(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2018年3月1日(01.03.2018)

WIPO | PCT

(10) 国際公開番号
WO 2018/037853 A1

(51) 国際特許分類:
H04N 19/119 (2014.01) *H04N 19/176* (2014.01)
H04N 19/136 (2014.01) *H04N 19/70* (2014.01)
H04N 19/157 (2014.01)

(21) 国際出願番号： PCT/JP2017/027877

(22) 国際出願日： 2017年8月1日(01.08.2017)

(25) 国際出願の言語： 日本語

(26) 国際公開の言語： 日本語

(30) 優先権データ：
特願 2016-166319 2016年8月26日(26.08.2016) JP
特願 2016-253890 2016年12月27日(27.12.2016) JP

(71) 出願人:シャープ株式会社(**SHARP KABUSHIKI KAISHA**) [JP/JP]; 〒5908522 大阪府堺市堺区匠町1番地 Osaka (JP).

(72) 発明者:山本 貴也(**YAMAMOTO, Yoshiya**). 八杉 将伸(**YASUGI, Yukinobu**). 猪飼 知宏(**IKAI, Tomohiro**). 青野 友子(**AONO, Tomoko**).

(74) 代理人:特許業務法人HARAKENZO WORLD PATENT & TRADEMARK (**HARAKENZO WORLD PATENT & TRADEMARK**); 〒5300041 大阪府大阪市北区天神橋2丁目北2番6号 大和南森町ビル Osaka (JP).

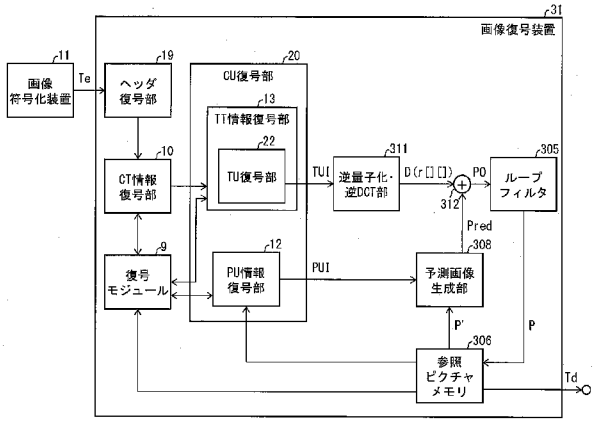
(81) 指定国(表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国(表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LR, LS,

(54) **Title:** IMAGE DECODING APPARATUS AND IMAGE CODING APPARATUS

(54) 発明の名称： 画像復号装置及び画像符号化装置

図 10



9 Decoding module
10 CT information decoding unit
11 Image coding apparatus
12 PU information decoding unit
13 TT information decoding unit
19 Header decoding unit
20 CU decoding unit
22 TU decoding unit
31 Image decoding apparatus
305 Loop filter
306 Reference picture memory
308 Prediction image generation unit
311 Inverse quantization/inverse DCT unit

(57) **Abstract:** An image decoding apparatus (31) is provided with a CT information decoding unit (11) that decodes division information indicating a division method for binary tree division of a coding node of a coding tree unit, and carries out the binary tree division of the coding node by referring to the division information about another decoded coding node.

(57) 要約：画像復号装置（31）は、符号化ツリーユニットの符号化ノードに対する二分木分割の分割方法を示す分割情報を復号し、符号化ノードに対して、復号済の他の符号化ノードの上記分割情報を参照して、二分木分割を行うCT情報復号部（11）を備える。

WO 2018/037853 A1

[続葉有]

MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

添付公開書類：

一　国際調査報告（条約第21条(3)）

明　細　書

発明の名称 ： 画像復号装置及び画像符号化装置

技術分野

[0001] 本発明は、予測画像生成装置、画像復号装置、及び画像符号化装置に関する。

背景技術

[0002] 動画像を効率的に伝送または記録するために、動画像を符号化することによって符号化データを生成する画像符号化装置、及び、当該符号化データを復号することによって復号画像を生成する画像復号装置が用いられている。

[0003] 具体的な動画像符号化方式としては、例えば、H.264/AVCやHEVC（High-Efficiency Video Coding）にて提案されている方式などが挙げられる。

[0004] このような動画像符号化方式においては、動画像を構成する画像（ピクチャ）は、画像を分割することにより得られるスライス、スライスを分割することにより得られる符号化単位（符号化ユニット（Coding Unit：CU）と呼ばれることもある）、及び、符号化単位を分割することより得られるブロックである予測ユニット（PU）、変換ユニット（TU）からなる階層構造により管理され、CUごとに符号化／復号される。

[0005] また、このような動画像符号化方式においては、通常、入力画像を符号化／復号することによって得られる局所復号画像に基づいて予測画像が生成され、当該予測画像を入力画像（原画像）から減算して得られる予測残差（「差分画像」または「残差画像」と呼ぶこともある）が符号化される。予測画像の生成方法としては、画面間予測（インター予測）、及び、画面内予測（イントラ予測）が挙げられる。

[0006] また、近年の動画像符号化及び復号の技術として非特許文献１が挙げられる。

[0007] さらに、近年では、スライスを構成する符号化ツリーユニット（CTU:CodingTree Unit）から符号化ユニット（CU:Coding Unit）への分割方式として、

４分木（quad tree）分割するQT分割に加えて、２分木（binary tree）分割するBT分割が導入されている。このBT分割には、水平分割と垂直分割とが含まれる。

先行技術文献

非特許文献

[0008] 非特許文献1：”AlgorithmDescription of Joint Exploration Test Model 2”, JVET-B1002, Joint VideoExploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC1/SC 29/WG 11, 20-26February 2016

発明の概要

発明が解決しようとする課題

[0009] CTUの分割方式として、QT分割に加えてBT分割が導入されたことにより、CUへの分割パターンが増大し、符号化／復号が複雑となる。

[0010] そこで、本発明は、上記の課題に鑑みてなされたものであり、その目的は、動画像の符号化／復号の複雑度を低減することができる画像復号装置及び画像符号化装置を提供することにある。

課題を解決するための手段

[0011] 本発明の一態様に係る画像復号装置は、上記の課題を解決するために、ピクチャを符号化ツリーユニット毎に復号する画像復号装置において、上記符号化ツリーユニットの符号化ノードに対する二分木分割の分割方法を示す分割情報を復号する分割情報復号部と、符号化ノードに対して、復号済の他の符号化ノードの上記分割情報を参照して、二分木分割を行う分割部と、を備える。

[0012] また、本発明の一態様に係る画像符号化装置は、上記の課題を解決するために、ピクチャを符号化ツリーユニットに分割して符号化する画像符号化装置において、上記符号化ツリーユニットの符号化ノードに対する二分木分割の分割方法を示す分割情報を符号化する分割情報符号化部と、符号化ノードに対して、符号化済の他の符号化ノードの上記分割情報を参照して、二分木

分割を行う分割部と、を備える。

発明の効果

[0013] 本発明の一態様によれば、動画像の符号化／復号の複雑度を低減することができる。

図面の簡単な説明

[0014] [図1]本発明の一実施形態に係る画像伝送システムの構成を示す概略図である。

[図2]本発明の一実施形態に係る符号化ストリームのデータの階層構造を示す図である。

[図3]PU分割モードのパターンを示す図である。（ａ）～（ｈ）は、それぞれ、PU分割モードが、2Nx2N、2NxN、2NxnU、2NxnD、Nx2N、nLx2N、nRx2N、及び、NxNの場合のパーティション形状について示している。

[図4]参照ピクチャ及び参照ピクチャリストの一例を示す概念図である。

[図5]本発明の一実施形態に係る画像符号化装置の構成を示すブロック図である。

[図6]本発明の一実施形態に係る画像復号装置の構成を示す概略図である。

[図7]本発明の一実施形態に係る画像符号化装置のインター予測画像生成部の構成を示す概略図である。

[図8]本発明の一実施形態に係る画像符号化装置を搭載した送信装置、及び、画像復号装置を搭載した受信装置の構成について示した図である。（ａ）は、画像符号化装置を搭載した送信装置を示しており、（ｂ）は、画像復号装置を搭載した受信装置を示している。

[図9]本発明の一実施形態に係る画像符号化装置を搭載した記録装置、及び、画像復号装置を搭載した再生装置の構成について示した図である。（ａ）は、画像符号化装置を搭載した記録装置を示しており、（ｂ）は、画像復号装置を搭載した再生装置を示している。

[図10]本発明の一実施形態に係る画像復号装置の構成を示すブロック図である。

[図11]本発明の一実施形態に係るCT情報復号部の動作を説明するフローチャートである。

[図12]本発明の一実施形態に係るQT情報のシンタックス表の構成図である。

[図13]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図14]BT分割の分割方法の一例を示す図である。

[図15]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図16]BT分割モードの導出テーブルを示す図である。

[図17]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図18]BT分割モードの導出テーブルを示す図である。

[図19]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図20]BT分割モードの導出テーブルを示す図である。

[図21]BT分割の分割方法の一例を示す図である。

[図22]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図23]BT分割モードの導出テーブルを示す図である。

[図24]BT分割の分割方法の一例を示す図である。

[図25]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図26]BT分割の分割方法の一例を示す図である。

[図27]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図28]BT分割の分割方法の一例を示す図である。

[図29]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図30]BT分割の分割方法の一例を示す図である。

[図31]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図32]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図33]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図34]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図35]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図36]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図37]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図38]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図39]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図40]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図41]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図42]BTツリーにおけるTemporal IDごとの最大BT深度の設定例を示す図である。

[図43]SPSのためのシンタックス構造図である。

[図44]本発明の一実施形態に係るCT情報復号部の動作を説明するフローチャートである。

[図45]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図46]本発明の一実施形態に係るQT情報のシンタックス表の構成図である。

[図47]本発明の一実施形態に係るBT情報のシンタックス表の構成図である。

[図48]本発明の一実施形態に係るQT情報のシンタックス表の構成図である。

発明を実施するための形態

[0015] （第１の実施形態）

以下、図面を参照しながら本発明の実施形態について説明する。

[0016] 図１は、本実施形態に係る画像伝送システム１の構成を示す概略図である。

[0017] 画像伝送システム１は、符号化対象画像を符号化した符号を伝送し、伝送された符号を復号し画像を表示するシステムである。画像伝送システム１は、画像符号化装置１１、ネットワーク２１、画像復号装置３１及び画像表示装置４１を含んで構成される。

[0018] 画像符号化装置１１には、単一レイヤもしくは複数レイヤの画像を示す画像Ｔが入力される。レイヤとは、ある時間を構成するピクチャが１つ以上ある場合に、複数のピクチャを区別するために用いられる概念である。たとえば、同一ピクチャを、画質や解像度の異なる複数のレイヤで符号化するとスケーラブル符号化になり、異なる視点のピクチャを複数のレイヤで符号化するとビュースケーラブル符号化となる。複数のレイヤのピクチャ間で予測（

インターレイヤ予測、インタービュー予測）を行う場合には、符号化効率が大きく向上する。また予測を行わない場合（サイマルキャスト）の場合にも、符号化データをまとめることができる。

[0019] ネットワーク２１は、画像符号化装置１１が生成した符号化ストリームＴｅを画像復号装置３１に伝送する。ネットワーク２１は、インターネット（internet）、広域ネットワーク（WAN:Wide Area Network）、小規模ネットワーク（LAN:Local Area Network）またはこれらの組み合わせである。ネットワーク２１は、必ずしも双方向の通信網に限らず、地上デジタル放送、衛星放送等の放送波を伝送する一方向の通信網であっても良い。また、ネットワーク２１は、DVD（Digital Versatile Disc）、BD（Blue-ray Disc）等の符号化ストリームTeを記録した記憶媒体で代替されても良い。

[0020] 画像復号装置３１は、ネットワーク２１が伝送した符号化ストリームTeのそれぞれを復号し、それぞれ復号した１または複数の復号画像Tdを生成する。

[0021] 画像表示装置４１は、画像復号装置３１が生成した１または複数の復号画像Tdの全部または一部を表示する。画像表示装置４１は、例えば、液晶ディスプレイ、有機ＥＬ（Electro-luminescence）ディスプレイ等の表示デバイスを備える。また、空間スケーラブル符号化、ＳＮＲスケーラブル符号化では、画像復号装置３１、画像表示装置４１が高い処理能力を有する場合には、画質の高い拡張レイヤ画像を表示し、より低い処理能力しか有しない場合には、拡張レイヤほど高い処理能力、表示能力を必要としないベースレイヤ画像を表示する。

[0022] ＜演算子＞

本明細書で用いる演算子を以下に記載する。

[0023] >>は右ビットシフト、<<は左ビットシフト、&はビットワイズAND、|はビットワイズOR、|=は別の条件との和演算（OR）である。

[0024] x ? y : zは、xが真（0以外）の場合にy、xが偽（0）の場合にzをとる３項演算子である。

[0025] Clip3(a, b, c）は、cをa以上b以下の値にクリップする関数であり、c<aの場合にはaを返し、c>bの場合にはbを返し、その他の場合にはcを返す関数である（ただし、a<=b）。

[0026] ＜符号化ストリームTeの構造＞

本実施形態に係る画像符号化装置１１及び画像復号装置３１の詳細な説明に先立って、画像符号化装置１１によって生成され、画像復号装置３１によって復号される符号化ストリームTeのデータ構造について説明する。

[0027] 図２は、符号化ストリームTeにおけるデータの階層構造を示す図である。符号化ストリームTeは、例示的に、シーケンス、及びシーケンスを構成する複数のピクチャを含む。図２の（ａ）～（ｆ）は、それぞれ、シーケンスSEQを既定する符号化ビデオシーケンス、ピクチャPICTを規定する符号化ピクチャ、スライスSを規定する符号化スライス、スライスデータを規定する符号化スライスデータ、符号化スライスデータに含まれる符号化ツリーユニット、符号化ツリーユニットに含まれる符号化ユニット（Coding Unit；CU）を示す図である。

[0028] （符号化ビデオシーケンス）

符号化ビデオシーケンスでは、処理対象のシーケンスSEQを復号するために画像復号装置３１が参照するデータの集合が規定されている。シーケンスSEQは、図２の（ａ）に示すように、ビデオパラメータセット（Video Parameter Set）、シーケンスパラメータセットSPS（Sequence Parameter Set）、ピクチャパラメータセットPPS（Picture Parameter Set）、ピクチャPICT、及び、付加拡張情報SEI（Supplemental Enhancement Information）を含んでいる。ここで＃の後に示される値はレイヤIDを示す。図２では、#0と#1すなわちレイヤ０とレイヤ１の符号化データが存在する例を示すが、レイヤの種類及びレイヤの数はこれによらない。

[0029] ビデオパラメータセットVPSは、複数のレイヤから構成されている動画像において、複数の動画像に共通する符号化パラメータの集合及び動画像に含まれる複数のレイヤ及び個々のレイヤに関連する符号化パラメータの集合が規

定されている。

[0030] シーケンスパラメータセットSPSでは、対象シーケンスを復号するために画像復号装置３１が参照する符号化パラメータの集合が規定されている。例えば、ピクチャの幅や高さが規定される。なお、SPSは複数存在してもよい。その場合、PPSから複数のSPSの何れかを選択する。

[0031] ピクチャパラメータセットPPSでは、対象シーケンス内の各ピクチャを復号するために画像復号装置３１が参照する符号化パラメータの集合が規定されている。例えば、ピクチャの復号に用いられる量子化幅の基準値（pic_init_qp_minus26）や重み付き予測の適用を示すフラグ（weighted_pred_flag）が含まれる。なお、PPSは複数存在してもよい。その場合、対象シーケンス内の各ピクチャから複数のPPSの何れかを選択する。

[0032] （符号化ピクチャ）

符号化ピクチャでは、処理対象のピクチャPICTを復号するために画像復号装置３１が参照するデータの集合が規定されている。ピクチャPICTは、図２の（ｂ）に示すように、スライスS0～S_{NS-1}を含んでいる（ＮＳはピクチャPICTに含まれるスライスの総数）。

[0033] なお、以下、スライスS0～S_{NS-1}のそれぞれを区別する必要が無い場合、符号の添え字を省略して記述することがある。また、以下に説明する符号化ストリームTeに含まれるデータであって、添え字を付している他のデータについても同様である。

[0034] （符号化スライス）

符号化スライスでは、処理対象のスライスSを復号するために画像復号装置３１が参照するデータの集合が規定されている。スライスSは、図２の（ｃ）に示すように、スライスヘッダSH、及び、スライスデータSDATAを含んでいる。

[0035] スライスヘッダSHには、対象スライスの復号方法を決定するために画像復号装置３１が参照する符号化パラメータ群が含まれる。スライスタイプを指定するスライスタイプ指定情報（slice_type）は、スライスヘッダSHに含ま

れる符号化パラメータの一例である。

[0036] スライスタイプ指定情報により指定可能なスライスタイプとしては、（１）符号化の際にイントラ予測のみを用いるIスライス、（２）符号化の際に単方向予測、または、イントラ予測を用いるPスライス、（３）符号化の際に単方向予測、双方向予測、または、イントラ予測を用いるBスライスなどが挙げられる。

[0037] なお、スライスヘッダSHには、上記符号化ビデオシーケンスに含まれる、ピクチャパラメータセットPPSへの参照（pic_parameter_set_id）を含んでいても良い。

[0038] （符号化スライスデータ）

符号化スライスデータでは、処理対象のスライスデータSDATAを復号するために画像復号装置３１が参照するデータの集合が規定されている。スライスデータSDATAは、図２の（ｄ）に示すように、符号化ツリーユニット（CTU:CodingTree Unit）を含んでいる。CTUは、スライスを構成する固定サイズ（例えば64x64）のブロックであり、最大符号化単位（LCU:Largest CodingUnit）と呼ぶこともある。

[0039] （符号化ツリーユニット）

図２の（ｅ）に示すように、処理対象の符号化ツリーユニットを復号するために画像復号装置３１が参照するデータの集合が規定されている。符号化ツリーユニットは、再帰的な４分木分割（QT分割）または２分木分割（BT分割）により符号化処理の基本的な単位である符号化ユニット（CU:Coding Unit）に分割される。再帰的な４分木分割または２分木分割により得られる木構造を符号化ツリー（CT:Coding Tree）、木構造のノードのことを符号化ノード（CN:Coding Node）と称する。４分木及び２分木の中間ノードは、符号化ノードであり、符号化ツリーユニット自身も最上位の符号化ノードとして規定される。

[0040] CTUは、QT分割を行うか否かを示すQT分割フラグ（cu_split_flag）、及びBT分割の分割方法を示すBT分割モード（split_bt_mode）を含む。cu_split_fl

ag及び／又はsplit_bt_modeは符号化ノードCNごとに伝送される。cu_split_flagが１の場合には、符号化ノードCNは４つの符号化ノードCNに分割される。cu_split_flagが０の場合には、符号化ノードCNは分割されない。一方、split_bt_modeが１の場合には、符号化ノードCNは２つの符号化ノードCNに水平分割される。split_bt_modeが２の場合には、符号化ノードCNは２つの符号化ノードCNに垂直分割される。split_bt_modeが０の場合には、符号化ノードCNは分割されず、１つの符号化ユニットCUをノードとして持つ。符号化ユニットCUは符号化ノードの末端ノード（リーフノード）であり、これ以上分割されない。

[0041] なお、本明細書では、分割に関して、「水平」および「垂直」という用語は、分割線の方向を指すものとする。したがって、「水平分割」「水平方向(ＨＯＲ)の分割」「水平に分割する」とは、水平な境界線による分割、すなわち、上と下の２つ以上のブロックに分割することを意味する。また、「垂直分割」「垂直方向（ＶＥＲ）の分割」「垂直に分割する」とは、垂直な境界線による分割、すなわち、左と右の２つ以上のブロックに分割することを意味する。

[0042] なお、本明細書では例示していないが別の用語法も存在する。すなわち、本明細書の水平分割を、あるブロックが垂直方向に並ぶ２つ以上のブロックに分割されるため、垂直に分割（split vertically）と呼ぶこともある。また、本明細書の垂直分割を、あるブロックが水平方向に並ぶ２つ以上のブロックに分割されるため水平に分割（split horizontally）と呼ぶこともある。上記、別の用語法における、垂直分割の記載が本明細書の水平分割を意味することがあるので注意が必要である（又はその逆）。この場合には、当該用語が意味するところにおいて適宜読み替える。

[0043] また、符号化ツリーユニットCTUのサイズが64x64画素の場合には、符号化ユニットのサイズは、64x64画素、64x32画素、32x64画素、32x32画素、64x16画素、16x64画素、32x16画素、16x32画素、16x16画素、64x8画素、8x64画素、32x8画素、8x32画素、16x8画素、8x16画素、8x8画素、64x4画素、4x64画素

、32x4画素、4x32画素、16x4画素、4x16画素、8x4画素、4x8画素、及び、4x4画素の何れかをとり得る。

[0044] （符号化ユニット）

図２の（ｆ）に示すように、処理対象の符号化ユニットを復号するために画像復号装置３１が参照するデータの集合が規定されている。具体的には、符号化ユニットは、予測ツリー、変換ツリー、CUヘッダCUHから構成される。CUヘッダでは予測モード、分割方法（PU分割モード）等が規定される。

[0045] 予測ツリーでは、符号化ユニットを１または複数に分割した各予測ユニット（PU）の予測情報（参照ピクチャインデックス、動きベクトル等）が規定される。別の表現でいえば、予測ユニットは、符号化ユニットを構成する１または複数の重複しない領域である。また、予測ツリーは、上述の分割により得られた１または複数の予測ユニットを含む。なお、以下では、予測ユニットをさらに分割した予測単位を「サブブロック」と呼ぶ。サブブロックは、複数の画素によって構成されている。予測ユニットとサブブロックのサイズが等しい場合には、予測ユニット中のサブブロックは１つである。予測ユニットがサブブロックのサイズよりも大きい場合には、予測ユニットは、サブブロックに分割される。たとえば予測ユニットが8x8、サブブロックが4x4の場合には、予測ユニットは水平に２分割、垂直に２分割からなる、４つのサブブロックに分割される。

[0046] 予測処理は、この予測ユニット（サブブロック）ごとに行ってもよい。

[0047] 予測ツリーにおける分割の種類は、大まかにいえば、イントラ予測の場合と、インター予測の場合との２つがある。イントラ予測とは、同一ピクチャ内の予測であり、インター予測とは、互いに異なるピクチャ間（例えば、表示時刻間、レイヤ画像間）で行われる予測処理を指す。

[0048] イントラ予測の場合、分割方法は、2Nx2N（符号化ユニットと同一サイズ）と、NxNとがある。

[0049] また、インター予測の場合、分割方法は、符号化データのPU分割モード（part_mode）により符号化され、2Nx2N（符号化ユニットと同一サイズ）、2NxN

、2NxnU、2NxnD、Nx2N、nLx2N、nRx2N、及び、NxNなどがある。なお、2NxN、Nx2Nは1:1の対称分割を示し、2NxnU、2NxnD及びnLx2N、nRx2Nは、1:3、3:1の非対称分割を示す。CUに含まれるPUを順にPU0、PU1、PU2、PU3と表現する。

[0050] 図３の（ａ）～（ｈ）に、それぞれのPU分割モードにおけるパーティションの形状（PU分割の境界の位置）を具体的に図示している。図３の（ａ）は、2Nx2Nのパーティションを示し、（ｂ）、（ｃ）、（ｄ）は、それぞれ、2NxN、2NxnU、及び、2NxnDのパーティション（横長パーティション）を示す。（ｅ）、（ｆ）、（ｇ）は、それぞれ、Nx2N、nLx2N、nRx2Nである場合のパーティション（縦長パーティション）を示し、（ｈ）は、NxNのパーティションを示す。なお、横長パーティションと縦長パーティションを総称して長方形パーティション、2Nx2N、NxNを総称して正方形パーティションと呼ぶ。

[0051] また、変換ツリーにおいては、符号化ユニットが１または複数の変換ユニットに分割され、各変換ユニットの位置とサイズとが規定される。別の表現でいえば、変換ユニットは、符号化ユニットを構成する１または複数の重複しない領域のことである。また、変換ツリーは、上述の分割より得られた１または複数の変換ユニットを含む。

[0052] 変換ツリーにおける分割には、符号化ユニットと同一のサイズの領域を変換ユニットとして割り付けるものと、上述したCUの分割と同様、再帰的な４分木分割によるものがある。

[0053] 変換処理は、この変換ユニットごとに行われる。

[0054] （予測パラメータ）

予測ユニット（PU：Prediction Unit）の予測画像は、PUに付随する予測パラメータによって導出される。予測パラメータには、イントラ予測の予測パラメータもしくはインター予測の予測パラメータがある。以下、インター予測の予測パラメータ（インター予測パラメータ）について説明する。インター予測パラメータは、予測リスト利用フラグpredFlagL0、predFlagL1と、参照ピクチャインデックスrefIdxL0、refIdxL1と、動きベクトルmvL0、mvL1から構成される。予測リスト利用フラグpredFlagL0、predFlagL1は、各々L0リ

スト、L1リストと呼ばれる参照ピクチャリストが用いられるか否かを示すフラグであり、値が１の場合に対応する参照ピクチャリストが用いられる。なお、本明細書中「ＸＸであるか否かを示すフラグ」と記す場合、フラグが０以外（たとえば１）をＸＸである場合、０をＸＸではない場合とし、論理否定、論理積などでは１を真、０を偽と扱う（以下同様）。但し、実際の装置や方法では真値、偽値として他の値を用いることもできる。

[0055] 符号化データに含まれるインター予測パラメータを導出するためのシンタックス要素には、例えば、PU分割モードpart_mode、マージフラグmerge_flag、マージインデックスmerge_idx、インター予測識別子inter_pred_idc、参照ピクチャインデックスrefIdxLX、予測ベクトルインデックスmvp_LX_idx、差分ベクトルmvdLXがある。

[0056] （参照ピクチャリスト）

参照ピクチャリストは、参照ピクチャメモリ３０６に記憶された参照ピクチャからなるリストである。図４は、参照ピクチャ及び参照ピクチャリストの一例を示す概念図である。図４の（ａ）において、矩形はピクチャ、矢印はピクチャの参照関係、横軸は時間、矩形中のI、P、Bは各々イントラピクチャ、単予測ピクチャ、双予測ピクチャ、矩形中の数字は復号順を示す。図に示すように、ピクチャの復号順は、I0、P1、B2、B3、B4であり、表示順は、I0、B3、B2、B4、P1である。図４の（ｂ）に、参照ピクチャリストの例を示す。参照ピクチャリストは、参照ピクチャの候補を表すリストであり、１つのピクチャ（スライス）が１つ以上の参照ピクチャリストを有してもよい。図の例では、対象ピクチャB3は、L0リストRefPicList0及びL1リストRefPicList1の２つの参照ピクチャリストを持つ。対象ピクチャがB3の場合の参照ピクチャは、I0、P1、B2であり、参照ピクチャはこれらのピクチャを要素として持つ。個々の予測ユニットでは、参照ピクチャリストRefPicListX中のどのピクチャを実際に参照するかを参照ピクチャインデックスrefIdxLXで指定する。図では、refIdxL0及びrefIdxL1により参照ピクチャP1とB2が参照される例を示す。

[0057] （マージ予測とAMVP予測）

予測パラメータの復号（符号化）方法には、マージ予測（merge）モードとAMVP（AdaptiveMotion Vector Prediction、適応動きベクトル予測）モードがある、マージフラグmerge_flagは、これらを識別するためのフラグである。マージ予測モードは、予測リスト利用フラグpredFlagLX（またはインター予測識別子inter_pred_idc）、参照ピクチャインデックスrefIdxLX、動きベクトルmvLXを符号化データに含めずに、既に処理した近傍PUの予測パラメータから導出する用いるモードであり、AMVPモードは、インター予測識別子inter_pred_idc、参照ピクチャインデックスrefIdxLX、動きベクトルmvLXを符号化データに含めるモードである。なお、動きベクトルmvLXは、予測ベクトルmvpLXを識別する予測ベクトルインデックスmvp_LX_idxと差分ベクトルmvdLXとして符号化される。

[0058] インター予測識別子inter_pred_idcは、参照ピクチャの種類及び数を示す値であり、PRED_L0、PRED_L1、PRED_BIの何れかの値をとる。PRED_L0、PRED_L1は、各々L0リスト、L1リストの参照ピクチャリストで管理された参照ピクチャを用いることを示し、１枚の参照ピクチャを用いること（単予測）を示す。PRED_BIは２枚の参照ピクチャを用いること（双予測BiPred）を示し、L0リストとL1リストで管理された参照ピクチャを用いる。予測ベクトルインデックスmvp_LX_idxは予測ベクトルを示すインデックスであり、参照ピクチャインデックスrefIdxLXは、参照ピクチャリストで管理された参照ピクチャを示すインデックスである。なお、LXは、L0予測とL1予測を区別しない場合に用いられる記述方法であり、LXをL0、L1に置き換えることでL0リストに対するパラメータとL1リストに対するパラメータを区別する。

[0059] マージインデックスmerge_idxは、処理が完了したPUから導出される予測パラメータ候補（マージ候補）のうち、いずれかの予測パラメータを復号対象PUの予測パラメータとして用いるかを示すインデックスである。

[0060] （動きベクトル）

動きベクトルmvLXは、異なる２つのピクチャ上のブロック間のずれ量を示

す。動きベクトルmvLXに関する予測ベクトル、差分ベクトルを、それぞれ予測ベクトルmvpLX、差分ベクトルmvdLXと呼ぶ。

[0061] （インター予測識別子inter_pred_idcと予測リスト利用フラグpredFlagLX）

インター予測識別子inter_pred_idcと、予測リスト利用フラグpredFlagL0、predFlagL1の関係は以下のとおりであり、相互に変換可能である。

[0062] inter_pred_idc = (predFlagL1<<１） + predFlagL0

predFlagL0 = inter_pred_idc& 1

predFlagL1 = inter_pred_idc>> 1

なお、インター予測パラメータは、予測リスト利用フラグを用いても良いし、インター予測識別子を用いてもよい。また、予測リスト利用フラグを用いた判定は、インター予測識別子を用いた判定に置き替えてもよい。逆に、インター予測識別子を用いた判定は、予測リスト利用フラグを用いた判定に置き替えてもよい。

[0063] （双予測biPredの判定）

双予測BiPredであるかのフラグbiPredは、２つの予測リスト利用フラグがともに１であるかによって導出できる。たとえば以下の式で導出できる。

[0064] biPred =(predFlagL0 == 1 && predFlagL1 == 1)

フラグbiPredは、インター予測識別子が２つの予測リスト（参照ピクチャ）を使うことを示す値であるか否かによっても導出できる。たとえば以下の式で導出できる。

[0065] biPred =(inter_pred_idc == PRED_BI) ? 1 : 0

上記式は、以下の式でも表現できる。

[0066] biPred =(inter_pred_idc == PRED_BI)

なお、PRED_BIはたとえば3の値を用いることができる。

[0067] （画像復号装置の構成）

次に、本実施形態に係る画像復号装置３１の構成について説明する。図５は、本実施形態に係る画像復号装置３１の構成を示す概略図である。画像復

号装置３１は、エントロピー復号部３０１、予測パラメータ復号部（予測画像復号装置）３０２、ループフィルタ３０５、参照ピクチャメモリ３０６、予測パラメータメモリ３０７、予測画像生成部（予測画像生成装置）３０８、逆量子化・逆DCT部３１１、及び加算部３１２を含んで構成される。

[0068] また、予測パラメータ復号部３０２は、インター予測パラメータ復号部３０３及びイントラ予測パラメータ復号部３０４を含んで構成される。予測画像生成部３０８は、インター予測画像生成部３０９及びイントラ予測画像生成部３１０を含んで構成される。

[0069] エントロピー復号部３０１は、外部から入力された符号化ストリームTeに対してエントロピー復号を行って、個々の符号（シンタックス要素）を分離し復号する。分離された符号には、予測画像を生成するための予測情報及び、差分画像を生成するための残差情報などがある。

[0070] エントロピー復号部３０１は、分離した符号の一部を予測パラメータ復号部３０２に出力する。分離した符号の一部とは、例えば、予測モードpredMode、PU分割モードpart_mode、マージフラグmerge_flag、マージインデックスmerge_idx、インター予測識別子inter_pred_idc、参照ピクチャインデックスrefIdxLX、予測ベクトルインデックスmvp_LX_idx、差分ベクトルmvdLXである。どの符号を復号するかの制御は、予測パラメータ復号部３０２の指示に基づいて行われる。エントロピー復号部３０１は、量子化係数を逆量子化・逆DCT部３１１に出力する。この量子化係数は、符号化処理において、残差信号に対してDCT（Discrete Cosine Transform、離散コサイン変換）を行い量子化して得られる係数である。

[0071] インター予測パラメータ復号部３０３は、エントロピー復号部３０１から入力された符号に基づいて、予測パラメータメモリ３０７に記憶された予測パラメータを参照してインター予測パラメータを復号する。

[0072] インター予測パラメータ復号部３０３は、復号したインター予測パラメータを予測画像生成部３０８に出力し、また予測パラメータメモリ３０７に記憶する。インター予測パラメータ復号部３０３の詳細については後述する。

[0073] イントラ予測パラメータ復号部３０４は、エントロピー復号部３０１から入力された符号に基づいて、予測パラメータメモリ３０７に記憶された予測パラメータを参照してイントラ予測パラメータを復号する。イントラ予測パラメータとは、CUを１つのピクチャ内で予測する処理で用いるパラメータ、例えば、イントラ予測モードIntraPredModeである。イントラ予測パラメータ復号部３０４は、復号したイントラ予測パラメータを予測画像生成部３０８に出力し、また予測パラメータメモリ３０７に記憶する。

[0074] イントラ予測パラメータ復号部３０４は、輝度と色差で異なるイントラ予測モードを導出しても良い。この場合、イントラ予測パラメータ復号部３０４は、輝度の予測パラメータとして輝度予測モードIntraPredModeY、色差の予測パラメータとして、色差予測モードIntraPredModeCを復号する。輝度予測モードIntraPredModeYは、３５モードであり、プレーナ予測（０）、DC予測（１）、方向予測（２～３４）が対応する。色差予測モードIntraPredModeCは、プレーナ予測（０）、DC予測（１）、方向予測（２～３４）、LMモード（３５）の何れかを用いるものである。イントラ予測パラメータ復号部３０４は、IntraPredModeCは輝度モードと同じモードであるか否かを示すフラグを復号し、フラグが輝度モードと同じモードであることを示せば、IntraPredModeCにIntraPredModeYを割り当て、フラグが輝度モードと異なるモードであることを示せば、IntraPredModeCとして、プレーナ予測（０）、DC予測（１）、方向予測（２～３４）、LMモード（３５）を復号しても良い。

[0075] ループフィルタ３０５は、加算部３１２が生成したCUの復号画像に対し、デブロッキングフィルタ、サンプル適応オフセット（SAO）、適応ループフィルタ（ALF）等のフィルタを施す。

[0076] 参照ピクチャメモリ３０６は、加算部３１２が生成したCUの復号画像を、復号対象のピクチャ及びCU毎に予め定めた位置に記憶する。

[0077] 予測パラメータメモリ３０７は、予測パラメータを、復号対象のピクチャ及び予測ユニット（もしくはサブブロック、固定サイズブロック、ピクセル）毎に予め定めた位置に記憶する。具体的には、予測パラメータメモリ３０

７は、インター予測パラメータ復号部３０３が復号したインター予測パラメータ、イントラ予測パラメータ復号部３０４が復号したイントラ予測パラメータ及びエントロピー復号部３０１が分離した予測モードpredModeを記憶する。記憶されるインター予測パラメータには、例えば、予測リスト利用フラグpredFlagLX（インター予測識別子inter_pred_idc）、参照ピクチャインデックスrefIdxLX、動きベクトルmvLXがある。

[0078] 予測画像生成部３０８には、エントロピー復号部３０１から入力された予測モードpredModeが入力され、また予測パラメータ復号部３０２から予測パラメータが入力される。また、予測画像生成部３０８は、参照ピクチャメモリ３０６から参照ピクチャを読み出す。予測画像生成部３０８は、予測モードpredModeが示す予測モードで、入力された予測パラメータと読み出した参照ピクチャを用いてPUの予測画像を生成する。

[0079] ここで、予測モードpredModeがインター予測モードを示す場合、インター予測画像生成部３０９は、インター予測パラメータ復号部３０３から入力されたインター予測パラメータと読み出した参照ピクチャを用いてインター予測によりPUの予測画像を生成する。

[0080] インター予測画像生成部３０９は、予測リスト利用フラグpredFlagLXが１である参照ピクチャリスト（L0リスト、もしくはL1リスト）に対し、参照ピクチャインデックスrefIdxLXで示される参照ピクチャから、復号対象PUを基準として動きベクトルmvLXが示す位置にある参照ピクチャブロックを参照ピクチャメモリ３０６から読み出す。インター予測画像生成部３０９は、読み出した参照ピクチャブロックをもとに予測を行ってPUの予測画像を生成する。インター予測画像生成部３０９は、生成したPUの予測画像を加算部３１２に出力する。

[0081] 予測モードpredModeがイントラ予測モードを示す場合、イントラ予測画像生成部３１０は、イントラ予測パラメータ復号部３０４から入力されたイントラ予測パラメータと読み出した参照ピクチャを用いてイントラ予測を行う。具体的には、イントラ予測画像生成部３１０は、復号対象のピクチャであ

って、既に復号されたPUのうち、復号対象PUから予め定めた範囲にある隣接PUを参照ピクチャメモリ３０６から読み出す。予め定めた範囲とは、復号対象PUがいわゆるラスタースキャンの順序で順次移動する場合、例えば、左、左上、上、右上の隣接PUのうちのいずれかであり、イントラ予測モードによって異なる。ラスタースキャンの順序とは、各ピクチャにおいて、上端から下端まで各行について、順次左端から右端まで移動させる順序である。

[0082] イントラ予測画像生成部３１０は、読み出した隣接PUについてイントラ予測モードIntraPredModeが示す予測モードで予測を行ってPUの予測画像を生成する。イントラ予測画像生成部３１０は、生成したPUの予測画像を加算部３１２に出力する。

[0083] イントラ予測パラメータ復号部３０４において、輝度と色差で異なるイントラ予測モードを導出する場合、イントラ予測画像生成部３１０は、輝度予測モードIntraPredModeYに応じて、プレーナ予測（０）、DC予測（１）、方向予測（２～３４）の何れかによって輝度のPUの予測画像を生成し、色差予測モードIntraPredModeCに応じて、プレーナ予測（０）、DC予測（１）、方向予測（２～３４）、LMモード（３５）の何れかによって色差のPUの予測画像を生成する。

[0084] 逆量子化・逆DCT部３１１は、エントロピー復号部３０１から入力された量子化係数を逆量子化してDCT係数を求める。逆量子化・逆DCT部３１１は、求めたDCT係数について逆DCT（InverseDiscrete Cosine Transform、逆離散コサイン変換）を行い、残差信号を算出する。逆量子化・逆DCT部３１１は、算出した残差信号を加算部３１２に出力する。

[0085] 加算部３１２は、インター予測画像生成部３０９またはイントラ予測画像生成部３１０から入力されたPUの予測画像と逆量子化・逆DCT部３１１から入力された残差信号を画素毎に加算して、PUの復号画像を生成する。加算部３１２は、生成したPUの復号画像を参照ピクチャメモリ３０６に記憶し、生成したPUの復号画像をピクチャ毎に統合した復号画像Tdを外部に出力する。

[0086] （画像符号化装置の構成）

次に、本実施形態に係る画像符号化装置１１の構成について説明する。図６は、本実施形態に係る画像符号化装置１１の構成を示すブロック図である。画像符号化装置１１は、予測画像生成部１０１、減算部１０２、DCT・量子化部１０３、エントロピー符号化部１０４、逆量子化・逆DCT部１０５、加算部１０６、ループフィルタ１０７、予測パラメータメモリ（予測パラメータ記憶部、フレームメモリ）１０８、参照ピクチャメモリ（参照画像記憶部、フレームメモリ）１０９、符号化パラメータ決定部１１０、予測パラメータ符号化部１１１を含んで構成される。予測パラメータ符号化部１１１は、インター予測パラメータ符号化部１１２及びイントラ予測パラメータ符号化部１１３を含んで構成される。

[0087] 予測画像生成部１０１は画像Ｔの各ピクチャについて、そのピクチャを分割した領域である符号化ユニットCU毎に予測ユニットPUの予測画像Ｐを生成する。ここで、予測画像生成部１０１は、予測パラメータ符号化部１１１から入力された予測パラメータに基づいて参照ピクチャメモリ１０９から復号済のブロックを読み出す。予測パラメータ符号化部１１１から入力された予測パラメータとは、例えばインター予測の場合、動きベクトルである。予測画像生成部１０１は、対象PUを起点として動きベクトルが示す参照画像上の位置にあるブロックを読み出す。またイントラ予測の場合、予測パラメータとは例えばイントラ予測モードである。イントラ予測モードで使用する隣接PUの画素値を参照ピクチャメモリ１０９から読み出し、PUの予測画像Pを生成する。予測画像生成部１０１は、読み出した参照ピクチャブロックについて複数の予測方式のうちの１つの予測方式を用いてPUの予測画像Ｐを生成する。予測画像生成部１０１は、生成したPUの予測画像Ｐを減算部１０２に出力する。

[0088] なお、予測画像生成部１０１は、既に説明した予測画像生成部３０８と同じ動作である。例えば、図７は、予測画像生成部１０１に含まれるインター予測画像生成部１０１１の構成を示す概略図である。インター予測画像生成部１０１１は、動き補償部１０１１１、重み予測部１０１１２を含んで構成

される。動き補償部１０１１１及び重み予測部１０１１２については、上述の動き補償部３０９１、重み予測部３０９４のそれぞれと同様の構成であるためここでの説明を省略する。

[0089] 予測画像生成部１０１は、予測パラメータ符号化部から入力されたパラメータを用いて、参照ピクチャメモリから読み出した参照ブロックの画素値をもとにPUの予測画像Pを生成する。予測画像生成部１０１で生成した予測画像は減算部１０２、加算部１０６に出力される。

[0090] 減算部１０２は、予測画像生成部１０１から入力されたPUの予測画像Ｐの信号値を、画像Ｔの対応するPUの画素値から減算して、残差信号を生成する。減算部１０２は、生成した残差信号をDCT・量子化部１０３に出力する。

[0091] DCT・量子化部１０３は、減算部１０２から入力された残差信号についてDCTを行い、DCT係数を算出する。DCT・量子化部１０３は、算出したDCT係数を量子化して量子化係数を求める。DCT・量子化部１０３は、求めた量子化係数をエントロピー符号化部１０４及び逆量子化・逆DCT部１０５に出力する。

[0092] エントロピー符号化部１０４には、DCT・量子化部１０３から量子化係数が入力され、予測パラメータ符号化部１１１から符号化パラメータが入力される。入力される符号化パラメータには、例えば、参照ピクチャインデックスrefIdxLX、予測ベクトルインデックスmvp_LX_idx、差分ベクトルmvdLX、予測モードpredMode、及びマージインデックスmerge_idx等の符号がある。

[0093] エントロピー符号化部１０４は、入力された量子化係数と符号化パラメータをエントロピー符号化して符号化ストリームTeを生成し、生成した符号化ストリームTeを外部に出力する。

[0094] 逆量子化・逆DCT部１０５は、DCT・量子化部１０３から入力された量子化係数を逆量子化してDCT係数を求める。逆量子化・逆DCT部１０５は、求めたDCT係数について逆DCTを行い、残差信号を算出する。逆量子化・逆DCT部１０５は、算出した残差信号を加算部１０６に出力する。

[0095] 加算部１０６は、予測画像生成部１０１から入力されたPUの予測画像Ｐの信号値と逆量子化・逆DCT部１０５から入力された残差信号の信号値を画素毎

に加算して、復号画像を生成する。加算部１０６は、生成した復号画像を参照ピクチャメモリ１０９に記憶する。

[0096] ループフィルタ１０７は加算部１０６が生成した復号画像に対し、デブロッキングフィルタ、サンプル適応オフセット（SAO）、適応ループフィルタ（ALF）を施す。

[0097] 予測パラメータメモリ１０８は、符号化パラメータ決定部１１０が生成した予測パラメータを、符号化対象のピクチャ及びCU毎に予め定めた位置に記憶する。

[0098] 参照ピクチャメモリ１０９は、ループフィルタ１０７が生成した復号画像を、符号化対象のピクチャ及びCU毎に予め定めた位置に記憶する。

[0099] 符号化パラメータ決定部１１０は、符号化パラメータの複数のセットのうち、１つのセットを選択する。符号化パラメータとは、上述した予測パラメータやこの予測パラメータに関連して生成される符号化の対象となるパラメータである。予測画像生成部１０１は、これらの符号化パラメータのセットの各々を用いてPUの予測画像Ｐを生成する。

[0100] 符号化パラメータ決定部１１０は、複数のセットの各々について情報量の大きさと符号化誤差を示すコスト値を算出する。コスト値は、例えば、符号量と二乗誤差に係数λを乗じた値との和である。符号量は、量子化誤差と符号化パラメータをエントロピー符号化して得られる符号化ストリームTeの情報量である。二乗誤差は、減算部１０２において算出された残差信号の残差値の二乗値についての画素間の総和である。係数λは、予め設定されたゼロよりも大きい実数である。符号化パラメータ決定部１１０は、算出したコスト値が最小となる符号化パラメータのセットを選択する。これにより、エントロピー符号化部１０４は、選択した符号化パラメータのセットを符号化ストリームTeとして外部に出力し、選択されなかった符号化パラメータのセットを出力しない。符号化パラメータ決定部１１０は決定した符号化パラメータを予測パラメータメモリ１０８に記憶する。

[0101] 予測パラメータ符号化部１１１は、符号化パラメータ決定部１１０から入

力されたパラメータから、符号化するための形式を導出し、エントロピー符号化部１０４に出力する。符号化するための形式の導出とは、例えば動きベクトルと予測ベクトルから差分ベクトルを導出することである。また予測パラメータ符号化部１１１は、符号化パラメータ決定部１１０から入力されたパラメータから予測画像を生成するために必要なパラメータを導出し、予測画像生成部１０１に出力する。予測画像を生成するために必要なパラメータとは、例えばサブブロック単位の動きベクトルである。

[0102] インター予測パラメータ符号化部１１２は、符号化パラメータ決定部１１０から入力された予測パラメータに基づいて、差分ベクトルのようなインター予測パラメータを導出する。インター予測パラメータ符号化部１１２は、予測画像生成部１０１に出力する予測画像の生成に必要なパラメータを導出する構成として、インター予測パラメータ復号部３０３（図６等、参照）がインター予測パラメータを導出する構成と一部同一の構成を含む。インター予測パラメータ符号化部１１２の構成については、後述する。

[0103] イントラ予測パラメータ符号化部１１３は、符号化パラメータ決定部１１０から入力されたイントラ予測モードIntraPredModeから、符号化するための形式（例えばMPM_idx、rem_intra_luma_pred_mode等）を導出する。

[0104] （画像復号装置の構成）

図１０に、本実施形態に係る画像復号装置の構成を示すブロック図を示す。本図では、図を簡略化するために、図１０に示したブロック図に含まれる一部の部材の図示を省略している。また、説明の便宜上、図５に示した部材と同じ機能を有する部材については、同じ符号を付記し、その説明を省略する。

[0105] 図１０に示すように、画像復号装置３１は、復号モジュール９、CT情報復号部１０（分割情報復号部、分割部）、予測画像生成部３０８、逆量子化・逆DCT部３１１、参照ピクチャメモリ３０６、加算部３１２、ループフィルタ３０５、ヘッダ復号部１９、及びCU復号部２０を備えている。CU復号部２０は、さらにPU情報復号部１２及びTT情報復号部１３を備えており、TT情報復

号部１３は、さらにTU復号部２２を備えている。

[0106] （復号モジュール）

以下、各モジュールの概略動作を説明する。復号モジュール９は、バイナリデータからシンタックス値を復号する復号処理を行う。復号モジュール９は、より具体的には、供給元から供給される符号化データ及びシンタックス種別に基づいて、CABAC等のエントロピー符号化方式により符号化されているシンタックス値を復号し、復号したシンタックス値を供給元に返す。

[0107] 以下に示す例では、符号化データ及びシンタックス種別の供給元は、CT情報復号部１０、CU復号部２０（PU情報復号部１２及びTT情報復号部１３）である。

[0108] （ヘッダ復号部）

ヘッダ復号部１９は、画像符号化装置１１から入力された符号化データのVPS（video parameter set）、SPS、PPS、スライスヘッダを復号する。

[0109] （CT情報復号部）

CT情報復号部１０は、復号モジュール９を用いて、画像符号化装置１１から入力された符号化データについて、符号化ツリーユニット及び符号化ツリーの復号処理を行う。CT情報復号部１０は、具体的には、以下の手順により符号化データから、CTU情報及びCT情報を復号する。

[0110] まず、CT情報復号部１０は、復号モジュール９を用いて、CTUに含まれるCTU情報からツリーユニットヘッダCTUHを復号する。次に、CT情報復号部１０は、CTに含まれるCT情報から、対象CTをQT分割するか否かを示すQT分割フラグ、及び対象CTのBT分割の分割方法を示すBT分割モードを復号し、QT分割フラグ及びBT分割モードがさらなる分割を通知しなくなるまで対象CTを再帰的に分割し復号する。最後に、CTU情報からツリーユニットフッタCTUFを復号する。

[0111] ツリーユニットヘッダCTUH及びツリーユニットフッタCTUFには、対象符号化ツリーユニットの復号方法を決定するために画像復号装置３１が参照する符号化パラメータが含まれる。また、CT情報には、QT分割フラグ及びBT分割

モードの他、対象CT及び下位の符号化ノードで適用されるパラメータを含んでいてもよい。

[0112] (CU復号部)

CU復号部２０は、PU情報復号部１２及びTT情報復号部１３から構成され、最下位の符号化ツリーCT（すなわちCU）のPUI情報及びTTI情報を復号する。

[0113] (PU情報復号部)

PU情報復号部１２では各ＰＵのＰＵ情報（マージフラグ（merge_flag）、マージインデックス（merge_idx）、予測動きベクトルインデックス（mvp_idx）、参照画像インデックス（ref_idx）、インター予測識別子（inter_pred_flag）、及び差分ベクトル（mvd）等）を、復号モジュール９を用いて復号する。

[0114] (TT情報復号部)

TT情報復号部１３は、各TTI（TU分割フラグSP_TU（split_transform_flag）、CU残差フラグCBP_TU（cbf_cb、cbf_cr、cbf_luma）等、及びＴＵ）を、復号モジュール９を用いて復号する。

[0115] また、TT情報復号部１３は、TU復号部２２を備えている。TU復号部２２は、TUに残差が含まれている場合に、QP更新情報（量子化補正値）を復号する。なお、QP更新情報は、量子化パラメータQPの予測値である量子化パラメータ予測値qPpredからの差分値を示す値である。またTU復号部２２は、量子化予測残差（residual_coding）を復号する。

[0116] (CT情報復号の処理)

CT情報復号部１０によるCT情報復号の動作について、図１１～図１３を参照して詳細に説明する。図１１は、本発明の一実施形態に係るCT情報復号部１０の動作を説明するフローチャートである。また、図１２は、本発明の一実施形態に係るQT情報のシンタックス表の構成例を示す図であり、図１３は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0117] CT情報復号部１０によるCT情報復号Ｓ１４００では、QT情報復号と、BT情

報復号とを行う。以下では、CT情報復号部１０によるQT情報復号と、BT情報復号とについて順に説明する。

[0118] まず、CT情報復号部１０は、符号化データからCT情報を復号し、再帰的に符号化ツリーCT（coding_quadtree）を復号する。具体的には、CT情報復号部１０は、QT情報を復号し、対象符号化ツリーcoding_quadtree(x0, y0, log2CbSize,cqtDepth)を復号する。なお、x0, y0は、対象符号化ツリーの左上座標、log2CbSizeは符号化ツリーのサイズであるCTサイズの２を底とした対数である対数CTサイズ（例えば、CTサイズが64,128, 256であれば6, 7, 8）である。cqtDepthは符号化ツリーの階層を示すCT階層（QT深度）である。

[0119] （Ｓ１４１１）CT情報復号部１０は、復号したCT情報にQT分割フラグがあるか否かを判定する。具体的には、CT情報復号部１０は、対数CTサイズlog2CbSizeが所定の最小CTサイズの対数値MinCbLog2SizeYより大きいか否かを判定する。対数CTサイズlog2CbSizeがMinCbLog2SizeYより大きい場合には、QT分割フラグがあると判定し、Ｓ１４２１に遷移する。それ以外の場合には、Ｓ１４２２に遷移する。

[0120] （Ｓ１４２１）CT情報復号部１０は、対数CTサイズlog2CbSizeがMinCbLog2SizeYより大きいと判定された場合には、シンタックス要素であるQT分割フラグ(split_cu_flag)を復号する。

[0121] （Ｓ１４２２）CT情報復号部１０は、それ以外の場合（対数CTサイズlog2CbSizeがMinCbLog2SizeY以下）、つまり、符号化データにQT分割フラグsplit_cu_flagが現れない場合には、符号化データからのQT分割フラグsplit_cu_flagの復号を省略し、QT分割フラグsplit_cu_flagを０として導出する。

[0122] （Ｓ１４３１）CT情報復号部１０は、QT分割フラグsplit_cu_flagが０以外（＝１）である場合には、後述の（Ｓ１４４１）を実施し、一階層下に移行して（Ｓ１４１１）以降の処理を繰り返す。それ以外の場合（QT分割フラグsplit_cu_flagが０の場合）には、Ｓ１４５１に遷移する。

[0123] （Ｓ１４４１）CT情報復号部１０は、QT分割を行う。具体的には、CT情報復号部１０は、CT階層cqtDepth + 1の位置(x0,y0)、(x1, y0)、(x0, y1)、(x

1, y1)で、対数CTサイズlog2CbSize- 1の４つの符号化ツリーCTを復号する。

[0124] coding_quadtree(x0, y0, log2CbSize - 1, cqtDepth + 1)

coding_quadtree(x1, y0, log2CbSize - 1, cqtDepth + 1)

coding_quadtree(x0, y1, log2CbSize - 1, cqtDepth + 1)

coding_quadtree(x1, y1, log2CbSize - 1, cqtDepth + 1)

ここで、x0, y0は、対象符号化ツリーの左上座標、x1, y1は、以下の式のように、(x0,y0)に、対数CTサイズ（1 <<log2CbSize）の1/2を加えて導出される。

[0125] x1 = x0 + (1<<(log2CbSize - 1))

y1 = y0 + (1<<(log2CbSize -1))

なお、<<は左シフトを示す。1<<Nは2のN乗と同値である（以下同様）。同様に>>は右シフトを示す。

[0126] そして、CT情報復号部１０は、符号化ツリーの階層を示すCT階層cqtDepthに１を加算し、符号化ユニットサイズの対数値である対数CTサイズlog2CbSizeを１だけ減算（CTサイズを1/2）して更新する。

[0127] cqtDepth = cqtDepth+ 1

log2CbSize = log2CbSize- 1

CT情報復号部１０は、下位の符号化ツリーCTにおいても、更新された左上座標、対数CTサイズ、CT階層を用いて、Ｓ１４１１から開始されるQT情報復号を継続する。

[0128] （Ｓ１４５１）CT情報復号部１０は、符号化データからCT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、BT情報を復号し、対象符号化ツリーcoding_binarytree(x0, y0,log2CbWidth, log2CbHeight, cqtDepth, cbtDepth)を復号する。なお、x0,y0は、対象符号化ツリーの左上座標、log2CbWidthは符号化ツリーの幅の対数値、log2CbHeightは符号化ツリーの高さの対数値、cqtDepthは符号化ツリーの階層を示すCT階層（QT深度）、cbtDepthは符号化ツリーの階層を示すCT階層（BT深度）である。

[0129] CT情報復号部１０は、復号したCT情報にBT分割モード（分割情報）があるか否かを判定する。具体的には、CT情報復号部１０は、符号化ツリーの高さ（1<<log2CbHeight）がminBTSizeよりも大きい、または符号化ツリーの幅（1<<log2CbWidth）がminBTSizeよりも大きい、且つ、符号化ツリーの幅（1<<log2CbWidth）がmaxBTSize以下、且つ、符号化ツリーの高さ（1<<log2CbHeight）がmaxBTSize以下、且つ、cbtDepthがmaxBTDepthよりも小さいか否かを判定する（以下では、所定条件と称す）。minBTSizeは最小BTサイズ（二分木のリーフノードの最小サイズ）、maxBTSizeは最大BTサイズ（二分木のルートノードの最大サイズ）、maxBTDepthは最大BT深度（二分木の深度の最大値）である。

[0130] 上記の所定条件を満たす場合には、BT分割モードがあると判定し、Ｓ１４６１に遷移する。それ以外の場合には、Ｓ１４６２に遷移する。

[0131] （Ｓ１４６１）CT情報復号部１０は、上記の所定条件を満たす場合には、シンタックス要素であるBT分割モードsplit_bt_mode[x0][y0]を復号する。

[0132] （Ｓ１４６２）CT情報復号部１０は、それ以外の場合（上記の所定条件を満たさない場合）、つまり、符号化データにBT分割モードsplit_bt_mode[x0][y0]が現れない場合には、符号化データからのBT分割モードsplit_bt_mode[x0][y0]の復号を省略し、BT分割モードsplit_bt_mode[x0][y0]を０として導出する。

[0133] （Ｓ１４７１）CT情報復号部１０は、BT分割モードsplit_bt_mode[x0][y0]が０以外（＝１または２）である場合には、後述の（Ｓ１４８１）を実施し、一階層下に移行して（Ｓ１４５１）以降の処理を繰り返す。それ以外の場合（BT分割モードsplit_bt_mode[x0][y0]が０の場合）には、CT情報復号部１０は、対象符号化ツリーを分割せず、処理を終了する。

[0134] （Ｓ１４８１）CT情報復号部１０は、BT分割を行う。具体的には、CT情報復号部１０は、BT分割モードsplit_bt_mode[x0][y0]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、幅及び高さの対数値がそれぞれlog2CbWidth及びlog2CbHeight - 1の２つの符号化ツリーCTを復号する。

[0135] coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)

coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)

一方、BT分割モードsplit_bt_mode[x0][y0]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、幅及び高さの対数値がそれぞれlog2CbWidth- 1及びlog2CbHeightの２つの符号化ツリーCTを復号する。

[0136] coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)

coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)

ここで、x0, y0は、対象符号化ツリーの左上座標、x1, y1は、以下の式のように、(x0,y0)に、（1 <<log2CbWidth）の1/2及び（1 <<log2CbHeight）の1/2をそれぞれ加えて導出される。

[0137] x1 = x0 + (1<<(log2CbWidth - 1))

y1 = y0 + (1<<(log2CbHeight -1))

なお、<<は左シフトを示す。1<<Nは2のN乗と同値である（以下同様）。同様に>>は右シフトを示す。

[0138] そして、CT情報復号部１０は、符号化ツリーの階層を示すCT階層cbtDepthに１を加算し、log2CbWidthまたはlog2CbHeightを１だけ減算して更新する。

[0139] cbtDepth = cbtDepth+ 1

log2CbWidth =log2CbWidth - 1

log2CbHeight =log2CbHeight - 1

CT情報復号部１０は、下位の符号化ツリーCTにおいても、更新された左上座標、CTの幅及び高さの対数値、BT階層を用いて、Ｓ１４５１から開始されるBT情報復号を継続する。

[0140] なお、以上で説明したCT情報復号は、QTツリーとBTツリーとが別レイヤで

ある場合を想定した復号方法である。この方法では、BT分割後にはQT分割を行うことができないが、BT分割後にはQT分割フラグの有無の判定が不要となる。しかし、本実施形態はこれに限定されず、QTツリーとBTツリーとが同一レイヤである場合を想定した復号方法を採用してもよい。この場合は、QT分割及びBT分割のいずれも選択できる、すなわちBT分割後にもQT分割を行うことができるが、毎度QT分割フラグの有無の判定が必要となる。

[0141] （BT分割の制限）

本実施形態では、動画像の符号化／復号の複雑度を低減するために、BT分割の分割方法を制限している。具体的には、CTUの符号化ノードを復号する際、復号済の他の符号化ノードにおけるBT分割の分割方法を参照してBT分割を行う。

[0142] 本実施形態では、CTUの符号化ノードを復号する際、復号済の他の符号化ノードにおけるBT分割の分割方法を参照してBT分割を行うため、符号化ノードのBT分割パターンが制限される。これにより、符号化ノードのBT分割に関する符号化／復号の複雑度を低減することができる。

[0143] 以下では、具体的な態様として、BTツリーにおけるルートノード（cbtDepth = 0の符号化ノード）の分割方法（水平分割または垂直分割）を用いて、下位の符号化ノードの分割方法を制限する態様を示す。

[0144] 図１４に、BT分割の分割方法の一例を示す。図中の実線は親ノードの分割を示し、点線は子ノードの分割を示す。本態様では、図１４の（ａ）に示すように、親ノードの分割方法が水平分割の場合は、子ノードの分割方法も水平分割に制限される。一方、図１４の（ｂ）に示すように、親ノードの分割方法が垂直分割の場合は、子ノードの分割方法も垂直分割に制限される。

[0145] （BT情報復号の処理－１）

本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図１５を参照して詳細に説明する。図１５は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0146] この場合、画像符号化装置１１においては、cbtDepth = 0であるルートノ

ードを符号化する際に、ルートノードの分割方向を示すBT分割方向フラグsplit_dir_flag[x0][y0]を符号化する。split_dir_flag[x0][y0]が０の場合には、垂直分割を表し、split_dir_flag[x0][y0]が１の場合には、水平分割を表す。

[0147] また、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。split_bt_flag[x0][y0]が０の場合には、BT分割を行わないことを表し、split_bt_flag[x0][y0]が１の場合には、BT分割を行うことを表す。

[0148] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth,cbtDepth)を復号する。

[0149] CT情報復号部１０は、所定条件を満たさない場合、つまり、符号化データにBT分割フラグsplit_bt_flag[x0][y0]が現れない場合には、符号化データからのBT分割フラグsplit_bt_flag[x0][y0]の復号を省略し、BT分割モードsplitBtMode[x0][y0]を０として導出する。

[0150] 一方、CT情報復号部１０は、所定条件を満たす場合には、シンタックス要素であるBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0151] CT情報復号部１０は、cbtDepth = 0（ルートノード）であり、且つ、BT分割フラグsplit_bt_flag[x0][y0]が０以外（＝１）である場合には、BT分割方向フラグsplit_dir_flag[x0][y0]を復号する。

[0152] CT情報復号部１０は、復号したBT分割方向フラグsplit_dir_flag[x0][y0]及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0]を導出する。ここで、BT分割モードsplitBtMode[x0][y0]の導出テーブルを図１６に示す。

[0153] 図１６に示すように、BT分割方向フラグsplit_dir_flag[x0][y0]及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0]が定まる。なお、BT_SPLIT_NONEは分割しないことを表し、spl

itBtMode[x0][y0] = 0となり、BT_SPLIT_HORは水平分割を表し、splitBtMode[x0][y0] = 1となり、BT_SPLIT_VERは垂直分割を表し、splitBtMode[x0][y0] = 2となる。

[0154] このようにして導出したBT分割モードsplitBtMode[x0][y0]に基づき、CT情報復号部１０は、BT分割を行う。CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0155] なお、CT情報復号部１０は、cbtDepth = 0（ルートノード）において復号したBT分割方向フラグsplit_dir_flag[x0][y0]をメモリに記憶する。CT情報復号部１０は、cbtDepth =0以外においては、BT分割フラグsplit_bt_flag[x0][y0]が０以外（＝１）である場合には、メモリに記憶されているBT分割フラグsplit_dir_flag[x0][y0]からBT分割モードsplitBtModeを導出し（つまり、ルートノードと同じ分割方向を設定し）、BT分割フラグsplit_dir_flag[x0][y0]は復号しない。

[0156] なお、以上では、cbtDepth =0の符号化ノードのBT分割方向フラグsplit_dir_flag[x0][y0]を参照して、下位の符号化ノードの分割を決定する構成について説明したが、本実施形態は必ずしもこれに限定されない。例えば、cbtDepth = iとした場合、当該符号化ノードの分割を、cbtDepth = j（j<i, j：０以上の整数）の符号化ノードのBT分割方向フラグsplit_dir_flag[x0][y0]を参照して決定してもよい。

[0157] なお、上記では、BT分割（分割情報）を示すシンタックスとして、分割するか否かを示すシンタックスsplit_bt_flagと分割方法を示すシンタックスsplit_dir_flagの２つのシンタックスを用いているが、両者をあわせたシンタックス（例えばsplit_bt_mode）を用いてもよい。この場合、split_bt_modeのバイナリ表現の先頭ビットが分割するか否か（split_bt_flagに相当）を示し、後続ビットが分割方法（split_dir_flagに相当）を示す。またsplit_bt_modeの先頭ビットが0、つまり、分割しないことを示す場合や、分割方法が制限されている場合（BT分割フラグsplit_dir_flagを含まない条件の場合）には、split_bt_modeの後続ビットは現れない。

[0158] （BT情報復号の処理－２）

本態様におけるCT情報復号部１０によるBT情報復号の他の動作例について、図１７を参照して詳細に説明する。図１７は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0159] この場合、画像復号装置３１においては、親ノードの分割方法を示すsplitConstrainを子ノードに継承する。splitConstrainが０の場合には、BT分割の分割方法を制限しない事を示し、splitConstrainが１の場合には、BT分割の分割方法を親ノードの分割方法である水平分割に制限することを示し、splitConstrainが２の場合には、BT分割の分割方法を親ノードの分割方法である垂直分割に制限することを示す。

[0160] また、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0161] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth,splitConstrain)を復号する。splitConstrainの初期値は０である。

[0162] CT情報復号部１０は、所定条件を満たさない場合、つまり、符号化データにBT分割モードsplit_bt_mode[x0][y0]またはBT分割フラグsplit_bt_flag[x0][y0]が現れない場合には、BT分割モードsplitBtMode[x0][y0]を０として導出する。

[0163] 一方、CT情報復号部１０は、所定条件を満たす場合には、シンタックス要素であるBT分割モードsplit_bt_mode[x0][y0]またはBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0164] CT情報復号部１０は、splitConstrainが０である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0]を対象CTのBT分割モードsplitBtMode[x0][y0]として設

定する。

[0165] 一方、CT情報復号部１０は、splitConstrainが０以外（＝１，２）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0]を導出する。ここで、BT分割モードsplitBtMode[x0][y0]の導出テーブルを図１８に示す。

[0166] 図１８に示すように、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0]が定まる。

[0167] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。

[0168] 一方、BT分割モードsplit_bt_mode[x0][y0]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。

[0169] CT情報復号部１０は、復号された各符号化ツリーCTのsplitConstrainにsplitBtMode[x0][y0]をセットし、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0170] なお、上記では、BT分割方法（分割情報）を示すシンタックスとして値域が２値の場合と多値の場合で異なるシンタックスを割り当てているが、どちらも同じシンタックス名（例えばsplit_bt_mode）を用いてもよい。つまり、分割するか否かのみを示すシンタックスsplit_bt_flagと、分割するか否かと分割の方向を示すシンタックスsplit_bt_modeに分けることなく同じシンタックスsplit_bt_modeを用いてもよい。シンタックスがどちらの意味であるかは、自明に判別できる。なお、この場合、同じシンタックスにおいても条件に応じてバイナリ表現が異なることになる。例えば、２値の場合はバイナリ表現は、0, 1のいずれかであり、３値の場合には0, 10, 11のいずれかである。以下、同様である。この場合、「BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]」を

復号（符号化）すると記載する部分は、単に、「BT分割情報split_bt_mode[x0][y0][cbtDepth]」を復号（符号化）すると置き換えればよい。

[0171] （BT情報復号の処理－３）

本態様におけるCT情報復号部１０によるBT情報復号のさらに他の動作例について、図１９を参照して詳細に説明する。図１９は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0172] この場合、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0173] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cbtDepth)を復号する。

[0174] CT情報復号部１０は、所定条件を満たさない場合、つまり、符号化データにBT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割フラグsplit_bt_flag[x0][y0]が現れない場合には、符号化データからのBT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割フラグsplit_bt_flag[x0][y0]の復号を省略し、BT分割モードsplitBtMode[x0][y0][cbtDepth]を０として導出する。

[0175] 一方、CT情報復号部１０は、所定条件を満たし、且つ、cbtDepth = 0である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]として設定する。

[0176] また、CT情報復号部１０は、所定条件を満たし、且つ、cbtDepth = 0以外（＝cbtDepth> 0）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、親ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth-1]及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出する。ここで、BT分割モードsp

litBtMode[x0][y0][cbtDepth]の導出テーブルを図２０に示す。

[0177] 図２０に示すように、親ノードのBT分割フラグsplitBtMode[x0][y0][cbtDepth-1]及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0][cbtDepth]が定まる。

[0178] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1の２つの符号化ツリーCTを復号する。

[0179] 一方、BT分割モードsplit_bt_mode[x0][y0]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1の２つの符号化ツリーCTを復号する。

[0180] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0181] 上記同様、分割情報のシンタックスをsplit_bt_flagとsplit_bt_modeの２つに分けることなく１つのシンタックスsplit_bt_modeを用い、バイナリ表現から両者を区別しても構わない。

[0182] （第２の実施形態）

本実施形態では、BT分割の分割方向を制限する、すなわちCTUの符号化ノードを復号する際、復号済の他の符号化ノードにおけるBT分割の分割方法を用いてBT分割を行う他の具体的な態様として、BTツリーにおいて、親ノードが共通の同じcbtDepthの符号化ノード（兄ノード）の分割方法（水平分割または垂直分割）に応じて、親ノードが共通の同じcbtDepthの他の符号化ノード（弟ノード）の分割方法を制限する態様を示す。

[0183] 上記の態様の具体例としては、弟ノードの分割方法を兄ノードと同じ分割方法に制限する態様、弟ノードの分割方法を兄ノードの分割方法とは異なる分割方法に制限する態様、及び兄ノードがBT分割されている場合には、弟ノードのBT分割を禁止する態様が挙げられる。ここで、親ノードが共通の同じcbtDepthの符号化ノードのうち、先に復号される符号化ノードが兄ノードであり、後に復号される符号化ノードが弟ノードである。

[0184] まず、BTツリーにおいて、親ノードが共通の同じcbtDepthの他の符号化ノード（弟ノード）の分割方法を、親ノードが共通の同じcbtDepthの符号化ノード（兄ノード）の分割方法（水平分割または垂直分割）に制限する態様を以下に説明する。図２１に、BT分割の分割方法の一例を示す。図中のPartUnitIdx = 0のノードは兄ノードを表し、PartUnitIdx =1のノードは弟ノードを表す。

[0185] 本態様では、図２１の（ａ）に示すように、兄ノードの分割方法が水平分割の場合は、弟ノードの分割方法も水平分割に制限される。一方、図２１の（ｂ）に示すように、兄ノードの分割方法が垂直分割の場合は、弟ノードの分割方法も垂直分割に制限される。

[0186] （BT情報復号の処理－１）

本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図２２を参照して詳細に説明する。図２２は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0187] この場合、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0188] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth,splitConstrain)を復号する。splitConstrainの初期値は０である。

[0189] CT情報復号部１０は、所定条件を満たさない場合、BT分割モードsplitBtMode[x0][y0][cbtDepth]を０として導出する。

[0190] 一方、CT情報復号部１０は、所定条件を満たす場合には、シンタックス要素であるBT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0191] CT情報復号部１０は、splitConstrainが０である場合には、符号化データ

からBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]として設定する。

[0192] 一方、CT情報復号部１０は、splitConstrainが０以外（＝１，２）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出する。ここで、BT分割モードsplitBtMode[x0][y0][cbtDepth]の導出テーブルを図２３に示す。

[0193] 図２３に示すように、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0][cbtDepth]が定まる。

[0194] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号するが、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、図２１の（ａ）に示すように、splitConstrainを兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]とする。

[0195] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号するが、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、図２１の（ｂ）に示すように、splitConstrainを兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]とする。

[0196] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0197] 上記同様、分割情報のシンタックスをsplit_bt_flagとsplit_bt_modeの２つに分けることなく１つのシンタックスsplit_bt_modeを用い、バイナリ表現から両者を区別しても構わない。

[0198] （BT情報復号処理－２）

続いて、BTツリーにおいて、親ノードが共通の同じcbtDepthの他の符号化ノード（弟ノード）の分割方法を、親ノードが共通の同じcbtDepthの符号化ノード（兄ノード）の分割方法（水平分割または垂直分割）とは異なる分割方法に制限する態様を説明する。図２４に、BT分割の分割方法の一例を示す。

[0199] 本態様では、図２４の（ａ）に示すように、兄ノードの分割方法が水平分割の場合は、弟ノードの分割方法は垂直分割に制限される。一方、図２４の（ｂ）に示すように、兄ノードの分割方法が垂直分割の場合は、弟ノードの分割方法は水平分割に制限される。

[0200] 本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図２５を参照して詳細に説明する。図２５は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0201] この場合、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0202] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth,cbtDepth, splitConstrain)を復号する。splitConstrainの初期値は０である。

[0203] CT情報復号部１０は、所定条件を満たさない場合、BT分割モードsplitBtMode[x0][y0][cbtDepth]を０として導出する。

[0204] 一方、CT情報復号部１０は、所定条件を満たす場合には、シンタックス要素であるBT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0205] CT情報復号部１０は、splitConstrainが０である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号し、復号したBT分

割モードsplit_bt_mode[x0][y0][cbtDepth]を対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]として設定する。

[0206] 一方、CT情報復号部１０は、splitConstrainが０以外（＝１，２）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0][cbtDepth]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出する。ここで、BT分割モードsplitBtMode[x0][y0][cbtDepth]の導出テーブルは、図２３に示した通りである。

[0207] 図２３に示したように、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0][cbtDepth]が定まる。

[0208] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth +1]が０以外（＝１，２）の場合は、splitConstrain = 3 - splitBtMode[x0][y0][cbtDepth + 1]、つまりsplitConstrainを兄ノードとは異なる分割方向とし（図２４の（ａ））、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０の場合は、splitConstrain = 0とする。

[0209] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０以外（＝１，２）の場合は、splitConstrain = 3 - splitBtMode[x0][y0][cbtDepth + 1]とし（図２４の（ｂ））、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０の場合は、splitConstrain = 0とする。

[0210] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0211] 上記同様、分割情報のシンタックスをsplit_bt_flagとsplit_bt_modeの２つに分けることなく１つのシンタックスsplit_bt_modeを用い、バイナリ表現から両者を区別しても構わない。

[0212] （BT情報復号処理－３）

続いて、BTツリーにおいて、親ノードが共通の同じcbtDepthの符号化ノード（兄ノード）がBT分割されている場合には、親ノードが共通の同じcbtDepthの他の符号化ノード（弟ノード）のBT分割を禁止する態様を説明する。図２６に、BT分割の分割方法の一例を示す。

[0213] 本態様では、図２６の（ａ）に示すように、兄ノードがBT分割されている場合は、弟ノードのBT分割は禁止される。一方、図２６の（ｂ）に示すように、兄ノードがBT分割されていない場合は、弟ノードのBT分割は禁止されない。

[0214] 本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図２７を参照して詳細に説明する。図２７は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0215] この場合、画像符号化装置１１においては、BT分割を許可するか否かを示すsplitConstrainを導出する。splitConstrainが０の場合には、BT分割を許可することを表し、splitConstrainが１の場合には、BT分割を禁止することを表す。

[0216] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth,cbtDepth, splitConstrain)を復号する。splitConstrainの初期値は０である。

[0217] CT情報復号部１０は、所定条件を満たさない場合、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]を０として導出する。

[0218] 一方、CT情報復号部１０は、所定条件を満たし、且つ、splitConstrain が０の場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号する。

[0219] CT情報復号部１０は、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth+ 1]が０以外（＝１，２）の場合は、splitConstrain = 1とし（図２６の（ａ））、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０の場合は、splitConstrain= 0とする。

[0220] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０以外（＝１，２）の場合は、splitConstrain = 1とし（図２６の（ｂ））、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０の場合は、splitConstrain= 0とする。

[0221] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0222] （第３の実施形態）

本実施形態では、BT分割の分割方向を制限する、すなわちCTUの符号化ノードを復号する際、復号済の他の符号化ノードにおけるBT分割の分割方法を用いてBT分割を行う、さらに他の具体的な態様として、BTツリーにおいて、親ノードと、親ノードが共通の同じcbtDepthの符号化ノード（兄ノード）との分割方法（水平分割または垂直分割）が同じである場合、当該分割方法に応じて、親ノードが共通の同じcbtDepthの他の符号化ノード（弟ノード）の分

割方法を制限する態様を示す。

[0223] 上記の態様の具体例としては、親ノードと兄ノードとの分割方法が同じである場合、弟ノードの分割方法を親ノード及び兄ノードの分割方法と同じ分割方法に制限する態様、及び弟ノードの分割方法を親ノード及び兄ノードの分割方法とは異なる分割方法に制限する態様が挙げられる。ここで、親ノードが共通の同じcbtDepthの符号化ノードのうち、先に復号される符号化ノードが兄ノードであり、後に復号される符号化ノードが弟ノードである。

[0224] まず、BTツリーにおいて、弟ノードの分割方法を、親ノードと兄ノードの分割方法（水平分割または垂直分割）に制限する態様を以下に説明する。図２８に、BT分割の分割方法の一例を示す。

[0225] 本態様では、図２８の（ａ）に示すように、親ノードの分割方法と兄ノードの分割方法とが同じ水平分割の場合は、弟ノードの分割方法も水平分割に制限される。一方、図２８の（ｂ）に示すように、親ノードの分割方法と兄ノードの分割方法とが同じ垂直分割の場合は、弟ノードの分割方法も垂直分割に制限される。

[0226] （BT情報復号の処理－１）

本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図２９を参照して詳細に説明する。図２９は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0227] この場合、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0228] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth,cbtDepth, splitConstrain)を復号する。splitConstrainの初期値は０である。

[0229] CT情報復号部１０は、所定条件を満たさない場合、BT分割モードsplitBtMo

de[x0][y0][cbtDepth]を０として導出する。

[0230] 一方、CT情報復号部１０は、所定条件を満たす場合には、splitConstrainを導出する。

[0231] CT情報復号部１０は、splitConstrainが０である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]として設定する。

[0232] 一方、CT情報復号部１０は、splitConstrainが０以外（＝１，２）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出する。ここで、BT分割モードsplitBtMode[x0][y0][cbtDepth]の導出テーブルは図２３に示した通りである。

[0233] 図２３に示したように、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0][cbtDepth]が定まる。

[0234] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１、すなわち対象CT（弟ノードから見た親ノード）の分割方法が水平分割の場合は、CT階層cbtDepth + 1の位置(x0, y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１、すなわち兄ノードの分割方法も水平分割の場合は、splitConstrain = 1、すなわち弟ノードの分割方法を水平分割に制限し（図２８の（ａ））、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１以外（＝０，２）の場合は、splitConstrain = 0とする。

[0235] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２、すなわち対象CT（弟ノードから見た親ノード）の分割方法が垂直分割の場合は、CT情報復

号部１０は、CT階層cbtDepth+ 1の位置(x0, y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２、すなわち兄ノードの分割方法も垂直分割の場合は、splitConstrain = 2、すなわち弟ノードの分割方法を垂直分割に制限し（図２８の（ｂ））、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２以外（＝０，１）の場合は、splitConstrain = 0とする。

[0236] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0237] （BT情報復号の処理－２）

続いて、親ノードが共通の同じcbtDepthの他の符号化ノード（弟ノード）の分割方法を、親ノードと、親ノードが共通の同じcbtDepthの符号化ノード（兄ノード）との分割方法（水平分割または垂直分割）とは異なる分割方法に制限する態様を以下に説明する。図３０に、BT分割の分割方法の一例を示す。

[0238] 本態様では、図３０の（ａ）に示すように、親ノードの分割方法と兄ノードの分割方法とが同じ水平分割の場合は、弟ノードの分割方法は垂直分割に制限される。一方、図３０の（ｂ）に示すように、親ノードの分割方法と兄ノードの分割方法とが同じ垂直分割の場合は、弟ノードの分割方法は水平分割に制限される。

[0239] 本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図３１を参照して詳細に説明する。図３１は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0240] この場合、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0241] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化

ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth,cbtDepth, splitConstrain)を復号する。splitConstrainの初期値は０である。

[0242] CT情報復号部１０は、所定条件を満たさない場合、BT分割モードsplitBtMode[x0][y0][cbtDepth]を０として導出する。

[0243] 一方、CT情報復号部１０は、所定条件を満たす場合には、シンタックス要素であるBT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0244] CT情報復号部１０は、splitConstrainが０である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]として設定する。

[0245] 一方、CT情報復号部１０は、splitConstrainが０以外（＝１，２）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出する。ここで、BT分割モードsplitBtMode[x0][y0][cbtDepth]の導出テーブルは図２３に示した通りである。

[0246] 図２３に示したように、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0][cbtDepth]が定まる。

[0247] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１、すなわち対象CT（弟ノードから見た親ノード）の分割方法が水平分割の場合は、CT階層cbtDepth + 1の位置(x0, y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１、すなわち兄ノードの

分割方法も水平分割の場合は、splitConstrain = 2、すなわち弟ノードの分割方法を垂直分割に制限し（図３０の（ａ））、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１以外（＝０，２）の場合は、splitConstrain = 0とする。

[0248] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２、すなわち対象CT（弟ノードから見た親ノード）の分割方法が垂直分割の場合は、CT情報復号部１０は、CT階層cbtDepth+ 1の位置(x0, y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２、すなわち兄ノードの分割方法も垂直分割の場合は、splitConstrain = 1、すなわち弟ノードの分割方法を水平分割に制限し（図３０の（ｂ））、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２以外（＝０，１）の場合は、splitConstrain = 0とする。

[0249] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0250] 上記同様、分割情報のシンタックスをsplit_bt_flagとsplit_bt_modeの２つに分けることなく１つのシンタックスsplit_bt_modeを用い、バイナリ表現から両者を区別しても構わない。

[0251] （第４の実施形態）

BT分割の分割方法は、対象CTのアスペクト比に応じて制限してもよい。本実施形態では、対象CTのアスペクト比（高さ／幅）が横長に大きい、すなわち対象CTの幅が所定の値以上（例えば、width>=8*height）の場合は、アスペクト比が１：１６となる水平分割を禁止して垂直分割のみに制限し、対象CTの縦長アスペクト比が縦長に大きい、すなわち対象CTの高さが所定の値以上（例えば、height>=8*width）の場合は、アスペクト比が１６：１となる垂直分割を禁止して水平分割のみに制限する態様を示す。

[0252] なお、アスペクト比制限がある場合には上記所定の値の範囲を超えること

はないため、対象CTの幅が所定の値（例えば、width==8*height）の場合は垂直分割のみに制限し、対象CTの高さが所定の値（例えば、height==8*width）の場合は水平分割のみに制限するとしても同じことである。

[0253] （BT情報復号の処理－１）

本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図３２を参照して詳細に説明する。図３２は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0254] この場合、画像符号化装置１１においては、各符号化ツリーCTにおいて、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割するか否かを示すBT分割フラグsplit_bt_flag[x0][y0]を符号化する。

[0255] CT情報復号部１０は、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth,cbtDepth, splitConstrain)を復号する。splitConstrainの初期値は０である。

[0256] CT情報復号部１０は、所定条件を満たさない場合、BT分割モードsplitBtMode[x0][y0]を０として導出する。

[0257] 一方、CT情報復号部１０は、所定条件を満たす場合には、シンタックス要素であるBT分割モードsplit_bt_mode[x0][y0][cbtDepth]またはBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0258] CT情報復号部１０は、splitConstrainが０である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0]を対象CTのBT分割モードsplitBtMode[x0][y0]として設定する。

[0259] 一方、CT情報復号部１０は、splitConstrainが０以外（＝１，２）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0]を導出する。ここで、BT分割モードsplitB

tMode[x0][y0]の導出テーブルは図２３に示した通りである。

[0260] 図２３に示したように、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]それぞれの値に応じて、BT分割モードsplitBtMode[x0][y0]が定まる。

[0261] CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、対数高さlog2CbHeight- 1である２つの符号化ツリーCTを復号する。この際、対象CTの幅が高さの４倍以上（log2CbWidth >= log2CbHeight+2）、すなわち水平分割後のCTの幅が高さの８倍以上の場合は、CTをさらに分割するとアスペクト比が１６：１となる。したがって、CTをさらに分割する場合は垂直分割のみに制限するためsplitConstrain = 2とし、それ以外の場合は、splitConstrain= 0とする。

[0262] 一方、BT分割モードsplit_bt_mode[x0][y0]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、対数幅log2CbWidth-1である２つの符号化ツリーCTを符号化する。この際、対象CTの高さが幅の４倍以上（log2CbHeight >= log2CbWidth+2）、すなわち垂直分割後のCTの高さが幅の８倍以上の場合は、CTをさらに分割するとアスペクト比が１：１６となる。したがって、CTをさらに分割する場合は水平分割のみに制限するためsplitConstrain = 1とし、それ以外の場合は、splitConstrain= 0とする。

[0263] なお、log2CbWidth >=log2CbHeight+2、log2CbHeight >= log2CbWidth+2の判定はそれぞれ、log2CbWidth == log2CbHeight+2、log2CbHeight ==log2CbWidth+2でもよい。

[0264] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0265] （BT情報復号の処理－２）

上述した対象CTのアスペクト比に応じたBT分割の制限は、スライスタイプがIスライスである場合に行ってもよい。以下では、スライスタイプがIスライスである場合に、アスペクト比に応じたBT分割の制限を行う形態を説明する。

[0266] 本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図

３３を参照して詳細に説明する。図３３は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図３２に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0267] CT情報復号部１０がBT分割モードsplitBtMode[x0][y0]を導出するまでの処理は、図３２のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割モードsplitBtMode[x0][y0]を導出した後、本態様では、CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、対象CTの幅が高さの４倍以上、すなわち水平分割後のCTの幅が高さの８倍以上の場合は、CTをさらに分割するとアスペクト比が１６：１となる。したがって、対象CTの幅が高さの４倍以上、且つ、スライスタイプがIスライスである場合（log2CbWidth >= log2CbHeight+2) && slice_type==I_SLICE）は、splitConstrain = 2とし、それ以外の場合は、splitConstrain= 0とする。

[0268] 一方、BT分割モードsplit_bt_mode[x0][y0]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを符号化する。この際、対象CTの高さが幅の４倍以上、すなわち垂直分割後のCTの高さが幅の８倍以上の場合は、CTをさらに分割するとアスペクト比が１：１６となる。したがって、対象CTの高さが幅の４倍以上、且つ、スライスタイプがIスライスである場合（log2CbHeight >= log2CbWidth+2) && slice_type==I_SLICE）は、splitConstrain = 1とし、それ以外の場合は、splitConstrain = 0とする。

[0269] 上記でもlog2CbWidth>= log2CbHeight+2、log2CbHeight >= log2CbWidth+2の判定はそれぞれ、log2CbWidth == log2CbHeight+2、log2CbHeight ==log2CbWidth+2でもよい。

[0270] CT情報復号部１０は、下位の符号化ツリーCTにおいても、同様の処理を行ってBT情報復号を継続する。

[0271] なお、以上ではアスペクト比が１：１６または１６：１となるBT分割を禁止する態様を示したが、本実施形態はこれに限定されない。例えば、アスペクト比が１：８または８：１となるBT分割を禁止する態様であってもよい。

[0272] また、以上では２つのノードともアスペクト比に応じてBT分割の制限する構成を示したが、本実施形態はこれに限定されない。例えば、兄あるいは弟ノードのみにおいてもアスペクト比に応じてBT分割の制限を行ってもよい。

[0273] （第５の実施形態）

上述した第１～第３の実施形態において、符号化／復号の複雑度を低減しながらも、符号化効率は低下しないことが好ましい。そこで、本実施形態では、BT深度に応じてBT分割の制限を行う。

[0274] 本実施形態では、BT深度に応じてBT分割の制限を行うことにより、BT分割パターンが制限されることに起因する符号化効率の低減を抑制することができる。したがって、本実施形態では、符号化効率への影響を押さえつつ、符号化／復号の複雑度を低減することができる。

[0275] 具体的な態様として、BT深度が閾値よりも大きい場合に第１～第３の実施形態におけるBT分割の制限を行う態様、及びBTツリーにおけるTemporal ID（時間階層、予測階層）ごとに設定した最大BT深度max_bt_depthに応じてBT分割の制限を行う態様が挙げられる。Temporal IDは、ピクチャ間の参照の深さを示す値であり、各ピクチャに割り振られる。あるTemporal IDがTID_currのピクチャの符号化・復号する場合、Temporal IDがTID_curr以下の参照ピクチャのみを参照可能であるという制限を持つ。Temporal IDが小さいピクチャの方がより多くの後続のピクチャで参照されることが多いため、符号化効率上、重要であるという性質を有する。また、Temporal IDを用いると、ピクチャを間欠的に再生する際に、特定のTemporalID以下のピクチャに限定して再生することで高速再生が可能である。

[0276] まず、BT深度が閾値よりも大きい場合に、第１～第３の実施形態におけるBT分割の制限を行う態様を以下に説明する。

[0277] （BT情報復号の処理－１）

はじめに、BT深度が閾値よりも大きい場合に、図１５のシンタックス表を用いたBT分割の制限を行う態様を説明する。図３４は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図１５に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0278] CT情報復号部１０がBT分割フラグsplit_bt_flag[x0][y0]を復号するまでの処理は、図１５のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割フラグsplit_bt_flag[x0][y0]を復号した後、本態様では、CT情報復号部１０は、cbtDepth <= THであり、且つ、BT分割フラグsplit_bt_flag[x0][y0]が０以外（＝１）である場合には、BT分割方向フラグsplit_dir_flag[x0][y0][cbtDepth]を復号する。ここで、THは、BT分割の制限を行う場合のBT深度の閾値である。

[0279] CT情報復号部１０は、復号したBT分割方向フラグsplit_dir_flag[x0][y0][cbtDepth]及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0]を導出する。これ以降の処理は、図１５のシンタックス表を用いた処理と同様である。

[0280] なお、CT情報復号部１０は、cbtDepth > THにおいては、BT分割フラグsplit_bt_flag[x0][y0]が０以外（＝１）である場合には、メモリに記憶されている親ノードのBT分割フラグsplit_dir_flag[x0][y0][cbtDepth-1]を対象CTのsplit_dir_flag[x0][y0][cbtDepth]として導出する。

[0281] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、対象CT（子ノード）の分割方法を親ノードの分割方法（水平分割または垂直分割）に制限する。

[0282] （BT情報復号の処理－２）

続いて、BT深度が閾値よりも大きい場合に、図１７のシンタックス表を用いたBT分割の制限を行う態様を説明する。図３５は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図１７に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は

省略する。

[0283] 本態様では、CT情報復号部１０は、splitConstrainが０、または、cbtDepth <= THである場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0]を対象CTのBT分割モードsplitBtMode[x0][y0]として設定する。これ以降の処理は、図１７のシンタックス表を用いた処理と同様である。

[0284] なお、CT情報復号部１０は、splitConstrainが０以外（＝１，２）、且つ、cbtDepth >THにおいては、BT分割フラグsplit_bt_flag[x0][y0]を復号し、splitConstrain及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0]を導出する。

[0285] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、対象CT（子ノード）の分割方法を親ノードの分割方法（水平分割または垂直分割）に制限する。

[0286] （BT情報復号の処理－３）

続いて、BT深度が閾値よりも大きい場合に、図１９のシンタックス表を用いたBT分割の制限を行う態様を説明する。図３６は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図１９に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0287] CT情報復号部１０は、所定条件を満たし、且つ、cbtDepth <= THである場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0]を復号し、復号したBT分割モードsplit_bt_mode[x0][y0]を対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]として設定する。

[0288] 一方、CT情報復号部１０は、所定条件を満たし、且つ、cbtDepth <= TH以外（＝cbtDepth > TH）である場合には、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号し、親ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth-1]及びBT分割フラグsplit_bt_flag[x0][y0]に基づき、対象CTのBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出する。これ以降の処理は、

図１９のシンタックス表を用いた処理と同様である。

[0289] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、対象CT（子ノード）の分割方法を親ノードの分割方法（水平分割または垂直分割）に制限する。

[0290] （BT情報復号の処理－４）

続いて、BT深度が閾値よりも大きい場合に、図２２のシンタックス表を用いたBT分割の制限を行う態様を説明する。図３７は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図２２に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0291] CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出するまでの処理は、図２２のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出した後、本態様では、CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）については、splitConstrain = 0とする。また、弟ノード（x0, y1）については、cbtDepth > THの場合は、splitConstrainを兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]とし、cbtDepth > TH以外（cbtDepth <= TH）の場合は、splitConstrain = 0とする。

[0292] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）については、splitConstrain =0とする。また、弟ノード（x1, y0）については、cbtDepth > THの場合は、splitConstrainを兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]とし、cbtDepth > TH以外（cbtDepth <= TH）の場合は、splitConstrain = 0とする。

[0293] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には

、弟ノードの分割方法を兄ノードの分割方法（水平分割または垂直分割）に制限する。

[0294] （BT情報復号の処理－５）

続いて、BT深度が閾値よりも大きい場合に、図２５のシンタックス表を用いたBT分割の制限を行う態様を説明する。図３８は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図２５に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0295] CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出するまでの処理は、図２５のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出した後、本態様では、CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０以外（＝１，２）、且つ、cbtDepth > THの場合は、splitConstrain = 3 - splitBtMode[x0][y0][cbtDepth + 1]とし、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０、または、cbtDepth> TH以外（cbtDepth <= TH）の場合は、splitConstrain = 0とする。

[0296] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０以外（＝１，２）、且つ、cbtDepth > THの場合は、splitConstrain = 3-splitBtMode[x0][y0][cbtDepth + 1]とし、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が０、または、cbtDepth> TH以外（cbtDepth <= TH）の場合は、s

plitConstrain = 0とする。

[0297] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、弟ノードの分割方法を兄ノードの分割方法（水平分割または垂直分割）とは異なる分割方法に制限する。

[0298] （BT情報復号の処理－５）

続いて、BT深度が閾値よりも大きい場合に、図２７のシンタックス表を用いたBT分割の制限を行う態様を説明する。図３９は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図２７に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0299] CT情報復号部１０がBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を導出するまでの処理は、図２７のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割モードsplit_bt_mode[x0][y0][cbtDepth]を導出した後、本態様では、CT情報復号部１０は、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０以外（＝１，２）、且つ、cbtDepth > THの場合は、splitConstrain = 1とし、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０、または、cbtDepth > TH以外（cbtDepth <= TH）の場合は、splitConstrain = 0とする。

[0300] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０以外（＝１，２）、且つ、cbtDepth > THの場合は、splitConstrain =1とし、兄ノード

のBT分割モードsplit_bt_mode[x0][y0][cbtDepth + 1]が０、または、cbtDepth <= THの場合は、splitConstrain = 0とする。

[0301] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、兄ノードがBT分割されていれば弟ノードのBT分割を禁止する。

[0302] （BT情報復号の処理－６）

続いて、BT深度が閾値よりも大きい場合に、図２９のシンタックス表を用いたBT分割の制限を行う態様を説明する。図４０は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図２９に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0303] CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出するまでの処理は、図２９のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出した後、本態様では、CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１、且つ、cbtDepth > THの場合は、splitConstrain = 1とし、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１以外（＝０，２）、または、cbtDepth <= THの場合は、splitConstrain = 0とする。

[0304] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを符号化する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２、且つ、cbtDepth > THの場合は、splitConstrain = 2とし、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２以外（＝０，１）、または、cbtDepth

<= THの場合は、splitConstrain = 0とする。

[0305] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、弟ノードの分割方法を、親ノードと兄ノードとの分割方法（水平分割または垂直分割）に制限する。

[0306] （BT情報復号の処理－７）

続いて、BT深度が閾値よりも大きい場合に、図３１のシンタックス表を用いたBT分割の制限を行う態様を説明する。図４１は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。なお、以下では、図３１に示したシンタックス表と異なる部分のみを説明し、他の部分の説明は省略する。

[0307] CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出するまでの処理は、図３１のシンタックス表を用いた処理と同様である。CT情報復号部１０がBT分割モードsplitBtMode[x0][y0][cbtDepth]を導出した後、本態様では、CT情報復号部１０は、BT分割モードsplitBtMode[x0][y0][cbtDepth]が１の場合は、CT階層cbtDepth + 1の位置(x0,y0)、(x0, y1)で、log2CbHeight - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x0, y1）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１、且つ、cbtDepth > THの場合は、splitConstrain = 2とし、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が１以外（＝０，２）、または、cbtDepth > TH以外（cbtDepth <= TH）の場合は、splitConstrain = 0とする。

[0308] 一方、BT分割モードsplit_bt_mode[x0][y0][cbtDepth]が２の場合は、CT情報復号部１０は、CT階層cbtDepth + 1の位置(x0,y0)、(x1, y0)で、log2CbWidth - 1である２つの符号化ツリーCTを復号する。この際、兄ノード（x0, y0）についてはsplitConstrain = 0とし、弟ノード（x1, y0）については、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２、且つ、cbtDepth > THの場合は、splitConstrain = 1とし、兄ノードのBT分割モードsplitBtMode[x0][y0][cbtDepth + 1]が２以外（＝０，１）、または、cbtDepth >

TH以外（cbtDepth <= TH）の場合は、splitConstrain = 0とする。

[0309] このように、本態様では、対象CTのBT深度が閾値THよりも大きい場合には、弟ノードの分割方法を、親ノードと兄ノードとの分割方法（水平分割または垂直分割）とは異なる分割方法に制限する。

[0310] （BT分割の制限）

以下には、BTツリーにおけるTemporal ID（時間階層、予測階層）ごとに設定した最大BT深度max_bt_depthに応じてBT分割の制限を行う態様を説明する。

[0311] 本実施形態では、Temporal IDごとに最大BT深度max_bt_depthを設定している。この最大BT深度max_bt_depthには、デフォルト値から各Temporal IDに応じたオフセット値（例えば、Temporal IDそのもの）を引いた値を設定する。このデフォルト値は、Temporal IDの最大値以上の値を任意に選択することが出来る。また、オフセット値はデフォルト値以下の値の中から任意に選択することができる。

[0312] 図４２に、BTツリーにおけるTemporal IDごとの最大BT深度max_bt_depthの設定例を示す。本図においては、デフォルト値 = 3とすると、Temporal ID = 0においては、max_bt_depth = デフォルト値 - Temporal ID = 3となる。すなわち、Temporal ID = 0においては、BT分割は３回実施できることになる。一方、Temporal ID = 1においては、max_bt_depth = デフォルト値 - Temporal ID = 2となるので、BT分割は２回実施できることになり、Temporal ID = 2においては、max_bt_depth = デフォルト値 - Temporal ID = 1となるので、BT分割は１回実施できることになる。Temporal ID = 3においては、max_bt_depth = デフォルト値 - Temporal ID = 0となるので、BT分割は実施できず、BTツリーの最後の階層となる。

[0313] このように、各Temporal IDに最大BT深度max_bt_depthを設定することにより、BT分割の階層を制限することができる。

[0314] （シーケンスパラメータセット（SPS））

以上のCT情報復号を行うために画像復号装置３１が参照する符号化パラメ

ータ（CTUサイズ、最小QTサイズ、最大BTサイズ、最大BT深度等）は、シーケンスパラメータセット（SPS）を用いて伝送される。図４３にSPSのためのシンタックス構造例を示す。

[0315] 図４３に示すように、SPSでは、CT情報復号を行うために画像復号装置３１が参照する符号化パラメータの集合が規定されている。ここで、本図から分かるように、最小QTサイズ（log2_minQT_...）、最大BTサイズ（log2_max_bt_size_...）、及び最大BT深度（max_bt_depth_...）については、Iスライスの輝度用（Luma）、Iスライスの色差用（Chroma）、その他（P,Bスライス）用それぞれについて定義されている。

[0316] 上述したBTツリーにおけるTemporal IDごとの最大BT深度max_bt_depthについても、SPSにおいて符号化することができる。図中では、Temporal IDごとの最大BT深度を示すパラメータとして、Iスライスの輝度用の最大BT深度max_bt_depth_i_slice_luma_minus2[i]、Iスライスの色差用の最大BT深度max_bt_depth_i_slice_chroma_minus2[i]、その他（P,Bスライス）用の最大BT深度max_bt_depth_minus2[i]が符号化されている。iは対応するTemporal IDを示している。

[0317] 例えば、Temporal IDが０の時の最大BT深度を４、Temporal IDが１～３の時の最大BT深度を３、Temporal IDが４の時の最大BT深度を２に設定したい場合は、SPSの最大BT深度を示すパラメータを以下のように設定する。

[0318]
```
max_bt_depth_i_slice_luma_minus2[0]= 4
max_bt_depth_i_slice_chroma_minus2[0]= 4
max_bt_depth_minus2[0]= 4
max_bt_depth_i_slice_luma_minus2[1]= 3
max_bt_depth_i_slice_chroma_minus2[1]= 3
max_bt_depth_minus2[1]= 3
max_bt_depth_i_slice_luma_minus2[2]= 3
max_bt_depth_i_slice_chroma_minus2[2]= 3
max_bt_depth_minus2[2]= 3
```

max_bt_depth_i_slice_luma_minus2[3]= 3

max_bt_depth_i_slice_chroma_minus2[3]= 3

max_bt_depth_minus2[3]= 3

max_bt_depth_i_slice_luma_minus2[4]= 2

max_bt_depth_i_slice_chroma_minus2[4]= 2

max_bt_depth_minus2[4]= 2

（エンコーダ制約）

以下には、符号化パラメータ決定部１１０における、各Temporal IDに設定した閾値であるスキップデプスSKIP_DEPTHに応じて分割の制限を行う態様を説明する。符号化パラメータ決定部１１０は、ＣＵをさらに分割した符号化パラメータのセットのコストを算出するか否か、即ち、ＣＵのさらなる分割を制限するか否かを、分割階層（例えば、CT階層やBT深度）に関する閾値であるスキップデプスSKIP_DEPTHに基づいて判定する。本実施形態では、スキップデプスSKIP_DEPTHをTemporal IDごとに設定する。

[0319] 例えば、Temporal IDが所定の閾値より小さい（０、１又は２）の場合には、符号化パラメータ決定部１１０は、スキップデプスSKIP_DEPTHを３に設定する。そして、ＣＵのBT分割回数BTDepthが３回以上であれば（BTDepth >= SKIP_DEPTH (=3)）、符号化パラメータ決定部１１０は、ＣＵのさらなる分割を制限する、つまり、符号化パラメータ決定部１１０は、コストを算出しない。また、Temporal IDが所定の閾値以上（３又は４）の場合には、符号化パラメータ決定部１１０は、スキップデプスSKIP_DEPTHを２に設定する。そして、ブロックのBT分割回数が２回以上であれば（BTDepth >= SKIP_DEPTH (=2)）、符号化パラメータ決定部１１０は、ＣＵのさらなる分割を制限する。

[0320] なお、ＣＵのさらなる分割を制限するのは、分割前のＣＵに対しコスト値が最小となる符号化パラメータのセットにおける予測モードがスキップモードである場合に限ってもよい。

[0321] このように、各Temporal IDにスキップデプスSKIP_DEPTHを設定することにより、コストを算出する符号化パラメータのセット数が減少し、符号化に係

る処理量が削減されるという効果を奏する。

（エンコーダ制約２）

続いて、符号化パラメータ決定部１１０における、符号化対象画像と参照画像の距離ｄに応じてスキップデプスSKIP_DEPTHを設定する態様を説明する。

[0322] 符号化パラメータ決定部１１０は、距離ｄが閾値ＴＨ以下か否かに応じて分割階層の閾値であるスキップデプスSKIP_DEPTHを設定する。具体的には、距離ｄが閾値ＴＨ以下である場合、つまり、符号化対象画像と参照画像が近い場合のスキップデプスSKIP_DEPTHは、距離ｄが閾値ＴＨより大きい場合、つまり、符号化対象画像と参照画像が遠い場合のスキップデプスSKIP_DEPTHより小さい値を設定する。例えば、閾値ＴＨを２と設定した場合、符号化対象画像と参照画像の距離ｄが２以下の場合（ｄ<=ＴＨが成立する場合）には、符号化パラメータ決定部１１０は、スキップデプスSKIP_DEPTHを２に設定する。また、距離ｄが３以上の場合（ｄ<=ＴＨが成立しない場合）には、符号化パラメータ決定部１１０は、スキップデプスSKIP_DEPTHを３に設定する。上記のより、参照画像が近い場合には、スキップデプスSKIP_DEPTHを比較的小さく設定することにより、分割階層が小さくなるように制限される。これにより、符号化に係る処理量が削減される。

[0323] ここで、符号化対象画像と参照画像の距離を算出する際に、対象画像が参照する参照画像が複数ある場合には、参照画像のうちで最も距離の小さい場合の値を距離とする。即ち、符号化対象画像と参照画像の距離ｄとは、ＣＵを含む対象画像と、ＣＵの参照ピクチャインデックスrefIdxL0、refIdxL1が示す参照画像との距離のうち値が小さい方である。また、距離の導出に用いる参照画像は、対象ＣＵが直接参照する画像（参照ピクチャインデックスrefIdxL0、refIdxL1が示す参照画像）だけではなく、対象画像が参照しうる参照画像のセット全体（例えば、参照ピクチャリストに含まれる参照画像）を含めてもよい。例えば、図４（ｂ）においては、対象画像Ｂ３上の対象ブロックが、参照ピクチャインデックスrefIdxL0、refIdxL1としてP1,B2を参照する

場合においても、対象画像Ｂ３が参照しうる画像I0についても距離を算出し、それらのうち最小の値を距離ｄとしてもよい。距離の算出方法としては、画像の表示時刻であるPOCの差分絶対値を求める方法がある。例えば、図４（ａ）においては、P1とI0の距離は|4 - 0|=4となり、B2とP1の距離は|2 - 4|=2となる。

[0324] なお、スキップデプスSKIP_DEPTHと対象ＣＵの分割階層に応じてＣＵの分割を制限する方法としては、ＣＵの分割階層(例えば、BT階層)がスキップデプスSKIP_DEPTH以上の場合に常に分割しない方法に限定されない。例えば、対象ＣＵの親のＣＵである参照ＣＵの予測モード（分割前のＣＵである参照ＣＵに対しコスト値が最小となる符号化パラメータのセットにおける予測モード）がスキップモードである場合に限ってもよい。すなわち、ＣＵの分割階層(例えば、BT階層)がスキップデプスSKIP_DEPTH以上、かつ、参照ＣＵの予測モードがスキップモードの場合に限定して、コスト導出を行わず、これ以上分割しないことを選択してもよい。

[0325] このように、符号化対象画像と参照画像の距離ｄに応じてスキップデプスSKIP_DEPTHを設定することにより、コストを算出する符号化パラメータのセット数が減少し、符号化に係る処理量が削減されるという効果を奏する。

[0326] （第６の実施形態）

上述した第１～第３の実施形態において、符号化／復号の複雑度を低減しながらも、符号化効率は低下しないことが好ましい。そこで、本実施形態では、対象CTサイズに応じてBT分割の制限を行う。具体的には、対象CTサイズ（1 << log2CbWidth、1 << log2CbHeight）が閾値THよりも小さい場合に第１～第３の実施形態におけるBT分割の制限を行う。

[0327] 本実施形態では、対象CTサイズに応じてBT分割の制限を行うことにより、符号化／復号効率への影響が大きい大サイズの対象CTにおいてはBT分割の制限を加えないため、符号化／復号効率の低減を抑制することができる。

[0328] 図３４～図４１のシンタックス表を用いた処理では、ビット深度（cbtDepth）が閾値よりも大きい場合にBT分割の制限を行っていたが、本実施形態では

、図３４～図４１のシンタックス表を用いた処理において、対象CTサイズ（1 << log2CbWidth、1 << log2CbHeight）が閾値THよりも小さい場合にBT分割の制限を行う。つまり、図３４～図３６における「cbtDepth <= TH」を「(1 << log2CbWidth) >= TH && (1 <<log2CbHeight) >= TH」に置き換え、図３７～図４１における「cbtDepth >TH」を「(1 << log2CbWidth) < TH || (1 <<log2CbHeight) < TH」に置き換えて処理を実行すればよい。

[0329] （第７の実施形態）

上述した第１～第３の実施形態において、符号化／復号の複雑度を低減しながらも、符号化効率は低下しないことが好ましい。そこで、本実施形態では、スライスタイプに応じてBT分割の制限を行う。具体的には、スライスタイプがIスライス以外（すなわち、B,Pスライス）の場合に第１～第３の実施形態におけるBT分割の制限を行う。

[0330] 本実施形態では、スライスタイプに応じてBT分割の制限を行うことにより、BT分割パターンが複雑なIスライスにおいてはBT分割の制限を加えないため、符号化効率の低減を抑制することができる。

[0331] 図３４～図４１のシンタックス表を用いた処理では、ビット深度（cbtDepth）が閾値よりも大きい場合にBT分割の制限を行っていたが、本実施形態では、図３４～図４１のシンタックス表を用いた処理において、スライスタイプがIスライス以外（すなわち、B,Pスライス）の場合にBT分割の制限を行う。つまり、図３４～図３６における「cbtDepth <= TH」を「slice_type == I_SLICE」に置き換え、図３７～図４１における「cbtDepth> TH」を「slice_type != I_SLICE」に置き換えて処理を実行すればよい。

[0332] （第８の実施形態）

Iスライスにおいては、輝度及び色差それぞれのQTBTツリーを持つため、符号化／復号の複雑度を低減することが好ましい。そこで、本実施形態では、Iスライスの輝度のBT分割に応じて、Iスライスの色差のBT分割を制限する。

[0333] 本実施形態では、Iスライスの輝度のBT分割に応じて、Iスライスの色差のBT分割を制限するため、色差のQTBTツリーのQTBTの分割パターンが制限される

。これにより、IスライスのQTBT分割に関する符号化／復号の複雑度を低減することができる。

[0334] なお、画像復号装置３１のCT情報復号部１０では、Iスライスの輝度のCT情報復号を行った後、Iスライスの色差のCT情報復号を行う。このCT情報復号部１０の動作を説明するフローチャートを図４４に示す。

[0335] CT情報復号部１０によるCTU情報復号Ｓ１３００では、CT情報復号部１０は、isLumaを１に設定する（Ｓ１３１０）。isLumaは、輝度及び色差のいずれを処理対象としているのかを示す変数であり、isLumaが１の場合は輝度が処理対象であり、isLumaが０の場合は色差が処理対象である。

[0336] 続いて、CT情報復号部１０は、CT情報復号Ｓ１４００を行う。このCT情報復号Ｓ１４００は、図１１に示したものと同じである。ここで、スライスタイプがIスライスである場合は、isLumaが１に設定されているため、CT情報復号部１０は、輝度についてCT情報復号Ｓ１４００を行う。

[0337] スライスタイプがIスライス以外（すなわち、B,Pスライス）である場合（Ｓ１３２０，ＮＯ）は、このまま処理を終了する。一方、スライスタイプがIスライスである場合（Ｓ１３２０，ＹＥＳ）は、CT情報復号部１０は、isLumaを０に設定する。

[0338] CT情報復号部１０は、isLumaが０に設定されているため、色差についてCT情報復号Ｓ１４００を行う。このCT情報復号Ｓ１４００も図１１に示したものと同じである。

[0339] 本実施形態の具体的な態様として、Iスライスの色差のBT深度がIスライスの輝度のBT深度以下となるように、Iスライスの色差のBT分割を制限する態様、及びIスライスの色差のQTBTツリーをIスライスの輝度のQTBTツリーと同じ木構造にする態様が挙げられる。

[0340] まず、Iスライスの色差のBT深度がIスライスの輝度のBT深度以下となるように、Iスライスの色差のBT分割を制限する態様を以下に説明する。

[0341] （BT情報復号の処理－１）

本態様におけるCT情報復号部１０によるBT情報復号の動作例について、図

４５を参照して詳細に説明する。図４５は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0342] （Iスライスの輝度の処理）

CT情報復号部１０は、Iスライスの輝度について、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth)を復号する。

[0343] CT情報復号部１０は、Iスライスの輝度について、所定条件を満たさない場合には、BT分割モードsplit_bt_mode[x0][y0]を０として導出する。

[0344] 一方、CT情報復号部１０は、Iスライスの輝度について、所定条件を満たす場合は、符号化データからBT分割モードsplit_bt_mode[x0][y0]を復号する。

[0345] このようにして導出したBT分割モードsplit_bt_mode[x0][y0]に基づき、Iスライスの輝度のCTを復号する。

[0346] ここで、BT分割モードsplit_bt_mode[x0][y0]が１又は２ではない場合（即ちBT分割を行わない場合）、CT情報復号部１０は、処理対象がIスライスの輝度（＝isLumaが１）であるので、Iスライスの輝度のQTBTツリーにおけるQT深度cqtDepthL[x0][y0]及びBT深度cbtDepthL[x0][y0]それぞれに、cqtDepth及びcbtDepthを設定する。

[0347] CT情報復号部１０は、Iスライスの輝度について、下位の符号化ツリーCTにおいても同様の処理を行ってBT情報復号を継続する。

[0348] （Iスライスの色差の処理）

続いて、CT情報復号部１０は、Iスライスの色差について、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0,log2CbWidth, log2CbHeight, cqtDepth, cbtDepth)を復号する。

[0349] CT情報復号部１０は、Iスライスの色差について、所定条件を満たさない場合、または、QT深度及びBT深度の和（cqtDepth + cbtDepth）が、Iスライスの輝度のQTBTツリーにおけるQT深度及びBT深度の和（cqtDepthL[x0][y0] + c

btDepthL[x0][y0]）以上である場合には、BT分割モードsplit_bt_mode[x0][y0]を０として導出する。

[0350] 一方、CT情報復号部１０は、Iスライスの色差について、所定条件を満たし、且つ、QT深度及びBT深度の和（cqtDepth + cbtDepth）が、Iスライスの輝度のQTBTツリーにおけるQT深度及びBT深度の和（cqtDepthL[x0][y0] + cbtDepthL[x0][y0]）未満である場合には、符号化データからBT分割モードsplit_bt_mode[x0][y0]を復号する。

[0351] このようにして導出したBT分割モードsplit_bt_mode[x0][y0]に基づき、CT情報復号部１０は、Iスライスの色差のBT分割を行う。

[0352] （BT情報復号の処理－２）

続いて、Iスライスの色差のQTBTツリーをIスライスの輝度のQTBTツリーと同じ木構造にする態様を説明する。なお、本態様では、Iスライスの色差のQTBTツリーをIスライスの輝度のQTBTツリーと同じ木構造にする場合、分割方法は同一である必要があるが、分割回数は輝度のQTBTツリーに対して少なくてもかまわない。つまり、Iスライスの輝度のQTBTツリーにおいて途中でQT分割またはBT分割が終了しているようなQTBTツリーをIスライスの色差のQTBTツリーとしてもよい。

[0353] 本態様におけるCT情報復号部１０によるQT情報復号及びBT情報復号の動作例について、図４６及び図４７を参照して詳細に説明する。図４６は、本発明の一実施形態に係るQT情報のシンタックス表の構成例を示す図であり、図４７は、本発明の一実施形態に係るBT情報のシンタックス表の構成例を示す図である。

[0354] （Iスライスの輝度の処理）

CT情報復号部１０は、Iスライスの輝度について、符号化データからQT情報を復号し、再帰的に符号化ツリーCT（coding_quadtree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_quadtree(x0, y0,log2CbSize, cqtDepth)を復号する。

[0355] CT情報復号部１０は、QT分割フラグがない場合には、QT分割フラグsplit_c

u_flag[x0][y0]を０として導出する。

[0356] 一方、CT情報復号部１０は、QT分割フラグがある場合には、符号化データからQT分割フラグsplit_cu_flag[x0][y0]を復号する。

[0357] ここで、CT情報復号部１０は、処理対象がIスライスの輝度（＝isLumaが１）の場合、対象CTのQT分割フラグsplitCuFlag及びIスライスの輝度のQT分割フラグsplitCuFlagL[x0][y0]それぞれに、split_cu_flag[x0][y0]を設定する。

[0358] このようにして導出したQT分割フラグsplitCuFlagに基づき、CT情報復号部１０は、Iスライスの輝度のQT分割を行う。

[0359] 続いて、CT情報復号部１０は、Iスライスの輝度について、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0,log2CbWidth, log2CbHeight, cqtDepth, cbtDepth)を復号する。

[0360] CT情報復号部１０は、所定条件を満たさない場合には、BT分割モードsplit_bt_mode[x0][y0]を０として導出する。

[0361] 一方、CT情報復号部１０は、所定条件を満たす場合は、符号化データからBT分割モードsplit_bt_mode[x0][y0]を復号する。ここで、CT情報復号部１０は、処理対象がIスライスの輝度（＝isLumaが１）の場合、対象CTのBT分割モードsplitBtModeに、split_bt_mode[x0][y0]を設定する。

[0362] このようにして導出したBT分割モードsplitBtModeに基づき、Iスライスの輝度のBT分割を行う。

[0363] CT情報復号部１０は、Iスライスの輝度について、下位の符号化ツリーCTにおいても同様の処理を行ってCT情報復号を継続する。

[0364] （Iスライスの色差の処理）

次に、CT情報復号部１０は、Iスライスの色差について、符号化データからQT情報を復号し、再帰的に符号化ツリーCT（coding_quadtree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_quadtree(x0, y0,log2CbSize, cqtDepth)を復号する。

[0365] CT情報復号部１０は、QT分割フラグがない場合には、QT分割フラグsplit_cu_flag[x0][y0]を０として導出する。

[0366] 一方、CT情報復号部１０は、QT分割フラグがある場合には、符号化データからQT分割フラグsplit_cu_flag[x0][y0]を復号する。

[0367] ここで、CT情報復号部１０は、処理対象がIスライスの色差（＝isLumaが０）の場合、対象CTのQT分割フラグsplitCuFlagを、splitCuFlagL[対応する輝度のX座標][対応する輝度のY座標]とsplit_cu_flag[x0][y0]のANDに設定する。対応する輝度の座標とは、色差の座標に対応する輝度の座標である。例えば、Y:Cb:Cr=4：4：4の場合は、対応する輝度の座標は(x0,y0)であり、Y:Cb:Cr=4：2：2の場合は、対応する輝度の座標は(x0<<１,y0)であり、Y:Cb:Cr=4：2：0の場合は、対応する輝度の座標は(x0<<1,y0<<1)である。

[0368] このようにして導出したQT分割フラグsplitCuFlagに基づき、CT情報復号部１０は、Iスライスの色差のQT分割を行う。

[0369] 続いて、CT情報復号部１０は、Iスライスの色差について、符号化データからBT情報を復号し、再帰的に符号化ツリーCT（coding_binarytree）を復号する。具体的には、CT情報復号部１０は、対象符号化ツリーcoding_binarytree(x0, y0,log2CbWidth, log2CbHeight, cqtDepth, cbtDepth)を復号する。

[0370] CT情報復号部１０は、所定条件を満たさない場合には、BT分割モードsplit_bt_mode[x0][y0]を０として導出する。

[0371] 一方、CT情報復号部１０は、所定条件を満たす場合は、符号化データからBT分割フラグsplit_bt_flag[x0][y0]を復号する。

[0372] ここで、CT情報復号部１０は、処理対象がIスライスの色差（＝isLumaが０）の場合、対象CTのBT分割モードsplitBtModeを、BT分割フラグsplit_bt_flag[x0][y0]が１の場合は、split_bt_mode[対応する輝度のX座標][対応する輝度のY座標]とし、BT分割フラグsplit_bt_flag[x0][y0]が０の場合は、０として設定する。

[0373] このようにして導出したBT分割モードsplitBtModeに基づき、Iスライスの色差のBT分割を行う。

[0374] CT情報復号部１０は、Iスライスの色差について、下位の符号化ツリーCTにおいても同様の処理を行ってCT情報復号を継続する。

[0375] なお、以上では、Iスライスの色差のQTBTツリーをIスライスの輝度のQTBTツリーと同じ木構造にしている態様を示したが、本実施形態はこれに限定されない。例えば、Iスライスの色差のQTツリーをIスライスの輝度のQTツリーと同じ木構造にしてもよい。この場合のQT情報のシンタックス表の構成例を図４８に示す。図４８に示すように、CT情報復号部１０は、処理対象がIスライスの色差（＝isLumaが０）の場合、対象CTのQT分割フラグsplitCuFlagを、splitCuFlagL[対応する輝度のX座標][対応する輝度のY座標]に設定する。対応する輝度の座標とは、色差の座標に対応する輝度の座標である。このようにして導出したQT分割フラグsplitCuFlagに基づき、CT情報復号部１０は、Iスライスの色差のQT分割を行う。

[0376] あるいは、Iスライスの色差のBTツリーをIスライスの輝度のBTツリーと同じ木構造にしてもよい。

[0377] （ソフトウェアによる実現例）

なお、上述した実施形態における画像符号化装置１１、画像復号装置３１の一部、例えば、エントロピー復号部３０１、予測パラメータ復号部３０２、ループフィルタ３０５、予測画像生成部３０８、逆量子化・逆DCT部３１１、加算部３１２、予測画像生成部１０１、減算部１０２、DCT・量子化部１０３、エントロピー符号化部１０４、逆量子化・逆DCT部１０５、ループフィルタ１０７、符号化パラメータ決定部１１０、予測パラメータ符号化部１１１をコンピュータで実現するようにしても良い。その場合、この制御機能を実現するためのプログラムをコンピュータ読み取り可能な記録媒体に記録して、この記録媒体に記録されたプログラムをコンピュータシステムに読み込ませ、実行することによって実現しても良い。なお、ここでいう「コンピュータシステム」とは、画像符号化装置１１、画像復号装置３１のいずれかに内蔵されたコンピュータシステムであって、OSや周辺機器等のハードウェアを含むものとする。また、「コンピュータ読み取り可能な記録媒体」とは、フ

レキシブルディスク、光磁気ディスク、ROM、CD-ROM等の可搬媒体、コンピュータシステムに内蔵されるハードディスク等の記憶装置のことをいう。さらに「コンピュータ読み取り可能な記録媒体」とは、インターネット等のネットワークや電話回線等の通信回線を介してプログラムを送信する場合の通信線のように、短時間、動的にプログラムを保持するもの、その場合のサーバやクライアントとなるコンピュータシステム内部の揮発性メモリのように、一定時間プログラムを保持しているものも含んでも良い。また上記プログラムは、前述した機能の一部を実現するためのものであっても良く、さらに前述した機能をコンピュータシステムにすでに記録されているプログラムとの組み合わせで実現できるものであっても良い。

[0378] また、上述した実施形態における画像符号化装置１１、画像復号装置３１の一部、または全部を、LSI（Large Scale Integration）等の集積回路として実現しても良い。画像符号化装置１１、画像復号装置３１の各機能ブロックは個別にプロセッサ化しても良いし、一部、または全部を集積してプロセッサ化しても良い。また、集積回路化の手法はＬＳＩに限らず専用回路、または汎用プロセッサで実現しても良い。また、半導体技術の進歩によりＬＳＩに代替する集積回路化の技術が出現した場合、当該技術による集積回路を用いても良い。

[0379] （応用例）

上述した画像符号化装置１１及び画像復号装置３１は、動画像の送信、受信、記録、再生を行う各種装置に搭載して利用することができる。なお、動画像は、カメラ等により撮像された自然動画像であってもよいし、コンピュータ等により生成された人工動画像（CG及びGUIを含む）であってもよい。

[0380] まず、上述した画像符号化装置１１及び画像復号装置３１を、動画像の送信及び受信に利用できることを、図８を参照して説明する。

[0381] 図８の（ａ）は、画像符号化装置１１を搭載した送信装置PROD_Aの構成を示したブロック図である。図８の（ａ）に示すように、送信装置PROD_Aは、動画像を符号化することによって符号化データを得る符号化部PROD_A1と、符

号化部PROD_A1が得た符号化データで搬送波を変調することによって変調信号を得る変調部PROD_A2と、変調部PROD_A2が得た変調信号を送信する送信部PROD_A3と、を備えている。上述した画像符号化装置１１は、この符号化部PROD_A1として利用される。

[0382] 送信装置PROD_Aは、符号化部PROD_A1に入力する動画像の供給源として、動画像を撮像するカメラPROD_A4、動画像を記録した記録媒体PROD_A5、動画像を外部から入力するための入力端子PROD_A6、及び、画像を生成または加工する画像処理部Ａ７を更に備えていてもよい。図８の（ａ）においては、これら全てを送信装置PROD_Aが備えた構成を例示しているが、一部を省略しても構わない。

[0383] なお、記録媒体PROD_A5は、符号化されていない動画像を記録したものであってもよいし、伝送用の符号化方式とは異なる記録用の符号化方式で符号化された動画像を記録したものであってもよい。後者の場合、記録媒体PROD_A5と符号化部PROD_A1との間に、記録媒体PROD_A5から読み出した符号化データを記録用の符号化方式に従って復号する復号部（不図示）を介在させるとよい。

[0384] 図８の（ｂ）は、画像復号装置３１を搭載した受信装置PROD_Bの構成を示したブロック図である。図８の（ｂ）に示すように、受信装置PROD_Bは、変調信号を受信する受信部PROD_B1と、受信部PROD_B1が受信した変調信号を復調することによって符号化データを得る復調部PROD_B2と、復調部PROD_B2が得た符号化データを復号することによって動画像を得る復号部PROD_B3と、を備えている。上述した画像復号装置３１は、この復号部PROD_B3として利用される。

[0385] 受信装置PROD_Bは、復号部PROD_B3が出力する動画像の供給先として、動画像を表示するディスプレイPROD_B4、動画像を記録するための記録媒体PROD_B5、及び、動画像を外部に出力するための出力端子PROD_B6を更に備えていてもよい。図８の（ｂ）においては、これら全てを受信装置PROD_Bが備えた構成を例示しているが、一部を省略しても構わない。

[0386] なお、記録媒体PROD_B5は、符号化されていない動画像を記録するためのものであってもよいし、伝送用の符号化方式とは異なる記録用の符号化方式で符号化されたものであってもよい。後者の場合、復号部PROD_B3と記録媒体PROD_B5との間に、復号部PROD_B3から取得した動画像を記録用の符号化方式に従って符号化する符号化部（不図示）を介在させるとよい。

[0387] なお、変調信号を伝送する伝送媒体は、無線であってもよいし、有線であってもよい。また、変調信号を伝送する伝送態様は、放送（ここでは、送信先が予め特定されていない送信態様を指す）であってもよいし、通信（ここでは、送信先が予め特定されている送信態様を指す）であってもよい。すなわち、変調信号の伝送は、無線放送、有線放送、無線通信、及び有線通信の何れによって実現してもよい。

[0388] 例えば、地上デジタル放送の放送局（放送設備など）／受信局（テレビジョン受像機など）は、変調信号を無線放送で送受信する送信装置PROD_A／受信装置PROD_Bの一例である。また、ケーブルテレビ放送の放送局（放送設備など）／受信局（テレビジョン受像機など）は、変調信号を有線放送で送受信する送信装置PROD_A／受信装置PROD_Bの一例である。

[0389] また、インターネットを用いたVOD（Video On Demand）サービスや動画共有サービスなどのサーバ（ワークステーションなど）／クライアント（テレビジョン受像機、パーソナルコンピュータ、スマートフォンなど）は、変調信号を通信で送受信する送信装置PROD_A／受信装置PROD_Bの一例である（通常、ＬＡＮにおいては伝送媒体として無線または有線の何れかが用いられ、ＷＡＮにおいては伝送媒体として有線が用いられる）。ここで、パーソナルコンピュータには、デスクトップ型PC、ラップトップ型PC、及びタブレット型PCが含まれる。また、スマートフォンには、多機能携帯電話端末も含まれる。

[0390] なお、動画共有サービスのクライアントは、サーバからダウンロードした符号化データを復号してディスプレイに表示する機能に加え、カメラで撮像した動画像を符号化してサーバにアップロードする機能を有している。すな

わち、動画共有サービスのクライアントは、送信装置PROD_A及び受信装置PROD_Bの双方として機能する。

[0391] 次に、上述した画像符号化装置１１及び画像復号装置３１を、動画像の記録及び再生に利用できることを、図９を参照して説明する。

[0392] 図９の（ａ）は、上述した画像符号化装置１１を搭載した記録装置PROD_Cの構成を示したブロック図である。図９の（ａ）に示すように、記録装置PROD_Cは、動画像を符号化することによって符号化データを得る符号化部PROD_C1と、符号化部PROD_C1が得た符号化データを記録媒体PROD_Mに書き込む書込部PROD_C2と、を備えている。上述した画像符号化装置１１は、この符号化部PROD_C1として利用される。

[0393] なお、記録媒体PROD_Mは、（１）HDD（HardDisk Drive）やSSD(Solid State Drive)などのように、記録装置PROD_Cに内蔵されるタイプのものであってもよいし、（２）SDメモリカードやUSB（Universal Serial Bus）フラッシュメモリなどのように、記録装置PROD_Cに接続されるタイプのものであってもよいし、（３）DVD（Digital Versatile Disc）やBD（Blu-ray Disc:登録商標）などのように、記録装置PROD_Cに内蔵されたドライブ装置（不図示）に装填されるものであってもよい。

[0394] また、記録装置PROD_Cは、符号化部PROD_C1に入力する動画像の供給源として、動画像を撮像するカメラPROD_C3、動画像を外部から入力するための入力端子PROD_C4、動画像を受信するための受信部PROD_C5、及び、画像を生成または加工する画像処理部PROD_C6を更に備えていてもよい。図９の（ａ）においては、これら全てを記録装置PROD_Cが備えた構成を例示しているが、一部を省略しても構わない。

[0395] なお、受信部PROD_C5は、符号化されていない動画像を受信するものであってもよいし、記録用の符号化方式とは異なる伝送用の符号化方式で符号化された符号化データを受信するものであってもよい。後者の場合、受信部PROD_C5と符号化部PROD_C1との間に、伝送用の符号化方式で符号化された符号化データを復号する伝送用復号部（不図示）を介在させるとよい。

[0396] このような記録装置PROD_Cとしては、例えば、DVDレコーダ、BDレコーダ、HDD（Hard DiskDrive）レコーダなどが挙げられる（この場合、入力端子PROD_C4または受信部PROD_C5が動画像の主な供給源となる）。また、カムコーダ（この場合、カメラPROD_C3が動画像の主な供給源となる）、パーソナルコンピュータ（この場合、受信部PROD_C5または画像処理部Ｃ６が動画像の主な供給源となる）、スマートフォン（この場合、カメラPROD_C3または受信部PROD_C5が動画像の主な供給源となる）なども、このような記録装置PROD_Cの一例である。

[0397] 図９の（ｂ）は、上述した画像復号装置３１を搭載した再生装置PROD_Dの構成を示したブロックである。図９の（ｂ）に示すように、再生装置PROD_Dは、記録媒体PROD_Mに書き込まれた符号化データを読み出す読出部PROD_D1と、読出部PROD_D1が読み出した符号化データを復号することによって動画像を得る復号部PROD_D2と、を備えている。上述した画像復号装置３１は、この復号部PROD_D2として利用される。

[0398] なお、記録媒体PROD_Mは、（１）HDDやSSDなどのように、再生装置PROD_Dに内蔵されるタイプのものであってもよいし、（２）SDメモリカードやUSBフラッシュメモリなどのように、再生装置PROD_Dに接続されるタイプのものであってもよいし、（３）DVDやBDなどのように、再生装置PROD_Dに内蔵されたドライブ装置（不図示）に装填されるものであってもよい。

[0399] また、再生装置PROD_Dは、復号部PROD_D2が出力する動画像の供給先として、動画像を表示するディスプレイPROD_D3、動画像を外部に出力するための出力端子PROD_D4、及び、動画像を送信する送信部PROD_D5を更に備えていてもよい。図９の（ｂ）においては、これら全てを再生装置PROD_Dが備えた構成を例示しているが、一部を省略しても構わない。

[0400] なお、送信部PROD_D5は、符号化されていない動画像を送信するものであってもよいし、記録用の符号化方式とは異なる伝送用の符号化方式で符号化された符号化データを送信するものであってもよい。後者の場合、復号部PROD_D2と送信部PROD_D5との間に、動画像を伝送用の符号化方式で符号化する符号

化部（不図示）を介在させるとよい。

[0401] このような再生装置PROD_Dとしては、例えば、DVDプレイヤ、BDプレイヤ、HDDプレイヤなどが挙げられる（この場合、テレビジョン受像機等が接続される出力端子PROD_D4が動画像の主な供給先となる）。また、テレビジョン受像機（この場合、ディスプレイPROD_D3が動画像の主な供給先となる）、デジタルサイネージ（電子看板や電子掲示板等とも称され、ディスプレイPROD_D3または送信部PROD_D5が動画像の主な供給先となる）、デスクトップ型PC（この場合、出力端子PROD_D4または送信部PROD_D5が動画像の主な供給先となる）、ラップトップ型またはタブレット型PC（この場合、ディスプレイPROD_D3または送信部PROD_D5が動画像の主な供給先となる）、スマートフォン（この場合、ディスプレイPROD_D3または送信部PROD_D5が動画像の主な供給先となる）なども、このような再生装置PROD_Dの一例である。

[0402] （ハードウェア的実現及びソフトウェア的実現）

また、上述した画像復号装置３１及び画像符号化装置１１の各ブロックは、集積回路（ICチップ）上に形成された論理回路によってハードウェア的に実現してもよいし、CPU（Central Processing Unit）を用いてソフトウェア的に実現してもよい。

[0403] 後者の場合、上記各装置は、各機能を実現するプログラムの命令を実行するCPU、上記プログラムを格納したROM（ReadOnly Memory）、上記プログラムを展開するRAM（RandomAccess Memory）、上記プログラム及び各種データを格納するメモリ等の記憶装置（記録媒体）などを備えている。そして、本発明の実施形態の目的は、上述した機能を実現するソフトウェアである上記各装置の制御プログラムのプログラムコード（実行形式プログラム、中間コードプログラム、ソースプログラム）をコンピュータで読み取り可能に記録した記録媒体を、上記各装置に供給し、そのコンピュータ（またはCPUやMPU）が記録媒体に記録されているプログラムコードを読み出し実行することによっても、達成可能である。

[0404] 上記記録媒体としては、例えば、磁気テープやカセットテープ等のテープ

類、フロッピー（登録商標）ディスク／ハードディスク等の磁気ディスクやCD-ROM（Compact Disc Read-Only Memory）／MOディスク（Magneto-Optical disc）／MD（Mini Disc）／DVD（Digital Versatile Disc）／CD-R（CD Recordable）／ブルーレイディスク（Blu-ray Disc：登録商標）等の光ディスクを含むディスク類、ICカード（メモリカードを含む）／光カード等のカード類、マスクROM／EPROM（Erasable Programmable Read-Only Memory）／EEPROM（Electrically Erasable andProgrammable Read-Only Memory：登録商標）／フラッシュROM等の半導体メモリ類、あるいはPLD（Programmable logic device）やFPGA（Field Programmable Gate Array）等の論理回路類などを用いることができる。

[0405] また、上記各装置を通信ネットワークと接続可能に構成し、上記プログラムコードを通信ネットワークを介して供給してもよい。この通信ネットワークは、プログラムコードを伝送可能であればよく、特に限定されない。例えば、インターネット、イントラネット、エキストラネット、LAN（Local Area Network）、ISDN（Integrated Services Digital Network）、VAN（Value-Added Network）、CATV（Community Antenna television/CableTelevision）通信網、仮想専用網（Virtual Private Network）、電話回線網、移動体通信網、衛星通信網等が利用可能である。また、この通信ネットワークを構成する伝送媒体も、プログラムコードを伝送可能な媒体であればよく、特定の構成または種類のものに限定されない。例えば、IEEE（Institute of Electrical and ElectronicEngineers）1394、USB、電力線搬送、ケーブルＴＶ回線、電話線、ADSL（Asymmetric Digital Subscriber Line）回線等の有線でも、IrDA（Infrared Data Association）やリモコンのような赤外線、BlueTooth（登録商標）、IEEE802.11無線、HDR（High Data Rate）、NFC（Near Field Communication）、DLNA（Digital Living Network Alliance：登録商標）、携帯電話網、衛星回線、地上デジタル放送網等の無線でも利用可能である。なお、本発明の実施形態は、上記プログラムコードが電子的な伝送で具現化された、搬送波に埋め込まれたコンピュータデータ信号の形態でも実現され得

る。

[0406] 本発明の実施形態は上述した実施形態に限定されるものではなく、請求項に示した範囲で種々の変更が可能である。すなわち、請求項に示した範囲で適宜変更した技術的手段を組み合わせて得られる実施形態についても本発明の技術的範囲に含まれる。

（関連出願の相互参照）

本出願は、2016年8月26日に出願された日本国特許出願：特願2016-166319、及び2016年12月27日に出願された日本国特許出願：特願2016-253890に対して優先権の利益を主張するものであり、それを参照することにより、その内容の全てが本書に含まれる。

産業上の利用可能性

[0407] 本発明の実施形態は、画像データが符号化された符号化データを復号する画像復号装置、及び、画像データが符号化された符号化データを生成する画像符号化装置に好適に適用することができる。また、画像符号化装置によって生成され、画像復号装置によって参照される符号化データのデータ構造に好適に適用することができる。

符号の説明

[0408] １０ CT情報復号部

１１ 画像符号化装置

２０ CU復号部

３１ 画像復号装置

４１ 画像表示装置

請求の範囲

[請求項1] ピクチャを符号化ツリーユニット毎に復号する画像復号装置において、

上記符号化ツリーユニットの符号化ノードに対する二分木分割の分割方法を示す分割情報を復号する分割情報復号部と、

符号化ノードに対して、復号済の他の符号化ノードの上記分割情報を参照して、二分木分割を行う分割部と、を備える

ことを特徴とする画像復号装置。

[請求項2] 上記分割部は、符号化ノードに対して、上記符号化ツリーユニットにおける二分木のルートノードの上記分割情報が示す分割方法を用いて二分木分割を行う

ことを特徴とする請求項１に記載の画像復号装置。

[請求項3] 上記分割部は、符号化ノードに対して、当該符号化ノードと同じ親ノードを持つ復号済の他の符号化ノードの上記分割情報が示す分割方法を用いて二分木分割を行う

ことを特徴とする請求項１に記載の画像復号装置。

[請求項4] 上記分割部は、符号化ノードに対して、当該符号化ノードと同じ親ノードを持つ復号済の他の符号化ノードの上記分割情報が示す分割方法とは異なる分割方法を用いて二分木分割を行う

ことを特徴とする請求項１に記載の画像復号装置。

[請求項5] 上記分割部は、符号化ノードに対して、当該符号化ノードと同じ親ノードを持つ復号済の他の符号化ノードの上記分割情報が示す分割方法が水平分割または垂直分割である場合は、二分木分割を行わない

ことを特徴とする請求項１に記載の画像復号装置。

[請求項6] 上記分割部は、符号化ノードに対して、当該符号化ノードの親ノードの上記分割情報が示す分割方法と、当該符号化ノードと同じ親ノードを持つ復号済の他の符号化ノードの上記分割情報が示す分割方法とが同じである場合は、上記親ノード及び上記復号済の他の符号化ノー

ドと同じ分割方法を用いて二分木分割を行う
ことを特徴とする請求項１に記載の画像復号装置。

[請求項7] 上記分割部は、符号化ノードに対して、当該符号化ノードの親ノードの上記分割情報が示す分割方法と、当該符号化ノードと同じ親ノードを持つ復号済の他の符号化ノードの上記分割情報が示す分割方法とが同じである場合は、上記親ノード及び上記復号済の他の符号化ノードとは異なる分割方法を用いて二分木分割を行う
ことを特徴とする請求項１に記載の画像復号装置。

[請求項8] 上記分割部は、符号化ノードの階層が予め定めた階層よりも深い場合は、当該符号化ノードに対して、復号済の他の符号化ノードの上記分割情報を参照して、二分木分割を行うことを特徴とする請求項１～７のいずれか１項に記載の画像復号装置。

[請求項9] 上記分割部は、符号化ノードのサイズが予め定めたサイズよりも小さい場合は、当該符号化ノードに対して、復号済の他の符号化ノードの上記分割情報を参照して、二分木分割を行う
ことを特徴とする請求項１～７のいずれか１項に記載の画像復号装置。

[請求項10] 上記分割部は、上記符号化ツリーユニットが含まれるスライスのスライスタイプがBスライスまたはPスライスである場合は、符号化ノードに対して、復号済の他の符号化ノードの上記分割情報を参照して、二分木分割を行う
ことを特徴とする請求項１～７のいずれか１項に記載の画像復号装置。

[請求項11] 上記分割部は、上記符号化ツリーユニットが含まれるスライスのスライスタイプがIスライスである場合は、色差の符号化ノードの四分木構造を、輝度の符号化ノードの四分木構造と同じにする
ことを特徴とする請求項１～７のいずれか１項に記載の画像復号装置。

[請求項12] ピクチャを符号化ツリーユニットに分割して符号化する画像符号化装置において、

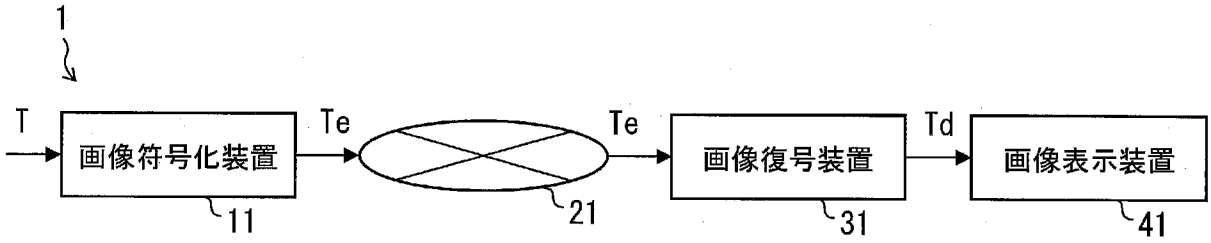
上記符号化ツリーユニットの符号化ノードに対する二分木分割の分割方法を示す分割情報を符号化する分割情報符号化部と、

符号化ノードに対して、符号化済の他の符号化ノードの上記分割情報を参照して、二分木分割を行う分割部と、を備える

ことを特徴とする画像符号化装置。

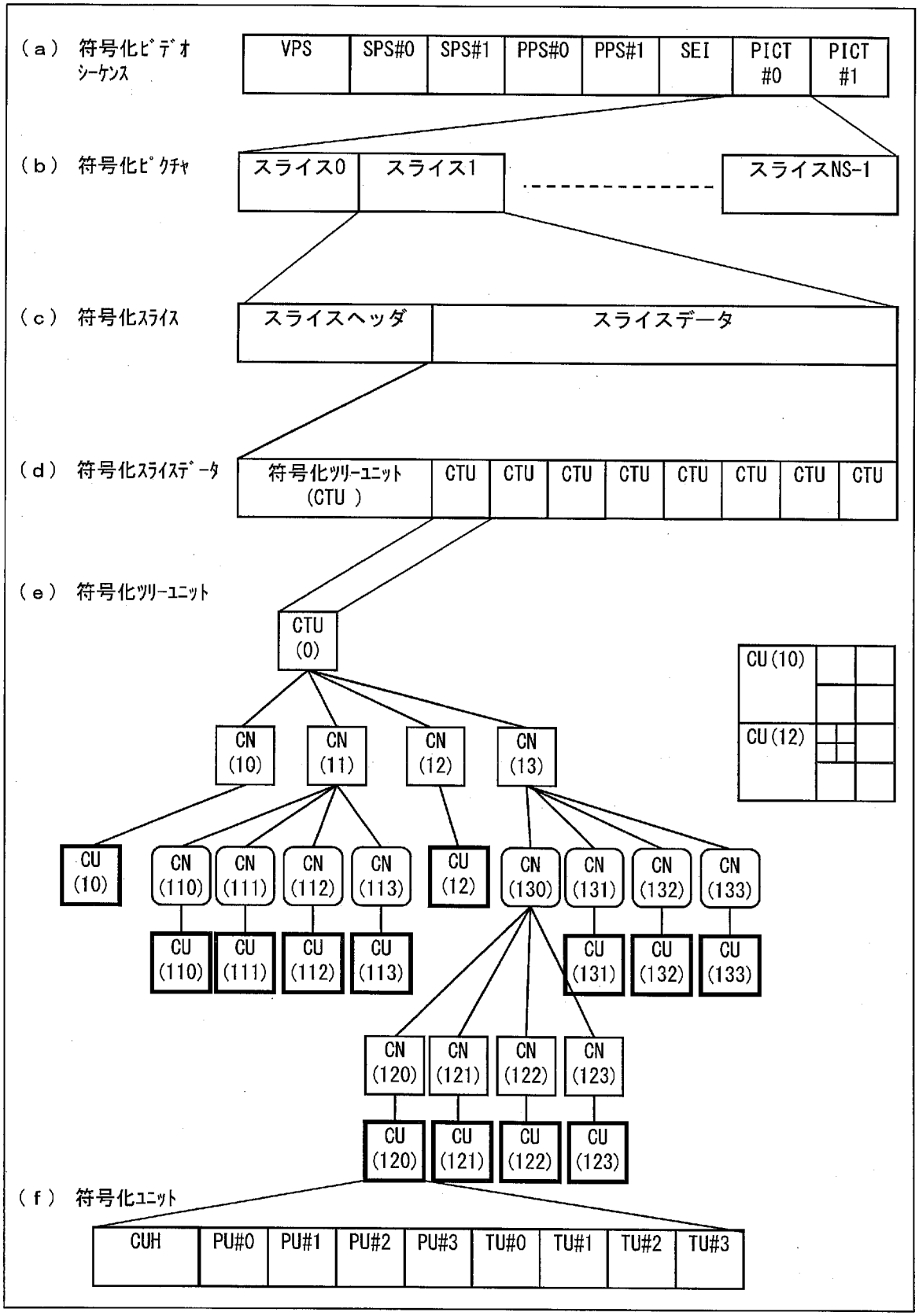
[図1]

図 1



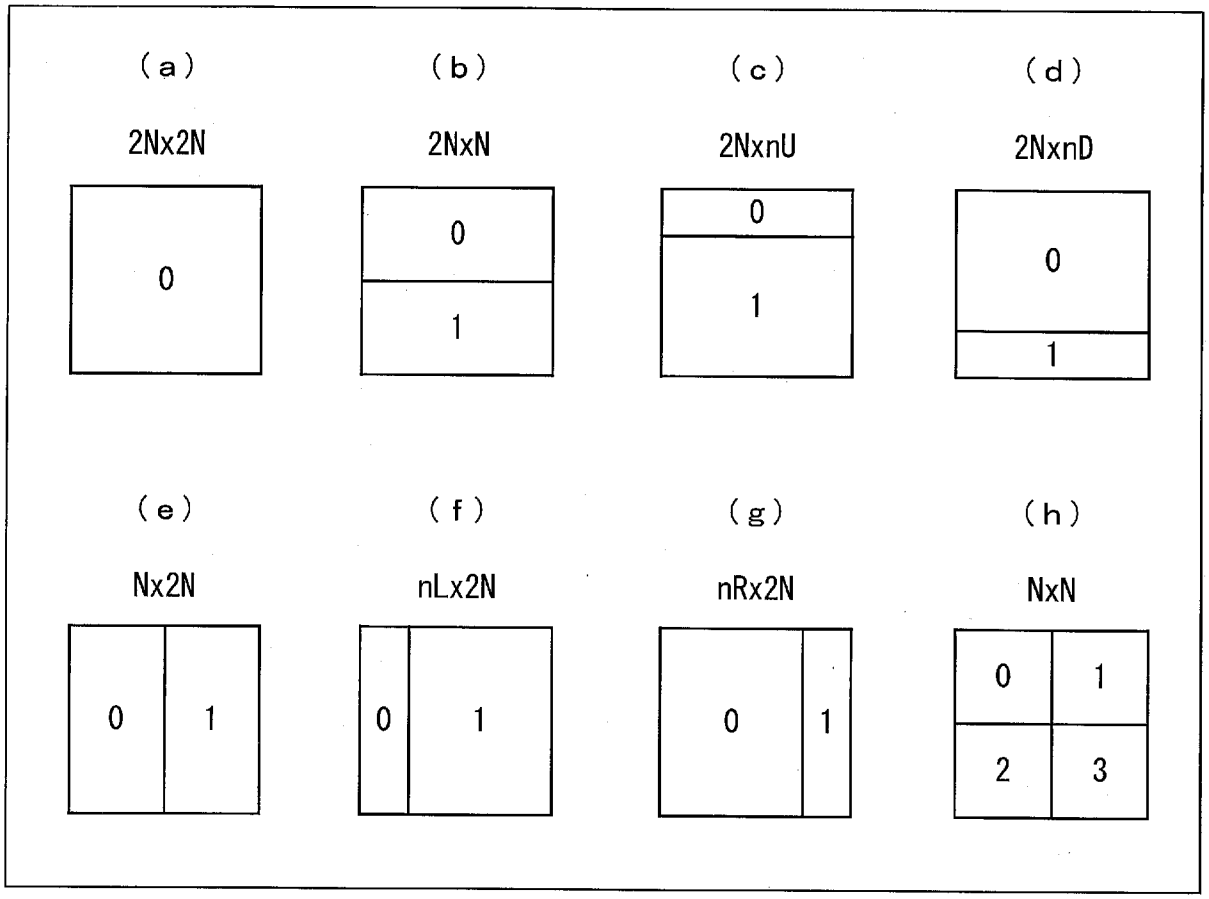
[図2]

図 2


(a) 符号化ビデオシーケンス
VPS
SPS#0
SPS#1
PPS#0
PPS#1
SEI
PICT #0
PICT #1
(b) 符号化ピクチャ
スライス0
スライス1
スライスNS-1
(c) 符号化スライス
スライスヘッダ
スライスデータ
(d) 符号化スライスデータ
符号化ツリーユニット (CTU)
CTU
(e) 符号化ツリーユニット
CTU (0)
CU (10)
CU (12)
CN (10)
CN (11)
CN (12)
CN (13)
CU (10)
CN (110)
CN (111)
CN (112)
CN (113)
CU (12)
CN (130)
CN (131)
CN (132)
CN (133)
CU (110)
CU (111)
CU (112)
CU (113)
CU (131)
CU (132)
CU (133)
CN (120)
CN (121)
CN (122)
CN (123)
CU (120)
CU (121)
CU (122)
CU (123)
(f) 符号化ユニット
CUH
PU#0
PU#1
PU#2
PU#3
TU#0
TU#1
TU#2
TU#3

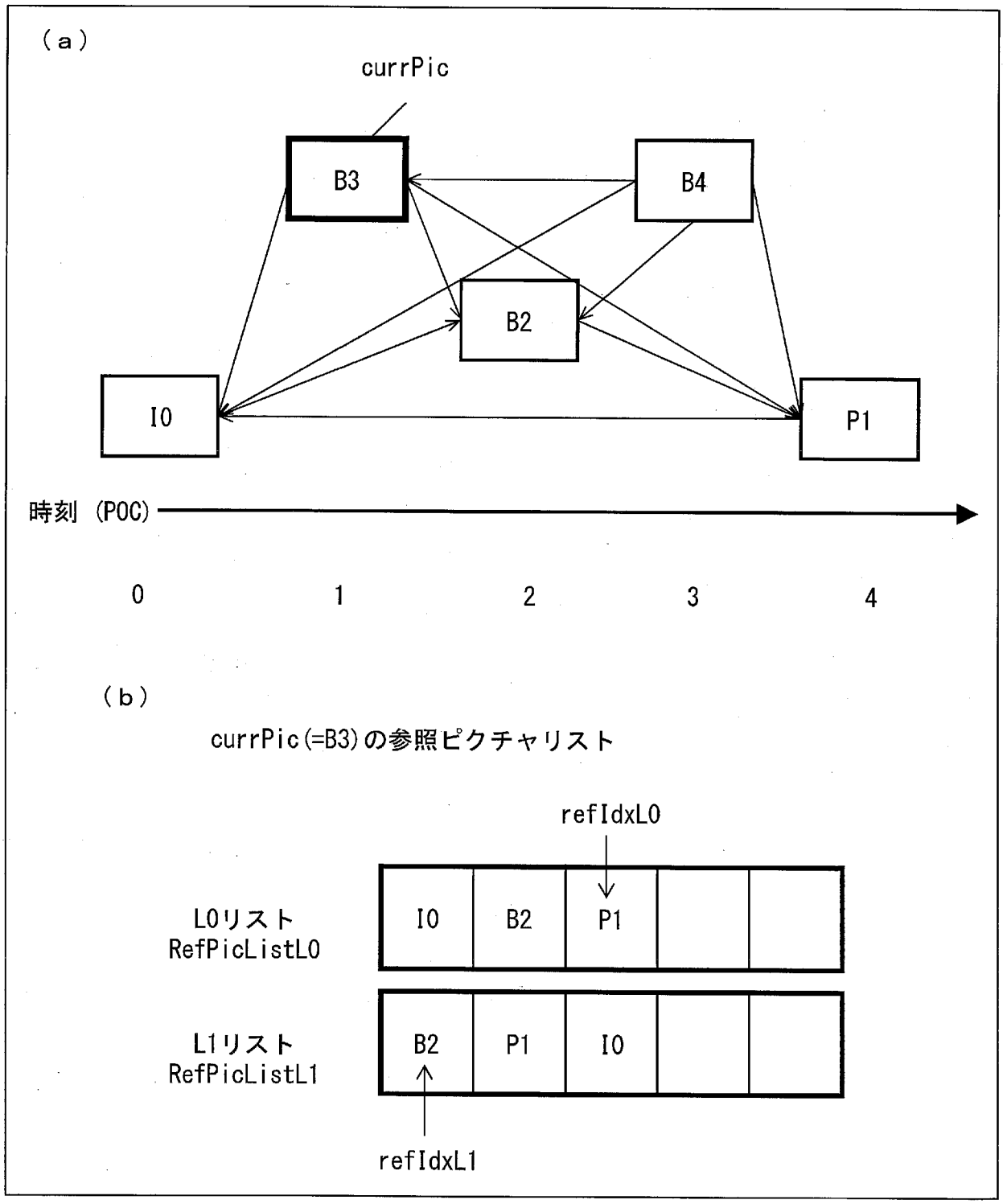
[図3]

図 3


(a)
2Nx2N
0
(b)
2NxN
0
1
(c)
2NxnU
0
1
(d)
2NxnD
0
1
(e)
Nx2N
0
1
(f)
nLx2N
0
1
(g)
nRx2N
0
1
(h)
NxN
0
1
2
3

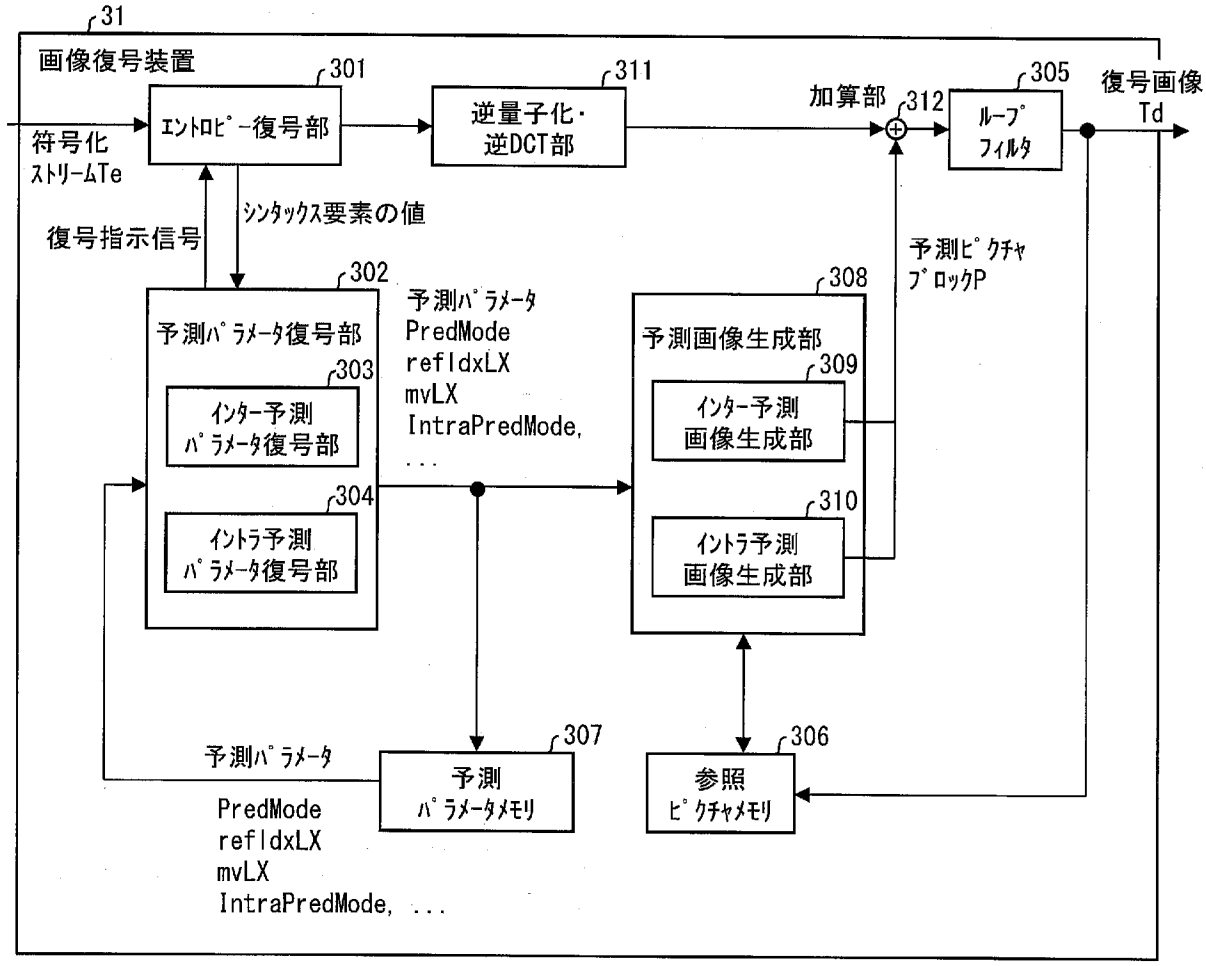
[図4]

図 4


(a)
currPic
B3
B4
B2
I0
P1
時刻 (POC)
0
1
2
3
4
(b)
currPic(=B3)の参照ピクチャリスト
refIdxL0
L0リスト
RefPicListL0
I0
B2
P1
L1リスト
RefPicListL1
B2
P1
I0
refIdxL1

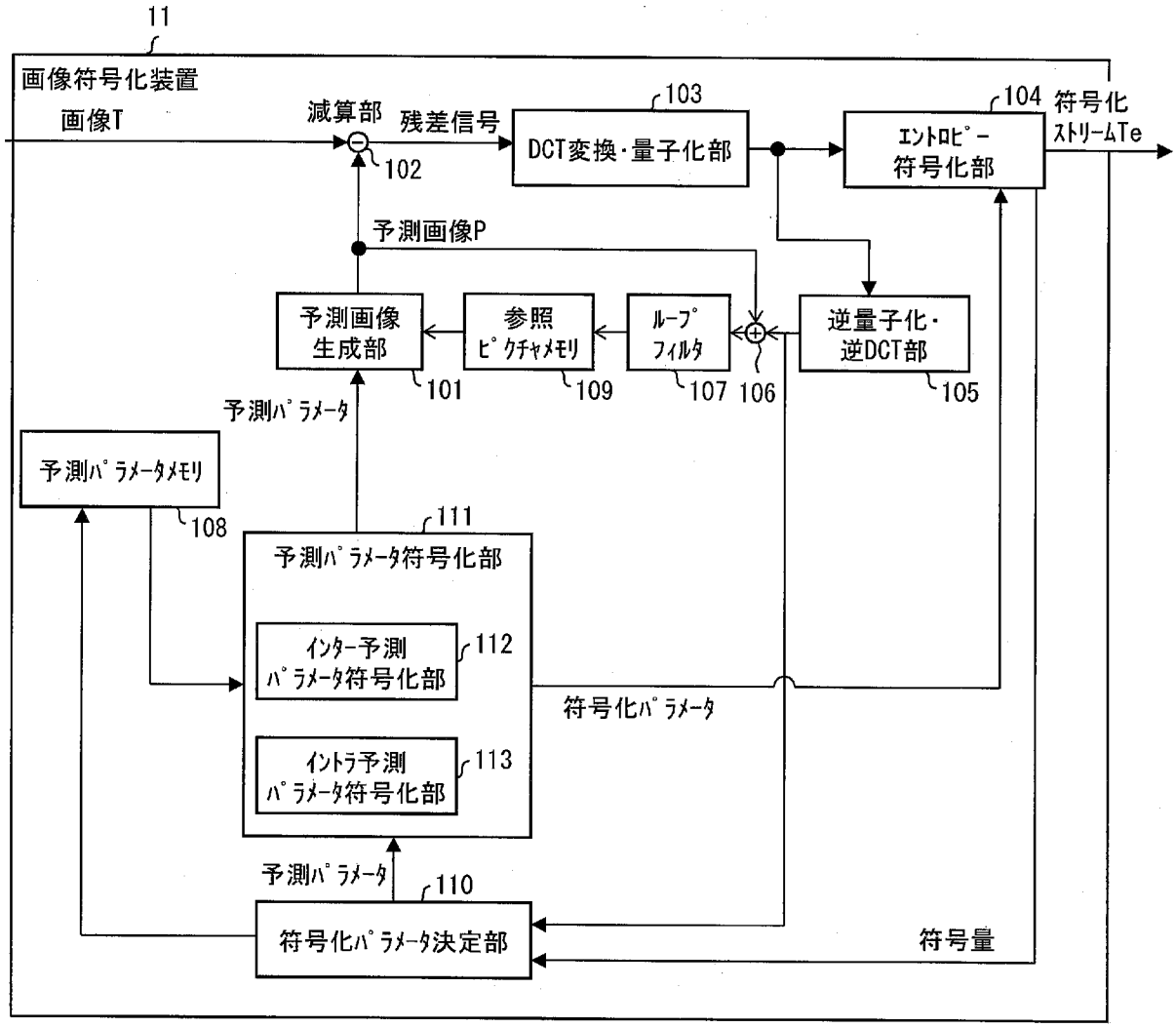
[図5]

図 5


31
画像復号装置
301
エントロピー復号部
311
逆量子化・逆DCT部
加算部
312
305
ループフィルタ
復号画像Td
符号化ストリームTe
復号指示信号
シンタックス要素の値
予測ピクチャブロックP
302
予測パラメータ復号部
303
インター予測パラメータ復号部
304
イントラ予測パラメータ復号部
予測パラメータ
PredMode
refIdxLX
mvLX
IntraPredMode,
...
308
予測画像生成部
309
インター予測画像生成部
310
イントラ予測画像生成部
307
予測パラメータメモリ
306
参照ピクチャメモリ
予測パラメータ
PredMode
refIdxLX
mvLX
IntraPredMode, ...

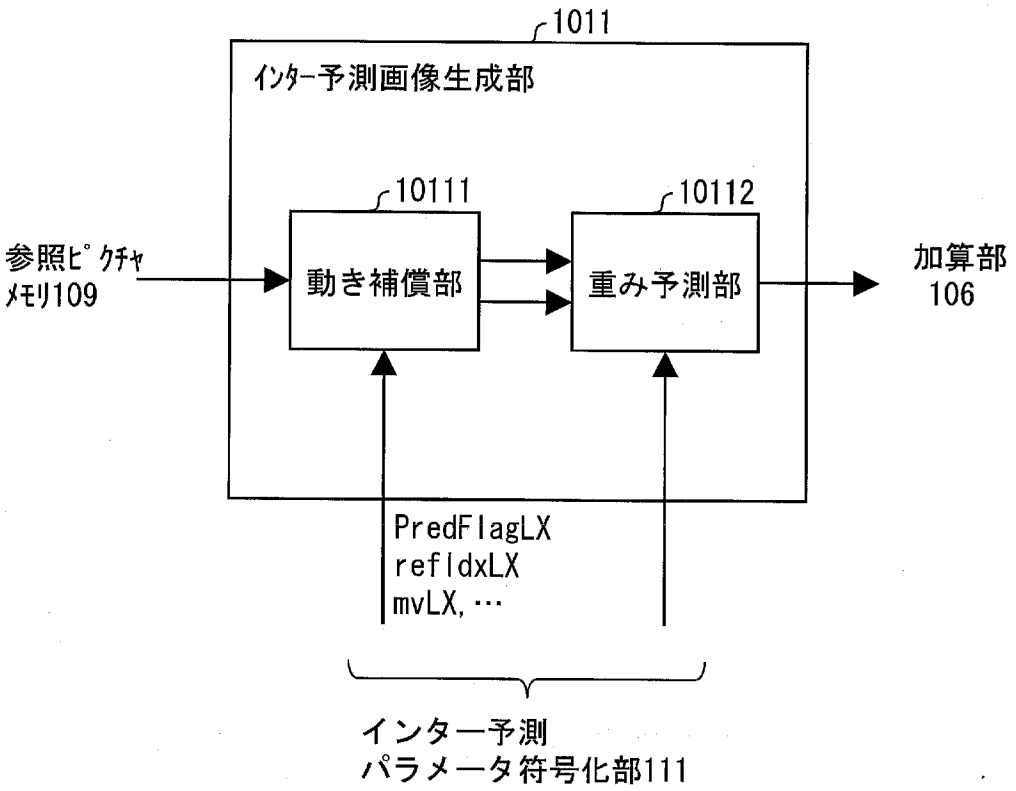
[図6]

図 6


11
画像符号化装置
画像T
減算部
残差信号
102
103
DCT変換・量子化部
104
エントロピー符号化部
符号化ストリームTe
予測画像P
予測画像生成部
101
参照ピクチャメモリ
109
ループフィルタ
107
106
逆量子化・逆DCT部
105
予測パラメータ
予測パラメータメモリ
108
111
予測パラメータ符号化部
インター予測パラメータ符号化部
112
イントラ予測パラメータ符号化部
113
符号化パラメータ
予測パラメータ
110
符号化パラメータ決定部
符号量

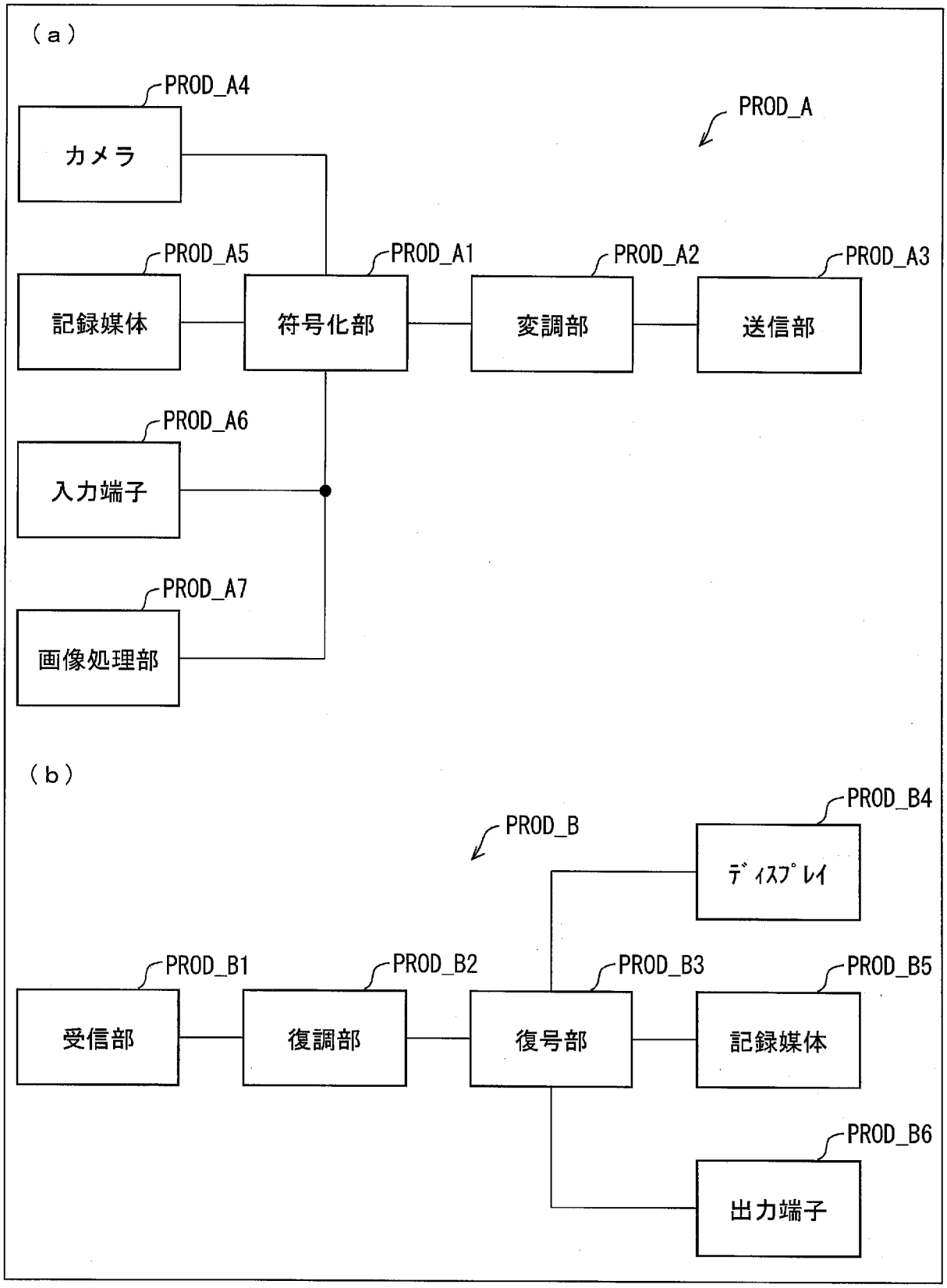
[図7]

図 7


1011
ｲﾝﾀｰ予測画像生成部
10111
10112
参照ﾋﾟｸﾁｬ
ﾒﾓﾘ109
動き補償部
重み予測部
加算部
106
PredFlagLX
refIdxLX
mvLX, …
インター予測
パラメータ符号化部111

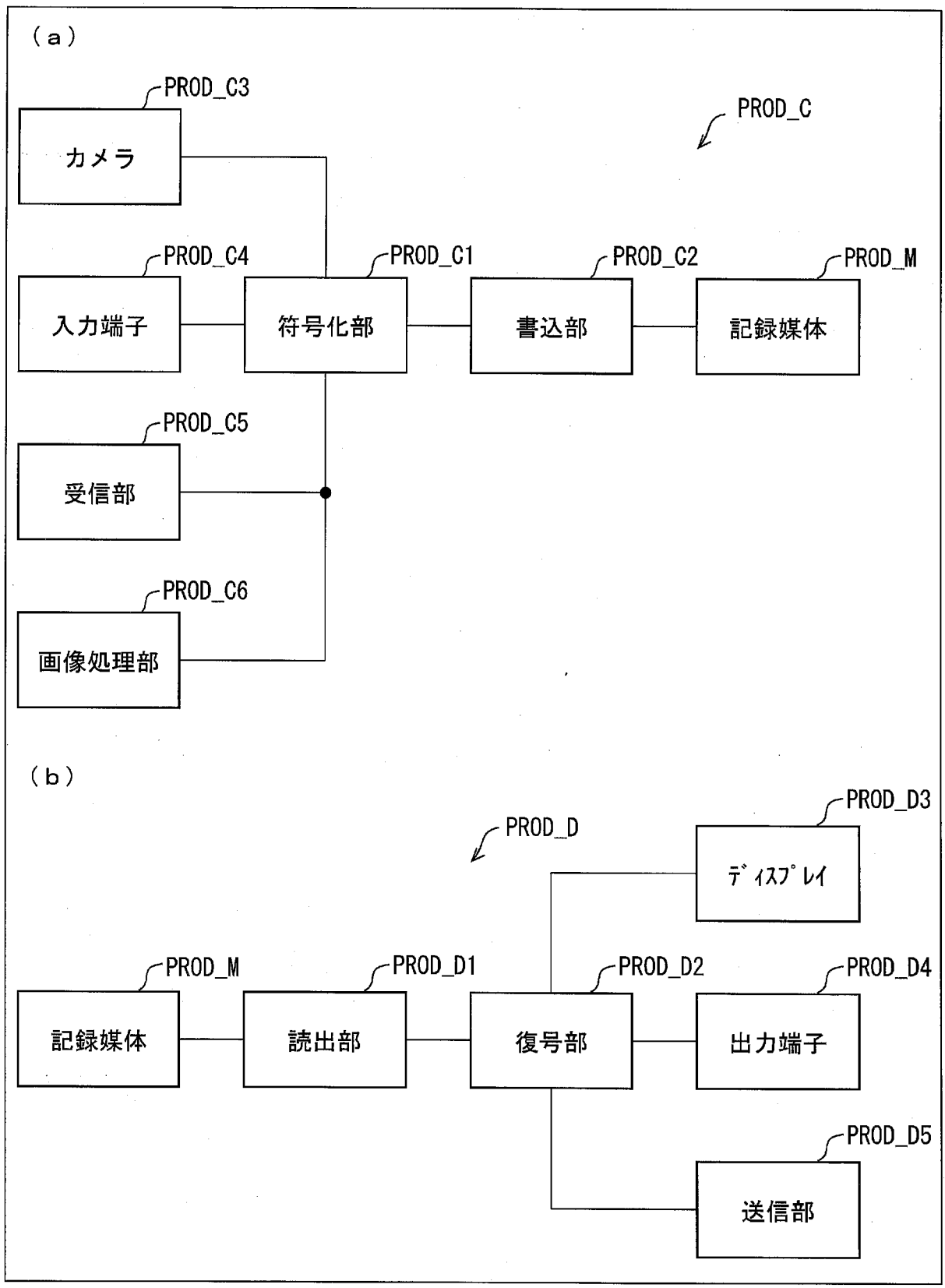
[図8]

図 8


（ａ）
PROD_A4
カメラ
PROD_A
PROD_A5
記録媒体
PROD_A1
符号化部
PROD_A2
変調部
PROD_A3
送信部
PROD_A6
入力端子
PROD_A7
画像処理部
（ｂ）
PROD_B
PROD_B4
ﾃﾞｨｽﾌﾟﾚｲ
PROD_B1
受信部
PROD_B2
復調部
PROD_B3
復号部
PROD_B5
記録媒体
PROD_B6
出力端子

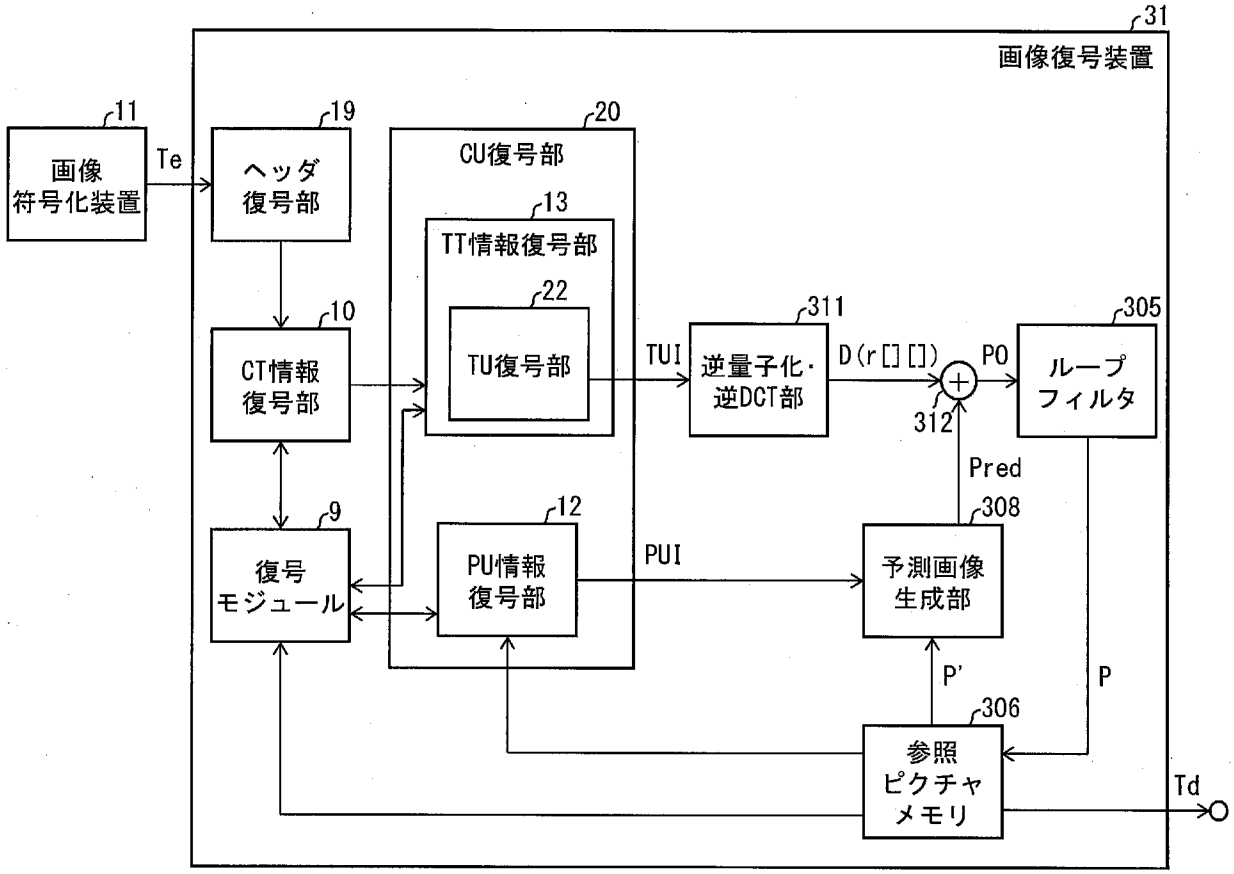
[図9]

図 9


(a)
PROD_C3
カメラ
PROD_C
PROD_C4
入力端子
PROD_C1
符号化部
PROD_C2
書込部
PROD_M
記録媒体
PROD_C5
受信部
PROD_C6
画像処理部
(b)
PROD_D
PROD_D3
ﾃﾞｨｽﾌﾟﾚｲ
PROD_M
記録媒体
PROD_D1
読出部
PROD_D2
復号部
PROD_D4
出力端子
PROD_D5
送信部

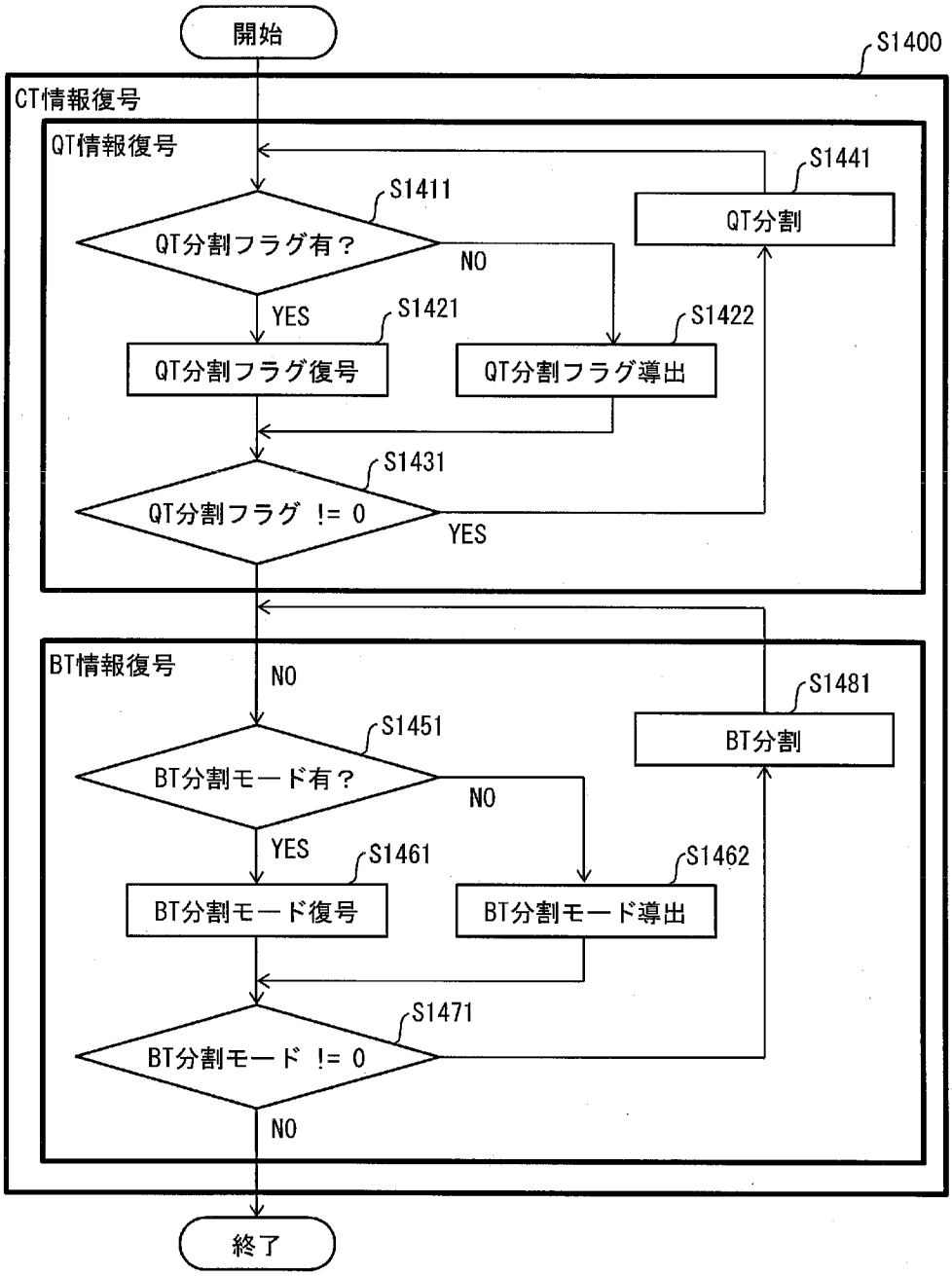
[図10]

図 10


31
画像復号装置
11
画像
符号化装置
Te
19
ヘッダ
復号部
20
CU復号部
13
TT情報復号部
22
TU復号部
10
CT情報
復号部
9
復号
モジュール
12
PU情報
復号部
TUI
311
逆量子化・
逆DCT部
D(r[][])
PO
312
305
ループ
フィルタ
Pred
308
予測画像
生成部
PUI
P'
P
306
参照
ピクチャ
メモリ
Td

[図11]

図 11


開始
S1400
CT情報復号
QT情報復号
S1411
QT分割フラグ有？
NO
YES
S1421
QT分割フラグ復号
S1422
QT分割フラグ導出
S1431
QT分割フラグ != 0
YES
S1441
QT分割
BT情報復号
NO
S1451
BT分割モード有？
NO
YES
S1461
BT分割モード復号
S1462
BT分割モード導出
S1471
BT分割モード != 0
NO
S1481
BT分割
終了

[図12]

図 12

coding_quadtree(x0, y0, log2CbSize, cqtDepth) {	**Descriptor**
if(x0 + (1 << log2CbSize) <= pic_width_in_luma_samples && y0 + (1 << log2CbSize) <= pic_height_in_luma_samples && log2CbSize > MinCbLog2SizeY	
split_cu_flag[x0][y0]	ae(v)
if(cu_qp_delta_enabled_flag && log2CbSize >= Log2MinCuQpDeltaSize) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
}	
if(cu_chroma_qp_offset_enabled_flag && log2CbSize >= Log2MinCuChromaQpOffsetSize)	
IsCuChromaQpOffsetCoded = 0	
if(split_cu_flag[x0][y0]) {	
x1 = x0 + (1 << (log2CbSize − 1))	
y1 = y0 + (1 << (log2CbSize − 1))	
coding_quadtree(x0, y0, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples)	
coding_quadtree(x1, y0, log2CbSize − 1, cqtDepth + 1)	
if(y1 < pic_height_in_luma_samples)	
coding_quadtree(x0, y1, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples && y1 < pic_height_in_luma_samples)	
coding_quadtree(x1, y1, log2CbSize − 1, cqtDepth + 1)	
} else	
coding_binarytree(x0, y0, log2CbSize, log2CbSize, cqtDepth, 0)	
}	

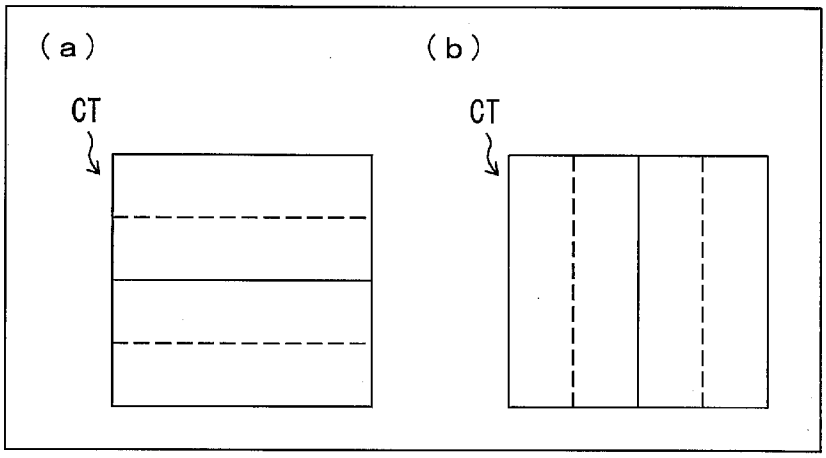
[図13]

図 13

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
split_bt_mode[x0][y0]	ae(v)
if (split_bt_mode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)	
} else if (split_bt_mode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図14]

図 14



[図15]

図 15

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cbtDepth) {	Descriptor
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	

if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) &&
(1 << log2CbWidth) <= maxBTSize &&
(1 << log2CbHeight) <= maxBTSize &&
cbtDepth < maxBTDepth) {	

split_bt_flag[x0][y0]	ae(v)
if (cbtDepth == 0 && **split_bt_flag**[x0][y0] == 1)	
split_dir_flag[x0] [y0]	ae(v)
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図16]

図 16

split_bt_flag	cbtDepth=0の split_dir_flag=0	cbtDepth=0の split_dir_flag=1
0	BT_SPLIT_NONE(=0)	BT_SPLIT_NONE(=0)
1	BT_SPLIT_HOR(=1)	BT_SPLIT_VER(=2)

[図17]

図 17

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	Descriptor
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	

if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) &&
(1 << log2CbWidth) <= maxBTSize &&
(1 << log2CbHeight) <= maxBTSize &&
cbtDepth < maxBTDepth) {	

if (splitConstrain == 0)	
split_bt_mode [x0][y0]	ae(v)
else	
split_bt_flag[x0][y0]	ae(v)
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cbtDepth + 1, splitBtMode[x0][y0])	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cbtDepth + 1, splitBtMode[x0][y0])	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1, splitBtMode[x0][y0])	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1, splitBtMode[x0][y0])	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図18]

図 18

split_bt_flag	splitConstrain=1	splitConstrain=2
0	BT_SPLIT_NONE(=0)	BT_SPLIT_NONE(=0)
1	BT_SPLIT_HOR(=1)	BT_SPLIT_VER(=2)

[図19]

図 19

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cbtDepth) {	Descriptor
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	

if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) &&
(1 << log2CbWidth) <= maxBTSize &&
(1 << log2CbHeight) <= maxBTSize &&
cbtDepth < maxBTDepth) {	

if (cbtDepth == 0)	
split_bt_mode [x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	ae(v)
}	
if (splitBtMode[x0][y0][cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
} else if (splitBtMode[x0][y0][cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

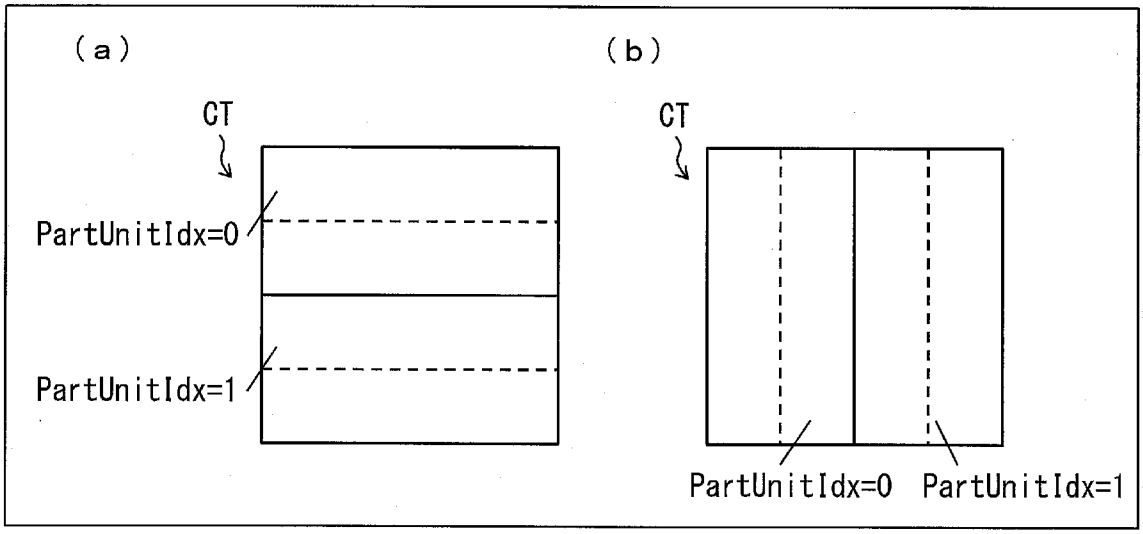
[図20]

図 20

split_bt_flag	splitBtMode[x0][y0][cbtDepth-1]=1	splitBtMode[x0][y0][cbtDepth-1]=2
0	BT_SPLIT_NONE(=0)	BT_SPLIT_NONE(=0)
1	BT_SPLIT_HOR(=1)	BT_SPLIT_VER(=2)

[図21]

図 21



[図22]

図 22

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1])	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1])	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

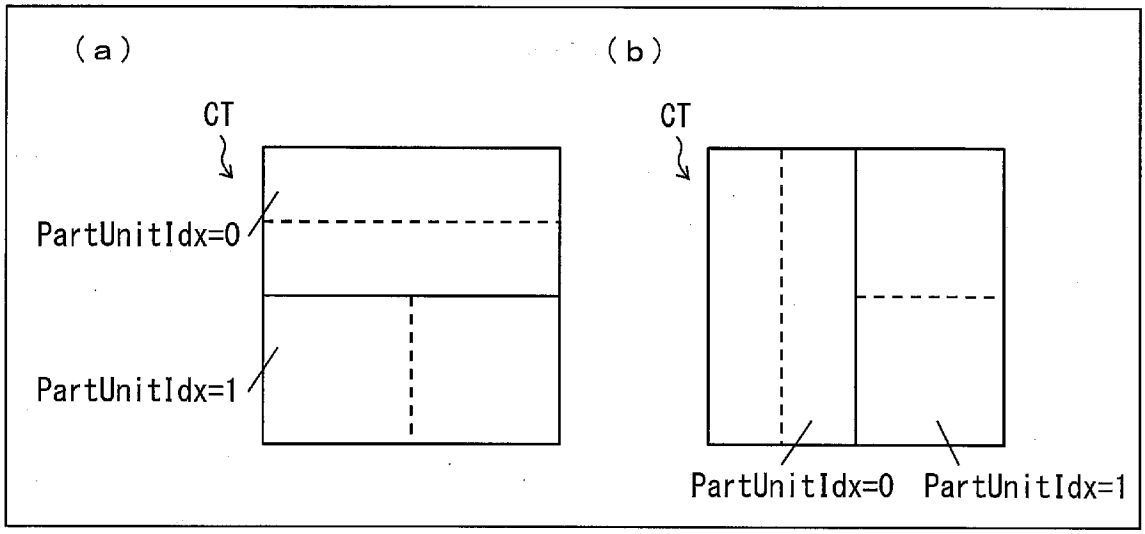
[図23]

図 23

split_bt_flag	splitConstrain=1	splitConstrain=2
0	BT_SPLIT_NONE(=0)	BT_SPLIT_NONE(=0)
1	BT_SPLIT_HOR(=1)	BT_SPLIT_VER(=2)

[図24]

図 24



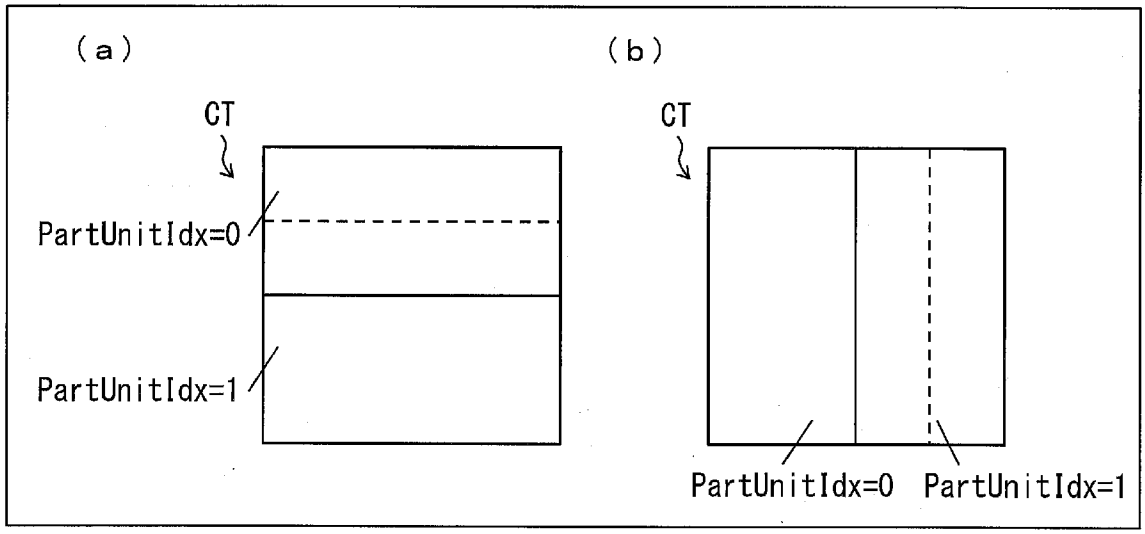
[図25]

図 25

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	Descriptor
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] != 0 ? 3-split_bt_mode[x0][y0][cbtDepth + 1] : 0)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode [x0][y0][cbtDepth + 1] != 0 ? 3-split_bt_mode[x0][y0][cbtDepth + 1] : 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図26]

図 26



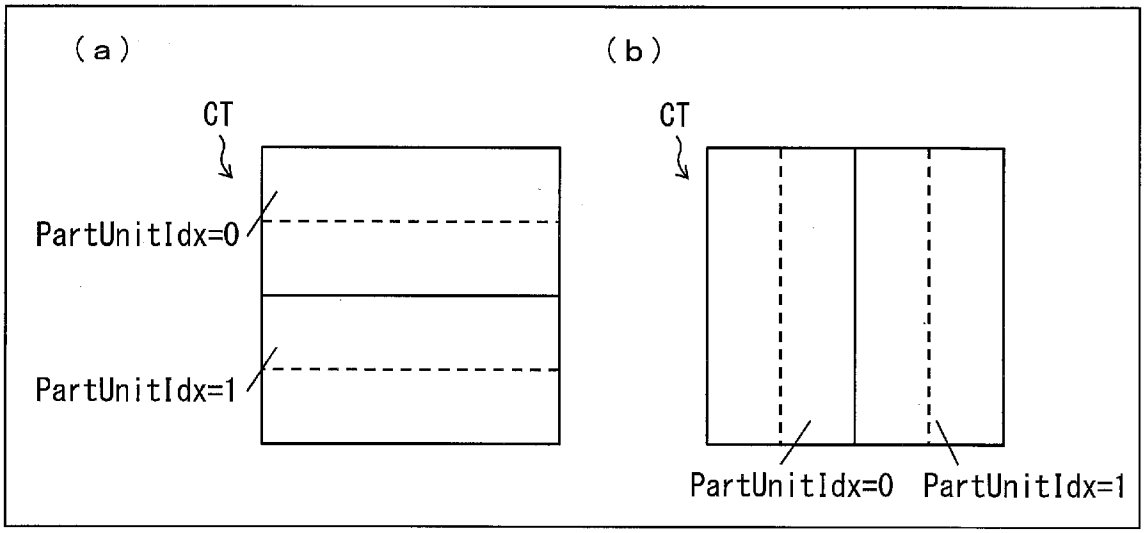
[図27]

図 27

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth **&& !splitConstrain**)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
if (**split_bt_mode**[x0][y0][cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] != 0 ? 1: 0)	
} else if (**split_bt_mode**[x0][y0][cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode [x0][y0][cbtDepth + 1] != 0 ? 1: 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図28]

図 28



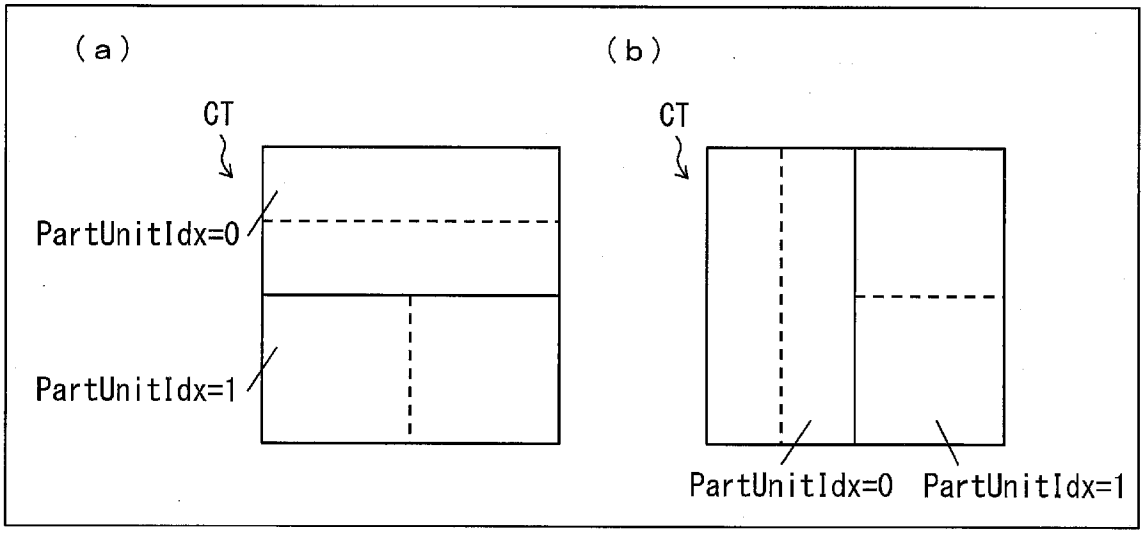
[図29]

図 29

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize)) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0][cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 1? 1: 0)	
} else if (splitBtMode[x0][y0][cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 2 ? 2: 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図30]

図 30



[図31]

図 31

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize)) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0][cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 1 ? **2**: 0)	
} else if (splitBtMode[x0][y0][cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 2 ? **1**: 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図32]

図 32

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize)) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, log2CbWidth >= log2CbHeight + 2 ? 2 : 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, log2CbWidth >= log2CbHeight + 2 ? 2 : 0)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, log2CbHeight >= log2CbWidth + 2 ? 1 : 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, log2CbHeight >= log2CbWidth + 2 ? 1 : 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図33]

図 33

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize)) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	ae(v)
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, log2CbWidth >= log2CbHeight + 2 && slice_type == I_SLICE ? 2: 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, log2CbWidth >= log2CbHeight + 2 **&& slice_type == I_SLICE** ? 2: 0)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, log2CbHeight >= log2CbWidth + 2 && slice_type == I_SLICE ? 1 : 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, log2CbHeight >= log2CbWidth + 2 **&& slice_type == I_SLICE** ? 1 : 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図34]

図 34

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cbtDepth) {	Descriptor
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	

if (((1 << log2CbHeight) > minBTSize ||
(1 << log2CbWidth) > minBTSize) &&
(1 << log2CbWidth) <= maxBTSize &&
(1 << log2CbHeight) <= maxBTSize &&
cbtDepth < maxBTDepth) {	

split_bt_flag [x0][y0]	ae(v)
if (cbtDepth <= TH && split_bt_flag [x0][y0] == 1)	
split_dir_flag [x0] [y0] [cbtDepth]	ae(v)
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図35]

図 35

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	

if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) &&
(1 << log2CbWidth) <= maxBTSize &&
(1 << log2CbHeight) <= maxBTSize &&
cbtDepth < maxBTDepth) {	

if (splitConstrain == 0 || **cbtDepth <=TH**)	
split_bt_mode [x0][y0]	ae(v)
else	
split_bt_flag[x0][y0]	ae(v)
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cbtDepth + 1, splitBTMode[x0][y0])	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cbtDepth + 1, splitBTMode[x0][y0]))	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1, splitBTMode[x0][y0]))	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1, splitBTMode[x0][y0]))	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図36]

図 36

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cbtDepth) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	

if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) &&
(1 << log2CbWidth) <= maxBTSize &&
(1 << log2CbHeight) <= maxBTSize &&
cbtDepth < maxBTDepth) {	

if (**cbtDepth <= TH**)	ae(v)
split_bt_mode [x0][y0]	
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] [cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cbtDepth + 1)	
} else if (splitBtMode[x0][y0] [cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cbtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図37]

図 37

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, **cbtDepth > TH** ? split_bt_mode[x0][y0][cbtDepth + 1] : 0)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, **cbtDepth > TH** ? split_bt_mode[x0][y0][cbtDepth + 1] : 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図38]

図 38

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] !=0 **&& cbtDepth > TH** ? 3-split_bt_mode[x0][y0][cbtDepth + 1] : 0)	
} else if (splitBtMode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] !=0 **&& cbtDepth > TH** ? 3-split_bt_mode[x0][y0][cbtDepth + 1] : 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図39]

図 39

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth **&& !splitConstrain)**	
split_bt_mode[x0][y0][cbtDepth]	
if (**split_bt_mode**[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] != 0 **&& cbtDepth > TH** ? 1: 0)	
} else if (**split_bt_mode**[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode [x0][y0][cbtDepth + 1] != 0 **&& cbtDepth > TH** ? 1: 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図40]

図 40

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize)) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0][cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 1 **&& cbtDepth > TH** ? 1: 0)	
} else if (splitBtMode[x0][y0][cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 2 **&& cbtDepth > TH** ? 2: 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

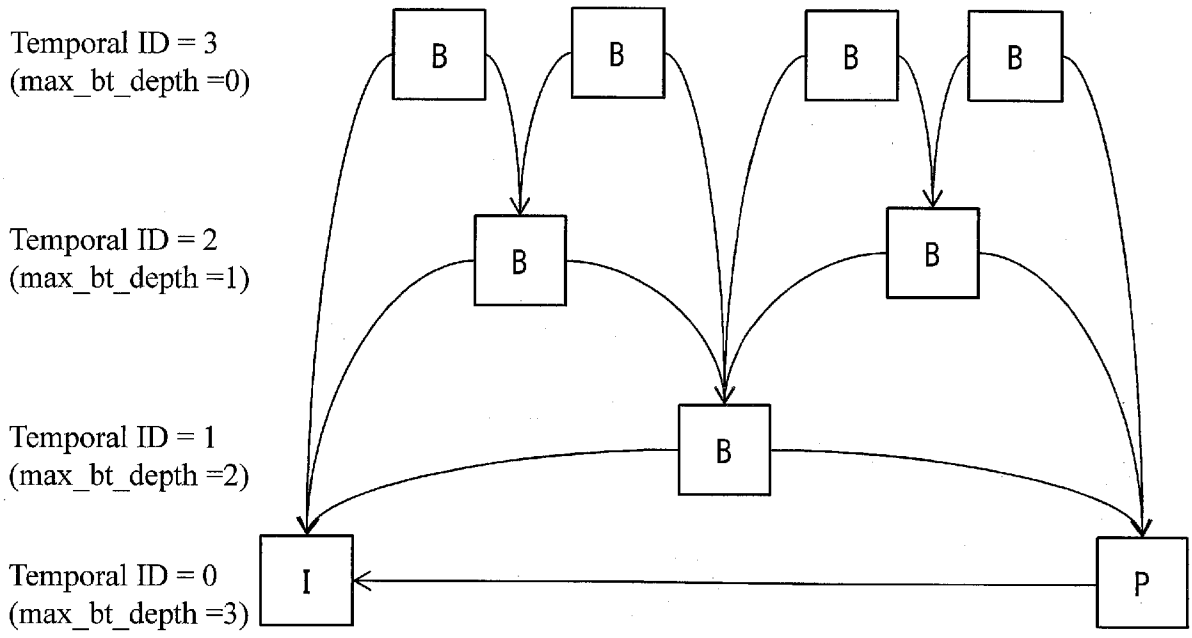
[図41]

図 41

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth, splitConstrain) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize)) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (splitConstrain == 0)	
split_bt_mode[x0][y0][cbtDepth]	
else	
split_bt_flag[x0][y0]	
}	
if (splitBtMode[x0][y0][cbtDepth] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 1 **&& cbtDepth > TH** ? 2: 0)	
} else if (splitBtMode[x0][y0][cbtDepth] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, 0)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1, split_bt_mode[x0][y0][cbtDepth + 1] == 2 **&& cbtDepth > TH** ? 1: 0)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図42]

図 42



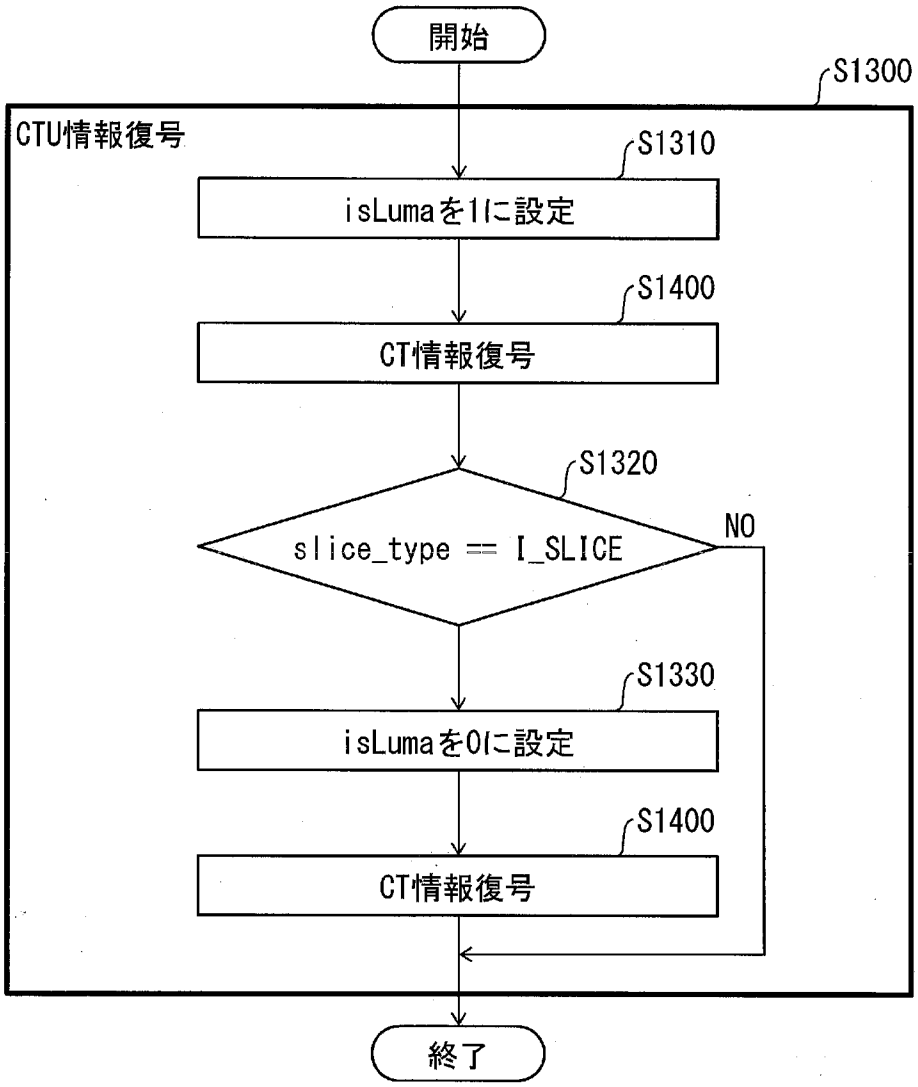
[図43]

図 43

seq_parameter_set_rbsp( ) {	Descriptor
・・・	
sps_max_latency_increase_plus1[i]	ue(v)
}	
log2_CTU_size_minus2	ue(v)
log2_minQT_ISliceLuma_minus2	ue(v)
log2_minQT_ISliceChroma_minus2	ue(v)
log2_minQT_PBSlice_minus2	ue(v)
log2_max_bt_size_minus2	ue(v)
log2_max_bt_size_i_slice_luma_minus2	ue(v)
log2_max_bt_size_i_slice_chroma_minus2	ue(v)
for (i=0; i < max_temporal_id; i++) {	
max_bt_depth_minus2[i]	ue(v)
max_bt_depth_i_slice_luma_minus2[i]	ue(v)
max_bt_depth_i_slice_chroma_minus2[i]	ue(v)
}	
log2_min_luma_transform_block_size_minus2	ue(v)
・・・	

[図44]

図 44


開始
S1300
CTU情報復号
S1310
isLumaを1に設定
S1400
CT情報復号
S1320
slice_type == I_SLICE
NO
S1330
isLumaを0に設定
S1400
CT情報復号
終了

[図45]

図 45

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth) {	**Descriptor**
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth && (slice_type != I_Slice || isLuma || (cqtDepth + cbtDepth) < (cqtDepthL[x0][y0] + cbtDepthL[x0][y0])))	
split_bt_mode[x0][y0]	ae(v)
if (split_bt_mode[x0][y0] == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)	
} else if (split_bt_mode[x0][y0] == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)	
} else	
if (isLuma) {	
cqtDepthL[x0][y0] = cqtDepth	
cbtDepthL[x0][y0] = cbtDepth	
}	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図46]

図 46

coding_quadtree(x0, y0, log2CbSize, cqtDepth) {	Descriptor
if(x0 + (1 << log2CbSize) <= pic_width_in_luma_samples && y0 + (1 << log2CbSize) <= pic_height_in_luma_samples && log2CbSize > MinCbLog2SizeY && (1<<log2CbSize) >= minQTSize)	
split_cu_flag[x0][y0]	ae(v)
if(cu_qp_delta_enabled_flag && log2CbSize >= Log2MinCuQpDeltaSize) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
}	
if(cu_chroma_qp_offset_enabled_flag && log2CbSize >= Log2MinCuChromaQpOffsetSize)	
IsCuChromaQpOffsetCoded = 0	
if(slice_type != I_Slice || isLuma) {	
splitCuFlag = split_cu_flag[x0][y0]	
splitCuFlagL[x0][y0] = split_cu_flag[x0][y0]	
} else	
splitCuFlag = splitCuFlagL [対応する輝度のX座標][対応する輝度のY座標] && split_cu_flag[x0][y0]	
if(splitCuFlag) {	
x1 = x0 + (1 << (log2CbSize − 1))	
y1 = y0 + (1 << (log2CbSize − 1))	
coding_quadtree(x0, y0, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples)	
coding_quadtree(x1, y0, log2CbSize − 1, cqtDepth + 1)	
if(y1 < pic_height_in_luma_samples)	
coding_quadtree(x0, y1, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples && y1 < pic_height_in_luma_samples)	
coding_quadtree(x1, y1, log2CbSize − 1, cqtDepth + 1)	
} else	
coding_binarytree(x0, y0, log2CbSize, log2CbSize, cqtDepth, 0)	
}	

[図47]

図 47

coding_binarytree (x0, y0, log2CbWidth, log2CbHeight, cqtDepth, cbtDepth) {	Descriptor
maxBTSize = (slice_type == I_Slice) ? (isLuma ? 1<<(log2_max_bt_size_i_slice_luma_minus2 + 2) : 1<<(log2_max_bt_size_i_slice_chroma_minus2 + 2)) : 1<<(log2_max_bt_size_minus2 + 2)	
minBTSize = 4	
maxBTDepth = (slice_type == I_Slice) ? (isLuma ? max_bt_depth_i_slice_luma_minus2 + 2 : max_bt_depth_i_slice_chroma_minus2 + 2) : max_bt_depth_minus2 + 2	
if (((1 << log2CbHeight) > minBTSize || (1 << log2CbWidth) > minBTSize) && (1 << log2CbWidth) <= maxBTSize && (1 << log2CbHeight) <= maxBTSize && cbtDepth < maxBTDepth)	
if (isLuma) {	
split_bt_mode[x0][y0]	ae(v)
splitBtMode = split_bt_mode[x0][y0]	
} else {	
split_bt_flag[x0][y0]	ae(v)
splitBtMode = split_bt_flag[x0][y0] ? split_bt_mode[対応する輝度のX座標][対応する輝度のY座標] : 0	
}	
if (splitBtMode == 1) { /*水平分割*/	
y1 = y0 + (1<< (log2CbHeight - 1))	
coding_binarytree(x0, y0, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)	
coding_binarytree(x0, y1, log2CbWidth, log2CbHeight - 1, cqtDepth, cbtDepth + 1)	
} else if (splitBtMode == 2) { /*垂直分割*/	
x1 = x0 + (1<< (log2CbWidth - 1))	
coding_binarytree(x0, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)	
coding_binarytree(x1, y0, log2CbWidth - 1, log2CbHeight, cqtDepth, cbtDepth + 1)	
} else	
coding_unit(x0, y0, log2CbWidth, log2CbHeight)	
}	

[図48]

図 48

coding_quadtree(x0, y0, log2CbSize, cqtDepth) {	Descriptor
if(x0 + (1 << log2CbSize) <= pic_width_in_luma_samples && y0 + (1 << log2CbSize) <= pic_height_in_luma_samples && log2CbSize > MinCbLog2SizeY && (1<<log2CbSize) >= minQTSize && isLuma)	
split_cu_flag[x0][y0]	ae(v)
if(cu_qp_delta_enabled_flag && log2CbSize >= Log2MinCuQpDeltaSize) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
}	
if(cu_chroma_qp_offset_enabled_flag && log2CbSize >= Log2MinCuChromaQpOffsetSize)	
IsCuChromaQpOffsetCoded = 0	
if(slice_type != I_Slice || isLuma)	
splitCuFlag = split_cu_flag[x0][y0]	
else	
splitCuFlag = split_cu_flag[対応する輝度のX座標][対応する輝度のY座標]	
if(splitCuFlag) {	
x1 = x0 + (1 << (log2CbSize − 1))	
y1 = y0 + (1 << (log2CbSize − 1))	
coding_quadtree(x0, y0, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples)	
coding_quadtree(x1, y0, log2CbSize − 1, cqtDepth + 1)	
if(y1 < pic_height_in_luma_samples)	
coding_quadtree(x0, y1, log2CbSize − 1, cqtDepth + 1)	
if(x1 < pic_width_in_luma_samples && y1 < pic_height_in_luma_samples)	
coding_quadtree(x1, y1, log2CbSize − 1, cqtDepth + 1)	
} else	
coding_binarytree(x0, y0, log2CbSize, log2CbSize, cqtDepth, 0)	
}	

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2017/027877

A. CLASSIFICATION OF SUBJECT MATTER
H04N19/119(2014.01)i, *H04N19/136*(2014.01)i, *H04N19/157*(2014.01)i, *H04N19/176*(2014.01)i, *H04N19/70*(2014.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04N19/119, H04N19/136, H04N19/157, H04N19/176, H04N19/70

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2017
Kokai Jitsuyo Shinan Koho	1971-2017	Toroku Jitsuyo Shinan Koho	1994-2017

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Jianle Chen,et al., Algorithm Description of Joint Exploration Test Model 1, Joint Video Exploration Team(JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG11 1nd Meeting:Geneva,CH, 19-21 October 2015, 2016.02.24, JVET-A1001	1-12
A	Jicheng An,Han Huang,Kai Zhang,et al., Quadtree plus binary tree structure integration with JEM tools, Joint Video Exploration Team(JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG11 2nd Meeting:San Diego,USA,20-26 February 2016, 2016.02.08, JVET-B0023	1-12

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
06 October 2017 (06.10.17)	17 October 2017 (17.10.17)

Name and mailing address of the ISA/	Authorized officer
Japan Patent Office 3-4-3,Kasumigaseki,Chiyoda-ku, Tokyo 100-8915,Japan	Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2017/027877

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Xin Zhao,Vadim Seregin,et al., Grouped signalling for transform in QTBT, Joint Video Exploration Team(JVET) of ITU-T SG 16 WP3 and ISO/IEC JTC 1/SC 29/WG11 3rd Meeting:Geneva,CH, 26 May-1 June 2016, 2016.05.17, JVET-C0054	1-12
A	C.Rosewarne,B.Bross,et al., High Efficiency Video Coding(HEVC)Test Model 16(HM 16)Improved Encoder Description Update 3, Joint Collaborative Team on Video Coding(JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11 21st Meeting:Warsaw,PL,19-26 Jun.2015, 2015.10.15, JCTVC-U1002, P.11-14	1-12
A	JP 2014-511628 A (Mitsubishi Electric Corp.), 15 May 2014 (15.05.2014), entire text; all drawings & US 2012/0281928 A1 the whole document & US 2013/0279820 A1 & WO 2012/150693 A1 & WO 2014/203805 A1 & EP 2705665 A1 & TW 201309025 A & CN 103503461 A	1-12

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２０１７／０２７８７７

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　H04N19/119(2014.01)i, H04N19/136(2014.01)i, H04N19/157(2014.01)i, H04N19/176(2014.01)i, H04N19/70(2014.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　H04N19/119, H04N19/136, H04N19/157, H04N19/176, H04N19/70

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２０１７年
日本国実用新案登録公報	１９９６－２０１７年
日本国登録実用新案公報	１９９４－２０１７年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
A	Jianle Chen,et al., Algorithm Description of Joint Exploration Test Model 1, Joint Video Exploration Team(JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG11 1nd Meeting:Geneva,CH,19-21 October 2015, 2016.02.24, JVET-A1001	1-12
A	Jicheng An,Han Huang,Kai Zhang,et al., Quadtree plus binary tree structure integration with JEM tools, Joint Video Exploration Team(JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG11 2nd Meeting:San Diego,USA,20-26 February 2016, 2016.02.08,	1-12

☑　Ｃ欄の続きにも文献が列挙されている。　　☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
０６．１０．２０１７	１７．１０．２０１７

国際調査機関の名称及びあて先
日本国特許庁（ＩＳＡ／ＪＰ）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員）　５Ｃ　９２４５
堀井　啓明
電話番号　０３－３５８１－１１０１　内線　３５４１

C（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求項の番号
	JVET-B0023	
A	Xin Zhao, Vadim Seregin, et al., Grouped signalling for transform in QTBT, Joint Video Exploration Team(JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG11 3rd Meeting:Geneva, CH, 26 May-1 June 2016, 2016.05.17, JVET-C0054	1-12
A	C. Rosewarne, B. Bross, et al., High Efficiency Video Coding(HEVC) Test Model 16(HM 16) Improved Encoder Description Update 3, Joint Collaborative Team on Video Coding(JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11 21st Meeting:Warsaw, PL, 19-26 Jun. 2015, 2015.10.15, JCTVC-U1002, P. 11-14	1-12
A	JP 2014-511628 A（三菱電機株式会社）2014.05.15, 全文全図 & US 2012/0281928 A1, the whole document& US 2013/0279820 A1& WO 2012/150693 A1& WO 2014/203805 A1& EP 2705665 A1& TW 201309025 A& CN 103503461 A	1-12