H.264 BASELINE PROFILE TRANSCODING OF MPEG2 MAIN PROFILE SEQUENCE

Subarna Tripathi*  Emiliano Mario Piccinelli#

*Advanced System Technology, STMicroelectronics Pvt. Ltd., GNoida, U.P. - 201308, India
#Advanced System Technology, STMicroelectronics, 20041 Agrate Brianza (MI), Italy
{subarna.tripathi, emiliano.piccinelli}@st.com

ABSTRACT

This paper describes a new transcoding algorithm, able to transcode any coded (e.g. MPEG-2) bit-stream into an H.264 Baseline Profile sequence. H.264 Baseline Profile is developed primarily for lower-cost applications demanding less computing resources. This profile is used widely in video conferencing and mobile applications. It can also be used in military and surveillance applications for its low latency. To maintain compatibility with lots of previously created MPEG2 materials and current transmission using MPEG2, we developed a “transcoding” system from MPEG2 main profile to H.264 baseline profile targeting mobile applications specifically. The visual quality at a given input and output bit-rate is close to the H.264 main profile transcoding (re-use of parameters like motion vectors and quantization parameters) and at the same time computational complexity is at least 10-50% less than that of H.264 main profile transcoding and at least 10-40% more compression efficient than MPEG2.

I. INTRODUCTION

H.264, MPEG-4 Part 10, or AVC, for Advanced Video Coding, is a digital video codec standard which is noted for very high compression efficiency. It was written by the ITU-T Video Coding Experts Group (VCEG) together with the ISO/IEC Moving Picture Experts Group (MPEG) as the product of a collective partnership effort known as the Joint Video Team (JVT). The standard includes the eleven sets of capabilities, which are referred to as profiles, targeting specific classes of applications. One of them is Baseline Profile (BP), which is developed primarily for low-cost applications, demanding less computing resources than higher profiles: for this reason this profile is suitable for videoconferencing and mobile applications, as well as military and surveillance applications for its intrinsic low latency. Current digital transmissions still use mainly MPEG2 [1] as compression standard and at the same time we have lots of already created MPEG2 contents. The “transcoding” system from MPEG2 to H.264 baseline profile described in this paper, is mainly aimed to provide system compatibility for video data sharing between terminals with different capabilities, as well as improving the compression ratio. Moreover, H.264 Baseline profile (BP) can be more memory-efficient than main profile because B pictures are not allowed: this removes the constraint of having at least two reference frames instead of just a single one. Finally, the algorithm herewith described may allow a set-top-box acting as home server or VCR (like the STM7109) to be connected to a mobile device (like ST Nomadik® family SOC) or a generic low resources client and to stream there the received content. With the term “MPEG-2 to H.264 Baseline transcoding” we indicate the conversion process of an MPEG-2 compressed video bit stream to a new H.264 Baseline Profile standard. The core features of this transcoder are:

• MPEG2 decoding done up to pixel level to avoid drift
• Transcoding of MPEG2 main profile sequence into H.264 baseline profile sequence.
• Temporal and spatial resolutions are maintained equal to the original MPEG2 sequence.
• Achieving compression efficiency as much as possible with very low processing delay.
II. STATE OF THE ART

The main issues of H.264 baseline transcoding are the following:

- conversion of MPEG2 B pictures in P picture of H.264 (at macro block level),
- picture order management (at frame level),
- conversion with full resolution (both in temporal and spatial), not merely throwing away any frame or pixel values (in both macroblock and frame level),
- Conversion of field-coding of MPEG2 into frame coding of H.264.

The main differences between our solutions of these issues compared with that available in literature are mentioned here below:

- Dynamic Bitstream Shaper [3] developed within STMicroelectronics can transcode into H.264 main profile sequence out of an MPEG2 main profile sequence but cannot transcode into H.264 baseline profile since absence of interlacing. B frames in this profile causes no easy one-to-one mapping of parameters like motion vectors.
- The Field-to-frame transcoding is done in [4], but it has been done with reduction in temporal and spatial resolution. It drops B frames thus reducing temporal resolution and takes alternate lines of a frame i.e. only one field of a frame to avoid field-frame mapping of motion vector and quantization parameters, thus reducing spatial resolution reduction.
- Work in [5] describes the way to manage reference frames issue, but it re-determines motion vectors because here prediction is always made from the last decoded frame.
- References [6] and [7] describe ways of changing B picture into P picture (or B\textsubscript{fwd} picture). But they always make macroblock with backward reference as Intra. Whereas our present algorithm selects among intra and zero forward motion vector whichever produces lesser residual, so efficient in terms of compression.

Our solution is capable of producing H.264 baseline profile given any MPEG2 (specially main profile) sequence, keeping the original resolution of the input and at the same time giving good compression efficiency with much less time complexity than that of full-decode-full-encode approach.

III. THE METHODOLOGY

Motion information coming from MPEG2 includes both forward and backward motions for any macroblock. But H.264 Baseline profile only allows forward motion, i.e. conversion of B frame into P frame is the heart of the procedure. For the rest of the paper we call it “converted P” frame.

III.1. Macroblock level strategy for converted P frame

To make this conversion we took the simple strategy that in a “converted P” frame:

- The macroblock will be intra if the corresponding macroblock was coded intra in B picture of MPEG2 [Figure 1].
- The macroblock will use the forward motion vector if the corresponding macroblock had only forward motion vector in the B picture of MPEG2 [Figure 2].
- The macroblock will use the forward motion vector only if the corresponding macroblock had both the forward and backward motion vector in the B picture of MPEG2 [Figure 3].
- If the corresponding macroblock had only backward motion vector in the B picture of MPEG2 then either code it with forward motion vector zero or intra in H.264 baseline which ever is giving lesser residual value of pixel differences [Figure 4].

![Figure 1: Intra MB conversion](image1)
![Figure 2: P MB conversion](image2)
![Figure 3: Bi-dir MB conversion](image3)
![Figure 4: conversion of MB with backward motion only](image4)

Figures 1 to 4 show the conversion strategy of B frame to ‘converted P’ frame at macroblock level. Left blocks correspond to the input MPEG2 macroblock type and the right blocks represent co-located output H.264 macroblocks. Here ‘I’ stands for Intra macroblock, B\textsubscript{fwd} stands for MPEG2 forward predicted macroblock,
B\textsubscript{fwd/bck} stands for MPEG2 bi-directional macroblock (which is having both forward and backward motion vectors). P stands for H.264 forward predicted macroblock and B\textsubscript{bck} stands for MPEG2 backward predicted macroblock.

### III.II. Picture Order Management

The unavailability of B frames in the H.264 baseline output means that display and coding order of H.264 baseline sequence must be the same, i.e. same as the display order of MPEG2 sequence. This means that the coding order of MPEG2, given to the transcoder system by the original encoder, must be modified at the time of encoding in the H.264 baseline stream. We used the approach of B frame transforming for display order management. Picture Order Management requires correct Picture Order Count to be set for each and every encoded H.264 frame. The picture management data flow is depicted in picture 5.

- Encode the first I picture as IDR picture in H.264 as it comes from MPEG2 decoded engine.
- Store the content and information of the anchor (I/P) picture in a temporary buffer as it comes from MPEG2 engine (I/P), and encode the previously stored anchor picture (if any).
- Encode the “converted P” picture as the B picture comes from MPEG2 engine.
- Finally, encode the anchor picture (I/P) if there is any in the stored buffer.

The main advantage here is that the frame rate is not changed, so all the incoming information is transformed into the output stream (no information is going to be lost), maximizing the quality of service and user experience.

We have to note that what we called since now in this paper “Picture Order Count” is not exactly the POC described by the H.264 standard that has to be written into the output stream, neither the temporal reference parameter of the MPEG2 stream. In fact, for H.264 baseline POC should be increased by two after encoding every frame, and should reset at every IDR occurrence, while the MPEG2 temporal reference parameter is reset at every GOP header occurrence and it increases by one for each frame: this means that an adjustment of this parameter is needed run time in the code, even if its meaning is equivalent to the MPEG2 temporal reference or the picture encoding order number.

### III.III. Reference Picture Management

In MPEG2, prediction is always done from last decoded anchor frame(s).

![Figure 5: display/transmission order of MPEG2](image)

Figure 5 shows a typical GOP structure in MPEG2, where transmission order is I1,P1,B1,B2,P2, B3,B4...so on and the intended display order is I1,B1,B2,P1,B3,B4,P2 so on. That is P1 in MPEG2 is predicted from I1, but in H.264 baseline the last coded picture of P1 is P’2. (P’2 is the converted P picture of the incoming B2 picture). So P1 should not be predicted from P’2 in H.264 baseline using the same motion vector information as in that case P1 was predicted from I1 in MPEG2. So either we have to signal the baseline coder that P1 should be predicted from I1 instead of P’2, or we have to make search for motion vector.

H.264 standard allows optional insertion of any decoded picture into the reference picture list: no constraint is there for last decoded picture only. While encoding the picture in H.264 baseline the entire frame is treated as a slice and this is encapsulated in one NAL unit. Every NAL unit containing a “converted P” picture will have the parameter “nal_ref_idc” set to 0: this tells to the final H.264 decoder to avoid inserting the current picture into the reference frames buffer (as normally it would do). In this way, we are able to transcode keeping the original temporal resolution (i.e. frame rate), avoiding throwing B frames at all.
III.IV. Field-to-Frame Conversion

III.IV.I. Interlaced Frame-coded MPEG2 Sequence

A single MPEG2 “frame picture” will be presented including both fields (top and bottom) together, as in the progressive case; each macroblock, anyhow, may have field associated motion vectors: in this case the coding type may also be field, so containing separate motion vectors for each 8x16 partitions a macroblock. For baseline output comp liancy, the encoding part of the transcoder is compelled to make all macroblocks with frame motion. In this case field motion vectors coming from MPEG2 decoder do not have proper relevance to frame motion vector of H.264, so the frame motion vector is taken the one amongst the two field’s motion vector which ever is giving lesser residue after calculating the SAD (Sum of absolute difference). Then refine the motion vector accurate up to the half pixel, into quarter pixel accuracy as in all the cases.

III.IV.II. Interlaced Field-coded MPEG2 Sequence

MPEG2 decoder decodes one field at a time, but the encoding part of the transcoder cannot start at this time, because the entire frame’s data is not available at this time. Encoder saves the related information in a temporary buffer and waits for next field of current frame, which necessarily will arrive immediately after the current field. In this case the motion vector of the first field decoded by the MPEG2 decoder is taken, its vertical component will be accordingly scaled (because the reference now is the frame instead of the field); finally it will be refined to quarter pixel accuracy. The first field was predicted from last frame in MPEG2, so this motion vector of it has more relevance than second field’s motion vector in making frame macroblocks in H.264 baseline out of those field macroblocks in MPEG2 (the second field MV may be pointing to the opposite field of current frame).

IV. RESULT AND CONCLUSION

H.264 baseline profile allows neither B frames, nor interlaced coding. It does not have several other features, which its main profile has (e.g. CABAC [2]). So compression efficiency of H.264 baseline is much lower than that of its main profile. But still it is better than MPEG2 main profile in term of compression efficiency. Proper re-use of information of the MPEG2 decoder leads us to create efficiently H.264 baseline sequences, without wasted HW/SW resources and/or video content information. To test the reliability of the proposed scheme, together with its quality and compression performances, the following MPEG2 reference test sequences have been used:

- Akina (720x480, 100 frames, 25fps,@7mbps)
- foreman (352x288, 250 frames, 25 fps @3mbps)
- empire (1280x720, 137 frames, 60 fps, @10 mbps)
- flowers ( 720x576, 79 frames, 25 fps, @ 4 mbps)
- walk (176x144, 300 frames, 25 fps, @ 900 kbps)
- Films (352x288, 908 frames, 25 fps, @2mbps)

Some more detailed observations are described below with supporting tables and graph.

IV.I Mpeg2 progressive sequences:

- H.264 Baseline profile transcoding always gives about 20 to 40 % better compression than that of original MPEG2 sequence if no rate control is used.
- Main profile transcoding [3] is always efficient in terms of compression than baseline profile. General performance wise it is about 10 to 15% better and main profile does not use CABAC for fare comparison.
- Baseline profile is always efficient in transcoding time. Always baseline profile gives more than 10% time gain than that of main profile [3].
- Quality of baseline profile is comparable with main profile. In most of the cases PSNR level falls less than 0.5 dB than that of main profile [3] transcoded output. Figure 6 shows the result for foreman sequence.

IV.II. Mpeg2 Interlaced (frame or field) sequences:

- H.264 Baseline profile gives about 10 to 40% more compression than that of original MPEG2 sequence if no rate control is used. But for some sequences, it is 10 – 20% worse than the original MPEG2. The reason behind it is field coding is actually not related to frame coding.
Main profile transcoding [3] is about 10 to 15% better than baseline transcoding in terms of compression efficiency (main profile does not use CABAC for fare comparison). For some cases main profile [3] output is much better than the baseline output due to the same reason of field-to-field conversion.

Always baseline profile gives more than 10% time gain than that of main profile, and at least 50% faster than full-encoding.

In most of the cases PSNR level falls less than 1 dB than that of main profile transcoded output.

![Figure 6: PSNR with varying bit-rate (FOREMAN)](image)

### Table 1: Comparison of BP with MP (AKINA: 3.33 MB)

<table>
<thead>
<tr>
<th>Profile</th>
<th>PSNR (Y) (dB)</th>
<th>PSNR (U) (dB)</th>
<th>PSNR (V) (dB)</th>
<th>Size (in mb)</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP 2-f</td>
<td>39.08</td>
<td>40.90</td>
<td>43.05</td>
<td>2.23</td>
<td>217</td>
</tr>
<tr>
<td>BP 2-f</td>
<td>38.79</td>
<td>40.62</td>
<td>42.67</td>
<td>2.46</td>
<td>176</td>
</tr>
<tr>
<td>BP 1-f</td>
<td>38.79</td>
<td>40.62</td>
<td>42.67</td>
<td>2.46</td>
<td>152</td>
</tr>
</tbody>
</table>

### Table 2: Comparison of BP with MP (WALK: 1.43 mb)

<table>
<thead>
<tr>
<th>Profile</th>
<th>PSNR (Y) (dB)</th>
<th>PSNR (U) (dB)</th>
<th>PSNR (V) (dB)</th>
<th>Size (in mb)</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP 2-f</td>
<td>38.48</td>
<td>43.94</td>
<td>43.07</td>
<td>1.02</td>
<td>48</td>
</tr>
<tr>
<td>BP 2-f</td>
<td>38.26</td>
<td>43.56</td>
<td>42.73</td>
<td>1.16</td>
<td>45</td>
</tr>
<tr>
<td>BP 1-f</td>
<td>38.26</td>
<td>43.56</td>
<td>42.73</td>
<td>1.16</td>
<td>34</td>
</tr>
</tbody>
</table>

### Table 3: Comparison of BP with MP (FILMS: 8.65 mb)

<table>
<thead>
<tr>
<th>Profile</th>
<th>PSNR (Y) (dB)</th>
<th>PSNR (U) (dB)</th>
<th>PSNR (V) (dB)</th>
<th>Size (in mb)</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP 2-f</td>
<td>38.22</td>
<td>42.17</td>
<td>43.13</td>
<td>5.31</td>
<td>541</td>
</tr>
<tr>
<td>BP 2-f</td>
<td>37.68</td>
<td>41.85</td>
<td>42.79</td>
<td>5.48</td>
<td>461</td>
</tr>
<tr>
<td>BP 1-f</td>
<td>37.67</td>
<td>41.85</td>
<td>42.77</td>
<td>5.49</td>
<td>393</td>
</tr>
</tbody>
</table>

*n-f -> 'n' number of reference frames
MP-> main profile, BP-> baseline profile

Xcoding time -> time includes decoding and encoding time

*results of all tables correspond to direct re-use of QP from MPEG2

V. REFERENCES