Application and implementation of an embedded subband coder

J. Hartung
Steven L. Grant
Missouri University of Science and Technology, sgrant@mst.edu
R. V. Cox
R. D. Gitlin

Follow this and additional works at: http://scholarsmine.mst.edu/faculty_work
Part of the Electrical and Computer Engineering Commons

Recommended Citation
Hartung, J.; Grant, Steven L.; Cox, R. V.; and Gitlin, R. D., "Application and implementation of an embedded subband coder" (1988). Faculty Research & Creative Works. Paper 330.
http://scholarsmine.mst.edu/faculty_work/330

This Article is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Faculty Research & Creative Works by an authorized administrator of Scholars' Mine. For more information, please contact weaverjr@mst.edu.
APPLICATION AND IMPLEMENTATION OF AN EMBEDDED SUBBAND CODER

Richard V. Cox, Steven L. Gay, Richard D. Gitlin, and John Hartung
AT&T Bell Laboratories
Holmdel, N. J. 07733

ABSTRACT

This paper describes an embedded subband speech coder with dynamic bit allocation and post-filtering. The coder has been implemented on an AT&T WE® DSP32 Digital Signal Processor in real time, and the performance of the coder in a statistical multiplexer environment has been simulated. A technique for eliminating low rate tandem encodings in a digital network is also described. These techniques yield improved end-to-end performance in environments with multiple tandem encodings and/or statistical multiplexing.

INTRODUCTION

The purpose of this paper is two-fold. First, an embedded subband speech coder with dynamic bit allocation and post-filtering with application to a statistical multiplexer environment is discussed. Second, a technique for eliminating tandems of speech coders in networks with digital connectivity is introduced. These techniques yield improved end-to-end performance in environments with multiple tandem encodings and/or statistical multiplexing.

The earliest system to exploit the talking/silence statistics of speech was TASl (Time Assignment Speech Interpolation) [1], which achieved a two-to-one compression over a cross-section of 74 input channels. In this system, a fixed amount of bandwidth was allocated to talking segments of speech and zero bandwidth was allocated to silent segments. When the short-term load exceeded the channel capacity talking segments of speech would be clipped or chopped. In later systems, such as that described by Sciulli and Campanella [2], excess load was handled by reducing the bandwidth allocated to talking segments. This provided more graceful degradation of service in the excess load condition. In [3], Goodman introduced an embedded ADPCM speech coder which provided an improved scheme for the variable bandwidth allocation.

Derby and Galand [4] have shown that variable rate embedded subband coding can also be successfully applied in the area of statistical multiplexing of speech. This paper discusses an alternate embedded subband coder that uses the dynamic bit allocation technique proposed by Ramstad [5]. This allows the coded speech to be more easily embedded at rates that are multiples of the sampling rate divided by the number of subbands rather than merely multiples of the sampling rate. The principle advantage of this more finely structured embedded coder is that the average rate of a channel consisting of multiple tandems of these multiplexers is higher.

The embedded subband coder with dynamic bit allocation and post-filtering is reviewed in section one, while sections two and three discuss the multiplexer environment and the coder's performance in a simulation of this environment, respectively.

Statistical multiplexing and low-rate coding of speech provide efficient use of available bandwidth at the price of delay and distortion. Multiple tandem links tend to exacerbate these problems. In section four a technique that eliminates tandem encodings at digital switches is presented. The technique can be realized entirely within the speech codec and is completely independent of the digital switch.

1. DYNAMIC BIT ALLOCATED SUBBAND CODING

Subband coding algorithms can be considered in two parts: the filter bank and the quantizer. This coder also includes a third section -- the post-filter. The filter bank and quantizer are based on techniques proposed by Rothweiler [6] and Ramstad [5], respectively, while the post-filter borrows the ideas of Ramamoorthy and Jayant [9]. Each of these techniques has been modified for this coder and is described below.

1.1 The Filter Bank

The filter bank is based on Rothweiler's polyphase quadrature filters which allow an arbitrary number of equal sized bands. The design is based on a low-pass prototype FIR filter which has a cut-off frequency of half of the desired subband bandwidth. The subband filters are obtained by modulating the low-pass filter with pairs of differently phased sinusoids. We use an eight band filter bank based on a 64 tap prototype. At 8 KHz sampling, each subband is 500 Hz wide and is sampled at 1 KHz.

In this coder, only the lower six bands are used. The frequency content above 3 KHz is discarded. Thus, a small amount of high frequency aliasing will be generated in the transition band around 3 KHz. Since the energy of the speech at these frequencies is fairly low, this does not cause a noticeable amount of distortion when the speech is passed through the filter bank without quantization.
1.2 The Quantizer

The block diagram of the subband coder with dynamic bit allocation (DSBC) is shown in figure 1. The quantization scheme is based on a frame size of 16 ms. Each of the six subbands has 16 samples per frame. Thus, there are a total of 96 samples to quantize. A side information scheme of the type proposed by Ramstad [5] is used. The energy for each of the subbands is computed and quantized. There are 6 bands, and 5 bits are used for quantizing the energy of each band, totaling 30 bits of information. The quantizer reconstruction levels are proportional to the square root of the energies, which gives an estimate of the standard deviation for each of the bands. This estimate is available at both the transmitter and the receiver and is the basis for the quantization of the subband signals. Quantization of the side information is essentially logarithmic over a 72 dB range.

Bit allocation is derived from the quantized energies. At each iteration, 16 bits (one per subband sample) are allocated to one of the subbands. Each iteration consists of finding the subband with the largest rms value, halving this value, storing the result in an rms table, and allocating 16 bits to that subband. There is one additional constraint -- no band can be allocated more than a specified maximum, typically either 4 or 5 bits per sample. When a band is allocated the maximum number of bits, its rms value is set to zero, insuring that no more bits will be allocated to that subband. Each iteration represents the allocation of 2 bits per Hertz of the bandwidth of a subband which, for this particular implementation, corresponds to 1 KHz.

Cox, et al in [7] have presented empirical evidence that the proper prioritization of the bit allocation is crucial for achieving high quality speech. The value of G, which corresponds to greater post filtering. We found that a value of G=0.6 yields good results at 16 KHz. The value of G can be adjusted so that the overall output level of the filter sounds the same as that of the original speech. A G in the range of 0.6 to 0.75 was found to be satisfactory. While post-filtering reduces the high frequency "hiss" that is present in the DSBC coder at rates below 20 KHz.

1.3 Adaptive Post-filtering

DSBC produces an error signal that is nominally spectrally flat because the bit allocation algorithm attempts to make the quantization noise equal in all bands. Speech's spectrum, in general, is not flat, thus the signal to quantization noise ratio varies as a function of frequency. Post-filtering, introduced by Jayant and Ramamoorthy [9] for use in ADPCM coding, is used to de-emphasize those portions of the spectrum which contain low signal to quantization noise ratios and emphasize those portions which contain high signal to quantization noise ratios.

We wish the quantized bit stream to be embedded because embedded quantizers provide the ability to change to lower rates by simply dropping undesired bits from the end of the coded frame. To do this the quantizer of the subbands must be embedded and the bits of the quantized samples must be properly prioritized.

The bit allocation procedure is used to construct the prioritized bit stream. For a 32 KHz coder, 16 ms of speech sampled at 8 KHz is represented by 512 bits. These 512 bits are organized as 32 individual 16-bit words (see figure 2). At the beginning of the stream are the 30 bits of side information. These 30 bits are preceded by two additional bits (used by the multiplexer to pass signaling and rate information) which are bundled together into the first two 16-bit words. Each sample in each subband is normalized by its standard deviation estimate and quantized to five bits by the embedded subband quantizer. The subband that is allocated the first bit has the most significant bit of its quantized samples placed in the third 16-bit word. The subband that is allocated the next bit has the most significant bit of its quantized samples, that have not already been transmitted, placed in the fourth 16-bit word. This process continues until all 512 bits of the frame are filled. The full frame represents a 32 KHz speech coder. Lower rates may be obtained by simply dropping 16-bit words from the end of the frame. Each word dropped represents a coding rate reduction of 1 KHz.

There are two requirements for an embedded quantizer. The first is that the binary representation of the output levels must be embedded. Many numbering schemes will work, such as sign-magnitude, 2's complement, or a natural binary code starting with 0 as the lowest level and then increasing by 1 for each new level. The second requirement is that the input thresholds for the n+1 bit quantizer must include the input thresholds and output values of the n bit quantizer. We can begin with any size non-uniform quantizers and from it build a family of embedded quantizers. Table 1 shows a family of non-uniform quantizers based on the 1 bit Max quantizer [8] for a normal distribution. In this case, the threshold levels for the 2 bit quantizer had to be -0.798, 0 and +0.798. Using Max's equations the output levels were then computed to be optimal for a normal distribution. Those output levels were then combined with the input thresholds to form the set of thresholds for the 3 bit quantizer. This procedure was repeated to obtain the 4 and 5 bit quantizers.

3.5.2.
It is important to note that post-filtering cannot be used in environments where multiple tandem encodings may occur. This is because repeated application of the post-filter is similar to using a very large g. Informal listening tests show that quality drops off dramatically after more than one tandem encoding of the algorithm with post-filtering.

1.4 Real-Time Implementation

The DSBC algorithm was implemented in real time on a 25 MHz AT&T DSP32 digital signal processor. The analysis and synthesis filter banks required 33% of the available real-time, while the quantization and dequantization required 25%, and the post filtering, 25%. Thus, 17% of the real-time remains for other tasks such as silence detection.

2. THE MULTIPLEXER ENVIRONMENT

Figure 3 is a block diagram of a statistical multiplexer. It is similar to that described in [4] and is a five-to-one compression statistical multiplexer where 120 channels coded with 64 Kbps mu-law or A-law PCM are compressed into a 1.536 Mbps bit stream. The multiplexer consists of 120 line interfaces each consisting of a single forty pin 25 MHz DSP32 within which a speech coder and silence detector are implemented. Each line interface is also equipped with an echo canceler because the round trip delay through the DSBC algorithm and the multiplexer is sufficiently long for reflections due to impedance mismatches at four-to-two wire interfaces to be perceived as echo. These line interfaces are connected to a central processor through the parallel ports of the DSP32s.

The central processor provides timing and synchronization to each of the 120 channels. All channels begin the 16 ms analysis frame at the same time. In the transmit direction, the central processor polls the line interfaces to determine the number and location of the talking channels. This information is used to compute the rate of each channel. If embedded coding is used, then the central processor must indicate to the talking channels the rate at which they will code their frames. This causes additional delay in the system since each talking channel must wait to be assigned a coding rate before proceeding. It embedded coding is used, then each channel codes the speech frame at the 32 Kbps rate, and once the central processor completes the rate calculations it merely reads the amount of data it requires from each line interface through the DSP32 parallel ports. Hence, the use of embedded coding reduces delay and simplifies the communications between the line interfaces and the central processor.

In the receive direction, the central processor computes the rate of each channel, then writes those rates to each of the line cards prior to sending the coded speech. The line cards decode the speech and send it to the connecting channels.

Speech frames that are found to be "silent" are coded at the reduced rate of 8 Kbps. This allows background noise to be transmitted during "silent" portions of speech while allowing a larger portion of the channel's bandwidth to be used by talking channels. Thus, the talking channels will normally operate above their lowest rate of 12 Kbps. In addition to the advantage of transmitting background noise, the 8 Kbps "silence" transmission rate is useful in lessening the impact of an incorrect decision by the silence detector.

Let B denote the total bandwidth available to the multiplexer (i.e. 1.536 Mbps), S be the rate at which silence is coded (8 Kbps), and R\(a\) be the average rate at which talking segments are coded (to be determined each frame). Also, let \(N_t\) and \(N_s\) be the number of talking channels and the number of silent channels respectively. Then,

\[ R_a = (B - SN_s)/N_t. \]

We denote the rates at which talking segments can be coded as \(R_i\). Here \(R_i\) is a member of the set \{12Kbps, 13Kbps, ..., 32Kbps\}. So, \(R_j < R_a \leq R_h\) where \(R_j\) and \(R_h\) are the lowest and highest permissible talking rates, respectively. In this case, \(R_j = 12\) Kbps and \(R_h = 32\) Kbps. The parameters B and S that have been selected guarantee that \(R_j \leq R_a \leq R_h\) for any combination of \(N_t\) and \(N_s\).

If \(R_a\) falls between two consecutive permissible rates \(R_j\) and \(R_h\) then the average talking rate \(R_t\) can be obtained by allowing an integer number of channels, \(N_i\), to operate at rate, \(R_j\), and an integer number of channels, \(N_h\), to operate at rate, \(R_h\).

\[ N_i = \frac{(B - SN_s)}{R_h - R_h}, \]

and

\[ N_h = N_t - N_i. \]

The state of each channel, whether talking or silent, must be sent over the channel. This is all the side information associated with the rate of the speech coding that is necessary to transmit over the channel since B and S are known a priori. The process of selecting which talking channels will operate at rates \(R_i\) and \(R_h\) can be determined by any number of rules (e.g. random selection) as long as both rules are available and operating synchronously at both the transmitting and receiving central processors.

3. MULTIPLEXER ENVIRONMENT SIMULATION

The performance of the speech coding algorithm in a fully loaded statistical multiplexer environment was studied by generating a table of coding rates at frame intervals from a monte-carlo experiment where talking and silence durations for the 120 voice channels were modeled with exponentially distributed statistics. This table was then used to provide coding rates for the real-time implementation of the algorithm, thus, providing a simulation of the algorithm's performance in a multiplexer environment in real-time.

The mean duration of talking and silent segments is a function of the sensitivity of the silence detector. If the silence detector used in [10] is used then, the measured mean talking duration is 400 ms and the speech activity factor is 36%. Here, to be conservative, we use a 500 ms talking duration and a 40% speech activity factor.

The real-time experiment did not include a silence detector and so the talking/silence statistics of this one channel was not taken into account in the rate calculation. Since this was only one of 120 channels the statistics were not significantly affected.
Figure 4 shows the histogram of rates for one channel where variable rate coding was performed at 1 Kbps and 8 Kbps intervals. While the average rate of these two schemes are both about 20 Kbps, the 1 Kbps interval scheme produced a segmental SNR of 18.74 dB while the 8 Kbps interval scheme produced a segmental SNR of 18.26 dB. In addition, informal listening tests show a slight advantage for the more finely structured embedded codec. However, the real difference between these two schemes shows up in multiple tandems of the multiplexer where the resulting end-to-end rate for a given frame and channel is the lowest rate allocated at any one tandem. Figure 5 shows the mean end-to-end rate for a given channel as a function of the number of tandems. The plot is based on the histograms of figure four and the assumption that the talking/silence statistics of each multiplexed link are independent. After only two tandems the difference in the mean end-to-end rate is already 1.2 Kbps. This increases to 1.8 Kbps after three tandems. Thus, there is a distinct advantage in using more finely structured embedded speech coders.

4. A TANDEM CODING BYPASS TECHNIQUE

Low rate speech coding introduces both distortion and delay to the encoded signal. Degradation at each tandem encoding using the 32 Kbps ADPCM coder limits its application to networks with less than four tandem encodings. Coders which operate at, or below, 16 Kbps also introduce significant delay (the processing frame length) to improve coding quality by removing additional temporal redundancy. Both the additional distortion and delay introduced by these coders imply additional constraints on the network configuration. The technique described in this section avoids tandeming of low rate speech coders in digital switches where there is digital connectivity between speech codecs. The technique can be completely contained within the speech codec and is transparent to any digital equipment (such as PBXs) between codecs. The idea is to surround the low rate speech frame with place-holder bits to form a 64 Kbps frame. These place-holder bits are either stripped off during compression or added during decompression. In either case, the low rate speech is left intact.

Figure 6 is a block diagram of a possible network configuration. It consists of three digital switches connected by two T-1 lines. Switch A is tied directly to switch B and switch B is tied directly to switch C. User 1 on switch A must be switched through switch B to speak to user 2 on switch C. If the multiplexers use low rate speech coders, then with the conventional approach, user 1’s voice is encoded and decoded twice. We would like to avoid the degradation in speech quality inherent in this approach without increasing the complexity of the switch or noticeably impairing the decoded speech.

In a purely digital environment a speech codec can be viewed as consisting of a decompressor which has a low rate input and a 64 Kbps output and a compressor which has a 64 Kbps input and a low rate output (see figure 7). The decompressor can be enabled to either decode its low rate input or to simply zero pad it up to 64 Kbps. Similarly, the compressor can be enabled to either encode the 64 Kbps mu-law PCM coded stream to the low rate or strip the padding bits from the 64 Kbps padded input stream.

The two directions of the codec will always operate in complementary states, that is, the codec can be either decoding and encoding or padding and stripping. It switches between these states by detecting different synchronization patterns. When the decompressor is decoding, it periodically "robs" the least significant bit (LSB) of the output mu-law sample and replaces it with a distinct synchronization pattern. This process will not significantly impair the quality of the speech since the low-rate speech coding/decoding will most likely render the LSB insignificant. The compressor always searches the LSB’s of the input 64 Kbps mu-law coded stream for this synchronization pattern. If it detects the pattern, it temporarily squelches the compressed signal and enables the decompressor to begin padding rather than decoding.

When the decompressor is padding the low-rate frame, a new synchronization pattern is inserted into one of the non-used bits. When the far end compressor detects this new synchronization pattern it disables the squelch on its compressor and begins stripping the padded bits.

There are several ways to increase the rate of the low rate bit stream to 64 Kbps. For coding rates that are multiples of 8 Kbps the low-rate signal can be placed into the least significant bits of what would ordinarily be the mu-law word of the 64 Kbps stream. For the case of 16 Kbps speech coding, only the two LSBs of the mu-law word would contain the coded speech. This insures that, if by chance, the system should treat this signal as mu-law coded speech, only a low level noise signal would be observed by a human listener. For rates other than multiples of 8 Kbps the best approach is to burst the coded frame across the switch at the 64 Kbps rate once per frame.

For the case where the multiplexers between switches A and B and switches B and C are both statistical, the end-to-end rate will be the lowest rate allocated by either of the two multiplexers.

In the terminating codecs (i.e. at codecs one and four in figure 6) the compressor does not find the robbed bit synchronization pattern in the 64 Kbps stream, and so the codec will never proceed to the pad and strip state. The decompressors of codecs 1 and 4 will continue to insert the robbed bit synchronization pattern into the 64 Kbps stream going to user 1 and user 2, respectively. The degradation in the speech quality due to this bit robbing is minimal, and actually could be disabled after a short decision interval at the beginning of a call.

The salient features of this by-pass scheme are:

1) Speech codec delay and quality degradations are eliminated in digital tandem switches.

2) No interaction with switch control (i.e. a PBX) is required.

3) Proper compression/decompression technique on tandem connections and local terminations is automatically selected.

4) Digital connectivity is required and the technique is not applicable to analog switches.

5. CONCLUSIONS

We have presented a subband coding algorithm that uses dynamic bit allocation and post filtering. The bit allocation procedure was used to easily embed the coder at rates that were multiples of twice the subband bandwidth rather than
simply multiples of the sampling rate. We have shown that this finer embedded structure results in higher average end-to-end rates for channels consisting of multiple tandems of statistical multiplexers. In addition a technique that eliminates tandem encodings at digital switches was presented.

6. ACKNOWLEDGMENTS

The authors gratefully acknowledge the assistance of J. G. Kneuer and J. W. Sawyer of AT&T Communications and Information Systems.

References:


Each Small Block Represents 16 Bits

Fig. 2 32 kb/s Embedded DSB-C Frame

Figure 4. A Comparison of Histograms of Rates for Coders Embedded with Steps of 1 Kb/s and 8 Kb/s

Number of Tandem Links

Average End-to-End Rate (Kbs)

16 17 18 19 20 21 22 23 24 25

16 17 18 19 20

Figure 5.

Fig. 1 Embedded Subband Coder with Dynamic Bit Allocation
LI : line interface
CP : central processor

Figure 3a. Statistical Multiplexer

Figure 3b. Line Interface

Table 1: Positive Input Thresholds & Output Levels for Embedded Quantizers Optimized for 1 Bit Performance

<table>
<thead>
<tr>
<th>Quantizer Size (Bits)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.053</td>
<td>1.200</td>
<td>1.125</td>
<td>1.053</td>
<td>0.921</td>
</tr>
<tr>
<td>2</td>
<td>1.366</td>
<td>1.366</td>
<td>1.366</td>
<td>1.366</td>
<td>1.366</td>
</tr>
<tr>
<td>3</td>
<td>1.825</td>
<td>1.825</td>
<td>1.825</td>
<td>1.825</td>
<td>1.825</td>
</tr>
<tr>
<td>4</td>
<td>2.219</td>
<td>2.219</td>
<td>2.219</td>
<td>2.219</td>
<td>2.219</td>
</tr>
<tr>
<td>5</td>
<td>2.569</td>
<td>2.569</td>
<td>2.569</td>
<td>2.569</td>
<td>2.569</td>
</tr>
</tbody>
</table>

Example: If a 3 bit quantizer is used, a sample 0.95 falls between 0.798 and 1.366 and so is quantized to 1.053.