Efficient MPEG-4 to H.264 transcoding exploiting MPEG-4 block modes, motion vectors, and residuals

Isabelle Metoevi and Stéphane Coulombe
Department of Software and IT Engineering
Ecole de Technologie Supérieure
1100 Notre-Dame Ouest, Montréal, Qc, H3C 1K3
E-mail: yehouessi-isabelle.metoevi1@etsmtl.ca, stephane.coulombe@etsmtl.ca

***Abstract*—In this paper, we present an efficient algorithm to transcode MPEG-4 to H.264. The algorithm exploits the information decoded from the MPEG-4 stream to reduce H.264 encoding complexity. This information includes the MPEG-4 block modes, motion vectors, and residuals. The algorithm proceeds in two steps. First, a small set of most probable H.264 block mode candidates are obtained from an MPEG-4 to H.264 block mode conversion table. Then, motion estimation is performed for the candidate modes where, based on the residual information, the MPEG-4 motion vectors are either reused or refined. Experimental results show that the algorithm can speed-up the transcoding of QCIF and CIF sequences, from MPEG-4 visual simple to H.264 baseline profiles, by a factor of 2 to 3, with an acceptable loss in quality compared to the cascade spatial domain transcoding approach. It also provides significantly improved quality relative to current state-of-the-art methods.**

I. Introduction

The diversity of multimedia applications and terminals inevitably causes interoperability problems. For instance, current mobile terminals support different video standards, such as H.263, MPEG-4 [1], and H.264/AVC [2]. The transcoding of video content to specific resolution, standard, and bit rate constraints has become a necessity in order to ensure the success of evolving multimedia communications. The MPEG-4 visual simple profile (VSP) is widely used in today's multimedia services, including mobile videoconferencing, multimedia message service (MMS), and streaming within the scope of 3GPP/3GPP2 [3]–[6]. The more recent H.264/AVC standard provides significant improvements in compression efficiency and is expected to replace the earlier standards, thereby making transcoding from MPEG-4 to H.264 inevitable.

Several studies have investigated the problem of video transcoding in general and MPEG-4 to H.264 in particular [7]–[13]. Their goal has been to reduce processing complexity while maintaining the best video quality. The most obvious transcoding approach, the cascade approach, consists of fully decoding the MPEG-4 video bitstream to the spatial (pixel) domain and then re-encoding it according to the H.264 specification. The best video quality is reached with this type of transcoding. Unfortunately, it is highly computationally complex, which is not always suitable for real-time applications. H.264 encoding is especially complex, because of its more sophisticated coding tools. H.264 uses several block modes: 4 inter modes (16x16, 16x8, 8x16, and 8x8), 4 sub-modes (8x8, 8x4, 4x8, and 4x4), a SKIP mode, and two intra prediction modes (16x16 and 4x4). To determine the best block coding mode, H.264 uses rate distortion optimization (RDO). So, for several candidate modes, it will perform motion estimation (ME) and motion compensation (MC) (up to 41 ME operations at quarter-pixel precision for a single macroblock (MB)).

Several methods have been proposed to reduce this computational complexity [8], [9], [11], [12]. The most efficient of these exploit the information available from the MPEG-4 decoder to reduce the number of block modes to evaluate, thereby reducing ME complexity. In [8], the authors exploit the frequency distribution of the H.264 block modes for a given MPEG-4 block mode in order to derive an MPEG-4 to H.264 block mode conversion table. An example of such a table is presented in Table I. Motion vectors (MVs) from MPEG-4 are then reused after a refinement process. However, they do not provide much detail on the refinement process, and the simulation results are not extensive. In [9], an arbitrary mapping between MPEG-4 block modes and H.264 candidate block modes is presented (without much justification), for both intra and inter blocks. MVs are either directly reused (in 16x16 mode) or become the starting points for ME (in 16x8 and 8x16 modes, for instance). They obtain very good speed-ups, but the quality is degraded by 1 to 2dB, which may be unacceptable in some applications.

In this paper, we propose to exploit the decoded residual information, in addition to the block modes and MV information gathered from the MPEG-4 decoding stage, to further improve MPEG-4 to H.264 transcoding performance in terms of speed and quality. The system is illustrated in Fig.1. The algorithm proceeds in two main steps. First, we reduce the number of H.264 candidate block modes based on the decoded MPEG-4 block modes using a table similar to that in [8], but enriched with the residual and MV information. Then, we determine the MVs for the candidate modes. The MVs are only refined when required based on residual data. We evaluate the SAD for all candidates and select the optimal block mode from conventional H.264 RDO.

II. Determination of candidate block modes

Current video compression standards use two key techniques: motion compensated predictive coding and transform coding. Predictive coding reduces the temporal redundancy

TABLE I
STATISTICAL MAPPING OF THE BLOCK MODES FROM MPEG-4 TO H.264 FOR QCIF CARPHONE VIDEO AT 128 KBIT/S USING INTEL'S H.264 ENCODER. THE CANDIDATE BLOCK MODES USED IN [8] ARE SHOWN IN BOLD.

MPEG-4 coding modes \ H264 coding modes	Intra 4x4	Intra 16x16	SKIP	Inter 16x16	Inter 16x8	Inter 8x16	Inter 8x8	sb8X8	sb8X4	sb4X8	sb4X4
Intra	**49.2%**	**41.4%**	2.1%	4.8%	1.1%	0.5%	0.9%	68.8%	0.0%	12.5%	18.7%
Inter16x16(64%)	0.0%	0.0%	31.3%	**61.6%**	**3.2%**	**2.9%**	**1.0%**	**92.7%**	1.9%	4.4%	1.0%
Inter8x8(25%)	0.0%	0.0%	2.8%	**37.7%**	**15.6%**	**19.3%**	**24.6%**	**88.7%**	**3.4%**	**4.9%**	**3.0%**
Skip(11%)	0.0%	0.0%	**89.3%**	**10.2%**	0.3%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%

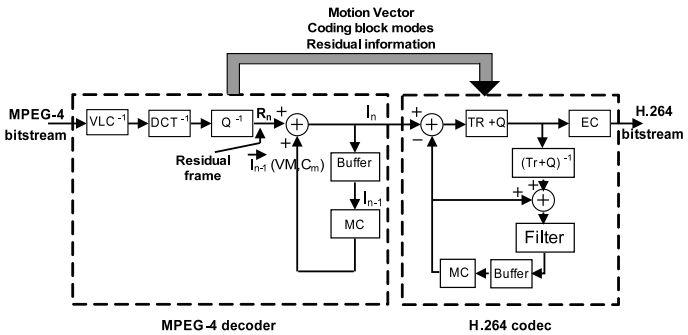

Fig. 1. Proposed fast MPEG-4 to H.264 transcoder architecture reusing MPEG-4 block modes, motion vectors, and residual information.

between frames by subtracting a predicted frame, obtained from the ME process, from the frame to encode in order to produce a prediction error frame (also know as residual information). The residual information typically has significantly less energy than the original image and can therefore be coded with fewer bits. The more accurate the prediction process is, the less energy will be contained in the residual information. Therefore, this information can be used as a measure of the efficiency of the ME process, including the suitability of the MV and the block mode (if we have selected the right block sizes).

Studying the cascade transcoding of MPEG-4 to H.264 led us to the following observations, which are exploited in our MPEG-4 to H.264 block mode conversion table:

- MPEG-4 blocks using 16x16 Inter mode are most often coded as either H.264 SKIP or 16x16 Inter blocks. Indeed, if 8x8 blocks would have been better in H.264, then this mode would most likely have been chosen for MPEG-4 too.
- MPEG-4 blocks using 16x16 Inter mode are most often coded as H.264 SKIP blocks if the residual energy is *low* and the MV is close to the predicted MV.
- MPEG-4 blocks using 16x16 Inter mode are most often coded as H.264 16x16 Inter blocks if the residual energy is *high* (but not so high that Intra mode is preferable).
- MPEG-4 blocks using 8x8 Inter mode may be coded in a variety of H.264 modes, including SKIP, 16x16, 18x8, 8x16, 8x8, etc. However, 8x4, 4x8, and 4x4 modes are rarely used [14].
- MPEG-4 SKIP and 16x16 Inter modes are used most often in video coding applications and have the most impact on computational complexity (which is not to say that 8x8 should be ignored) [14].
- MPEG-4 Intra blocks represent a small percentage of all coded blocks in a mobile video application, since key frames are infrequent to maintain coding efficiency and therefore have a small impact on computational complexity [2].

We now present some definitions. Let $I(x,y)$ and $J(x,y)$ with $0 \leq x,y \leq 15$ be MBs of the original and predicted images respectively. The residual for the MB is defined as:

$$R(x,y) = I(x,y) - J(x,y),\ 0 \leq x,y \leq 15 \tag{1}$$

The residual energy for the MB is defined as:

$$E = \sum_{x=0}^{15}\sum_{y=0}^{15} R^2(x,y) = \sum_{x=0}^{15}\sum_{y=0}^{15} [I(x,y) - J(x,y)]^2 \tag{2}$$

It is often useful to determine the residual energy for each 8x8 sub-block of an MB. Let us define E_k, the energy of the residual information of a k-th 8x8 sub-block of an MB, as follows:

$$E_k = \sum_{x=0}^{7}\sum_{y=0}^{7} R^2(x+p_{kx}, y+p_{ky}) \tag{3}$$

with $\mathbf{p}_k = [p_{kx}, p_{ky}]$ for $0 \leq i \leq 3$, where $\mathbf{p}_0 = [0,0]$, $\mathbf{p}_1 = [8,0]$, $\mathbf{p}_2 = [0,8]$, and $\mathbf{p}_3 = [8,8]$. Clearly, the residual energy E of an MB is the sum of the energies E_k of the four 8x8 blocks, expressed as $E = \sum_{k=0}^{3} E_k$.

We performed extensive simulations on QCIF (176×144) and CIF (352×288) videos at different bit rates with the cascade approach to analyze the probability distribution of mapping decisions from MPEG-4 information (block modes, MVs, and residual energy) to H.264 block modes. The test set included videos with various characteristics in terms of motion and details. We used Intel's video codecs for MPEG-4 and H.264 implementations (some details and justifications are presented in section IV). In order to classify MBs having *low* and *high* residual energy, we had to empirically set two thresholds Thr_low and Thr_high. The expectation was that if Thr_low was set properly, 16x16 Inter blocks with a residual energy below Thr_low and an MV similar to the predicted MV would be coded as SKIP with a very high probability, thereby eliminating the need to search for other candidates. Similarly, we were expecting that if Thr_high was set properly, blocks with a residual energy above Thr_high would

be coded as Inter16x16 with a very high probability. We have limited this strategy to 16x16 Inter blocks, since they represent the highest percentage of MPEG-4 block modes (for most mobile videos), and this alone brought important performance improvements. However, the concept of partitioning based on residual energy could be extended to 8x8 blocks. The thresholds have been empirically set to $\{Thr\_low = 125, Thr\_high = 5000\}$ through careful analysis and comparison of hundreds of simulations.

Table II shows the results obtained for the transcoding of the QCIF Carphone video sequence. This sequence was initially encoded in MPEG-4 VSP at 200kbit/s and then re-encoded in H.264 baseline at 128kbit/s. Although the specific values vary with each video sequence and bitrate, the distribution among modes remains mostly the same. In the table, the rows are partitioned as follows:

- Intra-I: MPEG-4 Intra MBs from an Intra frame. We can observe that they tend to be re-encoded in Intra mode.
- Intra-P: MPEG-4 Intra MBs from an Inter frame. We can observe that they tend to be re-encoded in SKIP or Inter16x16 modes.
- Inter16x16_case1: MPEG-4 Inter16x16 MB with $E < Thr\_low$, $|V_x - Vp_x| \leq 1$ and $|V_y - Vp_y| \leq 1$, where $\mathbf{V} = [V_x, V_y]$ is the decoded MPEG-4 MV and $\mathbf{Vp} = [Vp_x, Vp_y]$ is the predicted MV from the H.264 encoding stage. We can observe that, as expected, this type of MB tends to be re-encoded as SKIP most of the time.
- Inter16x16_case2: MPEG-4 Inter16x16 MB with $Thr\_low \leq E \leq Thr\_high$ or such that $E < Thr\_low$ but $|V_x - Vp_x| > 1$ or $|V_y - Vp_y| > 1$. This type of MB tends to be re-encoded as either SKIP or Inter16x16 most of the time.
- Inter16x16_case3: an Inter16x16 MPEG-4 with the $E > Thr\_high$. This type of MB tends to be re-encoded as Inter16x16 most of the time.
- Inter8x8: MPEG-4 Inter8x8 MB. Although half the time these blocks are re-encoded in Inter16x16 mode, the remaining half includes several modes with comparable probability. However, blocks smaller than 8x8 are not highly probable and could be ignored. This may be due to the Intel H.264 encoder's behavior.
- SKIP: MPEG-4 SKIP MB. Usually re-encoded as SKIP.

Note that in the table the values under sb8x8, sb8x4, sb4x8, and sb4x4 in the gray shaded area are respectively the mapping percentages of the sub-blocks 8x8, 8x4, 4x8, and 4x4 with respect to the Inter8x8 mode. The table also shows the distribution of each type of MB with respect to the Intra and Inter modes. For instance, 91% of MPEG-4 Intra MBs are Intra-I, while 9% are Intra-P. For MPEG-4 non Intra MBs, 17% are in Inter16x16_case1, 37% in Inter16x16_case2, 10% in Inter16x16_case3, 25% in Inter8x8, and 11% in SKIP.

In Table II, the probabilities in bold represent those with the highest probability. We propose to limit the set of H.264 candidate block modes to the ones associated with these bold values. More specifically, the sets of H.264 candidate block modes as a function of the various MPEG-4 block categories are as follows:

- Intra-I: The candidates are Intra16x16 and Intra4x4.
- Intra-P: Inter16x16
- Inter16x16_case1: SKIP
- Inter16x16_case2: SKIP and Inter16x16
- Inter16x16_case3: Inter16x16
- Inter8x8: SKIP, Inter16x16, Inter16x8, Inter8x16, Inter8x8.
- SKIP : remains SKIP.

As expected, the proposed method significantly reduces the number of candidate modes tested relative to previous methods [8], [9] where four candidate modes are typically tested.

III. Efficient ME based on residual information

Motion estimation is a very computationally intensive operation in a transcoder. In order to reduce the computation burden, state-of-the-art transcoding algorithms reuse the decoded MPEG-4 MVs as much as possible. However, the compression performance of an encoder highly depends on the MVs. A change in MV accuracy from quarter to half pixel can increase the video quality by ~ 2 dB, depending of the video type. In the H.264 standard, the MVs are at quarter-pixel accuracy, while in the MPEG-4 standard they can be at quarter- or half-pixel accuracy, depending of the profile supported: half-pixel for the visual simple profile and quarter-pixel for the advanced simple profile. In this paper, we consider the VSP supported by most MPEG-4 mobile applications [3]–[6]. To improve the accuracy of the MVs from the MPEG-4 decoder, we should refine them from half-pixel to quarter-pixel precision. Unfortunately, this refinement is quite demanding computationally. In order to decrease the complexity, we propose to refine the MVs only as needed. By doing so, we can significantly reduce the complexity, contrary to [9], where all the MVs are refined to quarter-pixel accuracy.

We propose to exploit the residual information once again, in order to determine whether or not an MV requires refinement. Indeed, we have already mentioned that the residual information can be used as an efficiency measure of ME. For each candidate mode, we propose to test the energy E or E_k for the various regions to code. If the energy is below a threshold, the MV will be kept as is, otherwise it will be refined from half-pixel to quarter-pixel accuracy. We also propose to use the fast refinement algorithm used in the Intel MPEG-4 encoder [15]. According to that method, 5 half-pixel positions p_i are evaluated instead of 8 quarter-pixel positions to find the best position. The refinement algorithm is described in Fig. 2 where b_i is the sum of absolute differences (SAD) of the position pixel p_i.

For the decision as to whether or not to refine, we used two thresholds, $Thr16$ and $Thr8$, for Inter16x16 and Inter8x8 decoded MPEG-4 block modes respectively. Through analysis and experimentation, we came to the conclusion that these thresholds had to be bitrate-dependent in order to maintain a certain level of quality. Indeed, as the bitrate is reduced, the H.264 encoder's RDO tends to map more MBs to the SKIP mode, which has the effect of decreasing quality. As a matter of fact, the smaller the bitrate, the smaller the SAD

TABLE II
PROPOSED STATISTICAL MAPPING OF THE BLOCK MODES FROM MPEG-4 TO H.264 FOR QCIF CARPHONE VIDEO AT 128KBIT/S. THE CANDIDATE BLOCK MODES ARE SHOWN IN BOLD.

MPEG-4 coding modes \ H264 coding modes	Intra 4x4	Intra 16x16	SKIP	Inter 16x16	Inter 16x8	Inter 8x16	Inter 8x8	sb8X8	sb8X4	sb4X8	sb4X4
Intra-I (91%)	**54.3%**	**45.7%**	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Intra-P (9%)	0.0%	0.0%	21.9%	**51.2%**	12.2%	4.9%	9.8%	68.8%	0.0%	12.5%	18.7%
Inter16x16_case0 (17%)	0.0%	0.0%	**73.6%**	25.3%	0.6%	0.5%	0.0%	0.0%	0.0%	0.0%	0.0%
Inter16x16_case1 (37%)	0.0%	0.0%	**26.4%**	**66.4%**	3.4%	3.2%	0.6%	98.0%	0.6%	0.6%	0.8%
Inter16x16_case2 (10%)	0.0%	0.0%	1.9%	**81.3%**	7.2%	5.7%	3.9%	89.2%	2.8%	6.8%	1.2%
Inter8x8 (25%)	0.0%	0.0%	**2.8%**	**37.7%**	**15.6%**	**19.3%**	**24.6%**	88.7%	3.4%	4.9%	3.0%
SKIP (11%)	0.0%	0.0%	**89.3%**	10.2%	0.3%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%

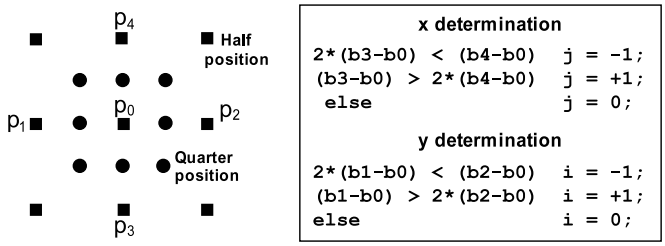

Fig. 2. Half- to quarter-pixel MPEG-4 MV refinement algorithm [15].

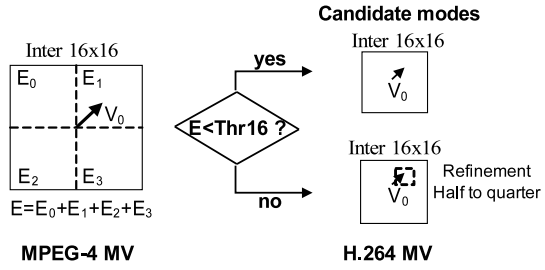

Fig. 3. Reuse and refinement of MPEG-4 MV in H.264 for the case of an Inter16x16 MB.

of an Inter block has to be in order to be assigned to Inter mode (the overhead cost associated with transmission of MVs becoming increasingly important). As a consequence, as the bitrate becomes smaller, we have to reduce the thresholds to increase the number of MVs that will be refined, leading to smaller SAD values, and consequently increasing quality. The determination of optimal thresholds as a function of the bitrate in accordance with the RDO process could be the topic for future research. Nevertheless, we obtained good results using the same methodology as before, showing the benefits of the proposed approach, by setting $\{Thr8 = 62.5, Thr16 = 500\}$ for small bitrates (64 kbits and below for QCIF sequences, 256 kbit/s and below for CIF sequences) and $\{Thr8 = 250, Thr16 = 2000\}$ for higher bitrates. It is worth noting that the threshold values have a direct impact on the tradeoffs the system will make between computation complexity and video quality. Small thresholds increase quality, but also computational complexity, and they can be adjusted to meet the specific transcoding system's requirements.

Figs. 3 and 4 illustrate the MV determination and refinement process of an MPEG-4 Inter16x16 and Inter 8x8 MB for Inter candidate modes. The process is as follows for the various decoded MPEG-4 MB modes:

- Inter16x16 with decoded MPEG-4 MV $\mathbf{V}_0$: $\mathbf{V}_0$ is refined to quarter-pixel accuracy when the residual energy $E > Thr16$; otherwise, we use $\mathbf{V_0}$.
- Intra-P: The Inter16x16 MV is found by performing ME (using the EPZS [18] algorithm in our simulations), since there is no initial MV. But, because there are so few of these types of MB, they have no noticeable impact on speed.
- Inter8x8 with decoded MPEG-4 MVs $\mathbf{V}_k$, $0 \leq k \leq 3$: For Inter8x8 candidate mode, a given $\mathbf{V}_k$ is refined to quarter-pixel accuracy when the residual energy $E_k > Thr8$; otherwise, we use $\mathbf{V}_k$. For the candidate modes with a partition larger than 8x8, we always refine the MV (although we could extend our residual energy-based method to these cases). For the case of the Inter16x16 candidate mode, we compute the SAD, at half-pixel accuracy, with the four MV candidates $\mathbf{V}_k$ and select the one with smallest SAD. The MV is then refined to quarter-pixel accuracy. A similar process is performed on Inter16x8 and Inter8x16 candidate modes.

Once the MVs and corresponding SADs have been determined for all candidate block modes, the final block mode is selected using H.264 RDO.

IV. EXPERIMENTAL RESULTS

The proposed method, along with other state-of-the-art methods, were implemented in the Intel IPP (Intel Integrated Performance Primitives) code samples, version 5.3 [15]. These video codecs are highly optimized compared to the MPEG-4 and H.264 reference codecs (MoMuSys [16] and JM [17]). Although the H.264 JM is an excellent reference to validate rate distortion performance, it is not optimized for speed and therefore cannot be used as a reliable reference to measure improvements in speed. The results on Intel's codecs are much more representative of the gains obtainable on a real transcoding product, although it may also use less exhaustive algorithms. The video sequences were initially encoded with

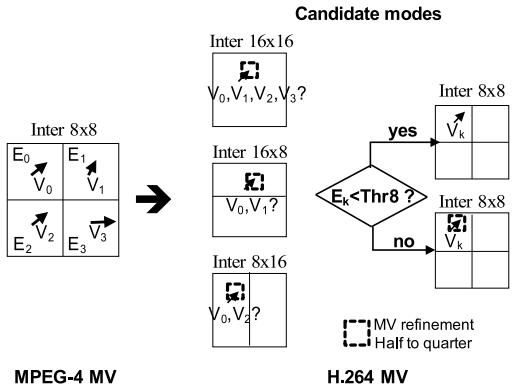

Fig. 4. Reuse and refinement of MPEG-4 MV in H.264 for the case of an Inter8x8 MB.

high quality using MPEG-4 VSP at 30fps with one Inter frame every 100 Inter frames (i.e. every 3.3s) at 200kbit/s and 720kbit/s for QCIF and CIF respectively (other initial rates were tested with small differences in final performance). No B frames were used. The H.264 encoding options were: RDO, maximum quality, one reference frame, and SADT (sum of absolute transform difference) instead of SAD.

We measured the quality (PSNR) and the computation times of the following methods: cascaded transcoding, MV refinement with mode selection (MS) [9], the statistical method with and without refinement [8], and the proposed method. The performance of each method was compared against the cascade method. The results for various video sequences are presented in Figs. 5 and 6. The results are quite impressive. The proposed algorithm is, on average, 2 to 3 times faster than the cascade method with only $\sim$ 0.5dB loss in quality. We observe that, as the bitrate increases, the difference in quality decreases with respect to the cascade method and the gains in speed increase. Only the statistical approach without refinement is faster than the proposed method. However, its PSNR is significantly lower (2.0dB less, on average, for QCIF and 2.5dB for CIF) than the proposed method. Such a loss in quality is often unacceptable in many applications. In addition, we could probably obtain similar gains in speed by changing the threshold values (to be validated in future work). Compared to the MV refinement with the MS algorithm, the proposed algorithm is 30% faster, on average, and provides better quality (1dB better, on average, for QCIF and 1.5dB for CIF). The differences in quality between the proposed method and state-of-the-art methods is particularly noticeable at low bitrates.

The results presented in [8] and [9], were obtained with the reference codecs MoMuSys and JM. Under a more optimized codec, such as that of Intel, their speed-ups are much less impressive. For instance, [9] (MV refinement and MS) obtained an average speed-up of 10.36, while we obtained an average of 2 using Intel codecs.

V. Conclusion

In this paper, we proposed an efficient algorithm for MPEG-4 to H.264 transcoding. By exploiting the residual information gathered in the MPEG-4 decoder in addition to the MVs and block modes, we were able to significantly improve the speed (by a factor of 2 to 3) while maintaining good quality compared to the cascade method. The method also provides superior results compared to state-of-the-art methods. The impressive speed-ups make our algorithm very suitable for real-time applications. The approach is expected to be applicable to other transcoding use cases as well, such as H.263 to H.264. In future research, we will investigate automatic threshold determination more deeply and extend the approach to smaller 8x8 blocks to improve performance even further.

VI. Acknowledgments

This work was funded by the Vantrix Corporation and by the Natural Sciences and Engineering Research Council of Canada under the Collaborative Research and Development Program (NSERC-CRD 326637-05).

References

[1] ISO/IEC 14496-2, "Information technology - Coding of audio-visual objects - Part 2: Visual," second edition, December 2001.

[2] ISO/IEC 14496-10 AVC and ITU-T rec. H.264, "Advanced video coding for generic audiovisual services," March 2005.

[3] 3GPP TS 26.234 v7.7.0, "Packet-switched Streaming Services (PSS); Protocols and codecs (Release 7)," March 2009.

[4] 3GPP TS 26.140 v7.1.0, "Multimedia Messaging Service (MMS); Media formats and codecs (Release 7)," June 2007.

[5] 3GPP2 C.S0045-A, "Multimedia Messaging Service (MMS) Media Format and Codecs for cdma200 Spread Spectrum Systems," version 1.0, March 2006.

[6] 3GPP2 C.S0046-0, "3G Multimedia Streaming Services," version 1.0, February 2006.

[7] B. Shen, "From 8-tap DCT to 4-tap integer-transform for MPEG-4 to H.264/AVC transcoding," *IEEE international conference on image processing,* Vol. 1, pp. 115-118, October 2004.

[8] Y. K. Lee, S. S. Lee and Y. L. Lee, "MPEG-4 to H.264 transcoding using macroblock statistics," *IEEE international conference on multimedia and expo,* pp. 57-60, July 2006.

[9] Y. Liang, X. Wei, I. Ahmad and V. Swaminahan, "MPEG-4 to H.264/AVC transcoding," *The International Wireless Communications and Mobile Computing Conference,* pp. 689-693, August 2007.

[10] T. N. Dinh, J. Yoo, S. Park, G. Lee, T. Y. Chang and H. J. Cho, "Reducing spatial resolution for MPEG-4 / H.264 transcoding with efficient motion reusing," *IEEE international conference on computer and information technology,* pp. 577-580, October 2007.

[11] S. E. Kim, J. K. Han and J. G. Kim, "Efficient motion estimation algorithm for MPEG-4 to H.264 transcoder," *IEEE international conference on image processing,* Vol. 3, pp. 659-702, September 2005.

[12] T. D. Nguyen, G. S. Lee, J. Y. Chang and H. J. Cho, "Efficient MPEG-4 to H.264/AVC transcoding with spatial downscaling," *ETRI,* Vol. 29, pp. 826-828, December 2007.

[13] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: an overview," *IEEE Signal Processing Magazine*, 20(2):18-29, 2003.

[14] B. G. Kim and S. K. Song, "Enhanced inter mode decision based on contextual prediction for P-slices in H.264/AVC video coding," ETRI journal, Vol.28, Number4. pp 425-434, August 2006.

[15] Intel Integrated Performance Primitives 5.3 - Code Samples. [Online]. http://software.intel.com/en-us/articles/intel-integrated-performance-primitives-code-samples/.

[16] ISO/IEC 14496-5:2001, "Information technology - Coding of audio-visual objects - Part 5: Reference Software," second edition, February 2005.

[17] H.264/AVC reference software JM 15.1. [Online]. http://iphome.hhi.de/suehring/tml/.

[18] A.M. Tourapis, "Enhanced predictive zonal search for single and multiple frame motion estimation," in Visual Communications and Image Processing, Jan.2002, pp. 1069-1079.

QCIF Videos		32 Kbit/s					64 Kbit/s					96 Kbit/s					128 Kbit/s				
		Cascade	MV+MS	Statistics	Statistics+ref	Proposed	Cascade	MV+MS	Statistics	Statistics+ref	Proposed	Cascade	MV+MS	Statistics	Statistics+ref	Proposed	Cascade	MV+MS	Statistics	Statistics+ref	Proposed
Akiyo	PSNR(dB)	39.18	-0.49	-2.34	-0.2	-0.46	42.27	-0.14	-1.7	-0.08	-0.33	43.98	-0.12	-1.16	-0.08	-0.48	45.28	-0.09	-0.89	-0.09	-0.35
	Speed up	1	1.54	2.28	1.41	1.9	1	1.59	2.44	1.4	2.03	1	1.66	2.55	1.42	2.35	1	1.73	2.6	1.44	2.34
Miss-America	PSNR(dB)	40.19	-1.23	-2.38	-0.48	-0.54	43.08	-0.61	-1.91	-0.33	-0.56	44.42	-0.39	-1.59	-0.24	-0.8	45.34	-0.33	-1.37	-0.21	-0.79
	Speed up	1	1.72	2.7	1.56	2.25	1	1.82	3.21	1.68	2.73	1	1.84	3.38	1.69	3.13	1	1.85	3.48	1.7	3.23
Mobile	PSNR(dB)	23.54	-3.54	-4.98	-0.68	-0.44	26.09	-1.01	-4.6	-0.27	-0.16	27.41	-0.59	-3.79	-0.13	-0.14	28.3	-0.36	-3.12	-0.06	-0.17
	Speed up	1	1.82	2.84	1.62	2.12	1	1.78	3.47	1.73	2.47	1	1.8	3.86	1.76	2.66	1	1.81	4.02	1.77	2.8
Carphone	PSNR(dB)	30.23	-7.34	-3.61	-1.38	-0.77	33.37	-2.68	-2.74	-0.65	-0.42	35.2	-1.5	-2.31	-0.39	-0.46	36.49	-0.98	-2	-0.29	-0.44
	Speed up	1	2.02	2.76	1.45	1.85	1	1.97	3.37	1.59	2.22	1	1.97	3.72	1.65	2.53	1	1.97	3.9	1.67	2.65
News	PSNR(dB)	31.89	-2.11	-2.61	-0.63	-0.46	35.93	-0.94	-2.2	-0.43	-0.43	38.24	-0.71	-1.93	-0.46	-0.64	39.78	-0.49	-1.49	-0.36	-0.54
	Speed up	1	1.74	2.45	1.45	1.87	1	1.8	2.81	1.54	2.08	1	1.86	2.95	1.5	2.31	1	1.82	2.89	1.5	2.27
Foreman	PSNR(dB)	29.22	-7.7	-3.83	-1.25	-0.6	32.65	-2.97	-3.01	-0.56	-0.29	34.31	-1.41	-2.39	-0.3	-0.34	35.48	-0.91	-2.09	-0.23	-0.36
	Speed up	1	2.1	2.86	1.56	1.94	1	2.04	3.57	1.65	2.29	1	1.99	3.87	1.67	2.62	1	1.96	3.94	1.7	2.69
Average PSNR(dB)		32.38	-3.74	-3.29	-0.77	-0.55	35.57	-1.39	-2.69	-0.39	-0.36	37.26	-0.79	-2.2	-0.27	-0.48	38.45	-0.53	-1.83	-0.21	-0.44
Average speed up		1	1.82	2.65	1.51	1.99	1	1.83	3.14	1.6	2.3	1	1.85	3.39	1.62	2.6	1	1.86	3.47	1.63	2.66

CIF Videos		128 Kbit/s					256 Kbit/s					384 Kbit/s					512 Kbit/s				
		Cascade	MV+MS	Statistics	Statistics+ref	Proposed	Cascade	MV+MS	Statistics	Statistics+ref	Proposed	Cascade	MV+MS	Statistics	Statistics+ref	Proposed	Cascade	MV+MS	Statistics	Statistics+ref	Proposed
Mother-daughter	PSNR(dB)	38.63	-2.11	-2.15	-0.62	-0.38	41.67	-0.87	-1.9	-0.33	-0.37	43.12	-0.54	-1.71	-0.19	-0.86	44.14	-0.39	-1.53	-0.15	-0.83
	Speed up	1	1.85	2.92	1.59	2.2	1	1.86	3.45	1.67	2.6	1	1.87	3.9	1.69	3.1	1	1.9	3.77	1.72	3.29
Stefan	PSNR(dB)	25.85	-6.29	-4.97	-1.09	-0.45	29.54	-2.3	-4.08	-0.63	-0.28	31.17	-1.17	-3.2	-0.37	-0.27	32.27	-0.79	-2.61	-0.26	-0.24
	Speed up	1	2	2.82	1.5	1.83	1	2.01	3.61	1.67	2.26	1	1.96	3.96	1.7	2.49	1	1.96	4.24	1.75	2.65
Waterfall	PSNR(dB)	32.19	-2.36	-4.23	-0.89	-0.26	34.83	-1.06	-3.73	-0.39	-0.16	36.08	-0.6	-3.07	-0.21	-0.78	36.88	-0.39	-2.38	-0.13	-0.68
	Speed up	1	1.95	3.36	1.73	2.29	1	1.89	4.1	1.81	2.7	1	1.9	4.39	1.85	3.29	1	1.92	4.39	1.85	3.44
Foreman	PSNR(dB)	31.15	-8.62	-4.33	-1.83	-1.13	34.5	-2.92	-3.05	-1	-0.67	36.2	-1.67	-2.5	-0.67	-0.79	37.36	-1.17	-2.13	-0.48	-0.75
	Speed up	1	2.11	2.88	1.55	1.99	1	2.03	3.5	1.66	2.36	1	2.02	3.8	1.71	2.71	1	2.01	3.96	1.7	2.84
Flower	PSNR(dB)	23.65	-3.05	-3.36	-0.43	-0.27	26.23	-0.98	-2.79	-0.2	-0.2	27.74	-0.67	-2.32	-0.17	-0.22	28.87	-0.49	-1.99	-0.14	-0.24
	Speed up	1	1.94	2.91	1.53	1.94	1	1.82	3.4	1.6	2.25	1	1.81	3.56	1.63	2.38	1	1.8	3.68	1.65	2.44
Tempete	PSNR(dB)	26.41	-6.24	-4.54	-0.95	-0.48	29.17	-1.95	-3.76	-0.41	-0.25	30.68	-1.12	-3.1	-0.25	-0.26	31.73	-0.77	-2.58	-0.14	-0.23
	Speed up	1	2.08	2.78	1.58	1.98	1	1.95	3.55	1.72	2.35	1	1.93	3.94	1.76	2.61	1	1.96	4.15	1.81	2.71
Average PSNR(dB)		29.65	-4.78	-3.93	-0.97	-0.5	32.66	-1.68	-3.22	-0.49	-0.32	34.17	-0.96	-2.65	-0.31	-0.53	35.21	-0.67	-2.2	-0.22	-0.5
Average speed up		1	1.99	2.94	1.58	2.04	1	1.93	3.6	1.69	2.42	1	1.91	3.93	1.72	2.76	1	1.93	4.03	1.75	2.89

Fig. 5. PSNR and speed-up results for various QCIF and CIF videos and bitrates. Numerous methods are compared against the cascade method: MV+MS [9], statistics with and without refinement [8], and proposed. The PSNR rows show differences from the PSNR values of the cascade method. The speed-up rows are defined as $T_{cascade}/T_{method}$, with T representing the transcoding time.

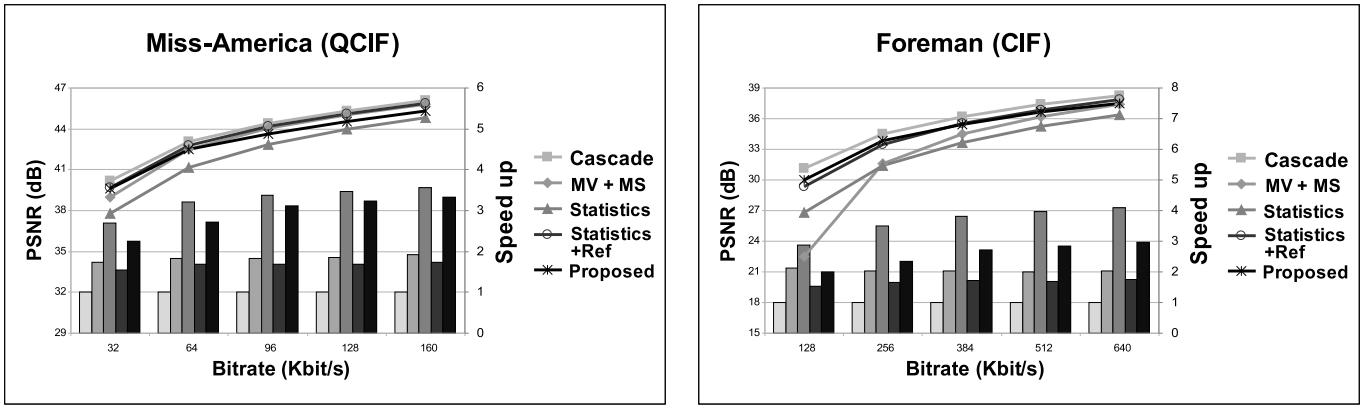

Fig. 6. PSNR and speed-up results for the Miss America (QCIF) and Foreman (CIF) videos at different bitrates. Speed-ups defined as $T_{cascade}/T_{method}$, with T representing the transcoding time. The methods are presented in the following order for speed-up results: cascade, MV+MS [9], statistics with and without refinement [8], and proposed.