MPEG Complexity Reduction by Scene Adaptive Motion Estimation

Shigeaki Yamaoka
Dept Electronics Eng. and Computer Science
Fukuoka University
Fukuoka 814-0180, Japan
yamaoka@vlab.tl.fukuoka-u.ac.jp

Vasily G. Moshnyaga
Dept Electronics Eng. and Computer Science
Fukuoka University
Fukuoka 814-0180, Japan
vasily@fukuoka-u.ac.jp

Abstract—This paper presents a new technique to reduce computations involved in MPEG motion estimation. Unlike existing schemes which apply the same computational pattern for all frames, we propose to link the motion estimation to the picture content. When a scene change is detected we stop processing and thus omit redundant calculations. Experiments on different benchmarks show that the technique can save up to 1/5 of the total number of motion estimation computations with almost no picture degradation.

I. INTRODUCTION

A. Motivation

In MPEG video coding, frames in a sequence are coded using three different algorithms [1]. Intra images (or I-frames) are self-contained and are coded using a DCT-based technique that gives the highest quality but the lowest compression ratio. Predicted images or (P-frames) are coded from a previous (I or P) frame using motion compensated prediction. Bidirectional images (or B-frames) are coded using two reference frames, a past and a future frame (which can be I or P frames). Because bidirectional coding provides the highest amount of compression, most of MPEG2 encoders incorporate two or three B-frames among the reference frames to achieve the highest compression ratio.

Motion estimation (ME) is a core task employed in coding of P and B frames to reduce temporal redundancy. Based on the block-matching, the task involves finding a candidate macroblock in a specified search area of the reference frame that is most similar to the current macroblock in the current frame. The difference in coordinates of matching blocks defines the motion vector, $V$, which is then transmitted jointly with prediction residual to the decoder. For each macroblock of P-frame, the ME produces a forward motion vector, $V_F$, only, while for the B-frame macroblocks, a pair of motion vectors (the forward vector, $V_F$, and the backward vector, $V_B$) are computed, as shown in Fig.1. If the number of bidirectionally predicted frames in the group of pictures is large, the computational complexity of the task becomes enormous. For example, at the MPEG2 MP@ML (30fps, two B-frames between the reference frames), the ME demands over 180 GOPS for the full search and 11 GOPS for the 3-step search, taking thus over 98% and 84% of the total encoding computations, respectively [2]. Therefore, methods capable of reducing computations of bidirectional ME are of a great interest.

B. Related Research

Many efforts have been put to lower computational load of ME. The methods proposed can be classified in three categories depending on level they focus on: pixel, macroblock, and frame. At the pixel level, the ME complexity is approached by reducing the number of bits in pixel representation and consequently in matching criterion [1-2]. Binary matching [3,4] is a typical technique exploited at this level. At the macroblock level, optimizations affect either the number of pixels used for macroblock representation (e.g. sub-sampling) or the number of locations searched (i.e. search-area). In turn, the algorithms can search only a small (fixed) subset of candidate macroblocks (e.g. [5]-[8]) or adjust the number of locations to the picture content by stopping the search when selected picture variation parameter overcomes a given threshold [9-17]. Our work differs from these techniques in that the ME complexity is optimized at the frame level. Namely, we focus on reducing redundant computations which occur when a scene change takes place by adaptively adjusting the motion estimation operation for the newly changed frame.

In video stream, scene change can happen at any time and can be either gradual or abrupt. While the former deals with picture transition that takes several frames to complete due...
to camera zooming, panning, tilting, fading, etc., the latter defines instant picture transformation. Traditional MPEG coders that use a fixed (e.g. IBBPBBPBB..) coding pattern usually become inefficient in the event of scene change. As bidirectional motion estimation fails, they additionally encode macroblocks using intra-coding and so require many extra computations.

Scene adaptive video coding has received significant attention recently. Cha, et al. [16] apply frame skipping (dropping) for adapting video to available bandwidth. Lee, et al. [17] use a prediction-based algorithm to control picture target bit-rate and compensate distortion at a sudden scene change. Kodiraka [18] replaces the first P-frame after a scene change with an I-type frame to increase coding quality. Farin, et al. [19] proposed to trade quality of the first B-frame after the scene transition to acquire target bit-ratio when rescheduling the GOP. The type of picture after the scene transition is decided based on measure of motion magnitude [20],[21] or co-linearity of displacements in successive frames [22].

In our work we also use adaptive frame skipping. However, in contrast to the related research which focuses on performance optimization in terms of the bit-rate, the speed and the quality of decoded video we target computational complexity.

C. Contribution

In this paper we present a new technique to reduce computations involved in MPEG motion estimation during abrupt scene changes. Unlike existing schemes, the technique adjusts the motion estimation pattern to the picture content in order to omit redundant calculations when abrupt scene change is detected. Experiments on different benchmarks show that the technique can save up to 1/5 of the total number of motion estimation computations with almost no picture degradation.

The paper is organized as follows. The next section describes the proposed technique. Section 3 analyzes the performance. Section 4 summarizes the results and outlines our future work.

II. THE PROPOSED TECHNIQUE

The technique we propose is based on observation that current motion estimation algorithms involve a large number of redundant computations at a scene change. The reason is that by using a fixed control pattern they try to predict the "unpredictable" content of a new picture frame \( F_t \), the prior reference frame \( F_{t-k} \), when a scene change occurs. If the frame \( F_t \) is coded as an I-frame there is no problem. However, if \( F_t \) is coded as a P- or a B-frame, the motion estimation causes unnecessary calculations. For example, consider Fig.2 which depicts coding relations within a sub-GOP. If a scene change takes place at the P-frame, as shown in Fig.2(a), the conventional motion estimation computes three redundant motion vectors for each macroblock in the frame: one during forward prediction and two during backward prediction. (In

to camera zooming, panning, tilting, fading, etc., the latter defines instant picture transformation. Traditional MPEG coders that use a fixed (e.g. IBBPBBPBB..) coding pattern usually become inefficient in the event of scene change. As bidirectional motion estimation fails, they additionally encode macroblocks using intra-coding and so require many extra computations.

Scene adaptive video coding has received significant attention recently. Cha, et al. [16] apply frame skipping (dropping) for adapting video to available bandwidth. Lee, et al. [17] use a prediction-based algorithm to control picture target bit-rate and compensate distortion at a sudden scene change. Kodiraka [18] replaces the first P-frame after a scene change with an I-type frame to increase coding quality. Farin, et al. [19] proposed to trade quality of the first B-frame after the scene transition to acquire target bit-ratio when rescheduling the GOP. The type of picture after the scene transition is decided based on measure of motion magnitude [20],[21] or co-linearity of displacements in successive frames [22].

In our work we also use adaptive frame skipping. However, in contrast to the related research which focuses on performance optimization in terms of the bit-rate, the speed and the quality of decoded video we target computational complexity.

C. Contribution

In this paper we present a new technique to reduce computations involved in MPEG motion estimation during abrupt scene changes. Unlike existing schemes, the technique adjusts the motion estimation pattern to the picture content in order to omit redundant calculations when abrupt scene change is detected. Experiments on different benchmarks show that the technique can save up to 1/5 of the total number of motion estimation computations with almost no picture degradation.

The paper is organized as follows. The next section describes the proposed technique. Section 3 analyzes the performance. Section 4 summarizes the results and outlines our future work.

II. THE PROPOSED TECHNIQUE

The technique we propose is based on observation that current motion estimation algorithms involve a large number of redundant computations at a scene change. The reason is that by using a fixed control pattern they try to predict the “unpredictable” content of a new picture frame \( F_t \), the prior reference frame \( F_{t-k} \), when a scene change occurs. If the frame \( F_t \) is coded as an I-frame there is no problem. However, if \( F_t \) is coded as a P- or a B-frame, the motion estimation causes unnecessary calculations. For example, consider Fig.2 which depicts coding relations within a sub-GOP. If a scene change takes place at the P-frame, as shown in Fig.2(a), the conventional motion estimation computes three redundant motion vectors for each macroblock in the frame: one during forward prediction and two during backward prediction. (In

the figure these redundant vectors are marked by crosses). If a scene change is detected at the first B frame (see Fig.2(b)) all forward predictions become redundant. Similarly, if a scene change occurs at the second B frame (Fig.2(c)), two forward predictions and one backward prediction are redundant. We propose to omit these redundant predictions based on frame classification.

The idea is following: we test statistical features for P/B frames to determine a scene change. If a scene change is detected, we terminate current GOP, by coding the “future” P-frame as an I-picture. If the current frame, \( F_t \), is a P-frame, we stop the forward prediction of \( F_t \) and the backward predictions of the B-frames, placed in between the reference frames \( F_{t-k} \) and \( F_{t-k+n} \), as shown in the left part of Fig.2(a). If \( F_t \) is the first B-type picture in the sub-GOP, then we perform only backward prediction for the current frame, \( F_t \), as well as all the consecutive B-frames \( F_{t+1}, F_{t+2},.. \), placed in between the reference frames \( F_{t-k} \) and \( F_{t-k+n} \). Else the forward prediction of \( F_t \) is performed.

To detect a scene change, a variety of measures can be used: likelihood ratio [23], histogram [24], mean absolute frame difference and its derivative [25], displaced frame difference [26], modified sign of frame [27], etc. To minimize the total number of computations, we use simple yet efficient criterion that counts the number of changed blocks in the frame. Namely, we compare each macroblock \( X_i \) of the frame \( F_t \) with corresponding macroblock \( X_{i,k} \) of the reference frame \( F_{t,k} \), calculating sum of absolute differences of their pixels:

\[
SAD_{i,j} = \sum_{k=1}^{n} | x_{i,j} - x_{i,k,j} | ,
\]

where \( x_{i,j} \in X_t \) and \( x_{i,j} \in X_{t,k} \) are pixels of \( X_t \) and \( X_{t,k} \) respectively; \( k \) is the integer; \( t \) is the frame number; \( i=1..N, j=1..N \). During motion estimation, we count macroblocks whose SAD value is larger than a given threshold, \( T_1 \). If the count exceeds a half of the total number of macroblocks in the frame, the correlation between the frames \( F_t \) and \( F_{t,k} \) is assumed to be very low. In this case, we detect a scene
For the sake of simplicity, we assume here that the sub-GOP contains two B-pictures in between reference frames. Even though these B-pictures precede the “future” P-frame in time, their coding is done after the P-frame, i.e. in the IPBB order. For each P and B frame, the algorithm counts macroblocks whose SAD value calculated in regards to the prior reference frame is larger than $T_1$. If the number of such macroblocks is less than half of the total number of macroblocks in the frame (threshold $T_2$), it runs conventional motion estimation. Otherwise, it detects a scene change ($s=1$) and transforms the coding pattern to stop redundant computations. The $k=0$ characterizes the first B-frame in the sub-GOP; while $k>0$ indicates that the scene change occurs at the second B-frame or at the P-frame. In the latter case, the algorithm performs only forward prediction for the B-frames, while omitting all backward predictions for the B-frames. As one can see, the algorithm does neither require extra memory nor large computational overhead.

III. EXPERIMENTAL RESULTS

We tested the proposed technique in the modified MPEG2 video codec environment on several video streams. In each stream, the GOP contained twelve frames; the distance between the past and the future reference frames was two. All the streams have been evaluated for the macroblock size of 16x16 pixels and the search region of ±16 pixels. The performance of proposed technique has been compared to conventional bidirectional full-search block matching algorithm. PSNR of the luminance has been considered to measure the objective quality of encoded video.

Table 1 shows the results in terms of computational complexity reduction, PSNR, and picture error normalized in comparison to conventional motion estimation scheme [29]. Note, the values in column 2 include additional computations required for detecting scene change based on criterion (1). We observe that efficiency of the proposed technique varies among sequences. The largest computational savings (up to 20% of the total number of computations) are achieved on sequences which exhibit scene transition count on the threshold $T_1$. For this purpose, we used a set of benchmarks which exhibit a few (or even none) scene changes (Foreman, Miss America, Salesman, Carphone). Fig.4 shows the results. Based on these results, we found that $T_1=5500$ provided correct scene change detection for all the sequences. Therefore, this value has been selected for the threshold.

Then, we evaluated efficiency of the proposed technique on five standard video sequences: Foreman (176x144, 298), Salesman (252x288, 300), Tennis (252x240, 110), Football (352x288, 298) and Mobile and Calendar (352x240, 447). All these sequences except Salesman exhibit scene changes, though the number and types of them vary significantly.

First, we empirically investigated the dependency of the scene transition count on the threshold $T_1$. For this purpose, we used a set of benchmarks which exhibit a few (or even none) scene changes (Foreman, Miss America, Salesman, Carphone). Fig.4 shows the results. Based on these results, we found that $T_1=5500$ provided correct scene change detection for all the sequences. Therefore, this value has been selected for the threshold.

Then, we evaluated efficiency of the proposed technique on five standard video sequences: Foreman (176x144, 298), Salesman (252x288, 300), Tennis (252x240, 110), Football (352x288, 298) and Mobile and Calendar (352x240, 447). All these sequences except Salesman exhibit scene changes, though the number and types of them vary significantly.

Table 1 shows the results in terms of computational complexity reduction, PSNR, and picture error normalized in comparison to conventional motion estimation scheme [29]. Note, the values in column 2 include additional computations required for detecting scene change based on criterion (1). We observe that efficiency of the proposed technique varies among sequences. The largest computational savings (up to 20% of the total number of computations) are achieved on sequences which exhibit scene changes.
Proposed with other methods and on other streams is also considered. To cover gradual scene transitions we are now considering frame insertion at the scene change, it demands higher bit rates than in average this bit-rate increase is not so high (less than 6% for the Carphone sequence). We also notice that the technique does not affect the PSNR much, providing quite a small picture error.

Fig.5 illustrates the PSNR variation with the picture frames on example of the Carphone sequence. We see that our technique over performs the conventional technique when scene change occurs (frame 52). The average PSNR gain for this frame is considerable (0.56dB).

Fig.6 shows variation of number of bits required to code the frames in the proposed scheme. Due to extra I-frame inserted at the scene change, it demands higher bit rates while in average this bit-rate increase is not so high (less than 6% for the Carphone sequence).

IV. CONCLUSION

We have presented a technique to reduce the amount of computations in bidirectional block-matching ME. As experiments showed, the technique allows us to avoid unnecessary ME operations for almost 1/5 of macroblocks without a large impact on picture quality and bit-rate. The current work, however, has been limited to abrupt scene changes. To cover gradual scene transitions we are now studying other criteria, such as [27]. Extending comparisons with other methods and on other streams is also considered.

REFERENCES