Edge-Directed Error Concealment

Mengyao Ma, Oscar C. Au, *Senior Member, IEEE,* S.-H. Gary Chan, *Senior Member, IEEE,* and Ming-Ting Sun, *Fellow, IEEE*

***Abstract*—In this paper we propose an edge-directed error concealment (EDEC) algorithm, to recover lost slices in video sequences encoded by flexible macroblock ordering. First, the strong edges in a corrupted frame are estimated based on the edges in the neighboring frames and the received area of the current frame. Next, the lost regions along these estimated edges are recovered using both spatial and temporal neighboring pixels. Finally, the remaining parts of the lost regions are estimated. Simulation results show that compared to the existing boundary matching algorithm [1] and the exemplar-based inpainting approach [2], the proposed EDEC algorithm can reconstruct the corrupted frame with both a better visual quality and a higher decoder peak signal-to-noise ratio.**

***Index Terms*—Edge extraction, error concealment, error resilience, FMO, image inpainting, structure region.**

I. Introduction

DELIVERING video of good quality over the Internet or wireless networks is very challenging, due to the use of predictive coding and variable length coding (VLC) in video compression [3], [4]. In the block-based video coding method, if we use INTER prediction mode, each macroblock (MB) is predicted from a previously decoded frame by *motion compensation*. If data loss occurs during the transmission, the corresponding frame will be corrupted, and this error will propagate to the subsequent frames because of INTER-prediction. In addition, a simple bit error in VLC can cause desynchronization; as a result, all the following bits cannot be used until a synchronization code arrives. Due to these facts, it is useful to develop error resilience (ER) and error concealment (EC) techniques to control and recover from the errors in video transmission. Error resilience is usually applied at the encoder side. The coding efficiency of an ER codec is lower than a normal codec, because the encoder needs to introduce some redundancy to the stream. In the case of error, the decoder would use this additional information to reconstruct the video. On the other hand, error concealment is applied at the decoder side. It requires no change to the encoder and does not increase the bit rate.

If each video frame is encoded into several slices which are packed and transmitted separately, when loss occurs during the transmission, it is possible that only a portion of a video frame is corrupted and thus can be estimated based on both spatial and temporal correlations between neighboring pixels. Flexible macroblock ordering (FMO) has been developed in part for such applications, which is one of the new features of the video compression standard H.264/advanced video coding (AVC). In FMO, the video frame is divided into several independently-decodable slices, and each slice consists of a sequence of MBs. Similar to other slice coding approaches, in FMO, the prediction beyond slice boundaries is also forbidden so as to prevent error propagation from intra-frame predictions, and thus can help to improve the error robustness of the compressed video [5], [6]. Fig. 1 shows two popularly used MB scan patterns in FMO, i.e., the checkerboard/scattered mode and the interleaving mode. One frame is encoded into two slices, and the blocks with the same color are grouped into one slice. When MBs are arranged in such fashions, error concealment schemes can perform very well as the lost MBs can be reconstructed by their surrounding MBs. It has been demonstrated that the distortion in a recovered block tends to increase with its distance to the nearest error-free blocks [5], [7]. The objective behind the use of FMO as an error resilience tool is to equally scatter possible errors to the whole frame, so as to make errors easily concealed compared to those concentrated in a small region [6].

When the checkerboard mode in Fig. 1(a) is used in FMO, predictions between neighboring MBs are not used so as to avoid error propagation from one slice to another. As a result, the coding efficiency is reduced due to the overhead bits accompanied with this scheme [6]. In an error-free environment, the bit rate penalty was found to be less than 5% for six of the seven common condition video sequences encoded at quantizer parameter (QP) = 16, and about 20% for the worst sequence encoded at QP = 28 [5]. These results were derived using H.26L, and we expect that similar results can be observed in H.264. When the interleaving mode in Fig. 1(b) is used to encode the MBs, prediction from upper MBs is

Manuscript received February 17, 2009; revised May 30, 2009. First version published November 3, 2009; current version published March 5, 2010. This work was supported in part by the Innovation and Technology Commission of the Hong Kong Special Administrative Region (HKSAR), China under Project GHP/048/08, the General Research Fund from the Research Grant Council of HKSAR (611209), and the Hong Kong Applied Science and Technology Research Institute (ASTRI). This paper was recommended by Associate Editor V. Bottreau.

M. Ma is with the Hong Kong Applied Science and Technology Research Institute Company Ltd., Shatin, Hong Kong (e-mail: mengyao.cb@gmail.com.).

O. C. Au is with the Multimedia Technology Research Center, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong (e-mail: eeau@ust.hk).

S.-H. G. Chan is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, and also with Microsoft Research Asia, Beijing, China (e-mail: gchan@ust.hk).

M.-T. Sun is with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: sun@ee.washington.edu).

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/TCSVT.2009.2035839

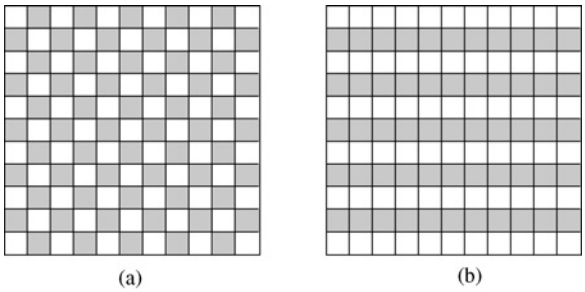


Fig. 1. FMO checkerboard mode and interleaving mode [6]. (a) FMO checker board mode. (b) FMO checker interleaving mode.

forbidden while prediction from the left one is still available. So a lost slice can be error-concealed using its upper and lower MB rows, if the slice containing these MBs is received. Compared to the checkerboard mode, the interleaving mode is better in terms of coding efficiency, but less capable in terms of error recovery ability [6]. It is worthy to note that these two FMO modes are essentially two implementations of the spatial sub-sampling multiple description coding (MDC) approach [8]. In MDC, the video stream is divided into independently-encoded streams (descriptions), which are sent to the destination through different channels. If error occurs during the transmission, only a subset of the descriptions will be received by the decoder, which can be used to reconstruct the video with lower but acceptable quality. Specially in spatial sub-sampling MDC, each frame is divided into multiple descriptions using, for example, the motion vectors of the blocks [9], the residues [10], [11], or even the blocks themselves [12]. The implementations of the last one and the checkerboard FMO mode are very similar.

Several error concealment algorithms have been proposed for MB losses in FMO, such as spatial interpolation based on edge direction [13] and motion-compensated temporal prediction based on boundary matching [14]. Image inpainting techniques have also been adopted in recent EC algorithms [15], [16]. In [2], an exemplar-based image inpainting approach is proposed. The authors demonstrate that the quality of the synthesized image is highly influenced by the order in which the filling process proceeds. So they propose to give higher synthesis priority to those regions lying on the continuation of image structures. The EC algorithms proposed in [15] and [16] are both based on this exemplar-based synthesis approach. A number of EC algorithms based on direction/edge information in image and video frames have also been developed to error-conceal lost blocks. In [17], the authors propose a spatial directional interpolation scheme in which the surrounding pixels of a lost block are first used to reveal the local geometric structure. The estimated edges can divide the lost block into several regions, and the missing pixels in each region are differently estimated using directional filtering. It is assumed that the number of cross-points between estimated edges and the lost block is always even, which may not be true for a natural texture image. This assumption is avoided in [18], as in this scheme one edge can meet another edge within the lost MB and does not exit the MB. The authors in [19] propose a spatial shape EC technique in the context of object-based image and video coding schemes. Based on the available surrounding contours, the missing shape data of lost blocks are interpolated using Bézier curves. All the approaches in [17]–[19] use only spatial information to estimate the corrupted structure/shape data. In [20], the boundary matching algorithm proposed in [1] is extended such that the best match of a lost MB is temporally searched based on the edge characteristics of neighboring MBs instead of their pixel values. In [21], the authors propose a two-stage EC algorithm. After an initial estimation of the missing blocks, a maximum *a posteriori* (MAP) estimator is used to further improve the reconstructed video quality. It uses an eight-pixel clique (with eight directions) around each pixel as the image *a priori* model. The MAP estimation of missing pixels is also used in the EC algorithm in [22], where previously interpolated pixels are also used in the recovery process of subsequent ones. This sequential approach can improve the capability of recovering important image features; nevertheless, it also causes the error propagation problem. To alleviate this, a linear average of eight recovered versions from different scanning orders is used as the final result.

In [23], an image compression framework is presented where inpainting techniques are used to remove visual redundancy inherent in natural images. Only edges and necessary exemplars (blocks) are extracted and compressed at the encoder side. Based on this assistant information, other regions of the image are intentionally skipped during encoding, and will be restored at the decoder using an edge-based inpainting approach. The structure regions along the obtained edges are inpainted first, and then the remaining regions are filled by texture synthesis. Motivated by this, we propose an edge-directed error concealment (EDEC) algorithm for FMO-encoded video streaming over the Internet. When packet (slice) loss occurs, the strong edges in a corrupted frame will be estimated first based on the edges in the neighboring frames and the received area of current frame. Then, the lost regions along these estimated edges will be recovered using both spatial and temporal neighboring pixels. Finally, the remaining parts of the lost regions are estimated. A patch-based approach is used to fill the erroneous regions, as compared to a pixel-based one, it can not only improve the execution speed but also improve the accuracy of the propagated structures [2]. Different from the approaches in [17], [18], and [20]–[22], edges in our algorithm may not be straight ones. An edge is defined to be a set of 8-connected pixels to make the EC algorithm more general for natural images. We will use the checkerboard mode in Fig. 1(a) to illustrate our proposed algorithm. However, it is worthy to note that our algorithm can be easily extended to error-conceal the lost MBs in other FMO modes.

The rest of this paper is organized as follows. In Section II, we will describe the proposed EDEC algorithm. Its performance will be demonstrated in Section III by both subjective and objective results. Section IV is the conclusion.

II. EDEC

Suppose each video frame is encoded into two slices based on the FMO mode in Fig. 1(a); slice 1 (L_1) contains all the

white MBs and slice 2 (L_2) contains all the grey MBs. Then, these two slices are packed and transmitted separately over the Internet. Without loss of generality, suppose L_1 of frame $\psi(n)$ is lost during the transmission. As predictions between neighboring MBs are abandoned in checkerboard FMO, L_2 of $\psi(n)$ can be correctly received and decoded. We will use both the decoded L_2 and the previous frame $\psi(n-1)$ to error-conceal the lost MBs in L_1 in our proposed algorithm. In detail: 1) the strong edges in frame $\psi(n)$ will be estimated first based on the edges in the previous frame $\psi(n-1)$ and the received area of frame $\psi(n)$; 2) then the lost regions along these estimated edges (*structure regions*) will be estimated using both spatial and temporal neighboring pixels; and 3) finally, the remaining parts of the lost regions are estimated. We name our proposed approach EDEC algorithm, which will be explained step by step in the subsequent sections. The intermediate result after each step is shown in Fig. 2 for a better illustration.

A. Initialization

1) *Hierarchical Boundary Matching:* In order to assist the execution of the three main steps of our algorithm, we first use a hierarchical boundary matching (HBM) approach to estimate the motion vector (MV) of each 4×4 block in frame $\psi(n)$. The original-size frame is first down-sampled to a coarse frame $\grave{\psi}(n)$, based on which the boundary matching algorithm in [1] will be applied to find the MV of each block. Then using these obtained MVs in $\grave{\psi}(n)$ as an initial estimation, the MV of each 4×4 block in $\psi(n)$ will be refined. By first applying motion estimation in the down-sampled video frame, we not only reduce the complexity of motion search but also improve the accuracy of the initially estimated MVs. As the low-pass filter used in down-sampling can help to reduce the noise in video frame, the number of erroneously estimated MVs can be reduced. Note that as *tree structured motion compensation* is used in the H.264/AVC standard [24], i.e., each MB can be partitioned into motion-compensated sub-blocks, each partition or sub-block can have a separate motion vector. So for each received 4×4 block, we will set its MV to that of the partition containing this block, which can be directly retrieved from the received bitstream. HBM is only applied to the corrupted blocks to estimate their MVs.

In detail, given a scale factor s, suppose the down-sampled frame of $\psi(n)$ is $\grave{\psi}_s(n)$. Then each MB in $\psi(n)$ becomes a smaller block in $\grave{\psi}_s(n)$, named $\grave{\text{MB}}$. Using the boundary matching algorithm in [1], we can estimate the MV of each corrupted $\grave{\text{MB}}$ in $\grave{\psi}_s(n)$. The median value of the MVs of received neighboring blocks is used as the starting point for MV search. After this, suppose the estimated MV for the ith $\grave{\text{MB}}$ is $\grave{MV}_i$. Then as the resolutions of $\grave{\psi}_s(n)$ and $\psi(n)$ are different, $\grave{MV}_i$ is scaled by factor s to get the initial estimation for the MV of the ith MB, MB_i, in $\psi(n)$, i.e., $MV_i^0 = \grave{MV}_i \times s$. Finally, MV_i^0 is used as the starting point to search the MV of each 4×4 block within MB_i. In our experiments, scale factor s is equal to 4 and pixel-averaging is used as the down-sampling approach.

After we obtain the MV of a corrupted 4×4 block, we will also error-conceal this block by copying pixels from $\psi(n-1)$ based on the estimated MV. So after the HBM process, an error-concealed frame for $\psi(n)$ will be obtained. Suppose it is $\psi_h(n)$. Both $\psi_h(n)$ and the estimated MVs of 4×4 blocks will be used to assist the *edge restoration* process introduced later.

2) *Attribute Vector:* For each pixel p in frame $\psi(n)$, we use an *attribute vector* $(Y_p, U_p, V_p, A_p, C_p, S_p)$ to keep its structure and pixel value information. Specifically, pixel value attributes Y_p, U_p, and V_p save the Y, U, and V components of p, respectively. For a lost pixel p, its initial values of Y_p, U_p, and V_p are *NA* (not available). Direct current (DC) attribute A_p saves the average Y-component values for a 3×3 neighborhood centered at p. Note that only the neighboring pixels with available Y values are used to calculate the DC attribute. C_p is the confidence attribute of p, which is initially set to be 1 for an error-free pixel and 0 for a lost pixel. The larger C_p is, the more trustworthy p becomes. Structure attribute S_p represents whether p is an edge pixel or not; its value can be $\widetilde{NE}$ (not edge pixel), $\widetilde{E}$ (edge pixel), or $\widetilde{PE}$ (possible edge pixel). We name a pixel p an *E-Pixel* if its structure attribute S_p is $\widetilde{E}$. To obtain the initial structure attribute values for all the pixels in $\psi(n)$, we first apply a canny edge detector [25] on frame $\psi_h(n)$ which is obtained by hierarchical boundary matching in Section II-A1. After this, for each pixel p in $\psi(n)$:

- If its corresponding pixel in $\psi_h(n)$ is marked as edge by the canny operator:
  - if p belongs to the received slice (L_2 of $\psi(n)$ in this example) but does not lie on any MB boundary, the structure attribute of p is set to be $\widetilde{E}$ (edge pixel);
  - otherwise, the structure attribute of p is set to be $\widetilde{PE}$ (possible edge pixel).
- Otherwise, its structure attribute S_p is set to be $\widetilde{NE}$ (not edge pixel).

Similarly, we also compute the attribute vector for each pixel q in frame $\psi(n-1)$. If $\psi(n-1)$ has been corrupted and error-concealed, each pixel will have a confidence attribute value computed by EDEC previously. Otherwise, the confidence attributes of the pixels in $\psi(n-1)$ are propagated from their reference pixels in $\psi(n-2)$ when $\psi(n-1)$ is decoded. If all the previous frames have been correctly received, the confidence attributes are set to 1. In addition, as it has been demonstrated in the literature that deblocking filters and sub-pixel motion compensation can help to attenuate the propagated error energy [26], [27], we will refresh the confidence attributes of all the pixels to be 1 if the time between the last corrupted frame and $\psi(n-1)$ is long enough. We also apply a canny edge detector on frame $\psi(n-1)$ to get the pixel structure attribute values. If a pixel q with $C_q = 1$ is marked as edge by the canny operator, its structure attribute S_q is set to be $\widetilde{E}$; otherwise, S_q is set to be $\widetilde{NE}$. One major difference between a pixel q in $\psi(n-1)$ and a pixel p in $\psi(n)$ is that the structure attribute of q can only be $\widetilde{NE}$ (not edge pixel) or $\widetilde{E}$ (edge pixel).

We define the *attribute difference* between two pixels p and q as

$$d_a(p,q)=\sqrt{(Y_p-Y_q)^2+(U_p-U_q)^2+(V_p-V_q)^2+(S_p-S_q)^2}. \tag{1}$$

In our experiments, we use $\widetilde{NE}=0$, $\widetilde{E}=255$, and $\widetilde{PE}=200$.

3) *P-Edges and R-Edges:* As we have introduced previously, we will estimate the strong edges in the corrupted frame $\psi(n)$ based on the edges in both the previous frame $\psi(n-1)$ and the received slice (L_2) of $\psi(n)$. To better illustrate our proposed algorithm, we define two kinds of edges, *potential edge* (P-Edge) and *reference edge* (R-Edge). P-Edges are the initial estimated edges in the corrupted frame $\psi(n)$, and R-Edges are the reference edges in $\psi(n-1)$ and will be transformed into $\psi(n)$ to refine the uncertain parts of related P-Edges. Specifically, a P-Edge $\hat{e}$ is defined to be a set of 8-connected pixels in $\psi(n)$ with similar DC attribute values and with structure attributes being $\widetilde{E}$ or $\widetilde{PE}$, i.e., $\hat{e}=\{p: p\in\psi(n), S_p=\widetilde{E} \text{ or } S_p=\widetilde{PE}, \text{ and } \exists\ p'\in\hat{e} \text{ s.t. } p'\neq p \text{ and } |A_p-A_{p'}|\le T_a \text{ and } \max(|x_p-x_{p'}|,|y_p-y_{p'}|)\le 1\}$. Here T_a is a threshold for DC attribute difference, and (x_p, y_p) is the coordinate of pixel p in frame $\psi(n)$. T_a is trained to be 30 in our experiments. By the definition, we can see that a P-Edge can contain some uncertain parts where the structure attributes of pixels are $\widetilde{PE}$. These uncertain parts will be refined by the *edge restoration* process in Section II-B. Compared with P-Edge, an R-Edge $\bar{e}$ is a set of 8-connected pixels in $\psi(n-1)$ with similar DC attribute values and with structure attributes being $\widetilde{E}$, i.e., $\bar{e}=\{q: q\in\psi(n-1), S_q=\widetilde{E}, \text{ and } \exists\ q'\in\bar{e} \text{ s.t. } q'\neq q \text{ and } |A_q-A_{q'}|\le T_a \text{ and } \max(|x_p-x_{p'}|,|y_p-y_{p'}|)\le 1\}$. We define $\mathbb{E}_P^n$ to be the set of all the P-Edges in $\psi(n)$, as shown in Fig. 3(b), and $\mathbb{E}_R^{n-1}$ to be the set of all the R-Edges in $\psi(n-1)$.

B. Edge Restoration

The key idea of the *edge restoration* process in our algorithm is that for a P-Edge $\hat{e}$ in frame $\psi(n)$, we search in the reference frame $\psi(n-1)$ to check whether there exists an R-Edge $\bar{e}$ that matches $\hat{e}$ well. If such a reference edge $\bar{e}$ exists, we will transform $\bar{e}$ into $\psi(n)$ to refine the uncertain parts of $\hat{e}$. Furthermore, in order to obtain $\bar{e}$, a motion vector $mv_{\hat{e}}$ needs to be estimated for $\hat{e}$ first, and then the reference edge of $\hat{e}$ will be searched around $mv_{\hat{e}}$.

In detail, for each P-Edge $\hat{e}_i\in\mathbb{E}_P^n$, the *edge restoration* process proceeds as follows.

1) Estimate the motion vector $mv_{\hat{e}_i}$ of $\hat{e}_i$.
   a) Calculate the initial MV of $\hat{e}_i$, i.e., $mv_{\hat{e}_i}^0$. Note that we have estimated and maintained the MVs of all the 4×4 blocks of $\psi(n)$ in the HBM of Section II-A1. We set $mv_{\hat{e}_i}^0$ to the medium MV of the blocks that $\hat{e}_i$ passes through.
   b) Using $mv_{\hat{e}_i}^0$ as the starting point, an integer motion search is performed in $\psi(n-1)$ to obtain the MV of $\hat{e}_i$. The search range used is $[-4, 4]$. Specifically, given an MV for $\hat{e}_i$, i.e., mv, the MV search distortion for $\hat{e}_i$ is defined to be

$$d_e(\hat{e}_i,mv)=\frac{\sum_{p\in G(\hat{e}_i,w)}\left(d_a(p,r_{p,mv}^{n-1})\right)^2}{\|G(\hat{e}_i,w)\|}\Bigg/\left(\frac{\|\mu(\hat{e}_i,mv)\|}{\|\hat{e}_i\|}\right) \tag{2}$$

   where $G(\hat{e}_i,w)$ is the set of available pixels in a region (with width $2w+1$) along $\hat{e}_i$, i.e., $G(\hat{e}_i,w)=\{p: p\in\psi(n), C_p>0, \text{ and } \exists\ p'\in\hat{e}_i \text{ s.t. } \max(|x_p-x_{p'}|,|y_p-y_{p'}|)\le w\}$. We use $w=2$ in our experiments. $\|\cdot\|$ denotes the number of elements in a set. $r_{p,mv}^{n-1}=r(p,mv,n-1)$ gives the reference pixel of p in $\psi(n-1)$ with displacement mv, and $d_a(p,r_{p,mv}^{n-1})$ is the attribute difference between p and $r_{p,mv}^{n-1}$ defined in (1). Specifically, if the structure attributes of p and $r_{p,mv}^{n-1}$ are both $\widetilde{E}$, i.e., $S_p=S_{r_{p,mv}^{n-1}}=\widetilde{E}$, $(p, r_{p,mv}^{n-1})$ is called an *E-Pixel pair* for $\hat{e}_i$, and p and $r_{p,mv}^{n-1}$ are named *target pixel* and *reference pixel* of this E-Pixel pair, respectively. $\mu(\hat{e}_i,mv)$ is a subset of $\hat{e}_i$ containing the target pixels of related E-Pixel pairs, $\mu(\hat{e}_i,mv)=\{p: p\in\hat{e}_i, S_p=\widetilde{E} \text{ and } S_{r_{p,mv}^{n-1}}=\widetilde{E}\}$. Note that in addition to the mean attribute difference between the pixels, the percentage of matched E-Pixel pairs given mv, i.e., $P_\mu=\|\mu(\hat{e}_i,mv)\|/\|\hat{e}_i\|$, is also considered in the selection of motion vector. The larger P_μ is, the smaller the MV search distortion is.

   After the motion search, the finally selected MV for $\hat{e}_i$ is the one that gives the minimum MV search distortion, i.e.,

$$mv_{\hat{e}_i}=\arg\min_{mv} d_e(\hat{e}_i,mv). \tag{3}$$

2) Search the reference edge $\bar{e}_{i'}^{n-1}$ of $\hat{e}_i$ in frame $\psi(n-1)$.
   a) With $mv_{\hat{e}_i}$, we can obtain the set of all the reference pixels of E-Pixel pairs for $\hat{e}_i$, i.e., $\bar{\mu}(\hat{e}_i,mv_{\hat{e}_i})=\{q: \exists\ p\in\hat{e}_i \text{ s.t. } q=r_{p,mv_{\hat{e}_i}}^{n-1}, S_p=\widetilde{E} \text{ and } S_q=\widetilde{E}\}$, and then select the R-Edge from $\mathbb{E}_R^{n-1}$ that overlaps the most with $\bar{\mu}(\hat{e}_i,mv_{\hat{e}_i})$. In other words, the reference edge of $\hat{e}_i$ in $\psi(n-1)$ is

$$\begin{aligned}\bar{e}_{i'}^{n-1}&=\arg\max_{\bar{e}\in\mathbb{E}_R^{n-1}}\|\bar{\mu}(\hat{e}_i,mv_{\hat{e}_i})\cap\bar{e}\|\\&=\arg\max_{\bar{e}\in\mathbb{E}_R^{n-1}}\|\{q: q\in\bar{\mu}(\hat{e}_i,mv_{\hat{e}_i}) \text{ and } q\in\bar{e}\}\|.\end{aligned} \tag{4}$$

   Note that we can search all the R-Edges in $\mathbb{E}_R^{n-1}$ to find the one that matches $\hat{e}_i$ the best, i.e., overlaps with $\bar{\mu}(\hat{e}_i,mv_{\hat{e}_i})$ the most. However, the complexity of this approach is very high as the number of R-Edges can be very large. In reality, there are only a few R-Edges in $\mathbb{E}_R^{n-1}$ having common pixels with $\bar{\mu}(\hat{e}_i,mv_{\hat{e}_i})$; therefore, we can just search among these R-Edges to find the best match.

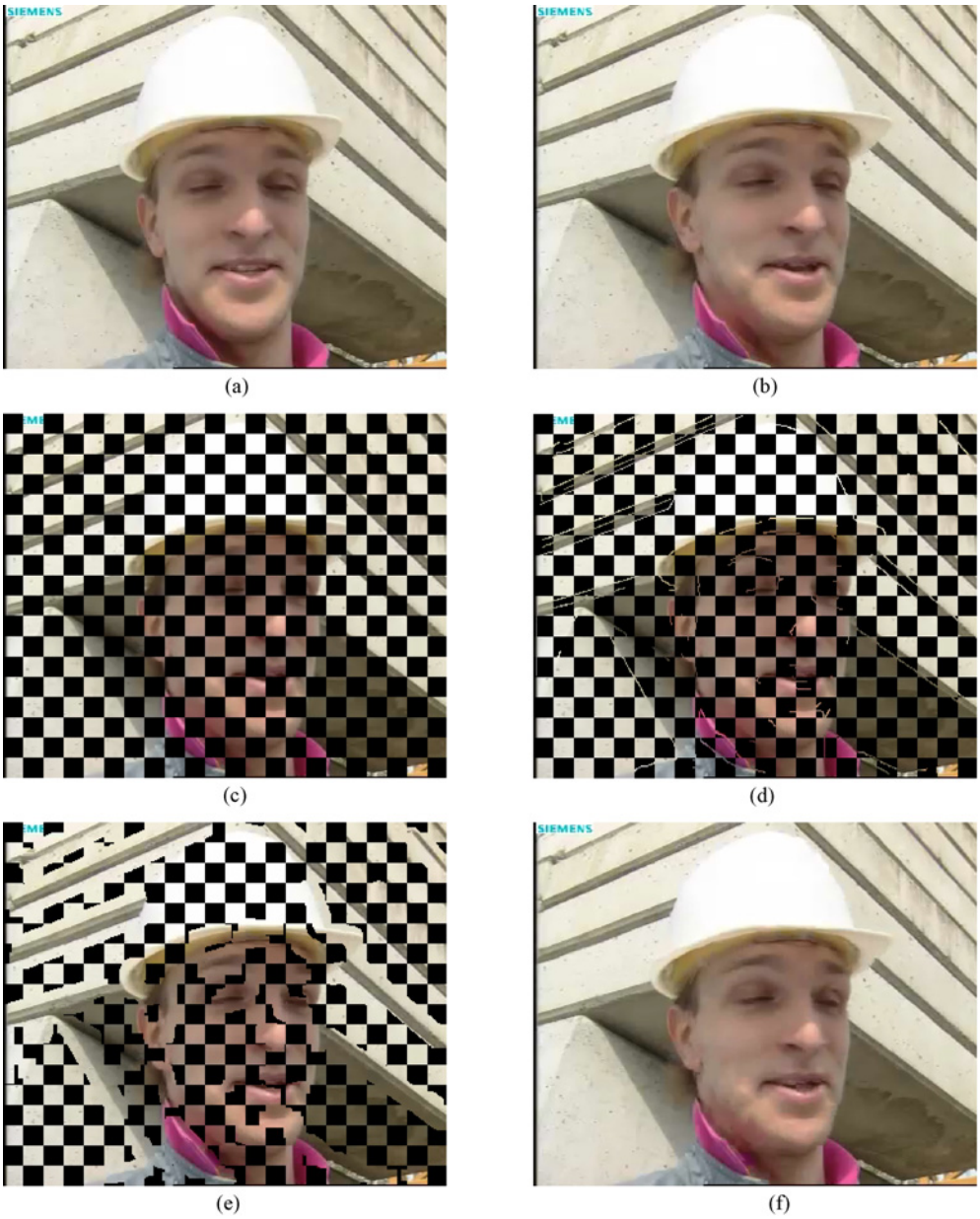


Fig. 2. Illustration for the three main steps of EDEC. (a) Reference frame $\psi(n-1)$ without error. (b) Reconstructed frame $\psi(n)$ without error. (c) Slice L_1 lost in frame $\psi(n)$. (d) Recovered edges in $\psi(n)$. (e) Structure region restoration based on (d). (f) Finally reconstructed frame $\psi(n)$ by EDEC.

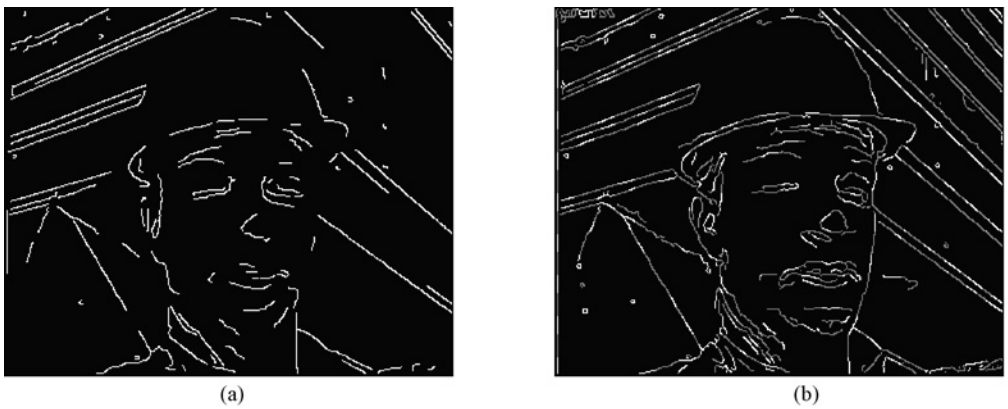


Fig. 3. Illustration for edge restoration. (a) Trustworthy R-Edges in frame $\underset{\sim}{\psi}(n-1)$ according to the P-Edges in frame $\underset{\sim}{\psi}(n)$. (b) All the P-Edges in $\psi(n)$. Pixels with white, grey, or black color denote pixels with structure attribute $\widetilde{E}$ (edge pixel), $\widetilde{PE}$ (possible edge pixel), or $\widetilde{NE}$ (not edge pixel), respectively.

b) After obtaining the reference edge of $\hat{e}_i$, i.e., $\bar{e}_{i'}^{n-1}$, in the previous step, we check whether $\bar{e}_{i'}^{n-1}$ is trustworthy or not. Specifically, we compute the percentage of matched E-Pixel pairs between $\hat{e}_i$ and $\bar{e}_{i'}^{n-1}$, i.e., $P_{\bar{\mu}} = \|\bar{\mu}(\hat{e}_i, mv_{\hat{e}_i}) \cap \bar{e}_{i'}^{n-1}\| / \|\bar{e}_{i'}^{n-1}\|$. Given a threshold T_m:

i) if $P_{\bar{\mu}} \geq T_m$, $\bar{e}_{i'}^{n-1}$ is a trustworthy reference edge for $\hat{e}_i$ and we will continue to go to step 3;

ii) otherwise, $\bar{e}_{i'}^{n-1}$ is not used and we will select the next P-Edge $\hat{e}_j$ from $\mathbb{E}_P^n$ and go to step 1.

T_m is trained to be 0.5 in our experiments.

3) Transform $\bar{e}_{i'}^{n-1}$ into $\psi(n)$ to refine the uncertain parts of $\hat{e}_i$:

a) Note that although we first estimate the motion vector $mv_{\hat{e}_i}$ for $\hat{e}_i$, and then search around $mv_{\hat{e}_i}$ to find the reference edge $\bar{e}_{i'}^{n-1}$ of $\hat{e}_i$, we do not assume that the motion from $\hat{e}_i$ to $\bar{e}_{i'}^{n-1}$ is a translational motion. Instead, we will use the matched E-Pixel pairs between $\hat{e}_i$ and $\bar{e}_{i'}^{n-1}$, i.e., $\varepsilon_i = \{(p, q) : p \in \hat{e}_i, q = r_{p,mv_{\hat{e}_i}}^{n-1}, q \in \bar{e}_{i'}^{n-1}, S_p = \widetilde{E}$ and $S_q = \widetilde{E}\}$, to calculate the transform matrix from $\bar{e}_{i'}^{n-1}$ to $\hat{e}_i$. The principle axes transform (PAT) algorithm proposed in [28] is used, with the coordinates of all the E-Pixel pairs from ε_i as input. PAT is chosen because it gives satisfactory results and requires very low time complexity, i.e., linear time to the number of E-Pixel pairs. Suppose the obtained transform matrix from $\bar{e}_{i'}^{n-1}$ to $\hat{e}_i$ is Φ_i;

b) For each pixel $q \in \bar{e}_{i'}^{n-1}$, with coordinate (x_q, y_q) in frame $\psi(n-1)$, the location of its corresponding pixel in $\psi(n)$ is calculated by $\Phi_i \times (x_q, y_q)^{\mathrm{T}}$. Then we round $\Phi_i \times (x_q, y_q)^{\mathrm{T}}$ to the integer coordinate and suppose pixel p lies here;

i) If $C_p = 0$, i.e., pixel p has not been error-concealed, we will update the attribute vector of p by setting $Y_p = Y_q$, $U_p = U_q$, $V_p = V_q$, $S_p = \widetilde{E}$, and $C_p = C_E$. Here C_E is a constant less than 1; it is trained to be 0.7 in our experiments;

ii) If $C_p = 1$, i.e., pixel p is error-free, we will set S_p to be $\widetilde{E}$ if it is not;

iii) Otherwise, we will select the next pixel q' from $\bar{e}_{i'}^{n-1}$ and go to step 3-b, as the attribute vector of p has already been updated by another edge that passes through it.

Note that in order to assist the *structure region restoration* process in Section II-C, we maintain the list of all the (p, q) pairs that we have obtained in this step, i.e., $\theta_i = \{(p, q) : q \in \bar{e}_{i'}^{n-1}$ and p lies in the rounded location of $\Phi_i \times (x_q, y_q)^{\mathrm{T}}$ in $\psi(n)\}$. Given a P-Edge $\hat{e}_i$, θ_i exists only when its reference edge $\bar{e}_{i'}^{n-1}$ exists. In other words, θ_i saves one edge pair, which includes an R-Edge $\bar{e}_{i'}^{n-1}$ and one transformed edge of $\bar{e}_{i'}^{n-1}$. And the latter one is actually one recovered edge during this *edge restoration* process. We define Θ to be the set of all the possible θ_i, i.e., all the obtained edge pairs. The *structure region restoration* process in Section II-C will be performed based on Θ.

Note that we have introduced how to match a P-Edge $\hat{e}$ in $\psi(n)$ to an R-Edge $\bar{e}$ in $\psi(n-1)$, and then transform $\bar{e}$ into $\psi(n)$ to refine the uncertain parts of $\hat{e}$. Analogously, for each R-Edge $\bar{e}'$ in $\psi(n-1)$, we can also check whether there exists a P-Edge $\hat{e}'$ in $\psi(n)$ that matches $\bar{e}'$ well. If such an $\hat{e}'$ exists, we can transform $\bar{e}'$ into $\psi(n)$ to refine $\hat{e}'$, and then add the obtained edge pair into set Θ. This complementary process can help to increase the number of recovered edges in $\psi(n)$. As it proceeds similarly as the one we have introduced previously, we will not go into the details here.

Edge restoration process consumes about 39% of the total EC time in EDEC, and this is mainly due to the motion vector estimation step (step 1 in Section II-B) and the reference edge search step (step 2 in Section II-B). The edge transform step (step 3 in Section II-B) consumes very little time. Fig. 2 shows the intermediate result after edge restoration. Suppose slice 1 in frame 1 of *Foreman* (common intermediate format (CIF), 15 frames/s, QP = 28) is lost, as shown in Fig. 2(c). The error-free reconstructed frame 0, Fig. 2(a), is used as the reference frame to error conceal frame 1. And the recovered edges are shown in Fig. 2(d) with estimated pixel values. We can see from the figure that edges in the corrupted frame can be well estimated by this edge restoration process. To give a better illustration for the R-Edges and P-Edges, we also show the initial estimated P-Edges of frame 1 in Fig. 3(b), and their trustworthy reference edges in frame 0, i.e., the extracted R-Edges from Θ, in Fig. 3(a).

C. Structure Region Restoration

In the previous *edge restoration* process, we have saved all the obtained edge pairs in Θ. For each edge pair $\theta_j \in \Theta$, we can extract the corresponding R-Edge $\bar{e}_{j'}^{n-1}$ in $\psi(n-1)$ and its transformed edge $\tilde{e}_j$ in $\psi(n)$. Here $\tilde{e}_j$ is actually a recovered edge. Define $R(\tilde{e}_j, \omega) = \{p : p \in \psi(n)$ and $\exists\ p' \in \tilde{e}_j$ s.t. $\max(|x_p - x_{p'}|, |y_p - y_{p'}|) \leq \omega\}$ to be the *structure region* along $\tilde{e}_j$, with region width parameter ω. During the *structure region restoration* process, we will refine the attribute vector of each pixel p in $R(\tilde{e}_j, \omega)$, using both temporal and spatial information, i.e., neighboring pixels surrounding $\bar{e}_{j'}^{n-1}$ and p, respectively. A patched-based filling approach will be used in our algorithm. As compared to a pixel-based one, it can not only improve the execution speed but also improve the accuracy of the propagated structures [2]. In addition, in order to reduce the occurrences of improperly propagated structures, we gradually increase the width of structure regions, and each new round of *structure region restoration* is based on the results of the previous one. In our experiments, the region width parameter ω goes from 1 to 6, as we find that a larger ω will not obviously increase the reconstructed video quality but increase the time complexity a lot.

In detail, given a region width parameter ω, for each edge pair $\theta_j \in \Theta$ with extracted $\bar{e}_{j'}^{n-1}$ in $\psi(n-1)$ and $\tilde{e}_j$ in $\psi(n)$, the *structure region restoration* process proceeds as follows.

1) Mark the initial status of each pixel in $\tilde{e}_j$. In detail, for each $p \in \tilde{e}_j$, suppose W_p is a $(2\omega+1) \times (2\omega+1)$ patch centered at p.
   a) If all the pixels in W_p are received, i.e., confidence attributes being 1, mark pixel p as *RCV* (neighbors are received).
   b) Otherwise, mark pixel p as *CRP* (neighbors are corrupted).
2) Search in $\tilde{e}_j$ to find the *CRP* pixel with the maximum priority and then refine its neighboring pixels. The priority of a pixel p is defined as the average confidence attributes of its neighboring pixels, i.e.,

$$\rho_p = \frac{\sum_{q \in W_p \cap \psi(n)} C_q}{\|W_p \cap \psi(n)\|}. \tag{5}$$

1) If such a *CRP* pixel exists and suppose it is p_k, refine each pixel in W_{p_k} as follows and mark the status of p_k as *RFN* (neighbors are refined).
   a) Search in frame $\psi(n-1)$ to find the patch that is most similar to W_{p_k}.
   Note that as $p_k \in \tilde{e}_j$ and θ_j maintains the list of all the pixel pairs for $\bar{e}_{j'}^{n-1}$ and $\tilde{e}_j$, we can obtain the corresponding pixel of p_k lying on edge $\bar{e}_{j'}^{n-1}$, i.e., $q_{k'}^{n-1}$. Define a search zone around $q_{k'}^{n-1}$ to be $Z_t(q_{k'}^{n-1}, \varpi_t) = \{q : q \in \psi(n-1)$ and $\exists q' \in \bar{e}_{j'}^{n-1}$ s.t. $\max(|x_{q'} - x_{q_{k'}^{n-1}}|, |y_{q'} - y_{q_{k'}^{n-1}}|) \le \varpi_t$ and $\max(|x_q - x_{q'}|, |y_q - y_{q'}|) \le \varpi_t\}$, where ϖ_t is the zone size parameter and is trained to be 6 in our experiments. Then the center of the most similar patch for W_{p_k}, i.e., the best match, is searched in $Z_t(q_{k'}^{n-1}, \varpi_t)$, and it is given by

$$b_k = \arg \min_{b \in Z_t(q_{k'}^{n-1}, \varpi_t)} d_p(W_{p_k}, W_b). \tag{6}$$

   Here $d_p(W_1, W_2)$ is the difference between two patches W_1 and W_2, and is simply defined as the average square YUV differences between the pixels with confidence attributes greater than zero. We use $D_k^0 = d_p(W_{p_k}, W_{b_k})$ to denote this minimum matching difference. Suppose T_p is a threshold for the early stop of patch matching. If $D_k^0 \le T_p$, use W_{b_k} to update the erroneous parts of W_{p_k}, change the confidence attributes of updated pixels to be C_S, and then go back to step 2. Here C_S is a constant and is trained to be 0.6 in our experiments.
   b) In order to obtain a better estimation for W_{p_k}, we use edge $\tilde{e}_j$ to split W_{p_k} into two halves, $W_{p_k}^1$ and $W_{p_k}^2$, and then search their best matches separately. Note that as $\tilde{e}_j$ may not be a straight edge, the shapes of $W_{p_k}^1$ and $W_{p_k}^2$ may not be rectangular. In addition to the temporal search zone $Z_t(q_{k'}^{n-1}, \varpi_t)$, we define a spatial search zone around p_k as $Z_s(p_k, \varpi_s) = \{p : p \in \psi(n)$ and $\max(|x_p - x_{p_k}|, |y_p - y_{p_k}|) \le \varpi_s\}$, where the zone size parameter ϖ_s is trained to be 8 in our experiments. Then the best matches of $W_{p_k}^1$ and $W_{p_k}^2$ will be searched separately in both $Z_t(q_{k'}^{n-1}, \varpi_t)$ and $Z_s(p_k, \varpi_s)$. Suppose these two best matches are $W_{b_1\ k}^1$ and $W_{b_2\ k}^2$, respectively, and the corresponding matching differences are D_k^1 and D_k^2, respectively.
   c) Compare the matching differences obtained in steps 2-a.i and 2-a.ii and, and then determine whether to refine W_{p_k} as a whole patch or not. In other words:
      i) if $(D_k^1 + D_k^2)/2 \le D_k^0$, use $W_{b_1\ k}^1$ and $W_{b_2\ k}^2$ to update the erroneous parts of $W_{p_k}^1$ and $W_{p_k}^2$, respectively;
      ii) otherwise, use W_{b_k} to update the erroneous parts of W_{p_k}. Change the confidence attributes of updated pixels in W_{p_k} to be C_S, and then go back to step 2.
2) Otherwise, all the pixels in $\tilde{e}_j$ have been marked as *RCV* (received) or *RFN* (refined), so the *structure region restoration* of θ_j with parameter ω has finished.

Note that although most of the edges in the corrupted frame can be well restored in Section II-B, some edges may still be improperly estimated, due to the inaccurate edge motion vectors or edge-pair transform matrices. To decrease the side effect of such false estimations, when the erroneous parts of a patch W_{p_k} are updated by its best match, e.g., W_{b_k}, both the zero-confidence pixels and those already error-concealed pixels are updated. Suppose pixel $p \in W_{p_k}$ and its corresponding pixel in W_{b_k} is b.

- If $C_p = 0$, p is error-concealed by copying the pixel value from b.
- If $0 < C_p < 1$, p is error-concealed by averaging its old pixel value with b.
- Otherwise, p is an error-free pixel and does not need to be error-concealed.

This continuous updating of erroneous pixels works like a low-pass filter and thus can effectively suppress the error brought by false edge estimation. On the other hand, as the *structure region restoration* process is directed by edge pairs, strong edges are still preserved in the reconstructed frame. The intermediate result after this step is presented in Fig. 2(e). The structure region restoration process consumes about 59% of the total time in EDEC. Although the splitting and separate-searching scheme (see item 2-a.ii in Section II-C) increases the time complexity by about 15%, it can help to increase the reconstructed video quality by about 0.3 dB.

D. Remaining Region Restoration

After recovering the edges in $\psi(n)$ and the structure regions along these edges, the next step of our algorithm is to recover the remaining erroneous regions of $\psi(n)$. Similarly as in Section II-C, we use a patch-based filling approach.

For each macroblock MB_c in $\psi(n)$, it is refined as follows if it belongs to the lost slice (L_1 in this example).

1) Search in MB_c among the pixels with zero confidence attributes and find the one with the maximum priority. Similarly as in Section II-C, the priority of pixel p is

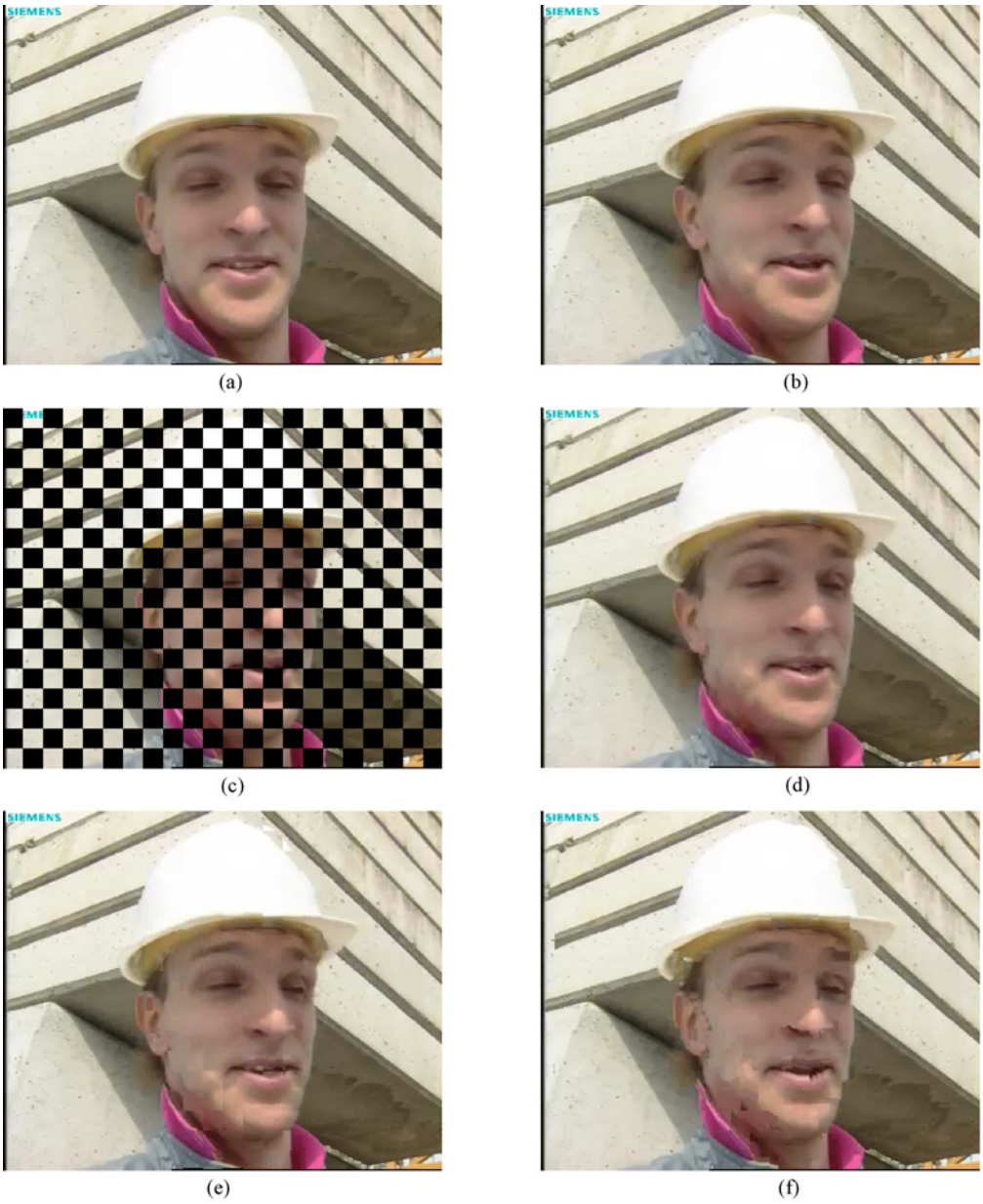


Fig. 4. Visual results of applying different error concealment algorithms on *Foreman* for one slice loss. (a) Original encoded frame 0 without error. (b) Original encoded frame 1 without error. (c) One slice lost in frame 1. (d)–(f) Reconstructed frame 1 using EDEC, BM, and EI, respectively. (d) EC by proposed EDEC, PSNR = 34.27. (e) EC by BM, PSNR = 32.04. (f) EC by EI, PSNR = 32.13.

calculated using (5), and W_p is a $(2\omega+1) \times (2\omega+1)$ patch centered at p. Parameter ω is trained to be 4 in our experiments, to make a compromise between EC performance and time complexity.

a) If such a pixel exists and suppose it is p_k, patch W_{p_k} will be refined.
Define the spatial search zone around p_k as $Z_s(p_k, \varpi) = \{p : p \in \psi(n)$ and $\max(|x_p - x_{p_k}|, |y_p - y_{p_k}|) \leq \varpi\}$, where ϖ is a zone size parameter. In addition, we also define a temporal search zone, $Z_t(q_{k'}^{n-1}, \varpi) = \{q : q \in \psi(n-1)$ and $\max(|x_q - x_{q_{k'}^{n-1}}|, |y_q - y_{q_{k'}^{n-1}}|) \leq \varpi\}$, where pixel $q_{k'}^{n-1}$ is the reference pixel of p_k in $\psi(n-1)$ and can be found using the MVs obtained in the HBM of Section II-A1. Then we will search the patch that is most similar to W_{p_k} in both $Z_s(p_k, \varpi)$ and $Z_t(q_{k'}^{n-1}, \varpi)$. Only the pixels with confidence attributes greater than zero are used to calculate the patch difference. Suppose the most similar patch is W_{b_k} centered at b_k, $b_k \in Z_s(p_k, \varpi) \cup Z_t(q_{k'}^{n-1}, \varpi)$. Then W_{b_k} will be used to update the erroneous parts of W_{p_k} using a similar approach as we have used in Section II-C. After this, we change the confidence attributes of updated pixels to be C_R. Here C_R is a constant and is trained to be 0.2 in our experiments. Then go to step 1.

b) Otherwise, all the pixels in MB_c have been refined or received. Go to step 2.

2) Continue to refine the next macroblock $MB_{c'}$ in $\psi(n)$.

After recovering the remaining erroneous regions of $\psi(n)$,

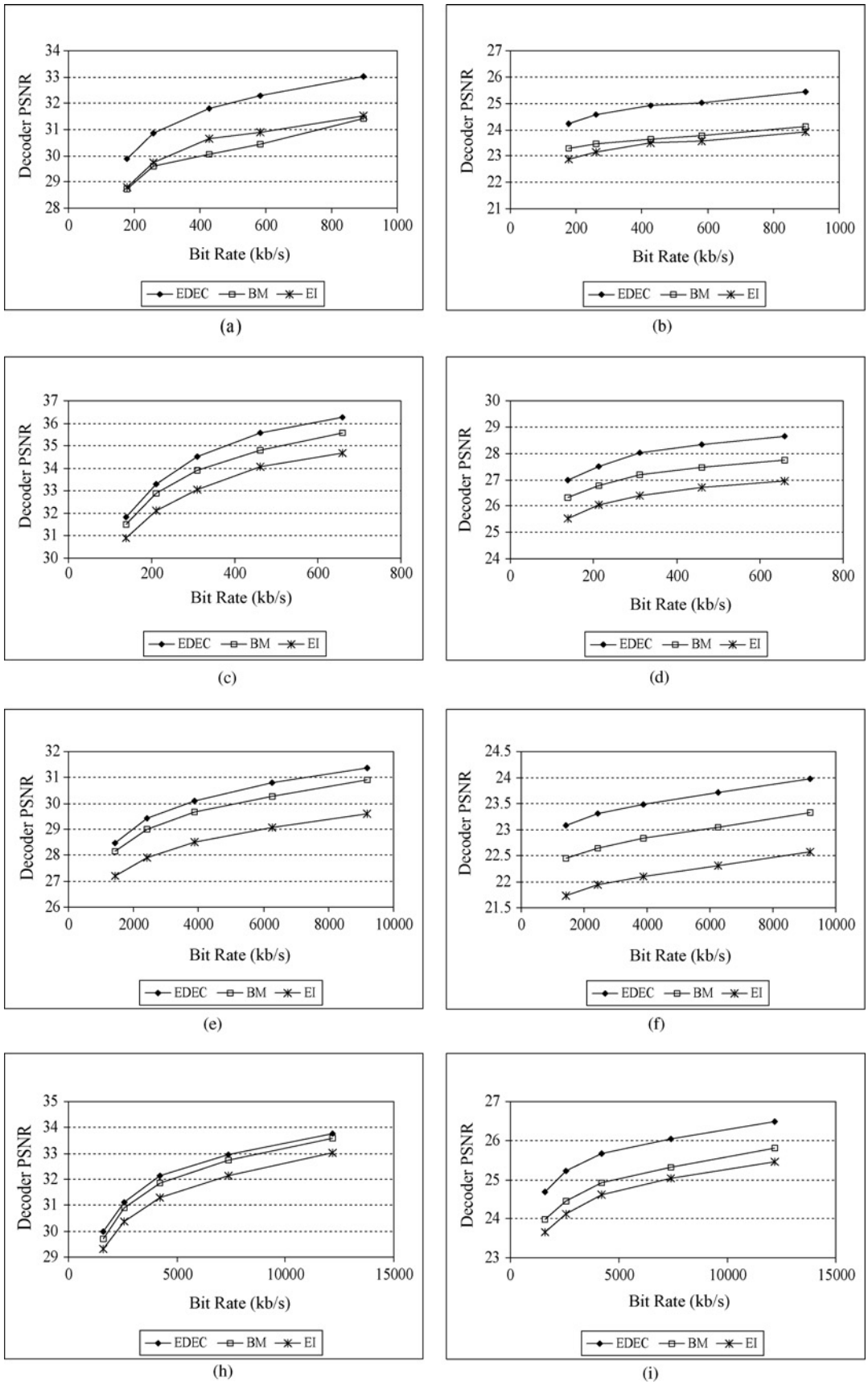


Fig. 5. Rate-distortion (RD) curves of different EC algorithms with packet loss rate $P = 3\%$ and $P = 10\%$. (a) *Foreman* CIF ($P = 3\%$). (b) *Foreman* CIF ($P = 10\%$). (c) *Sign_Irene* CIF ($P = 3\%$). (d) *Sign_Irene* CIF ($P = 10\%$). (e) *Harbour* 720p ($P = 3\%$). (f) *Harbour* 720p ($P = 10\%$). (g) *Night* 720p ($P = 3\%$). (h) *Night* 720p ($P = 10\%$).

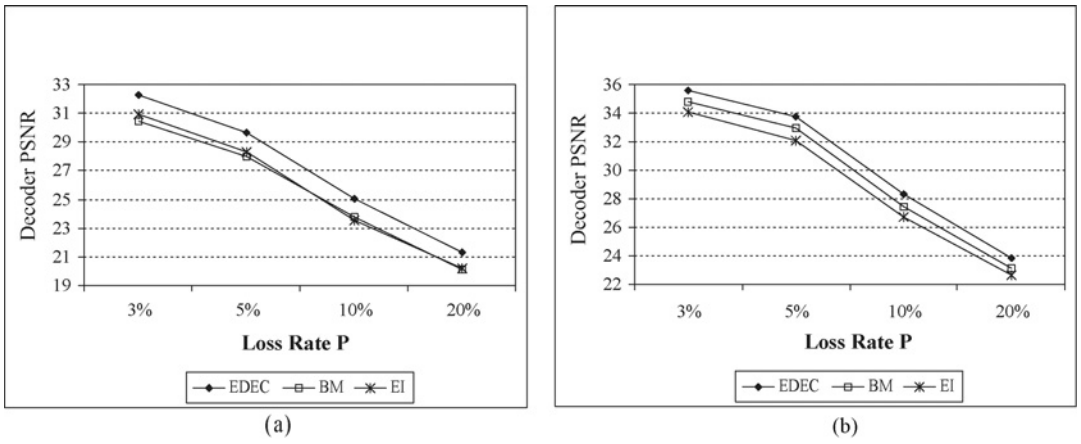


Fig. 6. Average decoder PSNRs for different packet loss rate (P = 3%, P = 5%, P = 10%, or P = 20%). QP = 25 is used to encode the sequences. (a) *Foreman*. (b) *Sign_Irene*.

we can obtain an error-concealed frame of $\psi(n)$. Suppose it is $\psi_e(n)$. Then we will average $\psi_e(n)$ with $\psi_h(n)$, which is obtained by hierarchical boundary matching in Section II-A1, to get the finally error-concealed frame of $\psi(n)$. This averaging operation can further help to reduce the error energy in the reconstructed frame. Fig. 2(f) shows the finally reconstructed frame by EDEC, which is very close to the error-free one shown in Fig. 2(b).

III. Simulation Results

In the simulation, we compare the proposed EDEC algorithm with two other algorithms. One is the boundary matching (BM) error concealment algorithm modified from [1], and the other one is the exemplar-based image inpainting (EI) approach modified from [2]. For BM, the boundary width for the motion estimation of lost MBs is set to 2. The search range is $[-16, 16]$ using the median MV of received neighboring blocks as the starting point. For EI, as a large portion of a video frame may be corrupted due to slice losses or a temporally propagated error, we improve the original algorithm in [2] by extending the source region for patch searching to both the previous frame and the received region of current frame. The patch size selected is 13×13 as we find it can propagate the structures most properly. The temporal search range is $[-32, 32]$ using zero motion as the search center, and the spatial search range is $[-4, 4]$. We use the joint video team reference software JM 12.4 (baseline profile) for the simulations [29], and 200 frames of video sequence *Foreman* (CIF), *Sign_Irene* (CIF), *Harbour* (720 p), and *Night* (720 p) are used for the testing, compressed at 15 frames/s. All the block sizes from 4×4 to 16×16 are allowed for the motion estimation/compensation and the search range is $[-32, 32]$. In the simulation, each frame is encoded into two slices, using the checkerboard FMO mode in Fig. 1. Only the first frame is encoded as an INTRA-frame, and all the subsequent ones are encoded as INTER-frames. One fixed QP is used for the whole sequence, and its value is adjusted to achieve different bit rates. *Random INTRA-MB refresh* is not used, i.e., parameter $RandomIntraMBRefresh = 0$ in the encoder configuration file. However, additional INTRA-MB can be encoded if it has a lower RD cost in the encoder mode-decision procedure.

The compressed video is supposed to be transmitted though a packet loss channel, and one packet contains the information of one slice. Note that as prediction beyond the slice boundaries is forbidden, within a same frame, error propagation from one slice to the other one can be prevented. We use [30] to generate the packet loss patterns, and the loss rate is P = 3%, 5%, 10%, or 20%. Decoder peak signal-to-noise ratio (PSNR) is used as the objective measurement, which is computed using the original uncompressed video as reference. Given a packet loss rate P, the video sequence is transmitted 20 times, and the average PSNR for the 20 transmissions is calculated at decoder.

Fig. 4 illustrates the visual quality after applying different error concealment algorithms on *Foreman* for one slice loss. QP = 28 is used to encode the whole sequence, and suppose slice 1 in frame 1 is lost, i.e., as in Fig. 4(c). For the sake of comparison, the original error-free reconstructed frames are also shown in Fig. 4(a) and (b). Then frame 0 is used to error-conceal the lost slice in frame 1 using EDEC, BM, and EI, and the reconstructed frames are shown in Fig. 4(d), (e), and (f), respectively, with corresponding decoder PSNR. We can observe from the figure that the error-concealed frame using EDEC looks much better than that using BM or EI, i.e., both the blocky artifacts and the artificial inpainting effects are greatly reduced. In addition, edges in the corrupted frame are more precisely estimated (e.g., the right boundary of the hat) and pixels in smooth regions are more reasonably filled (e.g., the region between the left neckline and the face). With EDEC, the PSNR gain can be more than 2 dB. In addition, as the width of structure regions is gradually increased during EC process, and the lost regions are recovered using both spatial and temporal neighboring pixels, the EDEC-reconstructed video can be smoothly displayed at normal frame rate. In other words, the flickering artifacts introduced by discontinuous temporal backgrounds can be avoided in EDEC. Interested readers can download the video sequences in Fig. 4 at [31].

Fig. 5 shows the RD curves of different EC algorithms with packet loss rates P = 3% and P = 10%. From

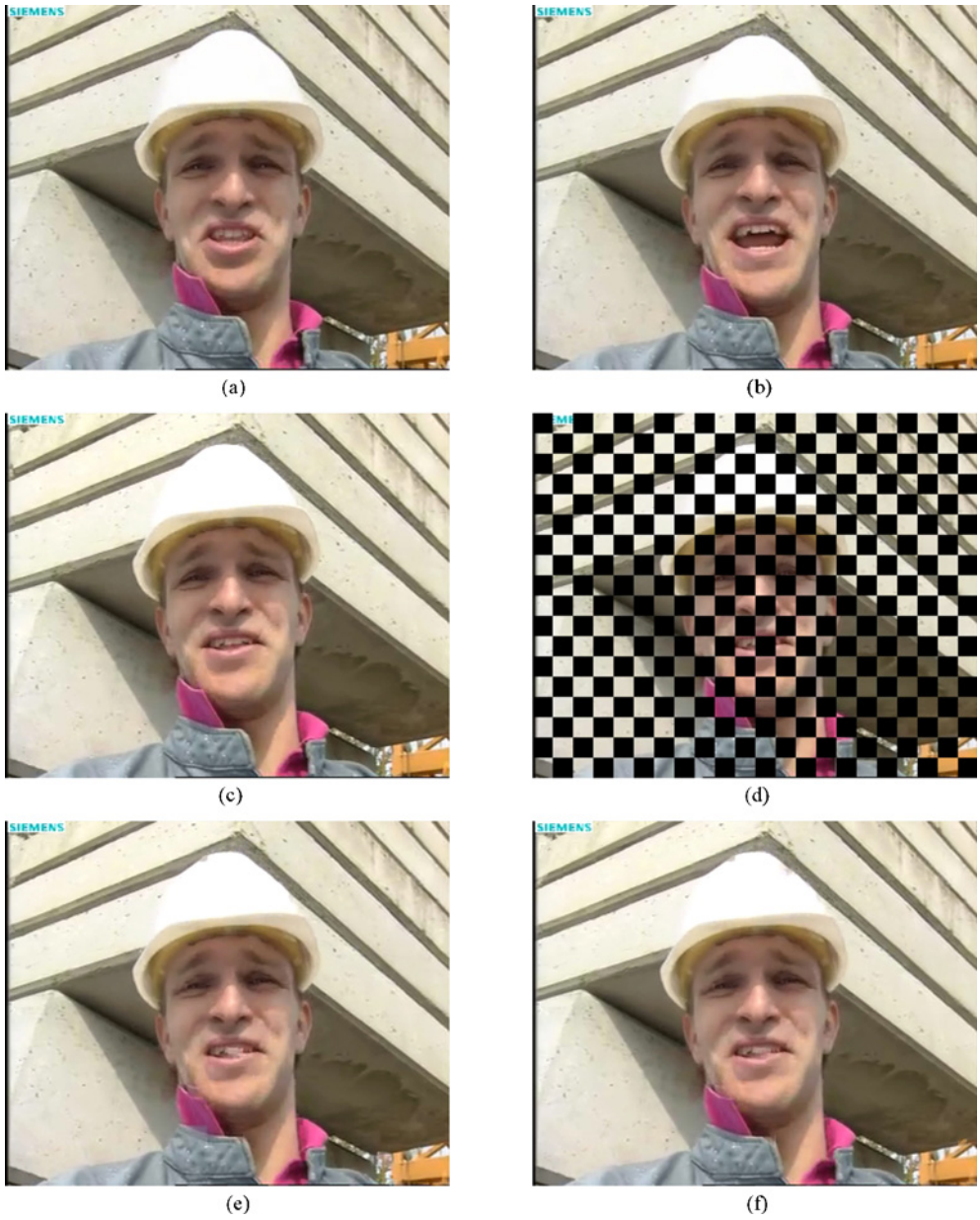


Fig. 7. Comparison between EDEC with one and with multiple reference frames. (a) Frame 27 without error. (b) Frame 28 without error. (c) Frame 29 without error. (d) One slice lost in frame 29. (e) Reconstructed frame 29 using frame 28 as reference; EC with one reference, PSNR = 32.46. (f) Reconstructed frame 29 using both frame 27 and frame 28 as references; EC with two references, PSNR = 32.89.

the figure we can see that the RD curves of EDEC are much higher than those of BM and SI, i.e., 0.68 dB and 1.22 dB on average for $P = 3\%$, and 0.84 dB and 1.35 dB for $P = 10\%$, respectively. Specifically, for a sequence with strong edges such as *Foreman*, EDEC performs especially well. For example, when the loss rate P is 3%, the PSNR gain of EDEC over BM can be as high as 1.84 dB. Fig. 6 compares the three EC algorithms for different packet loss rates, $P = 3\%$, 5%, 10%, or 20%. QP = 25 is used to encode the sequences. Similar to the results in Fig. 5, the curves of EDEC are always higher than those of BM and SI.

Note that in the previous simulations, only one reference frame, i.e., the immediately previous one, is used to error-conceal a corrupted frame. In order to recover more edges in the erroneous frame and obtain more trustworthy filling patches, we can extend the EDEC algorithm in Section II to utilize multiple previous frames as references. In other words, for a corrupted area, if we cannot find a suitable match in the previous frame, we can search further to find a better one. Fig. 7 shows the comparison between EDEC reconstructed frames using one and two reference frames. Suppose one slice is lost in frame 29 of *Foreman* (QP = 28). Its previous two reference frames and its error-free reconstructed frame are also shown for the sake of comparison. By comparing the results in Fig. 7(e) and (f) we can see that using multiple reference frames can indeed help to increase the reconstructed video quality. For example, when the teeth in the corrupted area are recovered, if only frame 28 is used as the reference, a right match for the teeth cannot be found as the *Foreman's* mouth is open widely at this time. On the other hand, in frame 27, as the

TABLE I
AVERAGE ERROR CONCEALMENT TIME FOR A CORRUPTED FRAME

	Compared to Decoding			Compared to BM		
Sequence	EDEC	BM	EI	EDEC	BM	EI
Foreman (CIF)	1288	97	1324	13.28	1.00	13.65
Sign_Irene (CIF)	1321	136	1786	9.71	1.00	13.13
Harbour (720p)	1043	99	1374	10.54	1.00	13.88
Night (720p)	624	80	1177	7.80	1.00	14.71

teeth are bared, we can find a suitable match to error-conceal the lost teeth in frame 29. By using one additional reference frame, a 0.43 dB gain can be obtained. However, since this additional reference will nearly double the computation time, we propose to use just one reference frame in reality as it can already provide a satisfactory result.

Finally, we compare the complexities of the three algorithms. For BM, the time complexity is determined by the boundary width and search range in the motion estimation of lost MBs. As the median MV of received neighboring blocks is used as the search center for a lost MB, we need to maintain the MVs of all the received blocks. Compared to BM, a confidence map (floating point, the same size as the video frame) needs to be maintained for EI. And its time complexity is determined by the patch size, search range used for region filling, and the contour length of the remaining unfilled regions. For EDEC, the time complexity is related to many factors, such as the maximum edge length, the number of edges in the corrupted frame, the structure region width, the patch size to fill the remaining regions, and the search ranges used in all the three main steps. During the EC process, we need to maintain the MVs of all the 4×4 blocks, the reconstructed frame $\psi_h(n)$ after HBM (Section II-A1), the attribute vectors and the obtained edge pairs (Section II-B). The YUV components of attribute vectors can be just saved in the original decoder buffer without occupying extra space. In the simulation, we use the average decoding time of an error-free frame as a time unit (around 11 ms for a CIF video frame and 131 ms for a 720 p video frame, on an Intel Core4 2.66 GHz central processing unit), and then normalize the average error concealment time of each algorithm in the previous simulations. The results are presented in Table I (columns under *Compared to Decoding*). The ratios of EDEC and EI to BM are also presented (columns under *Compared to BM*) for a clearer illustration. We can observe that the average EC time of EDEC for a corrupted frame is about 10.3 times that of BM. This is mainly due to the edge restoration and structure region restoration processes of EDEC, consuming about 39% and 59% of the total time, respectively. Although these processes increase the EC time considerably, they can precisely estimate most of the corrupted edges so as to reduce the artifacts brought by BM, and can help to increase the reconstructed video quality by as much as 2.23 dB (Fig. 4). In addition, it is worthy to note that there is still a large space to reduce the time complexity of EDEC by using techniques such as dynamic programming. For example, as edges can be broken into edge segments, their information (such as edge MV and best matched edge-pair) can be reused several times during the edge restoration process.

IV. CONCLUSION

In this paper, we have proposed an EDEC algorithm to recover the lost regions in a video frame. It consists of three main steps. First, the strong edges in the corrupted frame are estimated based on the edges in the neighboring frames and the received area of current frame. Second, the lost regions along these estimated edges are recovered using both spatial and temporal neighboring pixels. Finally, the remaining parts of the lost regions are estimated. Simulation results show that compared to the existing boundary matching algorithm and the exemplar-based inpainting approach, the proposed EDEC algorithm can reconstruct the corrupted frame with both a better visual quality and a higher decoder PSNR.

REFERENCES

[1] W. M. Lam, A. R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Mar. 1993, pp. 417–420.

[2] A. Criminisi, P. Prez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[3] Y. Wang, S. Wenger, J. Wen, and A. K. Katsaggelos, "Review of error resilient coding techniques for real-time video communications," *IEEE Signal Process. Mag.*, vol. 17, no. 4, pp. 61–82, Jul. 2000.

[4] Y. Wang and Q. F. Zhu, "Error control and concealment for video communication: A review," *Proc. IEEE*, vol. 86, no. 5, pp. 974–997, May 1998.

[5] *Scattered Slices: A New Error Resilience Tool for H.26L*, JVT-B027.doc, document of Joint Video Team of ISO/IEC MPEG and ITU-T Video Coding Experts Group, Feb. 2002.

[6] S. Kumar, L. Xu, M. K. Mandal, and S. Panchanathan, "Error resiliency schemes in H.264/AVC standard," *J. Vis. Commun. Image Represent.*, vol. 17, no. 2, pp. 425–450, Apr. 2006.

[7] S. Valente, C. Dufour, F. Groliere, and D. Snook, "An efficient error concealment implementation for MPEG-4 videostreams," *IEEE Trans. Consumer Electron.*, vol. 47, no. 3, pp. 568–578, Aug. 2001.

[8] Y. Wang, A. Reibman, and S. Lin, "Multiple description coding for video delivery," *Proc. IEEE*, vol. 93, no. 1, pp. 57–70, Jan. 2005.

[9] C. S. Kim and S. U. Lee, "Multiple description coding of motion fields for robust video transmission," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 5, pp. 999–1010, Sep. 2001.

[10] A. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri, "Multiple description coding for video using motion compensated prediction," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 1999, pp. 837–841.

[11] V. Vaishampayan, "Application of multiple description codes to image and video transmission over lossy networks," in *Proc. Int. Packet Video Workshop*, Apr. 1991, pp. 2857–2860.

[12] A. Tom, C. Yeh, and F. Chu, "Packet video for cell loss protection using deinterleaving and scrambling," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Apr. 1991, pp. 2857–2860.

[13] S.-C. Hsia, S.-C. Cheng, and S.-W. Chou, "Efficient adaptive error concealment technique for video decoding system," *IEEE Trans. Multimedia*, vol. 7, no. 5, pp. 860–868, Oct. 2005.

[14] C.-Y. Su, S.-H. Tsay, and C.-H. Huang, "Error concealment using direction-oriented candidate set and predicted boundary matching criteria," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2006, pp. 2221–2224.

[15] Y. Chen, X. Sun, F. Wu, Z. Liu, and S. Li, "Spatio-temporal video error concealment using priority-ranked region-matching," in *Proc. IEEE Int. Conf. Image Process.*, vol. 2. Sep. 2005, pp. 1050–1053.

[16] L. Chen, S. Chan, and H. Shum, "A joint motion-image inpainting method for error concealment in video coding," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2006, pp. 2241–2244.

[17] W. Zeng and B. Liu, "Geometric-structure-based error concealment with novel applications in block-based low-bit-rate coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 4, pp. 648–665, Jun. 1999.

[18] C.-C. Wei-Ying Kung, C.-S. Kim, and J. Kuo, "A spatial-domain error concealment method with edge recovery and selective directional interpolation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Apr. 2003, pp. 700–703.

[19] F. Soares and L. D. Pereira, "Spatial shape error concealment for object-based image and video coding," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 586–599, Apr. 2004.
[20] Y.-L. Huang and H.-Y. Lien, "An edge-based temporal error concealment for MPEG-coded video," in *Proc. Soc. Photo-Optic. Instrum. Eng. Vis. Commun. Image Process. (VCIP)*, Jul. 2005, pp. 992–1000.
[21] S. Shirani, F. Kossentini, and R. Ward, "A concealment method for video communications in an error-prone environment," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 1122–1128, Jun. 2000.
[22] X. Li and M. T. Orchard, "Novel sequential error-concealment techniques using orientation adaptive interpolation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 10, pp. 857–864, Oct. 2002.
[23] D. Liu, X. Sun, F. Wu, S. Li, and Y.-Q. Zhang, "Image compression with edge-based inpainting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1273–1287, Oct. 2007.
[24] I. E. G. Richardson, "H.264/MPEG4 Part 10," in *H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia*. Chichester, U.K.: Wiley, 2003, ch. 6, pp. 159–224.
[25] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, Nov. 1986.
[26] N. Farber, K. Stuhlmuller, and B. Girod, "Analysis of error propagation in hybrid video coding with application to error resilience," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 1999, pp. 550–554.
[27] B. Girod and N. Farber, "Wireless video," in *Compressed Video Over Networks*, M.-T. Sun and A. R. Reibman, Eds. New York: Marcel Dekker, 2000, ch. 12, pp. 465–512.
[28] D. Kennedy, N. M. Alpert, J. F. Bradshaw, and J. A. Correia, "The principal axes transformation: A method for image registration," *J. Nucl. Med.*, vol. 31, pp. 1717–1722, Oct. 1990.
[29] *Joint Model, The JVT Reference Software for H.264/AVC* [Online]. Available: http://iphome.hhi.de/suehring/tml/download
[30] *Error Patterns for Internet Experiments*, Q15-I-16r1.doc, document of ITU-T Video Coding Experts Group, Oct. 1999.
[31] *The Reconstructed YUV Sequences for Different EC Algorithms* [Online]. Available: http://www.cse.ust.hk/~myma/EDEC/EDEC.html

Mengyao Ma received the B.S. degree in computer science and technology from Peking University, Beijing, China, in 2003, and the Ph.D. degree in computer science and engineering from the Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, in 2009.

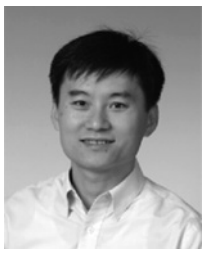
She is currently with the Hong Kong Applied Science and Technology Research Institute Company Ltd., Shatin, Hong Kong, as a Senior Engineer. Her research interests include error resilient video compression, error propagation analyses, and error concealment of video streams over packet loss channels.

Dr. Ma was a recipient of the Best Paper Award from the IEEE International Workshop on Signal Processing Systems in 2007 and also from the Pacific-Rim Conference on Multimedia in 2007.

Oscar C. Au (S'87–M'90–SM'01) received the B.A.Sc. degree from the University of Toronto, Toronto, Ontario, Canada, in 1986, and the M.A. and Ph.D. degrees from Princeton University, Princeton, NJ, in 1988 and 1991, respectively, all in electrical engineering.

After being a Postdoctoral Researcher in Princeton University for one year, he joined the Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology (HKUST), Clear Water Bay, Kowloon, Hong Kong, as an Assistant Professor, in 1992. He is/has been an Associate Professor with the Department of Electronic and Computer Engineering, Director of the Multimedia Technology Research Center, and Director of the Computer Engineering Program at HKUST. He has published about 260 technical journals and conference papers. His fast motion estimation algorithms were accepted into the ISO/IEC 14496-7 MPEG-4 International Video Coding Standard and the China AVS-M Standard. His light-weight encryption and error resilience algorithms are accepted into the AVS standard. He has three U.S. patents and is applying for more than 50 on his signal processing techniques. He has performed forensic investigations and has stood as an Expert Witness in the Hong Kong courts many times. His main research interests include video and image coding and processing, watermarking and light weight encryption, and speech and audio processing. His research topics include fast motion estimation for MPEG-1/2/4, H.261/3/4 and AVS, optimal and fast sub-optimal rate control, mode decision, transcoding, denoising, deinterlacing, post-processing, multiview coding, scalable video coding, distributed video coding, subpixel rendering, JPEG/JPEG2000, HDR imaging, compressive sensing, halftone image data hiding, GPU-processing, software-hardware co-design, etc.

Dr. Au has been an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS Part I, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and the IEEE TRANSACTIONS ON IMAGE PROCESSING. He is on the Editorial Boards of *Journal of Signal Processing Systems*, *Journal of Multimedia*, and *Journal of Franklin Institute*. He has been the Chairman of the Circuits and Systems (CAS) Technical Committee (TC) on Multimedia Systems and Applications, and a Member of the CAS TC on Video Signal Processing and Communications and the CAS TC on Digital Signal Processing, Signal Processing (SP) TC on Multimedia Signal Processing and SP TC on Image, Video and Multidimensional Signal Processing. He served on the Steering Committee of the IEEE TRANSACTIONS ON MULTIMEDIA and the IEEE International Conference on Multimedia and Expo. He also served on the Organizing Committee of the IEEE International Symposium on Circuits and Systems in 1997, the IEEE International Conference on Acoustics, Speech, and Signal Processing in 2003, the ISO/IEC MPEG 71st Meeting in 2004, the International Conference on Image Processing in 2010, and other conferences. He has been General Chair of Pacific-Rim Conference on Multimedia in 2007, and will be chairing the IEEE International Conference on Multimedia and Expo, and the Packet Video Workshop in 2010. He won best paper awards in the IEEE Workshop on Signal Processing Systems in 2007 and Pro Cycling Manager in 2007.



S.-H. Gary Chan (S'89–M'98–SM'03) received the B.S.E. degree (Highest Honors) in electrical engineering from Princeton University, Princeton, NJ, in 1993, with certificates in applied and computational mathematics, engineering physics, and engineering and management systems. He also received the M.S.E. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1994 and 1999, respectively, with a minor in business administration.

He is currently an Associate Professor with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, and an Adjunct Researcher with Microsoft Research Asia, Beijing, China. He was a Visiting Assistant Professor in Networking with the Department of Computer Science, University of California, Davis, from 1998 to 1999. From 1992 to 1993, he was a Research Intern with the Nippon Electric Company Research Institute, Inc., Princeton, NJ. His research interests include multimedia networking, peer-to-peer technologies and streaming, and wireless communication networks.

Dr. Chan was a William and Leila Fellow with Stanford University from 1993 to 1994. At Princeton University, he was the recipient of the Charles Ira Young Memorial Tablet and Medal, and the Photonic and Opto-Electronic Materials Newport Award of Excellence in 1993. He served as a Vice-Chair of the IEEE Communications Society Multimedia Communications Technical Committee from 2003 to 2006. He was a Guest Editor for the IEEE COMMUNICATION MAGAZINE (Special Issues on Peer-to-Peer Multimedia Streaming), in 2007, and Springer *Multimedia Tools and Applications* (Special Issue on Advances in Consumer Communications and Networking), in 2007. He was the Co-Chair for the Workshop on Advances in Peer-to-Peer Multimedia Streaming for the Association for Computing Machinery Multimedia Conference in 2005, the Multimedia Symposia for the IEEE Global Telecommunications in 2006, and the IEEE International Conference on Communications in 2005 and 2007.

Ming-Ting Sun (S'79–M'81–SM'89–F'96) received the B.S. degree from National Taiwan University, Taipei, Taiwan, in 1976, the M.S. degree from the University of Texas, Arlington, in 1981, and the Ph.D. degree from the University of California, Los Angeles, in 1985, all in electrical engineering.

He joined the University of Washington, Seattle, in August 1996, where he is currently a Professor with the Department of Electrical Engineering. Previously, he was the Director of the Video Signal Processing Research Group, Bellcore, Piscataway, NJ. He was a Chaired Professor with TsingHwa University, Beijing, China, and a Visiting Professor with Tokyo University, Tokyo, Japan, and National Taiwan University, Taipei, Taiwan. He holds 11 patents and has published over 200 technical papers, including 13 book chapters in the area of video and multimedia technologies. He has also coedited a book, *Compressed Video over Networks* (Marcel Dekker, 2000). His research interests include multimedia signal processing and networking.

Dr. Sun was the Editor-in-Chief of the IEEE TRANSACTIONS ON MULTIMEDIA and the Distinguished Lecturer of the IEEE CIRCUITS AND SYSTEMS SOCIETY (CAS-S) from 2000 to 2001. He was the recipient of the IEEE CAS-S Golden Jubilee Medal in 2000, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (TCVST) Best Paper Award in 1993, and the Award of Excellence from Bellcore for his work on the digital subscriber line in 1987. He was the General Co-Chair of the 2000 Visual Communications and Image Processing Conference. He was the Editor-in-Chief of the TCVST from 1995 to 1997. From 1988 to 1991, he was the Chairman of the IEEE Circuits and Systems Standards Committee and established the IEEE Inverse Discrete Cosine Transform Standard.