A LOW-COMPLEXITY VIDEO ENCODER WITH DECODER MOTION ESTIMATOR

Sibel Yaman and Ghassan AlRegib

Center for Signal and Image Processing
Georgia Institute of Technology
Atlanta, GA 30332-0250
E-mail: {syaman, gregib}@ece.gatech.edu

ABSTRACT

In this paper, we investigate the use of video compression methods that require a simple encoder and a complex decoder for applications such as video surveillance, smart spaces, and sensor networks. In our proposed method, the encoder tries to identify the locations whose content cannot be predicted at the decoder, and codes such areas at higher fidelity. Typically, high-motion macro-blocks[1] represent such significant state regions. A shape-adaptive (SA) SPIHT encoder is then used to efficiently code these regions. On the decoder side, we perform motion extrapolation using the previously decoded frames to construct an estimate for the current frame. This estimate then serves as the side information at the decoder. Receiving the SA-SPIHT coded and the motion extrapolated, the decoder fuses the information in these two frames to produce a frame that is of higher quality than the component images. Experimental results show that the proposed codec outperforms H.264 intra mode by 3 dB.

1. INTRODUCTION

In video coding standards such as MPEG-X and H.26X, similarities among successive frames are exploited by the inter-frame predictive coding techniques. Nevertheless, the existence of the motion estimation (ME) in these standards requires the encoder to be 5-10 times more complex than the decoder. This type of asymmetry in complexity is appropriate when the video is encoded once, at a powerful processing unit, but needs to be decoded at several receivers. The best example for this scenario is CATV and satellite TV systems, where the video is coded at the cable/satellite headend using expensive, professional-quality video coders, but decoded at several set-top boxes that are much cheaper. However, emerging applications such as surveillance systems, smart spaces and sensor networks will require the opposite as far as the complexity is concerned. In these applications, a decoder-centric compression mechanism with computationally simple encoder node at the expense of a complex decoder node would be preferred.

Even though information theoretic bounds [1, 2] prove the existence of such efficient lossless\lossy compression methods with side information, the research on practical code designs has been considered only recently. In [3], Pradhan and Ramchandran develop a method, Distributed Source Coding using Syndromes (DISCUS), based on the idea of transmitting the syndrome of the coset that the codeword falls into. In their recent work, they have developed a new method called Power-efficient, Robust, hIgh compression, Syndrome-based Multimedia coding (PRISM) [4]. In this method, they include a network device in between the encoder and the decoder that converts the PRISM bit-stream into a standard bit-stream (e.g.MPEG/H.26L). Using this method, they place the entire computational complexity into the introduced network device. In [5], Aaron *et al.* compress key frames of the video sequence using a standard H.263 codec and the rest of the sequence using their turbo-code based codec.

In this paper, we study a video compression method that can be used in a new emerging class of applications (*e.g.*, sensor networks and smart spaces). In the proposed method, with the use of frame difference (FD), the encoder predicts the regions that the decoder cannot successfully estimate. It then codes such regions at a high precision. Meantime, the decoder exploits temporal coherency over the successive frames by motion estimation. Using the two most recently reconstructed frames, the decoder obtains an estimate for the current frame. This estimate then serves as the side information.

This paper is organized as follows: Section 2 introduces the key ideas underlying our proposed solution. Sections 3 and 4 discuss the encoder and the decoder architecture, respectively. In Section 5, we provide the simulation results. Finally, Section 6 presents the concluding remarks.

[1] Following the MPEG terminology, we refer to 16 x 16 blocks as macroblocks

2. DECODER-CENTRIC VIDEO COMPRESSION

In recent years, with the growth of applications such as sensor networks and smart spaces, power-efficient communications, low channel occupancy, and low-complexity signal processing have received more attention in the academic community. In this paper, we propose a video codec that addresses these requirements. More specifically, we propose a decoder-centric video coder where a shape-adaptive 2D SPIHT encoder allocates bits in such a way that the transmitted bits allow the decoder to recover from the motion extrapolation errors.

Figure 1 illustrates the block diagram of the proposed decoder-centric video codec. This architecture is inspired by the following observations:

- Motion estimation (ME) often fails around the motion boundaries and occlusion regions. Except for these regions, the decoder usually obtains a satisfactory motion extrapolated estimate for the current frame using the two previously decoded frames. Hence, these regions must be detected and coded with a higher quality at the encoder. We refer to such regions as the "region of interest (ROI)", and the detection process of these regions as "ROI extraction".
- The available bit budget at the encoder should be allocated so as to supplement the side information as much as possible. That is, no bits should be spent to encode the information that is already present in the side-information.
- Using a simple frame difference operator, the encoder can roughly predict the regions that the decoder motion extrapolation is likely to fail.

These observations inspired new encoder and decoder architectures that are illustrated in the following sections.

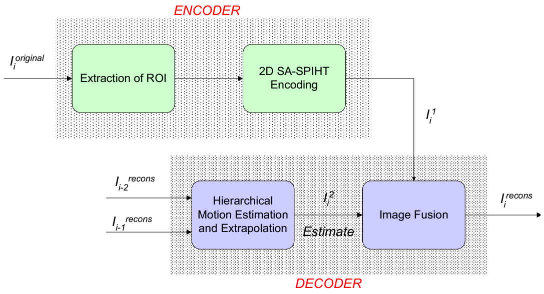


Fig. 1. Block diagram of the proposed codec.

3. ENCODER ARCHITECTURE

The task of the encoder in the proposed design is to extract the locations where the motion extrapolation at the decoder is likely to fail, and code these regions at a higher quality. In this section we illustrate how the encoder in the proposed codec achieves this task.

3.1. Region of Interest Extraction

As pointed out earlier, the encoder needs to identify the locations where the decoder cannot construct an accurate estimate of the current frame. Nevertheless, while doing so, the encoder should not resort to such computationally intensive operations as motion estimation or morphological region processing. Therefore, we used a frame difference (FD) operator, which serves as a simple tool that can predict these regions of interest. Not only FD is computationally simple, but it is also accurate in predicting the actual ROI even though it might fail at few cases. More accurate ROI estimates increase the compression efficiency at the expense of computational complexity. Consider the two residual images where the first one $R_i^{(prev,curr)}$ is between the current and the previous frame, and the second residual $R_i^{(est,curr)}$ is between the current frame and the motion extrapolated estimate. We observed that the distribution of energy in these two residuals are very similar, and both have residual error concentrated in certain areas of the frames.

In order to distinguish ROI from the remaining regions of the frame, we introduce a mask at the encoder with a value of 255 for opaque regions and a small but nonzero value for transparent regions[2]. Note that even in the regions where the motion extrapolation performs well, small errors will accumulate over time. Furthermore, the frame difference operator cannot always accurately locate the occlusion areas and motion boundaries. These errors will be handled with the use of such a binary mask, by which the encoder reserves most of its bit budget to encode opaque regions, but still allocates some bits to the remaining areas.

3.2. SA-SPIHT Encoding

The next task of the encoder is to encode the frame using shape-adaptive SPIHT (SA-SPIHT) algorithm [6]. Recall that the encoder codes the ROI with a higher quality than the rest of the frame. The way to achieve this is to flag transparent regions as "insignificant" during the shape-adaptive discrete wavelet transform (SA-DWT) so that the SPIHT algorithm processes these transparent regions in a manner identical to that of the other insignificant coefficients. In the literature, this approach has been successfully applied to a number of zerotree-based coders.

[2] A mask is composed of opaque and transparent regions. The distinction between opaque and transparent regions is that if a zero value is used for transparent regions, then the encoding algorithm processes these regions in a manner identical to other insignificant coefficients.

4. DECODER ARCHITECTURE

4.1. Motion Estimation and Extrapolation

The decoder constructs an estimate for the current frame I_i using the two most recently reconstructed frames I_{i-2}^{recons} and I_{i-1}^{recons} under the assumption that the motion trajectory (of each pixel) can be linearly approximated over short periods of time. In the proposed method, because of its relatively low computational complexity, we selected hierarchical motion estimation algorithm to find the motion vectors (MVs) from I_{i-1}^{recons} to I_{i-2}^{recons}. We begin with the assumption that the motion vectors from I_i to I_{i-2}^{recons} are the same as those from I_{i-1}^{recons} to $I_{i-1^{recons}}$. This is illustrated in Figure 2. Using this approach, if we seek sub-pixel accuracy, we face with the non-integer location problem. In this case to find the best integer-valued location, we construct a very small search region around each pixel position in the estimate. We then calculate the squared-error between the initial motion vector and all the motion vectors in the search region. The pixel location with the smallest squared-error is chosen.

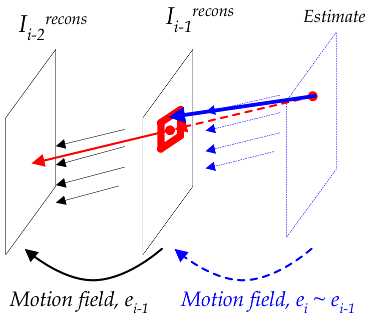


Fig. 2. Illustration of motion extrapolation used at the decoder. The dashed red motion vector (MV) from *Estimate* to I_{i-1}^{recons} represents the initial MV obtained by estimating MV from I_{i-1}^{recons} to I_{i-2}^{recons}. The thick blue line shows the location that the pixel is matched to after the search.

Note that we do not require the motion extrapolated estimate for the current frame to be very accurate. The encoded bit-stream will be used to correct the errors made by the motion extrapolation. This is the image fusion algorithm, which will be explained next.

4.2. Image Fusion

In SPIHT, the transform coefficients are compared to a threshold. The starting value for the threshold is T_0, and it is halved in each iteration. The coefficients that are larger than the current threshold are quantized to $1.5T_0$ and placed into the bit stream. After the entire subband pyramid is traversed, the threshold is halved, and the procedure is repeated. With each pass, the quantized coefficients become closer to the their original values by $0.25T_0$. Figure 3 depicts the first few steps of this process for a single transform coefficient.

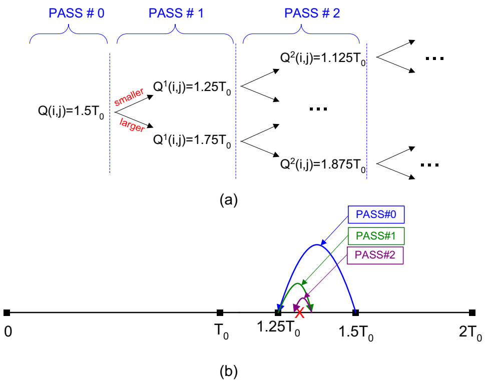


Fig. 3. (a) The refinement process of a coefficient (b) With each pass, the quantized value better approximates the original value

This operation at the SPIHT-encoder allows us to write an expression for the upper bound for the quantization noise at the n^{th} refinement pass as:

$$\delta_{max}^{n} = T_0 - \sum_{i=0}^{n} (\frac{1}{2})^{i+1} T_0$$

The fusion algorithm then makes use of the fact that the transform domain coefficients of the motion extrapolated estimate (for the current frame) should not differ from those of the encoded frame by more than δ_{max}^{n}. If the difference is greater than δ_{max}^{n}, then the coefficient value should be corrected so that the difference from the coefficient coming from the encoded frame is set equal to δ_{max}^{n}.

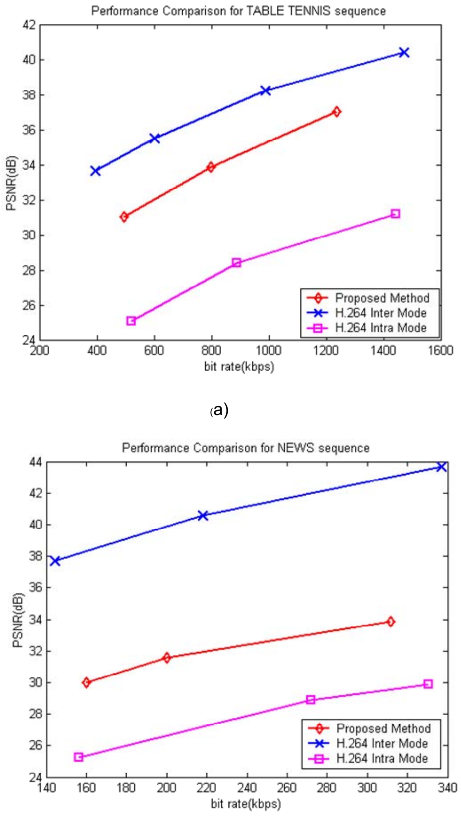
Note that the ROI locations are coded at a high precision at the encoder. Hence, with a high probability, the transform coefficients of the motion extrapolated estimate will be corrected by the transmitted coefficients. Meantime, the transform coefficients of the remaining regions are coarsely quantized, which scarcely causes any changes in the transform coefficients of the motion extrapolated estimate.

5. SIMULATION RESULTS

In this section, we present simulation results that illustrate the performance of the proposed scheme in comparison to H.264 [7] and 2D SPIHT[3]. We have applied the proposed method to several video sequences and here we report simulation results for 15 frames from CIF-resolution TABLE TENNIS, and QCIF-resolution NEWS sequences. The frame

[3] The source code is available at http:\\qccpack.sourceforge.net

rate is 30 frames per second in all simulations. The number of levels in the hierarchical motion estimation is set to 3. In the experiments with H.264, we used two B-type pictures between each P-P or I-P pictures. The GOP size is set to 15.



(a)

(b)

Fig. 4. Simulation results for the (a) TABLE TENNIS, (b) NEWS sequences.

The experimental results are shown in Figure 4. The rate and PSNR values have been averaged over 15 frames. As seen in these figures, the performance of the proposed method lies between the intra and inter modes of H.264. The proposed method outperforms H.264 intra-mode at all bit rates by more than 4 dB for the TABLE TENNIS sequence. Moreover, the performance of our method is close to the inter mode operation of H.264. The difference is around only 2 dB on average for TABLE TENNIS. Puri and Ramchandran in [4] claim that their codec performs 1.5-3 dB better than intra mode of H.264, while Aaron *et al.* in [5] claim that their improvement over the intra mode of H.263 as at most 3 dB.

These results demonstrate the fact that we achieve significant performance improvements with the proposed coder. Given that the decoder has side information obtained by motion extrapolation, we could encode ROI at a higher precision than the remaining regions. At the decoder side, the existence of an upper bound for the quantization error in transform domain allows us to accurately fuse the image at the decoder. Note that, because the encoder is required to be computationally simple as discussed in Section 3, we use frame difference operator and SA-SPIHT coding at the encoder. Both of these algorithms are computationally simple.

6. CONCLUSIONS

In this paper, we proposed a new approach to the utilization of the side information in a new class of video coders that are useful in such emerging applications as sensor networks and smart spaces. Side information enables the encoder to code the pictures at various precision levels. On the decoder side, we utilized the fact that there is an upper bound for the quantization parameters in SPIHT. This allowed us to fuse the encoded and the side information successfully. The simulation results have shown that the proposed method is superior to the intra mode operation of H.264 in terms of rate-PSNR performance with a low encoder computational complexity. Its performance is, on average, only 2 dB poorer than that of inter mode operation of H.264.

7. REFERENCES

[1] D. Slepian and J.K. Wolf, “Noiseless coding of correlated information sources,” *IEEE Transactions on Information Theory*, vol. IT-19, no. 4, pp. 471–480, July 1973.

[2] A.D. Wyner and J. Ziv, “The rate-distortion function for source coding with side information at the decoder,” *IEEE Transactions on Information Theory*, vol. IT-22, no. 1, pp. 1–10, May 1976.

[3] S.S. Pradhan and K. Ramchandran, “Distributed source coding using syndromes (discus): design and construction,” in *Proc. IEEE Data Compression Conference(DCC)*, March 1999, pp. 158–167.

[4] R. Puri and K. Ramchandran, “Prism: A new robust video coding architecture based on distributed compression principles,” in *Allerton Conf. Communication, Control, and Computing*, October 2002.

[5] Anne Aaron, Rui Zhang, and Bernd Girod, “Towards practical wyner-ziv coding of video,” in *Proc. International Conference on Image Processing*, September 2003.

[6] S. Li and W. Li, “Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding,” *IEEE Transactions on Circuits and Systems for Video Coding*, vol. vol. 10, no. 3, pp. pp. 725–743, August 2000.

[7] *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG Joint Model Reference Version 7.3.*