JOINT LEARNING FOR SIDE INFORMATION AND CORRELATION MODEL BASED ON LINEAR REGRESSION MODEL IN DISTRIBUTED VIDEO CODING

Xianming Liu[1], Debin Zhao[1], Yongbing Zhang[1], Siwei Ma[2],Qingming Huang[3],Wen Gao[2]
[1]Department of Computer Science and Technology, Harbin Institute of Technology, Harbin,150001, P.R. China
[2]Institute of Digital Media, Peking University, Beijing, 100871, P.R. China
[3]Graduate University, Chinese Academy of Science, Beijing, 100039, P.R. China

ABSTRACT

The coding efficiency of distributed video coding system is significantly determined by the side information quality and correlation model. Motivated by theoretical analysis of the maximum likelihood treatment for linear regression model, we propose a novel joint online learning model for side information generation and correlation model estimation in this paper. In our proposed scheme, each pixel in the side information is approximated as the linear weighted combination of samples within a local spatio-temporal neighboring space. Weights are trained in a self-feedback fashion, during which the correlation model parameters can also be achieved. The efficiency of the proposed joint learning model is confirmed experimentally.

Index Terms— Side information, correlation model, linear regression model, joint learning, distributed video coding

1. INTRODUCTION

Distributed video coding (DVC) is a new video coding paradigm which can shift the complexity from encoder to decoder. Because of its potential for several emerging applications such as wireless video surveillance and mobile camera phones, DVC has been receiving more and more attention in recent years. Many practical DVC schemes share a general architecture [1]. Typically, the encoder applies the turbo encoding on each Wyner-Ziv (WZ) frame, in which parity bits are generated. To achieve compression, only parts of these parity bits are sent. By exploiting the video correlation partially or totally, the decoder construct an estimation of the current WZ frame named side information (SI), which can be viewed as a noisy version of the current frame. Correlation model converts the SI into soft-input information needed for turbo decoding. The turbo decoder combines the received parity bits and soft-input information to decode the current frame.

From the process stated above we can see clearly that the performance of DVC system depends on two factors: the first one is the quality of SI, which relies considerably on the motion modeling between successive frames; the second one is the accuracy of correlation model estimated. Both are quite tough since original frames are not available at the decoder and the statistics of video source are dynamically varying in spatial and temporal domain.

For SI estimation, one of the most popular methods in the literature of DVC is motion compensated temporal interpolation (MCTI) based on block-matching algorithm (BMA) [1]. For correlation model parameters (CMP) estimation, the most popular approach is to use some estimated reliability information regarding the obtained motion vector based on BMA in the SI generation process [2]. BMA plays a key role in estimating SI and CMP in these fashions. The apparent advantage of BMA is its conceptual simplicity, and it can reflect some relationship between motion and estimated intensity values. However, since the original frames are not available at the decoder, BMA may not be effective locally, which usually results in some errors in estimated SI and CMP. To overcome this problem, some locally accurate motion models are introduced, e.g. the filter-based fashion [3]. Such localized estimation can be viewed as an implicit approach of exploiting motion-related temporal dependency, in which motion information is embedded into predictive coefficients.

In this paper, we propose a novel joint online learning strategy for SI and CMP based on linear regression model, which is performed at the decoder at pixel level. Model weights and CMP are trained in a self-feedback fashion, during which reconstructed pixels in neighboring spatio-temporal space are utilized as samples. Our method represents an important departure from previous work in the literature and provides a more efficient WZ video coding solution.

The rest of this paper is organized as follows. In Section 2, we give a theoretical analysis of the maximum likelihood treatment for linear regression model, which can bring us some motivation for jointly learning strategy. We describe our proposed model in detail in Section 3. In Section 4, the experimental results are presented to show the efficiency of our approach and Section 5 concludes this paper.

2. MAXIMUM LIKELIHOOD TREATMENT FOR LINEAR REGRESSION MODEL

Suppose $\{X(k_1, k_2, k_3)\}$ is the given video sequence,where $(k_1, k_2) \in [1, H] \times [1, W]$ are spatial coordinates and k_3 is the frame index. We denote the position of a pixel in SI by a vector $n_0 = (k_1, k_2, k_3)$. Linear regression model for pixel interpolation is formulated as:

$$y(n_0, \mathrm{w}) = \sum_{i=1}^{M} w_i \phi_i(n_0) = \mathrm{w}^T \phi(n_0), \tag{1}$$

where M is the model order which represents the number of samples used in interpolation, $\mathrm{w} = (w_1, \cdots, w_M)^T$ is the weight vector learned within a local training window sized of N, $\phi(n_0)$ denotes the intensity vector of neighboring samples of n_0. We assume the target intensity value of n_0 is

$$t = y(n_0, \mathrm{w}) + \varepsilon, \tag{2}$$

where ε is Gaussian noise with precision (inverse variance) β. In view of probability theory, t is a random variable satisfied Gaussian distribution and can be formulated as

$$p(t \mid n_0, \mathrm{w}, \beta) = \mathbb{N}(t \mid y(n_0, \mathrm{w}), \beta^{-1}). \tag{3}$$

Now consider a pixel set of input $X = \{n_0, \cdots, n_{N-1}\}$ with corresponding intensity values $T = \{t_0, \cdots, t_{N-1}\}$, which are in the same training window. Making the assumption that these pixels are drawn independently from the distribution (3), we obtain the following expression for the likelihood function:

$$p(T \mid X, \mathrm{w}, \beta) = \prod_{i=0}^{N-1} \mathbb{N}(t_i \mid \mathrm{w}^T \phi(n_i), \beta^{-1}). \tag{4}$$

Taking the logarithm of the likelihood function, we have

$$\begin{aligned} \ln p(T \mid X, \mathrm{w}, \beta) &= \sum_{i=0}^{N-1} \ln \mathbb{N}(t_i \mid \mathrm{w}^T \phi(n_i), \beta^{-1}) \\ &= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathrm{w}) \end{aligned} \tag{5}$$

where the sum-of-squares error function is defined by

$$E_D(\mathrm{w}) = \frac{1}{2} \sum_{i=0}^{N-1} \left\{ t_i - \mathrm{w}^T \phi(n_i) \right\}^2 \tag{6}$$

Then we can use maximum likelihood to determine w and β. The gradient of the log likelihood function (5) takes the form

$$\begin{cases} \dfrac{\partial \ln p(T \mid X, \mathrm{w}, \beta)}{\partial \mathrm{w}} = \sum_{i=0}^{N-1} \{t_i - \mathrm{w}^T \phi(n_i)\} \phi(n_i)^T = 0 \\ \dfrac{\partial \ln p(T \mid X, \mathrm{w}, \beta)}{\partial \beta} = \dfrac{N}{2\beta} - E_D(\mathrm{w}) = 0 \end{cases} \tag{7}$$

Solving for w and β we obtain

$$\mathrm{w} = \left(\Phi^T \Phi \right)^{-1} \Phi^T T \tag{8}$$

$$\frac{1}{\beta} = \sigma^2 = \frac{1}{N} \sum_{i=0}^{N-1} \left\{ t_i - \mathrm{w}^T \phi(n_i) \right\}^2 \tag{9}$$

where Φ is a pixel matrix sized of $N \times M$. From (8), we can see that the maximum likelihood treatment is identical with the least square solution for linear regression model [3]. From (8) and (9), we find model weights and correlation model parameters for each pixel can be estimated in the same process, which motivated the proposed joint learning model.

3. THE PROPOSED JOINT LEARNING MODEL

3.1. Model Parameters Selection

In the proposed model, the intensity value of each pixel in SI is approximated as the linear weighted combination of samples within its neighboring space, as described in (10).

$$\widehat{X}(k_1, k_2) = \sum_{i=1}^{M} w_i \phi(k_1, k_2) \tag{10}$$

where $\widehat{X}(k_1, k_2)$ stands for the interpolated value at location (k_1, k_2) in SI. w denote model weights which are trained in a local window, the size of which is set to 16×16 for QCIF sequence in our experiment. $\phi(k_1, k_2)$ are neighboring samples of the pixel at (k_1, k_2), which include not only the pixels within its two temporal neighborhoods in the forward and the backward key frames but also the available interpolated pixels within its spatial neighborhood in the current frame in the proposed model. This is different from the setting in [5]. Each temporal neighborhood is specified as a $(2L+1) \times (2L+1)$ region bounded by $(k_1 \pm L, k_2 \pm L)$, where L is the spatio-temporal order. Consequently, the spatial neighborhood is specified as a region with the size of $\lfloor (2L+1) \times (2L+1)/2 \rfloor$. Thus the model order M is:

$$M = 2 \times (2L+1) \times (2L+1) + \lfloor (2L+1) \times (2L+1)/2 \rfloor. \tag{11}$$

Fig.1 illustrates the case when $L = 1$.

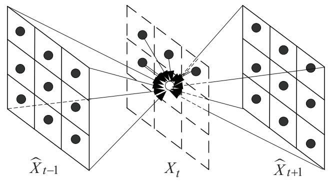

Fig.1 Side information interpolation with $L = 1$

It should be pointed out that the spatio-temporal order is closely related to the motion in the training window. Smaller spatio-temporal order will achieve good performance for stationary regions; however, for moving regions, larger spatio-temporal order is necessary. In our experiment, the motion vector in MCTI can be utilized to measure the motion level of the training window. The spatio-temporal order of the model is computed by

$$L = \max_{block_i \in S} \left\{ abs\left(\left\lfloor mvx_{block_i} \right\rfloor \right), abs\left(\left\lfloor mvy_{block_i} \right\rfloor \right) \right\} + 1 \tag{12}$$

where S represents the training window, mvx_{block_i} and mvy_{block_i} represent the horizontal and vertical motion vectors of the ith 8×8 block after performing MCTI.

3.2. Model Weight Training

In order to derive more accurate model weights, we group five successive frames as an interpolation unit (IU) and assume weights remain the same in each IU. An iterative self-feedback method proposed in our previous work is utilized for training [5]. Each iteration consists of two stages. In the first stage, we use the reconstructed key frames X_{t-2}, X_t, X_{t+2} to interpolate side information $\widehat{X}_{t-1}$ and $\widehat{X}_{t+1}$, as illustrated in Fig.2. In the second stage, as shown in Fig.1, the key frame X_t is re-interpolated by the side information $\widehat{X}_{t-1}$ and $\widehat{X}_{t+1}$ generated in the first stage. The weights for each training window are calculated by jointly minimizing the distortion between the interpolated frames and initial frames, which is formulated as follows.

$$\vec{W}_{\text{model}} = \arg \min_{\vec{W}, \hat{X}_{t-1}, \hat{X}_t, \hat{X}_{t+1}} \left\{ \left\| X_{t-1} - \hat{X}_{t-1} \right\|^2 + \left\| X_t - \hat{X}_t \right\|^2 + \left\| X_{t+1} - \hat{X}_{t+1} \right\|^2 \right\} \quad (13)$$

where X_t and $\widehat{X}_t$ are training windows in initial frame and interpolated frame on time t respectively. For key frames, we regard the H.264 intra decoded frames as the initial ones. $X_{t-1}, \widehat{X}_{t-1}$ and $X_{t+1}, \widehat{X}_{t+1}$ have the same meaning as X_t and $\widehat{X}_t$. For WZ frames, SI generated by MCTI is regarded as initial values of pixels. $\widehat{X}_{t-1}$, $\widehat{X}_t$ and $\widehat{X}_{t+1}$ are updated in each iteration until the difference of distortion in (13) is below a threshold or the maximum iteration time is reached . Model weight parameters are determined when the process of iteration is over.

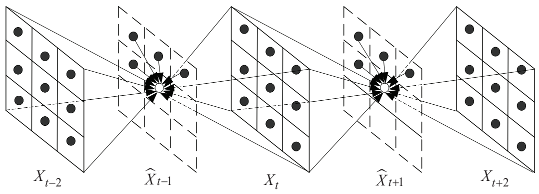

Fig.2 The first stage in weights training

3.3. Joint Learning for SI and CMP

In our proposed model, IU slides along the time axis at a step of GOP size as depicted in Fig.3. As a consequence, multiple SI can be generated through different UI for one WZ frame. For instance, there are two interpolated results $\widehat{X^{IUF}_{t+1}}$ and $\widehat{X^{IUB}_{t+1}}$ at time t+1, one is from IU forward (*IUF*) and the other is from IU backward (*IUB*). Note that $\widehat{X^{IUF}_{t+1}}$ contains more forward motion information as its weights are trained from frames X_{t-2}, X_t and X_{t+2}, while $\widehat{X^{IUB}_{t+1}}$ contains more backward motion information as its weights are trained from frame X_t, X_{t+2} and X_{t+4}. We consider that better interpolation performance can be achieved by averaging these two interpolated results. As a consequence, the SI estimated in our model is formulated as

$$\widehat{X}_{t+1} = (\widehat{X^{IUF}_{t+1}} + \widehat{X^{IUB}_{t+1}}) / 2 . \quad (14)$$

For CMP estimation, Eq.(9) implies a direct method. In order to derive more accurate CMP values, in our experiment, Eq.(9) is modified as (15)

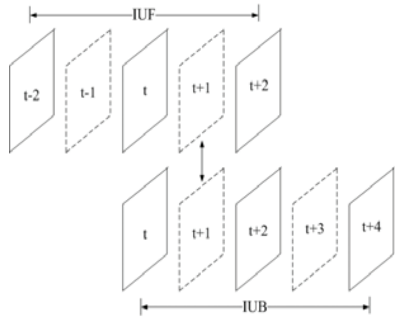

Fig.3 Two adjacent interpolation units

$$\sigma^2(k_1,k_2) = \frac{1}{M} \left\{ \begin{array}{l} \left\| \phi_{t-2}(k_1,k_2) - \hat{\phi}_{t-2}(k_1,k_2) \right\|^2 \\ + \left\| \phi_{t-1}(k_1,k_2) - \hat{\phi}_{t-1}(k_1,k_2) \right\|^2 \\ + \left\| \phi_t(k_1,k_2) - \hat{\phi}_t(k_1,k_2) \right\|^2 \end{array} \right\} . \quad (15)$$

where $\phi_{t-2}(k_1,k_2)$, $\phi_{t-1}(k_1,k_2)$ and $\phi_t(k_1,k_2)$ denote the actual values of the training samples for pixel at (k_1,k_2) on time *t*-2,*t*-1 and *t* respectively. Note that the actual values are intra decoded ones for key frames and MCTI results for SI. $\hat{\phi}_{t-2}(k_1,k_2)$, $\hat{\phi}_{t-1}(k_1,k_2)$ and $\hat{\phi}_t(k_1,k_2)$ represent the estimated values in the proposed model for corresponding samples.

The estimation process is illustrated in Fig.4. There are three kinds of points. The white one is the pixel for which the CMP is computed; the black ones stand for the actual values of training samples; the gray ones denote the values estimated. The average of SAD values between black points and gray ones is computed as σ^2 .

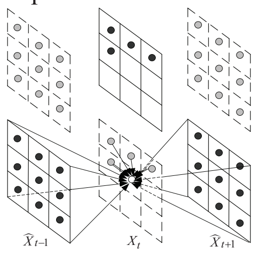

Fig.4 Correlation model parameters estimation

The Laplacian distribution parameter α is defined by

$$\alpha = \sqrt{\frac{2}{\sigma^2}} \quad (16)$$

4. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, experimental results are provided to demonstrate the performance of the proposed scheme. Results of two test sequences including *Foreman* and *Mobile* (QCIF, 30Hz, 4:2:0) are presented. In each sequence, 100 frames are selected and the GOP structure is IWI, where key frames are encoded in H.264 intra mode with three QP values:20,24,28. Table 1 includes the objective performance comparison for interpolated SI, where:

- MCTI_OBMC: the block-matching-based motion

Table 1 Objective quality comparison for interpolated side information (in dB)

	Foreman			Mobile		
	QP=28	QP=24	QP=20	QP=28	QP=24	QP=20
MCTI_OBMC	34.49	35.56	36.44	32.80	34.22	34.76
STAR	34.59	35.82	36.78	33.21	35.10	35.84
LearningWithAverage	**35.11**	**36.5**	**37.51**	**33.44**	**35.38**	**36.18**
Gain(overMCTI_OBMC)	**0.62**	**0.94**	**1.07**	**0.64**	**1.16**	**1.42**

compensation temporal interpolation, OBMC is used as a post process to smooth motion field [6].

- STAR: spatio-temporal auto regressive model proposed in [5] for frame interpolation.
- LearningWithAverage: the method we proposed in this paper.

From Table 1 we can see that our approach significantly outperforms the MCTI_OBMC approach for SI interpolation. Our method can improve up to 1.42dB for *Mobile* sequence and 1.07dB for *Foreman* sequence respectively. Overall simulation results presented in Fig.5 also show the efficiency of our method. Our model can improve 1dB for *Mobile* sequence at most and 0.5dB for *Foreman* sequence.

To evaluate the performance of online learning for CMP, we need to choose a benchmark. Offline learning provides insights on the maximum or "ideal" estimation performance since the CMP are obtained using both original data and SI. In our experiment, we utilize the method presented in [2] to do offline learning, which consists of two steps:

Step 1) Compute residual frame R between the WZ frame and the corresponding SI frame as follows.

$$R(k_1,k_2)=WZ(k_1,k_2)-SI(k_1,k_2) \tag{17}$$

Step 2) For each pixel in R frame, the Laplacian distribution parameter α is computed as follows.

$$\alpha(k_1,k_2)=\begin{cases}\sqrt{2}, & |R(k_1,k_2)|\le 1\\ \sqrt{\dfrac{2}{|R(k_1,k_2)|^2}}, & |R(k_1,k_2)|>1\end{cases} \tag{18}$$

The overall RD performances of *Foreman* and *Mobile* sequences are shown to demonstrate the efficiency of online learning in Fig.6. From the simulation results we can see our online learning strategy is close to offline learning.

5. CONCLUSION

In this paper, we proposed a novel strategy based on linear regression model to jointly estimate side information and correlation model in distributed video coding. In the proposed scheme, the intensity value and correlation model parameter of each pixel in side information are estimated simultaneously according to its neighboring spatio-temporal samples. Experimental results verify that the proposed method significantly improve the quality of SI and is close to offline learning approach for correlation model estimation.

6. ACKNOWLEDGEMENTS

This work was supported in part by National Science Foundation (60672088 and 60736043) and National Basic Research Program of China (973 Program, 2009CB320905). And the author would like to thank Dr. Yongpeng Li for his valuable suggestions.

7. REFERENCES

[1]B. Girod, A. Aaron, S. Rane, and D. R. Monedero, "Distributed video coding," *Proc.* IEEE, vol. 93, no.1, pp.71–83, Jan.2005
[2]C. Brites, F. Pereira, "Correlation Noise Modeling for Efficient Pixel and Transform Domain Wyner-Ziv Video Coding", IEEE Trans. Circuits Syst. Video Technol, vol. 18, No. 9, pp. 1177-1190, Sep. 2008.
[3]X. Li, "Least-square prediction for backward adaptive video coding," EURASIP Journal on Applied Signal Processing, 2006, special Issue on H.264 and Beyond.
[4]M.Bishop, Pattern recognition and machine learning. New York: Springer. (2006).
[5]Y. Zhang, D. Zhao, X. Ji, R. Wang, and X. Chen, "A spatial-temporal autoregressive frame rate up conversion," in Proc. IEEE Int. Conf. Image Processing, ICIP, 2007, pp. 441-444
[6]Michael T.Orchard, Gary J. Sullivan: Overlapped block motion compensation: an estimation-theoretic approach. IEEE Transact-ions on Image Processing 3(5): 693-699 (1994)

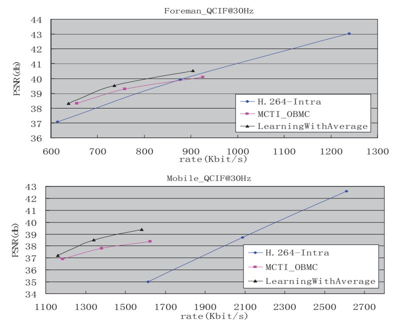

Fig.5 Overall simulation results for *Foreman* and *Mobile*

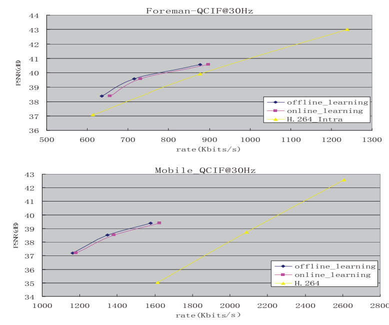

Fig.6 Overall RD performance of online and offline learning