

(19) World Intellectual Property Organization
International Bureau(43) International Publication Date
14 August 2003 (14.08.2003)

PCT

(10) International Publication Number
WO 03/067763 A2(51) International Patent Classification⁷: **H03M**

(21) International Application Number: PCT/EP03/01130

(22) International Filing Date: 5 February 2003 (05.02.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
02002641.5 5 February 2002 (05.02.2002) EP(71) Applicant (for all designated States except US):
SIEMENS AKTIENGESELLSCHAFT [DE/DE];
Wittelsbacherplatz 2, 80333 München (DE).

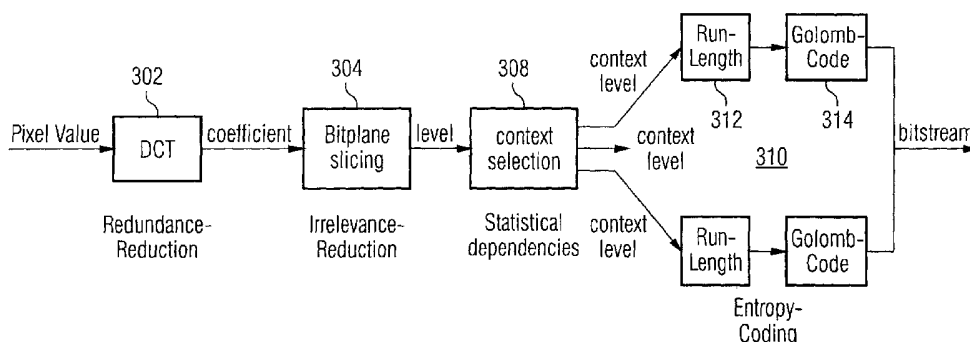
(72) Inventor; and

(75) Inventor/Applicant (for US only): **ILLGNER-FEHNS,**
Klaus [DE/DE]; Marquartsteiner Str. 12, 81549 München
(DE).(74) Common Representative: **SIEMENS AKTIENGE-**
SELLSCHAFT; Wittelsbacherplatz 2, 80333 München
(DE).(81) Designated States (national): AE, AG, AL, AM, AT, AU,
AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU,CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH,
GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC,
LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW,
MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE,
SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ,
VC, VN, YU, ZA, ZM, ZW.(84) Designated States (regional): ARIPO patent (GH, GM,
KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW),
Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM),
European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE,
ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, SE, SI,
SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN,
GQ, GW, ML, MR, NE, SN, TD, TG).**Declarations under Rule 4.17:**

— as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW, ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB,

[Continued on next page]

(54) Title: DATA COMPRESSION



(57) **Abstract:** Apparatus and method for quantizing and entropy encoding transform coefficients in the encoding of multimedia data, for example video and audio data. Pixel value data is input into a decorrelator, 302. The decorrelator generates reduced transform coefficients. A quantizer 304, for example a bit plane slicer, is used to remove irrelevant data from the reduced coefficients. Statistical dependencies in the bit planes are exploited by the application of a context selection means 308, which in turn generates multilevel context paths. The multilevel context paths are input into an entropy encoding mechanism 310. In one embodiment, the entropy encoding mechanism comprises: a run-length pre-coder 312, for receiving input data at each context level and outputs run-length symbols; and an analytical entropy coder 314, which receives the symbols and generates a compressed data stream. The above stages are reversed in decoding.



WO 03/067763 A2



GR, HU, IE, IT, LU, MC, NL, PT, SE, SI, SK, TR), OAPI
patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML,
MR, NE, SN, TD, TG)

— of inventorship (Rule 4.17(iv)) for US only

For two-letter codes and other abbreviations, refer to the "Guid-
ance Notes on Codes and Abbreviations" appearing at the begin-
ning of each regular issue of the PCT Gazette.

Published:

— without international search report and to be republished
upon receipt of that report

DATA COMPRESSION

Field of the invention

The present invention relates to a method of data compression and, in particular, but not restricted thereto, to a method of quantizing and entropy encoding transform coefficients in the encoding of multimedia data.

Background to the invention

Video coding systems are presently being developed for a diverse range of technologies including cable television, video conferencing and interactive services for mobile video handsets. As the capacity of communications networks to transfer large amounts of data increases, access to interactive video/audio systems has become increasingly common. Systems and standards capable of supporting such increasing data transfer needs are also being developed.

The Moving Picture Experts Group (MPEG) is a working group of researchers and developers who work in conjunction with the International Standards Organisation (ISO/IEC) in the development of standards for coded representation of digital audio and video. Other standards bodies are involved with the development of digital audio and video standards, a further example being ITU. To date these standards bodies are responsible for the creation and implementation of several audio and video standards including compression algorithms, processing schemes and authoring environments.

MPEG has produced: the MPEG-1 standard for storage and retrieval of moving pictures and audio; the MPEG-2 standard for digital television; and, more recently, the MPEG-4 standard for multimedia applications. MPEG are

currently developing MPEG-7, a content representation standard for multimedia content description interfaces.

MPEG-4 provides an object-based coding standard which gives flexibility to authors, service providers and end users of highly interactive, multimedia content. Whilst natural audio and video content (recorded audio and video) is still addressed by MPEG-4, the standard now also addresses the coded representation of synthetic (computer generated) audio and visual objects (AVOs). MPEG-4 was developed to provide the necessary facilities for specifying how AVOs, both natural and synthetic, can be composed together in space and time whilst allowing for interactive delivery. A further focus in the development of the MPEG-4 standard was the performance of coding and transmission of such content within a variety of different scenarios. The standard was consequently designed to accommodate “universal access” to multimedia information by taking into account possible bandwidth-specific problems presented by a wide variety of network situations.

Individual object streaming is another development within MPEG-4. Within any given MPEG-4 scene, a series of video objects may be streamed individually to an end user’s computer. Each of these individual streams can be time-stamped to enable synchronisation. Since the individual elements being streamed are independent of one another, they can easily be dropped or added in accordance with the interactive needs of the scene and the user’s involvement with that scene.

The presence of multiple streams engenders the need for information regarding the network itself: for example, the synchronisation of an entire scene could be jeopardised if congestion occurs. To circumvent potential congestion problems, MPEG-4 has included a set of descriptors within each of the streams which provide information relating to the required decoding

software, necessary timing information and required quality of service (QoS) requirements for the scene to be successfully streamed. Consequently, MPEG-4 is arranged to be capable of flexible delivery of multimedia content over a variety of operational environments and network situations. MPEG-4 is also arranged to have the ability to provide information as to the complexity of the scenes being transmitted, the QoS required to transmit such scenes and the synchronisation of the elements within the scene.

An important feature of data storage and transfer is the degree of compression of the data. Data compression reduces the bandwidth required to transmit and store data. Compression is a reversible operation, which converts data from an initial uncompressed data format to a compressed format that requires fewer bits. Data which has been compressed can be stored or transmitted more efficiently. When required, the operation can be reversed and the original uncompressed data can be reproduced.

Reproduction or reconstruction of the original data can be classed as either lossless or lossy. In a lossy compression system, the source data cannot be reconstructed perfectly: in a lossless system, it can be. Clearly, lossless compression is most suitable where perfect reconstruction is desirable, for instance the compression of text data or in the compression of medical or artistic images. By contrast, a certain degree of information loss may well be tolerable, as for example in the compression of most still image, audio or video data.

A compression operation can be considered as having two component stages. The first stage comprises an encoder model stage in which redundancy and other irrelevant detail is minimised. The second stage comprises an entropy encoding stage in which statistical properties are exploited to generate a very much reduced set of symbols which represent the

information distilled from the original uncompressed data and in which the symbols themselves are entropy encoded for the greatest encoding efficiency.

By careful choice of code, the statistical properties of the symbols, for instance the distribution of zero's and one's in the binary representation of the symbols, can be used to further compress the information of the original data. Runs of zero's can be encoded efficiently by transmitting a number representing the length of the current run rather than the zero values themselves. This technique is known as run-length coding (RLC).

RLC is often used as a pre-coding technique to prepare the data for a technique which exploits the frequency of occurrence of certain symbols. Variable length coding (VLC) encodes non-zero coefficients by representing the more frequently occurring symbols with short codes (codes having shorter bitlengths) and less frequently occurring symbols with longer codes. VLCs are not necessarily static and if the data being represented changes in its frequency characteristics the VLC technique can be arranged to allow for the assignment of codes to symbols to be adapted accordingly. The current set of assignments are stored in a VLC table.

VLC can be illustrated by analogy to the encoding of text for transmission along a telegraph wire using Morse code. When encoding substantial passages of text in the English language, a frequency distribution pattern will emerge. Certain letters are observed to occur very frequently, for example E, T, A, O and N, and certain other letters comparatively infrequently, for example Q and Z. So that telegraph operators took less time to transmit messages, Morse code assigns its shortest codes to more frequently occurring letters and considerably longer codes to the rarer letters. Morse code is an example of a static VLC.

As an entropy encoding technique, variable length coding (VLC) has advantages and disadvantages. VLC tables are easy to construct offline, for

example by the Huffman algorithm. Encoding using VLC tables is very simple since encoding simply comprises the indexing of a large table with an entry for each possible symbol. Decoding is a bit more complex as it requires the bitstream to be parsed and the identification of variable length code words. If memory is not a constraint, decoding can also be realised as a look-up table.

VLC tables do however require memory for storage. Symbol statistics, the probabilities of the occurrences of input symbols, vary between sequences of symbols. In the Morse code analogy, the language being represented may be different, say Polish rather than English, or the type of information being represented may change, for instance stock market data rather than business letters. A single table can reflect only a more or less well-matching average and will thus limit the compression efficiency. Employing several tables for different symbol statistics becomes infeasible as this requires a significant amount of memory for storing all the tables, regardless of the encoding (or decoding) approach. The underlying statistics of the VLC tables must therefore be calculated or measured online to adapt to the statistical changes measured between sequences symbols, in a process known as adaptive VLC. The calculation or measurement increases not only the computational load in that a complete new code table must be generate each time but also carries the risk that encoder/decoder mismatches will occur in the event of transmission errors. The potential for mismatches explains why this approach is not more widely utilised.

In VLC tables, at least one bit is invariably assigned to the most probable code word. This assignment in itself constitutes a significant source of redundancy. Encoding more than one symbol simultaneously would improve the performance but at the same time would increase VLC table size considerably.

An alternative approach to entropy encoding uses context-based arithmetic coding. Arithmetic coding has become a standard entropy encoding technique and is now included as an option in ITU-T Recommendation H.263.

5 Arithmetic coding is a method of encoding data using a variable number of bits. The number of bits used to encode each symbol varies according to the probability assigned to that symbol. Low probability symbols use many bits, high probability symbols use fewer bits. Unlike Huffman coding, however, arithmetic encoders do not have to use an integral
10 number of bits to encode each symbol. If the optimal number of bits for a symbol is 2.4, a Huffman coder will probably use 2 bits per symbol, whereas the arithmetic encoder may use very close to 2.4.

In arithmetic coding, a sequence of symbols, $X_1, X_2, \dots, X_n$, is processed as a vector from left to right in order. The symbols are not replaced by a
15 string of codebits but rather each symbol, X_i , is assigned a distinct subinterval, I_i , of the unit interval $[0,1]$. The subintervals are chosen so that $I_1 \supset I_2 \supset \dots \supset I_n$ and so that I_i is determined from the previously processed samples, $X_1, X_2, \dots, X_{n-1}$ in conjunction with the current sample X_i . Where the assignment of subintervals is altered in accordance with changes in the
20 probabilities of symbols in the previously processed samples, the arithmetic coding is said to be adaptive. Once the final interval I_n has been determined, a code word $(B_1, B_2, \dots, B_L)$ is assigned to the entire sequence such that: the rational number $.B_1B_2\dots B_L$ given by: $.B_1B_2\dots B_L = B_1/2 + B_2/4 + \dots + B_L/2^L$ is a point in interval I_n .

25 In summary, VLC (Huffman codes) and arithmetic coding yield the same performance from a theoretical point of view. Arithmetic coding however performs better in practical applications. The better performance stems from the fact that the minimal number of bits per symbol can be less

than one in arithmetic coding. In effect, arithmetic coding combines the fractional bits of consecutive symbols. This means an arithmetic coder can usually encode a message using fewer bits than a Huffman coder.

5 Rather than having to generate a new tables for each new statistical distribution pattern, arithmetic coding can be self-adaptive to the incoming data stream.

10 Furthermore, no code tables need be stored in arithmetic coding. In considering an alphabet of possible input symbols, only the probabilities of occurrences of each symbol of the alphabet must be known. Provided the alphabet remains small, no significant memory is needed. As will be demonstrated, this feature is exploited in context-based bitplane coding, where the alphabet is very small and for each context a different statistic is used.

15 The context selection is arranged to exploit statistical dependencies between neighbouring symbols, which in turn leads to an increase of the probability of one (or a limited number of) symbols. These increased probability symbols are then assigned very short theoretical code words (less than one bit per symbol when consecutive symbols are combined). For these symbols arithmetic coding can efficiently combine the fractional bits to give
20 very short real code words.

Arithmetic coding does have some notable deficiencies: in particular a complexity which is high in comparison to entropy encoding using RLC/VLC. Furthermore, arithmetic codes are very sensitive to bit errors. Whilst VLC functions can be adapted to have automatic resynchronisation characteristics or at least to allow for the identification of decoding errors,
25 arithmetic codes can not.

Other entropy encoding techniques are known. These techniques are not however enshrined in the current standards. Generally speaking they are

based on scanning the bits of the incoming data stream in a predefined order prior to Huffman coding.

The present invention seeks to provide an improved coding scheme. In particular, the present invention seeks to provide a coding scheme which has a reduced dependency upon large tables and to provide a simple and fast encoding and decoding mechanism. Further the present invention seeks to provide a coding scheme having a good capability to adjust to different signal statistics, the ability to recognise bit errors/decoding errors and the ability to assign less than one bit of data per symbol by exploiting statistical dependencies.

Statement of the invention

In accordance with one aspect of the present invention there is provided a method of encoding source data comprising the steps of: decorrelating source data to generate transform coefficient information; slicing said transform coefficient information into a plurality of bitplanes, said plurality of bitplanes including at least one least significant bitplane; context selecting from the bitplanes to generate a plurality of context levels; and for at least some of the plurality of context levels, applying run-length pre-coding to generate sequences of pre-coded symbols and then encoding said pre-coded symbol sequences by an analytical entropy coding scheme.

Advantageously the analytical entropy coding scheme is a Golomb coding scheme.

The symbol sequences are preferably binary symbol sequences.

The bitplane slicing is advantageously performed by the Zero-tree approach.

In accordance with a further aspect of the present invention there is provided a data encoding apparatus comprising:

a redundancy reduction processor,
a relevant data reduction processor,
a context selection processor; and
an entropy encoding processor:

5 characterised in that the context selection processor generates a plurality of context data output streams, each of said context data outputs having a corresponding context level; and in that at least some of the plurality of context data output streams is input into said entropy encoding processor.

The coding processor preferably comprises a run-length pre-coder and
10 an analytical entropy coding processor.

The context selection stage preferably includes measuring the frequency of zeros or alternatively ones for each of a plurality of contexts, generating a probability model and comparing the measured frequency of zeros for each of the contexts with the probability model. A selected context
15 is advantageously selected by establishing which of the available contexts best matches the probability model for which a Golomb Code of order m is optimal.

Brief description of the figures

20 The invention may be understood more readily, and various other aspects and features of the invention may become apparent, from consideration of the following description and the figures as shown in the following drawing sheets, wherein:

Figure 1 shows a block diagram of a source encoding process using
25 variable length encoding;

Figure 2A shows a block diagram of a source encoding process as it is used for example in the MPEG-4 FGS profile

Figure 2B shows a block diagram of a modified source encoding process with context-selective adaptive arithmetic coding; and

Figure 3 shows a block diagram of source encoding process in accordance with the present invention.

5

Detailed description of the present invention

Figure 1 diagrammatically illustrates a flow diagram of the stages in one well-known data compression technique. The encoder model stage first reduces redundancy as much as possible. To this end, a redundancy reduction function, for instance discrete cosine transform (DCT), is applied to the initial data. Redundance reduction is often referred to as decorrelation in the video coding literature since the transforms applied to the incoming data upset the spatial correlation between neighbouring pixels of the original image.

The redundancy reduction function outputs a reduced set of transform coefficients which still however contain a certain amount of subjectively irrelevant detail. The irrelevant data is then removed by the application of a quantization function. The removal of irrelevant data is irreversible and results in an overall loss of information.

The loss of information is represented schematically in Figure 1 by the introduction of a feature known as a quantizer. Quantizers, as the name suggests, have the effect of “rounding off” values - in effect reducing the number of symbols used to represent original data values in comparison to the ‘alphabet’ of possible values taken by transform coefficients. Quantizers generate output as quantizer level symbols.

Statistical dependencies still exist between the quantizer levels. The subsequent entropy encoding stage seeks to exploit those statistical dependencies.

A run-length pre-coding function (RLC) is applied to the quantizer level symbols and by exploiting statistical dependencies between the level symbols further reduces the number of bits required to represent the original data. Essentially “runs” of the same quantizer level symbol are represented as
5 (shorter) run-length symbols having a fixed bitlength.

The run-length symbols are further compacted by applying specifically designed variable length code tables, for example using Huffman coding. The unequal probabilities of the occurrences of each run-length symbol allows entropy coding by the assignment of the shortest VLC to the most frequently
10 occurring run-length symbol.

The compression technique described above has numerous applications, including still image and video coding and notably including the well-known JPEG standard.

Another form of coding, which is presently included in the MPEG-4
15 FGS profile and in the forthcoming JPEG 2000 standard, takes the reduced transform coefficients from the output of a DCT function and applies first bit-plane coding, then pre-coding for runs of identical values, and finally assigns variable length codes according to the probability of the occurrence of the run length symbols. This form of coding can be seen as a specific embodiment of
20 the scheme illustrated in Figure 1, the quantizer being a bit-plane slicer. As will be understood by the reader, the effect of discarding the least significant bit planes generated by a bit plane slicer is to round off the values of the transform coefficients and thereby reduce the number of symbols required to represent the data stream. A block diagram of the flow of the MPEG-4 FGS
25 profile is shown in Figure 2A.

Figure 2B shows a flow diagram of the stages in another well-known data compression technique. Here the entropy coding is achieved using context-based arithmetic coding rather than the RLC/VLC scheme of Figure

2A. To that end, the quantizer known from Figure 1 is again replaced by a mechanism for slicing transform coefficients into bitplanes.

A key aspect for the subsequent arithmetic coding is to maintain a small symbol alphabet.

5 Statistical dependencies are then selected by a context selection means which provides multilevel context paths to the subsequent entropy encoding mechanism. The selection of a certain context tries to predict the current symbol from its neighbouring symbols. The high probability of the current symbol occurring in a specific context can then be utilised efficiently by a
10 subsequent entropy encoding mechanism. In the case illustrated in Figure 2B the subsequent entropy encoding mechanism is an adaptive arithmetic coding mechanism.

Variants of bitplane slicing mechanisms exist, for example the Zero tree approach.

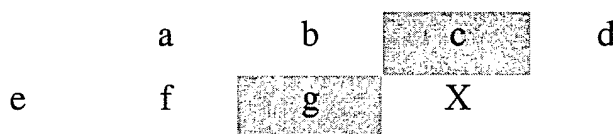
15 There will now be described, by way of example, the best mode contemplated by the inventors for carrying out the invention. In the following description, numerous specific details are set out in order to provide a complete understanding of the present invention. It will be apparent, however, to those skilled in the art, that the present invention may be put into
20 practice with variations of the specific details.

Referring now Figure 3, there is shown a block diagram of a process 300 for encoding image data in accordance with the present invention. As is known, pixel value data is input into a redundancy reduction processor 302: in Figure 3, the redundancy reduction processor is a DCT function. The
25 redundancy reduction processor 302 generates reduced coefficients. An irrelevant data reduction processor 304 takes the reduced coefficients as input. The reduction of irrelevant data is conveniently performed by a bit plane slicer. From the irrelevant data reduction processor 304, statistical

dependencies are selected in a context selection means 308 whereby to provide multilevel context paths in an entropy encoding mechanism 310.

By comparing Figures 2B and 3, one observes that the two encoding techniques differ in their entropy encoding mechanisms. In the inventive technique of Figure 3, a run-length pre-coder 312 receives input data at each context level. The run-length pre-coder 312 then outputs run-length symbols and passes the symbols to an analytical entropy coder 314. It should be noted that the context selection stage in Figure 2B differs fundamentally from that in Figure 3. Context selection in the present invention allows for progressive encoding and decoding.

To illustrate the distinction, consider a 2D arrangement of samples having the following appearance:



X indicates the pixel to be encoded. In the simplest case of binary encoding, the value at each location, a,b,c, etc., is either '0' or '1'. A context template is formed by considering the values of selected locations. For reasons of simplicity, the context template in this illustration comprises only the shaded locations, namely locations 'c' and 'g' (shaded grey).

A 'context' in this example is a single permutation of the values in the locations of the context template. Considering all possible permutations, we obtain the following table:

	c	g
Context 1	0	0
Context 2	0	1
Context 3	1	0
Context 4	1	1

The context selection technique shown in Figure 3 is intrinsically bound to the subsequent Golomb Code stage. For each context we measure the frequency of $X=0$ and $X=1$. Based on this measured frequency we estimate the parameter m of the Golomb-Code. The goal is to select contexts
 5 such that the measured frequency of 0 for each context gets closest to the probability model for which the Golomb code of order m is optimal.

By contrast the conventional context selection stage of Figure 2B operates to maximise the probability of finding $X=0$ (or 1).

In the embodiment illustrated in Figure 3, the analytical entropy coder
 10 314 makes use of Golomb codes (GC). After entropy coding has been performed the signals are streamed in a single bitstream.

In further embodiments of the invention, the source of the symbol stream upon which first the run length processor and then the analytical entropy coder operate need not be limited to the specific stages illustrated in
 15 Figure 3. The symbol stream can be provided as the results of scanning any binary map or can be the result from slicing coefficients into bitplanes followed by applying appropriately designed context. The symbol stream is not limited to transform coefficients but may be data of any type, including: discrete cosine, integer cosine transform (ICT), finite Fourier, or wavelet
 20 transform coefficients; image intensity values; or even two or three dimensional administration information, MPEG-4 descriptors for example.

The approach described above can be generalised to any kind of two-dimensional plane or three-dimensional array of binary or even non-binary symbols. A coded block pattern (CBP) can be encoded with this technique.

25 The separate processes performed by the processors shall now be explained in greater detail.

In the irrelevance reduction processor (the bitplane slicer), the transform coefficients $G[]$ are sliced into N bitplanes, $\alpha_k[]$:

$$G[i, j] = \sum_k \alpha_k[i, j] 2^k, \quad \alpha \in \{0, 1\}$$

where $k \in \{0, \dots, N - 1\}$.

Each bitplane can be considered as a binary array or map

$$\mathcal{M}_k = \{\alpha_k[i, j] | (i, j) \in \Lambda\},$$

where Λ is a 2D grid.

In the following step, the context selection processor operates on each
 5 bitplane and statistical dependencies between neighbouring pixels are
 exploited by forming contexts C_l . Let $\mathcal{S}_p$ denote one particular binary pattern
 formed by the pixel values in the neighbourhood $\mathcal{N}$ of $b[i, j]$

$$\mathcal{S}_p = \{b[i - n, j - m] | (m, n) \in \mathcal{N}\}$$

As the number of patterns $\mathcal{S}_p$ can get very large, depending on the
 number of elements of the neighbourhood, a context C_l is defined as set
 10 containing at least one pattern $\mathcal{S}_p$. The context must be causal with respect to
 the scan order for continuous decodability, which restricts the elements in the
 neighbourhood $\mathcal{N}$. Moreover, the context can even be extended to three
 dimensions by incorporating pixels in neighbouring higher order bitplanes.
 Associated with each context C_k is a bit buffer $\mathcal{B}_k$ that is operable to store the
 15 binary values identified as “belonging” to context C_k .

The context C_k is defined in order to maximise the probability of one of
 the two possible binary symbols, ‘0’ and ‘1’, with respect to this context. In
 other words, the context selection seeks to exploit statistical dependencies
 between neighbouring symbols. Context selection thus increases the
 20 probability of a correct prediction of the current pixel value. As a
 consequence, the generated bitstream for each context shows a high
 probability for one symbol. Furthermore, depending on the contexts defined,
 the resulting sequences of each context are almost completely statistically
 independent.

When entropy encoding such context selected sequences, arithmetic coding takes advantage of the presence of those symbols which have high probabilities of occurrence and which have a (theoretical) code word lengths of less than one bit.

5 In another embodiment of the present invention, a run-length pre-coder is applied to the context selected binary sequence. The run-length symbols are then input into a GC function to generate an output bitstream. A great advantage of this embodiment is that run-length symbols obtained from a binary sequence of statistically independent binary symbols always show an
10 (exponentially decaying) geometric distribution:

$$\text{pdf}(i) = p^i(1 - p), \text{ where } p < 1,$$

regardless of the underlying distribution of the transform coefficients, where p is the probability of the symbol '0' and i the length of run of consecutive '0's. It has been shown that GCs are optimal for geometrically distributed sources.
15 This embodiment therefore allows the encoding of binary maps to approach the theoretical optimum.

Advantageously, GCs allow a very simple adaption to varying symbols statistics: a different code order m is employed depending on the probability, p .

20 Whilst they have been known since the 1970's, GCs appear in only a small number of applications: one known application is in the JPEG 2000 standard.

GCs are prefix-free codes and can therefore be treated in a similar fashion to VLCs, offering the same advantages as these codes. Moreover, the
25 approach offers high coding efficiency exceeding the performance of arithmetic coding because the distribution of the source data matches the distribution of the code very well.

Finally, code conforming to the present invention has great advantages in terms of implementation. As there is no need to store any tables, the codes are very easy to generate, the codes are simple to decode and the codes can be adapted to varying symbol statistics by just changing the order of the code (a
5 single number).

CLAIMS:

1. A method of encoding source data comprising the steps of:
decorrelating source data to generate transform coefficient
5 information;
slicing said transform coefficient information into a plurality of
bitplanes, said plurality of bitplanes including at least one least significant
bitplane;
context selecting from the bitplanes to generate a plurality of context
10 levels; and
for at least some of the plurality of context levels, applying run-
length pre-coding to generate sequences of pre-coded symbols and then
encoding said pre-coded symbol sequences by an analytical entropy coding
scheme.
15
2. A method according to Claim 1, wherein said analytical entropy
coding scheme is a Golomb coding scheme.
3. A method according to Claims 1 or 2, wherein the symbol sequences
20 are binary symbol sequences.
4. A method according to any one of Claims 1 to 3, wherein the
bitplane slicing is performed by the Zero-tree approach.
- 25 5. A method according to any one of Claims 1 to 4, wherein the context
selection stage includes the steps of:

measuring the frequency of a given symbol for each of the plurality of context levels;

generating a probability model; and

5 comparing the measured frequency of the given symbol for each of the context levels with the probability model.

6. A method according to any one of the preceding claims, wherein the given symbol is a zero.

10

7. A method according to any one of the preceding claims, wherein each selected context level is selected by establishing which of the available context levels best matches the probability model for which a Golomb Code of order m is optimal.

15

8. A data encoding apparatus comprising:

a redundancy reduction processor (302),

a relevant data reduction processor (304),

a context selection processor (308); and

20 an entropy encoding processor (310):

characterised in that the context selection processor (308) generates a plurality of context data output streams, each of said context data outputs having a corresponding context level; and

25 in that at least some of the plurality of context data output streams is input into said entropy encoding processor (310).

20

9. An apparatus according to Claim 8, wherein the entropy encoding processor comprises a run-length pre-coder (312) and an analytical entropy coding processor (314).

FIG 1

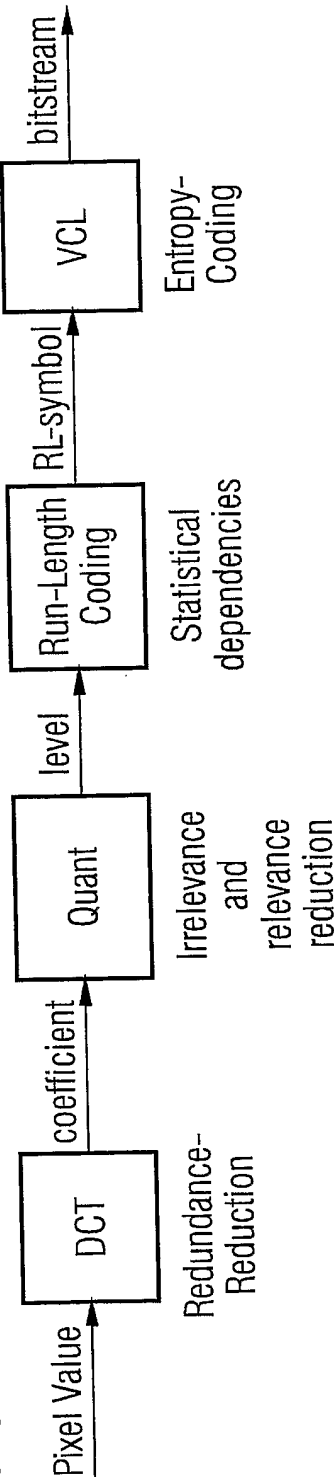


FIG 2A

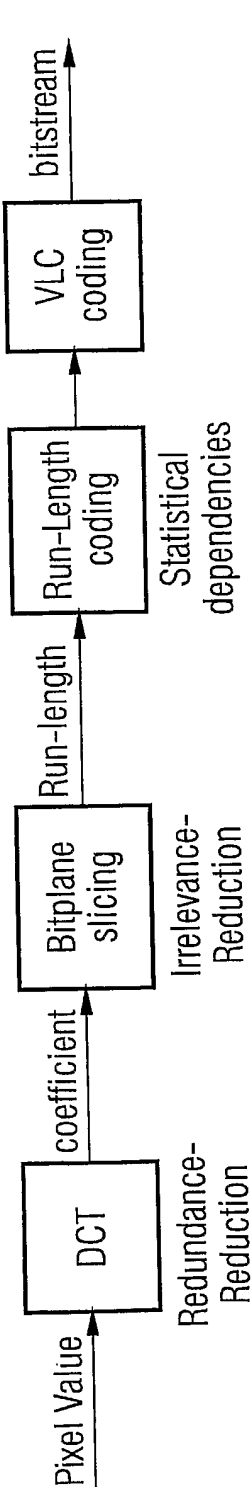
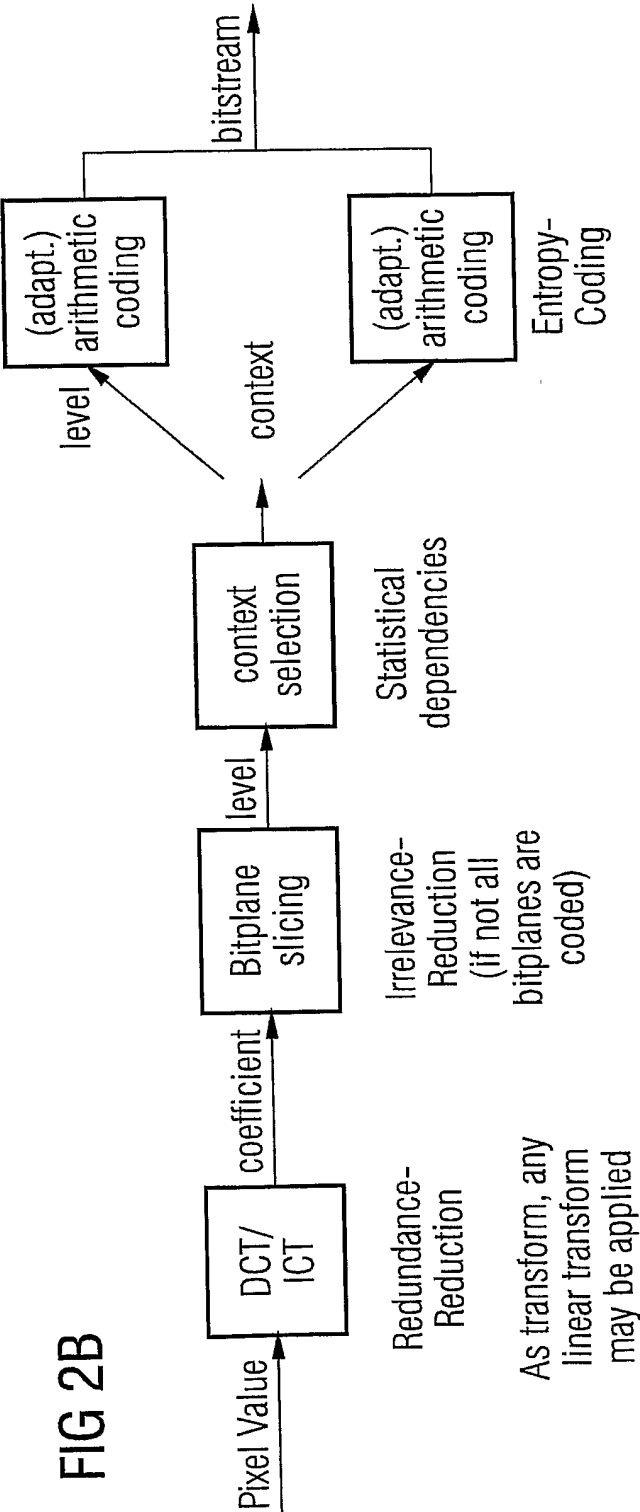


FIG 2B



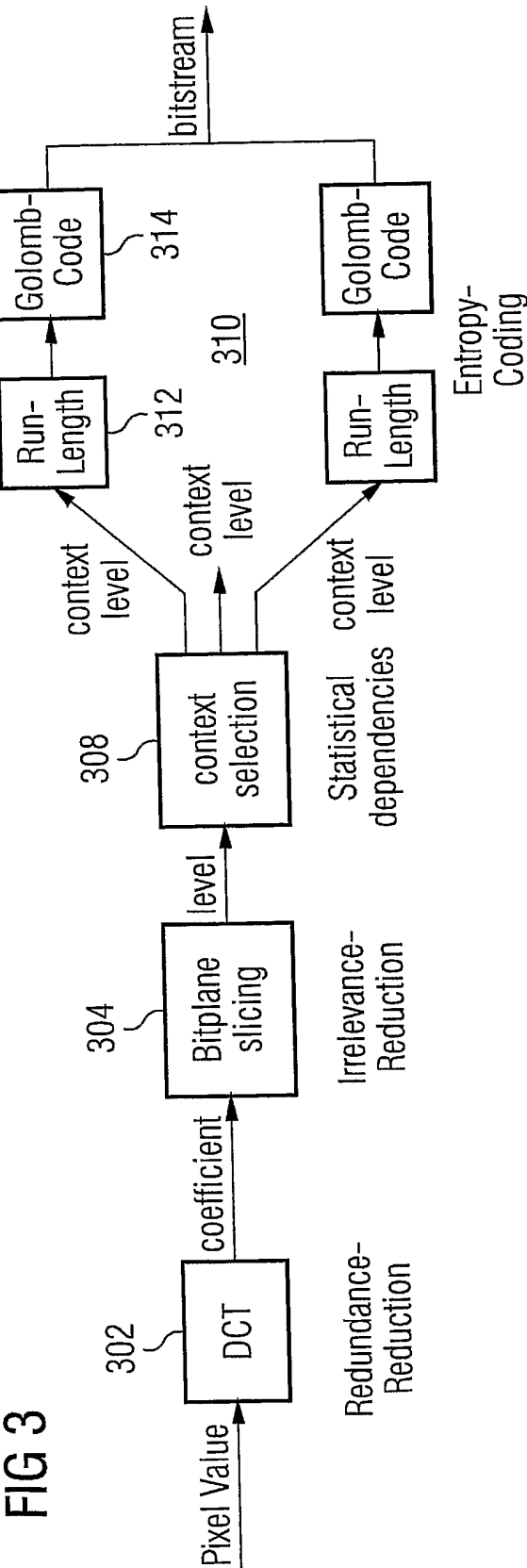


FIG 3