

(19)日本国特許庁(JP)

(12)公開特許公報(A)

(11)公開番号
特開2023-123747
(P2023-123747A)

(43)公開日 令和5年9月5日(2023.9.5)

(51)国際特許分類
H 0 4 N 19/70 (2014.01)

F I
H 0 4 N 19/70

審査請求 有 請求項の数 9 O L (全112頁)

(21)出願番号 (22)出願日 (62)分割の表示 原出願日 (31)優先権主張番号 (32)優先日 (33)優先権主張国・地域又は機関 (31)優先権主張番号 (32)優先日 (33)優先権主張国・地域又は機関 (31)優先権主張番号 (32)優先日	特願2023-107242(P2023-107242) 令和5年6月29日(2023.6.29) 特願2021-529357(P2021-529357))の分割 令和1年11月26日(2019.11.26) 62/774,050 平成30年11月30日(2018.11.30) 米国(US) 62/784,296 平成30年12月21日(2018.12.21) 米国(US) 62/791,227 平成31年1月11日(2019.1.11) 最終頁に続く	(71)出願人 (71)出願人	000005049 シャープ株式会社 大阪府堺市堺区匠町1番地 518446879 鴻穎創新有限公司 FG INNOVATION COMPANY LIMITED 中華人民共和國香港新界屯門海榮路22 號屯門中央廣場26樓2623室 Flat 2623, 26/F Tuen Mun Central Square, 22 Hoi Wing Road, Tuen Mun, New Territories, The Hong Kong Special Administrative Region 最終頁に続く
--	---	--------------------	--

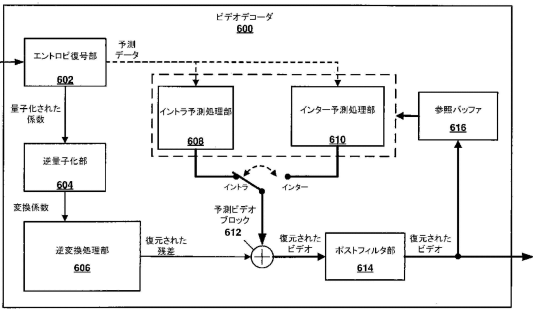
(54)【発明の名称】 符号化されたビデオのピクチャ用のタイル構造をシグナリングするシステム及び方法

(57)【要約】 (修正有)

【課題】 符号化されたビデオのピクチャ用のタイル構造をシグナリングする方法及びデバイスを提供する。

【解決手段】 ビデオデコーダによりビデオデータを復号する方法は、パラメータセット内の第1のフラグシンタックスを復号する。第1のフラグシンタックスは、値が第1の所定の値と等しい場合、少なくとも1つのスライスの各々におけるタイルがラスト走査順であることを指定し、値が第1の所定の値と異なる場合、少なくとも1つのスライスの各々におけるタイルがピクチャの矩形領域をカバーすることを指定する。方法はまた、第1のフラグシンタックスの値が第1の所定の値と等しい場合、スライスアドレスシンタックス、およびスライスヘッダに存在し、スライスに含まれるタイルの数を指定するナンバーシンタックスを復号する。

【選択図】 図9



【特許請求の範囲】**【請求項 1】**

ビデオデータを復号する方法であって、

パラメータセット内の第 1 のフラグシンタックスを復号することであって、前記第 1 のフラグシンタックスは、値が第 1 の所定の値と等しい場合、少なくとも 1 つのスライスの各々におけるタイルがラスト走査順であることを指定し、値が前記第 1 の所定の値と異なる場合、前記少なくとも 1 つのスライスの各々におけるタイルがピクチャの矩形領域をカバーすることを指定する、復号することと、

前記第 1 のフラグシンタックスの前記値が前記第 1 の所定の値と等しい場合、スライスアドレスシンタックス、およびスライスヘッダに存在するナンバーシンタックスであって、スライスに含まれるタイルの数を指定するナンバーシンタックスを復号することと、を含む、方法。

10

【請求項 2】

ビデオデータを符号化する方法であって、

パラメータセット内の第 1 のフラグシンタックスを符号化することであって、前記第 1 のフラグシンタックスは、値が第 1 の所定の値と等しい場合、少なくとも 1 つのスライスの各々におけるタイルがラスト走査順であることを指定し、値が前記第 1 の所定の値と異なる場合、前記少なくとも 1 つのスライスの各々におけるタイルがピクチャの矩形領域をカバーすることを指定する、符号化することと、

前記第 1 のフラグシンタックスの前記値が前記第 1 の所定の値と等しい場合、スライスヘッダに、スライスアドレスシンタックス、およびスライスに含まれるタイルの数を指定するナンバーシンタックスを符号化することと、を含む、方法。

20

【請求項 3】

前記第 1 のフラグシンタックスの値が前記少なくとも 1 つのスライスの各々の中の前記タイルがラスト走査順であることを指定し、且つ前記スライスヘッダに対応するピクチャのタイルの数が 1 より大きい場合に、前記スライスアドレスシンタックスは、前記スライスヘッダに存在する、請求項 1 に記載の方法。

【請求項 4】

前記ナンバーシンタックスの値に 1 加えた値が、前記スライスヘッダに対応する前記少なくとも 1 つのスライスの各々におけるタイルの数を指定する、請求項 1 に記載の方法。

30

【請求項 5】

前記パラメータセットはピクチャパラメータセットである、請求項 1 に記載の方法。

【請求項 6】

前記パラメータセットはシーケンスパラメータセットである、請求項 1 に記載の方法。

【請求項 7】

前記第 1 の所定の値は 0 である、請求項 1 に記載の方法。

40

【請求項 8】

ビデオデータを復号するデバイスであって、

少なくとも 1 つのプロセッサと、前記少なくとも 1 つのプロセッサに結合された記憶デバイスであって、前記少なくとも 1 つのプロセッサによって実行されるプログラムを格納する記憶デバイスと、を備え、前記プログラムの実行によって前記少なくとも 1 つのプロセッサは、

ピクチャパラメータセット内の第 1 のフラグシンタックスを復号することであって、前記第 1 のフラグシンタックスは、値が 0 の場合、前記第 1 のフラグシンタックスは少なく

50

とも1つのスライスの各々におけるタイルがラスト走査順であることを指定し、値が1の場合、前記第1のフラグシンタックスは前記少なくとも1つのスライスの各々におけるタイルがピクチャの矩形領域をカバーすることを指定する、復号を実行し、

前記第1のフラグシンタックスの値が0の場合、スライスアドレスシンタックス、およびおよびスライスヘッダに存在するナンバーシンタックスであって、スライスに含まれるタイルの数を指定するナンバーシンタックスを復号する、デバイス。

【請求項9】

前記第1のフラグシンタックスの値が前記少なくとも1つのスライスの各々の中の前記タイルがラスト走査順であることを指定し、且つ前記スライスヘッダに対応するピクチャのタイルの数が1より大きい場合に、前記スライスアドレスシンタックスは、前記スライスヘッダに存在し、

前記ナンバーシンタックスの値に1加えた値が、前記スライスヘッダに対応する前記少なくとも1つのスライスの各々におけるタイルの数を指定する、請求項8に記載のデバイス。

【発明の詳細な説明】

【技術分野】

【0001】

本開示は、ビデオ符号化に関し、より具体的には、符号化されたビデオのピクチャ用のタイル構造をシグナリングする技術に関する。

【0002】

背景技術

デジタルビデオ能力は、デジタルテレビ、ラップトップ又はデスクトップコンピュータ、タブレット型コンピュータ、デジタル記録デバイス、デジタルメディアプレーヤ、ビデオゲーミングデバイス、いわゆるスマートフォンを含むセルラー電話、医療用イメージングデバイスなどを含む、広範囲のデバイスに組み込むことができる。デジタルビデオは、ビデオ符号化規格に従って符号化することができる。ビデオ符号化規格は、ビデオ圧縮技術を組み込むことができる。ビデオ符号化規格の例としては、ISO/IEC MPEG-4 Visual及びITU-T H.264(ISO/IEC MPEG-4 AVCとしても公知である)並びにHigh-Efficiency Video Coding(HEVC)が挙げられる。HEVCは、High Efficiency Video Coding(HEVC), Rec. ITU-T H.

265(2016年12月)に記載されており、参照により本明細書に組み込まれ、本明細書ではITU-T H.265と称する。ITU-T H.265に関する拡張及び改良が、次世代ビデオ符号化規格の開発のために現在検討されている。例えば、ITU-T Video Coding Experts Group(VCEG)及びISO/IEC (Moving Picture Experts Group(MPEG)(Joint Video Exploration Team (JVET)と集合的に呼ばれる)は、現在のHEVC規格の圧縮能力を著しく上回る圧縮能力を有する将来のビデオ符号化技術の標準化についての潜在的必要性を検討している。The Joint Exploration Model 7 (JEM 7), Algorithm Description of Joint Exploration Test Model 7(JEM 7), ISO/IEC JTC1/SC29/WG11 Document:すなわち、参照により本明細書に組み込まれるJVET-G1001, July 2017, Torino, ITは、ITU-T H.265の能力を超えてビデオ符号化技術を向上させる可能性を有するものとして、JVETによって協調的試験モデル研究のさなかである符号化特徴を記載している。JEM 7の符号化特徴は、JEM参照ソフトウェアで実施されることに留意されたい。本明細書で使用されるとき、JEMという用語は、JEM 7に含まれるアルゴリズム及びJEM参照ソフトウェアの実施を集合的に表し得る。更に、VCEG and MPEGによって共に提起された「Joint Call for Proposals on Video Compression with Capabilities beyond HEVC」に応答して、ビデオ符号化に関する複数の説明が様々なグループによって、10th Meeting of ISO/IEC JTC1/SC29/WG11 16-20 April 2018, San Diego, CA.において提案されている。ビデオ符号化の複数の説明の結果として、ビデオ符号化仕様のドラフトテキストが、「Versatile Video Coding (Draft

1),」 10th Meeting of ISO/IEC JTC1/SC29/WG11 16-20 April 2018, San Diego, CA, document JVET-J1001-v2において説明され、これは本明細書に参考として組み込まれ、JVET-J1001として参照される。本明細書に参考として組み込まれ、JVET-K1001と称する、「Versatile Video Coding(Draft 2),」 11th Meeting of ISO/IEC JTC1/SC29/WG11 10-18 July 2018, Ljubljana, SI, document JVET-K1001-v7、がJVET-J1001の更新版である。更に、本明細書に参考として組み込まれ、JVET-L1001と称する、「Versatile Video Coding(Draft 3),」 12th Meeting of ISO/IEC JTC1/SC29/WG11 3-12 October 2018, Macao, CN, document JVET-L1001-v2、がJVET-K1001の更新版である。

【0003】

ビデオ圧縮技術は、ビデオシーケンス中の固有の冗長性を活用することによって、ビデオデータを記憶及び送信するためのデータ要件を低減させる。ビデオ圧縮技術は、ビデオシーケンスを連続的により小さな部分(すなわち、ビデオシーケンス内のフレームの群、フレームの群内のフレーム、フレーム内のスライス、スライス内の符号化ツリーユニット(例えば、マクロブロック)、符号化ツリーユニット内の符号化ブロックなど)に再分割することができる。イントラ予測符号化技術(例えば、ピクチャ内(空間的))及びインター予測技術(すなわち、ピクチャ間(時間的))を使用して、符号化されるビデオデータのユニットとビデオデータの参照ユニットとの間の差分値を生成することができる。差分値は、残差データと呼ばれることがある。残差データは、量子化された変換係数として符号化することができる。シンタックス要素は、残差データと参照符号化ユニットを関連付けることができる(例えば、イントラ予測モードインデックス、動きベクトル、及びブロックベクトル)。残差データ及びシンタックス要素は、エントロピ符号化することができる。エントロピ符号化された残差データ及びシンタックス要素は、準拠ビットストリームに含めることができる。準拠ビットストリーム及び関連メタデータは、データ構造に従ったフォーマットを有してもよい。

【0004】

発明の概要

一実施例では、ビデオデータを復号する方法であって、ピクチャパラメータセット内の第1のフラグシンタックスを復号することであって、前記第1のフラグシンタックスは、少なくとも1つのスライスの各々におけるタイルがラスト走査順であるか、又は前記少なくとも1つのスライスの各々におけるタイルがピクチャの矩形領域をカバーするかを指定する、復号することと、前記第1のフラグシンタックスの値に基づいて、スライスアドレスシンタックスがスライスヘッダに存在するかを判定することと、前記スライスアドレスシンタックスが前記スライスヘッダに存在する場合に、前記スライスアドレスシンタックスを復号することと、を含む、方法。

【0005】

一実施例では、ビデオデータを符号化する方法であって、ピクチャパラメータセット内の第1のフラグシンタックスを符号化することであって、前記第1のフラグシンタックスは、少なくとも1つのスライスの各々におけるタイルがラスト走査順であるか、又は前記少なくとも1つのスライスの各々におけるタイルがピクチャの矩形領域をカバーするかを指定する、符号化することと、前記第1のフラグシンタックスの値に基づいて、スライスヘッダ内にスライスアドレスシンタックスを符号化するかを判定することと、を含む、方法。

【図面の簡単な説明】

【0006】

【図1】本開示の1つ以上の技術による、ビデオデータを符号化及び復号するように構成することができるシステムの一例を示すブロック図である。

【図2A】本開示の1つ以上の技術による、符号化されたビデオデータ及び対応するデータ構造を示す概念図である。

【図2B】本開示の1つ以上の技術による、符号化されたビデオデータ及び対応するデー

10

20

30

40

50

タ構造を示す概念図である。

【図 2 C】本開示の 1 つ以上の技術による、符号化されたビデオデータ及び対応するデータ構造を示す概念図である。

【図 3】本開示の 1 つ以上の技術による、符号化ビデオデータ及び対応するメタデータをカプセル化するデータ構造を示す概念図である。

【図 4】本開示の 1 つ以上の技術による、ビデオデータを符号化及び復号するように構成することができるシステムの実装形態に含まれ得る構成要素の一例を示す概念図である。

【図 5】本開示の 1 つ以上の技術に係る、ビデオデータを符号化するように構成することができるビデオ符号化装置の一例を示すブロック図である。

【図 6】本開示の 1 つ以上の技術による、タイルセット内に含まれるタイルの秩序化の例を示す概念図である。

【図 7】本開示の 1 つ以上の技術による、タイルセット内に含まれるビデオブロックの順序付けの例を示す概念図である。

【図 8】本開示の 1 つ以上の技術による、バイト範囲情報のシグナリングの一例を示す概念図である。

【図 9】本開示の 1 つ以上の技術による、ビデオデータを復号するように構成することができるビデオデコーダの一例を示すブロック図である。 発明を実施するための形態

【0007】

概して、本開示は、ビデオデータを符号化する様々な技術を説明する。具体的には、本開示では、符号化されたビデオのピクチャのタイル構造をシグナリングするための技術を記載する。本明細書で使用されるとき、タイル構造という用語は、タイルへのピクチャの特定の分割を指し得る。以下に更に詳細に記載されるように、本明細書に記載される技術によれば、ピクチャは、可変サイズタイル及びタイル構造に分割されてもよい。本明細書に記載される技術によるタイル構造のシグナリングは、送信帯域幅を低減させること並びに/又はビデオエンコーダ及び/若しくはデコーダの並列化を促進することによって、ビデオ配信システム性能を改善するのに特に有用であり得る。本開示の技術は、ITU-T H.264 及び ITU-T H.265 に関して記載されているが、本開示の技術は、ビデオ符号化に一般に適用可能であることに留意されたい。例えば、本明細書で説明する符号化技術は、ITU-T H.

265 に含まれるもの以外のブロック構造、イントラ予測技術、インター予測技術、変換技術、フィルタリング技術、及び/又はエントロピ符号化技術を含むビデオ符号化システム(将来のビデオ符号化規格に基づくビデオ符号化システムを含む)に組み込むことができる。従って、ITU-T H.264 及び ITU-T H.265 への参照は、説明のためのものであり、本明細書で説明する技術の範囲を限定するように解釈すべきではない。更に、本明細書の文書の参照による組み込みは、本明細書で使用する用語に関して限定する又は曖昧さを生むように解釈されるべきではないことに留意されたい。例えば、組み込まれた参照が、別の組み込まれた参照のものとは異なる用語の定義を与える場合、かつ/又はその用語が本明細書で 사용되는ような場合には、その用語は、それぞれの対応する定義を幅広く含むように、及び/又は代わりに特定の定義のそれぞれを含むように解釈されたい。

一実施例では、タイルセット構造をシグナリングする方法は、タイルセットがビットストリームで有効になっていることを示すフラグをシグナリングすることと、ピクチャを区画する数タイルセット列を示すシンタックス要素をシグナリングすることと、ピクチャを区画する数タイルセット行を示すシンタックス要素をシグナリングすることと、を含む。

【0008】

一実施例では、デバイスは、タイルセットがビットストリームで有効になっていることを示すフラグをシグナリングし、ピクチャを区画する数タイルセット列を示すシンタックス要素をシグナリングし、ピクチャを区画する数タイルセット行を示すシンタックス要素をシグナリングするように構成された 1 つ以上のプロセッサを備える。

【0009】

一実施例では、非一時的コンピュータ可読記憶媒体は、その上に記憶された命令を含み

10

20

30

40

50

、これが実行された場合には、デバイスの1つ以上のプロセッサに、タイルセットがビットストリームで有効になっていることを示すフラグをシグナリングさせ、ピクチャを区画する数タイルセット列を示すシンタックス要素をシグナリングさせ、ピクチャを区画する数タイルセット行を示すシンタックス要素をシグナリングさせる。

【0010】

一実施例では、装置は、タイルセットがビットストリームで有効になっていることを示すフラグをシグナリングする手段と、ピクチャを区画する数タイルセット列を示すシンタックス要素をシグナリングする手段と、ピクチャを区画する数タイルセット行を示すシンタックス要素をシグナリングする手段と、を備える。

【0011】

一実施例では、ビデオデータを復号する方法は、タイルセットがビットストリームで有効になっていることを示すフラグをパースすることと、ピクチャを区画する数タイルセット列を示すシンタックス要素をパースすることと、ピクチャを区画する数タイルセット行を示すシンタックス要素をパースすることと、パースされたシンタックス要素の値に基づいてビデオデータを生成することと、を含む。

【0012】

一実施例では、デバイスは、タイルセットがビットストリームで有効になっていることを示すフラグをパースし、ピクチャを区画する数タイルセット列を示すシンタックス要素をパースし、ピクチャを区画する数タイルセット行を示すシンタックス要素をパースし、パースされたシンタックス要素の値に基づいてビデオデータを生成するように構成された1つ以上のプロセッサを備える。

【0013】

一実施例では、非一時的コンピュータ可読記憶媒体は、その上に記憶された命令を含み、実行されると、デバイスの1つ以上のプロセッサに、タイルセットがビットストリームで有効になっていることを示すフラグをパースさせ、ピクチャを区画する数タイルセット列を示すシンタックス要素をパースさせ、ピクチャを区画する数タイルセット行を示すシンタックス要素をパースさせ、パースされたシンタックス要素の値に基づいてビデオデータを生成させる。

【0014】

一実施例では、装置は、タイルセットがビットストリームで有効になっていることを示すフラグをパースする手段と、ピクチャを区画する数タイルセット列を示すシンタックス要素をパースする手段と、ピクチャを区画する数タイルセット行を示すシンタックス要素をパースする手段と、パースされたシンタックス要素の値に基づいてビデオデータを生成する手段と、を備える。

【0015】

1つ以上の実施例の詳細は、添付の図面及び以下の明細書に記述されている。他の特徴、目的、及び利点は、明細書及び図面から、並びに特許請求の範囲から明白である。

【0016】

ビデオコンテンツは、典型的には、一連のフレームからなるビデオシーケンスを含む。一連のフレームはまた、ピクチャ群(group of pictures、GOP)と呼ばれることがある。各ビデオフレーム又はピクチャは1つ以上のスライスを含むことができ、スライスは複数のビデオブロックを含む。ビデオブロックは、予測的に符号化することができる画素値(サンプルと呼ばれることもある)のアレイを含む。ビデオブロックは、走査パターン(例えば、ラスタ走査)に従って順序付けすることができる。ビデオエンコーダは、ビデオブロック及びその再分割に対して予測符号化を実行する。ITU-T H.264は、16×16のルマ(luma)サンプルを含むマクロブロックを規定する。ITU-T H.265は、類似の符号化ツリーユニット(Coding Tree Unit、CTU)構造(Largest Coding Unit(LCU)と呼ばれることがある)を規定するが、ピクチャは、等しいサイズのCTUに分割することができ、各CTUは、16×16、32×32、又は64×64のルマサンプルを有する符号化ツリーブロック(Coding Tree Block、CTB)を含むことができる。本明細書で使用されるとき、

10

20

30

40

50

ビデオブロックという用語は、一般に、ピクチャの領域を指すことがあり、又はより具体的には、予測的に符号化できる画素値の最大アレイ、その再分割、及び/又は対応する構造を指すことがある。更に、ITU-T H.265によれば、各ビデオフレーム又はピクチャは、1つ以上のタイルを含むように区画化してもよく、ここでのタイルは、ピクチャの矩形領域に対応する符号化ツリーユニットのシーケンスである。

【0017】

ITU-T H.265では、CTUは、ビデオデータのそれぞれの成分(例えば、ルマ(Y)及びクロマ(Cb及びCr))についてのそれぞれのCTB(Coding Tree Block)から構成される。更に、ITU-T

H.265では、CTUを四分ツリー(QT)分割構造に従って分割することができ、その結果、CTUのCTBが、符号化ブロック(CB:Coding Block)に分割される。すなわち、ITU-T H.265では、CTUを四分ツリーリーフノードに分割することができる。ITU-T H.265によれば、2つの対応するクロマCB及び関連するシンタックス要素を伴う1つのルマCBは、符号化ユニット(CU)と呼ばれる。ITU-T H.265では、CBの最小許容サイズをシグナリングすることができる。ITU-T H.265では、ルマCBの最も小さい最小許容サイズは、 8×8 のルマサンプルである。ITU-T H.265では、イントラ予測又はインター予測を用いてピクチャ領域を符号化する決定は、CUレベルで行われる。

【0018】

ITU-T H.265では、CUは、CUにおけるそのルートを有する予測ユニット(PU)構造に関連付けられている。ITU-T H.265では、PU構造は、対応する参照サンプルを生成する目的で、ルマCB及びクロマCBを分割することを可能にする。すなわち、ITU-T H.265では、ルマCB及びクロマCBを、それぞれのルマ及びクロマ予測ブロック(PB)に分割することができ、ここでPBは、同じ予測が適用されるサンプル値のブロックを含む。ITU-T H.265では、CBを1、2、又は4個のPBに分割することができる。ITU-T H.265は、 64×64 のサンプルから 4×4 のサンプルまでのPBサイズをサポートする。ITU-T H.265では、正方形のPBがイントラ予測のためにサポートされ、ここでCBはPBを形成することができるか、又はCBを4つの正方形のPBに分割することができる(すなわち、イントラ予測PBサイズタイプは、 $M \times M$ 又は $M/2 \times M/2$ を含む。但し、 M は正方形のCBの高さ及び幅である)。ITU-T H.265では、正方形のPBの他に、矩形のPBがインター予測のためにサポートされ、ここでCBを垂直又は水平に二等分してPBを形成することができる(すなわち、インター予測PBタイプは $M \times M$ 、 $M/2 \times M/2$ 、 $M/2 \times M$ 、又は $M \times M/2$ を含む)。更に、ITU-T H.265では、インター予測のために、4つの非対称PB分割がサポートされ、ここでCBは、CBの高さ(上部又は下部で)又は幅(左又は右)の4分の1で2つのPBに分割される(すなわち、非対称区分としては、 $M/4 \times M$ 左、 $M/4 \times M$ 右、 $M \times M/4$ 上部、及び $M \times M/4$ 下部を含む)ことに留意されたい。PBに対応するイントラ予測データ(例えば、イントラ予測モードシンタックス要素)又はインター予測データ(例えば、動きデータシンタックス要素)を使用して、PBに関する参照サンプル値及び/又は予測サンプル値が生成される。

【0019】

JEMは、 256×256 のルマサンプルの最大サイズを有するCTUを規定している。JEMは、四分ツリー+二分ツリー(QTBT)ブロック構造を規定している。JEMでは、QTBT構造は、四分ツリーリーフノードを二分ツリー構造(BT)によって更に分割することを可能にする。すなわち、JEMでは、二分ツリー構造は、四分ツリーリーフノードを垂直又は水平に再帰的に分割することを可能にする。したがって、JEMでの二分ツリー構造は、正方形のリーフノード及び矩形のリーフノードを可能にし、それぞれのリーフノードは、1つのCBを含む。図2Aに示すように、GOPに含まれるピクチャは複数のスライスを含むことができ、各スライスは一連のCTUを含み、各CTUはQTBT構造に従って分割することができる。JEMでは、CBは、追加の分割を行わない予測のために使用される。すなわち、JEMでは、CBは、同じ予測が適用されるサンプル値のブロックであってよい。したがって、JEM QTBTリーフノードは、ITU-T H.265におけるPBに類似するものであっ

てよい。

【0020】

イントラ予測データ(例えば、イントラ予測モードシンタックス要素)又はインター予測データ(例えば、動きデータシンタックス要素)は、PUを対応する参照サンプルに関連させることができる。残差データは、ビデオデータの各成分(例えば、ルマ(Y)及びクロマ(Cb及びCr))に対応する差分値のそれぞれのアレイを含むことができる。残差データは、画素領域内とすることができる。離散コサイン変換(discrete cosine transform、DCT)、離散サイン変換(discrete sine transform、DST)、整数変換、ウェーブレット変換、又は概念的に類似の変換などの変換を、画素差分値に適用して、変換係数を生成することができる。ITU-T H.265では、CUは、更に変換ユニット(Transform Unit、TU)に再分割できることに留意されたい。すなわち、画素差分値のアレイは、変換係数を生成するために再分割することができ(例えば、4つの8×8変換を、16×16のルマCBに対応する残差値の16×16のアレイに適用することができる)、そのような再分割は、変換ブロック(Transform Block、TB)と呼ばれることがある。変換係数は、量子化パラメータ(quantization parameter、QP)に従って量子化され得る。量子化された変換係数(これはレベル値と呼ばれることがある)は、エントロピ符号化技術(例えば、コンテンツ適応可変長符号化(content adaptive variable length coding、CAVLC)、コンテキスト適応2値算術符号化(context adaptive binary arithmetic coding、CABAC)、確率区間分割エントロピ符号化(probability interval partitioning entropy coding、PIPE)など)に従ってエントロピ符号化することができる。更に、シンタックス要素、例えば、予測モードを示すシンタックス要素なども、エントロピ符号化することができる。エントロピ符号化され量子化された変換係数及び対応するエントロピ符号化されたシンタックス要素は、ビデオデータを再生成するために使用することができる準拠ビットストリームを形成することができる。二値化プロセスを、エントロピ符号化プロセスの一部としてシンタックス要素に対して実行することができる。二値化は、シンタックス値を一連の1つ以上のビットに変換するプロセスを指す。これらのビットは、「ビン」と呼ばれることがある。

10

20

30

40

50

【0021】

上述したように、イントラ予測データ又はインター予測データは、サンプル値のブロックに対する参照サンプル値を生成するために使用される。現在のPB又は別の種類のピクチャ領域構造に含まれるサンプル値と、関連する参照サンプル(例えば、予測を用いて生成された参照サンプル)との差は、残差データと呼ばれることがある。上述したように、イントラ予測データ又はインター予測データは、ピクチャ(例えば、PB又はCB)の領域を、対応する参照サンプルと関連付けることができる。イントラ予測符号化のために、イントラ予測モードは、ピクチャ内の参照サンプルの位置を指定することができる。ITU-T H.265では、定義された可能なイントラ予測モードは、プレーナ(すなわち、表面適合)予測モード(predMode:0)、DC(すなわち、フラット全体平均化)予測モード(predMode:1)、及び33角度予測モード(predMode:2-34)を含む。JEMでは、定義された可能なイントラ予測モードは、プレーナ予測モード(predMode:0)、DC予測モード(predMode:1)、及び65角度予測モード(predMode:2-66)を含む。プレーナ及びDC予測モードは、非方向性予測モードと呼ばれることがあり、角度予測モードは、方向性予測モードと呼ばれることがあることに留意されたい。本明細書で説明する技術は、定義された可能な予測モードの数にかかわらず、一般的に適用可能であり得ることに留意されたい。

【0022】

インター予測符号化のために、動きベクトル(MV)は、符号化されるビデオブロックのピクチャ以外のピクチャ内の参照サンプルを識別し、それによってビデオの時間的冗長性を利用する。例えば、現在のビデオブロックは、以前に符号化されたフレーム(単数又は複数)内に位置する参照ブロック(単数又は複数)から予測することができ、動きベクトルは、参照ブロックの位置を示すために使用することができる。動きベクトル及び関連するデータは、例えば、動きベクトルの水平成分、動きベクトルの垂直成分、動きベクトルに

対する解像度(例えば、四分の一画素の精度、二分の一画素の精度、一画素の精度、二画素の精度、四画素の精度)、予測方向、及び/又は参照ピクチャのインデックス値を記述することができる。更に、例えば、ITU-T H.265などの符号化規格は、動きベクトル予測をサポートすることができる。動きベクトル予測は、隣接するブロックの動きベクトルを用いて動きベクトルを指定することを可能にする。動きベクトル予測の例としては、高度動きベクトル予測(advanced motion vector prediction)(AMVP)、時間動きベクトル予測(temporal motion vector prediction)(TMVP)、いわゆる「結合」モード、並びに「スキップ」及び「ダイレクト」動き推測が挙げられる。更に、JEMは、高度時間動きベクトル予測(advanced temporal motion vector prediction)(ATMVP)、時空間動きベクトル予測(spatial-temporal motion vector prediction)(STMVP)、フレームレートアップコンバージョン(FRUC)技術に基づく特別なマージモードであるパターンマッチ運動ベクトル導出(PMMVD)モード、及びアフィン変換動き補償予測をサポートする。

10

【0023】

残差データは、ビデオデータのそれぞれの成分に対応する差分値のそれぞれの配列を含むことができる。残差データは、画素領域内とすることができる。離散コサイン変換(discrete cosine transform)(DCT)、離散サイン変換(discrete sine transform)(DST)、整数変換、ウェーブレット変換、又は概念的に類似の変換などの変換を、差分値の配列に適用して、変換係数を生成することができる。ITU-T H.265では、CUは、CUレベルにおけるそのルートを有する変換ユニット(TU)構造に関連付けられている。すなわち、ITU-T H.265では、上記のように、変換係数を生成する目的で、差分値の配列を再分割することができる(例えば、4つの 8×8 変換を残差値の 16×16 の配列に適用することができる)。ITU-T H.265では、必ずしもTBをPBに整列する必要はないことに留意されたい。

20

【0024】

JEMでは、CBに対応する残差値を使用して、更なる分割を行うことなく変換係数が生成されることに留意されたい。すなわち、JEMでは、QTBTリーフノードは、ITU-T H.265におけるPB及びTBの両方に類似のものとすることができる。JEMでは、コア変換及び後続の二次変換を(ビデオエンコーダで)適用して変換係数を生成することができる。ビデオデコーダに対しては、変換の順序は逆転される。更に、JEMでは、二次変換を適用して変換係数を生成するか否かは、予測モードに依存し得る。

30

【0025】

量子化プロセスを、変換係数に対して実行することができる。量子化を、特定の値のセットに制限された振幅によって変換係数に近似する。量子化は、変換係数の群を表すために必要なデータの量を変化させるために使用されてもよい。量子化は、倍率による変換係数の除算及び任意の関連する丸め機能(例えば、最も近い整数への丸め)によって実現され得る。量子化された変換係数は、係数レベル値と呼ばれることがある。逆量子化(Inverse

quantization)(又は「脱量子化(dequantization)」)は、倍率による係数レベル値の乗算を含むことができる。本明細書で使用されるとき、量子化プロセスという用語は、いくつかの場合では、レベル値を生成する倍率による除算、又は、いくつかの場合では、変換係数を回復する倍率による乗算を指す場合があることに留意されたい。すなわち、量子化プロセスは、いくつかの場合では、量子化、及び、いくつかの場合では、逆量子化を指すことがある。

40

【0026】

本明細書で使用される式に関して、以下の算術演算子が使用され得る。

【0027】

+ 加算
- 減算
* 行列乗算を含む乗算

50

x^y べき乗。 x の y 乗を指定する。他のコンテキストでは、そのような表記は、べき乗としての解釈を意図していないスークリプトに使用される。

【0028】

/ ゼロへの結果切り捨てを伴う整数除算。例えば、 $7/4$ 及び $-7/-4$ は、1に切り捨てられ、 $-7/4$ 及び $7/-4$ は、-1に切り捨てられる。

【0029】

÷ 切り捨て又は四捨五入が意図されていない式において除算を表すために使用される。

【0030】

x/y 切り捨て又は四捨五入が意図されていない式において除算を表すために使用される。

【0031】

更に、以下の数学関数を使用することができる。

【0032】

$\text{Log}_2(x)$ 、底を2とする x の対数；

【0033】

【数1】

$$\text{Min}(x, y) = \begin{cases} x & ; x \leq y \\ y & ; x > y \end{cases}$$

$$\text{Max}(x, y) = \begin{cases} x & ; x \geq y \\ y & ; x < y \end{cases}$$

【0034】

$\text{Ceil}(x)$ は、 x 以上の最小整数である。

【0035】

本明細書で使用するシンタックスの例に関して、以下の論理演算子の定義が適用される。

【0036】

$x \& \& y$ x と y とのブール論理「積」

$x \mid \mid y$ x と y とのブール論理「和」

!ブール論理「否」

$x ? y : z$ x が真であるか又は0に等しくない場合は y の値を評価し、そうでない場合は z の値を評価する。

【0037】

更に、以下の関係演算子が適用されてもよい。

【0038】

> 大なり

>= 大なり又は等しい

< 小なり

<= 小なり又は等しい

== 等しい

!= 等しくない

更に、本明細書で使用するシンタックス記述子において、以下の記述子が適用されてもよいことに留意されたい。

【0039】

-u(n): n ビットを使用した符号なし整数。

【0040】

-ue(v):左ビットを最初に有する、符号なしの整数0次ゴロム符号化シンタックス要素。

10

20

30

40

50

【0041】

仮想現実(VR)アプリケーションは、ヘッドマウントディスプレイでレンダリングすることができるビデオコンテンツを含むことができ、ユーザの頭部の向きに対応する全天球映像の領域のみがレンダリングされる。VRアプリケーションは、360°球面ビデオとも呼ばれる、全方向ビデオによって可能とされ得る。全方位ビデオは、典型的には、最大360°のシーンをカバーする複数のカメラによってキャプチャされる。通常のビデオと比較した全方位ビデオの明確な特徴は、典型的には、キャプチャされたビデオ領域全体のサブセットのみが表示されること、すなわち、現在のユーザの視野(FOV)に対応する領域が表示されることである。FOVはまた、時に、ビューポートとも呼ばれる。他の場合では、ビューポートは、現在表示され、ユーザによって見られている球面ビデオの一部であってもよい。ビューポートのサイズは、視野以下でもよいことに留意されたい。

10

【0042】

全方向ビデオピクチャにおける最も関心のある領域は、その写真の提示時に(すなわち、FOVにある可能性が最も高い)ユーザにレンダリングされる可能性が統計的に最も高い、ビデオ領域全体のサブセットを指し得る。全方向ビデオの最も関心のある領域は、ディレクタ若しくはプロデューサの意図によって決定されてもよく、又はサービス若しくはコンテンツプロバイダによるユーザ統計から(例えば、全方向ビデオコンテンツがストリーミングサービスを通じて提供されたときに、ユーザによって最も多数のリクエストされた/見られた領域の統計を通じて)導出されてもよいことに留意されたい。最も関心のある領域は、エッジサーバ若しくはクライアントによる全方向ビデオ適応ストリーミングにおけるデータプリフェッチのために、及び/又は全方向ビデオが、例えば、異なるコーデック又はプロジェクションマッピングにトランスコードされるときにトランスコーディング最適化のために使用することができる。従って、全方向ビデオピクチャにおける最も関心のある領域をシグナリングすることは、送信帯域幅を低減させ、かつ、復号複雑性を低減させることによって、システム性能を改善することができる。ベース領域は、一般に、符号化ビデオデータの全体領域、例えば、ビデオ領域全体を指すことに留意されたい。

20

【0043】

上述のように、ITU-T H.265によれば、各ビデオフレーム又はピクチャは、1つ以上のスライスを含むように区画化してもよく、1つ以上のタイルを含むように更に区画化してもよい。図2A～図2Cは、スライスを含み、ピクチャを更にタイルに区画化しているピクチャ群の一例を示す概念図である。図2Aに示す例では、Pic₄は、2つのスライス(すなわち、Slice₁及びSlice₂)を含むものとして示されており、各スライスは(例えばラスタ走査順に)CTUのシーケンスを含む。スライスは、独立したスライスセグメントから始まる1つ以上のスライスセグメントのシーケンスであり、全ての後続の依存スライスセグメント(もしあれば)を含み、この後において同じアクセスユニット内の次の独立したスライスセグメント(もしあれば)が続くことに留意されたい。スライスのようなスライスセグメントは、符号化ツリーユニットのシーケンスである。本明細書に記載される実施例では、いくつかの場合では、スライス及びスライスセグメントという用語は、符号化ツリーユニットの結果を示すために互換的に使用され得る。図2Bに示す例では、Pic₄は、6つのタイル(すなわち、Tile₁～Tile₆)を含むものとして示されており、各タイルは矩形でありCTUのシーケンスを含む。ITU-T H.265では、タイルは、2つ以上のスライスに包含される符号化ツリーユニットからなってもよく、スライスは、2つ以上のタイルに包含される符号化ツリーユニットからなってもよいことに留意されたい。しかしながら、ITU-T H.265は、以下の条件のうちの1つ又は両方が満たされなければならないと規定している。(1)あるスライス中の全ての符号化ツリーユニットは同じタイルに属する、及び(2)あるタイル内の全ての符号化ツリーユニットは同じスライスに属する。したがって、例えば、図2Bに関して、タイルの全ては単一のスライスに属してもよく、又はタイルは複数のスライスに属してもよい(例えば、Tile₁～Tile₃は、Slice₁に属してもよく、Tile₄～Tile₆は、Slice₂に属してもよい)。JVET-L1001に関して、整数個の完全なCTUからなる必要があるだけでなく、スライスは、整数個の完全なタイルで構成されている

30

40

50

必要があることが提案されている。したがって、ピクチャの矩形領域を形成しない1組のCTUを含むスライス、いくつかのビデオ符号化技術でサポートされても、サポートされなくてもよい。更に、整数個の完全なタイルからなる必要があるスライスをタイルグループと呼ぶ。本明細書で説明する技術は、スライス、タイル、及び/又はタイルグループに適用可能であり得る。

【0044】

更に、図2Bに示すように、タイルはタイルセットを形成してもよい(すなわち、Tile₂及びTile₃はタイルセットを形成する)。タイルセットは、依存性を符号化するための境界(例えば、イントラ予測依存性、エントロピ符号化依存性など)を定義するために使用することができ、そのため、符号化及び関心領域符号化における並列処理を可能にすることができる。例えば、図2Bに示す例のビデオシーケンスが夜のニュース番組に対応する場合、Tile₂及びTile₃によって形成されたタイルセットは、ニュースを読んでいるニュースアンカーを含む視覚的関心領域に対応してもよい。ITU-T H.265は、動き制約タイルセット(MCTS)を有効にするシグナリングを定義する。動き制約タイルセットは、ピクチャ間予測依存性が参照ピクチャ内のコロケーションされたタイルセットに限定されるタイルセットを含んでもよい。この結果、所与のMCTSに対する動き補償を、MCTSの外側にある他のタイルセットの復号とは無関係に実行することが可能である。例えば、図2Bを参照すると、Tile₂及びTile₃によって形成されるタイルセットがMCTSであり、かつ、Pic₁~Pic₃のそれぞれがコロケーションされたタイルセットを含む場合、Tile₂及びTile₃上における動き補償は、Pic₄のTile₁、Tile₄、Tile₅及びTile₆、並びに、Pic₁~Pic₃のそれぞれにおけるタイルTile₁、Tile₄、Tile₅及びTile₆とコロケーションされたタイルの符号化とは無関係に実行されてもよい。MCTSに従ってビデオデータを符号化することは、全方向ビデオプレゼンテーションを含むビデオアプリケーションに有用であり得る。

【0045】

図2Cに示すように、Tile₁~Tile₆は、全方位ビデオの最も関心の高い領域を形成することができる。更に、Tile₂及びTile₃によって形成されるタイルセットは、最も関心のある領域内に含まれるMCTSであってもよい。ビューポート依存ビデオ符号化は、ビューポート依存部分ビデオ符号化とも呼ばれることもあり、ビデオ領域全体の一部のみの複合化を可能にするために使用することができる。すなわち、例えば、ビューポート依存ビデオ符号化を使用して、現在のFOVをレンダリングするのに十分な情報を提供することができる。例えば、全方向ビデオは、ビューポートをカバーする可能性のある各領域が経時的に他の領域から独立して復号化され得るように、MCTSを使用して符号化することができる。この場合、例えば、特定の現在のビューポートに対して、ビューポートをカバーするタイルの最小セットがクライアントに送信され、復号され、及び/又はレンダリングされてもよい。このプロセスは、単純タイルベース部分復号(STPD)と呼ばれることがある。

【0046】

ITU-T H.265では、符号化ビデオシーケンス(CVS)は、アクセスユニットのシーケンスとしてカプセル化(又は構造化)されてもよく、各アクセスユニットは、ネットワーク抽象レイヤ(NAL)ユニットとして構造化されたビデオデータを含む。ITU-T H.265では、ビットストリームは、1つ以上のCVSを形成するNALユニットのシーケンスを含むものとして説明されている。ITU-T H.265は、フォーマット範囲拡張(RExt)、スケーラビリティ(SHVC)、マルチビュー(MV-HEVC)及び3-D(3D-HEVC)を含むマルチレイヤ拡張をサポートすることに留意されたい。マルチレイヤ拡張は、ビデオ表示がベースレイヤ及び1つ以上の追加のエンハンスメントレイヤを含むことを可能にする。例えば、ベースレイヤは、基本レベルの品質(例えば、高解像度レンダリング)を有するビデオ表示を可能にすることができ、エンハンスメントレイヤは、強化された品質レベル(例えば、超高解像度レンダリング)を有するビデオ表示を可能にすることができる。ITU-T H.265では、ベースレイヤを参照することによってエンハンスメントレイヤを符号化することができる

。すなわち、例えば、エンハンスメントレイヤ内のピクチャは、ベースレイヤ内の1つ以上のピクチャ(その拡大/縮小したバージョンを含む)を参照することによって、(例えば、インター予測技術を使用して)符号化されてもよい。ITU-T H.265では、各NALユニットは、NALユニットが関連付けられているビデオデータのレイヤを示す識別子を含んでもよい。図2Aに示す例を参照すると、Pic₄に含まれるビデオデータの各スライス(すなわち、Slice₁及びSlice₂)は、NALユニットにカプセル化されているものとして示されている。更に、ITU-T H.265では、ビデオシーケンス、GOP、ピクチャ、スライス、及びCTUのそれぞれは、ビデオ符号化特性を記述するメタデータに関連付けられてもよい。ITU-T H.265は、ビデオデータ特性及び/又はビデオ符号化特性を記述するために使用され得るパラメータセットを定義する。ITU-T H.265では、パラメータセットは、特殊な種別のNALユニットとしてカプセル化されてもよく、又はメッセージとしてシグナリングされてもよい。符号化ビデオデータ(例えばスライス)を含むNALユニットは、VCL(ビデオ符号化レイヤ)NALユニットと呼ばれることがあり、メタデータ(例えばパラメータセット)を含むNALユニットは、非VCL NALユニットと呼ばれることがある。更に、ITU-T H.265は、補足エンハンスメント情報(SEI)メッセージがシグナリングされることを可能にする。ITU-T H.265では、SEIメッセージは、復号、表示、又は他の目的に関連するプロセスを支援するが、SEIメッセージは、復号プロセスによってルマ又はクロマサンプルを作成するために必要とされない場合がある。ITU-T H.265では、SEIメッセージが、非VCL NALユニットを使用してビットストリームでシグナリングされてもよい。更に、SEIメッセージは、ビットストリーム内に存在することによる以外の何らかの手段(すなわち、帯域外でシグナリングされる)によって伝達されてもよい。

10

20

【0047】

図3は、複数のCVSを含むビットストリームの一例を示しており、CVSは、それぞれのアクセスユニットに含まれるNALユニットによって表される。図3に例示する実施例では、非VCL NALユニットは、それぞれのパラメータセットユニット(すなわち、ビデオパラメータセット(VPS)、シーケンスパラメータセット(SPS)、及びピクチャパラメータセット(PPS)ユニット)、及びアクセスユニットデリミタNALユニットを含む。ITU-T H.265は、NALユニットに含まれる生バイトシーケンスペイロード(RBSP)データ構造の種別を明示するNALユニットヘッダセマンティクスを定義することに留意されたい。上述したように、MCTSを使用して、全方向ビデオを符号化することができる。サブビットストリーム抽出とは、受信ビットストリーム内のデータを破棄及び/又は修正することによって、ITU-T H.265準拠ビットストリームを受信しているデバイスが、新たなITU-T H.265準拠ビットストリームを形成するプロセスを指し得る。例えば、上述したように、特定の現在のビューポートに対して、ビューポートをカバーするタイルの最小セットがクライアントに送信されてもよい。サブビットストリーム抽出を使用して、タイルの最小セットを含む新しいITU-T H.265準拠ビットストリームを形成することができる。例えば、図2Cを参照すると、ビューポートが、Tile₂及びTile₃のみを含み、ビットストリーム内のアクセスユニットが、Tile₁~Tile₆のVCL NALユニットを含み、Tile₁、Tile₂、及びTile₃は、第1のスライスに含まれ、Tile₄、Tile₅、及びTile₆は、第2のスライスに含まれる場合、サブビットストリーム抽出プロセスは、Tile₂及びTile₃を含むスライスのVCL NALユニットのみを含む新しいビットストリームを生成することを含み得る(すなわち、Tile₄、Tile₅、及びTile₆を含むスライスを含むVCL NALユニットは、受信したビットストリームから除去される)。

30

40

【0048】

上述したように、タイル構造という用語は、タイルへのピクチャの特定の分割を指し得る。図2Bを参照すると、Pic₄のタイル構造は、図示されたタイルTile₁~Tile₆を含む。いくつかの場合では、異なるピクチャに異なるタイル構造を使用することが有用であり得る。ITU-T H.265では、ピクチャのタイル構造は、ピクチャパラメータセットを使用してシグナリングされる。表1は、タイル構造をシグナリングするための関連するシンタックス要素を含むITU-T H.265に指定されたPPSのシンタックスの一部である。

50

【 0 0 4 9 】

【 表 1 】

表1

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
...	
tiles_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tiles_columns_minus1; i++)	
column_width_minus1[i]	ue(v)
for(i = 0; i < num_tiles_rows_minus1; i++)	
row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
...	

【 0 0 5 0 】

ITU-T H.265は、表1に示されたそれぞれのシンタックス要素に対して以下の定義を規定する。pps_pic_parameter_set_idは、他のシンタックス要素による参照のためのPPSを指定する。pps_pic_parameter_set_idの値は、両端値を含む、0～63の範囲にあるものとする。

【 0 0 5 1 】

pps_seq_parameter_set_idは、アクティブSPSについてのsps_seq_parameter_set_idの値を指定する。pps_seq_parameter_set_idの値は、両端値を含む、0～15の範囲にあるものとする。

【 0 0 5 2 】

1に等しいtiles_enabled_flagは、PPSを参照して、各ピクチャに2つ以上のタイルが存在することを指定する。0に等しいtiles_enabled_flagは、PPSを参照して、各ピクチャに1つのタイルのみが存在することを指定する。CVS内でアクティブ化される全てのPPSについては、tiles_enabled_flagの値が同じであることが、ビットストリームの適合性の要件である。

【 0 0 5 3 】

num_tile_columns_minus1プラス1は、ピクチャを区画するタイル列の数を指定する。num_tile_columns_minus1は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_columns_minus1の値が存在しない場合、num_tile_columns_minus1の値は0に等しいと推測される。

【0054】

num_tile_rows_minus1プラス1は、ピクチャを区画するタイル行の数を指定する。num_tile_rows_minus1は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_rows_minus1の値が存在しない場合、num_tile_rows_minus1の値は0に等しいと推測される。tiles_enabled_flagが1に等しい場合、num_tile_columns_minus1及びnum_tile_rows_minus1は、いずれも0に等しくないものとする。

10

【0055】

1に等しいuniform_spacing_flagは、タイル列境界及び同様にタイルの行境界が、ピクチャにわたって均一に分布されることを指定する。0に等しいuniform_spacing_flagは、タイル列境界及び同様にタイルの行境界が、ピクチャにわたって均一に分布されないが、シンタックス要素column_width_minus1[i]及びrow_height_minus1[i]を使用して明示的にシグナリングされることを指定する。uniform_spacing_flagの値が存在しない場合、uniform_spacing_flagの値は1に等しいと推測される。

【0056】

column_width_minus1[i]プラス1は、符号化ツリーブロックの単位におけるi番目のタイル列の幅を指定する。

20

【0057】

row_height_minus1[i]プラス1は、符号化ツリーブロックの単位におけるi番目のタイル行の高さを指定する。

【0058】

更に、ITU-T H.265では、ビットストリーム内のエントリポイントに関する情報は、スライスセグメントヘッダを使用してシグナリングされる。表2は、エントリポイントをシグナリングするための関連するシンタックス要素を含むITU-T H.265に指定されたスライスセグメントヘッダのシンタックスの一部である。

【0059】

30

40

50

【表 2】

表2

slice_segment_header() {	記述子
...	
slice_pic_parameter_set_id	ue(v)
...	
if(tiles_enabled_flag entropy_coding_sync_enabled_flag) {	
num_entry_point_offsets	ue(v)
if(num_entry_point_offsets > 0) {	
offset_len_minus1	uc(v)
for(i = 0; i < num_entry_point_offsets; i++)	
entry_point_offset_minus1 [i]	u(v)
}	
}	
...	
}	

10

20

【0060】

ITU-T H.265は、表2に示されたそれぞれのシンタックス要素に対して以下の定義を規定する。

【0061】

slice_pic_parameter_set_idは、使用中のPPSのpps_pic_parameter_set_idの値を指定する。slice_pic_parameter_set_idの値は、両端値を含む、0～63の範囲にあるものとする。

30

【0062】

num_entry_point_offsetsは、スライスヘッダ内のentry_point_offset_minus1[i]シンタックス要素の数を指定する。num_entry_point_offsetsの値が存在しない場合、num_entry_point_offsetsの値は0に等しいと推測される。

【0063】

num_entry_point_offsetの値は、以下のように制約される。

【0064】

-tiles_enabled_flagが0に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、num_entry_point_offsetsの値は、両端値を含む、0～PicHeightInCtbsY-1の範囲であるものとする。

40

【0065】

-そうでなければ、tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが0に等しい場合、num_entry_point_offsetsの値は、両端値を含む、0～(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1)-1の範囲であるものとする。

【0066】

-そうでなければ、tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、num_entry_point_offsetsの値は、両端値を含む、0

50

～(num_tile_columns_minus1+1)*PicHeightInCtbsY-1の範囲であるものとする。

【0067】

offset_len_minus1プラス1は、entry_point_offset_minus1[i]シンタックス要素の長さ(ビット)を指定する。offset_len_minus1の値は、両端値を含む、0～31の範囲にあるもの

とする。

【0068】

entry_point_offset_minus1[i]プラス1は、i番目のエントリポイントオフセット(バイト)を指定し、offset_len_minus1プラス1ビットで表される。スライスセグメントデータは、スライスセグメントのヘッダに続き、num_entry_point_offset+1サブセットからなり、サブセットインデックス値は、両端値を含む0～num_entry_point_offsetの範囲である。スライスセグメントデータの第1のバイトは、バイト0と見なされる。存在する場合、符号化スライスセグメントNALユニットのスライスセグメントデータ部分に現れるエミュレーション防止バイトは、サブセット識別の目的でスライスセグメントデータの一部としてカウントされる。サブセット0は、符号化スライスセグメントデータのバイト0からentry_point_offset_minus1[0](両端値を含む)からなり、kが1～num_entry_point_offsets-1(両端値を含む)の範囲のkを有するサブセットkは、符号化スライスセグメントデータのバイトfirstByte[k]～lastByte[k](両端値を含む)からなり、firstByte[k]及びlastByte[k]は、以下のとおり定義される。

【0069】

【数2】

$$\text{firstByte}[k] = \sum_{n=1}^k (\text{entry_point_offset_minus1}[n-1] + 1)$$

$$\text{lastByte}[k] = \text{firstByte}[k] + \text{entry_point_offset_minus1}[k]$$

【0070】

最後のサブセット(num_entry_point_offsetに等しいサブセットインデックスを有する)は、符号化スライスセグメントデータのバイトからなる。

【0071】

tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが0に等しい場合、各サブセットは、同じタイル内にあるスライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなり、サブセットの数(すなわち、num_entry_point_offsets+1の値)は、符号化スライスセグメントである符号化ツリーユニットを含むタイルの数に等しいものとする。

【0072】

注6-tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが0に等しい場合、各スライスは、1つのタイルの符号化ツリーユニットのサブセット(この場合、シンタックス要素entry_point_offset_minus1[i]は存在しない)を含む必要があるか、又は完全なタイルの整数の全ての符号化ツリーユニットを含む必要がある。

【0073】

tiles_enabled_flagが0に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、kが0～num_entry_point_offsets(両端値を含む)の範囲のkを有する各サブセットkは、ピクチャの同じルマ符号化ツリーブロック行内にあるルマ符号化ツリーブロックを含むスライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなり、サブセットの数(すなわち、num_entry_point_offsets+1の値)は、符号化スライスセグメント内にある符号化ツリーユニットを含むピクチャの符号化ツリーブロック行の数に等しいものとする。

10

20

30

40

50

【0074】

注7-最後のサブセット(すなわち、num_entry_point_offsetsに等しいkのサブセットk)は、ピクチャのルマ符号化ツリーブロック行にあるルマ符号化ツリーブロックを含む全ての符号化ツリーユニットを含んでも含まなくてもよい。

【0075】

tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、kが1～num_entry_point_offsets(両端値を含む)の範囲のkを有する各サブセットkは、タイルの同じルマ符号化ツリーブロック行内にあるルマ符号化ツリーブロックを含むスライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなり、サブセットの数(すなわち、num_entry_point_offsets+1の値)は、符号化スライスセグメント内にある符号化ツリーユニットを含むタイルのルマ符号化ツリーブロック行の数に等しいものとする。

10

【0076】

上記のシンタックス及びセマンティックに示すように、ITU-T H.265では、タイル構造は多数の列及び数字の行によって指定され、したがって、各行及び列が同じ数のタイルを含むという点で制限される。このようにしてタイル構造を制限することは、理想的ではない場合がある。本明細書に記載される技術によれば、ビデオエンコーダは、増加した柔軟性を提供する様式でタイル構造及びタイルセットをシグナリングすることができる。

【0077】

図1は、本開示の1つ以上の技術による、ビデオデータを符号化する(符号化及び/又は復号する)ように構成することができる、システムの例を示すブロック図である。システム100は、本開示の1つ以上の技術に従って、ビデオデータをカプセル化することができるシステムの例を表す。図1に示すように、システム100は、ソースデバイス102と、通信媒体110と、目的デバイス120と、を含む。図1に示す例では、ソースデバイス102は、ビデオデータを符号化し、符号化したビデオデータを通信媒体110に送信するように構成された、任意のデバイスを含むことができる。目的デバイス120は、通信媒体110を介して符号化したビデオデータを受信し、符号化したビデオデータを復号するように構成された、任意のデバイスを含むことができる。ソースデバイス102及び/又は目的デバイス120は、有線及び/又は無線通信用に装備された演算デバイスを含むことができ、かつ、例えば、セットトップボックス、デジタルビデオレコーダ、テレビ、デスクトップ、ラップトップ、又はタブレットコンピュータ、ゲーム機、医療用撮像デバイス、及び、例えば、スマートフォン、セルラー電話、パーソナルゲームデバイスを含むモバイルデバイス、を含むことができる。

20

30

【0078】

通信媒体110は、無線及び有線の通信媒体並びに/又は記憶デバイスの任意の組み合わせを含むことができる。通信媒体110としては、同軸ケーブル、光ファイバケーブル、ツイストペアケーブル、無線送信機及び受信機、ルータ、スイッチ、リピータ、基地局、又は様々なデバイスとサイトとの間の通信を容易にするために有用であり得る任意の他の機器を挙げることができる。通信媒体110は、1つ以上のネットワークを含むことができる。例えば、通信媒体110は、ワールドワイドウェブ、例えば、インターネットへのアクセスを可能にするように構成されたネットワークを含むことができる。ネットワークは、1つ以上の電気通信プロトコルの組み合わせに従って動作することができる。電気通信プロトコルは、専用の態様を含むことができ、及び/又は規格化された電気通信プロトコルを含むことができる。標準化された電気通信プロトコルの例としては、Digital Video Broadcasting(DVB)規格、Advanced Television Systems Committee(ATSC)規格、Integrated Services Digital Broadcasting(ISDB)規格、Data Over Cable Service Interface Specification(DOCSIS)規格、Global System Mobile Communications(GSM)規格、符号分割多重アクセス(code division multiple access, CDMA)規格、第三世代パートナーシッププロジェクト(3rd Generation Partnership Project, 3GPP)規格、欧州電気通信標準化機構(European Telecommunicati

40

50

ons Standards Institute、ETSI)規格、インターネットプロトコル(Internet Protocol、IP)規格、ワイヤレスアプリケーションプロトコル(Wireless Application Protocol、WAP)規格、及びInstitute of Electrical and Electronics Engineers(IEEE)規格が挙げられる。

【0079】

記憶デバイスは、データを記憶することができる任意の種類のデバイス又は記憶媒体を含むことができる。記憶媒体は、有形又は非一時的コンピュータ可読媒体を含むことができる。コンピュータ可読媒体としては、光学ディスク、フラッシュメモリ、磁気メモリ、又は任意の他の好適なデジタル記憶媒体を挙げることができる。いくつかの例では、メモリデバイス又はその一部分は不揮発性メモリとして説明されることがあり、他の例では、メモリデバイスの一部分は揮発性メモリとして説明されることがある。揮発性メモリの例としては、ランダムアクセスメモリ(random access memory、RAM)、ダイナミックランダムアクセスメモリ(dynamic random access memory、DRAM)、及びスタティックランダムアクセスメモリ(static random access memory、SRAM)を挙げることができる。不揮発性メモリの例としては、磁気ハードディスク、光学ディスク、フロッピーディスク、フラッシュメモリ、又は電氣的プログラム可能メモリ(electrically programmable memory、EPROM)若しくは電氣的消去可能及びプログラム可能メモリ(electrically erasable and programmable、EEPROM)の形態を挙げることができる。記憶デバイス(単数又は複数)としては、メモリカード(例えば、セキュアデジタル(Secure Digital、SD)メモリカード)、内蔵/外付けハードディスクドライブ、及び/又は内蔵/外付けソリッドステートドライブを挙げることができる。データは、定義されたファイルフォーマットに従って記憶デバイス上に記憶することができる。

【0080】

図4は、システム100の一実装形態に含まれ得る構成要素の一例を示す概念的描画である。図4に示す例示的な実装形態では、システム100は、1つ以上の演算デバイス402A～402N、テレビサービスネットワーク404、テレビサービスプロバイダサイト406、ワイドエリアネットワーク408、ローカルエリアネットワーク410、及び1つ以上のコンテンツプロバイダサイト412A～412Nを含む。図4に示す実装形態は、例えば、映画、ライブスポーツイベントなどのデジタルメディアコンテンツ、並びにデータ及びアプリケーション及びそれらに関連付けられたメディアプレゼンテーションが、演算デバイス402A～402Nなどの複数の演算デバイスに配布され、かつ、それらによってアクセスされることが可能となるように構成され得るシステムの一例を表す。図4に示す例では、演算デバイス402A～402Nは、テレビサービスネットワーク404、ワイドエリアネットワーク408、及び/又はローカルエリアネットワーク410のうちの1つ以上からデータを受信するように構成されている任意のデバイスを含むことができる。例えば、演算デバイス402A～402Nは、有線及び/又は無線通信用に装備してもよく、1つ以上のデータチャネルを通じてサービスを受信するように構成してもよく、いわゆるスマートテレビ、セットトップボックス、及びデジタルビデオレコーダを含むテレビを含んでもよい。更に、演算デバイス402A～402Nは、デスクトップ、ラップトップ又はタブレットコンピュータ、ゲーム機、例えば「スマート」フォン、セルラー電話、及びパーソナルゲーミングデバイスを含むモバイルデバイスを含んでもよい。

【0081】

テレビサービスネットワーク404は、テレビサービスを含み得る、デジタルメディアコンテンツの配信を可能にするように構成されているネットワークの一例である。例えば、テレビサービスネットワーク404は、公共地上波テレビネットワーク、公共又は加入ベースの衛星テレビサービスプロバイダネットワーク、並びに公共又は加入ベースのケーブルテレビプロバイダネットワーク及び/又は頭越し型(over the top)サービスプロバイダ若しくはインターネットサービスプロバイダを含んでもよい。いくつかの実施例では、テレビサービスネットワーク404は、テレビサービスの提供を可能にするために主に使用され得るが、テレビサービスネットワーク404はまた、本明細書に記載された電気

通信プロトコルの任意の組み合わせに基づく他の種類のデータ及びサービスの提供も可能とすることに留意されたい。更に、いくつかの実施例では、テレビサービスネットワーク404は、テレビサービスプロバイダサイト406と、演算デバイス402A~402Nのうちの1つ以上との間の双方向通信を可能にすることができることに留意されたい。テレビサービスネットワーク404は、無線通信メディア及び/又は有線通信メディアの任意の組み合わせを含むことができる。テレビサービスネットワーク404は、同軸ケーブル、光ファイバケーブル、ツイストペアケーブル、無線送信機及び受信機、ルータ、スイッチ、リピータ、基地局、又は様々なデバイスとサイトとの間の通信を容易にするために有用であり得る任意の他の機器を含むことができる。テレビサービスネットワーク404は、1つ以上の電気通信プロトコルの組み合わせに従って動作することができる。電気通信プロトコルは、専用の態様を含むことができ、及び/又は規格化された電気通信プロトコルを含むことができる。規格化された電気通信プロトコルの例としては、DVB規格、ATSC規格、ISDB規格、DTMB規格、DMB規格、ケーブルによるデータサービスインターフェース標準(Data Over Cable Service Interface Specification、DOCSIS)規格、HbbTV規格、W3C規格、及びUPnP規格が挙げられる。

10

【0082】

図4を再び参照すると、テレビサービスプロバイダサイト406は、テレビサービスネットワーク404を介してテレビサービスを配信するように構成することができる。例えば、テレビサービスプロバイダサイト406は、1つ以上の放送局、ケーブルテレビプロバイダ、又は衛星テレビプロバイダ、又はインターネットベースのテレビプロバイダを含み得る。例えば、テレビサービスプロバイダサイト406は、衛星アップリンク/ダウンリンクを介したテレビプログラムを含む送信を、受信するように構成することができる。更に、図4に示すように、テレビサービスプロバイダサイト406は、ワイドエリアネットワーク408と通信することができ、コンテンツプロバイダサイト412A~412Nからデータを受信するように構成することができる。いくつかの実施例では、テレビサービスプロバイダサイト406は、テレビスタジオを含むことができ、コンテンツはそこから発信できることに留意されたい。

20

【0083】

ワイドエリアネットワーク408は、パケットベースのネットワークを含み、1つ以上の電気通信プロトコルの組み合わせに従って動作することができる。電気通信プロトコルは、専用の態様を含むことができ、及び/又は規格化された電気通信プロトコルを含むことができる。規格化された電気通信プロトコルの例としては、汎欧州デジタル移動電話方式(Global System Mobile Communications、GSM)規格、符号分割多元接続(code division multiple access、CDMA)規格、3rd Generation Partnership Project(3GPP)規格、欧州電気通信標準化機構(European Telecommunications Standards Institute、ETSI)規格、欧州規格(EN)、IP規格、ワイヤレスアプリケーションプロトコル(Wireless Application Protocol、WAP)規格、及び例えば、IEEE802規格のうちの1つ以上(例えば、Wi-Fi)などの電気電子技術者協会(Institute of Electrical and Electronics Engineers、IEEE)規格が挙げられる。ワイドエリアネットワーク408は、無線通信メディア及び/又は有線通信メディアの任意の組み合わせを含むことができる。ワイドエリアネットワーク408は、同軸ケーブル、光ファイバケーブル、ツイストペアケーブル、イーサネットケーブル、無線送信部及び受信部、ルータ、スイッチ、リピータ、基地局、又は様々なデバイス及びサイト間の通信を容易にするために有用であり得る任意の他の機器を含むことができる。一実施例では、ワイドエリアネットワーク408はインターネットを含んでもよい。ローカルエリアネットワーク410は、パケットベースのネットワークを含み、1つ以上の電気通信プロトコルの組み合わせに従って動作することができる。ローカルエリアネットワーク410は、アクセス及び/又は物理インフラストラクチャのレベルに基づいてワイドエリアネットワーク408と区別することができる。例えば、ローカルエリアネットワーク410は、セキュアホームネットワークを含んでもよい。

30

40

50

【0084】

図4を再び参照すると、コンテンツプロバイダサイト412A～412Nは、マルチメディアコンテンツをテレビサービスプロバイダサイト406及び/又は演算デバイス402A～402Nに提供することができるサイトの例を表す。例えば、コンテンツプロバイダサイトは、マルチメディアファイル及び/又はストリームをテレビサービスプロバイダサイト406に提供するように構成されている、1つ以上のスタジオコンテンツサーバを有するスタジオを含むことができる。一実施例では、コンテンツプロバイダのサイト412A～412Nは、IPスイートを使用してマルチメディアコンテンツを提供するように構成してもよい。例えば、コンテンツプロバイダサイトは、リアルタイムストリーミングプロトコル(RTSP)、HTTPなどに従って、マルチメディアコンテンツを受信デバイスに提供するように構成されてもよい。更に、コンテンツプロバイダサイト412A～412Nは、ハイパーテキストベースのコンテンツなどを含むデータを、ワイドエリアネットワーク408を通じて、受信デバイス、演算デバイス402A～402N、及び/又はテレビサービスプロバイダサイト406のうちの1つ以上に提供するように構成してもよい。コンテンツプロバイダサイト412A～412Nは、1つ以上のウェブサーバを含んでもよい。データプロバイダサイト412A～412Nによって提供されるデータは、データフォーマットに従って定義することができる。

10

【0085】

図1を再び参照すると、ソースデバイス102は、ビデオソース104と、ビデオエンコーダ106と、データカプセル化装置107と、インターフェース108とを含む。ビデオソース104は、ビデオデータをキャプチャ及び/又は記憶するように構成された任意のデバイスを含むことができる。例えば、ビデオソース104は、ビデオカメラ及びそれに動作可能に結合された記憶デバイスを含むことができる。ビデオエンコーダ106は、ビデオデータを受信し、ビデオデータを表す準拠ビットストリームを生成するように構成された、任意のデバイスを含むことができる。準拠ビットストリームは、ビデオデコーダが受信し、それからビデオデータを再生成することができるビットストリームを指すことがある。準拠ビットストリームの態様は、ビデオ符号化標準に従って定義することができる。準拠ビットストリームを生成するとき、ビデオエンコーダ106は、ビデオデータを圧縮することができる。圧縮は、非可逆的(視聴者に認識可能若しくは認識不可能)又は可逆的とすることができる。図5は、本明細書で説明するビデオデータを符号化するための技術を実施することができる、ビデオエンコーダ500の例を示すブロック図である。例のビデオエンコーダ500は、別個の機能ブロックを有するように示されているが、そのような例示は、説明のためのものであり、ビデオエンコーダ500及び/又はそのサブコンポーネントを特定のハードウェア又はソフトウェアアーキテクチャに限定するものではないことに留意されたい。ビデオエンコーダ500の機能は、ハードウェア、ファームウェア、及び/又はソフトウェアの実装形態の任意の組み合わせを用いて実現することができる。

20

30

【0086】

ビデオエンコーダ500は、ピクチャ領域のイントラ予測符号化及びインター予測符号化を実行する場合があり、そのためにハイブリッドビデオエンコーダと称される場合がある。図5に示す例では、ビデオエンコーダ500は、ソースビデオブロックを受信する。いくつかの例では、ソースビデオブロックは、符号化構造に従って分割されているピクチャの部分を含むことができる。例えば、ソースビデオデータは、マクロブロック、CTU、CB、その再分割、及び/又は別の同等の符号化ユニットを含むことができる。いくつかの例では、ビデオエンコーダ500は、ソースビデオブロックの追加の再分割を実行するように構成することができる。本明細書で説明する技術は、符号化の前及び/又は符号化中にソースビデオデータがどのように分割されるかにかかわらず、ビデオ符号化に一般的に適用可能であることに留意されたい。図5に示す例では、ビデオ符号化装置500は、加算部502と、変換係数発生装置504と、係数量子化部506と、逆量子化及び変換係数処理部508と、加算部510と、イントラ予測処理部512と、インター予測処理部514と、エントロピ符号化部516と、を含む。図5に示すように、ビデオエンコーダ500は、ソース

40

50

ビデオブロックを受信し、ビットストリームを出力する。

【0087】

図5に示す例では、ビデオエンコーダ500は、ソースビデオブロックから予測ビデオブロックを減算することにより、残差データを生成することができる。予測ビデオブロックの選択を、以下に詳細に記載する。加算器502は、この減算演算を実行するように構成された構成要素を表す。一例では、ビデオブロックの減算は、画素領域で行われる。変換係数発生装置504は、離散コサイン変換(DCT)、離散サイン変換(DST)、又は概念的に類似の変換などの変換を残差ブロック又はその再分割に適用し(例えば、4つの 8×8 の変換を残差値の 16×16 のアレイに適用することができる)、残差変換係数のセットを生成する。変換係数発生装置504は、離散三角変換の近似物を含め離散三角変換のファミリーに含まれる変換の任意の及び全ての組み合わせを実行するように構成することができる。変換係数発生装置504は、変換係数を係数量子化部506に出力することができる。係数量子化部506は、変換係数の量子化を実行するように構成することができる。量子化プロセスは、係数の一部又は全てに関連付けられたビット深度を低減することができる。量子化の程度により、符号化したビデオデータのレートディストーション(すなわち、ビデオのビットレート対品質)が変更され得る。量子化の程度は、量子化パラメータ(QP)を調節することにより、修正することができる。量子化パラメータは、スライスレベル値及び/又はCUレベル値(例えば、CUDeltaQP値)に基づいて判定することができる。QPデータは、特定の変換係数のセットを量子化するQPを判定するために使用される任意のデータを含み得る。図5に示すように、量子化された変換係数(レベル値と呼んでもよい)は、逆量子化及び変換係数処理部508に出力される。逆量子化及び変換係数処理部508は、逆量子化及び逆変換を適用して、復元された残差データを生成するように構成することができる。図5に示すように、加算器510において、復元された残差データを、予測ビデオブロックに加算することができる。このようにして、符号化ビデオブロックを復元することができる。結果として得られる復元されたビデオブロックを使用して、所与の予測、変換、及び/又は量子化についての符号化品質を評価することができる。ビデオ符号化装置500は、複数の符号化パスを実行する(例えば、予測、変換パラメータ及び量子化パラメータの1つ以上を変更しながら符号化を実行する)ように構成することができる。ビットストリームのレートディストーション又は他のシステムパラメータは、復元されたビデオブロックの評価に基づいて最適化することができる。更に、復元されたビデオブロックは、その後のブロックを予測するための参照として記憶して使用することができる。

【0088】

図5を参照すると、イントラ予測処理部512は、符号化されるビデオブロックに対してイントラ予測モードを選択するように構成することができる。イントラ予測処理部512は、現在のブロックを符号化するために、フレームを評価し、使用するイントラ予測モードを判定するように構成することができる。上述したように、可能なイントラ予測モードは、平面予測モード、DC予測モード、及び角度予測モードを含んでもよい。更に、いくつかの例では、クロマ成分に対する予測モードは、ルマ予測モードに対する予測モードから推測できることに留意されたい。イントラ予測処理部512は、1つ以上の符号化パスを実行した後にイントラ予測モードを選択してもよい。更に、一実施例では、イントラ予測処理部512は、レートディストーション解析に基づいて予測モードを選択してもよい。図5に示すように、イントラ予測処理部512は、イントラ予測データ(例えば、シンタックス要素)をエン트로ピ符号化部516及び変換係数発生装置504に出力する。上述のように、残差データに対して実行される変換は、モード依存性であってもよい(例えば、二次変換行列は、予測モードに基づいて判定することができる)。

【0089】

図5を再び参照すると、インター予測処理部514は、現在のビデオブロックに対してインター予測符号化を実行するように構成することができる。インター予測処理部514は、ソースビデオブロックを受信し、ビデオブロックのPUに対する動きベクトルを計算するように構成することができる。動きベクトルは、参照フレーム内の予測ブロックに対す

る、現在のビデオフレーム内のビデオブロックのPUの変位を示すことができる。インター予測符号化は、1つ以上の参照ピクチャを使用することができる。更に、動き予測は、単一予測(1つの動きベクトルを用いる)又は双予測(2つの動きベクトルを用いる)とすることができる。インター予測処理部514は、例えば、絶対差分和(sum of absolute difference)(SAD)、差分二乗和(sum of square difference)(SSD)、又は他の差の測定法によって判定された画素差を算出することにより、予測ブロックを選択するように構成することができる。上述したように、動きベクトルは、動きベクトル予測に従って判定及び判定することができる。インター予測処理部514は、上述したように、動きベクトル予測を実行するように構成することができる。インター予測処理部514は、動き予測データを用いて予測ブロックを生成するように構成することができる。例えば、インター予測処理部514は、フレームバッファ(図5に示さない)内に予測ビデオブロックを配置することができる。インター予測処理部514は、復元された残差ブロックに1つ以上の補間フィルタを適用して、動き予測に使用するための整数未満の画素値を算出するように更に構成することができることに留意されたい。インター予測処理部514は、計算された動きベクトルに対する動き予測データをエントロピ符号化部516に出力することができる。

10

【0090】

図5を再び参照すると、エントロピ符号化部518は、量子化された変換係数及び予測シタックスデータ(すなわち、イントラ予測データ、動き予測データ)を受信する。いくつかの例では、係数量子化部506は、係数がエントロピ符号化部518に出力される前に、量子化された変換係数を含む行列の走査を実行することができることに留意されたい。他の例では、エントロピ符号化部518は、走査を実行することができる。エントロピ符号化部518は、本明細書で説明する技術の1つ以上に従ってエントロピ符号化を実行するように構成することができる。このように、ビデオ符号化装置500は、本開示の1つ以上の技術に従って符号化ビデオデータを生成するように構成されているデバイスの一例を表す。一実施例では、ビデオ符号化器500は、動き制約タイルセットを含む符号化ビデオデータを生成することができる。

20

【0091】

図1を再び参照すると、データカプセル化部107は、符号化ビデオデータを受信し、定義されたデータ構造に従って準拠ビットストリーム、例えば、一連のNALユニットなどの準拠ビットストリームを生成することができる。準拠ビットストリームを受信するデバイスは、そこからビデオデータを再生成することができる。更に、上述のように、サブビットストリーム抽出とは、受信ビットストリーム内のデータを破棄及び/又は修正することによって、ITU-T H.265準拠ビットストリームを受信しているデバイスが、新たなITU-T H.265準拠ビットストリームを形成するプロセスを指し得る。適合ビットストリームという用語は、準拠ビットストリームという用語の代わりに使用され得ることに留意されたい。

30

【0092】

上述したように、ITU-T H.265では、タイル構造は、各行及び列が同じ数のタイルを含むという点で制限される。いくつかの場合では、行及び/又は列において様々な数のタイルを有することが有用であり得る。例えば、360°の球状ビデオを符号化するためには、球の赤道よりも極領域でタイルを少なくすることが有用であり得、この場合、これは、行から行までタイル列の数を変化させるのに有用であり得る。一実施例では、データカプセル化装置107は、本明細書に記載される1つ以上の技術に従ってタイル構造をシグナリングするように構成することができる。データカプセル化装置107は、ビデオエンコーダ106と同じ物理デバイス内に配置される必要はないことに留意されたい。例えば、ビデオエンコーダ106及びデータカプセル化部107によって実行されるものとして説明される機能は、図4に示すデバイス間で配布してもよい。

40

【0093】

本明細書に記載される技術によれば、データカプセル化装置107は、タイルセット構造のための以下の種類の情報のうちの1つ以上をシグナリングするように構成され得る。

50

【0094】

タイルセットが有効であるかどうかを示すフラグ。有効でない場合、ピクチャ全体は、1つのタイルセットであると推測される。

【0095】

タイルセットシグナリングが有効化された場合、以下をシグナリングすることができる。

【0096】

タイルセットの行の数は、シグナリングされてもよく、
タイルセットの列の数は、シグナリングされてもよい。

【0097】

各タイルセットにおいて、以下の情報をシグナリングすることができる。

【0098】

タイルセット内のタイル行の数、
タイルセット内のタイル列の数、
タイルがタイルセット内で行方向に沿って均一に離間されている場合にシグナリングするインジケータ/フラグ、及び/又は
タイルがタイルセット内で列方向に沿って均一に離間されている場合にシグナリングするインジケータ/フラグ、及び/又は
間隔が均一である場合、タイルセット毎に以下をシグナリングすることができる。

【0099】

タイルセット内の各タイル行内のCTBの数のタイル幅、
タイルセット内の各タイル列内のCTBの数のタイル高さ、
間隔が均一でない場合、各タイルセット内のタイル毎に以下をシグナリングすることができる。

【0100】

タイルセット内の各タイル内の行のCTBの数、
タイルセット内の各タイル内の列のCTBの数

表3は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表3に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表3に含まれる例示的なシンタックスは、VPS又はSPSに含まれてもよい。

【0101】

10

20

30

40

50

【表 3】

表3

parameter_set_rbsp() {	記述子
...	
tiles_enabled_flag	u(1)
tilesets_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
if(tilesets_enabled_flag) {	
num_tile_set_columns_minus1	ue(v)
num_tile_set_rows_minus1	ue(v)
}	
for(k = 0; k < num_tile_set_rows_minus1; k++)	
for(l = 0; l < num_tile_set_columns_minus1; l++) {	
num_tile_columns_minus1[k][l]	ue(v)
num_tile_rows_minus1[k][l]	ue(v)
uniform_spacing_flag[k][l]	u(1)
if(!uniform_spacing_flag[k][l]) {	
for(i = 0; i <= num_tile_columns_minus1[k][l]; i++)	
column_width_minus1[k][l][i]	
for(i = 0; i <= num_tile_rows_minus1[k][l]; i++)	
row_height_minus1[k][l][i]	
}	
else {	
tile_width_in_ctbsy_minus1[k][l]	ue(v)
tile_height_in_ctbsy_minus1[k][l]	ue(v)
}	
loop_filter_across_tiles_enabled_flag[k][l]	u(1)
}	
}	
}	
...	

10

20

30

40

【0102】

表3に関して、シンタックス要素tiles_enabled_flag、tilesets_enabled_flag、num_tile_columns_minus1[k][l]、num_tile_rows_minus1[k][l]、uniform_spacing_flag[k][l]、column_width_minus1[k][l][i]、row_height_minus1[k][l][i]、tile_width_in_ctbsy_minus1[k][l]、tile_height_in_ctbsy_minus1[k][l]、及びloop_filter_across_tiles_enabled_flag[k][l]は、以下の例示的な定義に基づくことができることに留意されたい。

【0103】

1に等しいtiles_enabled_flagは、パラメータセットを参照して、各ピクチャ内に2

50

つ以上のタイルが存在することを指定する。0に等しいtiles_enabled_flagは、パラメータセットを参照して、各ピクチャ内に1つのタイルのみが存在することを指定する。

【0104】

1に等しいtilesets_enabled_flagは、パラメータセットを参照して、各ピクチャ内に2つ以上のタイルセットが存在することを指定する。0に等しいtilesets_enabled_flagは、パラメータセットを参照して、各ピクチャ内に1つのタイルセットのみが存在することを指定する。

【0105】

CVS内でアクティブ化される全てのパラメータセットについては、tilesets_enabled_flagの値が同じであることが、ビットストリームの適合性の要件である。

10

【0106】

tiles_enabled_flagが0に等しい場合、tilesets_enabled_flagが0に等しいことがビットストリームの適合性の要件である。

【0107】

num_tile_set_columns_minus1プラス1は、ピクチャを区画するタイル列の数を指定する。num_tile_set_columns_minus1は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_set_columns_minus1の値が存在しない場合、num_tile_set_columns_minus1の値は0に等しいと推測される。

【0108】

num_tile_set_rows_minus1プラス1は、ピクチャを区画するタイル行の数を指定する。num_tile_set_rows_minus1は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_set_rows_minus1の値が存在しない場合、num_tile_set_rows_minus1の値は0に等しいと推測される。

20

【0109】

一実施例では、tilesets_enabled_flagが1に等しい場合、num_tile_set_columns_minus1及びnum_tile_set_rows_minus1は、いずれも0に等しくないものとする。

【0110】

num_tile_columns_minus1[k][1]プラス1は、インデックス(k,l)に関連付けられたタイルセットを区画するタイル列の数を指定する。num_tile_columns_minus1[k][1]は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_columns_minus1[k]

30

[1]の値が存在しない場合、num_tile_columns_minus1[k][1]の値は0に等しいと推測される。

【0111】

別の例では、num_tile_columns_minus1[k][1]は、両端値を含む0～PicWidthInCtbsY-num_tile_set_columns_minus1-1の範囲にあるものとする。

【0112】

num_tile_rows_minus1[k][1]プラス1は、インデックス(k,l)に関連付けられたタイルセットを区画するタイル行の数を指定する。num_tile_rows_minus1[k][1]は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_rows_minus1[k][1]の値が存在しない場合、num_tile_rows_minus1[k][1]の値は0に等しいと推測される。

40

【0113】

別の例では、num_tile_rows_minus1[k][1]は、両端値を含む、0～PicHeightInCtbsY-num_tile_set_rows_minus1-1の範囲にあるものとする。

【0114】

1に等しいuniform_spacing_flag[k][1]は、タイル列境界及び同様にタイルの行境界が、インデックス(k,l)に関連付けられたタイルセットにわたって均一に分布されることを指定する。0に等しいuniform_spacing_flag[k][1]は、タイル列境界及び同様にタイルの行境界が、インデックス(k,l)に関連付けられたタイルセットにわたって均一に

50

分布されないが、シンタックス要素 `column_width_minus1[k][1][i]` 及び `row_height_minus1[k][1][i]`

`i` を使用して明示的にシグナリングされることを指定する。`uniform_spacing_flag[k][1]` の値が存在しない場合、`uniform_spacing_flag[k][1]` の値は1に等しいと推測される。

【0 1 1 5】

`column_width_minus1[i]` プラス1は、インデックス (k, l) に関連付けられたタイルセットにおける符号化ツリーブロックの単位における i 番目のタイル列の幅を指定する。

【0 1 1 6】

`row_height_minus1[k][1][i]` プラス1は、インデックス (k, l) に関連付けられたタイルセット内の符号化ツリーブロックの単位における i 番目のタイル行の高さを指定する。

【0 1 1 7】

`tile_width_in_ctbsy_minus1[k][1]` プラス1は、符号化ツリーブロックの単位におけるインデックス (k, l) に関連付けられたタイルセット内の各タイル列の幅を指定する。

【0 1 1 8】

一実施例では、 $0 \sim \text{num_tile_set_rows_minus1}$ の範囲の各 k について、 $0 \sim \text{num_tile_set_columns_minus1}$ の範囲の l の `tile_width_in_ctbsy_minus1[k][1]` の合計の値が同じ値であることがビットストリームの適合性の要件である。

【0 1 1 9】

`tile_height_in_ctbsy_minus1[k][1]` プラス1は、符号化ツリーブロックの単位におけるインデックス (k, l) に関連付けられたタイルセット内の各タイル行の高さを指定する。

【0 1 2 0】

一実施例では、 $0 \sim \text{num_tile_set_columns_minus1}$ の範囲の各 l について、 $0 \sim \text{num_tile_set_rows_minus1}$ の範囲の k の `tile_height_in_ctbsy_minus1[k][1]` の合計の値が同じ値であることがビットストリームの適合性の要件である。

【0 1 2 1】

1に等しい `loop_filter_across_tiles_enabled_flag[k][l]` は、インループフィルタリング操作が、PPSを参照して、インデックス (k, l) に関連付けられたタイルセット内のタイル境界を越えて実行され得ることを指定する。0に等しい `loop_filter_across_tiles_enabled_flag` は、インループフィルタリング操作が、PPSを参照して、インデックス (k, l) に関連付けられたタイルセット内のタイル境界を越えて実行されないことを指定する。インループフィルタリング操作は、デブロッキングフィルタ及びサンプル適応オフセットフィルタ操作を含む。`loop_filter_across_tiles_enabled_flag[k][1]` の値が存在しない場合、`loop_filter_across_tiles_enabled_flag[k][1]` の値は1に等しいと推測される。

【0 1 2 2】

別の例では、タイルセット行当たりの数タイルセット列は、タイルセット行毎に異なることが可能であり得る。表4は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表4に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表4に含まれる例示的なシンタックスは、VPS又はSPSに含まれてもよい。

【0 1 2 3】

10

20

30

40

50

【表 4】

表4

parameter_set_rbsp() {	記述子
...	
tiles_enabled_flag	u(1)
tilesets_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
if(tilesets_enabled_flag) {	
num_tile_set_rows_minus1	ue(v)
}	
for(k=0; k< num_tile_set_rows_minus1 ; k++) {	
num_tile_set_columns_minus1[k]	ue(v)
for(l=0; l< num_tile_set_columns_minus1[k] ; l++) {	
num_tile_columns_minus1[k][l]	ue(v)
num_tile_rows_minus1[k][l]	ue(v)
uniform_spacing_flag[k][l]	u(1)
if(! uniform_spacing_flag[k][l]) {	
for(i=0; i<= num_tile_columns_minus1[k][l] ; i++)	
column_width_minus1[k][l][i]	ue(v)
for(i=0; i<= num_tile_rows_minus1[k][l] ; i++)	
row_height_minus1[k][l][i]	ue(v)
}	
else {	
tile_width_in_ctbsy_minus1[k][l]	ue(v)
tile_height_in_ctbsy_minus1[k][l]	ue(v)
}	
loop_filter_across_tiles_enabled_flag[k][l]	u(1)
}	
}	

【0124】

表4に関して、シンタックス要素**tiles_enabled_flag**、**tilesets_enabled_flag**、**num_tile_rows_minus1**、**uniform_spacing_flag[k][l]**、**column_width_minus1[k][l][i]**、**row_height_minus1[k][l][i]**、**tile_width_in_ctbsy_minus1[k][l]**、**tile_height_in_ctbsy_minus1[k][l]**、及び**loop_filter_across_tiles_enabled_flag[k][l]**は、表3に関して上記に提供された定義に基づくことができることに留意されたい。**num_tile_set_columns_minus1[k]**は、以下の例示的な定義に基づくことができる。

【0125】

num_tile_set_columns_minus1[k]プラス1は、タイルセット行kにおけるタイルセット列の数を指定する。**num_tile_set_columns_minus1[k]**は、両端値を含む、0

～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_set_columns_minus1[k]の値が存在しない場合、num_tile_set_columns_minus1[k]の値は、両端値を含む、0～num_tile_set_rows_minus1の範囲のkについて、0に等しいと推測される。

【0126】

別の例では、タイルセット列当たりの数タイルセット行は、タイルセット行毎に異なることが可能であり得る。表5は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表5に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表5に含まれる例示的なシンタックスは、VPS又はSPSに含まれてもよい。

【0127】

【表5】

表5

parameter_set_rbsp() {	記述子
...	
tiles_enabled_flag	u(1)
tilesets_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
if(tilesets_enabled_flag) {	
num_tile_set_columns_minus1	ue(v)
}	
for(l = 0; l < num_tile_set_columns_minus1; l++) {	
num_tile_set_rows_minus1[l]	ue(v)
for(k = 0; k < num_tile_set_rows_minus1[l]; k++) {	
num_tile_columns_minus1[k][l]	ue(v)
num_tile_rows_minus1[k][l]	ue(v)
uniform_spacing_flag[k][l]	u(1)
if(!uniform_spacing_flag[k][l]) {	
for(i = 0; i <= num_tile_columns_minus1[k][l]; i++)	
column_width_minus1[k][l][i]	ue(v)
for(i = 0; i <= num_tile_rows_minus1[k][l]; i++)	
row_height_minus1[k][l][i]	ue(v)
}	
else {	
tile_width_in_ctbsy_minus1[k][l]	ue(v)
tile_height_in_ctbsy_minus1[k][l]	ue(v)
}	
loop_filter_across_tiles_enabled_flag[k][l]	u(1)
}	
}	
}	

【0128】

表5に関して、シンタックス要素`tiles_enabled_flag`、`tilesets_enabled_flag`、`num_tile_columns_minus1`、`uniform_spacing_flag[k][l]`、`column_width_minus1[k][l][i]`、`row_height_minus1[k][l][i]`、`tile_width_in_ctbsy_minus1[k][l]`、`tile_height_in_ctbsy_minus1[k][l]`、及び`loop_filter_across_tiles_enabled_flag[k][l]`は、表3に関して上記に提供された定義に基づくことができることに留意されたい。`num_tile_set_rows_minus1`

[k]は、以下の例示的な定義に基づくことができる。

【0129】

`num_tile_set_rows_minus1[l]`プラス1は、タイルセット列1におけるタイルセット行の数を指定する。`num_tile_set_rows_minus1[l]`は、両端値を含む、0～`PicHeightInCtbsY-1`の範囲にあるものとする。`num_tile_set_rows_minus1[k]`の値が存在しない場合、`num_tile_set_rows_minus1[1]`の値は、両端値を含む、0～`num_tile_set_columns_minus1`の範囲のlについて、0に等しいと推測される。

【0130】

この例に関して、別の例では、シンタックス要素`num_tile_columns_minus1[k][l]`、`num_tile_rows_minus1[k][l]`、`uniform_spacing_flag[k][l]`、`tile_width_in_ctbsy_minus1[k][l]`、`tile_height_in_ctbsy_minus1[k][l]`、及び`loop_filter_across_tiles_enabled_flag[k][l]`についてのアレイインデックス[k][l]は、順序[l][k]のインデックスとして代わりにシグナリングされてもよく、そのため、シンタックス要素`num_tile_columns_minus1[l][k]`、`num_tile_rows_minus1[l][k]`、`uniform_spacing_flag[l][k]`、`tile_width_in_ctbsy_minus1[l][k]`、`tile_height_in_ctbsy_minus1[l][k]`、`loop_filter_across_tiles_enabled_flag[l][k]`としてシグナリングされてもよい。

【0131】

更に、この例に関して、別の例では、要素`column_width_minus1[k][l][i]`、`row_height_minus1[k][l][i]`についてのアレイインデックス[k][l][i]は、順序[l][k][i]のインデックスとして代わりにシグナリングされてもよく、そのため、シンタックス要素`column_width_minus1[l][k][i]`、`row_height_minus1[l][k][i]`としてシグナリングされてもよい。

【0132】

一実施例では、本明細書の技術によれば、タイルのラスタ順はタイルセット内で行ごとであってもよく、タイルセットは画像内でラスタ順序になる。これは、連続するタイルセット内の符号化データを形成し、タイルセットのビットストリームのスプライシング及びそれらのビットストリーム部分の並列復号を助けることができることに留意されたい。本明細書においてスプライシングという用語は、抽出された部分が1つ以上のタイルセットに対応し得る全体的なビットストリームの部分のみの抽出を指し得る。対照的に、ITU-T H.265では、タイルのラスタ順序は画像内で行ごとである。図6は、タイルのラスタ順はタイルセット内で行ごとであり、タイルセットは画像内でラスタ順序になるタイルラスタ走査を示す。図6では、各タイルに示される下付き文字数は、タイルがスキャナである順序を提供する。

【0133】

一実施例では、本明細書の技術によれば、符号化されたツリーブロック(CTB/CTU)のラスタ順序は、タイルセット内のタイルラスタ走査で行ごとであり得、タイルセットは、ピクチャ内でラスタ順序になる。これは、連続するタイルセット内の符号化データを形成し、タイルセットのビットストリームのスプライシング及び並列復号を助けることができることに留意されたい。本明細書においてスプライシングという用語は、抽出された部分が1つ以上のタイルセットに対応し得る全体的なビットストリームの部分のみの抽出を指し得る。対照的に、ITU-T H.265では、符号化されたツリーブロック(CTB/CTU)のラスタ順序は、ピクチャ内のタイルラスタ走査で行ごとである。図7は、符号化されたツリーブロック(CTB/CTU)のラスタ順序は、タイルセット内のタイルラスタ走査で行ごとで

あり得、タイルセットは、ピクチャ内でラスタ順序になることを示す。図7では、各CTUに示される数は、CTUがスキャンされる順序を提供する。図7の例は、図6に示すものと同じタイル構造を含むことに留意されたい。

【0134】

符号化されたツリーブロック(CTB/CTU)、タイル、及びタイルセットのラスタ順序は、以下の説明に基づいてもよい。

【0135】

このサブ項目は、VCL NALユニットの順序及び符号化されたピクチャへの関連を指定する。各VCL NALユニットは、符号化されたピクチャの一部である。

【0136】

符号化されたピクチャ内のVCL NALユニットの順序は、以下のように制約される。

【0137】

-符号化されたピクチャの第1のVCL NALユニットは、1に等しいfirst_slice_segment_in_pic_flagを有するものとする。

【0138】

-sliceSegAddrA及びsliceSegAddrBは、同じ符号化ピクチャ内の任意の2つの符号化スライスセグメントNALユニットA及びBのslice_segment_address値である。以下の条件のいずれかが真である場合、符号化スライスセグメントNALユニットAは、符号化スライスセグメントNALユニットBに先行するものとする。

【0139】

-TileId[CtbAddrRsToTs[sliceSegAddrA]]は、TileId[CtbAddrRsToTs[sliceSegAddrB]]未満である。

【0140】

-TileId[CtbAddrRsToTs[sliceSegAddrA]]は、TileId[CtbAddrRsToTs[sliceSegAddrB]]に等しく、CtbAddrRsToTs[sliceSegAddrA]は、CtbAddrRsToTs[sliceSegAddrB]未満である。

【0141】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのi番目のタイル列の幅を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_columns_minus1の範囲のiについてのリストcolWidtht[k][l][i]は、以下のように導出される。

【0142】

【表6】

```
for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        if( uniform_spacing_flag[ k ][ l ] )
            for(i=0; i <= num_tile_columns_minus1[ k ][ l ]; i++)
                colWidth[ k ][ l ][ i ] = tile_width_in_ctbsy[ k ][ l ] -
        else {
            for(i=0; i <= num_tile_columns_minus1[ k ][ l ]; i++) {
                colWidth[ k ][ l ][ i ] = column_width_minus1[ k ][ l ][ i ] + 1
            }
        }
    }
}
```

10

20

30

40

50

【0143】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのj番目のタイル行の高さを指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_rows_minus1の範囲のjについてのリストrowHeight[k][l][j]は、以下のよう
に導出される。

【0144】

【表7】

```
for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        if( uniform_spacing_flag[ k ][ l ])
            for(j=0;j <= num_tile_rows_minus1[ k ][ l ];j++)
                rowHeight[ k ][ l ][ j ]=tile_height_in_ctbsy[ k ][ l ]
        else {
            for(j=0;j <= num_tile_rows_minus1[ k ][ l ];j++) {
                rowHeight[ k ][ l ][ j ]=row_height_minus1[ k ][ l ][ j ]+1
            }
        }
    }
}
```

10

20

【0145】

タイルセットの数を示す変数NumTileSetsは、以下のように導出される。

【0146】

【数3】

$$\text{NumTileSets} = (\text{num_tile_set_rows_minus1} + 1) * (\text{num_tile_set_columns_minus1} + 1)$$

30

【0147】

ルマCTBにおけるピクチャ幅、ルマCTBにおけるピクチャ高さ、及びインデックス(k,l)に関連付けられたタイルセットのルマCTBにおけるピクチャサイズをそれぞれ指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについてのアレイpwctbsy[k][l]、phctbsy[k][l]、psizectbsy[k][l]、並びにインデックス(k,l)に関連付けられたタイルセットまでのCTBの累積カウントを指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについてのアレイctbAddrRSOffset[k][l]は、以下のように導出される。

【0148】

40

【表 8】

```

ctbAddrRSOffsetCalc=0; picSizeInCtbsY=0;
for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        for(pwctbsy[k][l]=0, i=0; i<=num_tile_columns_minus1[k][l]; i++){
            pwctbsy[k][l] += (colWidth[k][l][i])
        }
        for(phctbsy[k][l]=0, i=0; i<=num_tile_rows_minus1[k][l]; i++){
            phctbsy[k][l] += (rowHeight[k][l][i]);
        }
        psizectbsy[k][l]=pwctbsy[k][l]*phctbsy[k][l]
        picSizeInCtbsY+= psizectbsy[k][l]
        ctbAddrRSOffset[k][l]=ctbAddrRSOffsetCalc
        ctbAddrRSOffsetCalc+=psizectbsy[k][l]
    }
}

```

10

【0149】

j番目のタイルセットは、以下のようにインデックス(k,l)と関連付けられる。j番目のタイルセットは、インデックスjを有するタイルセットと呼ばれることがある。

20

【0150】

タイルセットインデックスjを考慮すると、ピクチャのタイルセット列の数k及びlは、以下のように導出される。

【0151】

【数 4】

$$k = j / (\text{num_tile_set_columns_minus1} + 1)$$

$$l = j \% (\text{num_tile_set_columns_minus1} + 1)$$

【0152】

インデックスk及びl並びにピクチャのタイルセット列の数を考慮すると、タイルセットインデックスjは、以下のように導出される。

30

【0153】

【数 5】

$$j = k * ((\text{num_tile_set_columns_minus1} + 1) + l)$$

【0154】

符号化ツリーブロックの単位で、インデックス(k,l)に関連付けられたタイルセットのi番目のタイル列境界の場所を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_columns_minus1+1の範囲のiについてのリストcolBd[k][l][i]は、以下のように導出される。

40

【0155】

【表 9】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        for(colBd[k][l][0]=0, i=0; i<=num_tile_columns_minus1[k][l]; i++){
            colBd[k][l][i+1]=colBd[k][l][i]+colWidth[k][l][i]
        }
    }
}

```

50

【0156】

符号化ツリーブロックの単位で、インデックス(k,l)に関連付けられたタイルセットのj番目のタイル行境界の場所を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_rows_minus1+1の範囲のjについてのリストrowBd[j]は、以下のように導出される。

【0157】

【表10】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        for( rowBd[k][l][0]=0,j=0;j <= num_tile_rows_minus1[k][l];j++)
            rowBd[k][l][j+1]=rowBd[k][l][j]+rowHeight[k][l][j]
    }
}

```

10

【0158】

ピクチャのCTBラスタ走査におけるCTBアドレスからタイルセット及びタイル走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrRsであるリストCtbAddrRsToTs[ctbAddrRs]は、以下のように導出される。

20

【0159】

【表11】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        for( ctbAddrRs = 0; ctbAddrRs < psizectbsy[k][l]; ctbAddrRs++) {
            tbX = ctbAddrRs % pwctbsy[k][l]
            tbY = ctbAddrRs / pwctbsy[k][l]
            for(i=0;i <= num_tile_columns_minus1[k][l];i++)
                if( tbX >= colBd[k][l][i])
                    tileX=i
            for(j=0;j <= num_tile_rows_minus1[k][l];j++)
                if( tbY >= rowBd[k][l][j])
                    tileY=j
            CtbAddrRsToTs[ ctbAddrRs + ctbAddrRSOffset[k][l] ] = 0
            for( i = 0; i < tileX; i++)
                CtbAddrRsToTs[ ctbAddrRs + ctbAddrRSOffset[ k ][ l ] ] +=
rowHeight[k][l][tileY]*colWidth[i]
            for(j=0;j<tileY;j++)
                CtbAddrRsToTs[ ctbAddrRs + ctbAddrRSOffset[ k ][ l ] ] += pwctbsy[k]
[ l ] * rowHeight[k][l][j]
            CtbAddrRsToTs[ ctbAddrRs + ctbAddrRSOffset[ k ][ l ] ] += (tbY - rowBd[k][l]
[ tileY ]) * colWidth[ k ][ l ][ tileX ] + tbX - colBd[ k ][ l ][ tileX ]
            CtbAddrRsToTs[ ctbAddrRs + ctbAddrRSOffset[ k ][ l ] ] += ctbAddrRSOffset[ k ][ l ]
        }
    }
}

```

30

40

【0160】

タイル走査におけるCTBアドレスからピクチャのCTBラスタ走査におけるCTBアドレ

50

スへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrTsであるリストCtbAddrTsToRs[ctbAddrTs]は、以下のように導出される。

【0161】

【表12】

```
for( ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++)
    CtbAddrTsToRs[ CtbAddrRsToTs[ ctbAddrRs ] ] = ctbAddrRs
```

【0162】

タイル走査におけるCTBアドレスからタイルIDへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrTsであるリストTileId[ctbAddrTs]は、以下のように導出される。

10

【0163】

【表13】

```
for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        for(j = 0, tileIdx = 0; j <= num_tile_rows_minus1[ k ][ 1 ]; j++)
            for(i = 0; i <= num_tile_columns_minus1[ k ][ 1 ]; i++, tileIdx++)
                for(y = rowBd[ k ][ 1 ][ j ]; y < rowBd[ k ][ 1 ][ j + 1 ]; y++)
                    for(x = colBd[ k ][ 1 ][ i ]; x < colBd[ k ][ 1 ][ i + 1 ]; x++)
                        TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x +
ctbAddrRsoffset[ k ][ 1 ] ] ] = tileIdx
    }
}
```

20

【0164】

本明細書の技術による一実施例では、データカプセル化装置107は、タイルセット内の各タイルセット及びタイルが独立して処理され得るように情報をシグナリングするように構成され得る。一実施例では、各タイルセットのバイト範囲情報はシグナリングされる。一実施例では、これはタイルセットエントリポイントオフセットのリストとしてシグナリングされてもよい。図8は、それぞれ、T[j]及びT[j+1]のそれぞれが、それぞれ、j番目及び(j+1)番目のタイルセットに属するバイトのサイズをそれぞれ示す実施例を示す。加えて、タイルセット内のタイル毎に、各タイルセットバイト範囲情報がシグナリングされる。一実施例では、これはタイルエントリポイントオフセットのリストとしてシグナリングされてもよい。図8では、b[j][k]及びb[j+1][k]は、それぞれ、j番目のタイルセットのk番目のタイルサブセット及び(j+1)番目のタイルセットのk番目のタイルサブセットに属するバイトのサイズをそれぞれ示す。表6は、タイルセット内のタイルセット及びタイルのそれぞれが、本明細書の技術に従って独立して処理され得るように、情報をシグナリングするために使用され得る、例示的なスライスセグメントヘッダを示す。

30

【0165】

40

【表 1 4】

表6

slice_segment_header() {	記述子
...	
if(tiles_enabled_flag entropy_coding_sync_enabled_flag) {	
if(tilesets_enabled_flag) {	
tileset_offset_len_minus1	ue(v)
for(j = 0; j < NumTileSets-1; j++)	
tileset_entry_point_offset_minus1[j]	u(v)
}	
for(j = 0; j < NumTileSets; j++){	
num_entry_point_offsets[j]	ue(v)
if(num_entry_point_offsets[j] > 0) {	
offset_len_minus1[j]	ue(v)
for(i = 0; i < num_entry_point_offsets[j]; i++)	
entry_point_offset_minus1[j][i]	u(v)
}	
}	
}	
...	
}	

10

20

30

【0166】

表6に関して、シンタックス要素tileset_offset_len_minus1、tileset_entry_point_offset_minus1、num_entry_point_offsets、offset_len_minus1、及びentry_point_offset_minus1は、以下の定義に基づくことができることに留意されたい。

【0167】

tileset_offset_len_minus1プラス1は、tileset_entry_point_offset_minus1[i]シンタックス要素の長さ(ビット)を指定する。tileset_offset_len_minus1の値は、両端値を含む、0～31の範囲にあるものとする。

【0168】

tileset_entry_point_offset_minus1[i]プラス1は、i番目のタイルセットについてi番目のエントリポイントオフセット(バイト)を指定し、tileset_offset_len_minus1プラス1ビットで表される。スライスセグメントデータは、スライスセグメントのヘッダに続き、NumTileSetsサブセットからなり、タイルセットインデックス値は、両端値を含む、0～NumTileSets-1の範囲である。スライスセグメントデータの第1のバイトは、バイト0と見なされる。存在する場合、符号化スライスセグメントNALユニットのス

40

50

ライスセグメントデータ部分に現れるエミュレーション防止バイトは、サブセット識別の目的でライスセグメントデータの一部としてカウントされる。タイルセット0は、符号化ライスセグメントデータのバイト0からtileset_entry_point_offset_minus1[0](両端値を含む)からなり、kが1～NumTileSets-1(両端値を含む)の範囲のkを有するタイルセットkは、符号化ライスセグメントデータのバイトfirstTileSetByte[k]～lastTileSetByte[k](両端値を含む)からなり、firstTileSetByte[k]及びlastTileSetByte[k]は、以下のとおり定義される。

【0169】

【表15】

$$\text{firstTileSetByte}[k] = \sum_{n=1}^k \text{tileset_entry_point_offset_minus1}[n-1] + 1$$

10

$$\text{lastTileSetByte}[k] = \text{firstTileSetByte}[k] + \text{tileset_entry_point_offset_minus1}[k]$$

【0170】

最後のサブセット(サブセットインデックスがNumTileSetsに等しい)は、符号化ライスセグメントデータの残りのバイト、すなわち、lastTileSetByte[NumTileSets-1]+1からライスセグメントデータの末尾までのバイトからなる。

【0171】

各サブセットは、同じタイルセット内にあるライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなるものとする。

20

num_entry_point_offsets[j]は、ライスヘッダ内のj番目のタイルセットのentry_point_offset_minus1[j][i]シンタックス要素の数を指定する。num_entry_point_offsets[j]の値が存在しない場合、num_entry_point_offsets[j]の値は0に等しいと推測される。

【0172】

変数k及びlは、以下のように導出される。

【0173】

【数6】

$$k = j / (\text{num_tile_set_columns_minus1} + 1)$$

30

$$l = j \% (\text{num_tile_set_columns_minus1} + 1)$$

【0174】

num_entry_point_offset[j]の値は、以下のように制約される。

【0175】

-tiles_enabled_flagが0に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、num_entry_point_offsets[j]の値は、両端値を含む、0～PicHeightInCtbsY-1の範囲であるものとする。

【0176】

-そうでなければ、tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが0に等しい場合、num_entry_point_offsets[j]の値は、両端値を含む、0～(num_tile_columns_minus1[k][l]+1)*(num_tile_rows_minus1[k][l]+1)-1の範囲であるものとする。

40

【0177】

-そうでなければ、tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、num_entry_point_offsetsの値は、両端値を含む、0～(num_tile_columns_minus1[k][l]+1)*phctbsy[k][l]-1の範囲であるものとする。

【0178】

offset_len_minus1[j]プラス1は、entry_point_offset_minus1[j][i]シンタッ

50

クス要素の長さ(ビット)を指定する。offset_len_minus1[j]の値は、両端値を含む、0～31の範囲にあるものとする。

【0179】

entry_point_offset_minus1[i]プラス1は、j番目のタイルセットにおけるi番目のエントリポイントオフセット(バイト)を指定し、offset_len_minus1[j]1プラス1ビットで表される。j番目のタイルセットに対応するスライスセグメントデータ内のデータは、tileset_entry_point_offset_minus1[j]の位置に従い、num_entry_point_offsets[j]+1のサブセットからなり、j番目のタイルセットのサブセットインデックス値は、両端値を含む、0～num_entry_point_offsets[j]の範囲である。スライスセグメントデータの第1のバイトは、バイト0と見なされる。存在する場合、符号化スライスセグメントNALユニットのスライスセグメントデータ部分に現れるエミュレーション防止バイトは、サブセット識別の目的でスライスセグメントデータの一部としてカウントされる。j番目のタイルセットのサブセット0は、符号化スライスセグメントデータのバイトtileset_entry_point_offset_minus1[j]+0からtileset_entry_point_offset_minus1[j]+entry_point_offset_minus1[0](両端値を含む)からなり、kが1～num_entry_point_offsets[j]-1(両端値を含む)の範囲のkを有するサブセットkは、符号化スライスセグメントデータのバイトfirstByte[j][k]～lastByte[j][k](両端値を含む)からなり、firstByte[j][k]及びlastByte[j][k]は、以下のとおり定義される。

10

【0180】

【数7】

20

$$\text{firstByte}[j][k] = \text{tileset_entry_point_offset}[j] + \sum_{n=1}^k \text{entry_point_offset_minus1}[j][n-1] + 1$$

$$\text{lastByte}[j][k] = \text{firstByte}[j][k] + \text{entry_point_offset_minus1}[j][k]$$

【0181】

最後のサブセット(サブセットインデックスが[j][num_entry_point_offsets[j]]に等しい)は、バイトlastByte[j][num_entry_point_offsets[j]-1]+1～tileset_entry_point_offset_minus1[j+1]-1からなり、0～NumTileSets-1の範囲のjの符号化スライスセグメントデータの両端値を含む。

30

【0182】

最後のタイルセットの最後のサブセット(サブセットインデックスが[NumTileSets-1][num_entry_point_offsets[NumTileSets]]に等しい)は、符号化スライスセグメントデータのバイト、すなわち、lastByte[NumTileSets-1][num_entry_point_offsets[j]-1]+1からスライスセグメントデータの末尾までのバイトからなる。

【0183】

tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが0に等しい場合、各サブセットは、同じタイル内にあるスライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなり、サブセットの数(すなわち、num_entry_point_offsets[j]+1の値)は、符号化スライスセグメント内のj番目のタイルセットにある符号化ツリーユニットを含むタイルの数に等しいものとする。

40

【0184】

注記:tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが0に等しい場合、各スライス、1つのタイルの符号化ツリーユニットのサブセット(この場合、シンタックス要素entry_point_offset_minus1[j][i]は存在しない)を含む必要があるか、又は完全なタイルの整数の全ての符号化ツリーユニットを含む必要がある。

【0185】

tiles_enabled_flagが0に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、kが1～num_entry_point_offsets[j](両端値を含む)の範囲のkを有する各サブセットkは、ピクチャの同じルマ符号化ツリーブロック行内にあるルマ符号化ツリ

50

ブロックを含むスライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなり、サブセットの数(すなわち、`num_entry_point_offsets[j]+1`の値)は、符号化スライスセグメント内にある符号化ツリーユニットを含むピクチャの符号化ツリーブロック行の数に等しいものとする。

【 0 1 8 6 】

注記:最後のサブセット(すなわち、`num entry point offsets[j]`に等しい`k`のサブセット`k`)は、ピクチャのルマ符号化ツリーブロック行にあるルマ符号化ツリーブロックを含む全ての符号化ツリーユニットを含んでも含まなくてもよい。

【 0 1 8 7 】

tiles_enabled_flagが1に等しく、entropy_coding_sync_enabled_flagが1に等しい場合、kが0～num_entry_point_offsets[j](両端値を含む)の範囲のkを有するj番目のタイルセットの各サブセットkは、j番目のタイルセット内のタイルの同じルマ符号化ツリーブロック行内にあるルマ符号化ツリーブロックを含むスライスセグメント内の全ての符号化ツリーユニットの全ての符号化ビットからなり、サブセットの数(すなわち、num_entry_point_offsets[j]+1の値)は、符号化スライスセグメント内にあるj番目のタイルセット内の符号化ツリーユニットを含むタイルのルマ符号化ツリーブロック行の数に等しいものとする。

【 0 1 8 8 】

一実施例では、各タイルセット内のタイルのバイト範囲シグナリング(タイルオフセットシグナリング)の固定長符号化に使用されるオフセット長さ情報は、1回のみシグナリングされ得、全てのタイルセットに適用される。この例のシンタックスを表7に示す。

【 0 1 8 9 】

【表 16】

表7

slice_segment_header() {	記述子
...	
}	
if(tiles_enabled_flag entropy_coding_sync_enabled_flag) {	
if(tilesets_enabled_flag) {	
tileset_offset_len_minus1	ue(v)
for(j = 0; j < NumTileSets-1; j++)	
tileset_entry_point_offset_minus1[j]	u(v)
}	
all_tile_offset_len_minus1	ue(v)
for(j = 0; j < NumTileSets; j++) {	
num_entry_point_offsets[j]	ue(v)
if(num_entry_point_offsets[j] > 0) {	
for(i = 0; i < num_entry_point_offsets[j]; i++)	
entry_point_offset_minus1[j][i]	u(v)
}	
}	
}	
...	

【0190】

表7に関して、シンタックス要素tileset_offset_len_minus1、tileset_entry_point_offset_minus1、num_entry_point_offsets、及びoffset_len_minus1は、表6に関して上記に提供された定義に基づくことができ、all_tile_offset_len_minus1及びentry_point_offset_minus1は、以下の定義に基づくことができることに留意されたい。

【0191】

all_tile_offset_len_minus1プラス1は、両端値を含む、0～NumTileSets-1の範囲のjの各値についてentry_point_offset_minus1[j][i]シンタックス要素の長さ(ビット)を指定する。all_tile_offset_len_minus1の値は、両端値を含む、0～31の範囲にあるものとする。

【0192】

entry_point_offset_minus1[j][i]プラス1は、j番目のタイルセットについてi番目のエントリポイントオフセット(バイト)を指定し、all_tile_offset_len_minus1プラス1ビットで表される。

【0193】

一実施例では、タイルバイト範囲情報は、ピクチャ内の全てのタイルについてループに関して単一にシグナリングされてもよい。この場合、単一のシンタックス要素は、シグナリングされるタイルのバイト範囲の数に対してシグナリングされてもよい。次に、他のシグナリングされたシンタックス要素を使用して、これらのタイルのバイト範囲要素の数が各タイルセットに属するかを決定することができる。

【0194】

一実施例では、ピクチャ内のタイルセット列の数(unm_tile_set_columns_minus1)及びルセット行の数(num_tile_set_rows_minus1)のシンタックス要素は、ue(v)符号化の代わりにu(v)符号化を使用して符号化された固定長であってもよい。一実施例では、この場合、追加のシンタックス要素は、これらの要素の固定長符号化に使用されるビットの長さを示すようにシグナリングされてもよい。別の例では、これらのシンタックス要素の符号化に使用されるビットの長さは、シグナリングされなくてもよく、代わりに以下に等しくなるように推測されてもよい。

【0195】

Ceil(Log2(PicSizeInCtbsY))ビット、ここで、PicSizeInCtbsYは、ピクチャ内のCTBの数を示す。

【0196】

一実施例では、ピクチャ内のインデックス(k,l)に関連付けられたタイルセットのCTBの数のタイル幅(tile_width_in_ctbsy_minus1[k][l])及びインデックス(k,l)に関連付けられたタイルセットのCTBの数のタイル高さ(tile_height_in_ctbsy_minus1[k][l])のシンタックス要素は、ue(v)符号化の代わりにu(v)符号化を使用して符号化された固定長であってもよい。一実施例では、この場合、追加のシンタックス要素は、これらの要素の固定長符号化に使用されるビットの長さを示すようにシグナリングされてもよい。別の例では、これらのシンタックス要素の符号化に使用されるビットの長さは、シグナリングされなくてもよく、代わりに以下に等しくなるように推測されてもよい。

【0197】

Ceil(Log2(PicSizeInCtbsY))ビット、ここで、PicSizeInCtbsYは、ピクチャ内のCTBの数を示す。

【0198】

一実施例では、CTBにおける列幅及び/又はCTBにおける行高さのシンタックス要素は、最後のタイルセットのピクチャ内の最後のタイルセット列(num_tile_columns_minus1[num_tile_set_rows_minus1][num_tile_set_columns_minus1])及び/又は最後のタイルセット行(num_tile_rows_minus1[num_tile_set_rows_minus1][num_tile_set_columns_minus1])についてシグナリングされない場合がある。この場

10

20

30

40

50

合、CTBのピクチャ高さ及び/又はCTBのピクチャ幅からそれらの値を推測することができる。

【0199】

表8は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表8に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表8に含まれる例示的なシンタックスは、VPS又はSPSに含まれてもよい。

【0200】

【表17】

表8

parameter_set_rbsp() {	記述子
...	
tiles_enabled_flag	u(1)
entropy_coding_sync_enabled_flag	u(1)
if(tiles_enabled_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
tilesets_enabled_flag	u(1)
if(tilesets_enabled_flag) {	
num_tile_set_rows_minus1	ue(v)
num_tile_set_columns_minus1	ue(v)
for(k = 0; k <= num_tile_set_rows_minus1; k++)	
num_tile_rows_in_tileset_minus1[k]	ue(v)
for(l = 0; l <= num_tile_set_columns_minus1; l++)	
num_tile_columns_in_tileset_minus1[l]	ue(v)
}	
...	
rbp_trailing_bits()	
}	

10

20

30

40

【0201】

表8に関して、それぞれのシンタックス要素は、以下の定義に基づくことができる。

【0202】

num_tile_columns_minus1プラス1は、ピクチャを区画するタイル列の数を指定す

50

る。num_tile_columns_minus1は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_columns_minus1の値が存在しない場合、num_tile_columns_minus1の値は0に等しいと推測される。

【0203】

num_tile_rows_minus1プラス1は、ピクチャを区画するタイル行の数を指定する。num_tile_rows_minus1は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_rows_minus1の値が存在しない場合、num_tile_rows_minus1の値は0に等しいと推測される。

【0204】

tiles_enabled_flagが1に等しい場合、num_tile_columns_minus1及びnum_tile_rows_minus1は、いずれも0に等しくないものとする。 10

【0205】

1に等しいuniform_spacing_flagは、タイル列境界及び同様にタイルの行境界が、ピクチャにわたって均一に分布されることを指定する。0に等しいuniform_spacing_flagは、タイル列境界及び同様にタイルの行境界が、ピクチャにわたって均一に分布されないが、シンタックス要素column_width_minus1[i]及びrow_height_minus1[i]を使用して明示的にシグナリングされることを指定する。uniform_spacing_flagの値が存在しない場合、uniform_spacing_flagの値は1に等しいと推測される。

【0206】

column_width_minus1[i]プラス1は、符号化ツリーブロックの単位におけるi番目のタイル列の幅を指定する。 20

【0207】

row_height_minus1[i]プラス1は、符号化ツリーブロックの単位におけるi番目のタイル行の高さを指定する。

【0208】

1に等しいloop_filter_across_tiles_enabled_flagは、インループフィルタリング操作が、PPSを参照して、ピクチャ内のタイル境界を越えて実行され得ることを指定する。0に等しいloop_filter_across_tiles_enabled_flagは、インループフィルタリング操作が、PPSを参照して、ピクチャ内のタイル境界を越えて実行されないことを指定する。インループフィルタリング操作は、デブロッキングフィルタ及びサンプル適応オフセットフィルタ操作を含む。loop_filter_across_tiles_enabled_flagが存在しない場合、loop_filter_across_tiles_enabled_flagの値は1に等しいと推測される。 30

【0209】

1に等しいtilesets_enabled_flagは、PPSを参照して、各ピクチャ内に2つ以上のタイルセットが存在することを指定する。0に等しいtilesets_enabled_flagは、PPSを参照して、各ピクチャ内に1つのタイルセットのみが存在することを指定する。

【0210】

別の例では、1に等しいtilesets_enabled_flagは、シンタックス要素num_tile_set_rows_minus1、num_tile_sets_columns_minus1、num_tile_rows_in_tileset_minus1[k]、num_tile_columns_in_tileset_minus1[l]が存在することを示す。0に等しいtilesets_enabled_flagは、シンタックス要素num_tile_set_rows_minus1、num_tile_sets_columns_minus1、num_tile_rows_in_tileset_minus1[k]、num_tile_columns_in_tileset_minus1[l]が存在しないことを示す。 40

【0211】

一実施例では、tilesets_enabled_flagが0に等しい場合、num_tile_set_rows_minus1は、0に等しいと推測され、num_tile_sets_columns_minus1は、0に等しいと推測される(すなわち、画像全体が単一タイルセットである)。

【0212】

一実施例では、CVS内でアクティブ化される全てのPPSについては、tilesets_enabled_flagの値が同じであることが、ビットストリームの適合性の要件である。tiles_ena 50

bled_flagが0に等しい場合、tilesets_enabled_flagは、0に等しいと推測される。

【0213】

別の例では、tiles_enabled_flagが0に等しい場合、tilesets_enabled_flagが0に等しいことがビットストリームの適合性の要件である。

【0214】

num_tile_set_rows_minus1プラス1は、ピクチャを区画するタイル行の数を指定する。num_tile_set_rows_minus1は、両端値を含む、0～num_tile_rows_minus1の範囲にあるものとする。別の例では、num_tile_set_rows_minus1は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_set_rows_minus1の値が存在しない場合、num_tile_set_rows_minus1の値は0に等しいと推測される。

10

【0215】

num_tile_set_columns_minus1プラス1は、ピクチャを区画するタイル列の数を指定する。num_tile_set_columns_minus1は、両端値を含む、0～num_tile_columns_minus1の範囲にあるものとする。別の例では、num_tile_set_columns_minus1は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_set_columns_minus1の値が存在しない場合、num_tile_set_columns_minus1の値は0に等しいと推測される。

【0216】

一実施例では、tilesets_enabled_flagが1に等しい場合、num_tile_set_columns_minus1及びnum_tile_set_rows_minus1は、いずれも0に等しくないものとする。

20

【0217】

num_tile_rows_in_tileset_minus1[k]プラス1は、両端値を含む、0～num_tile_set_rows_minus1の範囲にある各kについて、インデックス(k,l)に関連付けられたタイルセットにおけるタイル行の数を指定する。num_tile_rows_in_tileset_minus1[k]は、両端値を含む、0～num_tile_rows_minus1の範囲にあるものとする。別の例では、num_tile_rows_in_tileset_minus1[k]は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_rows_in_tileset_minus1[k]の値が存在しない場合、num_tile_rows_in_tileset_minus1[k]の値はnum_tile_rows_minus1に等しいと推測される。一実施例では、num_tile_rows_in_tileset_minus1[k]の値が存在しない場合、num_tile_rows_in_tileset_minus1[k]の値は0に等しいと推測される。

30

【0218】

一実施例では、両端値を含む、0～num_tile_set_rows_minus1の範囲にあるkについて、全ての(num_tile_rows_in_tileset_minus1[k]+1)の合計が、(num_tile_rows_minus1+1)に等しいことがビットストリームの適合性の要件である。

【0219】

別の例では、num_tile_rows_minus1[k]は、両端値を含む、0～PicHeightInCtbsY-num_tile_set_rows_minus1-1の範囲にあるものとする。

【0220】

num_tile_columns_in_tileset_minus1[l]プラス1は、両端値を含む、0～num_tile_set_columns_minus1の範囲にある各lについて、インデックス(k,l)に関連付けられたタイルセットにおけるタイル列の数を指定する。num_tile_columns_in_tileset_minus1[l]は、両端値を含む、0～num_tile_columns_minus1の範囲にあるものとする。別の例では、num_tile_columns_in_tileset_minus1[l]は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_columns_in_tileset_minus1[l]の値が存在しない場合、num_tile_columns_in_tileset_minus1[l]の値はnum_tile_columns_minus1に等しいと推測される。一実施例では、num_tile_columns_in_tileset_minus1[l]の値が存在しない場合、num_tile_columns_in_tileset_minus1[l]の値は0に等しいと推測される。

40

50

【0221】

一実施例では、両端値を含む、 $0 \sim \text{num_tile_set_columns_minus1}$ の範囲にある l について、全ての $(\text{num_tile_columns_in_tileset_minus1}[l]+1)$ の合計が、 $(\text{num_tile_columns_minus1}+1)$ に等しいことがビットストリームの適合性の要件である。

【0222】

別の例では、 $\text{num_tile_columns_minus1}[l]$ は、両端値を含む、 $0 \sim \text{PicWidthInCtbsY} - \text{num_tile_set_columns_minus1} - 1$ の範囲にあるものとする。

【0223】

一実施例では、タイル構造シンタックス(すなわち、ITU-T H.265からのシンタックス要素)は、PPSにシグナリングされてもよく、新たに提案されたタイルセット関連シンタックスは、SPSにおいてシグナリングされてもよい。

10

【0224】

一実施例では、上述のようにシンタックスがPPSでシグナリングされる場合、以下のよう、タイルセットの範囲について定義することができる:

ピクチャのセットPPSassociatedPicSetは、復号順序で連続し、(slice_pic_parameter_set_idを含めることによって)関連するPPSがスライスヘッダ内でアクティブ化されている全てのピクチャのセットである。次いで、PPS内でシグナリングされたタイルセットの範囲は、ピクチャのセットPPSassociatedPicSetである。

【0225】

別の実施例では、

20

PPSVaIAに等しいpps_pic_parameter_set_id値を有するPPSにおいてシグナリングされるタイルセットの範囲は、復号順序で連続し、そのスライスヘッダは、PPSVaIAに等しいslice_pic_parameter_set_idの値を有するピクチャのセットであり、復号順序のこのピクチャのセットの前後のピクチャは、PPSVaIAに等しい等しくないslice_pic_parameter_set_idを有する。符号化ビデオシーケンスには、複数のそのようなセットが存在してもよい。

【0226】

一実施例では、符号化ツリーブロックラスタ及びタイル走査変換プロセスは、以下のとおりであり得る。

【0227】

30

CTBの単位で、インデックス (k, l) に関連付けられたタイルセットの i 番目のタイル列の幅を指定する、両端値を含む、 $0 \sim \text{num_tile_set_rows_minus1}$ の範囲の k 、及び両端値を含む、 $0 \sim \text{num_tile_set_columns_minus1}$ の範囲の l 、両端値を含む、 $0 \sim \text{num_tile_columns_in_tileset_minus1}[l]$ の範囲の i についてのリスト $\text{colWidth}[k][l][i]$ は、以下のように導出される。

【0228】

40

50

【表 18】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for( i = 0; i  <=  num_tile_columns_in_tileset_minus1[l]; i-- )

                colWidth[ k ][ l ][ i ] = ( ( i + 1 ) * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 ) -
                    ( i * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 )
                    10

            else {

                colWidth[ k ][ l ][ num_tile_columns_minus1 ] = PicWidthInCtbsY

                for( i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++ ) {

                    colWidth[ k ][ l ][ i ] = column_width_minus1[ i ] + 1

                    colWidth[ k ][ l ][ num_tile_columns_minus1 ] -= colWidth[ k ][ l ][ i ]
                    20

                }

            }

        }

    }

}

```

【0229】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのj番目のタイル行の高さを指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_rows_in_tileset_minus1[k]の範囲のjについてのリストrowHeight[k][l][j]は、以下のように導出される。

【0230】

【表 19】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for(j = 0; j  <=  num_tile_rows_in_tileset_minus1[k]; j++)

                rowHeight[ k ][ l ][ j ] = ( ( j + 1 ) * PicHeightInCtbsY ) / ( num_tile_rows_minus1 + 1 ) -
                                                ( j * PicHeightInCtbsY ) / ( num_tile_rows_minus1 + 1 )
            10

        else {

            rowHeight[ k ][ l ][ num_tile_rows_minus1 ] = PicHeightInCtbsY

            for(j = 0; j <= num_tile_rows_in_tileset_minus1[k]; j++) {

                rowHeight[ k ][ l ][ j ] = row_height_minus1 [ j ] + 1

                rowHeight[ k ][ l ][ num_tile_rows_minus1 ] -= rowHeight[ k ][ l ][ j ]
                20

            }

        }

    }

}

```

【0 2 3 1】

別の例では、上記の導出は、以下のように実行され得る。

30

【0 2 3 2】

【数 8】

$$\text{NumTileSets} = (\text{num_tile_set_rows_minus1} + 1) * (\text{num_tile_set_columns_minus1} + 1);$$

【0 2 3 3】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのi番目のタイル列の幅を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_columns_in_tileset_minus1[l]の範囲のiについてのリストcolWidth[k][l][i]は、以下のように導出される。

40

【0 2 3 4】

【表 2 0】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for( i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++ )

                colWidth[ k ][ l ][ i ] = ( ( i + 1 ) * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 ) -
                    ( i * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 )
                    10

            else {

                colWidth[ k ][ l ][ num_tile_columns_minus1 ] = PicWidthInCtbsY

                for( i = 0; i < num_tile_columns_in_tileset_minus1[l]; i++ ) {

                    colWidth[ k ][ l ][ i ] = column_width_minus1[ i ] + 1

                    colWidth[ k ][ l ][ num_tile_columns_minus1 ] -= colWidth[ k ][ l ][ i ]
                    20

                }

                if(NumTileSets>1){

                    colWidth[ k ][ l ][ i ] = column_width_minus1[ i ] + 1

                    colWidth[ k ][ l ][ num_tile_columns_minus1 ] -= colWidth[ k ][ l ][ i ]

                }

            }
            30

        }

    }

}

```

【 0 2 3 5】

別の例では、上記の導出は、以下のように実行され得る。

【 0 2 3 6】

【数 9】

NumTileSets=(num_tile_set_rows_minus1+1)*(num_tile_set_columns_minus1+1);

【 0 2 3 7】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのi番目のタイル列の幅を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_columns_in_tileset_minus1[l]の範囲のiについてのリストcolWidth[k][l][i]は、以下のように導出される。

【 0 2 3 8】

【表 2 1】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for( i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++ )

                colWidth[ k ][ l ][ i ] = ( ( i + 1 ) * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 ) -
                    ( i * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 )
                    10

            else {

                colWidth[ k ][ l ][ num_tile_columns_minus1 ] = PicWidthInCtbsY

                for( i = 0; i < num_tile_columns_in_tileset_minus1[l]; i++ ) {

                    colWidth[ k ][ l ][ i ] = column_width_minus1[ i ] + 1

                    colWidth[ k ][ l ][ num_tile_columns_minus1 ] -= colWidth[ k ][ l ][ i ]
                    20

                }

                if(NumTileSets>1){

                    colWidth[ k ][ l ][ i ] = column_width_minus1[ i ] + 1

                    colWidth[ k ][ l ][ num_tile_columns_minus1 ] -= colWidth[ k ][ l ][ i ]

                }
                    30

            }

        }

    }

}

```

【0 2 3 9】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのj番目のタイル行の高さを指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_rows_in_tileset_minus1[k]の範囲のjについてのリストrowHeight[k][l][j]は、以下のように導出される。

【0 2 4 0】

【表 2 2】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for(j = 0; j <= num_tile_rows_in_tileset_minus1[k]; j++)

                rowHeight[ k ][ l ][ j ] = ((j + 1) * PicHeightInCtbsY) / ( num_tile_rows_minus1 + 1 ) -
                                                    (j * PicHeightInCtbsY) / ( num_tile_rows_minus1 + 1 )
                                                    10

        else {

            rowHeight[ k ][ l ][ num_tile_rows_minus1 ] = PicHeightInCtbsY

            for(j = 0; j < num_tile_rows_in_tileset_minus1[k]; j++) {

                rowHeight[ k ][ l ][ j ] = row_height_minus1 [ j ] + 1

                rowHeight[ k ][ l ][ num_tile_rows_minus1 ] -- rowHeight[ k ][ l ][ j ]
                                                    20

            }

            if(NumTileSets>1){

                rowHeight[ k ][ l ][ j ] = row_height_minus1 [ j ] ÷ 1

                rowHeight[ k ][ l ][ num_tile_rows_minus1 ] -- rowHeight[ k ][ l ][ j ]

                                                    30

            }

        }

    }

}

```

【0 2 4 1】

更に別の例では、上記の導出は、以下のように実行され得る。

【0 2 4 2】

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのi番目のタイル列の幅を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_columns_in_tileset_minus1[l]の範囲のiについてのリストcolWidth[k][l][i]は、以下のように導出される。

【0 2 4 3】

40

【表 2 3】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for( i = 0; i  <=  num_tile_columns_in_tileset_minus1[l]; i++)

                colWidth[ k ] [ l ] [ i ] = ( ( i + 1 ) * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 ) -
                                                    ( i * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 )
                                                    10

        else {

            colWidth[ k ] [ l ] [ num_tile_columns_minus1 ] = PicWidthInCtbsY

            for( i = 0; i < num_tile_columns_in_tileset_minus1[l]; i++) {

                colWidth[ k ] [ l ] [ i ] = column_width_minus1[ i ] + 1

                colWidth[ k ] [ l ] [ num_tile_columns_minus1 ] += colWidth[ k ] [ l ] [ i ]
                20
            }

        }

    }

}

}

```

【0 2 4 4】

30

CTBの単位で、インデックス(k,l)に関連付けられたタイルセットのj番目のタイル行の高さを指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_rows_in_tileset_minus1[k]の範囲のjについてのリストrowHeight[k][l][j]は、以下のように導出される。

【0 2 4 5】

40

50

【表 2 4】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        if( uniform_spacing_flag )

            for(j = 0; j  <=  num_tile_rows_in_tileset_minus1[k]; j++)

                rowHeight[ k ][ l ][ j ] = ((j + 1) * PicHeightInCtbsY) / ( num_tile_rows_minus1 + 1 ) -
                                                    (j * PicHeightInCtbsY) / ( num_tile_rows_minus1 + 1 )
                    10

        else {

            rowHeight[ k ][ l ][ num_tile_rows_minus1 ] = PicHeightInCtbsY

            for(j = 0; j < num_tile_rows_in_tileset_minus1[k] ; j++) {

                rowHeight[ k ][ l ][ j ] = row_height_minus1 [ j ] + 1

                rowHeight[ k ][ l ][ num_tile_rows_minus1 ]  -=  rowHeight[ k ][ l ][ j ]
                    20

            }

        }

    }

}

```

【0 2 4 6】

タイルセットの数を示す変数NumTileSetsは、以下のように導出される。

30

【0 2 4 7】

【数 1 0】

$$\text{NumTileSets} = (\text{num_tile_set_rows_minus1} + 1) * (\text{num_tile_set_columns_minus1} + 1)$$

【0 2 4 8】

ルマCTBにおけるピクチャ幅、ルマCTBにおけるピクチャ高さ、及びインデックス(k, l)に関連付けられたタイルセットのルマCTBにおけるピクチャサイズをそれぞれ指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについてのアレイpwctbsy[k][l]、phctbsy[k][l]、psizectbsy[k][l]、並びにインデックス(k,l)に関連付けられたタイルセットまでのCTBの累積カウントを指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについてのアレイctbAddrRSOffset

40

[k][l]は、以下のように導出される。

【0 2 4 9】

【表 2 5】

```

ctbAddrRsoffsetCalc=0; picSizeInCtbsY=0;

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        for(pwctbsy[ k ][ l ] = 0, i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++) {

            pwctbsy[ k ][ l ] += (colWidth [ k ][ l ][ i ])

        }

        for(phctbsy[ k ][ l ] = 0, i = 0; i <= num_tile_rows_in_tileset_minus1[k]

            ; i++) {

                phctbsy[ k ][ l ] += (rowHeight [ k ][ l ][ i ]);

            }

            psizectbsy[ k ][ l ] = pwctbsy[ k ][ l ] * phctbsy[ k ][ l ]

            picSizeInCtbsY += psizectbsy[ k ][ l ]

            ctbAddrRsoffset[ k ][ l ] = ctbAddrRsoffsetCalc

            ctbAddrRsoffsetCalc += psizectbsy[ k ][ l ]

        }

    }

```

【0 2 5 0】

一実施例では、上述のように、

【0 2 5 1】

40

50

【表 2 6】

picSizeInCtbsY will not be set equal to 0 and will not be updated as $\text{picSizeInCtbsY} += \text{psizectbsy}[k][l]$. In this case picSizeInCtbsY is calculated as follows as in ITU-T H.265:

$$\text{PicWidthInMinCbsY} = \text{pic_width_in_luma_samples} / \text{MinCbSizeY}$$

$$\text{PicWidthInCtbsY} = \text{Ceil}(\text{pic_width_in_luma_samples} \div \text{CtbSizeY})$$

$$\text{PicHeightInMinCbsY} = \text{pic_height_in_luma_samples} / \text{MinCbSizeY}$$

$$\text{PicHeightInCtbsY} = \text{Ceil}(\text{pic_height_in_luma_samples} \div \text{CtbSizeY})$$

$$\text{PicSizeInMinCbsY} = \text{PicWidthInMinCbsY} * \text{PicHeightInMinCbsY}$$

$$\text{PicSizeInCtbsY} = \text{PicWidthInCtbsY} * \text{PicHeightInCtbsY}$$

【0 2 5 2】

j 番目のタイルセットは、以下のようにインデックス(k,l)と関連付けられる。j 番目のタイルセットは、インデックスjを有するタイルセットと呼ばれることがある。

【0 2 5 3】

タイルセットインデックス又はタイルセット識別子jを考慮すると、ピクチャのタイルセット列の数k及びlは、以下のように導出される。

【0 2 5 4】

【数 1 1】

$$k = j / (\text{num_tile_set_columns_minus1} + 1)$$

$$l = j \% (\text{num_tile_set_columns_minus1} + 1)$$

【0 2 5 5】

インデックスk及びl並びにピクチャのタイルセット列の数を考慮すると、タイルセットインデックス又はタイルセット識別子jは、以下のように導出される。

【0 2 5 6】

【数 1 2】

$$j = k * (\text{num_tile_set_columns_minus1} + 1) + l$$

【0 2 5 7】

両端値を含む、 $0 \sim \text{num_tile_set_rows_minus1}$ の範囲のk、及び両端値を含む、 $0 \sim \text{num_tile_set_columns_minus1}$ の範囲のlについての各タイルセットにおけるタイルの数は、以下のように導出される。

【0 2 5 8】

10

20

30

40

50

【表 2 7】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        j = k*( num_tile_set_columns_minus1+1)+l

        NumTiles[j]=

        (num_tile_rows_in_tileset_minus1[k]+1)*(num_tile_columns_in_tileset_minus1[l]+1)

    }

}

```

10

【0 2 5 9】

符号化ツリーブロックの単位で、インデックス(k,D)に関連付けられたタイルセットのi番目のタイル列境界の場所を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_columns_in_tileset_minus1[k]+1の範囲のiについてのリストcolBd[k][l]

20

[i]は、以下のように導出される。

【0 2 6 0】

【表 2 8】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        for( colBd[k][l][0]=0,i=0;i <= num_tile_columns_in_tileset_minus1[l]; i++)

            colBd[k][l][i+1]=colBd[k][l][i]+colWidth[k][l][i]

    }

}

```

30

【0 2 6 1】

符号化ツリーブロックの単位で、インデックス(k,l)に関連付けられたタイルセットのj番目のタイル行境界の場所を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のl、両端値を含む、0～num_tile_rows_in_tileset_minus1[k]+1の範囲のjについてのリストrowBd[j]は、以下のように導出される。

40

【0 2 6 2】

50

【表 2 9】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        for(rowBd[k][l][0]=0,j=0;j <= num_tile_rows_in_tileset_minus1[k];j++)

            rowBd[k][l][j+1]=rowBd[k][l][j]+rowHeight[k][l][j]

    }

}

```

10

【0 2 6 3】

ピクチャのCTBラスタ走査におけるCTBアドレスからタイルセット及びタイル走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrRsであるリストCtbAddrRsToTs[ctbAddrRs]は、以下のように導出される。

【0 2 6 4】

20

30

40

50

【表 3 0】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        for( ctbAddrRs = 0; ctbAddrRs < psizectbsy[ k ][ 1 ]; ctbAddrRs++ ) {

            tbX = ctbAddrRs % pwctbsy[ k ][ 1 ]

            tbY = ctbAddrRs / pwctbsy[ k ][ 1 ]

            for( i = 0; i <= num_tile_columns_in_tileset_minus1[ l ]; i++ )

                if( tbX >= colBd[ k ][ 1 ][ i ] )

                    tileX = i

            for( j = 0; j <= num_tile_rows_in_tileset_minus1[ k ]; j++ )

                if( tbY >= rowBd[ k ][ 1 ][ j ] )

                    tileY = j

            CtbAddrRsToTs[ ctbAddrRs + ctbAddrROffset[ k ][ 1 ] ] = 0

            for( i = 0; i < tileX; i++ )

                CtbAddrRsToTs[ ctbAddrRs + ctbAddrROffset[ k ][ 1 ] ] += rowHeight[ k ][ 1 ]

            [ tileY ] * colWidth[ i ]

            for( j = 0; j < tileY; j++ )

                CtbAddrRsToTs[ ctbAddrRs + ctbAddrROffset[ k ][ 1 ] ] += pwctbsy[ k ][ 1 ] *

            rowHeight[ k ][ 1 ][ j ]

            CtbAddrRsToTs[ ctbAddrRs + ctbAddrROffset[ k ][ 1 ] ] += ( tbY - rowBd[ k ][ 1 ]

            [ tileY ] ) * colWidth[ k ][ 1 ][ tileX ] + tbX - colBd[ k ][ 1 ][ tileX ]

            CtbAddrRsToTs[ ctbAddrRs + ctbAddrROffset[ k ][ 1 ] ] += ctbAddrROffset[ k ][ 1 ]

        }

    }

}

```

【0 2 6 5】

タイル走査におけるCTBアドレスからピクチャのCTBラスト走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrTsであるリストCtbAddrTsToRs[ctbAddrTs]は、以下のように導出される。

【0 2 6 6】

【表 3 1】

```

for( ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++ )

    CtbAddrTsToRs[ CtbAddrRsToTs[ ctbAddrRs ] ] = ctbAddrRs

```

【0267】

タイル走査におけるCTBアドレスからタイルIDへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrTsであるリストTileId[ctbAddrTs]は、以下のように導出される。

【0268】

【表32】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        for(j=0,tileIdx=0;j <= num_tile_rows_in_tileset_minus1[k];j++)

            for(i=0;i <= num_tile_columns_in_tileset_minus1[l];i++,tileIdx++)

                for(y=rowBd[k][l][j];y<rowBd[k][l][j+1];y++)

                    for(x=colBd[k][l][i];x<colBd[k][l][i+1];x++)

                        TileId[CtbAddrRsToTs[y*PicWidthInCtbsY+x+ctbAddrRSOffset[k]

                            [l]]]=tileIdx

    }

}

```

10

20

【0269】

一実施例では、サブビットストリーム抽出又は他のプロセスのために、追加の計算が必要であり得る。一実施例では、更なる計算は、以下のとおりであり得る。

【0270】

両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについて、タイル列の数マイナス1及びタイル行の数マイナス1は、以下のように導出される。

【0271】

【表33】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){

    for(l=0;l<=num_tile_set_columns_minus1;l++){

        num_tile_columns_minus1[k][l]=num_tile_columns_in_tileset_minus1[l]

        num_tile_rows_minus1[k][l]=num_tile_rows_in_tileset_minus1[k]

    }

}

```

30

40

【0272】

タイルの単位におけるインデックス(*,l)境界に関連付けられたタイルセットの場所を指定する、両端値を含む、0～num_tile_columns_minus1+1の範囲のlについてのリストtilecolumnPos[l]は、以下のように導出される。

【0273】

50

【表 3 4】

```

for( l = 0, tilecolumnPos[0]=0; l <= num_tile_set_columns_minus1; l++)
    tilecolumnPos[l+1]= tilecolumnPos[l]+
num_tile_columns_in_tileset_minus1[l]

```

【0 2 7 4】

タイルの単位におけるインデックス(k,*)境界に関連付けられたタイルセットの場所を指定する、両端値を含む、0～num_tile_rows_minus1+1の範囲のkについてのリストtilerowPos[k]は、以下のように導出される。

【0 2 7 5】

10

【表 3 5】

```

for( k = 0, tilerowPos[0]=0; k <= num_tile_set_rows_minus1; k++)
    tilerowPos[k+1]=tilerowPos[k]+ num_tile_rows_in_tileset_minus1[k]

```

【0 2 7 6】

符号化ツリーブロックの単位で、インデックス(k,l)に関連付けられたタイルセットのタイル列境界の場所を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについてのリストcolBd

[k][l]、及び

20

符号化ツリーブロックの単位で、インデックス(k,l)に関連付けられたタイルセットのタイル行境界の場所を指定する、両端値を含む、0～num_tile_set_rows_minus1の範囲のk、及び両端値を含む、0～num_tile_set_columns_minus1の範囲のlについてのリストrowBd

[k][l]は、以下のように導出される。

【0 2 7 7】

【表 3 6】

```

for(k=0;k<=num_tile_set_rows_minus1;k++){
    for(l=0;l<=num_tile_set_columns_minus1;l++){
        colBd[ k ][ l ]= colWidth[tilecolumnPos[l]]
        rowBd[ k ][ l ]= rowHeight[tilerowPos[k]]
    }
}

```

30

【0 2 7 8】

表9は、タイルセット内のタイルセット及びタイルのそれぞれが、本明細書の技術に従って独立して処理され得るように、情報をシグナリングするために使用され得る、例示的なスライスセグメントヘッダを示す。この場合、スライスは常に、単一の完全なタイルセットを含む。この場合、スライスヘッダにおいて以下のシグナリングが行われる(代わりにタイルセットヘッダ又はセグメントヘッダ、又はそのような他の同様の名前と呼ばれる場合がある)。

40

【0 2 7 9】

50

【表 3 7】

表9

slice_segment_header() {	記述子
...	
if(tilesets_enabled_flag)	
tile_set_id	u(v)
...	
if(tiles_enabled_flag) {	
if(!tilesets_enabled_flag)	
num_entry_point_offsets	ue(v)
if(OffsetInfoPresent) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumOffsets; i++)	
entry_point_offset_minus1[i]	ue(v)
}	
...	

10

20

【0 2 8 0】

表9に関して、シンタックス要素**tile_set_id**、**num_entry_point_offsets**、**offset_len_minus1**、及び**entry_point_offset_minus1**は、以下の定義に基づくことができることに留意されたい。

【0 2 8 1】

tile_set_idは、このタイルセットのタイルセット識別子を指定する。タイルセットインデックス*k*及び*l*並びにピクチャのタイルセット列の数を考慮すると、タイルセットインデックス又はタイルセット識別子*j*は、以下のように導出される。

【0 2 8 2】

【数 1 3】

$$j = k * ((\text{num_tile_set_columns_minus1} + 1) + l)$$

30

【0 2 8 3】

タイルセット識別子は、代わりにタイルセットインデックスと呼ばれることがある。**tile_set_id**シンタックス要素の長さは $\text{Ceil}(\text{Log2}(\text{NumTileSets}))$ ビットである。

40

【0 2 8 4】

num_entry_point_offsets、**offset_len_minus1**、及び**entry_offset_minus1[i]**は、ITU-T H.265に提供されるものと同様のセマンティクスを有してもよい。

【0 2 8 5】

一実施例では、**tiles_enabled_flag**が1に等しいときに、以下が導出される。

【0 2 8 6】

tilesets_enabled_flagが1に等しい場合:

【0 2 8 7】

【数 1 4】

50

OffsetInfoPresent=1

NumOffsets = NumTilesInTileset[tile_set_id] - 1

【 0 2 8 8 】

そうでなければ(すなわち、tilesets_enabled_flagが0に等しい場合)

【 0 2 8 9 】

【表 3 8】

OffsetInfoPresent= (num_entry_point_offsets? 1:0)

NumOffsets = num_entry_offsets

10

【 0 2 9 0 】

tilesets_enabled_flagが0に等しい場合: OffsetInfoPresent=0, NumOffsets=0

別の例では、表10は、タイルセット内のタイルセット及びタイルのそれぞれが、本明細書の技術に従って独立して処理され得るように、情報をシグナリングするために使用され得る、例示的なスライスセグメントヘッダを示す。この場合、スライスは、整数個の完全なタイルセットを含んでもよい。この場合、スライスヘッダにおいて以下のシグナリングが行われる(代わりにタイルセットヘッダ又はセグメントヘッダ、又はそのような他の同様の名前と呼ばれる場合がある)。

【 0 2 9 1 】

【表 3 9】

20

表10

slice_segment_header() {	記述子
...	
if(tilesets_enabled_flag) {	
tile_set_id	u(v)
num_tile_set_ids_minus1	u(v)
}	
...	
if(tiles_enabled_flag) {	
if(!tilesets_enabled_flag)	
num_entry_point_offsets	ue(v)
if(OffsetInfoPresent) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumOffsets; i++)	
entry_point_offset_minus1[i]	ue(v)
}	
...	

30

40

【 0 2 9 2 】

表10に関して、シンタックス要素tile_set_id、num_entry_point_offsets、off

50

set_len_minus1、及びentry_point_offset_minus1は、以下の定義に基づくことができることに留意されたい。

【0293】

tile_set_idは、このタイルセットのタイルセット識別子を指定する。タイルセットインデックスk及びl並びにピクチャのタイルセット列の数を考慮すると、タイルセットインデックス又はタイルセット識別子jは、以下のように導出される。

【0294】

【数15】

$$j = k * ((\text{num_tile_set_columns_minus1} + 1) + 1)$$

10

【0295】

タイルセット識別子は、代わりにタイルセットインデックスと呼ばれることがある。

【0296】

tile_set_idシンタックス要素の長さはCeil(Log2(NumTileSets))ビットである。

【0297】

num_tile_set_ids_minus1プラス1は、スライス内に存在するタイルセットの数(タイルセットのラスト走査順)を指定する。num_tile_set_ids_minus1シンタックス要素の長さはCeil(Log2(NumTileSets-1))ビットである。

【0298】

tiles_enabled_flagが1に等しいときに、以下が導出される。

20

【0299】

tilesets_enabled_flagが1に等しい場合:

【0300】

【表40】

OffsetInfoPresent=1

for(i=tile_set_id, NumOffsets=0; i < (tile_set_id+num_tile_set_ids_minus1+1);

i++)

{NumOffsets += NumTilesInTileset[i]}

NumOffsets=NumOffsets-1;

30

【0301】

そうでなければ(すなわち、tilesets_enabled_flagが0に等しい場合)

【0302】

【表41】

OffsetInfoPresent= (num_entry_point_offsets? 1:0)

NumOffsets=num_entry_offsets

【0303】

tiles_enabled_flagが0に等しい場合: OffsetInfoPresent = 0, NumOffsets=0

「スライスセグメント」は、代わりに、「スライス」又は「タイルセット」又は「タイルセット」又は「セグメント」又は「マルチctugroup」又は「タイルグループ」又は「タイルリスト」又は「タイル収集」などと呼ばれ得ることに留意されたい。したがって、これらの単語は、いくつかの場合では、互換的に使用されてもよい。また、同様の名前のデータ構造名は互換可能である。「スライスセグメントヘッダ」は、代わりに、「スライスヘッダ」又は「タイルセットヘッダ」又は「タイルセットヘッダ」又は「セグメントヘッダ」又は「マルチctugroupヘッダ」又は「タイルグループヘッダ」又は「タイルリストヘッダ」又は「タイル収集ヘッダ」などと呼ばれ得ることに留意されたい。したがって、これらの単語は、互換的に使用される。また、同様の名前のデータ構造名は、いくつかの場合では、互換可能に使用されてもよい。

40

【0304】

50

表11は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表11に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表11に含まれる例示的なシンタックスは、VPS又はSPS又は他のパラメータセットに含まれてもよい。他の例では、表11に含まれる例示的なシンタックスは、タイルグループヘッダ又はスライスヘッダに含めることができる。

【0305】

【表42】

表11

parameter_set_rbsp() {	記述子
...	
tilesets_enabled_flag	u(1)
if(tilesets_enabled_flag) {	
num_tile_sets_in_pic_minus1	ue(v)
for(i = 0; i <= num_tile_sets_in_pic_minus1; i++) {	
top_left_tile_id[i]	u(v)
num_tile_rows_in_tileset_minus1[i]	u(v)
num_tile_columns_in_tileset_minus1[i]	u(v)
}	
}	
...	
rbp_trailing_bits()	
}	

10

20

【0306】

表11に関して、それぞれのシンタックス要素は、以下の定義に基づくことができる。

30

【0307】

1に等しいtilesets_enabled_flagは、パラメータセットを参照して、各ピクチャ内に2つ以上のタイルセットが存在することを指定する。0に等しいtilesets_enabled_flagは、パラメータセットを参照して、各ピクチャ内に1つのタイルセットのみが存在することを指定する。変形例では、0に等しいtilesets_enabled_flagは、各タイルが、PPSを参照して、各ピクチャ内のタイルセットであることを指定してもよい。

【0308】

CVS内でアクティブ化される全てのPPSについては、tilesets_enabled_flagの値が同じであることが、ビットストリームの適合性の要件である。tiles_enabled_flagが0に等しい場合、tilesets_enabled_flagは、0に等しいと推測される。

40

【0309】

num_tile_sets_in_pic_minus1プラス1は、ピクチャ内のタイルセットの数を指定する。

【0310】

top_left_tile_idは、i番目のタイルセットの左上の角に配置されるタイルのタイルIDを指定する。top_left_tile_id[i]の長さはCeil(Log2(num_tilesets_in_pic_minus1+1))ビットである。jに等しくない任意のiについて、top_left_tile_id[i]の値は、top_left_tile_id[j]の値に等しくないものとする。

【0311】

num_tile_rows_in_tileset_minus1[i]プラス1は、両端値を含む、0～(num_tile

50

sets in pic minus1)の範囲にある各*i*について、*i*番目のタイルセットにおけるタイル行の数を指定する。num_tile_rows_in_tileset_minus1[i]は、両端値を含む、0～num_tile_rows_minus1の範囲にあるものとする。num_tile_rows_in_tileset_minus1[i]の長さはCeil(Log2

(num_tile_rows_minus1+1))ビットである。

【0 3 1 2】

num_tile_rows_in_tileset_minus1[i]の値が存在しない場合、num_tile_rows_in_tileset_minus1[i]の値は0に等しいと推測される。

【0 3 1 3】

変形例では、num_tile_rows_in_tileset_minus1[i]の値が存在しない場合、num_tile_rows_in_tileset_minus1[i]の値はnum_tile_rows_minus1に等しいと推測される。

【0 3 1 4】

num_tile_columns_in_tileset_minus1[l]プラス1は、両端値を含む、0～(num_tile_sets_in_pic_minus1)の範囲にある各*i*について、*i*番目のタイルセットにおけるタイル列の数を指定する。num_tile_columns_in_tileset_minus1[l]は、両端値を含む、0～num_tile_columns_minus1の範囲にあるものとする。num_tile_columns_in_tileset_minus1[i]の長さはCeil(Log2(num_tile_columns_minus1+1))ビットである。

【0 3 1 5】

num_tile_columns_in_tileset_minus1[l]の値が存在しない場合、num_tile_columns_in_tileset_minus1[l]の値は0に等しいと推測される。

【0 3 1 6】

変形例では、num_tile_columns_in_tileset_minus1[i]の値が存在しない場合、num_tile_columns_in_tileset_minus1[l]の値はnum_tile_columns_minus1に等しいと推測される。

【0 3 1 7】

変形例では、num_tile_rows_m_tileset_minus1[i]及びnum_tile_columns_in_tileset_minus1[l]は、符号化されたue(v)である。

【0 3 1 8】

変形例では、1つ又は2つの別個の追加のシンタックス要素は、num_tile_rows_in_tileset_minus1[i]及び/又はnum_tile_columns_in_tileset_minus1[l]に使用されるビット数をシグナリングする。

【0 3 1 9】

表12は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表12に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表12に含まれる例示的なシンタックスは、VPS又はSPS又は他のパラメータセットに含まれてもよい。他の例では、表12に含まれる例示的なシンタックスは、タイルグループヘッダ又はスライスヘッダに含めることができる。

【0 3 2 0】

10

20

30

40

50

【表 4 3】

表12

parameter_set_rbsp() {	記述子
...	
tilesets_enabled_flag	u(1)
if(tilesets_enabled_flag) {	
num_tile_sets_in_pic_minus1	ue(v)
remaining_tiles_tileset_flag	u(1)
for(i=0; i < (num_tile_sets_in_pic_minus1+!remaining_tiles_tileset_flag); i++)	
{	
top_left_tile_id[i]	u(v)
num_tile_rows_in_tileset_minus1[i]	u(v)
num_tile_columns_in_tileset_minus1[i]	u(v)
}	
}	
...	
rbp_trailing_bits()	
}	

10

20

【0 3 2 1】

表12に関して、それぞれのシンタックス要素は、上記に提供された定義及び以下の定義に基づくことができる。

【0 3 2 2】

1に等しいremaining_tiles_tileset_flagは、シンタックス要素top_left_tile_id[i]、num_tile_rows_in_tileset_minus1[i]、num_tile_columns_in_tileset_minus1[i]によってシグナリングされた(num_tilesets_in_pic_minus1-1)タイルセット内で明示的に指定されたもの以外のピクチャ内の残りのタイルの全てが最後のタイルセットを形成することを指定する。0に等しいremaining_tiles_tileset_flagは、全てのnum_tilesets_in_pic_minus1タイルセットが、シンタックス要素top_left_tile_id[i]、num_tile_rows_in_tileset_minus1[i]、num_tile_columns_in_tileset_minus1[i]をシグナリングすることによって明示的に指定されることを指定する。

30

【0 3 2 3】

num_tile_rows_in_tileset_minus1[i]プラス1は、両端値を含む、0～(num_tile_sets_in_pic_minus1+!remaining_tiles_tileset_flag-1)の範囲にある各iについて、i番目のタイルセットにおけるタイル行の数を指定する。num_tile_rows_in_tileset_minus1[i]は、両端値を含む、0～num_tile_rows_minus1の範囲にあるものとする。num_tile_rows_in_tileset_minus1[i]の長さはCeil(Log2(num_tile_rows_minus1+1))ビットである。

40

【0 3 2 4】

num_tile_rows_in_tileset_minus1[i]の値が存在しない場合、num_tile_rows_in_tileset_minus1[i]の値は0に等しいと推測される。

【0 3 2 5】

変形例では、num_tile_rows_in_tileset_minus1[i]の値が存在しない場合、num_tile_rows_in_tileset_minus1[i]の値はnum_tile_rows_minus1に等しいと推測される。

【0 3 2 6】

50

`num_tile_columns_in_tileset_minus1[l]` プラス1は、両端値を含む、 $0 \sim (\text{num_tile_sets_in_pic_minus1} + !\text{remainmg_tiles_tileset_flag} - 1)$ の範囲にある各 l について、 i 番目のタイルセットにおけるタイル列の数を指定する。`num_tile_columns_in_tileset_minus1[l]` は、両端値を含む、 $0 \sim \text{num_tile_columns_minus1}$ の範囲にあるものとする。`num_tile_columns_in_tileset_minus1[i]` の長さは $\text{Ceil}(\text{Log2}(\text{num_tile_columns_minus1} + 1))$ ビットである。

【0327】

`num_tile_columns_in_tileset_minus1[l]` の値が存在しない場合、`num_tile_columns_in_tileset_minus1[l]` の値は0に等しいと推測される。

【0328】

変形例では、`num_tile_columns_in_tileset_minus1[i]` の値が存在しない場合、`num_tile_columns_in_tileset_minus1[l]` の値は `num_tile_columns_minus1` に等しいと推測される。

【0329】

変形例では、ピクチャ内の各タイルが、 $0 \sim \text{num_tile_set_minus1}$ (両端値を含む) の範囲のタイルセットを有するタイルセットのうちの1つ及び1つのみに属することを、ビットストリームの適合性の要件とする。

【0330】

表12Aは、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表12Aに含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表12Aに含まれる例示的なシンタックスは、VPS又はSPS又は他のパラメータセットに含まれてもよい。他の例では、表12Aに含まれる例示的なシンタックスは、タイルグループヘッダ又はスライスヘッダに含めることができる。

【0331】

【表44】

表12A

<code>parameter_set_rbsp() {</code>	記述子
<code>...</code>	
<code>tilesets_enabled_flag</code>	$u(1)$
<code>if(tilesets_enabled_flag) {</code>	
<code>num_tile_sets_in_pic_minus1</code>	$uc(v)$
<code>for(i = 0; i < num_tile_sets_in_pic_minus1; i++) {</code>	
<code>top_left_tile_id[i]</code>	$u(v)$
<code>num_tile_rows_in_tileset_minus1[i]</code>	$u(v)$
<code>num_tile_columns_in_tileset_minus1[i]</code>	$u(v)$
<code>}</code>	
<code>}</code>	
<code>...</code>	
<code>rbp_trailing_bits()</code>	
<code>}</code>	

【0332】

表12Aに関して、それぞれのシンタックス要素は、上記に提供された定義に基づくことができる。

【0333】

表13は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表13に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表13に含まれる例示的なシンタックスは、VPS又はSPS又は他のパラメータセットに含まれてもよい。他の例では、表13に含まれる例示的なシンタックスは、タイルグループヘッダ又はスライスヘッダに含めることができる。

【0334】

【表45】

表13

parameter_set_rbsp() {	記述子
...	
tile_set_flag	u(1)
if(tile_set_flag) {	
num_tile_sets_in_pic_minus1	ue(v)
remaining_tiles_tileset_flag	u(1)
for(i = 0; i < (num_tile_sets_in_pic_minus1+remaining_tiles_tileset_flag); i++)	
{	
top_left_tile_id[i]	u(v)
bottom_right_tile_id[i]	u(v)
}	
}	
...	
rbp_trailing_bits()	
}	

10

20

【0335】

表13に関して、それぞれのシンタックス要素は、上記に提供された定義及び以下の定義に基づくことができる。

【0336】

bottom_right_tile_id[i]は、i番目のタイルセットの右下の角に配置されるタイルのタイルIDを指定する。bottom_right_tile_id[i]の長さは $\text{Ceil}(\text{Log}_2(\text{num_tile_sets_in_pic_minus1}+1))$ ビットである。jに等しくない任意のiについて、bottom_right_tile_id[i]の値は、bottom_right_tile_id[j]の値に等しくないものとする。

【0337】

表14は、本明細書の技術によるタイル構造を信号化するために使用され得るパラメータセットのシンタックスの例を示す。一実施例では、表14に含まれる例示的なシンタックスは、PPSに含まれてもよい。他の例では、表14に含まれる例示的なシンタックスは、VPS又はSPS又は他のパラメータセットに含まれてもよい。他の例では、表14に含まれる例示的なシンタックスは、タイルグループヘッダ又はスライスヘッダに含めることができる。

40

【0338】

50

【表 4 6】

表14

parameter_set_rbsp() {	記述子
...	
tile_set_flag	u(1)
if(tile_set_flag) {	
num_tile_sets_in_pic_minus1	ue(v)
for(i = 0; i < num_tile_sets_in_pic_minus1; i++) {	
top_left_tile_id[i]	u(v)
bottom_right_tile_id[i]	u(v)
}	
}	
...	
rbp_trailing_bits()	
}	

10

【 0 3 3 9】

20

表14に関して、それぞれのシンタックス要素は、上記に提供された定義に基づくことができる。

【 0 3 4 0】

表11～14に関して、表14Aは、タイルグループヘッダの例示的なシンタックスを示す。

【 0 3 4 1】

30

40

50

【表 4 7】

表14A

tile_group_header() {	記述子
tile_group_pic_parameter_set_id	ue(v)
tile_set_idx	u(v)
tile_group_type	ue(v)
if(tile_group_type != I) {	
log2_diff_ctu_max_bt_size	ue(v)
if(sps_sbtmvp_enabled_flag) {	
sbtmvp_size_override_flag	u(1)
if(sbtmvp_size_override_flag)	
log2_sbtmvp_active_size_minus2	u(3)
}	
if(sps_temporal_mvp_enabled_flag)	
tile_group_temporal_mvp_enabled_flag	u(1)
if(tile_group_type == B)	
mvd_l1_zero_flag	u(1)
if(tile_group_temporal_mvp_enabled_flag) {	
if(tile_group_type == B)	
collocated_from_l0_flag	u(1)
}	
six_minus_max_num_merge_cand	ue(v)
}	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(num_tiles_in_tile_group_minus1 > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < num_tiles_in_tile_group_minus1; i++)	
entry_point_offset_minus1[i]	u(v)
}	
byte_alignment()	
}	

【0 3 4 2】

表14Aに関して、それぞれのシンタックス要素は、上記に提供された定義及び以下の定義に基づくことができる。

【0 3 4 3】

tile_set_idxは、このタイルセットのタイルセットインデックスを指定する。tile_set_idxシンタックス要素の長さはCeil(Log2(NumTileSets))ビットである。

【0 3 4 4】

表11に関して、表15は、タイルグループデータの例示的なシンタックスを示す。

【0 3 4 5】

10

20

30

40

50

【表 4 8】

表15

tile_group_data() {	記述子
tileIdx = top_left_tile_id[tile_set_idx]	
for(k = 0; k <= num_tile_rows_in_tileset_minus1[tile_set_idx]; k++, tileIdx += (num_tile_columns_minus1 - num_tile_columns_in_tileset_minus1[tile_set_idx])) {	
for(i = 0; i <= num_tile_columns_in_tileset_minus1[tile_set_idx]; i++, tileIdx++) {	
ctbAddrInTs = FirstCtbAddrTs[tileIdx]	
for(j = 0; j < NumCtusInTile[tileIdx]; j++, ctbAddrInTs++) {	
CtbAddrInRs = CtbAddrTsToRs[ctbAddrInTs]	
coding_tree_unit()	
}//for j- CTUs in one tile	
end_of_tile_one_bit /* equal to 1 */	ae(v)
if(k < num_tile_rows_in_tileset_minus1[tile_set_idx] j < num_tile_columns_in_tileset_minus1[tile_set_idx])	
Variant	
if(!((k == num_tile_rows_in_tileset_minus1[tile_set_idx]) && (j == num_tile_columns_in_tileset_minus1[tile_set_idx])))	
byte_alignment()	
}//for i	
}//for k	
}	

10

20

【0 3 4 6】

表15に関して(With respect to)、一実施例では、タイル走査におけるCTBアドレスからタイルIDへの変換は、以下のとおりであり得る。

30

【0 3 4 7】

40

50

【表 4 9】

```

for( j = 0; tileIdx = 0; j <= num_tile_rows_minus1; j++ )
    for( i = 0; i <= num_tile_columns_minus1; i++, tileIdx++ ) {
        for( y = RowBd[ j ]; y < RowBd[ j + 1 ]; y++ )
            for( x = ColBd[ i ]; x < ColBd[ i + 1 ]; x++ ) {
                TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x ] ] = tileIdx
                FirstCtbAddrTs[ tileIdx ] =
10         CtbAddrRsToTs[ RowBd[ j ] * PicWidthInCtbsY + ColBd[ i ] ]
            }
            TileIndex[ j*( num_tile_columns_minus1+1)+i ] = tileIdx
        }

for( l = 0; l <= num_tile_sets_in_pic_minus1; l++ ) {
    TopLeftTileId[ l ] = top_left_tile_id[ l ];
20    NumTileRowsInTileSetMinus1[ l ] = num_tile_rows_in_tileset_minus1 [ l ];
    NumTileColumnsInTileSetMinus1[ l ] = num_tile_columns_in_tileset_minus1 [ l ];
}

```

【 0 3 4 8 】

表 12～14 に関して、表 16 は、タイルグループデータの例示的なシンタックスを示す。

【 0 3 4 9 】

30

40

50

【表 5 0】

表16

tile_group_data() {	記述子
tileIdx = TopLeftTileId[tile_set_idx]	
for(k = 0; k <= NumTileRowsInTileSetMinus1[tile_set_idx]; k++, tileIdx += (num_tile_columns_minus1 - num_tile_columns_in_tileset_minus1[tile_set_idx]))	
{	
for(i = 0; i <= NumTileColumnsInTileSetMinus1[tile_set_idx]; i++, tileIdx++) {	
ctbAddrInTs = FirstCtbAddrTs[tileIdx]	
for(j = 0; j < NumCtusInTile[tileIdx]; j++, ctbAddrInTs++) {	
CtbAddrInRs = CtbAddrTsToRs[ctbAddrInTs]	
coding_tree_unit()	
}//for j= CTUs in one tile	
end_of_tile_one_bit /* equal to 1 */	ae(v)
if(k < num_tile_rows_in_tileset_minus1[tile_set_idx] j < num_tile_columnss_in_tileset_minus1[tile_set_idx])	
Variant	
if(!((k == num_tile_rows_in_tileset_minus1[tile_set_idx]) && (j == num_tile_columnss_in_tileset_minus1[tile_set_idx]))))	
byte_alignment()	
}//for i	
}//for k	
}	

10

20

【0 3 5 0】

表12～14に関して、表16Aは、タイルグループヘッダの別の例示的なシンタックスを示す。表16と表16Aとの主な違いは、シンタックス要素の一部が、導出された変数によって置き換えられることである。

30

【0 3 5 1】

40

50

【表 5 1】

表16A

tile_group_data() {	記述子
tileIdx = TopLeftTileId[tile_set_idx]	
for(k = 0; k <= NumTileRowsInTileSetMinus1[tile_set_idx]; k++, tileIdx += (num_tile_columns_minus1 - NumTileColumnsInTileSetMinus1[tile_set_idx])) {	
for(i = 0; i <= NumTileColumnsInTileSetMinus1[tile_set_idx]; i++, tileIdx++) {	
ctbAddrInTs = FirstCtbAddrTs[tileIdx]	
for(j = 0; j < NumCtusInTile[tileIdx]; j++, ctbAddrInTs++) {	10
CtbAddrInRs = CtbAddrTsToRs[ctbAddrInTs]	
coding_tree_unit()	
}//for j= CTUs in one tile	
end_of_tile_one_bit /* equal to 1 */	ae(v)
if(k < NumTileRowsInTileSetMinus1 [tile_set_idx] j < NumTileColumnsInTileSetMinus1 [tile_set_idx])	
Variant	
if(!((k == NumTileRowsInTileSetMinus1 [tile_set_idx]) &&	20
(j == NumTileColumnsInTileSetMinus1 [tile_set_idx])))	
byte_alignment()	
}//for i	
}//for k	
}	

【0352】

表12及び表16及び表16Aに関して、一実施例では、タイル走査におけるCTBアドレスからタイルIDへの変換は、以下のとおりであり得る。

【0353】

10

20

30

40

50

【表 5 2】

```

for(j = 0, tileIdx = 0; j <= num_tile_rows_minus1; j++)
    for(i = 0; i <= num_tile_columns_minus1; i++, tileIdx++) {
        for(y = RowBd[j]; y < RowBd[j + 1]; y++)
            for(x = ColBd[i]; x < ColBd[i + 1]; x++) {
                TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x ] ] = tileIdx
                FirstCtbAddrTs[ tileIdx ] =
                    CtbAddrRsToTs[ RowBd[j] * PicWidthInCtbsY + ColBd[i] ]
            }
                TileIndex[ j*( num_tile_columns_minus1+1)+i ] = tileIdx
                RemTiles[tileIdx]=1
            }

for(l = 0; l < (num_tile_sets_in_pic_minus1+!remaining_tiles_tileset_flag); l++) {
    for( k = 0, tileIdx = top_left_tile_id[l]; k <= num_tile_rows_in_tileset_minus1[l]; k++,
    tileIdx +=
        (num_tile_columns_minus1 * num_tile_columns_in_tileset_minus1[l] ) )
        for(i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++, tileIdx++){
            RemTiles[tileIdx]=0
        }
        TopLeftTileId[l] = top_left_tile_id[l];
        NumTileRowsInTileSetMinus1[l] = num_tile_rows_in_tileset_minus1[l];
        NumTileColumnsInTileSetMinus1[l] = num_tile_columns_in_tileset_minus1[l];
        NumTilesInSlice[l] = (NumTileRowsInTileSetMinus1[l]+1)*( NumTileColumnsInTileSetMinus1[l]+1)
    }
    if(remaining_tiles_tileset_flag) {
        for(i=0, tllId=(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1), brId=0; i<(num_tile_columns_minus1+1)*( num_tile_rows_minus1+1); i++){
            if(RemTiles[i]){
                if(i<tllId) tllId=i;
                if(i>brId) brId=i;
            }
        }
    }

```

【 0 3 5 4 】

【表 5 3】

```

TopLeftTileId[ num_tile_sets_in_pic_minus1 ]= tId;
NumTileRowsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]=(brId·tId)/( num_tile_columns_minus1 + 1);
NumTileColumnsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]=(brId·tId)/( num_tile_columns_minus1 + 1);
NumTilesInSlice[ num_tile_sets_in_pic_minus1 ] =
(NumTileRowsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]+1)*( NumTileColumnsInTileSetMinus1 [ num_tile_sets_in_pic_minus1 ]+1)
}

```

10

【 0 3 5 5】

表 12A 及び表 16 及び表 16A に関して、一実施例では、タイル走査における CTB アドレスからタイル ID への変換は、以下のとおりであり得る。

【 0 3 5 6】

【表 5 4】

```

for( j = 0; tileIdx = 0; j <= num_tile_rows_minus1; j++ )
    for( i = 0; i <= num_tile_columns_minus1; i++, tileIdx++ ) {
        for( y = RowBd[ j ]; y < RowBd[ j + 1 ]; y++ )
            for( x = ColBd[ i ]; x < ColBd[ i + 1 ]; x++ ) {
                TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x ] ] = tileIdx
                FirstCtbAddrTs[ tileIdx ] =
CtbAddrRsToTs[ RowBd[ j ] * PicWidthInCtbsY + ColBd[ i ] ]
            }
        TileIndex[ j*( num_tile_columns_minus1+1)+i ] = tileIdx
        RemTiles[tileIdx]=1
    }

```

20

```

for( l = 0; l < num_tile_sets_in_pic_minus1; l++ ) {
    for( k = 0, tileIdx= top_left_tile_id[l]; k <= num_tile_rows_in_tileset_minus1[ l ]; k++, tileIdx
+=

```

30

```

        (num_tile_columns_minus1 -num_tile_columns_in_tileset_minus1[ l ] )
        for( i = 0; i <= num_tile_columns_in_tileset_minus1[ l ]; i++, tileIdx++ ){
            RemTiles[tileIdx]=0
        }
        TopLeftTileId[ l ]= top_left_tile_id[ l ];
        NumTileRowsInTileSetMinus1[ l ] = num_tile_rows_in_tileset_minus1 [ l ];

```

```

NumTilesInSlice[ l ]=(NumTileRowsInTileSetMinus1[ l ]+1)*( NumTileColumnsInTileSetMinus1 [ l ]+1)
}

```

40

```

for(i=0,tId=(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1),brId=0;i<(num_tile_columns_minus1+1)*( num_tile_rows_minus1+1);i++){
    if(RemTiles[i]){
        if(i<tId) tId=i;
        if(i>brId) brId=i;
    }
}
}

```

【 0 3 5 7】

50

【表 5 5】

```

TopLeftTileId[ num_tile_sets_in_pic_minus1 ]= tId;
NumTileRowsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]=(brId-tId)/( num_tile_columns_minus1 + 1);
NumTileColumnsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]=(brId-tId)%( num_tile_columns_minus1 + 1);
NumTilesInSlice[ num_tile_sets_in_pic_minus1 ]
=
(NumTileRowsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]+1)*( NumTileColumnsInTileSetMinus1 [ num_tile_sets_in_pic_minus1 ]+1)

```

10

【 0 3 5 8】

表 13 及び表 16 及び表 16 A に関して、一実施例では、タイル走査における CTB アドレスからタイル ID への変換は、以下のとおりであり得る。

【 0 3 5 9】

タイル走査における CTB アドレスからタイル ID への変換を指定する、両端値を含む、0 ~ PicSizeInCtbsY-1 の範囲の ctbAddrTs であるリスト TileId[ctbAddrTs]、及びタイルにおけるタイルインデックスから CTU の数への変換を指定する、両端値を含む、0 ~ PicSizeInCtbsY-1 の範囲の tileIdx であるリスト NumCtusInTile[tileIdx] は、以下のように導出される。

【 0 3 6 0】

20

30

40

50

【表 5 6】

```

for( j = 0; tileIdx = 0; j <= num_tile_rows_minus1; j++ )
    for( i = 0; i <= num_tile_columns_minus1; i++, tileIdx++ ) {
        for( y = RowBd[ j ]; y < RowBd[ j + 1 ]; y++ )
            for( x = ColBd[ i ]; x < ColBd[ i + 1 ]; x++ ) {
                TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x ] ] = tileIdx
                FirstCtbAddrTs[ tileIdx ] =
CtbAddrRsToTs[ RowBd[ j ] * PicWidthInCtbsY + ColBd[ i ] ]
            }
            TileIndex[ j*( num_tile_columns_minus1+1)+i ] = tileIdx
            RemTiles[tileIdx]=1
        }

for( l = 0; l < (num_tile_sets_in_pic_minus1+!remaining_tiles_tileset_flag); l++ ) {
    for( k = 0, tileIdx= top_left_tile_id[l]; k <= num_tile_rows_in_tileset_minus1[l]; k++,
tileIdx +=
        (num_tile_columns_minus1 * num_tile_columns_in_tileset_minus1[l] )
            for( i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++, tileIdx++ ){
                RemTiles[tileIdx]=0
            }
        TopLeftTileId[ l ]= top_left_tile_id[ l ];
        BottomRightTileId[ l ]= bottom_right_tile_id[ l ];
        dtlId = TileIdToIdx[ bottom_right_tile_id[ l ] ] - TileIdToIdx[ top_left_tile_id[ l ]
NumTileRowsInTileSetMinus1 [ l ] = ( dtlId / ( num_tile_columns_minus1 + 1 ) )
NumTileColumnsInTileSetMinus1 [ l ] = ( dtlId % ( num_tile_columns_minus1 + 1 ) ) }

```

【 0 3 6 1 】

40

50

【表 5 7】

```

NumTilesInSlice[1] = (NumTileRowsInTileSetMinus1[1] + 1) * (NumTileColumnsInTileSetMinus1[1] + 1)
}
if(remaining_tiles_tileset_flag) {
for(i=0, tId=(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1), brId=0; i < (num_tile_columns_minus1+1)*(num_tile_rows_minus1+1); i++) {
    if(RemTiles[i]) {
        if(i < tId) tId = i;
        if(i > brId) brId = i;
    }
}
}

```

10

```

TopLeftTileId[num_tile_sets_in_pic_minus1] = tId;
BottomRightTileId[num_tile_sets_in_pic_minus1] = brId;

NumTileRowsInTileSetMinus1[num_tile_sets_in_pic_minus1] = (brId - tId) / (num_tile_columns_minus1 + 1);
NumTileColumnsInTileSetMinus1[num_tile_sets_in_pic_minus1] = (brId - tId) % (num_tile_columns_minus1 + 1);

NumTilesInSlice[num_tile_sets_in_pic_minus1] =
    (NumTileRowsInTileSetMinus1[num_tile_sets_in_pic_minus1] + 1) * (NumTileColumnsInTileSetMinus1[num_tile_sets_in_pic_minus1] + 1)
}

```

20

【0 3 6 2】

表 14 及び表 16 及び表 16 A に関して、一実施例では、タイル走査における CTB アドレスからタイル ID への変換は、以下のとおりであり得る。

30

【0 3 6 3】

タイル走査における CTB アドレスからタイル ID への変換を指定する、両端値を含む、0 ~ PicSizeInCtbsY-1 の範囲の ctbAddrTs であるリスト TileId[ctbAddrTs]、及びタイルにおけるタイルインデックスから CTU の数への変換を指定する、両端値を含む、0 ~ PicSizeInCtbsY-1 の範囲の tileIdx であるリスト NumCtusInTile[tileIdx] は、以下のように導出される。

【0 3 6 4】

40

50

【表 5 8】

```

for(j = 0; tileIdx = 0; j <= num_tile_rows_minus1; j++)
    for(i = 0; i <= num_tile_columns_minus1; i++, tileIdx++) {
        for(y = RowBd[j]; y < RowBd[j + 1]; y++)
            for(x = ColBd[i]; x < ColBd[i + 1]; x++) {
                TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x ] ] = tileIdx
                FirstCtbAddrTs[ tileIdx ] =
                    CtbAddrRsToTs[ RowBd[j] * PicWidthInCtbsY + ColBd[i] ]
            }
        TileIndex[ j*( num_tile_columns_minus1+1)+i ] = tileIdx
        RemTiles[tileIdx]=1
    }

for(l = 0; l < num_tile_sets_in_pic_minus1; l++) {
    for(k = 0, tileIdx= top_left_tile_id[l]; k <= num_tile_rows_in_tileset_minus1[l]; k++,
    tileIdx +=
        (num_tile_columns_minus1 - num_tile_columns_in_tileset_minus1[l]))
        for(i = 0; i <= num_tile_columns_in_tileset_minus1[l]; i++, tileIdx++){
            RemTiles[tileIdx]=0
        }

    TopLeftTileId[l] = top_left_tile_id[l];
    BottomRightTileId[l] = bottom_right_tile_id[l];
    dtlId = TileIdToIdx[ bottom_right_tile_id[l] ] - TileIdToIdx[ top_left_tile_id[l] ]
    NumTileRowsInTileSetMinus1[l] = ( dtlId / ( num_tile_columns_minus1 + 1 ) )
    NumTileColumnsInTileSetMinus1[l] = ( dtlId % ( num_tile_columns_minus1 + 1 ) )
    NumTilesInSlice[l] = (NumTileRowsInTileSetMinus1[l]+1)*( NumTileColumnsInTileSetMinus1[l]+1)
}

```

10

20

30

【 0 3 6 5 】

【表 5 9】

```

for(i=0,tlId=(num_tile_columns_minus1+1)*(num_tile_rows_minus1+1),brId=0;i<(num_tile_c
olumns_minus1+1)*( num_tile_rows_minus1+1);i++){
    if(RemTiles[i]){
        if(i<tlId) tlId=i;
        if(i>brId) brId=i;
    }
}

TopLeftTileId[ num_tile_sets_in_pic_minus1 ] = tlId;
BottomRightTileId[ num_tile_sets_in_pic_minus1 ] = brId;
NumTileRowsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ] = (brId-tlId)/( num_tile_colum
ns_minus1 + 1);
NumTileColumnsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ] = (brId-tlId)/( num_tile_c
olumns_minus1 + 1);
NumTilesInSlice[ num_tile_sets_in_pic_minus1 ] =
    (NumTileRowsInTileSetMinus1[ num_tile_sets_in_pic_minus1 ]+1)*( NumTileColumnsInTil
eSetMinus1[ num_tile_sets_in_pic_minus1 ]+1)

```

40

【 0 3 6 6 】

50

一実施例では、各タイルグループが1つのタイルのみからなることを示すフラグをシグナリングすることができる。タイルグループヘッダ内のタイルグループ内のタイルの数に対するシンタックス要素のシグナリングは、このフラグ上で調整されてもよい。これにより、ビットのソースが提供される。表17は、各タイルグループが1つのタイルのみからなることを示すフラグを含むタイル構造をシグナリングするために使用され得る、ピクチャパラメータセットのシンタックスの例を示す。

【0367】

【表60】

表17

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
one_tile_per_tile_group	u(1)
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbp_trailing_bits()	
}	

【0368】

表17に関して、それぞれのシンタックス要素は、上記に提供された定義及び以下の定義に基づくことができる。

【0369】

1に等しいtransform_skip_enabled_flagは、transform_skip_flagが残差符号化シンタックスに存在し得ることを指定する。0に等しいtransform_skip_enabled_flagは、transform_skip_flagが残差符号化シンタックスに存在しないことを指定する。

【0370】

1に等しいsingle_tile_in_pic_flagは、PPSを参照して、各ピクチャ内に1つのタイルのみが存在することを指定する。0に等しいsingle_tile_in_pic_flagは、PPSを参照して、各ピクチャ内に2つ以上のタイルが存在することを指定する。

【0371】

CVS内でアクティブ化される全てのPPSについては、single_tile_in_pic_flagの値が同じであることが、ビットストリームの適合性の要件である。

【0372】

1に等しいone_tile_per_tile_groupは、各タイルグループが1つのタイルを含むことを指定する。0に等しいone_tile_per_tile_groupは、タイルグループが2つ以上のタイルを含み得ることを指定する。

【0373】

変形例では、

1に等しいone_tile_per_tile_groupは、各タイルグループが1つのタイルを含むことを指定する。0に等しいone_tile_per_tile_groupは、少なくとも1つのタイルグループが2つ以上のタイルを含むことを指定する。

【0374】

tile_column_width_minus1プラス1は、符号化ツリーブロックの単位におけるi番目のタイル列の幅を指定する。

【0375】

tile_row_height_minus1[i]プラス1は、符号化ツリーブロックの単位におけるi番目のタイル行の高さを指定する。

【0376】

表17に関して、表18は、タイルグループヘッダの例示的なシンタックスを示す。

【0377】

【表61】

表18

tile_group_header() {	記述子
tile_group_pic_parameter_set_id	ue(v)
if(NumTilesInPic > 1) {	
tile_group_address	u(v)
if(!one_tile_per_tile_group)	
num_tiles_in_tile_group_minus1	ue(v)
}	
...	
}	

【0378】

表18に関して、それぞれのシンタックス要素は、上記に提供された定義及び以下の定義に基づくことができる。

【0379】

存在する場合、タイルグループヘッダシンタックス要素tile_group_pic_parameter_set_idの値は、符号化されたピクチャのタイルグループヘッダの全てにおいて同じであるものとする。

【0380】

tile_group_pic_parameter_set_idは、使用中のPPSのpps_pic_parameter_set_idの値を指定する。tile_group_pic_parameter_set_idの値は、両端値を含む、0～63の範囲にあるものとする。

【0381】

tile_group_addressは、タイルグループ内の第1のタイルのタイルアドレスを指定する。tile_group_addressの長さはCeil(Log2(NumTilesInPic))ビットである。tile_group_addressの値は、両端値を含む、0～NumTilesInPic-1の範囲にあるものとし、tile_group_addressの値は、同じ符号化ピクチャの任意の他の符号化タイルグ

10

20

30

40

50

ループNALユニットのtile_group_addressの値と等しくないものとする。tile_group_addressが存在しない場合、これは0に等しいと推測される。

【0382】

num_tiles_in_tile_group_minus1プラス1は、タイルグループ内のタイルの数を指定する。tile_group_minus1におけるタイルの数の値は、両端値を含む、0～NumTilesInPic-1の範囲にあるものとする。num_tiles_in_tile_group_minus1の値が存在しない場合、num_tiles_in_tile_group_minus1の値は0に等しいと推測される。

【0383】

変形例では、タイルグループ(minus1)内のタイルの数のシンタックス要素num_tiles_in_tile_group_minus1は、可変長符号化(すなわち、ue(v))の代わりに、固定長符号化(すなわち、u(v))を使用してシグナリングされる。これは、システムレベルでのより容易なパースを可能にし得る。このようなnum_tiles_in_tile_group_minus1は、以下の定義に基づくことができる。

【0384】

num_tiles_in_tile_group_minus1プラス1は、タイルグループ内のタイルの数を指定する。num_tiles_in_tile_group_minus1の長さはCeil(Log2(NumTilesInPic))ビットである。num_tiles_in_tile_group_minus1の値が存在しない場合、num_tiles_in_tile_group_minus1の値は0に等しいと推測される。

【0385】

一実施例では、タイルグループ内のタイルの数のシグナリングは、NumTilesInPic由来の変数の代わりに、single_tile_in_pic_flagシンタックス要素に基づいて調整されてもよい。シグナリングのためのシンタックス要素を使用すると、シンタックス要素が含まれているか否かを決定するための追加変数の導出及び使用を必要としないことによって、タイルグループヘッダの解析を容易にする。表19は、この実施例のタイルグループヘッダの例示的なシンタックスを示す。

【0386】

【表62】

表19

tile_group_header() {	記述子
tile_group_pic_parameter_set_id	ue(v)
if(!single_tile_in_pic_flag) {	
tile_group_address	u(v)
if(!one_tile_per_tile_group)	
num_tiles_in_tile_group_minus1	ue(v)
}	
...	
}	

【0387】

表19に関して、それぞれのシンタックス要素は、上記に提供された定義に基づくことができる。

【0388】

表20は、タイルグループヘッダの別の例示的なシンタックスを示す。

【0389】

10

20

30

40

50

【表 6 3】

表20

tile_group_header() {	記述子
tile_group_pic_parameter_set_id	ue(v)
if(!single_tile_in_pic_flag) {	
tile_set_idx	ue(v)
}	
...	
}	

10

【0 3 9 0】

表20に関して、それぞれのシンタックス要素は、上記に提供された定義及び以下の定義に基づくことができる。

【0 3 9 1】

tile_set_idxは、このタイルセットのタイルセットインデックスを指定する。tile_set_idシンタックス要素の長さは $\text{Ceil}(\text{Log2}(\text{NumTileSets}))$ ビットである。

【0 3 9 2】

上述したように、ITU-T H.265は、動き制約タイルセットを可能にするシグナリングを定義し、動き制約タイルセットは、ピクチャ間予測依存性が参照ピクチャ内のコロケーションされたタイルセットに限定されるタイルセットを含んでもよい。一実施例では、タイルセットがMCTSであるかどうかを示すフラグをシグナリングすることができる。表21は、タイルセットがMCTSであるかどうか示すフラグを含むタイル構造をシグナリングするために使用され得る、ピクチャパラメータセットのシンタックスの例がシグナリングされ得ることを示す。

20

【0 3 9 3】

30

40

50

【表 6 4】

表21

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
tile_id_len_minus1	ue(v)
explicit_tile_id_flag	u(1)
if(explicit_tile_id_flag)	
for(i = 0; i <= num_tile_rows_minus1; i++)	
for(j = 0; j <= num_tile_columns_minus1; j++)	
tile_id_val[i][j]	u(v)
if(!single_tile_in_pic_flag) {	
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
tile_set_flag	u(1)
if(tile_set_flag) {	
num_tile_sets_in_pic_minus1	ue(v)
signaled_tile_set_index_flag	u(1)
remaining_tiles_tileset_flag	u(1)
for(i = 0; i <= num_tile_sets_in_pic_minus1; i++) {	
if(!remaining_tiles_tileset_flag (remaining_tiles_tileset_flag && i < num_tile_sets_in_pic_minus1) {	
top_left_tile_id[i]	u(v)
bottom_right_tile_id[i]	u(v)
}	
ts_mcts_flag	u(1)
if(signaled_tile_set_index_flag)	
tile_set_index[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbp_trailing_bits()	
}	

10

20

30

40

【0394】

表21に関して、それぞれのシンタックス要素は、以下の定義に基づくことができる。

【0395】

pps_pic_parameter_set_idは、他のシンタックス要素による参照のためのPPSを指定する。pps_pic_parameter_set_idの値は、両端値を含む、0～63の範囲にあるものとする。

【0396】

pps_seq_parameter_set_idは、アクティブSPSについてのsps_seq_parameter_set_idの値を指定する。pps_seq_parameter_set_idの値は、両端値を含む、0～1

50

5の範囲にあるものとする。

【0397】

1に等しいtransform_skip_enabled_flagは、transform_skip_flagが残差符号化シンタックスに存在し得ることを指定する。0に等しいtransform_skip_enabled_flagは、transform_skip_flagが残差符号化シンタックスに存在しないことを指定する。

【0398】

1に等しいsingle_tile_in_pic_flagは、PPSを参照して、各ピクチャ内に1つのタイルのみが存在することを指定する。0に等しいsingle_tile_in_pic_flagは、PPSを参照して、各ピクチャ内に2つ以上のタイルが存在することを指定する。

10

【0399】

CVS内でアクティブ化される全てのPPSについては、single_tile_in_pic_flagの値が同じであることが、ビットストリームの適合性の要件である。

【0400】

num_tile_columns_minus1プラス1は、ピクチャを区画するタイル列の数を指定する。num_tile_columns_minus1は、両端値を含む、0～PicWidthInCtbsY-1の範囲にあるものとする。num_tile_columns_minus1の値が存在しない場合、num_tile_columns_minus1の値は0に等しいと推測される。

【0401】

num_tile_rows_minus1プラス1は、ピクチャを区画するタイル行の数を指定する。num_tile_rows_minus1は、両端値を含む、0～PicHeightInCtbsY-1の範囲にあるものとする。num_tile_rows_minus1の値が存在しない場合、num_tile_rows_minus1の値は0に等しいと推測される。

20

【0402】

変数NumTilesInPicは、 $(\text{num_tile_columns_minus1}+1) \times (\text{num_tile_rows_minus1}+1)$ に等しく設定される。

【0403】

single_tile_in_pic_flagが0である場合、NumTilesInPicは0よりも大きいものとする。

【0404】

tile_id_len_minus1プラス1は、存在する場合、PPS内のシンタックス要素tile_id_val[i][j]、top_left_tile_id[i]、及びbottom_right_tile_id[i]を表すために使用されるビット数を指定する。tile_id_len_minus1の値は、両端値を含む、Ceil(Log2(NumTilesInPic))～15の範囲にあるものとする。

30

【0405】

1に等しいexplicit_tile_id_flagは、各タイルのタイルIDが明示的にシグナリングされることを指定する。0に等しいexplicit_tile_id_flagは、タイルIDが明示的にシグナリングされないことを指定する。

【0406】

tile_id_val[i][j]は、i番目のタイル行及びj番目のタイル列のタイルのタイルIDを指定する。tile_id_val[i][j]の長さは、tile_id_len_minus1+1ビットである。

40

【0407】

0～num_tile_column_minus1(両端値を含む)までの範囲の任意の整数m、及び0～num_tile_row_minus1(両端値を含む)までの範囲の任意の整数nについて、iがmに等しくなく、又はjがnに等しくない場合、tile_id_val[i][j]は、tile_id_val[m][n]に等しくないものである、 $j \times (\text{num_tile_columns_minus1}+1) + i$ が、 $n \times (\text{num_tile_columns_minus1}+1) + m$ 未満である場合、tile_id_val[i][j]は、tile_id_val[m][n]未満であるものとする。

【0408】

1に等しいuniform_tile_spacing_flagは、タイル列境界及び同様にタイルの行境

50

界が、ピクチャにわたって均一に分布されることを指定する。0に等しいuniform_tile_spacing_flagは、タイル列境界及び同様にタイルの行境界が、ピクチャにわたって均一に分布されないが、シンタックス要素tile_column_width_minus1[i]及びtile_row_height_minus1[i]を使用して明示的にシグナリングされることを指定する。uniform_tile_spacing_flagが存在しない場合、uniform_tile_spacing_flagの値は1に等しいと推測される。

【0 4 0 9】

tile_column_width_minus1[i]プラス1は、CTBの単位におけるi番目のタイル列の幅を指定する。

【0 4 1 0】

tile_row_height_minus1[i]プラス1は、CTBの単位におけるi番目のタイル行の高さを指定する。以下の変数は、CTBラスタ及びタイル走査変換プロセスを呼び出すことによって導出される：

- CTBの単位で、i番目のタイル列の幅を指定する、両端値を含む、0～num_tile_columns_minus1の範囲のiについてのリストColWidth[i]、

- CTBの単位で、j番目のタイル行の高さを指定する、両端値を含む、0～num_tile_rows_minus1の範囲のjについてのリストRowHeight[j]、

- CTBの単位で、i番目のタイル列境界の場所を指定する、両端値を含む、0～num_tile_columns_minus1+1の範囲のiについてのリストColBd[i]、

- CTBの単位で、j番目のタイル行境界の場所を指定する、両端値を含む、0～num_tile_rows_minus1+1の範囲のjについてのリストRowBd[j]、

ピクチャのCTBラスタ走査におけるCTBアドレスからタイル走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrRsであるリストCtbAddrRsToTs[ctbAddrRs]、

- タイル走査におけるCTBアドレスからピクチャのCTBラスタ走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrTsであるリストCtbAddrTsToRs[ctbAddrTs]、

- タイル走査におけるCTBアドレスからタイルIDへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrTsであるリストTileId[ctbAddrTs]、

- タイルにおけるタイルインデックスからCTUの数への変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のtileIdxであるリストNumCtusInTile[tileIdx]、

- タイルIDからタイルインデックスへの変換を指定するNumTilesInPic tileId値のセットについてのセット[TileIdToIdx[tileId]、及びタイル内の第1のCTBのタイル走査におけるタイルIDからCTBアドレスへの変換を指定する、両端値を含む、0～NumTilesInPic-1の範囲のtileIdxについてのリストFirstCtbAddrTs[tileIdx]

- ルマサンプルの単位で、i番目のタイル列の幅を指定する、両端値を含む、0～num_tile_columns_minus1の範囲のiについてのリストColumnWidthInLumaSamples[i]、

- ルマサンプルの単位で、j番目のタイル行の高さを指定する、両端値を含む、0～num_tile_rows_minus1の範囲のjについてのリストRowHeightInLumaSamples[j]。

【0 4 1 1】

両端値を含む、0～num_tile_columns_minus1の範囲のiについてのColumnWidthInLumaSamples[i]の値、及び両端値を含む、0～num_tile_rows_minus1の範囲のjについてのRowHeightInLumaSamples[j]は、全て0を超えるものとする。

【0 4 1 2】

0に等しいtile_set_flagは、各タイルがタイルセットであることを指定する。1に等しいtile_set_flagは、タイルセットが、シンタックス要素num_tile_sets_in_pic_minus1、top_left_tile_id[i]、及びbottom_right_tile_id[i]によって明示的に指定されることを指定する。

【0 4 1 3】

10

20

30

40

50

別の例では、0に等しいtile_set_flagは、各ピクチャが1タイルであることを指定する。

【0 4 1 4】

num_tilesets_in_pic_minus1プラス1は、ピクチャ内のタイルセットの数を指定する。num_tilesets_in_pic_minus1の値は、両端値を含む、0～(NumTilesInPic-1)の範囲にあるものとする。存在しない場合、num_tilesets_in_pic_minus1は、(NumTilesInPic-1)であると推測される。別の実施例では、存在しない場合、num_tilesets_in_pic_minus1は、0に等しいと推測される。

【0 4 1 5】

1に等しいsignaled_tile_set_index_flagは、各タイルセットのタイルセットインデックスがシグナリングされることを指定する。0に等しいsignaled_tile_set_index_flagは、タイルセットインデックスがシグナリングされないことを指定する。

10

【0 4 1 6】

1に等しいremaining_tiles_tileset_flagは、シンタックス要素top_left_tile_id[i]、num_tile_rows_in_tileset_minus1[i]、num_tile_columns_in_tileset_minus1[i]によってシグナリングされた(num_tilesets_in_pic_minus1-1)タイルセットにおいて明示的に指定されたもの以外のタイルセット内の残りのタイル全てを指定する。

【0 4 1 7】

最終のタイルセットを形成する。0に等しいremaining_tiles_tileset_flagは、num_tilesets_in_pic_minus1タイルセット全てが、シンタックス要素top_left_tile_id[i]、num_tile_rows_in_tileset_minus1[i]、num_tile_columns_in_tileset_minus1[i]をシグナリングすることによって明示的に指定されることを指定する。

20

【0 4 1 8】

top_left_tile_id[i]は、i番目のタイルセットの左上の角に配置されるタイルのタイルIDを指定する。top_left_tile_id[i]の長さは、tile_id_len_minus1+1ビットである。jに等しくない任意のiについて、top_left_tile_id[i]の値は、top_left_tile_id[j]の値に等しくないものとする。

【0 4 1 9】

bottom_right_tile_id[i]は、i番目のタイルセットの右下の角に配置されるタイルのタイルIDを指定する。bottom_right_tile_id[i]の長さは、tile_id_len_minus1+1ビットである。

30

【0 4 2 0】

変数NumTileRowsInSlice[top_left_tile_id[i]]、NumTileColumnsInSlice[top_left_tile_id[i]]、及びNumTilesInSlice[top_left_tile_id[i]]は、以下のよう

に導出される：

【0 4 2 1】

【表 6 5】

$$\text{deltaTileIdx} = \text{TileIdToIdx}[\text{bottom_right_tile_id}[i]] - \text{TileIdToIdx}[\text{top_left_tile_id}[i]]$$

$$\text{NumTileRowsInSlice}[\text{top_left_tile_id}[i]] = (\text{deltaTileIdx} / (\text{num_tile_columns_minus1} + 1)) + 1$$

40

$$\text{NumTileColumnsInSlice}[\text{top_left_tile_id}[i]] = (\text{deltaTileIdx} \% (\text{num_tile_columns_minus1} + 1)) + 1$$

$$\text{NumTilesInSlice}[\text{top_left_tile_id}[i]] = \text{NumTileRowsInSlice}[\text{top_left_tile_id}[i]] * \text{NumTileColumnsInSlice}[\text{top_left_tile_id}[i]]$$

【0 4 2 2】

1に等しいis_mcts_flagは、i番目のタイルセットが動き制約タイルセットであることを指定する。0に等しいis_mcts_flagは、i番目のタイルセットが動き制約タイルセットでないことを指定する。

50

【0423】

一実施例では、0に等しいis_mcts_flagは、i番目のタイルセットが動き制約タイルセットであってもなくてもよいことを指定する。

【0424】

i番目のタイルセットが動き制約タイルセットである場合、それは、ITU-T H.265のClause D.3.30(すなわち、時間的動き制約タイルセットSEIメッセージ)に記載されているものなどの制約のうちの1つ以上を有し得る。一実施例では、動き制約タイルセットは、時間的動き制約タイルセットと呼ばれることがある。

【0425】

tile_set_index[i]は、i番目のタイルセットのタイルセットインデックスを指定する。tile_set_index[i]シンタックス要素の長さはCeil(Log2(num_tile_sets_in_pic_minus1+1))ビットである。tile_set_index[i]の値が存在しない場合、tile_set_index[i]の値は、両端値を含む、0～num_tile_sets_in_pic_minus1の範囲の各iについて、iに等しいと推測される。

【0426】

1に等しいloop_filter_across_tiles_enabled_flagは、インループフィルタリング操作が、PPSを参照して、ピクチャ内のタイル境界を越えて実行され得ることを指定する。0に等しいloop_filter_across_tiles_enabled_flagは、インループフィルタリング操作が、PPSを参照して、ピクチャ内のタイル境界を越えて実行されないことを指定する。インループフィルタリング操作は、デブロッキングフィルタ、サンプル適応オフセットフィルタ、及び適応ループフィルタ操作を含む。loop_filter_across_tiles_enabled_flagが存在しない場合、loop_filter_across_tiles_enabled_flagの値は1に等しいと推測される。

【0427】

一実施例では、表21の対応する部分を、以下の表21Aに示すように変更してもよい。

【0428】

【表66】

表21A

tile_set_flag	u(1)
if(tile_set_flag) {	
num_tile_sets_in_pic_minus1	ue(v)
signaled_tile_set_index_flag	u(1)
remaining_tiles_tileset_flag	u(1)
for(i = 0; i <= num_tile_sets_in_pic_minus1; i++) {	
if(!remaining_tiles_tileset_flag (remaining_tiles_tileset_flag && i < num_tile_sets_in_pic_minus1)) {	
top_left_tile_id[i]	u(v)
num_tile_rows_in_tileset_minus1[i]	u(v)
num_tile_columns_in_tileset_minus1[i]	u(v)
}	
tile_set_index[i]	u(v)
}	
}	

【0429】

それぞれのシンタックス要素は、上記に提供された定義を有し得る。更に、表 21 に関して、タイルグループヘッダのシンタックスは、以下の表 21B に示すとおりであってもよい。

【 0 4 3 0 】

【表 6 7 】

表21B

tile_group_header() {	記述子
tile_group_pic_parameter_set_id	ue(v)
tile_set_idx	u(v)
tile_group_type	ue(v)
if(tile_group_type != I) {	
log2_diff_ctu_max_bt_size	ue(v)
if(sps_sbtmvp_enabled_flag) {	
sbtmvp_size_override_flag	u(1)
if(sbtmvp_size_override_flag)	
log2_sbtmvp_active_size_minus2	u(3)
}	
if(sps_temporal_mvp_enabled_flag)	
tile_group_temporal_mvp_enabled_flag	u(1)
if(tile_group_type == B)	
mvd_l1_zero_flag	u(1)
if(tile_group_temporal_mvp_enabled_flag) {	
if(tile_group_type == B)	
collocated_from_l0_flag	u(1)
}	
six_minus_max_num_merge_cand	ue(v)
}	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(NumTilesInSlice[tile_set_idx] > 1) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumTilesInSlice[tile_set_idx]-1 ; i++)	
entry_point_offset_minus1[i]	u(v)
}	
byte_alignment()	
}	

10

20

30

40

【 0 4 3 1 】

表 21B に関して、それぞれのシンタックス要素は、以下のセマンティクス及び定義に基づくことができる。

【 0 4 3 2 】

存在する場合、タイルグループヘッダシンタックス要素tile_group_pic_paramete

50

`r_set_id`の値は、符号化されたピクチャのタイルグループヘッダの全てにおいて同じであるものとする。

【0 4 3 3】

`tile_group_pic_parameter_set_id`は、使用中のPPSの`pps_pic_parameter_set_id`の値を指定する。`tile_group_pic_parameter_set_id`の値は、両端値を含む、0～63の範囲にあるものとする。

【0 4 3 4】

`tile_set_idx`は、このタイルセットのタイルセットインデックスを指定する。`tile_set_idx`シンタックス要素の長さは、 $\text{Ceil}(\text{Log2}(\text{num_tile_sets_in_pic_minus1}+1))$ ビットである。

【0 4 3 5】

別の例では、`tile_set_idx`は`tile_group_idx`と呼ばれる場合があり、以下のようなセマンティクスを有する：

`tile_group_idx`は、このタイルセットのタイルセットインデックスを指定する。`tile_group_idx`シンタックス要素の長さは、 $\text{Ceil}(\text{Log2}(\text{num_tile_sets_in_pic_minus1}+1))$ ビットである。

【0 4 3 6】

`tile_group_type`は、表21Cに従ってタイルグループの符号化タイプを指定する。

【0 4 3 7】

【表 6 8】

表 21C

<code>tile_group_type</code>	<code>tile_group_type</code> の名前
0	B(B タイルグループ)
1	P(P タイルグループ)
2	I(I タイルグループ)

【0 4 3 8】

`nal_unit_type`がIRAP_NUTに等しい場合、すなわち、ピクチャはIRAPピクチャであり、`tile_group`タイプは2に等しい。

【0 4 3 9】

`log2_diff_ctu_max_bt_size`は、バイナリ分割を使用して分割され得る符号化ブロックのルマCTBサイズと最大ルマサイズ(幅又は高さ)との間の差を指定する。`log2_diff_ctu_max_bt_size`の値は、両端値を含む、0～ $\text{CtbLog2SizeY} - \text{MinCbLog2SizeY}$ の範囲にあるものとする。

【0 4 4 0】

`log2_diff_ctu_max_bt_size`の値が存在しない場合、`log2_diff_ctu_max_bt_size`の値は2に等しいと推測される。

【0 4 4 1】

変数`MinQtLog2SizeY`、`MaxBtLog2SizeY`、`MinBtLog2SizeY`、`MaxTtLog2SizeY`、`MinTtLog2SizeY`、`MaxBtSizeY`、`MinBtSizeY`、`MaxTtSizeY`、`MinTtSizeY`、及び`MaxMttDepth`は、以下のように導出される：

【0 4 4 2】

10

20

30

40

50

【表 6 9】

MinQtLog2SizeY = (tile_group_type == I) ? MinQtLog2SizeIntraY :
MinQtLog2SizeInterY

MaxBtLog2SizeY = CtbLog2SizeY - log2_diff_ctu_max_bt_size

MinBtLog2SizeY = MinCbLog2SizeY

MaxTtLog2SizeY = (tile_group_type == I) ? 5 : 6

MinTtLog2SizeY = MinCbLog2SizeY

10

MinQtSizeY = 1 << MinQtLog2SizeY

MaxBtSizeY = 1 << MaxBtLog2SizeY

MinBtSizeY = 1 << MinBtLog2SizeY

MaxTtSizeY = 1 << MaxTtLog2SizeY

MinTtSizeY = 1 << MinTtLog2SizeY

MaxMttDepth = (tile_group_type == I) ?

max_mtt_hierarchy_depth_intra_tile_groups :

20

max_mtt_hierarchy_depth_inter_tile_groups

【0 4 4 3】

1に等しいsbtmvp_size_override_flagは、シンタックス要素log2_sbtmvp_active_size_minus2が、現在のタイルグループに存在することを指定する。0に等しいsbtmvp_size_override_flagは、シンタックス要素log2_atmvp_active_size_minus2が存在せず、log2_sbtmvp_size_active_minus2がlog2_sbtmvp_default_size_minus2に等しいと推測されることを指定する。

【0 4 4 4】

log2_sbtmvp_active_size_minus2プラス2は、現在のタイルグループのサブブロックベースのTMVPの運動パラメータを導出するために使用されるサブブロックサイズの値を指定する。log2_sbtmvp_size_active_minus2が存在しない場合、log2_sbtmvp_default_size_minus2に等しいと推測される。変数は、以下のように導出される。

30

【0 4 4 5】

【数 1 6】

Log2SbtmvpSize = log2_sbtmvp_size_active_minus2 + 2

【0 4 4 6】

tile_group_temporal_mvp_enabled_flagは、時間的運動ベクトル予測因子がインター予測のために使用され得るかどうかを指定する。tile_group_temporal_mvp_enabled_flagが0に等しい場合、現在のピクチャのシンタックス要素は、現在のピクチャの復号に時間的運動ベクトル予測因子が使用されないように制約されるものとする。そうでなければ(tile_group_temporal_mvp_enabled_flagは1に等しい)、時間的運動ベクトル予測因子は、現在のピクチャの復号に使用されてもよい。log2_sbtmvp_default_size_minus2の値が存在しない場合、log2_sbtmvp_default_size_minus2の値は0に等しいと推測される。

40

【0 4 4 7】

1に等しいmvd_l1_zero_flagは、mvd_coding(x0,y0,1)シンタックス構造がパースされず、compIdx=0~1についてMvdL1[x0][y0][compIdx]が0に等しく設定され

50

ていることを示す。0に等しいmvd_l1_zero_flagは、mvd_coding(x0,y0,1)シンタックス構造がパースされたことを示す。

【0448】

1に等しいcollocated_from_10_flagは、時間的運動ベクトル予測に使用されるコロケートされたピクチャが参照ピクチャリスト0から導出されることを指定する。0に等しいcollocated_from_10_flagは、時間的運動ベクトル予測に使用されるコロケートされたピクチャが参照ピクチャリスト1から導出されることを指定する。collocated_from_10_flagが存在しない場合、これは1に等しいと推測される。

【0449】

six_minus_max_num_merge_candは、6から減算されたタイルグループでサポートされるマージ運動ベクトル予測(MVP)候補の最大数を指定する。MVP候補をマージする最大数MaxNumMergeCandは、以下のように導出される。

【0450】

【数17】

$$\text{MaxNumMergeCand} = 6 - \text{six_minus_max_num_merge_cand}$$

【0451】

MaxNumMergeCandの値は、両端値を含む、1～6の範囲にあるものとする。

【0452】

0に等しいdep_quant_enabled_flagは、依存量子化が無効化されることを指定する。1に等しいdep_quant_enabled_flagは、依存量子化が有効化されることを指定する。

【0453】

0に等しいsign_data_hiding_enabled_flagは、サインビットハイディングが無効化されることを指定する。1に等しいsign_data_hiding_enabled_flagは、サインビットハイディングが有効化されることを指定する。sign_data_hiding_enabled_flagが存在しない場合、これは0に等しいと推測される。

【0454】

offset_len_minus1プラス1は、entry_point_offset_minus1[i]シンタックス要素の長さ(ビット)を指定する。offset_len_minus1の値は、両端値を含む、0～31の範囲にあるものとする。

【0455】

entry_point_offset_minus1[i]プラス1は、i番目のエントリポイントオフセット(バイト)を指定し、offset_len_minus1プラス1ビットで表される。スライスデータは、スライスのヘッダに続き、NumTilesInSlice[tile_set_idx]サブセットからなり、サブセットインデックス値は、両端値を含む、0～NumTilesInSlice[tile_set_idx]-1の範囲である。スライスデータの第1のバイトは、バイト0と見なされる。存在する場合、符号化スライスNALユニットのスライスデータ部分に現れるエミュレーション防止バイトは、サブセット識別の目的でスライスデータの一部としてカウントされる。サブセット0は、符号化スライスセグメントデータのバイト0からentry_point_offset_minus1[0](両端値を含む)からなり、kが1～NumTilesInSlice[tile_set_idx]-2(両端値を含む)の範囲のkを有するサブセットkは、符号化スライスデータのバイトfirstByte[k]～lastByte[k](両端値を含む)からなり、firstByte[k]及びlastByte[k]は、以下のとおり定義される。

【0456】

【数18】

$$\text{lastByte}[k] = \text{firstByte}[k] + \text{entry_point_offset_minus1}[k]$$

【0457】

10

20

30

40

50

最後のサブセット(サブセットインデックスがNumTilesInSlice[tile_set_idx]-1に等しい)は、符号化スライスデータの残りのバイトからなる。

【0458】

各サブセットは、同じタイル内にあるスライス内の全てのCTUの全ての符号化ビットからなるものとする。

【0459】

更に、表21に関して、tile_group_data()のシンタックスは、以下の表21Dに示すとおりであってもよい。

【0460】

【表70】

10

表21D

tile_group_data() {	記述子
tileIdx = TopLeftTileId[tile_set_idx]	
for(j = 0; j <= NumTileRowsInTileSetMinus1[tile_set_idx]; j++, tileIdx += (num_tile_columns_minus1 - NumTileColumnsInTileSetMinus1[tile_set_idx])) {	
for(i = 0, ; i <= NumTileColumnsInTileSetMinus1[tile_set_idx]; i++, tileIdx++) {	
ctbAddrInTs = FirstCtbAddrTs[CurrTileIdx]	
for(k = 0; k < NumCtusInTile[CurrTileIdx]; k++, ctbAddrInTs++) {	
CtbAddrInRs = CtbAddrTsToRs[ctbAddrInTs]	
coding_tree_unit()	
}	
end_of_tile_one_bit /* equal to 1 */	ae(v)
if(i < NumTileRowsInTileSetMinus1[tile_set_idx] j < NumTileColumnsInTileSetMinus1[tile_set_idx])	
byte_alignment()	
}	
}	
}	

20

30

【0461】

表21Dに関して、それぞれのシンタックス要素は、以下の定義に基づくことができる。

【0462】

end_of_tile_one_bitは1に等しいとする。

【0463】

本明細書の説明は、スライス及びスライスヘッダという用語を使用するが、これらの用語は、タイルグループ及びタイルグループヘッダの用語にそれぞれ置き換えられてもよいことに留意されたい。加えて、シンタックス要素slice_typeは、シンタックス要素tile_group_typeに置き換えられてもよい。この場合、slice_typeを使用する条件及び他のシンタックス要素をtile_group_typeに変更することができる。加えて、変数NumTilesInSliceは、シンタックス要素NumTilesInTileGroupに置き換えられてもよい。更に、本明細書に記載される実施例の説明は、タイルセットという用語を使用するが、タイルセットの1つ以上の発生はタイルグループという用語と置き換えられてもよいことに

40

50

留意されたい。更に、単語tile_setを含む名前を有するシンタックス要素のうちの1つ以上は、代わりに単語tile_groupを含む名前で置き換えられてもよい。したがって、1つ以上の以下の変更が行われてもよい。

【0464】

「タイルセット」を「タイルグループ」に変更する。

【0465】

「tile_set」を「tile_group」に変更する。

【0466】

NumTilesInSliceをNumTilesInTileGroupに変更する。

【0467】

NumTileRowsInTileSetMinus1をNumTileRowsInTileGroupMinus1に変更する。

【0468】

NumTileColumnsInTileSetMinus1をNumTileColumnsInTileGroupMinus1に変更する。

【0469】

tile_set_idxをtile_group_idxに変更する。

【0470】

num_tile_sets_in_pic_minus1をnum_tile_groups_in_pic_minus1に変更する。

【0471】

signaled_tile_set_index_flagをsignaled_tile_group_index_flagに、又はexplicit_tile_group_index_flagに変更する。

【0472】

remaining_tiles_tileset_flagをremaining_tiles_tilegroup_flagに変更する。

【0473】

tile_set_indexをtile_group_indexに変更する。

【0474】

表22A及び表22Bは、本明細書の技術によるタイル構造を信号化するために使用され得るピクチャパラメータセットのシンタックスの例を示す。

【0475】

10

20

30

40

50

【表 7 1】

表22A

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	uc(v)
pps_seq_parameter_set_id	uc(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
num_tile_columns_minus1	uc(v)
num_tile_rows_minus1	uc(v)
}	
if(!single_tile_in_pic_flag) {	
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	uc(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	uc(v)
}	
one_tile_per_tile_group_flag	
rect_tile_group_flag	u(1)
if(rect_tile_group_flag) {	
num_tile_groups_in_pic_minus1	uc(v)
signalled_tile_group_index_length_minus1	uc(v)
signalled_tile_group_index_flag	u(1)
remaining_tiles_tile_group_flag	u(1)
for(i = 0; i <= num_tile_sets_in_pic_minus1; i++) {	
if(!remaining_tiles_tile_group_flag (remaining_tiles_tile_group_flag && i < num_tile_sets_in_pic_minus1)) {	
top_left_tile_id[i]	u(v)
if(!one_tile_per_tile_group_flag)	
bottom_right_tile_id[i]	u(v)
}	
if(signalled_tile_group_index_flag)	
tile_group_index[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits()	
}	

10

20

30

40

【 0 4 7 6 】

50

【表 7 2】

表22B

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
if(!single_tile_in_pic_flag) {	
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
one_tile_per_tile_group_flag	
rect_tile_group_flag	u(1)
if(rect_tile_group_flag) {	
num_tile_groups_in_pic_minus1	ue(v)
signalled_tile_group_index_flag	u(1)
if(signalled_tile_group_index_flag)	
signalled_tile_group_index_length_minus1	ue(v)
remaining_tiles_tile_group_flag	u(1)
for(i = 0; i <= num_tile_sets_in_pic_minus1; i++) {	
if(!remaining_tiles_tile_group_flag (remaining_tiles_tile_group_flag && i < num_tile_sets_in_pic_minus1)) {	
top_left_tile_id[i]	u(v)
if(!one_tile_per_tile_group_flag)	
bottom_right_tile_id[i]	u(v)
}	
if(signalled_tile_group_index_flag)	
tile_group_index[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rsbsp_trailing_bits()	
}	

10

20

30

40

【 0 4 7 7 】

50

【表 7 3】

表22C

pic_parameter_set_rbsp() {	記述子
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
if(!single_tile_in_pic_flag) {	
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
one_tile_per_tile_group_flag	
rect_tile_group_flag	u(1)
if(rect_tile_group_flag) {	
num_tile_groups_in_pic_minus1	ue(v)
signalled_tile_group_index_flag	u(1)
if(signalled_tile_group_index_flag)	
signalled_tile_group_index_length_minus1	ue(v)
for(i = 0; i < num_tile_sets_in_pic_minus1; i++) {	
top_left_tile_id[i]	u(v)
if(!one_tile_per_tile_group_flag)	
bottom_right_tile_id[i]	u(v)
}	
if(signalled_tile_group_index_flag)	
tile_group_index[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbbsp_trailing_bits()	
}	

【0 4 7 8】

表22A、22B、及び22Cに関して、それぞれのシンタックス要素は、上記に提供されたセマンティクス及び定義並びに以下のセマンティクス及び定義に基づくことができる。

【0 4 7 9】

1に等しいone_tile_per_tile_groupは、このPPSを指す各タイルグループが1つのタイルを含むことを指定する。0に等しいone_tile_per_tile_groupは、このPPSを指すタイルグループが2つ以上のタイルを含み得ることを指定する。

【0 4 8 0】

別の実施例では、

10

20

30

40

50

1に等しいone_tile_per_tile_group_flagは、PPSを指す各タイルグループが1つのタイルをちょうど含むことを指定する。0に等しいone_tile_per_tile_group_flagは、PPSを指す各タイルグループが1つ以上のタイルを含むことを指定する。

【0481】

0に等しいrect_tile_group_flagは、タイルグループ内のタイルがラスタ走査順であり、タイルグループ情報は、PPSにおいてシグナリングされないことを指定する。1に等しいrect_tile_group_flagは、矩形タイルグループ情報が、bottom_right_tile_id[i]が存在する場合、シンタックス要素num_tile_groups_in_pic_minus1、top_left_tile_id[i]によって明示的に指定されることを指定する。

【0482】

0に等しいtile_group_flagは、各タイルがタイルグループであることを指定する。1に等しいtile_group_flagは、タイルグループが、シンタックス要素num_tile_groups_in_pic_minus1、top_left_tile_id[i]、及びbottom_right_tile_id[i]によって明示的に指定されることを指定する。

【0483】

別の実施例では、

0に等しいtile_group_flagは、各タイルがタイルグループであることを指定する。1に等しいtile_group_flagは、タイルグループが、シンタックス要素num_tile_groups_in_pic_minus1、top_left_tile_id[i]、及びbottom_right_tile_id[i]によって明示的に指定されることを指定する。

【0484】

signalled_tile_group_index_length_minus1プラス1は、存在する場合、シンタックス要素tile_group_index[i]、及びtile_group_idを表すために使用されるビット数を指定する。signalled_tile_group_index_length_minus1の値は、両端値を含む、0～15の範囲にあるものとする。

【0485】

1に等しいsignalled_tile_group_index_flagは、各タイルグループのタイルセットグループがシグナリングされることを指定する。0に等しいsignalled_tile_group_index_flagは、タイルグループインデックスがシグナリングされないことを指定する。

【0486】

num_tile_groups_in_pic_minus1プラス1は、ピクチャ内のタイルグループの数を指定する。num_tile_groups_in_pic_minus1の値は、両端値を含む、0～(NumTilesInPic-1)の範囲にあるものとする。存在しない場合、num_tile_groups_in_pic_minus1は、(NumTilesInPic-1)であると推測される。

【0487】

1に等しいremaining_tiles_tile_group_flagは、シンタックス要素top_left_tile_id[i]、num_tile_rows_in_tile_group_minus1[i]、num_tile_columns_in_tile_group_minus1

[i]によってシグナリングされた(num_tile_groups_in_pic_minus1・1)タイルセット内で明示的に指定されたもの以外のタイルセット内の残りのタイルの全てが最後のタイルセットを形成することを指定する。0に等しいremaining_tiles_tile_group_flagは、全てのnum_tile_groups_in_pic_minus1タイルセットが、シンタックス要素top_left_tile_id[i]、num_tile_rows_in_tile_group_minus1[i]、num_tile_columns_in_tile_group_minus1[i]をシグナリングすることによって明示的に指定されることを指定する。

【0488】

top_left_tile_id[i]は、i番目のタイルセットの左上の角に配置されるタイルのタイルインデックスを指定する。jに等しくない任意のiについて、top_left_tile_id[i]の値は、top_left_tile_id[j]の値に等しくないものとする。

【0489】

10

20

30

40

50

`bottom_right_tile_id[i]`は、*i*番目のタイルセットの右下の角に配置されるタイルのタイルインデックスを指定する。`one_tile_per_tile_group_flag`が1に等しい場合、`bottom_right_tile_id[i]`は、`top_left_tile_id[i]`に等しいと推測される。

【0490】

タイルグループのタイルの数を指定する変数`NumTilesInTileGroup[i]`、及び関連する変数は、以下のように導出される。

【0491】

【表74】

<code>deltaTileIdx[i] = bottom_right_tile_id[i] - top_right_tile_id[i]</code>	
<code>NumTileRowsInTileGroupMinus1[i] = (deltaTileIdx[i] / (num_tile_columns_minus1 + 1))</code>	10
<code>NumTileColumnsInTileGroupMinus1[i] =</code>	
<code>(deltaTileIdx[i] % (num_tile_columns_minus1 + 1))</code>	
<code>NumTilesInTileGroup[i] = numTileRows[i] * numTileColumns[i]</code>	

【0492】

`tile_group_index[i]`は、*i*番目のタイルグループのタイルグループインデックスを指定する。`tile_group_index[i]`シンタックス要素の長さは、`signalled_tile_set_index_length_minus1 + 1`ビットである。`tile_group_index[i]`の値が存在しない場合、`tile_group_index[i]`の値は、両端値を含む、`0 ~ num_tile_groups_in_pic_minus1`の範囲の各*i*について、*i*に等しいと推測される。

【0493】

表22A、表22B、表22C、表23A及び表23Bに関して、タイルグループヘッダの例示的なシンタックスを示し、表24は、タイルグループデータの例示的なシンタックスを示す。

【0494】

30

40

50

【表 7 5】

表23A

tile_group_header() {	記述子
tile_group_pic_parameter_set_id	ue(v)
if(!rect_tile_group_flag) {	
if(NumTilesInPic > 1) {	
tile_group_address	u(v)
if(!one_tile_per_tile_group)	
num_tiles_in_tile_group_minus1	ue(v)
}	
}	
else	
tile_group_id	u(v)
tile_group_type	ue(v)
if(tile_group_type != 1) {	
log2_diff_ctu_max_bt_size	ue(v)
if(sps_sbtmvp_enabled_flag) {	
sbtmvp_size_override_flag	u(1)
if(sbtmvp_size_override_flag)	
log2_sbtmvp_active_size_minus2	u(3)
}	
if(sps_temporal_mvp_enabled_flag)	
tile_group_temporal_mvp_enabled_flag	u(1)
if(tile_group_type == B)	
mvd_l1_zero_flag	u(1)
if(tile_group_temporal_mvp_enabled_flag) {	
if(tile_group_type == B)	
collocated_from_l0_flag	u(1)
}	
six_minus_max_num_merge_cand	ue(v)
}	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
if(rect_tile_group_flag ? (NumTilesInTileGroup[tile_group_id] > 1):	
(num_tiles_in_tile_group_minus1 > 0)) {	
offset_len_minus1	ue(v)
for(i = 0; i < rect_tile_group_flag ?	
(NumTilesInTileGroup[tile_group_id]-1): num_tiles_in_tile_group_minus1; i++)	
entry_point_offset_minus1[i]	u(v)
}	
byte_alignment()	
}	

10

20

30

40

【 0 4 9 5 】

50

【表 7 6】

表23B

tile_group_header() {	記述子
...	
if(!rect_tile_group_flag) {	
if(NumTilesInPic > 1) {	
tile_group_address	u(v)
if(!one_tile_per_tile_group)	
num_tiles_in_tile_group_minus1	ue(v)
}	
}	
else	
tile_group_id	u(v)
tile_group_pic_parameter_set_id	ue(v)
tile_group_type	ue(v)
if(tile_group_type != I) {	
...	

10

20

【 0 4 9 6】

【表 7 7】

表24

tile_group_data() {	記述子
tileIdx = TopLeftTileId[tile_group_id]tile_group_address	
for(j = 0; j <= NumTileRowsInTileGroupMinus1[tile_group_id]; j++, tileIdx += (num_tile_columns_minus1 - NumTileColumnsInTileGroupMinus1[tile_group_id])) {	
for(i = 0; i <= NumTileColumnsInTileGroupMinus1[tile_group_id]; i++, tileIdx++) {	
ctbAddrInTs = FirstCtbAddrTs[tileIdx]	
for(k = 0; k < NumCtusInTile[tileIdx]; k++, ctbAddrInTs++) {	
CtbAddrInRs = CtbAddrTsToRs[ctbAddrInTs]	
coding_tree_unit()	
}	
end_of_tile_one_bit /* equal to 1 */	ae(v)
if(i < NumTileRowsInTileGroupMinus1[tile_group_id] j < NumTileColumnsInTileGroupMinus1[tile_group_id])	
byte_alignment()	
}	
}	
}	

30

40

【 0 4 9 7】

表 23 A、表 23 B、及び表 24 に関して、それぞれのシンタックス要素は、上記に提供さ

50

れたセマンティクス及び定義並びに以下のセマンティクス及び定義に基づくことができる。

【0498】

tile_group_idは、このタイルグループのタイルグループIDを指定する。Signalled_tile_group_index_flagが1に等しい場合、tile_group_idxシンタックス要素の長さは、signalled_tile_set_index_length_minus1+1ビットである。そうでなければ、tile_group_idxの長さは、 $\text{Ceil}(\text{Log}_2(\text{num_tile_groups_in_pic_minus1}+1))$ ビットに等しい。

【0499】

別の実施例では、

tile_group_idは、このタイルグループのタイルグループIDを指定する。tile_group_idxシンタックス要素の長さは、signalled_tile_set_index_length_minus1+1ビットである。

【0500】

offset_len_minus1プラス1は、entry_point_offset_minus1[i]シンタックス要素の長さ(ビット)を指定する。offset_len_minus1の値は、両端値を含む、0～31の範囲にあるものとする。

【0501】

entry_point_offset_minus1[i]プラス1は、i番目のエントリポイントオフセット(バイト)を指定し、offset_len_minus1プラス1ビットで表される。スライスヘッダに続くスライスデータは、rect_tile_group_flag?(NumTilesInTileGroup[tile_group_id]-1):num_tiles_in_tile_group_minus1+1サブセット、サブセットインデックス値は、0～rect_tile_group_flag?(NumTilesInTileGroup[tile_group_id]-1):num_tiles_in_tile_group_minus1(両端値を含む)。スライスデータの第1のバイトは、バイト0と見なされる。存在する場合、符号化スライスNALユニットのスライスデータ部分に現れるエミュレーション防止バイトは、サブセット識別の目的でスライスデータの一部としてカウントされる。サブセット0は、符号化スライスセグメントデータのバイト0～entry_point_offset_minus1[0](両端値を含む)からなり、サブセットkは、kが1～rect_tile_group_flag?(NumTilesInTileGroup[tile_group_id]-2):(num_tiles_in_tile_group_minus1-1)(両端値を含む)の範囲であり、符号化スライスデータのバイトfirstByte[k]～lastByte[k](両端値を含む)からなり、firstByte[k]及びlastByte[k]は、以下のとおり定義される。

【0502】

【数19】

$$\text{lastByte}[k] = \text{firstByte}[k] + \text{entry_point_offset_minus1}[k]$$

【0503】

最後のサブセット(サブセットインデックスがrect_tile_group_flag?(NumTilesInSliceNumTilesInTileGroup[tile_groupset_idx]-1):num_tiles_in_tile_group_minus1に等しい)は、符号化スライスデータの残りのバイトからなる。

【0504】

表22A～24に関して、一実施例では、CTBラスタ及びタイル走査プロセスは、以下のとおりであり得る。

【0505】

CTBの単位で、i番目のタイル列の幅を指定する、両端値を含む、0～num_tile_columns_minus1の範囲のiについてのリストColWidth[i]は、以下のように導出される。

【0506】

10

20

30

40

50

【表 7 8】

```

if( uniform_tile_spacing_flag )
    for( i = 0; i <= num_tile_columns_minus1; i++ )
        ColWidth[ i ] = ( ( i + 1 ) * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 ) -
            ( i * PicWidthInCtbsY ) / ( num_tile_columns_minus1 + 1 )
    else {
        ColWidth[ num_tile_columns_minus1 ] = PicWidthInCtbsY
        for( i = 0; i < num_tile_columns_minus1; i++ ) {
            ColWidth[ i ] = tile_column_width_minus1[ i ] + 1
            ColWidth[ num_tile_columns_minus1 ] -= ColWidth[ i ]
        }
    }

```

【0 5 0 7】

CTBの単位で、j 番目のタイル行の高さを指定する、両端値を含む、0～num_tile_rows_minus1の範囲のjについてのリストRowHeight[j]は、以下のように導出される。

【0 5 0 8】

【表 7 9】

```

if( uniform_tile_spacing_flag )
    for( j = 0; j <= num_tile_rows_minus1; j++ )
        RowHeight[ j ] = ( ( j + 1 ) * PicHeightInCtbsY ) / ( num_tile_rows_minus1 + 1 ) -
            ( j * PicHeightInCtbsY ) / ( num_tile_rows_minus1 + 1 )
    else {
        RowHeight[ num_tile_rows_minus1 ] = PicHeightInCtbsY
        for( j = 0; j < num_tile_rows_minus1; j++ ) {
            RowHeight[ j ] = tile_row_height_minus1[ j ] + 1
            RowHeight[ num_tile_rows_minus1 ] -= RowHeight[ j ]
        }
    }

```

【0 5 0 9】

CTBの単位で、i 番目のタイル列境界の場所を指定する、両端値を含む、0～num_tile_columns_minus1+1の範囲のiについてのリストColBd[i]は、以下のように導出される。

【0 5 1 0】

【表 8 0】

```

for( ColBd[ 0 ] = 0, i = 0; i <= num_tile_columns_minus1; i++ )
    ColBd[ i + 1 ] = ColBd[ i ] + ColWidth[ i ]

```

【0 5 1 1】

CTBの単位で、j 番目のタイル行境界の場所を指定する、両端値を含む、0～num_tile_rows_minus1+1の範囲のjについてのリストRowBd[j]は、以下のように導出される。

【0 5 1 2】

【表 8 1】

```

for( RowBd[ 0 ] = 0, j = 0; j <= num_tile_rows_minus1; j++ )
    RowBd[ j + 1 ] = RowBd[ j ] + RowHeight[ j ]

```

【0 5 1 3】

ピクチャのCTBラスタ走査におけるCTBアドレスからタイル走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のctbAddrRsで

あるリストCtbAddrRsToTs[ctbAddrRs]は、以下のように導出される。

【0514】

【表82】

```

for( ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++) {
    tbX = ctbAddrRs % PicWidthInCtbsY
    tbY = ctbAddrRs / PicWidthInCtbsY
    for( i = 0; i <= num_tile_columns_minus1; i++)
        if( tbX >= ColBd[ i ])
            tileX = i
    for( j = 0; j <= num_tile_rows_minus1; j++)
        if( tbY >= RowBd[ j ])
            tileY = j
    CtbAddrRsToTs[ ctbAddrRs ] = 0
    for( i = 0; i < tileX; i++)
        CtbAddrRsToTs[ ctbAddrRs ] += RowHeight[ tileY ] * ColWidth[ i ]
    for( j = 0; j < tileY; j++)
        CtbAddrRsToTs[ ctbAddrRs ] += PicWidthInCtbsY * RowHeight[ j ]
    CtbAddrRsToTs[ ctbAddrRs ] +=
        ( tbY - RowBd[ tileY ] ) * ColWidth[ tileX ] + tbX - ColBd[ tileX ]
}

```

【0515】

タイル走査におけるCTBアドレスからピクチャのCTBラスト走査におけるCTBアドレスへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のCtbAddrTsであるリストCtbAddrTsToRs[CtbAddrTs]は、以下のように導出される。

【0516】

【表83】

```

for( ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++)
    CtbAddrTsToRs[ CtbAddrRsToTs[ ctbAddrRs ] ] = ctbAddrRs

```

【0517】

タイル走査におけるCTBアドレスからタイルIDへの変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のCtbAddrTsであるリストTileId[ctbAddrTs]は、以下のように導出される。

【0518】

【表84】

```

for( j = 0; tileIdx = 0; j <= num_tile_rows_minus1; j++)
    for( i = 0; i <= num_tile_columns_minus1; i++, tileIdx++)
        for( y = RowBd[ j ]; y < RowBd[ j + 1 ]; y++)
            for( x = ColBd[ i ]; x < ColBd[ i + 1 ]; x++) {
                TileId[ CtbAddrRsToTs[ y * PicWidthInCtbsY + x ] ] = tileIdx
                RemTiles[ tileIdx ] = 1
            }
}

```

【0519】

タイルにおけるタイルインデックスからCTUの数への変換を指定する、両端値を含む、0～PicSizeInCtbsY-1の範囲のtileIdxであるリストNumCtusInTile[tileIdx]は、以下のように導出される。

【0520】

【表 8 5】

```

for( j = 0, tileIdx = 0; j <= num_tile_rows_minus1; j++ )
    for( i = 0; i <= num_tile_columns_minus1; i++, tileIdx++ )
        NumCtusInTile[ tileIdx ] = ColWidth[ i ] * RowHeight[ j ]

```

【0 5 2 1】

タイルIDからタイルインデックスへの変換を指定するNumTilesInPic tileId値のセットについてのセット[TileIdToIdx[tileId]、及びタイル内の第1のCTBのタイル走査におけるタイルIDからCTBアドレスへの変換を指定する、両端値を含む、0～NumTilesInPic1の範囲のtileIdxについてのリストFirstCtbAddrTs[tileIdx]は、以下のよう

10

【0 5 2 2】

【表 8 6】

```

for( ctbAddrTs = 0, tileIdx = 0, tileStartFlag = 1; ctbAddrTs < PicSizeInCtbsY; ctbAddrTs++ ) {
    if( tileStartFlag ) {
        TileIdToIdx[ TileId[ ctbAddrTs ] ] = tileIdx
        FirstCtbAddrTs[ tileIdx ] = ctbAddrTs
        tileStartFlag = 0
    }
    tileEndFlag = ctbAddrTs == PicSizeInCtbsY - 1 || TileId[ ctbAddrTs + 1 ] !=
    TileId[ ctbAddrTs ]
    if( tileEndFlag ) {
        tileIdx++
        tileStartFlag = 1
    }
}

```

20

```

for( l = 0; l < num_tile_groups_in_pic_minus1; l++ ) {
    for( k = 0, tileIdx = top_left_tile_id[l]; k <= NumTileRowsInTileGroupMinus1[l]; k++,
    tileIdx += (num_tile_columns_minus1 - NumTileColumnsInTileGroupMinus1[l]) )
        for( i = 0; i <= numTileColumnsMinus1[l]; i++, tileIdx++ ) {
            RemTiles[tileIdx] = 0
        }
    TopLeftTileId[l] = top_left_tile_id[l];
    BottomRightTileId[l] = bottom_right_tile_id[l];
    dtlId = TileIdToIdx[ bottom_right_tile_id[l] ] - TileIdToIdx[ top_left_tile_id[l] ]
    NumTileRowsInTileGroupMinus1[l] = ( dtlId / ( num_tile_columns_minus1 + 1 ) )
    NumTileColumnsInTileGroupMinus1[l] = ( dtlId % ( num_tile_columns_minus1 + 1 ) )
    NumTilesInSlice[l] = ( NumTileRowsInTileGroupMinus1[l] + 1 ) *
    ( NumTileColumnsInTileGroupMinus1[l] + 1 )
}

for( i = 0, tId = (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1), brId = -1;
i < (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1); i++ ) {
    if( RemTiles[i] ) {
        if( i < tId ) tId = i;
        if( i > brId ) brId = i;
    }
}

```

30

40

【0 5 2 3】

50

【表 8 7】

```

TopLeftTileId[ num_tile_groups_in_pic_minus1 ]= tId;
BottomRightTileId[ num_tile_groups_in_pic_minus1 ]= brId;
NumTileRowsInTileGroupMinus1[ num_tile_groups_in_pic_minus1 ]=(brId-tId)/( num_tile_columns_mi
nus1 + 1);
NumTileColumnsInTileGroupMinus1[ num_tile_groups_in_pic_minus1 ]=(brId-tId)%( num_tile_columns
_minus1 + 1);
NumTilesInTileGroup[ num_tile_groups_in_pic_minus1 ]=
(InTileGroupNumTileRowsInTileGroupMinus1[ num_tile_groups_in_pic_minus1 ]+1)*
(NumTileColumnsInTileGroupMinus1 [ num_tile_sets_in_pic_minus1 ]+1)

```

10

【0 5 2 4】

ルマサンプルの単位で、i番目のタイル列の幅を指定する、ColumnWidthInLumaSamples[i]の値は、両端値を含む、0～num_tile_columns_minus1の範囲のiについてColWidth[i] < <

CtbLog2SizeYに等しく設定される。

【0 5 2 5】

ルマサンプルの単位で、j番目のタイル行の高さを指定する、RowHeightInLumaSamples[j]の値は、両端値を含む、0～num_tile_rows_minus1の範囲のjについてRowHeight[j] < <CtbLog2SizeYに等しく設定される。

20

【0 5 2 6】

このようにして、ソースデバイス102は、タイルセットがビットストリームで有効になっていることを示すフラグをシグナリングし、ピクチャを区画する数タイルセット列を示すシンタックス要素をシグナリングし、ピクチャを区画する数タイルセット行を示すシンタックス要素をシグナリングするように構成されたデバイスの例を表す。

【0 5 2 7】

図1を再び参照すると、インターフェース108は、データカプセル化部107によって生成されたデータを受信し、そのデータを通信媒体に送信及び/又は記憶するように構成された任意のデバイスを含んでもよい。インターフェース108は、イーサネットカードなどのネットワークインターフェースカードを含むことができ、光送受信機、無線周波数送受信機、又は情報を送信及び/若しくは受信することができる任意の他の種類のデバイスを含むことができる。更に、インターフェース108は、ファイルを記憶デバイス上に記憶することを可能にすることができるコンピュータシステムインターフェースを含むことができる。例えば、インターフェース108は、Peripheral Component Interconnect(PCI)及びPeripheral Component Interconnect Express(PCIe)バスプロトコル、独自のバスプロトコル、ユニバーサルシリアルバス(Universal Serial Bus、USB)プロトコル、I²C、又はピアデバイスを相互接続するために使用することができる任意の他の論理構造及び物理構造をサポートする、チップセットを含むことができる。

30

【0 5 2 8】

図1を再び参照すると、目的デバイス120は、インターフェース122と、データデカプセル化部123と、ビデオデコーダ124と、ディスプレイ126とを含む。インターフェース122は、通信媒体からデータ受信するように構成されている任意のデバイスを含むことができる。インターフェース122は、イーサネットカードなどのネットワークインターフェースカードを含むことができ、光送受信機、無線周波数送受信機、又は情報を受信及び/若しくは送信することができる任意の他の種類のデバイスを含むことができる。更に、インターフェース122は、準拠ビデオビットストリームを記憶デバイスから取得することを可能にするコンピュータシステム用インターフェースを含むことができる。例えば、インターフェース122は、PCI及びPCIeバスプロトコル、独自のバスプロトコル、USBプロトコル、I²C、又はピアデバイスを相互接続するために使用することができる任意の他の論理構造及び物理構造をサポートする、チップセットを含むことができる。デー

40

50

タデカプセル化部123は、本明細書に記載される例示的なパラメータセットのうちのいずれかを受信及び解析するように構成されてもよい。

【0529】

ビデオ復号器124は、ビットストリーム(例えば、MCTSサブビットストリーム抽出)及び/又はその許容可能な変形を受信し、そこからビデオデータを再生成するように構成されている任意のデバイスを含むことができる。ディスプレイ126は、ビデオデータを表示するように構成された任意のデバイスを含むことができる。ディスプレイ126は、液晶ディスプレイ(liquid crystal display、LCD)、プラズマディスプレイ、有機発光ダイオード(organic light emitting diode、OLED)ディスプレイ、又は別の種類のディスプレイなどの、様々なディスプレイデバイスのうちの1つを含むことができる。ディスプレイ126は、高解像度ディスプレイ又は超高解像度ディスプレイを含むことができる。図1に示す例では、ビデオデコーダ124は、データをディスプレイ126に出力するように説明されているが、ビデオデコーダ124は、ビデオデータを様々な種類のデバイス及び/又はそのサブコンポーネントに出力するように構成することができることに留意されたい。例えば、ビデオデコーダ124は、本明細書で説明するような任意の通信媒体にビデオデータを出力するように構成することができる。

10

【0530】

図9は、本開示の1つ以上の技術による、ビデオデータを復号するように構成することができるビデオデコーダの一例を示すブロック図である。一実施例では、ビデオデコーダ600は、変換データを復号し、復号された変換データに基づいて変換係数から残差データを復元するように構成することができる。ビデオデコーダ600は、イントラ予測復号及びインター予測復号を実行するように構成される場合があり、そのために、ハイブリッドデコーダと称される場合がある。図9に示す例では、ビデオデコーダ600は、エンтроピ復号部602、逆量子化部及び変換係数処理部604、イントラ予測処理部606、インター予測処理部608、加算部610、ポストフィルタ部612、並びに参照バッファ614を含む。ビデオデコーダ600は、ビデオ符号化システムと合致するようにしてビデオデータを復号するように構成することができる。例のビデオデコーダ600が別個の機能ブロックを有するように示されているが、そのような例示は説明のためのものであり、ビデオデコーダ600及び/又はその下位構成要素を特定のハードウェア又はソフトウェアアーキテクチャに限定するものではないことに留意されたい。ビデオデコーダ600の機能は、ハードウェア、ファームウェア、及び/又はソフトウェアの実装形態の任意の組み合わせを用いて実現することができる。

20

30

【0531】

図9に示すように、エンтроピ復号部602は、エンтроピ符号化されたビットストリームを受信する。エンтроピ復号部602は、エンтроピ符号化プロセスとは逆の(reciprocal)プロセスに従って、ビットストリームからシンタックス要素及び量子化された係数を復号するように構成することができる。エンтроピ復号部602は、上述したエンтроピ符号化技術のいずれかに従ってエンтроピ復号を実行するように構成することができる。エンтроピ復号部602は、ビデオ符号化規格と合致するようにして、符号化されたビットストリーム中のシンタックス要素の値を判定することができる。図9に示すように、エンтроピ復号部602は、ビットストリームから量子化パラメータ、量子化された係数の値、変換データ、及び予測データを判定することができる。図9に示す実施例では、逆量子化部及び変換係数処理部604は、量子化パラメータ、量子化された係数の値、変換データ、及び予測データをエンтроピ復号部602から受信し、復元された残差データを出力する。

40

【0532】

図9を再び参照すると、復元された残差データが加算部610に提供されてもよく、加算部610は、復元された残差データを予測ビデオブロックに追加し、復元されたビデオデータを生成することができる。予測ビデオブロックは、予測ビデオ技術(すなわち、イントラ予測及びフレーム間予測)に従って判定することができる。イントラ予測処理部606

50

は、イントラ予測シンタックス要素を受信し、参照バッファ614から予測ビデオブロックを取得するように構成することができる。参照バッファ614は、ビデオデータの1つ以上のフレームを記憶するように構成されたメモリデバイスを含むことができる。イントラ予測シンタックス要素は、上述したイントラ予測モードなどのイントラ予測モードを識別することができる。インター予測処理部608は、インター予測シンタックス要素を受信し、動きベクトルを生成して、参照バッファ814に記憶された1つ以上の参照フレーム内の予測ブロックを識別することができる。インター予測処理部608は、場合によっては補間フィルタに基づく補間を実行して、動き補償されたブロックを生成することができる。シンタックス要素には、画素未満の精度を有する動き予測に使用されることになる補間フィルタの識別子を含めることができる。インター予測処理部808は、補間フィルタを使用して、参照ブロックのサブ整数の画素に対する補間された値を算出することができる。ポストフィルタ部612は、復元されたビデオデータに対してフィルタリングを実行するように構成することができる。例えば、ポストフィルタ部612は、例えば、ビットストリームで規定されたパラメータに基づいて、デブロッキング及び/又はサンプル適応オフセット(SAO)フィルタリングを実行するように構成することができる。更に、いくつかの例では、ポストフィルタ部612は、独自の任意フィルタリング(例えば、モスキートノイズリダクションなどの視覚強調)を実行するように構成することができることに留意されたい。図9に示すように、復元されたビデオブロックは、ビデオデコーダ600によって出力することができる。このようにして、ビデオデコーダ600は、タイルセットがビットストリームで有効になっていることを示すフラグをパースし、ピクチャを区画する数タイルセット列を示すシンタックス要素をパースし、ピクチャを区画する数タイルセット行を示すシンタックス要素をパースし、パースされたシンタックス要素の値に基づいてビデオデータを生成するように構成され得る。

10

20

【0533】

1つ以上の例では、記載された機能は、ハードウェア、ソフトウェア、ファームウェア、又はこれらの任意の組み合わせで実装することができる。ソフトウェアで実装される場合に、この機能は、コンピュータ可読媒体上の1つ以上の命令又はコードとして記憶するか又は伝送され、ハードウェアベースの処理部によって実行することができる。コンピュータ可読媒体は、例えば、通信プロトコルに従って、ある場所から別の場所へのコンピュータプログラムの転送を容易にする任意の媒体を含む、データ記憶媒体又は通信媒体などの有形の媒体に対応する、コンピュータ可読記憶媒体を含むことができる。このようにして、コンピュータ可読媒体は、一般に、(1)非一時的な有形のコンピュータ可読記憶媒体、又は(2)信号又は搬送波などの通信媒体に対応することができる。データ記憶媒体は、本開示中に記載された技術の実現のための命令、コード、及び/又はデータ構造を取り出すために、1つ以上のコンピュータ又は1つ以上のプロセッサによってアクセスされ得る任意の利用可能な媒体であり得る。コンピュータプログラム製品は、コンピュータ可読媒体を含むことができる。

30

【0534】

一例として、非限定的に、このようなコンピュータ可読記憶媒体は、RAM、ROM、EEPROM、CD-ROM、又は他の光学ディスク記憶装置、磁気ディスク記憶装置、他の磁気記憶装置、フラッシュメモリ、又は任意の他の媒体、すなわち命令若しくはデータ構造の形式で所望のプログラムコードを記憶するために使用可能であり、かつコンピュータによりアクセス可能な任意の他の媒体を含むことができる。また、任意の接続は、コンピュータ可読媒体と適切に呼ばれる。例えば、命令がウェブサイト、サーバ、又は他のリモートソースから、同軸ケーブル、光ファイバケーブル、ツイストペア、デジタル加入者線(digital subscriber line、DSL)、あるいは赤外線、無線及びマイクロ波などの無線技術を使用して伝送される場合、同軸ケーブル、光ファイバケーブル、ツイストペア、DSL、あるいは赤外線、無線及びマイクロ波などの無線技術は、媒体の定義に含まれる。しかし、コンピュータ可読媒体及びデータ記憶媒体は、接続、搬送波、信号、又は他の一過性媒体を含まないが、代わりに非一時的な有形記憶媒体を対象としていることを理解され

40

50

たい。本発明で使用する場合、ディスク(disk)及びディスク(disc)は、コンパクトディスク(Compact Disc、CD)、レーザーディスク(laser disc)、光学ディスク(optical disc)、デジタル多用途ディスク(Digital Versatile Disc、DVD)、フロッピーディスク(floppy disk)及びブルーレイ(登録商標)ディスク(Blu-ray(登録商標)disc)を含み、ディスク(disk)は通常データを磁氣的に再生し、ディスク(disc)はレーザを用いてデータを光学的に再生する。上記の組み合わせもまた、コンピュータ可読媒体の範囲内に含まれなければならない。

【0535】

命令は、1つ以上のデジタル信号プロセッサ(DSP)、汎用マイクロプロセッサ、特定用途向け集積回路(ASIC)、フィールドプログラマブルゲート配列(FPGA)、又は他の同等の集積又はディスクリトリック回路などの1つ以上のプロセッサによって実行することができる。したがって、本明細書で使用されるとき、用語「プロセッサ」は、前記の構造、又は本明細書で説明する技術の実装に好適な任意の他の構造のいずれかを指すことができる。加えて、いくつかの態様において、本明細書に記載の機能は、符号化及び復号化するように構成された、又は複合コーデックに組み込まれた専用のハードウェアモジュール及び/又はソフトウェアモジュール内に設けられ得る。また、この技術は、1つ以上の回路又は論理素子中に完全に実装することができる。

【0536】

本開示の技術は、無線ハンドセット、集積回路(integrated circuit、IC)、又はICのセット(例えば、チップセット)を含む多種多様なデバイス又は装置に実装することができる。様々なコンポーネント、モジュール、又はユニットを本開示中に示し、開示された技術を実行するように構成されたデバイスの機能的な態様を強調しているが、異なるハードウェアユニットによって実現することは必ずしも必要ではない。むしろ、前述したように、様々なユニットは、コーデックハードウェアユニットに組み合わせてもよく、又は好適なソフトウェア及び/又はファームウェアと共に、前述の1つ以上のプロセッサを含む、相互動作ハードウェアユニットの集合によって提供することができる。

【0537】

更に、上述の各実装形態で用いた基地局装置や端末装置の各機能ブロックや各種の機能は、一般的には集積回路又は複数の集積回路である電気回路によって実現又は実行することができる。本明細書に記載の機能を実行するように設計された回路は、汎用プロセッサ、デジタル信号プロセッサ(DSP)、特定用途向け又は汎用アプリケーション集積回路(ASIC)、フィールドプログラマブルゲート配列(FPGA)若しくは他のプログラマブルロジックデバイス、ディスクリトリックゲート若しくはトランジスタロジック、若しくは個々のハードウェアコンポーネント、又はそれらの組み合わせを備えていてもよい。汎用プロセッサは、マイクロプロセッサでもよく、あるいは、プロセッサは、従来のプロセッサ、コントローラ、マイクロコントローラ、又はステートマシンでもよい。上述した汎用プロセッサ又は各回路は、デジタル回路で構成されても、又はアナログ回路で構成されてもよい。更に、半導体技術の進歩により現時点での集積回路に置き換わる集積回路化技術が現れれば、この技術による集積回路もまた使用可能となる。

【0538】

様々な実施例について説明した。これら及び他の実施例は、以下の特許請求の範囲内である。

【0539】

〈相互参照〉

WO2020/111023号公報として公開された国際出願PCT/JP2019/046062(2019年11月26日出願)に基づく国内出願である本願は、米国特許仮出願第62/774,050号(2018年11月30日出願)、第62/784,296号(2018年12月21日出願)、第62/791,227号(2019年1月11日出願)、第62/806,502号(2019年2月15日出願)に基づく優先権を主張するものであり、その内容の全体が参照により本明細書に組み込まれる。

10

20

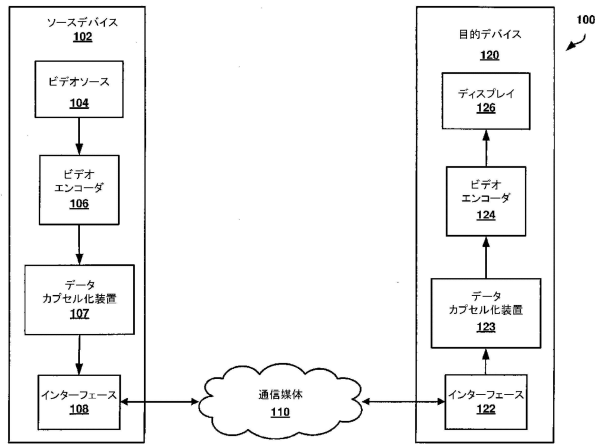
30

40

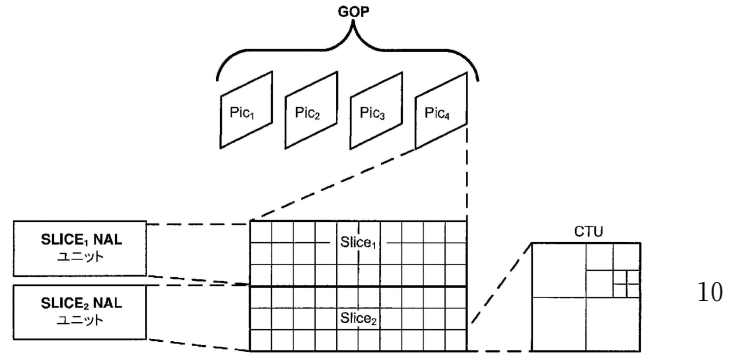
50

【図面】

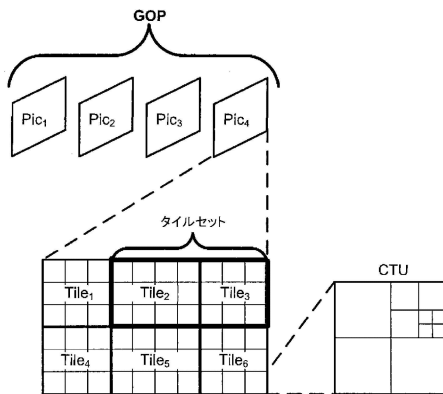
【図 1】



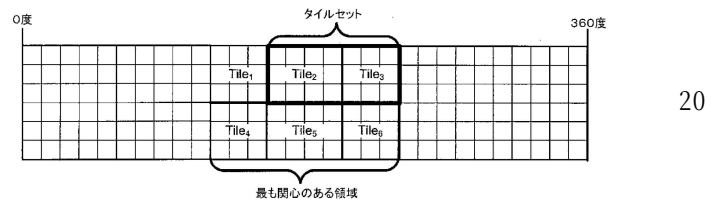
【図 2 A】



【図 2 B】



【図 2 C】



10

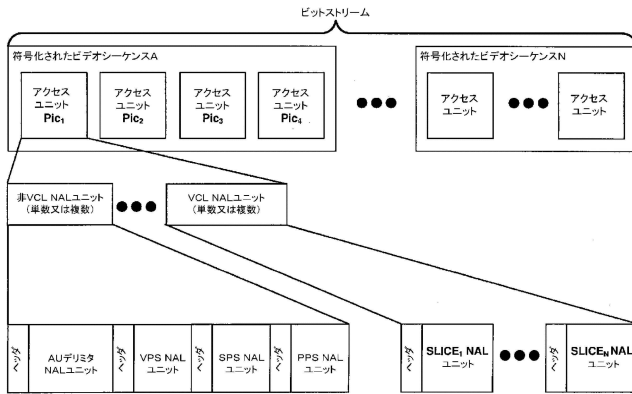
20

30

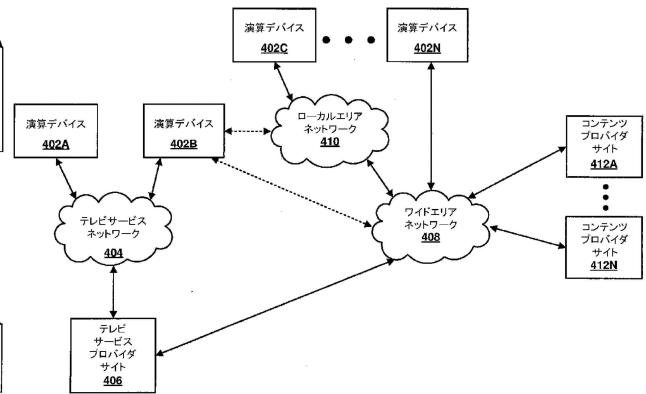
40

50

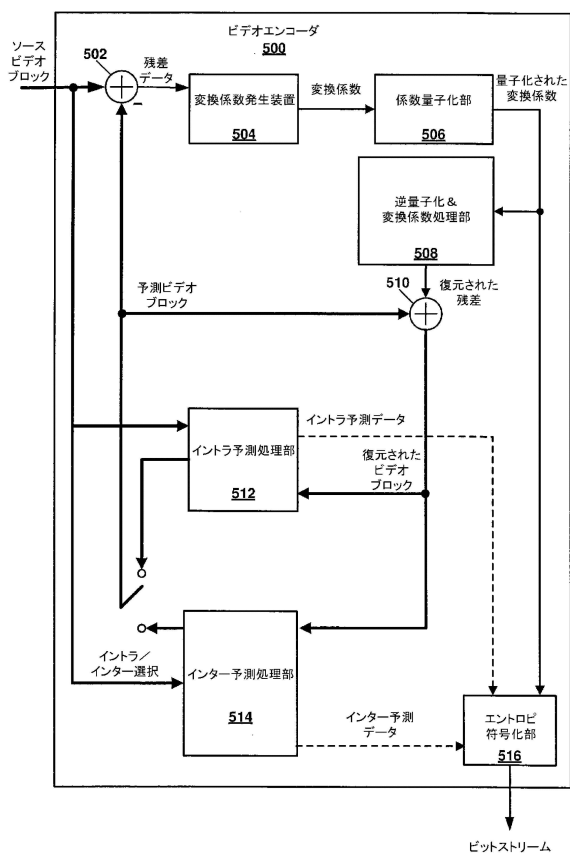
【図 3】



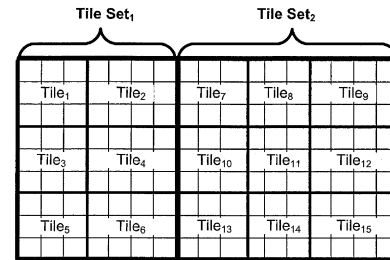
【図 4】



【図 5】



【図 6】



10

20

30

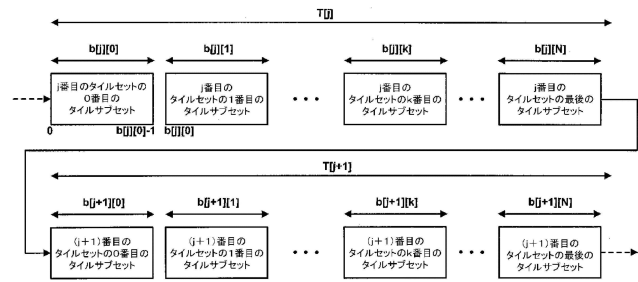
40

50

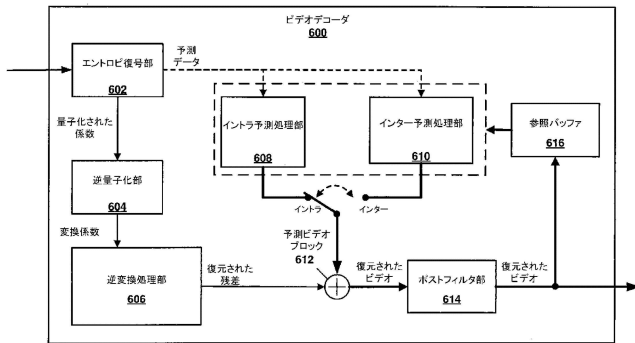
【図 7】

Tile Set ₁												Tile Set ₂											
0	1	2	9	10	11	12	63	64	65	72		81											
3	4	5																					
6	7	8	17	18	19	20				71		80											92
21	22	23	30				93			102		111											
			29				41			101		110											122
42			51				123			132		141											
			50				62			131		140											152

【図 8】



【図 9】



10

20

30

40

50

フロントページの続き

(33)優先権主張国・地域又は機関

米国(US)

(31)優先権主張番号 62/806,502

(32)優先日 平成31年2月15日(2019.2.15)

(33)優先権主張国・地域又は機関

米国(US)

(特許庁注：以下のものは登録商標)

1. 3 G P P

i v e R e g i o n o f t h e P e o p l e ' s R e p u b l i c o f C h i n a

(74)代理人 110000338

弁理士法人 H A R A K E N Z O W O R L D P A T E N T & T R A D E M A R K

(72)発明者 デシュパンダ サーチン ジー.

大阪府堺市堺区匠町1番地