Virtual Audio Chat: User Interaction and Audio Quality Evaluation

Lea Skorin-Kapov* and Saška Mateša**

*Ericsson Nikola Tesla
Krapinska 45, 10000 Zagreb, Croatia
**Zagrebačka Banka
Paromlinska 2, 10000 Zagreb, Croatia
e-mail: lea.skorin-kapov@etk.ericsson.se, saska.matesa@zaba.hr

Abstract

Virtual Audio Chat is an interactive Virtual Reality (VR)/Web application that enables real-time audio streaming between multiple Internet users. Users join the audio chat using a VR interface designed as a Virtual Reality Modeling Language (VRML) model of a mobile phone. Various interactions implemented using VRML allow users to choose the color/textures applied to the mobile phone and the audio codec used during the audio streaming session.

In this paper we address the developed application in terms of user interactions with the VR interface and evaluation of audio quality. Our primary focus is to evaluate and compare audio quality for different audio coders. Overall speech quality and network requirements depend in great deal on the type of codec being implemented. Different coders result in different data rates and Mean Opinion Scores (MOS) relating to subjective quality. We performed measurements of network throughput and subjective speech quality for three different coders: PCM, GSM, and G.723.1. Two sets of measurements were performed: the first set using previously recorded speech and music, and the second set during a live chat between two participants. Results have been analyzed and the coders have been compared in terms of network throughput and achieved subjective speech quality.

Keywords: audio quality, audio codec, virtual reality

1. Introduction

VR and multimedia applications on the Internet may be considered as having great market potential in areas such as entertainment, education/training, e-commerce, and data visualizations. Virtual Audio Chat (VAC) [6] is an example Web application that enables multiple users to join an audio chat by way of a Virtual Reality (VR) interface. User interactions with the VR interface range from the user's ability to choose a mobile phone model (from a variety of models and colors/textures) to be used when joining the chat, to choosing the type of coder to be used during the session.

The overall speech quality and network requirements of the audio chat depend in great deal on the type of coder being used, with different coders resulting in different data rates and Mean Opinion Scores (MOS). In general, parameters that may be used when comparing different coders include bit rate, quality of the decoded signal, overall delay, loss, complexity, use with a variety of sounds (speech, music, fax signals, etc.), and cost. The quality of the decoded signal is evaluated in both objective and subjective terms. Objective measurements are based on technical data, while subjective measurements describe human perception of a reconstructed signal and are often considered to be more meaningful than

objective measurements. The most widely used method for evaluating listening quality is the ITU-T standard MOS [3] based on a five-point opinion scale.

Our primary focus has been to evaluate and compare the audio quality for three different coders: PCM, GSM, and G.723.1 by performing measurements of network throughput and subjective speech quality. PCM is an example waveform coding technique defined by ITU-T standard G.711 based on the encoding of voice samples. GSM 06.10 (ETSI standard) and G.723.1 (ITU-T standard G.723,1) are examples of speech coders that transmit parameters relating to a model of the source signal, rather than sending samples or signal parameters. Such coders are known as vocoders. Two sets of measurements have been performed with the first set using previously recorded speech and music, and the second set during a live chat between two participants. In the first set of measurements a distinction was made between a recorded male voice, female voice, and music. Overall results have been analyzed and compared in terms of network throughput and achieved subjective speech quality for the three different coders we have focused on. The paper is organized as follows: Section 2 covers the design and development of the virtual audio chat application. A performance analysis is given in Section 3, including measurements of packet throughput and subjective evaluations of audio quality. Section 4 concludes the paper.

2. Multi-User Virtual Audio Chat

The VAC application enables real time audio communication over the Internet between a multiple number of users. It consists of three basic components:

- A (modified) Session Directory (sdr) tool [4] which enables a user to schedule and announce a multimedia session on the Multicast Backbone (MBone).
- A VRML model of a mobile phone.
- A Java applet that opens a Real-time Transport Protocol (RTP) based audio conference.

The *sdr* was used for the purposes of scheduling and announcing the audio chat on the Internet. A user wishing to initiate a session starts *sdr* and defines all necessary session parameters, such as the time the session is active and the media comprising the session. Once the parameters have been defined and the session announced, users connected to the Internet and running the *sdr* tool can choose to join (unless permission to join is denied), thus launching a Web browser from *sdr* and downloading a virtual gallery (Figure 1) of mobile phones from a Web server. The user navigates through the gallery and chooses to load any one of the displayed phones (Figure 2). By way of user interactions implemented in VRML, the user can first decide on the texture to be mapped to the model. After doing so, the user enters onto the phone the multicast address and port number specified by *sdr* (Figure 3). Three separate boxes allow the user to choose one of the offered coders to be used. A Java applet then opens an RTP based audio conference using this multicast address and port number.

2.1. Interaction with the VR interface

VRML is a file format that provides the means for describing interactive 3D worlds with integrated multimedia content in a distributed environment such as the Internet [2]. Figure 1 shows a virtual mobile phone gallery developed using VRML that allows a user to choose any one of the displayed phone models to be used when initializing an audio chat. After choosing and loading the desired phone model,

Figure 1. Virtual gallery of mobile phones

Figure 2. Mobile phone model with various textures

the user decides on the texture to be mapped to the model by clicking on one of the offered texture boxes (Figure 2). After having chosen the desired texture, a window is opened as shown in Figure 3 with the final phone model. User interactions include the following:

- Opening and closing of the active flip (on those models that have a flip).
- Interaction with the virtual keys.
- Lighting of the display.
- Choice of coder to be used during audio streaming.
- User control of buffer size during audio streaming.

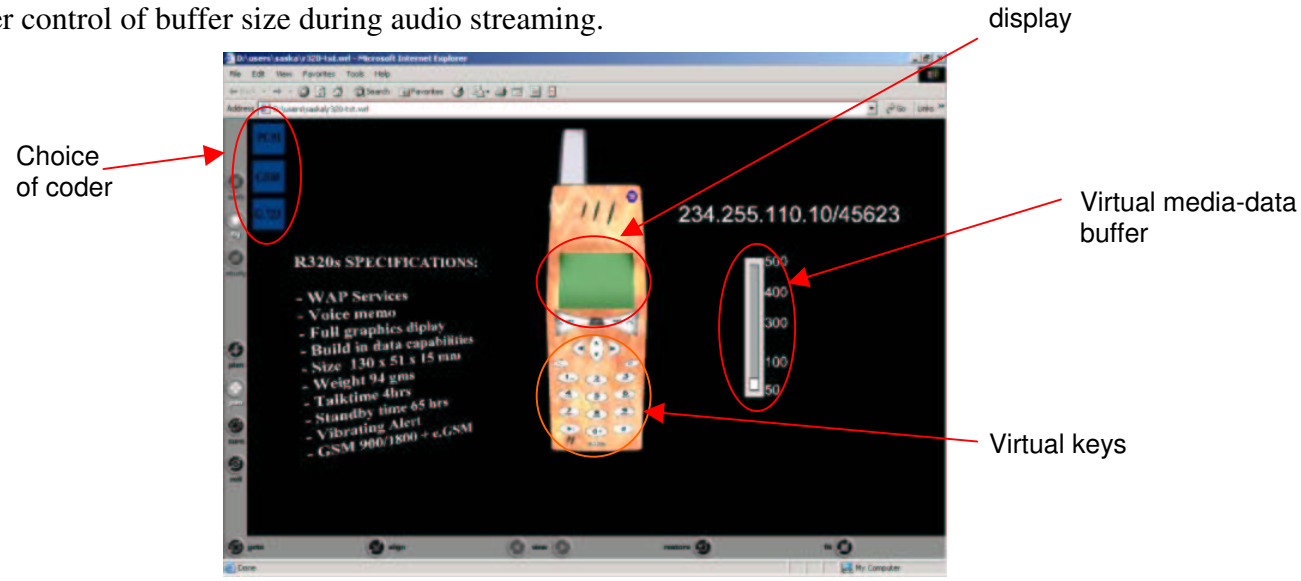

Figure 3. Mobile phone with virtual data buffer and coder indicators

Interactions and dynamic animations were enabled through the use of various Sensor nodes and Script nodes implemented in the VRML. Much use was made of the touch sensor, used to generate events in response to the position of the mouse or to mouse clicks on certain objects. Scripts written in JavaScript and Java were used to define how events related to user interactions were to be processed. The scroll bar

shown in the above figure represents a virtual media-data buffer. By setting the position of the bar, the user can control the amount of audio data that is buffered prior to being passed on to the next processing stage. This can be used to reduce the effect of jitter, or variations in latency, as perceived by the user during the chat.

2.2. Real time audio communication through the Internet

In order to join or initialize an audio chat on the Internet, a user enters a multicast address and port number by pressing the numbers on the virtual mobile phone with a mouse. The integration of the VRML and Java allowed us to make use of the Java Media Framework (JMF) API that provides a way for audio, video and other time-based media to be added to Java applications and applets [1]. An application for transmitting and receiving RTP data has been programmed to enable an audio conference between a multiple number of users. The RTPManager Java class defines methods for initializing, running, and closing an RTP session. Aside from the address and port number, the application requires parameters defining the location of the media source and the type of coder to be used for transmitting audio. Outgoing media streams may originate from a stored (local/remote) file or a capture device (i.e. microphone). At the receiving end, the application decodes an incoming media stream using any of the JMF supported coders. In this way, it is possible for multiple users taking part in an audio conference to use different coders when transmitting media. The speech quality corresponding to each participant depends on the particular codec being used and on network conditions.

The RTP [7] provides end-to-end delivery services for data with real-time characteristics such as interactive audio and video. It is designed so that the application may control the loss detection and recovery of packets (usually handled by lower levels in the communication architecture) enabling the use of simpler, unreliable protocols at the network and transport level. RTP works in combination with the RTP Control Protocol (RTCP) that monitors data delivery and provides control over data transport. RTP does not address resource reservation and does not guarantee QoS for real time services.

2.3. Modified session directory tool

Session Directory (sdr) is a tool designed for scheduling and announcing multimedia sessions on the MBone, a virtual multicast network that uses the physical infrastructure of the Internet and implements virtual multicast in software. A session may consist of a number of multimedia components, such as audio, video and shared text editing. Each component is represented by an appropriate user application that may be launched from sdr. In order to add support for our VAC application, we needed to modify *sdr* by enabling a new type of media, vrml_audio as seen in Figure 4.

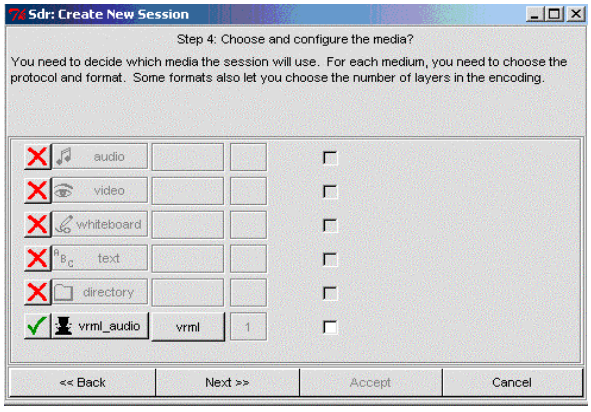

Figure 4. Sdr interface.

A new plug-in configuration file was written, associating a new media type vrml_audio, protocol RTP, user application vrml_audio.exe and file format vrml. When a user joins a session containing the media component vrml_audio, sdr then knows which user application to run. In this case, a Web browser is launched and a Web page containing our virtual mobile phone gallery is downloaded.

3. Audio Quality Evaluation

The overall speech quality and network requirements of the developed audio chat application depend in great deal on the type of coder being used. We focus our performance analysis on objective and subjective measurements conducted for three different coders:

- PCM: defined by ITU-T G.711 standard "Pulse code modulation for voice frequencies" (PCM) as a simple waveform coder based on the encoding of voice samples. It is characterized by a high quality of reconstructed signal (MOS score of 4.3), and small algorithm delay. The downside of this coder is a high bit rate of 64 kbit/s, making it not suitable for use on the Internet.
- GSM: defined by ETSI standard GSM 06.10: Full Rate Codec to be used for speech transmission in *Global System for Mobile communication* (GSM) systems. The coder is an example of a vocoder which transmits parameters relating to a model of the source signal. Data is transmitted at a rate of 13 kbit/s. The reconstructed signal for this coder is graded with a MOS score of 3.5.
- G.723.1: defined by ITU-T G.723.1 standard "Dual speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s". This is also a vocoder which gives a good quality of reconstructed speech (MOS score of 3.8) in spite of a low bit rate. Low bandwidth usage makes this coder suitable for Internet use.

Two sets of measurements have been performed: the first set using previously recorded speech and music, and the second set during a live chat between two participants taking part in a virtual audio chat. In both cases, network throughput and subjective evaluation of audio quality were determined for the mentioned coders. Throughput was measured using the Ethereal program (version 0.9) [5]. Ethereal enabled us to capture, filter, and analyze network traffic throughout all measurements.

3.1. Previously recorded speech and music

The first set of measurements was performed using previously recorded male speech, female speech, and music. Traffic was transmitted between two PCs (Pentium III, 733 MHz, 256 MB RAM) connected to a 10 Mbit/s Ethernet LAN. Packet capture was performed for each sample (male speech, female speech, and music) for a duration of 10 minutes. Ethereal enabled us to determine packet size and the time each packet was sent. Table I shows RTP packet size for each coder (including all headers: Ethernet, IP, UDP, RTP). The amount of data in each packet is also shown.

coder	packet size	data
PCM	534 bytes	492 bytes
GSM	153 bytes	111 bytes
G.723.1	102 bytes	60 bytes

Table I. RTP packets

RTP packet size corresponding to each coder remained unchanged for all three samples, while RTCP packet length varied depending on the type of report being sent (sender or receiver report, source description report, or goodbye report) and the information that it contained. The RTP/UDP bit rate for each coder remained independent of the sample being transmitted, while RTCP/UDP bit rate varied

depending on the sample. The generation of RTCP packets is also affected by network conditions. Results of bit rates for RTP and RTCP packets for all coders and samples are shown in Table II.

coder	RTP packet bit rate [bit/s]	RTCP packet bit rate [bit/s]		
		male voice	female voice	music
PCM	72919.04	298.53	312.88	313.55
GSM	20520.96	282.16	291.39	279.58
G.723.1	13926.40	287.84	287.39	283.79

Table II. Bit rates of RTP and RTCP packets

As shown in the above table, the greatest amount of network traffic was generated using the PCM coder while the G.723.1 coder generated the smallest amount of traffic. Slight differences in RTCP packet throughput are also visible for different samples encoded using the same codec. Figure 5, 6, and 7 show measured throughput for a female voice sample encoded with the three different coders.

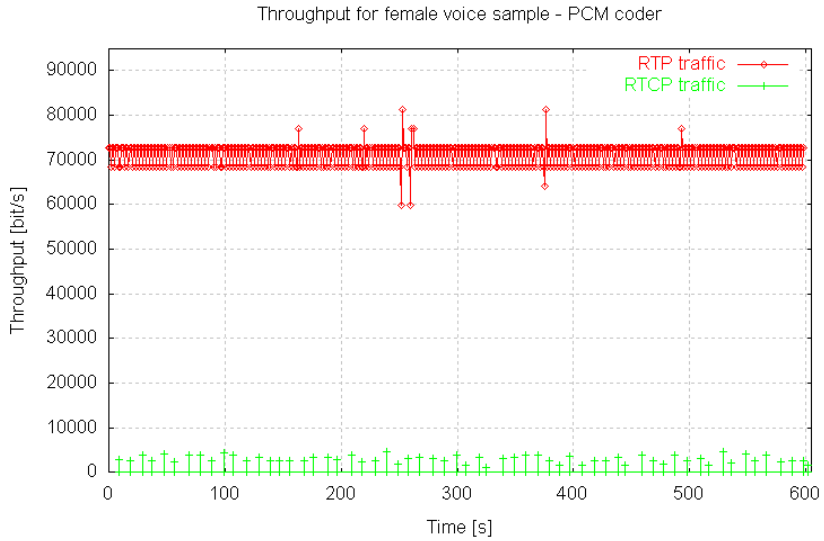

Figure 5. RTP and RTCP packet throughput for female voice sample encoded with the PCM coder

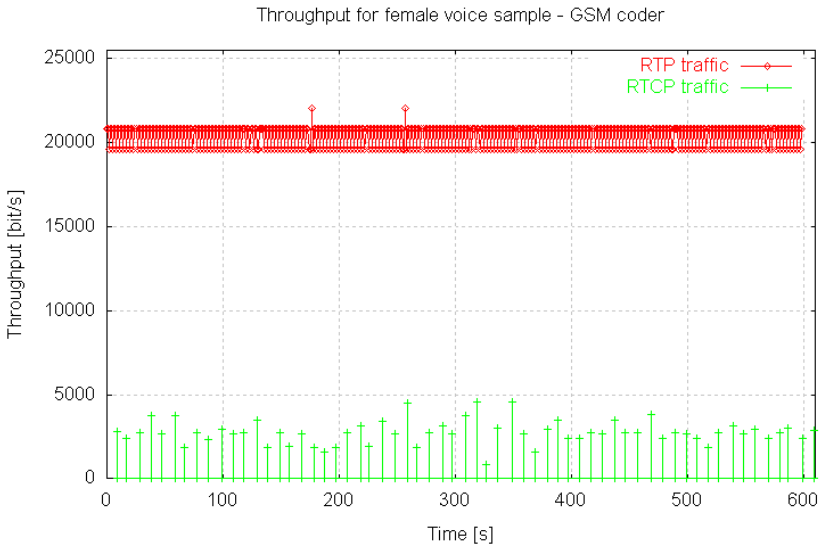

Figure 6. RTP and RTCP packet throughput for female voice sample encoded with the GSM coder

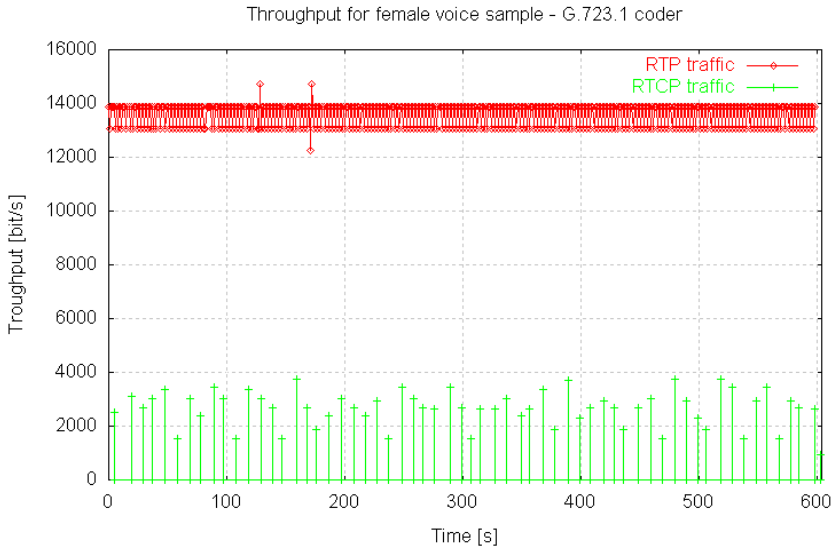

Figure 7. RTP and RTCP packet throughput for female voice sample encoded with the G.723.1 coder

The oscillation in the RTP traffic curve visible on all three graphs is related to a chosen time interval of 1 second for calculating throughput and to the effects of delay jitter (the choice of a finer time interval would have resulted in a "straighter" curve). Parallel to the generation of RTP traffic, we see the occasional generation of RTCP packets with information regarding QoS. Results such as these relating to measured throughput and RTCP packet information may then be used in the future for the purposes of resource reservation.

Subjective measurements focused on subjective estimation of speech (music) quality and comprehension of the decoded signal. The best quality and comprehension was achieved using the PCM coder, with decoded tones recorded to be slightly higher than the original. In the case of the other two coders, the decoded tones were smothered and deeper. GSM and G.723.1 coders gave similar results in quality and comprehension, even though the GSM coder has a higher bit rate. This shows that a higher bit rate may not necessarily give better quality. The quality of male and female voices encoded with the same coder did not noticeably differ, while all coders showed poorer quality when encoding music. Music samples have large and frequent frequency oscillations as opposed to speech samples that have different frequencies compared to one another, but with fewer oscillations. We can conclude therefore that frequent oscillations have a negative effect on the quality of the encoded signal. Results of subjective quality are summed up in Table III.

	male voice	**female voice**	**music**
PCM	good quality	good quality	good quality
G.723.1	fair quality (worse than PCM, approximately like GSM)	fair quality (worse than PCM, slightly better than GSM)	fair quality (worse than PCM, slightly better than GSM)
GSM	fair quality (worse than PCM, approximately like G.723.1)	fair quality (worse than PCM, little worse than G.723.1)	fair quality (worse than PCM, little worse than G.723.1)

Table III. Subjective quality of encoded samples.

3.2. Live audio chat

The second set of measurements was performed during a live chat between two participants using the VAC application. Both participants were connected to a 10 Mbit/s Ethernet LAN using a PC (Pentium III, 733 MHz, 256 MB RAM). The full VAC application was installed on both computers. Once again, Ethereal was used to capture all generated traffic. After choosing the mobile phone model, participants chose the coder to be used during the audio chat.

Three test scenarios were conducted for the three different coders: in each case both participants used the same coder. Table IV shows the bit rates for each participant in each scenario.

		average bit rate of RTP packets [bit/s]	average bit rate of RTCP packets [bit/s]
Scenario 1: PCM coder	Participant 1	71238.26	253.12
	Participant 2	71227.33	235.42
Scenario 2: GSM coder	Participant 1	20376.22	244.74
	Participant 2	20404.76	232.26
Scenario 3: G.723.1 coder	Participant 1	13602.75	269.08
	Participant 2	13601.80	232.40

Table IV. Average bit rates for live audio chat

As the above table shows, both participants in each scenario generated nearly the same amount of traffic. The application provides no support for silence detection, so traffic was generated regardless of whether users were taking or listening. It is visible that the generated throughput in the case of a live chat is nearly the same as when transmitting previously recorded speech using the same coder. The lowest demands on the network were imposed by the G.723.1 coder, which has a bit rate of 5.3 or 6.3 kbit/s. Figures 8 and 9 show graphical representations of network throughput generated by the two participants using the GSM coder.

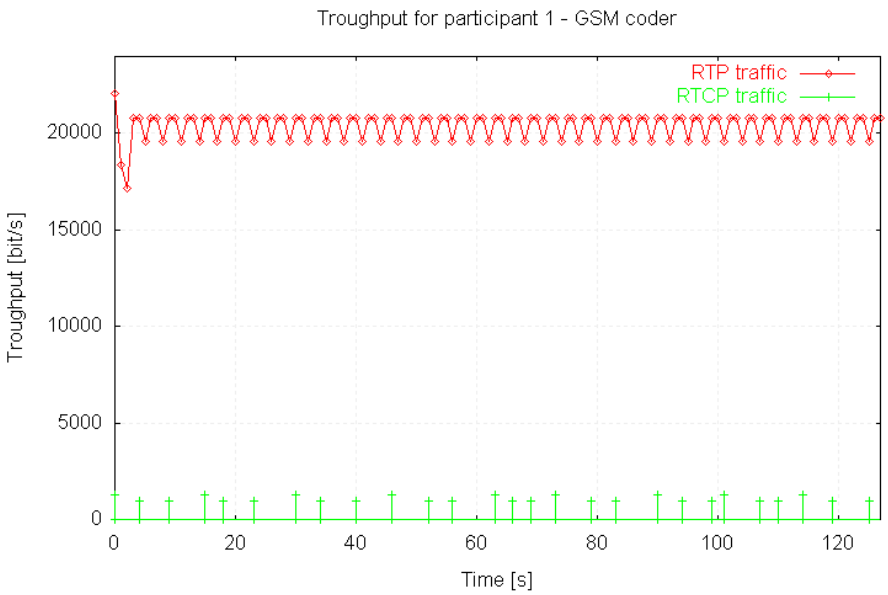

Figure 8. Throughput for participant 1 (GSM coder used).

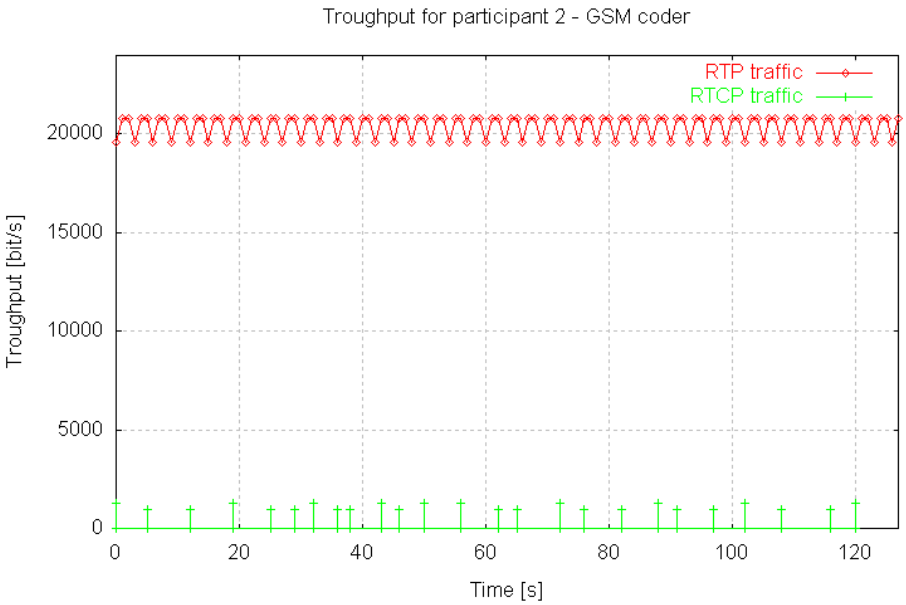

Figure 9. Throughput for participant 2 (GSM coder used).

A subjective analysis of speech comprehension gave good results for all three coders. Both participants understood each other and did not notice any difference in quality when using the GSM and G.723.1 coders, but noticed higher quality of speech while using the PCM coder. Although participants noticed changes in tones while using the GSM and G.723.1 coders, communication could go on normally without considerable effort.

4. Conclusions

In this paper we have described the development of a Virtual Audio Chat (VAC) application that enables real-time audio streaming between multiple Internet users. We first look at the development of VAC and the enabled interactivity with the VR interface. In the second part of the paper, we give a performance analysis based on evaluating audio quality and network requirements of the application using three different coders (PCM, GSM, and G.723.1) to encode audio. Measurements were first performed using previously recorded speech and music, and then during a live chat between two participants. Test results have been analysed for the purpose of comparing the three coders in terms of network throughput and achieved subjective speech quality.

References:

[1] --, Java Media Framework API, http://www.java.sun.com/products/java-media/jmf

[2] --, Information Technology – Computer graphics and image processing – The Virtual Reality Modelling Language (VRML) – Part 1: Functional specification and UTF-8 encoding, ISO/IEC 14772-1:1997, 1997.

[3] --, ITU-T Recommendation P.800, Methods for Subjective Determination of Transmission Quality, August 1998.

[4] --, Session Directory Tool, in Mbone Conferencing Applications, http://www-mice.cs.ucl.ac.uk/multimedia/software/sdr

[5] --, Ethereal, free network protocol analyzer, http://www.ethereal.com

[6] M. Matijasevic, L. Skorin-Kapov, Design and Evaluation of a Multi-User Virtual Audio Chat, to appear in Future Generation Computer Systems, 2003. Accepted for publication.

[7] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC 1889, January 1996.