Live Demonstration: Interactive Quality of Experience Evaluation in Kvazzup Video Call

Joni Räsänen, Aaro Altonen, Alexandre Mercat, and Jarno Vanne
Ultra Video Group, Tampere University, Tampere, Finland
{joni.rasanen, aaro.altonen, alexandre.mercat, jarno.vanne}@tuni.fi

Abstract—**This paper presents an interactive demonstration setup, which allows users to configure the video coding parameters of Kvazzup open-source video call software at runtime and evaluate their impact on Quality of Service (QoS) and Quality of Experience (QoE). The demonstration is carried out by implementing a new Kvazzup control panel for video call parameterization and visual quality, bit rate, latency, and frame rate evaluation.**

Keywords-Video call; Quality of Service (QoS); Quality of Experience (QoE); High Efficiency Video Coding (HEVC); Real-time Transport Protocol (RTP); Kvazzup video call application

I. INTRODUCTION

Video communication is increasing rapidly worldwide, and the COVID-19 pandemic has been reported to further accelerate the network traffic of VoIP and video conferencing by 210%-285% [1]. This growth is being mitigated by MPEG and ITU-T video coding standards, of which *High Efficiency Video Coding* (*HEVC*) [2] represents the state-of-the-art.

Quality of Service (*QoS*) and *Quality of Experience* (*QoE*) are key performance indicators in the current video call applications. Maximizing them in a video call requires careful optimization of subjective video and audio quality, but within the stringent limits of real-time performance, acceptable latency, and adequate bandwidth. Video coding has a particularly dominant role in these applications due to its high complexity and data-intensiveness. HEVC coding efficiency, complexity, and latency are all highly dependent on the selected coding parameters, so it is of utmost importance to examine the effects of individual HEVC parameters on QoS/QoE when developing automatic QoS/QoE management for video call applications.

The proposed demonstration addresses runtime video call parameterization by visualizing the effects of HEVC coding parameters on QoS/QoE. Our demonstration is carried out with Kvazzup [3] open-source video call software, which is built using the Qt application framework. It uses Kvazaar encoder [4] and OpenHEVC decoder [5] for video coding. For this demonstration, we implemented Video Settings that allow the users to modify the video coding parameters during the call and Statistics that allow users to analyze their effects on QoS/QoE. To the best of our knowledge, the proposed setup is the only open solution that supports dynamic video call parameterization.

The rest of this paper is organized as follows. Section II provides an overview of the Kvazzup video call configuration and Section III presents our interactive demonstration setup.

II. KVAZZUP VIDEO CALL CONFIGURATION

Fig. 1 illustrates an overview of the proposed Kvazzup control panel. It is composed of the *Video Settings* dialog box (on the left) and the *Statistics* window (on the right). The Kvazzup functionality is clarified by linking the windows to the respective stages of the implemented *Video Filter Graph* (*VFG*) (in the middle).

A. Video Settings

The Video Settings dialog box is used to configure the components of the VFG. It is organized into the following tabs.

- *Input*: selection of the camera input resolution, format, and frame rate. The coding parameters of Kvazaar are automatically set accordingly.
- *Parallelization*: parameterization of parallel coding tools of Kvazaar and assignment of the thread counts for the two video format conversions and HEVC decoding.
- *Structure*: assignment of the quantization and frame structure parameters of Kvazaar.
- *Compression*: selection of the Kvazaar preset and individual coding parameters.
- *Other*: selection between Qt's software rasterizer and OpenGL for video display.

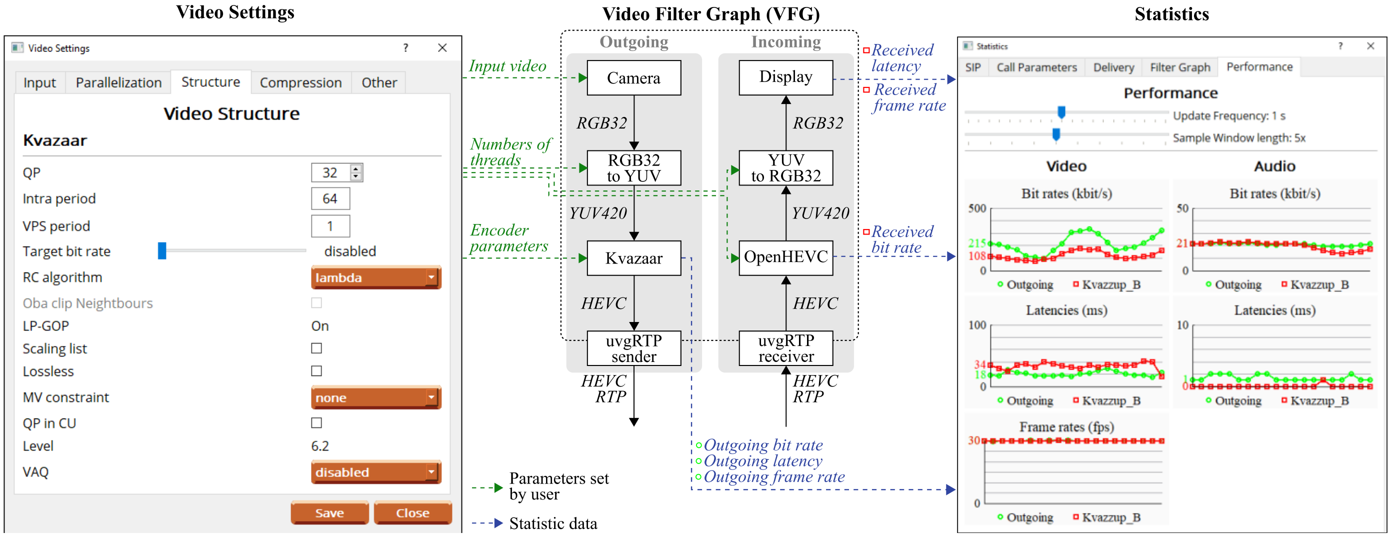
Kvazzup uses Qt to load and save these settings. If no previous settings exist when loading, default values are used.

B. Video Filter Graph

Kvazzup includes separate VFGs for outgoing and incoming video streams. The input video of the outgoing VFG is captured with Qt, converted from RGB32 to YUV 4:2:0 format, and encoded into HEVC by Kvazaar. The HEVC bitstream is then transported by the uvgRTP open-source library [6]. In the incoming VFG, the uvgRTP receiver filter receives the HEVC stream and OpenHEVC decodes it into YUV 4:2:0 raw video. Finally, this YUV video is converted to RGB32 and displayed to the user.

When the settings are changed, each filter in the VFG uses Qt to retrieve and apply the new settings. For example, Kvazaar is restarted with updated parameters. The changes take effect almost immediately after saving the settings and without interrupting the ongoing call, so the reparameterization is practically invisible to the user.

This work was supported by the Academy of Finland (decision no. 301820).



Video Settings

Input | Parallelization | Structure | Compression | Other

Video Structure

Kvazaar

QP	32
Intra period	64
VPS period	1
Target bit rate	disabled
RC algorithm	lambda
Oba clip Neightbours	□
LP-GOP	On
Scaling list	□
Lossless	□
MV constraint	none
QP in CU	□
Level	6.2
VAQ	disabled

Save Close

Statistics

SIP | Call Parameters | Delivery | Filter Graph | Performance

Performance

Update Frequency: 1 s

Sample Window length: 5x

Figure 1. Kvazzup control panel and linking to the actual implementation.

C. *Statistics*

Statistics window can be used to analyze QoS/QoE with objective metrics. It is divided into the following tabs:

- *SIP*: shows a log of Session Initiation Protocol messages.
- *Call Parameters*: shows network address details for media.
- *Delivery*: shows a graph of the incoming and outgoing bandwidth usage.
- *Filter Graph*: shows the status of the VFG filters in terms of their buffer size and total dropped frames.
- *Performance*: shows the performance statistics of the VFG for audio and video.

The Performance tab of the Statistics window is active in Fig. 1. It plots the average bit rate, latency, and frame rate values to graphs at defined intervals. The graphs show incoming (green) and outgoing (red) VFG statistics separately and they are updated at runtime. The analysis frequency can be set within the range of 0.1 s to 2.0 s and the sample window within the range of 1× to 10×.

III. Interactive Demonstration Setup

Fig. 2 depicts a practical setup of our two-way end-to-end Kvazzup demonstrator. It is composed of two laptop computers (with webcams) connected over a local area network. A Kvazzup video call is set up between the Kvazzup clients (i.e., Kvazzup *A* and *B*) run on the computers. At both ends, the Video Settings and Statistics windows are open for parameter setup and visualization. In addition, Windows Task Manager is used to monitor CPU usage.

During the demonstration, users can interact with Kvazzup clients by modifying the video call parameters in Video Settings dialog box and see how changes take effect in the ongoing call. When the user changes the settings of Kvazzup *A*, the effects become visible after a short period of time on the outgoing video of Kvazzup *A* and the incoming video of Kvazzup *B*, and vice versa. In practice, the user is able to evaluate the impacts of these changes on QoS/QoE in real-time by looking either at the outgoing and incoming VFG statistics or live video streams.

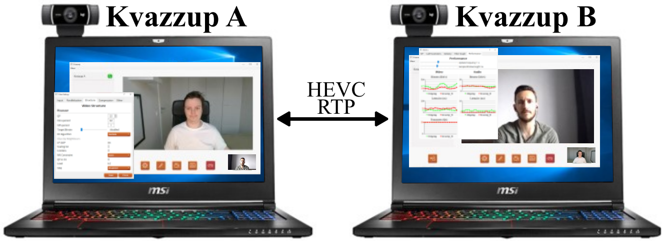


Figure 2. Demonstration setup.

This demonstration setup can be used to evaluate the effects of the parameters individually or the combined effects of multiple parameters. For instance, increasing the *Quantization Parameter* (*QP*) value decreases bit rate and latency, but it also deteriorates visual video quality.

In a broader sense, maximizing the QoS/QoE of a video call is a multidimensional design problem. The ultimate purpose of this work is to provide the designers with a reference framework to explore video coding settings interactively and thereby facilitate the development of automatic QoS/QoE management in the current and future video call applications.

References

[1] Comcast Corporation, "COVID-19 network update," May 2020, [Online], Available: https://corporate.comcast.com/covid-19/network/may-20-2020.

[2] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, Dec. 2012, pp. 1649–1668.

[3] J. Räsänen, M. Viitanen, J. Vanne, and T. D. Hämäläinen, "Kvazzup: open software for HEVC video calls," *in Proc. IEEE Int. Symp. Multimedia*, Taichung, Taiwan, Dec. 2017.

[4] A. Lemmetti, M. Viitanen, A. Mercat, and J. Vanne, "Kvazaar 2.0: fast and efficient open-source HEVC inter encoder," *in Proc. ACM Multimedia Syst. Conf.*, Istanbul, Turkey, June 2020.

[5] W. Hamidouche, M. Raulet, and O. Déforges, "4K real-time and parallel software video decoder for multilayer HEVC extensions," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, Sept. 2015, pp. 169–180.

[6] A. Altonen, J. Räsänen, J. Laitinen, M. Viitanen, and J. Vanne, "Open-source RTP library for high-speed 4K HEVC video streaming," *in Proc. IEEE Int. Workshop on Multimedia Signal Process.*, Tampere, Finland, Sept. 2020.