Forward/Backward Adaptive Context Selection with Applications to Motion Vector Field Encoding

Wenqing Jiang and Antonio Ortega
Integrated Media Systems Center
Department of Electrical Engineering-Systems,
University of Southern California
Los Angeles, CA 90089-2564
{wqjiang, ortega}@sipi.usc.edu

Abstract

In low rate motion-compensated video coding, the rate required to encode the motion field can become a significant portion of the overall rate budget. This motivates us to investigate methods to efficiently, and losslessly, encode the motion field. Reductions in the required motion field bitrate allow increasing the rate for the residue images or may permit a higher density motion field to be used. In this paper, we consider adaptive context modeling techniques, such as those recently proposed in image coding applications, and explore their effectiveness for coding the motion data. We rely on various forward/backward context selection algorithms, and demonstrate how forward adaptation, based on alphabet partitioning approaches, can result in performance improvements over purely backward adaptive methods. We observe substantial reductions in rate, especially for dense motion fields, when comparing with popular differential coding schemes using VLC tables, such as those in the H.263 standard.

1 Introduction

While at high rates, such as those used for MPEG-1 and MPEG-2 video coding, the proportion of bits devoted to coding the motion field is small, at low rates (e.g., H.261,H.263), this overhead can become significant. This motivates us to investigate efficient techniques for coding of the motion field. We concentrate on the lossless case and assume that the motion field has been already computed. Clearly, additional gains can be achieved by trading off the accuracy of the motion field (e.g. using variable sized blocks) with the cost of transmitting it [1, 2]. However we do not consider these trade-offs here, although they could be incorporated into our framework, and we concentrate on the constant block size case.

A common approach for motion field coding is differential coding, where each motion vector value is predicted from previously transmitted adjacent vectors and the prediction error is coded by a variable length codes (VLC) table [1]. Median, rather than linear, predictors tend to be favored and in general the scalar components of the prediction error vector are coded separately. Numerous approaches have been proposed to improve the performance of the motion field coding. For example, vector coding of the prediction error using a 2D VLC table and combined with magnitude-based classification of the error has been proposed in [3]. Improvements can also be achieved by optimizing the choice of variable block size, through joint optimization of the block size and the motion vector as in [2], where dynamic programming and hierarchical tree-pruning techniques are used. In [4], the block matching motion estimation algorithm is construed as a vector quantization problem and an iterative design algorithm is proposed. The temporal redundancy of the motion field data is exploited in [5], where a scheme for motion compensation of motion vectors is proposed.

A common point of all these algorithms is that they employ a fixed entropy coder to code the prediction error and are thus limited by the zeroth-order entropy (i.e. memoryless entropy) of the prediction error. Further reductions in bit rate can be achieved by taking advantage of local correlations in the source, through the use of higher-order entropy coders, for example. However, a large alphabet size input, such as that required by a motion vector field with dynamic range [-16,15.5] or a continuous-tone image, precludes this approach due to its complexity. As an alternative one can resort to the context adaptive techniques we outline in this paper. In context adaptive methods lower order entropy coders are used, but the probability model used in the entropy coding depends on the neighboring vectors. This allows capturing of local correlation while maintaining reasonable complexity. Our paper is organized as follows. A general introduction to available context adaptive methods is presented in Section 2. In Section 3 we present the proposed algorithms and compare their performance. Finally, we discuss the results and derive some conclusions in Section 4.

2 Context Adaptive Coding

We start by giving a general overview of available context adaptive techniques which have recently been shown to be a very effective tool in still image coding [6, 7]. The principle of context adaptive coding is to attempt to model the conditional probability of symbols based on their surrounding neighborhood. This can be seen as a generalized form of prediction where each context determines a complete probability model for the new data (as opposed to only a predicted value, which would be the mean of the conditional distribution).

Our goal is to, given a neighborhood $\mathcal{N}(x_i)$ of the current symbol x_i, model the probability distribution of the symbol, $p(x_i|\mathcal{N}(x_i))$. The number of different values for the neighborhood can be quite large, especially for large alphabet sources, and thus it is normal practice to partition the possible neighborhoods into a smaller set of classes. The appropriate number of classes has to be kept small based on complexity considerations but also to avoid "context dilution" situations. For each class the probability model can be obtained off-line, based on some modeling assumptions, or can be learned on the fly as input symbols are being transmitted.

There are two general approaches for context adaptive coding, namely, backward and forward methods. If backward adaptation is used then $\mathcal{N}(x_i)$ comprises only symbols that have already been transmitted. Thus $\mathcal{N}(x_i)$ is known to the decoder and no overhead has to be sent. Most likely there will be a rule Q_b which maps all possible $\mathcal{N}(x_i)$ into a smaller set of contexts and thus the coding will be based on $p(x_i|Q_b(\mathcal{N}(x_i)))$ (see for example [6, 7]).

Alternatively, a forward adaptive scheme will use a non-causal neighborhood, comprising data that has not yet been transmitted. Obviously, information has to be sent to the decoder so that it can adjust its coding parameters, and thus the number of classes will determine the necessary overhead. In this case a different function Q_f is used to reduce the number of possible neighborhoods considered in the modeling process. As above, the data will be coded based on $p(x|Q_f(\mathcal{N}(x_i)))$. A popular approach has been to group the input symbols into segments or blocks and to assign to each segment a label which corresponds to the probability distribution that best matches it. An example of this approach can be found in [8], where blocks within image subbands are modeled as being Laplacian distributed, with the Laplacian parameter chosen to be one among a discrete set of values known to the decoder. The classification map itself (i.e. the labels assigned to each block) can then be sent using efficient methods, including backward adaptive ones. A different forward adaptive approach is set partitioning [9, 10], where each groups of inputs (for example image blocks) is assigned to a class (for example according to the largest element in the set [10]) without requiring an explicit probability model to be introduced.

While backward adaptive approaches may suffer from inaccurate classification, forward classification methods require that extra overhead information be sent. This motivates us to consider also alphabet partitioning [10] as an alternative method for forward adaptation. Alphabet partitioning methods are based on sending a coarse version of the symbol itself. For example a coarsely quantized symbol is sent first and then the residue information is transmitted. It can be shown that for an i.i.d. source the entropy is the same whether the source is coded directly, or the combination of partitioning information and residue information is coded instead [10]. Thus, this form of partitioning carries no overhead. Alphabet partitioning is also useful as a method to reduce the complexity of the coding process (without loss in compression) because it allows lower order entropy coders to be used within each partition. In order for alphabet partitioning to result in a lower overall rate we need to find partitioning rules such that (i) the classification map can be encoded with few bits (e.g., it exhibits high spatial correlation) and (ii) the set information can be used to improve the encoding of the elements within the set. We will consider these issues in the context of coding of motion field. Effective combinations of alphabet partitioning and backward adaptation are thus our objective.

3 Motion Vector Field Coding

We use four standard QCIF sequences to compare the performance of the various algorithms: *MissAmerica*, *Claire*, *Foreman*, and *Carphone*. The first two are representative of videoconferencing applications, and thus do not contain much motion, the latter two have much stronger motion. We use block sizes of 16x16, 8x8 and 4x4, and the motion field data is estimated by exhaustive block matching, while skipping every two frames in the original sequence (i.e., we use $0, 3, 6, 9, \cdots, 150$). The search range is $[-16, 15.5]$ with half pixel accuracy. The intra/still/motion decision is made by comparing the original block variance v_0 with the minimum compensated error variance v_1. If v_0 is smaller than v_1 the block is tagged as intra block and no motion information needs to be sent. If classified as inter block, v_1 is then compared with the error variance v_2 of the zero motion compensation with a favor to zero motion. If $v_2 \leq (v_1 + 1.0)$, then this block is classified as a still block with zero motion vector. Otherwise, full motion information has to be sent. The results of different methods tested in our experiment are listed in the tables and are given in terms of average bits per frame to code the motion field.

We will refer to Fig.1, which summarizes the data flows in the various algorithms we discuss. The ellipses represent data processed in each stage and labels over the arrowhead lines connecting these ellipses indicate the specific coding modules used for this transition.

3.1 Baseline algorithm

A typical lossless coding system for motion vectors is depicted in Fig.1(a). The motion vector data is first classified as an intra (no motion compensation), still (motion compensation with zero vector), or motion (nonzero vector) block. An arithmetic coder (AC) is used to code this classification map. A median predictor (MP) is then used to remove the local mean and the prediction error is coded using the VLC table from [1]. We will call this algorithm *Method A* and its performance should be similar to that of the H.263 standard.

Note that the classification information is coded in a memoryless manner even though there exists substantial spatial correlation (e.g. "still" areas may occupy several consecutive blocks). Thus it is beneficial to use context adaptive entropy coding to transmit the classification map. We denote this approach *Method B*. *B* uses an arithmetic coder, instead of VLCs, to code the prediction error. Note that the classification used in *A* and *B* is a simple version alphabet partitioning as described in the previous section. As noted above every time alphabet partitioning is performed gains can be achieved by efficient encoding of the classification map and by taking advantage of the set information. Thus the gain in going from *A* to *B* is an example of alphabet partitioning gain. Other partitioning methods will be considered in the next two sections.

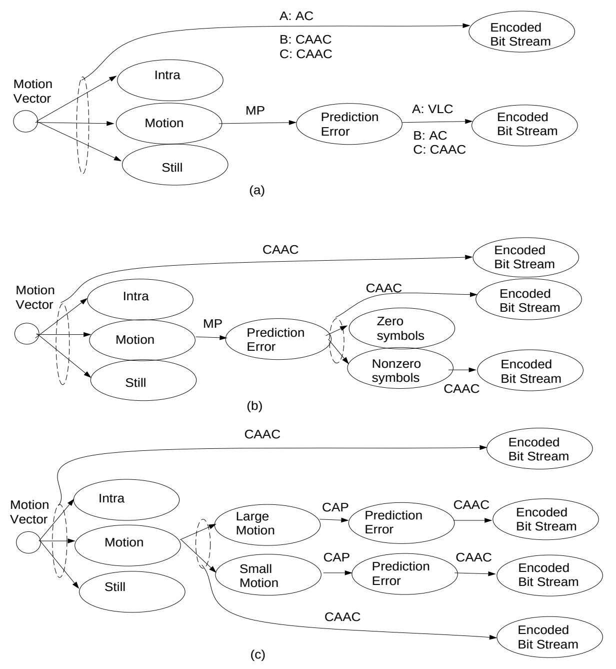


Figure 1: Coding flow of the proposed algorithms. (a) *Method A* encodes the initial classification map (still/motion/intra) using an arithmetic coder. The motion field is encoded using VLCs after prediction. *Method B* incorporates context adaptive coding for the classification map, *Method C* uses context adaptive techniques for both the classification map and the prediction error. (b) *Method D* uses alphabet partitioning combined with context adaptive techniques to encode the prediction error. (c) *Method E* also uses alphabet partitioning to separate vectors in to large and small before context adaptive predictive coding.

3.2 Prediction Error Coding

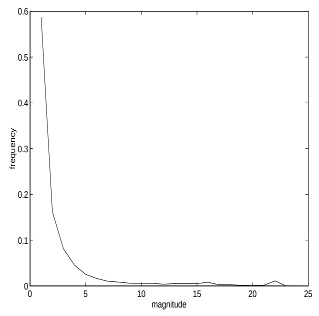
We first consider coding of the prediction residue with a context adaptive entropy coder. We select a backward adaptive approach, where an energy predictor $\hat{e}$ is calculated by taking the average prediction error magnitude from the current (causal) context neighborhood. Typically, large prediction errors tend to be clustered, and thus larger values of $\hat{e}$ generally indicate a higher probability that the current symbol will have a large prediction error value, thus the conditional density function is expected to have high variance. We use four classes which are determined by optimal quantization of the distribution of the $\hat{e}$, which is assumed to be Laplacian. We call this approach *Method C*. As can be seen from the results, *Method C* does not improve on *Method B*. Thus backward adaptive systems do not always guarantee significant bit rate reduction and the choice of classification is in fact critical to determine potential gains.

Although methods for optimal classification by training or probability modeling have been proposed [11, 7], we consider here a simple hybrid approach based on alphabet partitioning and backward adaptive classification (See Fig.1(b)). We focus on the most likely prediction error which, for a good prediction mechanism, will tend to be zero. Moreover a prediction error of zero is likely to be both frequent and spatially correlated, as regions of smooth motion will give consistently prediction error equal to zero. We thus consider the zero prediction error as a separate case and use alphabet partitioning to separate the zero and non-zero prediction error cases. For the non-zero case we then proceed as before, i.e. use a backward context based entropy coder and use the same classification rule as that in *Method C*. We call this approach *Method D*, and we observe that it outperforms *Method C*.

3.3 Class Adaptive Prediction

We have thus far considered a simple (motion/still/intra) classification accompanied with adaptive methods to encode the prediction error. We now present alternative forms of classification of the motion vectors. As a motivation, consider Fig.2(a) which depicts the histogram of motion vector values from 4 QCIF sequences (*Foreman, Carphone, MissAmerica and Claire*). It can be seen that most of the vectors have relatively small magnitude. We also notice that a single fixed predictor is not efficient, especially at the boundaries between regions of large and small motion. Thus, we propose to further classify the motion blocks into another two new classes, namely, those with small magnitudes and those with large magnitudes. This additional level of classification has the well known advantage that each set in the partition can have its own entropy coder with different alphabet size. This approach is called *Method E* and is shown in Fig.1(c).

Given this classification we can introduce a *class adaptive predictor* (CAP) to replace the fixed prediction scheme. Assume a predictor is being used which computes the predicted value based on the immediate neighbors of the current vector. Then, if information about the magnitude of the current vector is available, clearly we can eliminate any immediate neighbors *which do not belong to the same magnitude class.* Thus if any of the immediate neighbors does not belong to the same class it can be replaced in the prediction process by the latest transmitted vector with same class as the current vector. In other words, we can come up with a better predictor for the next symbol given that we know the partition to which this



(a)

```
1 2 2 2 2 2 2 0 0 0 0 1 1 0 2 2 2 2 0 2 2 2
1 2 2 2 2 2 0 0 0 0 0 0 1 0 2 2 2 2 2 3 2 2
1 2 2 2 2 0 0 0 0 0 0 0 0 2 2 2 0 0 2 2 2 2
1 2 2 2 2 0 0 0 0 0 0 0 0 2 2 0 0 0 2 2 2 2
0 2 0 2 2 0 0 1 1 0 0 0 0 0 2 0 0 1 2 3 3 2
2 2 2 2 0 0 0 0 0 0 0 0 0 2 2 2 0 0 2 3 2 2
2 2 0 0 2 0 0 0 0 0 0 0 0 2 0 2 0 0 2 2 3 2
0 1 1 1 2 2 0 0 0 0 0 1 0 0 0 2 0 0 2 3 1 2
2 1 1 1 2 1 0 0 0 0 1 1 0 0 0 2 0 0 0 0 0 0
0 1 1 1 0 0 2 2 0 1 1 1 1 0 2 0 0 0 0 0 0 0
2 1 1 0 0 0 2 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0
1 0 0 0 0 1 2 0 0 0 0 1 0 0 2 2 2 1 0 0 0 0
1 0 0 0 0 2 0 0 0 0 0 1 0 0 0 0 2 2 2 0 0 0
0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 0
2 0 2 0 0 0 2 0 0 0 0 0 0 0 0 2 0 2 0 2 2 1
2 0 2 0 0 0 2 0 0 0 0 0 0 0 0 2 2 2 0 2 0 0
2 2 0 0 0 0 0 0 0 0 0 0 1 0 2 1 2 2 2 0 0 2
2 2 0 0 0 0 2 0 0 1 0 0 0 1 0 0 2 2 2 0 2 2
```

(b)

Figure 2: (a) Motion vector data distribution. Distribution of vector magnitudes for the four sequences. (b) Motion field classification maps (block size 8x8). Code 0-Small Motion, 1-Large Motion, 2-Still, 3-Intra. between frames 69 and 72 from *Carphone*.

symbol belongs. To encode the classification map we resort to a context adaptive arithmetic coder, since there exists strong correlation between the classes of adjacent blocks, as can be seen in Fig.2(b)

Method	*Foreman*	*Carphone*	*MissAm*	*Claire*
A	736	525	366	146
B	699	493	307	123
C	707	501	321	129
D	676	488	305	122
E	700	509	317	121

Table 1: Motion Vector Encoding Comparison(16x16)

Method	*Foreman*	*Carphone*	*MissAm*	*Claire*
A	3050	2175	1400	1033
B	2778	1948	1150	570
C	2723	1934	1151	556
D	2610	1874	1109	502
E	2608	1952	1138	492

Table 2: Motion Vector Encoding Comparison(8x8)

4 Discussion

As a summary of the results we can first see that context adaptive coding of the classification maps (intra/still/motion) is very efficient compared to the zeroth order entropy coder. This is reflected in the rate reduction from A to B. The comparison between C and D has served to illustrate the shortcomings of backward classification and the benefits of selective applying forward classification. This leads to interesting issues in trade-offs in determining the choice between backward and forward classification. Except for the *Carphone* sequence, E has an average better performance than C. This indicates that forward classification of original motion vector data is more suitable for small motion sequences.

Method	*Foreman*	*Carphone*	*MissAm*	*Claire*
A	16428	11397	8923	3838
B	13438	9265	7036	2533
C	13272	9190	7018	2489
D	13021	9050	6902	2416
E	12863	9253	6854	2377

Table 3: Motion Vector Encoding Comparison(4x4).

Although context adaptive coding, both forward and backward, has been shown to be effective to code the motion field data, complexity issues have to be solved first to make it a practical alternative method. We further notice that in terms of overall bit rate, the motion field rates given here is still too high for low bitrate coding applications. More effort is necessary to make such a dense motion field possible in the future, in particular by improving classification methods and taking into account temporal dependencies.

References

[1] International Telecommunication Union, *ITU-T RECOMMENDATION H.263*, July 1995.

[2] M. C. Chen and A. N. Willson, “Design and optimization of a differentially coded variable block size motion compensation system,” in *Proc. of ICIP'96*, 1996.

[3] G. Y. Yu and C. T. Chen, “Two-dimentional motion vector coding for low-bit rate videophone applications,” in *Proc. of ICIP'95*, 1995.

[4] Y. Y. Lee and J. W. Woods, “Motion vector quantization for video coding,” *IEEE Trans. on IP*, vol. 4, pp. 379–382, Mar. 1995.

[5] J. Yeh, M. Vetterli, and M. Khansari, “Motion compensation of motion vectors,” in *Proc. of ICIP'95*, 1995.

[6] M. J. Weinberger, J. J. Rissanen, and R. B. Arps, “Applications of universal context modelling to lossless compression of gray-scale images,” *IEEE Trans. on IP*, vol. 5, pp. 575–586, Apr. 1996.

[7] C. Chrysafis and A. Ortega, “Efficient context-based entropy coding for lossy wavelet image compression,” in *Proc. of DCC'97*, (Snowbird, UT), Mar. 1997.

[8] R. L. Joshi, H. Jafarkhani, T. R. Fisher, N. Farvadin, M. W. Marcellin, and R. H. Bamberger, “Comparison of different methods of classification in subband image coding,” *Submitted to IEEE Trans. on IP*, 1995.

[9] A. Said and W. A. Pearlman, “A new fast and efficient codec based on set partitioning in hierarchical trees,” *IEEE Trans. on CAS for Video Tech.*, vol. 6, pp. 243–250, Jun. 1996.

[10] A. Said and W. A. Pearlman, “Low-complexity waveform coding via alphabet and sample-set partitioning,” in *Proc. of VCIP'97*, (San Jose, CA), Jan. 1997.

[11] X. Wu and N. Memon, “Calic-a context based adaptive lossless image codec,” in *ICASSP'96*, 1996.