Iterative Joint Source-Channel Decoding Aided Transmission of Losslessly Compressed Video

Chuan Zhu, Yongkai Huo, Rob G Maunder, Santosh Kawade and Lajos Hanzo
School of ECS, University of Southampton, SO17 1BJ, United Kingdom.
Email: {cz12g09,yh3g09,rm,lh}@ecs.soton.ac.uk, http://www-mobile.ecs.soton.ac.uk

***Abstract*—Iterative source-channel decoding (ISCD) exploits the residual redundancy of the source by iteratively exchanging extrinsic information with the channel codecs to achieve robust transmission. In this treatise we propose a video codec suitable for lossless video compression and ISCD. At the encoder the spatio-temporal redundancy is partially removed at a low complexity by simply evaluating the frame difference (FD), which is then variable-length encoded (VLC). At the receiver we invoke a three-stage concatenated ISCD scheme for exploiting the residual redundancy of the FD. A Markov Random Field (MRF) model based soft-in-soft-out (SISO) module is proposed for exploiting the spatial correlations amongst the adjacent video pixels, using intra-frame coding. More explicitly, the VLC SISO decoder operates by exchanging soft information with both the MRF module and the soft channel decoder. The convergence of the three-stage iterative decoding process is examined using 3D extrinsic information transfer (EXIT) charts. Finally, we show that our system exhibits a substantial E_b/N_0 improvement of about 5.2 dB compared to the corresponding benchmarker schemes.**

I. Introduction

The state-of-the-art video codecs [1], [2] used in wireless tranceivers tend to rely on Shannon's source and channel-coding separation theorem [3]. However, they often fail to remove all the redundancy of the video source, hence they typically exhibit a residual correlation. Furthermore, Shannon's source-and-channel-coding theorem is only valid for idealized Gaussian channels and for a potentially infinite encoding/decoding delay as well as complexity, which is unaffordable in practical applications. Hence, joint source-channel coding (JSCC) [4] was proposed for wireless scenarios, where the residual redundancy of the video source and the intentional redundancy of the channel codecs was jointly exploited for increasing the error resilience of the system. More specifically, soft bit source decoding (SBSD) was proposed in [5] and iterative source channel decoding (ISCD) in [6]. Both of these techniques were designed for exploiting the *a posteriori* probabilities (APPs) of the source parameters using a Markov process model. Furthermore, ISCD exchanged extrinsic information between the source decoder and channel decoder in a turbo-like iterative decoding process.

Variable Length Codes (VLCs) constitute a family of lossless source compression schemes and ISCD has been adopted in a number of VLC-aided systems as well. The authors of [7] proposed an ISCD scheme and the bit-based trellis structure aided VLC decoder of [8]. By contrast, a reversible VLC was invoked as the outer code and a convolutional code was utilized as the inner code in [9], [10]. In [11], Kliewer and Thobaben exploited the residual redundancy after source encoding using a symbol-based soft-input APP decoder for packetized variable-length encoded correlated source signals and demonstrated a substantial error protection improvement with the aid of this iterative decoding technique. The authors of [12] proposed a novel Irregular VLC (IrVLC) scheme for near-capacity JSCC and achieved an improved performance in their serially concatenated system upon combining IrVLC with Trellis Coded Modulation (TCM) by invoking iterative decoding. The authors of [13] also adopted the IrVLC scheme, but this time the IrVLC was concatenated with a rate-1 precoded fast frequency-hopping (FFH) M-ary frequency-shift keying (MFSK) scheme. Three-stage iterative decoding was adopted at the decoder, where extrinsic information was exchanged among the demodulator, the inner decoder and the outer decoder.

When applied to image or video transmission, the ISCD scheme requires sophisticated modeling of the source. In [14], a source decoder and maximum *a posteriori* probability (MAP) channel decoder was applied for jointly decoding the motion vectors of H.264 [2] coded video streams. Kliewer, Görtz and Mertins [15] exploited the *a priori* information inherent in images and invoked both MAP image estimation using Markov random field theory as well as the Markov–Gibbs correspondence relationship discovered in [16]. More explicitly, they modeled images using a Markov Random Field (MRF) for the sake of generating the APPs of the pixels by exploiting the correlation of the adjacent ones and they designed a soft-in-soft-out (SISO) module for exchanging extrinsic information with the adjacent modules in order to perform ISCD.

Against this background, in this treatise we propose a video codec suitable for lossless video compression and ISCD. Our main contributions are as follows:

1) *We further developed the soft source decoder of [15] for the transmission of video sequences rather than still video pictures. More explicitly, we used low complexity frame-differencing (FD) encoding for removing the temporal redundancy and a VLC for removing the residual spatial redundancy of the FD signal, while exploiting the correlations amongst the FD pixels within the current frame with the aid of our MRF model based SISO module.*
2) *A novel three-stage ISCD structure is proposed, which outperforms the two-stage architecture. Furthermore, we examined the convergence of the three-stage iterative decoding process using 3D extrinsic information transfer (EXIT) charts.*

The rest of this paper is organized as follows. In Section II, we introduce our video codec and the structure of the transmitter. Section III presents the algorithms conceived for all the decoding modules and then the corresponding JSCC algorithm is detailed in the context of loss-less video compression, followed by our 3D EXIT chart results. The performance of our system operating both with and without FD is compared to our benchmarkers in Section IV. Finally, we offer our conclusions in Section V.

II. Encoding And Transmission System

A. System Structure

The structure of the proposed system is shown in Fig. 1. First we consider an uncompressed video sequence consisting of N frames, represented in color *YUV* or *RGB* video format. Each sub-sequence is constituted by a single color component of the video (e.g. the Y component in a *YUV* format video), which is then processed by the system of Fig. 1 having color-specific parameter configurations. The n^{th} frame $\boldsymbol{F}_n$ of the single-color video sequence is considered here as an example, assuming $1 \leq n < N$, where $\boldsymbol{F}_n$ contains $(R \times C)$ pixels. Each frame is divided into a series of r-by-c sized macroblocks (MBs) (r and c divide R and C respectively), and each macroblock is scanned into a one-dimensional vector $\boldsymbol{U}_l^n = \left(u_{l,1}^n, ..., u_{l,M}^n\right)$, which contains $M = (r \times c)$ pixels. The vector index l obeys $0 \leq l < L$, where L is the total number of sub-blocks in $\boldsymbol{F}_n$, i.e. we have $L = (R \times C)/M$. Here $u_{l,m}^n$ is the random variable that represents a single source symbol / pixel value and its realization i_U belongs to the legitimate symbol / pixel set $\mathbf{I}_U = \{0, 1, \cdots, 2^K - 1\}$. Moreover, $u_{l,m}^n$ may also be represented by its corresponding bit-vector $\boldsymbol{u}_{l,m}^n$ consisting of K bits.

The financial support of British Telecom as well as the fiscal support of the European Research Concil's Advanced Fellow Grant are gratefully acknowledged.

978-1-4673-5939-9/13/$31.00 ©2013 IEEE

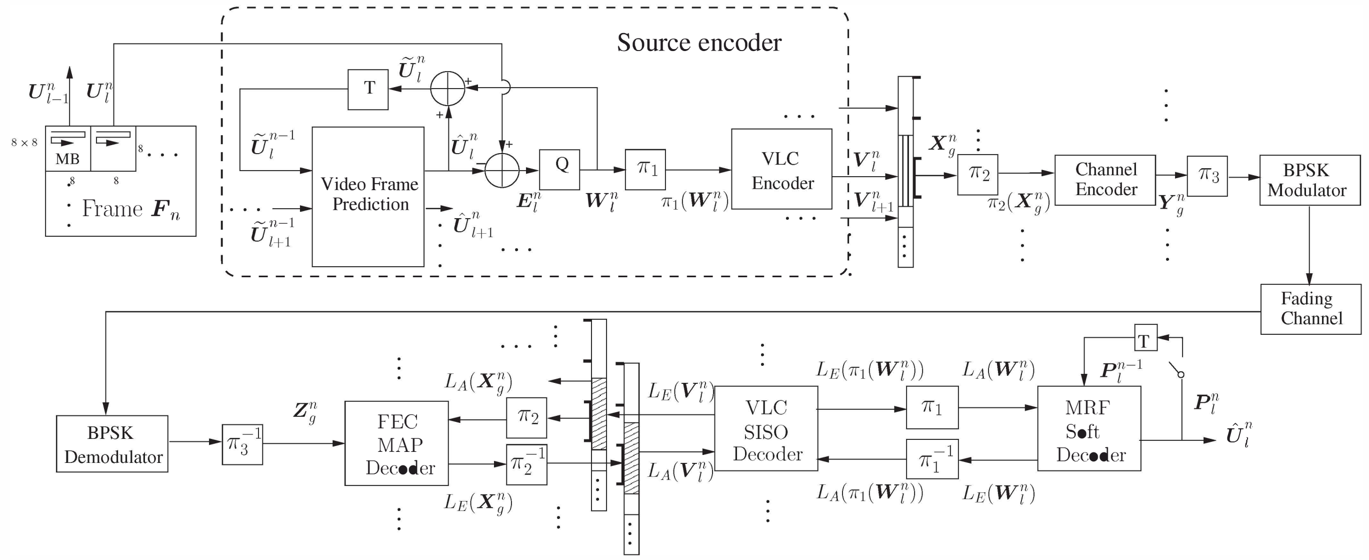

Figure 1. The proposed video encoder and the three-stage joint source-channel decoding receiver

Most of the inter-frame redundancy may be eliminated by subtracting $\hat{\boldsymbol{F}}_n$, namely the prediction of $\boldsymbol{F}_n$, from $\boldsymbol{F}_n$. Then the frame-difference has to be quantized, VLC-encoded and transmitted, as seen in Fig. 1. Firstly, the stored vectors $\tilde{\boldsymbol{U}}^{n-1}_{l=\{0,1,\cdots,L\}}$ generated from the previously reconstructed $(n-1)^{st}$ frame $\tilde{\boldsymbol{F}}_{n-1}$ are used for predicting $\boldsymbol{F}_n$. Although here we used simple frame-differencing, motion-vector-based motion-compensation may also be used, which is the norm in most standardized video codecs [1]. Then the corresponding vector $\hat{\boldsymbol{U}}^n_l = \left(\hat{u}^n_{l,1}, ..., \hat{u}^n_{l,M}\right)$ is extracted from the predicted frame $\hat{\boldsymbol{F}}_n$ and subtracted from the current vector $\boldsymbol{U}^n_l$, as seen in Fig. 1, in order to generate the FD vector $\boldsymbol{E}^n_l = \left(e^n_{l,1}, \cdots, e^n_{l,M}\right)$. The FD element $e^n_{l,m}$ can also be directly mapped into the $(K+1)$ bit vector $\boldsymbol{e}^n_{l,m}$. The specific realization i_E of each $e^n_{l,m} = u^n_{l,m} - \hat{u}^n_{l,m}$ value belongs to the set $\mathbf{I}_E = \{1-2^K, \cdots, -1, 0, 1, \cdots, 2^K - 1\}$. The FD $\boldsymbol{E}^n_l$ is then quantized by the quantizer Q of Fig. 1 and its quantized version $\boldsymbol{W}^n_l$ is added to $\hat{\boldsymbol{U}}^n_l$ for reconstructing $\boldsymbol{U}^n_l$. The reconstructed version $\tilde{\boldsymbol{U}}^n_l$ is then stored and will be used for the prediction of the next video frame. For the sake of avoiding severe error propagation due to the differential structure of the video encoder, $\tilde{\boldsymbol{U}}^{n-1}_{l=\{0,1,\cdots,L\}}$ are periodically reset in order to allow the insertion of intra-frame-coded frames, where the number of consecutive inter-frame-coded frames in a specific group of pictures is denoted by N_{ip}. In the intra-coded scenario, where each frame is encoded separately, the elements of the vectors $\tilde{\boldsymbol{U}}^{n-1}_{l=\{0,1,\cdots,L\}}$ are set to a constant.

The output of the quantizer Q is denoted by $\boldsymbol{W}^n_l = \left(w^n_{l,1}, ..., w^n_{l,M}\right)$, where each vector element results from the quantization of the m^{th} FD pixel, and $\boldsymbol{W}^n_l$ is interleaved on a symbol-by-symbol basis [17] before being encoded by the VLC encoder. The output bit vector of the VLC encoder is denoted by $\boldsymbol{V}^n_l$, which by definition has a variable length of J^n_l. We assume that the vector lengths J^n_l and the VLC codebooks are 'known' at the receiver. Before FEC coding, we first store the vectors $\boldsymbol{B}^n_l$ in a buffer. Then the interleaver π_2 of Fig. 1 having an input length of L_{π_2} is applied. Each time the interleaver π_2 reads L_{π_2} bits from the VLC encoder's output buffer, which are represented by the vector $\boldsymbol{X}^n_g$, $1 \leq g \leq G = \left\lceil \left(\sum_{l=1}^{L} J^n_l\right) / L_{\pi_2} \right\rceil$. The scrambled version of its input is passed to the FEC channel encoder of Fig. 1, yielding the encoded vector $\boldsymbol{Y}^n_g$ of Fig. 1, which has a length of L_{π_3}. Finally this vector is then interleaved by the channel interleaver π_3 before transmission over the channel.

B. VLC Encoder

We use two classic VLC codes, namely Huffman codes [18] and the reversible variable length code (RVLC) of [9], [10] in our design examples. With the advent of sophisticated VLC soft decoding algorithms [19], they constitute excellent candidates for characterizing our iterative decoding process, as it will be demonstrated in Section IV.

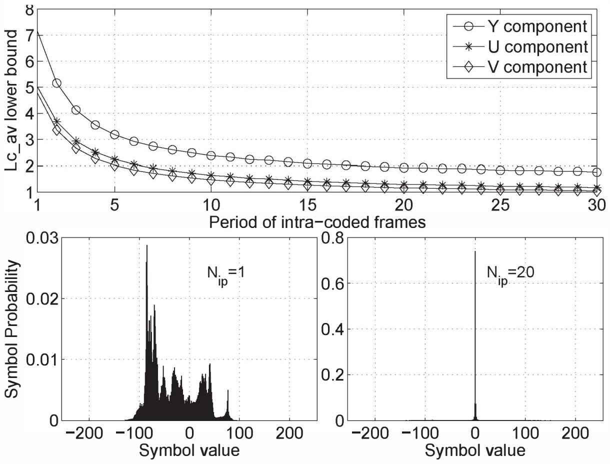

Figure 2. Lower-bound of the average codeword length of the YUV VLC coded 'Akiyo' video sequences and the histogram characterizing the symbol occurrence probabilities of the Y color component, for $N_{ip} = 1$ and $N_{ip} = 20$ of intra-coded frame periods. For the sake of generating our treatment, we handled the intra-frame-coded scenario as part of the inter-frame-coded framework and differentiated it with the aid of $N_p = 1$. This is why the intra-coded pixels are spread over $[-128, \ldots, 127]$, as detailed in the text.

As described in Section II-A with reference to Fig. 1, the VLC encoder takes its input symbols from the interleaved version of $\boldsymbol{W}^n_l$, which is the quantized version of $\boldsymbol{E}^n_l$. Naturally, the VLC codec has to be trained for the sake of generating the required codebooks. Since each color component of the video stream is processed separately, the VLC codec also has to be trained separately for each color component for the sake of determining the probability of occurrence for each legitimate symbol by observing $\boldsymbol{W}^n_l$, in order to generate the codebook as detailed in [1].

At this stage the redundancy exhibited by the correlation of the neighboring symbols is not removed, which however may be efficiently exploited for error-protection using the soft source decoder. More specifically, the VLC encoder ignores the spatial correlation amongst the adjacent symbols and considers the symbol sequence as being i.i.d. The average VLC codeword length L_{c_av} is lower-bounded by the 'entropy' of $\boldsymbol{W}^n_l$s, which is calculated as

$$H = -\sum_{w \in \mathbf{I}_E} P(w) \cdot \log_2 P(w), \qquad (1)$$

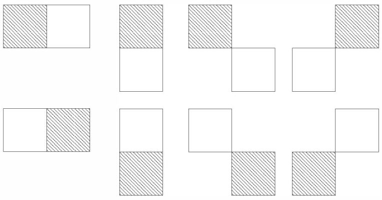

Figure 3. The cliques used in the MRF soft source coding of the proposed system, the shaded boxes represent the current pixel and the other one represents the neighboring pixels in the cliques

where $P(\cdot)$ stands for the probability of occurrence for a specific symbol. First we define the coding rate R_s of the source encoder as the length of the bit-representation of the encoder's input divided by that of its output, which yields $R_s = 8/L_{c\_av}$ in our case. Then we may define the compression ratio of the source encoder as $C_R = 1/R_s$. Let us consider for example the Y component of the 300-frame, (176×144)-pixel quarter common intermediate format (QCIF) and 4:2:0 YUV represented 'Akiyo' video sequence as an example and evaluate the average VLC-length as a function of the intra-frame-period N_{ip} in the top trace of Fig. 2. The intra-frame-encoding scheme constitutes a special case of the inter-frame-coded scheme which is associated with $N_{ip} = 1$. Since each source symbol is represented by $K = 8$ bits, its realization I_U is limited to the range of $\{0, 1, \ldots, 255\}$. As the default value stored in $\tilde{\boldsymbol{U}}_l^{n-1}$ is 128 when encoding the intra-coded frame, the system of Fig. 2 yields a differencing output within the range of $\{-128, \ldots, 0, \ldots, 127\}$. Accordingly, the legitimate range of the FD symbols' realization is spread over $\{-255, \ldots, 0, \ldots, 255\}$. The histogram portrayed in the lower half of Fig. 2 for comparing the $N_{ip} = 1$ and $N_{ip} = 20$ scenarios shows a highly peaked probability distribution for inter-frame-coding, also suggesting that more efficient compression is expected than for intra-frame coding, because the FD signal is close to zero with a high probability and the VLC tends to be more efficient for high-peak PDF, rather than for near-uniform PDFs. It is shown at the top of Fig. 2 that L_{c_av} decreases rapidly, when N_{ip} is increased from 1 to 5 and continues to drop as N_{ip} further decreases, which indicates a lower compression ratio C_R because of the increased temporal distance of the intra-frame coded frames. According to the upper trace of Fig. 2, for the Y component we may only achieve $L_{c\_av} = 7.1$ and $R_s = 1.13$ for intra-frame encoding, while we have $L_{c\_av} = 2$ and $R_s = 4$ for $N_{ip} = 20$ in the inter-frame encoding case.

III. Joint Source-Channel Decoding

A. MRF-Based SISO Source Decoding

In this section we will introduce our soft source decoder architecture invoked for exploiting the spatial correlations amongst the neighboring video or FD pixels and detail its decoding algorithm.

In this treatise we consider lossless VLC source encoding, which implies that the quantizer Q of Fig. 1 imposes no quantization distortion. Moreover, the following derivation may also be used for lossy encoding with minor amendments. We commence by formulating the decoding algorithm for frame-differencing based video source encoding, noting that the decoding method of its intra-frame-encoded counterpart may be readily obtained as a special case. For our low-complexity frame-differencing technique we have $\hat{\boldsymbol{U}}_l^n = \tilde{\boldsymbol{U}}_l^{n-1}$.

Given $e_{l,m}^n = u_{l,m}^n - \hat{u}_{l,m}^n = u_{l,m}^n - \tilde{u}_{l,m}^{n-1}$, we have

$$P_{apri}\left(u_{l,m}^n = i_U\right) = \sum_{\substack{i_U \in \mathbf{I}_U \\ i_E \in \mathbf{I}_E}} P\left(\tilde{u}_{l,m}^{n-1} = i_U - i_E\right) P_{apri}\left(e_{l,m}^n = i_E\right) \tag{2}$$

and

$$P_{apos}\left(e_{l,m}^n = i_E\right) = \sum_{\substack{i_U \in \mathbf{I}_U \\ i_E \in \mathbf{I}_E}} P\left(\tilde{u}_{l,m}^{n-1} = i_U - i_E\right) P_{apos}\left(u_{l,m}^n = i_U\right). \tag{3}$$

This SISO source decoding module may either exchange the symbol probabilities or the log-likelihood ratios (LLRs) of each bits with the inner modules in the concatenated system of Fig. 1. Let us denote the k^{th} bit of the FD signal's bit-representation $\boldsymbol{e}_{l,m}^n$ by $\boldsymbol{e}(k)$. Similarly, we can define the vector $\boldsymbol{i}_E$ as the realization of $\boldsymbol{e}_{l,m}^n$ and its k^{th} bit as $\boldsymbol{i}_E(k)$. If provided with the *a priori* LLR of $\boldsymbol{E}_l^n$ denoted by $L_{apri}(\boldsymbol{E}_l^n)$, we obtain the *a priori* probability (APrP) of $P_{apri}\left(e_{l,m}^n = i_E\right)$ in the form of:

$$P_{apri}\left(e_{l,m}^n = i_E\right) = \bar{c}_{l,m}^n \cdot \exp \sum_{k=0}^{K} \frac{\boldsymbol{i}_E(k)}{2} \cdot L_{apri}\left[\boldsymbol{e}(k)\right]. \tag{4}$$

On the other hand, if the *a posteriori* probability (APP) of $e_{l,m}^n$ is provided, the APP LLRs of $\boldsymbol{E}_l^n$ may be readily obtained as:

$$L_{apos}\left(\boldsymbol{e}(k)\right) = \ln \left[\frac{\sum\limits_{\forall i_E \in \mathbf{I}_E : \mathbf{e}(k)=0} P_{apos}\left(e_{l,m}^n = i_E\right)}{\sum\limits_{\forall i_E \in \mathbf{I}_E : \mathbf{e}(k)=1} P_{apos}\left(e_{l,m}^n = i_E\right)}\right]. \tag{5}$$

Here we briefly describe the soft decoding process of [15] using the MRF model, where the APP of each video pixel value may be calculated by combining the soft information gleaned from the channel with the intrinsic correlations between the pixel itself and the surrounding pixels. Let us simply express the event $u_{l,m}^n = i_U$ as i_U. Then the APP of each source pixel may be expressed as

$$\begin{aligned} P_{apos}\left(i_U\right) &= P\left[i_U \left|\mathcal{N}_{\bar{i}_U}, P_{apri}\left(i_U\right)\right.\right] \\ &= \bar{c} \cdot P\left(i_U \left|\mathcal{N}_{\bar{i}_U}\right.\right) \cdot P_{apri}\left(i_U\right), \end{aligned} \tag{6}$$

where $\mathcal{N}_{\bar{i}_U}$ represents the 8-pixel neighborhood set of the pixel $u_{l,m}^n$. However, in order to avoid an excessive complexity, we also defined a smaller subgroup of mutually dependent symbols, which we referred to as 'cliques' [15]. Then we proceed by examining the system using different cliques sizes for the sake of achieving the best possible performance. Based on our related investigations not included here for the sake of space economy, we adopted the size-2 cliques shown in Fig. 3, where each clique relates $u_{l,m}^n$ to a single adjacent pixel. The variable $P\left(i_U \left|\mathcal{N}_{\bar{i}_U}\right.\right)$ represents the APP of $u_{l,m}^n$ modeled using the so-called Markov-Gibbs correspondence [16], which is generated from the adjacent pixels, and it is expressed as

$$P\left(i_U \left|\mathcal{N}_{i_U}\right.\right) = \frac{1}{Z} \exp\left[-\frac{1}{T} \sum_{c \in C} V_c\left(i_U, i'_U\right)\right], \tag{7}$$

where Z is a normalization factor, T is defined as the 'temperature' parameter, and $V_c\left(i_U, i'_U\right)$ is the potential function, which was expressed as

$$V_c\left(i_U, i'_U\right) = \left|i_U - i'_U\right|^{\delta} \tag{8}$$

for our specific case, where i'_U represents the value of a particular pixel other than $u_{l,m}^n$ in the given clique c. When summing up the potential functions formulated in Eq. (8) over the entire collection of cliques C, we arrive at the summation term seen in the exponent of Eq. (7), which is referred to as the energy function. The choice of the parameters T in Eq. (7) and δ of Eq. (8) is video-content-dependent and the method provided in [15] may be invoked for finding the optimal value.

Finally, we describe the decoding procedure of the MRF soft decoder, which may contain I_{MRF} number of inner iteration loops:

1) Use Eq. (2) and Eq. (4) for generating the APrPs of $\boldsymbol{U}_l^n$, and carry out the MAP estimation of the pixels.
2) Use Eq. (6) in conjunction with Eq. (7) and Eq. (8) to generate the APPs of $\boldsymbol{U}_l^n$.
3) Re-estimate the pixel values in $\boldsymbol{U}_l^n$ with the aid of the APPs using MAP estimation.
4) If the number of affordable iterations I_{MRF} is not exceeded, go to Step 2), otherwise proceed to 5.
5) Store the APPs of $\boldsymbol{U}_l^n$ as $P\left(\tilde{u}_{l,m}^{n-1}\right)$ to be used in the decoding of the next frame.
    a) If the system-level iteration is completed, then we output the current estimate of $\boldsymbol{U}_l^n$s.

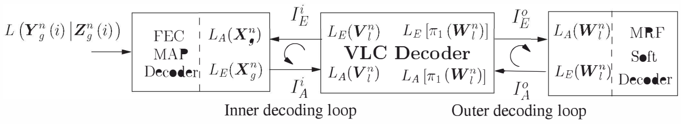
$L\left(\boldsymbol{Y}_g^n(i)\,|\boldsymbol{Z}_g^n(i)\right)$
FEC MAP Decoder
$L_A(\boldsymbol{X}_g^n)$
$L_E(\boldsymbol{X}_g^n)$
I_E^i
I_A^i
$L_E(\boldsymbol{V}_l^n)$
$L_E[\pi_1(\boldsymbol{W}_l^n)]$
VLC Decoder
$L_A(\boldsymbol{V}_l^n)$
$L_A[\pi_1(\boldsymbol{W}_l^n)]$
I_E^o
I_A^o
$L_A(\boldsymbol{W}_l^n)$
$L_E(\boldsymbol{W}_l^n)$
MRF Soft Decoder
Inner decoding loop
Outer decoding loop

Figure 4. Decoding modules that constitute the three-stage iterative decoder, with signal buffer and interleavers omitted

b) If the extrinsic probability of $\boldsymbol{E}_l^n$ is required, then we use Eq. (3) for generating the APPs of $\boldsymbol{E}_l^n$. As a result, we can generate the extrinsic probabilities of each element by using $P_{ext}\left(e_{l,m}^n = i_E\right) = \frac{P_{apos}\left(e_{l,m}^n = i_E\right)}{P_{apri}\left(e_{l,m}^n = i_E\right)}$

c) If the extrinsic LLRs of $\boldsymbol{e}_{l,m}^n$ are required, then Eq. (3) can be used for generating the APPs of $\boldsymbol{E}_l^n$. As a next step, the *a posteriori* LLRs of $\boldsymbol{e}_{l,m}^n$ can be obtained using Eq. (5). Finally, the extrinsic LLRs of each element can be obtained by using $L_{ext}[\boldsymbol{e}(k)] = L_{apos}[\boldsymbol{e}(k)] - L_{apri}[\boldsymbol{e}(k)]$.

B. Three Stage Iterative Decoding Process

As shown in Fig. 1 and Fig. 4, the three-stage iterative decoder consists of three main SISO decoding modules, namely the soft channel decoder, the VLC SISO decoder and the MRF-based soft source decoder. The decoding process includes two iterative decoding loops, which are the inner loop where the soft channel decoder exchanges its soft bit information estimates with the VLC SISO decoder, and the outer loop where the VLC SISO decoder exchanges the soft-information with the MRF-based source decoder in a iterative manner. Furthermore, the decoder operates on a frame-by-frame basis, which implies that both the channel decoder and the VLC decoder have to complete processing the current video frame, before commencing the processing of the next frame. Finally, if the current frame is the FD frame, its decoding will rely on the APP result of the previous frame.

In the three-stage iterative decoding process, the channel decoder invokes the BCJR algorithm and forwards the extrinsic bit-LLR information $L_E(\boldsymbol{X}_g^n)$ to the VLC SISO soft decoder, which is evaluated from the LLR $L\left(\boldsymbol{Y}_g^n(i)\,|\boldsymbol{Z}_g^n(i)\right),\ 0 < i < L_{\pi_3}$ of the de-interleaved channel output, while the *a priori* bit LLR $L_A(\boldsymbol{X}_g^n)$ is gleaned from the VLC decoder, as seen in Fig. 1. The interleaver π_2 de-interleaves $L_E(\boldsymbol{X}_g^n)$ of Fig. 1 received from the channel decoder, and again, stores the results in a buffer for the VLC decoder, which uses it as *a priori* information in the next phase after all the signal blocks $\boldsymbol{X}_{g=\{1,\cdots,G\}}^n$ of Fig. 1 have been processed. At the same time the interleaver π_2 reads its input from another buffer, which was previously loaded with the extrinsic output of the VLC decoder and generates the interleaved version as $L_A(\boldsymbol{X}_g^n)$.

Furthermore, as shown in Fig. 4, the four-port VLC SISO soft decoding module reads the *a priori* LLR of $\boldsymbol{V}_l^n$ from the frame buffer, and provides extrinsic LLRs for the FEC decoder module seen in Fig. 4 in order to participate in the inner iteration loop. As for the outer iterative loop, the VLC SISO decoder of Fig. 4 exchanges its extrinsic information of $\boldsymbol{W}_l^n$s with the MRF SISO decoder. As for the decoding algorithm of the VLC SISO decoder, a symbol-based trellis representation of the VLC decoding process as well as the corresponding symbol-by-symbol MAP decoding method were detailed in [19]. However, Bauer and Hagenauer only used a VLC as their outer code in the serially concatenated system, and the corresponding decoding module only exchanged soft information with the channel decoder. By contrast we adopted the concept of a SISO APP module [20] for further developing the method of [19] in order to arrange for the VLC decoder to act as an intermediate code, which exchanges extrinsic symbol probabilities with the soft source decoder as well. Moreover, for the convenience of observing the convergence behavior of the system using EXIT charts, we also allow the exchange of extrinsic bit LLRs between the VLC decoder and the MRF decoder. Hence, we use the natural mapping of the symbols in $\boldsymbol{W}_l^n$s to bits, and use an expression similar to Eq. (5) for directly generating the extrinsic bit LLRs from the extrinsic symbol probabilities.

Finally, we characterize the convergence behavior of the proposed three-stage iterative decoding process using the Akiyo sequence and Huffman VLCs, as mentioned in Section II-B. Let us denote the mutual information (MI) between $L_A(\boldsymbol{V}_l^n)$ and $\boldsymbol{V}_l^n$ by I_A^i, where the superscript i indicates that the information is exchanged in the inner decoding loop (o indicates the outer loop), and the subscript A indicates that the MI term represents the *a priori* information (E indicates the extrinsic information). Similarly, we can define I_E^i as the MI between $L_E(\boldsymbol{V}_l^n)$ and $\boldsymbol{V}_l^n$, I_A^o as the MI between $L_A[\pi_1(\boldsymbol{W}_l^n)]$ and $\pi_1(\boldsymbol{W}_l^n)$, while I_E^o as the MI between $L_E[\pi_1(\boldsymbol{W}_l^n)]$ and $\pi_1(\boldsymbol{W}_l^n)$. Since the FEC MAP decoder exploits the channel information $\boldsymbol{Z}_g^n$, its EXIT function inherently depends on E_b/N_0 and hence may be written as $I_A^i = f_{FEC}\left[I_E^i, E_b/N_0\right]$. By contrast, the inner VLC SISO decoder's EXIT function may be formulated as $I_E^i = f_{VLC}^i\left[I_A^i, I_A^o\right]$, while the outer VLC SISO decoder's EXIT function can be expressed as $I_E^o = f_{VLC}^o\left[I_A^i, I_A^o\right]$. Finally, the EXIT function of the soft source decoder may be formulated as $I_A^o = f_s^o\left[I_E^o\right]$.

The resultant 3D EXIT charts recorded at the E_b/N_0 value of 12.4 dB are presented in Fig. 5 and Fig. 6, while a Y frame of the 'Akiyo' sequence is used as the source for our example. The EXIT functions of the receiver components involved in the inner decoding loop are shown in Fig. 5, while Fig. 6 portrays the EXIT functions of the components involved in the outer decoding loop. We may observe from Fig. 6 that the EXIT curve of the MRF-based soft source decoder may not reach the (1,1) point in the I_A^o-I_E^o plane, thus the decoding trajectory is prevented from reaching the (1,1,1) point of perfect convergence in the 3D EXIT chart. However, the trajectory may still closely approach the (1,1) point in the I_A^i-I_E^i plane. It is shown in Fig. 6 that when we have $I_A^o = 0$, the EXIT curve of the FEC decoder and that of the Huffman VLC decoder intersect at the point of (0.93,0.42) in the I_A^i-I_E^i plane, which means that the two-stage (VLC-RSC) iterative decoder operating without the soft-source decoder may exhibit a residual error floor due to the poor codeword distances of the Huffman code. However, after four iterations using the three-stage decoding process of Fig. 5, the decoding trajectory reached the (0.98,0.91) point when $I_A^o = 0.68$, which suggests a substantial improvement in the BER performance.

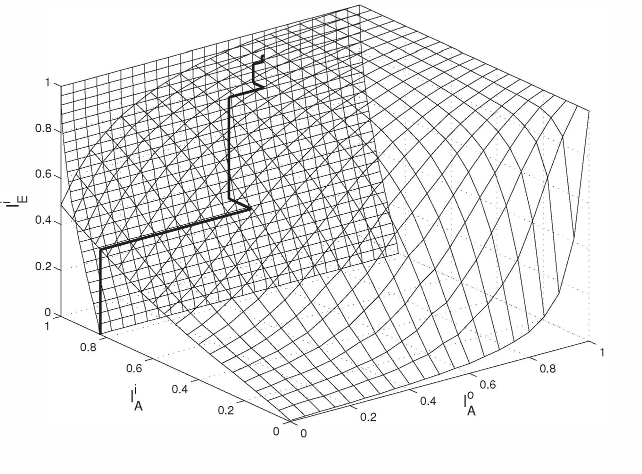

Figure 5. Inner decoding functions and decoding trajectory for an E_b/N_0 of 12.4 dB. The Y frame of the '*Akiyo*' sequence was used as our source, and a Huffman code was employed as the VLC code

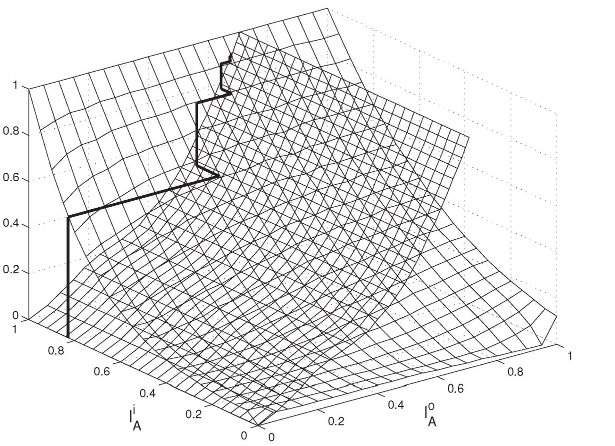

Figure 6. Outer decoding functions and decoding trajectory for an E_b/N_0 of 12.4 dB. The Y frame of the '*Akiyo*' sequence was used as our source, and a Huffman code was employed as the VLC code

IV. Performance Results

A. Parameter Configurations

This section details the parameters used in our following simulations. For the video source, the first 50 frames of the (176×144)-pixel quarter common intermediate format (QCIF) 4:2:0 YUV representation based 'Akiyo' sequence is employed. Thus the frame size is given by $R = 176$ and $C = 144$ for the QCIF luminance-frame and $R = 88$ and $C = 72$ for the chroma frame. The macroblocks size used is $(r \times c) = (8 \times 8) = 64$, thus each luminance frame is divided into $22 \times 18 = 396$ macroblocks, while each QCIF chroma-frame is divided into $(11 \times 9) = 99$ macroblocks.

As described in Section III-A, the search method provided in [15] is invoked for generating the optimal values for the MRF parameters T and δ of Eq. 7 and Eq. 8 according to the video sequence, but the details of the search process are omitted here for conciseness. As a result, $T = 2$ and $\delta = 0.3$ were chosen for the decoding of the QCIF luminance-frame, while $T = 2$ and $\delta = 0.4$ for the chroma-frames. The number of inner-loop iterations was $I_{MRF} = 1$ for the MRF soft source decoder throughout this treatise.

As for the FEC, a 2/3-rate RSC encoder using the generator polynomials of $g_1 = 1011$, $g_2 = 1101$ is employed, which can be represented as $\text{G} = [1, g_2/g_1]$, where g_1 denotes the feedback polynomial and g_2 is the feedforward polynomial. The corresponding puncturing pattern is $[1\,1; 1\,0]$, where the two rows correspond to the output of the systematic bit and that of the g_2, respectively. The interleaver length is $L_{\pi_2} = 512$ and random interleavers are used.

The BPSK modulated bits are transmitted over an uncorrelated non-dispersive Rayleigh channel. To specify the E_b/N_0 rario used in our simulations, first we have to define the coding rate of our system. Firstly, the concept of the 'natural' code rate (NCR) R_n [21] should be introduced. According to Shannon's channel capacity theorem [3] proposed for the transmission of i.i.d source bits, the true entropy of the video sequence should be considered when calculating the energy efficiency per bit. Let us denote the true entropy of the video source file by S_e, while the uncompressed video file has a size of S_r bits. Then the NCR becomes $R_n = S_e/S_r$, which may be physically interpreted as the code rate of a natural inherent channel code's redundancy. More explicitly, this redundancy is equivalent to an identical-rate external channel code's redundancy. This NCR will be exploited by our receiver. However, we have a predicament, since the true entropy of the video source cannot be readily evaluated. Therefore we generated the encoded version of the source sequence by the near-lossless coding mode of the H.264 codec [1], [2], which was utilized for approximating its entropy S_e. Since the compression ratio of the Akiyo clip described above using the H.264 codec is 8.7, the NCR R_n of the Akiyo clip becomes $1/8.7$. Finally, in the simulations the E_b/N_0 (dB) value is calculated as $E_b/N_0 = 10\log_{10}\frac{E_s}{N_{\bullet}R_{\text{total}}}$, where R_{total} is defined as

$$R_{\text{total}} = R_n R_s R_c. \tag{9}$$

Thus the BER versus E_b/N_0 curves should be shifted to the right according to the redundancy inherent in the encoded sequence, as well as according to the artificial redundancy imposed by both channel coding and VLC coding.

B. System Performance Results

In this section we present the simulation results characterizing our proposed scheme, which was introduced in Section III alongside a couple of benchmarkers, while using the parameter configurations described in Section IV-A and $I_{inner} = I_{outer} = 4$ iterations for both the outer and inner iteration loops. Here two types of measurements are utilized for characterizing the attainable video quality, namely the PSNR versus E_b/N_0 curves and the bit error ratio (BER) versus E_b/N_0 curves. When measuring the PSNR values of the reconstructed video, we employ the technique used in the H.264 reference software JM and set the minimum value of the total averaged mean squared error (MSE) value between the reconstructed and the original frame to 1, so that encountering infinite PSNR values resulting from a perfectly reconstructed video frame can be avoided. Hence the maximum unimpaired video PSNR that may be obtained at the receiver is about 48.1 dB.

Here we introduce the two benchmarkers that are used for comparison with our proposed systems, namely the Lossless-H.264-RSC system and the MRF-RSC two-stage iterative decoding system of [15]. For our first benchmarker, the same RSC codec is used for transmitting and receiving the video compressed in the near-lossless mode of the JM software, as in our proposed system. For the latter one, we used the MRF-based soft decoder introduced in [15] in conjunction with $I = 4$ iterations, where simple natural mapping of the source symbols to bits was used, along with the optimal MRF parameters.

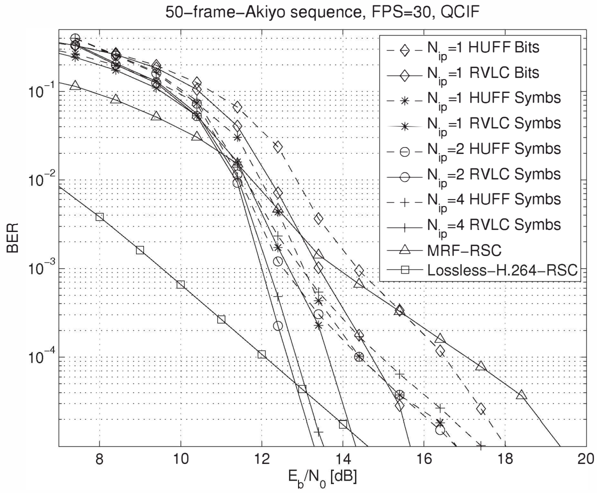

Figure 7. BER comparison of the MRF-VLC-RSC system and of the Lossless-H.264-RSC and MRF-RSC [15] benchmarkers using the *Akiyo* sequence for transmission over uncorrelated Rayleigh channels.

The BER versus E_b/N_0 performance of our scheme is presented in Fig. 7, while the PSNR versus E_b/N_0 performance is shown in Fig. 8. In order to choose the best system configurations, we carried out comparisons between the systems having different intra-coded frame periods, different VLC codes and different methods of information exchange between the soft source decoder and the VLC SISO decoder, which were described in Section III. As seen in Fig. 7 and Fig. 8 the dashed lines represent the system relying on Huffman codes, while the solid lines indicate the ones using the RVLC codes of [9], [10]. Since a RVLC has a symmetric construction, which supports both forward and backwards decoding, it is typically less susceptible to error-propagation than Huffman VLCs. Observe in Fig.

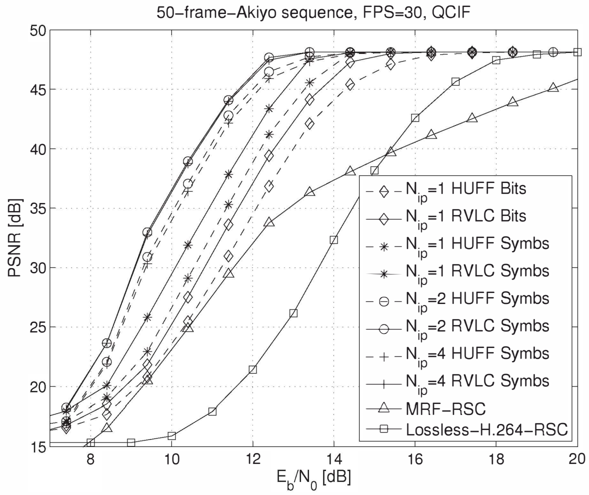

Figure 8. Reconstructed video quality of the MRF-VLC-RSC system and of the Lossless-H.264-RSC and MRF-RSC [15] benchmarkers using the *Akiyo* sequence for transmission over uncorrelated Rayleigh channels.

7 that the RVLC codes result in a steeper BER curve and better BER as well as PSNR performance. For example, at a BER of 1×10^{-4}, the RVLC aided system outperforms the Huffman coded system by about 1 dB, 2 dB and 2.2 dB in terms of the E_b/N_0 required for $N_{ip} = 1, 2, 4$, respectively. Furthermore, the system that exchanges extrinsic symbol probabilities rather than extrinsic LLRs has a better performance when $N_{ip} = 1$, while the former one outperforms the latter one by about 1 dB and 2.1 dB when using RVLC and Huffman codes, respectively. Finally, our comparisons between the systems having different N_{ip} values was also carried out, and the immediate conclusion emerged that $N_{ip} = 2$ provided a better performance than $N_{ip} = 1$ and $N_{ip} = 4$. This is because the low NCR of the uncompressed video $(N_{ip} = 1)$ radically and excessively reduces the bit-energy. By contrast, the error-propagation effects become severe for $N_{ip} = 4$. Thus the system associated with $N_{ip} = 2$ and relying on the RVLC code as well as on symbol-based information exchange within the inner decoding loop is preferred. It can also be seen in Fig. 7 and Fig. 8 that a BER of about 9×10^{-3} and a PSNR of about 44 dB can be achieved by our scheme at an E_b/N_0 of 11.4 dB in the best case.

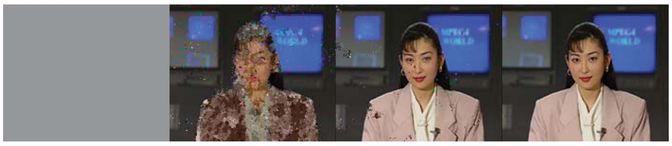

Figure 9. A frame comparison of the decoded *Akiyo* sequence at $E_b/N_0 = 11.4$ dB. The frames are reconstructed by Lossless-H.264-RSC, MRF-RSC [15], MRF-VLC-RSC($N_{ip} = 1$), MRF-VLC-RSC($N_{ip} = 2$), respectively.

By observing Fig. 7 we may see that our proposed scheme outperforms the MRF-RSC system by about 3.4 dB at a BER of 1×10^{-4}. Although the Lossless-H.264-RSC scheme achieves a better BER performance at low E_b/N_0 values, its bits are sensitive to transmission errors. It can be seen from Fig. 8 that at a PSNR of 45 dB our scheme outperforms the MRF-RSC and the Lossless-H.264-RSC scheme by about 8 dB and 5.2 dB, respectively.

Finally, our subjective comparison of the decoded Akiyo sequence at $E_b/N_0 = 11.4$ dB is portrayed in Fig. 9, demonstrating that the proposed MRF-VLC-RSC scheme is capable of substantially better recovering the error-infested video than the benchmarkers.

V. Conclusions

In this treatise a video codec suitable for lossless video compression and ISCD was proposed. At the video encoder the spatio-temporal redundancy is partially removed by low-complexity frame-differencing and VLC. A three-stage ISCD scheme was proposed for exploiting the residual source redundancy. The MRF model based SISO module was proposed for exploiting the spatial correlations of the video pixels. Simulations were carried out and the appropriate system parameters were chosen. Our results demonstrated that the system exhibits a substantial performance improvement compared to the corresponding benchmarkers.

In our future work we will improve our temporal prediction method of frames with the aid of sophisticated motion compensation, and we shall replace our VLC with more efficient entropy codecs in order to improve the attainable compression ratio.

References

[1] L. Hanzo, P. Cherriman, and J. Streit, *Video Compression and Communications: From Basics to H.261, H.263, H.264, MPEG2, MPEG4 for DVB and HSDPA-Style Adaptive Turbo-Transceivers*. New York: John Wiley, 2007.

[2] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, *ITU-T Rec. H.264/ISO/IEC 14496-10 AVC: Advanced Video Coding for Generic Audiovisual Services*, March 2010.

[3] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, June and October 1948.

[4] K. Sayood and J. Borkenhagen, “Use of residual redundancy in the design of joint source/channel coders,” *IEEE Transactions on Communications*, vol. 39, pp. 838–846, June 1991.

[5] T. Fingscheidt and P. Vary, “Softbit speech decoding: A new approach to error concealment,” *IEEE Transaction on Speech and Audio Processing*, vol. 9, pp. 240–251, March 2001.

[6] M. Adrat and P. Vary, “Iterative source-channel decoding: Improved system design using Exit charts,” *EURASIP Journal on Applied Signal Processing*, vol. 2005, pp. 928–941, January 2005.

[7] R. Bauer and J. Hagenauer, “On variable length codes for iterative source/channel decoding,” in *Data Compression Conference, 2001. Proceedings. DCC 2001.*, pp. 273 –282, 2001.

[8] V. Balakirsky, “Joint source-channel coding with variable length codes,” in *IEEE International Symposium on Information Theory*, p. 419, June 1997.

[9] Y. Takishima, M. Wada, and H. Murakami, “Reversible variable length codes,” *IEEE Transactions on Communications*, vol. 43, pp. 158 –162, Feb/Mar/Apr 1995.

[10] C.-W. Lin, Y.-J. Chuang, and J.-L. Wu, “Generic construction algorithms for symmetric and asymmetric RVLCs,” in *The 8th International Conference on Communication Systems, 2002. ICCS 2002.*, vol. 2, pp. 968 – 972 vol.2, November 2002.

[11] J. Kliewer and R. Thobaben, “Iterative joint source-channel decoding of variable-length codes using residual source redundancy,” *IEEE Transactions on Wireless Communications*, vol. 4, pp. 919–929, May 2005.

[12] R. Maunder, J. Wang, S. Ng, L.-L. Yang, and L. Hanzo, “On the performance and complexity of irregular variable length codes for near-capacity joint source and channel coding,” *IEEE Transactions on Wireless Communications*, vol. 7, pp. 1338 –1347, April 2008.

[13] S. Ahmed, R. Maunder, L.-L. Yang, and L. Hanzo, “Iterative detection of unity-rate precoded FFH-MFSK and irregular variable-length coding,” *IEEE Transactions on Vehicular Technology*, vol. 58, pp. 3765 –3770, September 2009.

[14] Y. Wang and S. Yu, “Joint source-channel decoding for H.264 coded video stream,” *IEEE Transactions on Consumer Electronics*, vol. 51, pp. 1273–1276, November 2005.

[15] J. Kliewer, N. Görtz, and A. Mertins, “Iterative source-channel decoding with Markov random field source models,” *IEEE Transactions on Signal Processing*, vol. 54, pp. 3688–3701, October 2006.

[16] S. Geman and D. Geman, “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, pp. 721 –741, November 1984.

[17] Y.-J. Wu and H. Ogiwara, “Symbol-interleaver design for turbo trellis-coded modulation,” *IEEE Communications Letters*, vol. 8, pp. 632 – 634, October 2004.

[18] D. Huffman, “A method for the construction of minimum-redundancy codes,” *Proceedings of the IRE*, vol. 40, pp. 1098 –1101, September 1952.

[19] R. Bauer and J. Hagenauer, “Symbol-by-symbol MAP decoding of variable length codes,” in *in Proc. 3. ITG Conf. on Source and Channel Coding*, pp. 111–116, 2000.

[20] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “A soft-input soft-output APP module for iterative decoding of concatenated codes,” *IEEE Communications Letters*, vol. 1, pp. 22 –24, January 1997.

[21] Y. Huo, C. Zhu, and L. Hanzo, “Spatio-temporal iterative source-channel decoding aided video transmission,” *IEEE Transactions on Vehicular Technology*, to appear. Available at `https://eprints.soton.ac.uk/344750/1/st_iscd.pdf` .