# A Viterbi Algorithm with Soft–Decision Outputs and its Applications

Joachim Hagenauer
Peter Hoeher

German Aerospace Research Establishment (DLR)
Institute for Communications Technology
D-8031 Oberpfaffenhofen, West-Germany
Tel. ++49/8153/28-893, Fax ++49/8153/28-243

## Abstract

The Viterbi algorithm (VA) is modified to deliver not only the most likely path sequence in a finite–state markov chain, but either the a posteriori probability for each bit or a reliability value. With this reliability indicator the modified VA produces soft decisions to be used in decoding of outer codes. The inner *Soft–Output Viterbi Algorithm* (SOVA) accepts *and delivers* soft sample values and can be regarded as a device for improving the SNR, similar to an FM demodulator. Several applications are investigated to show the gain over the conventional hard–deciding VA, including concatenated convolutional codes, concatenation of convolutional and simple block codes, concatenation of Trellis–Coded Modulation with convolutional FEC codes, and coded Viterbi equalization. For these applications we found additional gains of $1-4$ $dB$ as compared to the classical hard–deciding algorithms. For comparison, we also investigated the more complex symbol–by–symbol MAP algorithm whose optimal a posteriori probabilities can be transformed into soft outputs.

## 1 Introduction

The Viterbi algorithm [1] has become a standard tool in communication receivers, performing such functions as demodulation, decoding, equalization, etc. An increasing number of applications use two VA in a concatenated way. Examples are coded modulation systems without bandwidth expansion, such as coded quadrature amplidude modulation (QAM) [2] and continuous phase modulation (CPM) [3]. In these systems Viterbi receivers replace classical modulation schemes. An additional outer coding system could use convolutional codes with Viterbi decoding to perform forward error–correction (FEC) decoding.

There are normally two drawbacks with such a solution: first, the inner VA for demodulation produces bursts of errors against which the outer VA is very sensitive; second, the inner VA produces hard decisions prohibiting the outer VA from using its capability to accept soft decisions. The first drawback can be eliminated by using some interleaving between the inner and outer VA. To eliminate the second drawback, the inner VA needs to output soft-decisions, i.e., reliability information. This should improve the performance of the outer VA considerably.

Another important situation where a similar problem arises is when convolutional codes for FEC are used on channels requiring equalization. This is the case in the future pan–european mobile radio system (GSM) [4]. The normal Viterbi equalizer produces only hard decisions, leading to a reduced performance of the outer VA performing the FEC.

The performance of the above mentioned systems, and other systems such as multistage Viterbi schemes, FEC/ARQ schemes etc., will improve by using a *Soft–Output Viterbi Algorithm* (SOVA). This is a VA which uses soft (or hard) decisions to calculate its metrics, but also decides in a soft way by providing reliability information together with the output bits. The reliability information can be the log–likelihood function.

We wish to modify the VA as little as possible. The goal is to add to the Viterbi receiver a soft–deciding unit. In earlier work Forney considered "augmented outputs" from the VA itself [1], such as the depth at which all paths are merged, the difference in length between the best two paths, and a list of the $p$ best paths. The last topic was generalized in [5]. Ya-



CH2682-3/89/0000-1680 $1.00 

mamoto *et al.* derived a simple indicator for block errors by introducing labels [6]. This scheme is restricted to requesting a retransmission of the whole block. Schaub *et al.* [7] took up the ideas of Forney by declaring an erasure output on those bits to be decided, where the metric difference at the point of merging never exceeds a threshold.

# 2 The Soft–Output Symbol–by–Symbol MAP and Viterbi Algorithm (SOVA)

## 2.1 Detection with Reliablity Information

We assume that in the receiver chain a Viterbi detector is used. This could be a Viterbi equalizer, a Viterbi demodulator (i.e. for CPM or Trellis–Coded QAM), or the Viterbi decoder for an inner convolutional code. This device is followed by a second stage detector, which could be a demodulator or decoder after the equalizer, a decoder after the demodulator, an outer decoder after the inner decoder, or a source decoder. We assume that this second device can improve its performance if some reliability information is available in addition to the hard decisions from the first stage.

A straightforward way is shown in Fig. 1. The first stage detector delivers estimates $\hat{\mathbf{u}}'$ of the symbol sequence $\mathbf{u}'$ by processing the received sequence $\mathbf{y}$ in the Viterbi detector. We want the detector to deliver further for each symbol an estimate of the probability that this symbol has been incorrectly detected

$$p'_k = \text{Prob}\{\hat{u}'_k \neq u'_k \mid \mathbf{y}\}. \tag{1}$$

Since the VA of the first stage produces error events, and therefore correlated errors in $\hat{u}'_k$ and correlated values $\hat{p}'_k$ that might degrade the performance of the next stage, we apply sufficient deinterleaving to achieve statistical independence. A similar interleaving device is applied at the transmitter side. In the notation we drop the primes. At the dashed line A-A' in Fig. 1, the first stage detector delivers symbols $\hat{u}_k$ with statistically independent error probabilities $p_k$. Now, for the second stage detector the channel is a discrete (binary) memoryless compound channel [8] with output pairs $(\hat{u}_k, \hat{p}_k)$. If the second stage detector performs maximum–likelihood (ML) detection, an optimum ML metric is [9]

$$\sum_k x_k^{(m)} \cdot \hat{u}_k \log \frac{1-\hat{p}_k}{\hat{p}_k}, \tag{2}$$

where $x_k^{(m)} = \pm 1$ is the $k$-th symbol of the $m$-th information sequence. $\hat{u}_k$ is the hard ($\pm 1$) decision of the first Viterbi detector. We will call the first stage VA a *Soft–Output Viterbi Algorithm* (SOVA) because it delivers soft decisions

$$\hat{\Lambda}_k = \hat{u}_k \hat{L}_k, \tag{3}$$

with

$$\hat{L}_k = \log \frac{1-\hat{p}_k}{\hat{p}_k} \geq 0 \tag{4}$$

to be processed by the next ML-detector stage.

## 2.2 Algorithms

Given a finite–state discrete–time Markov process observed in white noise, the VA is the optimal recursive algorithm. It provides *the* state sequence with the lowest cost [1]. The VA is optimal in the ML sequence sense. It determines the symbol sequence $\hat{\mathbf{u}} = \{\hat{u}_k\}$ that maximizes the log–likelihood function $\log p(\mathbf{y} \mid \hat{\mathbf{u}})$.

### 2.2.1 The Soft–Output Symbol–by–Symbol MAP Algorithm

There exists an optimum general algorithm providing the a posteriori probabilities (APP's) for *each bit* to be decided [10], [1, Appendix]. This symbol–by–symbol MAP algorithm was originally developed to minimize the bit–error probability instead of the sequence error probability. The algorithm seems less attractive than the VA due to the increased complexity. However, we easily obtain the optimal APPs $P(u_k \mid \mathbf{y})$ for each bit to be decided. A soft decision as the log–likelihood ratio is obtained by

$$\hat{\Lambda}_k = \log \frac{P(\hat{u}_k \mid \mathbf{y})}{1 - P(\hat{u}_k \mid \mathbf{y})}, \tag{5}$$

where $\hat{u}_k$ is the MAP estimate for which $P(u_k \mid \mathbf{y})$ is maximum. The probability $P(u_k \mid \mathbf{y})$ can be calculated by a forward and a backward recursion following [10] and [1].

### 2.2.2 The Soft–Output Viterbi Algorithm (SOVA)

We shall first give a motivation for the algorithm before we state it formally. For the sake of simplicity we restrict ourselves to trellises with two branches ending in each node. This could include a $K/N$ code punctured from a $1/N$ code because it still uses the trellis of the $1/N$ code [11], [12]. The number of states $S$ is $S = 2^{\nu}$, where $\nu$ is the code memory.

Assume that the classical VA makes a final decision with delay $\delta$, $\delta$ being large enough so that all $2^\nu$ survivor paths have been merged with sufficiently high probability. As shown in Fig. 2, the VA has to select a survivor for state $s_k$, $0 \le s_k \le S = 2^\nu - 1$ at time $k$. It does so by selecting the path with the smallest ML metric, which for the AWGN channel, is

$$M_m = \frac{E_s}{N_0} \sum_{j=k-\delta}^{k} \sum_{n=1}^{N} (y_{jn} - x_{jn}^{(m)})^2, \; m = 1, 2 \,, \quad (6)$$

where $x_{jn}^{(m)}$ is the $n$-th bit of $N$ bits on the branch of the $m$-th path at time $j$, $y_{jn}$ is the received value at the same position, and $E_s/N_0$ is the signal-to-noise ratio (SNR). With this form we have

$$\text{Prob}\{\text{path } m\} \sim e^{-M_m}, \; m = 1, 2. \quad (7)$$

We label the path with the smaller metric by $m = 1$. This means $M_1 \le M_2$, which implies that the VA selects path 1 (neglecting ties). Then, the probability of selecting the wrong survivor path is

$$p_{sk} = \frac{e^{-M_2}}{e^{-M_1} + e^{-M_2}} = \frac{1}{1 + e^{M_2 - M_1}} = \frac{1}{1 + e^{\Delta}}, \quad (8)$$

with $\Delta = M_2 - M_1 \ge 0$. $p_{sk}$ approaches 0.5 if $M_1 \approx M_2$ and 0 if $M_2 - M_1 \gg 1$. With probability $p_{sk}$ the VA has made errors in all the $e$ positions where the information bits of path 2 differ from path 1; in other words if

$$u_j^{(1)} \ne u_j^{(2)} \,, \qquad j = j_1, \ldots, j_e. \quad (9)$$

Positions where $u_j^{(1)} = u_j^{(2)}$ are not affected by the survivor decision. Let $\delta_m$ be the length of those two paths until they merge. Then we have $e$ different information values, and $\delta_m - e$ nondifferent values. Assume we have stored the probabilities $\hat{p}_j$ of previous erroneous decisions with path 1. Under the assumption that path 1 has been selected we can update these probabilities for the $e$ differing decisions on this path according to

$$\hat{p}_j \leftarrow \hat{p}_j(1 - p_{sk}) + (1 - \hat{p}_j)p_{sk} \,, \quad j = j_1, \ldots, j_e, \quad (10)$$

$0 \le \hat{p}_j \le 0.5$. This formula requires statistical independence between the random variables $\hat{p}_j$ and $p_{sk}$, which is approximately true for most of the practical codes. The recursion could be directly performed on the log-likelihood ratio

$$\hat{L}_j = \log \frac{1 - \hat{p}_j}{\hat{p}_j} \,, \qquad 0 \le \hat{L}_j < \infty \,. \quad (11)$$

Using (8), (10), and (11) we obtain after some calculation

$$\hat{L}_j \leftarrow f(\hat{L}_j, \Delta) = \frac{1}{\alpha} \log \frac{1 + e^{(\alpha \hat{L}_j + \Delta)}}{e^{\Delta} + e^{\alpha \hat{L}_j}}, \quad (12)$$

with $\Delta = M_2 - M_1 \ge 0, j = j_1, \ldots, j_e$. The function $f(\hat{L}_j, \Delta)$ should be tabulated with $\hat{L}_j$ and $\Delta$ as input variables and need not to be calculated at each step. The factor $\alpha$ prevents overflow with increasing SNR. A proper choice to achieve asymptotically $E[\hat{L}_j] = 1$ is

$$\alpha = 4 d_{free} \frac{E_s}{N_0}, \quad (13)$$

where $d_{free}$ is the free distance of the code. A good approximation of (12) is

$$f(\hat{L}_j, \Delta) = \min\,(\hat{L}_j, \Delta/\alpha), \quad (14)$$

and requires no table and no knowledge of the SNR.

The *Soft-Output Viterbi Algorithm* (SOVA) can now be formulated using the notation in [1]. Only the steps marked by ($\star$) augment the classical Viterbi algorithm.

**Storage:**

$k$ (time index, modulo $\delta + 1$)
$\hat{\mathbf{u}}(s_k) = \{\hat{u}_{k-\delta}(s_k), \ldots, \hat{u}_k(s_k)\}, \quad 0 \le s_k \le S - 1$
(hard decision values, $\hat{u} \in \{\pm 1\}$)
$\hat{\mathbf{L}}(s_k) = \{\hat{L}_{k-\delta}(s_k), \ldots, \hat{L}_k(s_k)\}, \quad 0 \le s_k \le S - 1$
(soft reliability values, $0 \le \hat{L} \le \infty$) ($\star$)
$\Gamma(s_k), \quad 0 \le s_k \le S - 1$, (accumulated metric values)

**Recursion:**

a) Classical Viterbi step:
For each state $s_k$
Compute
$\Gamma(s_{k-1}, s_k) =$
$\Gamma(s_{k-1}) + \frac{E_s}{N_0} \sum_{n=1}^{N} (y_{kn} - x_{kn}^{(m)})^2$
for both transitions $(s_{k-1}, s_k)$.
Find $\Gamma(s_k) = \min \Gamma(s_{k-1}, s_k)$.
Store $\Gamma(s_k)$ and the corresponding survivor $\hat{u}_k(s_k)$.

b) Soft-deciding update: ($\star$)
For each state $s_k$
Store $\Delta = \max \Gamma(s_{k-1}, s_k) - \min \Gamma(s_{k-1}, s_k)$.
Initialize $\hat{L}_k(s_k) = +\infty$.
For $j = k - \nu$ to $j = k - \delta_m$
Compare the two paths merging in $s_k$
if $\hat{u}_j^{(1)}(s_j) \ne \hat{u}_j^{(2)}(s_j)$ then update
$\hat{L}_j := f(\hat{L}_j, \Delta)$

**Technical Realization:**
With an $n_s$ bit soft decision and fixed-point arithmetic, each survivor path of length $\delta$ consists of $n_s \cdot \delta$ bits. The first of $n_s$ bits is the sign bit or the hard-decision bit. The set of the likelihood values is then $\hat{L}_k \in \{0, 1, \ldots, 2^{n_s-1} - 1\}$. $\hat{L}_k = 0$ indicates the most unreliable value, and $\hat{L}_k = 2^{n_s-1} - 1$ the most reliable value. Given the metric difference, $\Delta$, quantized with $n_\Delta$ bits, the likelihood update is done with the table-look-up. The table is calculated only once by using Eqn. 12 and is stored in a ROM. Thus the extra effort of the SOVA relative to the VA is:

- Storage:
  - $2^\nu \cdot \delta \cdot n_s$ bits instead of $2^\nu \cdot \delta$ bits.
  - Look-up table with $2^{n_\Delta + n_s - 1}$ vectors each with $n_s - 1$ bits (unnecessary when using (14).
- Computation Complexity:
  - Maximal $2^\nu \cdot (\delta - \nu)$ bit comparisons.
  - $2^\nu \cdot e$ updates of $\hat{L}_k$.

The SOVA can be implemented in a pipelined structure [13] (clocked with symbol rate 1/T), see Fig. 3. Therefore, high speed implementation is possible. The Transition Metric Unit (TMU) and the Add-Compare-Select (ACS) unit remain unchanged. The novelity is a so called *Path Update Unit* (PAU). The PAU is used to "slide" over the stored (hard) information bits of the path RAM for the two paths diverging at time $k$ and remerging at time $k - \delta_m$, as shown in Fig. 2. The range where the information bits $\mathbf{u}^{(1)}$ and $\mathbf{u}^{(2)}$ *can* differ, is $j = k - \delta_m$ to $j = k - \nu$. *If* the bits differ a table-look-up is enabled to perform the update according to Eqn. 12 or 14. If the PAU is implemented in parallel the ACS unit remains the bottleneck. Therefore soft-deciding decoding does not nessesarily limit the speed.

#### 2.2.3 Input-Output SNR Conversion

The time-discrete analog output of the SOVA at a given point in time is a random variable, for which we can define an output SNR as the ratio $m^2/2\sigma^2$. This random variable is not always exactly Gaussian. However, for the next stage ML algorithm that decodes a code with distance $d$, the sum $\sum_{k=1}^{d} \hat{\Lambda}_k$ is the relevant decision variable. Because of the central limit theorem this sum is almost Gaussian and generates sequence errors with probability

$$P_d = \text{Prob}(\sum_{k=1}^{d} \hat{\Lambda}_k < 0) = \frac{1}{2}\text{erfc}\sqrt{d\frac{m^2}{2\sigma^2}}. \qquad (15)$$

Therefore the output SNR $= m^2/2\sigma^2$ is a useful figure of merit of the SOVA. This result can be interpreted in such a way that the input SNR (for the inner most section: the SNR of the channel) is converted into an output SNR. For closer examination we have plotted in Fig. 4 the simulated input/output conversion of the SNR for the best known $R = 1/3$ and $R = 1/2$ convolutional codes with memory $\nu = 3$. Simulation results with the symbol-by-symbol MAP algorithm show only minor differences of less than 0.3 $dB$.

The SNR input/output conversion is similar to that of analog FM. Similar to the FM-threshold we can define a SOVA-threshold. Enlarging the code distance by decreasing the code rate is analogous to an increased modulation index with FM-modulation. In both cases we exchange bandwidth with output SNR.

The common way to interpret channel coding is as a means for minimizing the BER. Concatenated coding is then viewed as a way to clean up the residual errors [14]. The observation of the preceding paragraph leads to a new interpretation of channel coding as *a means for improving the SNR*, since we have analog input- and output discrete-time signals in the inner sections, and perform hard decisions possibly only in the outer most section.

# 3 Applications

This novel receiver performs better than the conventional Viterbi decoder, -demodulator, or -equalizer whenever some form of concatenation is applied. This can include

- modulations with memory
  e.g. Trellis-Coded Modulation (TCM), Continuous Phase Modulation (CPM) such as Tamed FM (TFM), Gaussian MSK, etc.
- channels with memory
  e.g. filter-channels with intersymbol interference (ISI), frequency-selective channels, or storage media with memory like magnetic recording.
- coding with memory
  e.g. convolutional codes

and all possible combinations thereof.

## 3.1 Multistage Coding

#### 3.1.1 Convolutional Inner- and Outer Codes

Multistage or concatenated coding [15] with inner- *and* outer convolutional codes seems very promising.

The inner code uses the soft–quantized received samples, possibly supported by the channel state information in the case of channels with memory [9]. With the algorithms above, the outer decoder is also able to perform soft–decision ML decoding. First, we use the $R = 1/2$, $\nu = 3$ code with input/output SNR conversion presented in the last section as the inner code, and the punctured $R = 2/3$, $\nu = 3$ code [11] as the outer code. Both codes are separated by proper interleaving. The overall code rate is $R = R_i \cdot R_o = 1/3$.

For the simulations we used the SOVA and the MAP without significant differences. It is advantageous to use the same VA for the 1/2 and 2/3 punctured code with the PAU turned on only for the inner decoder. In [16] we presented a structure which also shares the interleaver in an efficient way and allows unequal error protection. The results are shown in Fig. 5. We have plotted the performance of the concatenated system derived from monte–carlo simulations. For comparison we also have plotted the performance curves for the best known $R = 1/3$ codes with memory $\nu = 3$, 4, and 6.

The question of optimal rate sharing between inner- and outer codes has been addressed elsewhere [17].

#### 3.1.2 Convolutional Codes and Outer Parity–Check Codes

ML decoding of block codes is not yet available in an elegant way. However, we give an example showing that with simple algorithms even parity–check codes gain from the soft decisions of the inner VA.

Suppose that a message of length $N$ bits is tailed by one bit resulting in a rate $R_o = N/(N+1)$ parity–check code. This code acts as the outer code. Sufficient interleaving and convolutional (inner) encoding is the second step of the encoding procedure. At the receiver we perform soft–decision decoding. A parity check is computed over the sign bits of the deinterleaved samples. If the parity–check fails we toggle the sign of the bit with the maximum probability of error $\hat{p}_k$ which is the bit with the lowest magnitude $\hat{L}_k$. Finally, we output the signs as the hard–decisions. We have plotted the curves for $R_i = 1/2$ and $R_o = 8/9$ in Fig. 6. An additional gain of 1.5 $dB$ at $10^{-6}$ was obtained.

#### 3.1.3 Coded Modulation and Outer FEC

Trellis–Coded Modulation [2] has gained strong interest in recent years, because it is power and bandwidth efficient. For example, 4PSK modulation can be replaced by Coded-8PSK to gain 3 to 6 $dB$ without bandwidth expansion. The drawback is that the conventional Viterbi demodulator of Coded-8PSK delivers only hard decisions. With the SOVA we should expect a gain in the next decoding stage.

We have investigated the performance of the 4-state rate 2/3 feedforward Ungerboeck code [2] with an 8PSK signal constellation and natural binary mapping. This code has a coding gain of 3 $dB$. The SOVA and the MAP work similar to Sections 2.2.1 and 2.2.2, except that we have to modify the metric transition unit, and treat the coded information bit and the uncoded bit (parallel transition) separately.

In Fig. 7 we have plotted the monte–carlo simulation results for the input/output SNR conversion for the SOVA. The plots show similar results as for the convolutional codes. Note that the parallel bit reaches the asymptotic gain of 3 $dB$. The asymptotic SNR improvement of the coded bit is about 4.7 $dB$. An outer FEC with ML–decoding would now work with the improved SNR.

### 3.2 Viterbi Equalizer and Coding

Equalization is a challenging problem in high–speed digital communications over time–dispersive channels, e.g. mobile channels. It is well established that the Viterbi equalizer performs the desired ML sequence estimation [18]. However, in coded systems the problem is obvious since the Viterbi equalizer delivers hard decisions to the outer decoder.

In this section we investigate the SOVA on frequency–selective fading channels. For the channel we suppose a tapped delay–line representation with $L+1$ independent Rayleigh tap gains. This channel, an idealization of the mobile multipath channel, can be viewed as the inner code. Now, the modified Gaussian likelihood metric reads (compare to (6))

$$M_m = \sum_k \frac{E_s(k)}{N_0} \mid y_k - \sum_{l=0}^{L} f_k^{(l)} x_{k-l}^{(m)} \mid^2, \qquad (16)$$

where $x_k^{(m)}$ is the symbol of the $m$-th path corresponding to the trellis, $f_k^{(l)}$ is the $l$-th tap gain, $0 \le l \le L$, and $y_k$ is the received value at the same time $k$, $x_k^{(m)}$, $y_k$ and $f_k^{(l)}$ denoted in complex notation. $E_s(k)/N_0$ is the instantaneous SNR. As outer code we chose the rate 1/2 convolutional code with memory $\nu = 4$ as proposed by GSM [4]. The results in Fig. 8, simulated for $L+1$ independent, Rayleigh–distributed tap gains, show the diversity gain for several values of $L$. However, more important in this context is the strong gain on the order of 4 $dB$ at $P_b = 10^{-3}$ due to the soft–output VA. Similar results have been derived for

trellis codes as outer codes [19]. As expected, the worse the channel, the greater is the gain due to soft decisions.

## References


[1] G.D. Forney, "The Viterbi Algorithm," *Proc. of the IEEE*, Vol. 61, No. 3, pp. 268-278, Mar. 73

[2] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. on Inf. Theory*, Vol. IT-28, No. 1, pp. 55-67, Jan. 82

[3] J.B. Anderson, T. Aulin, C.-E. Sundberg, *Digital Phase Modulation*, New York: Plenum Publishing Co., 1986

[4] Conference record of the "Digital Cellular Radio Conference," Hagen, Fern Universität, West-Germany, Oct. 1988

[5] T. Hashimoto, "A List-Type Reduced-Constraint Generalization of the Viterbi Algorithm," *IEEE Trans. on Inf. Theory*, Vol. IT-33, No. 6, pp. 866-876, Nov. 1987

[6] H. Yamamoto, K. Itoh, "Viterbi Decoding Algorithm for Convolutional Codes with Repeat Request," *IEEE Trans. on Inf. Theory*, Vol. IT-26, No. 5, pp. 540-547, Sep. 1980

[7] T. Schaub, J.W. Modestino, "An Erasure Declaring Viterbi Decoder and its Application to Concatenated Coding Systems," ICC'86, IEEE Cat. No. CH2314-3/86 , pp. 1612-1616, 1986

[8] R.G. Gallager, *Information Theory and Reliable Communication*, New York: John Wiley & Sons, 1968

[9] J. Hagenauer, "Viterbi Decoding of Convolutional Codes for Fading and Burst Channels," Proc. of the 1980 Zurich Seminar on Digital Comm., IEEE Cat. No. 80CH1521-4 COM, pp. G2.1-G2.7, 1980

[10] L.R. Bahl, J. Cocke, F. Jelinek, J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. on Inf. Theory*, Vol. IT-20, pp.284-287, Mar. 1974

[11] Y. Yasuda, K. Kashiki, Y. Hirata, "High Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding," *IEEE Trans. on Com.*, Vol. COM-32, pp. 315-319, Mar. 1984

[12] J. Hagenauer, "Rate-Compatible Punctured Convolutional Codes (RCPC Codes) and their Applications," *IEEE Trans. on Com.*, Vol. 36, No. 4, pp. 389-400, Apr. 1988

[13] J. Stahl, H. Meyr, M. Oerder, "Implementation of a High Speed Viterbi Decoder," EUSIPCO 86, conf. rec., pp. 1117-1120, 1986

[14] J.L. Massey, "The How and Why of Channel Coding," Proc. of the 1984 Zurich Seminar on Digital Communications, IEEE Cat. No. 84 CH 1998-4, pp. 67-73, 1984

[15] G.D. Forney, *Concatenated Codes*, Cambridge: M.I.T. Press, 1966

[16] J. Hagenauer, "Unequal Error Protection (UEP) for Statistically Time-Varying Channels," Proceedings ITG-Conference 'Stochastic Models and Methods in Information Technology', Nürnberg, W.-Germany, ITG-Fachbericht No. 107, VDE Verlag Berlin, pp. 253-262, Apr. 1989

[17] J. Hagenauer, P. Hoeher, "Concatenated Viterbi-Decoding," Proceedings of the 4. Joint Swedish-Soviet Int. Workshop on Inf. Theory, Gotland, Sweden, Studentlitteratur Lund, 27. Aug.-1. Sep. 1989

[18] G.D. Forney, "Maximum-Likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference," IEEE Trans. Inf. Theory, Vol. IT-18, No. 3, pp. 363-378, May 1972

[19] P. Hoeher, "Detection of Uncoded and Trellis-Coded PSK-Signals on Frequency-Selective Fading Channels," Proceedings ITG-Conference 'Stochastic Models and Methods in Information Technology', Nürnberg, W.-Germany, ITG-Fachbericht No. 107, VDE Verlag Berlin, pp. 225-232, Apr. 1989


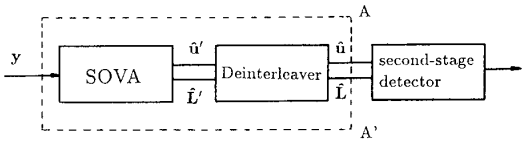


Figure 1: Detector with Reliability Information

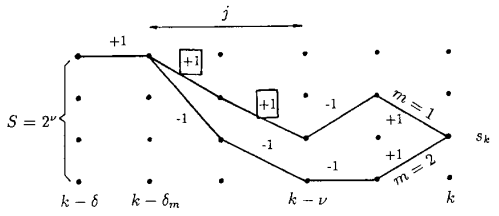


Figure 2: Example of the SOVA

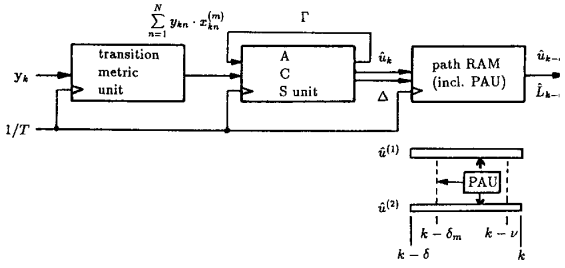


Figure 3: Blockdiagram of the SOVA with Pipelining and the Path Update Unit (PAU)

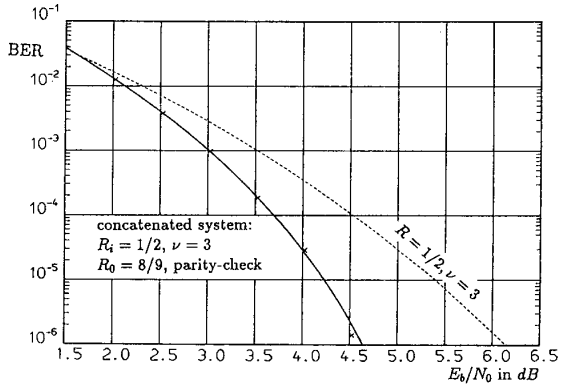


Figure 6: BER for the $R_i = 1/2$ Inner Convolutional Code with SOVA and the $R_o = 8/9$ Outer Parity-Check Code

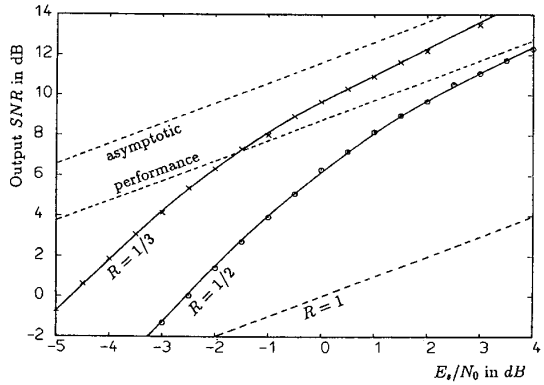


Figure 4: SNR Input/Output Conversion for Various Convolutional Codes

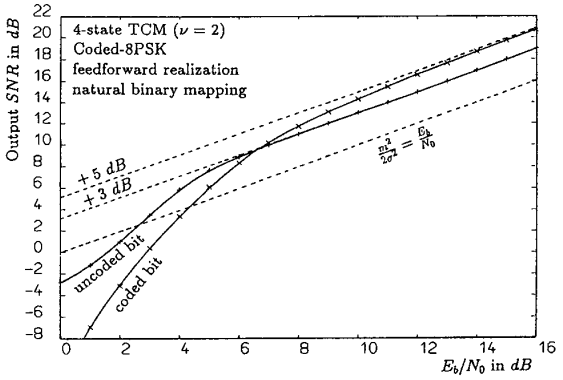


Figure 7: Input/Output SNR Conversion for 4-State Trellis-Coded Modulation

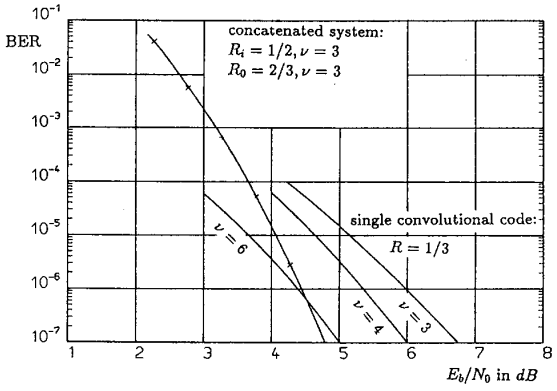


Figure 5: Simulated BER for Concatenated Convolutional Codes

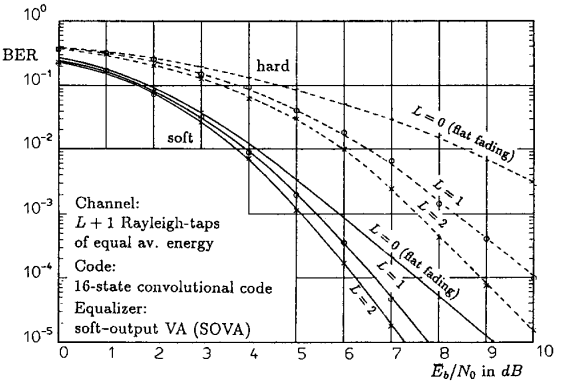


Figure 8: BER for Coded Equalization on a Frequency-Selective Fading Channel