Fast mode decision algorithm for spatial resolutions
down-scaling transcoding to H.264

BU Jia-jun, MO Lin-jian†, CHEN Chun, YANG Zhi
(School of Computer Science, Zhejiang University, Hangzhou 310027, China)
†E-mail: molin@zju.edu.cn
Received Dec. 15, 2005; revision accepted Feb. 17, 2006

Abstract: A fast mode decision algorithm is proposed in this paper to accelerate the process of transcoding videos into H.264 with arbitrary rate spatial resolution down-scaling. The proposed algorithm consists of three steps. First, an early-stop technique is introduced to determine the 16\times16-mode blocks, which take up about 70% of all the macroblocks; then, a bottom-up merging process is performed to determine the mode of rest non-early-stopped blocks; and then, we adopt half-pixel motion estimation to further refine the acquired predictive motion vectors. In order to obtain the predictive motion vectors for early-stop and merging processes, we propose a motion vector composition scheme, which can reuse the information in the input pre-encoded videos to handle the spatial resolution down-scaling. Experimental results showed that our algorithm is about four times faster than the Cascaded-Decoder-Encoder method and has negligible PSNR drop and little bit rate increase.

Key words: Transcoding, H.264, Mode decision, Motion composition
doi:10.1631/jzus.2006.AS0070

INTRODUCTION

The forthcoming multimedia telecommunication services are expected to use more efficient pre-encoded video for storage and transmission. H.264, as the emerging video coding standard, achieves much higher coding efficiency than previous standards. Thus, in some of the real-time video communication scenarios, it is desirable to rapidly transcode videos from other standards to H.264 without much video quality reduction.

In many applications, the pre-encoded video needs to be transcoded to lower spatial resolution, such as fitting for various display screen sizes on hand-held devices. Thus, To-H.264 transcoder needs to handle spatial resolution down-scaling. However, different from previous video coding standard, H.264 allows seven block-sizes motion estimation, which makes the transcoding task more challenging.

Many works addressed the spatial resolution reduction transcoding (Xin et al., 2002; Shanableh and Ghanbari, 2000; Takahashi et al., 2001). The motion information from input pre-encoded video can be reused during H.264 motion estimation processing, to reduce the computational complexity. The works by (Xin et al., 2002; Takahashi et al., 2001) describe techniques for handling motion vector (MV) re-use in transcoding to previous standards, such as MPEG2 and MPEG4. However, many new features are introduced that make the transcoding to H.264 very different. With the quad-tree structure and various block-sizes, the motion information re-use techniques should be re-designed while transcoding to H.264.

Seven block-sizes mode decision for INTER frame is firstly introduced by H.264. Chang et al.(2004) pointed out that variable block size motion estimation and mode decision take up about 70% computational resource in the encoding process. It is

* Project supported by the National Natural Science Foundation of China (No. 60573176), the Key Technologies R & D Program of Zhejiang Province (Nos. 2005C23047 and 2004C11052), China
necessary to explore fast algorithm. Some researchers tried to reduce the search range, especially with the early-stop technique which stops the process early when some conditions are satisfied. For instance, early-stop on 16×16-mode (MODE 0 or 1 of INTER Block in H.264) technique directly sets the block to 16×16-mode according to the texture or sub-block MV information (Kucukgoz and Sun, 2004). Others address this problem through a different approach named information re-use. Top-down splitting (Zhou et al., 2004) and bottom-up merging (Zhou et al., 2004; Tu et al., 2003) methods are introduced to reuse the up/low-layer MV in the quad-tree structure. These techniques can also be applied for To-H.264 transcoder; moreover, reusing the MVs of input pre-encoded videos will achieve better performance, because the input pre-encoded motion information predicts the MV better than that obtained from the spatially or temporally neighboring macroblocks.

This work is aimed at designing a fast mode decision algorithm in transcoding to H.264 with arbitrary-rate spatial resolution down-scaling. The motion estimation and mode decision process in H.264 are combined to avoid unnecessary time consumption. A novel predictive motion vector (PMV) composing scheme is suggested to handle the arbitrary-block-size multiple-to-one macroblocks merging. The proposed algorithm consists of three steps. First, we try to pre-detect the 16×16 mode blocks (SKIP, 16×16 Inter) which almost take up 70% of them all; then, a bottom-up merging processing is performed to handle the non-decided blocks in Step 1; and then, the obtained MV will be refined with half-pixel precision search. Experimental results showed that our algorithm acts about four times faster than the Cascaded-Decoder-Encoder method and has negligible PSNR drop and little bit rate increase.

The rest of the paper is organized as follows. Section 2 describes how to merge multiple MVs into one. The details of the fast mode decision algorithm are described in Section 3, and Section 4 presents the experimental results. We conclude this paper in Section 5.

PROPOSED MV COMPOSITION SCHEME

For transcoding into lower spatial resolution pictures, a new MV is to be calculated from a set of input MVs of the pre-encoded higher spatial resolution sequence. The acquired MV will act as PMV for the encoding process of the lower spatial resolution sequence. For instance, transcoding a bitstream in CIF format into QCIF format requires calculating a new MV from four input MVs. This corresponds to transcoding four macroblocks into one macroblock. The proposed MV composition scheme in this section can predict the PMVs for various size blocks in H.264 with spatial resolution down-scaling.

To efficiently reuse the input motion information for better prediction, various methods have been proposed, such as Random, Mean, Weighted Average, Weighted Median, DCmax, etc. (Ahmad et al., 2005). Here Weighted Median method yields good performance. Referring to Weighted Median method, we propose an MV composition scheme, which is based on the block size and covered area corresponding to the input MVs, as shown in Fig.1. The details are as follow:

\[
MV_{pred} \in \{ MV_i \},
\]

\[
WAMV = \left( \sum_i (CBSi \times MV_i) \right) / \left( \sum_i CBS_i \right), \]

\[
|MV_{pred} - WAMV| \leq |MV_i - WAMV|, \quad i=1, 2, \ldots, N,
\]

where \(WAMV\) is weighted average MV and \(MV_{pred}\) is the target PMV for the macroblock in down-scaling picture. \(N\) is the number of covered blocks. \(CBS_{i}\) and \(MV_{i}\) denote the block size in the covered area in the original picture and the corresponding MV, respectively. For instance, in Fig.1, \(CBS_{A}\) is 8×8=64, but \(CBS_{B}\) equals 16×8=128.

![Fig.1 Merging multiple MVs to one. MV on block B has bigger impact on the target predictive MV than that of block A. Slim continuous lines mean the covered area (The source picture consists of various size-blocks)](image)

It should be mentioned that if the input MVs are interlaced but the target MV is progressive; or the input MVs are progressive but target MV is interlaced,
some adjustments should be applied. This paper focuses on the former which are widely supported by actual applications.

Let \((MV_x, MV_y)\) be the original MV, \((MV'_x, MV'_y)\) be the adjusted MV; TF means top field and BF means bottom field.

(1) Field-to-frame, as shown in Fig.2:

\[
\begin{align*}
(MV'_x, MV'_y), \quad MB_{cu} & \in TF, MB_{ref} \in TF, \\
(MV'_x, MV'_y + 1), \quad MB_{cu} & \in TF, MB_{ref} \in BF, \\
(MV'_x, MV'_y - 1), \quad MB_{cu} & \in BF, MB_{ref} \in TF, \\
(MV'_x, MV'_y), \quad MB_{cu} & \in BF, MB_{ref} \in BF.
\end{align*}
\]

(2) Frame-to-field:

\[
\begin{align*}
(MV'_x, MV'_y), \quad MB_{cu} & \in TF, MB_{ref} \in TF, \\
(MV'_x, MV'_y - 1), \quad MB_{cu} & \in TF, MB_{ref} \in BF, \\
(MV'_x, MV'_y + 1), \quad MB_{cu} & \in BF, MB_{ref} \in TF, \\
(MV'_x, MV'_y), \quad MB_{cu} & \in BF, MB_{ref} \in BF.
\end{align*}
\]

PROPOSED MODE DECISION ALGORITHM

There is high correlativity between the up-layer and low-layer blocks in the quad-tree structure of the seven block types in H.264. Top-down splitting and bottom-up merging methods are introduced to reuse the up/lower-layer MV. Meanwhile, as in (Kucukgoz and Sun, 2004), 16×16-mode blocks usually take up about 70% of them all in H.264 encoding. If these 16×16-mode blocks are captured in the beginning of the mode decision, remaining procedures will be skipped and significant computation reduction will be achieved. Employing both the bottom-up merging and early-stop techniques, we proposed a novel fast predictive mode decision algorithm consisting of four steps:

Step 1: Prepare the PMVs
According to the scheme proposed in the previous section, we can get the PMVs for all size blocks in a macroblock, which are one 16×16, two 16×8 and two 8×16, four 8×8, eight 8×4 and eight 4×8, sixteen 4×4 size block MVs.

It should be mentioned that these PMVs are on-demand-calculated. For example, if the process is early-stop at 16×16 mode (proposed in Step 2), only one 16×16 size MV will be considered.

Step 2: Early-stop detection on 16×16-mode blocks
For the 16×16 size block, compute the SAD with the PMV. If the SAD value is less than a threshold, \(T\), the current macroblock will be decided to be 16×16-mode and the corresponding PMV acts as final MV without performing the following steps. Otherwise, the current macroblock will be regarded as a non-decided block, and go to Step 3.

According to the simulation results based on many sequences with \(QP=28, 32, 36, 40\), we find that usually threshold \(T\) is in \([450, 1000]\). However, different sequences, even different frames within the same sequence lead to different thresholds. As a result, an adaptive threshold for each frame is introduced to handle this problem. It can be described as follow:

(1) Initialization: \(T=AMAD=0, N=0\), where \(N\) denotes accumulated number of macroblock used mode-16×16.

(2) Repeat: For each macroblock, find the minimal SAD, \(SAD_{min}\), with the PMV. If \(SAD_{min}<T\), choose mode-16×16 as final block type and update \(T\) by the following steps:

\[
AMAD=AMAD+SAD_{min}, \quad N=N+1,
\]

\[
T=w[(AMAD+\alpha)/(N+\beta)],
\]

where \(w\) is a weight to adjust the threshold while different \(QP\) is used, \(\alpha\) and \(\beta\) are introduced to prevent a sharp alteration of \(T\) which may be caused by the sudden appearing of a small or large SAD. Usually, smaller \(QP\) causes smaller block types, such as 4×4
mode. Therefore, we set \( w \) to be 1.1, 1.2, 1.3, 1.4 for \( QP=28, 32, 36, 40 \) respectively. \( w \) should be larger than 1, otherwise \( T \) will never increase. \( \alpha, \beta \) can be regarded as the initial threshold because the initial AMAD, \( N \) are 0. Fig.3 shows a trade-off between quality and efficiency. We set \( \alpha=10000 \) and \( \beta=10 \).

\[
\alpha = \text{AMAD}, \quad \beta = 10. \tag{5}
\]

Experimental results (Fig.4) show that with our proposed pre-detection of 16\( \times \)16-mode blocks are captured.

Figure 4 Average performance of 16\( \times \)16-mode capture algorithm

Step 3: Bottom-up merging

Based on the achieved 4\( \times \)4-block MVs, a merging procedure is taken to finish the mode decision procedure. The principle is simple but efficient: If the distance of neighboring MVs, pred1, pred2, is less than a threshold, \( T_{\text{pred}} \), which means they are well aligned, merging will happen. First of all, let us define the distance between two MVs as follows:

\[
\text{Dist}_{\text{pred1,pred2}} = \max\{\|\text{MV}_1, \text{MV}_2\|, \|\text{MV}_1, \text{MV}_2\|, \|\text{MV}_1, \text{MV}_2\|\}, \tag{5}
\]

where \( \text{MV}_1, \text{MV}_1, \text{MV}_2, \) and \( \text{MV}_2 \), denote the \( x \) and \( y \) components of vectors \( \text{MV}_1 \) and \( \text{MV}_2 \) respectively.

Fig.6 depicts the flowchart of the merging from 4\( \times \)4-blocks to 8\( \times \)8-blocks. With the PMVs for all size blocks estimated by Step 1, the bottom-up merging process can be described as follows. For the vertical and horizontal merging in Fig.5, while \( \text{Dist}_{\text{pred1}, \text{pred2}} \leq T_{\text{pred}} \), merge the corresponding two blocks to an up-layer block; in particular, if \( \text{Dist}_{\text{pred1}, \text{pred2}} = 0 \), let the MV of merged block be \( \text{MV}_{\text{pred1}} \) instead of the PMV taken from the proposed scheme in Step 1. If \( \text{Dist}_{\text{pred1}, \text{pred2}} > T_{\text{pred}} \) in both of the vertical and horizontal merging processes, stop the merging and choose the current modes as the final block types (Status 1 in Fig.6); if both vertical and horizontal merging happen, these four 4\( \times \)4 blocks will be merged into an 8\( \times \)8 block without any additional checking (Status 3 in Fig.6).

The merging from 8\( \times \)8-blocks to 16\( \times \)16-blocks is similar, only the decide-to-be-merged 8\( \times \)8 blocks will participate in the following merging.

In order to determine the threshold \( T_{\text{pred}} \), statistical analysis is conducted. Statistical analysis (Table 1) showed that if \( \text{Dist}_{\text{pred1}, \text{pred2}} \leq 3 \), it is highly-probable that \( MD < 1 \) pixel (\( MD \) is defined in Table 1), otherwise the probability is very low. Hence, we let \( T_{\text{pred}} = 3 \), the unit is a pixel.

Table 1 Cumulative probability distribution of \( MD=\text{Dist}_{\text{predictor,actualMV}} \) for merged blocks by full search with \( QP=28 \) and search range=[−16, 15] where \( \text{Dist}_{\text{pred1}, \text{pred2}} \) denotes the distance of two MV predictors

<table>
<thead>
<tr>
<th>Sequence</th>
<th>( \text{Dist}_{\text{pred1}, \text{pred2}} \leq 3 )</th>
<th>( \text{Dist}_{\text{pred1}, \text{pred2}} &gt; 3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( MD=0 )</td>
<td>( MD \leq 1 )</td>
</tr>
<tr>
<td>News</td>
<td>89.38%</td>
<td>97.30%</td>
</tr>
<tr>
<td>M &amp; D</td>
<td>69.73%</td>
<td>89.34%</td>
</tr>
<tr>
<td>Foreman</td>
<td>63.71%</td>
<td>91.47%</td>
</tr>
<tr>
<td>M &amp; C</td>
<td>77.87%</td>
<td>97.01%</td>
</tr>
<tr>
<td>Coastguard</td>
<td>79.18%</td>
<td>97.01%</td>
</tr>
</tbody>
</table>
Step 4: Motion refinement
A half-pixel refinement is done on the merging results in Step 3. As in (Xin et al., 2002), it is enough to achieve a good quality.

EXPERIMENTAL RESULTS

In order to evaluate the performance of the proposed algorithm, we calculated the PSNR between each down-sampled sequence and corresponding original decoded video down-scaled to the same resolution. Moreover, the transcoding speed performance of Cascaded-Decoder-Encoder method and our proposed algorithm will also be tabulated in Table 2.

As the sequences to be transcoded, the test sequences are pre-encoded with an open source MPEG-4 encoder (XVID) with 4CIF (704×576) resolution using the simple profile on 30 fps frame rate, only the first 50 frames are encoded. These encoded sequences will be transcoded into H.264 baseline profile by JM8.6 with different resolutions, SQCIF (128×96), QCIF (176×144) and CIF (352×288). The JM8.6 encoder setting is as follow: frame rate is 30 fps, search range is 16 and QP=28.

In Table 2, the testing sequences can be classified into three categories: video-conferencing-like sequences such as News and Akiyo which contain smooth motions with static backgrounds; sequences with global motions such as Coastguard; and sequences with zooming, distorted large motion such as Foreman. We achieved the performance about four times faster than the Cascaded-Decoder-Encoder transcoder (Fast Full Search) with average 0.09 PSNR drop and 2% bit rate increase.

Table 2  Comparison of PSNR, bitrate and transcoding efficiency performance between Cascaded and proposed algorithm. The source sequences are in 4CIF (704×576) format (the units of PSNR and Bit rate are dB, kbps respectively)

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Down to</th>
<th>Cascaded PSNR</th>
<th>Bit rate</th>
<th>Cascaded PSNR</th>
<th>Bit rate</th>
<th>Proposed PSNR</th>
<th>Bit rate</th>
<th>Speed up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Akiyo</td>
<td>128×96</td>
<td>37.15</td>
<td>27.93</td>
<td>37.08</td>
<td>27.63</td>
<td>4.23</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>176×144</td>
<td>38.36</td>
<td>34.59</td>
<td>38.29</td>
<td>34.20</td>
<td>4.54</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>352×288</td>
<td>39.79</td>
<td>102.95</td>
<td>39.71</td>
<td>102.93</td>
<td>5.01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>News</td>
<td>128×96</td>
<td>35.75</td>
<td>63.03</td>
<td>35.64</td>
<td>63.41</td>
<td>3.71</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>176×144</td>
<td>36.72</td>
<td>71.52</td>
<td>36.71</td>
<td>71.38</td>
<td>3.97</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>352×288</td>
<td>38.18</td>
<td>202.99</td>
<td>38.09</td>
<td>206.99</td>
<td>4.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Foreman</td>
<td>128×96</td>
<td>36.26</td>
<td>44.54</td>
<td>36.14</td>
<td>45.13</td>
<td>3.84</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>176×144</td>
<td>35.47</td>
<td>112.98</td>
<td>35.39</td>
<td>114.50</td>
<td>3.62</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>352×288</td>
<td>36.61</td>
<td>381.55</td>
<td>36.55</td>
<td>399.76</td>
<td>3.89</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coastguard</td>
<td>128×96</td>
<td>33.05</td>
<td>119.83</td>
<td>33.01</td>
<td>121.04</td>
<td>2.79</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>176×144</td>
<td>33.82</td>
<td>481.21</td>
<td>33.72</td>
<td>483.82</td>
<td>3.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>352×288</td>
<td>34.33</td>
<td>1335.21</td>
<td>34.30</td>
<td>1337.84</td>
<td>3.39</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

with global motions such as Coastguard; and sequences with zooming, distorted large motion such as Foreman. We achieved the performance about four times faster than the Cascaded-Decoder-Encoder transcoder (Fast Full Search) with average 0.09 PSNR drop and 2% bit rate increase.

Fig.7 shows the bit rate comparison between proposed and cascaded algorithm on Foreman with the first 30 frames and QP=28. Figs.8 and 9 give the PSNR comparison. As Fig.9 shows, we find that the proposed works well on big QP values, such as QP=40.
Therefore, it can be concluded that our proposed algorithm can adapt to transcode with most kinds of down-sampled resolutions and motion scenarios.

CONCLUSION

In this paper, we proposed a fast mode decision algorithm for transcoding to H.264 with arbitrary rate spatial resolution down-scaling. With a PMV composition scheme to reuse the motion information in the input pre-encoded sequences, an early-stop on 16×16-block technique with adaptive threshold and a bottom-up merging processing are introduced to speed up the mode decision. Experimental results showed that the proposed algorithm can achieve a good trade-off between computational complexity and transcoding quality. The computational complexity can be reduced to about 25% of Cascaded-Decoder-Encoder method with Fast Full Search algorithm. PSNR drop and bit rate increase are 0.09 and 2%, respectively.

References


