general, a received packet $p \in s_{in}$ is referred to as *reordered packet* if and only if there is at least one packet $q \in s_{in}$ which was sent after p, i.e. $t_{s,p} < t_{s,q}$, but arrives before the packet p, i.e. $t_{r,q} < t_{r,p}$.

$$p \text{ is reordered} \Leftrightarrow \exists q \in s_{in} : t_{s,p} < t_{s,q} \wedge t_{r,q} < t_{r,p} \ . \tag{8}$$

The ratio $\rho(s_{in})$ of reordered packets within a stream of packets is denoted as *Type-P-Reordered-Ratio*, or reordering ratio in short. It is calculated as

$$\rho_{s_{in}} = \frac{|\{p \in s_{in} | p \text{ is reordered}\}|}{|s_{in}| - 1} \ . \tag{9}$$

The reordered ratio is a very simple metric, as it does not take into account how "much" a single packet is reordered. This can be illustrated with a simple example. Let s_A be packet stream with 8 packets, $s_{out,A} = \{p_1, \ldots, p_8\}$. If p_8 arrives for some reasons before p_4, but all other packets are sent in correct order, the received packet stream is $s_{in,A} = \{p_1, p_2, p_3, p_8, p_4, p_5, p_6, p_7\}$ and the resulting reordered ratio is $\rho_{s_{in,A}} = 1/2$, as $p_4, \ldots, p_7$ are reordered according to the definition above. However, an application might only drop packet p_8 while the other packets are processed correctly, as only the packet arriving out of order cannot be processed. If the stream s_A is received as $s_{in,B} = \{p_2, p_1, p_4, p_3, p_6, p_5, p_8, p_7\}$, we obtain the same reordered ratio $\rho_{s_{in,B}} = 1/2$. Later, we will see that this metric is sufficient to describe the relationship between packet reordering and QoE, as SJPhone seems to have problems with reordered packets.

A more complex metric to quantify packet reordering is the *mean reordering late time* of a packet stream [15]. The reordering late time is the maximum distance in time from a reordered packet to the earliest packet received that has a larger sequence number. If a packet is in-order, its reordering

late time is undefined. The first packet to arrive is in-order by definition and has undefined reordering late time. This metric seems appropriate to capture the network disturbance as perceived on application layer. A formal definition is

$$\tau = \frac{1}{|R|} \sum_{i \in R} t_{in,i} - t_{in,j} \tag{10}$$

with $R = \{p \in s_{in} : p \text{ is reordered}\}$, $j = \min\{k | 1 \leq k < i\}$.

5) Mean opinion scores: For the quantification of the QoE, we use a full reference metric, i.e. we compare the sent signal with the received one offline. Our measurement testbed allows to capture the audio signals on the sender and the receiver side and allows to apply the full reference metric after a measurement run. In particular, we use the mean opinion score (MOS) [1] to express the QoE of the VoIP call. Therefore, the audio file sent is compared with the received wav-file using the Perceptual Evaluation of Speech Quality (PESQ) method described in ITU-T P.862 [16]. The resulting PESQ value can be mapped into a subjective MOS value according to ITU-T Recommendation ITU-T P.862.1 [17]. The MOS can take the following values: (1) bad; (2) poor; (3) fair; (4) good; (5) excellent.

C. Fitting the QoS onto QoE Mapping Function

The model function $f(x) = \alpha \cdot e^{-\beta x} + \gamma$ as derived in Eq. 3 mathematically expresses the mapping from the value x of the considered QoS parameter to the QoE measure, i.e. MOS in this work. The parameters α, β, γ of the model function are retrieved by means of non-linear regression. We used the optimization toolbox of Matlab to find an optimal fitting function for the given measurement points. Optimal in this case means to find the unknown parameters α, β, γ in Eq. 3 such that the mean squarred error E^2 is minimized. The mean squarred error is defined as the average of the squared residuals $r_i^2 = (f(x_i) - y_i)^2$ for all n measurements (x_i, y_i) with a measured QoS value x_i and a measured MOS y_i:

$$E^2 = \frac{1}{n} \sum_{i=1}^{n} r_i^2 = \frac{1}{n} \sum_{i=1}^{n} (f(x_i) - y_i)^2 \ . \tag{11}$$

The goodness-of-fit for the model function $f(x)$ can be measured with different metrics, like the coefficient of correlation R between the model function and the measured data or the coefficient of determination R^2. It can be computed as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - f(x_i))^2}{\sum_{i=1}^{n} (y_i - \overline{y})^2} \tag{12}$$

with $\overline{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$. A value close to one means a perfect match between the model function and the measured data. Other common metrics are functions of the residuals which show a perfect match between model and measurements if the value is close to zero. Examples are the mean squarred error E^2 or the normalized mean squared error $NMSE = E^2/VAR[y_i]$ which is normalized by the variance of the measured MOS values. In this paper we use the coefficient of determination R^2 to test the IQX hypothesis and to show the goodness-of-fit of the proposed exponential model function for the obtained measurement results.

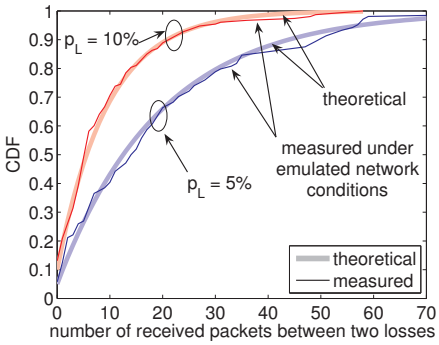

Fig. 2. Verifying the emulation of uncorrelated packet loss

D. Verification of the Emulation of Network Conditions

Although NIST Net is a common tool for emulating network conditions, we conducted several test runs to investigate whether the desired network condidtions are correctly emulated or not. Summarizing, NIST Net correctly emulates a) uncorrelated packet loss with input parameter for the packet loss probability p_L and correlation factor $\varrho_L = 0$, and b) correlated as well as uncorrelated delays with input parameters for the average delay μ_d, the standard deviation of the delay σ_d, and the correlation factor ϱ_d. However, correlated packet loss streams are not correctly emulated which is discussed in the following.

1) Packet loss: For verifying the emulation of uncorrelated packet loss, we investigate the inter-packet loss distance K, that is the number K of received packets between two consecutive packet losses. For a given packet loss probability p_L, the inter-packet loss distance follows a geometric distribution and $P(K = i) = p_L \cdot (1 - p_L)^i$ for $i = 0, 1, 2, \ldots$ in case of uncorrelated loss. Fig. 2 compares the theoretical and the measured cumulative distribution functions (CDF) of the inter-packet loss distance for $p_L = 0.1$ and $p_L = 0.05$. For sufficient long test runs, the measured packet loss ratio $\tilde{p}_L$ approaches the preset dropping probability.

2) Delay and Jitter: For verifying the emulation of delays, we consider the CDF of the one-way delay d_p for any packet p transmitted from sender to receiver. NIST Net offers the possibility to use different delay distributions. In our measurements, we use the normal distribution with parameter μ_d for the average delay and σ_d for the standard deviation of the delay. Fig. 3 shows the CDF of the one-way delays for $\mu_d \in \{0\,\text{ms}, 90\,\text{ms}\}$ and $\sigma_d \in \{1\,\text{ms}, 5\,\text{ms}, 10\,\text{ms}\}$. Again, we can see that the theoretical and the measured curves agree. Thus, NIST Net correctly emulates delay and jitter as desired.

Note that an average delay μ_d of 0 ms means that NIST Net does not add any additional delays before relaying a packet. As the packets are transmitted via Ethernet from sender to receiver, we obtain a minimal transmission time d_0 for the one-way delay which is around $d_0 = 0.3\,\text{ms}$. NIST Net inter-

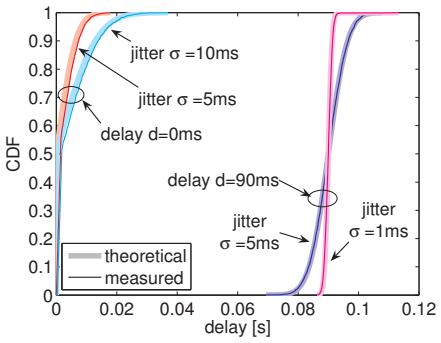

Fig. 3. Verifying the emulation of uncorrelated jitter and delay

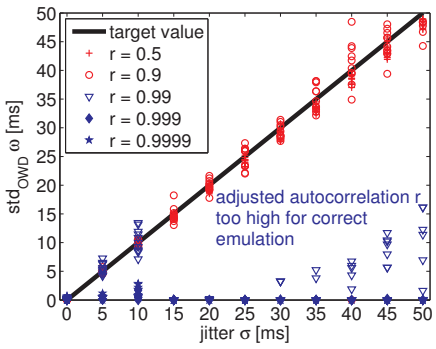

Fig. 4. Measured standard deviation of the one-way delays ω depending on jitter σ preset in network emulation package NistNet for different autocorrelation settings $r = \varrho_d$

nally generates pseudo random numbers following a normal distribution to delay packets. As negative delays do not make sense, NIST Net sets negative values to 0 ms which explains the probability of 50% for the minimal one-way delay d_0, $P(d_p = d_0) = 0.5$.

A deep inspection of the source code of NIST Net revealed that NIST Net is optimized w.r.t. computational time at the cost of accuracy. In particular, NIST Net can only generate random delay values which in case of the normal distribution lie in the interval $[\mu_d - 4\sigma_d; \mu_d + 4\sigma_d]$. However, the probability for delay values to be larger is negligible, $P(d_p > \mu_d + 4\sigma_d) = (1 + erf\frac{4}{\sqrt{2}})/2 = 3.17 \cdot 10^{-5}$ for normally distributed delays.

3) Autocorrelated Packet Streams: The emulation of autocorrelated packet streams is basically approximated by a first-order autoregressive process AR(1) with $y_i = x_i \cdot (1 - \varrho) + y_{i-1}\varrho$ in NIST Net. For generating the next random value y_i the fraction ϱ of the previous random value y_{i-1} is taken into account which leads to an autocorrelation of ϱ at lag 1.

However, the current implementation does not correctly emulate autocorrelated packet losses which would be one possibility to produce bursty losses. We deeply investigated the source code and found out that the error stems from internal conversions between 16 bit and 32 bit integer values. A formal mathematical proof that for a given packet loss probability p_L and a correlation factor ϱ_L NIST Net generates a packet stream with a measured packet loss ratio $\tilde{p}_L = p_L$ and $\tilde{\varrho}_L = 0$ (instead of $\tilde{\varrho}_L = \varrho_L$) can be found in the technical report [18].

Next, we check the emulation of autocorrelated delay values. Fig. 4 plots the measured standard deviation ω of one-way delays on the y-axis against the given jitter values σ_d passed to NIST Net on the x-axis. We varied the correlation factor ϱ_d assigned to NIST Net from 0.5 to 0.9999. Independently of the preset correlation factor ϱ_d, labeled with r in Fig. 4, the measurement results should lie on the line $\omega(\sigma_d) = \sigma_d$. For $\varrho_d < 0.9$, the generated delay values are as desired. However, for very large correlation factors $\varrho_d \geq 0.99$, NIST Net does not correctly emulate the given parameter settings. In this work, we therefore investigate the impact of autcorrelated delay values for correlation factors $\varrho_d \in \{0.5, 0.9\}$ only. In the following, we will investigate whether delay correlations in packet streams have an impact at all on the QoE.

IV. Measurement Results and Analysis

The measurements presented here were conducted during January 2007 and April 2007 at the Routerlab of the University of Würzburg. We test the IQX hypothesis for different preset QoS parameters, that are packet loss, delay and jitter. For quantifying these QoS parameters, we use the metrics as defined in Sec. III-B. For each of the QoS parameter setting ten individual measurement runs were repeated to gain statistical significant data. In the following figures, Fig. 5 – Fig. 13, a single dot represents a single measurement run with the measured QoS value as obtained by the packet trace on the x-axis and the observed mean opinion score on the y-axis.

To demonstrate whether an exponential interdependency between the QoS and the QoE can be observed when varying a single QoS parameter, we fit the measurement data as described in Sec. III-C. The resulting exponential model function is plotted in each corresponding figure, the obtained optimal parameters of Eq. 3 are annotated, as well as the coefficients of determination R^2 are given as goodness-of-fit measure.

A. Voice quality affected by loss

We start to investigate the influence of packet loss on the user perceived quality. Fig. 5 and Fig. 6 show the measurement results for the iLBC and the G.711 codec, respectively. In these experiments, the packet loss p_L was varied from 0% up to 40% in steps of 1%. Furthermore, we performed the measurements without any additional delay $\mu_d = 0\,\text{ms}$ and with an additional delay of $\mu_d = 90\,\text{ms}$ emulated by NIST Net.

The first observation is that there is a clear exponential relationship between the packet loss ratio and the MOS for iLBC as well as G.711. The results show that the IQX hypothesis holds for this scenario. Thus, the QoE degradation is very strong when the packet loss ratio increases slightly. For iLBC, the MOS is 4 without any loss, 3 for 1.6% packet loss, and 2 for 4.5% packet loss. For G.711, the MOS is also 4 without any loss, 3 for 1.4% packet loss, and 2 for 4% packet loss. The second observation is that the additional delay of 90 ms has no influence on this relationship – which is expected, as only large delays above 200 ms have an additional impact

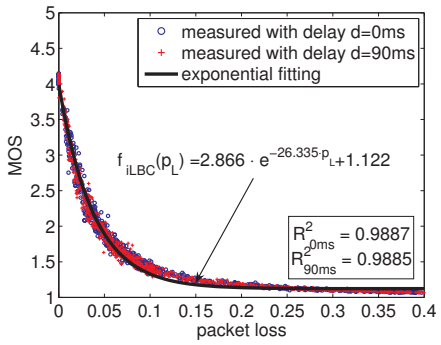

Fig. 5. Measurement results and obtained mapping function $f_{iLBC}(p_L)$ between packet loss ratio p_L and MOS for the iLBC codec

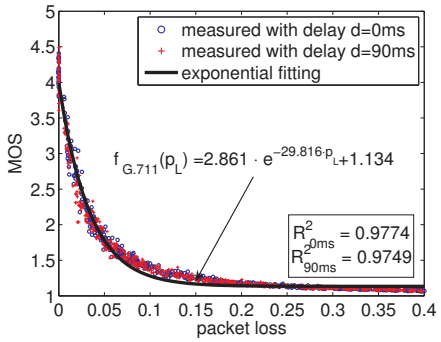

Fig. 6. Measurement results and obtained mapping function $f_{G.711}(p_L)$ between packet loss ratio p_L and MOS for the G.711 codec

on the QoE according to ITU-T recommendation G.114, cf [19].

B. Voice quality affected by jitter and reordering

Next, the influence of jitter on the QoE is investigated. In the experiments, we vary the jitter σ_d from 0 ms to 30 ms in steps of 1 ms, and afterwards in steps of 5 ms up to 80 ms. Again, we executed the measurements without any additional delay $\mu_d = 0\,\text{ms}$ and with an additional delay of $\mu_d = 90\,\text{ms}$. In this case, different results for both average delay values are expected as the variability of the delay values generated by NIST Net follows a normal distribution with parameters μ_d and σ_d. We will see that as a consequence of the jitter, packet reordering occurs, which decreases the user perceived quality. Describing this influence on the application with an appropriate packet reordering metric allows to verify again the IQX hypothesis for both codecs. However, their performance differs significantly, and we therefore start to provide the results for iLBC before the G.711 results are depicted.

1) iLBC: We first investigate the jitter value σ_d as QoS parameter to test the IQX hypothesis. Fig. 7 reveals that the measurement values scatter much more around the exponential fitting function than for the packet loss curves in the previous section. Obviously, for a certain jitter setting, no extra delay ($\mu_d = 0\,\text{ms}$) leads to higher MOS values than a scenario with an average delay of 90 ms.

Typically, real-time applications like VoIP or video streaming are able to handle jitter up to a certain level by using a jitter buffer. This explains why for small jitter values below 10 ms the curves are more flat and the QoE degradation is not so strong with increasing jitter, especially for $\mu_d = 0\,\text{ms}$. After that the MOS again show exponential decays. As the fitting is done for all variations of σ_d, the obtained mapping funcition from QoS to QoE shows a worse coefficient of determination.

However, in the experiments described above, the delay values are randomly generated and uncorrelated. Hence, packets might overtake each other and packet reordering occurs. Therefore, we use now as metric the packet reordering ratio ρ to quantify the QoS. To highlight this clearly, we use the MOSs and packet traces from the measurements as in Fig. 8, but as QoS metric we calculate ρ instead using σ_d.

As a result of Fig. 8, we clearly observe an exponential relationship between the QoE and the QoS. We obtain as large goodness-of-fit values as for packet loss and hence confirm again the IQX hypothesis. The main result of this section is that the important challenge consists in finding the appropriate QoS metric for describing the effect of the QoS influence on the QoE. In this particular case, this means that SJPhone gets into trouble when packets are reordered. Obviously, on application layer, packet reordering has a similar impact as packet loss. If packets are reordered, they are not processed any more by SJPhone. In particular, it is possible to convert the packet reordering ratio ρ to a packet loss ratio p_L such that the same MOS values are obtained $f(p_L) = f(g(\rho))$. From the results in Fig. 5 and Fig. 8, we compute the conversion function g for iLBC and $\mu_d = 90\,\text{ms}$:

$$p_L = g(\rho) = \max\{0.3837 \cdot \rho - 0.0054, 0\}\ . \qquad (13)$$

This means that the packet loss is a linear function of the reordering ratio.

TABLE III
MEAN SQUARRED ERRORS E^2 OF THE IQX HYPOTHESIS FOR DIFFERENT QOS METRICS APPLIED TO DESCRIBE THE IMPACT OF JITTER

	iLBC with delay		G.711 with delay	
	0 ms	90 ms	0 ms	90 ms
$ratio_{reordered}$ $\rho_{s_{in}}$	0.097	0.067	0.063	0.036
$mean_{n-reord}$	0.086	0.061	0.041	0.030
$mean_{reord-extent}$	0.089	0.072	0.040	0.035
$mean_{n-times-reord}$	0.087	0.066	0.040	0.032
$mean_{reordered-late-time}$ τ	0.108	0.091	0.056	0.036
$IPDV_{s_{in}}$	0.158	0.110	0.258	0.243
std_{OWD} $\omega_{s_{in}}$	0.158	0.112	0.259	0.241
jitter σ_d (NIST Net)	0.191	0.151	0.255	0.244

Table III shows the mean squared errors E^2 of the exponential mapping function between QoS and QoE when applying different QoS metrics to describe the impact of jitter. The QoS metrics are defined as in Sec. III-B. We additionally give the results for some more common metrics, as defined in [20], without explicitly showing the fittings due to lack of space. Table III includes the results for iLBC and G.711 while the delay is either 0 ms or 90 ms. From the table, we conclude

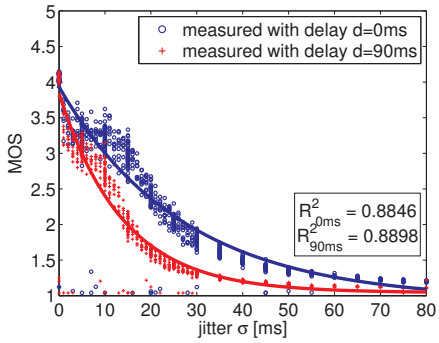

Fig. 7. Measurement results and obtained mapping functions between preset jitter σ and MOS for the iLBC codec

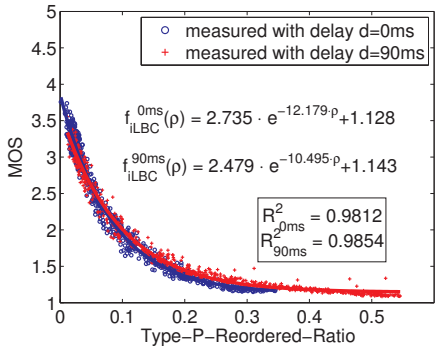

Fig. 8. Measurement results and obtained mapping function $f_{iLBC}(\rho)$ between Type-P-Reordered-Ratio ρ and MOS for the iLBC codec

that in all scenarios the packet reordering metrics reveal the relationship to the QoE better than the pure jitter metrics. However, this is not a general statement, in particular, this is caused by the fact that the used application has problems with packet reordering which affects the user perceived quality.

2) G.711: The impact of jitter on the QoE is analogously examined for the G.711 voice codec. In Fig. 9, the jitter value σ_d, which is passed as input parameter to NIST Net, is used as QoS parameter. The same observations as for iLBC are obtained. For a certain jitter value $\sigma_d > 0$, a lower average delay leads to a higher MOS. If the jitter values are below 10 ms, the curves are quite flat and the QoE degradation is not so strong with increasing jitter. For larger jitter values, the QoE in terms of MOS decays. However, the decay is not so strong as for iLBC. This is caused by the fact that as soon as jitter appears, i.e. even for $\sigma_d = 1\,\mathrm{ms}$, the MOS drops down to a value of 2, i.e. the quality is already poor. Note that for $\sigma_d = 0\,\mathrm{ms}$ the MOS is about 4, i.e. good quality.

An explanation for this can be found when investigating the sending pattern of the SJPhone application. Even though the G.711 codec is defined with a constant packet sending rate of $50\,\mathrm{s}^{-1}$, SJPhone uses intervals of length 32 ms to send packets. In order to achieve the desired bit rate, several packets are sent together. In detail, we observed the following pattern of time intervals in milliseconds between two consecutively sent packets: $0, 32, 32, 0, 32, 32, 0, 32$. In total, this leads to an average time of 20 ms between two packets. Thus, the codec mean bit rate is realized, but the single inter-packet delay varies. An inter-packet delay of 0 ms means that two packets are sent together at the same time. For the implementation of G.711 in SJPhone, this means that 37.5% of the packets are sent together. As a consequence, even a very small jitter like $\sigma_d = 1\,\mathrm{ms}$ might lead to packet reordering and causes a strong QoE degradation. For $\sigma_d = 1\,\mathrm{ms}$, we already obtain a packet reordering ratio of roughly 15%.

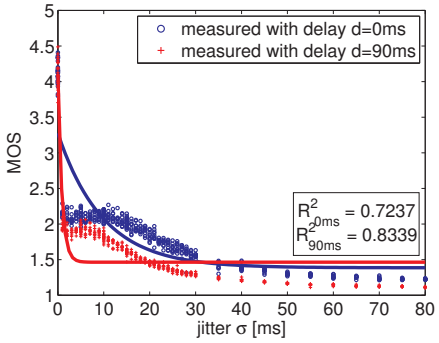

Fig. 9. Measurement results and obtained mapping function $f_{G.711}(\sigma)$ between preset jitter σ and MOS for the G.711 codec

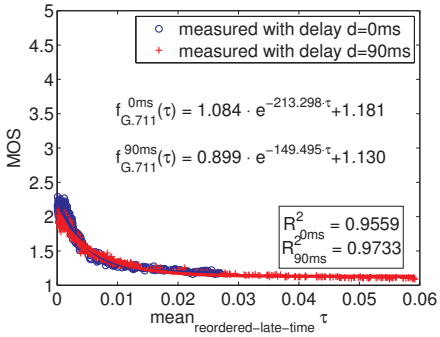

Fig. 10. Measurement results and obtained mapping function $f_{G.711}(\tau)$ between mean reordered late time τ and MOS for the G.711 codec

In Fig. 10, we use the mean reordered late time τ to describe the impact of jitter and the resulitng packet reordering as QoS parameter. Again, the IQX hypothesis can be confirmed and an exponential relationship between QoS and QoE is found.

C. Influence of Autocorrelated Packet Streams

Up to now we have investigated the impact of uncorrelated packet streams. In the context of packet loss, this means that packets are dropped randomly. As a consequence of uncorrelated delays, much more packet reordering occurs than for correlated delay which might be caused e.g. by queues at router along the end-to-end path. In the previous section, we have already seen that for the actual implementation of the G.711 codec in SJPhone very small jitter values result

in a high packet reordering ratio. For iLBC in contrast, this weird application phenomena was not observed. Therefore, we focus on the iLBC codec using SJPhone when investigating autocorrelated packet streams. As NIST Net does not correctly emulate autocorrelated packet loss, we generate bursty losses by dropping n subsequent packets. In particular, we investigate the impact of $n \in \{0, \ldots, 300\}$ consecutively lost voice datagrams on the QoE. Before that, we take a closer look at correlated delay values, which are correctly generated by NIST Net.

1) Autocorrelated delay values: We have already shown that NIST Net correctly emulates delay values for any correlation factor $\varrho_d \leq 0.9$, that is the measured delay values show an average delay $\tilde{\mu}_d$, a standard deviation $\tilde{\sigma}_d$, and a autocorrelation $\tilde{\varrho}_d$ which correspond to the parameter settings preset in NIST Net. In the scenario, we consider $\mu_d = 90\,\text{ms}$ and no packet loss $p_L = 0$, while the jitter is varied in the range $\sigma_d \in [0\,\text{ms}; 50\,\text{ms}]$. As correlation factor, we use either $r = \varrho_d = 0.5$ or $r = \varrho_d = 0.9$.

Fig. 11 shows the measured standard deviation ω of the one-way delay vs. MOS. The color of the dots indicates the preset NIST Net setting. It shows that independently of the correlation factor, the measured delays ω meet the preset jitter values σ_d. Furthermore, there is a clear difference between the curves for the different correlation factors. For $r = 0.9$ the obtained MOS values are larger than for $r = 0.5$. This is expected as a larger correlation reduces the reordering of packets.

Therefore, we describe the impact of the QoS on the QoE using the packet reordering ratio ρ. Fig. 12 shows the measurement results using ρ instead of ω. For a high correlation factor $r = 0.9$, we obtain a reorderin ratio $\rho \in [0; 0.15]$ according to the preset jitter σ_d. This means at most 15% of the packets are reordered even for a jitter of 50 ms. A lower correlation factor $r = 0.5$ means that the one-way delays of consecutive delays do not depend so strongly on each other. As a result, a packet reordering ratio up to 45% emerges for $\sigma = 50\,\text{ms}$. Nevertheless, for the same reordering ratio ρ, the observed MOS is higher for less correlated delay $r = 0.5$ than for strongly correlated ones, $r = 0.9$. Note that in Fig. 12, the majority of measurement results for $r = 0.9$ shows a reordering ratio $\rho < 5\%$ and MOS values larger than 2.5. In contrast, for $r = 0.5$, the reordering ratio goes up to 25% and MOS values are only larger than 1.5. As a main result, both curves can be well fitted by an exponential distribution. However, the actual curves strongly depend on the correlation factor. A more detailed analysis and the usage of different network emulator environments to investigate autocorrelated packet streams is a topic of future work.

2) Bursty Losses: Finally, the impact of bursty losses on the QoE is examined. As we know that NIST Net cannot be used for the emulation of bursty losses by adjusting the correlation factor for packet loss, this investigation was performed in a different way. On a local machine, we packetised the audio signal using the iLBC codec and dropped selected voice datagrams. To be more precise, we dropped n consecutive

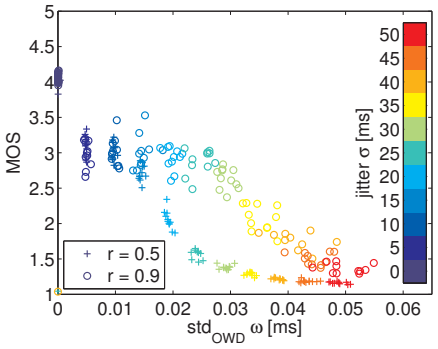

Fig. 11. Measurement results for the iLBC codec depending on the measured one-way delay ω

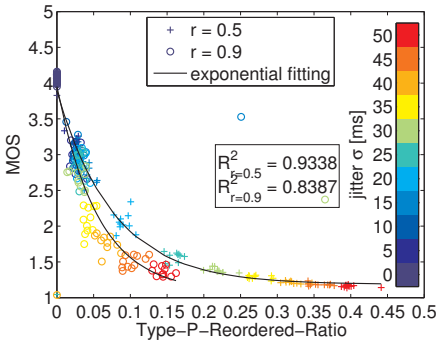

Fig. 12. Measurement results for the iLBC codec depending on the measured packet reordering ratio ρ

voice datagrams starting from voice datagram n_0. After that, the remaining voice datagrams were passed to the iLBC codec to be decoded as audio signal. Accordingly, the QoE was derived in the same manner as described in Sec. III-B.

Fig. 13 shows the number n of consecutively lost datagrams on the x-axis and the MOS on the y-axis. We also varied over n_0 which denotes the first lost packet. Obviously, the larger n, the worse the MOS becomes. For $n \leq 50$ there are no significant differences between the different curves for the first lost packet n_0. However, larger $n > 50$ make the curves disperse. Note that this corresponds to a silent period of 1.5 s and it is not clear whether the PESQ and MOS computation is able to correctly map this silence period on the real user experienced degree of satisfaction. Indeed, if the silence period is too long, a user will probably abort a call. However, this is hardly considered in this computation.

One more remarkable observation is that $n = 50$ consecutively lost packets means a packet loss ratio of $\tilde{p}_L = 3\%$, as the transmitted voice file has a length of 51 s consisting of 1700 iLBC voice datagrams. However, the observed MOS value of roughly 3.7 is much higher than for the same packet loss ratio with randomly dropped packets at a MOS value of 2.4. But it has to be noted that in this last experiment, the voice signals were locally encoded and decoded, but not transmitted via the test bed. Therefore, we suggest to modify NIST Net or use a

different network emulator which easily allows to investigate bursty loss models. This is also a topic of future work we want to focus on.

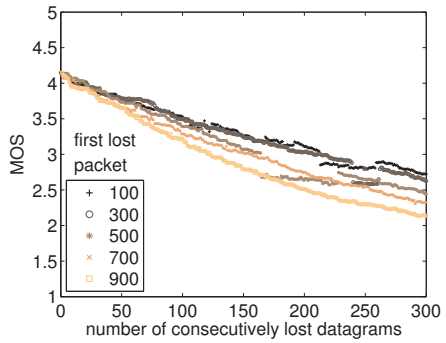

Fig. 13. Impact of bursty losses on the QoE for iLBC

V. Conclusion and Outlook

In this paper, we have validated an exponential interdependency of Quality of Experience (QoE), reflecting the degree of user satisfaction, from Quality of Service (QoS) disturbances formulated as IQX hypothesis. This validation was performed for the voice codecs iLBC and G.711 used in VoIP scenarios. The experimental setup consisted of two computers, each running the softphone SJPhone, interconnected by a third machine hosting the network emulator software NIST Net. This software allows among others for presetting the experimental conditions in terms of packet loss, one-way delay and delay jitter.

For each preset packet loss, delay and jitter setting, the received audio file is compared to the undistorted file by software determining the PESQ (Perceptual Evaluation of Speech Quality) and subsequent calculation of MOS (Mean Opinion Score). In case of packet loss, the exponential decay of QoE with growing QoS disturbance was clearly confirmed. While the effect of (constant) one-way delay is rather limited due to the fact that the receiver receives all packets with a constant delay, delay jitter also gives raise to exponentially-looking shapes, however with some remarkable deviations for small jitter values. A closer investigation of the traffic flow associated with SJPhone reveals the cause for this behavior, that is a pronounced sensitity of that particular softphone to packet reordering introduced by NIST Net. Plotting the QoE against the packet reordering ratio, we again observe a clear exponential interdependency.

In addition to these measurement results, we verified our testbed and in particular whether the emulated network conditions are emulated as desired. As a result, we found out that while NIST Net is capable of producing autocorrelated packet delay, it does not manage to impose autocorrelated packet loss. Thus, we cannot use the tool to emulate burst losses that can have a distinctive effect on QoE - the receiver misses a part of the speech.

Despite of these limitations, our investigations have shown that the IQX hypothesis appropriately captures the main vulnerabilities shown by the application SJPhone towards network-level disturbances, expressed in packet loss and reordering. This also shows the capability of the IQX hypothesis to identify the relevant performance metrics.

Future work will address autocorrelated packet streams and bursty loss models. Such studies require the incorporation of a different network emulator that allows for advanced queuing features, for instance avoiding packet reordering and thus allowing for the incorporation of Furthermore, the joint effect of several QoS disturbances should be studied quantitatively.

References

[1] ITU-T Recommendation, "ITU-T Rec. P.800.1: Mean Opinion Score (MOS) Terminology," 2003.

[2] M. E. Perkins, K. Evans, D. Pascal, and L. A. Thorpe, "Characterizing the subjective performance of the ITU-T 8 kb/s speech coding algorithm-ITU-T G.729," *IEEE Communications Magazine*, vol. 35, no. 9, 1997.

[3] R. Kwitt, T. Fichtel, and T. Pfeiffenberger, "Measuring perceptual VoIP speech quality over UMTS," in *4th International Workshop on Internet Performance, Simulation, Monitoring and Measurement (IPS-MoMe 2006)*, (Salzburg, Austria), 2006.

[4] T. Hoßfeld, A. Binzenhöfer, M. Fiedler, and K. Tutschku, "Measurement and analysis of skype voIP traffic in 3g UMTS systems," in *4th International Workshop on Internet Performance, Simulation, Monitoring and Measurement (IPS-MoMe 2006)*, (Salzburg, Austria), 2 2006.

[5] A. P. Markopoulou, F. A. Tobagi, and M. J. Karam, "Assessing the quality of voice communications over internet backbones," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, 2003.

[6] ITU-T Recommendation, "ITU-T Rec. G107: The E-model, a computational model for use in transmission planning," 1998.

[7] L. Ding and R. A. Goubran, "Speech quality prediction in VoIP using the extended E-model," in *GLOBECOM 2003 - IEEE Global Telecommunications Conference*, (San Francisco, USA), 2003.

[8] A. Richards and G. Rogers and M. Antoniades and V. Witana, "Mapping User Level QoS from a Single Parameter," in *In Proceedings of MMNS '98*, (Versailles, France), 1998.

[9] T. Hoßfeld, P. Tran-Gia, and M. Fiedler, "Quantification of quality of experience for edge-based applications," in *20th International Teletraffic Congress (ITC20)*, (Ottawa, Canada), 6 2007.

[10] Signalogic, "Speech codec wav samples." http://www.signalogic.com/melp/ EngSamples/Orig/male.wav.

[11] "http://www.sjlabs.com."

[12] G. Almes, S. Kalidindi, and M. Zekauskas, "A One-way Delay Metric for IPPM." RFC 2679 (Proposed Standard), September 1999.

[13] A. Binzenhöfer, D. Schlosser, K. Tutschku, and M. Fiedler, "An autonomic approach to verify end-to-end communication quality," in *Tenth IFIP-IEEE International Symposium on Integrated Network Management (IM 2007)*, (Munich, Germany), 5 2007.

[14] C. Demichelis and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)." RFC 3393 (Proposed Standard), Nov. 2002.

[15] A. Morton, L. Ciavattone, G. Ramachandran, S. Shalunov, and J. Perser, "Packet Reordering Metrics." RFC 4737 (Proposed Standard), Nov. 2006.

[16] ITU-T Recommendation P.862, "Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs," 2001.

[17] ITU-T Recommendation, "ITU-T Rec. P.862.1: Mapping function for transforming P.862 raw result scores to MOS-LQO," 2003.

[18] T. Hoßfeld, D. Hock, K. Tutschku, P. Tran-Gia, and M. Fiedler, "Investigating the exponential interdependency between qoe and qos for voip," tech. rep.

[19] I. T. Union, "ITU-T recommendation G.114: One-way transmission time," tech. rep., 5 2003.

[20] A. Morton, L. Ciavattone, G. Ramachandran, S. Shalunov, and J. Perser, "Packet reordering metrics." RFC 4737 (Proposed Standard), Nov. 2006.