SIViP (2013) 7:53–65
DOI 10.1007/s11760-011-0232-x

ORIGINAL PAPER

Gravity direction-based ultra-fast intraprediction algorithm for H.264/AVC video coding

Abderrahmane Elyousfi

Received: 17 March 2010 / Revised: 15 May 2011 / Accepted: 18 May 2011 / Published online: 8 June 2011

Abstract The H.264/AVC video coding standard uses in intraprediction, 9 directional modes for 4 × 4 and 8 × 8 luma blocks, and 4 directional modes for 16 × 16 luma macroblocks, and 8 × 8 chroma blocks. The use of the variable block size and multiple modes in intraprediction makes the intracoding of H.264/AVC very efficient compared with other compression standards; however, computational complexity is increased significantly. In this paper, we propose a fast mode selection algorithm for intracoding. This algorithm is based on the vector of the block's gravity center whose direction is used to select the best candidate prediction mode for intracoding. On this basis, only a small number of intraprediction modes are chosen for rate distortion optimization (RDO) calculation. Different video sequences are used to test the performance of proposed method. The simulation results show that the proposed algorithm increases significantly the speed of intracoding with negligible loss of peak signal-to-noise ratio quality.

Keywords Prediction modes · Intraprediction · H.264/AVC · Video coding · Encoder complexity reduction

1 Introduction

IUT-T Video Coding Experts Group (VCEG) and ISO/ IEC Moving Picture Experts Group (MPEG) formed the Joint Video Team (JVT) to develop a new video coding standard in 2001, known as Recommendation H.264 or 14496-10 (M-PEG-4 part 10) Advanced Video Coding (AVC) [1,2]. The core of H.264/AVC was frozen in 2003; compared with the previous video coding standards, H.264/AVC shows better performance in terms of peak signal-to-noise ratio (PSNR) and visual quality at the same bit rate [3]. This is accomplished mainly due to the consideration of integer transform, in-loop deblocking filter, context-based adaptive binary arithmetic coding (CABAC) and also due to better exploitation of the temporal and the spatial correlation that may exist between different frames and adjacent macroblocks (MB), respectively, with the multiple prediction modes and the variable block size in intra- and intercoding [4,5].

Intraprediction is an important technique in image and video compression to exploit spatial correlation within one picture. The intraprediction used in H.264/AVC is different from the one in the other standards. In previous video coding standards (namely H.263 and MPEG-4), intraprediction has been conducted in the transform domain. However, in H.264/AVC, the intraprediction is conducted by using spatially neighboring samples of a given block, which have already been transmitted and decoded. The H.264/AVC video coding standard supports intraprediction for various block sizes. For coding the luma signal, a 16 × 16 macroblock may be predicted as a whole using intra-16 × 16 modes, or the macroblock (MB) can be predicted as individual 4 × 4 blocks using nine intra-4 × 4 modes. In the profiles that support Fidelity Range Extension (FRExt) tools, a macroblock may also be predicted as individual 8x8 blocks using nine intra-8 × 8 modes [6]. The chroma intraprediction uses only intra-8 × 8 with four mode directions similar to those for luma intra-16 × 16 prediction.

The rate distortion optimization technique, RDO, has been employed in H.264/AVC for both intra- and interprediction

A. Elyousfi (✉)
National Engineering School of Applied Sciences,
Computer Science Department, University Ibn Zohr,
Agadir, Morocco
e-mail: elyousfiabdo@ieee.org; elyousfi.abdou@ensa-agadir.ac.ma

Present Address:
A. Elyousfi
E.N.S.A., BP 1136, Agadir, Morocco

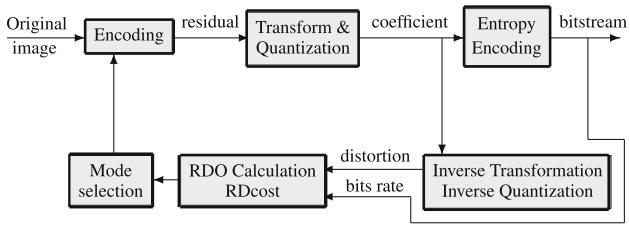

Fig. 1 Computation of RDcost

modes selection to achieve higher coding efficiency. In order to select the optimal encoding mode for an MB, H.264/AVC video encoder calculates the rate distortion cost (denoted as RDcost) of every possible mode and chooses the mode having the minimum value, and this process is repeatedly carried out for all the possible modes for a given MB. Figure 1 shows the RDO process [7,8]. As can be observed from this figure, in order to acquire each RDcost value of a possible mode, the value of the distortion and the rate are computed. The distortion computation involves integer discrete cosine transform, quantization, dequantization, and inverse integer discrete cosine transform; the rate computation requires the entropy coding to be calculated for prediction mode, motion vector, quantization parameter, quantized transform coefficients, and so on. Unfortunately, the computational burden of this type of exhaustively full searching algorithm is far more demanding than any other existing video coding standards.

To reduce this complexity, fast intraprediction algorithms have been proposed [9–19]. Pan et al. [13] proposed a fast mode decision scheme with a pre-processing technique, which measures the edge direction of a given block so as to reduce the number of probable intraprediction modes and thus diminishes complexity. Su et al. [15] propose a fast mode decision algorithm for H.264/AVC intraprediction based on integer transform and adaptive threshold. In this work, before the intraprediction, integer transform operations on the original image are performed to find the directions of local textures.

Li et al. [16,17] have proposed a fast intramode selection algorithm for H.264/AVC. It uses a fast edge detection method which is based on non-normalized Haar transform (NHT), to allow extracting edges for each subblock. For each block, the edge classification algorithm is used to determine the intra-prediction modes. Based on the edge mode, the corresponding prediction modes can be selected in terms of the possible edge directions. However, these approaches increase the bit rate or increase the complexity due to the precalculations needed and/or effect PSNR degradation heavily.

In this paper, we present ultra-fast intraprediction algorithm based on the vector of the block's gravity center. The direction of this vector is used to select the direction of intraprediction. For various video sequences, the simulation results show that the fast algorithm proposed in this paper can increase the encoding speed of intrapicture coding significantly with a negligible PSNR loss or an insignificant bit rate increment.

Subsequent sections are organized as below: Section 2 describes the intramode decision in H.264/AVC. In Sect. 3, we describe in detail the proposed fast intraprediction algorithm. In Sect. 4, we present the experimental results and discuss the performance of the proposed algorithm, and then, we conclude this paper in Sect. 5.

2 Overview of intracoding in H.264/AVC

The H.264/AVC standard exploits the spatial correlation between adjacent macroblocks/blocks for intraprediction; the current macroblock is predicted by the adjacent pixels in the upper and the left macroblocks that have been encoded earlier. For the luma prediction samples, the prediction block may be formed for each 4×4 subblock or for a 16×16 macroblock. The number of prediction modes used to select the best candidate depends on the subblock size: nine modes are used for a 4×4 luma subblock, and 4 modes for a 16×16 luma or 8×8 chroma subblocks.

2.1 4×4 Luma intraprediction modes

In 4×4 intraprediction modes, the values of each 4×4 block of luma samples are predicted from the neighboring pixels above and to the left of the 4×4 block. Nine different directional ways of performing the prediction can be selected by the encoder as illustrated in Fig. 2. The details of these nine modes are listed as follows:

- Mode 0 → Vertical Prediction.
- Mode 1 → Horizontal Prediction.
- Mode 2 → DC (or mean value) Prediction.
- Mode 3 → Diagonal Down-left Prediction.
- Mode 4 → Diagonal Down-right Prediction.
- Mode 5 → Vertical-right Prediction.
- Mode 6 → Horizontal-down Prediction.
- Mode 7 → Vertical-left Prediction.
- Mode 8 → Horizontal-up Prediction.

Each prediction direction corresponds to a particular set of spatially dependent linear combinations of previously encoded samples to be used as the prediction of each input sample. For the purpose of illustration, Fig. 2a shows a 4×4 block of pixels denoted a, b, c …p, belonging to a macroblock to be coded. Pixels denoted A, B, C …H, and I, J, K, L, M are the encoded neighboring pixels used in the prediction computation of the pixels of the current 4×4 block. Figure 2b depicts the eight directional modes. The vertical mode extrapolates a 4×4 block vertically with 4 neighboring

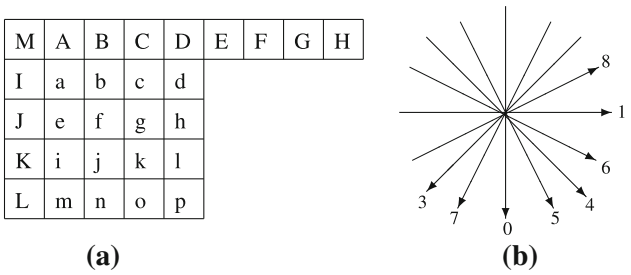

Fig. 2 The 4 × 4 luma *block* and prediction directions

pixels A, B, C, and D, whereas the horizontal mode utilizes the horizontal adjacent pixels I, J, K, and L to perform the prediction. With one exception of the DC mode, all other modes operate in a similar manner, according to their corresponding orientations. The DC prediction mode extrapolates all the pixels as (A+B+C+D+I+J+K+L)/8 [1–3].

In order to provide improved functionality for high fidelity video coding including lossless video coding, Joint Video Team developed extensions to the original H.264/AVC standard known as the Fidelity Range Extensions (FRExt) [6]. In the FRExt amendment, an additional intermediate prediction block size of 8 × 8 was introduced for spatial luma prediction by extending the concepts of 4 × 4 intraprediction to improve coding efficiency. For the 8 × 8 intraprediction, 9 prediction modes are used as in the prediction of a 4 × 4 block.

2.2 16 × 16 Luma and 8 × 8 chroma intraprediction modes

The 16 × 16 luma intraprediction modes are selected in relatively homogeneous area, and four prediction modes are supported and listed as follows:

- Mode 0 → Vertical Prediction.
- Mode 1 → Horizontal Prediction.
- Mode 2 → DC (or mean value) Prediction.
- Mode 3 → Plane Prediction.

These modes are specified in a similar manner to the modes in Intra-4 × 4 prediction except for the plane prediction [4,5]. For the chrominance (chroma) components, there are four prediction modes that are applied to both 8 × 8 chroma blocks (U and V), which are very similar to the 16 × 16 luma prediction except that the order of mode numbers is different:

- Mode 0 → DC (or mean value) Prediction.
- Mode 1 → Horizontal Prediction.
- Mode 2 → Vertical Prediction.
- Mode 3 → Plane Prediction.

To achieve the highest coding efficiency, H.264/AVC employs rate distortion optimization (RDO) technique to get the best coding result in terms of maximizing coding quality and minimizing resulting data bits. According to [13], the number of RDO calculation in a MB is equal to $N8 - chr \times (N4 \times 16 N8 \times 4 + N16)$, where N8-chr, N4, N8, and N16 represent the number of modes for 8 × 8 chroma blocks, 4 × 4, 8 × 8, and 16 × 16 luma blocks, respectively. It means that for an MB, it has to perform $4 \times (9 \times 16 + 9 \times 4 + 4) = 736$ different RDO calculations before a best RDO mode is determined [12,13]. As a result, the complexity of the encoder is extremely high.

3 Fast intramode selection algorithm

It is observed that the modes that provide the least residue energy will also result in minimum rate R and hence minimize the Lagrangian cost. Also, the pixels along the directional correlation of the block normally have similar values. Therefore, a good prediction could be attained if we predict the pixels using their neighboring pixels that are in the same directional correlation of the block. Several algorithms exist to get the directional correlation of the block [9–19]. However, these algorithms may increase the bit rate or may not perform sufficient complexity reduction due to the required pre-calculations and/or may affect heavily the PSNR degradation. The algorithm described in this paper is based on the vector of the block's gravity center. The direction of this vector is used to select the direction of intraprediction. Our algorithm leads to simplicity in terms of computational complexity and good performance in determining the best prediction mode. In the remaining part of this section, we explain in detail the fast intraprediction algorithm based on the vector of the block's gravity center.

3.1 Gravity center of the block theory

In order to obtain the directional correlation of the block to be predicted, the direction of the blocks gravity center is computed. In this study, the blocks are regarded as a system of material points in the plane. The pixel, in a luma (or chroma) block, is considered as a material point and the intensity of this pixel is regarded as the pixel mass. Hence, for every block in the image and at every instant in time, there is a unique location in space (G), representing the block's gravity center and computed as the average mass, i.e., the average position of the system's pixel values.

In an orthonormal coordinate system $(O, \vec{i}, \vec{j})$, for a block, the origin is chosen to be the block's center, i represents the horizontal direction to the right, and j the vertical direction to the bottom. The coordinates of each pixel in this block are a pair of numbers that define its exact location on a two-dimensional plane. Recall that the coordinate plane has

two axes at right angles to each other, called the x and y axes. We denote by $P_{x,y}$ the pixel for which the coordinates are x and y. x is the abscissa and it specifies how far along the x (horizontal) axis the pixel is. y is the ordinate and it specifies how far along the y (vertical) axis the pixel is. We note that x and y axes refer to the block. $\overrightarrow{OP}_{x,y}$ represents the vector of the pixel for which the coordinates are x and y.

The vector of the gravity center $\overrightarrow{OG}$ of the block is written in terms of the pixels values and their coordinates and is defined by:

$$\overrightarrow{OG} = \frac{\sum_{x=x_0}^{x_0+N}\sum_{y=y_0}^{y_0+M} I_{x,y}.\overrightarrow{OP}_{x,y}}{\sum_{x=x_0}^{x_0+N}\sum_{y=y_0}^{y_0+M} I_{x,y}} = X.\vec{i} + Y.\vec{j} \quad (1)$$

where

$$X = \frac{\sum_{x=x_0}^{x_0+N}\sum_{y=y_0}^{y_0+M}(I_{x,y}.x)}{\sum_{i=x_0}^{x_0+N}\sum_{j=y_0}^{y_0+M} I_{x,y}} \quad \text{and}$$

$$Y = \frac{\sum_{x=x_0}^{x_0+N}\sum_{y=y_0}^{y_0+M}(I_{x,y}.y)}{\sum_{x=x_0}^{x_0+N}\sum_{x=y_0}^{y_0+M} I_{x,y}},$$

where $I_{x,y}$ is the intensity of the pixel of location (x, y) of a block, (X, Y) is the coordinate of the gravity center of a block, (x_0, y_0) is the coordinate of the pixel up-left of a block, and (M, N) is the block dimension.

The direction of the block's gravity center vector (OG) is computed by

$$\text{Ang}(\overrightarrow{OG}) = \frac{180}{\pi}\arctan\left(\frac{Y}{X}\right) \quad (2)$$

The amplitude of this vector is measured by:

$$\text{Amp}(\overrightarrow{OG}) = \sqrt{X^2 + Y^2} \quad (3)$$

3.2 Gravity center direction and block correlation direction

To study the relation between the direction of the gravity center vector and the directional correlation of the block, we examine fewer correlation directions of the 4 × 4 blocks. The Eq. (1) that used to compute the vector of block gravity center is applied for the block containing the pixels of the block 4 × 4 (a to p), the four pixels of upper adjacent block (A to D), the four pixels of left adjacent block (I to L), and one pixel of up-left adjacent block (Q). The orthonormal coordinate system $(O, \vec{i}, \vec{j})$ is positioned as shown in the Fig. 3. The origin is chosen to be the center of block formed by block 4 × 4 and their neighboring pixels, i represents the horizontal direction to the right, and j the vertical direction to the bottom.

Figures 3a and b represent, respectively, the horizontal and vertical correlation block. In these figures, the dots represent the pixels of the current block and their neighboring pixels. Dots that have the same color have the same value.

In the block vertical correlation direction as shown in Fig. 3a, all pixels from the same column have the same value. For more explanation, each pixel of the set of pixels $\{Q, I, J, K, L\}$, $\{A, a, e, i, m\}$, $\{B, b, f, j, n\}$, $\{C, c, g, k, o\}$, and $\{D, d, h, l, p\}$ have, respectively, the value $\beta_1, \beta_2, \beta_3, \beta_4$, and β_5, where β_1 to β_5 are the constants values. So, to obtain the direction of the gravity center of this block, we applied the Eq. (1) for this block.

If we let M be the sum of values of the current block pixels and of the adjacent blocks pixels that are used for the intra-prediction, the formula of the equation used for computation this sum is defined as follow:

$$M = \sum_{x=-2}^{2}\sum_{y=-2}^{2} I_{x,y} \quad (4)$$

The Eq. (1) applied for the block of Fig. 3a can be written as follow:

$$\begin{aligned}\overrightarrow{OG} &= \frac{1}{M}\left[\sum_{x=-2}^{2}\sum_{y=-2}^{2} I_{x,y}.\overrightarrow{OP}_{x,y}\right] \\ &= \frac{1}{M}\left[\sum_{x=-2}^{2}\sum_{y=1}^{2}(I_{x,y}.\overrightarrow{OP}_{x,y} + I_{x,-y}.\overrightarrow{OP}_{x,-y}) + \sum_{x=-2}^{2} I_{x,0}.\overrightarrow{OP}_{x,0}\right] \\ &= \frac{1}{M}\left[\sum_{x=-2}^{2} 4x\beta_{x+3}.\vec{i} + \sum_{x=-2}^{2} x\beta_{x+3}.\vec{i}\right] \end{aligned} \quad (5)$$

By summing the above expressions, the vector $\overrightarrow{OG}$ can be expressed as:

$$\overrightarrow{OG} = \beta.\vec{i} \quad (6)$$

where β is a constant value.

From these computations, we found that the direction of the gravity center vector is perpendicular to the vertical direction correlation of the block.

In the block horizontal correlation direction as shown in Fig. 3b, all pixels from the same row have the same value. We let $\alpha_1, \alpha_2, \alpha_3, \alpha_4$, and α_5 be the values of the pixels in rows 1, 2, 3, 4, and 5, respectively. To compute the gravity center vector of this block, the Eq. (1) is applied for this block and

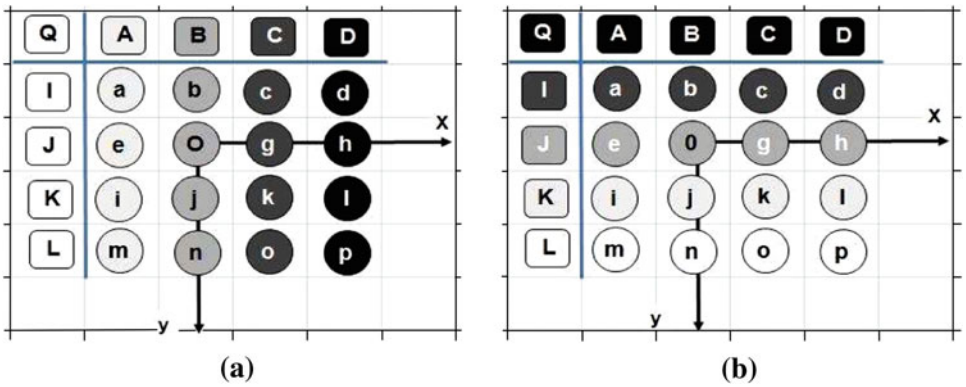

Fig. 3 The 4 × 4 luma *block* with the *vertical* and *horizontal* direction correlation

can be written as below:

$$\begin{aligned}\overrightarrow{OG} &= \frac{1}{M}\left[\sum_{x=-2}^{2}\sum_{y=-2}^{2} I_{x,y}.\overrightarrow{OP}_{x,y}\right] \\ &= \frac{1}{M}\left[\sum_{y=-2}^{2}\sum_{x=1}^{2}(I_{x,y}.\overrightarrow{OP}_{x,y} + I_{-x,y}.\overrightarrow{OP}_{-x,y}) \right. \\ &\quad \left. + \sum_{y=-2}^{2} I_{0,y}.\overrightarrow{OP}_{0,y}\right] \\ &= \frac{1}{M}\left[\sum_{y=-2}^{2} 4y\alpha_{y+3}.\overrightarrow{j} + \sum_{y=-2}^{2} y\alpha_{y+3}.\overrightarrow{j}\right] \end{aligned} \quad (7)$$

Hence, this Eq. (7) can be simplified by :

$$\overrightarrow{OG} = \alpha.\overrightarrow{j} \quad (8)$$

where α is a constant value.

From this above computation, we show that the direction of gravity center vector is perpendicular to the horizontal directional correlation of the block.

As shown in the previous results, we conclude that the direction of the gravity center vector of the block is perpendicular to the directional correlation of this block. This result is valid for all directional intraprediction of the block. (The same process is applied for all directional intraprediction.) Figure 4 shows examples of 4 × 4 blocks intraprediction directions and their corresponding gravity center direction. Hence, the correlation direction of the block is determined by the direction of its gravity center vector.

3.3 Gravity center and directional intraprediction of H.264/AVC

In $H.264/AVC$, intraprediction uses different size block, and each block has a limited number of the intraprediction direction. Hence, we use the angle of the vector of the block gravity center to determine the intraprediction mode of this block. However, we do not have exactly the directional intra-

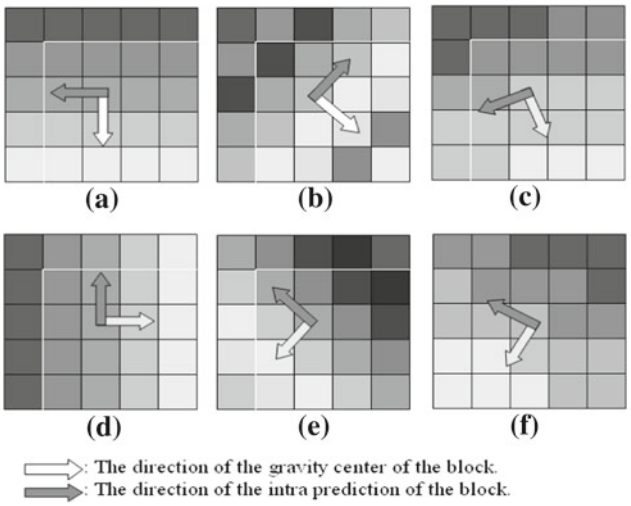

Fig. 4 Examples of 4 × 4 *blocks* intraprediction directions and their corresponding gravity center direction

prediction mode, due to the fact that the pixels in the same direction do not have exactly the same value. Also, the directional prediction modes in H.264/AVC are only represented in the half circle. The angle of the gravity center of the block can have the value between 0 and 2Π or this direction depends on the directional correlation and the position of the greater and the smaller pixel values. The directional correlation is circularly symmetric. In order to determine the best intraprediction candidate in H.264/AVC, the [0, 2Π] angle interval of the block's gravity center is partitioned into a set of subintervals, and each subinterval is associated with a prediction mode.

3.3.1 4 × 4 *Luma block directional correlation*

The 4×4 luma blocks are more suitable to predict the pictures with significant details. There are nine prediction modes, the DC prediction mode and eight directional prediction modes are specified as illustrated in Fig. 2. The prediction directions of these modes are only represented in the half circle as shown in Fig. 2b. The border between any two adjacent

directional prediction modes is the bisectrix of the two corresponding directions. It is important to note that Mode 3 and Mode 8 are adjacent due to circular symmetry of the prediction modes. The mode of each 4×4 block is determined by its gravity center direction. Therefore, the gravity center directions of the 4×4 block's and its preferred 4×4 intra-prediction mode are described as follow. For each 4×4 luma block, let θ_1 be the angle of the 4×4-block's gravity center, and let $\beta_1 = \theta_1 - \frac{\pi}{2}$, then:

$$I4 \times 4 = \begin{cases} \text{Mode 6} & \beta_1 \in]\frac{\pi}{16}, \frac{3\pi}{16}] \cup]\frac{\pi}{16} + \pi, \frac{3\pi}{16} + \pi] \\ \text{Mode 4} & \beta_1 \in]\frac{3\pi}{16}, \frac{5\pi}{16}] \cup]\frac{3\pi}{16} + \pi, \frac{5\pi}{16} + \pi] \\ \text{Mode 5} & \beta_1 \in]\frac{5\pi}{16}, \frac{7\pi}{16}] \cup]\frac{5\pi}{16} + \pi, \frac{7\pi}{16} + \pi] \\ \text{Mode 0} & \beta_1 \in]\frac{7\pi}{16}, \frac{9\pi}{16}] \cup]\frac{7\pi}{16} + \pi, \frac{9\pi}{16} + \pi] \\ \text{Mode 7} & \beta_1 \in]\frac{9\pi}{16}, \frac{11\pi}{16}] \cup]\frac{9\pi}{16} + \pi, \frac{11\pi}{16} + \pi] \\ \text{Mode 3} & \beta_1 \in]\frac{11\pi}{16}, \frac{13\pi}{16}] \cup]\frac{11\pi}{16} + \pi, \frac{13\pi}{16} + \pi] \\ \text{Mode 8} & \beta_1 \in]\frac{13\pi}{16}, \frac{15\pi}{16}] \cup]\frac{13\pi}{16} + \pi, \frac{15\pi}{16} + \pi] \\ \text{Mode 1} & \beta_1 \in [-\frac{\pi}{16}, \frac{\pi}{16}] \cup [-\frac{\pi}{16} + \pi, \frac{\pi}{16} + \pi] \end{cases}$$

Note that the DC mode has no direction. This mode is used for smoothness purposes of the block. So, in order to decide whether this mode is another prediction mode candidate, we can compare the amplitude of the block's gravity center with a threshold value. However, it is difficult to pre-define a universal threshold that suits for different block contexts and different video sequences. Since the 4×4 block has the smallest block size, the DC mode of a 4×4 block is likely to be the best prediction mode of the nine modes. Hence, the DC mode is always a prediction mode candidate for 4×4 intrablocks.

The 8×8 intrablock has nine directional predictions and are similar to those of the 4×4 intrablock. For this reason, we use the direction of the block's gravity center to determine the directional correlation of 8×8 intrablocks using similar steps used for 4×4 intrablocks.

3.3.2 16×16 *Luma and* 8×8 *chroma block directional correlation*

In the case of 16×16 luma and 8×8 chroma blocks, there are only two directional prediction modes, plus a plane prediction and a DC prediction mode. Therefore, the direction of the block's gravity center for this case will be based on three directions, i.e., horizontal, vertical, and diagonal (plane) directions. We note that both diagonal down-right and diagonal down-left prediction modes are associated with the plane prediction. So, to determine the prediction mode candidate, we associate the directional correlation for horizontal and vertical prediction modes to their corresponding areas of the block's gravity center direction and the rest of this areas is associated with the plane mode.

Therefore, for each 16×16 luma block (8×8 chroma block), the gravity center direction of this block and their corresponding prediction modes are represented as follow. For each 16×16 luma block (8×8 chroma block), let θ_2 be the gravity center angle and let $\beta_2 = \theta_2 - \pi/2$, then:

$$I16 \times 16 = \begin{cases} \text{Mode 1} & \beta_2 \in [-\frac{\pi}{8}, \frac{\pi}{8}] \cup [-\frac{\pi}{8} + \pi, \frac{\pi}{8} + \pi] \\ \text{Mode 0} & \beta_2 \in]\frac{3\pi}{8}, \frac{5\pi}{8}] \cup]\frac{3\pi}{8} + \pi, \frac{5\pi}{8} + \pi] \end{cases}$$

For the mode DC, the 16×16 luma and 8×8 chroma blocks chose always this mode as prediction mode candidate. This is due to the fact that 16×16 luma intraprediction is more suitable for coding very smooth areas of a picture and because the chroma signals are very smooth in most cases.

3.4 Mode decision for intraprediction

As mentioned previously, our algorithm uses the gravity center method to determine the correlation direction of the block. In addition, the proposed algorithm introduces also a technique which is based on the key observation of the fact that the correlation direction of a smaller block is similar to that of larger block, i.e., the correlation direction of 4×4 block within 8×8 block has the most probably to have the same direction as that of this 8×8 block. Based on these techniques, the ultra-fast intraprediction algorithm for H.264/AVC video coding selects a small number of the prediction modes as the candidates to be used in RDO computation. It should be noted that the actual RDO computation in H.264/AVC intracoding is based on the reconstructed images. So, we can determine the candidate mode for intracoding block size by the following rules:

Step 1: Obtain the directional correlation for the sixteen 4×4 blocks building the 16×16 MB by using the gravity center method and then calculate, for all 4×4 blocks, the RDO for the DC mode and the mode direction selected by gravity center method.

Step 2: For each 8×8 block, the prediction mode candidates of intra-8×8 are: the mode selected by the gravity center method of 8×8 block and the modes of the four 4×4 block making this 8×8 block and then calculate the RDO for these candidate modes.

Step 3: The prediction mode candidates of intra-16×16 are the DC mode, the mode selected by the gravity center method for 16×16 blocks, and the modes resulting from the 8×8 block making this 16×6 block as described in Table 1. The RDO is computed for these candidate modes.

Table 1 Candidates 16 × 16 modes according to 8 × 8 mode

8 × 8 Block	Candidates 16 × 16 block
Mode 7, 0, 5	Mode 0 (vertical)
Mode 8, 1, 6	Mode 1 (horizontal)
Mode 3, 4	Mode 3 (plane)

Step 4: The prediction mode candidates of chroma 8 × 8 are similar to those of intra 16 × 16. However, in chroma 8 × 8, the RDO is computed for the DC mode and for the best mode of luma intra 16 × 16.

4 Computational complexity assessment

In this section, numerical comparisons are made between our algorithm and the fast algorithms in terms of the RDO calculation and the calculation operations.

4.1 The number of the RDO calculation

Table 2 summarizes the number of candidate modes selected for RDO computation by the fast intraprediction methods. As it can be seen from Table 2, the encoder with our proposed algorithm needs to perform only a number of mode combinations for an MB $1 \times (2 \times 16 + 1 \times 4 + 2) = 38$ if the following conditions are verified:

- The four 4 × 4 blocks of each 8 × 8 block of the MB have the same best intramode prediction.
- The mode selected by the gravity center method for each 8 × 8 block and the best intraprediction mode of the four 4 × 4 blocks of this 8 × 8 block are identical.
- The mode selected by the gravity center method for the MB is identical to the modes of this MB that is resulted by the 8 × 8 blocks as described in Table 1.
- The best intraprediction mode of the MB is the DC mode.

The upper limit of RDO calculations in our algorithm is $2 \times (2 \times 16 + 5 \times 4 + 4) = 112$ if the following conditions are verified:

- The best intraprediction modes of the four 4 × 4 blocks of each 8 × 8 block are different. These modes are different to the mode selected by gravity center method for this 8 × 8 block.
- The mode selected by the gravity center method for the MB and the modes resulting from the 8 × 8 blocks making this MB have three modes different.
- The DC mode is not the best intraprediction mode of the MB.

From these data, our proposed algorithm significantly reduces the number of RDO calculation; compared with Rui Su et al.'s (between $2 \times (4 \times 16 + 4 \times 4 + 2) = 164$ and $2 \times (9 \times 16 + 9 \times 4 + 4) = 368$), Pan et al.'s (between $2 \times (4 \times 16 + 4 \times 4 + 2) = 264$ and $3 \times (4 \times 16 + 4 \times 4 + 2) = 396$), H. Li et al.'s (between $2 \times (1 \times 16 + 1 \times 4 + 0) = 40$ and $3 \times (4 \times 16 + 4 \times 4 + 4) = 252$), and to full search (FS) method of $H.264/AVC$ (equal to $4 \times (9 \times 16 + 9 \times 4 + 4) = 736$).

4.2 The number of the calculation operations

Table 3 gives the comparisons of the computational complexity of different methods. In Pan et al.'s algorithm, the Sobel edge operator for each of the luma or chroma samples needs 5 integer additions and 1 shifting operation. So, the horizontal and the vertical Sobel operations for each sample need at least 10 integer additions and 2 shifting operations. Hence, the Sobel operations for a sixteen 4 × 4 blocks need 2,560 integer additions and 512 shifting operations. To get the direction of the edge for each sample, a division operation is necessary. So, there are totally 256 division operations for a whole sixteen 4 × 4 blocks.

In Su et al.'s algorithm, the integer transform for a 4 × 4 block needs 64 integer addition and 16 shifting operations. Therefore, there are totally 1,024 integer addition and 256 shifting operations for a sixteen 4 × 4 blocks. To obtain the value of $\tan\theta$ for the two components, 1 float addition is employed. Certainly, 16 division operations are needed to obtain the local directions of the sixteen 4 × 4 blocks.

Li et al's algorithm needs 24 integer addition operations for computing the four NHT coefficients for a 4 × 4 block, 5 integer addition, and 3 division operations for a function used to perform a threshold, 1 shifting operation to choose EMA (edge model auxiliary diagonal) or EMP (edge model principal), and 4 addition operations on choosing EMA-IA, EMA-IB, or EMA-IC (first edge model auxiliary diagonal A, B, or C). Therefore, there are totally more than 508 integer addition, 16 shifting operations, and 48 division operations for a sixteen 4 × 4 blocks. However, this method needs a lot of comparisons to choose one model from 42 models.

However, in the gravity center method, for a 4 × 4 block, only 19 integer additions and 1 shifting operation are needed for one component of the gravity center vector. So, to compute the coordinates of the gravity center vector for each 4 × 4 block, it requires at least 38 integer additions and 2 shifting operations. In total, there are 608 integer additions and 32 shifting operations for a sixteen 4 × 4 blocks. To obtain the value of $\tan\theta$., for the two components, only 1 float division is employed. Certainly, 16 division operations are needed to obtain the local directions of the sixteen 4 × 4 blocks.

From these comparisons, it is obvious that the complexity of our algorithm of directional correlation determination is far less than Pan et al.'s and Su et al.'s. For Li al.'s algorithm,

Table 2 Comparison of the number of candidate modes

Algorithms	Luma 4 × 4	Luma 8 × 8	Luma 16 × 16	Chroma 8 × 8	Min and max number of RDO computation
FS [20]	9	9	4	4	736–736
Pan [13]	4	4	2	2–3	264–396
Su [15]	4–9	4–9	2–4	2	164–368
Li [17]	1–4	1–4	0–4	2–3	040–252
Proposed	2	1–5	2–4	1–2	038–112

Table 3 Comparison of operators number in determining the directional correlation for sixteen 4 × 4 blocks

Algorithms	Addition operations	Shift operations	Division operations
Pan [13]	2560	512	256
Su [15]	1024	256	16
Li [17]	More than 508	16	48
Proposed	608	32	16

the number of operations approximates the number of operations used in our algorithm. Hence, our algorithm is more suitable than all others algorithms that are using large number of operations to determine the best intraprediction mode candidate.

5 Experimental results

This section presents simulation results based on Pan et al.'s algorithm [13], Su et al.'s algorithm [15], Li et al.'s algorithm [17], and the proposed fast intraprediction algorithm.

All the algorithms were implemented into $H.264/AVC$ reference software $JM10.1$ [20]. The system platform is the Intel Pentium (R) D CPU Processor of speed 3.4 GHz, 0.97 Gbytes RAM, and Microsoft Windows XP. The test conditions are as follows:

- MV search range is ±32 pels for QCIF and CIF,
- RD optimization is enabled,
- Reference frame number equals to 1,
- Motion estimation scheme is Full Search,
- MV resolution is 1/4 pel,
- CABAC is enabled,
- GOP structure is full I, IPPP, or IBBPBBP,
- The number of frames of each sequence is 150,
- FREXT Profile: High profile,
- Two different frame formats, QCIF (144 × 176) and CIF(288 × 352) are used,
- All test sequences are in 4:2:0 formats.
- Frame Rate per second equals to 30 fps

A group of experiments were carried out on the test sequences with the 4 quantization parameters, i.e., QP = 28, 32, 36 and 40 as specified in [21].

The selected sequences in two different resolutions, namely, QCIF (144 × 176) and CIF (288 × 352) formats, are Foreman, Coastguard, Mother, Paris, News, Silent, Container an Hall. The averaged PSNR values of luma (Y) and chroma (U, V) are used and are based on the equations below:

$$\overline{\mathrm{PSNR}} = 10\log_{10}\left(\frac{255^2}{\overline{\mathrm{MSE}}}\right) \tag{9}$$

where the average mean square error ($\overline{\mathrm{MSE}}$) and the mean square error (MSE) are, respectively, defined by the following Eqs. (10) and (11).

$$\overline{\mathrm{MSE}} = \frac{4\times \mathrm{MSE}_Y + \mathrm{MSE}_U + \mathrm{MSE}_V}{6} \tag{10}$$

$$\overline{\mathrm{MSE}} = \frac{1}{N\times M}\sum_{n=1}^{N}\sum_{m=1}^{M}|S(n,m) - S^{'}(n,m)|^2 \tag{11}$$

Here, S and S' are the original and the reconstructed luma (Y) or chroma (U,V) pictures, (m, n) are the coordinates, and M and N are, respectively, the width and height of pictures.

The comparisons with the case of exhaustive search were performed with respect to the $PSNR$ difference (△PSNR), the data bits rate difference (△Bit), and the difference of coding time (△Time).

In order to evaluate the time saving of the fast prediction algorithm, the following calculation is defined to find the time differences. Let T_{JM} denote the coding time used by full search intraprediction algorithm of JM10.1 encoder and T_{FI} be the time taken by the fast mode decision algorithm, the time difference is defined as:

$$\triangle\mathrm{Time} = \frac{T_{\mathrm{FI}} - T_{\mathrm{JM}}}{T_{\mathrm{JM}}}\times 100\% \tag{12}$$

The differences between PSNR and bit rate are calculated according to the numerical averages between the RD curves

Table 4 Simulation results for IPPP frames sequences

Sequence	ΔTime				ΔPSNR				ΔBitrate			
	Sobel	DCT	NHT	Gravity	Sobel	DCT	NHT	Gravity	Sobel	DCT	NHT	Gravity
Foreman (CIF)	−18.923	−18.998	−20.026	-22.103	−0.069	−0.065	−0.093	−0.064	1.546	0.862	1.904	1.065
Coastguard (CIF)	−18.925	−17.693	−19.147	−22.825	−0.167	−0.165	−0.170	−0.084	2.846	1.884	2.877	2.566
Mother (CIF)	−19.069	−18.524	−20.355	−24.764	−0.098	−0.136	−0.117	−0.025	1.984	2.001	2.150	1.032
Paris (CIF)	−20.330	−19.814	−21.856	−26.905	−0.031	−0.041	−0.048	−0.012	0.647	0.514	0.706	0.323
News (CIF)	−20.223	−19.799	−23.088	−25.743	−0.047	−0.032	−0.039	−0.024	1.025	0.324	0.680	0.064
Silent (QCIF)	−20.245	−19.024	−24.390	−27.294	−0.065	−0.025	−0.072	−0.031	0.895	1.284	1.251	1.075
Container (QCIF)	−18.746	−18.986	−20.179	−22.980	−0.066	−0.065	−0.071	−0.063	1.547	0.858	1.227	1.065
Foreman (QCIF)	−18.127	−18.076	−19.922	−22.003	−0.054	−0.058	−0.100	−0.048	1.513	0.851	2.031	0.850
Hall (QCIF)	−19.986	−19.214	−21.250	−26.286	−0.039	−0.030	−0.042	−0.022	1.259	0.363	0.736	0.066
Average	**−19.397**	**−18.903**	**−21.134**	**−24.544**	**−0.070**	**−0.068**	**−0.083**	**−0.041**	**1.473**	**0.993**	**1.506**	**0.900**

derived from the JM 10.1 original encoder and the proposed fast algorithm, respectively. The detailed procedures for calculating these differences can be found in a JVT document by Bjontegaard [21], which is recommended by the JVT Test Model Ad Hoc Group [22].

5.1 Experiments on IPPPP sequences

The P-frame coding is a technique included in the baseline profile of H.264/AVC. Therefore, the IPPPPP structure is widely adopted in many applications, and the coding efficiency with this structure is improved in comparison with the all the I-frames structure. In the IPPPPP structure encoding, MBs in P-frame also can adopt intracoding as the possible coding modes in RDO operation [13,15].

Table 4 shows the tabulated performance comparison of the proposed algorithm, DCT [15], sobel [13], and NHT [16,17] algorithms for various sequences with IPPP type. In this experiment, the total number of frames is 150 for each sequence, and the period of I-frames is 50, i.e., there is one I-frame for every 50 coded frames. Note that in the tables, positive values mean increments and negative values mean decrements.

It can be seen from Table 4 that the proposed algorithm achieves very high encoding time saving (average 24.54%), which means that our algorithm only takes about 3/4 of the time that is needed by the default algorithm in JM10.1. This result is obtained with negligible losses in PSNR (average 0.0418 dB) in increments in bitrate (average 0.9010%).

With Pan et al.'s, Su et al.'s, and Li et al's algorithms, time saving is estimated in average to 19.3976, 18.9035 and 21.1354%, respectively. The results also showed that the average losses of PSNR and the average increment in bitrate are higher than the values obtained with our approach, about 0.0712, 0.0713, and 0.0840 dB, and about 1.4739, 0.9939, and 1.5072%, for Pan et al.'s, Su et al.'s and Li et al's algorithms, respectively.

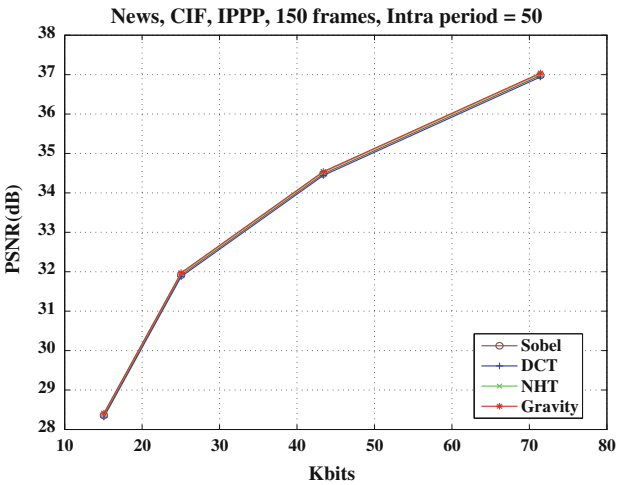

Fig. 5 Comparison of PSNR for the IPPP sequences of News

Figures 5 and 6 show the RD performance and the computation time for the IPPP sequences "NEWS". Figure 5 representing the RD curves found in the four cases: the proposed gravity method, Sobel method, DCT method, and NHT method, shows that these curves are almost overlapping each other. It means that the performance of the proposed algorithm is almost similar to that of the other methods, whereas Fig. 6 reveals that the encoding time with our fast prediction algorithm is distinctly less than that required by the other methods under the same test conditions.

5.2 Experiments on IBBPBB sequences

The B-frame coding is a technique included in the main profile of H.264/AVC, and this tool improves the coding efficiency in comparison with the P-frame encoding. In this

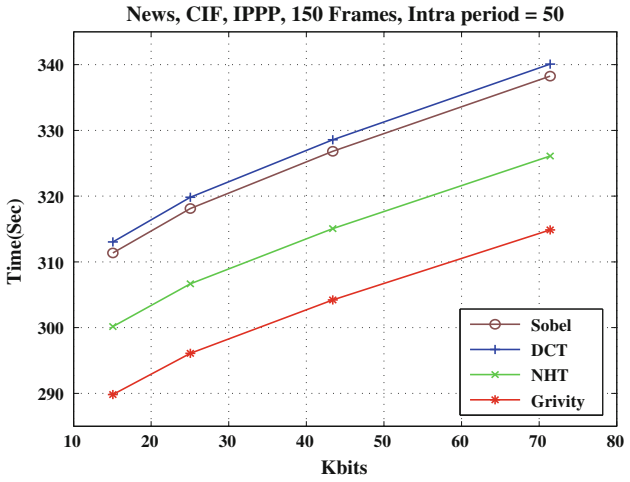

Fig. 6 The computational time comparison of news IPPP sequence

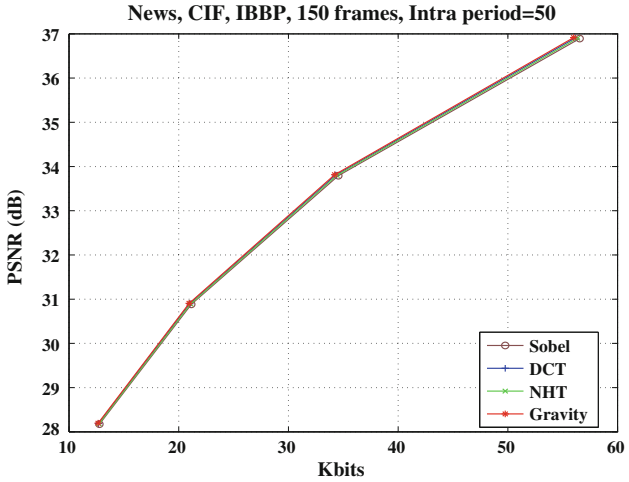

Fig. 7 Comparison of PSNR for the IBBPBB sequences of news

simulation, the picture type is set to IBBPB, i.e., there are two B-frames between any two I- or P-frames. The total number of the frames is 150 for each of the sequences. The period of I-frames is 50, i.e., one I-frame is inserted into every 50 frames.

Table 5 shows the performance comparison of the proposed algorithm with Pan et al's, Su et al's, and Li et al's algorithms for various sequences of IBBPB type. It is noted that the time saving for this type of sequence is much less than that of the IPPPP format. This is due to the fact that in H.264/AVC coding, in B-frame coding, the motion estimation takes much longer time than that in P-frame coding.

From Table 5, it is observed that our algorithm achieves a consistent time saving (in average 21.0189%), with negligible losses in PSNR (in average 0.0483 dB) in increments in bit rate (in average 0.9409%). Though Pan et al.'s, Su et al.'s, and Li et al's algorithms have a lesser time saving, 13.6473, 12.6487, and 15,1269% in average, respectively. These results are interpreted in terms of average losses of PSNR and the average increment in bitrate which show lower performance than ours proposed algorithm as witnessed by the values of 0.0855 dB and 1.5985%, 0.0882 dB and 1.2048%, and about 0.0992 dB and 1.8988% for Pan et al.'s, Su et al.'s, and Li et al's algorithms, respectively.

Figures 7 and 8 show the RD performance and the computation time for the IBBPBB sequence "News", respectively. From Fig. 8, it is observed that the proposed algorithm achieve faster encoding in intraprediction compared with the Pan et al's, Su et al's, and Li et al.'s algorithms. Again, in Fig. 7, these four RD curves show that our fast algorithm has the similar RDO performances as that of the other algorithms.

5.3 Experiments on all intraframes sequences

In this experiment, a total number of 150 frames are used for each sequence, and the period of I-frames is set to 1,

Table 5 Simulation results for IBBPBB frames sequences

Sequence	ΔTime				ΔPSNR				ΔBitrate			
	Sobel	DCT	NHT	Gravity	Sobel	DCT	NHT	Gravity	Sobel	DCT	NHT	Gravity
Foreman (CIF)	−12.044	−11.551	−13.701	−19.583	−0.083	−0.074	−0.085	−0.072	1.753	1.205	1.792	1.218
Coastguard (CIF)	−13.674	−11.978	−13.149	−19.217	−0.179	−0.179	−0.186	−0.089	2.963	2.406	3.110	2.635
Mother (CIF)	−13.790	−12.152	−14.244	−20.238	−0.128	−0.182	−0.190	−0.036	2.177	2.075	2.519	1.059
Paris (CIF)	−13.720	−12.233	−14.790	−21.763	−0.050	−0.058	−0.063	−0.023	0.859	0.792	1.301	0.468
News (CIF)	−13.933	−13.250	−16.800	−21.038	−0.054	−0.057	−0.061	−0.029	1.047	1.039	1.294	0.068
Silent (QCIF)	−14.376	−13.557	−17.669	−22.865	−0.078	−0.056	−0.076	−0.035	1.055	1.466	1.701	1.047
Container (QCIF)	−14.301	−13.828	−14.844	−21.759	−0.083	−0.089	−0.081	−0.069	1.608	1.057	1.908	1.069
Foreman (QCIF)	−11.277	−10.846	−13.919	−19.065	−0.075	−0.070	−0.078	−0.064	1.678	1.082	1.883	0.848
Hall (QCIF)	−15.708	−14.440	−17.022	−23.638	−0.037	−0.025	−0.069	−0.014	1.241	0.358	1.578	0.053
Average	**−13.647**	**−12.648**	**−15.126**	**−21.018**	**−0.085**	**−0.087**	**−0.098**	**−0.047**	**1.597**	**1.275**	**1.898**	**0.940**

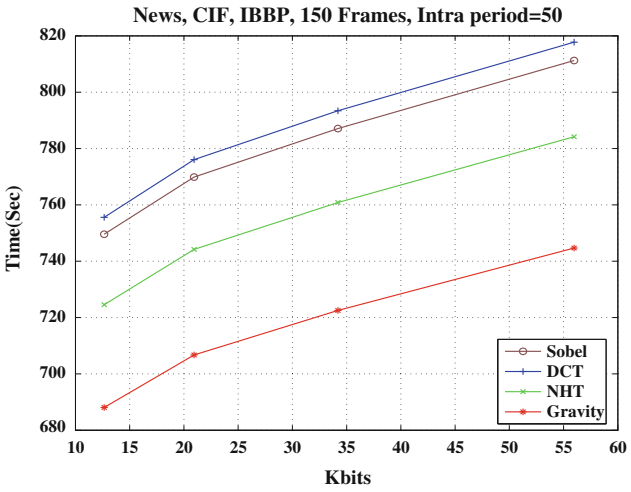

Fig. 8 The computational time comparison of news IBBPB sequence

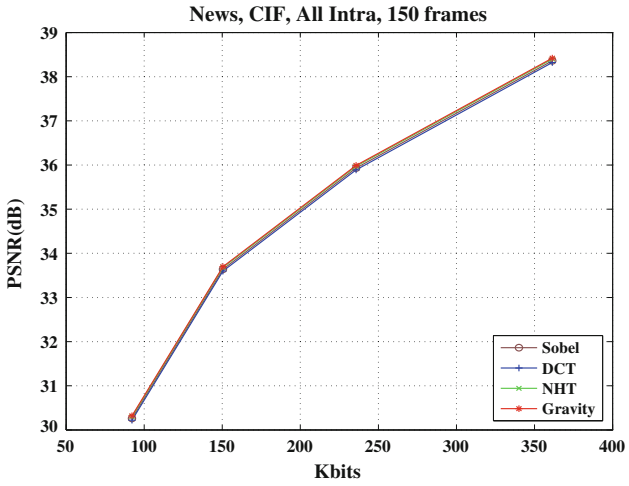

Fig. 9 Comparison of PSNR for all intrasequences of news

i.e., all the frames in the sequence are intracoded. In all the I-frames structure encoding, for each MB in each frame of this sequence, intracoding is chosen as the possible coding modes in RDO operation, thus great time saving is expected by using the fast intracoding algorithms for this structure encoding.

Table 6 shows the simulation results of the proposed algorithm, DCT [15], sobel [13], and NHT [16,17] algorithms for various sequences with all intra-frames type. Notice that in the tables, positive values mean increments and negative values mean decrements.

The results show that the proposed schemes reduced execution time greater than 80% with only an average of 0.1423 dB losses in PSNR and 2.0772% increments in bitrate only. These results are compared with those obtained with Pan et al.'s, Su et al.'s, and Li et al's algorithms where the time coding is reduced in average, respectively, to 62.0287, 61.3690, and 71.5028% with an average of 0.2008, 0.1978, and 0.2180 dB losses in PSNR and, respectively, 2.7446, 2.5289, and 3.02146% increments in bitrate.

The proposed schemes achieve faster encoding in intra-prediction compared with the Pan et al's, Su et al's and Li et al's, with little RD performance enhancement. Figures 9 and 10 show the RD performance and the computation time for the all I-frames sequence "News", respectively. In Fig. 9, four RD curves resulting from the pan et al's algorithm, the Su et al's algorithm, the Li et al's algorithm, and the proposed schemes are nearly overlapping each other which that our proposed algorithm has similar performances as compared with the other approaches in terms of PSNR and data bits, but offers higher computation time saving as shown in Fig. 10.

It can be seen that the proposed algorithm achieves, for all video sequences, very high encoding time saving compared with the Pan et al.'s, Rui Su et al.'s and H. Li et al's algorithms. It shows consistent gain in coding speed for all video

Table 6 Simulation results for all intra-frames sequences

Sequence	ΔTime				ΔPSNR				ΔBitrate			
	Sobel	DCT	NHT	Gravity	Sobel	DCT	NHT	Gravity	Sobel	DCT	NHT	Gravity
Foreman (CIF)	−60.755	−60.117	−69.355	−75.347	−0.237	−0.248	−0.252	−0.187	3.155	3.108	3.577	3.110
Coastguard (CIF)	−59.338	−59.016	−68.810	−75.066	−0.253	−0.249	−0.284	−0.159	3.957	3.765	4.109	3.790
Mother (CIF)	−62.957	−62.010	−70.224	−76.887	−0.216	−0.238	−0.256	−0.147	2.700	2.883	3.381	2.049
Paris (CIF)	−58.230	−57.009	−70.831	−76.884	−0.257	−0.198	−0.240	−0.094	3.478	3.166	3.744	2.178
News (CIF)	−65.461	−64.192	−73.934	−87.650	−0.163	−0.143	−0.145	−0.127	1.561	1.598	1.506	1.090
Silent (QCIF)	−64.184	−63.770	−74.571	−87.711	−0.130	−0.152	−0.164	−0.134	1.980	1.846	2.007	1.219
Container (QCIF)	−62.766	−62.790	−72.336	−80.062	−0.199	−0.190	−0.201	−0.148	2.507	1.372	2.942	1.369
Foreman (QCIF)	−58.357	−57.933	−68.861	−74.167	−0.218	−0.227	−0.239	−0.153	2.770	2.743	3.170	2.682
Hall (QCIF)	−66.207	−65.480	−74.600	−87.697	−0.130	−0.132	−0.178	−0.129	2.591	2.275	2.755	1.205
Average	**−62.028**	**−61.368**	**−71.502**	**−80.163**	**−0.200**	**−0.197**	**−0.217**	**−0.142**	**2.744**	**2.528**	**3.021**	**2.076**

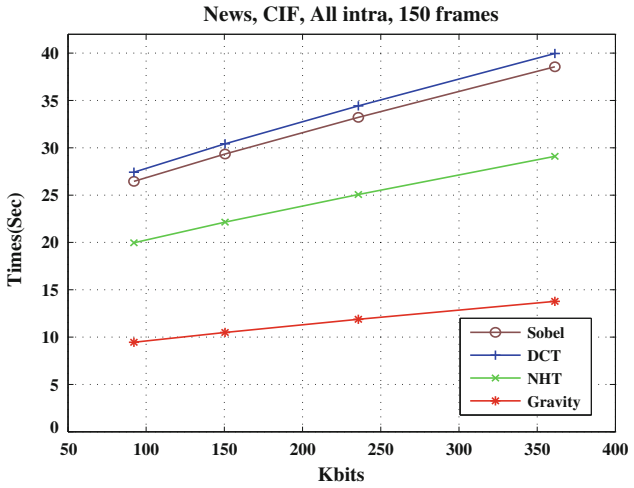

Fig. 10 The computational time comparison of news all intrasequence

sequences with the least gain of 75.0667% in "Coastguard" and most gain of 87.7112% in "Silent."

The sequences "Silent" and "News" show strong spatial homogeneity. The directional correlations of the small blocks for these sequences are similar to these of the bigger block containing those small blocks. Since the lowest number of RDO calculation by our algorithm is applied, the gain in coding speed is high with our algorithm for these sequences. On the other hand, the sequences "Coastguard" and "Foreman" have more non-homogeneous regions in frames. Therefore, the time saving for these sequences is not as much compared with previous sequences.

6 Conclusion

In this paper, we proposed a fast algorithm, namely, The ultra-fast intraprediction algorithm for H.264/AVC video coding. The algorithm improves the computational performance of intraframe coding, thus reducing the implementation requirements in real applications. Improvement is accomplished by discarding the least possible modes to be selected. The fast intraprediction algorithm selects in a smart way fewer candidate modes required to undergo expensive Lagrangian evaluation. It only uses the value of the angle of the block's gravity center to select the best intraprediction mode. This algorithm requires less computational complexity. The results of extensive simulations demonstrate that the proposed algorithm can attain a time saving that is very high than recent and referenced algorithms. This is achieved without sacrificing both picture quality and bit rate efficiency.

References

1. ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4) AVC, "Advanced Video Coding for Generic Audiovisual Services," (version 1: 2003, version 2: 2004) version 3: (2005)
2. Wiegand, T., Sullivan, G., Bjntegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Trans. Circuits Syst. Video Technol. **13**(7), 560–576 (2003)
3. Puri, A., Chen, X., Luthra, A.: Video coding using the H.264/MPEG-4 AVC compression standard. Signal Process. Image Commun. **19**(9), 793–849 (2004)
4. Richardson, Iain E.G.: H.264 and MPEG4 Video Compression: Video Coding for Next Generation Multimedia. Wiley, London (2003)
5. "Report of The Formal Verification Tests on AVC (ISO/IEC 14496-10 | ITU-T Rec. H.264)", ISO/IEC JTC1/SC29/WG11 MPEG2003/N6231, Waikoloa, Hawaii, USA, Dec. (2003)
6. Sullivan, G.J., Topiwala, P., Luthra, A.: The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions. In: SPIE Conference on Applications Of Digital Image Processing XXVII, vol. 5558, pp. 454–474. Denver, Colorado, USA (Aug. 2004)
7. Sullivan, G., Wiegand, T.: Rate distortion optimization for video compression. IEEE Signal Process. Mag. **15**(6), 74–90 (1998)
8. Wiegand, T., Schwarz, H., Joch, A., Kossentini, F., Sullivan, G.J.: Rate-constrained coder control and comparison of video coding standards. IEEE Trans. Circuits Syst. Video Technol. **13**(7), 688–703 (2003)
9. Chen, J.W., Chang, C.H., Lin, C.C., Yang, Y.H., Guo, J.I., Wang, J.S.: A condition-based intra prediction algorithm for H.264/AVC. In; IEEE International Conference on Multimedia and Expo, ICME, pp. 1077–1080. Hilton Toronto, Ontario, Canada (Jul. 2006)
10. Fu, F., Lin, X., Xu, L.: Fast Intra prediction algorithms in H.264/AVC. In: 7th International Conference on Signal Processing, ICSP04, pp. 1191–1194. Beijing, China (Sept. 2004)
11. Elyousfi, A. et al.: A new fast intra prediction mode decision algorithm for H.264/AVC encoders. Int. J. Comput. Syst. Sci. Eng. IJCSSE **4**, 89–95 (2008)
12. Pan, F., Lin, X., Rahardja, S., Lim, K.P., Li, Z.G., Feng, G.N., Wu, D.J., Wu, S.: Fast mode decision for intra prediction. In: JVT-G013, 7th JVT Meeting, Pattaya, Thailand (Mar. 2003)
13. Pan, F., Lin, X., Rahardja, S., Lim, K.P., Li, Z.G., Wu, D., Wu, S.: Fast mode decision algorithm for intraprediction in H.264/AVC video coding. IEEE Trans. Circuits Syst. Video Technol. **15**(7), 813–822 (2005)
14. Kim, J., Jeong, J.: Fast intra-mode decision in H.264 video coding using simple directional masks. In: Visual Communication and Image Procissing, VCIP, Beijing China, Proceedings of SPIE, vol. 5960, pp. 1071–1079 (July 2005)
15. Su, R., Liu, G., Zhang, T. : Fast mode decision algorithm for intra prediction in H.264/AVC with integer transform and adaptive threshold. J. Signal Image Video Process. **1**(1), 11–27 (Springer) (2007)
16. Wei, Z., Li, H., Ngi Ngan, K.: An efficient intra mode selection algorithm for H.264 based on fast edge classification. In: IEEE International Symposium on Circuits and Systems, ISCAS, PP. 3630–3633. Lafayette, USA, (May 2007)
17. Li, H., Ngan, K.N., Wei, Z.: Fast and efficient method for block edge classification and its application in H.264/AVC video coding. IEEE Trans. Circuits Syst. Video Technol. **18**(6), 756–768 (2008)
18. Kim, C., Shih, H., Kuo, C.J.: Fast H.264 intra-prediction mode selection using joint spatial and transform domain features. J. Vis. Commun. Image Represent. **17**(2), 291–310 (2006)

19. Yu, A.C., Ngi, N.K., Martin, G.R.: Efficient intra- and inter-mode selection algorithms for H.264/ AVC. J. Vis. Commun. Image Represent. **17**(2), 310–322 (2006)
20. JM Reference Software Version 10.1 "http://iphome.hhi.de/suehring/tml/download/".
21. JVT Test Model Ad Hoc Group: Evalution Sheet for Motion Estimation. ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Draft version 4 (Feb. 2003)
22. Bjontegaard, G.: Calculation of average PSNR differences between RD-curves, presented at the 13-th VCEG-M33 Meeting, Austin, Texas, USA, (Apr. 2001)