Rate-Compatible Punctured Convolutional Codes (RCPC Codes) and their Applications

JOACHIM HAGENAUER, SENIOR MEMBER, IEEE

Abstract—The concept of punctured convolutional codes is extended by puncturing a low rate 1/N code periodically with period P to obtain a family of codes with rate P/(P + I) where I can be varied between 1 and (N - 1)/P. A rate-compatibility restriction on the puncturing tables ensures that all code bits of high rate codes are used by the lower rate codes. This allows transmission of incremental redundancy in ARQ/FEC schemes and continuous rate variation to change from low to high error protection within a data frame. Families of RCPC codes with rates between 8/9 and 1/4 are given for memories M from 3 to 6 (8 to 64 trellis states) together with the relevant distance spectra. These codes are almost as good as the best known general convolutional codes of the respective rates. It is shown that the same Viterbi decoder can be used for all RCPC codes of the same M by controlling the metric memory access through the puncturing rule, the soft decision and the channel state information (CSI). The performance of RCPC codes on Gaussian and fading channels is shown under various quantization and CSI conditions. The application of RCPC codes to hybrid ARQ/FEC schemes is discussed for Gaussian and Rayleigh fading channels using CSI to optimize throughput. If a source coder supplies a source significance information (SSI) indicating the relative importance of bits, a flexible unequal error protection scheme can be designed with RCPC codes which minimizes the redundancy overhead.

I. INTRODUCTION

THE design of an error correction coding system usually consists of selecting a fixed code with a certain rate and correction capability matched to the protection requirement of all the data to be transmitted and adapted to the average or worst channel conditions to be expected [1]. In many cases, however, one would like to be more flexible because the data to be transmitted have different error protection needs and the channel is time varying or has insufficiently known parameters. Consequently, flexible channel encoding and an adaptive decoder are required. As Fig. 1 shows, the information to be transmitted might carry source significance information (SSI) indicating different protection requirements. Examples include control and signaling bits in a data stream, most significant bits in a PCM word, or side information like quantizer slopes, or filter coefficients in more sophisticated voice encoding schemes [2]. On the other hand, the channel characteristics or the channel state might vary considerably, as encountered in mobile or multipath radio transmission, in a jamming environment, during rain fading, or in HF transmission. This is indicated in Fig. 1 by the channel state information CSI. In rare cases the instantaneous CSI is available at the encoder where code adaptation could take place. Mostly the receiver only can use a CSI-like fading depth, noise level variation, short-term signal loss, or jammer activity. The CSI can significantly improve decoder performance together with soft decisions at the receiver. Whenever a return channel is available, the CSI can be indirectly relayed to the transmitter by asking for a retransmission as employed in automatic repeat request (ARQ) systems. Such ARQ systems can also be combined with forward error correction (FEC) to yield type I or type II ARQ/FEC hybrid schemes [3], [4]. The latter schemes nicely match the average channel rate or the throughput with the channel conditions and some of them use different FEC codes for repeated transmission attempts. The scenarios shown in Fig. 1 require variable codes adapted to the source and channel needs. We wish to change the code rate, i.e., the number of check bits, and hence the correction power of the code during transmission of an information frame according to source and channel needs. For practical purposes, we would like to have not just switching between a set of encoders and decoders, but one encoder and one decoder which can be modified without changing their basic structure. This can be achieved by not transmitting certain code bits, namely, by puncturing the code. Mandelbaum [5] was the first to propose punctured codes for transmitting redundancy in incremental steps by using Reed–Solomon codes. In order to accommodate soft decisions and CSI at the receiver, a maximum likelihood decoder is required. This motivates the use of convolutional codes and the Viterbi algorithm for decoding.

Punctured convolutional codes were first introduced by Cain, Clark, and Geist [6] mainly for the purpose of obtaining simpler Viterbi decoding for rate K/N codes with two branches arriving at each node instead of 2^k branches. They obtained codes of rate 2/3 and 3/4 by puncturing rate 1/2 codes. These punctured codes were almost as good as the best known codes. Some of the good codes used the same basic 1/2 generators. Later, Yasuda et al. [7], [8] found a family of (N - 1)/N codes by puncturing 1/2 codes for N up to 14, and built selectable rate encoders and Viterbi decoders using soft decisions.

In this paper, the concept of punctured convolutional codes is modified for the generation of a family of codes by adding a rate-compatibility restriction to the puncturing rule. The restriction implies that all the code bits of a high rate punctured
code are used by the lower rate codes; or in other words, the high rate codes are embedded into the lower rate codes of the family. If the higher rate codes are not sufficiently powerful to decode channel errors, only supplemental bits which were previously punctured have to be transmitted in order to upgrade the code. Furthermore, since codes are compatible, rate variation within a data frame is possible to achieve unequal error protection. In Section II, the necessary definitions and the performance criterion for the class of rate compatible punctured convolutional codes (RCPC codes) are given. Section III reports the results of a computer search for families of RCPC codes of rate $P/(P + 1)$, with rates between $1/4$ and $8/9$. The analytically calculated and simulated performance of some codes on Gaussian, Rayleigh, and Rice fading channels is shown in Section IV. Modified type II ARQ/FECC schemes with RCPC schemes shown in Section V allow a minimum number of transmitted bits as well as soft decision and CSI adaptive decoding and compare favorable to previous known schemes. Unequal error protection with RCPC codes is discussed in Section VI.

II. DEFINITION AND PERFORMANCE CRITERION OF RCPC CODES

In order to explain rate-compatible punctured codes, we start with the example of Fig. 2 where a rate $R = 1/2$ convolutional code with memory $M = 2$ is punctured periodically with period $P = 4$. We describe the binary information symbols and the binary code symbols $x_j$ and $x_j$ by values $\pm 1$. A zero in the puncturing table means that the code symbol is not to be transmitted. In the upper example of Fig. 2, the fourth bit of the upper branch and the second and third bit of the lower branch are not transmitted. The puncturing table can be viewed as a modulo $P$ rule for multiplexing the two streams of code bits. Instead of transmitting $2P = 2P = P + I = P + 1 = 5$ bits are transmitted per $P = 4$ information bits. Therefore, we have generated a code with rate $R = 4/5$ by using only $P + I = 4$ of the puncturing table, which can be described by the $N \times P$ matrix

$$a(1) = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$  

Suppose the code rate with $4/5$ and with the puncturing table $a(1)$ is not powerful enough to correct the channel errors. A more redundant and therefore more powerful code with lower rates $4/6$, $4/7$, or $4/8$ would be necessary. Instead of transmitting all the code bits of a completely different low rate code, the lower rate code should utilize the bits already transmitted. Then only additional incremental redundancy bits have to be transmitted. Additional $1$s in the puncturing tables of the lower rate codes can therefore be placed only where zeroes appeared in the puncturing matrix of the previous higher rate code, for example,

$$a(2) = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}, \quad a(3) = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix},$$

$$a(4) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$  

Thus, for $l = 1$ to $4$, we get a compatible family of codes derived from the mother code $1/2$ with rates $4/5$, $4/6$, $4/7$, $4/8$ which use only incremental redundancy. The question, of course, is whether all these codes in the family are good and whether noncatastrophic codes exist. This will be discussed in Section III. On the receiving side the decoder using the Viterbi algorithm (VA) has to know the current puncturing rule $a$. The VA receives real values $y_j$ when full soft decision is used. Alternatively, the decoder can use quantized values of $y_j$. The VA finds the path with the maximum likelihood metric for statistically independent $y_j$ by calculating

$$\max_{m \in J} \sum_{j=1}^{N} \lambda_j$$

where the metric increment is

$$\lambda_j = \sum_{i=1}^{N} a_{ij}x_j^{(m)}y_{ij}$$

and $a_{ij} = a_{ij}$ due to periodic puncturing. $a_{ij} = 0$ means that this $x_j^{(m)}$ has not been transmitted and $y_j$ is not available. Actually, it is not necessary to insert any dummy data at the receiver as suggested in [7]. The Viterbi algorithm operates on the trellis of the $1/N$ mother code with two branches leaving at each state. Equation (2) simply means that for $a_{ij} = 1$ the metric ROM is accessed using the address $x_j^{(m)}y_{ij}$ to add a metric increment and for $a_{ij} = 0$ it is not accessed.

For a fading channel, the transmitted signal is multiplied by a fading factor $a_p$. If interleaving is assumed, $a_p$ is an independent positive real random variable with density $p(a_p)$, i.e., Rayleigh or Rice. If this factor can be estimated at the receiver and is used as a CSI the ML-metric for coherent transmission is [9]

$$\lambda_j = \sum_{i=1}^{N} a_{ij}a_p x_j^{(m)}y_{ij}.$$  

As expected, a deep fade ($a_p \to 0$) has the same effect as puncturing ($a_{ij} = 0$).

A. General Definition of RCPC Codes

A family of RCPC codes is described by the mother code of rate $R = 1/N$ and memory $M$ having the generator tap matrix

$$g = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

with the tap connections $g_{ik} \in (0, 1)$ where a 1 represents a connection from the $k$th shift register stage to the $i$th output. Together with $N$, the puncturing period $P$ determines the range of code rates

$$R = \frac{P}{P + 1}$$

between $P/(P + 1)$ and $1/N$. The RCPC codes are punctured codes of the mother code with puncturing matrices

$$a(1) = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

with $a_{ij}(l) \in (0, 1)$ where 0 implies puncturing.
The rate-compatibility restriction implies the following rule:

\[ a_i(l) = 1 \text{ then } a_i(l) = 1 \text{ for all } l \leq l_0 \leq 1 \]  
(7a)

or equivalently

\[ a_i(l) = 0 \text{ then } a_i(l) = 0 \text{ for all } l \leq l_0 \leq (N-1)P - 1. \]  
(7b)

B. Criterion of Goodness for RCPC Codes

With Viterbi decoding the usual optimality criterion is a large free distance \( d_{\text{free}} \), a small number of paths \( a_d \), and a small information error weight \( c_d \) on all paths with \( d \geq d_{\text{free}} \). More specifically, one has Viterbi’s upper bound an error event probability [13]

\[ P_E \leq \frac{1}{P} \sum_{d=d_{\text{free}}}^{\infty} a_d P_d \]  
(8)

and for the bit error probability

\[ P_b \leq \frac{1}{P} \sum_{d=d_{\text{free}}}^{\infty} c_d P_d. \]  
(9)

\( P_d \) is the probability that the wrong path at distance \( d \) is selected. The so-called distance spectra \( \{a_d\} \) and \( \{c_d\} \) which should be as small as possible, depend only on the code, namely, on \( N, M, P, g, \) and \( a(l) \). Due to the time-varying nature of the RCPC codes, \( P \) different starting points for diverging paths have to be considered to obtain the total numbers \( a_d \) and \( c_d \). Since RCPC codes are a subclass of punctured codes, which in turn is a subclass of the general \( K/N \) codes, it has still to be shown that the RCPC restriction is not too severe to prevent “good” codes from being located in this subclass. A degree of freedom can be gained by increasing the period \( P \). RCPC codes constitute a specific class of time-varying codes with a fixed generator but periodically time-varying puncturing. We know that some time-varying codes are better than fixed codes [10] and some random coding theorems hold only for time-varying codes. Therefore, some improvement may be possible by using a higher period \( P \). Note that \( P \) has to be sufficiently large if codes of specific rates are to occur in the same RCPC family. The complexity of the VA decoder of an RCPC code in terms of add, compare and select operations per decoded bit is \( 2^{-P} K \), whereas a general \( K/N \) code requires \( 2^{P/2}\ln(K) \) such operations where \( M \) is the total encoder memory. At the RCPC decoder, an \( N/P \) ambiguity has to be resolved in the incoming data stream and the decision depths of punctured codes are generally longer [8]. The last disadvantage is not severe, when as in most applications, data frames are transmitted with sync words and proper termination of short frames.

III. RESULTS OF THE SEARCH FOR GOOD RCPC CODES

No constructive method is known for determining the generator matrix \( g \) and the puncturing matrices \( a(l) \) for a RCPC code family. Therefore a computer search has been performed under some restrictions. For distances \( d \geq d_{\text{free}} \), \( a_d \), \( c_d \), and the path length distribution has been determined by a stack algorithm similar to Cedervall’s algorithm [11].

The basic steps to find a family of good RCPC codes are:

Step 1: Select mother code with rate \( 1/N, \) memory \( M, \) and generator \( g \). Select puncturing period \( P \). For \( l = l_{\text{max}} = (N - 1)P \) set \( a(l) = (1). \)

For \( l = (N - 1)P \) to 2 perform steps 2-4.

Step 2: Amongst the elements \( a_i(l) \) which are “1” select one and set it to “0” [compatibility rule (7b)]. Observe that cyclic column permutations give the same code performance.

Step 3: Determine \( d_{\text{free}} \) and \( c_d \) for \( d \geq d_{\text{free}} \).

Step 4: Repeat steps 2 and 3 to find \( a(l - 1) \) with maximal \( d_{\text{free}} \) and minimal \( c_d \) for \( d = d_{\text{free}} \) and \( d_{\text{free}} + 1 \). If only catastrophic codes can be found, use next best code at level \( l \).

The results of the computer search for the \( M = 4 \) code family are given in some detail in Table I, whereas for the other codes of memory \( M = 3, 4, 5, \) only \( a(l) \) and the \( c_d \) values are given in Table II. The codes have a period \( P = 8 \) with a subperiod of \( P = 4 \) and are punctured mostly in steps \( l = 2 \) in such a way as to arrive at the best known code at rate 1/2. This implies that the mother codes of rate 1/3 or 1/4 are not necessarily the best known code of this rate. This is the case for \( M = 4 \) and 6 where one path exists at a distance which is one less than the best known free distance [3].

Table III shows a comparison of the most important distance parameters of RCPC codes, punctured codes, and general \( K/N \) codes for rates where results are available. This and other comparisons show that RCPC codes are almost as good as the other codes. By increasing the period \( P \) one can even obtain slightly better codes as shown for \( M = 3 \) and rate 2/3 code, \( P = 4 \) gives a greater variety in selecting the puncturing positions. In this case, \( c_{d'/P} \) for \( d' = d_{\text{free}}, \ldots, d_{\text{free}} + 5 \) is less than the corresponding numbers for the code with \( P = 2 \) and therefore a better BER performance can be expected over the whole SNR range. This result is a further confirmation that time-varying codes are better than fixed codes and that time-varying codes become better with increasing time period. In [10], an example is given where \( g \) is time varying. Here, we fix \( g \) and puncture in a time-varying manner.

One disadvantage of punctured codes is that error events at high rates and higher distances can be quite long. This was previously observed in [7]. To give an example, we have plotted in Fig. 3 the path length distribution of an \( R = 4/5 \) code with \( M = 4, \) \( P = 4 \). Paths at \( d_{\text{free}} + 3 = 6 \) can be up to 50 information bits long. For high rate codes, this requires a longer path memory.

IV. PERFORMANCE OF RCPC CODES ON GAUSSIAN AND FADING CHANNELS

The performance of RCPC codes on a Gaussian channel with soft decision and on interleaved Rayleigh and Rician channels using different degrees of channel state information has been evaluated using (9) and the distance spectrum \( \{c_d\} \) in Tables I and II. For the Gaussian case simulation results are also given. Fig. 4 shows the transmission scheme over a nonfrequency-selective fading channel with multiplicative distortion \( \delta_f \), with \( E\{x_i^2\} = 1 \) and

\[ \rho_0(\delta f) = 2a_0(1 + C/M) \text{ exp } [- \left( \alpha_x^2(1 + C/M) + C/M \right)] \]

\[ \cdot \ln(2\alpha_x\sqrt{C/M})(1 + C/M). \]

(10)

For a Rician channel, \( C/M \) is the ratio of the direct to diffusely reflected signal energy. With \( C/M = 0 \) we have Rayleigh fading, with \( C/M \to \infty \) we obtain the Gaussian channel. In the fading case we assume perfect interleaving, which means that the CSI values \( a_d \) and the received code values \( y \) are statistically independent with density function (10). Since we consider a varying code rate, we describe the channel SNR by \( E_s/N_0 \) rather than by \( E_s/N_0 = E_s/N_0 \). A reduced rate consequently means reduced throughput. The Viterbi algorithm for decoding the RCPC code uses the metric (3). In the following, we give the values for \( \delta f = \delta_f \) and \( y = y_d \) which the decoder uses in different cases, as well as the resulting \( P_b \) to be used in the evaluation of the BER in (9).

a) Gaussian channel [13]

\[ \delta_f = 1 \quad y \text{ analog or finely quantized} \]

\[ \frac{1}{2} \frac{1}{\text{erfc} \sqrt{dE_s/N_0}} \]

b) Rayleigh fading [10]

\[ \delta_f = 1 \quad y \text{ analog or finely quantized} \]

\[ \frac{1}{2} \frac{1}{\text{erfc} \sqrt{dE_s/N_0}} \]

IV. PERFORMANCE OF RCPC CODES ON GAUSSIAN AND FADING CHANNELS

The performance of RCPC codes on a Gaussian channel with soft decision and on interleaved Rayleigh and Rician channels using different degrees of channel state information has been evaluated using (9) and the distance spectrum \( \{c_d\} \) in Tables I and II. For the Gaussian case simulation results are also given. Fig. 4 shows the transmission scheme over a nonfrequency-selective fading channel with multiplicative distortion \( \delta_f \), with \( E\{x_i^2\} = 1 \) and

\[ \rho_0(\delta f) = 2a_0(1 + C/M) \text{ exp } [- \left( \alpha_x^2(1 + C/M) + C/M \right)] \]

\[ \cdot \ln(2\alpha_x\sqrt{C/M})(1 + C/M). \]

(10)

For a Rician channel, \( C/M \) is the ratio of the direct to diffusely reflected signal energy. With \( C/M = 0 \) we have Rayleigh fading, with \( C/M \to \infty \) we obtain the Gaussian channel. In the fading case we assume perfect interleaving, which means that the CSI values \( a_d \) and the received code values \( y \) are statistically independent with density function (10). Since we consider a varying code rate, we describe the channel SNR by \( E_s/N_0 \) rather than by \( E_s/N_0 = E_s/N_0 \). A reduced rate consequently means reduced throughput. The Viterbi algorithm for decoding the RCPC code uses the metric (3). In the following, we give the values for \( \delta f = \delta_f \) and \( y = y_d \) which the decoder uses in different cases, as well as the resulting \( P_b \) to be used in the evaluation of the BER in (9).

a) Gaussian channel [13]

\[ \delta_f = 1 \quad y \text{ analog or finely quantized} \]

\[ \frac{1}{2} \frac{1}{\text{erfc} \sqrt{dE_s/N_0}} \]
### Table I
**Puncturing Tables \( a(l) \), \( c_d \), and \( a_d \) Values for RCPC Codes with Memory \( M = 4 \). Period \( P = 8 \) and Rates \( R = P/(P + l) = 8/(8 + l) \), \( l = 1, 2, 4, 6, \cdots, 24 \)**

<table>
<thead>
<tr>
<th>( l )</th>
<th>( a(l) )</th>
<th>( c_d )</th>
<th>( a_d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b) Rayleigh and Rice fading with hard decisions on \( y \) (YH) and no CSI, \( a \) not used (AN): YHAN

\[ a_F = 1, \quad y = \pm 1. \]

\[
P_d = \begin{cases} 
\sum_{(d+1)/2}^{d} \binom{d}{e} P_0^e (1 - P_0)^{d-e} & d \text{ odd} \\
P_{d-1} & d \text{ even}
\end{cases}
\]

with the channel bit error rate

\[
P_0 = \frac{1}{2} \text{erfc} \sqrt{a_F^2} \cdot E_s/N_0 P_{er}(a_F) d_{er}.
\]

\[ P_d \leq \frac{1}{2} \left( \frac{1}{1 + E_s/N_0} \right)^d. \]

c) Rayleigh fading with soft decisions on \( y \) (YS) and full CSI (AS) with perfect estimation of the analogy amplitude \( a_F \): YSAS

\[ a_F = a_F \quad y \text{ analog or finely quantized} \]

\[ P_d \] is equivalent to the error probability of a \( d \)th order maximum ratio diversity system and can be upper bounded by [9]

\[
\]

d) Rayleigh fading with hard decision (YH) and hard CSI
### TABLE II
PUNCTURING TABLES \( a(i) \) AND \( c_d \) VALUES FOR RCPC CODES WITH MEMORY \( M = 3, 5, 6 \). PERIOD \( P = 8 \) AND RATES \( R = P/(P + 1) = 8/(8 + 1) \), \( I = 1, 2, 4, 6, \ldots \), 14, 16

#### RCPC CODE: \( M = 3 \)

<table>
<thead>
<tr>
<th>( 8/9 )</th>
<th>( 8/10 )</th>
<th>( 8/12 )</th>
<th>( 8/14 )</th>
<th>( 8/16 )</th>
<th>( 8/18 )</th>
<th>( 8/20 )</th>
<th>( 8/22 )</th>
<th>( 8/24 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
</tr>
<tr>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
<td>0001</td>
</tr>
<tr>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
</tr>
<tr>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
<td>1101</td>
</tr>
</tbody>
</table>

#### RCPC CODE: \( M = 5 \)

<table>
<thead>
<tr>
<th>( 8/9 )</th>
<th>( 8/10 )</th>
<th>( 8/12 )</th>
<th>( 8/14 )</th>
<th>( 8/16 )</th>
<th>( 8/18 )</th>
<th>( 8/20 )</th>
<th>( 8/22 )</th>
<th>( 8/24 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
<td>0000</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
</tbody>
</table>

(a) [Table for M = 3]

(b) [Table for M = 5]
### TABLE II (Continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>8/9</th>
<th>8/10</th>
<th>8/12</th>
<th>8/14</th>
<th>8/16</th>
<th>8/18</th>
<th>8/20</th>
<th>8/22</th>
<th>8/24</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>74</td>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>13327</td>
<td>376</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>21776</td>
<td>3464</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3315491</td>
<td>32512</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4827817</td>
<td>247234</td>
<td>1140</td>
<td>74</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>18907900</td>
<td>5104</td>
<td>385</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>24640</td>
<td>1162</td>
<td>388</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>109912</td>
<td>3542</td>
<td>0</td>
<td>82</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>11666</td>
<td>1688</td>
<td>182</td>
<td>44</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>320</td>
<td>80</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>11232</td>
<td>810</td>
<td>138</td>
<td>70</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>1882</td>
<td>444</td>
<td>90</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>750</td>
<td>172</td>
<td>160</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td></td>
<td>474</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td>424</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TABLE III

**Comparison of RCPC Codes with Punctured and General Convolutional Codes**

<table>
<thead>
<tr>
<th>Rate Memory</th>
<th>RCPC- Codes</th>
<th>Punctured Codes [8]</th>
<th>General K/N Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$d_{free}$</td>
<td>$d_{free}$</td>
<td>$P$</td>
</tr>
<tr>
<td>M = 3</td>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2/3</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>M = 6</td>
<td>6</td>
<td>6</td>
<td>1.5</td>
</tr>
<tr>
<td>M = 3</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4/5</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>M = 6</td>
<td>6</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>M = 3</td>
<td>2</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>8/9</td>
<td>6</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>M = 6</td>
<td>3</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>M = 3</td>
<td>4</td>
<td>0.5</td>
<td>4</td>
</tr>
<tr>
<td>4/7</td>
<td>5</td>
<td>6</td>
<td>1.25</td>
</tr>
</tbody>
</table>

*With memory to give an equal number of total branches per information bit in the VA.*
Fig. 3. Number of paths with length \( l_p \) at distance \( d \) for a RCPC code of rate \( R = 4/5, P = 4, M = 4 \).

Fig. 4. Coded data transmission using RCPC codes with interleaving and channel state information (CSI) on fading channels.

(AH): YHAH

\[
\hat{\delta}_F = \begin{cases} 
\log \frac{1-P_B}{P_B} & \text{for } \alpha_F < \alpha_T \\
\log \frac{1-P_G}{P_G} & \text{for } \alpha_F \geq \alpha_T
\end{cases} \tag{14}
\]

where \( \alpha_T \) is an optimized threshold. By partial integration in (12), we obtain \( P_G \) as the channel BER above the threshold \( \alpha_T \) and \( P_B \) as the BER below \( \alpha_T \). \( P_F \) can be calculated in closed form. The result is given in [9] and [14].

Fig. 5 shows the simulated RCPC performance for the Gaussian channel. Simulation and an analytical upper bound at a BER of \( 10^{-5} \) differ only by approximately 0.2 dB. The performance curves for the Rayleigh channel with the scheme described in Fig. 4 and in Section IV show that RCPC codes with rates between 4/5 and 1/3 cover a broad range of SNR and BER. At a BER of \( 10^{-5} \) a 1-bit CSI gains 7.9 dB at rate 2/3 and 2.6 dB at rate 1/3 relative to hard decisions without CSI. The Rice channel with \( C/M = 7 \) dB is typical for mobile satellite communications to ships with small antennas, to cars in open areas, and to airplanes with hemispherical antennas [14]. As Fig. 7 shows, the RCPC codes using basically the same encoder and the same Viterbi decoder with only 16 states and two branches per state can cover a SNR between 4 and 17 dB to achieve a BER of \( 10^{-5} \). This means that the same link can serve users with a gain variation of 13 dB by adaptively changing the rate between 4/5 and 1/3 at the expense of throughput.
This method goes back to Mandelbaum [5] and includes variations like code combining [15], [16] and memory ARQ [17].

We propose to use rate compatible punctured convolutional codes (RCPC Codes) with Viterbi algorithm (VA) decoding for successive parity transmission. The principle of the proposed ARQ/FEPCR code scheme is not to repeat information or parity bits if the transmission is unsuccessful as in previous type II ARQ/FEPCR schemes, but to transmit additional code bits of a lower rate RCPC code until the code is powerful enough to enable decoding. This includes several decoding attempts on the receive side, which seem especially feasible for applications in mobile or low rate systems where the data rate is in the kbit/s range and the VA decoder would run on a chip at a higher rate.

A. ARQ/FEPCR Protocol with RCPC Codes

The ARQ/FEPCR protocol performs the following steps while transmitting a block of information, which contains typically \( n = 100 \) to 1000 bits:

1. Encoding:
   1. Add \( n_c \) parity bits to form an error detection code \( C_0 \), i.e., 16 or more CRC bits.
   2. Add \( M \) “0” or known bits to properly terminate the encoder memory and the decoder trellis for the “information” block \( n + n_c \). \( M \) is the memory of the convolutional encoder.
   3. Use the \( 1/N \) encoder to encode the convolutional code and store it at the transmitter, possibly in a matrix as in Fig. 8. The matrix is only conceptual and can be modified in several ways to accommodate interleaving.

2. Transmitting and decoding:
   1. Transmit the first \( P \) columns of the matrix.
   2. Transmit additional columns of the matrix up to index \( l \) according to the puncturing table.
   3. Decode RCPC code of rate \( P(\ P + 1) \) with Viterbi algorithm.

4. Check the syndrome of code \( C_0 \). If the syndrome is zero output \( l \) and send ACK to the transmitter. If the syndrome of \( C_0 \) is not zero increase \( l \) and repeat steps (2)-(4).

If decoding is still not successful for \( l = (N - 1)P \), i.e., \( R = 1/N \), several possibilities exist. A higher order protocol could take over or ask for a repetition of the whole procedure starting at step 1). Code combining as suggested by Chase [15] could be used to repeat the whole \( 1/N \) code \( L_r \) times and combine it with the already transmitted code. Actually, the VA uses in this case the same trellis as for all the RCPC codes, but the ML metric (3) is changed to

\[
\lambda_{ij} = \sum_{r=1}^{L_r} \sum_{i=1}^{N} a_{ij} a_{ijr} x_i y_{jr} = \sum_{i=1}^{N} a_{ij} x_i \sum_{r=1}^{L_r} a_{ijr} y_{jr} \tag{15}
\]

where index \( r \) indicates the \( r \)th repeated transmission. Note that by repeating all code bits \( L_r \) times the free distance is increased by a factor of \( L_r \) [15] and the numbers of the weight spectrum are unchanged

\[
c_{d} = c_{d} \quad d_{s} = L_r \cdot d \quad d = d_{max} + l.
\]

Note that a repetition code is almost optimal only for low rates such as 1/8, 1/16 [15]. By halving the code rate the distance usually is more than twice as shown in Table I for memory \( M = 4 \). This enables a gain in \( E_s/N_0 \) on the Gaussian channel. This type of code combining as expressed in (15) can be viewed as maximum ratio combining of the received \( y_{w} \) values. The combined \( y \) values which equal the inner sum of the right side in (15) are then used by the original code for ML decoding.

In the following analysis, we will assume that decoding stops with the \( 1/N \) code in step 4). Consequently, there exists
a probability that the frame with information block I was not error-free decoded. This frame error rate FER will be calculated and should be sufficiently small. It should be pointed out that we are completely free in which steps we increase l. The number of steps depends on the ratio of frame arriving time (information rate) to frame decoding time (decoder speed). For extremely varying channel conditions an exponential increase could be useful, i.e., l = 1, 2, 4, 8, 16. This would mean to perform less FEC decoding attempts than the maximum number possible. This is at the cost of a slightly smaller throughput because in the example, the 10th attempt could have been successful, saving the transmission 11-16 in the above example.

With this type of protocol an acknowledgement (ACK) is only sent after successful decoding of a frame. The encoder would send more and more code bits until it receives the ACK. Assume that decoding is successful at step \( I_k \). If there is a round-trip delay \( T_D \) involved in the transmission with rate \( 1/T_S \), the encoder will transmit \( T_D/T_S \) unnecessary bits, which would slightly reduce the throughput. This will be shown in the analysis. If the ACK is lost, the encoder keeps transmitting unnecessary code bits until \( l = (N - 1)P \). If the probability of an ACK loss or error is small the effect on the throughput is negligible. Thus, the return channel is used only once per frame and a bad return channel does not confuse the basic protocol.

### B. Throughput Analysis

We will describe the performance of the hybrid ARQ/FEC scheme by two figures of merit: the average throughput \( R_{AV} \) and the probability that a frame or packet with \( n \) information bits is finally not correctly decoded, the so-called frame error rate FER. The throughput is

\[
R_{AV} = \frac{n}{n + n_c + M} \cdot \frac{P}{P + I_{AV}}.
\]

\( I_{AV} \) is the average number of additionally transmitted bits per \( P \) information bits. From Fig. 8 this can be interpreted as the average number of additionally transmitted columns. \( P/(P + I_{AV}) \) is the effective code rate of the FEC code. Due to the overhead of \( n_c \) parity check bits and \( M \) terminating bits with \( n_c + M < n \), the throughput is slightly smaller than the effective FEC code rate. We assume that the error detecting code \( C_0 \) achieves a very low probability of undetected errors which will be ignored in the subsequent analysis. If the \( (n + n_c, n) \) error detection code \( C_0 \) falls into the class which satisfies \( \sum_{d=2}^{\min(n-n_c-1, 2)} a_d < 2^{-n_c} \)

Let \( P_{EF}(I_k) \) be the probability that the FEC decoding at the \( k \)th step results in errors which are detected by \( C_0 \). We will give an approximate analysis where we assume that the \( K \) decoding attempts have statistically independent outcomes. Then we have

\[
l_{AV} = \sum_{k=1}^{K} l_k (1 - P_{EF}(I_k)) \cdot \frac{1}{P} \prod_{i=0}^{K} P_{EF}(I_i) + l_k \prod_{i=0}^{K} P_{EF}(I_i) \tag{17}
\]

where we define \( P_{EF}(I_0) = 1 \).

The frame error rate of the unsuccessful ARQ/FEC procedure is then

\[
FERR = P_{EF}(I_K).
\]

We can include in the analysis a delay \( T_D \) caused by the round-trip time and the decoding time by replacing \( l_k \) in (17) by

\[
l_k := \min \{ l_k, l_k + PT_D/(T_S(n + n_c + M)) \}.
\]

Similarly, a loss or error in the ACK signal on the return channel, causing unnecessary transmission up to \( l_k \), can be accommodated. For the probability \( P_{EF}(I_k) \), we only can give an upper bound by using (9) to calculate the probability that an error event occurs for the transmitted bits

\[
P_{EF}(I_k) < 1 - \left( 1 - \frac{1}{P} \sum_{d=2}^{\min(n-n_c-1, 2)} a_d(l_k) \cdot P_{H} \right)^{n_c + n_c + M} \tag{20}
\]

The values \( a_d \) depend on the code, \( P_H \) depends on the channel and the detection scheme and can be selected from the values given in Section IV and in (19). Recall that \( P_H \) usually assumes sufficient interleaving. By using (20) in (17) and (16), we obtain a lower bound for the throughput \( R_{AV} \) and by using (20) in (19), we get an upper bound on the FER.

Two examples will be given to show the performance of the RCPC codes in the hybrid ARQ/FEC scheme. The same code with \( M = 4, N = 3, P = 8 \) will be used, which has fairly low decoding complexity with 16 states and 32 add, compare and select operations per decoded bit. Fig. 9 shows the throughput on a Gaussian channel with hard and soft decisions as calculated by the lower bound method described. The simulations show that for this case the bound is accurate within 1 dB. The benefits of soft decisions are clearly observable. Due to the overhead \( R_{AV} \) approaches 0.97 for high SNR. At low SNR, \( R_{AV} \) approaches 0.323 and the frame error rate is high. The RCPC scheme which uses adaptive code rates between 1 and 1/3 has generally a better throughput than comparable schemes described in [4] and its throughput curve constitutes a kind of envelope to the corresponding fixed rate schemes. In Fig. 10 the performance on a fully interleaved Rayleigh channel is given. In this case, decoding starts with \( l = 1, \) i.e., the 8/9 code and ends with the 1/3 code. Several degrees of CSI are given as described in Section IV: YHAN, YHAH, and YSAS. Note that for a Rayleigh channel with \( E_b/N_0 = 10 \) dB a frame error rate of less than \( 10^{-6} \) and a throughput of 0.7 can be easily achieved. Simulation results and analytic bounds agree within 1 dB.

### VI. THE APPLICATION OF RCPC CODES TO UNEQUAL ERROR PROTECTION

In many applications it is required to provide different levels of error protection for different parts of an information sequence or block. Some examples have been given in the introduction. We assume that the source or the source coder supplies us with some information about the relative impor-
Fig. 9. ARQ/FEC throughput with RCP codes on a Gaussian channel with soft and hard decisions (BSC). Code: $P = 8, N = 3, l = 0, 1, 2, 4,\ldots, 16, M = 4$. Frame: $n = 8 \times 125, n_r = 28$.

Fig. 10. ARQ/FEC throughput and frame error rate FER on an interleaved Rayleigh channel, hard decision, and CSI. Code: $P = 8, N = 3, l = 1, 2, 4, \ldots, 16, M = 4$. Frame: $n = 8 \times 48, n_r = 34$.

Fig. 11. Data frame with information bits grouped according to their error protection requirement (source significance information $P_{MB}$). RCP codes assigned to the groups.

The rate-compatibility condition (7a) is crucial. In a transitional phase between two matrices $a(l_i)$ and $a(l_{i+1})$ where $l_{i+1} > l_i$ we have to guarantee, that despite the transition, the distance properties of all paths originating in code $l_i$ do not suffer a loss of distance due to transitions, thus guaranteeing at least the designed performance. Fig. 12 shows the same error paths relative to the correct all-zero path in RCP-code $l_i$: path 1 having distance $d_1$ is not affected by a transition, whereas path 2 extends into the lower rate code $l_{i+1}$. The rate-compatibility condition (7a) guarantees that the $l_{i+1}$ code does not puncture any "1" on the path. Due to less puncturing it might happen that additional "1"'s occur on the part $B$ of the path, which would cause a $d_{B} > d_{1}$. Therefore, some of the $a_{D}$ and $c_{D}$ values have to be assigned to higher distances if the path is not terminated at the transition. Thus, the rate compatibility restriction guarantees that transitional paths do have a distance which is at least the distance of the same path within the higher rate code and at most the distance of the same path within the lower rate code. If two punctured codes without the rate compatibility restriction are used at a rate transition, it can happen that a transitional path has a distance which is even lower than the distance of the same path.
within the higher rate code. This would lead to a bad error behavior in the transition region. One can easily construct such an example.

It is useful to order the information as described with increasing importance. Decoding errors typically occur at the beginning of paths and the last $M$ information bits of each error path are error free. However, the transitional paths (path 2) to lower rates generally have higher distances and therefore errors occur with lower probability. Simulations show that this type of ordering leads to a smaller number of total errors. Error paths can extend over many transitions. All paths will be properly terminated by the tail forcing the decoder to the all-zero state. Ordering the information bits from lower to higher rate codes would require proper zero termination at each change of rate. We conclude this section by giving an example of the simulated performance of the RCPC code with unequal error protection within one data frame of 128 bits. As Fig. 13 shows the code rate is changing from $4/5$ to $4/8$ while the measured short term BER is improving by a factor of 100. As discussed above, the bits close to a transition to the next lower rate get a better protection.

VII. CONCLUSIONS

We have described the concept of forward error correction codes which are obtained from low rate convolutional codes by puncturing them periodically in a rate-compatible manner. Surprisingly, this restriction still produced good codes relative to the best known codes of comparable complexity. In such a way, a whole family of codes with different rates is available using the same encoder and the same ML-decoder employing the Viterbi algorithm. Only the puncturing rule, i.e., the multiplexer switching pattern is changed. Since in most transmission schemes, channel rate is fixed due to modulation and channel filter requirements, changing the code rate means a change of information rate and a buffer for the incoming information stream is required. One disadvantage of punctured codes with high rate is that error paths can be quite long, as shown in Fig. 3 and in [7]. This requires either long path memories in the VA, framed transmission as employed in most packet-oriented transmission schemes, or periodically inserted synchronization patterns or known data bits. In a decoder implementation, the basic Viterbi algorithm runs on the same trellis for all the codes. This suggests a VLSI for the Viterbi-decoding algorithm where only the control and the memory access is changed according to the rate, the channel conditions, and the channel state information (CSI).

Since the compatibility condition ensures that only incremental redundancy is used for lower rate codes, RCPC codes are suitable for ARQ/FEC protocols and unequal error protection as shown in Sections V and VI. Of course, both methods can be combined to adapt unequal error protection to varying channel conditions. For instance, the transmission security of a frame with a set of unequal error protection codes using rates $(1, 8/10, 8/14)$ can be upgraded to a set of $(8/10, 8/12, 8/16) codes when the channel conditions get worse by transmitting another two check bits per 8 information bits.

The concept of RCPC codes can be combined with the concept of nested codes [25] where the Viterbi decoder can work also with $M - 1, M - 2, \ldots$ on a reduced set of states. Codes have been found which satisfy both the RCPC and the nested condition. Furthermore, the concept of RCPC codes could also be used to find a family of “good” codes with high memory suitable for sequential decoding [3]. “Good” would mean that all the codes with different rates need a rapidly growing distance profile [3]. In an ARQ/FEC application, the error detection via the $C_0$ code could be replaced by a time-out condition in the Fano or stack algorithm.

The design of combined source and channel coding as described for voice transmission in [23] and for image transmission in [24] gets a new degree of freedom by using RCPC codes because the code rate can be changed within a data frame and during transmission according to the needs [27]. In other words, source significance information SSI, channel state information CSI, and check information from the decoded bits can be used for the adaptation of the encoding and ML-decoding algorithm of RCPC codes.

ACKNOWLEDGMENT

The author would like to thank H. Osthoff for writing the computer search program used in Section III as part of his Dipl. Ing. thesis at the Technical University München.

REFERENCES


Joachim Hagenauer (M’78–SM’87) received the Ing.(grad) degree from Ohm-Polytechnic Nürnberg, in 1963, and the Dipl.-Ing. and the Dr.-Ing. degrees in electrical engineering from Technical University Darmstadt, in 1968 and 1974, respectively.

At Darmstadt University he served as an Assistant Professor and Dozent. From May 1975 to September 1976 he held a postdoctoral fellowship at the IBM T. J. Watson Research Center, Yorktown Heights, NY, working on error correction coding for magnetic recording. Since 1977 he has been with the German Aerospace Research (DFVLR) at Oberpfaffenhofen—since May 1980 as the Head of the Communication Theory Group. His research interests include convolutional coding, data transmission via fading channels, and mobile communications. During 1986/87 he spent a sabbatical year as an “Otto Lillienthal Fellow” at AT&T Bell Laboratories, Crawford Hill, NJ working on joint source/channel coding and on trellis coded modulation.