Accepted in IEEE Int. Conference on Image Processing (ICIP 2013), 2013

# A MAXIMUM LIKELIHOOD APPROACH TO CORRECTING TRANSMISSION ERRORS FOR JOINT SOURCE-CHANNEL DECODING OF H.264 CODED VIDEO

*François Caron and Stéphane Coulombe*

École de technologie supérieure, Université du Québec
Department of Software and IT Engineering
1100 Notre-Dame St. West, Montreal (Quebec), H3C 1K3, Canada

## ABSTRACT

Real time video applications need to handle transmission errors, as retransmissions are impractical. In this paper, we present a novel joint source channel decoding approach for video error correction. Soft-output information is combined with our syntax element-level maximum likelihood framework to effectively extract valid macroblocks from corrupted H.264 slices. Simulation results show that our video error correction strategy provides better visual quality by reducing the size of the concealment region. Our observations indicate an average PSNR improvement of 0.8 dB (QP=36), with peaks above 10 dB, over state-of-the-art error concealment at a bit error rate of $10^{-3}$.

***Index Terms—*** video error correction, joint source-channel decoding, maximum likelihood decoding, H.264, real-time video applications

## 1. INTRODUCTION

Error robustness and video error concealment are the de facto tools used to address transmission errors, especially in traditional real time video communication systems where corrupted packets are discarded.

A compromise between error robustness and coding efficiency is made on the encoding side, in an attempt to minimize both the occurrence and the impact of packet loss. For instance, smaller packets and Forward Error Correction (FEC) can be used to reduce the ratio of discarded packets, while H.264 [1] tools, such as Flexible Macroblock Ordering, can be used to surround missing macroblocks (MB) with additional neighbors. A careful compromise between error robustness and coding efficiency, as well as an adequate choice of FEC, is a difficult problem, mainly because channel conditions in wireless environments vary over time.

Video error concealment is performed at the decoder level as a last line of defense to handle missing MBs. The fact that it doesn't require additional bandwidth or reduce coding efficiency explains why the literature is rich with such solutions. However, its performance is greatly reduced when lost regions have low spatiotemporal correlation with neighboring regions that have been correctly decoded.

Video error correction can help alleviate these problems, as video communication systems belong to a class of applications that would benefit from using damaged packets [2] - [3]. Ideally, correcting these packets would lead to perfect reconstruction, but the goal is better described as minimizing the region where video error concealment is applied. However, the use of variable length codewords (VLC) makes this a challenging problem to tackle. This is because transmission errors tend to propagate, since they can alter both the length and the value of a VLC, modifying how the rest of the stream is interpreted.

Sabeva[4] and Nguyen[5] focussed on correcting CABAC encoded bits. Both solutions rely on channel information to generate and sort candidate slices, and use the video decoder as a validation mechanism. They select the first of the candidate slices that has been decoded without error, or the one for which the error was detected closest to the end of the slice. Wang[6] tackles a very specific scenario, where Data Partitioning (H.264 Extended Profile) is used to transmit motion vectors without prediction and residual information. By exploiting the spatial and temporal correlations between adjacent motion vectors, he builds a 1-D Markov process mimicking the behavior of a convolutional decoder, where the likeliest path is selected. Farrugia[7] combines list decoding and pixel-domain validation to select the best candidate slice. The solution as a whole is computationally complex, as multiple slices need to be reconstructed to select the final solution.

In this paper, we introduce a novel joint source-channel decoding (JSCD) approach to video error correction. Our solution combines the use of channel information and source semantics to select the sequence of likeliest codewords. The outline of the paper is as follows. Section 2 describes our syntax element-level error correction framework. Soft-output information is then combined into the framework in section 3. Section 4 presents an early termination strategy for the correction process. Experimental results are presented in section 5,

Financial support was provided by the Natural Sciences and Engineering Research Council of Canada (Ph.D. scholarship) and the Fonds de recherche du Québéc - Nature et technologies (grant #141698).

followed by concluding remarks in section 6.

## 2. MAXIMUM LIKELIHOOD SYNTAX-ELEMENT-LEVEL ERROR CORRECTION

Video error correction can be approached as an optimization problem. Assuming that fragmentation units [8] aren't used, each H.264 slice is sent in a distinct IP packet. Given that we know, from a failed checksum test, that a received slice $\tilde{S}$ is corrupted, we wish to find the likeliest valid slice $S^*$ that was transmitted. So, we have:

$$S^* = \arg\max_{\hat{S} \in H} \{P(\hat{S}|\tilde{S})\} \tag{1}$$

where $\hat{S}$ is a hypothetically transmitted slice, $H$ is the set of all possible transmitted slices having the same length as $\tilde{S}$, and $P(\hat{S}|\tilde{S})$ is the likelihood of transmitting $\hat{S}$, given the received slice $\tilde{S}$.

In our previous work [9], we showed that slices can be viewed as a sequence of transmitted codewords $\hat{c}_i$, such that $\hat{S} = \{\hat{c}_0, \hat{c}_1, \hat{c}_2, \ldots, \hat{c}_N\}$. Through mathematical derivations, the slice-level maximization problem (1) was decomposed into a series of syntax element-level maximization problems [9], as follows:

$$c_i^* = \arg\max_{\hat{c}_i \in C_i} \{P(\tilde{S}|\hat{c}_i) \cdot \frac{1}{2}^{\beta_i - L_B(\hat{c}_i)} \cdot P(\hat{c}_i| \cap_{k=1}^{i-1} c_k^*)\} \tag{2}$$

where $c_i^*$ is the value retained out of all the candidate symbols $\hat{c}_i$ in the codebook $C_i$, $L_B(\hat{c}_i)$ gives the binary length of the coded representation, and $\beta_i$ is the binary length of the longest coded representation in $C_i$. The decoder knows how to populate the codebook $C_i$ for each syntax element, as video standards describe the syntax an encoder has to conform to.

$P(\tilde{S}|\hat{c}_i)$ in (2) expresses the conversion cost to modify the received bits to match the coded representation of the hypothetical codeword $\hat{c}_i$. We compute this using the Hamming distance:

$$P(\tilde{S}|\hat{c}_i) = \rho^{d_i} \times (1-\rho)^{L_B(\hat{c}_i) - d_i} \tag{3}$$

where $d_i$ represents the number of differing bits between the received bits and the coded representation of the proposed codeword $\hat{c}_i$, and $\rho$ is the estimated bit error rate.

$\frac{1}{2}^{\beta - L_B(\hat{c}_i)}$ in (2) comes from the so-called random tail assumption, where the video encoder is considered to be a good binary source (i.e. $P(0) = P(1) = \frac{1}{2}$). $P(\hat{c}_i| \cap_{k=1}^{i-1} c_k^*)$ in (2) expresses the probability that the codeword $\hat{c}_i$ is used, given all the previously decoded codewords $c_k^*$. Each syntax element uses a different probability model.

## 3. JOINT SOURCE CHANNEL MAXIMUM LIKELIHOOD DECODING

In (3), and also in [9], it is assumed that the cost of flipping any bit in the corrupted packet is the same. However, JSCD provides the means to express individual flipping costs, as the log-likelihood ratios (LLRs) are shared by the channel decoder. Here, we introduce $T_n$ and $R_n$, two random binary variables representing a transmitted and a received bit at position $n$ in the bitstream. An LLR expresses the natural logarithm of the probability that a 1 was sent over the probability that a 0 was sent, given the same noise:

$$LLR_n = \log\left(\frac{P(T_n = 1|y)}{P(T_n = 0|y)}\right) \tag{4}$$

where $y$ represents a random noise signal present in the communication channel.

To better express the fact that we have a different confidence level for each received bit, we rewrite $P(\tilde{S}|\hat{c}_i)$ in (2) to replace (3):

$$P(\tilde{S}|\hat{c}_i) = \prod_{n=1}^{L_B(\hat{c}_i)} P(R_n = \tilde{b}_n | T_n = \hat{b}_n) \tag{5}$$

where $\tilde{b}_n$ is the $n^{th}$ bit from the initial position of $\hat{c}_i$, and $\hat{b}_n$ is the $n^{th}$ bit in the coded representation of the codeword $\hat{c}_i$. Using its Kolmogorov definition, we can expand (5) to:

$$P(\tilde{S}|\hat{c}_i) = \prod_{n=1}^{L_B(\hat{c}_i)} \frac{P(R_n = \tilde{b}_n \cap T_n = \hat{b}_n)}{P(T_n = \hat{b}_n)} \tag{6}$$

$P(R_n = \tilde{b}_n \cap T_n = \hat{b}_n)$ expresses the probability of receiving bit $\tilde{b}_n$, while having transmitted bit $\hat{b}_n$. There are four possible scenarios, two of which contain transmission errors. Assuming that 0s and 1s are equally subject to transmission errors, we have:

$$P(R = \tilde{b}_n \cap T = \hat{b}_n) = \begin{cases} \frac{\rho}{2}, & \text{if } \tilde{b}_n \neq \hat{b}_n \\ \frac{1-\rho}{2}, & \text{if } \tilde{b}_n = \hat{b}_n \end{cases} \tag{7}$$

From (4), the actual probabilities can be retrieved using:

$$P(T_n = \hat{b}_n|y) = \begin{cases} \frac{1}{\exp(LLR_n)+1}, & \text{if } \hat{b}_n = 0 \\ \frac{\exp(LLR_n)}{\exp(LLR_n)+1}, & \text{if } \hat{b}_n = 1 \end{cases} \tag{8}$$

The random noise signal in (4) and (8) is an actual constraint in our solution. Our goal isn't to find the signal that best explains the information received, but rather to find the actual message sent alongside the noisy signal. Therefore, let us assume that:

$$P(T = \hat{b}_n) \approx P(T = \hat{b}_n|y) \tag{9}$$

The conversion cost (6) is computed using (7) and (8).

## 4. CORRECTION PROCESS EARLY TERMINATION STRATEGY

In this work, we have modeled the probability distributions for the syntax elements used to communicate the prediction information (e.g. mb_type, mb_skip_run, mvd_l0, etc.). These distributions are added to the ones modeled in [9] for the syntax elements used in the slice header (e.g. first_mb_in_slice, slice_type, frame_number, etc.). The syntax elements used to describe the CAVLC residual information (e.g. coeff_token, total_zeros, run_before, etc.) haven't been modeled yet, but invalid values are detected while decoding.

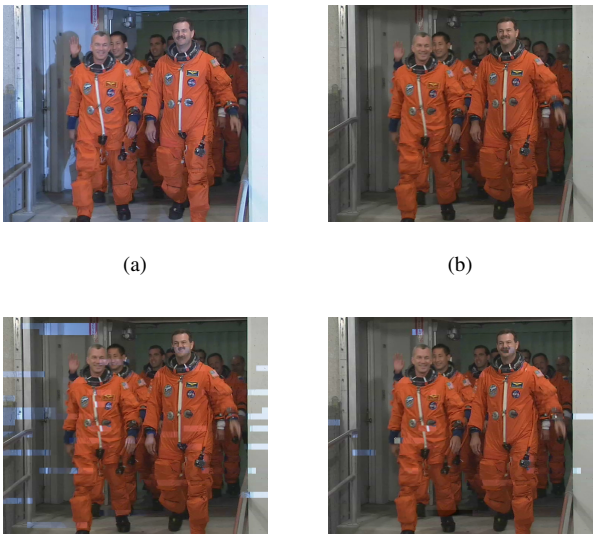
Errors in the residual information leading to a syntactically valid sequence of codewords are problematic, since the prediction information is interlaced with the residual information. Upon desynchronization, the correction process will force the use of a valid codeword in any modeled syntax element, no matter how unlikely. To address this problem, we use a threshold to stop the correction process when the selected codeword seems too unlikely. Exploiting the fact that most codewords are short, it becomes highly improbable that multiple bits inside the same codeword are erroneous, even at very high bit error rates. Thus, we propose to stop the correction process when the Hamming distance between the likeliest codeword and the received bits is greater than 1.

Furthermore, our implementation uses a greedy approach to find the series of likeliest codewords where a single codeword is retained at each step, and the remaining outcomes are discarded. Type I errors (i.e. changing intact bits) and type II errors (i.e. keeping corrupted bits) may desynchronize the bitstream, leading to an unlikely path requiring that multiple bits be flipped. To avoid following such a path, we stop the correction process when the ratio of flipped bits over interpreted bits exceeds $10^{-2}$. Once the decoding process stops, either because the slice was entirely decoded, an error was detected, or a threshold was reached, the extracted macroblocks are reconstructed and the missing macroblocks, if any, are marked for concealment.

## 5. EXPERIMENTAL RESULTS

To evaluate the gains from introducing soft-output information in our framework, we coded the 4CIF sequences *city*, *crew*, *harbour*, *ice* and *soccer*, the 480P sequences *driving*, *opening ceremony* and *whale show*, and the 720P sequences *shields* and *stockholm* using the JM18.2 [10] with four different parameter settings, and all using the Baseline profile.

We applied a combination of fixed QPs (24 and 36) and a maximum packet size (200 and 300 bytes) to vary the average number of macroblocks carried in a slice, as well as the amount of residual information transmitted. We chose this approach as a way to study the importance of modeling the residual information (i.e. CAVLC syntax elements). The MBs were coded following the raster scan order, and no additional H.264 error robustness tools were used. The coded sequences were submitted to a network simulator. The channel coder used QAM to encode 16-bit symbols using Gray codes without any additional FEC. The packets from one randomly selected picture were then submitted to an AWGN channel to simulate a bit error rate of $10^{-3}$. We used 20 different random error patterns for each coded sequence, targeting 5% of the slices in the pictures. The corrupted sequences were then decoded using three different approaches: 1) discarding all corrupted packets and applying state-of-the-art error concealment on the missing macroblocks (STBMA+PDE) [11], as, to our knowledge, this is the best and most cited method; 2) correcting the corrupted packets using the proposed maxi-



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

**Fig. 1**: Visual comparison of the $61^{th}$ frame of the *crew* sequence coded with QP = 24 and packet size $\leq$ 200 bytes; a) $60^{th}$ frame in the original sequence; b) $61^{th}$ frame in the original sequence; c) $61^{th}$ frame, where corrupted packets are discarded and 229 MBs are concealed using STBMA+PDE; d) $61^{th}$ frame, where 198 MBs were extracted using SO-MLD and 31 MBs are concealed using STBMA+PDE

(a) (b)

**Fig. 2**: Visual comparison of the top left corner of the $76^{th}$ frame in the *ice* sequence(QP=24); a) 69 MBs extracted with HO-MLD and 88 concealed MBs using STBMA+PDE; b) 27 Mbs extracted with SO-MLD and 130 concealed MBs using STBMA+PDE

**Table 1**: PSNR gains (dB) of HO-MLD and SO-MLD over STBMA+PDE (QP=24)

| Sequence | Size (bytes) | STBMA +PDE | HO-MLD Avg | HO-MLD Peak | SO-MLD Avg | SO-MLD Peak |
|---|---|---|---|---|---|---|
| city | 200 | 36.36 | +0.78 | +1.19 | +0.82 | +1.45 |
| | 300 | 36.23 | +0.69 | +2.52 | +0.83 | +2.30 |
| crew | 200 | 37.71 | +0.14 | +3.50 | +0.30 | +3.57 |
| | 300 | 37.57 | +0.27 | +2.55 | +0.50 | +3.15 |
| harbour | 200 | 36.99 | -0.40 | +0.57 | -0.10 | +0.72 |
| | 300 | 36.34 | -0.13 | +0.62 | -0.08 | +0.60 |
| ice | 200 | 39.50 | +0.55 | +2.95 | +0.69 | +2.95 |
| | 300 | 39.70 | +0.22 | +1.43 | +0.26 | +1.43 |
| soccer | 200 | 36.28 | -0.02 | +2.44 | +0.04 | +2.44 |
| | 300 | 35.12 | +0.12 | +1.60 | +0.53 | +2.18 |
| driving | 200 | 35.51 | +0.46 | +1.04 | +0.57 | +1.12 |
| | 300 | 35.26 | +0.44 | +3.24 | +0.76 | +3.24 |
| opening-ceremony | 200 | 36.93 | +0.12 | +0.42 | +0.13 | +0.47 |
| | 300 | 36.92 | +0.17 | +0.70 | +0.18 | +0.70 |
| whale-show | 200 | 35.17 | +0.78 | +2.03 | +0.90 | +2.31 |
| | 300 | 35.25 | +1.06 | +2.59 | +1.15 | +2.67 |
| shields | 200 | 38.00 | -0.80 | +0.53 | -0.77 | +0.63 |
| | 300 | 36.80 | -0.54 | +2.96 | -0.44 | +2.96 |
| stockholm | 200 | 37.45 | +0.19 | +0.81 | +0.09 | +0.48 |
| | 300 | 35.51 | +0.82 | +1.80 | +0.85 | +1.78 |

**Table 3**: PSNR gains (dB) of HO-MLD and SO-MLD over STBMA+PDE (QP=36)

| Sequence | Size (bytes) | STBMA +PDE | HO-MLD Avg | HO-MLD Peak | SO-MLD Avg | SO-MLD Peak |
|---|---|---|---|---|---|---|
| city | 200 | 29.81 | +0.49 | +8.31 | +0.55 | +8.33 |
| | 300 | 29.46 | +1.39 | +6.64 | +1.58 | +7.94 |
| crew | 200 | 32.75 | +0.06 | +5.40 | +0.14 | +6.64 |
| | 300 | 32.33 | +0.71 | +8.67 | +0.93 | +9.74 |
| harbour | 200 | 30.17 | +0.48 | +6.02 | +0.77 | +6.02 |
| | 300 | 30.19 | -0.32 | +13.99 | +0.50 | +10.11 |
| ice | 200 | 31.96 | +0.27 | +1.83 | +0.32 | +1.83 |
| | 300 | 30.81 | +0.07 | +2.69 | +0.04 | +2.96 |
| soccer | 200 | 29.55 | +0.87 | +3.14 | +0.90 | +2.80 |
| | 300 | 29.13 | +0.45 | +3.06 | +0.62 | +3.06 |
| driving | 200 | 30.05 | -0.33 | +5.00 | +0.17 | +5.00 |
| | 300 | 29.85 | +0.23 | +12.15 | +1.03 | +12.15 |
| opening-ceremony | 200 | 33.64 | +1.77 | +9.42 | +1.80 | +9.42 |
| | 300 | 25.32 | +1.30 | +7.05 | +1.47 | +7.05 |
| whale-show | 200 | 28.14 | +1.53 | +10.36 | +1.76 | +9.11 |
| | 300 | 28.09 | +2.56 | +8.24 | +2.79 | +8.02 |
| shields | 200 | 29.81 | +0.02 | +1.44 | +0.18 | +1.37 |
| | 300 | 27.66 | +0.03 | +2.08 | +0.20 | +1.09 |
| stockholm | 200 | 31.16 | -0.21 | +0.09 | -0.18 | +0.09 |
| | 300 | 24.80 | -0.01 | +0.07 | +0.10 | +0.27 |

mum likelihood framework, but without soft-output information (HO-MLD); and 3) correcting the corrupted packets using soft-output information (SO-MLD). The remaining missing MBs in 2) and 3) were concealed using STBMA+PDE.

The PSNR of the pictures containing corrupted slices were compared. Our observations on the test scenarios described indicate an overall PSNR improvement of 0.8 dB when using SO-MLD over STBMA+PDE with QP=36. Fig. 1 shows an excellent example of where minimizing the concealment region is beneficial, as the concealment method has trouble dealing with the photographic flash between frames (a) and (b) of Fig. 1. Tables 1 and 3 presents the average PSNR gains of HO-MLD and SO-MLD over STBMA+PDE with QPs 24 and 36 respectively. The performance of our approach decreases with lower QP values as the quantity of residual information increases. The highest peaks observed in the "harbour" and "driving" sequences (Table 3) are associated with two cases where the corrupted packets were perfectly corrected. They are a clear indication that modeling the CAVLC syntax elements would yield better results.

A comparison of HO-MLD and SO-MLD is presented in Tables 2 and 4. The results indicate that SO-MLD yields better PSNR values. However, a system without access to soft-information can still use our approach as HO-MLD also performs better than STBMA+PDE. Moreover, the fact that both approaches recover as many MBs with QP 24 yet provide a lower average PSNR gain compared to QP 36 is a clear indication that modeling the CAVLC syntax element is important. When errors in the residual information go undetected, the resulting images are visually less appealing than those where only error concealment was applied. In fact, the quality of the recovered MBs prevails over the quantity. An example is shown in Fig. 2 (near the cones and the skater's thighs).

**Table 2**: Average percentage of saved MBs in a corrupted slice and the average PSNR gain of SO-MLD over HO-MLD (QP=24)

| Sequence | Lost MBs | HO-MLD | SO-MLD | $\Delta$ PSNR |
|---|---|---|---|---|
| city | 3116 | 83% | 86% | 0.14 dB |
| crew | 3242 | 70% | 77% | 0.20 dB |
| harbour | 3075 | 81% | 84% | 0.26 dB |
| ice | 2526 | 10% | 15% | 0.09 dB |
| soccer | 3020 | 77% | 79% | 0.23 dB |
| driving | 2708 | 78% | 83% | 0.21 dB |
| ceremony | 2413 | 50% | 65% | 0.01 dB |
| whale show | 2711 | 84% | 87% | 0.10 dB |
| shields | 6984 | 69% | 74% | 0.06 dB |
| stockholm | 6762 | 69% | 77% | -0.03 dB |

## 6. CONCLUSION

We have shown that combining soft-input information with our maximum likelihood framework that exploits source semantics, in JSCD fashion, produces better visual quality. Our observations indicate that SO-MLD leads to an average increase of 0.8 dB (QP=36), with peaks above 10 dB, over the state-of-the-art error concealment method, STBMA+PDE.

**Table 4**: Average percentage of saved MBs in a corrupted slice and the average PSNR gain of SO-MLD over HO-MLD (QP=36)

| Sequence | Lost MBs | HO-MLD | SO-MLD | $\Delta$ PSNR |
|---|---|---|---|---|
| city | 8980 | 30% | 25% | 0.11 dB |
| crew | 9507 | 22% | 19% | 0.15 dB |
| harbour | 2920 | 55% | 56% | 0.56 dB |
| ice | 8726 | 3% | 3% | 0.00 dB |
| soccer | 5085 | 22% | 19% | 0.10 dB |
| driving | 12154 | 46% | 48% | 0.32 dB |
| ceremony | 10444 | 19% | 16% | 0.07 dB |
| whale show | 7657 | 71% | 71% | 0.22 dB |
| shields | 11225 | 6% | 8% | 0.16dB |
| stockholm | 15550 | 5% | 7% | 0.06dB |

## 7. REFERENCES


[1] International Telecommunications Union, "ITU-T Recommendation H.264 : Advanced video coding for generic audiovisual services," November 2007.

[2] L. Larzon, M. Degermark, and M. Pink, S. Degermark, "UDP Lite for Real Time Multimedia Applications," Tech. Rep., In Proceedings of the QoS mini-conference of IEEE International Conference of Communications, 1999.

[3] L. Trudeau, S. Coulombe, and S. Pigeon, "Pixel Domain Referenceless Visual Degradation Detection and Error Concealment for Mobile Video," in *18th International Conference on Image Processing*. IEEE, September 2011, pp. 2229–2232.

[4] G. Sabeva, S. Ben Jamaa, M. Kieffer, and P. Duhamel, "Robust Decoding of H.264 Encoded Video Transmitted over Wireless Channels," in *8th Workshop on Multimedia Signal Processing*. IEEE, October 2006, pp. 9–13.

[5] N. Nguyen, W. E. Lynch, and T. Le-Ngoc, "Iterative Joint Source-Channel Decoding for H.264 Video Transmission Using Virtual Checking Method at Source Decoder," in *23rd Canadian Conference on Electrical and Computer Engineering*. IEEE, May 2010, pp. 1–4.

[6] Y. Wang and S. Yu, "Joint Source-Channel Decoding for H.264 Coded Video Stream," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 4, pp. 1273–1276, November 2005.

[7] R.A. Farrugia and C.J. Debono, "Robust decoder-based error control strategy for recovery of H.264/AVC video content," *IET Communications*, vol. 5, no. 11, pp. 1928–1938, 2011.

[8] S. Wenger, M.M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, "RTP Payload Format for H.264 Video," http://www.ietf.org/rfc/rfc3984.txt, February 2005.

[9] F. Caron and S. Coulombe, "A Maximum Likelihood Approach to Video Error Correction Applied to H.264 Decoding," in *6th International Conference on Next Generation Mobile Applications, Services, and Technologies*. IEEE, September 2012, pp. 1–6.

[10] Joint Video Team, "H.264/AVC JM reference software," http://iphome.hhi.de/suehring/tml/, November 2011.

[11] Y. Chen, Y. Hu, O. Au, H. Li, and C.W. Chen, "Video Error Concealment Using Spatio-Temporal Boundary Matching and Partial Differential Equation," *IEEE Transactions on Multimedia*, vol. 10, no. 1, pp. 2–15, January 2008.