Compositing without Transcoding for H.265/HEVC in Cloud IPTV and VoD services

Alexandra Mikityuk[1,2], Oliver Friedrich[1]
[1]Telekom Innovation Laboratories, Berlin, Germany
[2]Security in Telecommunications, TU Berlin, Germany
{alexandra.mikityuk, oliver.friedrich}@telekom.de

Robert Skupin[3], Yago Sanchéz[3], Thomas Schierl[3]
[3]Multimedia Communications Group
Fraunhofer Heinrich-Hertz-Institute, Berlin, Germany
{robert.skupin, yago.sanchez, thomas.schierl}@hhi.fraunhofer.de

Abstract— **This paper presents an overview of ongoing efforts for creation and delivery of next generation IP television and Video on Demand services. The successful cloud computing paradigm and recent advances in video coding standards are ready to be applied to widely established media services and promise enhanced user experience and reduced cost for operators at the same time. This paper brings together a novel architecture for media services from Deutsche Telekom Innovation Laboratories with cutting edge compressed domain video processing techniques from Fraunhofer Heinrich-Hertz-Institute. In this context, this paper presents an alternative to costly and resource hungry server-side video transcoding, e.g. for insertion of advertisement overlays or user interfaces. The presented concept uses the newly emerged H.265/HEVC standard.**

***Index Terms*—Stereoscopic, Panorama, Streaming, MV-HEVC**

I. Introduction

The creation and delivery of next generation IP television (IPTV) or Video on Demand (VoD) services is of interest for many parties. On one hand, service operators want to reduce operational cost by applying successful approaches such as the cloud computing paradigm to their established IPTV and VoD infrastructure, specifically Set-Top Boxes (STB). The Virtual Set-Top Box (vSTB) project [1][2] at Deutsche Telekom Innovation Laboratories gives a prime example for such an operator driven change to the existing infrastructure. From the operator perspective, the vSTB approach enables better manageability of customer devices, simplification of service and application delivery and respective cost reduction. On the other hand, the new state-of-the-art video coding standard H.265/High Efficiency Video Coding (HEVC) is ready for service deployment and promises enhanced video quality and bitrate reductions compared to the ubiquitous predecessor video coding standards H.264/AVC. Furthermore, the design of HEVC allows for various traditional video processing tasks to be carried out in the compressed domain. For example, lightweight and fast video compositing [3] developed at Fraunhofer Heinrich-Hertz-Institute may function as an enabler for the cloudification of IPTV and VoD services and replace resource hungry transcoding for various use cases. Video compositing is the process of creating a composition of multiple video sources that is presented to the user. Common examples in IPTV and VoD services are picture-in-picture (PiP) compositing and alpha blending of overlays with video content, e.g. for advertisements or user interfaces (UIs).

From an infrastructure point of view, the complexity and limited lifetime of STBs in current IPTV or VoD service deployments are one of the major cost factors for the current deployments. The vSTB approach addresses this issue by shifting STB functionality, i.e. UI generation, application execution and service management, to a Cloud infrastructure. The vSTB approach might therefore help to reduce costs by using a simplified centralized management of services, as well as helping to increase the lifetime of STBs in the field through more simplified end devices with a longer lifetime by locating STB functionalities in the operator network.

Many STB functionalities have the potential to be shifted into the cloud environment, e.g. from the general processing of data such as Electronic Program Guide (EPG), over content protection mechanisms, to management tasks on session, resource, and network level or even UI rendering. The work on "Media Cloud" presented in [8] deals with a media distribution beyond the boundaries of a local network that interoperates with DLNA and UPnP. The middleware framework for STBs presented in this approach manages media files with a web application environment. Hereby the extension of the local network through a digital media server in this approach virtually serves as a physical shift of STB functionalities. However, rendering and service composition tasks are still left to be carried out on CPE.

Service composition is as a crucial part of STB

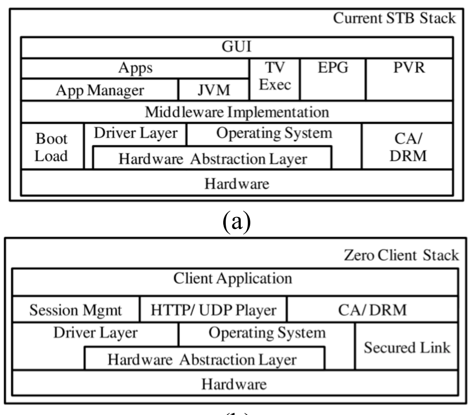

Figure 1: (a) current and (b) Zero Client STB stack.

functionality. Service composition in the cloud would encompass fusing of different media types, i.e. all the incoming video bitstreams with UIs and applications into one picture that is then to be transported and displayed to a user in a video frame. For carrying out this composition in a traditional transcoding-based fashion, the vSTB has to be able to decode and encode all relevant video bitstreams for a given user in real-time additionally to the video compositing task.

To reduce the workload from a full de- and encoding cycle, operation in the transform coefficient domain instead of the pixel domain was first proposed for Picture and Picture compositing in [6]. Since then, numerous techniques were proposed to fuse or cut short the individual compositing or transcoding steps and apply them to current video codecs with a good overview given in [7]. However, transcoding for the sake of carrying out some forms of compositing, particularly for state-of-the-art codecs such as HEVC, is still computationally complex and compromises system scalability. Depending on the transcoding approach, such processing may also impact rate distortion performance of the used video codec, i.e. disadvantages in video bitrate or quality. A novel technique for video compositing in the compressed domain with HEVC [4] enables the STB functionality of compositing to be carried out with much less demand on computational resources by operating in the compressed domain, i.e. no decoding or subsequent encoding has to be carried out at runtime. A demonstration of the compressed domain video compositing technique presented in this paper will be held at IBC 2015 exhibition.

The structure of the paper is a follows. An envisioned next generation service system architecture and client are presented in Sect. II. A novel technique for compressed domain video compositing with HEVC is described in Sect. III. Section IV deals with the integration of the technique into the envisioned architecture followed by a conclusion in Section V.

II. IPTV System Client and Architecture

STB architectures and their corresponding hard and software stacks, as depicted in Figure 1(a), have not fundamentally changed over the past 20 years. The key functionalities of video decoding, UI rendering, application execution, Conditional Access (CA) and Digital Rights Management (DRM) have been implemented on various hardware platforms, e.g. MIPS, ARM, x86. In parallel, one or multiple content protection systems, e.g. CA or DRM, were available to secure live and Video-on-Demand (VoD) services. On top of this, native, Java-based or even browser-based applications have been provided to drive different services, e.g. Electronic Program Guide (EPG), Personal Video Recorder (PVR), channel lists, etc. Standardized approaches have been later developed for the European market in the form of DVBs Multimedia Home Platform (MHP) [9] but were for various technical, political and IPR-related reasons never widely deployed. In parallel, different proprietary systems, e.g. Microsoft Mediaroom [10], have evolved and been introduced on several operator networks. These are the first generations of IPTV and VoD deployments implemented in the end-to-end service infrastructures, including the STB in customer premises.

While this first generation of deployments can provide End-to-End offering, ensure QoS in an operator controlled environment and provide fast enough channel change mechanism, they have the drawback of being limited to the specific vendor, not providing standard CPE and application environments and lack an ongoing roadmap concerning technical development.

A. Zero Client approach

Through the rapid development of the World Wide Web and corresponding web technologies in the recent years, service development paradigms have changed. In combination with the rise of mobile and TV platforms, these paradigms have also reached the embedded domain. In this context, web browsers have also become standard in embedded devices, and trends for the centralization of services in the cloud have yielded new approaches for reaching TV and mobiles as well.

For brevity, this paper focusses on the so-called Zero Client approach in which the CPE is kept as simple as possible hardware wise. For details on intermediate solutions, i.e. the Thin Client approach, the reader is referred to [1][2]. In the Zero Client approach, Web-Browser technology used for the service execution in the cloud which is closest to the idea of remote UI execution outsourcing as much as possible to the cloud thus ensuring maximum simplification on the CPE side. Here, the light weighting of the CPE hardware goes even further than the current state-of-the-art approach and locates not only the application layer in the virtualized environment, but also the middleware layer, leaving only video decoding to the client, in the ideal case. Therefore, in this approach, middleware is decoupled from hardware due to a virtualization (remote execution) that resolves HW dependencies and makes the system highly adaptive and flexible, providing a variety of application environments. This approach also solves the limitations of CPE capabilities and thus enables the operator to deliver new rich multimedia services to the legacy STBs and in general to any device owned by the end user, thereby enabling multi-screen support.

B. vSTB Architecture

With regard to the overall architecture of the Zero Client approach, the main advance over the current state-of-the-art STB architecture is that neither the UI rendering nor the application execution happen on the client itself. Instead, cloud resources are used to combine UI and video into a coded video bitstream on a server. Figure 1(b) gives an example of the reduced stack such an approach would allow for. The coded video bitstream is then delivered to the client, which in turn, only needs to perform video bitstream decoding. As evident from its description, the service transportation to the end user and the functionality of the end client is simpler than local execution. Therefore, the realization of multi-screen or companion screen applications becomes more affordable, assuming that the picture that the user will see is composed on a server and can be delivered to different devices owned by an end user.

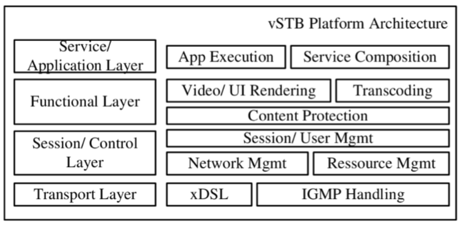

Figure 2: vSTB platform architecture overview.

Transmission of cloud rendered content, e.g. video with UI overlay, can be realized using two distinct approaches. It can be realized in an OS-dependent manner using proprietary bitmap streaming protocols such as RemoteFX, PC-over-IP (PCoIP) or High Definition User Experience (HDX) and therefore requiring the use of the local OS on the client. However, our experiments have shown that the proprietary bitmap streaming protocols, optimized for remote desktop applications, do not support all the required usage scenarios in which rich UIs must be supported. Therefore, an OS-independent solution, using only the hardware video decoder at the end device seems to be preferable over any proprietary solutions.

Considering the vSTB functionalities as mentioned earlier, the vSTB Platform architecture can be described through the four main layers depicted in Figure 2. The Service/ Application Layer is mainly responsible for Application Execution and Service Composition (or UI Generation), which comprises its discovery, execution and composition. The Functional Layer executes near-hardware functions, such as different Video/UI operations, Transcoding and Content Protection. The Session/Control Layer executes the management of Sessions/Users (Sessions regarding an interface to the Client: vSTB Client STB), to which the Client STBs are assigned, including resource management for hardware assignment tasks as well. The Session Manager is also in charge of interfacing with the Internet and manages vSTB sessions with Service/ Content Providers. The Network Manager coordinates the Transport Layer, controlling physical links and handling MC Requests and session handovers.

III. Video Compositing in the Compressed Domain

The novel method for compressed domain compositing of coded video bitstreams [3] takes a different approach than traditional transcoding and may therefore be a key component for the scalability of next-generation cloud IPTV or VoD services. Compositions are produced by merging coded input video bitstreams into a single common output bitstream and inserting pre-encoded, inter-predicted pictures that use the merged input bitstreams as reference. Video bitstreams in this context could contain any content from actual video to rendered applications or UI.

The proposed method relies on a number of features of HEVC [4], which includes the spatial segmentation of video pictures into a grid of so-called tiles. In contrast to the comparable H.264/AVC feature of flexible macroblock ordering, tiles are included in all currently existing HEVC profiles. These tiles are coded independently within a picture. They divide it along the borders of coding tree units that serve as basis for block-based coding. Tiles are especially interesting for another form of compressed domain video processing, namely video stitching as presented in [5] to allow application in standard compliant HEVC decoders and are applied in this work as well. HEVC further improves on the implicit reference picture management of H.264/AVC by introduction of the robust reference picture sets (RPSs) concept. Reference pictures in HEVC are explicitly signaled per picture slice, which enables not only a significantly easier loss detection but also an easier reference manipulation with respect to H.264/AVC, a key component of the proposed method. Another interesting new concept in HEVC that is used in the proposed method is a concept of non-output pictures. These pictures are decoded and can be used for reference by other pictures of the bitstream. However, these pictures are not output for presentation to the user.

The general outline of the proposed compressed domain video compositing method is as follows: first, pictures of the input videos are merged into a unified output video bitstream through multiplexing in the compressed domain as presented in subsection III.A. Second, pictures generating a composition of the input videos are inserted into the output bitstream as detailed in subsection III.B.

A. Merging sources

The process of merging *n* input bitstreams generates a single output bitstream that contains all pictures of the input bitstreams. Two approaches are available for bitstream merging: temporal multiplex (TM) or spatial multiplex (SM). For the sake of brevity, this paper focusses only on SM. For further details regarding TM, the reader is referred to [3].

The input bitstream pictures are referred to as source pictures (SPs) in this work and are not intended for display in the original form but for creation of a composition to be output by a standard compliant HEVC decoder. Alignment of coding parameters between input bitstreams is advantageous as the parameter sets have to be merged and values of different picture parameter sets within a coded video sequence are partly constrained, e.g. with respect to picture dimensions. The following assumes that input bitstreams are encoded in a synchronous fashion, i.e. with similar group of pictures (GOP) size and referencing structure. For VoD services that are in full control of the encoding of their content database, such an assumption is justified. In live IPTV services, input video from various sources might require extra precaution when applying the presented method.

The SM multiplexing approach is explained given the exemplary low delay IPP coding structure shown in Figure 3(a)

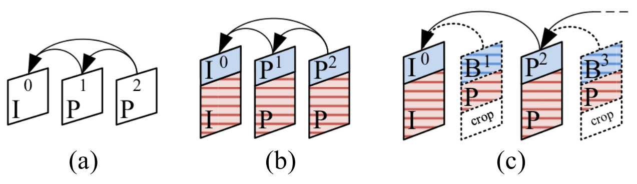

Figure 3: (a) exemplary IPP video, (b) merged input bitstream pictures and (c) position of dashed CPs.

indicating prediction dependencies as solid arrows and denoting the picture order count (POC) in the top of each picture. However, it should be noted that the technique is applicable to other coding structures with Random Access capability and hierarchical B-frames that are commonly used in IPTV and VoD services.

The merging approach SM is illustrated in Figure 3(b). The underlying technique is also referred to as compressed domain stitching. For SM, a single picture dimension of the input video bitstreams is required to be sized equally along which pictures can be stitched. POC values and RPS can remain unchanged in the merged bitstreams. Slice segment data of the input bitstreams is copied to slices or tiles of the output bitstream, depending on the stitching layout. The required adjustments to high-level syntax are lightweight. For example, slice addresses in the merged slice headers need to be adjusted to the new tile or slice positions within the merged picture, and slice delta quantization parameters (QPs) might need adjustment to reflect a possible change of the initial QP as signaled in the merged parameter set.

Furthermore, to allow for such compressed domain stitching without prediction mismatches between the multiple individual encoders and the single decoder of the merged output bitstream, the encoders have to be constrained. Below is a short summary of the constraints for HEVC coded bitstreams as detailed in [5]:

1) Motion Vector (MV) constraints: MVs should not point to samples outside the picture borders or sub-pel sample positions, for which the encoder-side invoked sub-pel interpolation filter kernel overlaps with the picture borders.

2) Prediction units: the rightmost prediction units within a picture shall not use the MV prediction candidate that corresponds to a temporal motion vector prediction (TMVP) candidate or the spatial MV candidate at the position of a non-existent TMVP candidate.

3) In-loop filters: slice segment and tile borders (if present) shall not be crossed by in-loop filters such as the deblocking and SAO filter.

It is important to note that the alternative multiplexing approach TM allows for multiplexing of video sequences that do not fulfil the above encoding constraints. For details, the reader is referred to [3].

The output flag in slice segment headers of input bitstream pictures is disabled and therefore input bitstream pictures are not output. The input bitstream referencing structures present a crucial point of the merging process. When the input bitstreams are not encoded in a synchronous fashion, merging may require

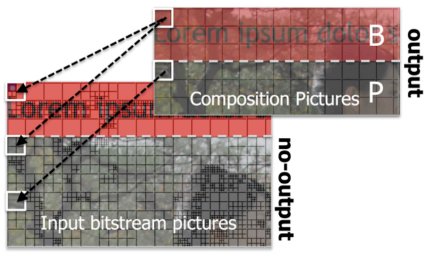

Figure 4: CP inter-prediction and structure.

heavy adjustments to RPS structures up to the point of infeasibility due to level constraints regarding the DPB size or an incompatible order of coded pictures.

B. Insert Composition Pictures

The second step of the proposed method is the addition of pre-encoded composition pictures (CPs) to the output bitstream. These pictures may be encoded beforehand and added to a merged output bitstream at runtime. Picture area samples of a set of input bitstream pictures are copied to the associated CP picture area via block-based inter-prediction to form the desired composition.

Figure 4 illustrates the construction of a CP (top right) via inter-prediction from input bitstream pictures (bottom left) that are merged via SM. The exemplary CP consists of a bi-predictive slice segment at the picture top and a uni-predictive slice segment at the bottom. Inter-prediction MVs are depicted as dashed arrows for the first prediction block of each CP slice segment. The depicted setup leads to a transparent overlay of the first input video on top of the second input video. As can be seen from the resulting composition given in Figure 4, CP MVs are invariant throughout the CP slice segment to seamlessly copy the intended input bitstream pictures samples. For example, samples that are collocated to the given CP sample positions (e.g. as for the top dashed arrow) or samples with an optional invariant spatial offset to the collocated CP sample positions (e.g. as for the two bottom dashed arrows).

Since CPs slice segment data only depends on high-level parameters such as picture size and desired composition, a data set matching the targeted bitstream parameters can be pre-encoded beforehand for subsequent insertion. The remaining workload of the proposed method consists of simple parameter set and slice segment header adjustments for correct reference picture management. However, without many of the time-consuming steps of regular video encoding such as motion estimation or loop filtering, CP generation is of comparatively low complexity and thus could even be carried out on-demand at runtime.

As shown in Figure 3 (c), the slice segments of CPs are inserted into the output bitstream after the associated set of input bitstream pictures in coding order. Therefore, POC values and RPS of input bitstream pictures have to be slightly adjusted to accommodate the additional CPs in the output bitstream. Additional RPSs are required to signal references of the CPs, namely the one active reference picture used for compositing plus all reference pictures signaled in the RPS of the associated set of input bitstream pictures in coding order. It is likely that picture dimensions of at least one of the input videos are also the desired output picture dimensions, e.g. when adding a UI overlay. As illustrated in Figure 3 (c), unused picture areas of the CP might be required to be hidden from the user. The HEVC picture cropping procedure allows output of the desired picture area of CPs only in this case.

It was reported in [3] that the presented composition method allows for bitrate savings compared to transcoding-based compositing in the uncompressed domain of 20% on average using an exemplary overlay. Furthermore, the

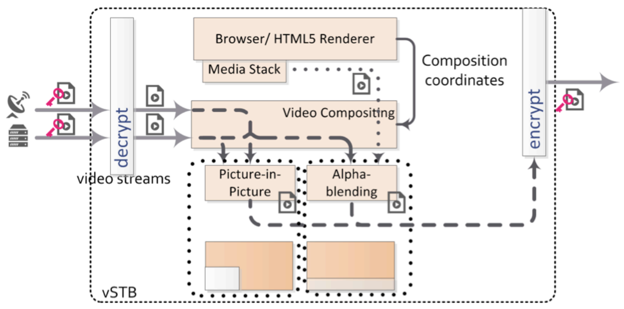

Figure 5: Compositing without transcoding for vSTB architecture

compressed domain compositing leads to a higher video quality at given input video quality.

C. Integration into vSTB Architecture

From system perspective, carrying out video compositing such as UI insertion on service operator side, e.g. on cloud resources, instead of CPE reduces the requirements for the customer equipment to mere video decoding. In such a system design as envisioned for the vSTB, transcoding based compositing scales poorly and can potentially impair RD performance. The proposed compositing method on the other hand presents a lightweight alternative to transcoding based compositing.

In context of the vSTB platform architecture presented in Figure 5, the presented technique for compressed domain compositing resides in the Service/Application and Functional layer as described in Sect. II. Here, it replaces compositing related decoding and encoding, which does not interfere with CA and DRM protection mechanisms.

The integration of the described above video composition into the vSTB architecture is in detail presented in Figure 5. The encrypted video streams from different video sources are delivered to the vSTB. The CA and DRM libraries running in the cloud enable the decryption function. After the video streams have been decrypted, each of them is then sent to the Video Compositing (VC) component. The VC component is responsible for two major TV use cases:

a) Picture-in-Picture (PiP): merging and composition of the video streams,
b) Alpha blending: overlapping the video stream with an UI overlay with a certain value of transparency.

With regard to the PiP use case, both of the steams are delivered to the VC component. The video stream that is to be presented to the end user as a channel or video preview is available at a lower resolution. The video stream in a smaller resolution is merged with the other video stream. The HTML5 Renderer or the Browser sends the coordinates of the merging position or the so-called composition coordinates to the VC component.

With regard to the alpha blending use case, the required video stream here is overlapped with a UI element. The Browser interprets the HTML5 UI code and renders this code into a video stream via the running in the cloud Media Stack. Since UI content can often be reused between users, e.g. channel or program information, the UI video stream is not necessarily encoded live for each user.

This UI video stream is afterwards sent to the VC and overlaid on top of the video stream. Analog to the PiP use case, the VC also receives the alpha blending transparency and the composition coordinates from the Browser, e.g. bottom left, top right, etc. After processing in VC, the final composite stream is sent back to the CA and DRM libraries that in turn encrypt the video to send it to the client over an operator's network. The client decodes the video and presents it to the user.

IV. Conclusion

This paper has presented a detailed analysis of pain points of current IPTV deployments. The cloud-based rendering concept, virtual Set-Top Box, has been presented in this paper as a solution to overcome the fragmentation and limitation of the current STBs. To reduce costs and computer resource consumption, the video compositing without transcoding for H.265/HEVC has been introduced into the vSTB architecture. The video compositing enables efficient implementation of the Picture-in-Picture and alpha blended UI use cases.

References

[1] Mikityuk, Alexandra, Jean-Pierre Seifert, and Oliver Friedrich. "The virtual Set-Top Box: On the shift of IPTV service execution, service & UI composition into the cloud." Intelligence in Next Generation Networks (ICIN), 2013 17th International Conference on. IEEE, 2013.

[2] Mikityuk, Alexandra, Jean-Pierre Seifert, and Oliver Friedrich. "Paradigm shift in IPTV service generation: Comparison between locally-and Cloud-rendered IPTV UI." Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th. IEEE, 2014.

[3] Skupin, Robert, Yago Sanchez, and Thomas Schierl: "Compressed Domain Video Compositing with HEVC." Picture Coding Symposium (PCS), 2015.

[4] Sullivan, G. J., Ohm, J., Han, W. J., & Wiegand, T. "Overview of the high efficiency video coding (HEVC) standard." Circuits and Systems for Video Technology, IEEE Transactions on 22.12 (2012): 1649-1668.

[5] Sanchez, Yago, et al. "Low complexity cloud-video-mixing using HEVC." Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th. IEEE, 2014.

[6] Chang, Shih-Fu, and David G. Messerschmitt. "Manipulation and compositing of MC-DCT compressed video." Selected Areas in Communications, IEEE Journal on 13.1 (1995): 1-11.

[7] Vetro, Anthony, Charilaos Christopoulos, and Huifang Sun. "Video transcoding architectures and techniques: an overview." Signal Processing Magazine, IEEE 20.2 (2003): 18-29.

[8] Diaz-Sanchez, Daniel, et al. "Media Cloud: Sharing contents in the large." 2011 IEEE International Conference on Consumer Electronics (ICCE). 2011.

[9] ETSI ES 201 812 V1.1.2 (2006-08). Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.0.3. Sophia Antipolis, 2006.

[10] Microsoft Mediaroom; http://www.microsoft.com/mediaroom/