IMAGE COMPRESSION SYSTEMS A JPEG PERSPECTIVE

Jan Čapek, Peter Fabian

Department of Information Systems, Faculty of Economics and Administration, University of Pardubice

1. Introduction

Basic structure of image compression systems can be described following manner.
Image compression is the art and science of reducing the number of bits required describing an image. The two basic components of an image compression system are illustrated in Figure 1. The device that compresses (reduced) the "source" image (the original digital image) is called an encoder and the output of this encoder is called compressed data (or coded data). The compressed and/or reduced data may be either stored or transmitted, but are at some point fed to a decoder. The decoder is a device that recreates or "reconstructs" an image from the compressed data.

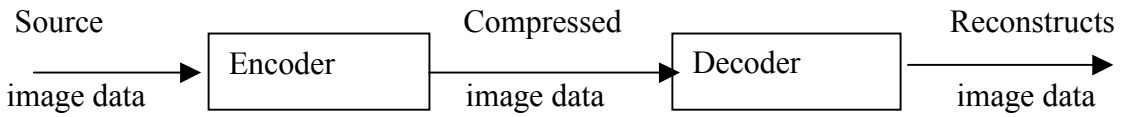

Figure 1. Image compression system

1.1 Encoder structure

In general, a data compression encoding system can be broken into the following basic parts: an encoder model, an encoder statistical model, and an entropy encoder (Fig.2). The encoder model generates a sequence of "descriptors" that is an abstract representation of the image. The statistical model converts these descriptors into symbols and passes them on to the entropy encoder. The entropy encoder compresses the symbols to form the compressed data.

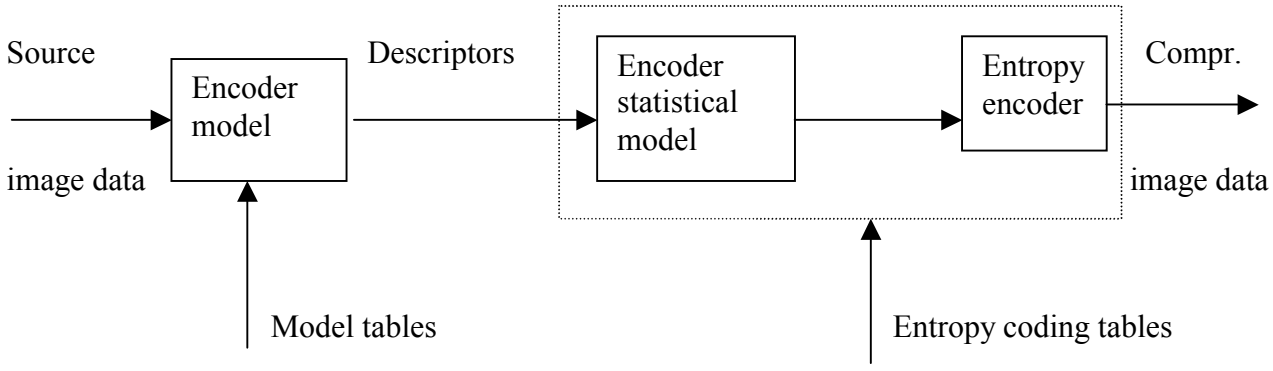

Figure 2. The encoder with basic parts

The two main functional blocks in Figure 2 divide the system in a way that is slightly different from the classical division found in many image compression books. Figure 2 follows the convention used in the JPEG "Joint Photographic Experts Group" documentation, where the division of the system is made at a point that cleanly separates the modelling from parts that are dependent on the particular form of entropy coding used.

The encoder may require external tables-that is, tables specified externally when the encoder is invoked. We define two classes of these tables. Model tables are those tables that are needed in the procedures that generate the descriptors. Entropy-coding tables are those tables needed by the JPEG entropy-coding procedures.

1.2 Decoder structure

A data compression decoding system can be broken into basic parts that have an inverse function relative to the parts of the encoding system. The entropy decoder decodes a sequence of descriptors that exactly matches the sequence that was compressed by the entropy encoder. The sequence of descriptors is converted to a reconstructed image by the decoder model. The decoder requires the same model tables and entropy-coding tables as the encoder. If they are not known to both a priori, the encoder must transmit the tables as part of the compressed data.

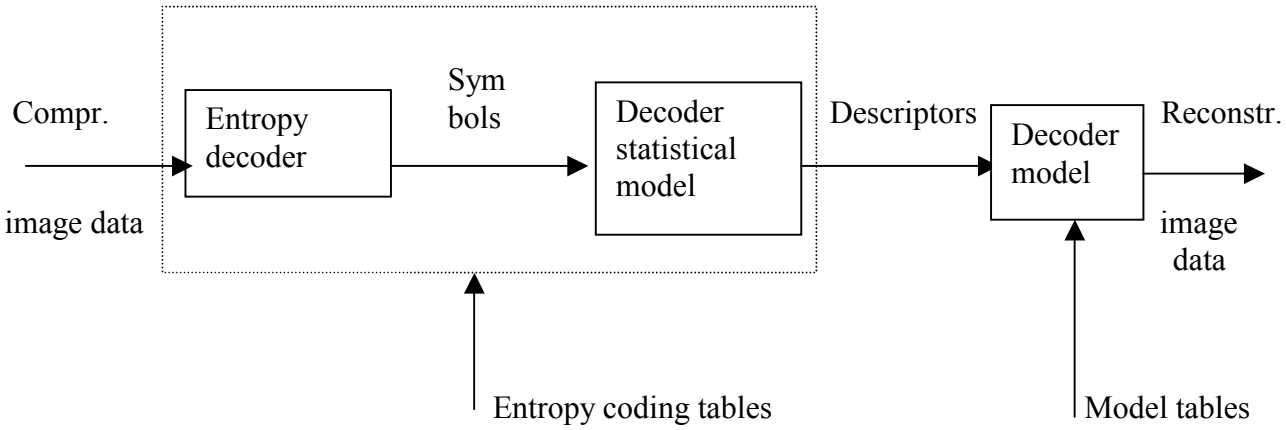

Figure 3. The decoder with basic parts

Unless the compressed image data have been corrupted (by errors introduced during storage or transmission), the descriptor input to the entropy encoder and the output from the entropy decoder are identical. Any loss or distortion, any difference between the source and only the encoder and decoder models introduce reconstructed image data. One of the goals of image compression is to minimise the amount of distortion, especially the amount of visible distortion, for a given amount of compression.

Models that introduce distortion are termed "lossy" models, whereas models that allow no distortion are termed "lossless." The entropy encoding and decoding systems are also "lossless," as this is a term applied to any coding system or subsystem in which the input to the encoder is exactly matched by the output from the decoder.

2 Image compression models

The encoder model is defined as a unit that generates a sequence of descriptors. In this section we shall give some examples that should make this definition a little less abstract. Suppose

we have some image data that we need to code. We could simply feed the sample values to the entropy encoder, in which case the encoder model in Figure 2 is essentially an empty box. This is the simplest possible encoder model; it is also one of the poorest from the stand- point of compression of greyscale images. It is known as "pulse code modulation," or PCM. For PCM the set of descriptors is all possible values of the samples.

A much better model is realised if one attempts to predict the sample value from a combination of samples already coded in the image (the decode must know those values too, so it can make the same prediction). For example, assuming we are coding from left to right, we might use the sample to the left as an estimate for the current sample. Then, the descriptor fed to the entropy encoder can be the difference between the sample being coded and the sample to the left. Since in a continuous-tone image the differences between one sample and the next are likely to be small, it is more efficient to encode the difference values than to encode each sample independently. This class of coding models is known a "differential pulse code modulation," or DPCM. For DPCM the set o descriptors is all possible difference values.

Given a known starting value, differences from one sample to the next can be used to exactly reconstruct the original intensity values. Therefore the set of differences is a representation that is entirely equivalent to the original intensity values. However, creating a representation in which few events or symbols are very probable is important for data compression The essence of data compression is the assignment of shorter code word to the more probable symbols, and longer code words to the less probable symbols. We might suspect, therefore, that DPCM gives better compression than does PCM, and indeed, it does. Phrased another way, the better performance of DPCM relative to PCM is due to the high degree of correlation found in most images. Only if there is strong correlation will the small differences be very probable.

A schematic of a DPCM encoder model that uses the neighbouring sample to the left as the prediction is shown in Figure 4. The corresponding decoder model is shown in Figure 5.

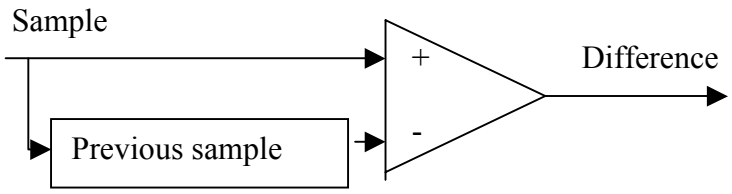

Figure 4 . DPCM encoder model

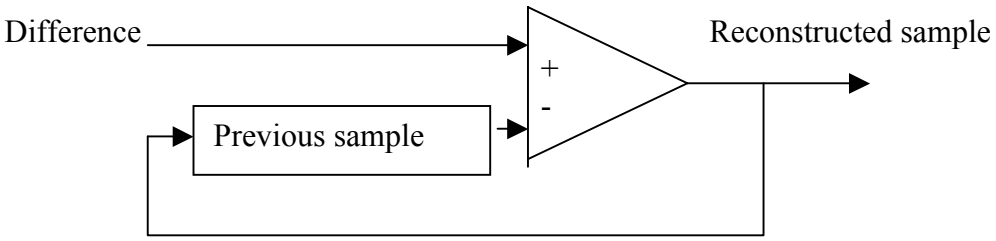

Figure 5. DPCM decoder model

In these figures subtraction is indicated by "-" and addition by "+". There is an implicit storage of at least one sample in both schematics, a differences are always calculated relative to the previous sample.

In two places JPEG uses forms of DPCM similar to that shown in Figure 4: in the lossless coding mode and in the coding of the DCT DC coefficient. Note that in lossless coding other predictors besides the value of the neighbouring sample to the left are allowed. For many images the most effective predictor is an average of the samples immediately above and to the left.

2.1 DCT model

The DCT "Discrete Cosine Transform" (Composed from the FDCT "Forward Discrete Cosine Transform" and IDCT "Inverse Discrete Cosine Transform", as shown in following math. forms) and the associated quantization dequantization of the coefficients C(u) are part of the encoder and decoder models used in the lossy JPEG modes. This encoder model is shown Figure 6.

FDCT:

$$S(u) = \frac{C(u)}{2} \sum_{x=0}^{7} s(x) \cos\left[\frac{(2x+1)u\pi}{16}\right]$$

IDCT:

$$s(x) = \sum_{u=0}^{7} \frac{C(u)}{2} S(u) \cos\left[\frac{(2x+1)u\pi}{16}\right]$$

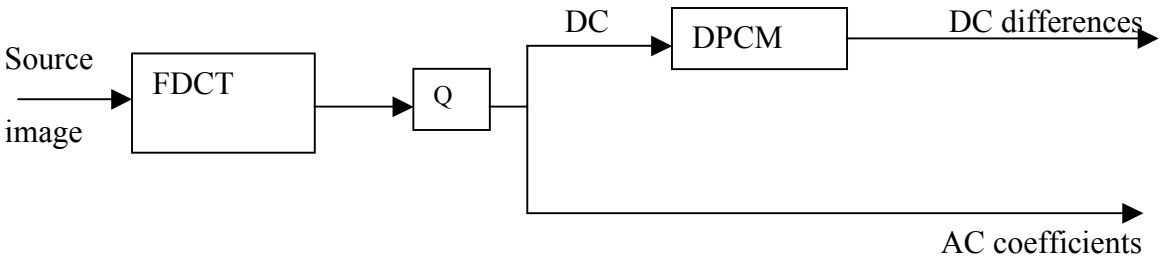

Figure 6. Encoder model for DCT systems

Quantization is done in the box labelled "Q." Note that the DC is fed separately to an additional stage containing a DPCM model. DC differences and AC coefficients are the descriptors that are fed the entropy-coding block. The entropy-coding block codes these two classes of descriptors differently.

The corresponding decoder model for the lossy JPEG modes is shown in Figure 7. The dequantization is done in the box labelled "DQ.' The FDCT, quantization, dequantization, and IDCT are the cause of the distortion in the images reconstructed by a JPEG lossy decoder. Arithmetic approximations in the integer arithmetic typically use computing the FDCT and IDCT

introduce a small amount of this distortion. The principal source of loss or distortion, however, is quantization and dequantization of the coefficients.

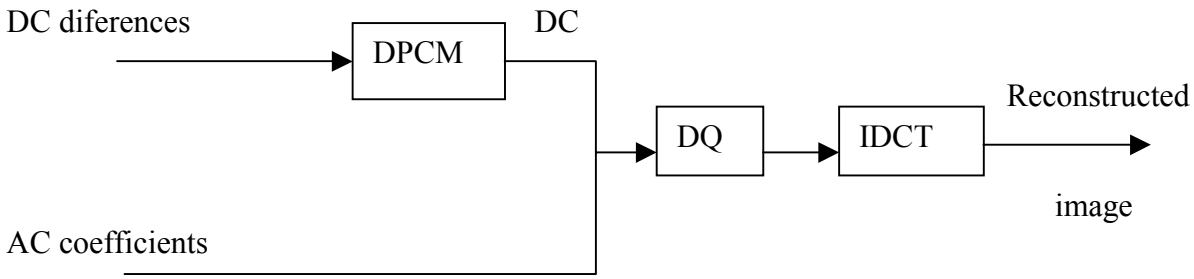

Figure 7. Decoder model for DCT systems

The quantization of each coefficient is done independently and can therefore be matched to the response of the human visual system. It is this aspect that makes the JPEG lossy modes so effective. However, the reader should recognise that the quantization rules are defined for a particular of viewing conditions, image content, and perhaps even application demands. If these change, the distortion introduced by quantization may become a problem. There are additional attributes of DCT models that help to simple the statistical modelling, one of them being the almost ideal decorrelation between the DCT basis functions.

2.2 Other models

The DPCM model is one particular instance of a general class of cod known as predictive coding. In predictive coding information already s is used to predict future values and the differences are coded. The D model is also a particular form of a general class of coding known transform coding. There are many other classes of coding models. Because our goal in this book is to provide a complete review of technologies that apply specifically to JPEG, we shall not discuss the ml other models that have been devised for image coding.We note, however that excellent implementations of block truncation coding, vector quantization, other transform coding schemes, sub-band coding, and several predictive coding schemes were considered by JPEG as candidates for the standard. JPEG's selection in January, 1988, of a DCT-based approach for lossy compression reflects the fact that of all the proposals submitted, the DCT provided the best image quality for a given bit rate.

3 JPEG entropy encoder and entropy decoder structures

JPEG uses two techniques for entropy coding: Huffman coding and arithmetic coding. Huff'man coding, devised about 40 years ago, is the more familiar of the two, is computationally simpler, and is simpler to implement. However, it requires the code tables to be known at the start of entropy coding. Arithmetic coding provides systematically higher compression performance (typically 10% or more) and one-pass adaptive coding in which the "code book" adapts dynamically to the data being coded. In this section we shall only introduce the major functional blocks in these entropy coders. The principles of entropy coding and the actual coding procedures will be described in a later.

Inside an entropy coder there are four basic building blocks: a "statistical model," an "adaptor," a storage area, and an encoder. The way the building blocks are linked together is a function of the

particular entropy coding. The statistical model translates the descriptors into "symbols", each symbol being assigned a particular code word or probability. The adapter is responsible for the assignments of code words (Huffman coding) or probability estimates (arithmetic coding) needed the encoder. The storage area contains the Huffman code table or arithmetic-coding probability estimates. With the help of the code words or probability estimates in the storage area, the encoder converts the symbols to bits of compressed data.

The symbols created by the statistical model are members of "alphabet" that contains all possible symbols that the model can generate. Usually, some of these symbols are quite probable and some are v improbable. The objective in entropy coding is to assign short code words to the very probable symbols and longer code words to the less probable symbols. Samuel Morse did this in an intuitive way more than a cent, ago when he invented Morse code. When the code assignment is done an optimal way, the result is entropy coding.

3.1 Huffman entropy encoder and decoder structures.

The four basic building blocks of the Huffman entropy encoder are illustrated in Figure 8. The Huffman statistical model converts descriptors into the actual symbols that are coded. For example, the JPEG Huffman statistical model converts 16 contiguous zero AC coefficients into a single symbol meaning a run of 16 zeros. This type of conversion is usually done to improve the efficiency of the Huffman codes. The symbols created by the statistical model are directed by switch S1 to either the Huffman encoder or the Huffman adapter.

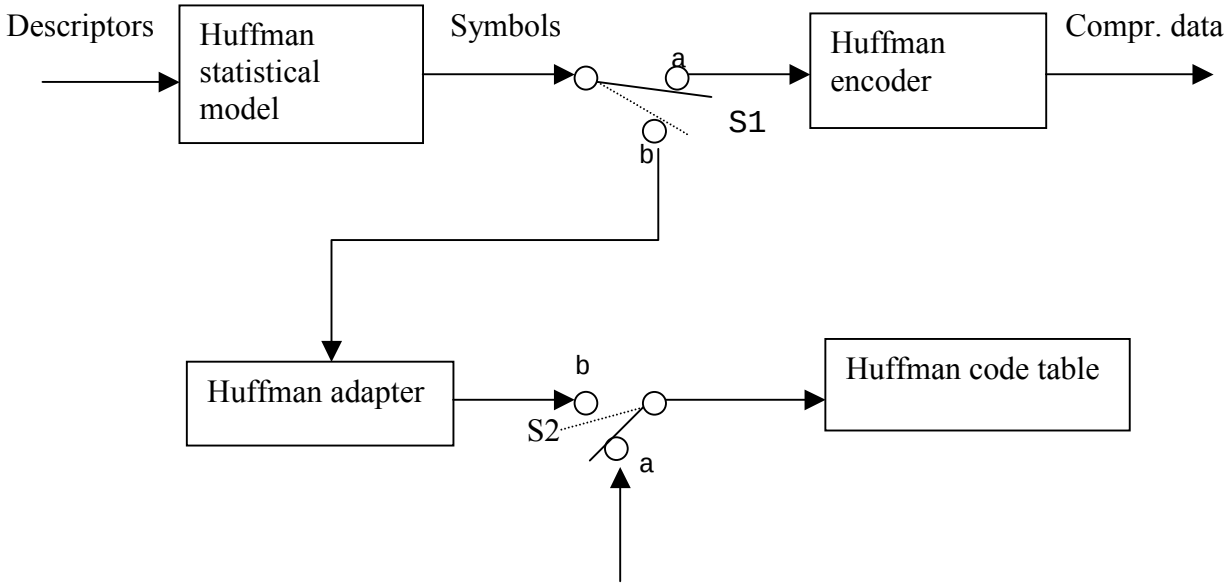

Figure 8. Huffman entropy encoder

Whenever the adapter is used, two passes through the data are required. In the first pass the symbols are fed to the adapter, in which the data is analysed and used to create a "custom" Huffman code table. (Huffman code tables are the "entropy coding tables" for a Huffman entropy coder.) The second pass then encodes the data. Since two passes through the data are not always convenient, switch S2 allows "fixed" Huffman code tables from an external source to be used instead. Custom tables typically improve the coding efficiency by a few percent relative to the efficiency achieved with fixed tables.

The Huffman entropy decoder is illustrated in Figure 9.

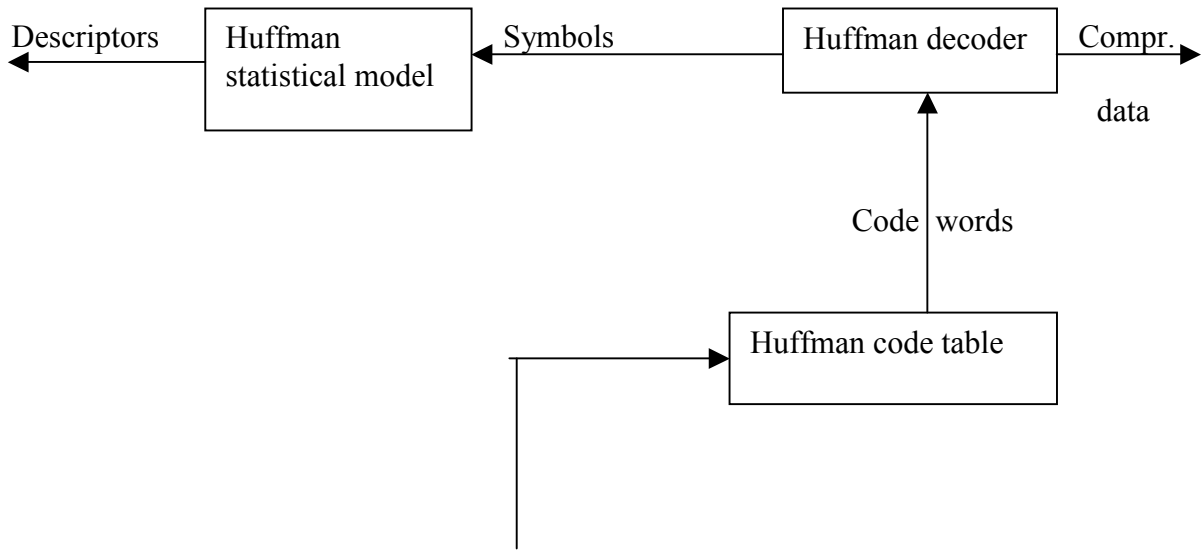

Figure 9. Huffman entropy decoder

The Huffman compressed data are decoded into symbols by the Huffman decoder block. The code words always come from a fixed Huffman code table that is loaded into the Huffman code table storage area before decoding begins. The data needed to generate this table may be incorporated into the compressed data. The symbols are translated back into the descriptors by the Huffman statistical model block.

3.2 Arithmetic entropy encoder and decoder structures

The four basic building blocks of the arithmetic-coding entropy encoder are illustrated in Figure 10. JPEG uses an adaptive binary arithmetic coder that can code only two symbols, 0 and 1. (This may seem restrictive until you realise that computers also use only binary symbols.) Therefore the arithmetic coc statistical model must translate the descriptors into a set of binary decisions. The statistical model also generates a "context" that selects a particular probability estimate to be used in coding the binary decision binary decisions and contexts are fed in parallel to both the arithmetic coding adapter and the arithmetic coder. The arithmetic-coding "conditioning table" (the arithmetic coding version of the entropy-coding table provides parameters needed in generating the contexts.

The arithmetic encoder encodes the symbol using the probability estimate supplied to it. In the process of coding, it also keeps an approximate count of the 0's and 1's, and occasionally signals the adapter to tell it to make the probability estimate for a particular context larger or smaller. The arithmetic coder is identical to that adopted by JBIG known as "Joint Bi-level Image Experts Group". Consequently, everything but the statistical model and perhaps the size of the probability storage area are unchanged when used by JBIG. Note that the JPEG/JBIG arithmetic entropy coder is a single-pas adaptive coder.

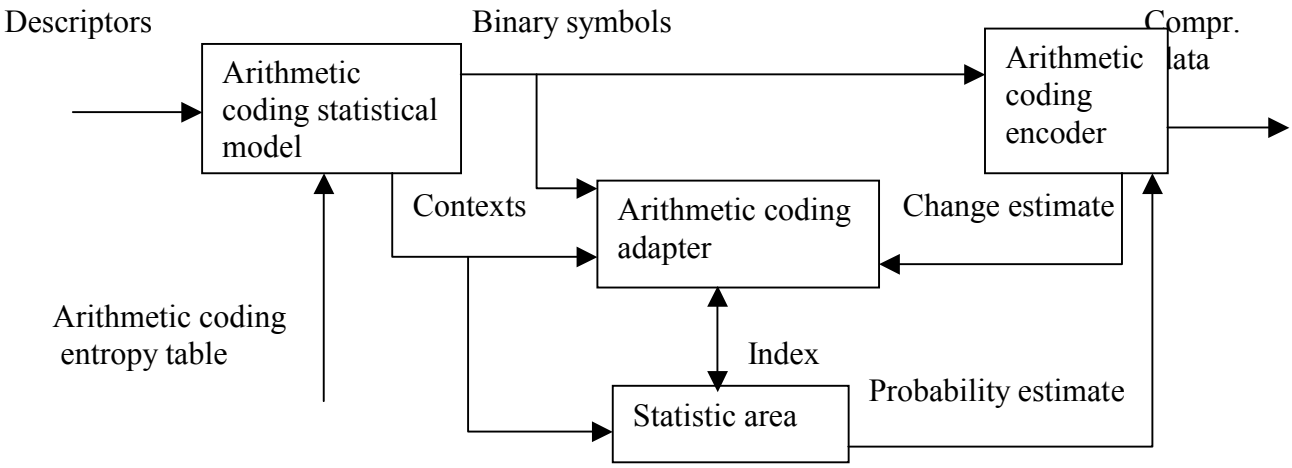

Figure 10. Arithmetic entropy encoder

The arithmetic entropy decoder is illustrated in Figure 11. The compressed data are fed to the decoder, which, with the benefit of the same probability estimates used for encoding, determines the sequence of binary symbols. The binary symbols are in turn translated back into descriptor: by the arithmetic-coding statistical model. Note that the decoder and encoder statistical models generate the same context for a given binary decision. In general, the contexts, adaptation, and probability estimate must be identical in the entropy encoder and decoder

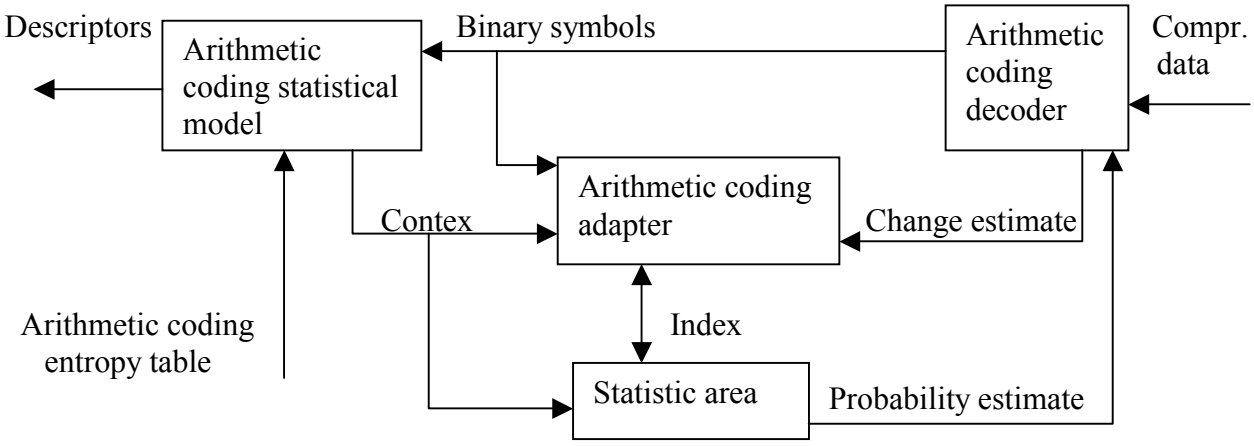

Figure 11. Arithmetic entropy decoder

4 Transcoding

Because the Huffman and arithmetic entropy coders encode and decode the same set of descriptors, it is possible to "transcode" between these systems Figure 12 illustrates this procedure for a conversion from Huffman compressed data to arithmetic-coding compressed data. Of course, transcoding is a reversible process and compressed data generated by an arithmetic coder can be transcoded to Huffman coded compressed data.

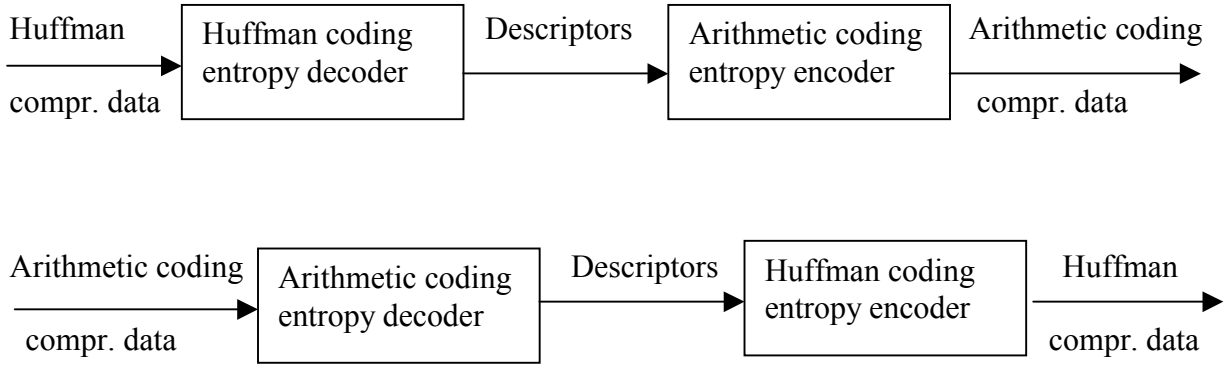

Figure 12. Transcoding between entropy coders

5 Conclusion

JPEG defines both lossy and lossless compression processes. The lossy JPEG processes are based on the Discrete Cosine Transform (DCT), whereas the lossless processes are based on forms of DPCM.

Lossy JPEG DCT-based compression is particularly effective in minimising visible distortion. The improvement in compression that results when a small amount of visible distortion is allowed can be quite significant. Colour images that can be compressed by a ratio of about 2:1 with the JPEG lossless techniques can often be compressed with JPEG lossy techniques by more than 20:1 yet have nearly imperceptible levels of visible distortion in the reconstructed images.

When judging compression for a given degree of visual distortion compression ratios can be misleading. Compression ratios can be artificially inflated by using excessively high precision or too many samples in the chrominance components. It would be better to state that nearly imperceptible distortion is achieved at compressions of a little more than 1 bit/pixel with the JPEG lossy techniques. Of course, bits/pixel can also be misleading, as it can be radically affected by changes in viewing conditions and picture resolution. Really meaningful comparison between compression systems requires a fixed set of viewing conditions a carefully calibrated display system, and a set of images with scene content and resolution that are representative of the application. The real cost in storage and transmission comes from the amount of compressed data required to represent the image at some particular quality.

6 References

[1] Pennebaker,W.,B., Mitchel, J .,L.: JPEG Still image data compression standard. Van Nostrand Reinhold, New York, 1993 ISBN 0-442-01272-1
[2] Fabian,P., Capek,J.: Information Systems: Coding and Compression. INSYPA, TEMPUS JEP 11572 Workshop Proceedings, Huddersfield 1999, ISBN 80-7194-203-0
[3] Kállay,F.,Peniak,P.: Počítačové sítě a jejich aplikace, Grada 1999, ISBN 80-7169-407-X
[4] Held,G., Marshall,R.T.: Data and Image Compression Techniques, John Wiley &sons, Chichester, 1996.

Kontaktní adresa:

doc.Ing.Jan Čapek,CSc., Katedra informačních systémů, FES, UPa,
Studentská 84, 532 10 Pardubice
e-mail: Jan.Capek@upce.cz

Ing.Peter Fabian,CSc., Katedra informačních systémů, FES, UPa,
Studentská 84, 532 10 Pardubice
☎ 040-603 6038, fax: 040-603 6010

Recenzoval: doc.Ing.Karel Šotek,CSc., Katedra informatiky v dopravě, DFJP, UPa