# LOSS CONCEALMENT BASED ON VIDEO INPAINTING FOR ROBUST VIDEO COMMUNICATION

*Mounira Ebdelli*[*], *Olivier Le Meur*[**], *Christine Guillemot*[*]

INRIA [*], University of Rennes 1[**]
Campus Universitaire de Beaulieu
35042 Rennes Cedex France

## ABSTRACT

This paper presents an error concealment algorithm combining the advantages of motion-compensated spatio-temporal interpolation and examplar-based video inpainting. The algorithm first estimates the motion information of erroneous/lost blocks in a video sequence using a state-of-the-art method called BMFI (Bilinear Motion Field Interpolation). Then, the video inpainting algorithm estimate the texture of each lost blocks as a linear combination of the most similar blocks in a motion-compensated window. Experiments on several videos show more accurate and visually pleasing results. In terms of PSNR (Peak of Signal-to-Noise Ratio), the average gain is about 2dB compared to state-of-art methods.

***Index Terms***— concealment, examplar-based, motion-interpolation.

## 1. INTRODUCTION

Error/loss concealment methods aim at reconstructing the lost parts of a received video using the high temporal and spatial correlation of video sequences [1]. It is usually performed at the decoder. Then, new error concealment algorithms can be incorporated as standard-compatible enhancements to conventional decoders. A review of classical methods can be found in [2]. In general, a concealment method requires first error or loss detection. The impairments can be detected at the transport protocol level in the case of losses (lost packets - erasure channels), via the detection of missing transport packets using the packet sequence numbers. In the case of bit errors, the impairments can be detected at the VLC (Variable Length Coding) decoding level. In the pixel domain, after the decoding process, the difference between adjacent blocks can be computed. When a difference exceeds a certain threshold, a transmission error has been detected.

To handle detected lost information, classical error concealment methods make use of some form of spatial, temporal or spatio-temporal interpolation. The spatial interpolation consists in estimating the missing pixels by smoothly interpolating surrounding pixels. While the temporal interpolation consists in repeating co-located pixels in previously correctly decoded frames. This method is only efficient on the static parts of images. In presence of motion, spatial and temporal interpolation can be combined [3] in a motion-compensated temporal interpolation to provide better concealment. Missing blocks are estimated by motion-compensating blocks from previous frames. If the motion vector (MV) is also lost, one has to estimate the MV first, typically by copying the MV of the block above or by interpolating the MV of previously decoded frames. This method does not take into account the texture similarity between blocks. Therefore errors in motion estimation may lead to severe artefacts. Examplar-based inpainting techniques which are used to fill-in missing regions (holes) can be considered to address the loss concealment problem by jointly using motion and texture information. Indeed, these techniques select for a given corrupted patch the best matching patches belonging to the uncorrupted parts of the images [4–7]. This search can be efficiently performed by using the texture similarities as well as the motion information.

In this paper, we use a modified version of the video inpainting algorithm proposed in [8, 9] to fill-in missing blocks of a received video. This algorithm uses motion information of each undamaged pixel in the video in order to determine whether a pixel $p$ belongs to a moving object ($M_c(p) = 1$) or not ($M_c(p) = 0$). It then proceeds by first inpainting the moving objects. Each corrupted patch is filled in with a linear combination of the most similar patches in the neighboring images. Motion information $M_c(p)$ of each filled pixel is then updated with the $M_c$ values of the copied pixel values. This strategy of updating the $M_c$ values is not sufficiently robust since any error in the motion information may lead to propagate the moving foreground information in the stationary background. To deal with this problem, we propose here to estimate the motion information of each corrupted block before inpainting the texture. We also use this motion estimated information to limit the search space for the best matching patches in a motion-compensated window. Finally, a new priority scheme is proposed to first inpaint moving objects in frames having less corrupted moving pixels.

The rest of the paper is organized as follows. State-of-the-art

methods of error concealment using spatio-temporal interpolation are described in Section 2. The proposed algorithm of error concealment is discussed in Section 3. Performances are illustrated and commented in Section 4. Section 5 concludes the paper.

## 2. STATE-OF-THE-ART METHODS : ERROR CONCEALMENT BASED ON SPATIO-TEMPORAL INTERPOLATION

In this section, we only focus on state-of-the-art methods based on saptio-temporal interpolation. Typically, motion-compensated spatio-temporal interpolation methods utilize the smoothness across the boundary of the lost regions to perform error concealment. Motion vectors (MV) are first estimated using MV of the neighboring blocks. The lost block is then recovered by performing the motion compensation of the block from the previous image. Lam and Reibman [10] proposed a boundary matching algorithm (BMA) to estimate the missing motion vectors. For each MV of the neighboring blocks, motion compensation is used. The side matched distortion (SMD) across boundaries of the block to be filled in and the compensated one is computed. The candidate MV that minimizes SMD is selected as the MV of the damaged block. The SMD measure usually used is the average of absolute (or squared) differences.

This method can fail if the motion within the corrupted block is not purely translational or if the actual motion vector is not close to any of the neighboring vectors. Bilinear motion field interpolation (BMFI) may produce more accurate motion estimation for different types of motion [1]. This method estimates the motion vector of each pixel $p(x, y)$ in the lost or corrupted block $B_c$ using its coordinates in the block and a bilinear interpolation of MV of the left, right, top and bottom (respectively $V_L$,$V_R$,$V_T$ and $V_B$) neighboring blocks. The motion vector $V(x, y)$ of the pixel $p(x, y)$ is computed using the following formula: $V(x, y) = \frac{1}{2}((1 - x_n)V_L + x_nV_R + (1 - y_n)V_T + y_nV_B)$, where $x_n = \frac{x-x_L}{x_R-x_L}$ and $y_n = \frac{y-y_T}{y_B-y_T}$. $x_L$ and $x_R$ are respectively the x-coordinates of the left and right borders of $B_c$ and $y_T$ and $y_B$ are respectively the y-coordinates of the top and bottom borders.

In the rest of the paper, we consider the BMFI method to estimate lost motion vectors of corrupted blocks. As mentioned above a simple compensation of the corrupted block using estimated motion vectors may lead to severe visually artefacts because of the propagation of motion estimation errors. The BMFI method is then used to provide estimated motion vectors which are then used by the examplar-based video inpainting algorithm. This one uses both texture similarities and motion information to select the best matching patches in the correctly received parts of the video sequence. Corrupted parts of moving objects in each image is filled-in before inpainting the static background. Figure 1 gives a synthetic flow chart of the proposed approach. This two-steps algorithm leads to better results than a simple motion-compensation. The next section describes in more details the video inpainting algorithm that we use for error concealment.

## 3. ERROR CONCEALMENT BASED ON VIDEO INPAINTING TECHNIQUE

### 3.1. VIDEO INPAINTING ALGORITHM

As proposed in [8], the horizontal ($V_x$) and vertical ($V_y$) components of the motion vectors are thresholded to determine whether the pixel $p$ belongs to the moving foreground object ($M_c(p) = 1$) or not ($M_c(p) = 0$). The proposed algorithm consists of three steps: (i) filling-in the moving foreground; (ii) filling-in the stationary background; (iii) filling-in the remaining holes with a spatial image inpainting technique. These steps are described in next sections.

#### *3.1.1. Inpainting the moving foreground object*

The algorithm starts by filling-in the moving objects in each image $I_t$ using the following steps.

1. Filling order: for each pixel $p$ of the fill front $\delta\Omega$, compute the confidence and data terms of the patch $\Psi_p$ centered at $p$. The confidence term $C(p)$ is defined as in [6]: $C(p) = \frac{\sum_{q\in\Psi_p\cap(I-\Omega)} C(q)}{|\Psi_p|}$. This is the ratio between the number of known pixels with respect to the total number of pixels in the patch to be filled-in. While the data term $D(p)$ is defined as: $D(p) = \frac{|\nabla M_c^{\perp}\cdot n_p|}{\alpha}$. Where $n_p$ is a unit vector orthogonal to the fill front $\delta\Omega$ centered at $p$ and $\alpha$ is a normalizing constant ($\alpha = 255$). The term $D(p)$ aims at giving more priority to patches for which motion direction is perpendicular to the fill front. Then, the priority of filling-in of each patch $\Psi_p$ centered at $p$ is given by: $P(p) = C(p)D(p)$.

2. Texture synthesis: once the priority has been computed, we select the highest priority patch $\Psi_{\hat{p}}$ to be inpainted ($\hat{p} = \arg\max_{p\in\delta\Omega} P(p)$). For this patch, we seek the most similar patches $\Psi_q$ using the known region of $\Psi_{\hat{p}}$ in a motion compensated search window in the previous and next images. The similarity between the current patch $\Psi_{\hat{p}}$ and the candidate $\Psi_q$ is computed using the sum of squared differences (SSD) between the corresponding 5 components vectors (R,G,B, $V_x$,$V_y$).
   As proposed in [9], the most similar patch is either copied using template matching (TM), or $K$ most similar patches linearly combined using local linear embedding (LLE) and non-negative matrix factorization (NMF) methods. The best estimate (note $\Psi_q$) obtained by one of these three methods that minimize the distance with the corrupted block belonging to the known pixels is selected. The unknown moving pixel values of

$\Psi_{\hat{p}}$ are updated by their co-located pixel values of $\Psi_q$. For each inpainted pixel in $\Psi_{\hat{p}}$, the confidence term is updated by computing the new ratio of known versus unknown pixels in $\Psi_{\hat{p}}$.

The above steps are repeated until all unknown pixels of the moving objects are filled-in.

#### 3.1.2. *Inpainting the stationary background*

This second step consists in filling-in the stationary background. The missing blocks are filled-in by copying available information from the known parts of the co-located blocks in the neighboring images. Here, the confidence term is defined as $C(p) = 0$ if the pixel $p$ is either in the moving foreground or a damaged pixel and $C(p) = 1$ otherwise. The data term $D(p)$ is defined as :

$$D(p) = \frac{\sum_{p \in \delta\Omega, t=-\delta n \ldots \delta n} M_t(p)}{\beta}$$

where $M_t(p) = 0$ if the pixel $p$ is either damaged or moving and $M_t(p) = 0$ otherwise. The time index $t$ is the relative position of the neighboring image to the current inpainted image. $n$ is the number of previous and next neighboring images considered and $\beta$ is a normalized factor equal to $(2n + 1)$. In this step, the data term $D(p)$ measures the amount of information that can be copied from neighboring images. These two terms are used to compute the priority of all corrupted patches in the video sequence to be filled-in as : $P(p) = C(p)D(p)$. The highest priority of $P(p)$ indicates both the patch $\Psi_{\hat{p}}$ and the image $I_{\hat{f}}$ to be first filled in i.e.

$$\left\{\hat{p}, \hat{f}\right\} = \underset{p \in \delta\Omega_f, f=1 \ldots N}{\arg\max} P(p)$$

where $\delta\Omega_f$ is the fill front of the missing region in the image $f$ and $N$ is total number of images in the video sequence. $\Psi_{\hat{p}}$ is inpainted using information of the patch centered at $p$ in the neighboring images having the highest confidence term. The algorithm iterates until no more temporal information is available to fill in the remaining patches ($D(p) = 0$). The last step consists in filling-in the remaining holes using a spatial inpainting algorithm (as described in [6]) on each image.

### 3.2. VIDEO INPAINTING FOR ERROR CONCEALMENT

The previous sections describe how texture of missing areas is retrieved. This algorithm requires motion vectors (MV) of each pixel to compute the similarities between patches. In addition, MV play an important role to avoid the propagation of foreground information in the background region. They are also used to adapt the motion-compensated search window. However, in an error concealment context, if a block is lost, its MV can also be lost. Then, to estimate a lost block using the video inpainting algorithm, we first estimate the MV of each damaged block using the BMFI method as described in Section 2. Figure 1 describes the different steps of the proposed error concealment algorithm, namely a pre-processing step followed by the proposed video inpainting method.

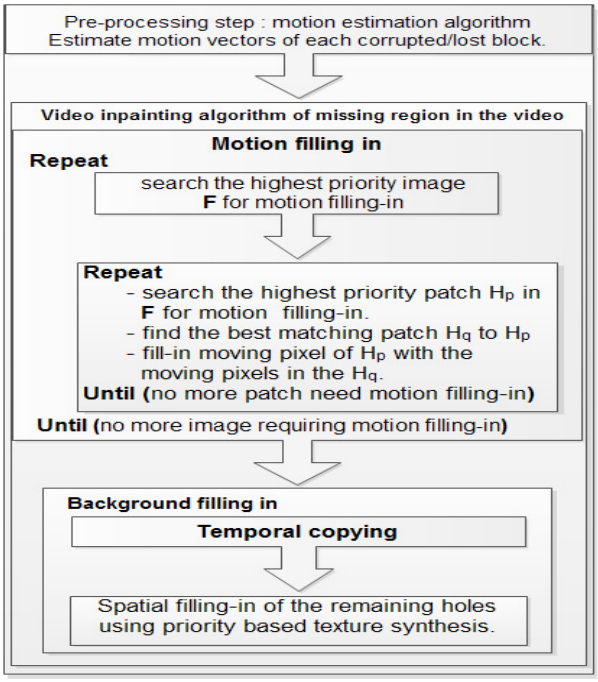



**Fig. 1**. Overview of the proposed error concealment algorithm.

## 4. RESULTS

The proposed approach is compared to state-of-the-art methods [2, 3, 6]. Fig. 4 illustrates three images results of different methods. The first row shows the original images while the second row represents the corresponding corrupted images where 20% of blocks of $16 \times 16$ pixels are lost. Results of concealment simply using a motion-compensation method [2], or using only spatial inpainting algorithm described in [6] present a lot of artefacts. One can observe, in the last column, that these artefacts do not appear in images obtained using the proposed approach. Moving object as well as static background are correctly recovered. Fig. 3 shows that even in the case of a video sequence with 50% of lost blocks, the proposed technique performs better than the motion compensation method. Table 1 summarizes the PSNR values obtained using each of the listed methods for the two test sequences with different percentage of lost blocks. We can notice that the proposed approach presents an average gain of about 2dB compared to state-of-the-art methods. A comparison with the spatio-temporal selective extrapolation method presented in [3] is also shown in Fig. 2. Our proposed method show more natural looking results.

**Fig. 2**. Original image (first column); corrupted image (second column), PSNR=13.2 dB ; image recovered using the algorithm in [3] (third column), PSNR=20.59 dB ; image recovered using the proposed algorithm (last column), PSNR=20.65 dB.

**Fig. 3**. Image with 50% of blocks are lost (first column), PSNR=9.01 dB; image recovered using motion compensation (second column), PSNR=28.26 dB; image recovered using the proposed algorithm (third column), PSNR=32.23 dB.

| Fig. | Percentage of lost blocks | Motion compensation | Spatial inpainting | Proposed method |
|---|---|---|---|---|
| 3 | 20 | 31.47 | 23.45 | 34.17 |
| 3 | 50 | 27.27 | 19.37 | 29.27 |
| 4 | 20 | 30.74 | 23.84 | 33.39 |
| 4 | 50 | 27.13 | 19.90 | 28.84 |

**Table 1**. PSNR values of concealed videos presented in figures 3 and 4.

## 5. CONCLUSION

This paper describes a new error concealment algorithm based on examplar-based video inpainting avided by motion interpolation. Once the motion information of the lost blocks in the video sequence is estimated, the missing texture is approximated using the video inpainting algorithm. The results show improved performances compared to using a simple motion compensation of blocks from previous frames. This method limits error propagation, caused by the uncertainties on the estimated motion information, especially in the first step of moving object inpainting. Some artefacts still appear in the concealed video due to the errors in the estimation of the motion vectors of lost blocks. The erroneous motion vectors lead to consider some lost moving pixels as background pixels and vice versa. Future work will be dedicated to more robust method for motion vectors estimation of corrupted blocks.

## ACKNOWLEDGMENT

This work is supported by the ANR-09-VERS-019-02 grant (AR-SSO project) and by the Inria - Alcatel Lucent Bell Labs joint laboratory.

## 6. REFERENCES

[1] M. E. Al-Mulla, N. Canagarajah, and D. R. Bull, "Error concealment using motion field interpolation," 1998, pp. 512–516.

[2] Y. Wang and Q. Zhu, "Error control and concealment for video communication: A review," in *Proceedings of the IEEE*, May. 1998, vol. 86(5).

[3] K. Meisinger and A. Kaupp, "Spatio-temporal selective extrapolation for 3-d signals and its applications in video communication," in *IEEE Trans. On Image Processing*, Sept. 2007, vol. 16(9), pp. 2348–2360.

[4] M. Bertalmio, G. Sapiro, C. Ballester, and V. Caselles, "Image inpainting," in *In Proceedings of SIGGRAPH*, July 2000, pp. 417–424.

[5] R. Bornard, E. Lecan, L. Laborelli, and J. Chenot, "Missing data correction in still images and image sequences," in *In ACM International Conference on Multimedia*, Dec.2002.

[6] A. Criminisi, P. Perez, and K. Toyama, "Object removal by examplar-based image inpainting," in *In Proc. of International Conference on Computer Vision and pattern Recognition*. CVPR, Dec. 2003, pp. 721–728.

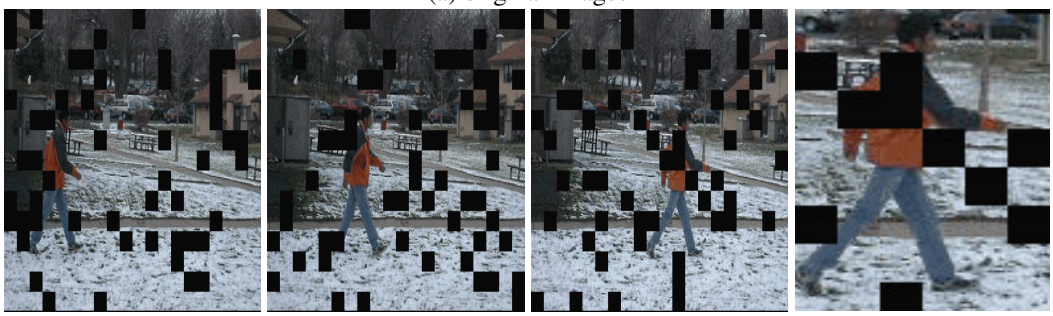
[7] J. Sun, L. Yuan, J. Jia, and H. Shum, "Image completion with structure propagation," in *ACM Transactions on Graphics*, vol. 24(3).

[8] K. A. Patwardhan, G. Sapiro, and M. Bertalmio, "Video inpainting of occluding and occluded objects," in *Proc. of the IEEE Intl. Conf. on Image Processing, ICIP*, 2005, vol. 2, pp. 69–72.

[9] M. Ebdelli, C. Guillemot, and O. Le Meur, "Examplar-based video inpainting with motion-compensated neighbor embedding," in *Proc. of the IEEE Intl. Conf. on Image Processing, ICIP*, 2012.

[10] W.M. Lam, A.R. Reibman, and B. Liu, "Recovery of lost or erroneously received motion vectors," in *Proceedings of ICASSP*, 1993, p. 417420.

(a) original images



(b) corrupted images, PSNR=14.22 dB

(c) Recovered images with motion-compensation, PSNR=31.23 dB

(d) Images inpainted with the TM-based algorithm of [6], PSNR=23.41 dB

(e) Images inpainted with video inpainting [8, 9] and a motion estimation step, PSNR=34.96 dB

**Fig. 4**. Comparison of different methods for recovering corrupted video with 20% of loss.