Chapter Eight

Loss Compression Algorithms

Introduction

As discussed in Chapter 7, the *compression ratio* for image data using lossless compression techniques (e.g., Huffman Coding, Arithmetic Coding, LZW) *is* low when the image histogram is relatively flat. For image compression in multimedia applications, where a higher compression ratio is required, lossy methods are usually adopted. In lossy compression, the compressed image is usually not the same as the original image but is meant to form a close approximation to the original image *perceptually.* To quantitatively describe how close the approximation is to the original data, some form of distortion measure is required.

Distortion Measures

A *distortion measure* is a mathematical quantity that specifies how close an approximation is to its original, using some distortion criteria. When looking at compressed data, it is natural to think of the distortion in terms of the numerical difference between the original data and the reconstructed data. However, when the data to be compressed is an image, such a measure may not yield the intended result.

For example, if the reconstructed image is the same as original image except that it is shifted to the right by one vertical scan line, an average human observer would have a hard time distinguishing it from the original and would therefore conclude that the distortion is small. However, when the calculation is carried out numerically, we find a large distortion, because of the large changes in individual pixels of the reconstructed image. The problem is that we need a measure of *perceptual distortion*, not a more naive numerical approach.

The rate distortion theory

Lossy compression always involves a tradeoff between rate and distortion. Rate is the average number of bits required to represent each source symbol. Within this framework, the tradeoff between rate and distortion is represented in the form of a rate-distortion function *R(D)*.

Intuitively, for a given source and a given distortion measure, if *D* is a tolerable amount of distortion, *R(D)* specifies the lowest rate at which the source data can be encoded while keeping the distortion bounded above by *D*. It is easy to see that when $D = 0$, we have a lossless compression of the source. The rate-distortion function

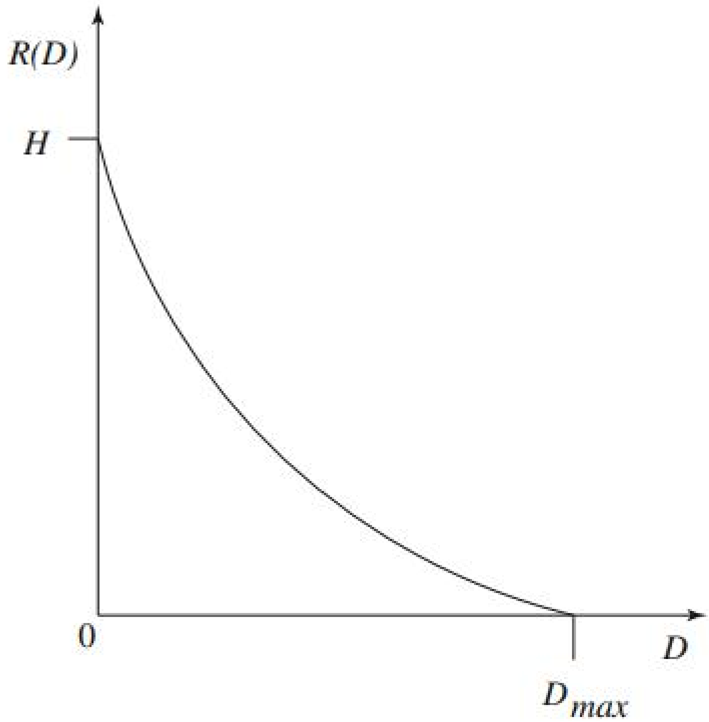

Fig. 8.1 Typical rate-distortion function

is meant to describe a fundamental limit for the performance of a coding algorithm and so can be used to evaluate the performance of different algorithms

Figure 8.1 shows a typical rate-distortion function. Notice that the minimum possible rate at $D = 0$, no loss, is the entropy of the source data. The distortion corresponding to a rate $R(D) \Rightarrow 0$ is the maximum amount of distortion incurred when “nothing” is coded

Quantization

Quantization in some form is the heart of any lossy scheme. Without quantization, we would indeed be losing little information. Here, we embark on a more detailed discussion of quantization than in chapter 6

The source we are interested in compressing may contain a large number of distinct output values (or even infinite, if analog). To efficiently represent the source output, we have to reduce the number of distinct values to a much smaller set, via quantization

Uniform Scalar Quantization

A uniform scalar quantizer partitions the domain of input values into equally spaced intervals, except possibly at the two outer intervals. The endpoints of partition intervals are called the quantizer's *decision boundaries.*

Nonuniform Scalar Quantization

If the input source is not uniformly distributed, a uniform quantizer may be inefficient. Increasing the number of decision levels within the region where the source is densely distributed can effectively lower granular distortion. In addition, without having to increase the total number of decision levels, we can enlarge the region in which the source is sparsely distributed. Such *nonuniform quantizers* thus have nonuniformly defined decision boundaries

Vector Quantization*

One of the fundamental ideas in Sharmon's original work on inf0l1llation theory is that any compression system performs better if it operates on vectors or groups of samples rather than on individual symbols or samples. We can form vectors of input samples by concatenating a number of consecutive samples into a single vector. For example, an input vector might be a segment of a speech sample, a group of consecutive pixels in an image, or a chunk of data in any other format.

Transform Coding

From basic principles of information theory, we know that coding vectors is more efficient than coding scalars (see chapter 7). To cany out such an intention, we need to group blocks of consecutive samples from the source input into vectors

Discrete Cosine Transform (DCT)

The Discrete Cosine Transform (DCT), a widely used transform coding technique, is able to perform decorrelation of the input signal in a data-independent manner. Because of this, it has gained tremendous popularity.

Karhunen-Loeve Transform*

The Karhunen-Loeve Transfonn (KLT) is a reversible linear transform that exploits the statistical properties of the vector representation. Its primary property is that it optimally decorrelates the input. To do so, it fits an n-dimensional ellipsoid around the (mean-subtracted) data. The main ellipsoid axis is the major direction of change in the data