

Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

The Lecture Contains:

- ☰ The Need for Video Coding
- ☰ Elements of a Video Coding System
- ☰ Elements of Information Theory
- ☰ Symbol Encoding
- ☰ Run-Length Encoding
- ☰ Entropy Encoding

◀ Previous Next ▶

Video Coding Basics

The Need for Video Coding

We consider the raw data rates of a number of typical video formats that are shown in Table 8.1. Table 8.2 shows a number of typical video applications and the bandwidths available to them. It is immediately evident from the tables that video coding (or compression) is a key enabling technology for such applications. As an example we consider a 2-hour CCIR-601 color movie. Without compression, a 5-Gbit compact disc (CD) can hold only 30 seconds of this movie. To store the entire movie on the same CD requires a compression ratio of about 240:1. Without compression, the same movie will take about 36 days to arrive at the other end of a 384 Kbits/s Integrated Services Digital Network (ISDN) channel. To achieve real-time transmission of the movie over the same channel, a compression ratio of about 432:1 is required.

Table 8.1: Raw data rates of typical video formats

Format	Raw data rate
HDTV	1.09 Gbits/s
CCIR-601	165.89 Mbits/s
CIF @ 15 f.p.s	18.24 Mbits/s
QCIF @ 10 f.p.s	3.04 Mbits/s

Table 8.2: Typical video applications

Application	Bandwidth
HDTV (6-MHz channel)	20 Mbits/s
Desktop video (CD-ROM)	1.5 Mbits/s
Videoconferencing (ISDN)	384 kbits/s
Videophone (PSTN)	56 kbits/s
Videophone (GSM)	10 kbits/s

Elements of a Video Coding System

The aim of video coding is to reduce, or compress, the number of bits used to represent video. Video signals contain three types of redundancy: statistical, psychovisual, and coding redundancy. Statistical redundancy is present because certain data patterns are more likely than others. This is mainly due to the high spatial (intraframe) and temporal (interframe) correlations between neighboring pixels (pels). Psychovisual redundancy is due to that fact that the HVS is less sensitive to certain visual information than to other visual information. If video is coded in a way that uses more and/or longer code symbols than absolutely necessary, it is said to contain coding redundancy. Video compression is achieved by reducing or eliminating these redundancies.

Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

Figure 8.1 shows the main elements of a video encoder. Each element is designed to reduce one of the three basic redundancies.

The mapper (or transformer) transforms the input raw data into a representation that is designed to reduce statistical redundancy and make the data more amenable to compression in later stages. The transformation is a one-to-one mapping and is, therefore, reversible.

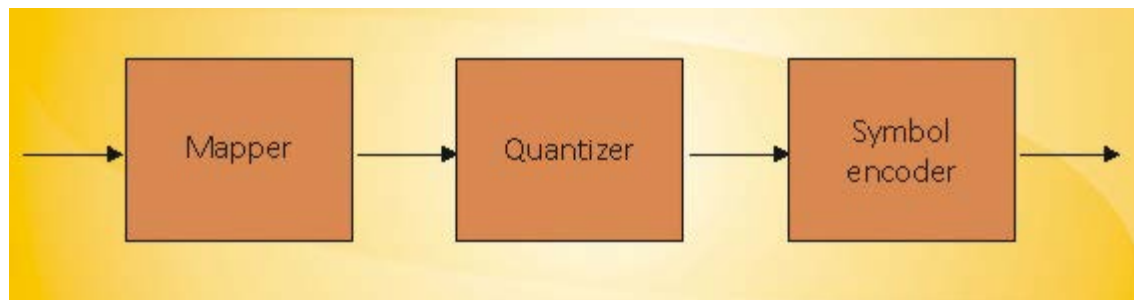


Figure 8.1: Elements of a video encoder

The quantizer reduces the accuracy of the mapper's output, according to some fidelity criterion, in an attempt to reduce psychovisual redundancy. This is a many-to-one mapping and is, therefore, irreversible.

The symbol encoder (or codeword assigner) assigns a codeword, a string of binary bits, to each symbol at the output of the quantizer. The code must be designed to reduce coding redundancy. This operation is reversible.

In general, compression methods can be classified into **lossless** methods and **lossy** methods. In lossless methods the reconstructed (compressed-decompressed) data is identical to the original data. This means that such methods do not employ a quantizer. Lossless methods are also known as bit-preserving or reversible methods.

In lossy methods the reconstructed data is not identical to the original data; that is, there is loss of information due to the quantization process. Such methods are therefore irreversible, and they usually achieve higher compression than lossless methods.

◀ Previous Next ▶

Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

Elements of Information Theory

A source S with an alphabet A can be defined as a discrete random process $S = S_1, S_2, \dots$, where each random variable S_i takes a value from the alphabet A .

In a discrete memoryless source (DMS) the successive symbols of the source are statistically independent. Such a source can be completely defined by its alphabet $A = \{a_1, a_2, \dots, a_N\}$ and the associated probabilities $P = \{p(a_1), p(a_2), \dots, p(a_N)\}$, where $\sum_{i=1}^N p(a_i) = 1$. According to information theory, the information I contained in a symbol a_i is given by

$$I(a_i) = \log_2 p(a_i) \text{ (bits)} \quad (8.1)$$

and the average information per source symbol $H(S)$, also known as the entropy of the source, is given by

$$H(S) = \sum_{i=1}^N p(a_i) \log_2 p(a_i) \text{ (bits/symbol)}. \quad (8.2)$$

A more realistic approach is to model sources using Markov-K random processes. In this case the probability of occurrence of a symbol depends on the values of the K preceding symbols. Thus, a Markov-K source can be specified by the conditional probabilities $p(S_j = a_i | S_{j-1}, \dots, S_{j-K})$, for all $j, a_i \in A$. In this case, the entropy is given by

$$H(S) = \sum_{S^K} p(S_{j-1}, \dots, S_{j-K}) H(S | S_{j-1}, \dots, S_{j-K}). \quad (8.3)$$

Where S^K denotes all possible realizations of S_{j-1}, \dots, S_{j-K} , and

$$H(S | S_{j-1}, \dots, S_{j-K}) = - \sum_{a_i \in A} p(S_j = a_i | S_{j-1}, \dots, S_{j-K}) \log p(S_j = a_i | S_{j-1}, \dots, S_{j-K}). \quad (8.4)$$

The performance bound of a lossless coding system is given by the lossless coding theorem :

Lossless coding theorem: The minimum bit rate $R_{min} = H(S) + \epsilon$, where ϵ is a positive quantity that can be made arbitrarily close to zero.

For a DMS, this lower bound can be approached by coding symbols independently, whereas for a Markov-K source, blocks of K symbols should be encoded at a time.

Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

The performance bounds of lossy coding systems are addressed by a branch of information theory known as rate-distortion theory. This theory provides lower bounds on the achievable average distortion for a given average bit rate, or vice versa. It also promises that codes exist that approach the theoretical bounds when the code dimension and delay become large. An important theorem in this branch is the source coding theorem:

Source coding theorem: There exists a mapping from source symbols to codeword such that for a given distortion D , $R(D)$ bits/symbol are sufficient to achieve an average distortion that is arbitrarily close to D .

The function $R(D)$ is known as the rate-distortion function. It is a convex, continuous, and strictly decreasing function of D . This function is normally computed using numerical methods analytically. Although rate-distortion theory does not give an explicit method for constructing practical optimum coding systems, it gives very important hints about the properties of such systems.

Symbol Encoding

Another key element of video coding systems is the symbol encoder. This assigns a codeword to each symbol at the output of the quantizer. The symbol encoder must be designed to reduce the coding redundancy present in the set of symbols. Following are a number of commonly used techniques that can be applied individually or in combinations.

 **Previous** **Next** 

Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

The output of the quantization step may contain long runs of identical symbols. One way to reduce this redundancy is to employ run-length encoding (RLE). There are different forms of RLE. For example, if the quantizer output contains long runs of zeros, then RLE can represent such runs with intermediate symbols of the form (RUN, LEVEL). For example, a run of the form 0, 0, 0, 0, 0, 9 can be represented by the intermediate symbol (5,9).

Entropy Encoding

The quantizer can be considered a DMS L that can be completely specified by its alphabet $R = \{r_1, r_2, \dots, r_N\}$, where r_i are the reconstruction levels and the associated probabilities of occurrence $P = \{p(r_1), p(r_2), \dots, p(r_N)\}$. The information contained in a symbol r_i is given by Equation (8.1), whereas the entropy of the source $H(L)$ is given by Equation (8.2).

Now consider a symbol encoder that assigns a codeword c_i of length $l(c_i)$ bits to symbol r_i . Then the average word length \bar{L} of the code is given by

$$\bar{L} = \sum_{i=1}^N p(r_i) l(c_i) \text{ (Bits)}, \quad (8.5)$$

And, the efficiency (η) of the code is

$$\eta = \frac{H(L)}{\bar{L}} \quad (8.6)$$

Thus, an optimal ($\eta = 1$) code must have an average word length that is equal to the entropy of the source; i.e., $\bar{L} = H(L)$. Clearly, this can be achieved if each codeword length is equal to the information content of the associated symbol, that is, $l(c_i) = I(r_i)$. Since $I(r_i)$ is inversely proportional to $p(r_i)$ (from Equation (8.1),

then an efficient code must assign shorter codewords to more probable symbols, and vice versa. This is known as entropy encoding or variable-length coding (VLC) (as opposed to fixed-length coding (FLC)).



Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

The most commonly used VLC is Huffman coding. Given a finite set of symbols and their probabilities, Huffman coding yields the optimal integer-length prefix code. The basic principles of Huffman coding can be illustrated using the example given in Figure 8.2. In each stage, the two least probable symbols are combined to form a new symbol with a probability equal

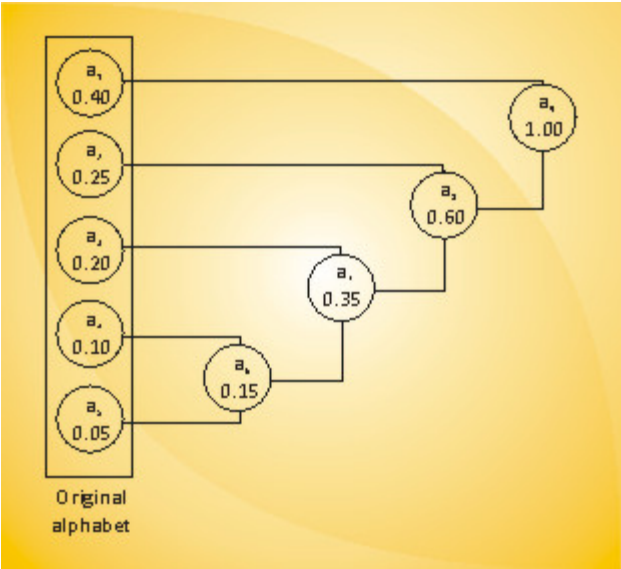


Figure 8.2: Huffman coding example

Table 8.3: Comparison between VLC (of Figure 8.2) and a 3-bit FLC

r_i	$p(r_i)$	$I(r_i)$	VLC c_i	FLC c_i
a_1	0.40	1.32 bits	0 (1 bit)	000
a_2	0.25	2.00 bits	10 (2 bits)	001
a_3	0.20	2.32 bits	111 (3 bits)	010
a_4	0.10	3.32 bits	1101 (4 bits)	011
a_5	0.05	4.32 bits	1100 (4 bits)	100

$H(R) \approx 2.04$ bits/symbol

$\bar{L}_{FLC} = 3$ bits/word

$\eta_{FLC} \approx 0.68$

$\bar{L}_{VLC} \approx 2.1$ bits/word


$\eta_{VLC} \approx 0.97$

Module 8: Video Coding Basics

Lecture 40: Need for video coding, Elements of information theory, Lossless coding

to the sum of their probabilities. This new symbol creates a new node in the tree, with two branches connecting it to the original two nodes. A "0" is assigned to one branch and a "1" is assigned to the other. The original two nodes are then removed from the next stage. This process is continued until the new symbol has a probability of 1. Now, to find the codeword for a given symbol, start at the right-hand end of the tree and follow the branches that lead to the symbol of interest combining the "0"s and "1"s assigned to the branches. Table 8.3 shows the obtained VLC and compares it to an FLC of 3 bits. Clearly, the Huffman VLC is much more efficient than the FLC.

There are more efficient implementations of Huffman coding. For example, in many cases, most of the symbols of a large symbol set have very small probabilities. This leads to very long codewords and consequently to large storage requirements and high decoding complexity. In the modified Huffman code, the less probable symbols (and their probabilities) are lumped into a single symbol like ESCAPE. A symbol in this new ESCAPE category is coded using the VLC codeword for ESCAPE followed by extra bits to identify the actual symbol. Standard video codecs also use 2-D and 3-D versions of the Huffman code. For example, the H.263 standard uses a 3-D Huffman code where three different symbols (LAST, RUN, LEVEL) are lumped into a single symbol (EVENT) and then encoded using one VLC codeword.

One disadvantage of the Huffman code is that it can only assign integer-length codewords. This usually leads to a suboptimal performance. For example, in Table 8.3, the symbol  was represented with a 3-bit codeword, whereas its information content is only 2.32 bits. In fact, Huffman code can be optimal only if all the probabilities are integer powers of 1/2. An entropy code that can overcome this limitation and approach the entropy of the source is **arithmetic coding**. In Huffman coding there is a one-to-one correspondence between the symbols and the codewords. In arithmetic coding, however, a single variable-length codeword is assigned to a variable-length block of symbols.

◀ Previous Next ▶