US 20150181218A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2015/0181218 A1

(43) Pub. Date: Jun. 25, 2015

Okawa

(54) **IMAGE ENCODING APPARATUS, METHOD OF IMAGE ENCODING, AND RECORDING MEDIUM, IMAGE DECODING APPARATUS, METHOD OF IMAGE DECODING, AND RECORDING MEDIUM**

(71) Applicant: **CANON KABUSHIKI KAISHA**, Tokyo (JP)

(72) Inventor: **Koji Okawa**, Tokyo (JP)

(73) Assignee: **CANON KABUSHIKI KAISHA**, Tokyo (JP)

(21) Appl. No.: **14/411,287**

(22) PCT Filed: **Jun. 26, 2013**

(86) PCT No.: **PCT/JP2013/004003**

§ 371 (c)(1),
(2) Date: **Dec. 24, 2014**

(30) **Foreign Application Priority Data**

Jun. 29, 2012 (JP) 2012-147153

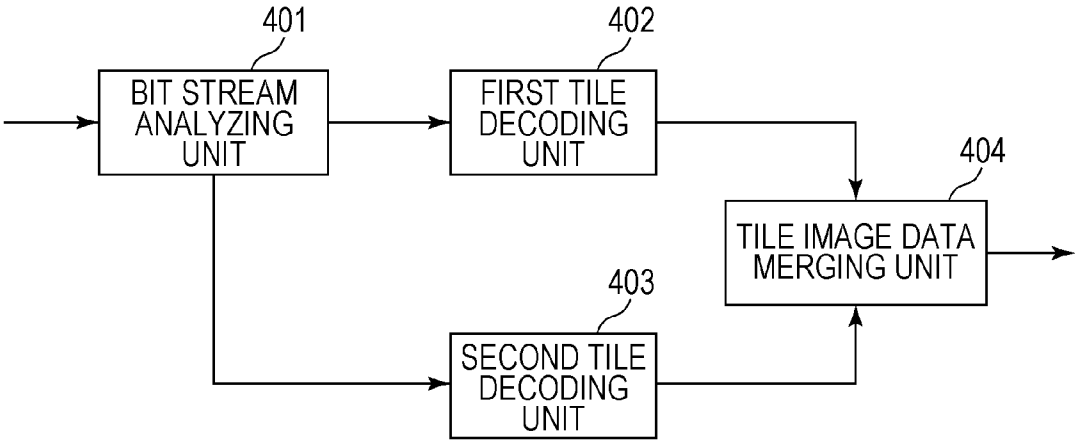
Publication Classification

(51) **Int. Cl.**
H04N 19/436 (2006.01)
H04N 19/513 (2006.01)

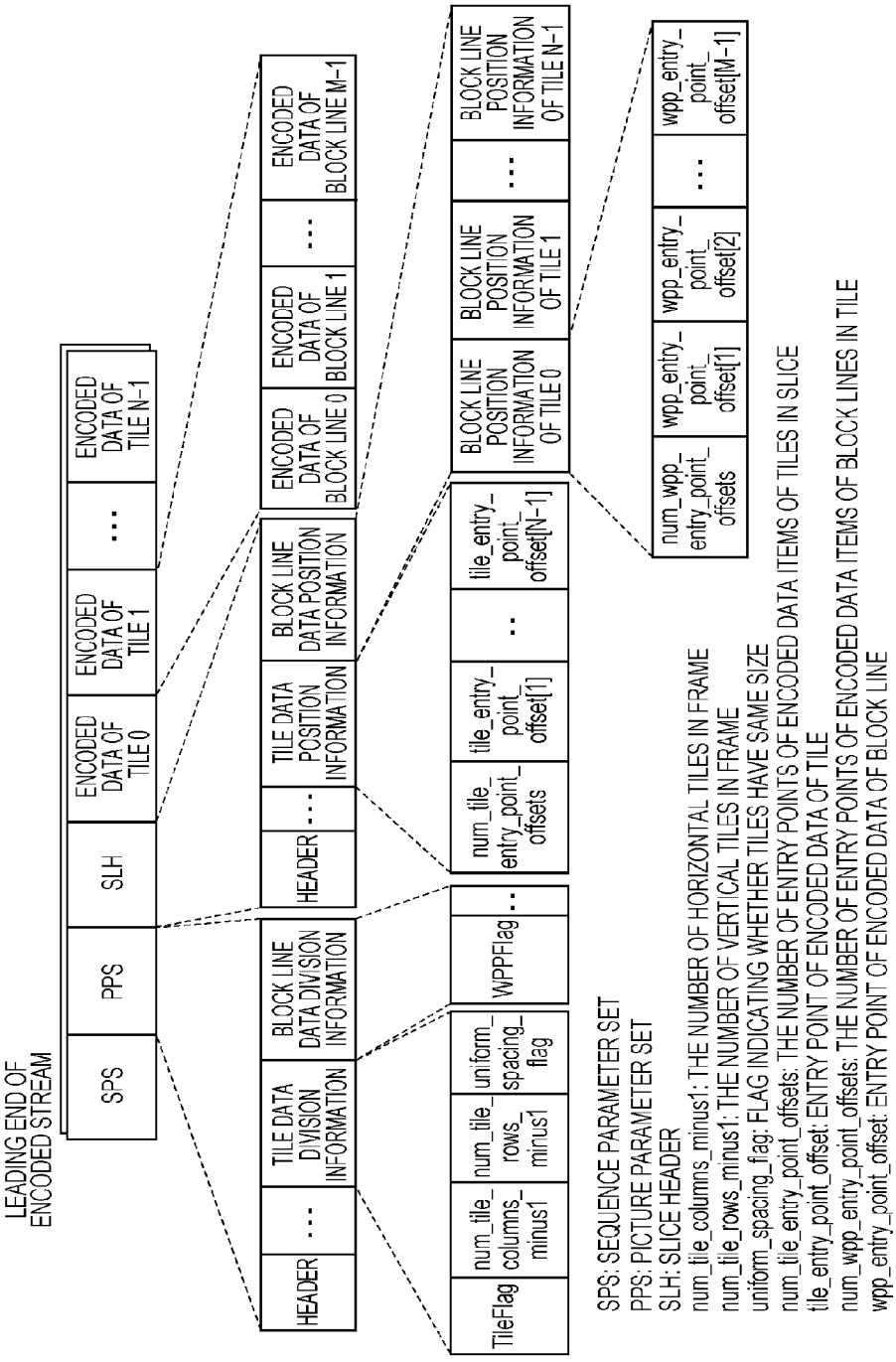
(52) **U.S. Cl.**
CPC ***H04N 19/436*** (2014.11); ***H04N 19/513*** (2014.11)

(57) **ABSTRACT**

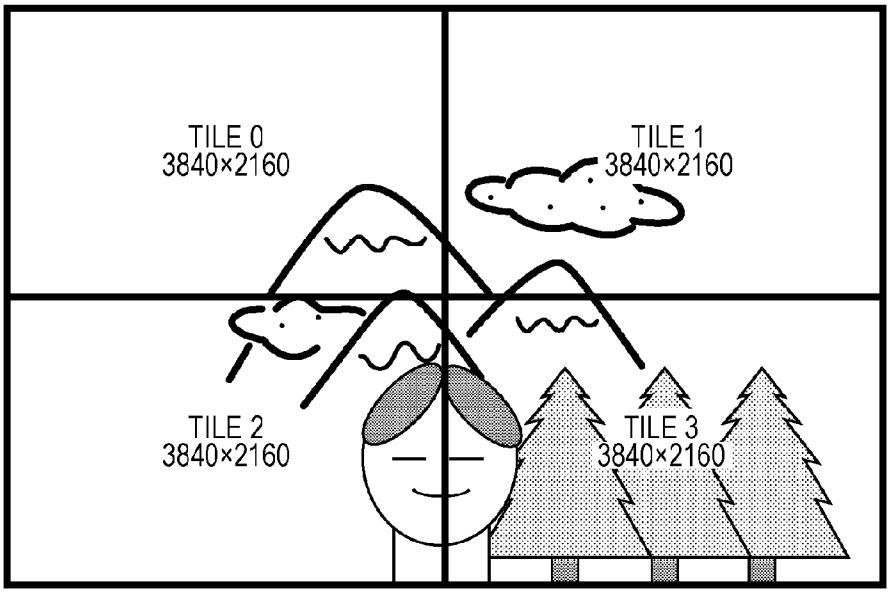
Encoded data is decoded based on tile data division information, tile data position information, block line data division information, and block line data position information. The tile data division information indicates whether the encoded data is composed of tile data items that serve as encoded data items of tiles. The tile data position information indicates positions of the tile data items. The block line data division information indicates whether each tile data item is composed of first block line data and second block line data. The first block line data serves as encoded data of a first block line that is a set of blocks arranged linearly. The second block line data serves as encoded data of a second block line next to the first block line. The block line data position information indicates a position of the second block line data.



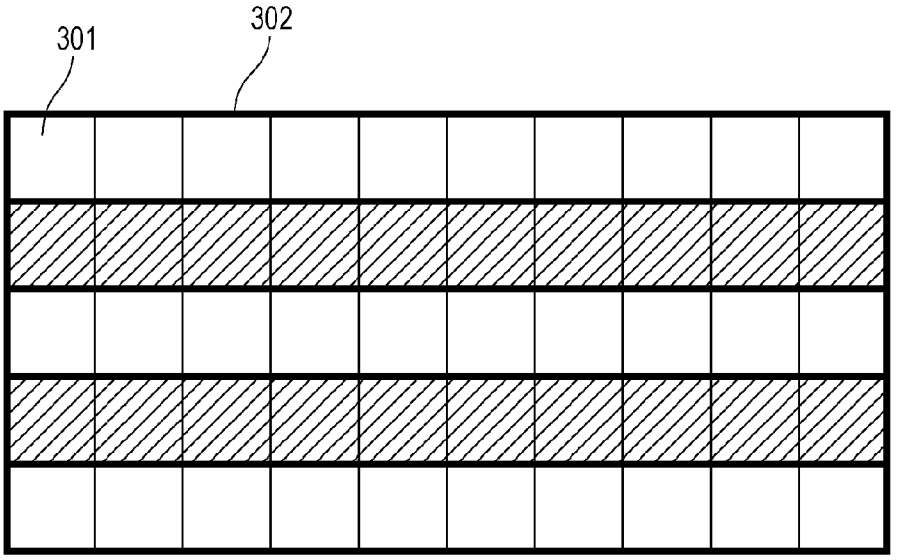
[Fig. 1]


LEADING END OF ENCODED STREAM
SPS
PPS
SLH
ENCODED DATA OF TILE 0
ENCODED DATA OF TILE 1
...
ENCODED DATA OF TILE N-1
HEADER
...
TILE DATA DIVISION INFORMATION
BLOCK LINE DATA DIVISION INFORMATION
HEADER
...
TILE DATA POSITION INFORMATION
BLOCK LINE DATA POSITION INFORMATION
ENCODED DATA OF BLOCK LINE 0
ENCODED DATA OF BLOCK LINE 1
...
ENCODED DATA OF BLOCK LINE M-1
TileFlag
num_tile_columns_minus1
num_tile_rows_minus1
uniform_spacing_flag
WPPFlag
...
num_tile_entry_point_offsets
tile_entry_point_offset[1]
..
tile_entry_point_offset[N-1]
BLOCK LINE POSITION INFORMATION OF TILE 0
BLOCK LINE POSITION INFORMATION OF TILE 1
...
BLOCK LINE POSITION INFORMATION OF TILE N-1
num_wpp_entry_point_offsets
wpp_entry_point_offset[1]
wpp_entry_point_offset[2]
...
wpp_entry_point_offset[M-1]
SPS: SEQUENCE PARAMETER SET
PPS: PICTURE PARAMETER SET
SLH: SLICE HEADER
num_tile_columns_minus1: THE NUMBER OF HORIZONTAL TILES IN FRAME
num_tile_rows_minus1: THE NUMBER OF VERTICAL TILES IN FRAME
uniform_spacing_flag: FLAG INDICATING WHETHER TILES HAVE SAME SIZE
num_tile_entry_point_offsets: THE NUMBER OF ENTRY POINTS OF ENCODED DATA ITEMS OF TILES IN SLICE
tile_entry_point_offset: ENTRY POINT OF ENCODED DATA OF TILE
num_wpp_entry_point_offsets: THE NUMBER OF ENTRY POINTS OF ENCODED DATA ITEMS OF BLOCK LINES IN TILE
wpp_entry_point_offset: ENTRY POINT OF ENCODED DATA OF BLOCK LINE

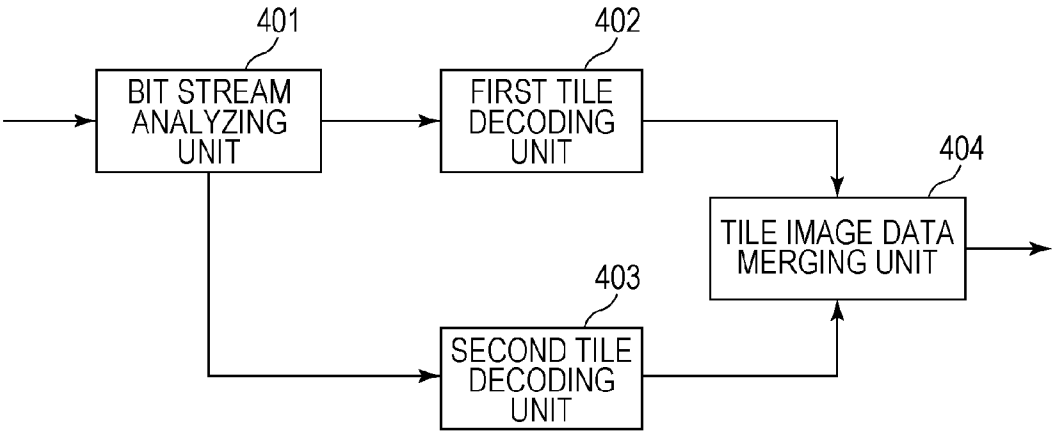
[Fig. 2]


TILE 0
3840×2160
TILE 1
3840×2160
TILE 2
3840×2160
TILE 3
3840×2160

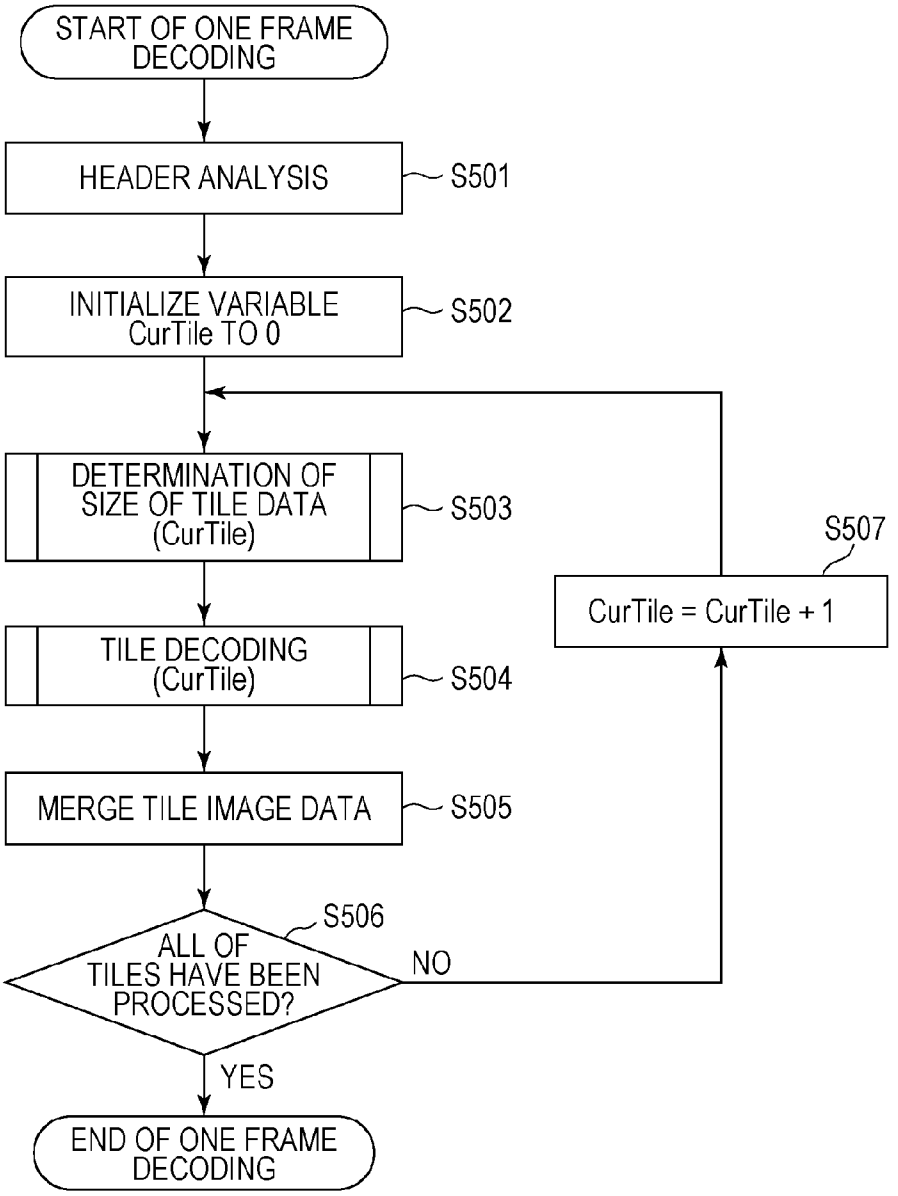
[Fig. 3]


301
302

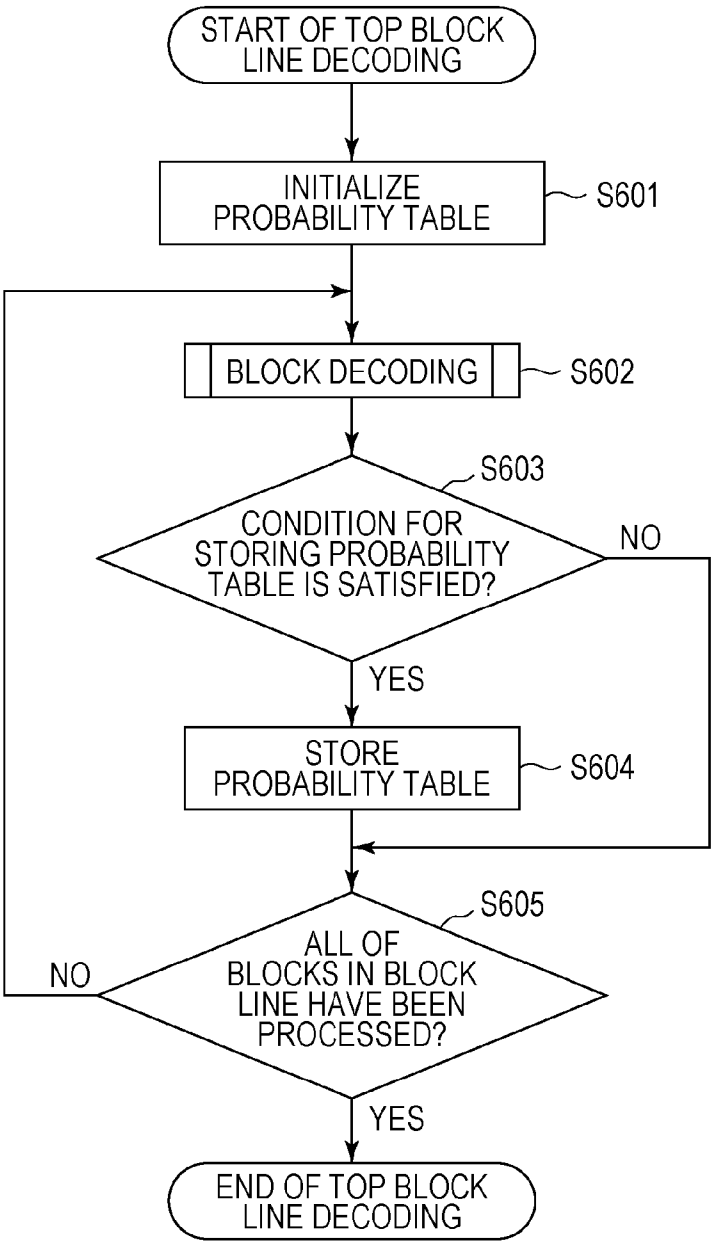
[Fig. 4]


401
BIT STREAM ANALYZING UNIT
402
FIRST TILE DECODING UNIT
404
TILE IMAGE DATA MERGING UNIT
403
SECOND TILE DECODING UNIT

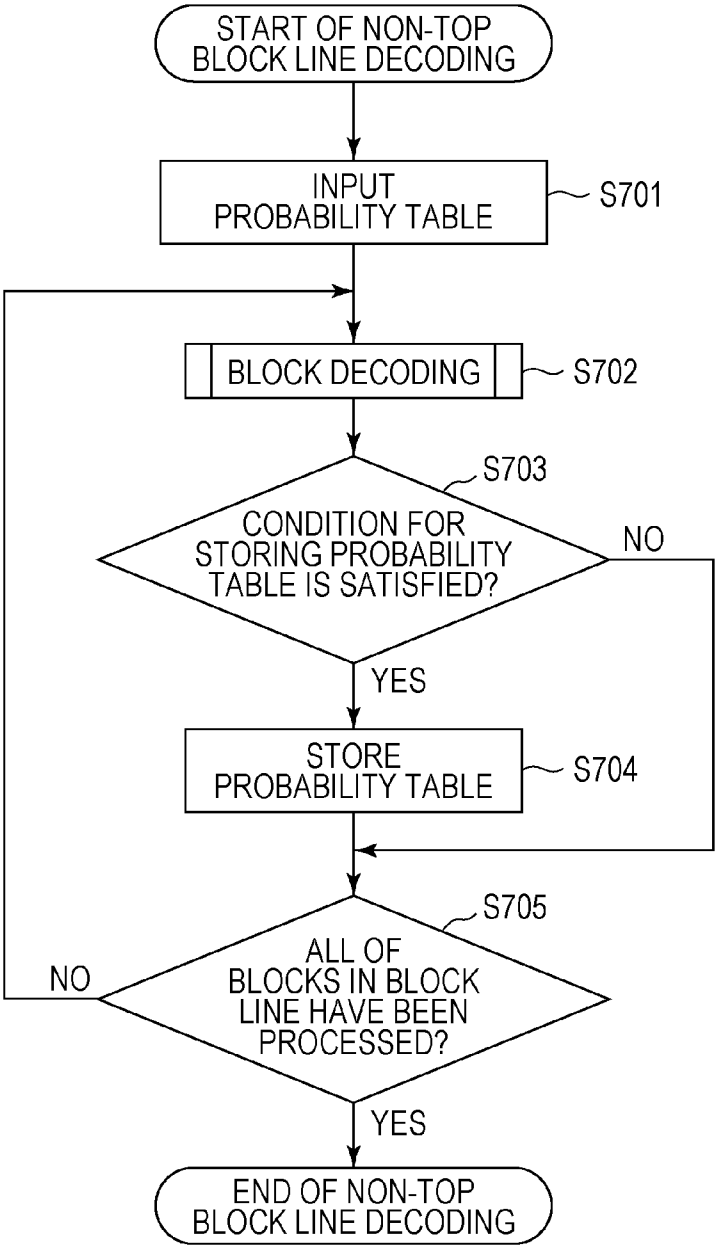
[Fig. 5]


START OF ONE FRAME DECODING
HEADER ANALYSIS
S501
INITIALIZE VARIABLE CurTile TO 0
S502
DETERMINATION OF SIZE OF TILE DATA (CurTile)
S503
TILE DECODING (CurTile)
S504
MERGE TILE IMAGE DATA
S505
ALL OF TILES HAVE BEEN PROCESSED?
S506
NO
YES
CurTile = CurTile + 1
S507
END OF ONE FRAME DECODING

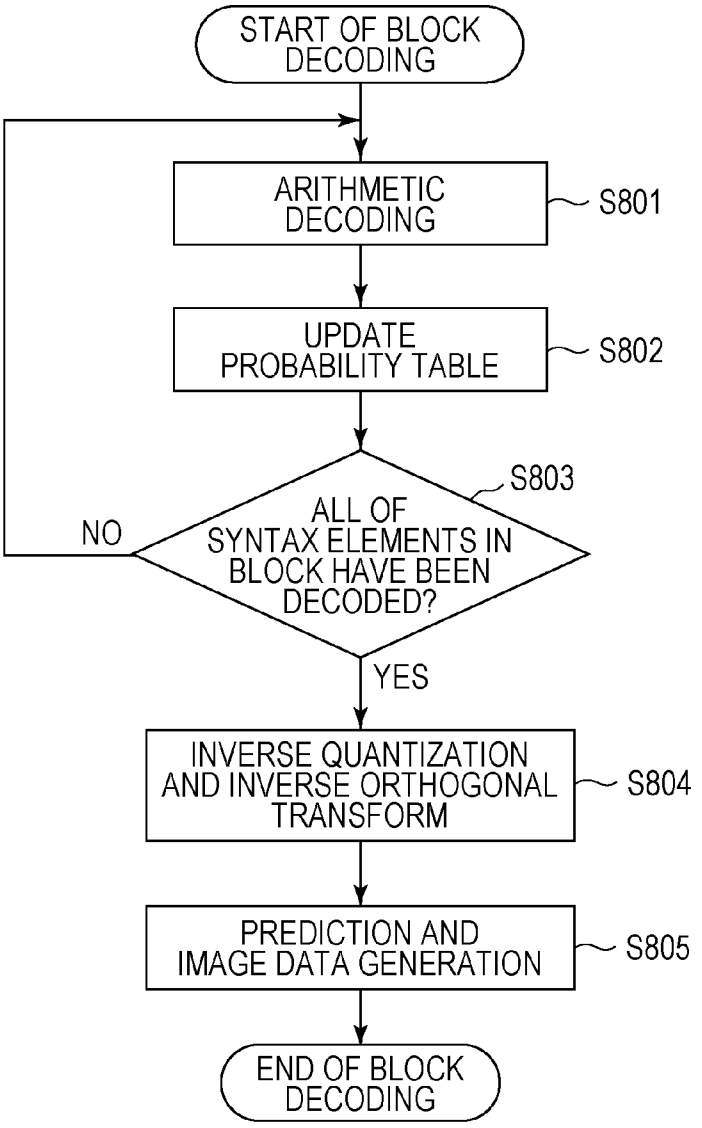
[Fig. 6]


START OF TOP BLOCK LINE DECODING
INITIALIZE PROBABILITY TABLE
S601
BLOCK DECODING
S602
S603
CONDITION FOR STORING PROBABILITY TABLE IS SATISFIED?
NO
YES
STORE PROBABILITY TABLE
S604
S605
ALL OF BLOCKS IN BLOCK LINE HAVE BEEN PROCESSED?
NO
YES
END OF TOP BLOCK LINE DECODING

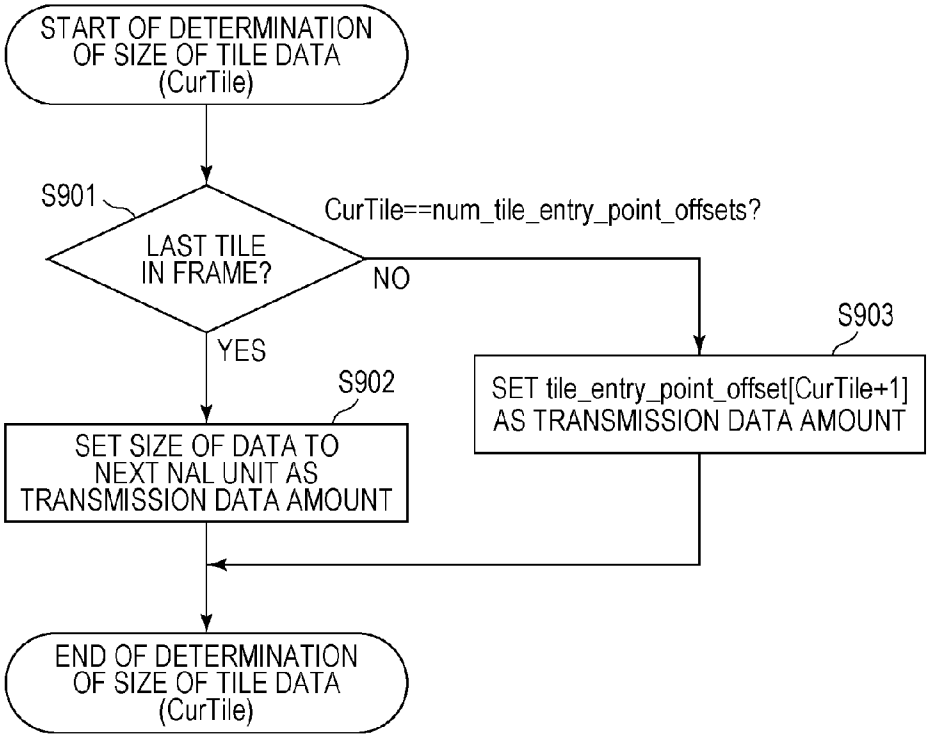
[Fig. 7]


START OF NON-TOP BLOCK LINE DECODING
INPUT PROBABILITY TABLE
S701
BLOCK DECODING
S702
S703
CONDITION FOR STORING PROBABILITY TABLE IS SATISFIED?
NO
YES
STORE PROBABILITY TABLE
S704
S705
ALL OF BLOCKS IN BLOCK LINE HAVE BEEN PROCESSED?
NO
YES
END OF NON-TOP BLOCK LINE DECODING

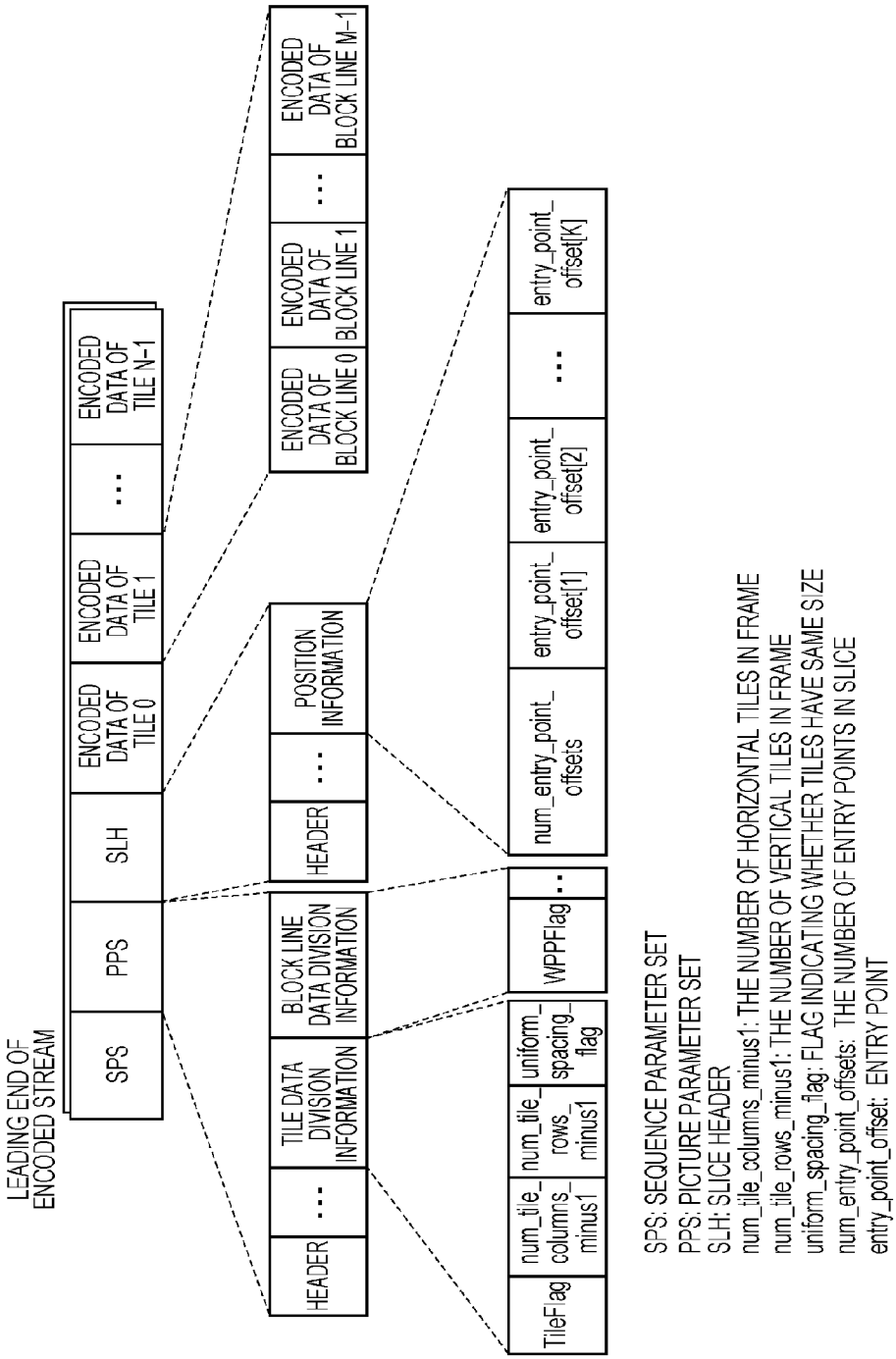
[Fig. 8]


START OF BLOCK DECODING
ARITHMETIC DECODING
S801
UPDATE PROBABILITY TABLE
S802
S803
ALL OF SYNTAX ELEMENTS IN BLOCK HAVE BEEN DECODED?
NO
YES
INVERSE QUANTIZATION AND INVERSE ORTHOGONAL TRANSFORM
S804
PREDICTION AND IMAGE DATA GENERATION
S805
END OF BLOCK DECODING

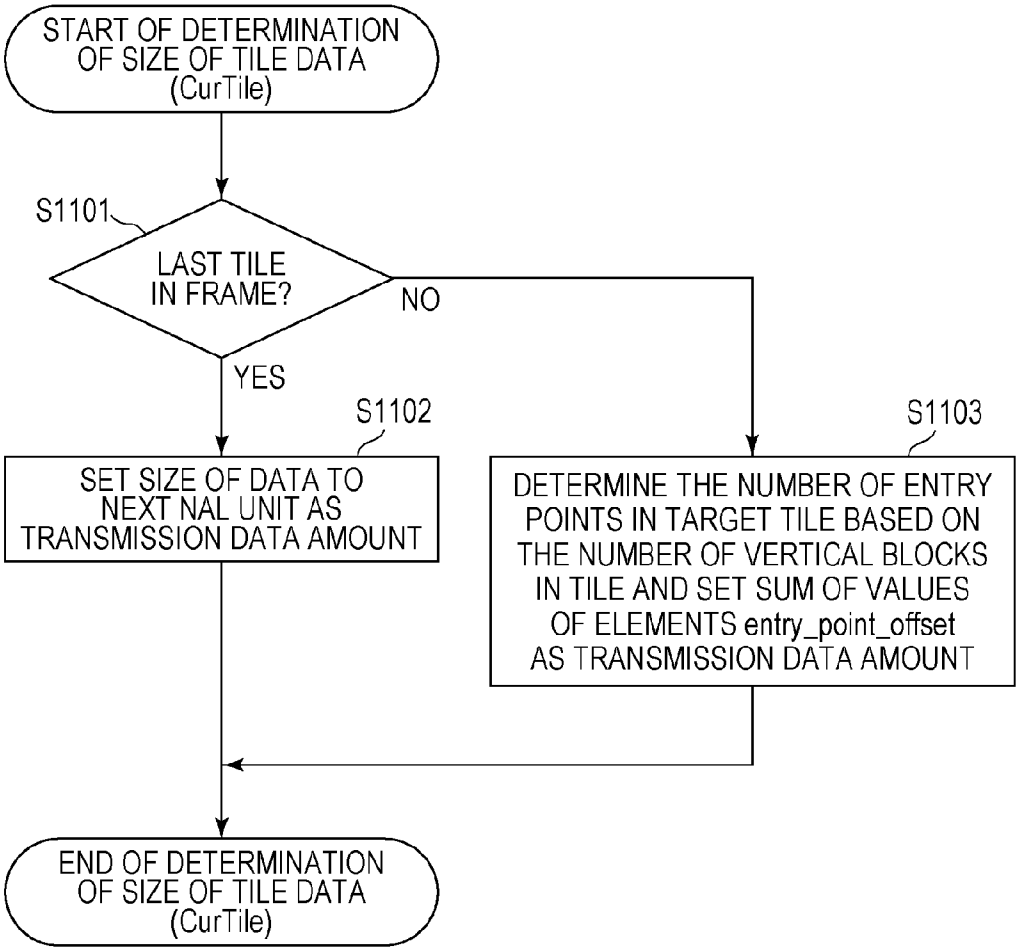
[Fig. 9]


START OF DETERMINATION OF SIZE OF TILE DATA (CurTile)
S901
LAST TILE IN FRAME?
CurTile==num_tile_entry_point_offsets?
NO
YES
S902
SET SIZE OF DATA TO NEXT NAL UNIT AS TRANSMISSION DATA AMOUNT
S903
SET tile_entry_point_offset[CurTile+1] AS TRANSMISSION DATA AMOUNT
END OF DETERMINATION OF SIZE OF TILE DATA (CurTile)

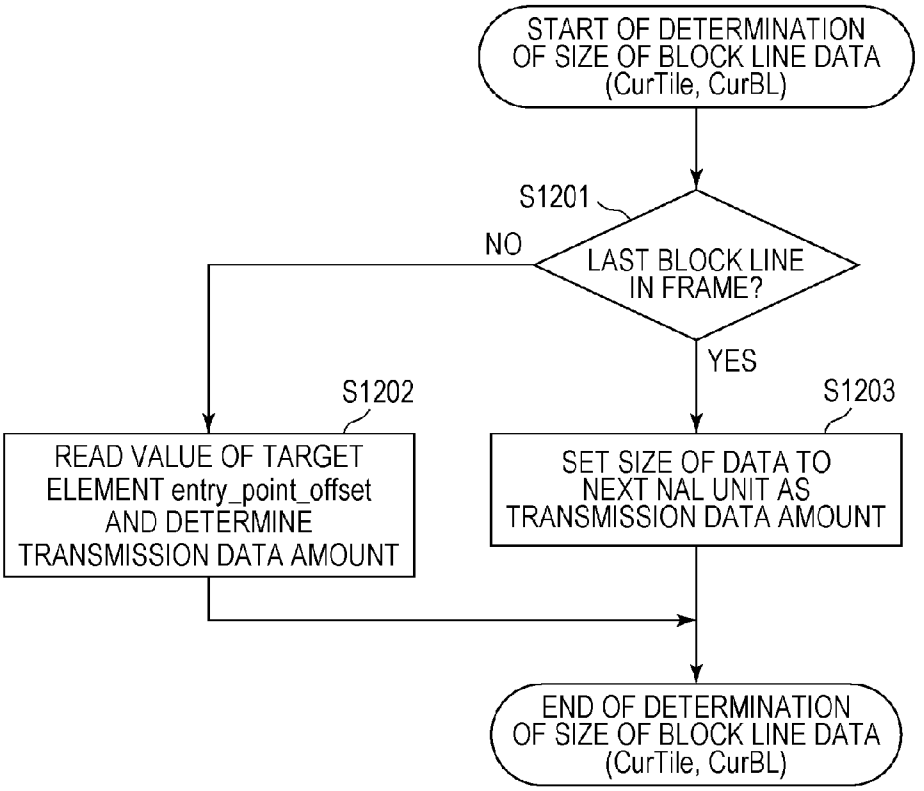
[Fig. 10]


LEADING END OF ENCODED STREAM
SPS
PPS
SLH
ENCODED DATA OF TILE 0
ENCODED DATA OF TILE 1
...
ENCODED DATA OF TILE N-1
HEADER
...
TILE DATA DIVISION INFORMATION
BLOCK LINE DATA DIVISION INFORMATION
HEADER
...
POSITION INFORMATION
ENCODED DATA OF BLOCK LINE 0
ENCODED DATA OF BLOCK LINE 1
...
ENCODED DATA OF BLOCK LINE M-1
TileFlag
num_tile_columns_minus1
num_tile_rows_minus1
uniform_spacing_flag
WPPFlag
...
num_entry_point_offsets
entry_point_offset[1]
entry_point_offset[2]
...
entry_point_offset[K]
SPS: SEQUENCE PARAMETER SET
PPS: PICTURE PARAMETER SET
SLH: SLICE HEADER
num_tile_columns_minus1: THE NUMBER OF HORIZONTAL TILES IN FRAME
num_tile_rows_minus1: THE NUMBER OF VERTICAL TILES IN FRAME
uniform_spacing_flag: FLAG INDICATING WHETHER TILES HAVE SAME SIZE
num_entry_point_offsets: THE NUMBER OF ENTRY POINTS IN SLICE
entry_point_offset: ENTRY POINT

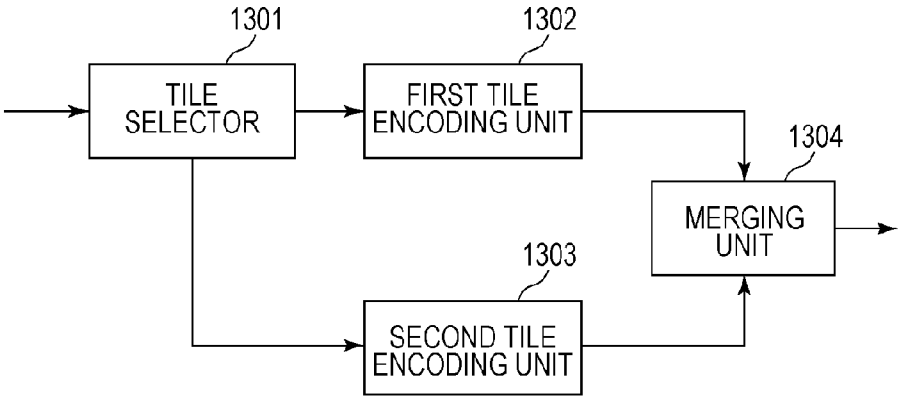
[Fig. 11]


START OF DETERMINATION OF SIZE OF TILE DATA (CurTile)
S1101
LAST TILE IN FRAME?
NO
YES
S1102
SET SIZE OF DATA TO NEXT NAL UNIT AS TRANSMISSION DATA AMOUNT
S1103
DETERMINE THE NUMBER OF ENTRY POINTS IN TARGET TILE BASED ON THE NUMBER OF VERTICAL BLOCKS IN TILE AND SET SUM OF VALUES OF ELEMENTS entry_point_offset AS TRANSMISSION DATA AMOUNT
END OF DETERMINATION OF SIZE OF TILE DATA (CurTile)

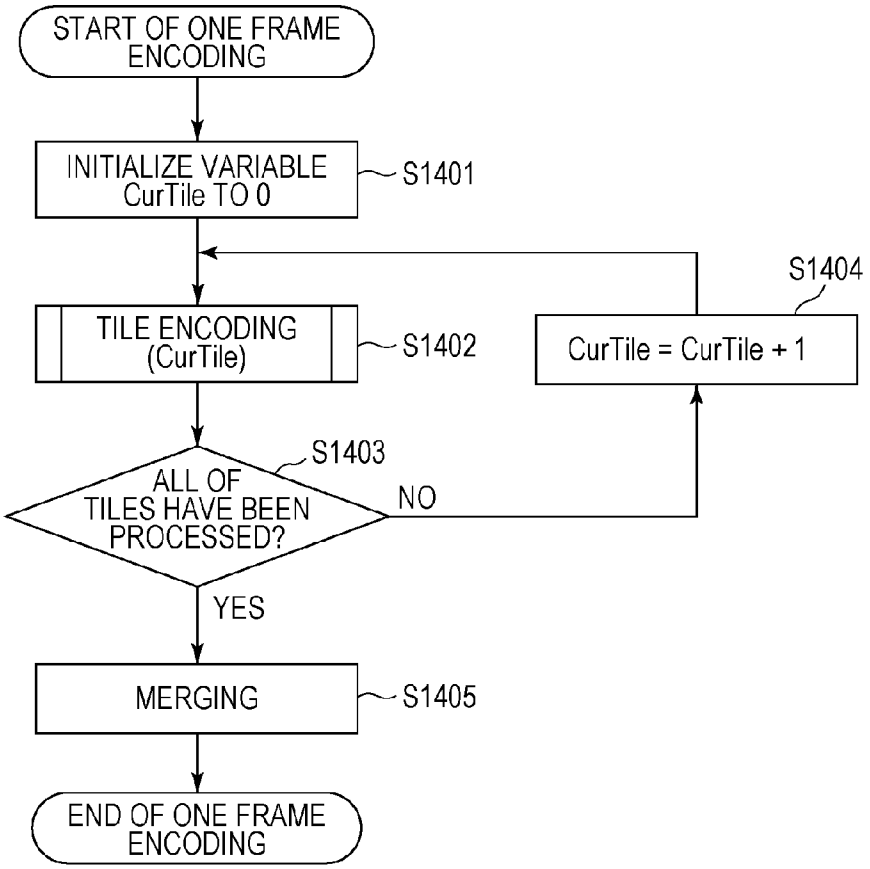
[Fig. 12]


START OF DETERMINATION OF SIZE OF BLOCK LINE DATA (CurTile, CurBL)
S1201
LAST BLOCK LINE IN FRAME?
NO
YES
S1202
READ VALUE OF TARGET ELEMENT entry_point_offset AND DETERMINE TRANSMISSION DATA AMOUNT
S1203
SET SIZE OF DATA TO NEXT NAL UNIT AS TRANSMISSION DATA AMOUNT
END OF DETERMINATION OF SIZE OF BLOCK LINE DATA (CurTile, CurBL)

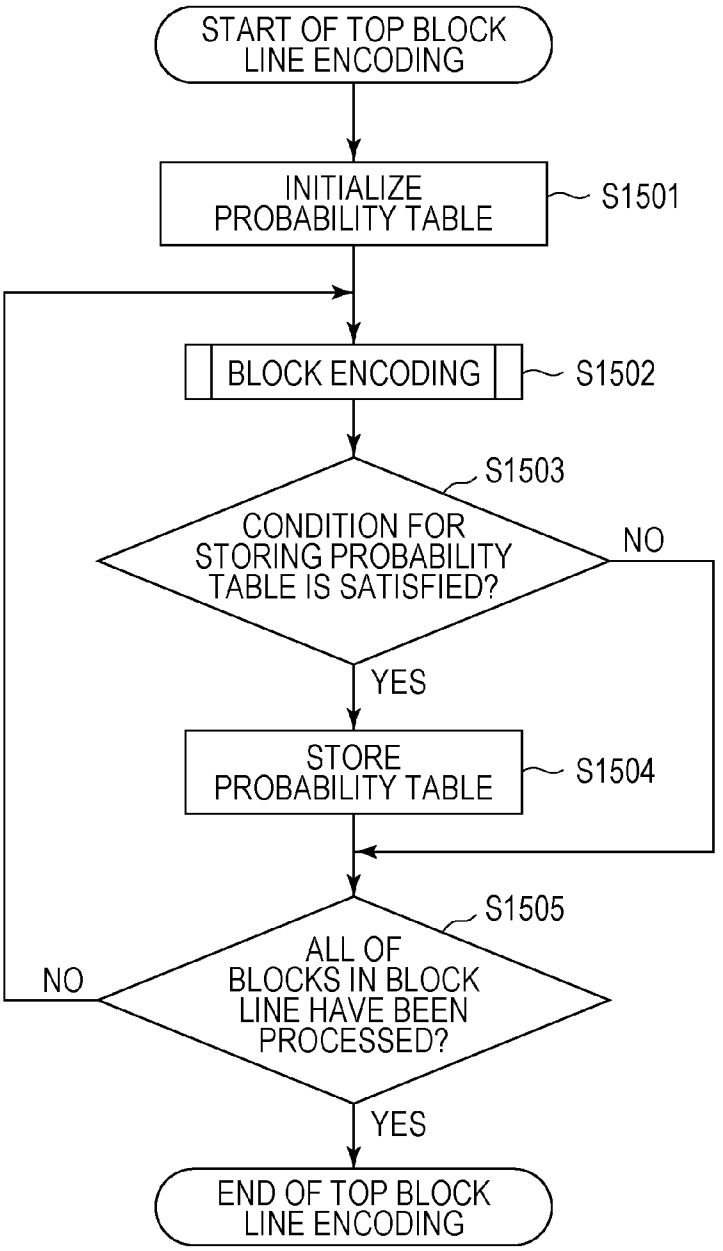
[Fig. 13]


1301
TILE SELECTOR
1302
FIRST TILE ENCODING UNIT
1304
MERGING UNIT
1303
SECOND TILE ENCODING UNIT

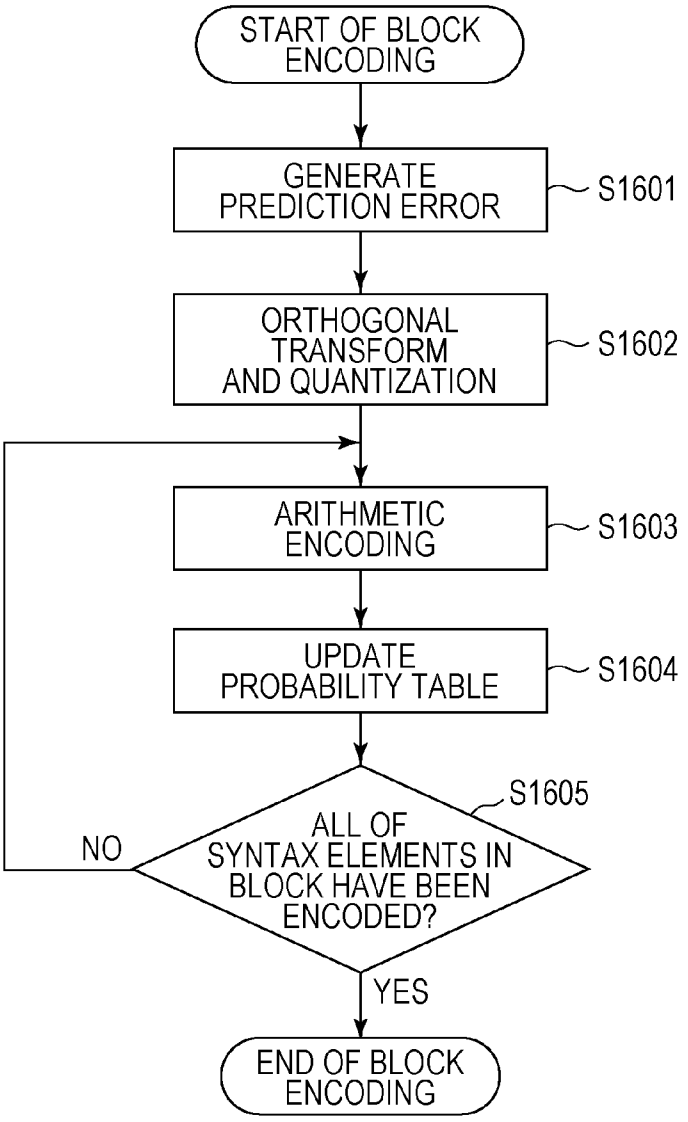
[Fig. 14]


START OF ONE FRAME ENCODING
INITIALIZE VARIABLE CurTile TO 0
S1401
TILE ENCODING (CurTile)
S1402
S1403
ALL OF TILES HAVE BEEN PROCESSED?
NO
YES
S1404
CurTile = CurTile + 1
MERGING
S1405
END OF ONE FRAME ENCODING

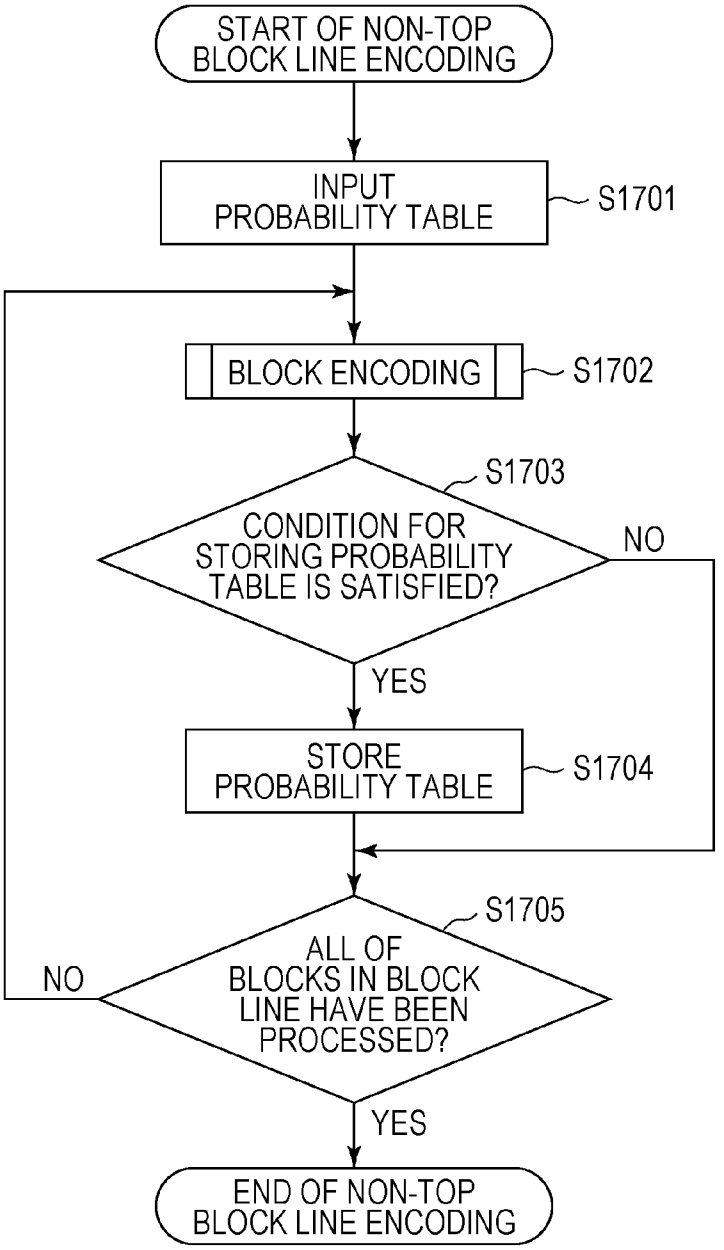
[Fig. 15]


START OF TOP BLOCK LINE ENCODING
INITIALIZE PROBABILITY TABLE
S1501
BLOCK ENCODING
S1502
S1503
CONDITION FOR STORING PROBABILITY TABLE IS SATISFIED?
NO
YES
STORE PROBABILITY TABLE
S1504
S1505
ALL OF BLOCKS IN BLOCK LINE HAVE BEEN PROCESSED?
NO
YES
END OF TOP BLOCK LINE ENCODING

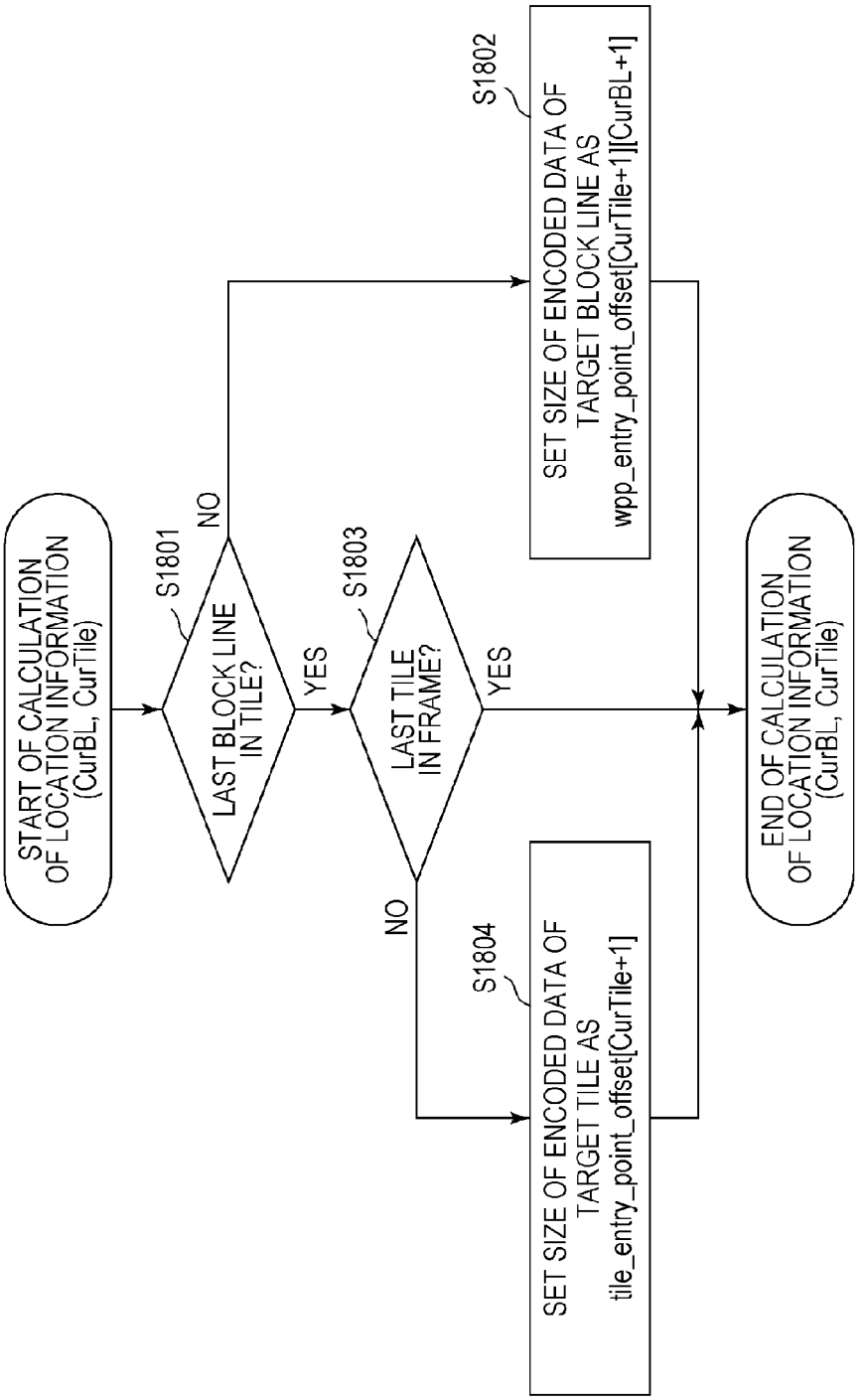
[Fig. 16]


START OF BLOCK ENCODING
GENERATE PREDICTION ERROR
S1601
ORTHOGONAL TRANSFORM AND QUANTIZATION
S1602
ARITHMETIC ENCODING
S1603
UPDATE PROBABILITY TABLE
S1604
S1605
ALL OF SYNTAX ELEMENTS IN BLOCK HAVE BEEN ENCODED?
NO
YES
END OF BLOCK ENCODING

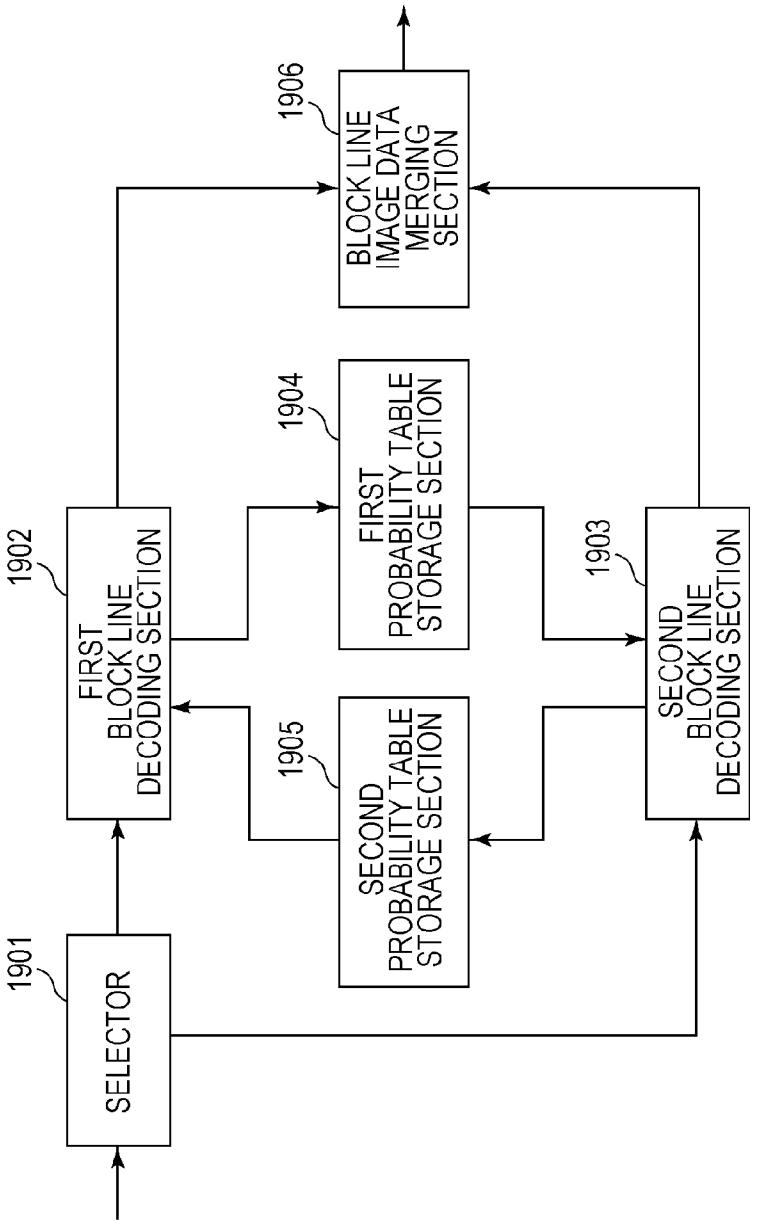
[Fig. 17]


START OF NON-TOP BLOCK LINE ENCODING
INPUT PROBABILITY TABLE
S1701
BLOCK ENCODING
S1702
S1703
CONDITION FOR STORING PROBABILITY TABLE IS SATISFIED?
NO
YES
STORE PROBABILITY TABLE
S1704
S1705
ALL OF BLOCKS IN BLOCK LINE HAVE BEEN PROCESSED?
NO
YES
END OF NON-TOP BLOCK LINE ENCODING

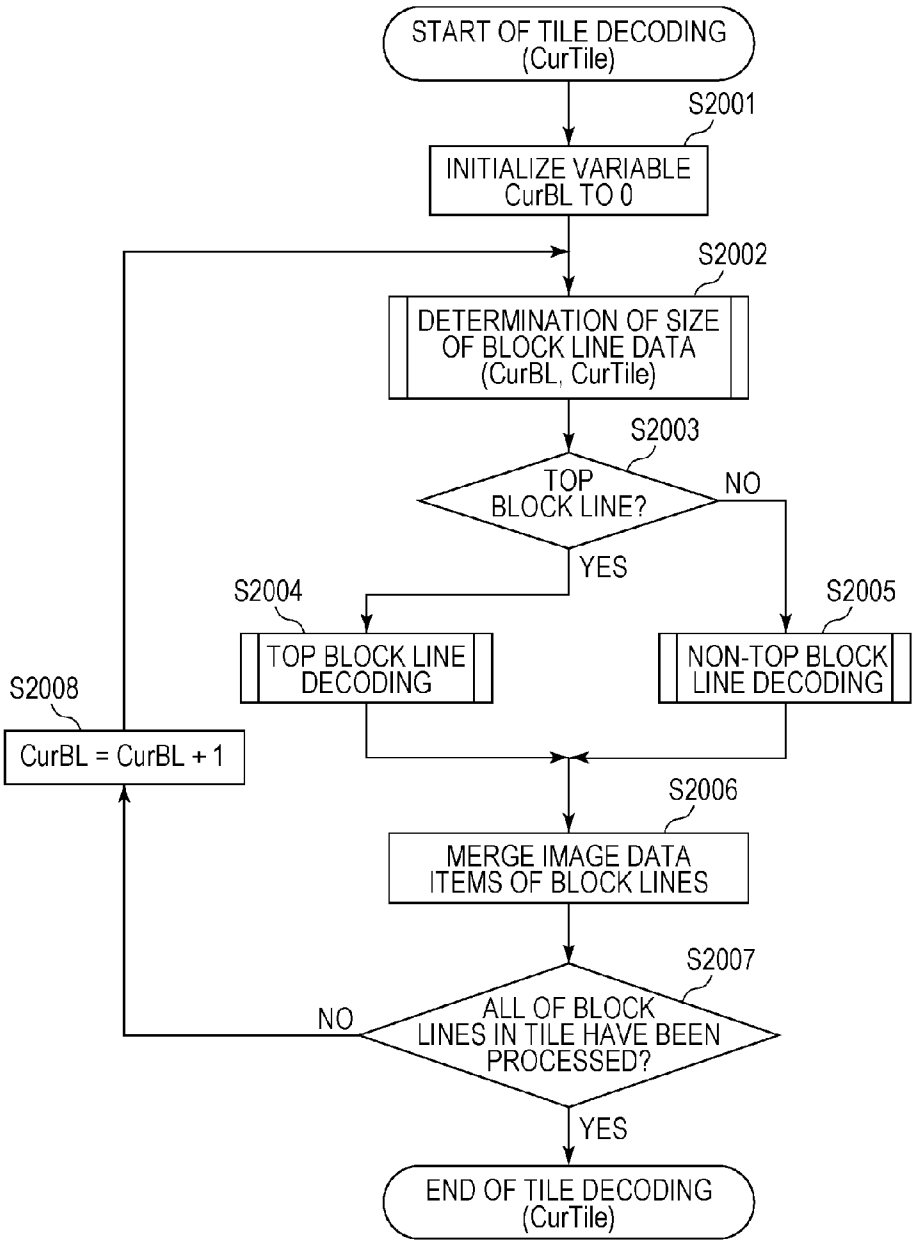
[Fig. 18]


START OF CALCULATION OF LOCATION INFORMATION (CurBL, CurTile)
S1801
LAST BLOCK LINE IN TILE?
NO
YES
S1802
SET SIZE OF ENCODED DATA OF TARGET BLOCK LINE AS wpp_entry_point_offset[CurTile+1][CurBL+1]
S1803
LAST TILE IN FRAME?
NO
YES
S1804
SET SIZE OF ENCODED DATA OF TARGET TILE AS tile_entry_point_offset[CurTile+1]
END OF CALCULATION OF LOCATION INFORMATION (CurBL, CurTile)

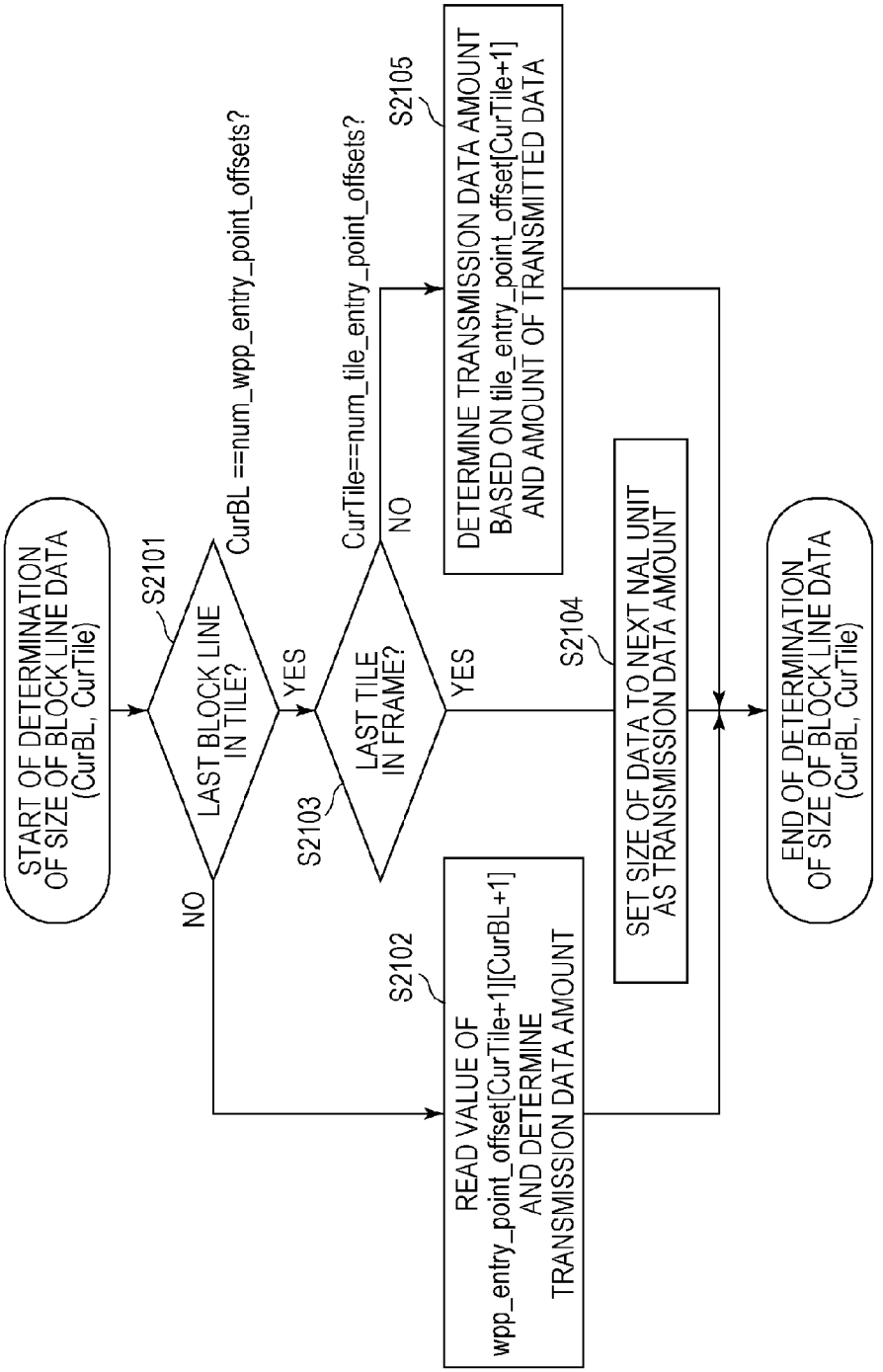
[Fig. 19]


1901
SELECTOR
1902
FIRST BLOCK LINE DECODING SECTION
1903
SECOND BLOCK LINE DECODING SECTION
1904
FIRST PROBABILITY TABLE STORAGE SECTION
1905
SECOND PROBABILITY TABLE STORAGE SECTION
1906
BLOCK LINE IMAGE DATA MERGING SECTION

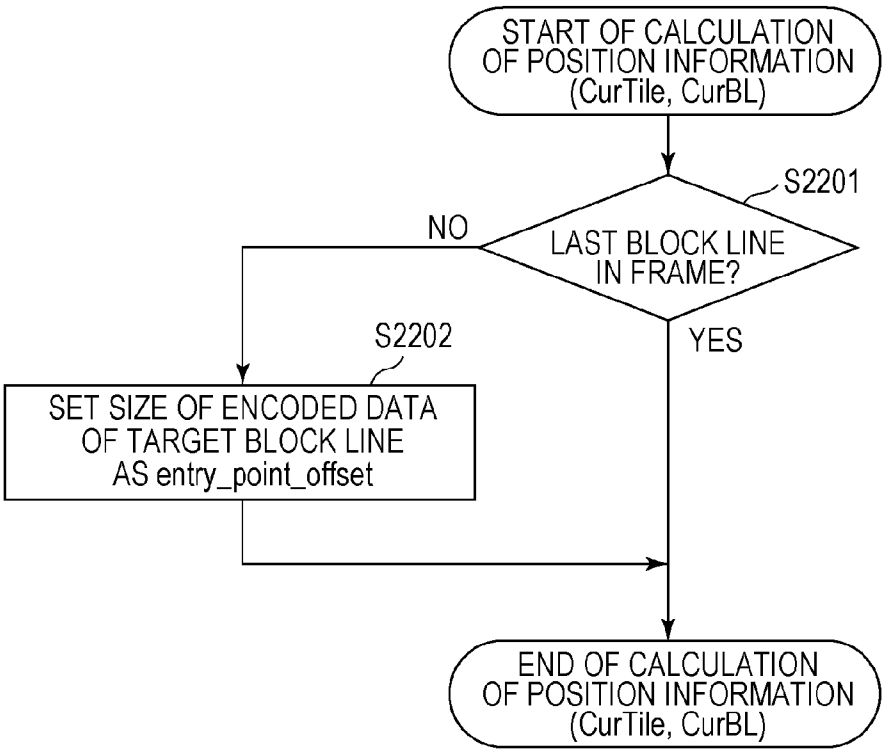
[Fig. 20]


START OF TILE DECODING (CurTile)
S2001
INITIALIZE VARIABLE CurBL TO 0
S2002
DETERMINATION OF SIZE OF BLOCK LINE DATA (CurBL, CurTile)
S2003
TOP BLOCK LINE?
NO
YES
S2004
TOP BLOCK LINE DECODING
S2005
NON-TOP BLOCK LINE DECODING
S2006
MERGE IMAGE DATA ITEMS OF BLOCK LINES
S2007
ALL OF BLOCK LINES IN TILE HAVE BEEN PROCESSED?
NO
YES
S2008
CurBL = CurBL + 1
END OF TILE DECODING (CurTile)

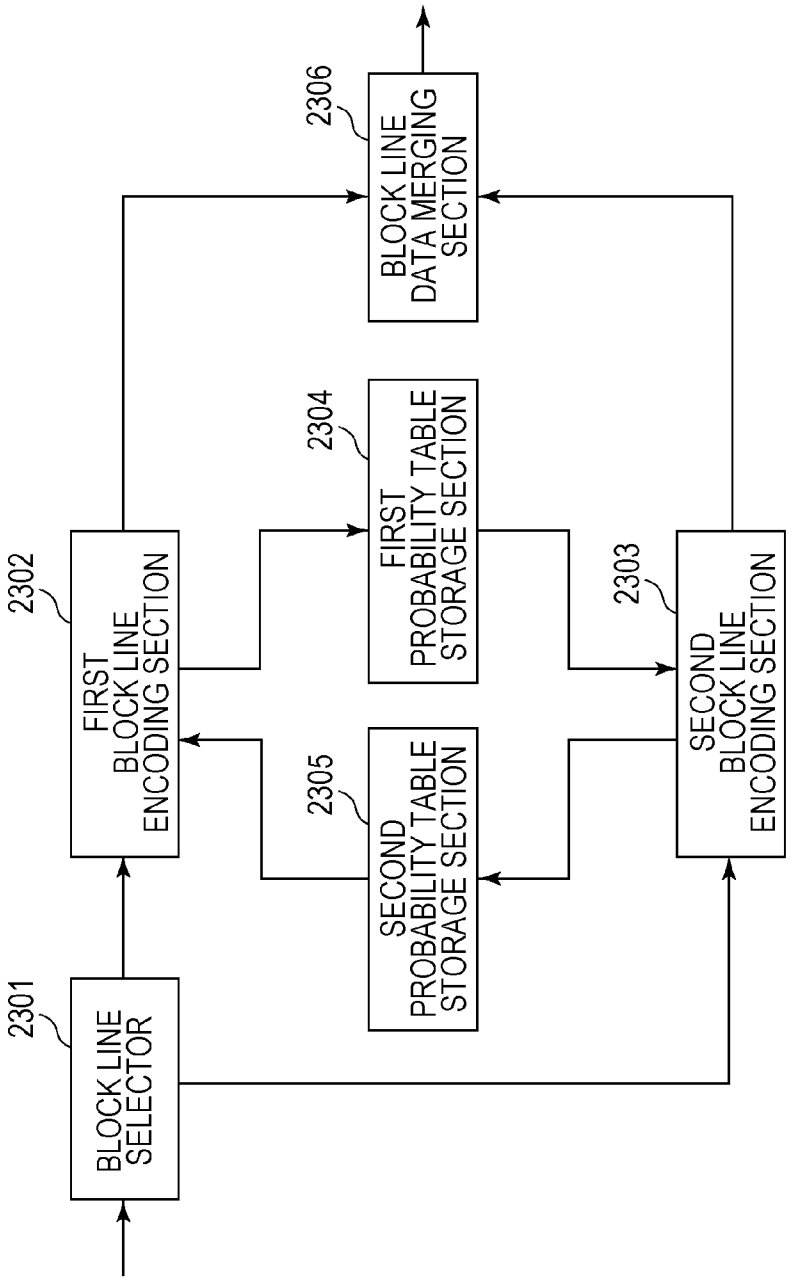
[Fig. 21]


START OF DETERMINATION OF SIZE OF BLOCK LINE DATA (CurBL, CurTile)
S2101
LAST BLOCK LINE IN TILE?
CurBL ==num_wpp_entry_point_offsets?
YES
NO
S2103
LAST TILE IN FRAME?
CurTile==num_tile_entry_point_offsets?
YES
NO
S2104
S2105
DETERMINE TRANSMISSION DATA AMOUNT BASED ON tile_entry_point_offset[CurTile+1] AND AMOUNT OF TRANSMITTED DATA
S2102
READ VALUE OF wpp_entry_point_offset[CurTile+1][CurBL+1] AND DETERMINE TRANSMISSION DATA AMOUNT
SET SIZE OF DATA TO NEXT NAL UNIT AS TRANSMISSION DATA AMOUNT
END OF DETERMINATION OF SIZE OF BLOCK LINE DATA (CurBL, CurTile)

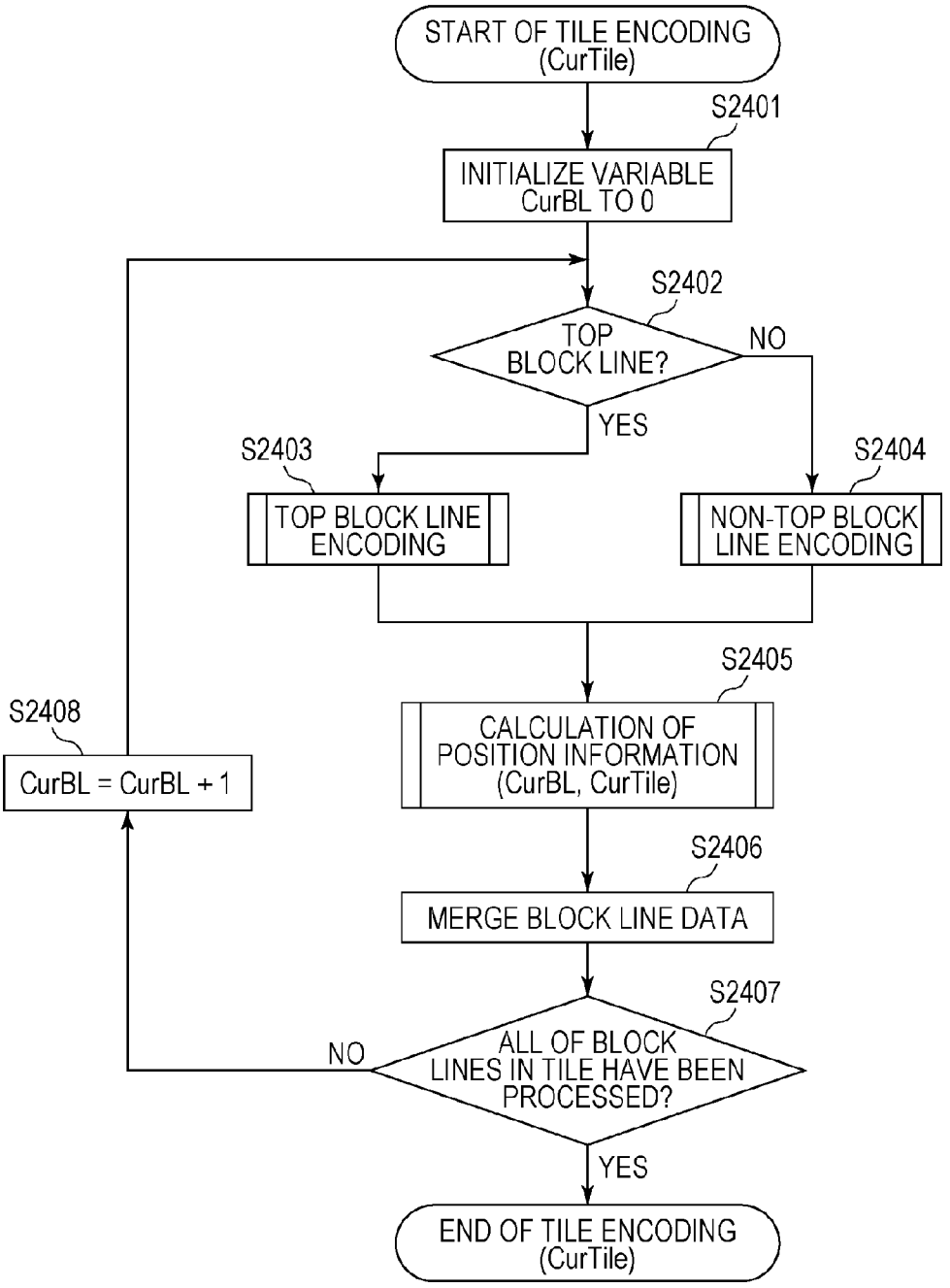
[Fig. 22]


START OF CALCULATION OF POSITION INFORMATION (CurTile, CurBL)
S2201
LAST BLOCK LINE IN FRAME?
NO
YES
S2202
SET SIZE OF ENCODED DATA OF TARGET BLOCK LINE AS entry_point_offset
END OF CALCULATION OF POSITION INFORMATION (CurTile, CurBL)

[Fig. 23]


2306
BLOCK LINE DATA MERGING SECTION
2304
FIRST PROBABILITY TABLE STORAGE SECTION
2302
FIRST BLOCK LINE ENCODING SECTION
2303
SECOND BLOCK LINE ENCODING SECTION
2305
SECOND PROBABILITY TABLE STORAGE SECTION
2301
BLOCK LINE SELECTOR

[Fig. 24]


START OF TILE ENCODING (CurTile)
S2401
INITIALIZE VARIABLE CurBL TO 0
S2402
TOP BLOCK LINE?
NO
YES
S2403
TOP BLOCK LINE ENCODING
S2404
NON-TOP BLOCK LINE ENCODING
S2405
CALCULATION OF POSITION INFORMATION (CurBL, CurTile)
S2406
MERGE BLOCK LINE DATA
S2407
ALL OF BLOCK LINES IN TILE HAVE BEEN PROCESSED?
NO
YES
S2408
CurBL = CurBL + 1
END OF TILE ENCODING (CurTile)

[Fig. 25]

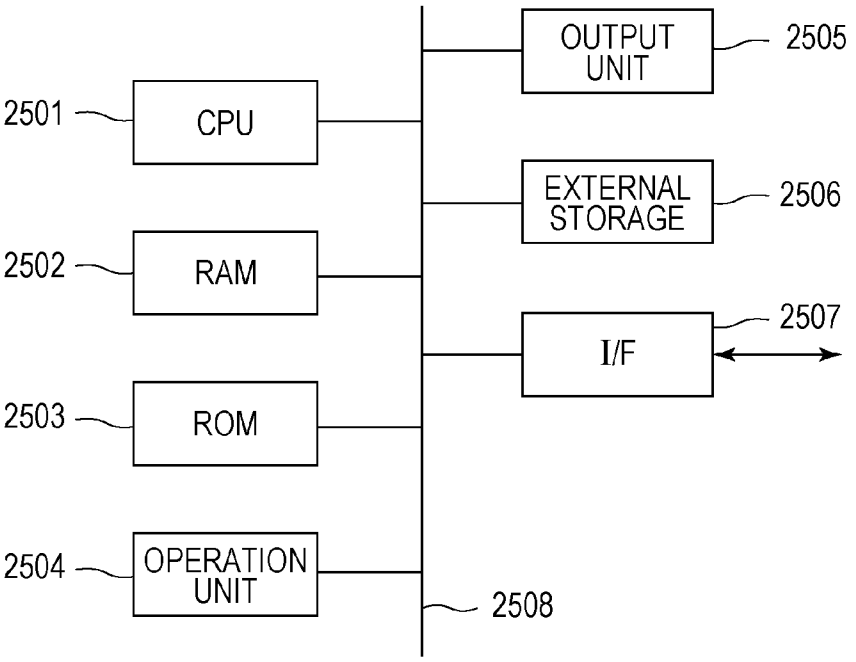

OUTPUT UNIT
2505
2501
CPU
EXTERNAL STORAGE
2506
2502
RAM
2507
I/F
2503
ROM
2504
OPERATION UNIT
2508

IMAGE ENCODING APPARATUS, METHOD OF IMAGE ENCODING, AND RECORDING MEDIUM, IMAGE DECODING APPARATUS, METHOD OF IMAGE DECODING, AND RECORDING MEDIUM

TECHNICAL FIELD

[0001] The present invention relates to an image encoding apparatus, a method of image encoding, and a recording medium, an image decoding apparatus, a method of image decoding, and a recording medium. In particular, the invention relates to encoding and decoding based on parallel processing of images in which each picture is divided into rectangular tiles.

BACKGROUND ART

[0002] One of known encoding methods used for movie compression recording is H.264/MPEG-4 AVC (hereinafter, "H.264") (ITU-T H.264 (03/2010) Advanced video coding for generic audio visual services). According to H.264, each picture can be divided into a plurality of slices and image data can be encoded on a slice-by-slice basis. The slices have little dependence on one another. Accordingly, the slices can be subjected to encoding or decoding in parallel. One of significant benefits of the division into slices is that the execution of parallel processing by, for example, a multicore CPU enables reduction of processing time.

[0003] Furthermore, each slice is encoded by a related-art binary arithmetic encoding scheme used in H.264. Specifically, each syntax element is binarized, thus generating a binary signal. Each syntax element is assigned the probability of occurrence as a table (hereinafter, "probability table") in advance. The binary signal is subjected to arithmetic encoding based on the probability table. The probability table is used as decoding information for decoding subsequent symbols during decoding and is used as encoding information for encoding subsequent symbols during encoding. Each time encoding is performed, the probability table is updated based on statistical information indicating whether the encoded binary signal is a symbol assigned higher probability or not.

[0004] Recently, activities for international standardization of higher-efficiency encoding as a successor to H.264 have been started and Joint Collaborative Team on Video Coding (JCT-VC) has been established by ISO/IEC and ITU-T. According to JCT-VC, the standardization of High Efficiency Video Coding (HEVC) is in progress.

[0005] For the standardization of HEVC, various encoding tools have been broadly examined from viewpoints of not only improvement of encoding efficiency but also ease of implementation and reduction of processing time. For reduction of processing time, schemes for enhancing parallelism have been examined. One of them is a scheme, called Wavefront, for parallel entropy encoding/decoding (JCT-VC document JCTVC-I1003.doc, the Internet <http://phenix.int-evry.fr/jct/doc). Since a target binary signal to be encoded has to be encoded using the probability table updated constantly, parallel processing cannot be performed if statistical information is not reset. If the statistical information is reset, the encoding efficiency is unfortunately reduced. On the other hand, according to Wavefront, the probability table upon encoding of a plurality of blocks at predetermined positions is used for the left block in the next line, thus achieving parallel encoding of blocks on a line-by-line basis while preventing a reduction in encoding efficiency. Although Wavefront has been mainly described with respect to encoding, the same applies to decoding.

[0006] According to HEVC, the schemes for enhancing the parallelism include tiling. According to this technique, tiles allow a picture to be divided into rectangular regions that can independently be processed. Thus, high throughput can be achieved by parallel encoding/decoding and the capacity of a memory included in each of an encoding apparatus and a decoding apparatus can be reduced.

[0007] According to HEVC, processing, such as tiling or Wavefront, has been exclusively performed using a symbol tiles_or_entropy_coding_sync_idc. If the symbol has a value of 0, this means that a picture is composed of a single tile and Wavefront parallel processing is not performed. If the symbol has a value of 1, this means that the picture is composed of a plurality of tiles but Wavefront parallel processing is not performed. If the symbol has a value of 2, this means that the picture is composed of a single tile and parallel processing, such as Wavefront, is performed. If the symbol has a value of 3, this means that the picture is composed of a single tile, Wavefront parallel processing is not performed, and entropy slices which can independently be decoded are used. Other values cannot be used. The reason is as follows. If a plurality of processes are performed in parallel on a sufficiently low definition image, controlling such parallel processing would be complicated. Because the degree of complexity would be too high for the picture size, exclusive processing is performed. In particular, real-time decoding of a very large image, such as an 8k or 4k image, requires high parallelism. The parallelism can be enhanced by dividing the image into very small tiles, but the encoding efficiency is reduced because information available for prediction is limited at the boundary between tiles. Wavefront parallel processing performed on such a very large image divided into block lines requires a large-capacity line buffer and, unfortunately, does not enable the parallelism to be enhanced more than that using tiles, because the block lines have a dependence on each other, the dependence including the probability table. As regards such a large image, it is necessary to divide the image into segments having a certain size, allocate the segments to computer nodes, and operate a plurality of processors for each node. For example, in the case where tiles are allocated to nodes and processing is performed on a node-by-node basis, unfortunately, parallel processing, such as Wavefront, cannot be performed in the tiles.

[0008] The present invention provides an encoding format that enables multi-stage parallel processing to enhance parallelism of processes and achieve high-speed encoding and decoding.

SUMMARY OF INVENTION

[0009] The present invention provides an image decoding apparatus that decodes encoded data obtained by dividing a frame of image data into rectangular tiles and encoding the image data on a tile-by-tile basis, the apparatus including the following elements. A deriving unit is configured to derive tile data division information, tile data position information, block line data division information, and block line data position information from the encoded data. The tile data division information indicates whether the encoded data is composed of tile data items that serve as encoded data items of the tiles. The tile data position information indicates positions of the tile data items. The block line data division information indi-

cates whether each tile data item is composed of first block line data and second block line data, the first block line data serving as encoded data of a first block line that is a set of blocks arranged linearly, the second block line data serving as encoded data of a second block line next to the first block line. The block line data position information indicates a position of the second block line data. A first decoding unit is configured to decode the first block line data based on the information derived by the deriving unit. A second decoding unit is configured to decode the second block line data based on the information derived by the deriving unit and decoding information obtained by decoding encoded data of a block in a predetermined position belonging to the first block line.

[0010] According to the present invention, an encoding format that enables multi-stage parallel processing is provided, thus enhancing the parallelism of processes and achieving high-speed encoding and decoding.

[0011] Further features of the present invention will become apparent from the following description of exemplary embodiments with reference to the attached drawings.

BRIEF DESCRIPTION OF DRAWINGS

[0012] FIG. **1** is a diagram illustrating an image encoding format to be decoded by an image decoding apparatus according to a first embodiment.

[0013] FIG. **2** is a diagram illustrating division of an image into tiles.

[0014] FIG. **3** is a diagram illustrating division of a tile into block lines.

[0015] FIG. **4** is a block diagram illustrating a configuration of each of the image decoding apparatus according to the first embodiment and an image decoding apparatus according to a second embodiment.

[0016] FIG. **5** is a flowchart illustrating an operation of the image decoding apparatus according to each of the first and second embodiments.

[0017] FIG. **6** is a flowchart illustrating a process of top block line decoding in each of the first and second embodiments.

[0018] FIG. **7** is a flowchart illustrating a process of non-top block line decoding in the first and second embodiments.

[0019] FIG. **8** is a flowchart illustrating a process of block decoding in the first and second embodiments.

[0020] FIG. **9** is a flowchart illustrating a process of determination of the size of tile data in the first embodiment.

[0021] FIG. **10** is a diagram illustrating an image encoding format to be decoded by the image decoding apparatus according to the second embodiment.

[0022] FIG. **11** is a flowchart illustrating a method of determination of the size of tile data in the second embodiment.

[0023] FIG. **12** is a flowchart illustrating a method of determination of the size of block line data in the second embodiment.

[0024] FIG. **13** is a block diagram illustrating a configuration of an image encoding apparatus according to each of a third embodiment and a fourth embodiment.

[0025] FIG. **14** is a flowchart illustrating an operation of the image encoding apparatus according to the third embodiment.

[0026] FIG. **15** is a flowchart illustrating a process of top block line encoding in the third embodiment.

[0027] FIG. **16** is a flowchart illustrating a process of block encoding in the third embodiment.

[0028] FIG. **17** is a flowchart illustrating a process of non-top block line encoding in the third embodiment.

[0029] FIG. **18** is a flowchart illustrating a process of calculating position information in the third embodiment.

[0030] FIG. **19** is a block diagram illustrating a configuration of each tile decoding unit in the first and second embodiments.

[0031] FIG. **20** is a flowchart illustrating a process of tile decoding in the first and second embodiments.

[0032] FIG. **21** is a flowchart illustrating a process of determination of the size of block line data in the first embodiment.

[0033] FIG. **22** is a flowchart illustrating a process of calculating position information in the fourth embodiment.

[0034] FIG. **23** is a block diagram illustrating a configuration of each tile encoding unit in the third and fourth embodiments.

[0035] FIG. **24** is a flowchart illustrating a process of tile encoding in the third and fourth embodiments.

[0036] FIG. **25** is a block diagram illustrating an exemplary configuration of hardware of a computer applicable to each decoding apparatus according to the present invention.

DESCRIPTION OF EMBODIMENTS

[0037] Preferred embodiments of the present invention will be described in detail below with reference to the accompanying drawings. Configurations illustrated in the following embodiments are intended for purposes of illustration only and the present invention is not limited to the illustrated configurations.

First Embodiment

[0038] In an image encoding format to be decoded by an image decoding apparatus according to a first embodiment of the present invention, one frame is divided into a plurality of tiles that are rectangular regions. FIG. **2** illustrates tile arrangement in the first embodiment. Referring to FIG. **2**, a frame of 7680 horizontal pixels×4320 vertical pixels is divided into four tiles arranged in an array of two horizontal tiles×two vertical tiles. In the encoding format in the present invention, each block is composed of 16 horizontal pixels×16 vertical pixels and encoding or decoding is performed on a block-by-block basis. Although each block is composed of 16×16 pixels in the present embodiment, the present invention is not limited thereto. Each block may be composed of 32×32 pixels or 64×64 pixels. The size of each tile is a multiple of the size of the block. In FIG. **2**, each tile is composed of **240** horizontal blocks×135 vertical blocks. The tile is further divided into a plurality of block lines. Each block line is a set of blocks arranged linearly in the tile as illustrated in FIG. **3**. Referring to FIG. **3**, each square defined by thin lines denotes a block **301** and each rectangle defined by thick lines denotes a block line **302**.

[0039] FIG. **1** illustrates a format of HEVC-encoded data in the present embodiment. In FIG. **1**, the encoded data includes a sequence parameter set that is header information containing information about sequence encoding. The encoded data further includes a picture parameter set that is header information containing information about picture encoding, a slice header that is header information containing information about slice encoding, and multiple encoded data items of tiles. The picture parameter set includes tile data division information and block line data division information. The

slice header includes tile data position information and block line data position information.

[0040] The picture parameter set includes, as the tile data division information, a flag TileFlag indicating whether a frame has been divided into tiles and image data of each tile has been encoded. If the flag TileFlag has a value of 1, this indicates that the frame has been divided into tiles and image data of each tile has been encoded. If the flag TileFlag does not have a value of 1, this indicates that the frame encoded has not been divided into tiles. In the case where the flag TileFlag has a value of 1, information num_tile_columns_minus1 indicating the number of horizontal tiles in the frame, information num_tile_rows_minus1 indicating the number of vertical tiles, and a flag uniform_spacing_flag indicating the shape of a tile are arranged. The information num_tile_columns_minus1 has a value obtained by subtracting 1 from the actual number of horizontal tiles in the frame. The information num_tile_rows_minus1 has a value obtained by subtracting 1 from the actual number of vertical tiles. Referring to FIG. **1**, N denotes the number of tiles in a slice, the number of tiles being uniquely determined by (num_tile_columns_minus1+1)×(num_tile_rows_minus1 +1). The flag uniform_spacing_flag indicates whether the tiles in the frame have the same size. If this flag has a value of 1, this indicates that the tiles have the same size. FIG. **2** illustrates a case where the flag uniform_spacing_flag =1, namely, the tiles having the same size are used. If this flag has a value of 0, this indicates that the tiles have difference sizes. Accordingly, the different sizes have to be encoded as syntax element.

[0041] The picture parameter set includes, as the block line data division information, a flag WPPFlag. If the flag WPPFlag has a value of 1, this indicates that each of the tiles has been divided into block lines and image data of each block line has been encoded. Although the block line data division information and the tile data division information include different flags in the present embodiment, the present invention is not limited thereto. It is only required to know that tile data division and block line data division are used in combination. For example, a syntax element indicating the combination of the two flags may be provided and be defined as follows.

[0042] If this syntax element has a value of 0, this means the absence of tile data division and block line data division. If the syntax element has a value of 1, this means the presence of tile data division and the absence of block line data division. If the syntax element has a value of 2, this means the absence of tile data division and the presence of block line data division. If the syntax element has a value of 3, this means the presence of tile data division and block line data division.

[0043] The slice header includes the tile data position information if the flag TileFlag has a value of 1, and further includes the block line data position information if the flag WPPFlag has a value of 1. The tile data position information includes a syntax element num_tile_entry_point_offsets and syntax elements tile_entry_point_offset. The block line data position information includes sets equal in number to the tiles, each set including a syntax element num_wpp_entry_point_offsets and syntax elements wpp_entry_point_offset. The syntax element num_tile_entry_point_offsets has a value obtained by subtracting 1 from N that indicates the number of tiles in the slice. The syntax element num_wpp_entry_point_offsets has a value obtained by subtracting 1 from M that indicates the number of block lines in the tile. If the flag TileFlag has a value of 0, the syntax element num_tile_entry_point_offsets is treated as having a value of 0. If the flag WPPFlag has a value of 0, a symbol corresponding to the syntax element num_wpp_entry_point_offsets is omitted and the syntax element num_wpp_entry_point_offsets is treated as having a value of 0.

[0044] The syntax element num_tile_entry_point_offsets indicates the number of entry points of encoded data items of the tiles in the slice. The number of entry points of encoded data items of the tiles is uniquely determined by the number of tiles included in the slice. When the number of tiles is 2, the number of entry points is 1. When the number of tiles is 4, the number of entry points is 3. In the present embodiment, it is assumed that the frame is composed of a single slice including four tiles. That is, the syntax element num_tile_entry_point_offsets has a value of 3. The syntax elements tile_entry_point_offset each indicate an entry point of encoded data of the tile, namely, a leading position of the encoded data of the tile. The number of syntax elements tile_entry_point_offset is equal to the value indicated by the syntax element num_tile_entry_point_offsets. A syntax element tile_entry_point_offset[i] indicates an entry point of encoded data of the ith tile. Since it is known that encoded data of the zeroth tile (Tile **0**) immediately follows the slice header, an entry point of the zeroth tile is omitted. The size of encoded data of the (i−1)th tile has been encoded as the syntax element tile_entry_point_offset[i]. If the syntax element num_tile_entry_point_offsets has a value of 0, no mention is made of the syntax elements tile_entry_point_offset because the syntax elements tile_entry_point_offset are not needed.

[0045] The syntax element num_wpp_entry_point_offsets indicates the number of entry points of encoded data items of block lines belonging to the tile. Since the number of vertical blocks in each tile is 135 in the present embodiment, the syntax element num_wpp_entry_point_offsets has a value of 134. The syntax elements wpp_entry_point_offset each indicate an entry point of encoded data of the block line, namely, a leading position of the encoded data of the block line. A syntax element wpp_entry_point_offset[j] indicates an entry point of encoded data of the jth block line. Since a leading position of encoded data of the zeroth block line is the same as the leading position of encoded data of a tile to which the block line belongs, the syntax element wpp_entry_point_offset is omitted. The size of encoded data of the (j−1)th block line has been encoded as the syntax element wpp_entry_point_offset[j]. If the syntax element num_wpp_entry_point_offsets has a value of 0, no mention is made of the syntax elements wpp_entry_point_offset because the syntax elements wpp_entry_point_offset are not needed.

[0046] FIG. **4** illustrates a configuration of the image decoding apparatus for decoding encoded data in the above-described image encoding format.

[0047] In FIG. **4**, a bit stream analyzing unit **401** is configured to analyze a header including a sequence parameter set, a picture parameter set, and a slice header of input encoded data. Decoding units **402** and **403** are configured to decode the input encoded data based on input encoded parameters on a tile-by-tile basis. Although the two tile decoding units are arranged in the present embodiment, the present invention is not limited thereto. A tile image data merging unit **404** is configured to merge image data generated by the first tile decoding unit **402** and image data generated by the second tile decoding unit **403** and output the resultant data.

[0048] In the above-described configuration, the bit stream analyzing unit **401** analyzes the picture parameter set and the

slice header to derive tile data division information, block line data division information, tile data position information, and block line data position information. The bit stream analyzing unit **401** supplies the encoded data following the header to the first tile decoding unit **402** or the second tile decoding unit **403** based on the above-described information. Note that a leading tile is the zeroth tile. If target blocks belong to an even-numbered tile, the encoded data of the tile is supplied to the first tile decoding unit **402**. If the target blocks belong to an odd-numbered tile, the encoded data of the tile is supplied to the second tile decoding unit **403**. In FIG. **2**, the encoded data items of Tile **0** and Tile **2** are decoded by the first tile decoding unit **402** and the encoded data items of Tile **1** and Tile **3** are decoded by the second tile decoding unit **403**.

[0049] The first tile decoding unit **402** and the second tile decoding unit **403** will be described in detail below with reference to FIG. **19**, which is a block diagram of the decoding unit.

[0050] A selector **1901** is configured to determine whether a target block belongs to an even-numbered block line in the tile. The selector **1901** is supplied with encoded data on a tile-by-tile basis. If the block belongs to an even-numbered block line, the selector **1901** outputs the encoded data of the block line to a first block line decoding section **1902**. If the block does not belong to an even-numbered block line, the selector **1901** outputs the encoded data of the block line to a second block line decoding section **1903**. The first and second block line decoding sections **1902** and **1903** are configured to decode input encoded data of a block line on a block-by-block basis. As regards decoding, arithmetic decoding is performed. The arithmetic decoding generates and updates a probability table. A first probability table storage section **1904** is configured to store a probability table, generated by the first block line decoding section **1902**, as decoding information. A second probability table storage section **1905** is configured to store a probability table, generated by the second block line decoding section **1903**, as decoding information. A block line image data merging section **1906** is configured to merge image data generated by the first block line decoding section **1902** and image data generated by the second block line decoding section **1903** and output the resultant image data.

[0051] In FIG. **3**, encoded data of white blocks belonging to even-numbered block lines including the top block line (zeroth line) is decoded by the first block line decoding section **1902** and encoded data of hatched blocks belonging to odd-numbered block lines is decoded by the second block line decoding section **1903**. In other words, a first block line and a second block line are arranged next to each other.

[0052] In each block line decoding section, first, a probability table for a binary signal of encoded data to be decoded is selected and arithmetic decoding is performed based on the probability table to generate quantization coefficients. Subsequently, the quantization coefficients are subjected to inverse quantization based on quantization parameters, thus generating transform coefficients. Then, the transform coefficients are subjected to inverse orthogonal transform, thus generating prediction errors. Lastly, intra-frame prediction with reference to pixels surrounding the target block or inter-frame prediction with reference to another frame is performed, thus generating image data of the target block.

[0053] An operation of the image decoding apparatus according to the present embodiment will be described in detail with reference to a flowchart of FIG. **5**. In the present embodiment, encoded data is supplied on a frame-by-frame basis. Each frame is composed of a plurality of tiles, each tile is divided into blocks, and encoded data is decoded on a block-by-block basis. Although data is input on a frame-by-frame basis in the present embodiment, data may be input on a slice-by-slice basis, the slice being obtained by dividing the frame. Although only intra-prediction decoding will be described in the present embodiment for ease of explanation, decoding is not limited thereto. The present embodiment may be applied to inter-prediction decoding.

[0054] In step S**501**, the bit stream analyzing unit **401** analyzes a header of an input bit stream. In this case, the sequence parameter set, the picture parameter set, and the slice header are analyzed. In particular, the flags TileFlag and WPPFlag in the picture parameter set are analyzed and the tile data position information and the block line data position information are derived from the slice header.

[0055] In step S**502**, a variable CurTile is initialized to 0. The variable CurTile indicates the number of a target tile. In step S**503**, the bit stream analyzing unit **401** determines the amount of transmission data (or the size of encoded data of the target tile) to be transmitted to the first tile decoding unit **402** or the second tile decoding unit **403**.

[0056] A process of determination of the size of tile data in the bit stream analyzing unit **401** will be described with reference to a flowchart of FIG. **9**. Referring to FIG. **9**, in step S**901**, whether the target tile is the last tile in the frame is determined. If the target tile is the last tile (YES in step S**901**), the process proceeds to step S**902**. If NO in step S**901**, the process proceeds to step S**903**.

[0057] In step S**902**, the size of data from the leading end of encoded data of the target tile to the next NAL unit is set as the amount of transmission data. The NAL unit is a container that stores the sequence parameter set, the picture parameter set, an encoded slice and so on. Since a specific bit sequence, for example, 0x000001, is included in the leading end of data of the NAL unit, the leading end of data can be correctly identified. In step S**903**, the value of a syntax element tile_entry_point_offset[CurTile+1] indicating the size of encoded data of the (CurTile)th tile is read and the transmission data amount is determined.

[0058] Referring again to FIG. **5**, in step S**504**, tile decoding is performed in the first tile decoding unit **402** or the second tile decoding unit **403**. In this case, the bit stream analyzing unit **401** outputs encoded data of an even-numbered tile to the first tile decoding unit **402** and outputs encoded data of an odd-numbered tile to the second tile decoding unit **403** such that data as much as the size determined in step S**503** is transmitted. The tile decoding will be described in detail later.

[0059] In step S**505**, the tile image data merging unit **404** merges tile image data output from the first tile decoding unit **402** and tile image data output from the second tile decoding unit **403** to generate decoded image data and outputs the data. In step S**506**, whether decoding of encoded data of all of the tiles in the frame has been completed is determined. If the one frame decoding has been completed (YES in step S**506**), the one frame decoding is terminated. If the decoding has not been completed, the process proceeds to step S**507**. In step S**507**, the variable CurTile is incremented by one. This means that the next tile becomes a target tile.

[0060] The tile decoding in step S**504** will now be described in detail with reference to a flowchart of FIG. **20**. Referring to

FIG. **20**, in step S**2001**, a variable CurBL is initialized to 0. The variable CurBL indicates the number of a target block line.

[0061] In step S**2002**, the amount of transmission data (or the size of encoded data of the target block line) to be transmitted to the first block line decoding section **1902** or the second block line decoding section **1903** by the selector **1901** is determined. A process of determination of the transmission data amount will be described in detail later. In step S**2003**, whether the target block line is the top block line in the tile is determined. If it is the top block line (YES in step S**2003**), the process proceeds to step S**2004**. If NO in step S**2003**, the process proceeds to step S**2005**.

[0062] In step S**2004**, encoded data of the top block line in the tile is decoded to generate image data of the block line. Details of processing in step S**2004** will be described later. In step S**2005**, encoded data of a block line other than the top block line is decoded to generate image data of the block line. As regards block line decoding, whether the target block line is an even-numbered or odd-numbered block line is determined based on the variable CurBL by the selector **1901**. If the target block line is an even-numbered block line, the encoded data of the target block line is decoded by the first block line decoding section **1902**. If the target block line is an odd-numbered block line, the encoded data of the target block line is decoded by the second block line decoding section **1903**. The encoded data items of the block lines are decoded in parallel on a block-line-by-block-line basis by the first and second block line decoding sections **1902** and **1903**. Details of the decoding will also be described later.

[0063] In step S**2006**, the block line image data merging section **1906** merges image data of the block lines output from the first block line decoding section **1902** and image data of the block lines output from the second block line decoding section **1903** to generate tile image data and outputs the data. In step S**2007**, whether decoding of encoded data of all of the block lines in the tile has been completed is determined. If the decoding has been completed (YES in step S**2007**), the tile decoding is terminated. If NO in step S**2007**, the process proceeds to step S**2008**.

[0064] In step S**2008**, the variable CurBL is incremented by one. This means that the next block line in the target tile becomes a target block line. In step S**2002**, the processing or process of determination of the size of block line data in the bit stream analyzing unit **401** will be described in detail with reference to a flowchart of FIG. **21**. In this process, the variable CurBL indicating the number of a target block line and the variable CurTile indicating the number of a target tile are given as inputs.

[0065] In step S**2101**, whether the target block line is the last block line in the tile is determined. If it is the last block line (YES in step S**2101**), the process proceeds to step S**2103**. If NO in step S**2101**, the process proceeds to step S**2102**.

[0066] In step S**2102**, the value of a syntax element wpp_entry_point_offset[CurTile+1][CurBL+1] indicating the size of encoded data of the (CurBL)th block line in the (CurTile)th tile is read and the amount of transmission data is determined. In step S**2103**, whether the target tile is the last tile in the frame is determined. If it is the last tile (YES in step S**2103**), the process proceeds to step S**2104**. If NO in step S**2103**, the process proceeds to step S**2105**.

[0067] In step S**2104**, the size of data from the leading end of encoded data of the target block line to the next NAL unit is set as the amount of transmission data of the last block line in the last tile. Since a specific bit sequence, such as 0x000001, is included in the leading end of data of the NAL unit, the leading end of data can be correctly identified.

[0068] In step S**2105**, the size of data of the last block line in a tile other than the last tile is determined as the amount of transmission data. The size of data of the last block line in the tile is not given as a syntax element wpp_entry_point_offset. The size of data of the last block line is therefore calculated by subtracting the sizes of data of the processed tiles from the value of a syntax element tile_entry_point_offset[CurTile+1] indicating the size of encoded data of the target tile.

[0069] Processing (process of top block line decoding) in step S**2004** in FIG. **20** will be described in detail below with reference to a flowchart of FIG. **6**. Since the top block line is an even-numbered block line, encoded data of the target block line is supplied through the selector **1901** to the first block line decoding section **1902**, in which the data is decoded. In this case, the selector **1901** outputs the encoded data as much as the transmission data amount (or the size of encoded data of the target block line) determined in step S**2002** to the first block line decoding section **1902**.

[0070] Referring to FIG. **6**, in step S**601**, the probability table is initialized by a predetermined method. The initialized probability table is used for arithmetic decoding of the first binary signal in the left block of the block line and is updated at any time in step S**602**, which will be described later. In the following description, the probability table used for arithmetic decoding of a binary signal in the first block of a block line will be referred to as a "block line reference probability table".

[0071] In step S**602**, the encoded data is decoded on a block-by-block basis by the first block line decoding section **1902**, thus generating image data. The block-by-block basis decoding in step S**602** will now be described in detail with reference to a flowchart of FIG. **8**.

[0072] Referring to FIG. **8**, in step S**801**, encoded data is subjected to arithmetic decoding based on the probability table, thus generating a binary signal. Furthermore, the binary signal, binarized by any of various binarization schemes such as unary binarization and fixed length binarization, is decoded for each syntax element in a manner similar to H.264, thus generating a syntax element including a quantization coefficient. In step S**802**, the probability table is updated based on whether the arithmetically decoded binary signal is a symbol with higher probability. In step S**803**, whether all of the syntax elements in the block have been arithmetically decoded is determined. If all of the syntax elements have been arithmetically decoded (YES in step S**803**), the process proceeds to step S**804**. If NO in step S**803**, the process returns to step S**801**. The next syntax element is decoded. In step S**804**, the quantization coefficients are subjected to inverse quantization, thus generating transform coefficients. Furthermore, the transform coefficients are subjected to inverse orthogonal transform, thus generating prediction errors. In step S**805**, intra-prediction is performed based on pixels surrounding the target block, thus generating predicted image data. Furthermore, the prediction errors are added to the predicted image data, thus generating image data on a block-by-block basis.

[0073] Referring again to the flowchart of FIG. **6**, in step S**603**, whether a condition for storing the probability table is satisfied is determined. In the present embodiment, a criterion as to whether a predetermined number of blocks are located between the block decoded in step S**602** and the left end of the

block line is the condition for storing the probability table. If the above condition is satisfied (YES in step S**603**), the process proceeds to step S**604** and the probability table is stored as a first probability table in the first probability table storage section **1904**. If the condition is not satisfied, the process proceeds to step S**605**. The first probability table is used as a block line reference probability table used for decoding of encoded data of the left block in the next block line.

[0074] In step S**605**, whether decoding of encoded data of all of the blocks in the target block line has been completed is determined. If the decoding has been completed (YES in step S**605**), the top block line decoding is terminated. If NO in step S**605**, the process returns to step S**602** and the next block in the raster order is decoded.

[0075] Processing (or process of non-top block line decoding) in step S**2005** in FIG. **20** will be described in detail with reference to a flowchart of FIG. **7**. If the target block line is an even-numbered block line, encoded data of the target block line is supplied to and decoded by the first block line decoding section **1902**. If the target block line is an odd-numbered block line, the encoded data of the target block line is supplied to and decoded by the second block line decoding section **1903**. The selector **1901** outputs the encoded data as much as the transmission data amount (or the size of the encoded data of the target block line) determined in step S**2002** to the first block line decoding section **1902** or the second block line decoding section **1903**. A process of decoding encoded data of an odd-numbered block line in the second block line decoding section **1903** will now be described below.

[0076] Referring to FIG. **7**, in step S**701**, the first probability table is supplied as the block line reference probability table from the first probability table storage section **1904**.

[0077] Since processing in steps S**702** and S**703** is the same as that in steps S**602** and S**603**, explanation is omitted. In step S**704**, the probability table is stored as a second probability table to the second probability table storage section **1905**. The second probability table is used as a block line reference probability table for arithmetic decoding of encoded data of the left block in the next block line. Processing in step S**705** is the same as that in step S**605**. In step S**705**, whether decoding of encoded data of all of the blocks in the odd-numbered block line has been completed is determined. A process of decoding encoded data of an even-numbered block line in the first block line decoding section **1902** will now be described below. In step S**701**, the second probability table is supplied as the block line reference probability table from the second probability table storage section **1905**. Since processing in steps S**702** to S**705** is the same as that in steps S**602** to S**605**, explanation is omitted.

[0078] The above-described configuration and operation enable the leading end of encoded data of each tile and the leading end of encoded data of each block line to be correctly identified. Accordingly, the data can be decoded in parallel on a tile-by-tile basis and the data of each tile can be further decoded in parallel on a block-line-by-block-line basis. Consequently, the division of a frame into tiles and the division of each tile into block lines allow enhancement of both parallelism and encoding efficiency.

[0079] Although the present embodiment has been described with respect to the case where the two tile decoding units and the two block line decoding sections are arranged, it is clear that additional arrangement of, for example, a third tile decoding unit, a third block line decoding section, and a third probability table storage section enables parallel processing by more decoding units and sections.

Second Embodiment

[0080] In an image encoding format to be decoded by an image decoding apparatus according to a second embodiment of the present invention, one frame is divided into a plurality of tiles that are rectangular regions in the same way as in the first embodiment. Since tile arrangement in the second embodiment is the same as that in the first embodiment illustrated in FIG. **2**, explanation is omitted.

[0081] FIG. **10** illustrates a format of HEVC-encoded data in the present embodiment. The encoded data of FIG. **10** is the same as that in the first embodiment, except for the slice header. Accordingly, only the slice header will be described below. In FIG. **10**, when the flag TileFlag has a value of 1 or the flag WPPFlag has a value of 1, the slice header includes, as position information, a syntax element num_entry_point_offsets and syntax elements entry_point_offset.

[0082] If the flag TileFlag has a value of 1 and the flag WPPFlag has a value of 0, this means that the frame has been divided into a plurality of tiles, each tile has not been divided into block lines, and image data of each tile has been encoded. In this case, the syntax element num_entry_point_offsets has a value obtained by subtracting 1 from N that indicates the number of tiles in a slice. A syntax element entry_point_offset[i] indicates an entry point of encoded data of the ith tile. Since it is known that encoded data of the zeroth tile immediately follows the slice header, an entry point of the zeroth tile is omitted. The size of encoded data of the (i−1)th tile has been encoded as the syntax element entry_point_offset[i].

[0083] If the flag TileFlag has a value of 0 and the flag WPPFlag has a value of 1, this means that the frame is composed of a single tile, the tile has been divided into block lines, and image data of each block line has been encoded. In this case, the syntax element num_entry_point_offsets has a value obtained by subtracting 1 from the number of block lines in the slice. The syntax element entry_point_offset[i] indicates an entry point of encoded data of the ith block line. Since it is known that encoded data of the zeroth block line immediately follows the slice header, an entry point of the zeroth block line is omitted. The size of encoded data of the (i−1)th block line has been encoded as the syntax element entry_point_offset[i].

[0084] If the flag TileFlag has a value of 1 and the flag WPPFlag has a value of 1, this means that the frame has been divided into a plurality of tiles, each tile has been divided into block lines, and image data of each block line has been encoded. In this case, the syntax element num_entry_point_offsets has a value obtained by subtracting 1 from the product of the number N of tiles in the slice and the number M of block lines in each tile. The syntax element entry_point_offset[i] indicates an entry point of encoded data of the ith block line. Since it is known that encoded data of the zeroth block line immediately follows the slice header, an entry point of the zeroth block line is omitted. The size of encoded data of the (i−1)th block line has been encoded as the syntax element entry_point_offset[i].

[0085] If the flag TileFlag has a value of 0 and the flag WPPFlag has a value of 0, namely, in the case where the frame is composed of a single tile, the tile has not been divided into block lines, and image data of the tile has been encoded, the syntax element num_entry_point_offsets is treated as having a value of 0.

[0086] In the present embodiment, the flag TileFlag and the flag WPPFlag each have a value of 1. A slice of encoded data in the present embodiment is composed of four tiles. Since the number of vertical blocks in each tile is 135, the syntax element num_entry_point_offsets has a value of 4×135−1=539.

[0087] Encoded data in the image encoding format is decoded. Since the image decoding apparatus according to the present embodiment has the same configuration as that in the first embodiment illustrated in FIGS. **4** and **19**, explanation is omitted.

[0088] An operation of the image decoding apparatus according to the present embodiment will be described in detail with reference to the flowchart of FIG. **5**. Referring to FIG. **5**, in step S**501**, the flags TileFlag and WPPFlag in the picture parameter set are analyzed in the same way as in the first embodiment, thus deriving position information from the slice header. In step S**502**, a variable CurTile is initialized to 0. In step S**503**, the amount of transmission data to be transmitted to the first tile decoding unit **402** or the second tile decoding unit **403** is determined in the bit stream analyzing unit **401** by a different process from that in the first embodiment.

[0089] The process of determining the size of tile data in the present embodiment will be described with reference to a flowchart of FIG. **11**. Since processing in steps S**1101** and S**1102** in FIG. **11** is the same as that in steps S**901** and S**902** in FIG. **9**, explanation is omitted.

[0090] In step S**1103**, the number of entry points in a target tile is determined based on the number of vertical blocks in the tile and the sum of values of syntax elements entry_point_offset is set as the amount of transmission data. In the present embodiment, the number of vertical blocks in the tile is 135, which is fixed. Accordingly, the sum of values of syntax elements entry_point_offset[CurTile*135+1] to entry_point_offset[(CurTile+1)*135] is set as the transmission data amount. In the case where the tiles have different numbers of vertical blocks, the numbers of block lines of the tiles which have been processed are added to determine the position of a syntax element entry_point_offset[] from which addition starts.

[0091] Referring again to FIG. **5**, in step S**504**, tile decoding is performed by the first tile decoding unit **402** or the second tile decoding unit **403** in the same way as in the first embodiment. The tile decoding in step S**504** will be described with reference to the flowchart of FIG. **20** in a manner similar to the first embodiment. The process of FIG. **20** in the present embodiment is the same as that in the first embodiment, except for processing of determination of the size of block line data in step S**2002**. Accordingly, only step S**2002** will be described with reference to FIG. **12** in the present embodiment. Referring to FIG. **12**, in step S**1201**, whether a target block line is the last block line in the frame is determined by the bit stream analyzing unit **401**. If the target block line is the last block line (YES in step S**1201**), the process proceeds to step S**1203**. If NO in step S**1201**, the process proceeds to step S**1202**.

[0092] In step S**1202**, the value of a syntax element entry_point_offset indicating the size of encoded data of the target block line is read to determine the transmission data amount. Thus, the process of determination of the size of block line data is terminated. In the present embodiment, since the number of vertical blocks in each tile is 135, which is fixed, the syntax element entry_point_offset[CurTile×135+CurBL+1] is read. In the case where the tiles have different numbers of vertical blocks, the sum of the numbers of vertical blocks of the tiles processed has to be replaced with a value of CurTile× 135.

[0093] In step S**1203**, the size of data from the leading end of encoded data of the target block line to the next NAL unit is set as the amount of transmission data. The process of determination of the size of block line data is terminated. Since a specific bit sequence, for example, 0x000001, is included in the leading end of data of the NAL unit, the leading end of data can be correctly identified.

[0094] Since processing in steps S**505** to S**507** in FIG. **5** is the same as that in the first embodiment, explanation is omitted.

[0095] The above-described configuration and operation enable the leading end of encoded data of each tile and the leading end of encoded data of each block line to be correctly identified. Accordingly, the data can be decoded in parallel on a tile-by-tile basis and the data in each tile can be further decoded in parallel on a block-line-by-block-line basis. In the first embodiment, it is necessary to provide a syntax element indicating the number of entry points of tiles and syntax elements indicating the entry points of the tiles and further provide syntax element sets equal in number to the tiles, each syntax element set including a syntax element indicating the number of entry points of block lines in the tile and syntax elements indicating the entry points of the block lines. According to the present embodiment, each entry point is represented by the size of encoded data of a block line. Accordingly, it is unnecessary to separately provide information related to the entry points of tiles and information related to the entry points of block lines. Consequently, parallel decoding can be achieved with a smaller amount of information.

[0096] Although the present embodiment has been described with respect to the case where the two tile decoding units and the two block line decoding sections are arranged, it is clear that additional arrangement of, for example, a third tile decoding unit, a third block line decoding section, and a third probability table storage section enables parallel processing by more decoding units and sections.

Third Embodiment

[0097] In an image encoding format for encoding by an image encoding apparatus according to a third embodiment of the present invention, one frame is divided into a plurality of tiles that are rectangular regions in the same way as in the first embodiment. Since tile arrangement in the third embodiment is the same as that in FIG. **2** in the first embodiment, explanation is omitted. Note that the tile arrangement is not limited to that in FIG. **2**.

[0098] Although FIG. **1** illustrates the format of an HEVC-encoded stream in the present embodiment, explanation is omitted because it is the same as that in the first embodiment. FIG. **13** illustrates a configuration of the image encoding apparatus for generating an encoded stream in the image encoding format.

[0099] Referring to FIG. **13**, a tile selector **1301** is configured to determine whether a target block belongs to an even-numbered tile. If the block belongs to an even-numbered tile, the tile selector **1301** outputs data of the block to a first tile encoding unit **1302**. If the block does not belong to an even-numbered tile, the tile selector **1301** outputs the data of the block to a second tile encoding unit **1303**.

[0100] The first and second tile encoding units **1302** and **1303** are configured to encode input image data of blocks on a tile-by-tile basis, each block being composed of n×n pixels (n is a positive integer greater than or equal to 2). Although arrangement of two tile encoding units is described in the present embodiment, the present invention is not limited thereto. In FIG. **2**, image data items of Tile **0** and Tile **2** are encoded by the first tile encoding unit **1302** and image data items of Tile **1** and Tile **3** are encoded by the second tile encoding unit **1303**.

[0101] The first tile encoding unit **1302** and the second tile encoding unit **1303** will be described in detail with reference to a block diagram of FIG. **23**.

[0102] A block line selector **2301** is configured to determine whether a target block belongs to an even-numbered block line in the tile and select an output destination depending on the result of determination. A first block line encoding section **2302** is configured to encode image data of a block belonging to an even-numbered block line. A second block line encoding section **2303** is configured to encode image data of a block belonging to an odd-numbered block line. A first probability table storage section **2304** is configured to store a probability table generated by the first block line encoding section **2302** as encoding information. In the following description, the probability table stored in the first probability table storage section **2304** will be referred to as a first probability table. A second probability table storage section **2305** is configured to store a probability table generated by the second block line encoding section **2303** as encoding information. In the following description, the probability table stored in the second probability table storage section **2305** will be referred to as a second probability table. A block line data merging section **2306** is configured to merge encoded data generated by the first block line encoding section **2302** and encoded data generated by the second block line encoding section **2303** and output the merged data together with encoding parameters including position information.

[0103] The block line selector **2301** is supplied with image data on a tile-by-tile basis. If a target block belongs to an even-numbered block line, the block line selector **2301** outputs data of the block to the first block line encoding section **2302**. If the target block does not belong to an even-numbered block line, the block line selector **2301** outputs the data of the block to the second block line encoding section **2303**. Referring to FIG. **3**, image data of the white blocks indicating the even-numbered block lines is encoded by the first block line encoding section **2302** and image data of the hatched blocks indicating the odd-numbered block lines is encoded by the second block line encoding section **2303**. Since the top block line is the zeroth line, image data of the top block line is encoded by the first block line encoding section **2302**. A process of encoding using the first block line encoding section **2302** and the second block line encoding section **2303** will be described in detail later.

[0104] Referring again to FIG. **13**, a merging unit **1304** is configured to merge encoded data and encoding parameters generated by the first tile encoding unit **1302** and encoded data and encoding parameters generated by the second tile encoding unit **1303** into a bit stream and output the bit stream. The merging unit **1304** merges division information, such as the flags TileFlag and WPPFlag, into a picture parameter set and merge position information into a slice header and outputs the bit stream.

[0105] An operation of the image encoding apparatus according to the present embodiment will be described in detail with reference to flowcharts of FIGS. **14** to **17**. In the present embodiment, images are input on a frame-by-frame basis. Each frame is composed of a plurality of tiles, each tile is divided into a plurality of blocks, and image data is encoded on a block-by-block basis. Although the images are input on a frame-by-frame basis in the present embodiment, the images may be input on a slice-by-slice basis, the slices being obtained by dividing the frame. Furthermore, although only intra-prediction encoding will be described in the present embodiment for ease of explanation, encoding is not limited thereto. The present embodiment may be applied to inter-prediction encoding.

[0106] One frame encoding will be described with reference to FIG. **14**. In step S**1401**, a variable CurTile indicating the number of a target tile is initialized to 0. In step S**1402**, tile encoding is performed in the first tile encoding unit **1302** or the second tile encoding unit **1303**. In this case, the tile selector **1301** transmits data of an even-numbered tile to the first tile encoding unit **1302** and transmits data of an odd-numbered tile to the second tile encoding unit **1303**. The tile encoding will be described in detail with reference to a flowchart of FIG. **24**.

[0107] Referring to FIG. **24**, in step S**2401**, a variable CurBL indicating the number of a target block line is initialized to 0. In step S**2402**, whether the target block line is the top block line in the tile is determined. If the target block line is the top block line (YES in step S**2402**), the process proceeds to step S**2403**. If NO in step S**2402**, the process proceeds to step S**2404**.

[0108] In step S**2403**, the top block line in the tile is encoded. In step S**2404**, data of a block line other than the top block line is encoded. Encoding in these steps will be described in detail later. In step S**2405**, position information (syntax elements tile_entry_point_offset and syntax elements wpp_entry_point_offset) to be contained in the slice header is set. Processing in this step will also be described in detail later.

[0109] In step S**2406**, encoded data output from the first block line encoding section **2302** and encoded data output from the second block line encoding section **2303** are merged by the block line data merging section **2306**. In addition, the merged encoded data is merged with encoding parameters including the position information calculated in step S**2405** and the resultant encoded data is output.

[0110] In step S**2407**, whether encoding of data of all of the block lines in the tile has been completed is determined. If the encoding has been completed (YES in step S**2407**), the tile encoding is terminated. If NO in step S**2407**, the process proceeds to step S**2408**.

[0111] In step S**2408**, the variable CurBL is incremented by one. This means that the next block line in the target tile becomes a target block line.

[0112] Referring again to FIG. **14**, in step S**1403**, whether encoding of data of all of the tiles in the frame has been completed is determined. If the encoding has been completed (YES in step S**1403**), the process proceeds to step S**1405**. If NO in step S**1403**, the process proceeds to step S**1404**. In step S**1404**, the variable CurTile indicating the tile number is incremented by one. This means that the next tile becomes a target tile. In step S**1405**, encoded data output from the first tile encoding unit **1302** and encoded data output from the second tile encoding unit **1303** are merged by the merging

unit **1304**. In addition, the merged encoded data and encoding parameters including the position information calculated in step S**2405** are merged into a bit stream and the bit stream is output.

[0113] The block line encodings included in the process of tile encoding in FIG. **24** will be described in detail below. First, the processing (or process of top block line encoding)in step S**2403** will be described in detail with reference to a flowchart of FIG. **15**. Since the top block line is an even-numbered block line, image data of the target block line is supplied through the tile selector **1301** to the first tile encoding unit **1302** and the image data is encoded by the first tile encoding unit **1302**.

[0114] Referring to FIG. **15**, in step S**1501**, the probability table is initialized by a predetermined method. The initialized probability table is used for arithmetic encoding of a first binary signal of the left block in the block line and is updated at any time in step S**1502**, which will be described later. In the following description, the probability table used for arithmetic encoding of a binary signal of the first block in the block line will be referred to as a block line reference probability table.

[0115] In step S**1502**, image data is encoded on a block-by-block basis by the first tile encoding unit **1302**, thus generating encoded data. Block encoding in step S**1502** will be described in detail with reference to a flowchart of FIG. **16**.

[0116] Referring to FIG. **16**, in step S**1601**, intra-prediction using pixels surrounding the block is performed on input image data of the block, thus generating prediction errors. In step S**1602**, the prediction errors are subjected to orthogonal transform, thus generating transform coefficients. Furthermore, the transform coefficients are quantized using quantization parameters determined based on, for example, image characteristics and the amount of symbols, thus generating quantization coefficients. In step S**1603**, a syntax element indicating, for example, a quantization parameter or a prediction mode, is binarized, thus generating a binary signal. Various binarization schemes, such as unary binarization and fixed length binarization, are used while being switched for each syntax element in a manner similar to H.264. Furthermore, the binary signal is subjected to arithmetic encoding based on the probability table. In step S**1604**, the probability table is updated based on whether the arithmetically encoded binary signal is a symbol with higher possibility. In step S**1605**, whether all of the syntax elements in the block have been arithmetically encoded is determined. If all of the syntax elements have been arithmetically encoded (YES in step S**1605**), the block encoding is terminated. If NO in step S**1605**, the process returns to step S**1603**, in which the next syntax element is encoded.

[0117] Referring again to the flowchart of FIG. **15**, in step S**1503**, whether a condition for storing the probability table is satisfied is determined. In the present embodiment, a criterion as to whether a predetermined number of blocks are located between the block encoded in step S**1502** and the left end of the block line is the condition for storing the probability table. If the above condition is satisfied (YES in step S**1503**), the process proceeds to step S**1504**. The probability table is stored as a first probability table to the first probability table storage section **2304**. If the condition is not satisfied, the process proceeds to step S**1505**. The first probability table is used as the block line reference probability table for encoding of image data of the left block in the next block line.

[0118] In step S**1505**, whether encoding of data of all of the blocks in the target block line has been completed is determined. If the encoding has been completed (YES in step S**1505**), the top block line encoding is terminated. If NO in step S**1505**, the process returns to step S**1502** and image data of the next block in the raster order is encoded.

[0119] Processing (or process of non-top block line encoding) in step S**2404** in FIG. **24** will be described in detail with reference to a flowchart of FIG. **17**. If a target block line is an even-numbered block line, image data of the target block line is supplied to and encoded by the first block line encoding section **2302**. If the target block line is an odd-numbered block line, the image data of the target block line is supplied to and encoded by the second block line encoding section **2303**. First, a process of encoding image data of an odd-numbered block line in the second block line encoding section **2303** will now be described below.

[0120] In step S**1701**, the first probability table is supplied as the block line reference probability table from the first probability table storage section **2304**. Since processing in steps S**1702** and S**1703** is the same as that in steps S**1502** and S**1503**, explanation is omitted. In step S**1704**, the probability table is stored as a second probability table to the second probability table storage section **2305**. The second probability table is used as a block line reference probability table for arithmetic encoding of image data of the left block in the next block line. Processing in step S**1705** is the same as that in step S**1505**. In step S**1705**, whether encoding of image data of all of the blocks in the odd-numbered block line has been completed is determined.

[0121] Next, a process of encoding image data of an even-numbered block line in the first block line encoding section **2302** will be described below. In step S**1701**, the second probability table is supplied as the block line reference probability table from the second probability table storage section **2305**. Since processing in steps S**1702** to S**1704** is the same as that in steps S**1502** to S**1504**, explanation is omitted. In step S**1705**, whether encoding of image data of all of the blocks in the even-numbered block line has been completed is determined.

[0122] Processing or process of calculating position information in step S**2405** in FIG. **24** will be described in detail with reference to a flowchart of FIG. **18**. In this process, the variable CurBL and the variable CurTile are given as inputs.

[0123] Referring to FIG. **18**, in step S**1801**, whether a target block line is the last block line in a target tile is determined. If it is the last block line (YES in step S**1801**), the process proceeds to step S**1803**. If NO in step S**1801**, the process proceeds to step S**1802**.

[0124] In step S**1802**, the size of encoded data of the target block line is set as a syntax element wpp_entry_point_offset [CurTile+1][CurBL+1]. A syntax element wpp_entry_point_offset[CurTile][CurBL] indicates the size of encoded data of the (CurBL)th block line in the (CurTile)th tile.

[0125] In step S**1803**, whether the target tile is the last tile in the frame is determined. If it is the last tile, the process of calculating position information is terminated. If the target tile is not the last tile, the process proceeds to step S**1804**. If the target tile is the last tile, this means that the target block line is the last block line in the last tile. Information about the entry point of the block line is not encoded as a syntax element.

[0126] In step S**1804**, the size of encoded data of the target tile is set as a syntax element tile_entry_point_offset

[CurTile+1] that indicates the size of encoded data of the (CurTile)th tile. Values of syntax elements wpp_entry_point_offset[] and syntax elements tile_entry_point_offset[] are merged into the slice header in step S**1405** in FIG. **14**.

[0127] The above-described configuration and operation enable division of an image into tiles and parallel encoding on a block-line-by-block-line basis. Accordingly, high-speed encoding can be achieved. Furthermore, since the leading end of encoded data of each tile and the leading end of encoded data of each block line can be correctly identified, a bit stream capable of being decoded in parallel on a tile-by-tile basis and being decoded in parallel on a block-line-by-block-line basis in each tile can be encoded. Thus, since a frame can be divided into tiles and each tile can be divided into block lines, both parallelism and encoding efficiency can be enhanced.

[0128] Although the present embodiment has been described with respect to the case where the two tile encoding units and the two block line encoding sections are arranged, it is clear that additional arrangement of, for example, a third tile encoding unit, a third block line encoding section, and a third probability table storage section enables parallel processing by more encoding units and sections.

Fourth Embodiment

[0129] In an image encoding format for encoding by an image encoding apparatus according to a fourth embodiment of the present invention, one frame is divided into a plurality of tiles that are rectangular regions in the same way as in the first embodiment. Since tile arrangement in the fourth embodiment is the same as that in the first embodiment illustrated in FIG. **2**, explanation is omitted. Note that the tile arrangement is not limited to this arrangement.

[0130] The format of an HEVC-encoded stream in the present embodiment is illustrated in

[0131] FIG. **10**. The encoded stream in FIG. **10** is the same as that in the second embodiment. All of the entry points are indicated by entry points of block lines.

[0132] FIGS. **13** and **23** illustrate a configuration (components) of the image encoding apparatus according to the present embodiment, the image encoding apparatus generating an encoded stream in the image encoding format according to the present embodiment. Since the image encoding apparatus according to the present embodiment has the same configuration as that in the third embodiment, explanation is omitted.

[0133] An operation of the image encoding apparatus according to the present embodiment is the same as that in the third embodiment, except for the processing of calculating position information in step S**2405** in FIG. **24**. Accordingly, only processing in step S**2405** will be described with reference to FIG. **22**.

[0134] Referring to FIG. **22**, in step S**2201**, whether a target block line is the last block line in a frame is determined. If it is the last block line (YES in step S**2201**), the process of calculating position information is terminated. If the target block line is not the last block line, the process proceeds to step S**2202**. If the target block line is the last block line, information about an entry point of the block line is not encoded as a syntax element.

[0135] In step S**2202**, the size of encoded data of the target block line is set as a syntax element entry_point_offset. In the present embodiment, since the number of vertical blocks in each tile is 135 which is fixed, the size of encoded data of the (CurBL)th block line in the (CurTile)th tile is set as a syntax element entry_point_offset[CurTile×135+CurBL+1]. If the tiles have different numbers of vertical blocks, it is necessary to replace the sum of the numbers of vertical blocks in the tiles processed with a value of CurTile×135.

[0136] The position information entry_point_offset obtained in step S**2405** in FIG. **24** is merged as a syntax element into a slice header in step S**1405** in the same way as in the third embodiment.

[0137] The above-described configuration and operation enable division of an image into tiles and parallel encoding on a block-line-by-block-line basis. Accordingly, high-speed encoding can be achieved. Furthermore, since the leading end of encoded data of each tile and the leading end of encoded data of each block line can be correctly identified, a bit stream capable of being decoded in parallel on a tile-by-tile basis and being decoded in parallel on a block-line-by-block-line basis in each tile can be encoded.

[0138] In the third embodiment, it is necessary to provide a syntax element indicating the number of entry points of tiles and syntax elements indicating the entry points of the tiles and further provide syntax element sets equal in number to the tiles, each syntax element set including a syntax element indicating the number of entry points of block lines in each tile and syntax elements indicating the entry points of the block lines. According to the present embodiment, each entry point is represented by the size of encoded data of a block line. Accordingly, it is unnecessary to separately provide information related to the entry points of the tiles and information related to the entry points of the block lines. Consequently, a stream capable of being subjected to parallel decoding with a smaller amount of information can be encoded.

[0139] Although the present embodiment has been described with respect to the case where the two tile encoding units and the two block line encoding sections are arranged, it is clear that additional arrangement of, for example, a third tile encoding unit, a third block line encoding section, and a third probability table storage section enables parallel processing by more encoding units and sections.

Fifth Embodiment

[0140] The above-described embodiments have been described on the assumption that the processing units and sections illustrated in FIGS. **4**, **13**, **19**, and **23** are constructed by hardware. Processes performed by the processing units and sections in FIGS. **4**, **13**, **19**, and **23** may be implemented by computer programs.

[0141] FIG. **25** is a block diagram illustrating an exemplary configuration of hardware of a computer applicable to the image processing (decoding and encoding) apparatuses according to the above-described embodiments.

[0142] A CPU **2501** is configured to perform overall control of the computer based on computer programs and data stored in a RAM **2502** and a ROM **2503** and execute the above-described processes, performed by the image processing apparatuses according to the embodiments. In other words, the CPU **2501** functions as the processing units and sections illustrated in FIGS. **4**, **13**, **19**, and **23**.

[0143] The RAM **2502** has an area for temporarily storing a computer program and data loaded from an external storage **2506** and data obtained through an interface (UF) **2507** from an external device. The RAM **2502** further has a work area used for execution of various processes by the CPU **2501**. In other words, the RAM **2502** can function as a frame memory or appropriately provide various areas.

[0144] The ROM **2503** stores setting data of the computer, a boot program, and the like. An operation unit **2504** includes a keyboard and a mouse. When operated by a user of the computer, the operation unit **2504** can input various instructions to the CPU **2501**. An output unit **2505** outputs a result of processing by the CPU **2501**. The output unit **2505** can be constructed by a display device, e.g., a liquid crystal display, such that the result of processing can be displayed.

[0145] The external storage **2506** functions as a mass storage, such as a hard disk drive. The external storage **2506** stores an operating system (OS) and computer programs for allowing the CPU **2501** to achieve functions of the units and sections illustrated in FIGS. **4**, **13**, **19**, and **23**. The external storage **2506** may further store images to be processed.

[0146] The computer programs and data stored in the external storage **2506** are appropriately loaded to the RAM **2502** under the control of the CPU **2501** and serve as targets to be processed by the CPU **2501**. The I/F **2507** can be connected to networks, such as a local area network (LAN) and the Internet, and other devices, such as a projector and a display device. The computer can receive and transmit various pieces of information through the I/F **2507**. A bus **2508** connects the above-described components.

[0147] The operations described with reference to the foregoing flowcharts are controlled mainly by the CPU **2501**.

Other Embodiments

[0148] Aspects of the present invention can also be realized by a computer of a system or apparatus (or devices such as a CPU or MPU) that reads out and executes a program recorded on a memory device to perform the functions of the above-described embodiment(s), and by a method, the steps of which are performed by a computer of a system or apparatus by, for example, reading out and executing a program recorded on a memory device to perform the functions of the above-described embodiment(s). For this purpose, the program is provided to the computer for example via a network or from a recording medium of various types serving as the memory device (e.g., computer-readable medium).

[0149] While the present invention has been described with reference to exemplary embodiments, it is to be understood that the invention is not limited to the disclosed exemplary embodiments. The scope of the following claims is to be accorded the broadest interpretation so as to encompass all such modifications and equivalent structures and functions.

[0150] This application claims the benefit of Japanese Patent Application No. 2012-147153, filed Jun. 29, 2012, which is hereby incorporated by reference herein in its entirety.

1. An image decoding apparatus that decodes encoded data obtained by dividing a frame of image data into rectangular tiles and encoding the image data on a tile-by-tile basis, the apparatus comprising:

a deriving unit configured to derive tile data division information, tile data position information, block line data division information, and block line data position information from the encoded data, the tile data division information indicating whether the encoded data is composed of tile data items that serve as encoded data items of the tiles, the tile data position information indicating positions of the tile data items, the block line data division information indicating whether each tile data item is composed of first block line data and second block line data, the first block line data serving as encoded data of a first block line that is a set of blocks arranged linearly, the second block line data serving as encoded data of a second block line next to the first block line, the block line data position information indicating a position of the second block line data;

a first decoding unit configured to decode the first block line data based on the information derived by the deriving unit; and

a second decoding unit configured to decode the second block line data based on the information derived by the deriving unit and decoding information obtained by decoding encoded data of a block in a predetermined position belonging to the first block line.

2. The apparatus according to claim **1**, wherein the position of data of the zeroth block line belonging to the Nth (N is greater than or equal to 1) tile in the frame is indicated by the Nth tile data position information and positions of data items of the Mth (M is greater than or equal to 1) block line and subsequent block lines are indicated as block line data position information by the size of encoded data of the (M−1)th block line.

3. An image decoding apparatus that decodes encoded data obtained by dividing a frame of image data into rectangular tiles and encoding the image data on a tile-by-tile basis, the apparatus comprising:

a deriving unit configured to derive tile data division information, block line data division information, and size information from the encoded data, the tile data division information indicating whether the encoded data is composed of first tile data and second tile data, the first and the second tile data serving as encoded data of a corresponding tile, the block line data division information indicating whether each tile data item is composed of first block line data and second block line data, the first block line data serving as encoded data of a first block line that is a set of blocks arranged linearly, the second block line data serving as encoded data of a second block line next to the first block line, the size information indicating a size of the first tile data or the first block line data;

a first decoding unit configured to decode the first block line data based on the information derived by the deriving unit;

a second decoding unit configured to decode the second block line data based on the information derived by the deriving unit and decoding information obtained by decoding encoded data of a block in a predetermined position belonging to the first block line; and if the encoded data is encoded with tile data division and the tile data is encoded without block line data division, the size information indicates the first tile data size, otherwise, the size information indicates the first block line data size.

4. The apparatus according to claim **3**, wherein if the encoded data is encoded with tile data division and the tile data is encoded with block line division, the position of Mth (M is greater than or equal to 1) block line data from 0th block line data is derived from sum of information from 0th to (M−1)th block line data size and the position of Nth (N is greater than or equal to 1) tile data from 0th tile data is derived from sum of block data size information of corresponding block data in 0th to (N−1)th tile data.

5. A method of image decoding for an image decoding apparatus that decodes encoded data obtained by dividing a

frame of image data into rectangular tiles and encoding the image data on a tile-by-tile basis, the method comprising the steps of:

deriving tile data division information, tile data position information, block line data division information, and block line data position information from the encoded data, the tile data division information indicating whether the encoded data is composed of tile data items that serve as encoded data items of the tiles, the tile data position information indicating positions of the tile data items, the block line data division information indicating whether each tile data item is composed of first block line data and second block line data, the first block line data serving as encoded data of a first block line that is a set of blocks arranged linearly, the second block line data serving as encoded data of a second block line next to the first block line, the block line data position information indicating a position of the second block line data;

decoding the first block line data based on the derived information; and

decoding the second block line data based on the derived information and decoding information obtained by decoding encoded data of a block in a predetermined position belonging to the first block line.

6. A recording medium storing a program that allows a computer to function as the image decoding apparatus according to claim **1** when read and executed by the computer.

* * * * *