A New Spatio-Temporal Predictor for Motion Estimation in H.264 Video Coding

Spyridon K. Kapotas and Athanassios N. Skodras
Digital Systems and Media Computing Laboratory, School of Science and Technology
Hellenic Open University, Patras, Greece
s.kapotas@eap.gr, skodras@eap.gr

Abstract

The new JVT H.264 video coding standard is becoming the dominant standard, mainly due to its high coding efficiency. Among all modules in the encoder, motion estimation plays a key role since it can significantly affect the output quality of an encoded sequence. However, motion estimation requires a significant part of the encoding time. Thus, various fast motion estimation algorithms have been proposed. The H.264 reference encoder has currently adopted three fast motion estimation algorithms. These algorithms aim at reducing the search area by predicting the search points using suitable predictors. In this paper we propose a new spatio-temporal predictor and we study the effectiveness of different predictors in order to find out whether these predictors truly contribute to the decrease of motion estimation time without affecting video quality and adding undesirable complexity to the encoder.

Keywords— video coding, motion estimation, H.264

1. Introduction

The recently established H.264/AVC is the newest video coding standard [1]. H.264 has various motion-compensation units in sizes of 16×16, 16×8, 8×16, 8×8, and sub8×8. For sub8×8, there are further four sub-partitions of sub8×8, sub8×4, sub4×8, and sub4×4. Such wide block choices greatly improve coding efficiency but at the cost of largely increased motion estimation time. The computational complexity becomes even higher when larger search ranges, bi-directional and/or multiple reference frames are used. Such high computational complexity is often a bottle-neck for real-time conversational applications.

Block-matching is the process of seeking the best-matched block from the reference frame within certain search area (W). The matching criterion is usually called block distortion measure (BDM) such as sum of absolute (transformed) differences (SAD) or Lagrange cost. There have been many fast motion estimation (FME) techniques proposed in the literature [2-7]. Two popular approaches are chosen to reduce computation in block matching motion estimation. The first approach reduces the number of candidate blocks in the search window (fast searching techniques). Those algorithms usually show good speed gain but have relatively larger rate-distortion (R-D) performance degradation. The second approach reduces the complexity of SAD computation (fast matching techniques). Those algorithms often achieve good coding efficiency but have limited speed up gain. Other techniques include predicted spatial-temporal search, adaptive early termination, and dynamic search range adjustment. It is possible to combine several of the above techniques to form a hybrid search method. For example, PMVFAST [8] and UMHexagonS [9] utilize prediction, diamond search, hexagon search, partial distortion, and adaptive early termination. They are proven to be more robust than a single search strategy.

Reference software is often optimized for coding efficiency rather than encoding speed because R-D performance is the paramount concern during standardization process. On the other hand, reference software is often used as a benchmark for researchers because of its public availability. JM (joint model) adopted three FME algorithms due to their competitive R-D performance over Full Search. The three FME algorithms are the UMHexagonS [9], the simplified UMHexagonS [10] and the EPZS [11]. The first two algorithms make use of the conventional median predictor, described in the standard, in order to find a better search center and then perform a limited search around this center. On the other hand the EPZS algorithm defines some sets of predicted search points, which are likely to give the best
match. For that purpose the EPZS uses various predictors, such as the median predictor and temporal predictor(s).

In this paper we study the effectiveness of different predictors used by the EPZS algorithm in order to verify that their use is justified. Moreover, we propose a new predictor, which may substitute the median predictor used in [9] and [10] or may also be included in the set of the predictors used in [11]. The rest of the paper is organized as follows. In Section 2 we propose a new predictor, which may give either a search center for [9] and [10] or another predicted search point for [11]. In Section 3 we present the simulation results of the new predictor. In Section 4 we study the effectiveness of the predictors, which are used in [11]. Finally, in Section 5 conclusions are drawn.

2. Spatio-Temporal predictor

It has been found that the median predictor is more reliable and has higher probability to be the true predictor, especially for nonzero biased sequences. On the other hand, the (0, 0) prediction is more suitable for sequences, which contain a lot of stationary data. Finally, the prediction based on the motion vector of the collocated macroblock is better in a number of cases. The proposed predictor combines the aforementioned predictors in order to estimate a new predictor which covers a wider range of video sequences.

Let $mv$ the desired predictor and $col_{mv}$ the motion vector of the collocated macroblock and $med_{mv}$ the median predictor estimated by the motion vectors $a, b, c$ of the adjacent blocks, as shown in Figure 1. We distinguish the following cases:

1. Stationary block
   - Condition: Both of the coordinates $x$, $y$ of the $col_{mv}$ are zero.
   - Choice: $mv(x, y) = (0, 0)$

2. Vertical movement
   - Condition: Both of the $x$ coordinates of the $col_{mv}$ and the $med_{mv}$ are zero.
   - Choice: If $col_{mv_y} > 2$ then we consider the movement to be fast and we set $mv(x, y) = (0, \max(col_{mv_y}, med_{mv_y}))$
   - Otherwise we set $mv(x, y) = (0, \min(col_{mv_y}, med_{mv_y}))$

3. Horizontal movement
   - Condition: Both of the $y$ coordinates of the $col_{mv}$ and the $med_{mv}$ are zero.
   - Choice: If $col_{mv_x} > 2$ then we consider the movement to be fast and we set $mv(x, y) = (\max(col_{mv_x}, med_{mv_x}), 0)$
   - Otherwise we set $mv(x, y) = (\min(col_{mv_x}, med_{mv_x}), 0)$

4. The current block is moving at the same direction and at the same speed with the collocated block
   - Conditions:
     - Same direction: Both of the $col_{mv}$ and the $med_{mv}$ lie in the same quadrant as in Figure 1.
     - Same speed:
       - $med_{mv_x} - 2 \leq col_{mv_x} \leq med_{mv_x} + 2$
       - $med_{mv_y} - 2 \leq col_{mv_y} \leq med_{mv_y} + 2$
   - Choice: $mv(x, y) = (col_{mv_x}, col_{mv_y})$

5. The current block is moving at the same direction with the collocated block but at different speed
   - Conditions:
     - As above
     - $med_{mv_x} - 2 \leq col_{mv_x} \leq med_{mv_x} + 2$
     - $med_{mv_y} - 2 \leq col_{mv_y} \leq med_{mv_y} + 2$
   - Choice: $mv(x, y) = (\frac{col_{mv_x} + med_{mv_x}}{2}, \frac{col_{mv_y} + med_{mv_y}}{2})$

6. All other cases
   - Conditions:
     - As above
   - Choice: $mv(x, y) = (med_{mv_x}, med_{mv_y})$

The proposed predictor leads to a considerable reduction of the time devoted to motion estimation, with a comparable rate-distortion performance. Extended comparisons with existing predictors are conducted and useful conclusions are drawn in the following sections.
3. Simulation results

The proposed scheme was integrated within version 11.0 of the reference JVT software [12]. The reference code uses three fast motion estimation algorithms, namely those in [9], [10] and [11]. All of these algorithms consider the Median Predictor as the initial search point and then they perform a fast search around this point. In our tests we substituted the Median Predictor by our Spatio-Temporal Predictor and then we let the three algorithms do the fast search around the predicted point. The encoder was configured with the default parameters of the baseline profile as these are described by the encoder_baseline.cfg file, which is included in the reference code. We have tested the proposed scheme on different video sequences and the results are shown in Table 1. To simplify the comparisons we present the percentage differences of the PSNR (Y frames) and of the motion estimation speed of the three Motion Estimation algorithms when the Median Predictor and the proposed Predictor are used. The minus sign in the results denotes an increment while the positive sign denotes a decrement.

From the results we observe that the proposed scheme does not actually affect the PSNR. This was expected since the PSNR is affected mainly by the search pattern of the FME algorithm rather than its initial search point. We also observe that the proposed scheme speeds up the Motion Estimation in most of the test cases. In the vast majority of the cases a speedup was observed, which varied from 0.6% to 7.3 %.

This is a considerable improvement of the existing FME algorithms [9], [10] and [11], taking into account that the proposed scheme leaves the main core of the FME algorithms as is and it simply modifies the initial search point. However, in some cases the proposed scheme was proved to be ineffective since it increased the motion estimation time.

4. Effectiveness of the EPZS predictors

The EPZS [11] is considered to be the most advanced Fast Motion Estimation algorithm among the three, which are used by the H.264 reference code [12]. The basic idea of EPZS is to reduce the candidate search points by predicting search points, which are likely to give good results. For that purpose EPZS uses various search points predictors such as the well-known median predictor, the (0,0) position, the motion vectors of the adjacent blocks in the current frame, the motion vectors of the collocated block and of its adjacent blocks in the reference frame and many others. We decided to study the effectiveness of these predictors in order to find out whether all of them truly contribute to the decrease of motion estimation time. We examined 11 video sequences in QCIF format. The H.264 encoder was configured with the default parameters of the baseline profile as these are described by the encoder_baseline.cfg file, and the results are shown in Figure 2. This figure shows the % contribution of each predictor over the different video sequences.

It is clear that the median predictor is the dominant predictor. The second best predictor seems to be the (0, 0) position. Moreover, the motion vectors of the adjacent blocks in the current frame (Left, Up, UpRight, UpLeft, Mem Left, Mem Up, Mem UpRight) have a significant contribution. The contribution of the motion vector of the collocated block and the contributions of the motion vectors of its adjacent blocks in the reference frame are all summed over the label “Collocate” for simplicity. In practice this number is spread over 9 different predictors. Moreover, the predictors of Block...
Type also have considerable contribution. Finally, the Window type predictors seem to have negligible contribution to the motion estimation and they might have been skipped.

5. Conclusions

In this paper we present a new spatio-temporal predictor. The proposed predictor may be used as the initial search center by [9] and [10] and by other FME algorithms of this type. Moreover, it may be used as an additional search candidate by [11]. The proposed scheme in conjunction with the study of the EPZS predictors shows that it is possible to combine different spatial and temporal predictors in order to form a new better predictor.

6. References


Table 1. Evaluation of the Proposed Scheme

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Format</th>
<th>FME</th>
<th>PSNR Y</th>
<th>Speed up</th>
</tr>
</thead>
</table>
|Hex bridge close QCIF HEX 0.00 1.4 SHEX 0.00 4.7 EPZS 0.00 1.9
|Hex bridge far QCIF HEX 0.00 1.0 SHEX 0.00 6.7 EPZS 0.00 -0.7
|Hex highway QCIF HEX 0.10 4.4 SHEX 0.10 7.3 EPZS 0.00 6.4
|Hex salesman QCIF HEX 0.00 1.9 SHEX 0.00 3.4 EPZS 0.00 2.5
|QCF carphone QCIF HEX 0.20 1.0 SHEX 0.00 6.4 EPZS 0.00 5.0
|QCF news QCIF HEX 0.00 0.6 SHEX 0.10 1.8 EPZS 0.00 -3.0
|QCF grandma QCIF HEX 0.00 3.5 SHEX 0.00 3.3 EPZS 0.00 1.5
|QCF container QCIF HEX 0.10 1.3 SHEX 0.00 2.9 EPZS 0.00 -1.7
|QCF claire QCIF HEX 0.14 -3.0 SHEX 0.00 4.5 EPZS 0.10 6.1
|QCF silent QCIF HEX 0.18 3.4 SHEX 0.00 3.2 EPZS 0.00 1.4
|QCF foreman QCIF HEX 0.00 -1.0 SHEX 0.00 -0.7 EPZS 0.2 3.9