ADAPTIVE PIECEWISE LINEAR BITS ESTIMATION MODEL FOR MPEG BASED VIDEO CODING

Jia-Bao Cheng and Hsueh-Ming Hang

Dept. of Electronics Eng., National Chiao-Tung Univ.
Hsin-Chu 300, Taiwan, Republic of China
Email: hmhang@cc.nctu.edu.tw

ABSTRACT

We propose an adaptive piecewise linear bits estimation model whose structure is similar to a tree-structured piecewise linear filter. Each node in the tree is associated with a linear relationship between macroblock *bits* and *activity/stepsize*. The parameters in this tree structure are adjusted by a modified LMS algorithm. Computer simulation results indicate that the adaptive bits model is able to precisely estimate the compressed bits regardless how the image contents vary along time. Also, when compared to the table-look-up bits model derived based on cluster analysis, the adaptive piecewise linear bits model has a much lower complexity to achieve about the same high performance.

1. INTRODUCTION

In many video compression applications, it is essential to control precisely the bit rate produced by the encoder, for example, in the cases of constant channel rate video transmission and storage. In a typical motion-compensated transform coding scheme such as *Simulation Model 3 (SM3)* used in the process of defining the international video compression standard, MPEG [2], its output bit rate is mainly controlled by adjusting quantizer stepsize to avoid buffer overflow/underflow. The simplest method to decide the quantization stepsize is based on the buffer status. That is, set the quantization stepsize small when the buffer is nearly empty and set the stepsize large when the buffer is nearly full. However, this control scheme often results in non-uniform picture quality because 1) it does not have an overall stepsize plan before execution, and 2) it does not take image contents into consideration. To achieve a uniform perceptual picture quality, we need a stepsize planning strategy or bits allocation scheme, which pre-analyzes the entire image content and allocate bits accordingly. In this approach, the bits model plays an important role in deciding the stepsizes because it predicts the final compressed bits when a certain quantization stepsize is in use before the real quantization and variable (word) length coding (VLC) steps are actually performed. When the predicted bit number does not match the bits budget, the selected set of stepsizes have to be altered.

This work was in part supported by a grant from Computer & Communication Research Labs., ITRI, Taiwan, ROC.

In order to use one-pass or nearly one-pass coding structures which cost much less than the multiple-pass coding structures in hardware, an accurate bits estimation model becomes very critical. There are two approaches to construct such a model. One is the analytic approach that derives a mathematical model based on information theory, for example, the model in [1]. The other approach is to derive an experimental expression based on test data. Because of the nonlinearity of quantizer and VLC and the variation of picture contents, it is rather difficult to fully describe an accurate and self-adaptive bits estimation model based only on mathematical theory. The experimental approach is taken in this paper. We propose an adaptive piecewise linear bits estimation model whose structure is similar to a tree-structured piecewise linear filer [4, 5]. Each node in the tree represents a linear relationship between image (macro)block bits and activity/stepsize. The parameters in this relationship can be adjusted by a modified LMS (least mean squares) algorithm. Simulation results show that due to its tree structure this bits model has a rather fast adaptation speed even in the case of scene changes. In addition, compared to the bits model derived based on cluster analysis, the adaptive piecewise linear bits model has a much lower complexity to achieve about the same high performance.

2. ACTIVITY MEASURES

The activity function is a measurement of image contents complexity; a high activity value indicates a hard-to-compress image (block). Several types of activity functions have been proposed. For example, the measure could be calculated based on the variance or the absolute value of DCT coefficients of an image block. It was reported that the activity measure based on the absolute values of the AC coefficients is relatively accurate [3]; hence, it is adopted here as follows: $ACT = \sum |$ AC coefficient $|$.

Using the activity function defined in the above, the number of coded bits per (macro)block is almost proportional to the activity measure and is nearly inverse-proportional to the quantization stepsize. An empirical first-order bits model is thus derived:

$$\widetilde{BIT} = m * \frac{ACT}{Q} + n, \tag{1}$$

where $\widetilde{BIT}$ is the estimated coded bits, ACT is the activity measure, Q is the quantization stepsize, and m, n are two constants derived from training data to minimize a selected error criterion. There are two main drawbacks of this simple first-order model. One, the parameters, m, n are picture-dependent; and two, the linear expression becomes less accurate outside a certain range of ACT and Q. To enhance its performance, we need to develop an adaptive scheme to adjust the parameters from time to time. Also, instead of a single model that covers the entire range of interests, we partition the data space (ACT, Q) into segments and design parameter sets for each segment separately. Therefore, a tree-structured piecewise linear model is adopted, in which a tree structure is used to combine several linear bits estimation models, each corresponding to a selected range of (ACT, Q) values and the parameters of each linear model can be adjusted after each macroblock being coded.

3. ADAPTIVE BITS ESTIMATION MODEL

The tree-structured piecewise linear filter is first proposed for adaptive equalization in digital communication by Gelfand et al. [4, 5]. Each tree node is associated with a linear filter and a corresponding threshold value is chosen to partition the domain of the filter. The filter coefficients and the threshold at each node are updated by the modified LMS algorithm. Through sequential and hierarchical partitioning of the input space, this approach is computationally efficient and converges rather rapidly as compare to many other nonlinear adaptive filters. A typical example of the tree-structured piecewise linear filter for our application is illustrated in Fig. 1 and explained below.

To construct a tree-structured piecewise linear filter, we specify three elements: a tap weight vector $\mathbf{c_t}$, an offset d_t, and a threshold θ_t for each node in the *Tree* T. Let $\mathbf{x}$ be an input data vector, then node t is associated with a linear filter

$$\widetilde{y_t} = \mathbf{c_t}'\mathbf{x} + \mathbf{d_t},$$

where $\widetilde{y_t}$ is the filter output at node t. The final output of this piecewise liner filter is defined by

$$\widetilde{y_T} = \widetilde{y_{t^*}},$$

where t^* is the terminal node in the tree T obtained through the following process. We start from the root node and use the rule

$$\widetilde{y_t} > \theta_t, \text{ go to } r(t)$$
$$\widetilde{y_t} \le \theta_t, \text{ go to } l(t),$$

where $r(t)$ is the right child stemmed from node t and, similarly, $l(t)$, the left child. Therefore, each node in a tree corresponds to a filter with inputs restricted to a polygonal domain denoted by χ_t. In general, the filter output $\widetilde{y_t}$ at the node t is determined by the filter weight $\mathbf{c_t}$ and the offset d_t, whereas the domain χ_t is determined by the weights $\mathbf{c}_s$, offsets d_s, and thresholds θ_s of all the ancestor nodes s of node t.

The above piecewise linear filter can be made adaptive by updating the filter input domain and the filter coefficients when new samples arrive. That is, the values of $\mathbf{c_t}$, d_t, and θ_t are adjusted by applying the least mean squares algorithm (LMS) to the input data sequentially [4, 5].

In the proposed adaptive piecewise linear bits estimation model, each node in the tree is associated with a first-order macroblock bits model restricted to a certain range of ACT/Q values. In other words, the filter output $\widetilde{y}$ now represents the estimated bits $\widetilde{BIT}$ in our bits estimation model. The input vector $\mathbf{x}$ now has the form of ACT/Q (a scalar) and the coefficient vector $\mathbf{c}$ and the offset d are now parameters m and n, respectively. Our adaptive bits estimation algorithm is similar to the original adaptive filter algorithm except for the initialization. To ensure a reasonable initial performance, we initialize each node in the tree with the same parameters that are derived from the constant coefficient first-order macroblock bits estimation model obtained off-line by the method described in Sec. 2.

The adaptive algorithm for piecewise linear bits estimation model is summarized below.

<*Initialization:*>
Let m_0 be the slope and n_0 be the bias term of the initial macroblock bits model.
We initialize each node in the tree by:
$p_t(0) = \frac{1}{2^{depth(t)}}$,
$m_t(0) = m_0$, $n_t(0) = n_0$, $\theta_t(0) = 0$,
where p_t is the probability of the input domain associated with node t.

< *Updating:*>
Let $(BIT(k), \frac{ACT}{Q}(k))$ be the $(k+1)$-th arriving coded data pair. Assume $\widetilde{BIT}_t(k) = m_t(k)\frac{ACT}{Q}(k) + n_t(k)$.
Propagate the data sample from the root node to a terminal node of T according to the rule:
$\widetilde{BIT}_t > \theta_t$, go to $r(t)$
$\widetilde{BIT}_t \le \theta_t$, go to $l(t)$.
If the data sample passes through node t, then its associated parameters are updated:
$p_t(k+1) = p_t(k) + \mu(1 - p_t(k))$
$m_t(k+1) = m_t(k) + \frac{\mu}{p_t(k+1)}(BIT(k) - \widetilde{BIT}_t(k))\frac{ACT}{Q}(k)$
$n_t(k+1) = n_t(k) + \frac{\mu}{p_t(k+1)}(BIT(k) - \widetilde{BIT}_t(k))$
$\theta_t(k+1) = \theta_t(k) + \frac{\mu}{p_t(k+1)}(BIT(k) - \theta_t(k))$.
Otherwise, the above parameters remain the same except that

$$p_t(k+1) = p_t(k) - \mu p_t(k).$$

4. ADAPTIVE BITS ALLOCATION

We now describe how the adaptive bits estimation model works together with bits allocation policy. The problem of bits allocation is to distribute bits to each macroblock properly so that the following two goals can be achieved: (1) the total coded bits should meet the bits budget, and (2) the perceptual quality of every coded image block should be almost equal. The above two requirements lead to following two additional problems: (1) how to come up with an adequate bits budget for each picture frame and (2) how to decide the coded image perceptual quality for every block. We do not attempt to solve these two additional problems here, but rather, for a given bits budget and a given picture quality measure, we want to allocate bits to each block so that the total coded bits would match the pre-assigned bits budget.

The bits allocation problem can be solved by a multi-pass approach that simply tries several promising quantization stepsizes for each image block to find the best stepsize that generates the desired bits. Each iteration (*pass*) comprises quantization and VLC operations performed on every image block. Although it could be rather accurate, this approach requires a large amount of computations. The one-pass approach is to predict the quantization stepsize based upon a *bits model.* If the bits model used is quite accurate, the results could be close to those of the multi-pass approach at a much lower computational complexity. The operation of our bits model consists of two modes: *estimation mode* and *updating mode.* In the estimation mode, the appropriate quantization stepsize is sequentially searched from the root node to the terminal node in the tree and the estimated bits number is the output of the terminal node. The best estimated stepsize is the one that produces the estimated bits closest to the bits budget. Then we use the estimated stepsize to quantize the current macroblock. In the updating mode, the bit estimation error which is the difference between the coded bits and the estimated bits is used to update the piecewise linear bits estimation model.

There are three main advantages of using the tree-structured filter structure. First, the piecewise linear filter is clearly superior to the single linear filter in solving nonlinear problems. Second, since the tree structure employs standard linear adaptive filtering techniques at each node, it is simpler than many other nonlinear adaptive filters such as polynomial filters. Third, the most important feature of the tree-structured adaptive algorithm is that it provides "piecewise" adaptation. It means that when each training sample arrives, we update the parameters of the nodes by passing through a data-dependent path from the root node to a terminal node. As a result, only one component filter (terminal node) in the filter bank (that makes up the entire piecewise linear filter) should be modified, and the other component filters remain unchanged. In the meanwhile, the updating process modifies slightly the domain of the corresponding filter and those of its neighbors'. Hence, the whole process is relatively simple but effective. Since the bits model is gradually dominated by the arriving coded macroblocks, the influence of the far-away coding samples becomes less important.

5. SIMULATIONS

Computer simulations have been conducted to evaluate the performance of the proposed bits estimation scheme. The error bits are defined as the difference between the coded bits and the estimated bits for each macroblock. Since the coding behavior is rather different for I-frame, P-frame, and B-frame in MPEG2 coding, three tree-structured bits estimation models are constructed for each of them separately. In Fig. 2, we first show the error bits for every macroblock without and with adaptation for the I-frames in encoding the video sequence *flowergarden* In the cases of "without adaptation" we simply use tree constant-coefficients estimation models with coefficents trained off-line using the same data. It is clear from this figure that the adaptive piecewise linear bits estimation model decreases the error bits significantly. Similar results are obtained also for P-frames and B-frames.

In Fig. 3, we demonstrate the adaptation capability of the adaptive piecewise linear bits estimators when scene change occurs. In the middle of the test sequence, the scene changes from *flowergarden* to *football.* The robustness of the adaptive bits estimators is due to their rapid and good adaptation capability. Therefore, suboptimal model parameters can be used as the initial values of this piece-wise linear model without degrading its long-term performance. Table 1 is the average error bits per macroblock with/without adaptation. It shows that the piecewise linear bits estimation scheme can cope with the variations of image sequence in real-time coding.

Although the above simulation results demonstrate the advantages of the adaptive bits estimation model over the constant coefficient model, we like to make one more comparison against a rather complicated but potentially better approach. This new bits estimation scheme is designed based on clustering analysis. First, we collect the data pairs in the form of $(BIT, ACT/Q)$ and generate a table that contains the representatives of these data pairs using the K-means cluster algorithm. This table is called TABLE1, and its entries are denoted by $(Tbl1_{BIT}, Tbl1_{ACT/Q})$. Second, we expand the dimension of each entry in the TABLE1 into the form of $(Tbl2_{BIT}, Tbl2_{ACT}, Tbl2_{Q})$. At this stage, each and every coding data triplet (BIT, ACT, Q) is compared against TABLE1 to find the best matching entry using the MSE measure. Then we obtain a new table with entry $(Tbl2_{BIT}, Tbl2_{ACT}, Tbl2_{Q})$, where $Tbl2_{BIT}$ and $Tbl2_{ACT}$ are the average bits and activity values of all the coding data triplets classified to the same table entry $(Tbl1_{BIT}, Tbl1_{ACT/Q})$ and quantization stepsize Q. This new table is called TABLE2. Table 2 shows the average error bits by using this approach with different table sizes. Compared to the simulation results in Table 1, the adaptive bits estimation model not only approaches the same performance limit but also use much lower memory than the table-look-up method. In addition, it is not easy to design adaptive method for the table approach. Therefore, the proposed adaptive bits estimation model is superior to the pre-trained clustering/table method.

6. REFERENCES

[1] J.-J. Chen and H.-M. Hang, "A transform video coder source model and its applications," *IEEE International Conf. on Image Processing '94*, pp.967–971, Austin, Texas, Nov. 1994.

[2] "MPEG video simulation model three (SM3)," *ISO/IECJTCI/SC2/WG11 Doc. MPEG 90/41*, Apr. 1991.

[3] W.-Y. Sun, H.-M. Hang, and C.-B. Fong, "Scene adaptive parameters selection for MPEG syntax based HDTV coding," *Int'l Workshop on HDTV '93*, Ottawa, Canada, Oct. 1993.

[4] S. B. Gelfand, C. S. Ravishankar, and E. J. Delp, "Tree-structured piecewise linear adaptive equalization," *IEEE Trans. Commun.*, Vol. 41, pp. 70-82, Jan. 1993.

sequence	*I-frm*		*P-frm*		*B-frm*	
	w/o	with	w/o	with	w/o	with
flowergarden	149	60	83	39	54	43
football	80	53	123	78	109	63
flower+foot	146	57	107	58	74	52

Table 1: *The average prediction error bits/macroblock of test images with/without adaptation.*

TABLE1 size	*TABLE2 size*	*flowergarden*			*football*		
		I	*P*	*B*	*I*	*P*	*B*
16	480	83	65	50	72	78	73
32	960	68	43	35	57	63	58
64	1920	63	41	40	61	70	66
128	3840	67	40	43	67	70	69

Table 2: *The average prediction error bits/macroblock using clustering/table method.*

[5] S. B. Gelfand and C. S. Ravishankar, "Tree-structured piecewise linear adaptive filter," *IEEE Trans. Inform. Theory,* Vol. 39, pp. 1907-1922, Nov. 1993.

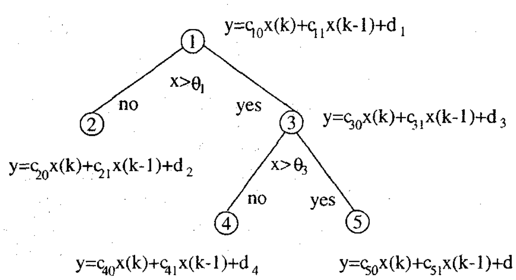
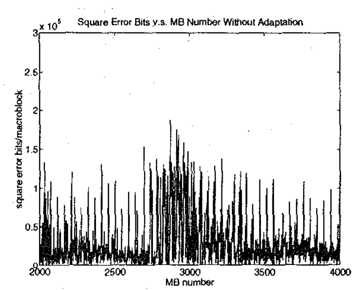
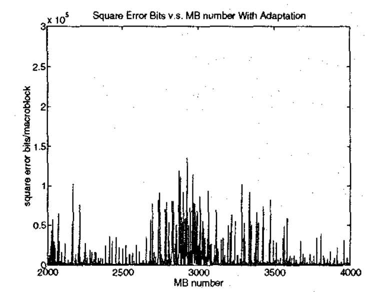


Figure 1: *A tree-structured piecewise linear filter.*

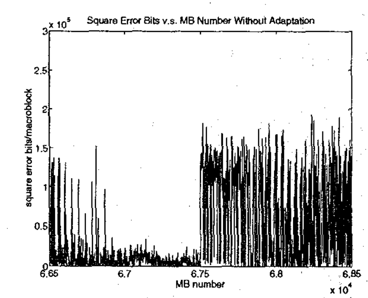


(a)

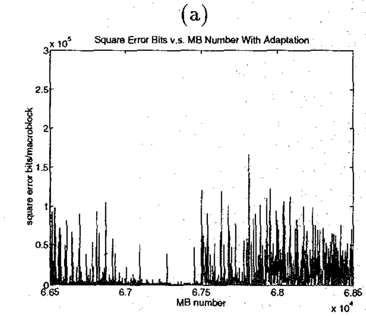


(b)

Figure 2: *The prediction error bits/macroblock of I-frame (a) using constant-coefficient linear model, and (b) using adaptive piecewise linear model.*



(a)



(b)

Figure 3: *The prediction error bits/macroblock at scene change (a) using constant-coefficient linear model, and (b) using adaptive piecewise linear model.*