An efficient Video Coding using Phase-matched Error from Phase Correlation Information

Manoranjan Paul\textsuperscript{1} and Golam Sorwar\textsuperscript{2}

\textit{School of Multimedia and Information Technology, Southern Cross University}
Hogbin Drive, Coffs Harbour, NSW 2457, Australia
\textsuperscript{1}Manoranjan.Paul@infotech.monash.edu.au
\textsuperscript{2}GSorwar@scu.edu.au

\textbf{Abstract}—The H.264 video coding standard exhibits high performance in terms of compression and image quality compared to the other existing standard such as H.263, MPEG-X. This improved performance is achieved due to the mainly enormous computations in multiple mode motion estimation and compensation. Recent research tried to reduce the computational time using predictive motion estimation, early zero motion vector detection, fast motion estimation, fast mode decision etc. These approaches successfully reduced the computational time by degrading the image quality. Phase correlation technique is used to find the shift between two pictures. In this paper we used phase correlation technique to indicate the motion information between current and reference block and then we devise an algorithm to predict the motion estimation block size. Using phase correlation we are able to successfully predict the motion estimation mode directly instead of using exhaustive motion estimation by all possible modes, thus we save a huge amount of computational time. The experimental results show that we can save around 50\% time in motion estimation without degrading the image quality.

\section{I. INTRODUCTION}

The variable block size motion estimation and compensation in H.264/AVC \cite{1} is one of the most significant innovations towards the improving coding efficiency when compared to the previous video coding standards. The block sizes (popularly known as modes) are chosen in the range from 16×16 to 4×4 pixels. Choosing larger partition sizes (16×16, 16×8, 8×16, and 8×8) requires a smaller number of bits to encode the motion vectors and the type of partition, at the expense of significant number of bits in motion compensated residual error in areas of high detail. By contrast, choosing smaller partition sizes (8×4, 4×8, and 4×4) may result in a lower number of bits for encoding the residual error at the expense of a larger number of bits to encode the motion vectors and the type of partitions. Thus the choice of mode has a crucial role on rate-distortion optimization.

The method of Lagrange Multipliers (LM) has been used for mode selection to trade off between the quality of the compressed video and the bit rate \cite{1–3}. To find the best mode, bits and distortion are determined by performing motion estimation (ME) and motion compensation (MC) using seven inter modes. It is reported that motion estimation, irrespective of a scene’s complexity, typically comprises more than 60\% of the processing overhead required to encode an inter picture with a software codec using the DCT \cite{10}, when full search is used. Thus, any attempt to improve this computational time eventually extends the video coding applications in real time using low-powered and low-processing electronics devices such as hand-held PDA, mobile-phone etc.

Some fast mode selection algorithms have been proposed in \cite{4}–\cite{8}. Most of the algorithms reduced computational time with sacrificing some quality or increasing bit streams. For example, Ahmad \textit{et al}. \cite{4} proposed a fast mode selection algorithm using motion vector cost and previous frame information. The experimental results showed that this approach reduced encoding time by around 32\% while increasing bitstream size by around 18\% without any degradation of image quality. The main disadvantages of this approach are the extra memory requirement to store the previous information and the increase in bitstream size due to scene changes. Yang \textit{et al}. \cite{5} proposed another fast mode selection algorithm which stopped motion searches early for some cases, thus skipping a large number of search points. Early termination of the search process reduced the encoding time by 12–20\% without increasing of bitstream size for the same image quality throughout the whole bit rate ranges when applied for some standard CIF video sequences (e.g., Foreman, Football, Tennis, and Coastguard). This process would suffer large performance degradation if there is a mistake in early termination due to the presence of a local minimum in the performance surface. Moreover, there is no significant encoding time saving for high motion video sequences with full search motion estimation.

There are two ways to reduce the computational time in encoding. One is reducing the search points in motion estimation and another is reducing operations in the mode selection. The above mentioned algorithms together with fast motion estimation algorithms try to reduce computational time by reducing the search points in motion estimation. Undoubtedly, reducing the searching points also degrades the image quality and increases the size of the bit stream. On the other hand, Paul \textit{et al}. \cite{12} reduces mode selection computational time using only the distortion and motion vector. However, they also reported that around 0.2dB image quality was degraded compared to the original H.264 standard while 31\% operations were reduced only in mode selection.

Phase correlation technique \cite{11,14} provides shifting information between two correlated images using Fourier transformation. To \textit{et al}. \cite{13} also confirmed that phase information provides reliable motion estimation while other
system may fail due to the sensitivity over circumstances. In this paper we used phase correlation technique to indicate the motion information between current and reference block and then we devise an algorithm to predict the motion estimation block size. Using phase correlation we are able to successfully predict the motion estimation mode directly, thus we save huge amount of computational time. The experimental results show that we can save around 50% time in motion estimation without degrading the image quality.

Paper is organised as follows. Section II will detail the proposed technique, computational complexity is analysed in the Section III, simulations result is discussed in the Section III, and Section IV concludes the paper.

II. PROPOSED TECHNIQUE

In our technique, we generate a binary matrix from the motion information using phase correlation information such that ‘0’ indicates no motion and ‘1’ indicates single or multiple motions. If the resolution of an image is \( H \times W \) pixels, then the size of the binary matrix will be \( H/8 \times W/8 \). We assume that for a 16×16-pixel block (popularly known as macroblock), which is considered as a video encoding processing unit, comprises four values. For each macroblock (MB), based on the binary matrix, we will decide which mode it would be for ME&MC.

![Phase correlation matrix](image)

Figure 1: Phase correlation between two corresponding blocks from reference and current frames where (a) current blocks of current frames and (b) two peaks in phase correlation matrix represent two motions, (c) one peak represent motion, and (d) one big magnitude peak at exact middle represent no motion.

A. Binary Matrix Generation using Phase Correlation

The phase correlation technique can estimate initial displacement between adjacent frames [11][14]. It correlates two images by first performing Fourier transformation on each image, multiplying them with corresponding frequency components, and performing inverse Fourier transformation on the resulting image. The result is two dimensional array which will have a peak at the coordinates corresponding to the shift between the two pictures. The sharpness of the peak can be significantly increased by normalizing the amplitude of each frequency component prior to performing the inverse transform. Figure 1 shows phase correlation information using different blocks where different motions are observed. From Figure 1 it is clear that block (5,6), (5,7), and (6,5) represents multiple, one, and no motion respectively. Using phase correlation we also observed that two peaks (b), one peak (c), and one big magnitude peak at middle (d) successfully represent the different motions in corresponding blocks. Note that here we consider 32×32-pixel block for clear visualisation.

Our binary matrix is calculated using phase-matched error [13]. While conventional residual error is calculated by subtracting the matching block from the original, in a phase-matched error the phase component of the matching block is replaced by the phase component of the original block before subtraction. Let \( r \), \( c \), and \( e \) be reference, current blocks, and phase-matched error respectively. Note that we use four 8×8 blocks to process a MB. The phase-matched error is calculated as follows:

\[
R = F(r).
\]

\[
C = F(c).
\]

\[
r' = \frac{r - |r|}{|r|} r e^{j \theta}.
\]

\[
e = c - r'.
\]

\[
E = F(e).
\]

In phase-matched error, normally the energy will be concentrated on the left-top corner (i.e., upper triangle) if there is no displacement between two MBs, otherwise, the energy will be scattered through out the whole area. Based on this property, we can easily calculate the energy concentration ratio on the upper triangle with respect to the whole energy and predict the motion. The ratio is calculated as follows:

\[
D = \frac{\sum_{u=1}^{6} \sum_{v=1}^{7} (E(u,v))^2}{\sum_{x=1}^{8} \sum_{y=1}^{8} (E(x,y))^2}
\]

Based on the ratio of energy we can easily find the binary matrix as follows:

\[
M(x,y) = \begin{cases} 
0 & \text{if } D < T \\
1 & \text{otherwise}
\end{cases}
\]
where \( T \) is a threshold. Too low value of \( T (\approx 0) \) assigns ‘1’, for almost all blocks assuming that they have single or multiple motions, and on the other hand, too high value of \( T (\approx 1) \) assigns ‘0’ for almost all blocks assuming that they have no motion.

Figure 2 shows the motion and non-motion blocks for standard Tennis video sequence using the binary matrix generated by the phase-matched error where we used \( T=0.3 \). It is clear from the figure that only the motion dominating areas are identified by the green blocks. For example, tennis ball, bat, portion of hand, and some border lines are identified as motion areas.

B. Mode Selection from the Binary Matrix

As we mention earlier that a 16x16-pixel MB is used as an encoding processing unit. It is also observed that phase-matched error is not effective when we used any block less than 8x8 pixels level. In our experiment we use 8x8 pixels block for binary matrix generation. Thus in a MB, we will get four sub-blocks. Binary decision using sub-blocks, we have to select final mode among 16x16, 16x8, 8x16, and 8x8. The direct mode selection strategy is shown in Figure 3. When a MB has all zeros or only one ‘1’ we assume it 16x16 block mode because ME&MC using whole 16x16 should capture this single motion if any. When we have two ones vertically or horizontally we assume 16x8 or 8x16 is suitable as half portion has no motion and other half has some motion. For vertical position we select 16x8 by assuming that two motions are identical and for the same reason we select 8x16 for horizontally two ones. To inspect details we use 8x8 for any other combinations. Obviously by assuming 8x8 mode in other combinations, we give slightly advantage to this mode but the experimental results show that it enhances the coding performance. Note that for smaller mode (such as 8x4, 4x8, and 4x4) we used same strategy as phase-matched error and binary matrix. We need 3\( N^2 \) operations for assigning ‘1’ or ‘0’. Thus total \( 2kN^2 \) operations for each mode. Thus 16x16 is suitable as half block we 8x16 should capture this single motion if any. When we have two ones vertically or horizontally we assume 16x8 or 8x16 is suitable as half portion has no motion and other half has some motion. For vertical position we select 16x8 by assuming that two motions are identical and for the same reason we select 8x16 for horizontally two ones. To inspect details we use 8x8 for any other combinations. Obviously by assuming 8x8 mode in other combinations, we give slightly advantage to this mode but the experimental results show that it enhances the coding performance. Note that for smaller mode (such as 8x4, 4x8, and 4x4) we used same strategy as phase-matched error system is not effective for small size of blocks.

Figure 4 shows the comparative results by the exhaustive mode selection technique using the H.264 and direct mode selection technique using proposed strategy. From the figure it is clear that our approach clearly identifies motion dominating areas in the both sequences where as the H.264 identifies a portion of them. For example tennis ball and bat are clearly identified in Figure 4 (b) and moving train in Figure 4 (a) is also clearly identified by the proposed technique. On the other hand, the motion dominating areas in ball and bat of Tennis sequences or moving train in Mobile&Calendar sequences are not clearly identified by the H.264 exhaustive mode selection technique. This happens because the large block generates smaller Lagrangian cost function compared to the other small block. Note that the cost function is calculated by adding distortion with bits multiplied by Lagrangian multiplier.

III. COMPUTATIONAL COMPLEXITY

According to our strategy we only use one ME and MC for larger blocks and if 8x8 mode is selected than we apply ME&MC for all small blocks. For fair comparison we need to find overhead for calculating phase-matched error and binary matrix. We need to find Fourier transform of each block, replace reference block phase with current block phase, calculate the ratio of upper triangular error compared to the total error, and assign ‘1’ or ‘0’ to a matrix. For \( N\times N \) block we need 3\( N^2 \) operations for Fourier transform/inverse transform, \( N^2 \) operations for replacing phase, 0.25 \( N^2 \) operations for ratio calculations, \( N^2 \) operations for assigning ‘1’ or ‘0’. Thus total 5.25 \( N^2 \) operations for up to binary matrix calculations.

For full search motion estimation using \( d \) width search length, we need 3\( N^2 (2d+1)^2 \) operations for each mode. Thus the H.264 needs \( k_H (3N^2 (2d+1)^2) \) operations where \( k_H \) is the average modes ME&MC per MB using H.264. The proposed technique needs \( k_p (3N^2 (2d+1)^2 + 5.25N^2) \) operations where \( k_p \) is the average modes per MB. As 5.25 is negligible compared to the \( 3(2d+1)^2 \), \( k_H - k_P \) would be reduction of computational complexity using proposed technique compared to the H.264. The experimental results show that the proposed technique requires around half number of modes per MB at the middle range of bit rates. At whole range of bit rates, the average number of modes in proposed technique is
less sensitive compared to the H.264 standard. In the H.264, the average number of modes is increasing with the bit rates. Thus around 50% computational time is reduce using the proposed technique.

IV. SIMULATION RESULTS

In this paper, experimental results in Figure 5 are presented using the first 100 frames of two standard video sequences, comprising I- and P-type frames of CIF digital video format [9]. Full-search motion estimation and H.264 recommended ‘baseline’ profile were employed to obtain the encoding results for standalone H.264 and proposed technique.

Figure 5 shows rate-distortion performance using the both techniques for two standard video sequences. For tennis sequence the H.264 performs better whereas for Mobile&Calendar sequence proposed technique performs better. As we know the Lagrangian multiplier \( \lambda = 0.85 \times 2^{(b_0-12)/3} \) is used to select the mode using a cost function comprising bits and distortion. This multiplier is derived using experimental values from various standard and non-standard video sequences. This is an average value which more or less works for all sequences. As we observed that Mobile&Calendar sequence has relatively more motion compared to the other sequences, for example, Tennis sequence, thus it requires more bits to encode a MB. Due to the limitation of multiplier, it selects those modes which are biased to fewer bits. Eventually it surfaces the quality. The proposed technique selects the mode based on the motion, thus it performs better. On the other hand, the Tennis sequence has moderate motion and thus the H.264 performs better.

V. CONCLUSIONS AND FUTURE WORKS

In this paper, we proposed a video coding technique where motion estimation and compensation modes are directly selected using phase correlation information instead of exhaustive motion estimation and compensation using all possible modes. The experimental results show that proposed technique reduces around 50% computational time without losing any image quality compared to the H.264 video coding standard.

We believe that further computational complexity can be reduced if we can use motion vector which is generated by the phase correlation. It would not be straightforward because some motion vector might be too long so that ultimate coding efficiency could not be achieved using available motion vector.

We also believe that using same threshold in binary matrix generation for entire bit rates could not optimise the coding efficiency. Further research is necessary to see the effect of threshold on different bit rates.

REFERENCES


