A Video Coding Scheme Based on Joint Spatiotemporal and Adaptive Prediction

Wenfei Jiang, Longin Jan Latecki, *Senior Member, IEEE*, Wenyu Liu, *Member, IEEE*, Hui Liang, and Ken Gorman

***Abstract*—We propose a video coding scheme that departs from traditional Motion Estimation/DCT frameworks and instead uses Karhunen–Loeve Transform (KLT)/Joint Spatiotemporal Prediction framework. In particular, a novel approach that performs joint spatial and temporal prediction simultaneously is introduced. It bypasses the complex H.26x interframe techniques and it is less computationally intensive. Because of the advantage of the effective joint prediction and the image-dependent color space transformation (KLT), the proposed approach is demonstrated experimentally to consistently lead to improved video quality, and in many cases to better compression rates and improved computational speed.**

***Index Terms*—APT, AVC, compression, H.264, joint predictive coding, video.**

I. Introduction

MOST video coding schemes build upon the motion estimation (ME) and discrete cosine transform (DCT) frameworks popularized by ITU-T's H.26x family of coding standards. Although the current H.264/AVC video coding standard achieves significantly better compression over its predecessors, namely, H.263, it still relies on traditional ME/DCT methods. This compression improvement is not without cost. By sacrificing computational time for high complexity motion estimation and mode selection, H.264 reduces the output bit-rate by 50% while increasing the encoding time significantly. Granted, the complex DCT of H.263 was abandoned for the simpler integer transform in H.264, however, the overall time spent calculating motion estimation is increased from 34% to 70% on average [1]. This shows it is costly for H.264 to achieve current compression performance by pursuing accurate motion estimation. The compression efficiency is not likely to be improved significantly by modifying the ME module, and, consequently, the development potential of the ME/DCT coding mechanism may be limited.

Manuscript received June 24, 2008; revised January 28, 2009. Current version published April 10, 2009. This work was supported in part by the National High Technology Research and Development Program of China (Grant 2007AA01Z223) and in part by the National Natural Science Foundation of China (Grant 60572063, 60873127). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Pier Luigi Dragotti.

W. Jiang, W. Liu, and H. Liang are with the Department of Electronics and Information Engineering, Huazhong University of Science and Techology, Wuhan, Hubei 430074, China (e-mail: flyinbati@hotmail.com).

L. J. Latecki and K. Gorman are with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19094 USA (e-mail: latecki@temple.edu; kengorman@yahoo.com).

Digital Object Identifier 10.1109/TIP.2009.2016140

This paper proposes a video coding scheme that reduces the computation complexity of the H.264 interprediction and mode selection calculations by substituting it with a method based on Karhunen–Loeve Transform (KLT) and interpolative prediction framework called Joint Predictive Coding (JPC). It follows a new coding procedure that starts with an image-dependent color space transformation, followed by a pixel-to-pixel interpolative prediction, and ends in entropy coding in the spatial domain. The proposed framework introduces a new way of prediction that takes both temporal and spatial correlation into consideration simultaneously. Since, as mentioned above, efforts on pursuing accurate ME seem to have reached their limit in increasing the compression rate, the proposed prediction method does not follow conventional methods. It performs intraprediction using Adaptive Prediction (Robinson [2]) but not in each frame separately, but instead in both the current and reference frame. The pixels in both frames are linked by MPEG like motion estimation. The calculated error of the reference frame is then used to estimate the intraprediction error in the current frame, and, using these values determines the joint predicted value.

JPC has at least three advantages in comparison to the traditional scheme.

1. It utilizes both temporal and spatial correlation for prediction of each pixel, which leads to more accurate prediction, smaller error, and makes better compression.
2. The encoding is done on a pixel-to-pixel basis instead of on a block-to-block basis in traditional coding, avoiding visually sensitive blocking effects. Moreover, it tends to have good quality on videos with complicated textures and irregular contours, which is a difficult problem for block-based codecs.
3. Interpolative coding allows video frames to be reconstructed with increasing pixel accuracy or spatial resolution. This feature allows the reconstruction of video frames with different resolutions and pixel accuracy, as needed or desired, for different target devices.

Fig. 1 shows the main parts of a JPC coder, including K-L color space transformation, motion estimation module, mode selector, joint predictor, adaptive quantizer and entropy coder. The basic differences between JPC and traditional H.26x family of coding schemes are the prediction and transform module, in the darker boxes. In JPC encoding, pixels in video frames are coded hierarchically. Each pixel is predicted by several available pixels that are most temporally or spatially correlated. The prediction errors are quantized, and entropy coded. This scheme leads to good compression performance at a relatively small computation cost. When using a single frame reference and MPEG-2 level ME techniques [3], such as single block

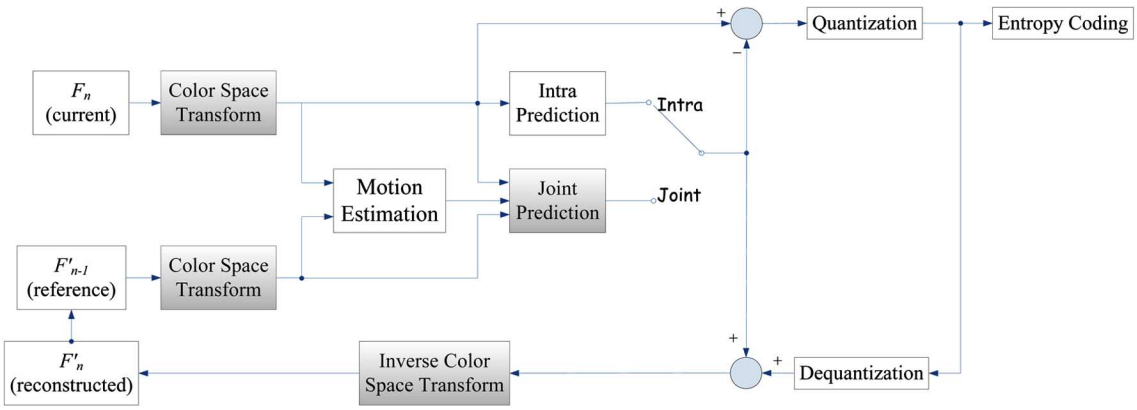


Fig. 1. Block diagram of JPC encoder.

size and half-pel precision ME, JPC significantly outperforms MPEG-2 at the cost of 30% more complexity. It is expected that JPC will do even better if more advanced techniques such as support for variable block sizes, multiple frame reference, and sub-pixel (i.e., quarter-pel) precision motion estimation are used.

Section II briefly introduces interpolative coding and Adaptive Prediction Trees forming the basis of JPC. Section III describes and analyzes the JPC scheme in detail. Section IV presents experimental result and performance comparison. Section V concludes.

II. Interpolative Coding and Adaptive Prediction Trees

Interpolative coding for image compression is a kind of predictive coding dating back over 20 years and was first introduced by Hunt [4]. Several years later, Burt and Adelson built upon the coding scheme [5]. It was further developed by a few different groups [6]–[11]. In such methods, pixels in an image are ordered in a series of groups called bands in increasing sample density and the encoder goes through all the bands from coarse to fine. The main idea is to use grids of points, oriented both squarely and diagonally, to predict their midpoints recursively. These groupings of square and diagonal points are alternately used for neighboring bands.

Fig. 2 shows the band structure of interpolative coding. It represents a 5×5 image which is divided into bands denoted by letters $\boldsymbol{a}$ to $\boldsymbol{e}$. Pixels in each band are predicted by those in higher bands. Pixels labeled $\boldsymbol{a}$ form a square with its center $\boldsymbol{b1}$; thus, $\boldsymbol{b1}$, and all equivalent $\boldsymbol{b}$s throughout the image are predicted by $\boldsymbol{a}$s; similarly, $\boldsymbol{c}$s are predicted by $\boldsymbol{a}$s and $\boldsymbol{b}$s, and so on. Different bands have different quantization intervals for the prediction errors. Pixels in primary bands (especially Band $\boldsymbol{a}$), of more importance, have smaller quantization intervals, while pixels in latter bands with lower entropy are quantized coarsely.

In Fig. 3, each dashed square represents a predictive grid. It helps to explain the algorithm of prediction. Pixel X is the center of the square grid (A, B, C, D). Pixels U and V are earlier processed neighbors in the same band as X while pixels A, B, C, D belong to a higher band. To predict X, bilinear prediction $X_{\text{pred}} = (A + B + C + D)/4$ is usually ineffective. A

a1	**e1**	**c1**	**e2**	**a2**
e3	**d1**	**e4**	**d2**	**e5**
c2	**e6**	**b1**	**e7**	**c3**
e8	**d3**	**e9**	**d4**	**e10**
a3	**e11**	**c4**	**e12**	**a4**

Fig. 2. Band structure.

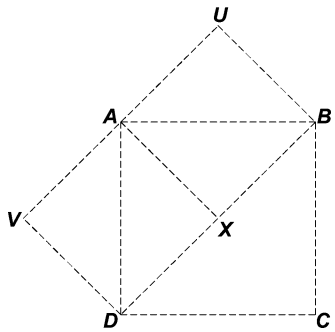


Fig. 3. Predictive grid.

nonlinear approach that selects a subset of its predictor set $P_x = \{A, B, C, D, U, V\}$ as predictors is proposed by Robinson in his image compression scheme called Adaptive Prediction Trees (APT) [2]. The value of a pixel can either be less than, greater than, or equal to its closest neighbor. Let $\mathrm{R} = (>, =, <)$, we have four pixel relationships ($A\mathrm{R}B, B\mathrm{R}C, C\mathrm{R}D$, and $D\mathrm{R}A$). This represents $3^4 = 81$ distinct possibilities. However, several of the relationships are impossible (e.g., $A > B > C > D > A$). Eliminating the impossible relationships, experimental results using training images were obtained to determine the best predictor based upon different combinations of values of pixels in P_x, which is defined as *structure*. These predictors were then combined, duplicates were eliminated, and a small set of "rules" remained.

For the sake of the joint spatiotemporal prediction description in Section III, we split the APT prediction process into two parts. First, get the structure feature in terms of the order of amplitudes of the predictors. Represent this with a function Get_Structure(), with its return value *structure* indicating which

of the six possible pixels are used for prediction. Second, predict X with the predictors based on the determined *structure*. This course is represented by a function Adaptive_Pred(), which returns the estimate of X, X_{pred}. The prediction method in APT can be expressed as follows:

$$\begin{aligned} \text{structure} &= \text{Get\_Structure}(P_x) \\ X_{\text{pred}} &= \text{Adaptive\_Pred}(P_x, \text{structure}). \end{aligned} \tag{1}$$

The Get_Structure() and Adaptive_Pred() functions together not only determine which of the six neighboring pixels are to be used in the prediction, but specify how the prediction is calculated based upon the *structure* determined.

III. Joint Predictive Coding Scheme

This section introduces the joint predictive coding in detail. Subsection *A* describes the joint spatiotemporal prediction technically. Subsections *B* and *C* explain the design process of the mode configuration and the filter for better subjective quality. The main functional stages of JPC encoding are presented in Sub-section D. Our JPC method is compared to binary tree residue coding [12] and temporal prediction trees [13] in Subsections E and F, respectively. A theoretical analysis on joint prediction and traditional motion compensation is shown in the Appendix. Analysis and experiments show that JPC outperforms the other methods.

A. Joint Spatiotemporal Prediction

Classical video coders choose between intra- and interprediction to remove the correlation in the video frames and, thus, achieve greater compression. Unfortunately, there are such cases that neither the temporal correlation nor the spatial correlation holds the dominant position. We propose a joint spatiotemporal prediction that consider both of them and lead to better accuracy in most circumstances.

As shown in Fig. 3, the pixel $X(x, y)$ has six possible neighboring predictors, forming its predictor set $P_x = \{A, B, C, D, U, T\}$. Its motion vector (mv_x, mv_y) can be obtained by motion estimation. The matching pixel (interpolation generated as well) in the reference frame, $X' = (x + mv\_x, y + mv\_y)$, similarly, has its predictor set $P'_x = \{A', B', C', D', U', T'\}$. Joint prediction takes both P_x and P'_x to estimate X. To get the joint predicted value X_{jp}, JPC executes the following steps (see Fig. 4).

First, for a pixel X in the current frame, implement adaptive prediction with its predictor set P_x and acquire the *structure* feature and prediction error. The following functions are defined in Section II. X_{intra} is the intrapredicted value calculated as (2) in the same way as APT and intra_error is the prediction error

$$\begin{aligned} \text{structure} &= \text{Get\_Structure}(P_x) \\ X_{\text{intra}} &= \text{Adaptive\_Pred}(P_x, \text{structure}) \\ \text{intra\_error} &= X - X_{\text{intra}}. \end{aligned} \tag{2}$$

Then, use the same *structure* to estimate the referring pixel X' with its predictor set P'_x as (3). For example, if the estimate of X is $(A+B+C+D)/4$, the estimate of X' shall be $(A'+B'+C'+D')/4$, and if the estimate of X is $(A+C)/2$, the estimate of X' shall be $(A'+C')/2$

$$\begin{aligned} X'_{\text{intra}} &= \text{Adaptive\_Pred}(P'_x, \text{structure}) \\ \text{intra\_error}' &= X' - X'_{\text{intra}}. \end{aligned} \tag{3}$$

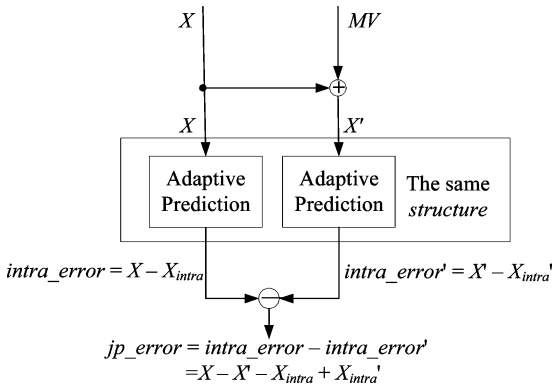


Fig. 4. Illustration of joint spatiotemporal prediction.

Finally, the joint predicted value, X_{jp} is computed by the ME prediction of X from the previous frame, plus the prediction of X using APT from the current frame, and minus the prediction of X using APT from the previous frame. The estimated value X_{jp} and the prediction error *jp*_error can be expressed as follows:

$$\begin{aligned} X_{jp} &= X' + X_{\text{intra}} - X'_{\text{intra}} \\ jp\_\text{error} &= X - X_{jp} \\ &= X - (X' + X_{\text{intra}} - X'_{\text{intra}}) \\ &= (X - X') - (X_{\text{intra}} - X'_{\text{intra}}) \\ &= mc\_\text{error} - pmc\_\text{error}. \end{aligned} \tag{4}$$

In (4), mc_error and pmc_error can be regarded as the prediction error of motion estimation in the actual domain and predicted domain respectively.

All predictive coding strives to make the prediction error as small as possible. Since APT does well on still images and motion compensation does well on videos with regular motion, JPC tries a joint predictive approach that combines the advantages of both. This joint prediction error, jp_error, takes the place of the traditional motion compensated residual mc_error and is supposed to be smaller.

To decode a joint predicted pixel, JPC executes following steps.

1) Predict X with P_x, determine *structure* and get the estimated value X_{intra}.
2) Predict X' with P'_x with the same *structure* in 1) and get the estimated value X'_{intra}.
3) The joint predicted value, $X_{jp} = X' + X_{\text{intra}} - X'_{\text{intra}}$
4) The decoded value, $X_{\text{dec}} = X_{jp} + jp\_\text{error}$.

B. Mode Configuration

Different prediction methods fit different cases. Although joint prediction is more precise than motion estimation in many cases, however, for some sequences with very regular or fast

TABLE I
EFFECT OF DISABLING EACH MODE

Sequence	No *Skip*	No *MC*	No *Intra*
Foreman	2.77%	7.86%	41.38%
News	21.8%	131.67%	10.39%
Bus	0.12%	12.17%	15.56%
Football	2.04%	-5.77%	55.27%
Stefan	3.16%	13.17%	11.43%
Tempete	2.45%	21.11%	9.99%
Average	5.39%	30.04%	24.00%

object motion, direct compensation or intraprediction does even better. Multiple mode prediction is necessary to make good compression. JPC considers four candidate modes aiming at different features for blocks of video frames.

Skip mode: values of pixels in such blocks are copied from pixels in the same position in reference frame directly.

Motion Compensation (MC) mode: pixels in such blocks are predicted as X' in the reference frame. It is the same with the intermode in traditional coding.

Joint Prediction mode: pixels in such blocks are predicted as X_{jp}, which is introduced in Section III-A. Experiments show this mode covers most of the images in video sequences.

Intra mode: pixels in such blocks are predicted as X_{intra}, which is acquired by APT prediction.

To investigate the necessity of the candidate modes besides *Joint Prediction* mode, different kinds of video sequences were tested at various quality parameters. Each mode was disabled and the increase in the sum of absolute difference (SAD) of the prediction was calculated respectively. Table I shows the SAD increase on average caused by each mode disabling. It shows the three modes have significant effect on the prediction accuracy, and, thus, all of them are brought into the system.

With the help of mode selection, the prediction error of JPC gets 24.6% smaller than the traditional inter/intraprediction residual.

C. "Despeckling" Filter Design

The calculation of Peak Signal-to-Noise ratio (PSNR) of the reconstructed video to the original is widely used to objectively determine video quality. However, because of the characteristics of the human visual system, PSNR does not correlate sometimes with perceived quality. Section IV.C will present comparisons between JPC and H.264 at the equivalent PSNRs where JPC shows additional detail without blocking artifacts associated with H.264. However, like all interpolative prediction based coding, JPC has its own problem: speckling artifacts which is typified in the example shown in Fig. 5(a). When compressed at low bit-rate, the noise would be obvious and has to be removed. Therefore, a "despeckling" filter is proposed.

Because of the similarity of speckling artifact and salt-and-pepper noise, order-statistic filtering [14] is chosen as a basic solution in both video and image coding. Each pixel is processed as follows:

(a)

(b)

Fig. 5. Effect of the "despeckling" filter on the tenth frame of FOREMAN. (a) Speckling artifact caused by original JPC. $\mathrm{PSNR} = 32.9$ dB. (b) Result of JPC compression with the "despeckling" filter. $\mathrm{PSNR} = 32.9$ dB.

Sort the values in the 3×3 neighborhood. If the value of the central pixel is higher/lower than the kth highest/lowest value, it is changed into the value of the kth value. Note that k is an empirical parameter which is set to 3 or 4 adaptively in APT [15].

Studies on a number of video cases show that the lack of correlation between content on successive frames indicated by noisy pixels is the key problem to visual quality. To improve the continuity in successive frames, JPC processes the 3×3 neighborhood of the target pixel along with the 3×3 neighborhood of its reference pixel. Thus, 18 pixels are sorted, followed by regular operations.

There are three different filtering methods. A prefilter processes the image before the actual coding; a postfilter processes it during the reconstruction; or thirdly, a loop-filter which not only processes the reconstructed image but also replaces the original frame with the filtered frame as a reference. The three methods were tested experimentally and loop filtering proved superior to the others. Experimental results showed that loop filtering not only improves the subjective quality, but also significantly improves the objective quality by 0.5–1 dB in many cases. Because of the experimental successes, we opted for the loop-filter design. As shown in Fig. 5(b), the "despeckling" filter, although still preliminary, removes most of the noise.

D. Procedure of Joint Predictive Coding

1) Preparation: First, the JPC codec computes the quantization intervals based upon a user-selectable quality parameter, which is derived in APT system [2]. Then, it reads YUV data from video sequence in 4:4:4 format (because KLT requires the same sample interval on all components). Each frame is extended by copying the border pixels into the extended region for the following motion estimation and prediction.

2) Color space Transformation: For video frames whose quality parameter is below an empirical threshold, Principal Component Analysis (PCA), i.e., KLT, is used to achieve even greater compression. Each pixel in current frame is added as a 3-D sample for the covariance matrix, whose eigenvalues and eigenvectors are then calculated. Based on the eigenvalues and eigenvectors, the matrix is diagonalized to determine the basis of the PCA color space. The eigenvector with the highest eigenvalue is the principal component of the data set.

Both the current and reference frame are transformed into the new color space in which the first component of the current frame has the highest energy. Thus, motion estimation and *structure* determination are performed on the first component. The PCA color space transformation is used in other image coding schemes [16]. However, JPC does motion estimation and joint prediction on the image-dependent PCA color space instead of the traditional YUV color space, which achieves higher level of compression.

3) Motion Estimation: The JPC encoder does a half-pel precision motion vector search for every 16×16 block based on the primary component of the current frame. After this stage, each pixel has a motion vector (mv_x, mv_y).

4) Mode Selection: JPC has four modes for blocks in video sequences of different features, *Skip, Motion Compensation, Joint Prediction* and *Intra*.

First, the encoder determines whether to choose *Skip* mode based on the SAD of motion estimation. Then, it prepredicts the current frame in *Motion Compensation, Joint Prediction* and *Intra* mode. The mean square error (MSE) of the three modes in each 16×16 block, $\mathrm{MSE}_{\mathrm{MC}}, \mathrm{MSE}_{\mathrm{JP}}$ and $\mathrm{MSE}_{\mathrm{Intra}}$ are computed and the mode with smallest MSE is selected (considering the tradeoff between MSE and motion vectors, $\mathrm{MSE}_{\mathrm{Intra}}$ is multiplied by an empirical coefficient before the comparison). Mode information and motion vectors are written into the bit-stream at the end of this stage.

5) Predictions and Reconstruction: JPC predicts the three components one by one and goes through all of the pixels in the current frame band by band like all interpolative coders do. X_{pred}, the estimate value, is determined by its position and mode. To initialize the predictors, pixels on the top band (Band ***a***) are estimated to be the referring pixels (X') or the nearest neighbors (N). Assume the sample interval of the top band is L

$$X_{\mathrm{pred}} = \begin{cases} N(x, y-L) & (\mathrm{mode} = \mathrm{intra} \text{ and } x = 0) \\ N(x-L, y) & (\mathrm{mode} = \mathrm{intra} \text{ and } x \neq 0) \\ X' & (\mathrm{mode} \neq \mathrm{intra}). \end{cases} \tag{5}$$

For the rest of bands, each pixel is predicted based on its mode, which is determined in *Step 4)*

$$X_{\mathrm{pred}} = \begin{cases} X' & (\text{Motion Compensation mode}) \\ X_{jp} & (\text{Joint Prediction mode}) \\ X_{\mathrm{intra}} & (\text{Intra mode}). \end{cases} \tag{6}$$

The prediction error, $\mathrm{error}_{\mathrm{pred}} = X - X_{\mathrm{pred}}$, is quantized by the quantization interval computed previously in *Step 1)* and reconstructed to be $\mathrm{error}_{\mathrm{rec}}$. The reconstructed value of $X, X_{\mathrm{rec}} = X_{\mathrm{pred}} + \mathrm{error}_{\mathrm{rec}}$. At the decoder, the output of the pixel X, X_{dec}, will equal this X_{rec}. X_{rec} and X vary because of the quantization of $\mathrm{error}_{\mathrm{pred}}$. Therefore, X_{rec} is used for the downward prediction instead of X.

To save the encoding time, the prediction on the second and the third components does not do mode selection or *structure* determination. Instead, it follows the mode and *structure* on the principal component.

After this stage, three matrixes (three components) of quantized *jp*_errors are formed and written into the stream. The current frame is then reconstructed, transformed to YUV color space, "despeckling" filtered (optional), interpolated by a 6-tap Finite Impulse Response filter [17], and then saved as a reference frame.

6) Reorder and Entropy Coding: The prediction errors are reordered in advance of entropy coding. They are rearranged by the band order so there might be more consecutive zeroes, especially on coarsely quantized bands. This leads to better efficiency for runlength coding. Last, three entropy coders perform ordinary Huffman-runlength coding [14] on the three components independently.

7) Go to the Next Frame.

E. Comparison With Binary Tree Residue Coding

A prior codec that attempts to apply interpolative prediction to video coding is called Binary Tree Residue Coding (BTRC), developed by Robinson, Druet, and Gosset. It simply regards the motion compensated residue as an image and codes it with Binary Tree Predictive Coding (BTPC) [18], a predecessor of APT. However, as residue is not a natural image any more; its intracorrelation gets weaker. Consequently, for this kind of image, APT or BTPC does not perform as well as it does on natural images. In the proposed approach, the prediction is performed only on original pixels. An explanation follows detailing why joint prediction leads to better compression than BTRC.

Let's define a function $\mathrm{Pred}(X)$ that returns the prediction error based upon APT prediction process as described in Section II above. Then, for joint prediction, the error is defined as

$$\begin{aligned} jp\_\mathrm{error} &= (X - X_{\mathrm{intra}}) - (X' - X'_{\mathrm{intra}}) \\ &= \mathrm{Pred}(X) - \mathrm{Pred}(X'). \end{aligned} \tag{7}$$

Alternatively, for comparison the error returned for BTPC is

$$\mathrm{btrc\_error} = \mathrm{Pred}(X - X'). \tag{8}$$

Although extremely efficient on static images, the difficulty encountered with using BTPC on video lies in the nature of how the pixels are predicted. The locations of pixels used for

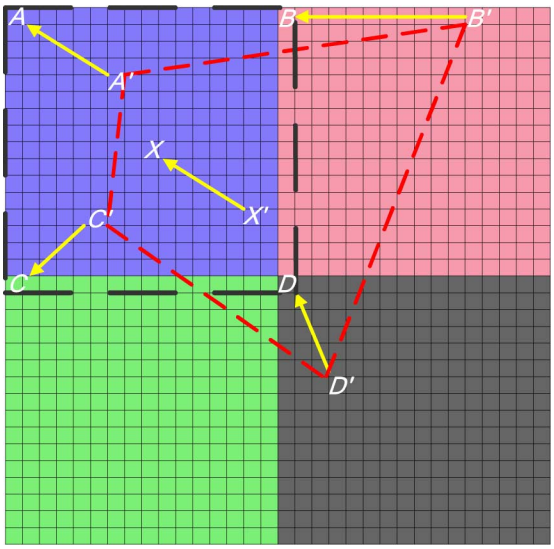


Fig. 6. BTRC difficulty with integrating motion vector predictions.

prediction in one frame may change significantly in successive frames because the pixels may have different motion vectors.

Consider the btrc_error defined above as being comprised of two parts, that is, the Pred() of the current frame minus the prediction of the reference frame. Note that we do not define the Pred_1 of the reference frame the same as the Pred of the current frame

$$\mathrm{Pred}(X - X') = \mathrm{Pred}(X) - \mathrm{Pred\_1}(X'). \tag{9}$$

BTRC reconstructs an image (frame) by recursively using the prediction error on previous encoded pixels. For example, in Fig. 6, the Pixel X is to be reconstructed from the 4 neighboring pixels that form a square, i.e., A, B, C, and D, but in a video, a very likely scenario is that these pixels may have completely different motion vectors. Let A', B', C', and D' be the corresponding pixels in the previous frame. Consequently, the function Pred_1(X') uses four unrelated pixels $A', \quad B', C'$, and D' (they do not form a square) to predict X'. In other words, the facts that differences $A - A', B - B', C - C'$, and $D - D'$ are used to predict $X - X'$, and the pixels A', B', C', D', and X' may be unrelated in the previous frame render the BTRC reconstruction algorithm inefficient. In contrast the proposed JPC uses only the motion vector of pixel X.

We have experimentally demonstrated that joint prediction leads to an average of a 20.5% bit-rate reduction compared to this approach.

Fig. 7 shows the rate-distortion curves of JPC and BTRC on the sequence *News* and *Football*, which verify the advantage of the Joint Predictive method over BTRC. These results are not unexpected. BTRC and its successor APT were not designed with video in mind and their difficulties with video are summarized above. By utilizing the adaptive structure empirical prediction of APT and combining it with the joint prediction methods, we can demonstrate a marked improvement over BTRC/APT alone.

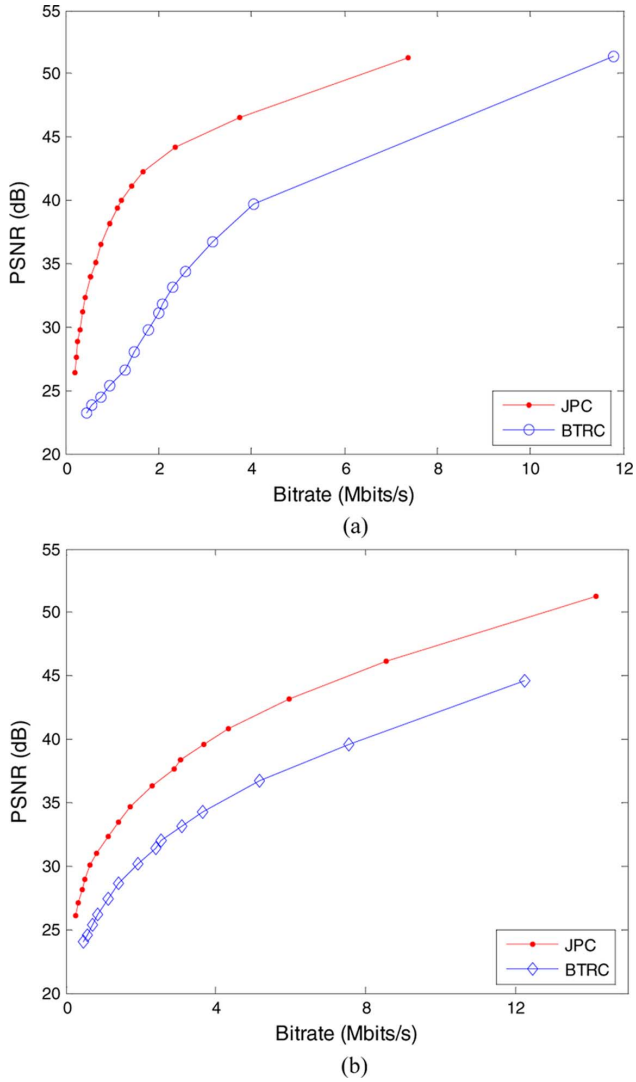


Fig. 7. Comparison of JPC and BTRC.

F. Comparison With Temporal Prediction Trees

Day and Robinson developed another approach to video compression called temporal prediction trees (TPT) [13]. The main idea is to determine temporal/spatial prediction mode for each pixel based on a threshold (calculated by several parameters such as desired quality and motion vectors) and perform an adaptive temporal/spatial coding. Temporal prediction, which simply predicts the current pixel as its corresponding pixel in the reference frame, is the same as the *Motion Compensation* mode in JPC. Spatial prediction uses the APT spatial predictor for the current pixel. Since this approach also separates the spatial and temporal prediction, it is not as precise as the proposed joint spatiotemporal prediction. As can be seen in Fig. 8, TPT is not able to match the performance of JPC. (Luma-weighted PSNR weights the luminance component four times as highly as each of the chrominance components.)

Another reason for the performance difference is the tree structure. APT and TPT use a so called "hex-tree" structure to arrange pixels that efficiently represent large areas of an image with a zero prediction errors. They introduce a "terminator"

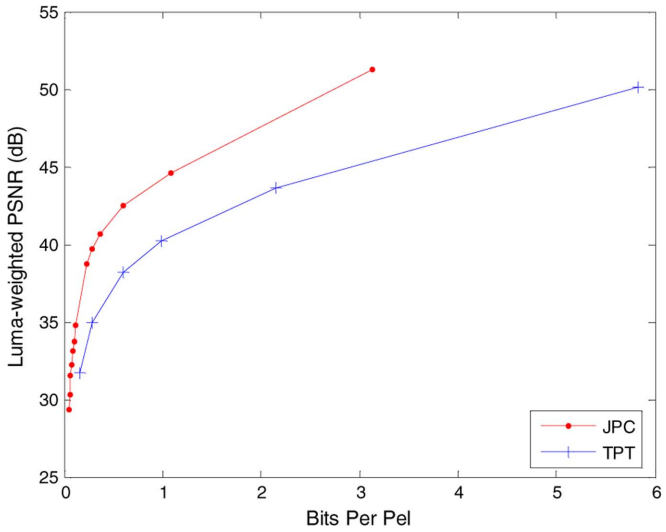


Fig. 8. Comparison of JPC and TPT on the 300-frame sequence Mother and Child (352 × 288).

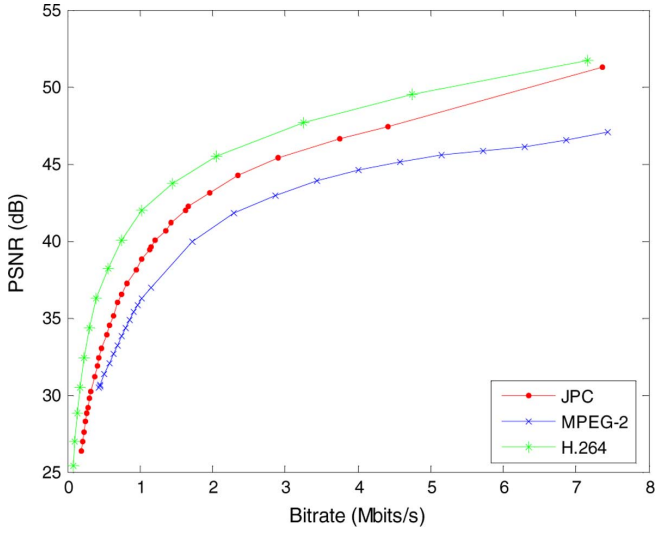


Fig. 9. Comparison of JPC, MPEG-2 and H.264 on the beginning 100 frames of the sequence *News*.

flag to represent the sub-trees whose nodes are all zero values [15]. They use a flag for every nonterminated node indicating whether its prediction error is zero and whether it is a terminator. This makes good result on still image compression. Video sequences, however, have much more redundancy between close frames. Consequently, after motion compensation, there are many more groupings of zeroes in prediction errors. A higher number of consecutive zeroes lend itself better to use alternative methods of compression, including traditional runlength coding. In this case, the tree-flag mechanism leads to inefficiency. As a result, in JPC, the basics of APT prediction (adaptive structures, prediction, etc.) were all used; however, the use of trees was abandoned. Instead we used the runlength coding.

IV. Experimental Result and Comparison

A. Rate-Distortion Performance

JPC is compared with MPEG-2 and H.264 (high 4:4:4 profile) on six 100-frame CIF (352 × 288 30 frame/s) video sequences, *Football, News, Foreman, Stefan, bus,* and *Tempete*. MPEG-2 V1.2 [19] and JM 12.4 [20] reference software is used respectively for MPEG-2 and H.264. A coding structure of IPPPPIPPP... is chosen. We take PSNR in dependency of bit-rate as the rate-distortion performance standard and investigate a large range of performance up to 50 dB PSNR. This may help indicate the possible applications for our new codec. Overall, H.264 proves superior to the other methods; however, JPC shows promising results. Figs. 9–11 present the results of comparative tests on the sequences *News, Foreman* and *Football*. All coders make good compression on the sequence *News* (with low activity). For the sequence *Foreman* (the part with rapid camera motion) and *Football* (with fast object motion), the coding efficiency gets poorer but the relative performances remain the same. It can also be seen that JPC is able to catch up with H.264 when coding videos in extremely high quality. In general, JPC outperforms MPEG-2 in all cases but cannot match H.264 now. Nevertheless, several areas of improvement

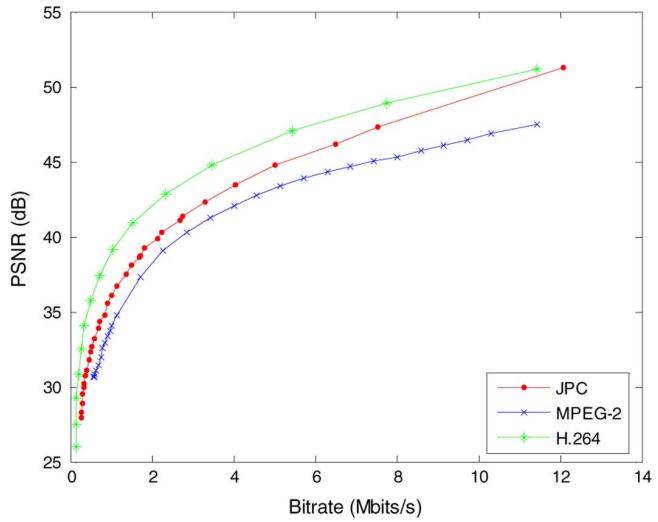


Fig. 10. Comparison of JPC, MPEG-2 and H.264 on the 130th to 230th frames of the sequence *Foreman*.

are already being researched and experimented with, including variable block sizes, multiple frame reference and more precise motion estimation (i.e., quarter pel) and mode selection. Surely, these improvements will definitely result in better compression.

B. Speed

Tables II and III present an overview of the main modules employed by JPC that differ from the corresponding modules for H.264 and MPEG-2. Modules that are common to all three schemes (e.g., entropy coding) are not presented in those tables.

The JPC source code has not been optimized; therefore, it is disadvantaged in speed comparisons to other coding schemes. We do, however, present the timing information for informational purposes. MPEG-2 V1.2 [19] and JM 12.4 [20] reference software are used for MPEG-2 and H.264 codecs, respectively. The test was done on a PC with Pentium 4 2.6-GHz CPU and

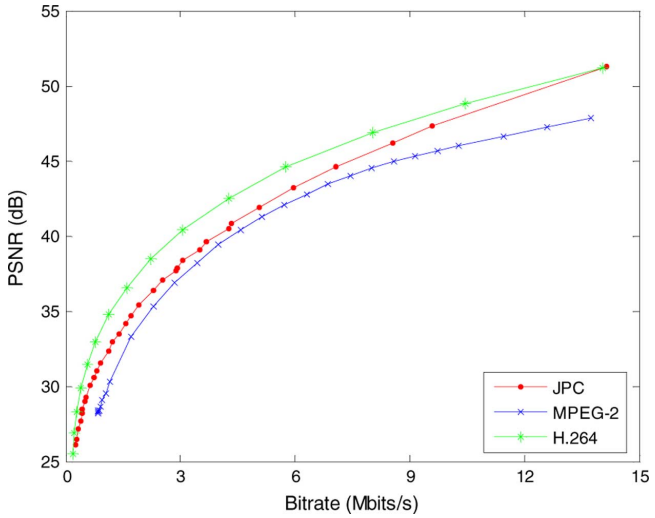


Fig. 11. Comparison of JPC, MPEG-2 and H.264 on the beginning 100 frames of the sequence *Football*.

TABLE II
MODULES FOR ENCODING COMPLEXITY COMPARISON

JPC encoder	H.264 encoder	MPEG-2 encoder
Color space transform	H.264 motion estimation	MPEG-2 motion estimation
JPC motion vector search	H.264 mode Selection	MPEG-2 mode selection
JPC mode selection	Integer DCT	DCT
Prediction and Quantization	Quantization	Quantization

TABLE III
MODULES FOR DECODING COMPLEXITY COMPARISON

JPC decoder	H.264 decoder	MPEG-2 decoder
Inverse Quantization	Inverse Quantization	Inverse Quantization
Prediction and Reconstruction	Inverse integer DCT	Integer DCT
Inverse Color space transform	Reconstruction	Reconstruction

512-M memory. All the three coders perform full search for motion estimation and the filters are disabled.

As can be seen from the timing data in Tables IV and V, the good performance of the nonoptimized JPC codec demonstrates that traditional coding schemes can be further refined.

We tested six video sequences on a full range of quality parameters and recorded the average time cost of each sequence. Table IV shows the encoding time cost for I frame and P frame respectively while Table V shows the decoding time cost of the three codecs on the modules listed above. For encoding, JPC is 57.6% faster than H.264 while 39.9% slower than MPEG-2; for decoding, it is 7.9% faster than H.264 and 18.0% faster than

TABLE IV
AVERAGE TIME COST OF ENCODING ONE FRAME (MS)

Encoder	JPC		H.264		MPEG-2	
type	I	P	I	P	I	P
Foreman	37.2	258.0	75.9	621.3	75.9	189.7
News	37.9	186.3	76.2	446.8	73.9	125.9
Bus	37.6	311.1	76.4	661.2	81.1	205.4
Football	37.6	280.6	77.5	745.2	75.4	207.2
Stefan	38.0	269.6	78.4	617.9	73.8	171.9
Tempete	38.8	259.2	76.1	627.2	72.0	151.7

TABLE V
AVERAGE TIME COST OF DECODING ONE FRAME (MS)

Decoder	JPC	H.264	MPEG-2
Foreman	24.3	32.4	63.1
News	29.3	22.9	61.3
Bus	28.0	35.9	61.8
Football	29.1	28.2	61.5
Stefan	28.7	37.6	62.4
Tempete	29.3	31.9	62.7

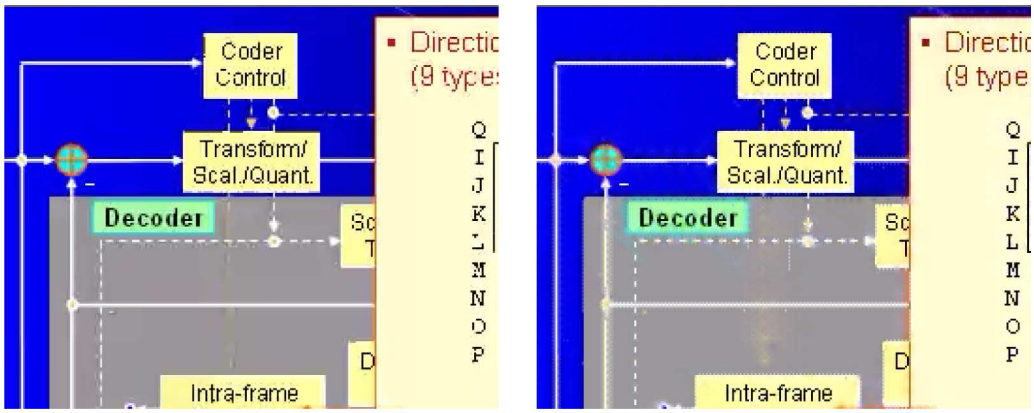
MPEG-2. It is clear that the computational complexity of JPC is acceptable. The methods presented in this paper show promise for practical applications and future enhancements will enhance the practicality even further. The current implementation has some inefficiencies in the entropy coding process. JPC entropy coder goes through all the prediction errors twice, once to collect statistics and once to code the values. For better efficiency, the entropy code table for prediction errors should be designed in advance. The encoder would then go through the values only once, which is the case in MPEG and H.26x series implementations.

C. Subjective Quality

Although the compression efficiency of JPC is currently worse than H.264, it leads to much better subjective quality in some specific domains. After conducting a large number of comparative tests on various types of video, we conclude that both JPC and H.264 each have each their strong points in subjective quality. However, we find that JPC shows significant advantage over H.264 on coding videos with text as well as videos with complicated textures and irregular contours.

Fig. 12 presents the result on one reconstructed frame of a video of a slide presentation. Fig. 12(a) is reconstructed by H.264, with the luma-weighted PSNR 31.1 dB. Fig. 12(b) is reconstructed by JPC, with the luma-weighted PSNR 31.3 dB. It shows that JPC preserves the texts well in compressing slides while H.264 fails to keep good quality. Fig. 13 shows H.264 leads to obvious blocking artifact on videos with complicated textures and irregular contours, while JPC does much better in subjective quality.

The main problem of JPC is that it generates temporal noise (speckling artifacts) during high compression, which makes it



(a) (b)

Fig. 12. Subjective quality comparison of H.264 and JPC on a video of slides.

(a) (b)

Fig. 13. Subjective quality comparison of H.264 and JPC on a landscape video.

under perform H.264 in some cases. We note that H.264 uses sophisticated filters while the current JPC implementation has only recently been implemented and has not been optimized fully. However, the filter shows promising result on removing the noise. It shows at least that the speckling artifact problem is not unsolvable. Because of the principal advantage over the block-based coding, JPC is very likely to be able to obtain better subjective quality in many domains.

V. Conclusion

The good performance of the proposed joint predictive coding shows that joint spatiotemporal prediction is a serious competitor to the current block based video coding techniques. With prediction error reduced by 24.6% on average compared to the traditional inter/intraprediction approach, joint spatiotemporal prediction is obviously superior to the conventional inter/intraprediction. JPC not only introduces a new prediction method but also proposes a new KLT/Joint Prediction framework to replace the traditional ME/DCT mechanism. This opens up a new, very promising direction in video coding. With the help of joint prediction technique, JPC performs much better than MPEG-2 using the equivalent ME technique. It is at least a promising alternative in related applications such as DVD and digital satellite broadcasting. After further research, specifically in better filtering technique, JPC has a strong potential to become a very competitive video coding scheme.

Appendix
Theoretical Analysis of Joint Prediction

The conventional way of prediction is to use motion estimation to find the best matching block in a reference frame and take the corresponding pixel as the estimate of current one. However, this is not enough for the cases where temporal correlation is limited. Joint spatiotemporal prediction is a method that utilizes both temporal and spatial correlation and usually yields a better estimated value. To prove that the joint prediction is a better approach than motion compensation, we compare the variances of joint predicted error and motion compensated residue.

Assume that $D[jp\_\text{error}]$ is the variance of joint predicted error and $D[mc\_\text{error}]$ is the variance of motion compensated residue. $E[X]$ presents the expected value of the X. Motion compensated residue is recognized to follow a zero-mean Laplacian distribution [21], i.e., $E[mc\_\text{error}] = 0$. Assume P_i is one of the pixels used for the prediction of X and k_i is its weight. Then the APT estimated value X_{intra} of X is computed as follows:

$$X_{\text{intra}} = \sum_{i=1}^{n} k_i P_i.$$

According to the rules of APT prediction

$$\sum_{i=1}^{n} k_i = 1(\mathrm{n} = 1, 2, 4)$$

and

$$\begin{aligned} pmc\_\mathrm{error} &= X_{\mathrm{intra}} - X'_{\mathrm{intra}} \\ &= \sum_{i=1}^{n} k_i P_i - \sum_{i=1}^{n} k_i P'_i \\ &= \sum_{i=1}^{n} k_i (P_i - P'_i) \\ &= \sum_{i=1}^{n} k_i mc\_\mathrm{error} \\ \therefore E[mc\_\mathrm{error}] &= E[pmc\_\mathrm{error}] \\ D[mc\_\mathrm{error}] &= E[mc\_\mathrm{error}^2] - E^2[mc\_\mathrm{error}] \\ &= E[mc\_\mathrm{error}^2]. \end{aligned}$$

Similarly, $D[pmc\_\mathrm{error}] = E[pmc\_\mathrm{error}^2]$.

For $n = 2$

$$\begin{aligned} &pmc\_\mathrm{error} \\ &\quad = k_1 \cdot mc\_\mathrm{error}_1 \\ &\qquad + k_2 \cdot mc\_\mathrm{error}_2 \quad (k_1 + k_2 = 1) \\ &E[pmc\_\mathrm{error}^2] - E[mc\_\mathrm{error}^2] \\ &\quad = E[(k_1 \cdot mc\_\mathrm{error}_1 + k_2 \cdot mc\_\mathrm{error}_2)^2] \\ &\qquad - 1 \cdot E[mc\_\mathrm{error}^2] \\ &\quad = (k_1^2 + k_2^2) E[mc\_\mathrm{error}^2] \\ &\qquad + 2k_1 k_2 E[mc\_\mathrm{error}_1 \cdot mc\_\mathrm{error}_2] \\ &\qquad - (k_1 + k_2)^2 E[mc\_\mathrm{error}^2] \\ &\quad = 2k_1 k_2 (E[mc\_\mathrm{error}_1 \\ &\qquad \cdot mc\_\mathrm{error}_2] - E[mc\_\mathrm{error}^2]). \end{aligned}$$

Since $mc\_\mathrm{error}_1, mc\_\mathrm{error}_2$ and $mc\_$ error follow the same Laplacian distribution. By Schwartz inequality, we obtain $E[mc\_\mathrm{error}_1 \cdot mc\_\mathrm{error}_2] \leq E[mc\_\mathrm{error}^2]$

$$E[pmc\_\mathrm{error}^2] - E[mc\_\mathrm{error}^2] \leq 0.$$

For $n = 4$,

$E[pmc\_\mathrm{error}^2] = E[(\sum_{i=1}^{n} k_i mc\_\mathrm{error})^2] \leq E[mc\_\mathrm{error}^2]$ can be obtained by mathematical induction

$$\begin{aligned} &D[jp\_\mathrm{error}] - D[mc\_\mathrm{error}] \\ &\quad = E[(mc\_\mathrm{error} - pmc\_\mathrm{error})^2] \\ &\qquad - (E[mc\_\mathrm{error} - pmc\_\mathrm{error}])^2 - E[mc\_\mathrm{error}^2] \\ &\quad = E[pmc\_\mathrm{error}^2] \\ &\qquad - 2E[mc\_\mathrm{error} \cdot pmc\_\mathrm{error}]. \end{aligned}$$

Define ρ as the correlation coefficient of mc_error and pmc_error (see the equation shown at the bottom of the page). With the above substitution, we obtain

$$\begin{aligned} &D[jp\_\mathrm{error}] - D[mc\_\mathrm{error}] \\ &\quad = E[pmc\_\mathrm{error}^2] \\ &\qquad - 2\rho \cdot \sqrt{E[mc\_\mathrm{error}^2]} \cdot \sqrt{E[pmc\_\mathrm{error}^2]} \\ &\quad = \sqrt{E[pmc\_\mathrm{error}^2]} \\ &\qquad \cdot (\sqrt{E[pmc\_\mathrm{error}^2]} - 2\rho \cdot \sqrt{E[mc\_\mathrm{error}^2]}) \\ &\quad (\text{As proved, } E[pmc\_\mathrm{error}^2] \leq E[mc\_\mathrm{error}^2]) \\ &\quad \leq \sqrt{E[pmc\_\mathrm{error}^2]} \\ &\qquad \cdot (\sqrt{E[mc\_\mathrm{error}^2]} - 2\rho \cdot \sqrt{E[mc\_\mathrm{error}^2]}) \\ &\quad = (1 - 2\rho) \cdot \sqrt{E[pmc\_\mathrm{error}^2]} \cdot \sqrt{E[mc\_\mathrm{error}^2]} \\ &\rho > 0.5 \Rightarrow D[jp\_\mathrm{error}] < D[mc\_\mathrm{error}]. \end{aligned}$$

The above derivation proves the following theorem:

Theorem: As long as the correlation coefficient of mc_error and pmc_error, $\rho > 0.5$, it holds that $D[jp\_\mathrm{error}] < D[mc\_\mathrm{error}]$, i.e., the variance of joint predicted error is smaller than the variance of motion compensated residue.

Since $pmc\_\mathrm{error} = \sum_{i=1}^{n} k_i mc\_\mathrm{error}_i$ and the intraprediction of APT is accurate, it is expected that mc_error and pmc_error are highly correlated, which means that the assumption of $\rho > 0.5$ should be satisfied in most cases.

We investigated eight 100-frame video sequences from static to dynamic for correlation of mc_error and pmc_error. The values of ρ are computed empirically from each couple of successive frames and the statistical distribution is shown in Fig. 14. It can be observed that the value of ρ is seldom smaller than 0.5 in either low activity sequence *News* or very dynamic sequence *Football*. As a result, $D[jp\_\mathrm{error}] < D[mc\_\mathrm{error}]$

$$\begin{aligned} \rho &= \frac{E[mc\_\mathrm{error} \cdot pmc\_\mathrm{error}] - E[mc\_\mathrm{error}] \cdot E[pmc\_\mathrm{error}]}{\sqrt{D[mc\_\mathrm{error}]} \cdot \sqrt{D[pmc\_\mathrm{error}]}} \\ &= \frac{E[mc\_\mathrm{error} \cdot pmc\_\mathrm{error}]}{\sqrt{D[mc\_\mathrm{error}]} \cdot \sqrt{D[pmc\_\mathrm{error}]}} \\ \therefore E[mc\_\mathrm{error} \cdot pmc\_\mathrm{error}] &= \rho \sqrt{D[mc\_\mathrm{error}]} \sqrt{D[pmc\_\mathrm{error}]} \end{aligned}$$

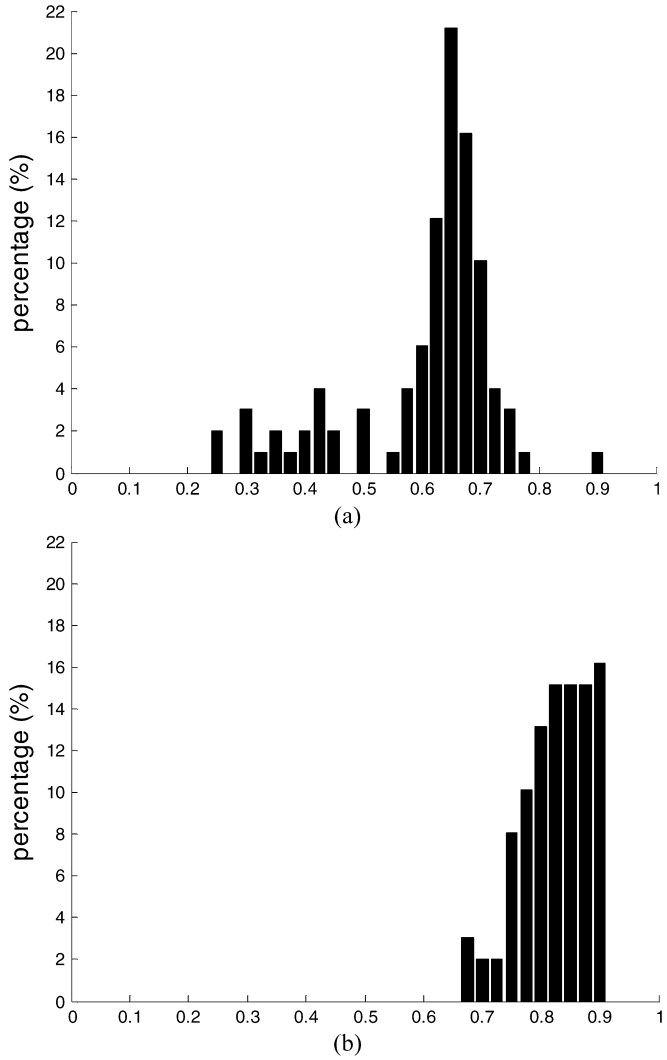


Fig. 14. Distribution of ρ of 100 frames in sequence News and Football.

is true in most cases. Therefore, joint prediction makes better compression than motion compensation for majority of videos.

REFERENCES

[1] W. Pu, Y. Lu, and F. Wu, "Joint power-distortion optimization on devices with MPEG-4 AVC/H.264 codec," in *Proc. IEEE Int. Conf. Communications*, June 2006, vol. 1, pp. 441–446.

[2] J. A. Robinson, "Adaptive prediction trees for image compression," *IEEE Trans. Image Process.*, vol. 15, no. 8, pp. 2131–2145, Aug. 2006.

[3] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. IEEE*, vol. 83, no. 6, pp. 858–876, Jun. 1995.

[4] B. R. Hunt, "Optical computing for image bandwidth compression: Analysis and simulation," *Appl. Opt.*, vol. 15, pp. 2944–2951, Sept. 1978.

[5] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Commun.*, vol. COM-3l, no. 4, Apr. 1983.

[6] W. Woods and S. D. O'Neil, "Subband coding of images," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 5, pp. 1278–1288, Oct. 1986.

[7] P. Roos, M. A. Viergever, M. C. A. can Dijke, and J. H. Peters, "Reversible intraframe coding of medical images," *IEEE Trans. Med. Imag.*, vol. 7, no. 12, pp. 328–336, Dec. 1988.

[8] L. Arnold, "Interpolative coding of images with temporally increasing resolution," *Signal Process.*, vol. 17, pp. 151–160, 1989.

[9] P. G. Howard and J. S. Vitter, "New methods for lossless image compression using arithmetic coding," *Inf. Process. Manag.*, vol. 28, no. 6, pp. 765–779, 1992.

[10] P. G. Howard and J. S. Vitter, "Fast progressive lossless image compression," presented at the Data Comp. Conf., Snowbird, UT, Mar. 1994.

[11] E. A. Gifford, B. R. Hunt, and M. W. Marcellin, "Image coding using adaptive recursive interpolative DPCM," *IEEE Trans. Image Process.*, vol. 4, no. 8, pp. 1061–1069, Aug. 1995.

[12] J. A. Robinson, A. Druet, and N. Gosset, "Video compression with binary tree recursive motion estimation and binary tree residue coding," *IEEE Trans. Image Process.*, vol. 9, no. 7, Jul. 2000.

[13] M. G. Day and J. A. Robinson, "Residue-free video coding with pixel-wise adaptive spatio-temporal prediction," *IET Image Process.*, vol. 2, no. 3, pp. 131–138, Jun. 2008.

[14] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2005, Section 3.6.2 and Section 8.4.

[15] *APT Online Reference Code*, [Online]. Available: http://www.intuac.com/userport/john/apt/index.html, [Online]

[16] C. L. Yang, L. M. Po, D. H. Cheung, and K. W. Cheung, "A novel ordered-SPIHT for embedded color image coding," in *Proc. IEEE Int. Conf. Neural Networks and Signal Processing*, Nanjing, China, Dec. 14–17, 2003, pp. 1087–1090.

[17] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[18] J. A. Robinson, "Efficient general-purpose image compression with binary tree predictive coding," *IEEE Trans. Image Process.*, vol. 6, no. 4, pp. 601–608, Apr. 1997.

[19] *MPEG-2 Reference Software*, [Online]. Available: http://www.mpeg.org/MSSG, MPEG-2 version 1.2, MPEG Software Simulation Group.

[20] *JM Software, JM 12.4, H.264/AVC Software Coordination*, [Online]. Available: http://iphome.hhi.de/suehring/tml/

[21] I.-M. Pao and M.-T. Sun, "Modeling DCT coefficients for fast video encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 4, pp. 608–616, Jun. 1999.

Wenfei Jiang received the B.S. degree in telecommunication engineering in 2007 and the M.S. degree in communication and information systems in 2009 from the Huazhong University of Science and Technology, Wuhan, China.

His research interests include image processing and video coding.

Longin Jan Latecki (M'02–SM'07) received the Ph.D. degree in computer science from the Hamburg University, Germany, in 1992.

He has published over 160 research papers and books. He is an Associate Professor of computer science at Temple University, Philadelphia, PA. His main research areas are shape representation and similarity, robot perception, and digital geometry and topology.

Dr. Latecki is the winner of the Pattern Recognition Society Award with A. Rosenfeld for the best article published in Pattern Recognition in 1998. He received the main annual award from the German Society for Pattern Recognition (DAGM), the 2000 Olympus Prize. He is an Editorial Board Member of *Pattern Recognition*.

Wenyu Liu (M'08) received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 1986, and the M.S. and Ph.D. degrees in electronics and information engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively.

He is now a Professor and Associate Dean of the Department of Electronics and Information Engineering, HUST. His current research areas include multimedia information processing, and computer vision.

Hui Liang received the B.S. degree in electronics and information engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2008. He is currently pursuing the M.S. degree in the Department of Electronics and Information Engineering, HUST.

His research interests include image compression and wireless communication.

Ken Gorman received the B.S.E.E. degree from Villanova University and the M.S. degree in computer science from Temple University, Philadelphia, PA, where he is pursuing the Ph.D. degree.

He is a Systems Staff Engineer for Honeywell International. His research interests include imaging, video streaming and compression.