ENCODER-ASSISTED ADAPTIVE VIDEO FRAME INTERPOLATION

Gökçe Dane, Khaled El-Maleh, Yen-Chi Lee**

Electrical and Computer Engineering Department, University of California, San Diego
*Qualcomm Inc., San Diego, CA
gdane@ucsd.edu, {kelmaleh, yclee}@qualcomm.com

ABSTRACT

In low bandwidth video coding applications, frame rate is reduced to increase the spatial quality of the frames. However, video sequences that are encoded at low frame rates demonstrate motion jerkiness artifacts when displayed. Therefore, a mechanism is required at the decoder to increase the frame rate while keeping an acceptable level of spatial quality. In this paper, we present a new method to perform video frame interpolation by sending effective side information for frame rate up conversion applications. The proposed scheme encodes the skipped frames lightly by sending motion vectors and an important information map which indicates the decoder the type of interpolation method to perform. We also propose a novel overhead reduction method to keep the side information cost low. Experimental results show that the proposed algorithm outperforms decoder-only frame rate up conversion methods and gives better performance in terms of PSNR and visual quality over encoding at full frame rate without frame skipping.

I. INTRODUCTION

In order to meet low bandwidth requirements, video applications such as video telephony or video streaming reduce the bit rate by encoding the video at a lower frame rate. However, low frame rate video produces artifacts in the form of motion jerkiness. For that reason, temporal frame interpolation, also known as frame rate up conversion (FRUC) is necessary to display the video at a higher frame rate.

FRUC methods can be divided broadly into two categories. The first category reconstructs video frames without taking motion information into account. This class includes methods such as frame repetition (FR) and frame averaging (FA). Although these algorithms perform well in no or low motion content, they produce either motion jerkiness in FR or blurring of the objects in FA when there is high motion. The second category uses motion-compensated conversion techniques like [1]. In this category, the spatial quality of the interpolated frames heavily depends on how close the estimated motion is to the true object motion. In low-complexity standard compliant (i.e. decoder-only) FRUC applications, the motion information that is used to interpolate the skipped frames comes from the motion information of the previous and/or subsequent frames. However, these motion vectors are not always reliable to use directly to interpolate the skipped frames. If they are used without processing, artifacts are introduced due to incorrect motion vectors. Furthermore, for intra-coded blocks or for frames that comes right before or after intra (I) frames there is no motion information available and extra motion estimation or processing is necessary at the decoder.

One approach to eliminate the artifacts and enhance video quality is to perform motion processing before FRUC at the decoder [2, 3]. If complexity is not an issue for the decoder; instead of using the received motion vectors, new motion estimation algorithm like bi-directional or true motion estimation can be performed [4]. However, motion estimation-based FRUC at the decoder is limited to the information available in the bit-stream. Since the frame is already skipped during encoding, the motion estimation algorithms will not be able to describe the actual motion for the skipped frame.

In this paper, we propose a new Encoder-Assisted (E-A) video coding framework for FRUC application. The proposed algorithm calculates and sends effective side information associated with the skipped frames to improve FRUC quality at the decoder. The objectives of the proposed E-A video interpolation algorithm can be summarized as follows: (i) to reduce the PSNR fluctuation between interpolated frames (that are skipped at the encoder) and the consecutive encoded frames, (ii) to reduce the blur artifacts in bi-directional interpolation that occurs in any motion-compensated FRUC application due to non-matching forward and backward motion vectors, (iii) to reduce the visual artifacts due to the lack of prediction error, which is not available for the skipped frames at the decoder. The side information transmitted to the decoder contains motion vectors of the skipped frame

and an information map which signals the decoder the type of interpolation method to perform for each block. In order to keep the cost of side information low, we propose a novel overhead reduction method by adaptively picking which information to send.

In the rest of the paper, Section II presents the details of the proposed framework. Section III describes the overhead reduction algorithm. Simulation results with comparison to other methods in the literature are presented in Section IV, which is followed by conclusions in Section V.

II. THE PROPOSED FRAMEWORK

The proposed framework is composed of two main components at the encoder and another one at the decoder as demonstrated in Fig. 1. The shaded areas in Fig. 1 represent the blocks where the proposed scheme introduces additional processing. S-frames refer to the frames that are skipped at the encoder. The encoder performs motion estimation between S_2 and P_1 as shown in Fig. 2 and obtains motion vectors (mv_{12}) for S-frames. For the next P-frame (P_3), motion estimation is performed between $\hat{S}_{12}$ and P_3. $\hat{S}_{12}$ refers to the motion compensated prediction of S by using P_1 and mv_{12} only. The actual S frame is going to be obtained by different interpolation equations which are derived form the most general interpolation equation which is given in (1).

$$S_2(x, y) = \alpha_1 P_1(x+mv_{12x}, y+mv_{12y}) + \alpha_2 P_1(x+mv_{23x}, y+mv_{23y}) + \alpha_3 P_3(x-mv_{23x}, y-mv_{23y}) + \alpha_4 P_3(x-mv_{12x}, y-mv_{12y}) \quad (1)$$

In the above equation, the odd numbered frames (i.e., P-frames) are encoded with high fidelity (i.e. both motion vectors and prediction residuals are sent), whereas for the even numbered frames (i.e., S-frames) only motion vectors and interpolation equation map are sent. The weighting coefficients in (1) are subject to $\Sigma_i \alpha_i = 1$ to keep the intensity values of the pixels normalized. The skipped frames are recovered by using motion vectors estimated for that particular frame (mv_{12}) and/or also using motion vector (mv_{23}) estimated for the consecutive frame by using the following equations which are subsets of (1).

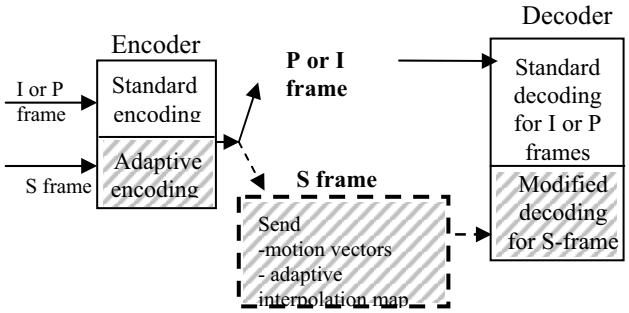


Fig.1. E-A video frame interpolation system diagram

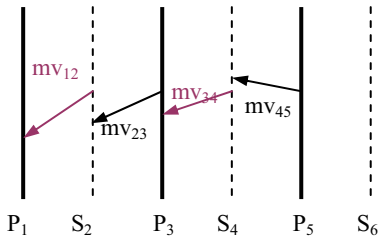


Fig.2. Illustration of reference and current frames and motion vector directions for P and S frames

$$S_2(x, y) = P_1(x+mv_{12x}, y+mv_{12y}) \quad (2.a)$$

$$S_2(x, y) = P_3(x-mv_{23x}, y-mv_{23y}) \quad (2.b)$$

$$S_2(x, y) = \tfrac{1}{2} P_1(x+mv_{12x}, y+mv_{12y}) + \tfrac{1}{2} P_3(x-mv_{23x}, y-mv_{23y}) \quad (2.c)$$

$$S_2(x, y) = \tfrac{1}{3} P_1(x+mv_{12x}, y+mv_{12y}) + \tfrac{1}{3} P_1(x+mv_{23x}, y+mv_{23y}) + \tfrac{1}{3} P_3(x-mv_{23x}, y-mv_{23y}) \quad (2.d)$$

$$S_2(x, y) = \tfrac{1}{4} P_1(x+mv_{12x}, y+mv_{12y}) + \tfrac{1}{4} P_1(x+mv_{23x}, y+mv_{23y}) + \tfrac{1}{4} P_3(x-mv_{23x}, y-mv_{23y}) + \tfrac{1}{4} P_3(x-mv_{12x}, y-mv_{12y}) \quad (2.e)$$

The interpolation techniques presented in (2) are checked to predict the S-frame at the encoder. Subsequently, the label of the equation which gives the highest peak signal to noise ratio (PSNR) is selected and transmitted for each block. The equation labels make up the interpolation map. When the decoder receives this information, it interpolates the missing frame based on this map by using the interpolation methods which are also available at the decoder. The enhancement obtained by multi interpolation equations increases for cases where the content of S_2 can not be predicted from P_1 but it may be predicted form P_3. In this case, using (2.b) or (2.c) will be more useful than using (2.a) alone. Note that although motion estimation is performed for 16x16 blocks, the interpolation equation label can be sent for blocks that are as small as 2x2. This offers the advantage of low motion vector overhead, since motion estimation is performed only once for bigger block size. As the block size for equation label assignment decreases better performance both in terms of PSNR and visual quality can be obtained. However, using smaller block size for labels increases the cost of the side information. In the next section, we describe a novel algorithm to reduce the cost of the side information.

III. SIDE INFORMATION OVERHEAD REDUCTION

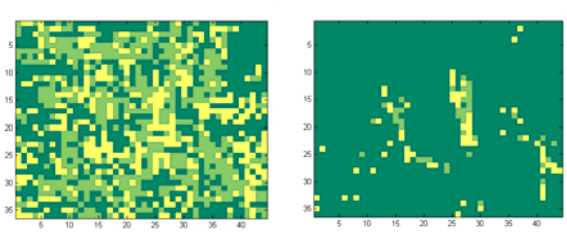
In order to understand the cost of the side information, let's consider the following example. Assume that only the first three equations of (2) are used in interpolation of a QCIF image. In the worst case, 2 bits will be transmitted for each block. If 4x4 block size is used for assigning interpolation labels, a total of 36×44×2=3168 bits (or 396 bytes) per frame will be used for the equation label map

which is not reasonable for low bit rate applications. If we analyze the equation label map shown in the left side of Fig. 3(i) left closely, we see that some of the 4x4 blocks in a close neighborhood (as in upper right corner of left figure) share the same equation label. Therefore these blocks can be grouped and assigned a common equation label. However, in that case a mechanism is required to signal the decoder to indicate which blocks are grouped together. This implies that there will be an additional cost to transmit this grouping information. Instead of grouping similar blocks, we developed a novel scheme for equation map overhead reduction which does not require sending the grouping information. The proposed method makes use of the correlation between forward and backward motion compensated prediction to locate the blocks which need equation labels to transmit. Since forward and backward predictions are available both at the encoder and decoder, there is not any need or additional cost to send the difference map.

Let **F** be the forward motion compensation of reference frame and **B** is the backward motion compensation of the future frame as illustrated in the flow chart in Fig. 4. In the overhead reduction algorithm, we first form a difference map by calculating the absolute value of the difference between frames **F** and **B**. Then the difference map is thresholded with a value th_1 to obtain a binary difference map. The binary difference map is downscaled by a scaling factor k (such as k=2, 4, 8) to match the size of the equation label map. Downscaling is carried out by replacing each value in the binary map with the sum of the values within its neighborhood. The goal of this step is to match the size of the difference map with the size of the equation label map. Another thresholding operation with th_2 is performed on the smaller size map, and for each location in that map, an equation label is found by comparing the PSNR values of the interpolated blocks that we obtain by using each equation. Subsequently, the total amount of bytes spent for the equation labels is calculated. If the total map size is above a given bit budget say R, then threshold th_2 is increased and total amount of bytes spent is re-calculated. If it is less than the bit budget R, the algorithm stops and the equation labels are packed in raster scan order. One other advantage of the proposed overhead reduction method is its precise rate control mechanism.

IV. EXPERIMENTAL RESULTS

In this section, we compare the proposed E-A FRUC framework to decoder-only FRUC [2] and standard encoding without frame skipping. The experiments focus on doubling the frame rate. The simulations are performed by coding five QCIF sequences at 10 fps, at 48kbps with an MPEG-4 based codec. Motion vectors are estimated for a block size of 16x16 and equation labels are assigned for



(i) Equation label map before (on the left, 310 bytes), and after overhead redcution (on the right, 81 bytes) with the proposed algorithm. Different intensity values illustrate different equation labels.

(ii) The predicted frame by using deterministic equation (i.e. always 2.a) (on the left), and using adaptive equation (on the right)

Fig.3. An example of frame prediction with overhead reduction

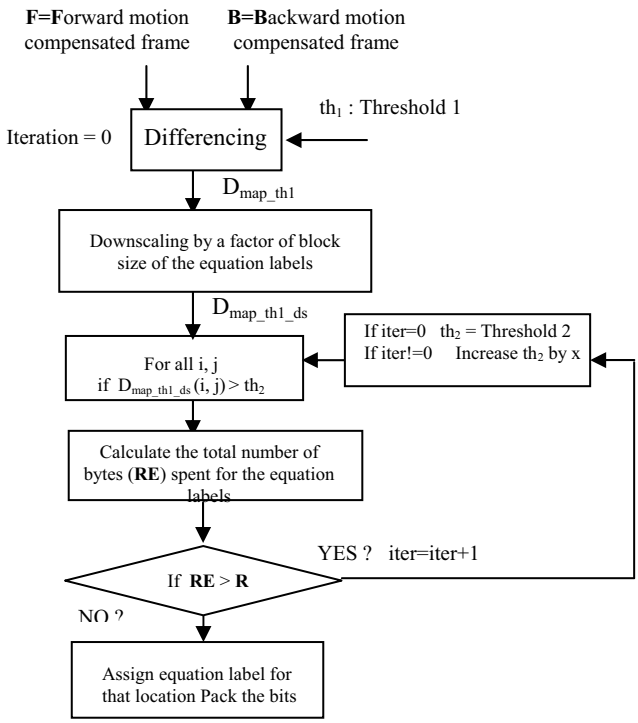


Fig.4. Flow chart of the interpolation map overhead reduction

a block size of 4x4. In the first experiment, Carphone sequence is decoded at 48kbps fixed bit rate with various FRUC algorithms. The first four methods presented in the

Table I. PSNR comparison of E-A FRUC with decoder-only FRUC

FRUC METHOD	PSNR (averaged over 150 frames)
Frame Repeat (FR)	29.51 dB
FRUC with no mv processing	29.26 dB
FRUC with mv processing (bi-directional)	30.53 dB
FRUC with mv processing (uni-directional)	29.75 dB
E-A FRUC	**31.57 dB**
Encoding at 10fps (no frame skipping)	31.11 dB

first four rows of Table I refer to decoder-only FRUC. For decoder-only algorithms the sequence is encoded at 5fps and the frame rate is doubled at the decoder by the following methods which are, frame repetition, motion-compensated interpolation without motion vector processing, bi-directional and uni-directional motion-compensated prediction with motion vector processing [2]. The fifth row in Table I shows the results of the proposed algorithm (denoted as E-A FRUC) and the last row is regular encoding at 10fps without frame skipping. The proposed algorithm performs 0.46 dB better than regular full frame rate encoding. Moreover it decreases the PSNR fluctuation that takes place in decoder-only FRUC algorithms. Table II compares the E-A approach to regular 10fps encoding at 48 kbps for different QCIF sequences. From the last column, we can observe that 0.2 to 0.9 dB gain can be obtained for various video sequences. The bytes spent for motion vectors for these sequences occupy 3-25% of the total bit rate. (Specifically 3.4% in Akiyo, 3.6% in Salesman, 15% in Coastguard, 18.5% in Carphone and 25% in Foreman). As motion activity increases, overall PSNR gain decreases. Visual quality comparison of the proposed algorithm and standard encoding is demonstrated in Fig. 5.

V. CONCLUSION

In this paper, we proposed a novel encoder-assisted video frame interpolation method and its accompanying overhead cost reduction algorithm. The proposed algorithm encodes the skipped frames lightly by sending motion vectors and an important information map which signals the decoder the type of interpolation method to perform. The algorithm can benefit from small block sizes such as 2x2, even though the motion estimation is performed for larger block sizes like 16x16. Future work will focus on comparing the E-A framework with generalized B-frames in H.264 codec.

Table II. PSNR improvement over standard full frame rate encoding

Clip, 48kbps	**A.** Standard 10fps encoding	**B.** Encoding at 5fps with proposed method	**B-A** PSNR improvement
Akiyo	37.28 dB	38.19 dB	0.91 dB
Coastguard	27.92 dB	28.52 dB	0.60 dB
Salesman	31.88 dB	32.58 dB	0.70 dB
Carphone	31.11 dB	31.57 dB	0.46 dB
Foreman	29.25 dB	29.46 dB	0.21 dB

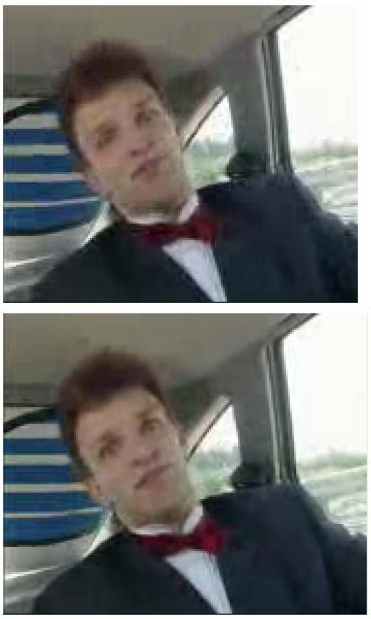


Fig. 5.Visual comparison of regular 10 fps encoding (at the top), with encoder-assisted FRUC (at the bottom.)

REFERENCES

[1] S. Liu, J.W. Kim, and C.-C. J. Kuo, "Non-linear motion-compensated interpolation for low bit rate video," *SPIE Proc. of Int. Symp. on Optic. Scc., Eng. and Inst., App. of Digital Image Proc. XXIII,* San Diego, July, 2000.

[2] G. Dane, and T. Q. Nguyen "Motion vector processing for frame rate up conversion," *IEEE Int. Conf. on Acous. Speech and Signal Proc., ICASSP'04*, vol.3, pp.309-312, May 2004.

[3] H. Sasai, S. Kondo, and S. Kadono, "Frame-rate up-conversion using reliable analysis of transmitted motion information," *IEEE Int. Conf. on Acous. Speech and Signal Proc. ICASSP'04,*, vol.5, pp.257-260, May 2004.

[4] G. de Haan, P. Biezen, H. Huijgen, and O. Ojo, , "True-motion estimation with 3D-recursive search block matching," *IEEE Trans. On Circuits and Systems*, vol. 3, no.5, pp. 368-379, Oct 1993.