Open SVC decoder: a flexible SVC library

Médéric Blestel, Mickaël Raulet

HAL Id: hal-00560027

https://hal.archives-ouvertes.fr/hal-00560027

Submitted on 27 Jan 2011

Open SVC Decoder: a Flexible SVC Library

Médéric Blestel
IETR/Image group Lab
UMR CNRS 6164/INSA
France
mblestel@insa-rennes.fr

Mickaël Raulet
IETR/Image group Lab
UMR CNRS 6164/INSA
France
mraulet@insa-rennes.fr

ABSTRACT

This paper describes the Open SVC Decoder project, an open source library which implements the Scalable Video Coding (SVC) standard, the latest standardized by the Joint Video Team (JVT). This library has been integrated into open source players The Core Pocket Media Player (TCPMP) and mplayer, in order to be deployed over different platforms with different operating systems.

Categories and Subject Descriptors

D.2.0 [**Software Engineering**]: General

General Terms

Algorithms, Performance

Keywords

Scalable Video Coding, mplayer, TCPMP, Joint Video Team, Joint Scalable Video Model

1. INTRODUCTION

In October 2007, the Joint Video Team (JVT), composed of the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) has standardized a Scalable Video Coding (SVC) [1] extension of the H.264/AVC standard [2]. The two groups have also developed a reference software, named the Joint Scalable Video Model (JSVM) [3] which implements this standard.

The SVC standard extends the AVC standard with scalability features allowing easy adaptation of the bit-stream to network and terminal constraints. SVC provides a high degree of flexibility in terms of scalability dimensions supporting various temporal or spatial resolutions and quality levels.

The Open SVC Decoder [4] [5], an open source library, aims to implement this new standard. It has been developed using a data flow methodology and tested over differ-

ent platforms like x86 platform, Personal Data Assistant, PlayStation 3 and Digital Signal Processor.

In this paper, a brief description of the SVC standard is done, followed by a presentation of the Open SVC Decoder (OSD) and its installation procedure.

2. SVC OVERVIEW

Scalable Video Coding (SVC), the new scalable codec based on H.264/AVC standard, has been developed to broadcast a video to multiple users with multiple displays and connected through multiple networks using a single bit-stream. Thanks to this standard, it is possible to partially transmit or decode a video bit-stream resulting in various video qualities.

In the SVC standard, the video compression is performed by generating a unique hierarchical bit-stream structured in several layers of information, consisting of a base layer and several enhancement layers. The base layer provides basic quality while the enhancement layers provide improved quality at increased computational cost. The particularity of the bit-stream is that the enhancement layers can be removed without compromising the decoding process of the resulting bit-stream.
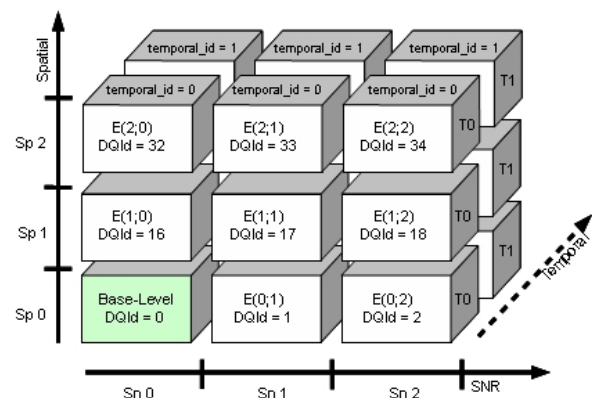

Figure 1: Evolution of layers parameters

Figure 1 shows an example of a bit-stream with several enhancement layers: 2 spatial, 2 SNR and 1 temporal. Each layer E(dependency_id; quality_id), can be easily identified thanks to two parameters, the "DQId" (derived by: (dependency_id $<< 4$) + quality_id) and the "temporal_id". These

parameters refer to the different scalabilities present in the layer among the three types of scalability specified in the standard:

- *Temporal*: The temporal scalability which was already present in H.264/AVC, has been improved by partitioning the hierarchical B and P frames in different temporal layers. The temporal scalability of each frame can be determined thanks to the "temporal_id" parameter.

- *Spatial*: This type of scalability tries to increase coding efficiency by exploiting the redundant information between layers. In addition of the basic prediction mode defined in H.264/AVC, a new macroblock coding mode has been defined to predict from a base layer, macroblock's information thanks to upsampling mechanisms. Each spatial enhancement should have a different "dependency_id".

- *Quality*: The quality scalability is very similar to the spatial scalability but without changing of resolution between layers. The same inter-layer predictions are used except the up sampling mechanisms. The quality enhancement can be referred either by the quality_id or by the dependency_id following the SNR scalability chosen.

In the SVC amendment [2], temporal and quality scalabilities should be supported by decoders without any restrictions. Only the spatial scalability is limited according to the profile defined. There is three profiles in the SVC amendment:

- *The Scalable Baseline profile*: In this profile, the base layer must conform to H.264 / AVC Baseline profile. Resolution ratios between successive spatial layers in both horizontal and vertical direction must be 1.5 or 2.

- *The Scalable High profile*: In this profile, the base layer must conform to H.264 / AVC High profile. No restrictions have been defined for the enhancement layers.

- *The Scalable High Intra profile*: Only intra pictures in all layers are authorized. As the previous profile, the base layer should be conforming to H.264 / AVC High profile.

3. OPEN SVC DECODER

IETR is developing since July 2006 the Open SVC Decoder, a C language Scalable Baseline profile library supporting all tools to deal with spatial, temporal and fidelity scalabilities. It is based on a fully compliant H.264 / AVC Baseline library with most of Main profile tools.

In this section, the library features and its conformance are presented.

3.1 Open SVC Decoder features

Contrary to the JSVM which decodes only the layer with the highest scalability, i.e the enhancement layer with the highest spatial, temporal and quality scalability, the Open SVC Decoder can decode partially the bit-stream until a

specific layer with a specific temporal scalability. This particularity provides an adaptability of the decoder over different platforms by selecting the right layer in order to have a real-time decoding.

The library contains also several mechanisms to switch of layer during the decoding process which allows the user to select the layer to display by specifying commands. However, the decoder is also able to change of layer by itself when a missing enhancement occurs due to transmission errors.

In the case of a partial decoding of a bit-stream, the decoder will dismiss discardable layer. Figure 2 shows the data flow graph of the decoding process when the top layer of a four layers stream is not decoded. Variable Length Coding and Texture Decoding are processes for the first three layers but not for the fourth.
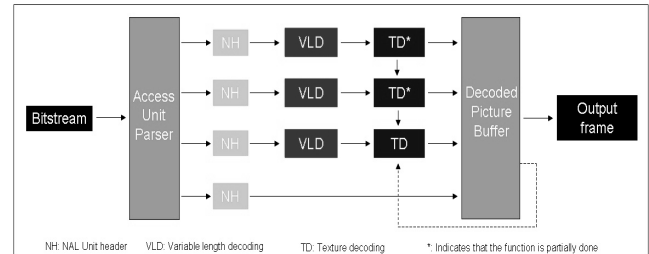

Figure 2: Open SVC Decoder data flow graph

3.2 OSD conformance and benchmarks

The Open SVC Decoder has been compared to the JSVM 9.19 to benchmark and to test the conformance of the library using conformance sequences which can be found on the JVT site [6]. A list of the conformance sequences in which Open SVC Decoder is compatible with, is available on the web site [7].

Table 1 shows the results of the comparison between both decoders on several conformance sequences. The benchmarks were executed on a PC with Intel Core 2 Duo CPU at 2.4GHz.

Table 1: Open SVC Decoder Benchmark

Sequence	Decoding time (s)		Speed up
	JSVM	OSD	
SVCBST-1	31.2	0.87	35
SVCBST-2	23.3	0.87	26
SVCBST-14	137	2.69	50
SVCBST-15	50	2.11	23

These benchmarks show the speeding up between the Open SVC Decoder and the JSVM decoder on several conformance sequences with different configurations. Indeed, the performance of the library is up to 50 times faster than the JSVM decoder.

4. OSD INSTALLATION AND EXECUTION

The library has been built into two open source players, TCPMP and mplayer which were using ffmpeg video codec [8]. This codec has been removed and replaced by the Open SVC Decoder in order to decode AVC and SVC bit-streams.

In this section, only the installation and execution of mplayer is presented. Others installation procedures are presented on the project web site [4].

4.1 Installation

We have decided to only present mplayer installation which is simplest than the TCPMP one. Moreover, this player is compatible with all major operating systems, including Linux and other Unix-like systems, Microsoft Windows and Mac OS X, and has been tested over different platforms.

The mplayer version proposed in this project is based on the 31411th revision of the mplayer SVN repository [9]. No major changes have been done except for the interfacing of the library into the mplayer.

The installation proceeds in several steps:

- *Download the sources*: The first step consists in downloading the source of the project [10]. The latest release, 1_08 should be downloaded. The package is organized in three different directories: the library sources (Libs), mplayer directory and TCPMP directory.

- *Download a MinGW package*: This step is necessary only if the operating system is Windows. Indeed, mplayer under Windows, is only compatible with MinGW build environment which can be downloaded on this page [11]. The compilation with CygWin or others built environments is not working.

- *Download SDL library*: This step is necessary only if the operating system is different from Windows. The SDL library [12] is used to manage the layer to display.

- *Configure the player*: Once all packages downloaded, the next step consists in configuring the player by using the following the command line "./configure –enable-svc" in a console (MinGW if the operating system is Windows) at the root of the mplayer directory. The option "–enable-svc" allows the player to be configured with the SVC library located in the "Libs" directory. Without specifying this option, mplayer will be configured with the ffmpeg library and only the base layer will be decoded in case of a SVC bit-stream.

- *Start the compilation*: The compilation of all sources can be started using the command line: "make". Once the compilation done, the program mplayer is ready for use.

4.2 Execution

Once the installation finished, the program can be executed. Contrary to TCPMP which can only read multimedia file formats, mplayer can also read raw streams. The command line to execute mplayer is quite simple: "./mplayer file_name".

However, to decode SVC raw streams like conformance sequences from the JVT site, the option "-fps" followed by the frame rate (25 as default) has to be specified in the command line. Moreover, if the operating system is Mac OS X, the option "-vo sdl" should be added in order to use SDL library to correctly display the output video.

As default, the decoding process starts by displaying the enhancement layer with the highest scalability. Options can be added to the command line to select a specific layer to decode with a specific temporal scalability.

For instance, the option "-setlayer" followed by a DQId value, allows to select the layer with the DQId specified present in the bit-stream. To select the temporal level to decode, you can specify the option "-settemporalid" followed by the temporal_id wanted.

For example, the following command line "./mplayer -fps 25 mystream.264 -setlayer 16 -settemporalid 2 -vo sdl" allows to decode all layers necessaries to display the layer with the DQId equals to 16, of the bit-stream named "mystream.264". The decoder will display at 25 frames per second and using the SDL library, all frames of this layer with a "temporal_id" inferior or equals to 2.

The configuration of the DQId and the temporal scalability can be changed during the decoding process thanks to hot-keys:

- *"l"*: To select a layer with a lowest spatial or quality scalability if the layer currently displayed has a DQId different from 0.

- *"b"*: To select a layer with a higher spatial or quality scalability if the currently displayed layer has not the maximum value of DQId present in the bit-stream.

- *";"*: To switch down of temporal scalability. The frame rate will not change, so the bit-stream will be decoded faster.

- *","*: To switch up of temporal scalability.

Using this library, the user can change of temporal layer at any time whereas he can change of spatial and quality layer only on frames set as IDR (Instantaneous Decoding Refresh). This type of frame occurs occasionally in the sequence from the JVT site, that's why video bit-streams with multiple IDR frames are available on the project download page [10]. These sequences allow to regularly switch between layers.

5. AUDIENCE

The Open SVC Decoder has been developed in the framework of Scalim@ges [13] project in a modular manner [14]. This project aimed to promote SVC standard in order to reduce the number of formats manipulated in production, distribution, and use of video compatible with existing solutions. Partners of this project had actively contributed to the standardization of the SVC standard and its transport stream protocol [15].

Currently, others French and international projects like SVC4QoE [16] and ScalNet [17] are using Open SVC Decoder. ScalNet tries to focus on designing a streaming system based on the SVC standard. The system is designed to cope with streaming scenarios that can be classified in four use cases: session handover, network congestion, receiver heterogeneity and user driven adaptation.

The SVC4QoE project aims at associating Scalable Video Coding and Quality of Experience (QoE) evaluation techniques in order to maximize the user experience when receiving audiovisual content over broadcast networks. The purpose of the project is to explore and demonstrate how to optimize the broadcast network infrastructure in various receiving conditions by lowering the necessary bandwidth, while providing the best quality of the signal to a wide range of portable and hand-held receivers.

The Open SVC Decoder is likewise used as a starting point in the development of a new SVC library in the framework of MPEG-Reconfigurable Video Coding [18].

6. CONCLUSION

Since May 2009 when this project had been uploaded on sourceforge web site, Open SVC Decoder has been downloaded 7,100 times and the SVN repository has next to 2500 read transactions. These results show the interest of the community into this video codec and open source code.

The web site has also permitted us to meet and work with foreign teams in order to enhance the library. Optimized algorithms are regularly added to the Open SVC Decoder source code.

The speeding up's library compared to the JSVM, and its portability over different platforms make the Open SVC Decoder a relevant candidate for research projects

7. ACKNOWLEDGMENTS

8. REFERENCES

[1] H. Schwarz, D. Marpe and T. Wiegand, "Overview of the Scalable Extension of the H.264 / AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, Sept 2007.

[2] ITU-T and ISO/IEC JTC 1, ITU-T Recommendation H.264 - ISO/IEC 14496-10(AVC), "Advanced Video Coding for Generic Audio Visual services, Amendment 3: Scalable Video Coding," 2007.

[3] Joint Video Team, "Reference software," `http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm`.

[4] M. Blestel, "Open SVC Decoder Web site," `http://opensvcdecoder.sourceforge.net/`, 2009.

[5] M. Blestel and M.Raulet, "Open source code of an svc decoder," *SIGMultimedia Rec.*, vol. 1, no. 4, pp. 24, December 2009.

[6] Joint Video Team, "Conformance testing," `http://wftp3.itu.int/av-arch/jvt-site/bitstream_exchange/SVC/`, 2008.

[7] M. Blestel, "Open SVC Decoder Conformance," `http://opensvcdecoder.sourceforge.net/JVT-AB023.xls`.

[8] ffmpeg, "ffmpeg web site," `http://ffmpeg.org/`.

[9] mplayer, "mplayer svn repository," `svn://svn.mplayerhq.hu/mplayer/trunk`.

[10] M. Blestel, "Open SVC Decoder download page," `https//sourceforge.net/projects/opensvcdecoder/files/`, 2009.

[11] MPlayer for Win32, "Mplayer for win32 web site," `http://sourceforge.net/project/downloading.php?groupname=mplayer-win32&filename=MinGW-full-gcc-4.2.4.7z&use_mirror=freefr`.

[12] Simple Direct Media Layer, "Simple Direct Media Layer," `http://www.libsdl.org/`.

[13] "Scalim@ges project," `http://www.images-et-reseaux.com/en/les-projets/fiche-projets-finances.php?id=125`.

[14] M. Pelcat, M. Blestel, M. Raulet, "From AVC decoder to SVC: Minor impact on a data flow graph description," *Picture Coding Symposium*, November 2007.

[15] ISO/IEC 13818-1:2007/Amd 3:2009, *Transport of scalable video over Rec. ITU-T H.222.0 | ISO/IEC 13818-1*, 2009.

[16] M. Barkowsy, M. Blestel, M. Carnec, A. Ksentini, P. Le Callet, G. Madec, R. Monnier, JF. Nezan, R. Pepion, Y., JF. Travers, M. Raulet, and A. Untersee, "Overview of the svc4qoe project," in *Workshop on Impact of Scalable Video Coding on Multimedia Provisioning (SVCVision 2010)*, Lisbon, Portugal, September 2010.

[17] M. Ransburg, E. Martínez Graciá, T. Sutinen, J. Ortiz, M. Sablatschan, and H. Hellwagner, "Scalable Video Coding Impact on Networks," in *Workshop on Impact of Scalable Video Coding on Multimedia Provisioning (SVCVision 2010)*, Lisbon, Portugal, September 2010.

[18] M. Mattavelli, I. Amer, and M. Raulet, "The Reconfigurable Video Coding standard [Standards in a Nutshell]," *IEEE Signal Processing Magazine*, pp. 159 – 167, 05 2010.

[19] F. Pescador del Oso, D. Samper, M. Garrido Gonzalez, E. Juarez Martinez, and M. Blestel, "A DSP based SVC IP STB using Open SVC Decoder ," *International Symposium on Consumer Electronics*, June 2010.