# Deadline dependent coding-a framework for wireless real-time communication

Elisabeth Uhlemann, Per-Arne Wiberg , Tor M. Aulin and Lars K. Rasmussen



Uhlemann E, Wiberg P, Aulin T, Rasmussen L. Deadline dependent coding-a framework for wireless real-time communication. In: Seventh International Conference on Real-Time Computing Systems and Applications: proceedings : 12-14 December, 2000, Cheju Island, South Korea. Piscataway, NJ: IEEE; 2000. p. 135-142.

DOI: http://dx.doi.org/10.1109/RTCSA.2000.896381
Copyright: IEEE

# Deadline Dependent Coding - A Framework for Wireless Real-Time Communcation*

Elisabeth Uhlemann and Per-Arne Wiberg
*Halmstad University,*
*Computing and Communications Laboratory,*
*Box 823*
*SE-301 18 Halmstad, Sweden*
*{elisabeth, per-arne.wiberg}@ide.hh.se*

Tor M. Aulin and Lars K. Rasmussen
*Chalmers University of Technology,*
*Telecommunication Theory Group,*
*Dept. of Computer Engineering,*
*SE-412 96 Göteborg, Sweden*
*{tor, larsr}@ce.chalmers.se*

## Abstract

*A framework for real-time communication over a wireless channel is proposed. The concept of deadline dependent coding (DDC), previously suggested by the authors, is further developed using soft decision decoding of block codes to maximize the probability of delivering information before a given deadline. The strategy of DDC is to combine different coding and decoding methods with automatic repeat request (ARQ) in order to fulfil the application requirements. These requirements are formulated as two Quality of Service (QoS) parameters: deadline ($t_{DL}$) and probability of correct delivery before the deadline ($P_d$), leading to a probabilistic view of real-time communication. An application can negotiate these QoS parameters with the DDC protocol, thus creating a flexible and dependable scheme.*

## 1. Introduction

The tremendous development in wireless communication has provided opportunities in many related fields. This wireless evolution offers improvements for industrial applications, where traditional wiring causes inhibiting problems in terms of costs and feasibility. Some applications even require a wireless connection in order to function. A recognized problem is the harsh communication environment encountered by a wireless connection. This has prevented extensive use of wireless access in real-time systems. In some real-time communication systems it is often assumed that the deadline must be met with unit probability. This is a situation that cannot be achieved with any physical system due to noisy channel conditions [1].

Given sufficient control over the quality of the communication channel, wireless communication can replace cables in a vast number of real-time systems. A class of industrial applications is measurement and control of moving objects. Another application is communication to and from different kinds of vehicles in factory automation situations. Implementations of wireless communication systems for industrial use have been attempted, however, the problem of guaranteeing real-time delivery is usually solved in a somewhat *ad hoc* manner. As a consequence, it is not straightforward to evaluate the dependability of the entire system. In this paper, we describe a framework for dependable real-time communication over an unreliable wireless communication channel, exemplified here by a radio channel.

A real-time communication system is characterized by the fact that it is equally important to deliver in time, as it is to deliver correct data. The literature in the real-time field often discusses two different classes of systems, *hard* and *soft* real-time systems. In a hard real-time system, late delivery cannot be tolerated. In contrast, in a soft real-time system a certain low probability of late delivery is tolerated, but leads to performance degradation in terms of the real-time constraints. In our framework we introduce a probabilistic view of the real-time constraints. This means it is no longer meaningful to talk about hard or soft real-time systems. We rather talk about deadline for delivery and the probability of succeeding in delivering correct information before this deadline. We therefore introduce two parameters: deadline ($t_{DL}$) and probability of correct delivery before deadline ($P_d$). They can be viewed as Quality of Service (QoS) parameters of the real-time communication system. It follows that a protocol layer can negotiate the parameters with an underlying layer. The protocol can then accept a request, guaranteeing the delivery based on the given QoS parameters, or reject it. If the request is rejected the application can possibly re-negotiate. One of the objectives of the real-time communication protocol is to maximize the probability that the communication system will be able to accept the transmission request.

* This work was supported in part by ARTES (A network for Real-Time research and graduate Education in Sweden) "www.artes.uu.se".

The main idea behind the concept of Deadline Dependent Coding (DDC) [2] is to make the communication protocol deadline dependent. The protocol should also attempt to minimize the bandwidth, the transmitted energy and the time required to successfully deliver the information. The QoS parameters $t_{DL}$ and $P_d$ are mapped onto a retransmission protocol, which plays the role of maximizing the probability of correct delivery before deadline and still being able to reject requests that cannot be handled. The DDC protocol performs a series of transmissions triggered by the retransmission protocol, providing increasingly more information for decoding the closer we come to the deadline. We denote this series of transmissions a *transmission suite*. Aspects that must be considered are not only static and dynamic channel properties like throughput and error rate, but also multiple access interference (MAI). In a multiple access radio system it is important to limit the interference created by the acting nodes. This is another reason for getting the information delivered with as little use of the communication resources as possible. As the deadline approaches, it becomes increasingly more important to get the information delivered regardless of possible undesirable MAI. As a consequence, increasingly more channel resources are allocated in order to meet the probabilistic requirements for delivery before the deadline.

The DDC scheme differs from existing communication protocols in a number of ways. Most importantly, DDC explicitly uses the deadline to control the transmission suite. There are a number of real-time communication protocols as [3] and [4] that are best effort protocols and consequently do not give any guarantees or explicit predictions on the probability of delivery. Other protocols that are guarantee seeking can thus guarantee hard deadlines [5], but rely on a dependable channel. DDC strives to maximize the probability of correct delivery over an undependable channel. In this paper the concepts of DDC are further developed and new improved methods are used within the scheme.

In order to develop the DDC scheme for real-time communication over a radio channel, we need to define a framework of methodologies and terminology based on communication and coding theory. Consequently, section 2 contains the necessary background in telecommunication theory to provide tools for integrating real-time constraints in a wireless environment. The choice of decoding strategy is also motivated here. Section 3 contains a description of retransmission protocols. The DDC scheme is then analyzed by means of the QoS parameters in section 4 and results obtained by computer simulation are presented. Finally section 5 contains our conclusions.

## 2. Telecommunication framework

We need to derive notation and concepts that can be used to unite the area of real-time systems with that of telecommunication theory. This telecommunication framework will then be used to map the real-time constraints onto a communication protocol for a digital radio channel.

### 2.1. System model

The communication system used throughout this work can be described by the block diagram in Figure 1. The *information source* generates a sequence of information symbols. These symbols are grouped into blocks $k$ symbols long, denoted $\mathbf{u}$. The *encoder* then adds a controlled amount of redundant symbols to each of these blocks, producing a longer $n$-symbol code block, $\mathbf{c}$. In the *modulator* each symbol in the $n$-symbol code block is associated with a corresponding signal waveform, $s_i(t)$, for transmission over the *channel*. We have used binary phase shift keying (BPSK) modulation [1] throughout this work, not because it is the best possible modulation method, but because it is a commonly used method in many existing hardware platforms. The channel corrupts the signal waveforms in a random manner. A simple and frequently used mathematical model for the communication channel is the additive white Gaussian noise (AWGN) channel [1] which models thermal noise present in all electronic equipment. It adds a Gaussian random noise process $n(t)$ to the transmitted signal $s(t)$. At the *demodulator* the corrupted signal waveforms are reduced to a sequence of numbers, $\mathbf{r}$, that represent sufficient statistics for detection of the transmitted symbols. The vector $\mathbf{r}$ of observables contains all statistical information relevant for optimal detection of $\mathbf{u}$. The sequence of observables is then fed to the *decoder*, which attempts to reconstruct the original information sequence, $\hat{\mathbf{u}}$, using the redundant symbols.

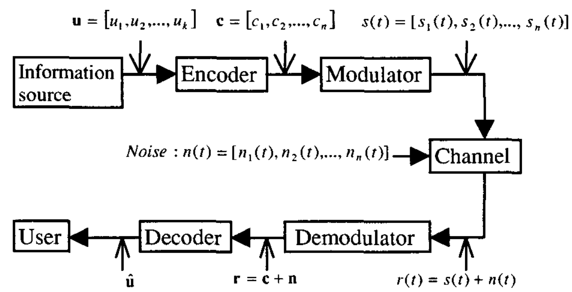


**Figure 1. Block diagram of the communication system.**

In many cases the received symbol vector, $\mathbf{r}$, is first sent through a two-level quantizer providing the decoder with only digital zeros and ones. When a two-level quantizer is used the decoder is said to make *hard decisions* and the resulting channel (consisting of the modulator, the AWGN channel, the demodulator and the quantizer) is called a binary symmetric channel, (BSC) [1]. Decisions based directly on the unquantized demodulator output, so-called *soft decision decoding*, require a more complex decoder that can handle continuous or non-binary inputs, but as we shall see, offers a significant performance improvement over hard decision decoding.

### 2.2. Linear block codes

Block codes introduce controlled amounts of redundancy into a transmitted data stream, enabling the receiver to make more accurate estimates of the transmitted sequence although it is corrupted by noise over the communication channel. The linear block codes used in the DDC scheme are Reed-Solomon (RS) codes [6]. These codes are especially good at handling noise burst, which is a common phenomenon in a wireless communication system due to fading. Reed-Solomon codes are also maximum-distance codes; i.e. its code words are at maximum achievable symbol distance from each other for a given number of information symbols and a fixed block length. This distance between the code words has a direct influence on the performance of the code.

The information data stream is divided into blocks of $k$ symbols, $\mathbf{u}=[u_1,u_2,\ldots,u_k]$, where each symbol takes values from the Galois field, GF($q$) [6]. Such symbols are $q$-ary, representing $\log_2(q)$ bits each. Each message block of $k$ symbols is then encoded, generating a code word of $n$ symbols $\mathbf{c}=[c_1,c_2,\ldots,c_n]$, where $n>k$, each symbol again taking values from GF($q$). Consequently, the total amount of redundancy introduced is $r = n - k$. Reed-Solomon codes has a minimum distance between the code words of $d_{min}=n-k+1$. Each $(n,k)$ block code, $C$, can be described by a generator matrix $\mathbf{G}$, which when multiplied by the information symbol block, $\mathbf{u}$, produces a code word, $\mathbf{c}$, according to $\mathbf{c}=\mathbf{u} * \mathbf{G}$ where "$*$" denotes multiplication using Galois field arithmetic [6]. Associated with each $(n,k)$ block code is also a parity check matrix $\mathbf{H}$, which when multiplied with a valid code word, $\mathbf{c}$, returns the all zero vector as $\mathbf{c} * \mathbf{H}^T=\mathbf{0}$.

The Reed-Solomon (RS) code word symbols are always $q$-ary, where $q>2$. Before they can be transmitted onto the channel they have to be modulated, here by using a binary phase shift keying (BPSK) modulation technique. This means that the $q$-ary code word symbols are translated into a sequence of $\log_2(q)$ binary channel symbols before transmission. At the receiver, the channel symbols are demodulated and can then be translated back to $q$-ary code word symbols. The notation RS(7,3) implies a Reed-Solomon (RS) code with three information symbols that are coded into a code block of seven symbols. The symbols in this work are defined over GF(8) which means that $q$=8 and there are consequently eight different symbols available. Each octal code symbol is thus translated into three binary channel symbols for modulation.

### 2.3. Maximum likelihood decoding

Assume that the decoder has received a vector $\mathbf{r}$. The optimum decoder will then select the code sequence that minimizes the probability of error, more precisely we want the decoder to select the sequence $\hat{\mathbf{c}} = \mathbf{c}_k$ iff [1]:

$$P[\mathbf{c}_k \mid \mathbf{r}] > P[\mathbf{c}_i \mid \mathbf{r}]; \qquad \forall i \neq k \tag{1}$$

This is known as the maximum aposteriori probability (MAP) criterion. From Bayes rule [1] we get:

$$P[\mathbf{c}_i \mid \mathbf{r}] = \frac{P[\mathbf{c}_i] p_{\mathbf{r}}(\mathbf{r} \mid \mathbf{c} = \mathbf{c}_i)}{p_{\mathbf{r}}(\mathbf{r})} \tag{2}$$

Since $p_{\mathbf{r}}(\mathbf{r})$ is independent of $i$ it has no impact on the decoding procedure and can thus be ignored. In case the a priori probabilities $P[\mathbf{c}_i]$ can be assumed to be equal for all $i$, the MAP criterion simplifies to the maximum likelihood (ML) criterion and the optimum receiver then sets $\hat{\mathbf{c}} = \mathbf{c}_k$ iff:

$$p_{\mathbf{r}}(\mathbf{r} \mid \mathbf{c} = \mathbf{c}_k) > p_{\mathbf{r}}(\mathbf{r} \mid \mathbf{c} = \mathbf{c}_i); \qquad \forall i \neq k \tag{3}$$

The AWGN channel where $\mathbf{r} = \mathbf{c} + \mathbf{n}$ yields:

$$p_{\mathbf{r}}(\mathbf{r} \mid \mathbf{c} = \mathbf{c}_i) = p_{\mathbf{n}}(\mathbf{r} - \mathbf{c}_i \mid \mathbf{c} = \mathbf{c}_i) = \tag{4}$$

$$p_{\mathbf{n}}(\mathbf{r} - \mathbf{c}_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-|\mathbf{r}-\mathbf{c}_i|^2/2\sigma^2}$$

Consequently the optimum ML receiver will set $\hat{\mathbf{c}} = \mathbf{c}_k$ iff:

$$|\mathbf{r} - \mathbf{c}_k|^2 < |\mathbf{r} - \mathbf{c}_i|^2, \tag{5}$$

where

$$|\mathbf{r} - \mathbf{c}_i|^2 = \sum_{j=1}^{n} (r_j - c_{ij})^2. \tag{6}$$

The expression in (6) is called the squared Euclidean distance. A decoder that calculates the squared Euclidean distance for a sequence transmitted over an AWGN channel is called a maximum-likelihood sequence detector (MLSD) for the AWGN channel. If hard decisions are made on $\mathbf{r}$ prior to decoding by means of a two-level quantizer, then the decoder experiences a BSC. In this case a MLSD for the BSC will instead select $\hat{\mathbf{c}} = \mathbf{c}_k$ iff [1]:

$$p_{\mathbf{r}'}(\mathbf{r}' \mid \mathbf{c} = \mathbf{c}_k) > p_{\mathbf{r}'}(\mathbf{r}' \mid \mathbf{c} = \mathbf{c}_i); \qquad \forall i \neq k \tag{7}$$

where $\mathbf{r}'$ is the received sequence delivered to the decoder according to Figure 2. The BSC changes a binary "0" to a binary "1" with probability $p$. This so-called cross-over probability $p$ can easily be determined for a two-level quantized AWGN channel using a probability density function similar to that of equation (4), [1]. Whenever $\mathbf{r}'$ differs from $\mathbf{c}_i$ in $d_l$ coordinates:

$$P[\mathbf{r}' \mid \mathbf{c}_i] = p^{d_l} q^{N-d_l} = q^N \left(\frac{p}{q}\right)^{d_l}; \quad q \triangleq 1 - p. \tag{8}$$

where $N$ is the number of components in $\mathbf{r}$. The quantity $d_l$ is known as the *Hamming distance* between $\mathbf{r}'$ and $\mathbf{c}_i$, which is the number of positions in which two vectors differ.

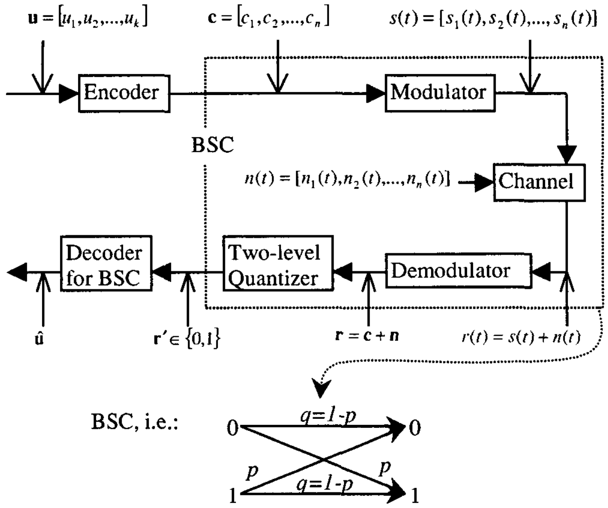


**Figure 2. Block diagram of a BSC.**

A MLSD for an AWGN channel has to compute the squared Euclidean distance between the received sequence $\mathbf{r}$ and all the $q^k$ available code words in $C$ and for a BSC, the Hamming distance is required instead. For both cases this is a complex operation even for small $k$. There exists, however, algorithms that reduces the complexity of the MLSD to a manageable level. They usually require a graphical representation of the code.

## 2.4. Trellis representation of linear block codes

A graphical representation of a block code, $C$, called a trellis can be constructed using either the generator matrix $\mathbf{G}$ or the parity check matrix $\mathbf{H}$ associated with $C$, [7]. A trellis is a directed graph consisting of a particular collection of states and branches, in which every path through the trellis represents a code word in the set $C$. In Figure 3, a trellis of a binary (8,4) Reed-Muller code [7] is shown for illustration as the trellis for a RS(7,3) code is significantly more complex.

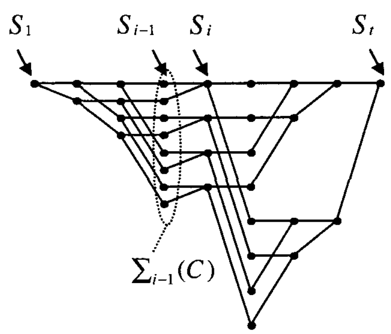


**Figure 3. Trellis for a binary (8,4) block code constructed from the parity check matrix H.**

The states in the trellis are grouped into different sets. The states in set $i$, denoted $\sum_i(C)$, are said to be at *depth i*. A branch in the $i$-th section of the trellis connects a particular state $S_{i-1} \in \sum_{i-1}(C)$ to a particular state $S_i \in \sum_i(C)$ and is labeled with the corresponding code bit $c$ that represents the encoder output at the interval from time ($i$-1) to time $i$. Each branch represents a state transition. The branches diverging from the same state have different labels, representing different code symbols. Each path from the initial state $S_1$ to the final state $S_t$ is thus a code word in $C$.

The concept of trellises was first used to decode convolutional codes [6] and the fact that block codes can be interpreted as trellises was first described in [8] and later in [9]. The complexity of a decoding algorithm operating on the code trellis is dependent on the number of states and the number of branches. A trellis for an $(n,k)$ block code defined over GF($q$) and constructed from the generator matrix $\mathbf{G}$ will have at most $q^k$ states at any time interval, while a trellis constructed from the parity check matrix $\mathbf{H}$ of the block code will have at most $q^{n-k}$ states. Consequently, to reduce decoding complexity a code trellis is constructed from the $\mathbf{H}$ matrix whenever the number of redundant symbols are less than the number of information symbols, otherwise the $\mathbf{G}$ matrix is used.

The construction of a code trellis makes it possible to implement maximum likelihood sequence detection (MLSD) with manageable complexity. We simply search through the trellis looking for the maximum-likelihood sequence. However, we do not have to consider all $q^k$ possible code sequences. We can use the Viterbi algorithm [10] to eliminate certain paths that merge in the same state in the trellis with another more likely path. For each branch in the trellis we compute the corresponding branch metric. For the BSC the branch metric corresponds to the Hamming distance between the received symbol and the particular code symbol considered and for the AWGN channel the squared Euclidean distance obtained directly from the unquantized signal is used as branch metric.

## 2.5. Computer simulations

Figure 4 shows computer simulated results of MLSD using the Viterbi algorithm for two different channels, BSC and AWGN. Two different Reed-Solomon codes are also compared, RS(7,5) and RS(7,3). The code trellis for RS(7,5), having ($k$>$n$-$k$), is constructed from the $\mathbf{H}$ matrix whereas the code trellis for RS(7,3) is based on $\mathbf{G}$. The code word blocks are transmitted using BPSK over an AWGN channel. The information bit error rate is plotted as a function of $E_b/N_0$, where $E_b$ is the energy used per information bit and $N_0$ the one-sided spectral density of the channel noise.

As a reference the uncoded case is also plotted in Figure 4.

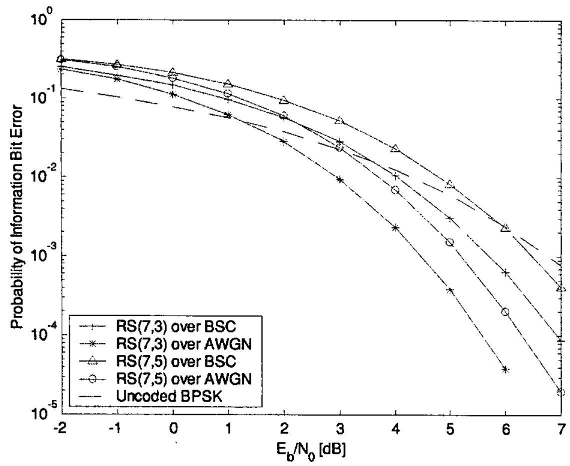


**Figure 4. MLSD Decoding of RS-codes for different channels.**

We can see from Figure 4 that we gain more by using soft decision decoding (unquantized signal) of RS(7,5) compared to hard decisions, than by changing to the RS(7,3) code even though the latter code contains more redundant symbols, i.e. it has a lower *code rate*. This is of particular interest in a real-time system, as more time is required to transmit a longer code block, which results in a reduced throughput.

## 3. Hybrid automatic repeat request

When data is transmitted in packets an Automatic Repeat reQuest (ARQ) scheme [11] can be used. Whenever a packet arrives, the receiver may choose *not* to accept the packet, but instead request a retransmission through a feedback channel. To determine whether a retransmission should be requested or not the receiver checks the *quality* of the received packet, usually this is done by means of an error detecting code, like a cyclic redundancy check code (CRC). This two-way communication goes on until the receiver obtains a packet that is considered *reliable*. If the packet is accepted, an acknowledgement message is sent.

A hybrid ARQ (HARQ) scheme, first suggested in [12], uses an error control code in conjunction with the retransmission scheme. Consequently, it tries to *decode* the received code word first and only requests a retransmission if the uncertainty of the decoding decision is considered too high, i.e. if the detection is below a certain *reliability threshold*. Hard decision decoding of Reed-Solomon codes applied in a HARQ scheme is considered in [13] for fading channels.

There are different methods of determining whether the reliability of the decoding decision is too low and hence different methods of demanding a retransmission. The choice of method highly affects the character of the retransmission scheme. In a non-ARQ scheme, after decoding a packet we either succeed, $P_d$, or the received packet is interpreted as another valid code word, resulting in a block error, $P_e$. These probabilities are related as:

$$P_e + P_d = 1 \tag{9}$$

In a HARQ scheme, we choose not to accept the decoded packet if the reliability of the decoding decision is too low, and instead request a retransmission. This results in a certain *probability of retransmission*, here denoted $P_{ret}$. If $P_{ret}$ is increased consequently $P_e$ and $P_d$ is reduced, as:

$$P_e^i + P_{ret}^i + P_d^i = 1, \tag{10}$$

where $i$ represent the $i$:th retransmission. Even though we desire $P_e$ to be minimized, a large $P_{ret}$ will result in numerous retransmissions, yielding a very slowly increasing $P_d$ and a low throughput, see Figure 5.

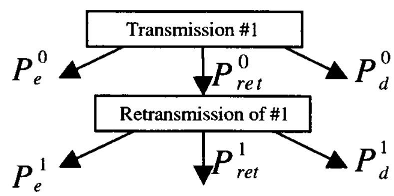


**Figure 5. The different probabilities involved in a retransmission scheme.**

In this paper we will use the concept of *bounded distance* to determine the reliability of the decoding decision. A bounded distance decoder [6] will select the code word closest in Hamming distance to the received word if and only if that distance is less than a certain bounded distance, typically $(d_{min}\text{-}1)/2$. If there is no code word within the bounded distance a *decoder failure* is declared and consequently a retransmission can be requested. A bounded distance decoder will make an error whenever the received word is within the bounded distance of a code word different from the one which was sent, see Figure 6. The DDC-protocol was analyzed by the authors in [2], using the bounded distance decoder based on hard decisions (i.e. a two-level quantizer).

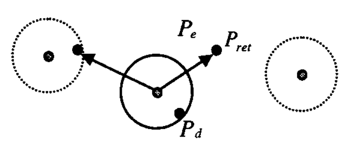


**Figure 6. Bounded Distance Decoding. In the centre of each sphere is a valid code word.**

We will use the bounded distance concept and hence request a retransmission whenever the received packet is at a Hamming distance greater than $(d_{min}\text{-}1)/2$ from the code word selected by the detector. This retransmission strategy is used for the MLSD for both the BSC and the AWGN channel in order to make a reasonable comparison between the two channel types in a hybrid ARQ scheme.

There are also different ways of using the information in the previously received packets, i.e. different packet combining techniques, in order to improve performance. The retransmission scheme described in [2] uses a bitwise majority voting procedure whenever three or more packets have been received. There are, however, other methods that yield better performance for soft decisions. If the branch metric used in the trellis for a newly received packet, $\mathbf{r}_2$, is added to the corresponding branch metric of a previously received packet, $\mathbf{r}_1$, a mean of the current and previous package is obtained as:

$$|\mathbf{r}_1 - \mathbf{c}_i|^2 + |\mathbf{r}_2 - \mathbf{c}_i|^2 = |(\mathbf{r}_1 + \mathbf{r}_2) - \mathbf{c}_i|^2 \qquad (11)$$

The expression in (11) is known as *equal gain combining,* [14] [15]. A packet combining strategy based on code word by code word combining is referred to as a *code combining* method and was first considered in [16] for ARQ-protocols. A code symbol by code symbol combining method is called *diversity combining* and different varieties of this method is considered in the literature, for example in [17]. In the computer simulations presented below, we use the equal gain combining method to take advantage of the information in previous packets for the AWGN channel. The bitwise majority voting method is used for the BSC case.

## 3.1. Computer simulations

We have again used BPSK modulation over the AWGN channel and the BSC and plotted the information bit error rate as a function of $E_b/N_0$. The RS(7,3) using a bounded distance retransmission criterion of $(d_{min}\text{-}1)/2$ over the AWGN channel is plotted in Figure 7 and over the BSC in Figure 8. For simplicity, we have assumed an error free feedback channel for both the AWGN channel case and BSC case. Three scenarios are plotted. In the first only one transmission is allowed. In the second, one retransmission can be made when requested for, but after that the packet is accepted. In the third, a maximum of two retransmissions is made and consequently the same packet is transmitted at most three times. Thereafter the packet is accepted unconditionally.

In the AWGN case the information from each previous packet is taken into account by equal gain combining before the decoding procedure takes place. In contrast, the detector for the BSC performs a bit-wise majority voting procedure among the rejected copies as soon as it has received at least *three* packets. This is why the MLSD for the BSC gains more from the third transmission than from the second.

When a packet is considered to be unreliable by the retransmission scheme it is not accepted and consequently no errors are made at this time. Instead a retransmission is made and a new decoding procedure takes place, but this time with more information available. If accepted, this newly decoded packet is less likely to contain errors than the one available after the first transmission. This is why the information bit error rate is reduced when retransmissions are allowed. Obviously, the decoding strategy still influences the procedure and soft decision decoding on the unquantized signal is still superior to hard decision decoding.

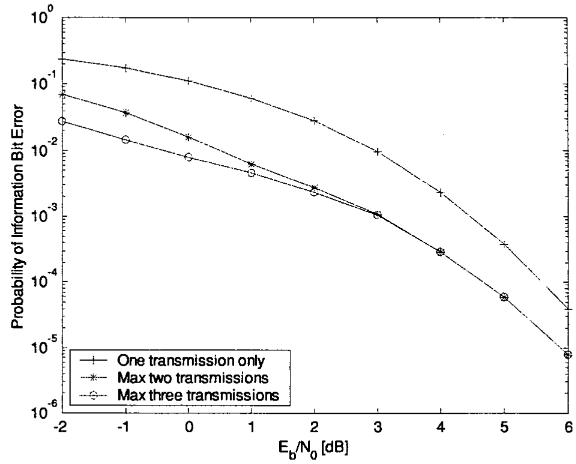


**Figure 7. The information bit error rate using a bounded distance retransmission criteria for RS(7,3) over the AWGN channel.**

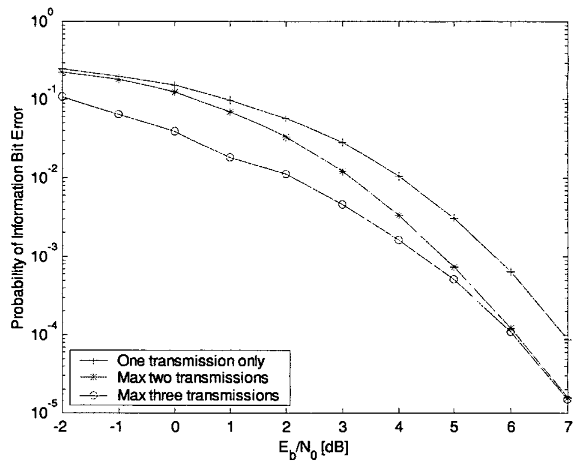


**Figure 8. The information bit error rate using a bounded distance retransmission criteria for RS(7,3) over the BSC channel.**

We can see from the figures that the curves allowing a third transmission tend towards the ones allowing only two transmissions as $E_b/N_0$ increases. This is due to the fact that the higher the $E_b/N_0$, the need for another retransmission is smaller and hence also the gain from it.

It should be noted that each retransmission actually *increases* the total energy spent for transmission of a single information bit. This is because of the fact that the information bits in the second packet serves as redundant information for the information bits in the first packet, thus the rate of the code is actually increased. In the simulations in Figure 7 and Figure 8 the information bit error rate is plotted as a function of the information bit energy-to-noise ratio $E_b/N_0$, but with reference to the same code rate, hence the increased information bit energy has not been compensated for. The information bits for each retransmission is regarded as independent of each other in terms of the consecutive energy used.

## 4. The DDC scheme based on the QoS parameters

Deadline Dependent Coding (DDC) [2] is a communication scheme that strives to meet the real-time demands on deadline, here denoted ($t_{DL}$) and probability of a correct delivery before the deadline ($P_d$), while keeping the utilization factor of the network as low as possible. As the name DDC implies, the message to be transmitted is coded differently depending on the deadline and the requested probability of correct delivery. The code rate and the number of retransmissions allowed are thus determined by the requested $t_{DL}$ and $P_d$. In the beginning of the time window, Figure 9, a high rate coded message, i.e. with few redundant bits, is transmitted. If the message received is considered too uncertain and the receiver does not succeed in decoding it with sufficient reliability, a retransmission made. This procedure will be repeated until the requested level of $P_d$ has been achieved. Even though our retransmission mechanism might tell us that we need another retransmission, we choose to ignore that fact if the required level of $P_d$ has already been achieved and instead deliver the current result in order to meet the deadline. It should be noted that once a request has been accepted by the protocol the probability of delivering a packet, with the required QoS parameters $P_d$ and $t_{DL}$, is guaranteed to hold.

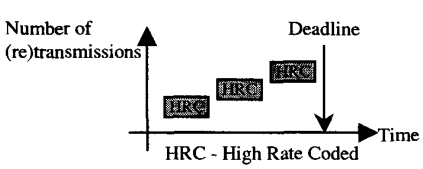


**Figure 9. The Deadline Dependent Coding (DDC) Transmission Suite.**

The application designer states the probability of correct delivery together with the deadline as requirements. The communication mechanism then transforms this request into appropriate actions in the protocol, e.g. number of retransmissions allowed and required code rate. If the transform can be made, the request is accepted, guaranteeing the delivery based on the given QoS parameters. The DDC scheme was analyzed by the authors using hard decision decoding (i.e. BSC) in [2].

If we assume that we transmit the RS-codes using BPSK over an AWGN-channel the probability of correct delivery before deadline, $P_d$, can be plotted as a function of time. We assume again an error free feedback channel and we do not take into account the time to decode a packet. Even though the trellis based decoding algorithms are faster in our simulation environment there exists very fast non-trellis based algebraic decoders and hence these aspects will not be considered in this paper. In Figure 10 and Figure 11, $P_d$ is plotted as a function of time for RS(7,3) over the AWGN channel and the BSC. In the AWGN case the information from each previous packet is taken into account by equal gain combining before decoding. The detector for the BSC performs a bit-wise majority voting procedure among the rejected copies as soon as it has received at least three packets.

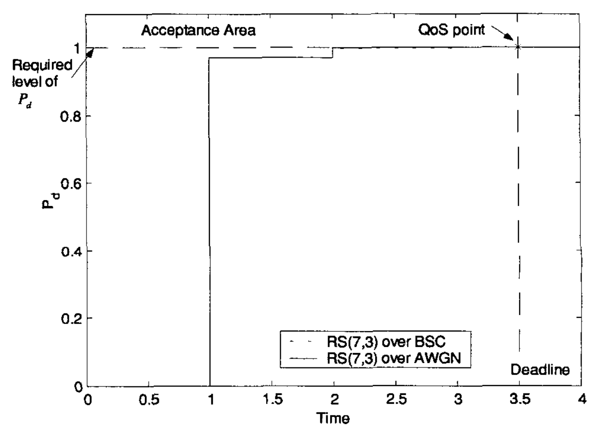


**Figure 10. $P_d$ plotted as a function of time for RS(7,3) over different channels. ($E_b/N_0$=5dB)**

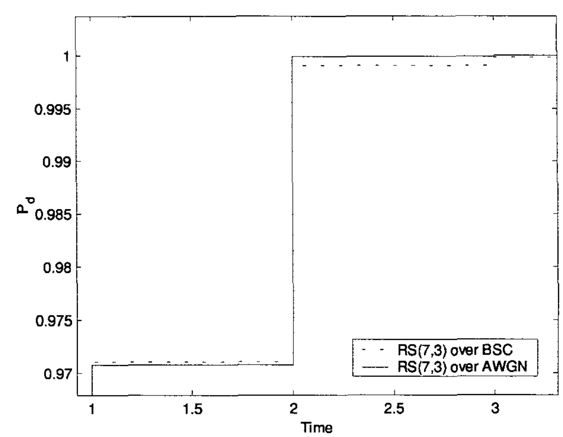


**Figure 11. Enlargement of Figure 10.**

The area above the required level of $P_d$ and before deadline we refer to as the *acceptance area.* In this area we can keep a message even though it is considered unreliable by the chosen retransmission mechanism. Consequently we do not have to do any more retransmissions because we have already reached the level of $P_d$ required by the application. The QoS parameters determine the location of the *QoS point* as seen in Figure 10 and hence the acceptance area. A deadline, $t_{DL}$, and a required level of $P_d$ are also plotted in Figure 10 as examples to illustrate the acceptance area and the QoS point. $P_d$ is typically: $P_d - \varepsilon = 1$, where $\varepsilon$ is a very small number. If we transmit at, for example, 1 Mbps and assume that the acknowledge message requires a maximum of 7 bits, then 1 time unit in Figure 10 and Figure 11 equals 28 $\mu$s (the RS(7,3) using BPSK consists of 21 bits and the acknowledge of 7 bits).

We see that the function has a staircase characteristic. The length of a step in the staircase depends on the packet length, i.e. the longer the code word, the wider the step. The height of each step is dependent on the amount of redundancy, i.e. the code rate, the decoding strategy and the retransmission criterion. Increased redundancy results in a higher step. If a low rate code is used, the throughput will be lower, but we have an opportunity to reach a higher value of $P_d$ after a series of retransmissions. Soft decision decoding yields a lower bit error rate on the accepted packets, which of course will increase the height of a step if the packet is accepted. If the packet is considered unreliable too often the bit error rate will decrease, but number of steps will increase and at the same time the height of each step will be reduced. As can be seen in Figure 11 the AWGN channel case is actually inferior to the BSC case even though it has a lower bit error rate. This has to do with the retransmission criterion. The BSC has accepted more packets after the first transmission and consequently has more correct packets, but also more bit errors. The bit errors are only visible in this figure as the small gap that prevents us from reaching $P_d \equiv 1$. A smaller amount of packets has been accepted for the AWGN channel case and hence a smaller amount of packets are correct. After the second and third transmission the AWGN channel case is superior again.

A series of small steps may be advantageous in many application compared to one large step, because at every step there is an increased probability that the packet is correct and retransmissions will cease. Also, a slight synchronization problem could result in the large step being taken just after the deadline. We typically want a new step just before the deadline - not directly after.

## 5. Conclusions

We have put forward a framework for transmitting real-time data over a radio channel based on the Quality of Service parameters deadline ($t_{DL}$) and probability for correct delivery before deadline ($P_d$). Initial studies show a high probability of delivering the correct data before deadline. We use a set of tools for handling information, such as, trellis representation of linear block codes, maximum-likelihood sequence detection, the Viterbi algorithm, hybrid ARQ and packet combining techniques. These methods are used in conjunction with the deadline dependent coding (DDC) scheme in order to maximize the probability of delivering a message before a given deadline. The required deadline and probability of correct delivery can be negotiated by the application, thus forming a flexible transmission mechanism. We believe that DDC is a way of making it possible to use radio channels in time- and safety critical applications.

## 6. References


[1] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*, John Wiley & Sons, NY, USA, 1965.

[2] H. Bengtsson, E. Uhlemann and P-A. Wiberg, "Protocol for wireless real-time systems", *Proc. of the 11th Euromicro Conf. Real Time Systems*, York, UK, June, 1999.

[3] W. Zhao, J. A. Stankovic, K Ramamritham, "A window protocol for transmission of time-constrained messages" *IEEE Trans. Computers*, Sept. 1990, pp 1186 -1203.

[4] W. Zhao, K Ramamritham: "Virtual time CSMA protocols for hard real-time communication", *IEEE Trans. Software Eng.*, Aug. 1987, pp. 938-952.

[5] G. Agrawal, B. Chen and W. Zhao, "Local synchronous capacity allocation for guaranteeing message deadlines with the time token protocol", *Proc. INFOCOM*, pp. 186-193, NJ 1993.

[6] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*, Prentice Hall Inc. 1995.

[7] S. Lin, T. Kasami, T. Fujiwara and M. Fossorier, *Trellises and Trellis-Based Decoding Algorithms for Linear Block Codes*, Kluwer Aca. Publ., 1998.

[8] L. R. Bahl, J. Cocke, F.Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Trans. Info. Th.*, pp. 284-287, March 1974.

[9] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes using a trellis", *IEEE Trans. Info. Th.*, pp.76-80, 1978.

[10] G. D. Forney, Jr., "The Viterbi algorithm", *Proc. IEEE*, vol. 61, pp. 268-278, 1973.

[11] S. Lin, D. J. Costello Jr. and M. J. Miller, "Automatic-repeat -request error control schemes", *IEEE Commun. Mag.*, pp. 5-16, Dec. 1984.

[12] J. M. Wozencraft and M. Horstein, *Coding for two-way channels*, Research Lab. Elec., MIT, MA, 1961. Tech. Rep. 383.

[13] S. B. Wicker, "High-reliability data transfer over land mobile radio channel using interleaved hybrid-ARQ error control". *IEEE Trans. Veh. Tech.*, pp. 48-55, February 1990.

[14] M. Schwartz and W. R. Bennett, *Communication Systems and Techniques*, McGraw-Hill, Inc. 1966.

[15] B. A. Harvey and S. B. Wicker, "Packet Combining Systems Based on the Viterbi Decoder", *IEEE Trans.Commun.*, pp. 1544 -1557, Feb./Mar./Apr. 1994.

[16] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications", *IEEE Trans.Commun.*, pp. 389-400, April 1988.

[17] G. Benelli, "An ARQ scheme with memory and sort error detectors", *IEEE Trans. Commun.*, pp. 285-288, March 1985.