Efficient Motion Vector Interpolation for Error Concealment of H.264/AVC

Jie Zhou, Bo Yan, *Senior Member, IEEE*, and Hamid Gharavi, *Fellow, IEEE*

***Abstract*—In video communications the compressed video stream is very likely to be corrupted by channel errors. In order to combat these channel errors, many methods have been proposed. In this paper, a new method for motion vectors (MVs) interpolation with the help of MV extrapolation (MVE), is presented to conceal the corrupted frame of H.264/AVC. In our proposed method, the MVs of lost blocks can be obtained after interpolation with the extrapolated MVs. Experimental results show that this method is able to provide better performance than existing algorithms in terms of PSNR.**

***Index Terms*—Error concealment, error-resilient video transmission, H.264/AVC.**

I. Introduction

THE H.264/AVC [1], [2] standard is the latest video coding technology, which performs much better than previous standards for wired and wireless video applications by including some error resilient techniques [3]. In video communications, error-prone networks are very likely to cause packet loss, which may degrade the video quality significantly. In order to alleviate this effect, our error concealment (EC) technique is capable of combating packet loss in video transmissions.

EC techniques take advantage of the spatial or temporal correlation in the received video data to recover lost data. For an INTRA coded frame (I-frame), only spatial correlation will be exploited. In the case of an INTER coded frame (P-frame or B-frame), both spatial and temporal correlations can be used. In H.264/AVC video coding, the motion vector (MV) is highly significant in the decoding process. If MVs are received with errors, not only the current frame will be corrupted, but also succeeding frames.

Recently many methods have been proposed to recover the lost MVs in order to improve the decoded video quality [4]–[8]. Among them, [9] proposes a hybrid temporal error concealment method, which is able to switch or blend the methods of boundary matching algorithm (BMA) and decoder motion vector estimation (DMVE). However, this method mainly discusses the criterion of MV selection; it doesn't consider the construction of MV candidates.

Based on BMA, [10] and [11] propose to define an order function using the Sobel operator or texture intensity, which decides the concealment priority of corrupted macro-blocks (MBs). However, experimental results show that the performance improvements of these two methods are not significant as compared to the conventional method, which conceals the lost MBs from both sides to the middle used in H.264/AVC JM reference software [12].

In order to estimate the lost MVs more accurately, some methods propose to divide the lost MB (16×16) into smaller blocks (4×4) for temporal error concealment. For example, Zheng *et al.* propose to recover the MVs of the blocks by Lagrange interpolation (LI) [13]. Although this method is able to interpolate the lost MVs with the MVs of neighboring blocks, its performance is still unsatisfactory.

In this paper, we propose a new method to recover the MVs of lost blocks, which is implemented based on MV extrapolation (MVE). With the help of extrapolated MVs from the previous frame, our method is able to interpolate the MVs of lost blocks. As a result, it is able to provide more accurate MVs compared with BMA and LI. Experimental results show that the proposed algorithm significantly outperforms existing methods in concealing the corrupted frame.

The remainder of this paper is organized as follows. In Section II, we briefly introduce BMA and LI methods. In Section III, we propose our algorithm, which can improve the decoded video quality significantly after transmission over error-prone channels. We then evaluate the proposed method by simulations and present the results in Section IV. Finally, in Section V, we draw the conclusions.

II. Existing Methods

A. Boundary Matching Algorithm

BMA can recover the lost MVs starting from the leftmost image block, proceeding to the right and then downwards. Let the size of an image block be $N \times N$. Let the pixel values of the reconstructed frame n be denoted by $P_R(x, y, n)$, where (x, y) is the spatial coordinate. Suppose the MV of an image block at frame n is lost or received in error. We denote the estimate of the lost MV by $\hat{d}$, and its x, y components by $\hat{d}_x$ and $\hat{d}_y$ respectively. The estimated reconstructed image block using an estimated MV is then

$$P_R(x, y, n) = P_R(x + \hat{d}_x, y + \hat{d}_y, n - 1)$$
$$x_0 \le x < x_0 + N, \quad y_0 \le y < y_0 + N \quad (1)$$

where (x_0, y_0) is the upper left coordinate of the block. We denote the variations between the current image block and the one above it, the one to its left, and the one below it by D_A, D_L and D_B respectively as shown in Fig. 1 [14]–[16]. In this figure, the

Manuscript received June 07, 2010; revised August 01, 2010; accepted September 23, 2010. Date of publication November 11, 2010; date of current version February 23, 2011.

J. Zhou and B. Yan are with the School of Computer Science, Fudan University, Shanghai 200433, China (e-mail: byan@fudan.edu.cn).

H. Gharavi is with the National Institute of Standards and Technology, Gaithersburg, MD 20899 USA (e-mail: gharavi@nist.gov).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBC.2010.2086771

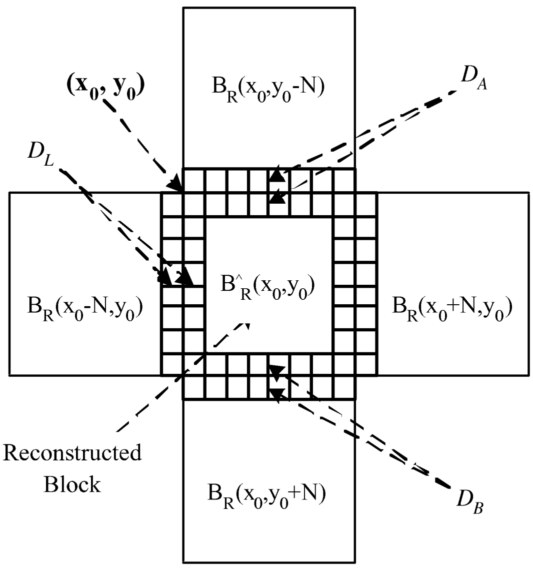

Fig. 1. Choice of the recovered MV by boundary match criterion.

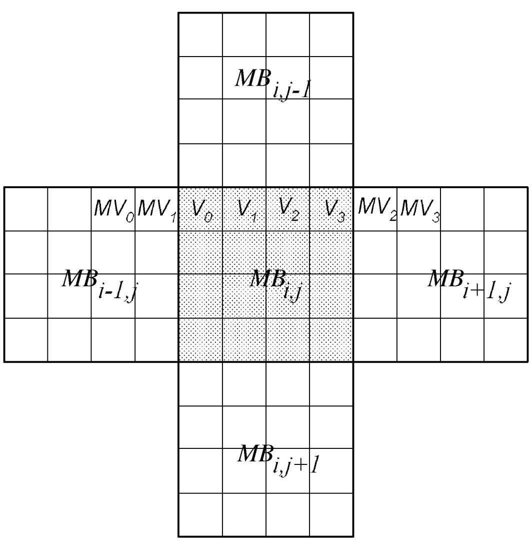

Fig. 2. Lagrange interpolation for MVs recovery [13].

estimated reconstructed image block is denoted by $\hat{B}_R(x_0, y_0)$, where (x_0, y_0) is the upper left coordinate of the block.

$$\begin{cases} D_A = \sum_{x=x_0}^{x_0+N-1} \left(\hat{P}_R(x, y_0, n) - P_R(x, y_0-1, n)\right)^2 \\ D_L = \sum_{y=y_0}^{y_0+N-1} \left(\hat{P}_R(x_0, y, n) - P_R(x_0-1, y, n)\right)^2 \\ D_B = \sum_{x=x_0}^{x_0+N-1} \left(\hat{P}_R(x, y_0+N-1, n) - P_R(x, y_0+N, n)\right)^2 \end{cases} \tag{2}$$

Then the BMA chooses the MV $\hat{d}(\hat{d}_x, \hat{d}_y)$ among a set of candidate vectors, which produces the smallest total variation

$$D(d_x, d_y) = D_A + D_L + D_B \tag{3}$$

The algorithm is based on the fact that the image pixels in the original image are highly correlated. Thus if the correct motion is chosen, the pixel variation among neighboring blocks should be at a minimum. The choice of including each individual component variation function in D depends on the availability of the block [17]–[19].

B. Lagrange Interpolation

LI is an efficient method for recovering the MVs of the small blocks (4×4) [13]. As shown in Fig. 2, $MB_{i,j}$ is the lost MB (in gray), where i and j denote the horizontal and vertical coordinates respectively. $MB_{i-1,j}$, $MB_{i,j-1}$, $MB_{i+1,j}$ and $MB_{i,j+1}$ represent the surrounding MBs around the corrupted MB. MV_0, MV_1, MV_2 and MV_3 represent the MVs of corresponding 4×4 blocks respectively.

In order to achieve the MVs of the lost blocks with LI, the interpolation function $I(x)$ is defined as [13]:

$$\begin{aligned} I(x) = {} & MV_0 \left[-\frac{(x+2.5)(x-2.5)(x-3.5)}{42}\right] \\ & + MV_1 \left[\frac{(x+3.5)(x-2.5)(x-3.5)}{30}\right] \\ & + MV_2 \left[-\frac{(x+3.5)(x+2.5)(x-3.5)}{30}\right] \\ & + MV_3 \left[\frac{(x+3.5)(x+2.5)(x-2.5)}{42}\right] \end{aligned} \tag{4}$$

Then the MVs of two lost blocks V_0 and V_3 are obtained as [13]:

$$\begin{cases} V_0 = I(-1.5) \\ V_3 = I(1.5) \end{cases} \tag{5}$$

In order to make the movement change smoothly in the lost MB, LI uses V_0, V_3 and neighboring MVs to calculate V_1 and V_2 [13]:

$$V_1 : \begin{cases} x = -0.5 \\ I(x) = MV_1\left[-(x+1.5)\right] + V_0[x+2.5] \end{cases} \tag{6}$$

$$V_2 : \begin{cases} x = 0.5 \\ I(x) = V_3\left[-(x-2.5) + MV_2[x-1.5]\right. \end{cases} \tag{7}$$

By iterating this method, all the MVs of 4 × 4 blocks in the lost MB, which are represented as $V_0^h, V_1^h, \ldots, V_{15}^h$, are recovered in the horizontal direction. Similarly, we can get the $V_0^v, V_1^v, \ldots, V_{15}^v$ by using the same approach in the vertical direction [13].

Finally, LI uses the average of estimated MVs in two directions to recover the MVs of every 4×4 block in the corrupted MB as [13]:

$$V_i^{rec} = \frac{\left(V_i^H + V_i^V\right)}{2}, \qquad i = 0, 1, \ldots, 15 \tag{8}$$

III. Our Proposed Method

LI is a simple method that recovers the MVs by assuming that the lost MVs are limited in the range of the MVs of the

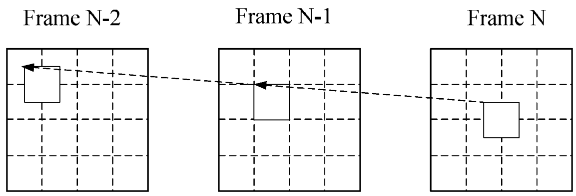

Fig. 3. The extrapolated MB from previous frames [20].

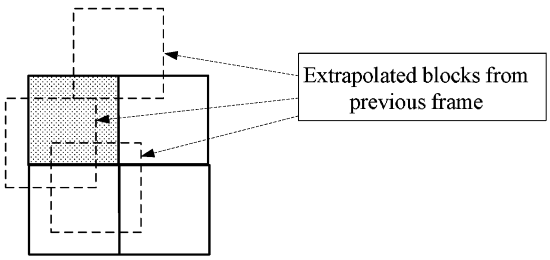

Fig. 4. Extrapolated blocks overlap with the lost block.

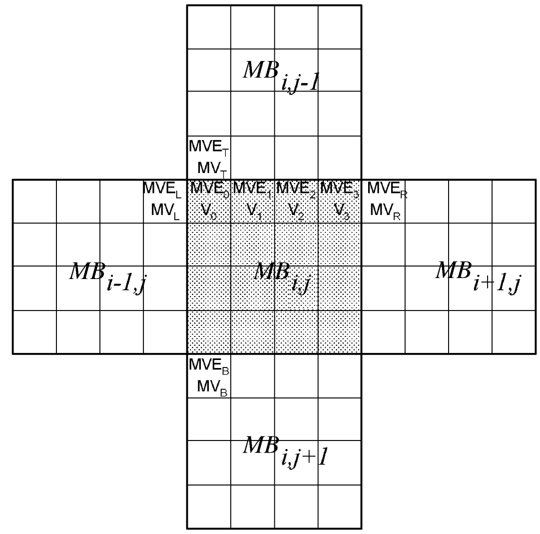

Fig. 5. MV estimation with the extrapolated MVs.

surrounding blocks. However, it may not be true for some sequences. For example, according to LI, if $MV_1 < MV_2$, then V_0, V_1, V_2 and V_3 as shown in Fig. 2 should satisfy:

$$MV_1 < V_0 < V_1 < V_2 < V_3 < MV_2 \tag{9}$$

However, V_0, V_1, V_2 and V_3 may not be in the range of (MV_1, MV_2) limited by MV_1 and MV_2 for some sequences. As a result, the estimated MVs of the lost blocks may not be accurate. In order to solve this problem and further improve the performance, we propose a new method to recover the MVs of the lost blocks.

Peng *et al.* propose to use MVE in order to achieve the MVs of the lost MBs [20]. In this method, the MVs of MBs are first extrapolated from the last decoded frame to the corrupted frame, as shown in Fig. 3. Then the overlapped area between the missing MB and the motion extrapolated MB is estimated and the best MV is selected. This method is able to obtain lost MVs, but the MB (16×16) based MV is too rough to cause block artifacts.

In H.264/AVC, the smallest unit for motion estimation and compensation is a block (4×4). Thus in our proposed algorithm, we use a block as the concealment unit. As shown in Fig. 4, the MV of the lost block (in gray) is estimated according to the extrapolated blocks, which occupy the missing block. The number of pixels in the overlapped areas is used to obtain the weight for the estimation. Let EB_n^j denote the extrapolated 4×4 block from the jth block in the reference frame to the corrupted frame n, $MV(EB_n^j)$ denotes the MV of EB_n^j, and B_n^i denotes the ith 4×4 block in frame n. Then the weight is given by [20]:

$$w_n^{i,j} = \sum_{p \in B_n^i} f_j(p), \quad i, j = 1, 2, \ldots, M \tag{10}$$

where

$$f_j(p) = \begin{cases} 1, & p \in EB_n^j \\ 0, & p \notin EB_n^j \end{cases} \tag{11}$$

M is the total number of blocks in a video frame.

Then the MV of the lost block B_n^i, denoted by $MV(B_n^i)$, is given by:

$$MV\left(B_n^i\right) = MV\left(EB_n^{j^*}\right) \tag{12}$$

where

$$j^* = \arg\max\left\{w_n^{i,j}\right\}, \quad j = 1, 2, \cdots, M$$

If block B_n^i has not been overlapped by any extrapolated blocks, the MV of this block will be the average MV of the surrounding blocks.

After motion extrapolation, the extrapolated MVs of all the blocks in the corrupted frame can be achieved. As shown in Fig. 5, MVE_L and MVE_R are the extrapolated MVs of the left and right 4×4 blocks of the corrupted MB respectively. MV_L and MV_R denote the actual MVs of these two blocks respectively. MVE_0, MVE_1, MVE_2 and MVE_3 denote the extrapolated MVs of the corresponding lost blocks.

Then we will judge if the extrapolated MVs satisfy the following conditions:

$$\begin{cases} |MVE_L(x) - MV_L(x)| + |MVE_R(x) - MV_R(x)| \le Th \\ |MVE_L(y) - MV_L(y)| + |MVE_R(y) - MV_R(y)| \le Th \end{cases} \tag{13}$$

In our simulation, the typical value of Th is set as 16.

If the extrapolated MVs don't satisfy either condition of (13), it means that the MVE is not accurate enough for MVs recovery. In this case, BMA is used for error concealment by considering the MVs of neighboring blocks as the MV candidates.

Conversely, if both conditions of (13) are satisfied, it means that the extrapolated MVs are accurate enough to be used in MV recovery. In this case, our proposed method will take the following steps for temporal error concealment.

The MV of the lost block, such as V_0, should satisfy:

$$\frac{(V_0 - MV_L)}{(MVE_0 - MVE_L)} = \frac{(MV_R - MV_L)}{(MVE_R - MVE_L)} \tag{14}$$

TABLE I
COMPARISON OF THE AVERAGE PSNR PERFORMANCE OVER ERRONEOUS FRAMES ONLY FOR DIFFERENT PLRS

Sequence	Method	Packet Loss Rate			
		5%	10%	15%	20%
Mobile	BMA	24.53	23.04	22.46	22.01
	LI	23.34	22.13	21.37	21.01
	Proposed	25.42	24.08	23.76	23.14
Flower	BMA	26.00	24.54	24.23	23.12
	LI	26.65	25.05	24.25	23.21
	Proposed	26.90	25.44	24.99	23.75

TABLE II
COMPARISON OF THE AVERAGE PSNR PERFORMANCE OVER ALL FRAMES FOR DIFFERENT PLRS

Sequence	Method	Packet Loss Rate			
		5%	10%	15%	20%
Mobile	BMA	32.02	30.19	26.91	25.53
	LI	31.7	29.72	26.12	24.78
	Proposed	32.34	30.69	27.76	26.37
Flower	BMA	33.14	31.45	28.54	26.61
	LI	33.47	31.77	28.54	26.71
	Proposed	33.48	31.90	29.05	27.09

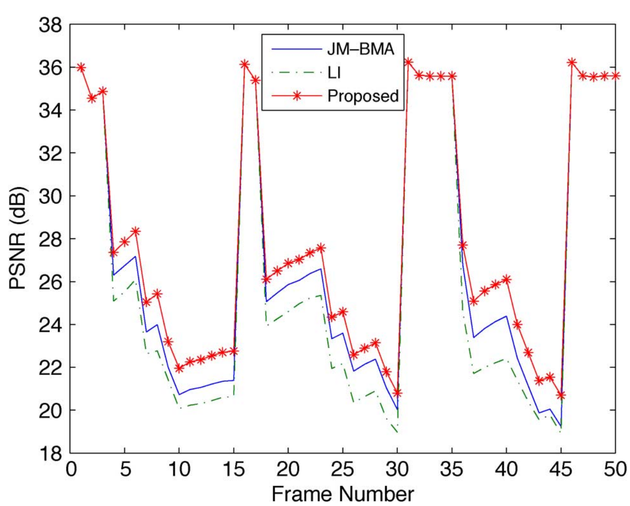

Fig. 6. "Mobile" sequence PSNR comparison vs. frame number when $\mathrm{PLR} = 15\%$.

Thus V_0 can be calculated as:

$$V_0 = \frac{(MV_R - MV_L)}{(MVE_R - MVE_L)}(MVE_0 - MVE_L) + MV_L \quad (15)$$

By using the same approach, we can obtain the MVs of 16 4×4 blocks as $V_0^h, V_1^h, \ldots, V_{15}^h$. If the vertical neighboring blocks are not corrupted, we can also apply our proposed method in the vertical direction. As a result, $V_0^v, V_1^v, \ldots, V_{15}^v$ can also be achieved.

Finally, the reconstructed MV of each lost block is obtained by Eq. (8). If one direction is not available for MV interpolation, we may use the other available one. If neither is available, we may use BMA for MV recovery. After all MVs of lost blocks are recovered, the corrupted frame can be concealed.

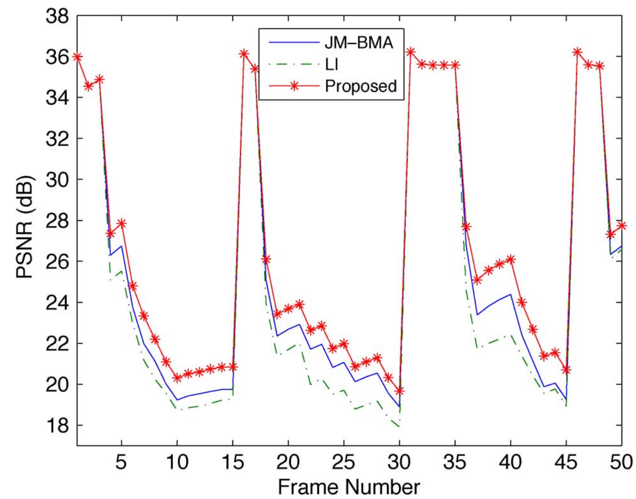

Fig. 7. "Mobile" sequence PSNR comparison vs. frame number when $\mathrm{PLR} = 20\%$.

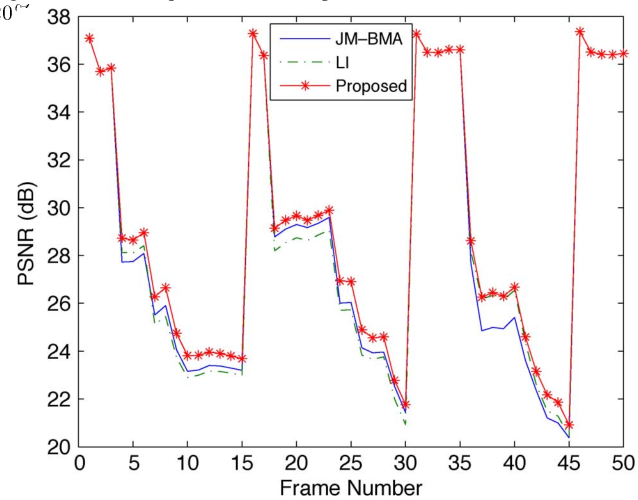

Fig. 8. "Flower" sequence PSNR comparison vs. frame number when $\mathrm{PLR} = 15\%$.

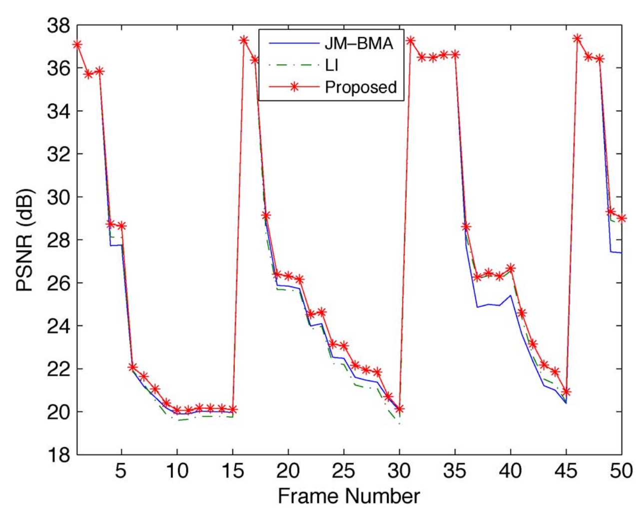

Fig. 9. "Flower" sequence PSNR comparison vs. frame number when $\mathrm{PLR} = 20\%$.

IV. SIMULATION RESULTS

In order to evaluate the performance of the proposed method, we simulated it with JM 15.0 H.264/AVC codec [12]. Two

(a) (b) (c) (d) (e)

Fig. 10. Restored 4th frame of "Mobile" sequence: (a) original frame; (b) corrupted frame; (c) recovered with BMA; (d) recovered with LI; (e) recovered with our proposed method. (Gains over BMA and LI are 1.06 dB and 2.27 dB respectively.)

(a) (b) (c) (d) (e)

Fig. 11. Restored 4th frame of "Flower" sequence: (a) original frame; (b) corrupted frame; (c) recovered with BMA; (d) recovered with LI; (e) recovered with our proposed method. (Gains over BMA and LI are 1.00 dB and 0.60 dB respectively.)

standard CIF (352×288) video sequences, including "Mobile" and "Flower", are encoded by H.264/AVC codec. The frame rate is 30 frames/second and the period of I frame reset is 15. A constant quantization parameter (QP) of 28 is maintained for all frames. The dispersed FMO is enabled, thus a frame contains two slices. As far as the packetization scheme is concerned, all the compressed video streams related to one slice are encapsulated into one packet. In this case, packet loss means slice loss. Different packet loss rates (PLRs) of 5%, 10%, 15% and 20% are simulated in order to evaluate the various network conditions.

In this simulation, we compare performances of the proposed method with BMA used in the JM software package [12] and LI [13]. Table I presents the average PSNR performances over erroneous frames only for different PLRs, which are defined as frames corrupted by slice losses. As shown in this table, the proposed algorithm yields a higher average PSNR performance than BMA and LI, and is able to provide up to 1.3 dB and 2.39 dB better PSNR performances than BMA and LI respectively. Therefore our proposed method is quite effective in concealing erroneous frames.

In order to evaluate the improvement of video quality, Table II presents the PSNR performances averaged over all frames for different sequences with different PLRs. As shown in this table, the proposed method can improve the average PSNR performance by up to 0.85 dB and 1.64 dB, compared with BMA and LI, respectively.

Figs. 6–9 show the simulation results with PSNR vs. frame number for different test sequences with $\mathrm{PLR} = 15\%$ and 20% respectively. Due to accurate MV interpolation with the help of MVE, the proposed algorithm is able to capture more accurate MVs than LI, and leads to concealed frames with a much higher PSNR performance. Experimental results report that the proposed algorithm significantly outperforms BMA and LI for different sequences.

For subjective evaluation, Fig. 10 shows the results of one frame extracted from the "Mobile" sequences, where (a) and (b) are the original and erroneous frames respectively. (c) to (e) are the frames reconstructed using BMA, LI and the proposed algorithm respectively. In Fig. 11, it can be seen that the image quality is consistent with the PSNR measurement. In this figure, (e) is perceptually superior to (c) and (d), especially around the edges of the image objects.

It should be noted that although the proposed method assumes that four neighboring MBs of the corrupted MB are correct (e.g. dispersed FMO mode), it can also work if the two neighboring MBs in the horizontal or vertical direction are correct. If the PLR is so high that no neighboring MBs are correct, simple motion copy can be used for concealing the lost MB.

V. Conclusion

In this paper, we present an efficient method for MVs recovery to combat packet loss during video transmission over error-prone networks. As opposed to conventional methods, our proposed method is able to recover the MVs of lost blocks more accurately by MV interpolation with the help of extrapolated MVs. Simulation results have demonstrated that our proposed method is capable of providing better visual quality than other methods for different sequences with different PLRs. In conclusion, our method is highly effective and efficient for video transmission.

References

[1] T. Wiegand, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC," ITU-T Rec.H.264 | ISO/IEC 14496-10 AVC, Mar. 2003.

[2] M. H. Pinson, S. Wolf, and G. Cermak, "DTV subjective quality of H.264 vs. MPEG-2, with and without packet loss," *IEEE Trans. Broadcast.*, vol. 56, no. 1, pp. 86–91, Mar. 2010.

[3] D. A. Adjeroh and S. D. Sawant, "Error-resilient transmission for 3D DCT coded video," *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 178–189, Jun. 2009.

[4] B. Yan and H. Gharavi, "A hybrid frame concealment algorithm for H.264/AVC," *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 98–107, Jan. 2010.

[5] M. Friebe and A. Kaup, "Fading techniques for error concealment in block-based video decoding systems," *IEEE Trans. Broadcast.*, vol. 53, no. 1, pp. 286–296, Mar. 2007.

[6] T. Thaipanich, P. H. Wu, and C.-C. J. Kuo, "Low-complexity video error concealment for mobile applications using OBMA," *IEEE Trans. Consum. Electron.*, vol. 54, no. 2, May 2008.

[7] S. Valente, C. Dufour, F. Groliere, and D. Snook, "An efficienct error concealment implementation for MPEG-4 video streams," *IEEE Trans. Consum. Electron.*, vol. 47, no. 3, Aug. 2001.

[8] S. C. Huang and S. Y. Kuo, "Optimization of Hybridized Error Concealment for H. 264," *IEEE Trans. Broadcast.*, vol. 54, no. 3, pp. 499–516, Sep. 2008.

[9] M. C. Hwang, J. H. Kim, D. T. Duong, and S. J. Ko, "Hybrid temporal error concealment methods for block-based compressed video transmission," *IEEE Trans. Broadcast.*, vol. 54, no. 2, pp. 198–207, Jun. 2008.

[10] G. Dane, Y. Ye, and Y. C. Lee, "Low-complexity temporal error concealment by motion vector processing for mobile video applications," in *IEEE Int. Conf. Image Process. (ICIP')*, Oct. 2008, pp. 3056–3059.

[11] Z. Huang and B. Yi, "A new temporal error concealment algorithm for {H.264/AVC} using motion strength of neighboring area," in *IEEE Int. Symp. Knowledge Acquisition and Modeling Workshop*, Dec. 2008, pp. 818–821.

[12] H.264/AVC Reference Software JM15.0 [Online]. Available: http://iphome.hhi.de/suehring/tml/download

[13] J. Zheng and L. P. Chau, "A temporal error concealment algorithm for {H.264} using Lagrange interpolation," in *Int. Symp. Circuits Syst. (ISCAS)*, May 2004, vol. 2, pp. II–133–II–136.

[14] M.-J. Chen, L.-G. Chen, and R.-M. Weng, "Error concealment of lost motion vectors with overlapped motion compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 3, pp. 560–563, Jun. 1997.

[15] L.-W. Kang and J.-J. Leou, "A new hybrid error concealment scheme for {MPEG}-2 video transmission," in *Proc. IEEE Workshop Multimedia Signal Process.*, 2002, pp. 29–32.

[16] Y. Zhang and K. K. Ma, "Error concealment for video transmission with dual multiscale markov random field modeling," *IEEE Trans. Image Process.*, vol. 12, no. 2, pp. 236–242, Feb. 2003.

[17] C.-J. Huang, H.-S. Wong, and J.-H. Wang, "Selective block matching for error concealment in video system," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Oct. 2002, vol. 3, pp. 360–364.

[18] J. Zhang, J. F. Arnold, and M. R. Frater, "A cell-loss concealment technique for MPEG-2 coded video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 4, pp. 659–665, Jun. 2000.

[19] W.-Y. Kung, C.-S. Kim, and C.-C. J. Kuo, "A dynamic error concealment for video transmission over noisy channels," in *Proc. of IEEE GLOBECOM'02*, Nov. 2002, vol. 2, pp. 17–21.

[20] Q. Peng, T. Yang, and C. Zhu, "Block-based temporal error concealment for video packet using motion vector extrapolation," in *IEEE Int. Conf. Commun., Circuits Syst. West Sino Expo.*, Jul. 2002, vol. 1, pp. 10–14.