H.263 to VP6 Video Transcoder

Chris Holder and Hari Kalva
Dept. of Computer Science and Engineering
Florida Atlantic University, Boca Raton, FL 33431

ABSTRACT

VP6 is a video coding standard developed by On2 Technologies. It is the preferred codec in the Flash 8/9 format used by many popular online video services and user generated content sites. The wide adoption of Flash video for video delivery on the Internet has made VP6 one of the most widely used video compression standards on the Internet. With the wide adoption of VP6 comes the need for transcoding other video formats to the VP6 format. This paper presents algorithms to transcode H.263 to the VP6 format. This transcoder has applications in media adaptation including converting older Flash video formats to Flash 8 format. The transcoding algorithms reuse the information from the H.263 decoding stage and accelerate the VP6 encoding stage. Experimental results show that the proposed algorithms are able to reduce the encoding complexity by up to 52% while reducing the PSNR by at most 0.42 dB in the worst case.

Keywords: H.263, VP6, transcoding , complexity reduction

1. INTRODUCTION

The VP6 video coding standard was developed by On2 Technologies and is the codec of choice for use in Flash 8 & 9. The VP6 video format, by virtue of its inclusion in Flash, has seen explosive adoption and is now one of the most widely used compression formats on the internet, especially in the burgeoning area of user generated content. The suitability of VP6 for online video applications is well summarized by Uro in his blog on the reasons for why VP6 was picked over H.264 [1]. While VP6 is widely used for online video services, MPEG codecs such as MPEG-2 and H.264 are used in virtually all broadcast video applications as well as DVD and high definition DVD. The coexistence of multiple video formats requires transcoding and efficient transcoding is necessary for scalable services. Previous version of Flash video is based on H.263 and converting those videos to the current Flash format also requires video transcoding. The key problem addressed in transcoding applications is complexity reduction – how to efficiently transcode video without sacrificing quality. Transcoding algorithms typically reuse the information from the decoding stage to accelerate the encoding stage.

This paper presents algorithms for transcoding H.263 video to the VP6 format. The goal is to effectively reuse the information gathered during the H.263 decoding stage and speed up the VP6 encoding stage. The effectiveness of this reuse depends on the similarities and differences between the input and output video formats. The similarities can help exploit prior results in video transcoding. When the input and output video formats are very similar, transform domain transcoding can be used [2]. Transform domain transcoding requires only partial decoding and transform coefficients are converted from the input format to the output format. If the differences in coding formats are significant, transform domain transcoding becomes complex and pixel domain transcoding is necessary. In pixel domain transcoding, video is fully decoded and the encoding stage is accelerated by reusing the information from the input video [3]. The differences in H.263 and VP6 make it complex to use transform domain transcoding and pixel domain transcoding is employed. An overview of related research is presented in Section 2 and a short summary of H.263 and VP6 codecs is presented in Section 3. The transcoding algorithms and results are presented in Section 4 and conclusions in Section 5.

2. RELATED RESEARCH

There is no publish work on VP6 coding or transcoding. The related research is in the area of pixel domain video transcoding that reuses macro block (MB) coding mode information. Transcoding H.263 to H.264 is presented in [4]. This approach uses the motion compensated residual from the H.263 stream to determine the MB coding modes in H.264. Transcoding between H.264 and MPEG-2 reported in [5] reuses H.264 motion information to reduce the complexity of MPEG-2 encoding stage. The MPEG-4 to H.264 transcoding reported in [6] and the VC-1 to H.264

This work was supported through a grant from On2 Technologies Inc.

Visual Communications and Image Processing 2008, edited by William A. Pearlman, John W. Woods, Ligang Lu,
Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 6822, 68222B, © 2008 SPIE-IS&T · 0277-786X/08/$18

transcoding reported in [7] are also based on pixel domain transcoding and are effective in reducing transcoding complexity. Of all the prior work, the transcoding techniques developed in this paper are similar to the H.264 to MPEG-2 transcoding reported in [5]. However, the results cannot be directly applied because of the codec differences.

3. OVERVIEW OF H.263 AND VP6

3.1. H.263 Video Coding

H.263 is a video coding standard developed under the auspice of the ITU. The standard was developed targeting low bitrate video applications in telecommunications. A brief overview of H.263 baseline profile follows. H.263 is a hybrid video coding format similar to the MPEG video codecs. Video frames are encoded as a set of macro blocks that represent a non-overlapping 16x16 region of the frame. A video frame can be encoded as Intra (I) or Inter (P) frame. Motion compensation is performed on per macro block basis and each MB can have either 1 or 4 motion vectors (MV). The motion compensated differences are transformed with DCT and the coefficients are coded with variable length codes. The baseline profile of H.263 with Annex-D, unrestricted motion vector mode, and Annex-F, Advanced prediction mode, is used as the input to the transcoder.

3.2. VP6 Video Coding

This section provides a brief overview of VP6 with emphasis on the features that are relevant to the proposed transcoding algorithms. VP6 is also a hybrid codec that uses motion compensated transform coding at its core. The codec has Intra and Inter pictures similar to MPEG video codecs. Intra pictures are coded independent of other coded pictures and Inter pictures use previously coded pictures for prediction. Motion compensation supports 16x16 and 8x8 blocks similar to H.263 but the Inter 8x8 macro blocks can have mixed blocks; i.e., one or more 8x8 blocks can be coded in Intra mode without using any prediction. The Inter MBs in VP6 can be coded using 9 different modes. The modes are characterized by the number of motion vectors (1 vs. 4), reference frame used, whether motion vectors are coded. Where motion vectors are not coded, the motion vectors are predicted from previously decoded MBs. The VP6 codec uses 8x8 DCT for transform coding and de-blocking filter is applied at the block boundaries. We do not provide a complete overview of VP6 features due to space considerations.

Table 1 summarizes the similarities and differences between H.263 baseline profile and the VP6 encoder. The large number of coding modes supported in VP6, the ¼ pixel resolution motion vectors, and multiple reference frames allowed make transcoding from H.263 to VP6 difficult. The transcoding algorithms are designed for Inter frames and focus on reusing the motion information.

Table 1: Comparison of VP6 and H.263 Coding Features

Feature	H.263 Baseline	VP6
Picture type	I, P	I, P
Transform size	8x8	8x8
Transform	DCT	Integer DCT
Intra prediction	None	None
Motion comp.	16x16, 8x8	16x16, 8x8
Total MB modes	4	10
Motion vectors	½ pixel	¼ pixel
Deblocking filter	None	Yes
Reference frames	1	Max 2

4. TRANSCODING H.263 TO VP6

The similarities and differences between H.263 and VP6 provide opportunities for reusing H.263 MB coding mode details for reducing the transcoder complexity. The fact that both H.263 and VP6 support 1 MV and 4 MV modes means that motion vectors can be reused to some extent. However, the fact that VP6 supports large number of MB modes compared to H.263 means that the H.263 MB mode and motion vectors cannot be used directly. The differences in the codecs mean that an Inter 16x16 MB in H.263 is not necessarily coded as an Inter 16x16 MB. Table 2 shows the typical example of MB coding modes when encoding H.263 decoder output using VP6. For this example, a Foreman video sequence at 352x288 resolution and 297 frames is encoded using H.263 at 384 Kbps and then transcoded to VP6 using full re-encoding at 291 Kbps. The full details of VP6 modes are not given here due to space considerations. In brief, Nearest and Near MB modes do not code motion vectors and derive their MVs from previously coded MBs; Golden frames are long term reference frames, and Inter 0,0 forces the use of a 0,0 motion vector. Each row corresponds to a H.263 MB coding mode and the columns give the VP6 mode used to code those MBs. For example, of all the MBs that are coded as Inter 4V in H.263, 3% were coded as Inter0,0 mode, 1% coded as Intra, 30% coded as Inter+MV, 11% nearest, 7% near, and 47% are coded as Inter 4V MBs. Thus, if an Inter 4V MB in H.263 is mapped to Inter 4V in VP6, it is likely to map correctly only in 50% of the cases. Thus direct mode mapping will lead to poor results and more efficient algorithms are necessary.

Table 2: MB Mode Mapping for Foreman Sequence (CIF, 297 frames) in H.263 to VP6 Transcoding

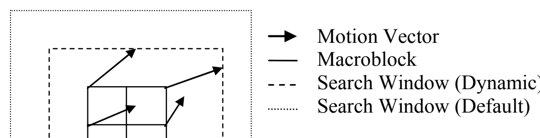
	Inter 0,0	Intra	Inter+ MV	Nearest	Near	Golden	Golden MV	Inter 4V	Golden Nearest	Golden Near
Inter	17%	0%	21%	38%	21%	0%	1%	2%	0%	0%
Inter 4V	3%	1%	30%	11%	7%	1%	1%	47%	0%	0%
Intra	11%	33%	4%	9%	4%	11%	3%	19%	4%	2%
Skipped	60%	0%	0%	31%	8%	0%	0%	0%	0%	0%

Based on these observations, we have developed transcoding algorithms that reuse H.263 motion vectors and reduce the search space in VP6 based on H.263 MB modes and motion vectors. As discussed, the simple mode mapping of Intra to Intra, Inter 4V to Inter 4V etc. will lead to poor RD performance due to a large mismatch in MB coding modes. From table 2, we can also see patterns that allow us to restrict which modes are evaluated for a given H.263 modes. Near and Nearest are computationally inexpensive to evaluate and are allowed in all cases. Inter 4V, on the other hand, takes significant computation and is evaluated only when input MB is also in the Inter 4V mode. The transcoding algorithms thus reduce the complexity by placing constraints on MB modes evaluated and further reduce the complexity by using: 1) Dynamic search range and 2) Dynamic search window.

All the results reported in this paper use the TMN 3.2 H.263 encoder from University of British Columbia which is based on Telenor's H.263 implementation. The input video is coded at 384 Kbps in baseline profile with advanced motion options and one I frame (first frame). A decoder based on the same H.263 implementation is used in the decoding stage of the transcoder. The VP6 encoding stage is based on the optimized VP6 encoder software provided by ON2 Technologies. The VP6 video is encoded with I frame frequency of 120 and at multiple bitrates to assess the RD performance of the transcoder. The results are compared with the baseline transcoder that performs full encoding in the VP6 stage.

4.1. Complexity Reduction Using Dynamic Search Range

The dynamic search range approach sets the search range used for motion estimation for each MB. Typically this range is fixed through out the encoding process and is set to 15 in the experiments. With the knowledge of motion vectors in H.263, the search range no longer has to be fixed. The search range is changed based on the maximum motion vector component for the current MB. Figure 1 shows the dynamic search range selection based on H.263 motion vectors. The RD performance of the proposed method is compared to the baseline transcoder in Figure 3. The results for three of the sequences evaluated are show and the performance of the proposed algorithm closely tracks the RD performance of the baseline transcoder. The PSNR drop is higher for the Stefan sequence because of large motion in the sequence.



$$Range_x = \max|MV_x|$$

$$Range_y = \max|MV_y|$$

Figure 1. Dynamic search range

4.2. Complexity Reduction Using Dynamic Search Window

Using a dynamic refinement window further reduces the complexity by reusing the H.263 motion vectors. Unlike the dynamic search range method where window location is fixed and the window size or search range is varied, the dynamic search window approach uses the H.263 motion vectors to determine the position of the fixed sized window. Window sizes of 1x1 and 3x3 are evaluated. This approach reduces the complexity more than the dynamic range approach due to an even smaller search space. This reduction in complexity comes at a slight increase in PSNR loss. Figure 2 shows the dynamic window derived based on the H.263 motion vectors of a MB. Figure 4 shows a RD plot comparing the dynamic window approach to the baseline approach. The performance of 3x3 and 1x1 windows is very close and 1x1 window can be used without significantly affecting the RD performance.

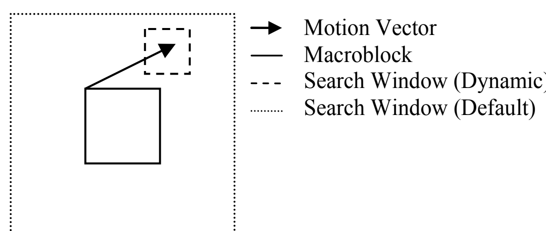


Figure 2. Dynamic search window

Table 3 shows the performance summary of the proposed algorithms. The performance of the dynamic range (DR) algorithm is better in sequences with large motion compared to low motion sequences as can be seen in the case of Stefan. The dynamic range also suffers the least PSNR drop as indicated by a max PSNR drop of 0.11 dB for Stefan sequence. The dynamic window reduces the total transcoding time by as much as 52% for the Stefan sequence. The Foreman sequence shows PSNR loss of as high as 0.42 dB for dynamic window. The max PSNR loss and max bitrate increase are reported for the full QP range that lead to very high bitrates at lower QPs. For a practical bitrate range of up to 600 Kbps for CIF sequences, the RD performance is almost identical to the reference transcoder.

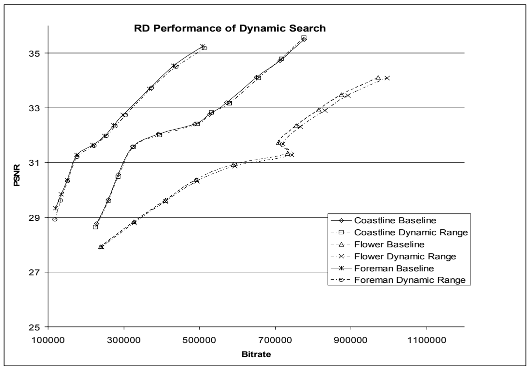


Figure 3. RD performance of dynamic search

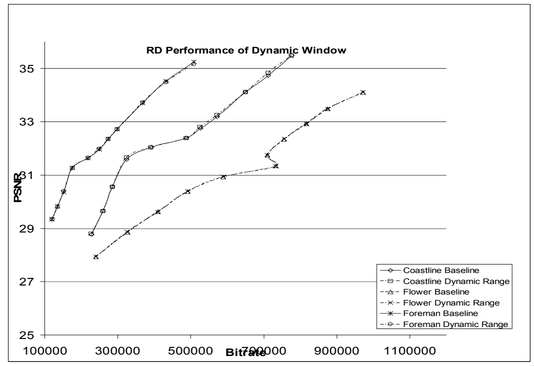


Figure 4. RD performance of dynamic window

Table 3: Transcoder performance summary

	Coastguard			**Foreman**			**Stefan**		
	DR	DW1x1	DW3x3	DR	DW1x1	DW3x3	DR	DW1x1	DW3x3
Max PSNR loss	0.02	0.09	0.12	0.08	0.41	0.42	0.11	0.14	0.16
Max Bitrate incr	0.1%	1.6%	1.2%	0.4%	2.2%	1.5%	0.6%	0.47%	0.46%
Ave Complexity Reduction	10.2%	47.1%	46.9%	3.5%	39%	39%	21%	52%	51%

5. CONCLUSIONS

This paper presents algorithms for transcoding H.263 to VP6 format. The wide use of VP6 in online video services creates a need for transcoding and the low complexity transcoding is critical to support large scale video services. The similarities and differences between H.263 and VP6 were exploited to reduce the complexity. The large differences in MB coding modes makes direct mapping of MB modes very inefficient. Based on the mode mapping likelihood for typical video sequences, a constrained set of MB modes are evaluated for a given H.263 MB mode. The evaluation cost is reduced by using dynamic range and dynamic window approaches. The results show that the proposed transcoder is able to reduce the complexity by more than 50% without a significant loss in PSNR. Given that the VP6 implementation used is highly optimized, the resulting savings of 50% is considered significant. Transcoders based on this approach will be able to transcoder at least 50% more streams for the same hardware configuration.

REFERENCES

1. T. Uro, "The quest for a new video codec in Flash 8," http://www.kaourantin.net/2005/08/quest-for-new-video-codec-in-flash-8.html, August 13, 2005.
2. A. Vetro, C. Christopulos, and H. Sun, "Video transcoding architectures and techniques: An overview," IEEE Signal Process. Mag., vol. 20, no. 2, pp. 18–29, Mar. 2003.
3. G. Fernandez-Escribano, H. Kalva, P. Cuenca, and L. Orozco-Barbosa, "Very Low Complexity MPEG-2 to H.264 Transcoding Using Machine Learning," Proceedings of the ACM Multimedia 2007, Santa Barbara, CA, October 2006, pp. 931-940.
4. J. Bialkowski, A. Kaup, K. Illgner, "Overview of Low-Complexity Video Transcoding from H.263 to H.264," Proceedings of the IEEE International Conference on Multimedia & Expo (ICME) 2006, pp. 49-52.
5. P. Kunzelmann and H. Kalva, "Reduced Complexity H.264 to MPEG-2 Transcoder," Proceedings of the IEEE International Conference on Consumer Electronics, Las Vegas, USA, January 10-14, 2007.
6. Y.-K. Lee, S.-S. Lee, and Y.-L. Lee, "MPEG-4 to H.264 Transcoding using Macroblock Statistics," Proceedings of the IEEE International Conference on Multimedia & Expo (ICME) 2006, pp. 57-60.
7. J.B. Lee and H. Kalva, "An Efficient Algorithm for VC-1 to H.264 Transcoding in Progressive Video Compression," Proceedings of the IEEE International Conference on Multimedia & Expo (ICME) 2006, pp. 53-56.