The FlexCode Channel Coding Approach

Laurent Schmalen[†], *Birgit Schotsch*[†], *Marco Jeub*[†], *Peter Vary*[†], *Thorsten Clevorn*[⋆]

[†] Institut für Nachrichtengeräte und Datenverarbeitung (IND), RWTH Aachen, 52056 Aachen
E-Mail: {schmalen,vary}@ind.rwth-aachen.de
Web: www.ind.rwth-aachen.de
[⋆] *is now with* Infineon Technologies, Duisburg, Germany
E-Mail: thorsten.clevorn@infineon.com

Abstract

The *FlexCode* project is a joint research project of KTH Stockholm, RWTH Aachen University, Ericsson AB, Nokia Siemens Networks, and Orange/France Telecom under the umbrella of the sixth framework programme of the European Commission (http://www.flexcode.eu). In this paper we present the channel coding approach used in the *FlexCode* project. Furthermore, a brief introduction to the channel model utilized in *FlexCode* is given. The presented channel encoder enables iterative source-channel decoding at the receiver in order to achieve near-capacity transmission of the source coder parameters. The structure of the encoder enables to flexibly select the coding rate as well as the size of the input block. This joint source-channel coding approach is able to handle both considered types of quantization in the *FlexCode* project: constrained entropy and constrained resolution. On the other hand, the channel coding approach presented in this paper is able to achieve near Shannon-limit performance for arbitrary bit streams which is shown by a simulation example.

1 Introduction

The increasing heterogeneity of communication networks and the variability in user requirements form a challenge for source and channel coding algorithms. To address this challenge, the aim of the *FlexCode* project is to create a speech and audio coder that can adapt instantaneously to network and user requirements. Depending on the current network conditions, the available data rate and the computational power available at the terminal or base station, the channel coder can flexibly select the required coding rate and thus set up the source coder in a way that the required quality of service can be achieved. Therefore, a highly flexible channel coder with considerably high performance under all conditions has to be developed.

With the discovery of Turbo codes, channel coding close to the Shannon limit has become possible with moderate computational complexity. In the past years, the Turbo principle of exchanging extrinsic information between separate channel decoders has also been extended to other receiver components. In a Turbo-like process the residual redundancy of source codec parameters such as scale factors or predictor coefficients for speech, audio, and video signals can be exploited by *iterative source-channel decoding* (ISCD) [1, 2]. This residual redundancy occurs due to imperfect source encoding resulting for instance from delay and complexity constraints. It can be utilized by a *soft decision source decoder* (SDSD) [3] which exchanges extrinsic reliabilities with a channel decoder.

A joint source-channel coding approach with iterative decoding has been selected for the protection against transmission errors in the *FlexCode* project. However, several modifications had to be made in order to make the approach feasible for the application to the source coder. These modifications will be explained and highlighted within the present paper.

The paper is organized as follows: In Section 2 the *FlexCode* channel model is briefly explained while in Section 3 the channel coder is introduced. The paper concludes with simulation examples in Section 4.

This work was financed by the European Union under Grant FP6-2002-IST-C 020023-2 FlexCode

2 The FlexCode Channel Model

Owing to the possibility of the channel coding approach to be adoptable to any kind of transmission channel a generic channel model has been developed. It is capable of modeling both circuit switched and packet switched transmission.

The circuit switched transmission channel delivers soft information on bit level to the channel decoder. It consists of a *binary input soft output (BISO) channel* in the baseband supporting a wide variety of analytical models (e.g., AWGN, Rayleigh fading, Rice fading) and modulation schemes (e.g., PSK, QAM) [4].

Furthermore an IP layer is integrated to simulate packet erasures with predefined error pattern stored in an ITU-T G.192 Error Insertion Device (EID) compatible file format [5]. The use of UDP-Lite is most suitable for the channel model because unlike the standard UDP protocol it allows partial check sums covering only the most relevant parts of the IP packet (e.g., header information) [6]. Therefore the iterative channel decoder can decode the payload even when it is partly corrupted. The handling of corrupted headers is out of the scope of *FlexCode* and not considered.

The predefined error patterns are generated using the Gilbert-Elliot or the Bellcore model. Additionally traces of real-channel measurements recorded in different networks have been analyzed with respect to packet loss rate, delay and jitter and are stored as EID as well.

3 The FlexCode Channel Coder

The channel coding concept has to be adapted to the basic structure of the source encoder. The baseline *FlexCode* source coding concept is described for instance in [7] and [8]. For each frame, the source encoder provides a set of parameters which can be grouped into two main parts: model parameters and transform coefficients. The model parameters include for example the LP coefficients and gain factors. Using the model parameters the source encoder determines the quantizer setup for the transform coefficients:

- In the case of constrained resolution (CR) quantization, the source encoder determines the bit allocation of the transform coefficients, i.e., the number of quantization levels to be used for the considered parameter.

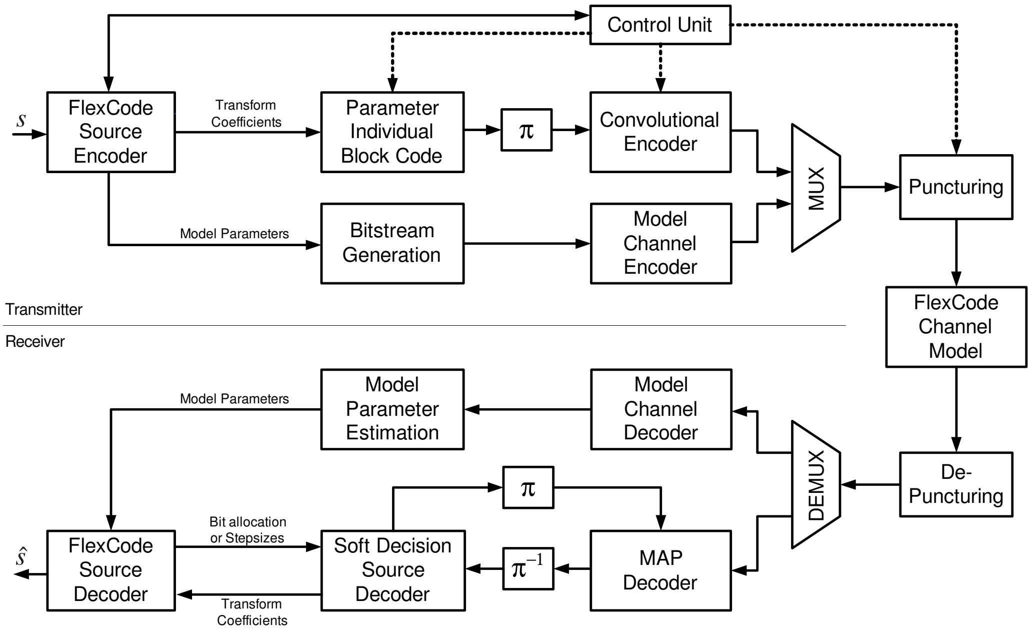

Figure 1: Block diagram of the *FlexCode* baseline channel coder

- In the case of constrained entropy (CE) quantization, the source encoder uses the model parameters to determine the distribution of the transform coefficients and the step size of the uniform quantizer. Using this information, an entropy coder (for example an arithmetic coder) can efficiently generate a compressed bit stream.

As a result of this source coding concept, it has been found that it is not feasible to perform joint source-channel decoding of the model parameters and the transform coefficients. The source-channel decoder requires knowledge about the model in order to determine the encoding parameters of the transform coefficients like bit allocation and step sizes. Therefore, we propose to utilize a separate transmission of the model parameters and the transform coefficients. The resulting structure which defines the *FlexCode* baseline channel coder is depicted in Fig. 1.

The model parameters are grouped and if entropy constrained quantization of the model parameters is utilized, compressed using an arithmetic coder. On the other hand, if resolution constrained quantization is employed, arithmetic encoding is not required. The generation of the bit stream using either arithmetic coding or not is summarized in the block *Bitstream Generation* in Fig. 1. Afterwards, the grouped bit stream is encoded using a strong conventional channel code. This channel code could be for instance an iteratively decodable code such as a Turbo code or an LDPC code. The bit rate for transmitting the model parameters is rather small and more or less fixed (around 5 kbit/s, see [9]). As LDPC codes and Turbo codes might show a considerably high error floor due to the small block size (and interleaver), it might be advantageous to deploy a "conventional" channel coding scheme such as the concatenation of a Reed-Solomon code and a convolutional code. This concatenation has been widely employed in existing communication systems [10]: The convolutional decoder at the receiver, which might be a Viterbi decoder, produces burst errors at its output which can be efficiently corrected by the Reed-Solomon decoder. By puncturing the convolutional code, the rate and the robustness requirements can be efficiently adjusted.

The transform coefficients on the other hand are encoded using an iterative source-channel coding system. For a detailed description and implementational details of this joint source-channel coding approach with iterative decoding, we refer the reader to the literature, e.g., [11]. As the approach depends on the type of quantization (constrained resolution or constrained entropy) two cases have to be considered. The details for both cases will be given in Sections 3.1 and 3.2. The basic concept, however, is the same: A bit stream is generated and a block encoder adds a certain amount of artificial redundancy (depending on the overall coding rate) to the bit stream. This bit stream is interleaved using the interleaver presented in Sec. 3.3 and then encoded by a convolutional encoder. At the receiver, a MAP decoder and an SDSD (which may exploit the residual redundancy of the transform coefficients, if available) iteratively exchange extrinsic information. After a certain number of iterations have been carried out, the transform coefficients are estimated using the MAP rule.

The information about the bit allocation (in the case of constrained resolution quantization) or the quantizer step sizes and parameter distribution (in the case of constrained entropy quantization) is derived by the *FlexCode* source decoder from the model parameters which are decoded first. This information is then used by the soft decision source decoder in the iterative source-channel decoding process.

3.1 Constrained Resolution (CR) Quantization

In the case of CR quantization, the source encoder determines the number of quantization levels (and thus also the number of required bits) for each transform coefficient. The bit stream generation in this case is simply the assignment of the natural binary representation of the quantized index to the coefficient. If, due to adverse channel conditions, additional channel coding redundancy shall be added, one or several parity check bits are added to each transform coefficient. This is performed by the block *Parameter Individual Block Code* in Fig. 1. At the receiver, the SDSD can exploit all available statistical information on the transform coefficients such as an unequal distribution or correlation as well as the artificial redundancy added by the parameter individual block code.

3.2 Constrained Entropy (CE) Quantization

In the case case of CE quantization, an arithmetic encoder generates a variable-length bit stream using the statistical information on the transform coefficients. This bit stream can be partitioned into groups of several bits. A small block code which adds one or several parity bits depending on the number of available redundancy bits, is then assigned to these groups. Therefore, the block *Parameter Individual Block Code* performs a block code not on parameter basis but on blocks of bits, which can be considered as parameters. The assignment of block codes to the groups of bits can be optimized using the concept of irregular codes and index assignments [12], [13]. At the receiver, the SDSD cannot exploit any statistical properties as the bits after arithmetic coding are assumed to be equiprobable. The SDSD reduces in this case to a MAP decoding of the single block codes.

3.3 Flexible Interleavers

The *FlexCode* channel coder uses the Turbo principle of exchanging extrinsic information between two component codes (here between channel decoder and soft decision source decoder). An essential element of a transmission or storage system employing the Turbo principle is the interleaver. As the *FlexCode* source coder can adapt on the fly to different scenarios and source conditions, the size of the data to be channel coded might be subject to frequent changes. Furthermore, the constrained entropy quantizer leaves different amounts of redundancy in the quantized data such that, after possible data compression (if, e.g., arithmetic codes, are used) the size of the packets to transmit varies significantly. Therefore, interleavers are needed which can change their size on the fly with moderate computational complexity. Such interleavers are called *prunable* interleavers [14].

Several communication systems, such as UMTS, already employ prunable, variable-size interleavers [15]. These interleavers are based on the Zech logarithm and utilize Galois Field arithmetic. However, a different approach for generating prunable interleavers has been chosen for the *FlexCode* channel coder. The *FlexCode* interleaver is based on [16] which extends an S-random interleaver by adding additional entries such that the S-condition remains fulfilled. The pruning can be easily performed on the fly during (de-)interleaving. For implementational details, we refer to [17].

3.4 Future Extensions

In the current baseline version of the channel coder, only scalar quantization of the model parameters and transform coefficients is included. However, further gains are expected by employing lattice quantization (see [7]). The developed channel coder is able to handle vector quantized parameters without changes.

In order to combat the negative effects of packet losses, multiple description coding will be used by the *FlexCode* source encoder. The integration of multiple description coding to the *FlexCode* channel coder is currently part of ongoing research work.

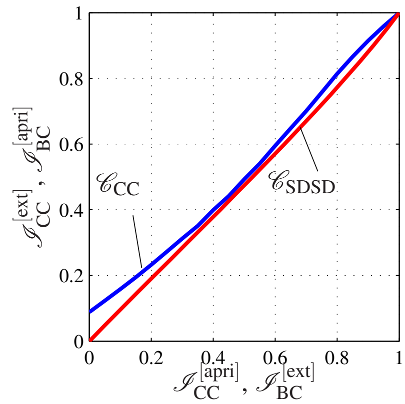

Figure 2: EXIT chart analysis (snapshot, CR case) at $E_s/N_0 = -2.6\,\text{dB}$

4 Simulation Examples

The first simulation example shows a snapshot of the *FlexCode* coding system: The *FlexCode* source coder operates at a coding rate of 24 kbit/s and the channel coder at a coding rate of 1/2. Model parameters and transform coefficients are encoded with rate 1/2 such that the total amount of data to be transmitted on the channel amounts to 48 kbit/s. Figure 2 shows a snapshot of the EXIT chart [18] analysis of the system. It can be seen that a (narrow) decoding tunnel exists between the characteristic of the channel code $\mathscr{C}_{CC}$ (rate-1 recursive non-systematic with generator polynomials $G^{\mathsf{C}}(D) = \left(1, \frac{1}{1+D+D^2+D^3}\right)$) and the characteristic of the SDSD $\mathscr{C}_{SDSD}$. Note that the SDSD in this example does not exploit any residual redundancy in the quantized parameters. Better performance is expected if redundancy such as unequal parameter distribution and correlation is exploited.

In a second simulation, the generic *FlexCode* channel coder is utilized for the transmission of an arithmetically coded bit stream. However, we do not perform this simulation using the *FlexCode* system but using a Bernoulli source which emits blocks consisting of 20000 equiprobable data bits. The utilized channel code is the rate-1 code taken from [19]: A systematic recursive convolutional code with generator polynomials $G^{\mathsf{C}}(D) = \left(1, \frac{D+D^2+D^2}{1+D+D^2+D^3}\right)$ is punctured to rate-1 using the puncturing matrix

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 1 \end{pmatrix}$$

with $\dim \mathbf{P} = 2 \times 100$ which means that each 100th output bit is a systematic bit. The overall coding rate shall be 1/2. The utilized codes are the (n,k) multiple parity check codes with $(n_i,k_i) \in \{(10,1);(6,2);(5,2);(4,2);(3,2)\}$. The result of the optimization according to [12] are weights from which the number of bits assigned to the different codes can be determined. These are called N_{B_i} with $N_{B_i} \in \{176; 4616; 1816; 3336; 10056\}$ which means that $N_{B_1} = 176$ bits are encoded with the $(10,1)$ code and so on. The simulation results in terms of bit error rate are depicted in Fig. 3.

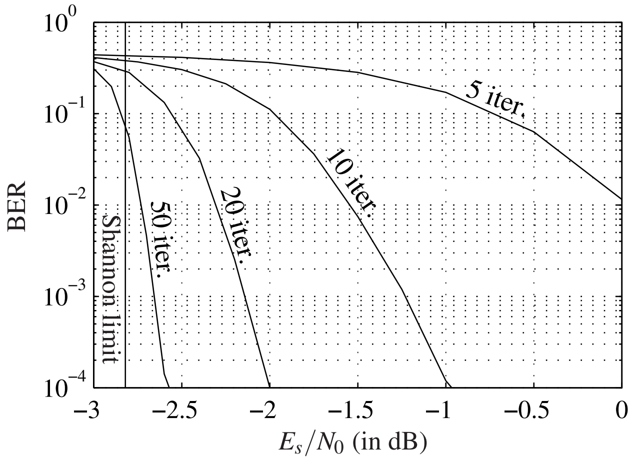

Figure 3: Bit error rate performance of the generic *FlexCode* channel coder

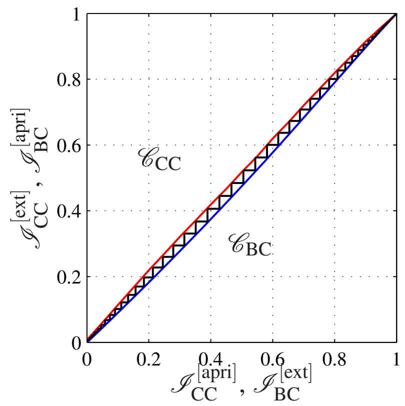

Figure 4: EXIT chart analysis (CE case) at $E_s/N_0 = -2.6\,\mathrm{dB}$

Within 50 iterations, the channel code is able to closely reach the Shannon bound. If the finite block size of only 20000 bits is considered, e.g., using the sphere packing bound [20], it can be seen that the channel code is able to reach the theoretical bound by only 0.2 dB. The EXIT chart analysis of the simulation example is depicted in Fig. 4. It can be seen that a quite narrow decoding tunnel is present.

5 Conclusion

In this paper we have presented the channel coding approach utilized in the *FlexCode* project. The proposed channel code represents a joint source-channel coding approach with iterative decoding. The two main types of parameters – model and transform coefficients – are encoded and transmitted separately. We have proposed a flexible channel encoder which can instantaneously adapt to different channel conditions and change the coding rate on the fly. Two simulation examples have been presented showing the good performance of the proposed channel coder.

6 Acknowledgments

The authors want to thank Dipl.-Ing. Thomas Schlien for providing the simulation results.

References

[1] M. Adrat, P. Vary, and J. Spittka, "Iterative Source-Channel Decoder Using Extrinsic Information from Softbit-Source Decoding," in *IEEE ICASSP*, Salt Lake City, USA, May 2001.

[2] N. Görtz, "On the Iterative Approximation of Optimal Joint Source-Channel Decoding," *IEEE J. Select. Areas Commun.*, vol. 19, no. 9, pp. 1662–1670, Sept. 2001.

[3] T. Fingscheidt and P. Vary, "Softbit Speech Decoding: A New Approach to Error Concealment," *IEEE Trans. Speech Audio Processing*, vol. 9, no. 3, pp. 240–251, Mar. 2001.

[4] FlexCode, "Deliverable D2.1: BISO Channel Model," RWTH Aachen University, Tech. Rep., July 2007, available at http://www.flexcode.eu.

[5] "ITU-T Recommendation G.192," International Telecommunication Union, Geneva, Tech. Rep., 1995.

[6] L. Larzon, M. Degermark, S. Pink, L. Jonsson, and G. Fairhurs, "The Lightweight User Datagram Protocol (UDP-Lite)," Tech. Rep., 2004.

[7] S. Bruhn, V. Grancharov, W. B. Kleijn, J. Klejsa, M. Li, H. Pobloth, and S. Ragot, "The FlexCode Source Coding Approach," in *ITG Fachtagung Sprachkommunikation*, Aachen, Germany, Oct. 2008.

[8] FlexCode, "Deliverable D1.1: Baseline Source Coder," KTH Stockholm, Tech. Rep., Feb. 2008, available at http://www.flexcode.eu.

[9] W. B. Kleijn and A. Ozerov, "Rate Distribution between Model and Signal," in *Proc. of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, Nov. 2007, pp. 243–246.

[10] D. J. Costello, Jr., J. Hagenauer, H. Imai, and S. B. Wicker, "Applications of Error-Control Coding," vol. 44, no. 6, pp. 2531–2560, Oct. 1998.

[11] M. Adrat, T. Clevorn, and L. Schmalen, "Iterative Source-Channel Decoding & Turbo-DeCodulation," in *Advances in Digital Speech Transmission*, R. Martin, U. Heute, and C. Antweiler, Eds. John Wiley & Sons, Ltd., Jan. 2008, ch. 13, pp. 365–398.

[12] M. Tüchler and J. Hagenauer, "EXIT Charts of Irregular Codes," in *Proc. of Conf. on Information Sciences and Systems (CISS)*, Princeton University, Mar. 2002.

[13] L. Schmalen, P. Vary, T. Clevorn, and B. Schotsch, "Efficient Iterative Source-Channel Decoding Using Irregular Index Assignments," in *Proc. of 8th International ITG Conference on Source and Channel Coding (SCC)*, Ulm, Germany, Jan. 2008.

[14] A. Tarable, L. Dinoi, and S. Benedetto, "Design of Prunable Interleavers for Parallel Turbo Decoder Architectures," *IEEE Commun. Lett.*, vol. 11, no. 2, Feb. 2007.

[15] 3GPP, *3GPP TS 25.222 V3.4.0 (2000-09)*, available at http://www.3gpp.org.

[16] M. Ferrari, F. Scalise, and S. Bellini, "Prunable S-Random Interleavers," in *Proc. of International Conference on Communications (ICC)*, New York City, NY, USA, Apr. 2002.

[17] FlexCode, "Deliverable D2.2: Baseline Channel Coder," RWTH Aachen University, Tech. Rep., Feb. 2008, available at http://www.flexcode.eu.

[18] S. ten Brink, "Convergence Behaviour of Iteratively Decoded Parallel Concatenated Codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[19] S. ten Brink, "A Rate One-Half Code for Approaching the Shannon Limit by 0.1dB," *IEE Electronics Letters*, vol. 36, no. 15, pp. 1293–1294, July 2000.

[20] T. Clevorn, L. Schmalen, P. Vary, and M. Adrat, "On The Optimum Performance Theoretically Attainable for Scalarly Quantized Correlated Sources," in *Proc. of ISITA*, Seoul, Korea, Nov. 2006.