Description-Based Substitution Methods for Emulating Temporal Scalability in State-of-the-Art Video Coding Formats

W. De Neve[1], D. De Schrijver[1], D. Van de Walle[1], P. Lambert[1], and R. Van de Walle[2]

[1] Ghent University - IBBT, ELIS, Multimedia Lab
Gaston Crommenlaan 8 bus 201, B-9050 Ledeberg-Ghent, Belgium
[2] Ghent University - IBBT - IMEC, ELIS, Multimedia Lab
Gaston Crommenlaan 8 bus 201, B-9050 Ledeberg-Ghent, Belgium

Abstract MPEG-21 BSDL provides a solution for discovering the structure of a binary media resource to generate its XML description, and for the generation of an adapted media resource using a transformed description. So far, most literature related to XML-driven video content adaptation deals with the exploitation of temporal scalability by dropping certain pictures. This paper takes it a step further by focusing on how description-driven exploitation of temporal scalability can be realized by replacing coded pictures by placeholder pictures. Such an approach allows for a transparent integration with the systems layer in modern multimedia architectures. Our XML-based replacement technique will be discussed from a high-level point of view for MPEG-{1, 2} Video, MPEG-4 Visual, and SMPTE's Video Codec 1 (VC-1). A more detailed analysis, including some performance measurements, will be provided for H.264/MPEG-4 AVC as its technical design is the most challenging one.

1 Introduction

The traditional view of digital video coding is to find an optimal trade-off between bit rate and distortion, given a maximum allowed delay and complexity. However, due to a growing diversity in the present-day multimedia landscape in terms of terminals, networks, and media formats used, there is a strong tendency to draw another important aspect into this relationship, i.e. adaptivity. The subject of this paper is a discussion of description-driven extraction of video streams of multiple picture rates from a single coded bitstream (i.e., temporal scalability). Special attention is paid to the deployment of placeholder pictures to avoid conflicts with the systems layer (e.g., file formats, streaming protocols). As such, the substitution of certain coded pictures by dummy pictures makes it, for instance, straightforward to maintain synchronization with other media streams in a file container, even when a varying GOP structure is in use.

This paper is organized as follows. Section 2 discusses the MPEG-21 Bitstream Syntax Description Language (MPEG-21 BSDL). The concept of placeholder pictures is introduced in Section 3. The creation of placeholder pictures for several widely-used coding formats is elaborated on in Section 4, hereby putting the emphasis on H.264/AVC. Finally, Section 5 discusses some performance results while Section 6 concludes this paper.

2 MPEG-21 BSDL

The MPEG-21 community is currently working towards an efficient, secure, and interoperable description-driven framework for media format-agnostic content adaptation. One of the building blocks of this architecture is MPEG-21 BSDL [1]. Sitting at the intersection between the metadata world and the world of media representation, this schema language is used to describe the (high-level) structure of a certain media format (file format, coding format). This gives birth to a document that is called a Bitstream Syntax Schema (BS Schema). It contains the necessary information for translating the structure of a binary media resource into an XML-based Bitstream Syntax Description (BSD), and for the creation of a tailored media resource by relying on a transformed description.

The generation of the XML-based description is done by a software module called BintoBSD Parser, while the adapted bitstream is constructed by a module that is called BSDtoBin Parser. How to transform a BSD, is not specified by MPEG-21 BSDL. The functioning of both parsers is guided by a particular BS Schema for a certain media format. As such, a BintoBSD and BSDtoBin Parser constitute the pillars of a generic software architecture for media format-unaware content adaptation, and of which the operation is entirely steered by XML-based technologies. This concept is similar to how XHTML and Cascading Style Sheet (CSS) documents steer a browser in order to render a web page.

```
<xsd:schema>
   <xsd:element name="bitstream" bs2:startContext="'bitstream'">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="parse_unit"
                  bs2:startContext="'pu'" bs2:stopContext="'old_pu'"
                  bs2:redefineMarker="'tmp_pu' 'old_pu' 'pu' 'tmp_pu'">
               <xsd:complexType>
                  <xsd:choice>
                     <xsd:element ref="coded_header_data" />
                     <xsd:element ref="coded_video_data" />
                  </xsd:choice>
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

Fig. 1 Generic BS Schema for a packet-based format

In Figure 1, a generic template is provided for a BS Schema describing the structure of a typical packet-based coding format. The high-level structure of all video coding formats as discussed in this paper, is in line with this template [e.g., a Parse Unit (PU) in H.264/AVC equals to a Network Abstraction Layer Unit (NALU)]. The template is also annotated by attributes that enable an intelligent context management by a BintoBSD Parser. These extensions to BSDL are currently under consideration for standardization by MPEG. They allow an efficient BSD generation in terms of processing time and memory consumption needed (see the performance results in Section 5 for the BintoBSD Parser) [2].

3 Placeholder Pictures

A placeholder or dummy picture can be defined as a picture that is identical to a certain reference picture, or that is reconstructed by relying on a well-defined interpolation process between different reference pictures. This implies that only a limited amount of information has to be stored or transmitted in order to identify placeholder pictures. Dummy pictures are typically used for achieving bit rate savings when the content of successive pictures is very similar. Because these pictures are able to introduce an additional delay, they can also be used for synchronization purposes. Hence, they are sometimes called delay or skipped pictures as well. In this paper, placeholder pictures are used to fill up the gaps that are created in a bitstream due to the removal of certain pictures (temporal scalability). This approach makes it straightforward to maintain synchronization with other media streams in a particular container format, especially when a varying GOP structure is in use.

The operational flow of our XML-driven content adaptation approach, independent of the coding format used, can be described as follows. Given a bitstream and a BS Schema for the format of the media resource, a description of the high-level structure of the bitstream is created by a generic BintoBSD Parser. The descriptions of the different PUs in the BSD are then processed by relying on a hybrid solution that involves the use of a Streaming Transformations for XML (STX) and an Extensible Stylesheet Language Transformations (XSLT) stylesheet. Note that an XSLT-only approach fails in case of large BSDs due to its high memory requirements. A full STX approach is the subject of further research.

The STX stylesheet is used for iterating through the descriptions of the different PUs and for passing them to the XSLT stylesheet. As such, the use of STX makes it possible to acquire a low memory footprint during the processing of a BSD (see Section 5). The XSLT stylesheet, embedded in the STX stylesheet, is subsequently used for doing look-ahead operations in the description of a particular PU. This allows doing a more detailed analysis of the description of a PU in an easy way. For instance, in case of the detection of a description of a bidirectionally coded picture, the XSLT stylesheet might decide to replace or to adapt this PU description to obtain a description of a PU representing a placeholder picture. Finally, the customized BSD is provided to a BSDtoBin Parser which in its turn generates an adapted bitstream, suited for a certain usage environment. Note that the BS Schema has to include a syntax description of a dummy picture to generate a tailored bitstream (due to the use of a few additional, low-level syntax elements; see also Figure 2). This fragment in the BS Schema is only used by a BSDtoBin Parser.

4 Creation of Placeholder Pictures

In the next subsections, the creation of placeholder pictures will be discussed from a syntax point of view for a few well-known coding formats. Special attention will be paid to the creation of dummy pictures in H.264/AVC.

4.1 MPEG-1 Video and MPEG-2 Video

Signaling a placeholder picture in MPEG-2 Video can be realized by using so-called pseudo-skipped pictures. This concept, introduced by Lee *et al.* in [3], boils down to creating an artificial picture that consists of multiple slices. Hereby, every slice contains two macroblocks that mark the beginning and the end of the slice. In-between macroblocks are automatically considered as skipped. The macroblocks at the start and end of the slice are to be coded in backward or forward motion compensated mode, and with zero-valued motion vectors and quantized coefficients. Because MPEG-2 Video can be considered as a superset of MPEG-1 Video, the technique for creating dummy pictures in MPEG-1 Video is very similar. When creating such pictures in MPEG-1 Video, there are only some minor syntactical differences to be taken into account at the macroblock layer. The more flexible definition of a slice in MPEG-1 Video also allows for a few optimizations since an entire picture can be coded as a single slice. A description of a placeholder picture in MPEG-1 Video is depicted in Figure 2. Note the resolution dependency in the macroblock addresses.

```
<pseudo_skipped_picture_352x288>
   <picture_header>
   <!-- ... -->
   </picture_header>
   <one_slice_352x288>
      <slice_start_code>00000101</slice_start_code>
      <quantiser_scale_code>1</quantiser_scale_code>
      <extra_bit_slice>0</extra_bit_slice>
      <first_macroblock>
         <macroblock_address_increment>1</macroblock_address_increment>
         <macroblock_type>2</macroblock_type>
         <motion_hor_backward_code>1</motion_hor_backward_code>
         <motion_vert_backward_code>1</motion_vert_backward_code>
      </first_macroblock>
      <last_macroblock>
         <macroblock_escape>8</macroblock_escape>
         <!-- ... -->
         <macroblock_escape>8</macroblock_escape>
         <macroblock_address_increment>25</macroblock_address_increment>
         <macroblock_type>2</macroblock_type>
         <motion_hor_backward_code>1</motion_hor_backward_code>
         <motion_vert_backward_code>1</motion_vert_backward_code>
      </last_macroblock>
      <bit_stuffing>0</bit_stuffing>
   </one_slice_352x288>
</pseudo_skipped_picture_352x288>
```

Fig. 2 A pseudo-skipped picture in MPEG-1 Video

4.2 MPEG-4 Visual and Video Codec 1

In MPEG-4 Visual, the description of any coded Video Object Plane (VOP) can be transformed into a description of a non-coded VOP (N-VOP), i.e. a dummy picture, by enabling the `vop_coded flag` in the VOP header and by stripping all further information for that picture. Because SMPTE's VC-1 specification provides an own picture type for signaling a placeholder picture (i.e., a Skipped picture), it is rather straightforward to replace the description of a picture by the description of a Skipped picture. Both specifications allow creating a resolution-independent description of a dummy picture.

4.3 H.264/MPEG-4 AVC

Before diving into the construction of placeholder pictures in H.264/AVC, it is interesting to have a closer look at some of its design features. First, in H.264/AVC there is no such thing as an explicit I, P, or B picture since only I, P, and B slices are defined. In addition, a coded picture can comprise a mixture of different types of slices. Finally, B slices can be used as a reference for the reconstruction of other slices. Therefore, the recommended way for picture dropping in H.264/AVC is to rely on sub-sequences and Supplemental Enhancement Information messages (SEI messages) [4].

The flexible design of H.264/AVC does not only have an impact on how to implement and exploit temporal scalability; it also has an influence regarding the creation of placeholder pictures. While the generic nature of our XML-driven substitution approach is constrained by a resolution dependency in MPEG-{1, 2} Video, there are a few other aspects to deal with when using our proposed method for H.264/AVC. Besides the picture resolution and the number of macroblocks used in a slice, it is also important to take into account parameters such as the variable length coding scheme, the slice type, and whether or not a slice is used as a reference for the reconstruction of other slices. Therefor, the idea is to keep the creation of placeholder slices as simple as possible, hereby minimizing the impact on the different decoding processes (reference picture and display order management, deblocking filter). As such, this enables the use of skipped slices into an H.264/AVC bitstream with a small number of changes at the level of the different parameter sets and the `slice_header()` syntax structures.

```
-------------------- original non-reference B slice ----------------------
<coded_slice_of_a_non_IDR_picture>
   <slice_layer_without_partitioning_rbsp>
      <slice_header>
         <first_mb_in_slice>0</first_mb_in_slice>
         <slice_type>1</slice_type>
         <pic_parameter_set_id>0</pic_parameter_set_id>
         <frame_num xsi:type="b5">3</frame_num>
         <!-- ... -->
      </slice_header>
      <slice_data>
         <bit_stuffing>3</bit_stuffing>
         <slice_payload>20605 4</slice_payload>
      </slice_data>
   </slice_layer_without_partitioning_rbsp>
</coded_slice_of_a_non_IDR_picture>
---------- resulting placeholder: skipped non-reference P slice ----------
<coded_slice_of_a_skipped_non_IDR_picture>
   <skipped_slice_layer_without_partitioning_rbsp>
      <slice_header>
         <first_mb_in_slice>0</first_mb_in_slice>
         <slice_type>5</slice_type>
         <pic_parameter_set_id>0</pic_parameter_set_id>
         <frame_num xsi:type="b5">3</frame_num>
         <!-- ... -->
      </slice_header>
      <skipped_slice_data>
         <mb_skip_run>544</mb_skip_run>
         <rbsp_trailing_bits>
            <rbsp_stop_one_bit>1</rbsp_stop_one_bit>
            <rbsp_alignment_zero_bit>0</rbsp_alignment_zero_bit>
         </rbsp_trailing_bits>
      </skipped_slice_data>
   </skipped_slice_layer_without_partitioning_rbsp>
</coded_slice_of_a_skipped_non_IDR_picture>
```

Fig. 3 Slice manipulation in H.264/AVC

The use of placeholder pictures in H.264/AVC will be exemplified with the translation of a non-reference B slice to a non-reference dummy P slice. This conversion process is partly clarified in Figure 3. Firstly, the `nal_ref_idc` syntax element in a NALU header is not changed, implying that the property whether a slice is used as a reference or not, is maintained. This is a first step in keeping the management of reference pictures consistent in case such pictures are involved in the conversion process[1]. Secondly, the `slice_type` is changed in order to signal a P slice, hereby avoiding an interpolation between the reference pictures in case a skipped B slice would have been used. The `frame_num` syntax element can be merely copied to the target slice header (no reference pictures are removed), as well as all information related to the display order management (slices are only replaced by dummy ones), information pertaining to the deblocking filter, and information concerning memory management control operations. The parameters related to B slices, used in the management of the reference lists and the reconstruction of a slice (weighted prediction), have to be dropped. To save some extra bits, the value of `slice_qs_delta` is initialized to zero as it is

[1] Pictures of which the reconstruction is dependent on skipped reference pictures also need to be signaled as skipped.

Table 1 Performance measurements (non-reference B slices replaced by non-reference skipped P slices)

bitstream characteristics					BintoBSD$_m$					STX/XSLT					BSDtoBin$_m$		
resolution	#slices/ picture	#PUs	file$_o$ (MB)	file$_m$ (MB)	ET (s)	ET (PU/s)	MC (MB)	BSD$_o$ (MB)	cBSD$_o$ (KB)	ET (s)	ET (PU/s)	MC (MB)	BSD$_m$ (MB)	cBSD$_m$ (KB)	ET (s)	ET (PU/s)	MC (MB)
848x352	5	17819	37.4	26.6	131	136	1.7	35.0	352	1865	10	2.7	36.2	243	32	557	1.9
1280x544	7	24945	110.0	73.9	204	122	1.7	45.9	498	2746	9	2.7	47.6	341	44	567	1.9
1904x800	9	32071	163.0	109.0	278	115	1.7	59.8	607	3340	10	2.7	62.0	419	59	544	1.9

not used during the reconstruction of a skipped slice. Finally, the `slice_data()` structure is replaced such that all macroblocks are flagged as skipped. Note that our approach only works when Context-Adaptive Variable Length Coding (CAVLC) is in use. Arithmetic coding leads to a context-adaptive representation of the syntax element that communicates a skipped macroblock run, something not easy to express in BSDL.

5 Experimental Results

The use of placeholder pictures has been successfully verified for all coding formats discussed, among which is the use of pseudo-skipped pictures for DVD content. Due to place constraints, we will only provide some measurements for H.264/AVC. The results as shown in Table 1 were obtained on a PC with an Intel Pentium IV 2.61 GHz CPU and 512 MB of memory (ET stands for Execution Time; MC for peak Memory Consumption; PU for Parse Unit, i.e. a NALU; and cBSD for compressed BSD). The BSD and bitstream generation was done by in-house optimized versions of the parsers as available in the MPEG-21 reference software. The Joost STX processor (version 20050521), having a built-in XSLT engine, was used for transforming BSDs; WinRAR 3.0's default text compression algorithm was used for compressing them [the BSDs could not be compressed with current implementations of MPEG-7 BiM (Binary Format for Metadata)]. The resources involved are 3 different versions of the same movie trailer (The New World), as used in a real-world simulstore scenario. Each version has a varying GOP pattern. However, a P slice coded picture is mostly alternated with a non-reference B slice coded picture. At the start of the bitstream, every resource has one Sequence and one Picture Parameter Set, as well as one buffering period SEI message. The picture rate is 23.98 Hz, resulting in a length of 148 s.

A first observation is that the parsers are able to generate BSDs and customized bitstreams with a feasible computational and memory complexity. Transforming a BSD can be done efficiently in terms of memory requirements; an XSLT-only approach would have required lots of memory [5]. However, the processing speed is currently unacceptable, caused by the large number of slices per picture, the use of XSLT, and the complexity of the transformation. The hybrid STX/XSLT approach also results in an increase of the BSD size, mainly due to unwanted namespace declarations for each PU. Finally, it is worth paying attention to the decrease in bitstream sizes. The column labeled *file$_o$* contains the sizes of the original H.264/AVC bitstreams; *file$_m$* denotes the sizes of the modified bitstreams containing placeholders.

6 Conclusion

In this paper, a discussion was provided regarding the use of placeholder pictures in an XML-driven content adaptation chain, hereby aiming at transparently exploiting temporal scalability. The construction of skipped pictures was outlined for several state-of-the-art video coding formats. Their use was stressed for H.264/AVC as its design is the most challenging one for our substitution technique. Some performance measurements were provided for real-life H.264/AVC bitstreams. The results illustrate that the generation of a BSD and the construction of an adapted H.264/AVC bitstream with placeholder slices can be done with a feasible complexity in terms of system memory and processing time needed.

Acknowledgement: The research as described in this paper was funded by Ghent University, the Interdisciplinary Institute for Broadband Technology (IBBT), the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research-Flanders (FWO-Flanders), the Belgian Federal Science Policy Office (BFSPO), and the European Union.

References

1. S. Devillers, C. Timmerer, J. Heuer, H. Hellwagner, "Bitstream Syntax Description-Based Adaptation in Streaming and Constrained Environments," *IEEE Transactions on Multimedia*, vol. 7, pp. 463–470, 2005.
2. D. De Schrijver, W. De Neve, K. De Wolf, R. Van de Walle, "Generating MPEG-21 BSDL Descriptions Using Context-Related Attributes," *Proc. of the 7th IEEE ISM conference*, pp. 79–86, USA, 2005.
3. Y. Lee, J. Lee, H. Chang, J. Y. Nam, "A New Scene Change Control Scheme based on Pseudo-skipped Picture," *Proc. of SPIE/Visual Communications and Image Processing*, vol. 3024, pp. 159–166, USA, 1997.
4. W. De Neve, D. Van Deursen, D. De Schrijver, K. De Wolf, R. Van de Walle, "Using Bitstream Structure Descriptions for the Exploitation of Multi-layered Temporal Scalability in H.264/MPEG-4 AVC's Base Specification," *Proc. of PCM 2005*, Springer-Verlag, pp. 641–652, Chejudo, 2005.
5. W. De Neve, D. De Schrijver, D. Van Deursen, R. Van de Walle, "XML-Driven Bitstream Extraction Along the Temporal Axis of SMPTE's Video Codec 1," *Proc. of WIAMIS 2006*, Accepted for publication, Seoul, 2006.