

REAL-TIME SOFTWARE MPEG-2 TO H.264 VIDEO TRANSCODING

K.H. Goh, D.J. Wu, J.Y. Tham, T.K. Chiew and W.S. Lee

Institute for Infocomm Research

21 Heng Mui Keng Terrace, Singapore 119613

Email: khgoh@i2r.a-star.edu.sg

ABSTRACT

The paper describes a fast MPEG-2 to H.264 transcoder by leveraging the MPEG-2 metadata set to reduce the computational complexity of the H.264 encoding process. With the proposed fast transcoding algorithms, our software-based transcoder can perform real-time transcoding a MPEG-2 movie from a DVD disc (at D1 resolution and full frame-rate) and simultaneously performing RTP/UDP streaming of the H.264 video with AC-3 audio over IP networks using only a 1.6 GHz PC notebook. We can also achieve about 20-30% faster than the full re-encoding method even at the same target bit-rate and without any visible quality degradation.

Index Terms—Transcoding, Transcaling and H.264

1. INTRODUCTION

Video transcoding is about transforming compressed video bit-streams from one format to another format. It can also include transforming into different bit-rates and different picture sizes. Video transcoding enables video content provider to transform video sources from different formats to the desired transmission format, and also empowers them to provide the content to heterogeneous devices with different hardware capabilities such as PDA, pocket PC and mobile phones. Almost all the digital video contents for broadcasting as well as in DVDs are encoded using MPEG-2 now. It is also known that H.264 is becoming more attractive to broadcasters due to its superior coding efficiency, hence motivating the demand for MPEG-2 to H.264 format transcoding. Very high speed transcoding is also desired in broadcasting industries as they are usually dealing with hundreds of channels simultaneously.

2. PRIOR ARTS AND OBJECTIVES

A simplistic approach towards transcoding is to decode a bit-stream to obtain the reconstructed images and perform a full re-encoding on these images. This method is far from ideal as the bulk of the re-encoding step has been done in the original encoder and information on the various decisions made can be found in the decoded bit-stream. How this information is used will determine the efficiency of various transcoding schemes. Some existing methods [1,2] analyses the MPEG-2 8x8 DCT DC & AC coefficients

of the neighboring blocks to decide the Intra prediction direction. However this method could still be computational. Other existing research such as in [3] is on performing the transcoding in the DCT domain for Intra coding, i.e. fast conversion algorithm of the DCT coefficients to integer transform coefficients. However, there might be drifting error problem in such approach. Method such as [2] proposed checking the AC coefficients of MPEG-2 8x8 DCT to see if the MB is smooth. Some other fast method using previously coded information is also proposed [4], depending on the activity of the various block size.

This paper focus on using pixel-domain transcoding methods in which MPEG-2 sequences are decoded to obtain the reconstructed frames and by just using a simple meta-data set to provide the H.264 encoder with additional information to reduce encoding complexity. Three area of fast transcoding is presented in this paper. One objective is to transcode to H.264 baseline profile which has no B-picture encoding targeting for live transcoding applications. Another objective is to make fast inter and intra-coding decisions. Simple information from MPEG2 is used to determine the MB activity and help to make fast coding decisions. The third objective is to perform fast transcoding from 4CIF to CIF which will convert the MPEG-2 bit-stream to a down-sized H.264 bit-stream for real-time streaming to mobile device applications.

3. DETAILED FAST TRANSCODING ALGORITHMS

The MPEG-2/H.264 transcoder described in this paper is a pixel-domain transcoder which consists of an optimized MPEG-2 decoder which outputs both reconstructed frames as well as a meta-data set pertaining to the decoder stream. The re-encoder makes use of the meta-data set to perform fast motion vector determination and mode selection for encoding of the reconstructed frame. The detailed transcoder architecture is shown in figure 1.

3.1. Fast motion vectors derivations

The baseline profile H.264 is used in this paper for live transcoding application. As baseline profile uses only P-picture, for pictures coded in B-directional type (B-picture), they are to be encoded as P-picture in H.264. The motion vectors used in MPEG-2 will be used to predict the motion vectors to be used in the H.264 encoding. However these

predicted motion vectors (PMV) are not expected to be accurate for some cases, therefore further re-estimation for such cases will be performed by using the PMV as the centre point for a finer search. The algorithm of predicting the PMV and the decision making of re-estimation is described in the flow diagram figure 2. For each MPEG-2 inter-coded MB, the available forward motion vectors will be used to derive the predicted MV for the H.264 P-picture encoding. These forward motion vectors are scaled to one-frame reference picture distance, as the encoding in H.264 are P-picture referencing the previous picture. For example, referring to figure 3, the B2 forward MV having reference picture distance of 2 frames will be scaled by half and used as the PMV for P2 in H.264.

To keep the motion estimation process in H.264 to the minimal, motion re-estimation flag is set when the MPEG-2 forward reference picture distance is greater than 1-frame, or there is only backward MV available, i.e. the B-picture is backward predicted, which could mean that the forward prediction does not have a good match. The backward MV will still be used to predict the forward MV by simple “negating” the MV and scaled to 1-frame distance.

To further reduce the unnecessary re-estimation, the SAD of the PMV is first checked, the motion re-estimation is performed using the PMV as the search centre only if the SAD at PMV is greater than a certain threshold.

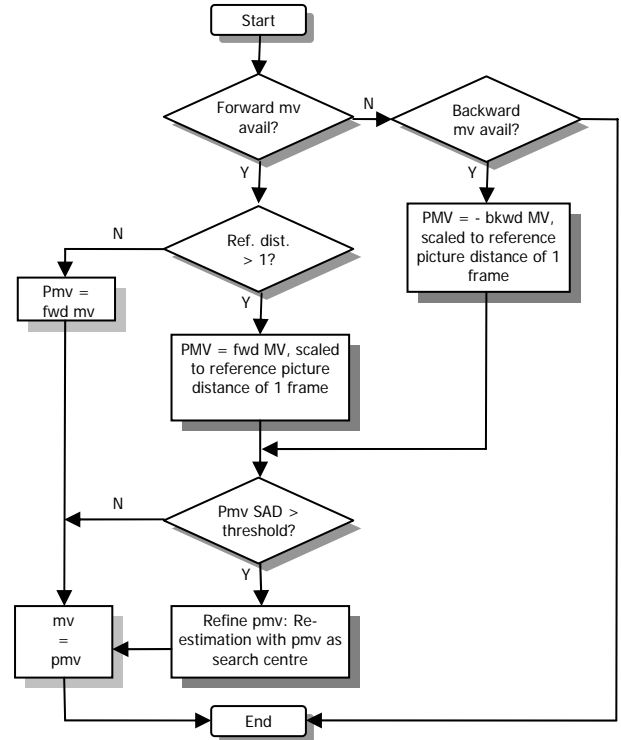


Figure 2: MV Derivation flow diagram

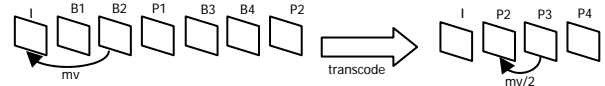


Figure 3: MV derivations from B to P picture

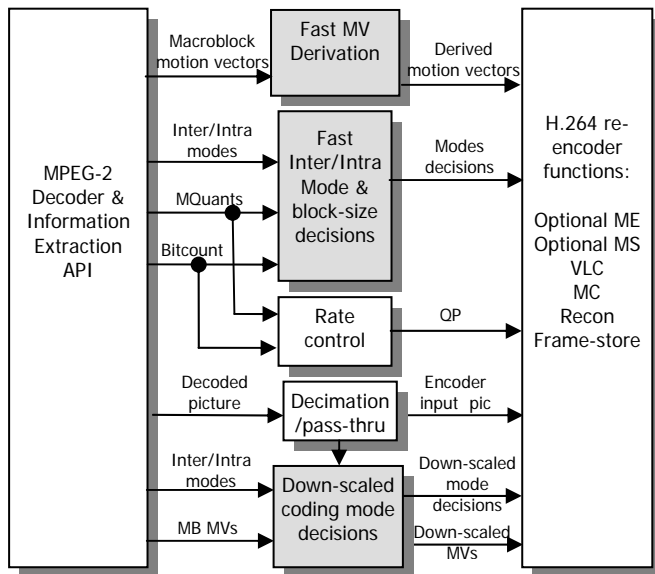


Figure 1: Transcoder system architecture

3.2 Fast Coding Mode Selection

As the MPEG-2 metadata contains inter/intra mode-decision information, they are re-used during the H.264 encoding. It should be noted that the MPEG-2 ‘skip’ mode shall not be used directly as H.264’s skip mode as the skip mode definition in each standards are different. When encounter MPEG-2 skip mode, the transcode will use inter mode with motion vectors of the previous MB instead.

3.2.1 Fast Intra-mode decision

For Intra-mode coding, checking of the directional prediction mode for all 16x16 and 4x4 modes will be too computational for live transcoding objective. Therefore the modes are limited to 16x16_DC-mode or 4x4_DC-mode. MPEG-2 intra-coding is based on 8x8 DCT (4 blocks for each MB), if the MB is smooth, coding in H.264 as 16x16 DC mode will be beneficial in terms of coding efficiency. If the macroblock has high spatial activity, coding as 16x16 might not be as efficient. A fast method is used whereby the bits-used information from the MPEG-2 bit-stream is used to decide 16x16 or 4x4 DC-mode. Generally, with the same quantization value, a smoother macroblock will use lesser bits than a more detailed macroblock type. Therefore the number of bits used to encode per macroblock in MPEG-2 and the quantization-step value for the macroblock are used to check on whether the macroblock is smooth or detailed, and hence decide if the MB is to be intra-code as 16x16 or 4x4 DC-mode. The decision method is illustrated as the following pseudo code:

```

If (Bit_count_for_MB > Activity_thres[Quant_step] ) then
    Intra_mode_decision = 4x4_DC_mode;
Else
    Intra_mode_decision = 16x16_DC_mode;

```

The `Bit_count_for_MB` is the bits used to encode the MB in MPEG-2 which could be easily extracted from the MPEG-2 bits-stream. The `Activity_thres[.]` is a look-up table function of the quantization-step-size used to encode the MPEG-2 intra-coded macroblock.

3.2.2 Conditional inter to intra mode-switching

The bit-count is again used to check if the MPEG2 inter/intra decision is good, e.g. inter-coded macroblock using too many bits might be better coded in intra instead. A simple threshold method can be used to decide the inter-to-intra mode switching:

*If (MPEG2_inter_MB_bit_count > inter_threshold) then
Switch to intra-coded MB*

The `inter_threshold` can be a function of the quantization-step-size used, such that a higher threshold is used for a lower quantization-step-size. This conditional switching method is specifically useful for bad inter-mode decision in the MPEG-2 encoding. For example some not so efficient software MPEG-2 encoder chooses the inter-coding mode for all P and B pictures without checking, for simplicity reason. Therefore, for such bitstreams that contains bad inter-mode decision (i.e. no intra-coding in P/B frames), this fast conditional inter to intra mode-switching will be useful so that MB that supposed to be coded as intra will be detected and coded as intra to achieve better coding efficiency.

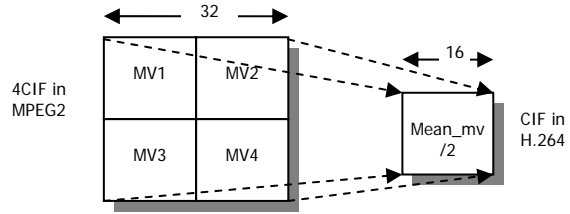
3.3 Adaptive Rapid Decimated Coding Decisions

For transcaling to smaller picture, for example down-sizing from 4CIF to CIF, as each macroblock is down-sized from 4 MB, the Inter/Intra mode decision and the predicted motion vectors is required to be determined from the 4 MBs' information. In this fast algorithm, the MPEG-2 information used consists of the MB inter/intra mode decision and the MB motion vectors. If any of the 4 MBs is intra-coded, then intra mode is chosen for the down-sized MB, else the inter mode is chosen.

3.3.1 Adaptive inter block-size and MV determination for down-sized MB

As in the motion vector for the down-sized picture a single decimated macroblock is derived from 4 macroblocks, there are 4 motion vectors to choose from. Logically, if the 4 motion vectors are all moving in the similar directions, i.e., they are very close to each other, then the motion vector for the MB downsized from these 4 macroblocks can be determined by the mean of these 4 respective motion vectors, and the MB is coded as inter-16x16 mode. However, if any one of these 4 motion vectors are moving in a different directions or they are not close enough to each other, then the MB is coded as 4 inter-8x8 mode where each of these 8x8 motion vectors are the scaled-down version of the original 4 MBs' motion vectors. This is illustrated in the figure below:

If the 4 MVs are all close enough to each other =>
Then choose **inter-16x16 mode** using (mean_MV)/2



If the 4 MVs are not close enough to each other =>
Then choose **inter-8x8 mode** using respective MVs

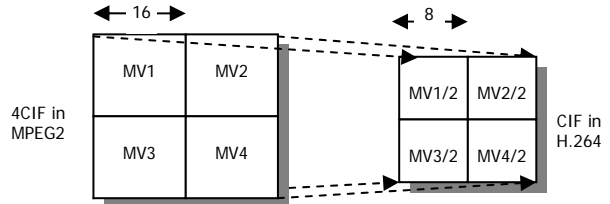


Figure 4: Adaptive inter-mode block-size determination

The detailed decision algorithm to decide the whether inter-16x16 or inter-8x8 mode is described in the following pseudo code:

*If [max{d(mv1,mv2), d(mv1,mv3), d(mv1,mv4), d(mv2,mv3), d(mv2,mv4), d(mv3,4)} > threshold] then
Choose inter-8x8;*

Else

Choose inter-16x16 mode;

where `max()` is the function that finds the maximum value of the inputs, and `d(x, y)` is the function that compute the distance between motion vectors `x` and `y`. The function `d(MV1, MV2)` is to compute the distance of 2 motion vectors, two methods can be used:

Method 1: The precise distance computation,

$$d(mv1[x,y], mv2[x,y]) = (mv1[x] - mv2[x])^2 + (mv1[y] - mv2[y])^2$$

Or

Method 2: Simplified fast computation,

$$d(mv1[x,y], mv2[x,y]) = Abs(mv1[x] - mv2[x]) + Abs(mv1[y] - mv2[y])$$

Therefore, as long as the distance between any two of the four corresponding motion vectors exceeds a certain threshold, then it is considered that the 4 corresponding MVs are not near to each other, and the inter-8x8 is chosen for the decimated MB using the corresponding MVs (scaled to half as shown in figure 4) for each of the 8x8 blocks. Similarly, if all the motion vector distances between each other are smaller than the threshold, then they are considered as near to each other and the inter-16x16 is chosen for the decimated MB using the mean of the 4 MVs, scaled to half.

4. RESULTS AND OBSERVATIONS

4.1 Fast MV Derivation Speed Improvement

The simulation results of the fast MV derivation as described in section 3.1, compared to the full re-encode

scheme without using the motion vectors metadata, are shown in the following table, with both the C-code running on a 1.6GHz notebook PC, and the MPEG-2 bit-streams used are CIF format encoded at 1 Mbps:

Test Stream	Without MV Metadata (Full re-encoding)			Using our fast MV algorithm (maintaining similar PSNR)		
	fps	PSNR-Y	Kbps	fps	AVG Δ kbps	Δ fps
Mobile	42.81	38.81	1481	49.55	-4.44%	15.59%
Stefan	46.93	28.81	998.2	55.46	2.04%	18.05%
Foreman	52.27	33.37	419.2	63.18	6.03%	21.01%

The transcoding speed improvement is around 15% - 21% by using the motion vector meta-data; however, there is also slight increase in bit-rate, meaning that the motion vectors in the MPEG-2 is not as accurate as the re-motion estimation performed in the H.264 encoding which is expected as the forward motion vectors for H.264 are determined from motion vectors not referencing the same temporal reference frame due to bi-directional motion estimation. However, subjective evaluations have shown that there is no visible quality difference between full re-encoding and our fast transcoder when maintaining the same bit-rate.

4.2 Fast Coding Decisions Performance

The following table shows the results of the fast Intra mode and conditional inter to intra switching (as described in 3.2):

Test-stream	With only Intra_16x16_DC			Adaptive Intra_16x16_DC/ Intra_4x4_DC		Inter to intra switching
	fps	PSNR-Y	Kbps	AVG Δ kbps	Δ fps	AVG Δ kbps
Mobile	46.49	38.81	1790	-0.48%	0.93%	-0.01%
Stefan	46.14	26.03	1072	-4.20%	1.36%	-4.68%
Foreman	50.86	29.49	502.5	-4.05%	-0.09%	0.00%

The results verified that the simple and fast adaptive intra mode decision can improve the coding efficiency with a reduced bit-rate of more than 4%, while maintaining similar PSNR. The adaptive inter to intra switching also helps to reduce the bit-rate (Stefan) when the motion search for inter-mode coding might not perform well for fast motion.

4.3 Fast Decimated Coding Decision Results

By using the algorithms as described in section 3.3 to decide the intra/inter mode and adaptively choosing the inter-16x16 or inter-8x8 mode, the complexity can be reduced quite significantly, as shown in the following table:

Test Stream	Without using MPEG2 metadata (Re-encoding)			Speed improvement using the fast decimated coding decisions		
	fps	PSNR_Y	Kbps	Δ PSNR	Δ kbps	Δ fps
Mobile	45.45	38.81	1481	0.01	-0.38%	12.9%
Stefan	42.56	28.81	998	-0.12	1.96%	8.32%
Foreman	46.01	37.83	1640	-0.15	2.21%	6.9%

For the adaptive method described in section 3.3.1, the difference of non-adaptive and adaptive method is shown in the next result table (decimated from CIF to QCIF):

Test-stream	Non adaptive inter-block size (fixed 16x16)		With adaptive inter-16x16 and inter-8x8
	fps	Kbps	AVG Δ kbps
Mobile	142.9	611	-1.53%
Stefan	136.14	352	-0.03%
Foreman	154.86	525	-1.19%

4.4 Real-time Performance and Subjective Evaluations

Combining all the fast algorithms, the transcoder is able to achieve constantly 20-30% faster than the full re-encoding method. With further code optimization of the software code, it has been shown with live demo that our fast transcoder is able to perform 'live' full DVD resolution transcoding of a DVD movie at full frame-rate on a 1.6GHz notebook PC with simultaneous RTP/UDP streaming of the H.264 video stream and AC-3 audio over IP networks to another PC client decoder. The table below shows the average frame per second and PSNR performance for real-time transcoding of three test DVD movie clips:

Original MPEG-2 DVD@D1	Cartoon (1-min, 2.9Mbps) Transcoded @ 2.9 Mbps		Star Wars (10-mins, 4.3Mbps) Transcoded @ 2.7 Mbps		Matrix Trailer (5-mins, 4.5Mbps) Transcoded @ 2.8 Mbps	
	fps	PSNR	fps	PSNR	fps	PSNR
Using Fast Transcoder	31.6	38.7	26.0	39.9	27.3	39.3

Subjective quality evaluations of transcoding using the full re-encode method and the fast transcoding method were made. It was found that when transcoded at similar bit-rate, no significant visual difference is observed between the full re-encode and our fast transcoder.

5. CONCLUSION

This paper proposed a fast MPEG-2 to H.264 transcoder which is capable of performing real-time software-only transcoding and streaming of D1-resolution video. Simulation results have shown that it performs very well in terms of speed improvement, and has no visual degradation when compared to the full decode and re-encode method at similar bit-rate.

6. REFERENCES

- [1] Kalva H. and Petljanski B, "Exploiting the directional features in MPEG-2 for H.264 intra transcoding", IEEE Transactions on Consumer Electronics, Volume 52, Issue 2, May 2006 Pg.706-711.
- [2] Li Wang, Qi Wang, Yu Liu and Wei Lu, "A fast Intra Mode Decision Algorithm for MPEG-2 to H.264 Video Transcoding", IEEE 10th International Symposium on Consumer Electronics (ISCE) 2006, 28-01 June 2006, Pg.1-5.
- [3] Gao Chen, Shouxun Lin and Yongdong Zhang, "A Fast Coefficients Conversion Method for the Transform Domain MPEG-2 to H.264 Transcoding", International Conference on Digital Telecommunications (ICDT) 2006, 29-31 Aug. 2006 Pg.17-17.
- [4] Xiaolan Lu, Tourapis, A.M., Peng Yin and Boyce, J., "Fast mode decision and motion estimation for H.264 with a focus on MPEG-2/H.264 transcoding", IEEE International Symposium on Circuits and Systems (ISCAS) 2005, 23-26 May 2005, Pg.1246-1249 Vol. 2.