Differential Encoding

Related terms:

Image Compression, Vector Quantization, Amplifier, Compression Scheme, Difference Sequence, Encoding Scheme, Phase Difference, Quantization Error, Quantization Noise

View all Topics

Differential Encoding

Khalid Sayood, in Introduction to Data Compression (Fourth Edition), 2012

11.7 Speech Coding

Differential encoding schemes are immensely popular for speech encoding. They are used in the telephone system, voice messaging, and multimedia applications, among others. Adaptive DPCM is a part of several international standards (ITU-T G.721, ITU G.723, ITU G.726, ITU-T G.722), which we will look at here and in later chapters.

Before we do that, let's take a look at one issue specific to speech coding. In Figure 11.7, we see that there is a segment of speech that looks highly periodic. We can see this periodicity if we plot the autocorrelation function of the speech segment (Figure 11.15).

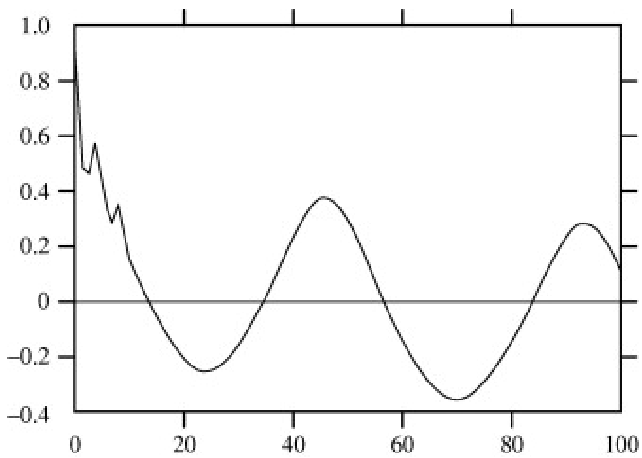


Figure 11.15. Autocorrelation function for test.snd.

Figure 11.15. Autocorrelation function for test.snd.

The autocorrelation peaks at a lag value of 47 and multiples of 47. This indicates a periodicity of 47 samples. This period is called the *pitch period*. The predictor we originally designed did not take advantage of this periodicity, as the largest predictor was a third-order predictor, and this periodic structure takes 47 samples to show up. We can take advantage of this periodicity by constructing an outer prediction loop around the basic DPCM structure as shown in Figure 11.16. This can be a simple single coefficient predictor of the form, where is the pitch period. Using this system on testm.raw, we get the residual sequence shown in Figure 11.17. Notice the decrease in amplitude in the periodic portion of the speech.

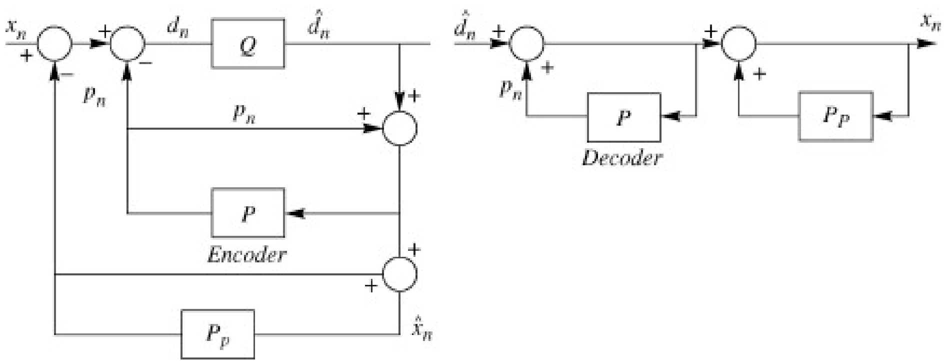


Figure 11.16. The DPCM structure with a pitch predictor.

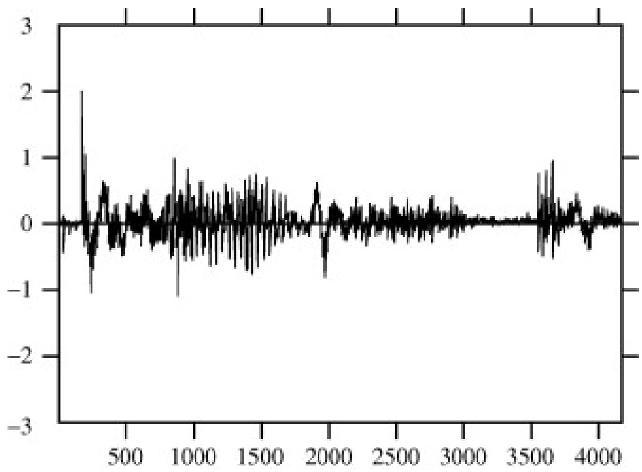


Figure 11.17. The residual sequence using the DPCM system with a pitch predictor.

Finally, remember that we have been using mean squared error as the distortion measure in all of our discussions. However, perceptual tests do not always correlate with the mean squared error. The level of distortion we perceive is often related to the level of the speech signal. In regions where the speech signal is of higher amplitude, we have a harder time perceiving the distortion, but the same amount of distortion in a different frequency band, where the speech is of lower amplitude, might be very perceptible. We can take advantage of this by shaping the quantization error so that most of the error lies in the region where the signal has a higher amplitude. This variation of DPCM is called *noise feedback coding* (NFC) (see [134] for details).

11.7.1 G.726

11.7.1 G.726

The International Telecommunications Union has published several recommendations for a standardized ADPCM system, including recommendations G.721, G.723, and G.726. G.726 supersedes G.721 and G.723. In this section we will describe the G.726 recommendation for ADPCM systems at rates of 40, 32, 24, and 16 kbits.

The Quantizer

The recommendation assumes that the speech output is sampled at the rate of 8000 samples per second, so the rates of 40, 32, 24, and 16 kbits per second translate 5 bits per sample, 4 bits per sample, 3 bits per sample, and 2 bits per sample. Comparing this to the PCM rate of 8 bits per sample, this would mean compression ratios of 1.6:1, 2:1, 2.67:1, and 4:1. Except for the 16 kbits per second system, the number of levels in the quantizer are , where is the number of bits per sample. Thus, the number of levels in the quantizer is odd, which means that for the higher rates we use a midtread quantizer.

The quantizer is a backward adaptive quantizer with an adaptation algorithm that is similar to the Jayant quantizer. The recommendation describes the adaptation of the quantization interval in terms of the adaptation of a scale factor. The input is normalized by a scale factor. This normalized value is quantized, and the normalization removed by multiplying with . In this way the quantizer is kept fixed and is adapted to the input. Therefore, for example, instead of expanding the step size, we would increase the value of .

The fixed quantizer is a nonuniform midtread quantizer. The recommendation describes the quantization boundaries and reconstruction values in terms of the log of the scaled input. The input-output characteristics for the 24 kbit system are shown in Table 11.2. An output value of in the table corresponds to a reconstruction value of 0.

Table 11.2. Recommended input-output characteristics of the quantizer for 24-kbits-per-second operation.

Input Range	Label	Output
	3	2.91
	2	2.13
	1	1.05
	0	

The adaptation algorithm is described in terms of the logarithm of the scale factor:

(60)

The adaptation of the scale factor of the log depends on whether the input is speech or speechlike, where the sample-to-sample differences can fluctuate considerably, or whether the input is voice-band data, which might be generated by a modem, where the sample-to-sample fluctuation is quite small. In order to handle both these situations, the scale factor is composed of two values, a *locked* slow scale factor for when the sample-to-sample differences are quite small, and an *unlocked* value for when the input is more dynamic:

(61)

The value of depends on the variance of the input. It will be close to one for speech inputs and close to zero for tones and voice-band data.

The unlocked scale factor is adapted using the Jayant algorithm with one slight modification. If we were to use the Jayant algorithm, the unlocked scale factor could be adapted as

(62)

where is the multiplier. In terms of logarithms, this becomes

(63)

The modification consists of introducing some memory into the adaptive process so that the encoder and decoder converge following transmission errors:

(64)

where , and .

The locked scale factor is obtained from the unlocked scale factor through

(65)

The Predictor

The recommended predictor is a backward adaptive predictor that uses a linear combination of the past two reconstructed values as well as the six past quantized differences to generate the prediction:

(66)

The set of predictor coefficients is updated using a simplified form of the LMS algorithm:

(67)

(68)

where

(69)

(70)

The coefficients are updated using the following equation:

(71)

Notice that in the adaptive algorithms we have replaced products of reconstructed values and products of quantizer outputs with products of their signs. This is computationally much simpler and does not lead to any significant degradation of the adaptation process. Furthermore, the values of the coefficients are selected such that multiplication with these coefficients can be accomplished using shifts and adds. The predictor coefficients are all set to zero when the input moves from tones to speech.

> Read full chapter

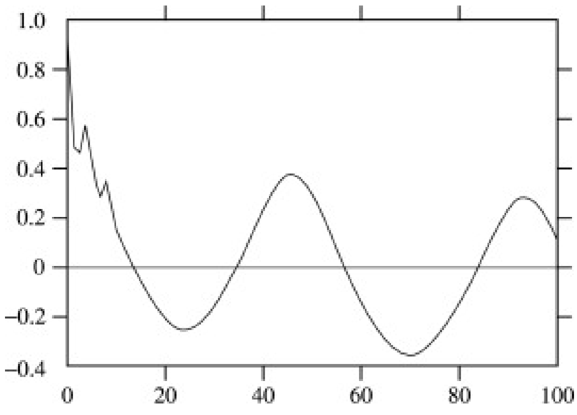
Differential Encoding

Khalid Sayood, in Introduction to Data Compression (Fifth Edition), 2018

11.7 Speech Coding

Differential encoding schemes are immensely popular for speech encoding. They are used in the telephone system, voice messaging, and multimedia applications, among others. Adaptive DPCM is a part of several international standards (ITU-T G.721, ITU G.723, ITU G.726, ITU-T G.722), which we will look at here and in later chapters.

Before we do that, let's take a look at one issue specific to speech coding. In Fig. 11.7, we see that there is a segment of speech that looks highly periodic. We can see this periodicity if we plot the autocorrelation function of the speech segment (Fig. 11.15).



1.0
0.8
0.6
0.4
0.2
0
−0.2
−0.4
0
20
40
60
80
100

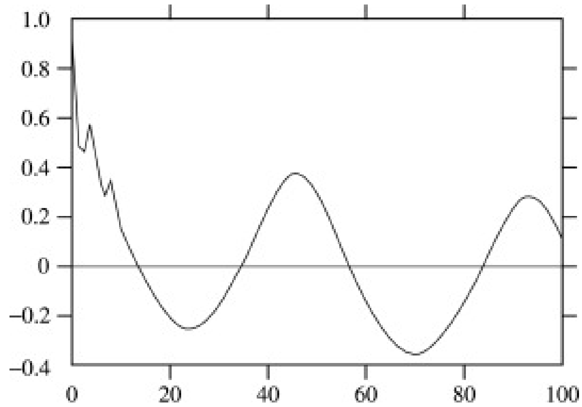


Figure 11.15. Autocorrelation function for test.snd.

The autocorrelation peaks at a lag value of 47 and multiples of 47. This indicates a periodicity of 47 samples. This period is called the *pitch period*. The predictor we originally designed did not take advantage of this periodicity, as the largest predictor was a third-order predictor, and this periodic structure takes 47 samples to show up. We can take advantage of this periodicity by constructing an outer prediction loop around the basic DPCM structure as shown in Fig. 11.16. This can be a simple single coefficient predictor of the form , where P is the pitch period. Using this system on testm.raw, we get the residual sequence shown in Fig. 11.17. Notice the decrease in amplitude in the periodic portion of the speech.

Figure 11.16. The DPCM structure with a pitch predictor.

Figure 11.17. The residual sequence using the DPCM system with a pitch predictor.

Finally, remember that we have been using mean squared error as the distortion measure in all of our discussions. However, perceptual tests do not always correlate with the mean squared error. The level of distortion we perceive is often related to the level of the speech signal. In regions where the speech signal is of higher amplitude, we have a harder time perceiving the distortion, but the same amount of distortion in a different frequency band, where the speech is of lower amplitude, might be very perceptible. We can take advantage of this by shaping the quantization error so that most of the error lies in the region where the signal has a higher amplitude. This variation of DPCM is called *noise feedback coding* (NFC) (see [156] for details).

level of the speech signal. In regions where the speech signal is of higher amplitude, we have a harder time perceiving the distortion, but the same amount of distortion in a different frequency band, where the speech is of lower amplitude, might be very perceptible. We can take advantage of this by shaping the quantization error so that most of the error lies in the region where the signal has a higher amplitude. This variation of DPCM is called *noise feedback coding* (NFC) (see [156] for details).

11.7.1 G.726

The International Telecommunications Union has published recommendations for a standard ADPCM system, including recommendations G.721, G.723, and G.726. G.726 supersedes G.721 and G.723. In this section we will describe the G.726 recommendation for ADPCM systems at rates of 40, 32, 24, and 16 kbits.

The Quantizer

The recommendation assumes that the speech output is sampled at the rate of 8000 samples per second, so the rates of 40, 32, 24, and 16 kbits per second translate 5 bits per sample, 4 bits per sample, 3 bits per sample, and 2 bits per sample. Comparing this to the PCM rate of 8 bits per sample, this would mean compression ratios of 1.6:1, 2:1, 2.67:1, and 4:1. Except for the 16 kbits per second system, the number of levels in the quantizer are , where is the number of bits per sample. Thus, the number of levels in the quantizer is odd, which means that for the higher rates we use a midtread quantizer.

The quantizer is a backward adaptive quantizer with an adaptation algorithm that is similar to the Jayant quantizer. The recommendation describes the adaptation of the quantization interval in terms of the adaptation of a scale factor. The input is normalized by a scale factor . This normalized value is quantized, and the normalization removed by multiplying with . In this way the quantizer is kept fixed and is adapted to the input. Therefore, for example, instead of expanding the step size, we would increase the value of .

The fixed quantizer is a nonuniform midtread quantizer. The recommendation describes the quantization boundaries and reconstruction values in terms of the log of the scaled input. The input-output characteristics for the 24 kbit system are shown in Table 11.2. An output value of −∞ in the table corresponds to a reconstruction value of 0.

Table 11.2. Recommended Input–Output Characteristics of the Quantizer for 24-kbits-per-Second Operation

Input Range	Label	Output	Label

	$\lvert I_k \rvert$		$\lvert I_k \rvert$
[2.58,∞)	3	2.91	3
[1.70,2.58)	2	2.13	2
[0.06,1.70)	1	1.05	1
(−∞,−0.06)	0	−∞	0

The adaptation algorithm is described in terms of the logarithm of the scale factor

(11.60)

The adaptation of the scale factor or its log depends on whether the input is speech or speechlike, where the sample-to-sample difference can fluctuate considerably, or whether the input is voice-band data, which might be generated by a modem, where the sample-to-sample fluctuation is quite small. In order to handle both these situations, the scale factor is composed of two values, a *locked* slow scale factor for when the sample-to-sample differences are quite small, and an *unlocked* value for when the input is more dynamic:

(11.61)

The value of depends on the variance of the input. It will be close to one for speech inputs and close to zero for tones and voice band data.

The unlocked scale factor is adapted using the Jayant algorithm with one slight modification. If we were to use the Jayant algorithm, the unlocked scale factor could be adapted as

(11.62)

where is the multiplier. In terms of logarithms, this becomes

(11.63)

The modification consists of introducing some memory into the adaptive process so that the encoder and decoder converge following transmission errors:

(11.64)

where , and .

The locked scale factor is obtained from the unlocked scale factor through

(11.65)

The Predictor

The recommended predictor is a backward adaptive predictor that uses a linear combination of the past two reconstructed values as well as the six past quantized differences to generate the prediction

The recommended predictor is a backward adaptive predictor that uses a linear combination of the past two reconstructed values as well as the six past quantized differences to generate the prediction

(11.66)

The set of predictor coefficients is updated using a simplified form of the LMS algorithm.

(11.67)

(11.68)

where

(11.69)

(11.70)

The coefficients are updated using the following equation:

(11.71)

Notice that in the adaptive algorithms we have replaced products of reconstructed values and products of quantizer outputs with products of their signs. This is computationally much simpler and does not lead to any significant degradation of the adaptation process. Furthermore, the values of the coefficients are selected such that multiplication with these coefficients can be accomplished using shifts and adds. The predictor coefficients are all set to zero when the input moves from tones to speech.

> Read full chapter

Entropy, Predictive Coding and Quantization

Suhel Dhanani, Michael Parker, in Digital Video Processing for Engineers, 2013

11.5 Differential Encoding

Suppose we take the actual pixel value and subtract the predicted pixel value. This is differential encoding, and this difference is used to represent the pixel value for that location. For example, to store a video frame we would perform the differential encoding process, and store the results.

that location. For example, to store a video frame we would perform the differential encoding process, and store the results.

To restore the original video data for display, we would simply need to compute the predicted pixel value (using the previous pixel data) and then add to the stored differential encoded value. This is the original pixel data. (The three pixels on the upper left corner of the frame are stored in their original representation, and are used to create the initial predicted pixel data during the restoration process).

All of this sounds unnecessarily complicated, so what is the purpose here? The differential encoder outputs are most likely to be much smaller values than the original pixel data, due to the correlation between nearby pixels. Since statistically the differentially encoded data is just representing the errors of the predictor, this signal is likely to have the great majority of values concentrated around zero, or to have a very compact histogram. In contrast, the original pixel values are likely to span the entire color value space, and therefore are a high entropy data source with an equal or uniform probability distribution.

Differential encoding sends less information than if the video frame was simply sent pixel by pixel across the whole frame. The entropy of the data stream has been reduced through use of differential encoding, as the correlation between adjacent pixels has been largely eliminated. Without differential encoding, we have no information as to the type of video frames that will be processed, the initial pixel value outcomes are assumed to be equally distributed, meaning each of the 256 possible values has a probability of 1/256, and an entropy of eight bits per pixel. The possible outcomes of the differential encoder tend to be much more probable for small values (due to the correlation to nearby pixels) and much less likely for larger values. As we saw in our simple example above, when the probabilities are not evenly distributed, the entropy is lower. Lower entropy means less information. Therefore, on average, significantly fewer bits are required to represent the image, and the only cost is increased complexity due to the predictor and differential encoding computations.

Let's assume that averaged over a complete video frame, the probabilities work out thus:

Pixel color value Probability = 1/256 for values in range of 0 to 255

And that after differential encoding, the probability distribution comes out as:

Differential color	**for value equal to 0**	**Probability = 1/16**
Differential color	for values in the range −8 to −1, 1 to 8	Probability = 1/25
Differential color	for values in the range −32 to −9, 9 to 32	Probability = 1/400
Differential color		Probability = 1/5575

composing a signal into four bands would result in four signals instead of one. Fortunately, the Nyquist's sampling theorem (discussed in Chapter 12) comes to our aid allowing us to subsample the signals in each subband. The subsampling process known as *downsampling* and its counterpart *upsampling*, which is necessary to re-compose the signal, are simple processes in practice. However, their mathematical representation can be bit complicated. For those interested in the theory we have included it in starred sections for which a review of z-transforms from Chapter 12 is highly recommended. We have also included starred sections which describe some of the theory behind the design of practical perfect reconstruction filters. While not essential, for those so inclined this part of the chapter can be a window into the fascinating world of digital signal processing.

composing a signal into four bands would result in four signals instead of one. Fortunately, the Nyquist's sampling theorem (discussed in Chapter 12) comes to our aid allowing us to subsample the signals in each subband. The subsampling process known as *downsampling* and its counterpart *upsampling*, which is necessary to recompose the signal, are simple processes in practice. However, their mathematical representation can be bit complicated. For those interested in the theory we have included it in starred sections for which a review of z-transforms from Chapter 12 is highly recommended. We have also included starred sections which describe some of the theory behind the design of practical perfect reconstruction filters. While not essential, for those so inclined this part of the chapter can be a window into the fascinating world of digital signal processing.

Finally, we look at applications of subband coding to speech, audio, and image compression. The application to audio compression—the popular *mp3* algorithm—is only mentioned as we will return to it in Chapter 17.

> Read full chapter

Subband Coding

Khalid Sayood, in Introduction to Data Compression (Fourth Edition), 2012

14.12.2 Coding the Subbands

Once we have decomposed an image into subbands, we need to find the best encoding scheme to use with each subband. The coding schemes we have studied to date are scalar quantization, vector quantization, and differential encoding. Let us encode some of the decomposed images from the previous section using two of the coding schemes we have studied earlier, scalar quantization and differential encoding.

Example 14.12.3

In the previous example we noted the fact that the eight-tap Johnston filter did not compact the energy as well as the 16-tap Johnston filter or the eight-tap Smith-Barnwell filter. Let's see how this affects the encoding of the decomposed images.

When we encode these images at an average rate of 0.5 bits per pixel, there are bits available to encode four values, one value from each of the four subbands. If we use the recursive bit allocation procedure on the eight-tap Johnston filter outputs, we end up allocating 1 bit to the low-low band and 1 bit to the high-low band. As the pixel-to-pixel difference in the low-low band is quite small, we use a DPCM encoder for the low-low band. The high-low band does not show this behavior, which means we can simply use scalar quantization for the high-low band. As there are no bits available to encode the other two bands, these bands can be discarded. This results in the image shown in Figure 14.32, which is far from pleasing. However, if we use the same compression approach with the image decomposed using the eight-tap Smith-Barnwell filter, the result is Figure 14.33, which is much more pleasing.

pixel-to-pixel difference in the low-low band is quite small, we use a DPCM encoder for the low-low band. The high-low band does not show this behavior, which means we can simply use scalar quantization for the high-low band. As there are no bits available to encode the other two bands, these bands can be discarded. This results in the image shown in Figure 14.32, which is far from pleasing. However, if we use the same compression approach with the image decomposed using the eight-tap Smith-Barnwell filter, the result is Figure 14.33, which is much more pleasing.

Figure 14.32. Sinan image coded at 0.5 bits per pixel using the eight-tap Johnston filter.

Figure 14.33. Sinan image coded at 0.5 bits per pixel using the eight-tap Smith-Barnwell filter.

To understand why we get such different results from using the two filters, we need to look at the way the bits were allocated to the different bands. In this implementation, we used the recursive bit allocation algorithm. In the image decomposed using the Johnston filter, there was significant energy in the high-low band. The algorithm allocated 1 bit to the low-low band and 1 bit to the high-low band. This resulted in poor encoding for both, and subsequently poor reconstruction. There was very little signal content in any of the bands other than the low-low band for the image decomposed using the Smith-Barnwell filter. Therefore, the bit allocation algorithm assigned both bits to the low-low band, which provided a reasonable reconstruction.

in poor encoding for both, and subsequently poor reconstruction. There was very little signal content in any of the bands other than the low-low band for the image decomposed using the Smith-Barnwell filter. Therefore, the bit allocation algorithm assigned both bits to the low-low band, which provided a reasonable reconstruction.

If the problem with the encoding of the image decomposed by the Johnston filter is an insufficient number of bits for encoding the low-low band, why not simply assign both bits to the low-low band? The problem is that the bit allocation scheme assigned a bit to the high-low band because there was a significant amount of information in that band. If both bits were assigned to the low-low band, we would have no bits left for use in encoding the high-low band, and we would end up throwing away information necessary for the reconstruction. □

The issue of energy compaction becomes a very important factor in reconstruction quality. Filters that allow for more energy compaction permit the allocation of bits to a smaller number of subbands. This in turn results in a better reconstruction.

The coding schemes used in this example were DPCM and scalar quantization, the techniques generally preferred in subband coding. The advantage provided by subband coding is readily apparent if we compare the results shown in Figure 14.33 to results in the previous chapters where we used either DPCM or scalar quantization without prior decomposition.

It would appear that the subband approach lends itself naturally to vector quantization. After decomposing an image into subbands, we could design separate codebooks for each subband to reflect the characteristics of that particular subband. The only problem with this idea is that the low-low subband generally requires a large number of bits per pixel. As we mentioned in Chapter 10, it is generally not feasible to operate the nonstructured vector quantizers at high rates. Therefore, when vector quantizers are used, they are generally used only for coding the higher frequency bands. This may change as vector quantization algorithms that operate at higher rates are developed.

> Read full chapter

Demonstration of Successful LEO/GEO/Small Satellites/UAV/Airplanes to Ground Optical Communication Experiments

Arun K. Majumdar, in Optical Wireless Communications for Broadband Global Internet Connectivity, 2019

6.3.1.2.4 Differential Phase Keying Modulation for Free-Space Optical Communication Link: Bit Error Rate in Atmospheric Turbulence

6.3.1.2.4 Differential Phase Keying Modulation for Free-Space Optical Communication Link: Bit Error Rate in Atmospheric Turbulence

The DPSK can be viewed as the noncoherent version of the phase-shift keying. It eliminates the need for a coherent reference signal at the receiver by combining two basic operations at the transmitter: (1) differential encoding of the input binary wave, and (2) phase-shift keying. DPSK has gained considerable interest for FSO communications due to a □3 dB sensitivity improvement over the popular OOK, and reduced peak power. Two of the symbols are represented by the relative phase between two differentially encoded bits or by the binary information: a 0 represented by no-phase change and a 1 by a phase difference (or vice versa). The receiver is equipped with a storage capability so that it can measure the relative phase difference between the waveforms received during two successive bit intervals. The phase difference between waveforms received in two successive bit intervals will be independent of the unknown phase □, if □ contained in the received wave varies slowly (slow enough for it to be considered essentially constant over 2 bit intervals). DPSK is another example of noncoherent modulation when it is considered over 2 bit intervals. The average probability of error or BER for DPSK is given by

(6.11)

where E_b and N_0 are the signal and noise energy per bit, respectively, and N_{avg} is the SNR, equal to the average number of photons per bit; that is, $N_{avg} = SNR_{DPSK}$.

In the case of atmospheric scintillation, the unconditional BER is obtained by averaging Eq. (6.11) over the irradiance fluctuation statistics given by P(I). The BER result in presence of atmospheric turbulence is given by [22]

(6.12)

where $P(I)$ can be assumed as gamma-gamma distribution discussed earlier in this chapter. The BER can be computed from Eq. (6.12) by a numerical solution where $P(I)$ can be assumed as gamma-gamma distribution discussed earlier in this section.

For the LCRD project, the data rates for PPM will be up to 311 Mb/s uplink and downlink, and DPSK up to 1.244 Gb/s (2.880 Gb/s coded) uplink and downlink. The ground segment for the operation of this mission will have two ground stations and the mission operations center to schedule, command, and control the communication payload and the two ground stations. Uplink beacon beams from the ground stations to the GEO terminal will help to point toward correct locations on Earth. One of the ground stations in Southern California will be using the JPL Optical Communications Telescope Laboratory equipped with AO systems as described before for the LEO satellite and OGS communication links. The AO system will be needed to correct for the atmospheric turbulence-induced distortions of the optical signals close to the earth. The other ground station in Hawaii will be developed by MIT/LL. The ultimate goal of this LCRD will be to prove the development of the potential future optical service provider in space. The optical links in the future will even be able to support between any two user remote platforms to establish global Internet connectivity. The ATP operations will include establishing optical links between the payload and the fixed OGSs on the earth and between the payload and a moving user platform such as space-borne LEO or an airborne platform. The latter is much more complex. The acquisition sequence will be coordinated by the mission operation center. When using multiple OGSs on the earth, a seamless handover between ground stations will be necessary. This will be important for establishing future potential connectivity to global remote locations based entirely on optical wave technology.

be using the JPL Optical Communications Telescope Laboratory equipped with AO systems as described before for the LEO satellite and OGS communication links. The AO system will be needed to correct for the atmospheric turbulence-induced distortions of the optical signals close to the earth. The other ground station in Hawaii will be developed by MIT/LL. The ultimate goal of this LCRD will be to prove the development of the potential future optical service provider in space. The optical links in the future will even be able to support between any two user remote platforms to establish global Internet connectivity. The ATP operations will include establishing optical links between the payload and the fixed OGSs on the earth and between the payload and a moving user platform such as space-borne LEO or an airborne platform. The latter is much more complex. The acquisition sequence will be coordinated by the mission operation center. When using multiple OGSs on the earth, a seamless handover between ground stations will be necessary. This will be important for establishing future potential connectivity to global remote locations based entirely on optical wave technology.

> Read full chapter

Modulation and Demodulation Technologies

Vijay K. Garg, Yih-Chen Wang, in The Electrical Engineering Handbook, 2005

4.5 The π/4 Differential Phase Shift Keying

We can design a phase shift keying system to be inherently differential and thus solve the detection problems. The π/4 differential quadrature phase shift keying (π/4-DQPSK) is a compromise modulation method because the phase is restricted to fluctuate between ±π/4 and ±3π/4 rather than the ±π/2 phase changes for OQPSK. The method has a spectral efficiency of about 20% more than the Gaussian minimum shift keying (see upcoming section) modulation used for DECT and GSM.

The π/4-DQPSK is essentially a π/4-shifted QPSK with differential encoding of symbol phases. The differential encoding mitigates loss of data due to phase slips. However, differential encoding results in the loss of a pair of symbols when channel errors occur. This can be translated to approximately a 3-dB loss in E_b/N_0 relative to coherent π/4-QPSK.

A π/4-shifted QPSK signal constellation consists of symbols corresponding to eight phases. These eight phase points can be considered to be formed by superimposing two QPSK signal constellations offset by 45 degrees relative to each other. During each symbol period, a phase angle from only one of the two QPSK constellations is transmitted. The two constellations are used alternately to transmit every pair of bits (di-bits). Thus, successive symbols have a relative phase difference that is one of the four phases shown in Table 4.1.

each other. During each symbol period, a phase angle from only one of the two QPSK constellations is transmitted. The two constellations are used alternately to transmit every pair of bits (di-bits). Thus, successive symbols have a relative phase difference that is one of the four phases shown in Table 4.1.

FIGURE 4.8. The π/4-DQPSK Modulation

TABLE 4.1. Phase Transitions of π/4-DQPSK

Symbol	π/4-DQPSK phase transition
00	45°
01	135°
10	–45°
11	–135°

Figure 4.8 shows the π/4-shifted QPSK signal constellation. When the phase angles of π/4-shifted QPSK symbols are differentially encoded, the resulting modulation is π/4-shifted DQPSK. This can be done either by differential encoding of the source bits and mapping them onto absolute phase angles or, alternately, by directly mapping the pairs of input bits onto relative phase (±π/4 ± 3π/4) as shown in Figure 4.8. The binary data stream entering the modulator $b_M(t)$ is converted by a serial-to-parallel converter into two binary streams $b_o(t)$ and $b_e(t)$ before the bits are differentially encoded (see Figure 4.9):

FIGURE 4.9. Differential Encoding of π/4-DQPSK

(4.25)

(4.26)

(4.26)

The I_k and Q_k are the in-phase and quadrature components of the π/4-shifted DQPSK signal corresponding to the *k*th symbol. The amplitudes of I_k and Q_k are . Because the absolute phase of (*k*−1)th symbol is $□_{k-1}$, the in-phase and quadrature components can be expressed as:

(4.27)

(4.28)

These component signals (I_k, Q_k) are then passed through baseband filters having a raised cosine frequency response as:

(4.29).

In equation 4.29 □ is the roll-off factor, and T_s is the symbol duration.

If $g(t)$ is the response to pulses I_k and Q_k at the filter input, then the resultant transmitted signal is given as:

(4.30)

(4.31)

where 2π□ is the carrier frequency of transmission.

The component $□_k$ results from differential encoding (i.e., $□_k = □_{k-1} + \Delta□_k$).

Depending on the detection method (coherent detection or differential detection), the error performance of π/4-DQPSK can either be the same or 3 dB worse than QPSK.

> Read full chapter

Submarine terminal

Arnaud Leroy, Omar Ait Sab, in Undersea Fiber Communication Systems (Second Edition), 2016

Cycle slip mitigation

Cycle slip mitigation

Coherent optical transmission systems today use powerful DSP algorithms to recover the transmitted signal and compensate for transmission impairments. One of the main blocks of the coherent receiver is carrier phase recovery, which is done in the digital electrical domain. The carrier phase is corrupted by the laser phase noise and also by the transmission nonlinear phase noise. The received signal after carrier recovery may be affected by phase slips (also called cycle slips (CS)) [43]. When a cycle slip occurs, it propagates and all the following decided data will be corrupted and cannot be properly recovered. Such catastrophic error propagation can be avoided by using differential encoding [44].

The probability of the cycle slips depends on the accumulated nonlinear noise phase introduced by the optical transmission link.

Differential encoding (DE) is the first and the most common approach that has been used to mitigate the cycle slips in optical submarine transmission systems. DE encodes the transitions between two consecutive symbols and renders a CS to a single error event that could be handled by FEC decoder. However, as the symbols are correlated by DE, a single transmission error will be doubled (with hard decision) after differential decoding of the received data, leading to DE penalty.

Figure 14.11 shows an example of differential encoder and decoder and the differential encoding induced penalty with BPSK/QPSK modulations.

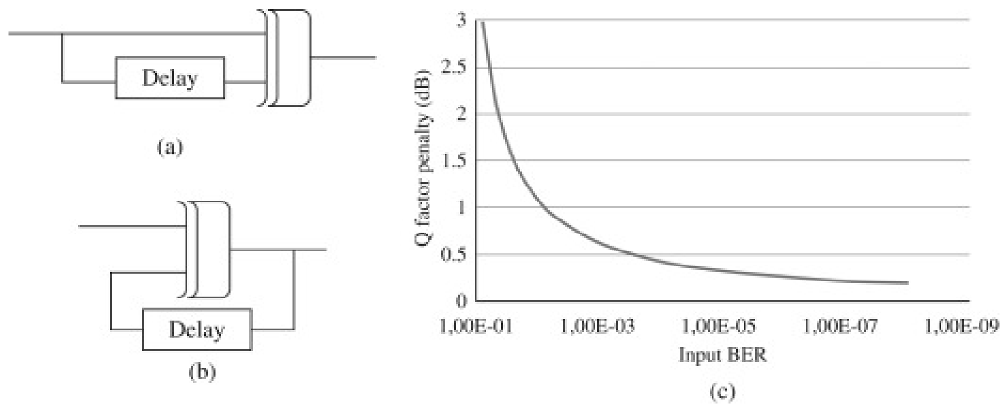


Figure 14.11. (a) Differential encoder, (b) differential decoder, and (c) DE induced penalty for BPSK/QPSK.

As illustrated in Figure 14.11, the penalty grows dramatically at higher input BER. For an input 10^{-2} BER the induced penalty is around 1 dB, which means that 1 additional dB Q factor is needed to achieve the same performance as a direct coding scheme transmission. In the range of input BER that is targeted for the upcoming submarine SLTE generation, the penalty grows up to 1.5 dB. Furthermore, soft LLR metrics, on which soft decoding FECs rely, should take advantage of the correlation introduced by DE to avoid additional performance degradation [45].

In order to reduce the DE-induced penalty, iterative SISO decoding between differential decoder and FEC decoder (called turbo differential decoding) has been used and showed that a large part of the DE penalty can be recovered while maintaining a good tolerance to cycle slips even at high CS probability [46]. However, this solution leads to an increase of the receiver complexity, as several SISO differential turbo decoding iterations have to be carried out. Another approach considers the introduction of pilot symbols to help the carrier phase estimator mitigating cycle slips.

Pilot-assisted carrier phase estimation: Pilot symbols, having known data information, are periodically inserted after D-length data transmissions as shown in Figure 14.12. These regular pilots are first used to remove the phase ambiguity stemming from the blind CPE (often based on constellation symmetry), thus enabling direct decoding of the transmitted symbols. By adjusting absolute phase reference periodically, they also avoid subsequent propagation due to cycle slip occurrence. Therefore, the use of pilot-assisted CPE could avoid the need of differential encoding and thus enhance the receiver sensitivity.

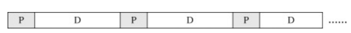


Figure 14.12. Transmitted data frame with Pilot symbols.

Pilot symbols are an extra overhead added to the transmitted signal, thus increasing the signal bit rate and leading to received OSNR penalty. The number of pilot symbols P depends on the nonlinear phase noise induced by the optical transmission link. In other words, the rate of P increases with the CS probability that has to be mitigated. CS probability depends on several parameters: CPE taps number, launched power, neighboring channels modulation format, optical link dispersion management. For example, uncompensated dispersion links induce a very low CS probability. However, CS probability increases drastically ($>10^{-3}$) for managed dispersion systems especially around zero cumulated dispersion wavelength [43,47].

With pilot symbols strategy, the maximum number of corrupted symbols by a CS is *D* symbols. So the FEC has to be designed to handle such a burst of errors. In general, the rate of added pilot symbols is a design trade-off between OSNR induced penalty, CS probability, and FEC code. Typically, adding 1 to 3% of pilot symbols is sufficient to support 10^{-3} CS probability with no need of DE [47]. However, it may not be sufficient for all managed dispersion system configurations.

Recently, multidimensional coded modulation has been introduced and showed that it can enable direct absolute phase detection without the need of pilot symbols [48].

> Read full chapter

Transceiver Requirements

$R(n)$ and the output of the encoder is $E(n)$, the $E(n)$ is generated from $R(n)$ using the following equation:

$R(n)$ and the output of the encoder is $E(n)$, the $E(n)$ is generated from $R(n)$ using the following equation:

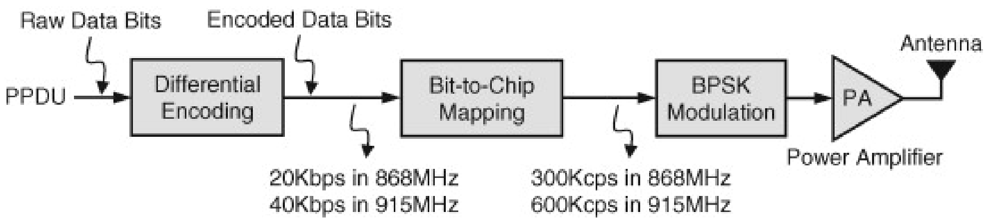


Figure 4.12. The Spreading and Modulation Steps for the 868/915 MHz DSSS/BPSK Mode of Operation

(4.7)

where □ indicates the exclusive *OR* operation. The value of $E(0)$ is equal to zero. For example, if the raw data entering the differential encoder is the following:

then the encoded output will become:

When the receiver of the packet receives the encoded data $E(n)$, it can simply recover the original data $R(n)$ by performing an exclusive *OR* operation:

(4.8)

Equation 4.8 indicates that the original data R(n) is recovered by using two consecutive bits of the received encoded information. In other words, the original bit is determined by comparing the phase difference of two consecutive signals instead of using the absolute phase value, which is the main goal of using differential encoding.

In Figure 4.12, the encoded data bits are mapped using the DSSS mechanism. But in contrast to case 1, the bits are not grouped together to form a multibit symbol. Instead, each single bit is mapped to a 15-bit chip sequence. The mapping table is provided in Appendix C. The processing gain associated with cases 2 and 3 is 11.76 dB:

(4.9)

Implementation of BPSK modulation, similar to O-QPSK, requires pulse-shaping filters to shape the PSD of the modulated signal.

ASK is used in optional cases 4 and 5. In ASK modulation, the information is embedded in the signal amplitude instead of the signal phase. The spreading method for cases 4 and 5 is parallel sequence spread spectrum (PSSS) instead of the direct sequence spread spectrum (DSSS) used in all other cases.

We can understand the basic concept of PSSS by comparing PSSS to DSSS. Both of these spreading techniques rely on orthogonal sequences to spread the signal before transmission. In DSSS, a single chip sequence is transmitted, but PSSS sends superposition of multiple orthogonal sequences in parallel. Figure 4.13 shows the PSSS mapping mechanism. First, each bit of the binary data that needs to be transmitted is converted to bipolar data. In bipolar data, the logic level 0 is replaced by −1, whereas logic level +1 is kept the same. Therefore, the data that enters the multiplier stage is an array with +1 and −1 levels. Each bipolar data is multiplied by a unique sequence. The PSSS sequences are identified by sequence numbers. (The table of PSSS sequences is provided in Appendix A.) In the 868 MHz frequency band, the sequence numbers 0 to 19 are utilized. Therefore, the value of parameter *n* in Figure 4.13 is equal to 20. The 915 MHz band uses only sequence numbers 0 to 4, and the value of parameter *n* in Figure 4.13 is set to 5. In other words, every 20 bits in 868 MHz are grouped together to form a single symbol. In 915 MHz, every 5 bits form a single symbol.

signal before transmission. In DSSS, a single chip sequence is transmitted, but PSSS sends superposition of multiple orthogonal sequences in parallel. Figure 4.13 shows the PSSS mapping mechanism. First, each bit of the binary data that needs to be transmitted is converted to bipolar data. In bipolar data, the logic level 0 is replaced by −1, whereas logic level +1 is kept the same. Therefore, the data that enters the multiplier stage is an array with +1 and −1 levels. Each bipolar data is multiplied by a unique sequence. The PSSS sequences are identified by sequence numbers. (The table of PSSS sequences is provided in Appendix A.) In the 868 MHz frequency band, the sequence numbers 0 to 19 are utilized. Therefore, the value of parameter *n* in Figure 4.13 is equal to 20. The 915 MHz band uses only sequence numbers 0 to 4, and the value of parameter *n* in Figure 4.13 is set to 5. In other words, every 20 bits in 868 MHz are grouped together to form a single symbol. In 915 MHz, every 5 bits form a single symbol.

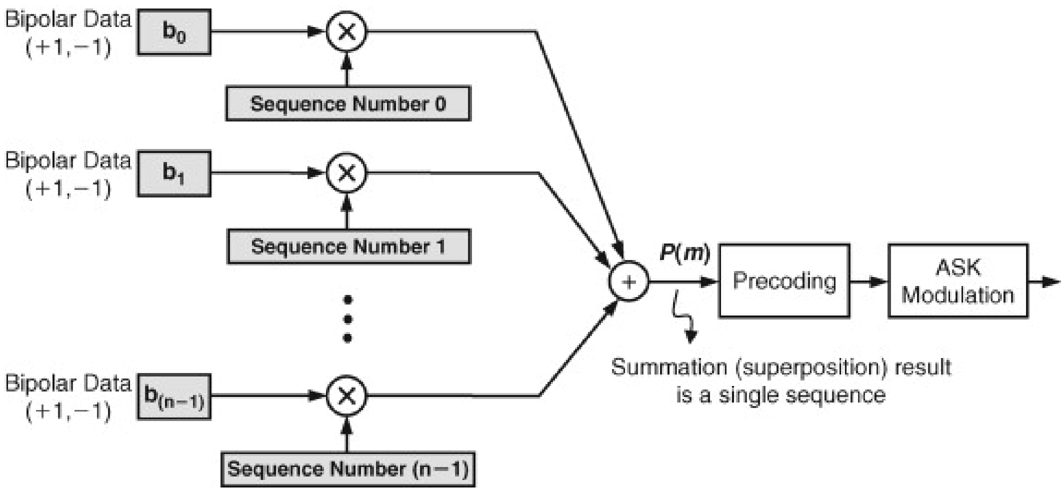


Figure 4.13. The PSSS Modulator in a Transmitter

The result of the summation (i.e., superposition) in Figure 4.13 is a single sequence. Each entry in the summation sequence is a multilevel value. The orthogonal characteristic of the sequences allows the receiver to be capable of recovering each bipolar data from this single sequence. Because the sequences used in PSSS are nearly orthogonal, the PSSS method is also referred to as the orthogonal code division multiplexing (OCDM) method. Each sequence provided for the 868 MHz band in Appendix A has an offset of one-half of a chip compared to the next sequence. Therefore, these 32-bit sequences are sometimes referred to as 64 *half-chip sequences*.

Figure 4.14a shows the beginning of an example of the summation sequence $P(m)$. These different levels will determine the amplitude levels in the ASK modulator. The maximum and minimum levels of the sequence $P(m)$ are random numbers. If the maximum and minimum of $P(m)$ are symmetric about zero, the implementation of ASK will be optimal. Therefore, after the summation in Figure 4.13, there is a precoding block. The role of this precoding block is to add a constant value to the sequence $P(m)$ to ensure that the maximum and minimum are symmetric about zero. For example, in Figure 4.14a, where the maximum is +5 and the minimum is –3, subtracting a constant 1 from the sequence $P(m)$ will make the maximum and minimum symmetric about zero:

sequence $P(m)$ to ensure that the maximum and minimum are symmetric about zero. For example, in Figure 4.14a, where the maximum is +5 and the minimum is –3, subtracting a constant 1 from the sequence $P(m)$ will make the maximum and minimum symmetric about zero:

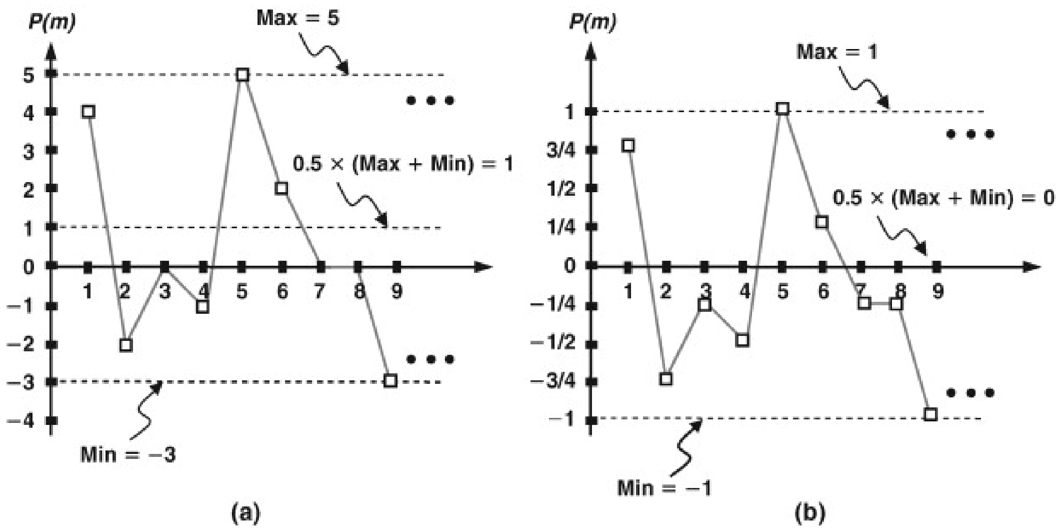
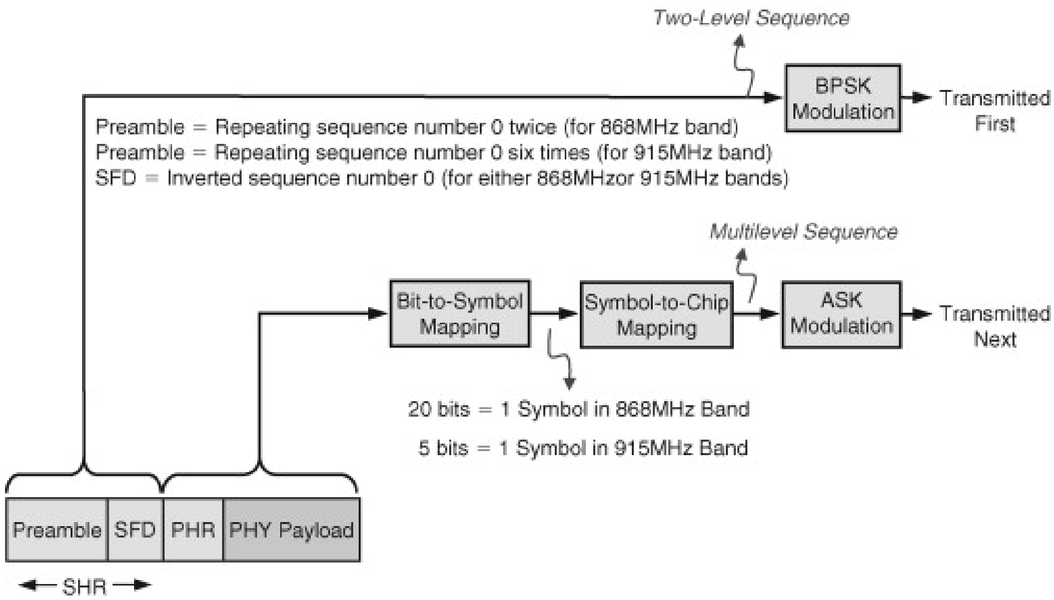


Figure 4.14. The precoding is used to ensure that the maximum and minimum are symmetric about zero

(4.10)

The adjusted sequence is shown in Figure 4.14b. The precoding block also normalizes the sequence $P(m)$ to make the maximum and minimum of the sequence $P(m)$ equal to +1 and –1, respectively.

Figure 4.15 shows that in 868 and 915 MHz, the PHY header (PHR) and the PHY payload are modulated using PSSS/ASK. But the SHR is directly modulated using the BPSK without any mapping stage. The reason is that the content of SHR is already a spread sequence and does not need any additional spreading. Recall from Chapter 3 that in the PSSS/ASK mode of operation, the preamble is constructed by repeating the sequence number 0 in Appendix A. In 868 MHz, the sequence number 0 is repeated twice to form the preamble sequence. In 915 MHz, the preamble is generated by repeating sequence number 0 six times. The start-of-frame delimiter (SFD) is also generated using the sequences in Appendix A. In both 868 MHz and 915 MHz, the SFD contains the inverted sequence number 0 in Appendix A. Since the content of SHR has only two levels (+1 and –1), the ASK modulation is reduced to BPSK modulation. The pulse shaping performed for SHR is the same as the pulse shaping used for PHR and the PHY payload.


Two-Level Sequence
BPSK Modulation
Transmitted First
Preamble = Repeating sequence number 0 twice (for 868MHz band)
Preamble = Repeating sequence number 0 six times (for 915MHz band)
SFD = Inverted sequence number 0 (for either 868MHzor 915MHz bands)
Multilevel Sequence
Bit-to-Symbol Mapping
Symbol-to-Chip Mapping
ASK Modulation
Transmitted Next
20 bits = 1 Symbol in 868MHz Band
5 bits = 1 Symbol in 915MHz Band
Preamble
SFD
PHR
PHY Payload
SHR

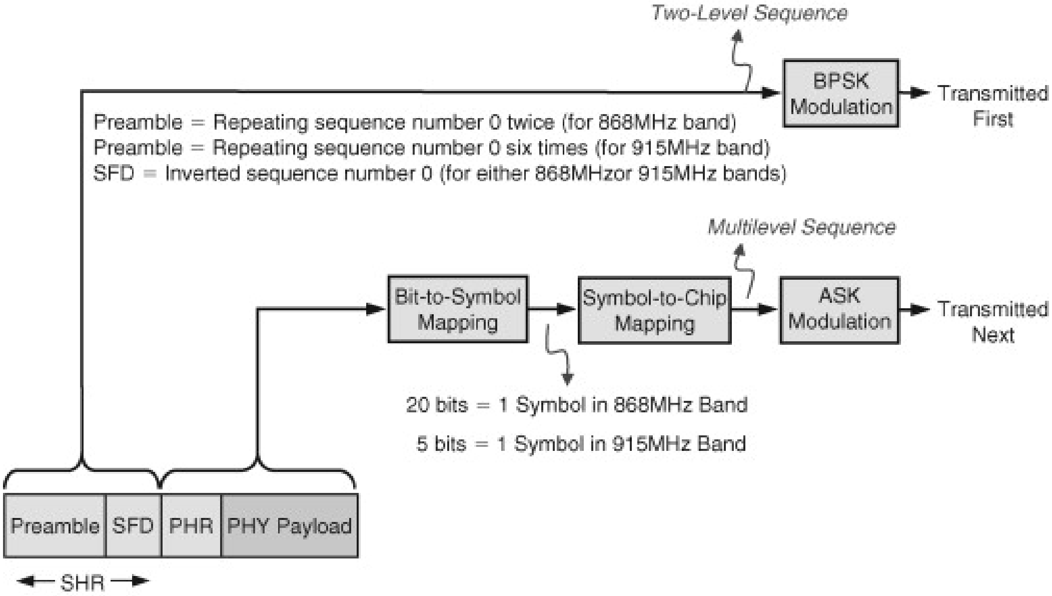


Figure 4.15. In the PSSS Optional Mode of Operation, the SHR is Modulated using BPSK

Availability of PSSS/ASK modulation as an optional mode of operation in sub-GHz provides the opportunity of increasing the bit rate to 250 Kbps from 40 Kbps (maximum) in DSSS/BPSK mode. In PSSS, in addition to processing gain due to increase in signal bandwidth, there is coding gain as well. The coding gain is the amount of additional SNR that would be required to provide the same BER performance for an uncoded signal. However, comparing the PSSS/ASK versus DSSS/BPSK and DSSS/OQPSK in terms of processing gain will depend on the transceiver nonlinearity performance. The reason is that in the BPSK and OQPSK, in contrast to multilevel ASK modulation, there is no information in the amplitude, and the nonlinearity of the transmitter and receiver has less impact on the quality of the signal. Therefore, the actual SNR improvement for the PSSS/ASK mode of operation must be determined by measurement or simulation.

Cases 6 and 7 in Tables 4.16 and 4.17 are based on the DSSS/OQPSK modulation method similar to case 1. But the chip sequences used in cases 6 and 7 are half the size of the length of the chip sequence used in case 1. Every 4 bits (a symbol) in cases 6 and 7 are mapped to a 16-bit chip sequence. (The mapping table is provided in Appendix C.) Since the length of the chip sequence in cases 6 and 7 is less than in case 1, the processing gain is reduced to 6 dB.

> Read full chapter

Video Compression

Khalid Sayood, in Introduction to Data Compression (Fourth Edition), 2012

19.3 Motion Compensation

19.3 Motion Compensation

In most video sequences there is little change in the contents of the image from one frame to the next. Even in sequences that depict a great deal of activity, there are significant portions of the image that do not change from one frame to the next. Most video compression schemes take advantage of this redundancy by using the previous frame to generate a prediction for the current frame. We have used prediction previously when we studied differential encoding schemes. If we try to apply those techniques blindly to video compression by predicting the value of each pixel by the value of the pixel at the same location in the previous frame, we will run into trouble because we would not be taking into account the fact that objects tend to move between frames. Thus, the object in one frame that was providing the pixel at a certain location with its intensity value might be providing the same intensity value in the next frame to a pixel at another location. If we don't take this into account, we can actually increase the amount of information that needs to be transmitted. **Example 19.3.1** Consider the two frames of a motion video sequence shown in Figure 19.1. The only differences between the two frames are that the devious looking individual has moved slightly downward and to the right of the frame, while the triangular object has moved to the left. The differences between the two frames are so slight, you would think that if the first frame was available to both the transmitter and receiver, not much information would need to be transmitted to the receiver in order to reconstruct the second frame. However, if we simply take the difference between the two frames, as shown in Figure 19.2, the displacement of the objects in the frame results in an image that contains more detail than the original image. In other words, instead of the differencing operation reducing the information, there is actually more information that needs to be transmitted. Figure 19.1. Two frames of a video sequence. Figure 19.2. Difference between the two frames.

In order to use a previous frame to predict the pixel values in the frame being encoded, we have to take the motion of objects in the image into account. Although a number of approaches have been investigated, the method that has worked best in practice is a simple approach called *block-based motion compensation*. In this approach, the frame being encoded is divided into blocks of size . For each block, we search the previous reconstructed frame for the block of size that most closely matches the block being encoded. We can measure the closeness of a match, or distance, between two blocks by the sum of absolute differences between corresponding pixels in the two blocks. We would obtain the same results if we used the sum of squared differences between the corresponding pixels as a measure of distance. Generally, if the distance from the block being encoded to the closest block in the previous reconstructed frame is greater than some prespecified threshold, the block is declared uncompensable and is encoded without the benefit of prediction. This decision is also transmitted to the receiver. If the distance is below the threshold, then a *motion vector* is transmitted to the receiver. The motion vector is the relative location of the block to be used for prediction obtained by subtracting the coordinates of the upper-left corner pixel of the block being encoded from the coordinates of the upper-left corner pixel of the block being used for prediction.

tion. This decision is also transmitted to the receiver. If the distance is below the threshold, then a *motion vector* is transmitted to the receiver. The motion vector is the relative location of the block to be used for prediction obtained by subtracting the coordinates of the upper-left corner pixel of the block being encoded from the coordinates of the upper-left corner pixel of the block being used for prediction.

Suppose the block being encoded is an block between pixel locations (24, 40) and (31, 47); that is, the upper-left corner pixel of the block is at location (24, 40). If the block that best matches it in the previous frame is located between pixels at location (21, 43) and (28, 50), then the motion vector would be (−3, 3). The motion vector was obtained by subtracting the location of the upper-left corner of the block being encoded from the location of the upper-left corner of the best matching block. Note that the blocks are numbered starting from the top-left corner. Therefore, a positive component means that the best matching block in the previous frame is to the right of the location of the block being encoded. Similarly, a positive component means that the best matching block is at a location below that of the location of the block being encoded.

Example 19.3.2 Let us again try to predict the second frame of Example 10.3.1 using motion compensation. We divide the image into blocks and then predict the second frame from the first in the manner described above. Figure 19.3 shows the blocks in the previous frame that were used to predict some of the blocks in the current frame. Figure 19.3. Motion-compensated prediction.Notice that in this case all that needs to be transmitted to the receiver are the motion vectors. The current frame is completely predicted by the previous frame. □

We have been describing motion compensation where the displacement between the block being encoded and the best matching block is an integer number of pixels in the horizontal and vertical directions. There are algorithms in which the displacement is measured in half pixels. In order to do this, pixels of the coded frame being searched are interpolated to obtain twice as many pixels as in the original frame. This “doubled” image is then searched for the best matching block.

The doubled image is obtained as follows: Consider Table 19.1. In this image , and are the pixels of the original frame. The pixels , and are obtained by interpolating between the two neighboring pixels:

Table 19.1. “Doubled” image.

(1)

The pixel is obtained as the average of the four neighboring pixels from the coded original:

We have described motion compensation in very general terms in this section. The various schemes in this chapter use specific motion compensation schemes that differ from each other. The differences generally involve the region of search for the matching block and the search procedure. We will look at the details with the study of the compression schemes. But before we begin our study of compression schemes, we briefly discuss how video signals are represented in the next section.

We have described motion compensation in very general terms in this section. The various schemes in this chapter use specific motion compensation schemes that differ from each other. The differences generally involve the region of search for the matching block and the search procedure. We will look at the details with the study of the compression schemes. But before we begin our study of compression schemes, we briefly discuss how video signals are represented in the next section.

> Read full chapter
> Read full chapter

ScienceDirect is Elsevier's leading information solution for researchers.
Copyright © 2018 Elsevier B.V. or its licensors or contributors. ScienceDirect ® is a registered trademark of Elsevier B.V. Terms and conditions apply