(19) DANMARK

(10) DK/EP 3121966 T3

(12) Oversættelse af europæisk patentskrift

Patent- og Varemærkestyrelsen

(51) Int.Cl.: *H 03 M 7/24 (2006.01)* *H 04 N 19/126 (2014.01)* *H 04 N 19/136 (2014.01)* *H 04 N 19/176 (2014.01)* *H 04 N 19/18 (2014.01)* *H 04 N 19/184 (2014.01)* *H 04 N 19/42 (2014.01)*

(45) Oversættelsen bekendtgjort den: **2018-03-12**

(80) Dato for Den Europæiske Patentmyndigheds bekendtgørelse om meddelelse af patentet: **2018-01-03**

(86) Europæisk ansøgning nr.: **16186353.5**

(86) Europæisk indleveringsdag: **2002-08-08**

(87) Den europæiske ansøgnings publiceringsdag: **2017-01-25**

(30) Prioritet: **2001-08-09 US 311436 P** **2001-11-30 US 319018 P** **2002-05-02 US 139036**

(62) Stamansøgningsnr: **06019033.7**

(84) Designerede stater: **AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LU MC NL PT SE SK TR**

(73) Patenthaver: **Dolby International AB, Apollo Building, 3E , Herikerbergweg 1-35, 1101 CN Amsterdam Zuidoost, Holland**

(72) Opfinder: **Kerofsky, Louis Jospeh, 1009 NW Goldendale, Camas, WA 98607, USA**

(74) Fuldmægtig i Danmark: **RWS Group, Europa House, Chiltern Park, Chiltern Hill, Chalfont St Peter, Bucks SL9 9FG, Storbritannien**

(54) Benævnelse: **FÆLLES KVANTISERINGS- OG HELTALSTRANSFORMATIONSNORMALISERING UNDER BENYTTELSE AF EN MANTISSE-EKSPONENTREPRÆSENTATION AF EN KVANTISERINGSPARAMETER**

(56) Fremdragne publikationer:
US-A- 5 230 038
US-A- 5 764 553
"H.26L TEST MODEL LONG TERM NUMBER 8 (TML-8) DRAFT0", ITU-T TELECOMMUNICATION STANDARIZATION SECTOR OF ITU, GENEVA, CH, 2 April 2001 (2001-04-02), pages 1-54, XP001089814,
LIANG J., TRAN T., TOPIWALA P.: "A 16-bit architecture for H.26L, treating DCT transforms and quantization", DOCUMENT VCEG-M16, ITU - TELECOMMUNICATIONS STANDARDIZATION SECTOR, STUDY GROUP 16 QUESTION 6, VIDEO CODING EXPERTS GROUP (VCEG) , 2 April 2001 (2001-04-02), pages 1-17, XP002332050, AUSTIN, TEXAS, USA Retrieved from the Internet: URL:http://www.ensc.sfu.ca/people/faculty/ jiel/papers/H26L_Proposal.doc [retrieved on 2005-06-08]
KEROFSKY L., LEI S.: "Reduced bit-depth quantization", DOCUMENT VCEG-N20, ITU - TELECOMMUNICATIONS STANDARDIZATION SECTOR, STUDY GROUP 16 QUESTION 6,VIDEO CODING EXPERTS GROUP (VCEG) , 24 September 2001 (2001-09-24), pages 1-14, XP002332051, SANTA BARBARA, CA, USA Retrieved from the Internet: URL:ftp3.itu.int/av-arch/video-site/0109_S an/VCEG-N20.doc [retrieved on 2005-06-08]
HALLAPURO: "Transform and Quantizer - part 1: Basics", 2. JVT MEETING; 29-01-2002 - 01-02-2002; GENEVA,

Fortsættes ...

CH; (JOINT VIDEO TEAMOF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16),, no. JVT-B038r2, 1 February 2002 (2002-02-01), XP030005037, ISSN: 0000-0443

DESCRIPTION

BACKGROUND OF THE INVENTION

1. Field of the Invention

[0001] This invention generally relates to video compression techniques and, more particularly, to a method for reducing the bit size required in the computation of video coding transformations.

2. Description of the Related Art

[0002] A video information format provides visual information suitable to activate a television screen, or store on a video tape. Generally, video data is organized in a hierarchical order. A video sequence is divided into group of frames, and each group can be composed of a series of single frames. Each frame is roughly equivalent to a still picture, with the still pictures being updated often enough to simulate a presentation of continuous motion. A frame is further divided into slices, or horizontal sections which helps system design of error resilience. Each slice is coded independently so that errors do not propagate across slices. A slice consists of macroblocks. In H.26P and Motion Picture Experts Group (MPEG)-X standards, a macroblock is made up of 16 x 16 luma pixels and a corresponding set of chroma pixels, depending on the video format. A macroblock always has an integer number of blocks, with the 8 x 8 pixel matrix being the smallest coding unit.

[0003] Video compression is a critical component for any application which requires transmission or storage of video data. Compression techniques compensate for motion by reusing stored information in different areas of the frame (temporal redundancy). Compression also occurs by transforming data in the spatial domain to the frequency domain. Hybrid digital video compression, exploiting temporal redundancy by motion compensation and spatial redundancy by transformation, such as Discrete Cosine Transform (DCT), has been adapted in H.26P and MPEG-X international standards as the basis.

[0004] As stated in US Patent 6,317,767 (Wang), DCT and inverse discrete cosine transform (IDCT) are widely used operations in the signal processing of image data. Both are used, for example, in the international standards for moving picture video compression put forth by the MPEG. DCT has certain properties that produce simplified and efficient coding models. When applied to a matrix of pixel data, the DCT is a method of decomposing a block of data into a weighted sum of spatial frequencies, or DCT coefficients. Conversely, the IDCT is used to transform a matrix of DCT coefficients back to pixel data.

[0005] Digital video (DV) codecs are one example of a device using a DCT-based data compression method. In the blocking stage, the image frame is divided into N by N blocks of pixel information including, for example, brightness and color data for each pixel. A common block size is eight pixels horizontally by eight pixels vertically. The pixel blocks are then "shuffled" so that several blocks from different portions of the image are grouped together. Shuffling enhances the uniformity of image quality.

[0006] Different fields are recorded at different time incidents. For each block of pixel data, a motion detector looks for the difference between two fields of a frame. The motion information is sent to the next processing stage. In the next stage, pixel information is transformed using a DCT. An 8-8 DCT, for example, takes eight inputs and returns eight outputs in both vertical and horizontal directions. The resulting DCT coefficients are then weighted by multiplying each block of DCT coefficients by weighting constants.

[0007] The weighted DCT coefficients are quantized in the next stage. Quantization rounds off each DCT coefficient within a certain range of values to be the same number. Quantizing tends to set the higher frequency components of the frequency matrix to zero, resulting in much less data to be stored. Since the human eye is most sensitive to lower frequencies, however, very little perceptible image quality is lost by this stage.

[0008] The quantization stage includes converting the two-dimensional matrix of quantized coefficients to a one-dimensional linear stream of data by reading the matrix values in a zigzag pattern and dividing the one-dimensional linear stream of quantized coefficients into segments, where each segment consists of a string of zero coefficients followed by a non-zero quantized coefficient. Variable length coding (VLC) then is performed by transforming each segment, consisting of the number of zero coefficients and the amplitude of the non-zero coefficient in the segment, into a variable length codeword. Finally, a framing process packs every 30 blocks of variable length coded quantized coefficients into five fixed-length synchronization blocks.

[0009] Decoding is essentially the reverse of the encoding process described above. The digital stream is first deframed. Variable length decoding (VLD) then unpacks the data so that it may be restored to the individual coefficients. After inverse quantizing the coefficients, inverse weighting and an inverse discrete cosine transform (IDCT) are applied to the result. The inverse weights are the multiplicative inverses of the weights that were applied in the encoding process. The output of the inverse weighting function is then processed by the IDCT.

[0010] Much work has been done studying means of reducing the complexity in the calculation of DCT and IDCT. Algorithms that compute two-dimensional IDCTs are called "type I" algorithms. Type I algorithms are easy to implement on a parallel machine, that is, a computer formed of a plurality of processors operating simultaneously in parallel. For example, when using N parallel processors to perform a matrix multiplication on N x N matrices, N column multiplies can be simultaneously performed. Additionally, a parallel machine can be designed so as to contain special hardware or software instructions for performing fast matrix transposition.

[0011] One disadvantage of type I algorithms is that more multiplications are needed. The computation sequence of type I algorithms involves two matrix multiplies separated by a matrix transposition which, if N=4, for example, requires 64 additions and 48 multiplications for a total number of 112 instructions. It is well known by those skilled in the art that multiplications are very time-consuming for processors to perform and that system performance is often optimized by reducing the number of multiplications performed.

[0012] A two-dimensional IDCT can also be obtained by converting the transpose of the input matrix into a one-dimensional vector using an L function. Next, the tensor product of constant a matrix is obtained. The tensor product is then multiplied by the one-dimensional vector L. The result is converted back into an N x N matrix using the M function. Assuming again that N=4, the total number of instructions used by this computational sequence is 92 instructions (68 additions and 24 multiplications). Algorithms that perform two-dimensional IDCTs using this computational sequence are called "type II" algorithms. In type II algorithms, the two constant matrices are grouped together and performed as one operation. The advantage of type II algorithms is that they typically require fewer instructions (92 versus 112) and, in particular, fewer costly multiplications (24 versus 48). Type II algorithms, however, are very difficult to implement efficiently on a parallel machine. Type II algorithms tend to reorder the data very frequently and reordering data on a parallel machine is very time-intensive.

[0013] There exist numerous type I and type II algorithms for implementing IDCTs, however, dequantization has been treated as an independent step depending upon DCT and IDCT calculations. Efforts to provide bit exact DCT and IDCT definitions have led to the development of efficient integer transforms. These integer transforms typically increase the dynamic range of the calculations. As a result, the implementation of these algorithms requires processing and storing data that consists of more than 16 bits.

[0014] It would be advantageous if intermediate stage quantized coefficients could be limited to a maximum size in transform processes.

[0015] It would be advantageous if a quantization process could be developed that was useful for 16-bit processors.

[0016] It would be advantageous if a decoder implementation, dequantization, and inverse transformation could be implemented efficiently with a 16-bit processor. Likewise, it would be advantageous if the multiplication could be performed with no more than 16 bits, and if memory access required no more than 16 bits.

SUMMARY OF THE INVENTION

[0017] The present invention is an improved process for video compression. Typical video coding algorithms predict one frame from previously coded frames. The error is subjected to a

transform and the resulting values are quantized. The quantizer controls the degree of compression. The quantizer controls the amount of information used to represent the video and the quality of the reconstruction.

[0018] The problem is the interaction of the transform and quantization in video coding. In the past the transform and quantizer have been designed independently. The transform, typically the discrete cosine transform, is normalized. The result of the transform is quantized in standard ways using scalar or vector quantization. In prior work, MPEG-1, MPEG-2, MPEG-4, H.261, H.263, the definition of the inverse transform has not been bit exact. This allows the implementer some freedom to select a transform algorithm suitable for their platform. A drawback of this approach is the potential for encoder/decoder mismatch damaging the prediction loop. To solve this mismatch problem portions of the image are periodically coded without prediction. Current work, for example H.26L, has focused on using integer transforms that allow bit exact definition. Integer transforms may not normalized. The transform is designed so that a final shift can be used to normalize the results of the calculation rather than intermediate divisions. Quantization also requires division. H.26L provides an example of how these integer transforms are used along with quantization.

[0019] In the current H.26L Test Model Long-term (TML), normalization is combined with quantization and implemented via integer multiplications and shifts following forward transform and quantization and following dequantization and inverse transform. See in particular document "H.26L TEST MODEL LONG TERM NUMBER 8 (TML-8) DRAFTO",ITU-T TELECOMMUNICATION STANDARIZATION SECTOR OF ITU, GENEVA, CH, 2 April 2001. H.26L TML uses two arrays of integers A(QP) and B(QP) indexed by quantization parameter (QP), see Table 1. These values are constrained by the relation shown below in Equation 1.

Table 1 TML quantization parameters

QP	$A_{TML}(QP)$	$B_{TML}(QP)$
0	620	3881
1	553	4351
2	492	4890
3	439	5481
4	391	6154
5	348	6914
6	310	7761
7	276	8718
8	246	9781
9	219	10987
10	195	12339
11	174	13828
12	155	15523

QP	$A_{TML}(QP)$	$B_{TML}(QP)$
13	138	17435
14	123	19561
15	110	21873
16	98	24552
17	87	27656
18	78	30847
19	69	34870
20	62	38807
21	55	43747
22	49	49103
23	44	54683
24	39	61694
25	35	68745
26	31	77615
27	27	89113
28	24	100253
29	22	109366
30	19	126635
31	17	141533

Equation 1 Joint Normalization/Quantization relation

$$A(QP) \cdot B(QP) \cdot 676^2 \approx 2^{40}$$

[0020] Normalization and quantization are performed simultaneously using these integers and divisions by powers of 2.

[0021] Transform coding in H.26L uses a 4x4 block size and an integer transform matrix T, Equation 2. For a 4x4 block X, the transform coefficients K are calculated as in Equation 3. From the transform coefficients, the quantization levels, L, are calculated by integer multiplication. At the decoder the levels are used to calculate a new set of coefficients, K'. Additional integer matrix transforms followed by a shift are used to calculate the reconstructed values X'. The encoder is allowed freedom in calculation and rounding of the forward transform. Both encoder and decoder must compute exactly the same answer for the inverse calculations.

Equation 2 H.26L test model 8 transform matrix

$$T = \begin{pmatrix} 13 & 13 & 13 & 13 \\ 17 & 7 & -7 & -17 \\ 13 & -13 & -13 & 13 \\ 7 & -17 & 17 & -7 \end{pmatrix}$$

Equation 3 TML DCT_LUMA and IDCT_LUMA

$$Y = T \cdot X$$
$$K = Y \cdot T^T$$
$$L = (A_{TML}(QP) \cdot K)/2^{20}$$
$$K' = B_{TML}(QP) \cdot L$$
$$Y' = T^T \cdot K'$$
$$X' = (Y' \cdot T)/2^{20}$$

[0022] Where the intermediate result Y is the result of a one dimensional transform and the intermediate result Y' is the result of a one dimensional inverse transform.

[0023] The dynamic range required during these calculations can be determined. The primary application involves 9-bit input, 8 bits plus sign, the dynamic range required by intermediate registers and memory accesses is presented in Table 2.

Table 2 Dynamic range of TML transform and inverse transform (bits)

9-bit input	LUMA Transform	Inverse Transform
Register	30	27
Memory	21	26

[0024] To maintain bit-exact definitions and incorporate quantization, the dynamic range of intermediate results can be large since division operations are postponed. The present invention combines quantization and normalization, to eliminate the growth of dynamic range of intermediate results. With the present invention the advantages of bit exact inverse transform and quantization definitions are kept, while controlling the bit depth required for these calculations. Reducing the required bit depth reduces the complexity required of a hardware implementation and enables efficient use of single instruction multiple data (SIMD) operations, such as the Intel MMX instruction set.

[0025] Accordingly, a method is provided for the quantization of a coefficient. The method comprises: supplying a coefficient K; supplying a quantization parameter (QP); forming a quantization value (L) from the coefficient K using a mantissa portion (Am(QP)) and an exponential portion ($x^{Ae(QP)}$). Typically, the value of x is 2.

[0026] In some aspects of the method, forming a quantization value (L) from the coefficient K includes:

$$L = K \cdot A(QP) = K \cdot Am(QP) \cdot (2^{Ae(QP)})$$

$L = K \cdot A(QP) = K \cdot Am(QP) \cdot (2^{\ldots})$

[0027] In other aspects, the method further comprises: normalizing the quantization value by 2^N as follows:

$$Ln = L/2^N = K \cdot Am(QP)/2^{(N-Ae(QP)}$$

[0028] In some aspects, forming a quantization value includes forming a set of recursive quantization factors with a period P, where A(QP+P) = A(QP)/x. Therefore, forming a set of recursive quantization factors includes forming recursive mantissa factors, where Am(QP) = Am(QP mod P). Likewise, forming a set of recursive quantization factors includes forming recursive exponential factors, where Ae(QP) = Ae(QP mod P) - QP/P.

[0029] More specifically, supplying a coefficient K includes supplying a coefficient matrix K[i][j]. Then, forming a quantization value (L) from the coefficient matrix K[i][j] includes forming a quantization value matrix (L[i][j] using a mantissa portion matrix (Am(QP)[i][j]) and an exponential portion matrix ($x^{Ae(QP)[i][j]}$),

[0030] Likewise, forming a quantization value matrix (L[i][j]) using a mantissa portion matrix (Am(QP)[i][j]) and an exponential portion matrix ($x^{Ae(QP)[i][j]}$) includes, for each particular value of QP, every element in the exponential portion matrix being the same value. Every element in the exponential portion matrix is the same value for a period (P) of QP values, where Ae(QP) = Ae(P * (QP/P)).

[0031] Additional details of the above-described method, including a method for forming a dequantization value (X1), from the quantization value, using a mantissa portion (Bm(QP)) and an exponential portion ($x^{Be(QP)}$) are provided below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0032]

Fig. 1 is a flowchart illustrating the present invention method for the quantization of a coefficient.

Figs. 2 to 9 show embodiments or aspects of the present invention comprise systems and methods for video coding.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0033] The dynamic range requirements of the combined transform and quantization is reduced by factoring the quantization parameters A(QP) and B(QP) into a mantissa and exponent terms as shown in Equation 4. With this structure, only the precision due to the mantissa term needs to be preserved during calculation. The exponent term can be included in the final normalization shift. This is illustrated in the sample calculation Equation 5.

Equation 4 Structure of quantization parameters

$$A_{proposal}(QP) = A_{mantissa}(QP) \cdot 2^{A_{exponent}(QP)}$$

$$B_{proposal}(QP) = B_{mantissa}(QP) \cdot 2^{B_{exponent}(QP)}$$

Equation 5 reduced bit_depth LUMA Transform

$$Y = T \cdot X$$

$$K = Y \cdot T^T$$

$$L = (A_{mantissa}(QP) \cdot K)/2^{20-A_{exponent}(QP)}$$

$$K' = T^T \cdot L$$

$$Y' = T^T \cdot K'$$

$$X' = (B_{mantissa}(QP) \cdot Y')/2^{20-B_{exponent}(QP)}$$

[0034] To illustrate the present invention, a set of quantization parameters is presented that reduce the dynamic range requirement of an H.26L decoder to 16-bit memory access. The memory access of the inverse transform is reduced to 16 bits. Values for $A_{manissa}$, $A_{exponent}$, $B_{manissa}$, $B_{exponent}$, $A_{proposed}$, $B_{proposed}$ are defined for QP=0-5 as shown in Table 3. Additional values are determined by recursion, as shown in Equation 6. The structure of these values makes it possible to generate new quantization values in addition to those specified.

Table 3 Quantization values 0-5 for TML

QP	$A_{mantissa}$	$A_{exponent}$	$B_{mantissa}$	$B_{exponent}$	$A_{proposed}$	$B_{proposed}$
0	5	7	235	4	640	3760
1	9	6	261	4	576	4176
2	127	2	37	7	508	4736
3	114	2	165	5	456	5280
4	25	4	47	7	400	6016
5	87	2	27	8	348	6912

Equation 6 Recursion relations

$$A_{mantissa}(QP + 6) = A_{mantissa}(QP)$$

$$B_{mantissa}(QP + 6) = B_{mantissa}(QP)$$

$$A_{exponent}(QP + 6) = A_{exponent}(QP) - 1$$

$$B_{exponent}(QP + 6) = B_{exponent}(QP) + 1$$

[0035] Using the defined parameters, the transform calculations can be modified to reduce the dynamic range as shown in Equation 5. Note how only the mantissa values contribute to the growth of dynamic range. The exponent factors are incorporated into the final normalization and do not impact the dynamic range of intermediate results.

[0036] With these values and computational method, the dynamic range at the decoder is reduced so only 16-bit memory access is needed as seen in Table 4.

Table 4 Dynamic range with low-bit depth quantization (QP>6) **8-bit LUMA Transform LUMA Inverse Transform**

8-bit	LUMA Transform	Inverse Transform
Register	28	24
Memory	21	16

[0037] Several refinements can be applied to the joint quantization/normalization procedure described above. The general technique of factoring the parameters into a mantissa and exponent forms the basis of these refinements.

[0038] The discussion above assumes all basis functions of the transform have an equal norm and are quantized identically. Some integer transforms have the property that different basis functions have different norms. The present invention technique has been generalized to support transforms having different norms by replacing the scalars A(QP) and B(QP) above by matrices A(QP)[i](j] and B(QP)[i][j]. These parameters are linked by a normalization relation of the form shown below, Equation 7, which is more general than the single relation shown in Equation 1.

Equation 7 Joint quantization/normalization of matrices

$$A(QP)[i][j] \cdot B(QP)[i][j] = N[i][j]$$

[0039] Following the method previously described, each element of each matrix is factored into a mantissa and an exponent term as illustrated in the equations below, Equation 8.

Equation 8 Factorization of matrix parameters

$$A(QP)[i][j] = A_{mantissa}(QP)[i][j] \cdot 2^{A_{exponent}(QP)[i][j]}$$
$$B(QP)[i][j] = B_{mantissa}(QP)[i][j] \cdot 2^{B_{exponent}(QP)[i][j]}$$

[0040] A large number of parameters are required to describe these quantization and dequantization parameters. Several structural relations can be used to reduce the number of free parameters. The quantizer growth is designed so that the values of A are halved after each period P at the same time the values of B are doubled maintaining the normalization

relation. Additionally, the values of $A_{exponent}(QP)[i][j]$ and $B_{exponent}(QP)[i][j]$ are independent of i, j and (QP) in the range [0,P-1]. This structure is summarized by structural equations, Equation 9. With this structure there are only two parameters $A_{exponent}[0]$ and $B_{exponent}[0]$.

Equation 9 Structure of exponent terms

$$A_{exponent}(QP)[i][j] = A_{exponent}[0] - QP/P$$
$$B_{exponent}(QP)[i][j] = B_{exponent}[0] + QP/P$$

[0041] A structure is also defined for the mantissa values. For each index pair (i,j), the mantissa values are periodic with period P. This is summarized by the structural equation, Equation 10. With this structure, there are P independent matrices for $A_{mantissa}$ and P independent matrices for $B_{mantissa}$ reducing memory requirements and adding structure to the calculations.

Equation 10 Structure of mantissa terms

$$A_{mantissa}(QP)[i][j] = A_{mantissa}(QP\%P)[i][j]$$
$$B_{mantissa}(QP)[i][j] = B_{mantissa}(QP\%P)[i][j]$$

[0042] The inverse transform may include integer division that requires rounding. In cases of interest the division is by a power of 2. The rounding error is reduced by designing the dequantization factors to be multiples of the same power of 2, giving no remainder following division. Dequantization using the mantissa values $B_{mantissa}(QP)$ gives dequantized values that are normalized differently depending upon QP. This must be compensated for following the inverse transform. A form of this calculation is shown in Equation 11.

Equation 11 Normalization of inverse transform I

$$K[i][j] = B_{mantissa}(QP\%P)[i][j] \cdot Level[i][j]$$
$$X = (T^{-1} \cdot K \cdot T)/2^{(N-QP/P)}$$

[0043] In Equation 11, Level[i][j] is the quantized version of the transform coefficients and is called as "quantization value". K[i][j] is the scaled version of the transform coefficients and is called as "dequantization value".

[0044] To eliminate the need for the inverse transform to compensate for this normalization difference, the dequantization operation is defined so that all dequantized values have the same normalization. The form of this calculation is shown in Equation 12.

Equation 12 Normalization of Inverse transform II

$$K[i][j] = B_{mantissa}(QP\%P)[i][j] \cdot 2^{QP/P} \cdot Level[i][j]$$

$$X = (T^{-1} \cdot K \cdot T)/2^{N}$$

[0045] The power of 2 can be calculated by using left-shift operation and the dequantization value K[i][j] in Equation 12 will then be given as follows.

$$K[i][j] = [B_{mantissa} \cdot Level[i][j]] \ll (QP/P)$$

[0046] An example follows that illustrates the present invention use of quantization matrices. The forward and inverse transforms defined in Equation 13 need a quantization matrix rather than a single scalar quantization value. Sample quantization and dequantization parameters are given. Equation 14 and 16, together with related calculations, illustrate the use of this invention. This example uses a period P=6. In Equation 14, $A_{mantissa}$ is represented by Q and QP is represented by m. In Equation 16, $B_{mantissa}$ is represented by R and QP is represented by m.

Equation 13 transforms

$$T_{forward} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix}$$

$$T_{reverse} = \begin{pmatrix} 2 & 2 & 2 & 1 \\ 2 & 1 & -2 & -2 \\ 2 & -2 & -2 & 2 \\ 2 & -1 & 2 & -1 \end{pmatrix}$$

$$)[i][j] = M_{m,2}\ otherwise$$

$$M = \begin{bmatrix} 21844 & 8388 & 13108 \\ 18724 & 7625 & 11650 \\ 16384 & 6989 & 10486 \\ 14564 & 5992 & 9532 \end{bmatrix}$$

ıtion 16 Dequantization parameters

$$'(m)[i][j] = S_{m,0}\ for\ (i,j) = \{(0,0),(0,2),(2,0),(2,2)\}$$

$$'(m)[i][j] = S_{m,1}\ for\ (i,j) = \{(1,1),(1,3),(3,1),(3,3)\}$$

$$'(m)[i][j] = S_{m,2}\ otherwise$$

$$= \begin{bmatrix} 6 & 10 & 8 \\ 7 & 11 & 9 \\ 8 & 12 & 10 \\ 9 & 14 & 11 \\ 10 & 16 & 13 \end{bmatrix}$$

[0047] The description of the forward transformation and forward quantization, Equation 18, are given below assuming input is in X, quantization parameter QP.

Equation 17 forward transform

$$K = T_{forward} \cdot X \cdot T_{forward}^{T}$$

Equation 18 forward quantization

$$period = QP/6$$
$$phase = QP - 6 \cdot period$$
$$Level[i][j] = (Q(phase)[i][j] \cdot K[i][j])/2^{(17+period)}$$

[0048] The description of dequantization, inverse transform, and normalization for this example is given below, Equation 19 and 20.

Equation 19 Dequantization

$$period = QP/6$$
$$phase = QP - 6 \cdot period$$
$$K[i][j] = R(phase)[i][j] \cdot Level[i][j] \cdot 2^{period}$$

Equation 20 IDCT and normalization

$$X' = T_{reverse} \cdot K \cdot T_{reverse}^{T}$$
$$X^{\cdot} = X'[i][j]/2^{T}$$

[0049] Fig. 1 is a flowchart illustrating the present invention method for the quantization of a coefficient. Although this method is depicted as a sequence of numbered steps for clarity, no order should be inferred from the numbering unless explicitly stated. It should be understood that some of these steps may be skipped, performed in parallel, or performed without the requirement of maintaining a strict order of sequence. The methods start at Step 100. Step 102 supplies a coefficient K. Step 104 supplies a quantization parameter (QP). Step 106 forms a quantization value (L) from the coefficient K using a mantissa portion (Am(QP)) and an exponential portion ($x^{Ae(QP)}$). Typically, the exponential portion ($x^{Ae(QP)}$) includes x being the value 2.

[0050] In some aspects of the method, forming a quantization value (L) from the coefficient K using a mantissa portion (Am(QP)) and an exponential portion ($x^{Ae(QP)}$) in Step 106 includes:
$L = K \cdot A(QP) = K \cdot Am(QP) \cdot (2^{Ae(QP)})$.

[0051] Some aspects of the method include a further step. Step 108 normalizes the quantization value by 2^N as follows:
$Ln = L/2^N = K \cdot Am(QP)/2^{(N-Ae(QP)}$.

[0052] In other aspects, forming a quantization value in Step 106 includes forming a set of recursive quantization factors with a period P, where A(QP+P) = A(QP)/x. Likewise, forming a set of recursive quantization factors includes forming recursive mantissa factors, where Am(QP) = Am(QP mod P). Then, forming a set of recursive quantization factors includes forming recursive exponential factors, where Ae(QP) = Ae(QP mod P) - QP/P.

[0053] In some aspects, forming a quantization value includes forming a set of recursive quantization factors with a period P, where A(QP+P) = A(QP)/2. In other aspects, forming a set of recursive quantization factors includes forming recursive mantissa factors, where P = 6. Likewise, forming a set of recursive quantization factors includes forming recursive exponential factors, where P = 6.

[0054] In some aspects of the method, supplying a coefficient K in Step 102 includes supplying a coefficient matrix K[i][j]. Then, forming a quantization value (L) from the coefficient matrix K[i][j] using a mantissa portion (Am(QP) and an exponential portion ($x^{Ae(Qp)}$) in Step 106 includes forming a quantization value matrix (L[i][j]) using a mantissa portion matrix (Am(QP)[i][j]) and an exponential portion matrix ($x^{Ae(QP)[i][j]}$). Likewise, forming a quantization value matrix (L[i][j]) using a mantissa portion matrix (Am(QP)[i][j]) and an exponential portion matrix ($x^{Ae(QP)[i][j]}$) includes, for each particular value of QP, every element in the exponential portion matrix being the same value. Typically, every element in the exponential portion matrix is the same value for a period (P) of QP values, where Ae(QP) = Ae(P * (QP/P)).

[0055] Some aspects of the method include a further step. Step 110 forms a dequantization value (X1) from the quantization value, using a mantissa portion (Bm(QP)) and an exponential portion ($x^{Be(QP)}$). Again, the exponential portion ($x^{Be(QP)}$) typically includes x being the value 2.

[0056] In some aspects of the method, forming a dequantization value (X1) from the quantization value, using a mantissa portion (Bm(QP)) and an exponential portion ($2^{Be(QP)}$) includes:
$X1 = L \cdot B(QP) = L \cdot Bm(QP) \cdot (2^{Be(QP)})$.

[0057] Other aspects of the method include a further step, Step 112, of denormalizing the quantization value by 2^N as follows:

$$X1d = X1/2^N = X1 \cdot Bm(QP)/2^N.$$

[0058] In some aspects, forming a dequantization value in Step 110 includes forming a set of recursive dequantization factors with a period P, where B(QP+P) = x*B(QP). Then, forming a set of recursive dequantization factors includes forming recursive mantissa factors, where Bm(QP) = Bm(QP mod P). Further, forming a set of recursive dequantization factors includes forming recursive exponential factors, where Be(QP) = Be(QP mod P) + QP/P.

[0059] In some aspects, forming a set of recursive quantization factors with a period P includes the value of x being equal to 2, and forming recursive mantissa factors includes the value of P being equal to 6. Then, forming a set of recursive dequantization factors includes forming recursive exponential factors, where Be(QP) = Be(QP mod P) + QP/P.

[0060] In some aspects of the method, forming a dequantization value (X1), from the quantization value, using a mantissa portion (Bm(QP)) and an exponential portion ($x^{Be(QP)}$) in Step 110 includes forming a dequantization value matrix (X1[i][j]) using a mantissa portion matrix (Bm(QP)[i][j]) and an exponential portion matrix ($x^{Be(QP)[i][j]}$). Likewise, forming a dequantization value matrix (X1[i][j] using a mantissa portion matrix (Bm(QP)[i][j]) and an exponential portion matrix ($x^{Be(QP)[i][j]}$) includes, for each particular value of QP, every element in the exponential portion matrix being the same value. In some aspects, every element in the exponential portion matrix is the same value for a period (P) of QP values, where Be(QP) = Be(P * (QP/P)).

[0061] Another aspect of the invention includes a method for the dequantization of a coefficient. However, the process is essentially the same as Steps 110 and 112 above, and is not repeated in the interest of brevity.

[0062] A method for the quantization of a coefficient has been presented. An example is given illustrating a combined dequantization and normalization procedure applied to the H.26L video coding standard with a goal of reducing the bit-depth required at the decoder to 16 bits. The present invention concepts can also be used to meet other design goals within H.26L. In general, this invention has application to the combination of normalization and quantization calculations.

[0063] Embodiments of the present invention may be implemented as hardware, firmware, software and other implementations. Some embodiments may be implemented on general purpose computing devices or on computing devices specifically designed for implementation of these embodiments. Some embodiments may be stored in memory as a means of storing the embodiment or for the purpose of executing the embodiment on a computing device.

[0064] Some aspects of the present invention comprise systems and methods for video encoding, as shown in Figure 2. In these aspects, image data 130 is subtracted from 132 with data representing prior video frames 145 resulting in a differential image 133, which is sent to a transform module 134. Transform module 134 may use DCT or other transform methods to transform the image. Generally, the result of the transform process will be coefficients (K), which are then sent to a quantization module 136 for quantization. Quantization module 136 may have other inputs, such as user inputs 131 for establishing quantization parameters (QPs) and for other input. Quantization module 136 may use the transformation coefficients and the quantization parameters to determine quantization levels (L) in the video image. Quantization module 136 may use methods employing a mantissa portion and an exponential portion, however, other quantization methods may also be employed in the quantization modules 136 of embodiments of the present invention. These quantization levels 135 and quantization parameters 137 are output to a coding module 138 as well as a dequantization module (DQ) 140.

[0065] Output to the coding module 138 is encoded and transmitted outside the encoder for immediate decoding or storage. Coding module 138 may use variable length coding (VLC) in its coding processes. Coding module 138 may use arithmetic coding in its coding process. Output from coding module 138 is encoded data 139 which may be transmitted to the decoder or stored in the storage device.

[0066] Output from quantization module 136 is also received at dequantization module 140 to begin reconstruction of the image. This is done to keep an accurate accounting of prior frames. Dequantization module 140 performs a process with essentially the reverse effect as quantization module 136. Quantization levels or values (L) are dequantized yielding transform coefficients. Dequantization modules 140 may use methods employing a mantissa portion and an exponential portion as described herein.

[0067] The transform coefficients output from dequantization module 140 are sent to an inverse transformation (IT) module 142 where they are inverse transformed to a differential image 141. This differential image 141 is then combined with data from prior image frames 145 to form a video frame 149 that may be input to a frame memory 146 for reference to succeeding frames.

[0068] Video frame 149 may also serve as input to a motion estimation module 147, which also receives image data 130. These inputs may be used to predict image similarities and help compress image data. Output from motion estimation module 147 is sent to motion compensation module 148 and combined with output data from coding module 138, which is sent out for later decoding and eventual image viewing.

[0069] Motion compensation module 148 uses the predicted image data to reduce frame data requirements; its output is subtracted from input image data 130.

[0070] Some embodiments of the present invention comprise systems and methods for video

decoding, as shown in Figure 3. A decoder of embodiments of the present invention may receive encoded data 150 to a decoder module 152. Encoded data 150 may comprise data that has been encoded by an encoder 100 such as that described with reference to Figure 2. Decoding module 152 may employ variable length decoding methods if they were used in the encoding process. Other decoding methods may also be used as dictated by the type of encoded data 150. Decoding module 152 performs essentially the reverse process as coding module 138. Output from decoding module 152 may comprise quantization parameters 156 and quantization values 154. Other output may comprise motion estimation data and image prediction data that may be sent directly to a motion compensation module 166.

[0071] Typically, quantization parameters 156 and quantization values 154 are output to a dequantization module 158, where quantization values are converted back to transform coefficients. Dequantization module 158 may use methods employing a mantissa portion and an exponential portion as described herein. These coefficients are then sent to an inverse transformation module 160 for conversion back to spatial domain image data 161.

[0072] The motion compensation unit 166 uses motion vector data and the frame memory 164 to construct a reference image 165.

[0073] Image data 161 represents a differential image that must be combined with prior image data 165 to form a video frame 163. This video frame 163 is output 168 for further processing, display or other purposes and may be stored in frame memory 164 and used for reference with subsequent frames.

[0074] In some aspects of the present invention, as illustrated in Figure 4, image data 102 may be sent to an encoder or encoding portion 104 for the various transformation, quantization, encoding and other procedures typical of video encoding as described above for some aspects of the present invention. Output from the encoder may then be stored on any computer-readable storage media 106. Storage media 106 may act as a short-term buffer or as a long-term storage device.

[0075] When desired, encoded video data may be read from storage media 106 and decoded by a decoder or decoding portion 108 for output 110 to a display or other device. In some aspects of the present invention, as illustrated in Figure 5, image data 112 may be sent to an encoder or encoding portion 114 for the various transformation, quantization, encoding and other procedures typical of video encoding as described above for some aspects of the present invention. Output from the encoder may then be sent over a network, such as a LAN, WAN or the Internet 116. A storage device such as storage media 106 may be part of a network. Encoded video data may be received and decoded by a decoder or decoding portion 118 which also communicates with network 116. Decoder 118 may then decode the data for local consumption 120.

[0076] In some aspects of the present invention, as illustrated in Figure 6, a quantization method or apparatus comprises a mantissa portion 172 and an exponential portion 174.

Quantization parameters 176 are input to both portions 172 & 174. A coefficient K 170 is input to the mantissa portion 172 where it is modified using the quantization parameter and other values as explained above. The result of this operation is combined with the result produced in the exponential portion using the quantization parameter thereby producing a quantization level or value L 178.

[0077] In some aspects of the present invention, as illustrated in Figure 7, a quantization method or apparatus comprises a mantissa portion 182 and a shifting portion 184. Quantization parameters 186 are input to both portions 182 & 184. A coefficient, K 180 is input to the mantissa portion 182 where it is modified using the quantization parameter and other values as explained above. The result of this operation is further processed in the shifting portion using the quantization parameter thereby producing a quantization level or value, L 188.

[0078] Some embodiments of the present invention, as illustrated in Figure 8, comprise a dequantization method or apparatus with a mantissa portion 192 and an exponential portion 194. Quantization parameters 196 are input to both portions 192 & 194. A quantization value, L 190 is input to the mantissa portion 192 where it is modified using the quantization parameter and other values as explained above. The result of this operation is further processed in the exponential portion using the quantization parameter thereby producing a coefficient, X1 198.

[0079] Some embodiments of the present invention, as illustrated in Figure 9, comprise a dequantization method or apparatus with a mantissa portion 202 and a shifting portion 204. Quantization parameters 206 are input to both portions 202 & 204. A quantization value, L 200 is input to the mantissa portion 202 where it is modified using the quantization parameter and other values as explained above. The result of this operation is further processed in the exponential portion using the quantization parameter thereby producing a coefficient, X1 208.

[0080] Some embodiments of the present invention may be stored on computer-readable media such as magnetic media, optical media, and other media as well as combinations of media. Some embodiments may also be transmitted as signals across networks and operation of embodiments of the present invention or as a way of transmitting the embodiment to a destination.

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US6317767B [0004]

Non-patent literature cited in the description

- H.26L TEST MODEL LONG TERM NUMBER 8 (TML-8) DRAFTOITU-T TELECOMMUNICATION STANDARIZATION SECTOR OF ITU, 2001, [0019]

Patentkrav

1. Videoafkoder til rekonstruktion af en prøve X' fra en kvantiseret værdi L, idet apparatet er konfigureret til:
at modtage den kvantiserede værdi L (190);
at udføre en invers heltalstransformation af værdien L for at udlede en værdi Y', hvori
alle basisfunktioner fra den inverse heltalstransformation har en ens norm, og
at udføre kombineret afkvantisering og normalisering af den inverst transformerede værdi Y' af den kvantiserede værdi L for at udlede en afkvantiseret og normaliseret værdi X', der repræsenterer den rekonstruerede prøve, hvori den kombinerede afkvantisering og normalisering indbefatter:
at indføre en kvantiseringsparameter QP (196); og
at udlede den afkvantiserede og normaliserede værdi X', der repræsenterer den rekonstruerede prøve;
kendetegnet ved, at:
den afkvantiserede og normaliserede værdi X', der repræsenterer den rekonstruerede prøve, bliver udledt ved at benytte en mantissedel Bm(QP) (192), der er en funktion af kvantiseringsparameteren QP, og en eksponentdel Be(QP) (194), der er en funktion af kvantiseringsparameteren QP og repræsenterer et normaliseringsskifte som $X' = Y' * Bm(QP) * 2^{Be(QP)} * 2^{-N}$,
hvori N er et heltal, multiplikationen med 2^{-N} repræsenterer et endeligt normaliseringsskifte, og mantissedelen Bm(QP) (192) og eksponentdelen Be(QP) (194) opfylder de rekursive relationer
$Bm(QP + 6) = Bm(QP)$, og
$Be(QP + 6) = Be(QP) + 1$.

DRAWINGS

FIG.1

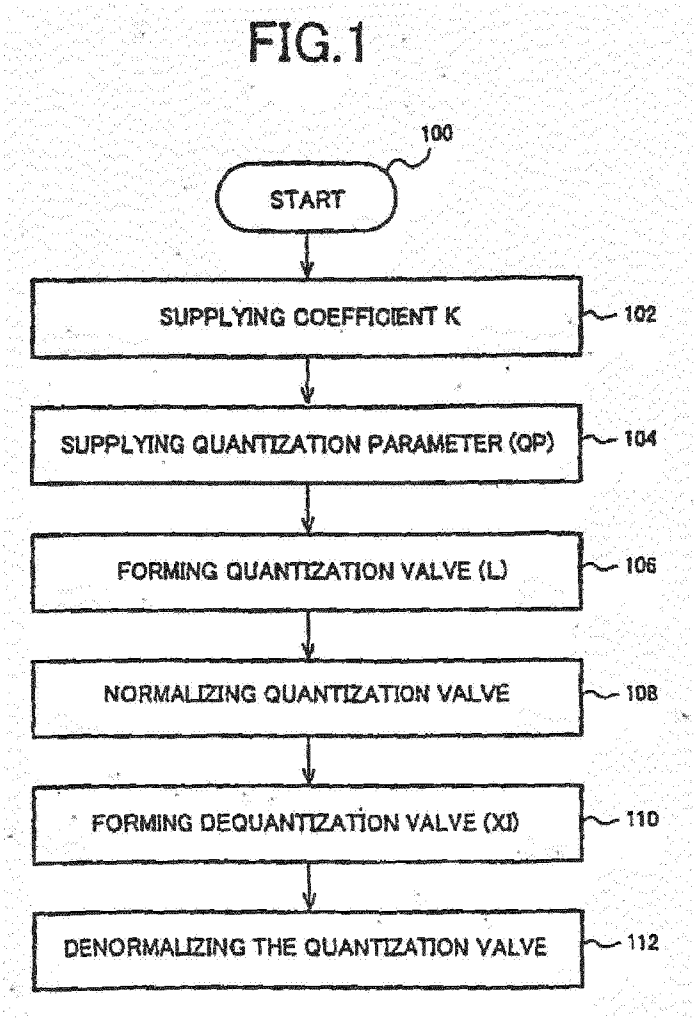


FIG.2

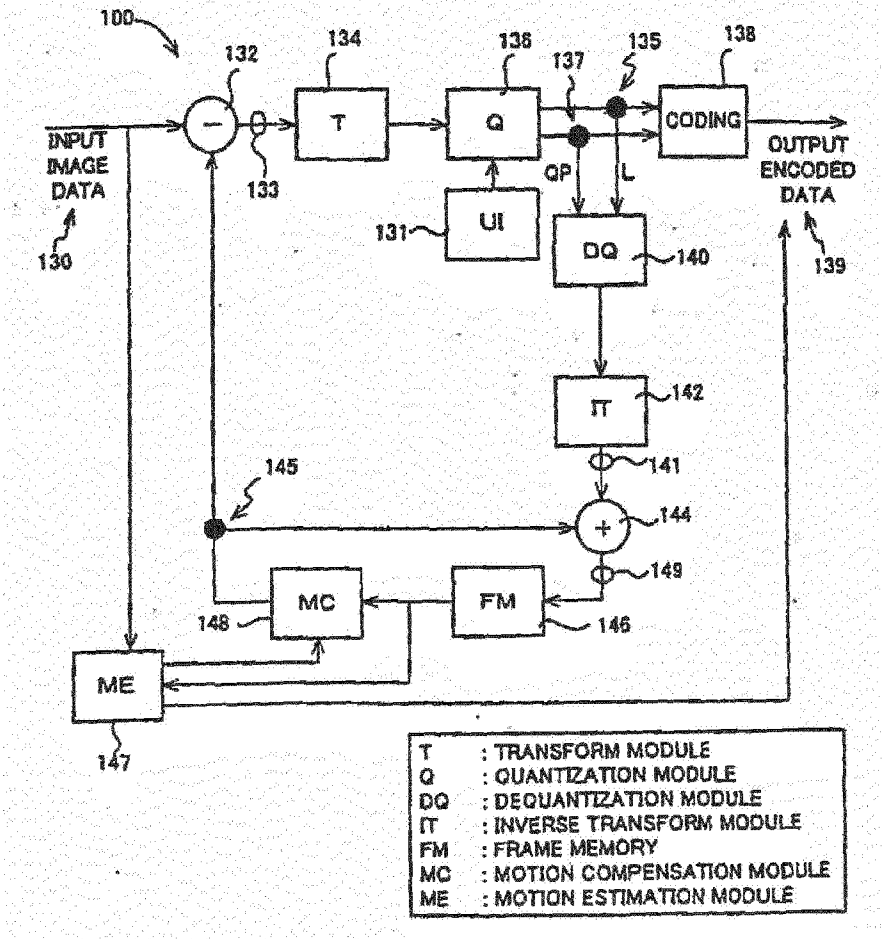

100
132
134
136
137
135
138
INPUT IMAGE DATA
130
133
T
Q
CODING
OUTPUT ENCODED DATA
139
QP
L
131
UI
DQ
140
IT
142
141
145
144
149
MC
FM
148
146
ME
147
T : TRANSFORM MODULE
Q : QUANTIZATION MODULE
DQ : DEQUANTIZATION MODULE
IT : INVERSE TRANSFORM MODULE
FM : FRAME MEMORY
MC : MOTION COMPENSATION MODULE
ME : MOTION ESTIMATION MODULE

FIG.3

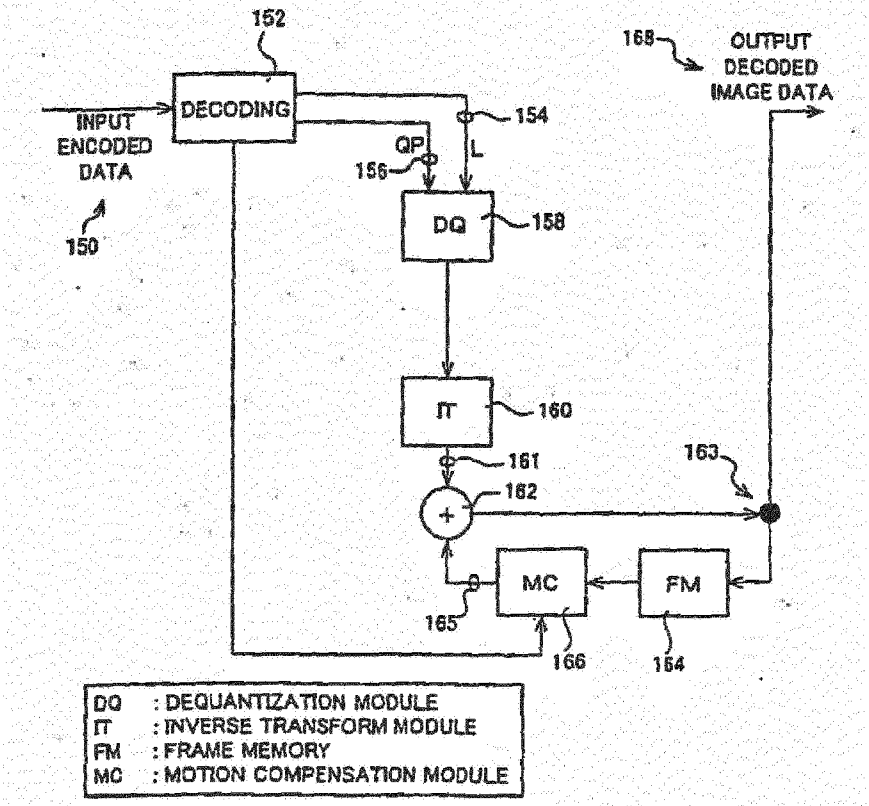

152
DECODING
INPUT ENCODED DATA
150
154
L
QP
156
DQ
158
IT
160
161
162
163
168
OUTPUT DECODED IMAGE DATA
MC
FM
165
166
164
DQ : DEQUANTIZATION MODULE
IT : INVERSE TRANSFORM MODULE
FM : FRAME MEMORY
MC : MOTION COMPENSATION MODULE

FIG.4

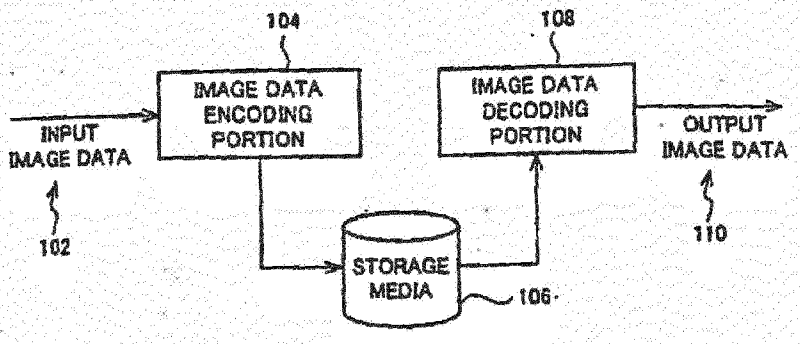

104
108
IMAGE DATA ENCODING PORTION
IMAGE DATA DECODING PORTION
INPUT IMAGE DATA
OUTPUT IMAGE DATA
102
110
STORAGE MEDIA
106

FIG.5

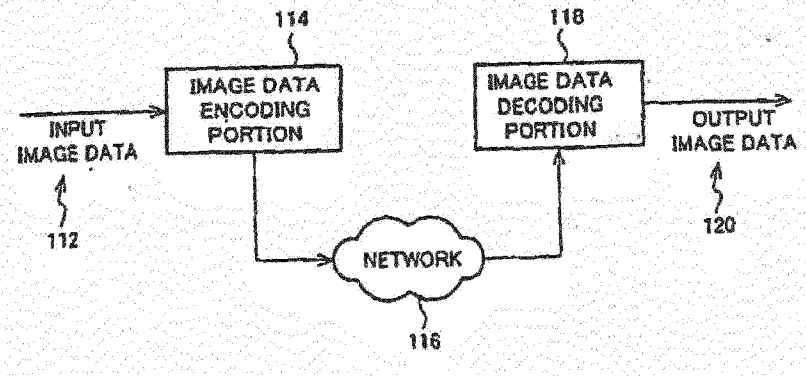

114
118
IMAGE DATA ENCODING PORTION
IMAGE DATA DECODING PORTION
INPUT IMAGE DATA
OUTPUT IMAGE DATA
112
120
NETWORK
116

FIG.6

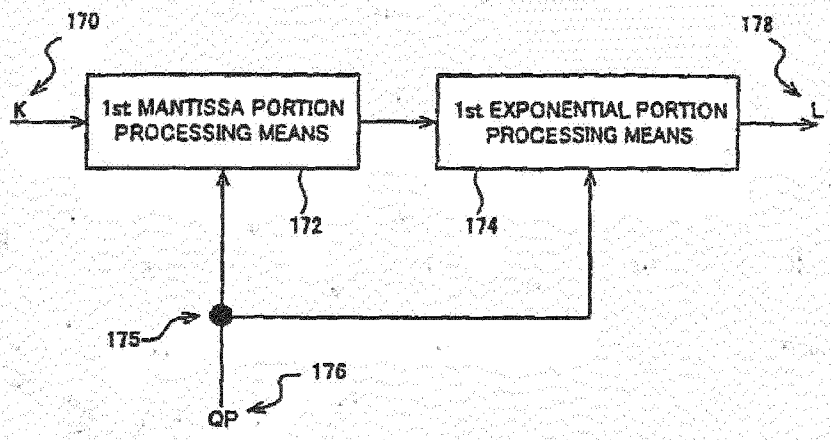

170
K
1st MANTISSA PORTION PROCESSING MEANS
172
1st EXPONENTIAL PORTION PROCESSING MEANS
174
178
L
175
176
QP

FIG.7

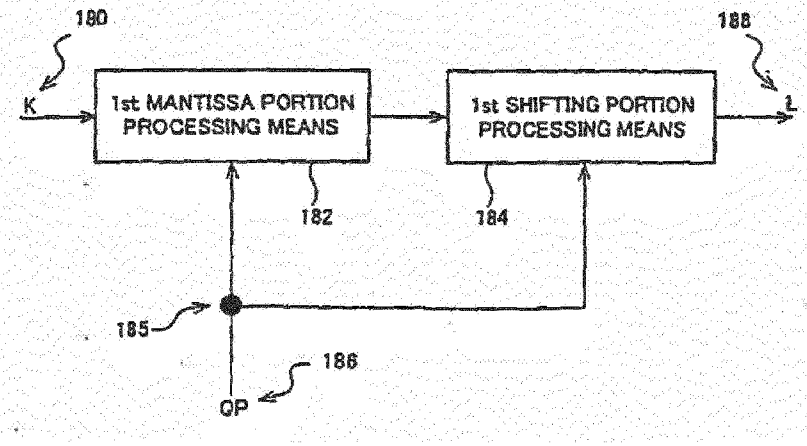

180
K
1st MANTISSA PORTION PROCESSING MEANS
182
1st SHIFTING PORTION PROCESSING MEANS
184
188
L
185
186
QP

FIG.8

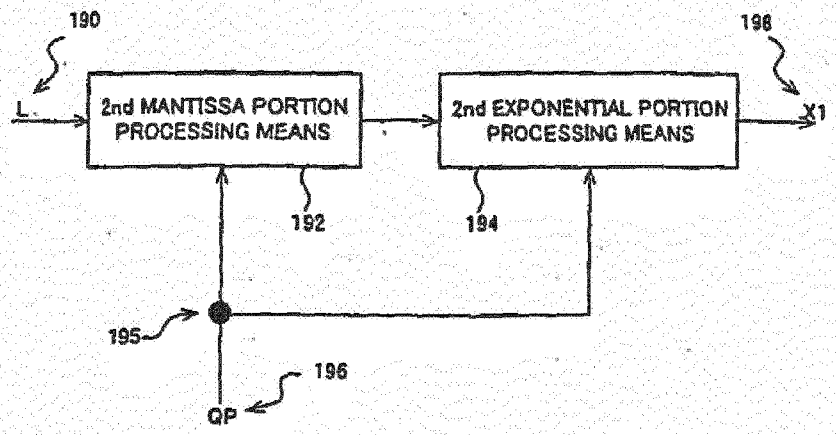
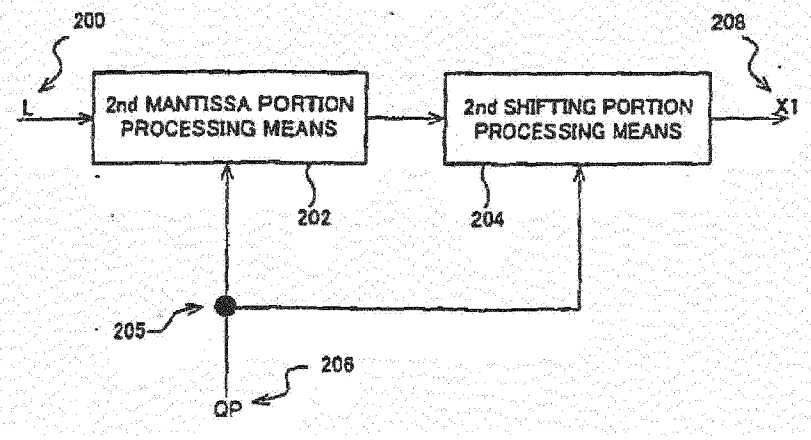

190
198
L
2nd MANTISSA PORTION PROCESSING MEANS
2nd EXPONENTIAL PORTION PROCESSING MEANS
X1
192
194
195
196
QP

FIG.9


200
208
L
2nd MANTISSA PORTION PROCESSING MEANS
2nd SHIFTING PORTION PROCESSING MEANS
X1
202
204
205
206
QP