USING SCALABLE VIDEO CODING FOR DYNAMIC ADAPTIVE STREAMING OVER HTTP IN MOBILE ENVIRONMENTS

Christopher Müller[1], Daniele Renzi[2], Stefan Lederer[1], Stefano Battista[2], and Christian Timmerer[1]

[1]Alpen-Adria-Universität Klagenfurt, Universitätsstraße 65-67, 9020 Klagenfurt, Austria, *{firstname.lastname}@itec.aau.at*
[2]bSoft ltd, 156 via Velini, 62100 Macerata, Italy, *{firstname.lastname}@bsoft.net*

ABSTRACT

Dynamic Adaptive Streaming over HTTP (DASH) is a convenient approach to transfer videos in an adaptive and dynamic way to the user. As a consequence, this system provides high bandwidth flexibility and is especially suitable for mobile use cases where the bandwidth variations are tremendous. In this paper we have integrated the Scalable Video Coding (SVC) extensions of the Advanced Video Coding (AVC) standard into the recently ratified MPEG-DASH standard. Furthermore, we have evaluated our solution under restricted conditions using bandwidth traces from mobile environments and compared it with an improved version of our MPEG-DASH implementation using AVC as well as major industry solutions.

Index Terms— Dynamic Adaptive Streaming over HTTP, MPEG-DASH, Scalable Video Coding, Evaluation, Mobile Networks, Vehicular Mobility

1 INTRODUCTION

Dynamic Adaptive Streaming over HTTP (DASH) has the potential to play a major role in networks with fluctuating bandwidth. The major industry players (e.g., Microsoft, Apple, and Adobe) have already adopted it and several streaming providers like Netflix, Hulu, Vudu, and Amazon, which are using the Hypertext Transfer Protocol (HTTP) for their streaming service. One major advantage of HTTP is its convenience for the end user, as well as for the streaming provider. Nevertheless, TCP and HTTP introduces a significant overhead compared to RTP and UDP, which is definitely a disadvantage [1]. Another fact that cannot be handled by traditional HTTP streaming (i.e., progressive download) are varying bandwidth conditions.

Dynamic Adaptive Streaming over HTTP aims to address this fact with a quite simple yet effective approach. Instead of having one media file encoded at a single bitrate, the same media file will be encoded at several bitrates, resolutions, etc. These multiple versions of same media will be then chopped into segments that can be individually requested by the client through HTTP. This enables the client to switch between different qualities, resolutions, etc. during the streaming session. Furthermore, the clients can be served through ordinary Web servers, which let the system scale very well. As mentioned at the beginning the industry has already deployed several solutions and also MPEG has recently ratified DASH as international standard [2].

Typically, AVC will be used to generate multiple qualities of the media for DASH but also the scalable extensions [3] of AVC are suitable and can potentially bring some major advantages due to its layered architecture, which enhances the flexibility of the segment selection. That is, in comparison to AVC it is possible to cancel a segment request at the layer boundaries. That advantage could simplify the adaptation process because this allows the client trying to download up to the highest quality and in case of insufficient bandwidth it could cancel the request at the segment boundaries. This is not possible with AVC because when the client cancels a segment download, the video data of that segment becomes useless.

The goal of this paper is to improve our existing MPEG-DASH implementation using AVC [4] and evaluate a SVC-based solution on top of it. In particular, we have compared both solutions among themselves as well as with the major industry solutions. In anticipation of the results we can conclude that we have achieved a major improvement of our own implementation [4], which performs now better than all industry solutions. Furthermore, our SVC-based solution could utilize a higher overall bandwidth compared to AVC.

The remainder of this paper is organized as follows. Section 2 describes related work and Section 3 describes our integration of SVC into MPEG-DASH. The experimental setup and results are described and discussed in Section 4. Finally, the paper is concluded in Section 5.

2 RELATED WORK

Sanchez et al. [5] has already described potential benefits of SVC with DASH. However, their main focus is on cache performance and encoding while we focus on the adaption process itself within mobile environments. Kofler et al. [6] has evaluated the implications of the ISO Base Media File Format (ISOBMFF) on adaptive HTTP Streaming of SVC and shown that the ISOBMFF is not suitable for SVC streams below 1 Mbps. Akhsabi et al. [7] evaluated Microsoft Smooth Streaming, Adobe HTTP Dynamic Streaming, and the Netflix Player using simulated


```
<Representation id="0" width="704" height="480" bandwidth="451231">
  <SegmentList>
    <SegmentURL media="bunny_480p24.vh4" mediaRange="141-74012" />
  </SegmentList>
</Representation>
<Representation id="1" dependencyId="0" width="704" height="480" bandwidth="550737">
  <SegmentList>
    <SegmentURL media="bunny_480p24.vh4" mediaRange="306392-647144"/>
  </SegmentList>
</Representation>
```

Figure 1. Excerpt of a Simplified MDP for SVC.

bandwidth traces. They used different test content for each system in question and, thus, the results are difficult to compare. Yao et al. [8] evaluated the possibility of using HTTP streaming under vehicular mobility within 3G mobile networks. The evaluation is based on real-world bandwidth traces using their own, proprietary client. However, their evaluation focused on the comparison of their system with non-adaptive HTTP streaming, i.e., progressive download whereas our evaluation compares AVC and SVC as well as the major industry solutions with real-world bandwidth traces under vehicular mobility. In our previous work [4] we have already evaluated our MPEG-DASH implementation using AVC with the major industry solutions. In this paper we have improved our solution and compared it also with SVC.

3 INTEGRATION OF SVC INTO MPEG-DASH

This section describes the integration of SVC into MPEG-DASH. Based on the observations by Kofler et al. [6] we have only used elementary streams and described the layer dependency in the Media Presentation Description (MPD) as depicted in Figure 1 which shows an excerpt of a simplified MPD. The MPD contains one base layer with a bandwidth of 451231 bps and an enhancement layer with a cumulative bandwidth of 550737 bps that depends on the base layer. The segments in Figure 1 correspond to 2 seconds of video data, which can be obtained through byte range requests, e.g., the base layer segment starts from byte 141 and ends at byte 74012 and comprises multiple base layer Network Abstraction Layer units (NALU). When the client selects the representation with 550737 bps it would initially download the corresponding segment of the representation which this representation depends on, i.e., the representation with 451231 bps followed by the enhancement layer. However, this download scheme would produce a bitstream – if segments are simply concatenated – that is not valid for the decoder due to the fact that the decoder would get a bunch of base layer NALUs followed by a bunch of enhancement layer NALUs and therefore looses all dependencies. As a consequence we had to reorder the NAL units at the client within our C++ dynamic link library (DLL) libdash that is open source available at [9]. The DLL handles the whole connection setup, XML parsing, and bandwidth adaptation process. Furthermore, with its internal buffer it provides a stable stream for the

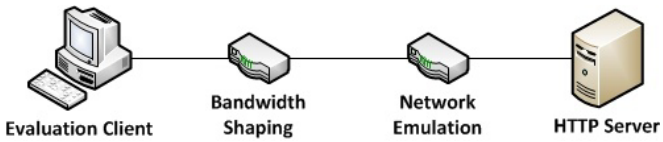

Figure 2. Experimental Setup [4].

caller of the library and it supports HTTP/1.1 persistent connections and pipelining. We have modified this library so that it could also handle the described SVC case and would produce a valid bitstream, e.g., in our example this would mean that the output of libdash is always one base layer NALU followed by the corresponding enhancement layer NALU. Without that modification we would have to describe each NALU in the MPD and request each NALU with an individual byte range request which is very ineffective due to the fact that each of these HTTP requests contains a header. This modification enables the efficient streaming of media with a lower bitrate than 1 Mbps which is needed for our experiments described in the following.

4 EXPERIMENTS

Please note that in this paper we have adopted the experimental setup, methodology, and content from our previous paper [4]. The most important aspects are described here for the sake of completeness but for details the interested reader is referred to [4].

4.1 DASH Content

We have encoded Big Buck Bunny for the AVC experiment with x264 at 14 different bitrates (100, 200, 350, 500, 700, 900, 1100, 1300, 1600, 1900, 2300, 2800, 3400, and 4500 kbps) with a Group of Pictures (GOP) size of 48 frames resulting into segments of 2 seconds length. For the SVC experiment the same content has been encoded with our own encoder using a GOP size of 48 frames and at 13 different bitrates with one base layer at 450 kbps and 12 enhancement layers (550, 650, 700, 1100, 1300, 1500, 1800, 2300, 2600, 3000, 3500 and 4700 kbps) where each layer depends on the previous one. An optimal combined selection of Coarse-Grain (CGS) and Medium-Grain Scalability (MGS) has been used, in order to act only on the fidelity of the encoded video stream, as we did in the AVC experiment, instead of on resolution or framerate.

4.2 Bandwidth Traces

All experiments have been evaluated under three different network emulation settings (tracks 1-3) that have been recorded during separate freeway car drives with a HUAWEI E169 HSPDA USB Stick using a SIM-card of the Austrian cellular network provider A1 [4].

4.3 Evaluation Metrics

Our main metric is the **average bitrate** that could be seen as the overall performance of the system at a particular test setup. The **number of quality switches** is another metric that describes the variance of the session. High values indicate very frequent switching which can lead to a decreased Quality of Experience (QoE) [10]. The **buffer level** describes the current fill state of the buffer and is an indicator for the stability of the system. The **number of unsmooth seconds** describes the smoothness of the session and may also influence the QoE. It can be derived from the buffer level and describes the time when the buffer is empty. Therefore, a high value of unsmooth seconds indicates a more jerky session.

4.4 Experimental Setup

The experiments have been performed using the setup depicted in Figure 2. This setup comprises four nodes, namely the evaluation client, bandwidth shaping, network emulation, and HTTP server.

The bandwidth shaping controls the maximum achievable bandwidth for the client and the network emulation controls all network related parameters such as round trip time (RTT). Based on our measurements the RTT has been set to 150ms [11]. For the HTTP server we used the Apache Web server which handles the HTTP requests from the client. The HTTP server, the bandwidth shaping node, and the network emulation node are based on Ubuntu 10.04. The evaluation client slightly differs for the individual experiments using AVC and SVC and will be described in the following subsections.

4.5 MPEG-DASH and AVC

The evaluation of MPEG-DASH AVC is based on our VLC plugin and the content has been generated using our DASHEncoder – both available at [9] – which is a wrapper tool for x264 and MP4Box. The client for the experiment is based on Ubuntu 10.04 and the DASH VLC Plugin has been modified to support HTTP/1.1 persistent connections and pipelining. Furthermore, we have improved our implementation from [4] and used a new buffer model which is depicted in Figure 3 in combination with the average measured bandwidth for the adaptation.

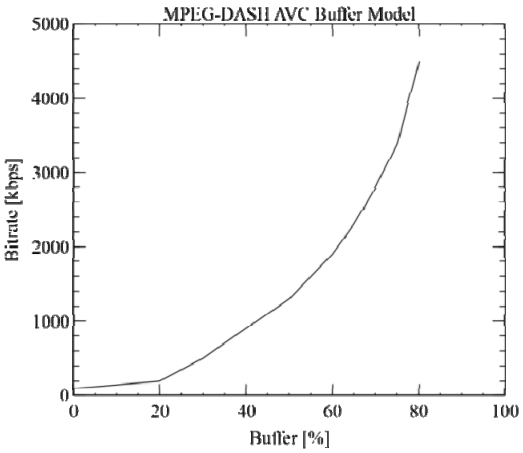

Figure 3. MPEG-DASH Buffer Model for AVC.

The buffer model has an exponential characteristic with the aim to reduce the number of quality switches and to enable a smooth playback. The model has been specified following an explorative approach where we have tested different models, e.g., linear, exponential, etc. In particular, the model must be fitted to the network conditions but this could be done on demand also, e.g., the client could start with a conservative approach which means that the turning point of the curve is near to 100% of the buffer. When the client is able to fill the buffer with that curve it could shift

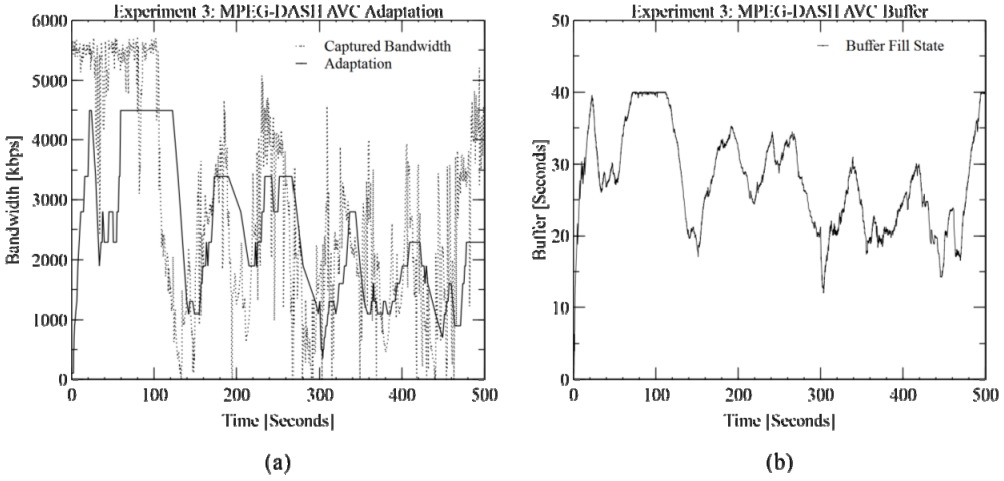

Figure 4. Results for MPEG-DASH AVC Experiment.

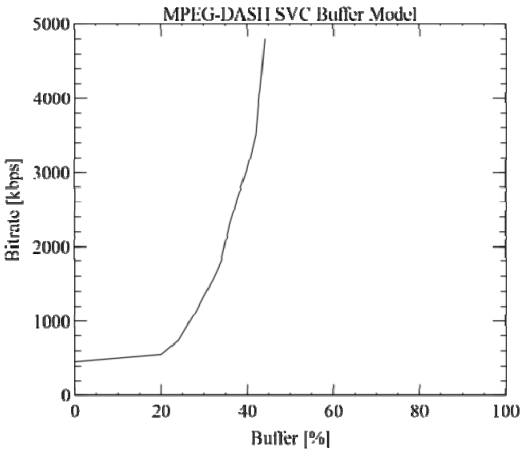

Figure 5. MPEG-DASH Buffer Model for SVC.

the curve to the left until the buffer reaches a predefined minimum.

Our adaption logic is mainly based on the buffer model depicted in Figure 3 and the average measured bandwidth. The first step is that the adaption logic identifies the quality levels that are allowed due to the buffer state, e.g., when the buffer is exactly at 40% all quality levels below 900 kbps are allowed by the buffer model. The second step would be to find a matching quality level based on the average bandwidth inside the range of the lowest quality level and a quality level that is below 900 kbps.

Figure 4 shows the behavior of our MPEG-DASH AVC implementation based on the above mentioned adaption process. Due to page count limits we only show the evaluation of experiment 3 / track 3 but the interested reader is referred to our supplemental material [12] which contains the evaluations of the other experiments / tracks as well as the evaluations of the industry solutions. Figure 4 (a) shows the adaption process (adaptation) and the available bandwidth (captured bandwidth) and Figure 4 (b) shows the buffer fill state in seconds. Interestingly, the buffer never falls below 10 seconds during the whole experiment which indicates that the adaption process is very stable and the probability of producing stalls is very low. Furthermore, the adaption reacts very accurately to bandwidth variations, e.g., second 190 or 290 where the available bandwidth collapses and as a consequence the adaptation process reduces the quality to keep the buffer in a stable state and guarantees a smooth playback. Additionally, it recovers also fast from low quality levels when the available bandwidth increases, e.g., second 150 and 300. In comparison to our implementation from [4] we have achieved a more accurate adaptation with less switches and utilized a higher average quality.

4.6 MPEG-DASH and SVC

In comparison to the MPEG-DASH AVC experiment the client for the SVC experiment is based on windows, due to the fact that we have used our dash library that is only available for windows. This library handles the whole DASH session management and it also supports HTTP/1.1 persistent connections and pipelining. This library has been integrated into our SVC client application.

A consequence of the layered architecture of SVC is that we are able to cancel requested segments at the layer boundaries which make the system more flexible. Therefore, we were able to use a more aggressive buffer model as depicted in Figure 5. We have also used the buffer model from the AVC experiment (cf. Section 4.5) but SVC achieved with the AVC buffer model nearly the same average bitrate as AVC. Hence, we have used the aggressive buffer model which has been specified following an explorative approach like the AVC buffer model. The adaptation process follows the same logic as described in Section 4.5 where the buffer model restricts the available

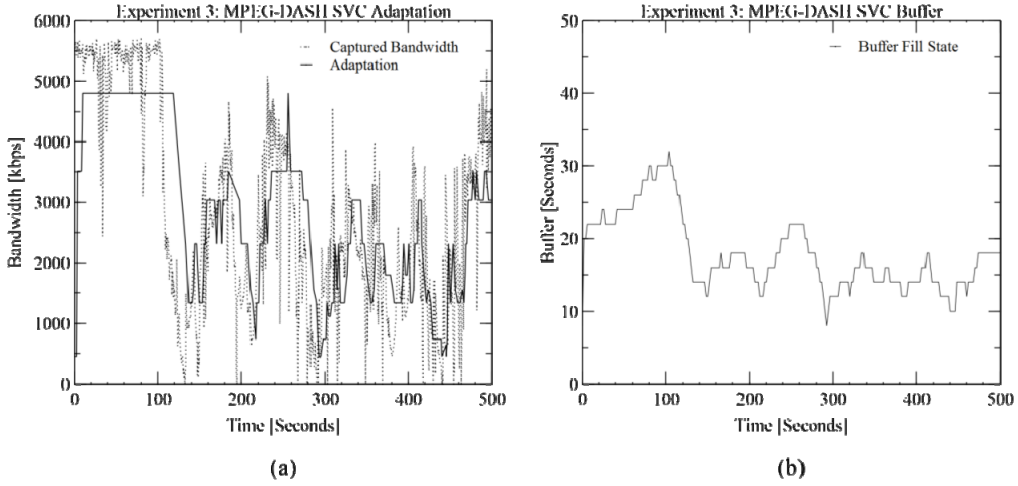

Figure 6. Results for MPEG-DASH SVC Experiment.

Table 1. Comparison with Existing Approaches.

Name	Average Bitrate [kbps]	Average Switches [Number of Switches]	Average Unsmoothness [Seconds]
Microsoft Smooth Streaming	1522	51	0
Adobe HTTP Dynamic Streaming	1239	97	64
Apple HTTP Live Streaming	1162	7	0
DASH AVC [4]	1464	166	0
DASH AVC	2341	81	0
DASH SVC	2738	101	0

quality levels. Figure 6 (a) shows the results for experiment 3 / track 3 and additional results are available at [12]. In comparison to the AVC experiment SVC achieves better bandwidth utilization with a lower yet stable buffer. Additionally, it also reacts very accurately to bandwidth variations, e.g., second 200 and 450 and it recovers fast from low quality levels when the available bandwidth increases, e.g., second 300 and 220.

4.7 Comparison with Existing Approaches

The overview of all three experiments / tracks are shown in Table 1. From left to right the first column depicts the name of the system, the average bitrate, the number of quality switches, and the number of unsmooth seconds. All values have been calculated as an average of all three experiments. Interestingly, our improved MPEG-DASH implementation using AVC and the smooth buffer model outperforms all industry solutions and our previous implementation with respect to average bitrate. Finally, the MPEG-DASH SVC-based solution achieves a higher average bitrate over all three experiments as a consequence of the more aggressive buffer model but requires more representation switches. Please note that using the same buffer model for AVC would produce an unsmooth session.

5 CONCLUSIONS AND FUTURE WORK

This paper provides an evaluation of our improved MPEG-DASH implementation using AVC and SVC. Furthermore, both solutions have been compared with the major industry solutions that we have already evaluated in [5]. All systems have been evaluated under the same conditions with our real-world mobile network traces that have been captured under vehicular mobility.

As seen from the experimental results, an exponential buffer model allows for a better utilization of the available bandwidth and SVC – thanks to its layered coding structure – allows for more flexibility and, consequently, a more aggressive buffer model. Minor modifications to the MPD combined with reordering of NALUs at the client enable the efficient usage of SVC for bitrates below 1 Mbps.

Finally, the actual impact on the QoE is subject to future work which may help us to improve our adaptation logic for the dynamic adaptive streaming over HTTP.

6 ACKNOWLEDGMENTS

This work was supported in part by the EC in the context of the ALICANTE (FP7-ICT-248652), SocialSensor (FP7-ICT-287975) projects and partly performed in the Lakeside Labs research cluster at AAU.

7 REFERENCES

[1] B. Wang et al., "Multimedia Streaming via TCP: An Analytic Performance Study," *ACM Transactions on Multimedia Computing, Communication and Applications*, vol. 4, no. 2, May 2008, pp. 16:1-16:22.

[2] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet", *IEEE Multimedia*, vol. 18, no. 4, Oct.-Dec. 2011, pp. 62-67.

[3] H. Schwarz et al., "Overview of the Scalable Video Coding Extensions of the H.264/AVC Standard," *IEEE Trans. on CSVT*, vol. 17, no. 9, Sep. 2007, pp. 1103-1120.

[4] C. Müller, S. Lederer, C. Timmerer, "An Evaluation of Dynamic Adaptive Streaming over HTTP in Vehicular Environments," *In Proce. of 4th Workshop on Mobile Video*, Chapel Hill, NC, USA, Feb. 2012.

[5] Y. Sanchez, T. Schierl, C. Hellge, T. Wiegand, D. Hong, D. D. Vleeschauwer, W. V. Leekwijck, Y. L. Leuedec, "iDASH: Improved Dynamic Adaptive Streaming over HTTP using Scaleable Video Coding," *In Proc. of ACM Multimedia Systems*, San Jose, CA, USA, Feb. 2011.

[6] I. Kofler, R. Kuschnig, H. Hellwagner, "Implications of the ISO Base Media File Format on Adaptive HTTP Streaming of H.264/SVC," *In Proc. of 9th IEEE Consumer Communications and Networking Conference*, Los Alamitos, CA, USA, Jan. 2012.

[7] S. Akhshabi, A. Begen, C. Dovrolis, "An Experimental Evaluation of Rate-Adaptation Algorithms in Adaptive Streaming over HTTP," *In Proc. of ACM Multimedia Systems*, San Jose, CA, USA, Feb. 2011.

[8] J. Yao, S. Kanhere, I. Hossain, M. Hassan, "Empirical evaluation of HTTP adaptive streaming under vehicular mobility," *In Proc. of 10th Int'l IFIP TC 6 Conference on Networking*, Valencia, Spain, May 2011, pp. 92-105.

[9] MPEG-DASH at ITEC/AAU, http://dash.itec.aau.at, (last access: Mar. 2012).

[10] P. Ni, R. Eg, A. Eichhorn, C. Griwodz, P. Halvorsen, "Spatial Flicker Effect in Video Scaling," *In Proc. of 3rd Int'l Workshop on Quality of Multimedia Experience*, Mechelen, Belgium, Sep. 2011, pp. 55-60.

[11] P. Romirer-Maierhofer, A. Coluccia, T. Witek, "On the Use of TCP Passive Measurements for Anomaly Detection: A Case Study from an Operational 3G Network," *Traffic Monitoring and Analysis Workshop TMA 2010*, Zürich, Switzerland, Apr. 2010, pp. 183-197.

[12] Additional Results, http://www-itec.uni-klu.ac.at/dash/eusipco/additional_results.pdf (last access: Mar. 2012).