MOTION VECTOR PROCESSING FOR FRAME RATE UP CONVERSION

Gökçe Dane and Truong Q. Nguyen

University of California at San Diego
Department of Electrical and Computer Engineering
9500 Gilman Drive, La Jolla, CA, 92093-0407

ABSTRACT

In this paper, the effect of motion vector accuracy on the efficiency of motion compensated frame rate up conversion (MC-FRUC) is studied. The motion vector processing problem is formulated and analyzed via motion vector modelling. A processing scheme is proposed to improve the interpolated frame quality at the decoder. Practical application of integrating motion vector processing algorithm to a standard H.263 decoder for MC-FRUC is also discussed. Experimental results show that 0.4-0.6 dB gain in H.263 codec, and 0.5-2 dB gain in off-line frame rate conversion can be obtained by the proposed algorithm.

1. INTRODUCTION

Frame rate up conversion (FRUC) is the conversion process between any two display formats with different frame rates. Besides the scanning format applications [1], FRUC can also be used in low bandwidth video coding. In low bandwidth applications, a number of frames are skipped in the encoding stage and missing frames are later interpolated during the decoding process.

Many FRUC algorithms have been developed, which are divided broadly into two categories. The first category uses combination of video frames without taking the object motion into account. This class includes methods such as frame repetition and frame averaging. Although these algorithms can be perfect in the absence of motion, they produce motion jerkiness and blurring of the moving objects when there is motion. The second category uses advanced conversion technique that employs motion [2]. The interpolator that predicts the missing frame can be either linear such as Haar filter, or nonlinear such as median operation [3]. However as one interpolates the additional frames to increase the frame rate of the video, artifacts are introduced due to incorrect motion vectors. As expected, the *true* motion estimation is most crucial in MC-FRUC applications.

The problem of motion compensated FRUC for standard video codecs (e.g. H.263, MPEG) is different and more challenging than off-line FRUC applications. The received motion vectors are not very reliable to use directly in FRUC. The reason is that in most standard video coding applications, motion estimation is performed by minimizing an error criterion such as sum of absolute difference or mean square error. Although this reduces the bit rate of video, it does not completely describe the actual motion between frames. Furthermore, a new motion estimation at the decoder is computationally expensive. Therefore, the motion information contained in the received bitstream should be fully utilized to obtain better quality in the interpolated frame.

An MPEG-2 video codec that uses frame rate up conversion is presented in [4], where natural motion vector estimation [5] is used during the video encoding process. An object based refinement of motion vector fields is given in [8]. It uses the same 3-D recursive motion estimation algorithm given in [5], and has segmentation as a post processing module which increases the complexity. An adaptive interpolation scheme is described in [6], where bidirectional motion estimation is used, and either forward or backward blocks are chosen based on mean absolute difference threshold in final interpolation. The general framework of adapting temporal filter taps that combines forward and backward blocks for MC-FRUC is discussed in [7].

In this paper, we assume that we do not have access to the encoder, which means motion estimation algorithm is not changed. We give a general motion vector processing framework and propose a motion vector processing scheme that is applied on any received motion vector field. We analyze the cases in which the proposed scheme reduces prediction errors theoretically by motion vector modelling. Integrating MC-FRUC via motion vector processing algorithm to a standard H.263 decoder for MC-FRUC is also studied.

The paper is organized as follows. The relation between efficiency of MC-FRUC with motion vector accuracy is presented in Section II. The motion vector field models and how to process them analytically is described in Section III, which is followed by the proposed algorithm in Section IV. The experimental results are discussed in Section V. Finally, last section concludes our work.

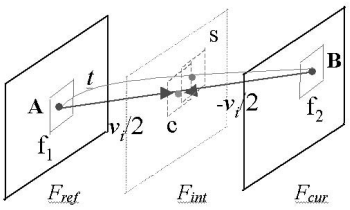


Fig. 1. Motion compensated frame interpolation in FRUC

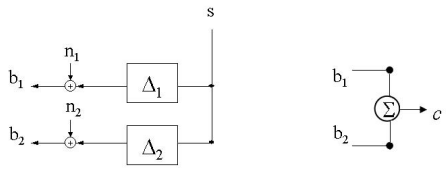


Fig. 2. Statistical error model

2. MOTION COMPENSATED FRAME RATE UP CONVERSION (MC-FRUC)

In MC-FRUC, the missing frame is interpolated based on the motion vector information calculated between the reference and current frame (Fig. 1). If FRUC is applied on standard codecs, then the reference and current frame can be any I, B or P frames. The assumption is that the motion between the reference frame and the interpolated frame, similarly the motion between the interpolated frame and current frame is half of the motion vector v_i in amplitude (for 1:2 up conversion) and it lies on the same trajectory as v_i. Explicitly, if block f_1 moves by v_i from frame F_{ref} to F_{cur}, then it is most likely that the block f_1 moves by v_i/2 from frame F_{ref} to F_{int}, and f_2 moves by -v_i/2 from frame F_{cur} to F_{int}. If we let b_1 = $f_1(v_i/2)$ and b_2 = $f_2(-v_i/2)$, then block c in the corresponding frame F_{int} can be interpolated by $c = 1/2(b_1 + b_2)$ by using half and half temporal interpolation filter, where b_1 is forward, and b_2 is backward motion compensated block.

For the analysis, we adapted the statistical model (Fig. 2) that is used in [9] for multi-hypothesis video coding. The difference from [9] is in the analysis, where the forward and backward blocks are compensated with different motion vector accuracy. In FRUC applications, we do not have access to the missing frame. All the blocks are not compensated equally in frame interpolation, hence motion vector error variance differs. Let Δ_1=(Δx_1, Δy_1) and Δ_2=(Δx_2, Δy_2) be the forward and backward motion vector errors respectively. They represent the deviation of estimated motion vectors from the real motion trajectory *t*. Let s be the video signal (i.e. block) in the missing frame F_{int} that we want to interpolate at the decoder, and c be the prediction of s. The aim is to minimize the prediction error $e(x, y) = s(x, y) - c(x, y)$. Associating the motion vector errors with motion compensated blocks b_1 and b_2 (Fig. 2) and replacing them in $c(x, y)$ = 1/2 $b_1(x, y)$ + 1/2 $b_2(x, y)$, we have

$$e(x,y) = s(x,y) - 1/2 \cdot (s(x - \Delta x_1, y - \Delta y_1) + s(x - \Delta x_2, y - \Delta y_2)) + \eta(x,y) \quad (1)$$

The mean square error in prediction can be found by calculating the power spectral density psd of the error. The psd is defined as the fourier transform of the expectation of ee^H. Let the error in motion vector be D_i = $e^{j\omega_x \Delta x_i - j\omega_y \Delta y_i}$ in frequency domain, then psd of prediction error is:

$$\Phi_{ee}(\omega_x,\omega_y) = \Phi_{ss}(\omega_x,\omega_y) \cdot [1 + 1/4E(D_1 D_2^H) + 1/4 + 1/4E(D_2 D_1^H) + 1/4 - 1/2E(D_1) - 1/2E(D_1^H) - 1/2E(D_2) - 1/2E(D_2^H)] \quad (2)$$

We assume that motion vector errors have independent Gaussian distribution $p_i(\Delta x, \Delta y)$ with fourier transform $P_i(\omega_x, \omega_y)$=$e^{-\frac{(\omega_x^2+\omega_y^2)\sigma_i^2}{2}}$. Note that the expectation of motion vector error is actually the fourier transform of the motion vector *pdf*. Hence, $E(D_1 D_2^H) = E(D_1^H D_2)$, $E(D_1)$=$E(D_1^H)$, $E(D_i D_j)$=P_i P_j and $E(D_i)$=P_i. And equation (2) becomes

$$\Phi_{ee}(\omega_x,\omega_y) = \Phi_{ss}(\omega_x,\omega_y)[3/2+1/2P_1P_2-P_1-P_2] \quad (3)$$

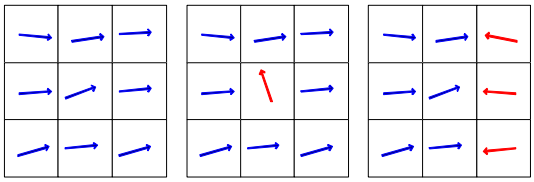
In equation (3), as each σ_i goes to zero, P_i goes to one and $\frac{\Phi_{ee}}{\Phi_{ss}}$ goes to zero. Hence prediction error decreases. In the next section, motion vector processing is related to motion vector error variance via motion vector models.

3. ANALYSIS OF MOTION VECTOR FIELD MODELLING AND PROCESSING

In MC-FRUC applications, the ground truth is not available and one needs to model the motion vector in order to determine motion vector errors. Motion fields are usually smooth functions except at the object boundaries. In [10], an $AR(1)$ motion vector model is introduced in the context of video coding, where the motion vector is predicted by the motion vector of previous block. In this work, we want to utilize a larger number of motion vectors, hence we will consider $AR(p)$ motion vector field models for different regions such as smooth, outlier and object boundary. Let $v_r(j)$ be the motion vector of the j^{th} (center) block of any one of the 3x3 blocks given in Fig. 3. Then, non-causal $AR(p)$ model can be written as

$$v_r(j) = \mathbf{A}\mathbf{v_r}(j) + \eta \quad (4)$$

where $\mathbf{A}$=[a_1 a_2 a_3 ... a_p], $\mathbf{v_r}(j)$=[$v_r(j-m)$, $v_r(j-m+1)$ ... $v_r(j-1)$, $v_r(j+1)$ ... $v_r(j+m)$]^T, and $p = 2m$.



(i) smooth region (ii) outlier (iii) object boundary

Fig. 3. Motion Vector Field Models

Let $v_m(j)$=**B** $\mathbf{v_m}(j) + \eta$ be the measured (or received) motion vectors, $v_p(j)$=**C** $\mathbf{v_m}(j) + \eta$ be the processed motion vectors, and $v_r(j)$ be the real motion vectors as given in (4). The motion vector processing problem can be formulated as follows.

Problem statement: Find a motion vector processing function g that acts on received motion vectors $\mathbf{v_m}$ and produces $v_p(j)$, (i.e. $v_p(j)$=$g(\mathbf{v_m})$), such that the variance σ^2_{ep} of motion vector error $v_{ep}(j) = v_r(j) - v_p(j)$ after processing should be less than the variance σ^2_{em} of motion vector error $v_{em}(j) = v_r(j) - v_m(j)$ before processing.

In this work, vector median [11] filtering (VMF) and low pass filtering (LPF) are considered for motion vector processing functions. For VMF, the motion processing vector **C** for $AR(p)$ model can be given as **C**=[0 . . . 1 . . . 0]. The n^{th} coefficient being 1 implies that n^{th} vector in a window of received motion vectors is picked as an output of VMF. For LPF, the processing vector **C** is given as **C**=[1/N 1/N . . . 1/N], which implies that the received motion vectors in a window are simply averaged. The error variance of j^{th} motion vector before and after processing can be written as

$$\sigma^2_{em} = E((v_r(j) - v_m(j))^2) = \sigma^2_{rj} + \sigma^2_{mj} - 2E(v_r(j)v_m(j)) \quad (5)$$

$$\sigma^2_{ep} = E((v_r(j) - v_p(j))^2) = \sigma^2_{rj} + \sigma^2_{pj} - 2E(v_r(j)v_p(j)) \quad (6)$$

In order to understand how to reduce error variance by VMF and LPF, the statistical properties of each motion vector model is analyzed next. Let σ^2_{rj} be the variance of motion vector $v_r(j)$, ρ_{jk} be the correlation coefficient between j^{th} and k^{th} block.

(i) smooth region: $\sigma_{rj} \approx \sigma_{rk}$, $\rho_{jk} \approx 1$, $\forall\ k$ in the window.

(ii) outlier: $\sigma_{rj} \gg \sigma_{rk}$, $\rho_{jk} \ll 1$, $\forall\ k \neq j$ in the window.

(ii) boundary: σ_{rj} and ρ_{jk} within each object is similar to case *(i)*, and across objects is similar to case *(ii)*.

Assume that VMF is applied on an outlier $v_m(j)$, then σ_{pj} in (6) will be smaller than σ_{mj} in (5) due to variance condition imposed in case (ii). Furthermore, $E(v_r(j)v_p(j))$= $\mathbf{ARC}^T$ will be bigger then $E(v_r(j)v_m(j))$=$\mathbf{ARB}^T$ due to correlation condition of case (ii), with $\mathbf{R}$=$E(\mathbf{v_r}(j)\mathbf{v_m}(j))$. Therefore after processing, the motion vector error variance σ^2_{ep} will be reduced, which in turn will reduce prediction error in MC-FRUC. LPF has smoothing effect on the motion vector field. Application of **C**=[1/N 1/N . . . 1/N] on received motion vectors will increase the correlation between motion vectors, and reduce motion vector error variance.

4. PROPOSED MOTION VECTOR PROCESSING

The overall system diagram is given in Figure 4. The first step of the algorithm windows the incoming motion vector field. The size of the processing window changes according to the image and block size. If motion vector field is not very dense, the window size should not be very large, since the correlation between blocks is not enough to make use of motion vector information in that neighborhood. If image size is big, and the motion vector field is dense, window size should accordingly. The angle variance of the motion vectors are calculated for motion field classification. Next outliers are determined and corrected by the VMF step. A motion vector that is classified as outlier can sometimes represent a small object like a ball in a sports sequence, or a flying bird in a natural scene, which can be determined by motion vector tracking. We do not consider this case here, in this work we assume low delay and small buffer, hence the processing is done only on the current received motion vector field.

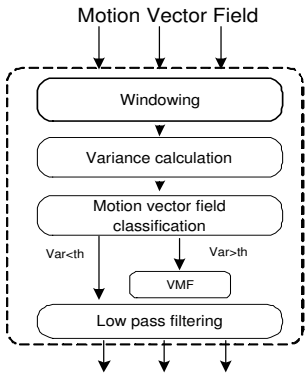


Fig. 4. Proposed scheme

5. EXPERIMENTAL RESULTS

In this section, two sets of experiments; off-line FRUC, and FRUC in H.263 codec are presented. In the off-line case, full search and hierarchical search algorithms are used for motion estimation on un-compressed video sequences. The block size for full search is 8x8, and the search range is 16 pixels for both horizontal and vertical directions. For hierarchical search 3 levels are used. The block size is 2x2, 4x4, 8x8 in the first, second and third levels respectively. Every other frame is skipped during the encoding process and later they are interpolated using two methods. The first one uses the standard motion compensated interpolation (MCI)

	Full Search Motion Estimation				Hierarchical Search Motion Estimation			
	Foreman	City	Zoom	Ballet	Foreman	City	Zoom	Ballet
Avg. Psnr before proc.	33.1	27.17	33.08	26.25	31.88	25.56	31.91	25.81
Proposed scheme	34.58	29.65	34.46	27.15	34.5	27.83	34.4	27.19
Avg. Psnr increase	1.48	2.48	1.38	0.9	2.62	2.27	2.49	1.38

Table 1. PSNR (dB) comparisons for off-line MC-FRUC

	No Annex			Annex J (deblocking filter mode)	
	Foreman (CIF) 225 kb/sec	Foreman (QCIF) 79 kb/sec	Stefan (CIF) 270 kb/sec	Foreman (CIF) 225 kb/sec	Foreman (QCIF) 79 kb/sec
Avg. PSNR increase	0.48	0.51	0.57	0.49	0.44

Table 2. PSNR (dB) results for MC-FRUC in H.263

[2] with received motion vectors and is denoted as 'before processing'. The second one uses the motion vectors obtained by the proposed algorithm and is denoted as 'after processing'. The performance of the algorithm is tested on several video sequences such as foreman, ballet, zoom, and city. Simulations are performed over 150 frames of foreman sequence (CIF), 40 frames of zoom and city sequences, 10 frames of ballet sequence. The results are given in Table I.

Next, we have applied the MC-FRUC on a H.263 codec [12]. In a standard codec, macroblock types, GOP structures, the type of annexes are practical issues. We have used H.263 with no Annex on, and with Annex J so that for some macroblocks 4 motion vectors were sent. The GOP structure was IPPP and all the motion vectors were unidirectional. The psnr increase for foreman and stefan sequence compressed at different rates for different sizes are given in Table II. By the proposed algorithm, we are able to achieve 0.5 dB on the average. Besides the psnr value, subjective picture quality is also important. The proposed algorithm is able to get rid of dominant blocking artifacts caused by outliers. Low pass filtering of motion vectors instead of the frames help reduce artifacts without blurring. The video files in .avi format are available at "http://video processing.ucsd.edu/fruc_demo.htm".

6. CONCLUSION

In this paper, the effect of motion vector processing on motion compensated frame rate up conversion is analyzed theoretically. MC-FRUC is investigated by combining forward and backward blocks with different motion vector error variances. The motion vector processing problem that reduces the error variance is formulated via motion vector modelling. Motion vector processing algorithm will be extended to cover more models and object-based structures in the future work. The experimental results show that the proposed algorithm generate higher quality frames with less blocking and blurring artifacts.

7. REFERENCES

[1] E. B. Bellers and G. de Haan, *De-interlacing: A Key Technology for Scan Rate Conversion, Advances in Image Communication*, Volume 9, Elsevier, MA: Wellesley-Cambridge Press, 2000.

[2] C. Cafforio, F. Rocca, and S. Tubaro, "Motion-compensated image interpolation," *IEEE Trans. on Communications*, vol.38, no.2, pp. 215-221, 1990.

[3] O. A. Ojo and G. de Haan, "Robust motion-compensated up conversion," *IEEE Trans. on Consumer Electronics*, vol.43, no.4, pp. 1045-1056, 1997.

[4] F. J. de Bruijn, W. H. A. Bruls, D. D. Burazerovic, and G. de Haan, "Efficient video coding integrating MPEG-2 and picture rate conversion," *IEEE Trans. on Consumer Electronics*, 2002.

[5] G. de Haan, P. W. Biezen, H. Huijgen, and O. A. Ojo, "True-motion estimation with 3-D recursive search block matching," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 3, pp. 368379, Oct. 1993.

[6] T. Chen, "Adaptive temporal interpolation using bidirectional motion estimation and compensation," *Proc. of IEEE Conference on Image Processing*, Vol. II, pp. 313-316, Sept. 2002.

[7] G. Dane, T. Nguyen, "Analysis of motion vector errors for frame rate up conversion," *Proc. of 37th Asilomar Conf. on Signals, Systems and Computers*, Nov. 2003.

[8] H. Blume, G. Herczeg, O. Erdler, and T. G. Noll, "Object based refinement of motion vector fields applying probabilistic homogenization rules," *IEEE Trans. on Consumer Electronics*, vol. 48, No. 3, pp. 694-701, Aug. 2002.

[9] B. Girod, "Efficiency analysis of multihypothesis motion-compensated prediction for video coding," *IEEE Trans. on Image Processing*, vol. 9, no. 2, pp. 173-183, 2000.

[10] J. Ribas-Corbera, D.L. Nenhoff, "Optimizing block size in motion-compensated video coding," *Journal of Electronic Imaging*, vol.7, no.1, pp.155-65, Jan. 1998.

[11] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proc. IEEE*, vol. 78, pp. 678-689, April 1990.

[12] ITU-T Recommendation H.263 Version II, "Video coding for low bitrate communication," Nov. 1998.