

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7681615号
(P7681615)

(45)発行日 令和7年5月22日(2025.5.22)

(24)登録日 令和7年5月14日(2025.5.14)

(51)国際特許分類

F I

H 0 4 N 19/70 (2014.01)

H 0 4 N 19/70

請求項の数 17 (全42頁)

(21)出願番号	特願2022-554887(P2022-554887)	(73)特許権者	511050697
(86)(22)出願日	令和3年3月31日(2021.3.31)		アリババ グループ ホウルディング リ
(65)公表番号	特表2023-520141(P2023-520141		ミテッド
	A)		英国領ケイマン諸島 グランド ケイマン
(43)公表日	令和5年5月16日(2023.5.16)		ジョージ タウン ピーオーボックス 8
(86)国際出願番号	PCT/US2021/025092		4 7 ワン キャピタル プレイス フォー
(87)国際公開番号	WO2021/202667		ス フロア
(87)国際公開日	令和3年10月7日(2021.10.7)	(74)代理人	100079108
審査請求日	令和6年2月13日(2024.2.13)		弁理士 稲葉 良幸
(31)優先権主張番号	63/005,305	(74)代理人	100109346
(32)優先日	令和2年4月4日(2020.4.4)		弁理士 大貫 敏史
(33)優先権主張国・地域又は機関	米国(US)	(74)代理人	100117189
			弁理士 江口 昭彦
(31)優先権主張番号	63/002,594	(74)代理人	100134120
(32)優先日	令和2年3月31日(2020.3.31)		弁理士 内藤 和彦
	最終頁に続く		最終頁に続く

(54)【発明の名称】 パレット予測のための方法

(57)【特許請求の範囲】

【請求項1】

映像データ処理方法であって、
処理するために映像フレームを受信することと、
前記映像フレームの1つ又は複数のコード化単位を生成することと、
パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することと、
を含み、
前記1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、
各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定され、再利用フラグの前記固定数は、スライスタイプ及びデュアルツリーモード設定に基づいて31又は63に設定される、映像データ処理方法。

【請求項2】

前記パレット予測子が、
現パレットのすべてのパレットエントリを前記パレット予測子の第1のエントリのセットとして追加すること、及び、
前記現パレットで再利用されない前のパレット予測子からのエントリを、前記パレット予測子の第2のエントリのセットであって、前記第1のエントリのセットの後にある前記第2のエントリのセットとして追加すること

によって更新される、請求項 1 に記載の方法。

【請求項 3】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第 1 のエントリのセットとして追加すること、及び、

前のパレット予測子からのパレットエントリを、前記前のパレット予測子のパレットエントリの再利用フラグの値に関わらず、前記パレット予測子の第 2 のエントリのセットであって、前記第 1 のエントリのセットの後にある前記第 2 のエントリのセットとして追加すること

によって更新される、請求項 1 に記載の方法。

10

【請求項 4】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第 1 のエントリのセットとして追加すること、

第 1 の範囲であって、前のパレット予測子の第 1 のパレットエントリから始まり、前記前のパレット予測子の、再利用フラグセットを有する第 1 のパレットエントリで終わる前記第 1 の範囲内にある、前記前のパレット予測子の 1 つ又は複数のパレットエントリを、前記パレット予測子の第 2 の 1 つ又は複数のエントリのセットとして前記パレット予測子に追加すること、及び、

第 2 の範囲であって、前記前のパレット予測子の、再利用フラグセットを有する最後のパレットエントリから始まり、前記前のパレット予測子の最後のパレットエントリで終わる前記第 2 の範囲内にある、前記前のパレット予測子の 1 つ又は複数のパレットエントリを、前記パレット予測子の第 3 の 1 つ又は複数のエントリのセットとして前記パレット予測子に追加すること

20

によって更新され、前記第 2 のエントリのセット及び前記第 3 のエントリのセットが、前記第 1 のエントリのセットの後にある、請求項 1 に記載の方法。

【請求項 5】

前記固定数が、前記スライスタイプ及び前記デュアルツリーモード設定に基づいて 3 1 又は 6 3 に設定されることが、

前記スライスタイプが E スライスであり、前記デュアルツリーモードがオンであるとき、前記固定数を 3 1 に設定すること、あるいは

30

前記スライスタイプが B スライス／P スライスであるか、又は、前記スライスタイプが E スライスであり、前記デュアルツリーモードがオフであるとき、前記固定数を 6 3 に設定すること、

をさらに含む、請求項 1 に記載の方法。

【請求項 6】

前記 1 つ又は複数のパレット予測子のうちの 1 つのパレット予測子のサイズが、非ウェーブフロントの場合にはスライスの初めに、事前定義された値に初期化される、請求項 1 に記載の方法。

【請求項 7】

40

映像のビットストリームを記憶する方法であって、前記方法は、

映像シーケンスを受信することと、

前記映像シーケンスの 1 つ又は複数のピクチャを符号化することと、

ビットストリームを生成することと、

前記ビットストリームを非一時的コンピュータ可読記憶媒体に記憶することと、を含み、

前記符号化することは、

映像フレームの 1 つ又は複数のコード化単位を生成することと、

パレットエントリを有する 1 つ又は複数のパレット予測子を使用して、前記 1 つ又は複数のコード化単位を処理することと、を含み、

前記 1 つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグ

50

を有し、

各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定され、再利用フラグの前記固定数は、スライスタイプ及びデュアルツリーモード設定に基づいて 3 1 又は 6 3 に設定される、方法。

【請求項 8】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第 1 のエントリのセットとして追加すること、及び、

前記現パレットで再利用されない前のパレット予測子からのエントリを、前記パレット予測子の第 2 のエントリのセットであって、前記第 1 のエントリのセットの後にある前記第 2 のエントリのセットとして追加すること

によって更新される、請求項 7 に記載の方法。

【請求項 9】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第 1 のエントリのセットとして追加すること、及び、

前のパレット予測子からのパレットエントリを、前記前のパレット予測子のパレットエントリの再利用フラグの値に関わらず、前記パレット予測子の第 2 のエントリのセットであって、前記第 1 のエントリのセットの後にある前記第 2 のエントリのセットとして追加すること

によって更新される、請求項 7 に記載の方法。

【請求項 10】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第 1 のエントリのセットとして追加すること、

第 1 の範囲であって、前のパレット予測子の第 1 のパレットエントリから始まり、前記前のパレット予測子の、再利用フラグセットを有する第 1 のパレットエントリで終わる前記第 1 の範囲内にある、前記前のパレット予測子の 1 つ又は複数のパレットエントリを、前記パレット予測子の第 2 の 1 つ又は複数のエントリのセットとして前記パレット予測子に追加すること、及び、

第 2 の範囲であって、前記前のパレット予測子の、再利用フラグセットを有する最後のパレットエントリから始まり、前記前のパレット予測子の最後のパレットエントリで終わる前記第 2 の範囲内にある、前記前のパレット予測子の 1 つ又は複数のパレットエントリを、前記パレット予測子の第 3 の 1 つ又は複数のエントリのセットとして前記パレット予測子に追加すること

によって更新され、前記第 2 のエントリのセット及び前記第 3 のエントリのセットが前記第 1 のエントリのセットの後にある、請求項 7 に記載の方法。

【請求項 11】

前記固定数が、前記スライスタイプ及び前記デュアルツリーモード設定に基づいて 3 1 又は 6 3 に設定されることが、

前記スライスタイプが I スライスであり、前記デュアルツリーモードがオンであるとき、前記固定数を 3 1 に設定すること、あるいは

前記スライスタイプが B スライス／P スライスであるか、又は、前記スライスタイプが I スライスであり、前記デュアルツリーモードがオフであるとき、前記固定数を 6 3 に設定すること

をさらに含む、請求項 7 に記載の方法。

【請求項 12】

映像ストリームのための 1 つ又は複数のピクチャを出力するためにビットストリームを復号する方法であって、前記方法は、

ビットストリームを受信することと、

前記ビットストリームのコード化情報を使用して、1つ又は複数のピクチャを復号することと、

を含み、前記復号することは、

パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理すること

を含み、

前記1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、

各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定され、再利用フラグの前記固定数は、スライスタイプ及びデュアルツリーモード設定に基づいて3 1又は6 3に設定される、方法。

10

【請求項13】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第1のエントリのセットとして追加すること、及び、

前記現パレットで再利用されない前のパレット予測子からのエントリを、前記パレット予測子の第2のエントリのセットであって、前記第1のエントリのセットの後にある前記第2のエントリのセットとして追加すること

によって更新される、請求項12に記載の方法。

20

【請求項14】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第1のエントリのセットとして追加すること、及び、

前のパレット予測子からのパレットエントリを、前記前のパレット予測子のパレットエントリの再利用フラグの値に関わらず、前記パレット予測子の第2のエントリのセットであって、前記第1のエントリのセットの後にある前記第2のエントリのセットとして追加すること

によって更新される、請求項12に記載の方法。

【請求項15】

前記パレット予測子が、

現パレットのすべてのパレットエントリを前記パレット予測子の第1のエントリのセットとして追加すること、

第1の範囲であって、前のパレット予測子の第1のパレットエントリから始まり、前記前のパレット予測子の、再利用フラグセットを有する第1のパレットエントリで終わる前記第1の範囲内にある、前記前のパレット予測子の1つ又は複数のパレットエントリを、前記パレット予測子の第2の1つ又は複数のエントリのセットとして前記パレット予測子に追加すること、及び、

30

第2の範囲であって、前記前のパレット予測子の、再利用フラグセットを有する最後のパレットエントリから始まり、前記前のパレット予測子の最後のパレットエントリで終わる前記第2の範囲内にある、前記前のパレット予測子の1つ又は複数のパレットエントリを、前記パレット予測子の第3の1つ又は複数のエントリのセットとして前記パレット予測子に追加すること

40

によって更新され、前記第2のエントリのセット及び前記第3のエントリのセットが、前記第1のエントリのセットの後にある、請求項12に記載の方法。

【請求項16】

前記固定数が、前記スライスタイプ及び前記デュアルツリーモード設定に基づいて3 1又は6 3に設定されることが、

前記スライスタイプがIスライスであり、前記デュアルツリーモードがオンであるとき、前記固定数を3 1に設定すること、あるいは

前記スライスタイプがBスライス／Pスライスであるか、又は、前記スライスタイプが

50

スライスであり、前記デュアルツリーモードがオフであるとき、前記固定数を 63 に設定すること

をさらに含む、請求項 12 に記載の方法。

【請求項 17】

前記 1 つ又は複数のパレット予測子のうちの 1 つのパレット予測子のサイズが、非ウェーブフロントの場合にはスライスの初めに、事前定義された値に初期化される、請求項 12 に記載の方法。

【発明の詳細な説明】

【技術分野】

【0001】

関連出願の相互参照

【0001】 本開示は、2020年4月4日に出願された米国仮特許出願第63／005，305号、及び2020年3月31日に出願された米国仮特許出願第63／002，594号の優先権の利益を主張するものであり、両方の仮出願は、その全体が参照により本明細書に組み込まれる。

【0002】

技術分野

【0002】 本開示は、概して、映像処理に関し、より詳細には、映像符号化及び復号化におけるパレットモードの使用に関する。

【背景技術】

【0003】

背景

【0003】 映像は、視覚情報を捕捉する静止ピクチャ（又は「フレーム」）のセットである。記憶メモリ及び伝送帯域幅を減らすために、映像は、記憶又は伝送の前に圧縮し、表示前に解凍することができる。圧縮プロセスは通常、符号化と呼ばれ、解凍プロセスは通常、復号化と呼ばれる。規格化された映像コード化技術、最も一般的には、予測、変換、量子化、エントロピーコード化及びループ内フィルタリングに基づく技術を使用する様々な映像コード化形式がある。特定の映像コード化形式を指定するHEVC（High Efficiency Video Coding）（HEVC／H.265）規格、VVC（Versatile Video Coding）（VVC／H.266）規格、及びAVS規格などの映像コード化規格が、規格化組織によって策定されている。一層進化した映像コード化技術が映像規格に採用されるにつれて、新たな映像コード化規格のコード化効率が一層高くなっている。

【発明の概要】

【課題を解決するための手段】

【0004】

開示の概要

【0004】 本開示の実施形態は、パレット予測子のためのコンピュータ実装方法を提供する。いくつかの実施形態では、方法は、処理するために映像フレームを受信することと、映像フレームの1つ又は複数のコード化単位を生成することと、パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することと、を含み、1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、各パレット予測子の再利用フラグの数は、対応するコード化単位に対して固定数に設定される。

【0005】

【0005】 本開示の実施形態は、装置を提供する。いくつかの実施形態では、装置は、命令を記憶するように構成されたメモリと、メモリに結合されたプロセッサとを含み、プロセッサは、命令を実行して、装置に、処理するために映像フレームを受信すること、映像フレームの1つ又は複数のコード化単位を生成すること、及び、パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することを実行させるように構成され、1つ又は複数のパレット予測子の各パレットエントリが

10

20

30

40

50

、対応する再利用フラグを有し、各パレット予測子の再利用フラグの数は、対応するコード化単位に対して固定数に設定される。

【0006】

【0006】 本開示の実施形態は、命令のセットを記憶する非一時的コンピュータ可読記憶媒体を提供し、命令のセットは、装置に映像データ処理を実行するための方法を開始させるために、装置の1つ又は複数のプロセッサによって実行可能である。いくつかの実施形態では、方法は、処理するために映像フレームを受信することと、映像フレームの1つ又は複数のコード化単位を生成することと、パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することと、を含み、1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、各パレット予測子の再利用フラグの数は、対応するコード化単位に対して固定数に設定される。

10

【0007】

【0007】 本開示の実施形態は、パレットモードのフィルタをデブロッキングするためのコンピュータ実装方法を提供する。いくつかの実施形態では、方法は、処理するために映像フレームを受信することと、映像フレームの1つ又は複数のコード化単位であって、それぞれのコード化単位が1つ又は複数のコード化ブロックを有する1つ又は複数のコード化単位を生成することと、隣接する2つのコード化ブロックのうちの少なくとも第1のコード化ブロックがパレットモードでコード化され、隣接する2つのコード化ブロックのうちの第2のコード化ブロックがパレットモードとは異なるコード化モードを有することに応答して、境界フィルタ強度を1に設定することと、を含む。

20

【0008】

【0008】 本開示の実施形態は、装置を提供する。いくつかの実施形態では、装置は、命令を記憶するように構成されたメモリと、メモリに結合されたプロセッサとを含み、プロセッサは、命令を実行して、装置に、処理するために映像フレームを受信することと、映像フレームの1つ又は複数のコード化単位であって、それぞれのコード化単位が1つ又は複数のコード化ブロックを有する1つ又は複数のコード化単位を生成することと、及び、隣接する2つのコード化ブロックのうちの少なくとも第1のコード化ブロックがパレットモードでコード化され、隣接する2つのコード化ブロックのうちの第2のコード化ブロックがパレットモードとは異なるコード化モードを有することに応答して、境界フィルタ強度を1に設定することを実行させるように構成される。

30

【0009】

【0009】 本開示の実施形態は、命令のセットを記憶する非一時的コンピュータ可読記憶媒体を提供し、命令のセットは、装置に映像データ処理を実行するための方法を開始させるために、装置の1つ又は複数のプロセッサによって実行可能である。いくつかの実施形態では、方法は、処理するために映像フレームを受信することと、処理するために映像フレームを受信することと、映像フレームの1つ又は複数のコード化単位であって、それぞれのコード化単位が1つ又は複数のコード化ブロックを有する1つ又は複数のコード化単位を生成することと、隣接する2つのコード化ブロックのうちの少なくとも第1のコード化ブロックがパレットモードでコード化され、隣接する2つのコード化ブロックのうちの第2のコード化ブロックがパレットモードとは異なるコード化モードを有することに応答して、境界フィルタ強度を1に設定することと、を含む。

40

【0010】

図面の簡単な説明

【0010】 本開示の実施形態及び様々な態様を以下の詳細な説明及び添付の図面に示す。図に示されている様々な特徴は、縮尺通りに描かれていない。

【図面の簡単な説明】

【0011】

【図1】 【0011】 本開示のいくつかの実施形態による映像シーケンスの一例の構造を図示する概略図である。

【図2A】 【0012】 本開示の実施形態と合致するハイブリッドビデオコード化システムの例

50

示的な符号化プロセスを図示する概略図である。

【図 2 B】[0013]本開示の実施形態と合致するハイブリッドビデオコード化システムの別の例示的な符号化プロセスを図示する概略図である。

【図 3 A】[0014]本開示の実施形態と合致するハイブリッドビデオコード化システムの例示的な復号化プロセスを図示する概略図である。

【図 3 B】[0015]本開示の実施形態と合致するハイブリッドビデオコード化システムの別の例示的な復号化プロセスを図示する概略図である。

【図 4】[0016]本開示のいくつかの実施形態による、映像を符号化又は復号化するための例示的な装置のブロック図である。

【図 5】[0017]本開示のいくつかの実施形態による、パレットモードでコード化されたブロックを図示したものである。

【図 6】[0018]パレット予測子更新プロセスの一例を示す。

【図 7】[0019]パレットコード化シンタックスの一例を示す。

【図 8】[0020]本開示のいくつかの実施形態による、パレット予測子更新プロセスのフローチャートを示す。

【図 9】[0021]本開示のいくつかの実施形態による、パレット予測子更新プロセスの一例を示す。

【図 10】[0022]パレットモードの復号化プロセスの一例を示す。

【図 11】[0023]本開示のいくつかの実施形態による、パレットモードの復号化プロセスの一例を示す。

【図 12】[0024]本開示のいくつかの実施形態による、別のパレット予測子更新プロセスのフローチャートを示す。

【図 13】[0025]本開示のいくつかの実施形態による、別のパレット予測子更新プロセスの一例を示す。

【図 14】[0026]本開示のいくつかの実施形態による、パレットコード化シンタックスの一例を示す。

【図 15】[0027]本開示のいくつかの実施形態による、パレットモードの復号化プロセスの一例を示す。

【図 16】[0028]本開示のいくつかの実施形態による、パレット予測子更新プロセスの一部分を実装するパレットモードのための復号器のハードウェア設計を図示したものである。

【図 17】[0029]本開示のいくつかの実施形態による、パレットモードの復号化プロセスの一例を示す。

【図 18】[0030]パレットモードの初期化プロセスの一例を示す。

【図 19】[0031]本開示のいくつかの実施形態による、パレットモードの初期化プロセスの一例を示す。

【図 20】[0032]パレット予測子更新、及び対応する再利用フラグのランレングスコード化の一例を示す。

【図 21】[0033]本開示のいくつかの実施形態による、パレット予測子更新、及び対応する再利用フラグのランレングスコード化の一例を示す。

【図 22】[0034]本開示のいくつかの実施形態による、パレット予測子更新、及び対応する再利用フラグのランレングスコード化の一例を示す。

【図 23】[0035]本開示のいくつかの実施形態による、パレットコード化シンタックスの一例を示す。

【図 24】[0036]パレットコード化セマンティックスの一例を示す。

【図 25】[0037]本開示のいくつかの実施形態による、パレットコード化セマンティックスの一例を示す。

【発明を実施するための形態】

【0012】

詳細な説明

【0038】 ここで、例示的な実施形態について詳細に言及するが、それらの例が添付の図

10

20

30

40

50

面に図示されている。以下の説明は添付の図面を参照し、添付の図面では、別段の記載がない限り、異なる図面の同じ番号が同じ又は類似の要素を表す。例示的な実施形態についての以下の説明に記載される実装形態は、本発明と合致するすべての実装形態を表すわけではない。むしろ、それらは、添付の特許請求の範囲で列挙する本発明に関係する態様と合致する装置及び方法の例に過ぎない。下記に、本開示の特定の態様をより詳細に説明する。本明細書に記載されている用語及び定義が、参照により組み込まれている用語及び／又は定義と矛盾する場合には、本明細書の記載が優先される。

【0013】

【0039】 ITU-T Video Coding Expert Group (ITU-T VCEG) 及びISO/IEC Moving Picture Expert Group (ISO/IEC MPEG) のJoint Video Experts Team (JVET) は、VVC (Versatile Video Coding) (VVC/H. 266) 規格を現在開発している。VVC規格は、その前身の規格であるHEVC (High Efficiency Video Coding) (HEVC/H. 265) 規格の圧縮効率を2倍にすることを目指している。言い換えれば、VVCの目標は、半分の帯域幅を使用してHEVC/H. 265と同じ主観的品質を実現することである。

10

【0014】

【0040】 半分の帯域幅を使用してHEVC/H. 265と同じ主観的品質を実現するために、JVETは、ジョイントエクスプロレーションモデル(JEM) 参照ソフトウェアを使用してHEVCを超える技術を開発してきた。コード化技術がJEMに組み込まれたため、JEMは、HEVCよりも大幅に向上したコード化性能を実現した。

20

【0015】

【0041】 VVC規格は、最近開発されてきており、より優れた圧縮性能を提供するより多くのコード化技術を含み続けている。VVCは、HEVC、H. 264/AVC、MPEG2、H. 263、等々といったような最新の映像圧縮規格で使用されてきたものと同じハイブリッド映像コード化システムに基づいている。

【0016】

【0042】 映像とは、視覚情報を記憶するために時系列順に配列された静止ピクチャ（又は「フレーム」）のセットである。それらのピクチャを時系列順に捕捉し、記憶するために、映像捕捉デバイス（例えば、カメラ）を使用することができ、このようなピクチャを時系列順に表示するために、映像再生デバイス（例えば、テレビ、コンピュータ、スマートフォン、タブレットコンピュータ、ビデオプレーヤ、又は表示機能を有する任意のエンドユーザ端末）を使用することができる。また、いくつかの用途では、監視、会議、又は生放送などのために、映像捕捉デバイスは、捕捉した映像を映像再生デバイス（例えば、モニタを有するコンピュータ）にリアルタイムで伝送することができる。

30

【0017】

【0043】 このような用途で必要とされる記憶空間及び伝送帯域幅を減らすために、映像は、記憶及び伝送の前に圧縮することができ、表示の前に解凍することができる。この圧縮及び解凍は、プロセッサ（例えば、汎用コンピュータのプロセッサ）又は専用ハードウェアによって実行されるソフトウェアによって実装することができる。圧縮のためのモジュールは一般に「符号器」と呼ばれ、解凍のためのモジュールは一般に「復号器」と呼ばれる。符号器及び復号器は、総称して「コーデック」と呼ぶことができる。符号器及び復号器は、様々な適切なハードウェア、ソフトウェア、又はそれらの組み合わせのいずれかとして実装することができる。例えば、符号器及び復号器のハードウェア実装形態としては、1つ又は複数のマイクロプロセッサ、デジタル信号プロセッサ(DSP)、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アレイ(FPGA)、ディスクリットロジック、又はそれらの任意の組み合わせなどの回路類を挙げることができる。符号器及び復号器のソフトウェア実装形態としては、プログラムコード、コンピュータ実行可能命令、ファームウェア、又はコンピュータ可読媒体に固定された任意の適切なコンピュータ実装アルゴリズム若しくはプロセスを挙げることができる。映像の圧縮及び解凍は、MPEG-1、MPEG-2、MPEG-4、H. 26xシリーズなどといったよ

40

50

うな様々なアルゴリズム、又は規格によって実装することができる。いくつかの用途では、コーデックが第1のコード化規格から映像を解凍し、第2のコード化規格を使用して、解凍された映像を再圧縮することができるが、この場合、コーデックは「トランスコーダ」と呼ぶことができる。

【0018】

【0044】 映像符号化プロセスは、ピクチャを再構築するために使用可能な有用な情報を識別し、保持することができ、再構築に重要でない情報を無視することができる。無視された重要でない情報を完全に再構築できない場合、このような符号化プロセスは、「不可逆」と呼ぶことができる。そうでない場合、符号化プロセスは、「可逆」と呼ぶことができる。ほとんどの符号化プロセスは不可逆であり、これは、必要な記憶空間及び伝送帯域幅を減らすためのトレードオフである。

10

【0019】

【0045】 符号化されているピクチャ（「現ピクチャ」と呼ぶ）の有用な情報は、参照ピクチャ（例えば、前に符号化され、再構築されたピクチャ）に対する変化を含む。このような変化は、ピクセルの位置変化、光度変化、又は色変化を含むことができ、そのうちの位置変化が最も重要である。オブジェクトを表すピクセル群の位置変化は、参照ピクチャと現ピクチャとの間のオブジェクトの動きを反映し得る。

【0020】

【0046】 別のピクチャを参照することなくコード化されるピクチャ（すなわち、ピクチャが自らの参照ピクチャである）は「Iピクチャ」と呼ばれる。ピクチャ内の一部又はすべてのブロック（例えば、映像ピクチャの部分を一般に指すブロック）が、1つの参照ピクチャを用いたイントラ予測又はインター予測を使用して予測される場合（例えば、片方向予測）、ピクチャは「Pピクチャ」と呼ばれる。ピクチャ内の少なくとも1つのブロックが、2つの参照ピクチャを用いて予測される場合（例えば、双方向予測）、ピクチャは「Bピクチャ」と呼ばれる。

20

【0021】

【0047】 図1は、本開示のいくつかの実施形態による映像シーケンス100の一例の構造を示す。映像シーケンス100は、ライブ映像又は捕捉されアーカイブされている映像とすることができる。映像100は、現実の映像、コンピュータによって生成される映像（例えば、コンピュータゲーム映像）、又はそれらの組み合わせ（例えば、拡張現実効果を有する現実の映像）とすることができる。映像シーケンス100は、映像捕捉デバイス（例えば、カメラ）、前に捕捉された映像を含む映像アーカイブ（例えば、記憶デバイス内に記憶されている映像ファイル）、又は映像コンテンツプロバイダから映像を受信するための映像供給インタフェース（例えば、映像ブロードキャストトランシーバ）から入力することができる。

30

【0022】

【0048】 図1に示されているように、映像シーケンス100は、ピクチャ102、104、106、及び108を含む、タイムラインに沿って時間的に配列された一連のピクチャを含むことができる。ピクチャ102～106は、連続的であり、ピクチャ106とピクチャ108との間にさらに多くのピクチャがある。図1では、ピクチャ102は、Iピクチャであり、その参照ピクチャは、ピクチャ102自身である。ピクチャ104は、Pピクチャであり、矢印によって示すように、その参照ピクチャはピクチャ102である。ピクチャ106はBピクチャであり、矢印によって示すように、その参照ピクチャはピクチャ104及び108である。いくつかの実施形態では、ピクチャ（例えば、ピクチャ104）の参照ピクチャは、そのピクチャの直前又は直後になくてもよい。例えば、ピクチャ104の参照ピクチャは、ピクチャ102に先行するピクチャとすることができる。ピクチャ102～106の参照ピクチャは例に過ぎず、本開示は参照ピクチャの実施形態を図1に示す例として限定しないことに留意されたい。

40

【0023】

【0049】 典型的には、映像コーデックは、ピクチャ全体を一度に符号化又は復号化しな

50

いが、それは、このようなタスクの計算が複雑なためである。もっと正確に言えば、映像コーデックは、ピクチャを基本セグメントに分け、ピクチャをセグメントごとに符号化又は復号化することができる。このような基本セグメントは、本開示では基本処理単位（「BPU: Basic Processing Unit」）と呼ばれる。例えば、図1の構造110は、映像シーケンス100のピクチャ（例えば、ピクチャ102～108のうちのいずれか）の構造の一例を示す。構造110では、ピクチャが4×4の基本処理単位に分割されており、その境界が破線で示されている。いくつかの実施形態では、基本処理単位は、いくつかの映像コード化規格（例えば、MPEGファミリー、H. 261、H. 263、又はH. 264／AVC）では「マクロブロック」と呼ぶことができ、又は他のいくつかの映像コード化規格（例えば、H. 265／HEVC又はH. 266／VVC）では「コード化ツリー単位」

10

【0024】

[0050] 基本処理単位は、コンピュータメモリ内（例えば、映像フレームバッファ内）に記憶される様々なタイプの映像データ群を含み得る論理単位とすることができる。例えば、カラーピクチャの基本処理単位は、無彩色の輝度情報を表すルマ成分（Y）、色情報を表す1つ又は複数のクロマ成分（例えば、Cb及びCr）、並びにルマ成分及びクロマ成分が同じサイズの基本処理単位を有し得る関連シンタックス要素を含むことができる。いくつかの映像コード化規格（例えば、H. 265／HEVC又はH. 266／VVC）では、ルマ成分及びクロマ成分を「コード化ツリーブロック」（「CTB: Coding Tree Block」）と呼ぶことができる。基本処理単位に対して実行されるどの操作も、そのルマ成分及びクロマ成分のそれぞれに対して繰り返し実行することができる。

20

【0025】

[0051] 映像のコード化は、複数の操作段階を有するが、それらの例が図2A～図2B及び図3A～図3Bに示されている。それぞれの段階ごとでは、基本処理単位のサイズがまだ大き過ぎて処理できない可能性があるため、本開示では「基本処理副単位」と呼ばれるセグメントにさらに分割することができる。いくつかの実施形態では、基本処理副単位は、いくつかの映像コード化規格（例えば、MPEGファミリー、H. 261、H. 263、又はH. 264／AVC）では「ブロック（block）」と呼ばれる場合もあれば、いくつかの他の映像コード化規格（例えば、H. 265／HEVC又はH. 266／VVC）では「コード化単位」（「CU: Coding Unit」）と呼ばれる場合もある。基本処理副単位は、基本処理単位と同じサイズ又はそれよりも小さいサイズを有することができる。基本処理単位と同様に、基本処理副単位もまた論理単位であり、コンピュータメモリ内（例えば、映像フレームバッファ内）に記憶される様々なタイプの映像データの群（例えば、Y、Cb、Cr及び関連シンタックス要素）を含むことができる。基本処理副単位に対して実行されるどの操作も、そのルマ成分及びクロマ成分のそれぞれに対して繰り返し実行することができる。処理上の必要性に応じて、このような分割は、さらなるレベルに対して実行可能であることに留意されたい。様々な段階が様々な方式を使用して基本処理単位を分割できることにまた留意されたい。

30

40

【0026】

[0052] 例えば、モード判定段階（その一例を図2Bに示す）において、符号器は、基本処理単位に対してどの予測モード（例えば、イントラピクチャ予測又はインターピクチャ予測）を使用するのかを判定することができるが、このような判定を下すには基本処理単位が大き過ぎる場合がある。符号器は、基本処理単位を複数の基本処理副単位（例えば、H. 265／HEVC又はH. 266／VVCにおけるようなCU）に分け、それぞれの個々の基本処理副単位に対して予測のタイプを判定することができる。

【0027】

50

【0053】 別の例では、予測段階（その一例を図2A～図2Bに示す）において、符号器は、基本処理副単位（例えば、CU）のレベルで予測操作を実行することができる。しかしながら、場合によっては、基本処理副単位がまだ依然として大き過ぎて処理できない可能性がある。符号器は、基本処理副単位を（例えば、H. 265/HEVC又はH. 266/VVCで「予測ブロック」又は「PB」と呼ばれる）より小さいセグメントにさらに分けることができ、そのレベルで予測操作を実行することができる。

【0028】

【0054】 別の例では、変換段階（その一例を図2A～図2Bに示す）において、符号器は、残差基本処理副単位（例えば、CU）に対する変換操作を実行することができる。しかしながら、場合によっては、基本処理副単位が依然として大き過ぎて処理できない可能性がある。符号器は、基本処理副単位を（例えば、H. 265/HEVC又はH. 266/VVCで「変換ブロック」又は「TB」と呼ばれる）より小さいセグメントにさらに分けることができ、そのレベルで変換操作を実行することができる。同じ基本処理副単位の分割方式は、予測段階と変換段階とで異なり得ることに留意されたい。例えば、H. 265/HEVC又はH. 266/VVCでは、同じCUの予測ブロック及び変換ブロックは、異なるサイズ及び数を有する場合がある。

【0029】

【0055】 図1の構造110では、基本処理単位112が3×3の基本処理副単位にさらに分割されており、その境界が点線で示されている。同じピクチャの異なる基本処理単位を異なる方式で基本処理副単位に分割することができる。

【0030】

【0056】 いくつかの実装形態では、映像の符号化及び復号化に並列処理及び誤り耐性の能力を提供するために、ピクチャを処理のための領域に分割することができ、これにより、符号化又は復号化プロセスが、ピクチャのある領域に対して、ピクチャの他のどの領域の情報にも依存しないようにすることができる。言い換えれば、ピクチャの各領域を独立して処理することができる。そうすることで、コーデックは、ピクチャの様々な領域を並列に処理することができ、コード化効率を高めることができる。また、領域のデータが処理中に破損するか又はネットワーク伝送中に失われると、コーデックは、破損したデータ又は失われたデータに依存することなく、同じピクチャの他の領域を正しく符号化又は復号化することができ、誤り耐性の機能を提供することができる。いくつかの映像コード化規格では、ピクチャを様々なタイプの領域に分割することができる。例えば、H. 265/HEVC及びH. 266/VVCは、2つのタイプの領域、すなわち、「スライス」及び「タイル」を提供する。映像シーケンス100の様々なピクチャは、ピクチャを領域に分割するための様々な区画方式を有し得ることにまた留意されたい。

【0031】

【0057】 例えば、図1では、構造110が3つの領域114、116、及び118に分割されており、その境界が構造110内の実線で示されている。領域114は、4つの基本処理単位を含む。領域116及び118はそれぞれ、6つの基本処理単位を含む。図1の構造110の基本処理単位、基本処理副単位、及び領域は例に過ぎず、本開示はその実施形態を限定しないことに留意されたい。

【0032】

【0058】 図2Aは、本開示の実施形態と合致する符号化プロセス200Aの一例の概略図を示す。例えば、符号化プロセス200Aは、符号器によって実行することができる。図2Aに示されているように、符号器は、プロセス200Aに従って映像シーケンス202を映像ビットストリーム228内に符号化することができる。図1の映像シーケンス100と同様に、映像シーケンス202は、時系列順に配列されたピクチャのセット（「元のピクチャ」と呼ばれる）を含むことができる。図1の構造110と同様に、映像シーケンス202のそれぞれの元のピクチャは、符号器によって基本処理単位、基本処理副単位、又は処理のための領域に分割することができる。いくつかの実装形態では、符号器は、映像シーケンス202のそれぞれの元のピクチャごとの基本処理単位のレベルで、プロセ

ス 200A を実行することができる。例えば、符号器は、プロセス 200A を反復して実行することができる、その場合、符号器は、プロセス 200A の 1 回の反復において基本処理単位を符号化することができる。いくつかの実施形態では、符号器は、映像シーケンス 202 のそれぞれの元のピクチャの領域（例えば、領域 114～118）に対して、プロセス 200A を並列に実行することができる。

【0033】

【0059】 図 2A では、符号器は、映像シーケンス 202 の元のピクチャの基本処理単位（「元の BPU」と呼ばれる）を予測段階 204 に供給して、予測データ 206 及び予測された BPU 208 を生成することができる。符号器は、元の BPU から、予測された BPU 208 を減算して、残差 BPU 210 を生成することができる。符号器は、残差 BPU 210 を変換段階 212 及び量子化段階 214 に供給して、量子化された変換係数 216 を生成することができる。符号器は、予測データ 206 及び量子化された変換係数 216 をバイナリコード化段階 226 に供給して、映像ビットストリーム 228 を生成することができる。構成要素 202、204、206、208、210、212、214、216、226 及び 228 は、「順方向経路」と呼ぶことができる。プロセス 200A の間、量子化段階 214 の後に、符号器は、量子化された変換係数 216 を逆量子化段階 218 及び逆変換段階 220 に供給して、再構築された残差 BPU 222 を生成することができる。符号器は、再構築された残差 BPU 222 を予測された BPU 208 に加算して、予測基準 224 を生成することができる、この予測基準は、プロセス 200A の次の反復のための予測段階 204 で使用される。プロセス 200A の構成要素 218、220、222 及び 224 は、「再構築経路」と呼ぶことができる。再構築経路は、符号器及び復号器の両方が、予測に同じ参照データを確実に使用するために使用することができる。

【0034】

【0060】 符号器は、プロセス 200A を反復して実行して、元のピクチャのそれぞれの元の BPU を（順方向経路で）符号化し、元のピクチャの次の元の BPU を（再構築経路で）符号化するための予測基準 224 を生成することができる。元のピクチャの元の BPU をすべて符号化した後、符号器は、映像シーケンス 202 内の次のピクチャの符号化に進むことができる。

【0035】

【0061】 プロセス 200A を参照すると、符号器は、映像捕捉デバイス（例えば、カメラ）によって生成された映像シーケンス 202 を受信することができる。本明細書で使用する「受信（する）」という用語は、受信すること、入力すること、獲得すること、取り出すこと、取得すること、読み出しすること、アクセスすること、又はデータを入力するための任意のやり方の任意の行為を指すことができる。

【0036】

【0062】 予測段階 204 では、現在の反復において、符号器は、元の BPU 及び予測基準 224 を受信し、予測操作を実行して予測データ 206 及び予測された BPU 208 を生成することができる。予測基準 224 は、プロセス 200A の前の反復の再構築経路から生成することができる。予測段階 204 の目的は、予測データ 206 を抽出することにより、情報の冗長性を低減することであり、予測データ 206 は、予測データ 206 及び予測基準 224 から、予測された BPU 208 として抽出することにより、元の BPU を再構築するために使用可能である。

【0037】

【0063】 理想的には、予測された BPU 208 は、元の BPU と同一となり得る。しかしながら、予測操作及び再構築操作が理想的ではないことにより、予測された BPU 208 は、概して、元の BPU とはわずかに異なっている。このような差を記録するために、予測された BPU 208 を生成した後、符号器は、それを元の BPU から減算して、残差 BPU 210 を生成することができる。例えば、符号器は、予測された BPU 208 のピクセルの値（例えば、グレースケール値又は RGB 値）を元の BPU の対応するピクセルの値から減算することができる。残差 BPU 210 の各ピクセルは、元の BPU の対応す

るピクセルと、予測されたBPU208との間の、このような減算の結果としての残差値を有することができる。元のBPUと比較して、予測データ206及び残差BPU210は、有するビット数が少なくなる可能性があるが、著しい品質の劣化なしに元のBPUを再構築するためにそれらを使用することができる。このようにして、元のBPUは圧縮される。

【0038】

[0064] 残差BPU210をさらに圧縮するために、変換段階212において、符号器は、残差BPU210を、2次元「基底パターン」のセットであって、各基底パターンが「変換係数」に関連する基底パターンのセットに分解することにより、残差BPU210の空間的冗長性を低減することができる。基底パターンは、同じサイズ（例えば、残差BPU210のサイズ）を有することができる。それぞれの基底パターンは、残差BPU210の変動周波数（例えば、輝度変動周波数）成分を表すことができる。基底パターンはいずれも、他のどの基底パターンのどの組み合わせ（例えば、線形結合）からも再現することができない。言い換えれば、分解は、残差BPU210の変動を周波数領域に分解することができる。このような分解は、関数の離散フーリエ変換に類似し、基底パターンは、離散フーリエ変換の基底関数（例えば、三角関数）に類似し、変換係数は、基底関数に関連する係数に類似する。

【0039】

[0065] 変換アルゴリズムが異なれば、使用する基底パターンも異なる可能性がある。変換段階212では、例えば、離散コサイン変換、離散サイン変換、などといったような、様々な変換アルゴリズムを使用することができる。変換段階212における変換は、逆に行うことが可能である。すなわち、符号器は、変換の逆操作（「逆変換」と呼ばれる）によって残差BPU210を復元することができる。例えば、残差BPU210のピクセルを復元するために、逆変換は、基底パターンの対応するピクセルの値を、関連するそれぞれの係数で乗算し、積を加算して加重和を生み出すこと、とすることができる。映像コード化規格では、符号器及び復号器の両方が同じ変換アルゴリズム（したがって同じ基底パターン）を使用することができる。したがって、符号器は、変換係数のみを記録することができ、復号器は、符号器から基底パターンを受信することなく、変換係数から残差BPU210を再構築することができる。残差BPU210と比較して、変換係数は、より少ないビット数とすることができ、著しい品質の劣化なしに残差BPU210を再構築するためにそれらを使用することができる。このようにして、残差BPU210がさらに圧縮される。

【0040】

[0066] 符号器は、量子化段階214において変換係数をさらに圧縮することができる。変換プロセスでは、様々な基底パターンが様々な変動周波数（例えば、輝度変動周波数）を表すことができる。人間の目は、概して、低周波変動の方が良く認識できるため、符号器は、復号化の際に著しい品質劣化を引き起こすことなく、高周波変動の情報を無視することができる。例えば、量子化段階214において、符号器は、各変換係数を整数値（「量子化スケール因子」と呼ばれる）で除算し、商をその最も近い整数に丸めることにより、量子化された変換係数216を生成することができる。このような操作後、高周波基底パターンのいくつかの変換係数をゼロに変換することができ、低周波基底パターンの変換係数をより小さい整数に変換することができる。符号器は、ゼロ値の量子化された変換係数216を無視することができ、これにより、変換係数がさらに圧縮される。量子化プロセスもまた逆に行うことが可能であり、その場合、量子化された変換係数216は、量子化の逆操作（「逆量子化」と呼ばれる）で変換係数に再構築することができる。

【0041】

[0067] 符号器は、丸め操作でのこのような除算の剰余を無視するため、量子化段階214は、不可逆となり得る。通常、量子化段階214は、プロセス200Aにおける最大の情報損失の原因となる可能性がある。情報損失が大きいほど、量子化された変換係数216が必要とし得るビット数が少なくなる。様々なレベルの情報損失を取得するために、

符号器は、様々な値の量子化パラメータ又は量子化プロセスの任意の他のパラメータを使用することができる。

【0042】

【0068】 バイナリコード化段階226において、符号器は、例えば、エントロピーコード化、可変長コード化、算術コード化、ハフマンコード化、コンテキスト適応バイナリ算術コード化、又は任意の他の可逆若しくは不可逆圧縮アルゴリズムといったような、バイナリコード化技法を使用して、予測データ206及び量子化された変換係数216を符号化することができる。いくつかの実施形態では、予測データ206及び量子化された変換係数216に加えて、符号器は、バイナリコード化段階226において、例えば、予測段階204で使用される予測モード、予測操作のパラメータ、変換段階212における変換のタイプ、量子化プロセスのパラメータ（例えば、量子化パラメータ）、符号器制御パラメータ（例えば、ビットレート制御パラメータ）、などといったような、他の情報を符号化することができる。符号器は、バイナリコード化段階226の出力データを使用して、映像ビットストリーム228を生成することができる。いくつかの実施形態では、映像ビットストリーム228は、ネットワーク伝送のためにさらにパケット化することができる。

10

【0043】

【0069】 プロセス200Aの再構築経路を参照すると、逆量子化段階218において、符号器は、量子化された変換係数216に対して逆量子化を実行して、再構築された変換係数を生成することができる。逆変換段階220において、符号器は、再構築された変換係数に基づいて、再構築された残差BPU222を生成することができる。符号器は、再構築された残差BPU222を、予測されたBPU208に加算して、プロセス200Aの次の反復で使用される予測基準224を生成することができる。

20

【0044】

【0070】 プロセス200Aの他の変形例を使用して、映像シーケンス202を符号化することが可能であることに留意されたい。いくつかの実施形態では、プロセス200Aの各段階は、符号器によって異なる順序で実行することができる。いくつかの実施形態では、プロセス200Aの1つ又は複数の段階は、単一の段階にまとめることができる。いくつかの実施形態では、プロセス200Aの単一の段階は、複数の段階に分割することができる。例えば、変換段階212と量子化段階214とを単一の段階にまとめることができる。いくつかの実施形態では、プロセス200Aは、追加の段階を含むことができる。いくつかの実施形態では、プロセス200Aは、図2Aの1つ又は複数の段階を省略することができる。

30

【0045】

【0071】 図2Bは、本開示の実施形態と合致する符号化プロセス200Bの別の例の概略図を示す。プロセス200Bは、プロセス200Aから修正することができる。例えば、プロセス200Bは、ハイブリッド映像コード化規格（例えば、H.26xシリーズ）に準拠する符号器によって使用することができる。プロセス200Aと比較して、プロセス200Bの順方向経路は、モード判定段階230を追加で含み、予測段階204を空間的予測段階2042と、時間的予測段階2044と、に分割する。プロセス200Bの再構築経路は、ループフィルタ段階232及びバッファ234を追加で含む。

40

【0046】

【0072】 概して、予測技法は、空間的予測及び時間的予測の2つのタイプに分類することができる。空間的予測（例えば、イントラピクチャ予測又は「イントラ予測」）は、同じピクチャ内のすでにコード化された1つ又は複数の隣接するBPUからのピクセルを使用して、現BPUを予測することができる。すなわち、空間的予測における予測基準224は、隣接するBPUを含むことができる。空間的予測は、ピクチャに固有の空間的冗長性を低減することができる。時間的予測（例えば、インターピクチャ予測又は「インター予測」）は、すでにコード化された1つ又は複数のピクチャからの領域を使用して、現BPUを予測することができる。すなわち、時間的予測における予測基準224は、コード化されたピクチャを含むことができる。時間的予測は、ピクチャに固有の時間的冗長性を

50

低減することができる。

【0047】

【0073】 プロセス200Bを参照すると、順方向経路において、符号器は、空間的予測段階2042及び時間的予測段階2044で予測操作を実行する。例えば、空間的予測段階2042では、符号器は、イントラ予測を実行することができる。符号化されているピクチャの元のBPUでは、予測基準224は、同じピクチャ内の（順方向経路で）符号化され、（再構築された経路で）再構築されている1つ又は複数の隣接するBPUを含むことができる。符号器は、隣接するBPUを外挿することにより、予測されたBPU208を生成することができる。外挿技法は、例えば、線形外挿又は直線補間、多項式外挿又は多項式補間などを含むことができる。いくつかの実施形態では、符号器は、予測されたBPU208のピクセルごとに対応するピクセルの値を外挿することなどによって、ピクセルレベルで外挿を実行することができる。外挿に使用される隣接するBPUは、垂直方向に（例えば、元のBPUの上に）、水平方向に（例えば、元のBPUの左に）、対角線方向に（例えば、元のBPUの左下、右下、左上、若しくは右上に）、又は使用される映像コード化規格で定義される任意の方向に、といったような、様々な方向から元のBPUに対して位置し得る。イントラ予測では、予測データ206は、例えば、使用される隣接するBPUの位置（例えば、座標）、使用される隣接するBPUのサイズ、外挿のパラメータ、元のBPUに対する使用される隣接するBPUの方向などを含むことができる。

【0048】

【0074】 別の例では、時間的予測段階2044において、符号器は、インター予測を実行することができる。現ピクチャの元のBPUでは、予測基準224は、（順方向経路で）符号化され、（再構築された経路で）再構築されている1つ又は複数のピクチャ（「参照ピクチャ」と呼ばれる）を含むことができる。いくつかの実施形態では、参照ピクチャをBPUごとに符号化し、再構築することができる。例えば、符号器は、再構築された残差BPU222を予測されたBPU208に加算して、再構築されたBPUを生成することができる。同じピクチャの再構築されたBPUがすべて生成されると、符号器は、参照ピクチャとして再構築されたピクチャを生成することができる。符号器は、「動き推定」の操作を実行して、参照ピクチャ内のある範囲（「探索窓」と呼ばれる）において一致領域を探索することができる。参照ピクチャ内の探索窓の位置は、現ピクチャ内の元のBPUの位置に基づいて決定することができる。例えば、探索窓は、参照ピクチャ内で、現ピクチャ内の元のBPUと同じ座標を有する位置に中心を置くことができ、所定の距離にわたって広げることができる。符号器が探索窓内で元のBPUと同様の領域を（例えば、画素再帰アルゴリズム、ブロックマッチングアルゴリズム、などを使用することによって）識別すると、符号器は、このような領域を一致領域として決定することができる。一致領域は、元のBPUとは異なる寸法（例えば、それよりも小さい、等しい、大きい又は異なる形状）を有することができる。参照ピクチャ及び現ピクチャは、（例えば、図1に示されているように）タイムライン内で時間的に隔てられているため、時間が経つにつれて一致領域が元のBPUの位置に「移動する」と見なすことができる。符号器は、このような動きの方向及び距離を「動きベクトル」として記録することができる。（例えば、図1のピクチャ106のように）複数の参照ピクチャが使用されるとき、符号器は、それぞれの参照ピクチャごとに一致領域を探索し、その関連する動きベクトルを求めることができる。いくつかの実施形態では、符号器は、それぞれの一致する参照ピクチャの一致領域のピクセル値に重みを割り当てることができる。

【0049】

【0075】 動き推定を使用して、例えば、平行移動、回転、拡大縮小、などといったような、様々なタイプの動きを識別することができる。インター予測では、予測データ206は、例えば、一致領域の位置（例えば、座標）、一致領域に関連する動きベクトル、参照ピクチャの数、参照ピクチャに関連する重み、などを含むことができる。

【0050】

【0076】 予測されたBPU208を生成するために、符号器は、「動き補償」の操作を

実行することができる。動き補償を使用して、予測データ206（例えば、動きベクトル）及び予測基準224に基づき、予測されたBPU208を再構築することができる。例えば、符号器は、動きベクトルに従って参照ピクチャの一致領域を動かすことができ、その場合、符号器は、現ピクチャの元のBPUを予測することができる。（例えば、図1のピクチャ106のように）複数の参照ピクチャが使用されるとき、符号器は、それぞれの動きベクトルに従って参照ピクチャの一致領域を動かし、一致領域のピクセル値を平均することができる。いくつかの実施形態では、符号器が、それぞれの一致する参照ピクチャの一致領域のピクセル値に重みを割り当てた場合、符号器は、動かした一致領域のピクセル値の加重和を加算することができる。

【0051】

【0077】 いくつかの実施形態では、インター予測は、片方向又は双方向とすることができる。片方向インター予測は、現ピクチャに対して同じ時間的方向にある1つ又は複数の参照ピクチャを使用することができる。例えば、図1のピクチャ104は片方向インター予測されたピクチャであり、参照ピクチャ（例えば、ピクチャ102）がピクチャ104に先行する。双方向インター予測は、現ピクチャに対して両方の時間的方向にある1つ又は複数の参照ピクチャを使用することができる。例えば、図1のピクチャ106は双方向インター予測されたピクチャであり、参照ピクチャ（例えば、ピクチャ104及び108）がピクチャ104に対して両方の時間的方向にある。

【0052】

【0078】 プロセス200Bの順方向経路を引き続き参照すると、空間的予測2042及び時間的予測段階2044の後、モード判定段階230において、符号器は、プロセス200Bの現在の反復のための予測モード（例えば、イントラ予測又はインター予測のうちの1つ）を選択することができる。例えば、符号器は、レート歪み最適化技法を実行することができる。その場合、符号器は、候補予測モードのビットレート及び候補予測モード下の再構築された参照ピクチャの歪みに応じて、予測モードを選択して、コスト関数の値を最小化することができる。選択された予測モードに応じて、符号器は、対応する予測されたBPU208及び予測されたデータ206を生成することができる。

【0053】

【0079】 プロセス200Bの再構築経路において、順方向経路内でイントラ予測モードが選択されている場合、予測基準224（例えば、現ピクチャ内で符号化され、再構築されている現BPU）を生成した後、符号器は、後で使用するために（例えば、現ピクチャの次のBPUを外挿するために）、予測基準224を空間的予測段階2042に直接供給することができる。符号器は、予測基準224をループフィルタ段階232に供給することができる。この段階で、符号器は、ループフィルタを予測基準224に適用して、予測基準224のコード化の間に引き起こされる歪み（例えば、ブロッキングアーティファクト）を低減又は除去することができる。符号器はループフィルタ段階232で、例えば、デブロッキング、サンプル適応オフセット、適応ループフィルタ、などといったような、様々なループフィルタ技法を適用することができる。ループフィルタされた参照ピクチャは、後で使用するために（例えば、映像シーケンス202のこの先のピクチャのためのインター予測参照ピクチャとして使用するために）、バッファ234（又は「復号化されたピクチャバッファ」）内に記憶することができる。符号器は、1つ又は複数の参照ピクチャをバッファ234内に記憶して、時間的予測段階2044で使用するすることができる。いくつかの実施形態では、符号器は、量子化された変換係数216、予測データ206、及びその他の情報とともに、バイナリコード化段階226でループフィルタのパラメータ（例えば、ループフィルタの強度）を符号化することができる。

【0054】

【0080】 図3Aは、本開示の実施形態と合致する復号化プロセス300Aの一例の概略図を示す。プロセス300Aは、図2Aにおける圧縮プロセス200Aに対応する解凍プロセスとすることができる。いくつかの実施形態では、プロセス300Aは、プロセス200Aの再構築経路と同様とすることができる。復号器は、プロセス300Aに従って映

10

20

30

40

50

像ビットストリーム 228 を映像ストリーム 304 に復号化することができる。映像ストリーム 304 は、映像シーケンス 202 と非常に類似したものとすることができる。しかしながら、圧縮及び解凍プロセス（例えば、図 2A～図 2B の量子化段階 214）における情報損失により、概して、映像ストリーム 304 は、映像シーケンス 202 と同一ではない。図 2A～図 2B のプロセス 200A 及び 200B と同様に、復号器は、映像ビットストリーム 228 内で符号化されるそれぞれのピクチャごとに、基本処理単位（BPU）のレベルでプロセス 300A を実行することができる。例えば、復号器は、プロセス 300A を反復して実行することができ、その場合、復号器は、プロセス 300A の 1 回の反復において基本処理単位を復号化することができる。いくつかの実施形態では、復号器は、映像ビットストリーム 228 内で符号化されるそれぞれのピクチャの領域（例えば、領域 114～118）に対してプロセス 300A を並列に実行することができる。

10

【0055】

【0081】 図 3A では、復号器は、符号化されたピクチャの基本処理単位（「符号化された BPU」と呼ばれる）に関連する映像ビットストリーム 228 の一部分を、バイナリ復号化段階 302 に供給することができる。バイナリ復号化段階 302 において、復号器は、その一部分を予測データ 206 及び量子化された変換係数 216 に復号化することができる。復号器は、量子化された変換係数 216 を逆量子化段階 218 及び逆変換段階 220 に供給して、再構築された残差 BPU 222 を生成することができる。復号器は、予測データ 206 を予測段階 204 に供給して、予測された BPU 208 を生成することができる。復号器は、再構築された残差 BPU 222 を、予測された BPU 208 に加算して、予測基準 224 を生成することができる。いくつかの実施形態では、予測基準 224 は、バッファ（例えば、コンピュータメモリ内の復号化されたピクチャバッファ）内に記憶することができる。復号器は、予測基準 224 を、プロセス 300A の次の反復で予測操作を実行するための予測段階 204 に供給することができる。

20

【0056】

【0082】 復号器は、プロセス 300A を反復的に実行して、符号化されたピクチャのそれぞれの符号化された BPU を復号化し、符号化されたピクチャの次の符号化された BPU を符号化するための予測基準 224 を生成することができる。符号化されたピクチャのすべての符号化された BPU を復号化した後、復号器は、表示するためにピクチャを映像ストリーム 304 に出力し、映像ビットストリーム 228 内の次の符号化されたピクチャの復号化に進むことができる。

30

【0057】

【0083】 バイナリ復号化段階 302 において、復号器は、符号器によって使用されるバイナリコード化技法（例えば、エントロピーコード化、可変長コード化、算術コード化、ハフマンコード化、コンテキスト適応バイナリ算術コード化、又は任意の他の可逆圧縮アルゴリズム）の逆操作を実行することができる。いくつかの実施形態では、予測データ 206 及び量子化された変換係数 216 に加えて、復号器は、バイナリ復号化段階 302 において、例えば、予測モード、予測操作のパラメータ、変換のタイプ、量子化プロセスのパラメータ（例えば、量子化パラメータ）、符号器制御パラメータ（例えば、ビットレート制御パラメータ）、などといったような、他の情報を復号化することができる。いくつかの実施形態では、映像ビットストリーム 228 がネットワーク上においてパケット単位で伝送される場合、復号器は、映像ビットストリーム 228 をデパケット化してから、それをバイナリ復号化段階 302 に供給することができる。

40

【0058】

【0084】 図 3B は、本開示の実施形態と合致する復号化プロセス 300B の別の例の概略図を示す。プロセス 300B は、プロセス 300A から修正することができる。例えば、プロセス 300B は、ハイブリッド映像コード化規格（例えば、H. 26x シリーズ）に準拠する復号器によって使用することができる。プロセス 300A と比較して、プロセス 300B は、予測段階 204 を空間的予測段階 2042 及び時間的予測段階 2044 に追加で分割し、ループフィルタ段階 232 及びバッファ 234 を追加で含む。

50

【0059】

[0085] プロセス300Bにおいて、復号化中の符号化されたピクチャ（「現ピクチャ」と呼ばれる）の符号化された基本処理単位（「現BPU」と呼ばれる）では、復号器によってバイナリ復号化段階302から復号化される予測データ206は、現BPUを符号化するためにどの予測モードが符号器によって使用されたかに応じて、様々なタイプのデータを含むことができる。例えば、現BPUを符号化するために符号器によってイントラ予測が使用された場合、予測データ206は、イントラ予測、イントラ予測操作のパラメータ、などを示す予測モード指標（例えば、フラグ値）を含むことができる。イントラ予測操作のパラメータは、例えば、基準として使用される1つ又は複数の隣接するBPUの位置（例えば、座標）、隣接するBPUのサイズ、外挿のパラメータ、元のBPUに対する隣接するBPUの方向、などを含むことができる。別の例では、現BPUを符号化するためにインター予測が符号器によって使用された場合、予測データ206は、インター予測、インター予測操作のパラメータ、などを示す予測モード指標（例えば、フラグ値）を含むことができる。インター予測操作のパラメータは、例えば、現BPUに関連する参照ピクチャの数、参照ピクチャにそれぞれ関連する重み、それぞれの参照ピクチャ内の1つ又は複数の一致領域の位置（例えば、座標）、一致領域にそれぞれ関連する1つ又は複数の動きベクトル、などを含むことができる。

10

【0060】

[0086] 予測モード指標に基づき、復号器は、空間的予測段階2042で空間的予測（例えば、イントラ予測）を実行するのか、又は時間的予測段階2044で時間的予測（例えば、インター予測）を実行するのか、を判定することができる。このような空間的予測又は時間的予測の実行の詳細は、図2Bで説明しているので、以下では繰り返さない。このような空間的予測又は時間的予測を実行した後、復号器は、予測されたBPU208を生成することができる。図3Aで説明したように、復号器は、予測されたBPU208と、再構築された残差BPU222とを加算して、予測基準224を生成することができる。

20

【0061】

[0087] プロセス300Bにおいて、復号器は、予測基準224を、プロセス300Bの次の反復で予測操作を実行するための空間的予測段階2042又は時間的予測段階2044に供給することができる。例えば、現BPUが空間的予測段階2042においてイントラ予測を使用して復号化される場合、予測基準224（例えば、復号化された現BPU）を生成した後、復号器は、後で使用するために（例えば、現ピクチャの次のBPUを外挿するために）、予測基準224を空間的予測段階2042に直接供給することができる。現BPUが時間的予測段階2044においてインター予測を使用して復号化される場合、予測基準224（例えば、すべてのBPUが復号化されている参照ピクチャ）を生成した後、復号器は、予測基準224をループフィルタ段階232に供給して、歪み（例えば、ブロッキングアーティファクト）を低減又は除去することができる。復号器は、図2Bで説明したやり方で予測基準224にループフィルタを適用することができる。ループフィルタされた参照ピクチャは、後で使用するために（例えば、映像ビットストリーム228のこの先の符号化されたピクチャのためのインター予測参照ピクチャとして使用するために）バッファ234（例えば、コンピュータメモリ内の復号化されたピクチャバッファ）内に記憶することができる。復号器は、1つ又は複数の参照ピクチャをバッファ234内に記憶して、時間的予測段階2044で使用するすることができる。いくつかの実施形態では、予測データは、ループフィルタのパラメータ（例えば、ループフィルタの強度）をさらに含むことができる。いくつかの実施形態では、予測データ206の予測モード指標が、現BPUを符号化するためにインター予測が使用されたことを示すとき、予測データは、ループフィルタのパラメータを含む。

30

40

【0062】

[0088] 図4は、本開示の実施形態と合致する、映像を符号化又は復号化するための装置400の一例のブロック図である。図4に示されているように、装置400は、プロセッサ402を含むことができる。プロセッサ402が本明細書に記載の命令を実行すると

50

き、装置 400 は、映像符号化又は復号化のための専用機器となることができる。プロセッサ 402 は、情報を取り扱い又は処理することが可能な任意のタイプの回路類とすることができる。例えば、プロセッサ 402 は、任意の数の中央処理ユニット（すなわち「CPU」）、グラフィックス処理ユニット（すなわち「GPU」）、ニューラル処理ユニット（「NPU」）、マイクロコントローラユニット（「MCU」）、光プロセッサ、プログラマブル論理コントローラ、マイクロコントローラ、マイクロプロセッサ、デジタル信号プロセッサ、アイピー（IP：Intellectual Property）コア、プログラマブル論理アレイ（PLA）、プログラマブルアレイ論理（PAL）、汎用アレイ論理（GAL）、複合プログラマブル論理デバイス（CPLD）、フィールドプログラマブルゲートアレイ（FPGA）、システムオンチップ（SoC）、特定用途向け集積回路（ASIC）、などの任意の組み合わせを含むことができる。いくつかの実施形態では、プロセッサ 402 は、単一の論理構成要素としてグループ化されるプロセッサのセットとすることもまた可能である。例えば、図 4 に示されているように、プロセッサ 402 は、プロセッサ 402a と、プロセッサ 402b と、プロセッサ 402n と、を含む複数のプロセッサを含むことができる。

【0063】

【0089】 装置 400 は、データ（例えば、命令、コンピュータコード、中間データ、などのセット）を記憶するように構成されたメモリ 404 もまた含むことができる。例えば、図 4 に示されているように、記憶されたデータは、プログラム命令（例えば、プロセス 200A、200B、300A、又は 300B における段階を実装するためのプログラム命令）及び処理用データ（例えば、映像シーケンス 202、映像ビットストリーム 228、又は映像ストリーム 304）を含むことができる。プロセッサ 402 は、プログラム命令及び処理用データに（例えば、バス 410 を介して）アクセスし、プログラム命令を実行して、処理用データに対する操作又は取り扱いを実行することができる。メモリ 404 は、高速ランダムアクセス記憶デバイス又は不揮発性記憶デバイスを含むことができる。いくつかの実施形態では、メモリ 404 は、任意の数のランダムアクセスメモリ（RAM）、読取専用メモリ（ROM）、光学ディスク、磁気ディスク、ハードドライブ、ソリッドステートドライブ、フラッシュドライブ、セキュリティデジタル（SD）カード、メモリスティック、コンパクトフラッシュ（登録商標）（CF）カード、などの任意の組み合わせを含むことができる。メモリ 404 は、単一の論理構成要素としてグループ化されたメモリの群（図 4 には示されていない）とすることもまた可能である。

【0064】

【0090】 バス 410 は、内蔵バス（例えば、CPU メモリバス）、外付けバス（例えば、USB（Universal Serial Bus）ポート、PCI（Peripheral Component Interconnect）エクスプレスポート）、などといったような、装置 400 内の構成要素間でデータを転送する通信デバイスとすることができる。

【0065】

【0091】 曖昧さを招くことなく説明を分かり易くするために、プロセッサ 402 及び他のデータ処理回路は、本開示では、総称して「データ処理回路」と呼ばれる。データ処理回路は、完全にハードウェアとして、又はソフトウェア、ハードウェア、若しくはファームウェアの組み合わせとして実装することができる。加えて、データ処理回路は、単一の独立したモジュールとすることもできるし、或いは、装置 400 の任意の他の構成要素に完全に、若しくは部分的にまとめることもできる。

【0066】

【0092】 装置 400 は、ネットワーク（例えば、インターネット、イントラネット、ローカルエリアネットワーク、モバイル通信ネットワーク、など）との有線通信又はワイヤレス通信を提供するためにネットワークインタフェース 406 をさらに含むことができる。いくつかの実施形態では、ネットワークインタフェース 406 は、任意の数のネットワークインタフェースコントローラ（NIC）、無線周波数（RF）モジュール、トランスポンダ、トランシーバ、モデム、ルータ、ゲートウェイ、有線ネットワークアダプタ、ワ

イヤレスネットワークアダプタ、Bluetooth（登録商標）アダプタ、赤外線アダプタ、近距離無線通信（「NFC」）アダプタ、セルラネットワークチップ、などの任意の組み合わせを含むことができる。

【0067】

[0093] いくつかの実施形態では、任意選択的に、装置400は、1つ又は複数の周辺デバイスへの接続を提供するために周辺機器インタフェース408をさらに含むことができる。図4に示されているように、周辺デバイスは、カーソル制御デバイス（例えば、マウス、タッチパッド、又はタッチスクリーン）、キーボード、ディスプレイ（例えば、ブラウン管ディスプレイ、液晶ディスプレイ、又は発光ダイオードディスプレイ）、映像入力デバイス（例えば、映像アーカイブに結合されたカメラ又は入力インタフェース）、な

10

【0068】

[0094] 映像コーデック（例えば、プロセス200A、200B、300A、又は300Bを実行するコーデック）は、装置400内の任意のソフトウェア又はハードウェアモジュールの任意の組み合わせとして実装できることに留意されたい。例えば、プロセス200A、200B、300A、又は300Bのいくつかの段階又はすべての段階は、メモリ404にロード可能なプログラム命令などの、装置400の1つ又は複数のソフトウェアモジュールとして実装することができる。別の例では、プロセス200A、200B、300A、又は300Bのいくつかの段階又はすべての段階は、専用データ処理回路（例えば、FPGA、ASIC、NPU、など）といった、装置400の1つ又は複数のハードウェアモジュールとして実装することができる。

20

【0069】

[0095] 図5は、パレットモードでコード化されたCUを図示したものである。VVCTドラフト8では、パレットモードは、単色、4:2:0、4:2:2及び4:4:4のカラー形式で 사용할 ことができる。パレットモードが有効化されているとき、CUサイズが64x64以下であり、パレットモードが使用されているかどうかを表示するサンプルが16個よりも多い場合、フラグがCUレベルで送信される。パレットモードが（現）CU500のコード化に利用されている場合、CU内の各位置におけるサンプル値は、代表カラー値のsmallセットによって表される。このセットは、パレット510と呼ばれる。パレットカラー501、502、503に近似する値を有するサンプル位置について、対応するパレットインデックスがシグナリングされる。エスケープインデックス504をシグナリングすることによって、パレット外のカラー値を指定することもまた可能である。次に、エスケープカラーインデックスを使用するCU内のすべての位置について、（量子化された）カラー成分値が、これらのそれぞれの位置ごとにシグナリングされる。

30

【0070】

[0096] パレットのコード化のために、パレット予測子が維持される。図6は、各コード化単位600を符号化した後にパレット予測子を更新するための例示的なプロセスを示す。予測子は、非ウェーブフロントの場合には各スライスの始めに、ウェーブフロントの場合には各CTU行の始めに0（すなわち、空）に初期化される。パレット予測子のエントリごとに、それを現CUの現パレットに含めるかどうかを表示するために再利用フラグがシグナリングされる。再利用フラグは、ゼロのランレングスコード化を使用して送られる。この後、新たなパレットエントリの数、及び新たなパレットエントリの成分値がシグナリングされる。パレットコード化されたCUを符号化した後、パレット予測子は、現パレットを使用して更新されることになり、現パレットで再利用されない前のパレット予測子からのエントリは、許容される最大サイズに達するまで、新たなパレット予測子の末尾で追加される。

40

【0071】

[0097] エスケープフラグがCUごとにシグナリングされ、エスケープ記号が現CU内に存在するかどうかを表示する。エスケープ記号が存在する場合、パレットテーブルは、エスケープ記号となるように割り当てられた最後の1つのインデックスによって拡張され

50

る。図5の例に示されているように、CU内のサンプルのパレットインデックスは、パレットインデックスマップを形成している。インデックスマップは、水平横断スキャン又は垂直横断スキャンを使用してコード化されている。スキャン順序は、`palette_transpose_flag`を使用して、ビットストリームで明確にシグナリングされる。パレットインデックスマップは、インデックス実行モード又はインデックスコピーモードを使用してコード化される。

【0072】

[0098] VVCドラフト8では、デブロッキングフィルタプロセスは、ブロックの境界を定義すること、定義されたブロックの境界に沿って隣接する2つのブロックのコード化モードに基づいて境界フィルタ強度を導出すること、フィルタの対象となるサンプルの数を導出すること、及びデブロッキングフィルタをサンプルに適用することを含む。エッジがコード化単位、コード化サブブロック単位、又は変換単位の境界であるとき、そのエッジをブロックの境界として定義する。次に、以下の6つの規則に従って、隣接する2つのブロックのコード化モードに基づいて、境界フィルタ強度を計算する。(1) 2つのコード化ブロックが両方ともBDPCMモードでコード化されている場合、境界フィルタ強度は0に設定される。(2) そうでない場合、コード化ブロックの一方がイントラモードでコード化されている場合、境界フィルタ強度は2に設定される。(3) そうでない場合、コード化ブロックの一方がCIIPモードでコード化されている場合、境界フィルタ強度は2に設定される。(4) そうでない場合、コード化ブロックの一方が、1つ又は複数の非ゼロ係数レベルを含んでいる場合、境界フィルタ強度は1に設定される。(5) そうでない場合、ブロックの一方がIBCモードでコード化され、もう一方のブロックがインターモードでコード化されている場合、境界フィルタ強度は1に設定される。(6) そうでない場合(2つのブロックの両方がIBCモード又はインターモードでコード化されている)、2つのブロックの参照ピクチャ及び動きベクトルを使用して、境界フィルタ強度を導出する。

【0073】

[0099] VVCドラフト8では、境界フィルタ強度を計算するプロセスがより詳細に記載されている。具体的には、VVCドラフト8では、8つの順次の、網羅的なシナリオが与えられている。シナリオ1では、`cldx`が0に等しく、サンプル p_0 及び q_0 の両方が、`intra_bdpcm_luma_flag`が1に等しいコード化ブロックにある場合、`bs[xDi][yDj]`は、0に等しく設定される。そうでない場合、シナリオ2において、`cldx`が0よりも大きく、サンプル p_0 及び q_0 の両方が、`intra_bdpcm_chroma_flag`が1に等しいコード化ブロックにある場合、`bs[xDi][yDj]`は、0に等しく設定される。そうでない場合、シナリオ3において、サンプル p_0 又は q_0 がイントラ予測モードを用いてコード化されたコード化単位のコード化ブロックにある場合、`bs[xDi][yDj]`は、2に等しく設定される。そうでない場合、シナリオ4において、ブロックエッジがコード化ブロックのエッジでもあり、サンプル p_0 又は q_0 が、`ciip_flag`が1に等しいコード化ブロックにある場合、`bs[xDi][yDj]`は、2に等しく設定される。そうでない場合、シナリオ5において、ブロックエッジが変換ブロックのエッジでもあり、サンプル p_0 又は q_0 が、1つ又は複数の非ゼロ変換係数レベルを含んでいる変換ブロックにある場合、`bs[xDi][yDj]`は、1に等しく設定される。そうでない場合、シナリオ6において、サンプル p_0 を含んでいるコード化サブブロックの予測モードが、サンプル q_0 を含んでいるコード化サブブロックの予測モードとは異なる(すなわち、コード化サブブロックのうちの一方がIBC予測モードでコード化されており、もう一方がインター予測モードでコード化されている)場合、`bs[xDi][yDj]`は、1に等しく設定される。

【0074】

[0100] そうでない場合、シナリオ7において、`cldx`が0に等しく、`edgeFlags[xDi][yDj]`が2に等しく、以下の条件のうちの1つ又は複数が、真である場合、`bs[xDi][yDj]`は、1に等しく設定される。

【0075】

10

20

30

40

50

【0101】 条件（１）：サンプル p_0 を含んでいるコード化サブブロック、及びサンプル q_0 を含んでいるコード化サブブロックの両方が、IBC 予測モードでコード化されており、２つのコード化サブブロックの予測で使用するブロックベクトルの水平成分又は垂直成分の間の絶対差が、 $1/16$ ルマサンプル単位で 8 以上である。

【0076】

【0102】 条件（２）：サンプル p_0 を含んでいるコード化サブブロックの予測に対して、サンプル q_0 を含んでいるコード化サブブロックの予測に対するのとは異なる参照ピクチャ、又は異なる数の動きベクトルが使用される。条件（２）については、２つのコード化サブブロックに使用される参照ピクチャが同じか、異なるかの判定は、予測が参照ピクチャリスト 0 へのインデックスを使用して形成されるのか、参照ピクチャリスト 1 へのインデックスを使用して形成されるのかに関係なく、また、参照ピクチャリスト内のインデックス位置が異なっているかどうかに関係なく、どのピクチャが参照されているかだけに基づくことに留意されたい。また、条件（２）についても、左上サンプルが (x_{Sb}, y_{Sb}) をカバーするコード化サブブロックの予測に使用される動きベクトルの数は、 $PredFlagL0[x_{Sb}][y_{Sb}] + PredFlagL1[x_{Sb}][y_{Sb}]$ に等しいことに留意されたい。

【0077】

【0103】 条件（３）：１つの動きベクトルが、サンプル p_0 を含んでいるコード化サブブロックを予測するために使用され、１つの動きベクトルが、サンプル q_0 を含んでいるコード化サブブロックを予測するために使用され、使用される動きベクトルの水平成分又は垂直成分の間の絶対差は、 $1/16$ ルマサンプル単位で 8 以上である。

【0078】

【0104】 条件（４）：２つの動きベクトル及び２つの異なる参照ピクチャが、サンプル p_0 を含んでいるコード化サブブロックを予測するために使用され、同じ２つの参照ピクチャの２つの動きベクトルが、サンプル q_0 を含んでいるコード化サブブロックを予測するために使用され、同じ参照ピクチャの２つのコード化サブブロックの予測で使用する２つの動きベクトルの水平成分又は垂直成分の間の絶対差は、 $1/16$ ルマサンプル単位で 8 以上である。

【0079】

【0105】 条件（５）：同じ参照ピクチャの２つの動きベクトルが、サンプル p_0 を含んでいるコード化サブブロックを予測するために使用され、同じ参照ピクチャの２つの動きベクトルが、サンプル q_0 を含んでいるコード化サブブロックを予測するために使用され、以下の条件が両方とも真である。条件（５．１）：２つのコード化サブブロックの予測で使用するリスト 0 の動きベクトルの水平成分又は垂直成分の間の絶対差は、 $1/16$ ルマサンプル単位で 8 以上であるか、或いは、２つのコード化サブブロックの予測で使用するリスト 1 の動きベクトルの水平成分又は垂直成分の間の絶対差は、 $1/16$ ルマサンプル単位で 8 以上である。条件（５．２）：サンプル p_0 を含んでいるコード化サブブロックの予測で使用するリスト 0 の動きベクトルと、サンプル q_0 を含んでいるコード化サブブロックの予測で使用するリスト 1 の動きベクトルと、の水平成分又は垂直成分の間の絶対差は、 $1/16$ ルマサンプル単位で 8 以上であるか、或いは、サンプル p_0 を含んでいるコード化サブブロックの予測で使用するリスト 1 の動きベクトルと、サンプル q_0 を含んでいるコード化サブブロックの予測で使用するリスト 0 の動きベクトルと、の水平成分又は垂直成分の間の絶対差は、 $1/16$ ルマサンプル単位で 8 以上である。

【0080】

【0106】 そうでない場合、つまり前述の 7 つのシナリオがどれも満たされない場合、シナリオ 8 において、変数 $bS[xDi][yDj]$ が 0 に等しく設定される。境界フィルタ強度を導出した後に、フィルタの対象となるサンプルの数が導出され、デブロッキングフィルタがサンプルに適用される。ブロックがパレットモードでコード化されたとき、サンプルの数は 0 に等しく設定されることに留意されたい。これは、デブロッキングフィルタが、パレットモードでコード化されたブロックに適用されないことを意味する。

【0081】

【0107】 前述したように、ブロックの現パレットの構造は、2つの部分を含んでいる。第1に、現パレット内のエントリは、パレット予測子から予測することができる。パレット予測子のエントリごとに、再利用フラグがシングナリングされ、このエントリが現パレットに含まれるか否かを表示する。第2に、現パレットエントリの成分値は、直接シングナリングすることができる。現パレットを得た後、パレット予測子は、現パレットを使用して更新される。図7は、VVCドラフト8の節7.3.10.6（「パレットコード化シンタックス」）の一部分を示す。パレットコード化ブロックのシンタックスを構文解析するとき、再利用フラグ（すなわち、図7の「palette_predictor_run」701）が最初に復号化され、その後が続いて、新たなパレットエントリの成分値（すなわち、図7の「num_signalled_palette_entries」702及び「new_palette_entries」703）が、復号化される。VVCドラフト8では、再利用フラグを構文解析する前に、パレット予測子のエントリ（すなわち、図7の「PredictorPaletteSize[startComp]」704）の数を知っておくことが必要である。それは、隣接する2つのブロックが両方ともパレットモードでコード化されているとき、第2のブロックのシンタックスは、第1のブロックのパレット予測子の更新プロセスが終了するまで構文解析できないことを意味する。

【0082】

【0108】 しかしながら、従来のパレット予測子の更新プロセスでは、各エントリの再利用フラグを検査することが必要とされる。より悪いシナリオでは、最大63回もの検査が必要である。これらの従来の設計は、少なくとも次の2つの理由でハードウェアフレンドリでない場合がある。第1に、コンテキスト適応的二進算術符号化（CABAC：Context-based Adaptive Binary Arithmetic Coding）構文解析は、パレット予測子が完全に更新されるまで待つ必要がある。通常、CABAC構文解析は、ハードウェアで最も遅いモジュールである。これにより、CABACの処理能力が低減する場合がある。第2に、パレット予測子更新プロセスをCABAC構文解析段階で実装すると、ハードウェアに負担がかかる場合がある。

【0083】

【0109】 加えて、これらの従来の設計の別の課題は、境界フィルタ強度がパレットモードに対して定義されないことである。隣接するブロックのうちの一方がパレットモードでコード化されており、もう一方の隣接するブロックが、IBCモード又はインターモードでコード化されているとき、境界フィルタ強度は定義されない。

【0084】

【0110】 本開示の実施形態は、上述した1つ又は複数の課題の解決に取り組むための実装形態を提供する。これらの実施形態は、パレット予測子更新プロセスを改良し、上述したパレット予測プロセス、又は同様のプロセスを実装するシステムの効率、速度、及びリソース消費を高める場合がある。

【0085】

【0111】 いくつかの実施形態では、パレット予測子更新プロセスを簡素化して、その複雑さを低減することにより、ハードウェアリソースを他の目的のために使えるようにする。図8は、本開示の実施形態による、パレット予測子更新プロセス800のフローチャートを示す。方法800は、符号器によって（例えば、図2Aのプロセス200A又は図2Bの200Bにより）、又は復号器によって（例えば、図3Aのプロセス300A又は図3Bの300Bにより）実行することができるし、或いは、装置（例えば、図4の装置400）の1つ又は複数のソフトウェア若しくはハードウェア構成要素によって実行することができる。例えば、プロセッサ（例えば、図4のプロセッサ402）は、方法800を実行することができる。いくつかの実施形態では、方法800は、コンピュータ（例えば、図4の装置400）によって実行されるプログラムコードなどのコンピュータ実行可能命令を含む、コンピュータ可読媒体内に具現化されるコンピュータプログラム製品によって実装することができる。図8を参照すると、方法800は、以下のステップ802～804を含んでもよい。

【0086】

【0112】 ステップ802において、パレット予測子を更新すると、現パレットのすべてのパレットエントリが、新たなパレット予測子の前に第1のエントリのセットとして追加される。ステップ804において、前のパレット予測子からのすべてのパレットエントリが、そのエントリが第2のエントリのセットとして現パレットで再利用されるか否かに関わらず、新たなパレット予測子の末尾に追加され、第2のエントリのセットは第1のエントリのセットの後にある。

【0087】

【0113】 例えば、図9は、図8で説明したプロセスと合致する、簡素化されたパレット予測子更新プロセスを提示している。現パレットは、プロセス901及びプロセス902によって生成されている。現パレットは、前のパレット予測子から再利用されたエントリ、及びシグナリングされた新たなパレットエントリを含む。プロセス903では、現パレットのすべてのパレットエントリが新たなパレット予測子の第1のエントリのセットとして追加される（ステップ802に対応）。プロセス904において、エントリが現パレットで再利用されるか否かに関わらず、前のパレット予測子からのパレットエントリが新たなパレット予測子の第2のエントリのセットとして追加される（ステップ804に対応）。

【0088】

【0114】 恩恵は、再利用フラグの値を検査せずに、現パレットのサイズ及び前のパレット予測子のサイズを追加することによって、新たなパレット予測子のサイズが計算されることである。更新プロセスは、（図10に示されているような）VVCDラフト8の従来の設計よりもはるかに簡素になるので、CABACの処理能力を高める可能性がある。

【0089】

【0115】 例えば、図11は、本開示の実施形態と合致する、パレットモードの復号化プロセスの一例を示す。図10の従来の設計と図11の開示されている設計との間の、削除された部分1101を含めた変更点が、取消し線が引かれたテキストで強調されている。

【0090】

【0116】 いくつかの実施形態では、パレット予測子更新プロセスは簡素化されているが、再利用フラグの検査をしないことが原因で、パレット予測子は2つ以上の同一のエントリ（すなわち、パレット予測子の冗長性）を有する場合がある。それでも改良されたとはいえ、これは、パレット予測子の予測効果が低下し得ることを意味する場合がある。

【0091】

【0117】 図12は、本開示の実施形態による、別のパレット予測子更新プロセス1200のフローチャートを示す。方法1200は、符号器によって（例えば、図2Aのプロセス200A又は図2Bの200Bにより）、又は復号器によって（例えば、図3Aのプロセス300A又は図3Bの300Bにより）実行することができるし、或いは、装置（例えば、図4の装置400）の1つ又は複数のソフトウェア若しくはハードウェア構成要素によって実行することができる。例えば、プロセッサ（例えば、図4のプロセッサ402）は、方法1200を実行することができる。いくつかの実施形態では、方法1200は、コンピュータ（例えば、図4の装置400）によって実行されるプログラムコードなどのコンピュータ実行可能命令を含む、コンピュータ可読媒体内に具現化されるコンピュータプログラム製品によって実装することができる。図12を参照すると、方法1200は、以下のステップ1202～1206を含んでもよい。

【0092】

【0118】 この例示的な実施形態では、パレット予測子の冗長性を排除し、パレット予測子更新プロセスを簡素にしておくために、第1の再利用エントリと、最後の再利用エントリとの間の前のパレット予測子のエントリが、直接廃棄される。第1のエントリから第1の再利用エントリまでのエントリ、及び最後の再利用エントリから前のパレット予測子の最後のエントリまでのエントリだけが、新たなパレット予測子に追加される。要約すると、パレット予測子更新プロセスは以下のように修正される。ステップ1202において、現パレットの各エントリが、第1のエントリのセットとして新たなパレット予測子に追加される。ステップ1204において、前のパレット予測子の第1のエントリから第1の再

利用エントリまでの各エントリが、第1のエントリのセットの後に、第2のエントリのセットとして新たなパレット予測子に追加される。ステップ1206において、前のパレット予測子の最後の再利用エントリから最後のエントリまでの各エントリが、第2のエントリのセットの後に、第3のエントリのセットとして新たなパレット予測子に追加される。

【0093】

【0119】例えば、図13は、図12で説明したプロセスと合致する、簡素化されたパレット予測子更新プロセスを提示している。プロセス1301において、現パレットのすべてのパレットエントリが、新たなパレット予測子の第1のエントリのセットとしてパレット予測子に追加される（ステップ1202に対応）。プロセス1302において、前のパレット予測子の第1のエントリから第1の再利用エントリまでの各エントリが、第1のエントリのセットの後に、第2のエントリのセットとして新たなパレット予測子に追加される（ステップ1204に対応）。プロセス1303において、前のパレット予測子の最後の再利用エントリから最後のエントリまでの各エントリが、第2のエントリのセットの後に、第3のエントリのセットとして新たなパレット予測子に追加される（ステップ1206に対応）。いくつかの実施形態では、第3のエントリのセットが第1のエントリのセットの後にあり、第2のエントリのセットが第3のエントリのセットの後にある。

【0094】

【0120】第1の再利用エントリ及び最後のエントリは、再利用フラグを構文解析するときに導出可能であることに留意されたい。再利用フラグの値を検査する必要はない。新たなパレット予測子のサイズは、現パレットのサイズ、0から第1の再利用エントリまでのサイズ、及び最後の再利用エントリから最後のエントリまでのサイズを加算することによって計算される。図14は、本開示の実施形態と合致する、パレットコード化シンタックスの一例を示す。図7の従来の設計と図14の開示されている設計との間の、追加された部分1401を含めた変更点が破線で囲まれている。図15は、本開示の実施形態と合致する、パレットモードの復号化プロセスの一例を示す。図10の従来の設計と図15の開示されている設計との間の、削除された部分1501を含めた変更点が、取消し線が引かれたテキストで強調され、追加された部分1502が破線で囲まれている。

【0095】

【0121】先に開示した実施形態では、パレット予測子更新プロセスは簡素化されているが、このプロセスは、ハードウェア設計のCABAC構文解析段階で実装することが必要である。いくつかの実施形態では、再利用フラグの数は固定値に設定される。したがって、CABACは、パレット予測子更新プロセスを待たずに、構文解析を続けることができる。さらに、パレット予測子更新プロセスは、CABAC構文解析段階以外の異なるパイプライン段階で実装することができ、これにより、ハードウェア設計の柔軟性を高めることが可能になる。

【0096】

【0122】図16は、パレットモードの復号器ハードウェア設計を図示したものである。データ構造1601及びデータ構造1602が、CABAC及び復号化パレットピクセルについて例示的に図示されている。復号器ハードウェア設計1603は、予測子更新モジュール1631、及びCABAC構文解析モジュール1632を含み、予測子更新プロセス及びCABAC構文解析プロセスは、同時進行している。したがって、CABACは、パレット更新プロセスを待たずに、構文解析を続けることができる。

【0097】

【0123】再利用フラグの数をそれぞれのコード化ブロックごとに固定値に設定するために、いくつかの実施形態では、パレット予測子のサイズは、非ウェーブフロントの場合には各スライスの初め、及びウェーブフロントの場合には各CTU行の初めに事前定義された値に初期化される。事前定義された値は、パレット予測子の最大サイズに設定される。一例では、再利用フラグの数は、スライスタイプ及びデュアルツリーモード設定に応じて、31又は63である。スライスタイプがエスライスであり、デュアルツリーモードがオンである（ケース1と呼ばれる）とき、再利用フラグの数は、31に設定される。そうで

ない場合（スライスタイプがBスライス／Pスライスであるか、又は、スライスタイプがEスライスであり、デュアルツリーモードがオフである（ケース2と呼ばれる））、再利用フラグの数は、63に設定される。2つのケース以外の数の再利用フラグを使用することができ、ケース1の再利用フラグの数と、ケース2の再利用フラグの数との間が2倍である関係を維持することが有益な場合がある。それに加え、パレット予測子を初期化するとき、各エントリの値及び各成分が、0又は（ $1 < (シーケンスビット深度 - 1)$ ）に設定される。

【0098】

【0124】例えば、図17は、本開示の実施形態と合致する、パレットモードの復号化プロセスの一例を示す。図10の従来の設計と図17の開示されている設計との間の、削除された部分1701を含めた変更点が、取消し線が引かれたテキストで強調され、追加された部分1702が破線で囲まれている。同様に、図19は、本開示の実施形態と合致する、初期化プロセスの一例を示す。図18の従来の設計と図19の開示されている設計との間の、削除された部分1901を含めた変更点が、取消し線が引かれたテキストで強調され、追加された部分1902が破線で囲まれている。

【0099】

【0125】いくつかの実施形態では、再利用フラグが固定値に設定されている場合、又は図9の実施形態が実装されている場合、何らかの冗長性がパレット予測子に発生する場合がある。いくつかの実施形態では、それらの冗長なエントリが符号器側で予測に選択されることはないので、これは問題にならない場合がある。パレット予測子更新の例が、図20、図21、及び図22に示されている。図20は、VVCドラフト8で指定されている手順に従った一例を示し、図21は、図9に提案されている方法のいくつかの実装形態（第1の実施形態と呼ばれる）に従った一例を示し、図22は、再利用フラグが固定値に設定されている実施形態（第3の実施形態と呼ばれる）のいくつかの実装形態に従った別の例を示す。図20、図21、及び図22の比較が示すように、新たなシグナリングされたパレットエントリがないと仮定して、第1の実施形態（図21）に提案されている方法は、再利用フラグ2101をシグナリングするためのビット数を増やすことができるが、他方、第3の実施形態（図22）に提案されている方法は、VVCドラフト8の設計における方法（図20）の再利用フラグ2001をシグナリングするためのビット数と同じビット数が、再利用フラグ2201をシグナリングするために保たれている。

【0100】

【0126】いくつかの実施形態では、パレット予測子は、最初に固定値に初期化される。それは、パレット予測子に冗長なエントリがあり得ることを意味する。前述したように、冗長なエントリは問題にならない場合もあるが、この実施形態の設計は、符号器がそれらの冗長なエントリを使用するのを防ぐわけではない。符号器がこれらの冗長なエントリのうちの1つを偶然選択した場合、パレット予測子のコード化性能、したがって、パレットモードの性能もまた、低下する可能性がある。このケースを防ぐために、いくつかの実施形態では、再利用フラグをシグナリングするときに、ビットストリーム適合性が追加される。いくつかの実施形態では、ビットストリーム適合性は、再利用フラグをシグナリングするときに、パレット予測子のサイズの値がパレット予測子の最大サイズに等しい。より具体的には、再利用フラグの2値化値に範囲制約が加わっている。

【0101】

【0127】例えば、図23は、本開示の実施形態と合致する、パレットコード化シンタックスの一例を示す。図7の従来の設計と図23の開示されている設計との間の、削除された部分2301を含めた変更点が、取消し線が引かれたテキストで強調され、追加された部分2302が破線で囲まれている。同様に、図25は、本開示の実施形態と合致する、パレットコード化セマンティックスの一例を示す。図24の従来の設計と図25の開示されている設計との間の、削除された部分2501を含めた変更点が、取消し線が引かれたテキストで強調され、追加された部分2502が破線で囲まれている。

【0102】

【0128】 さらにその上に、本開示は、以下の方法を提供して、パレットモードのデブロッピングフィルタの課題に対処する。

【0103】

【0129】 いくつかの実施形態では、パレットモードは、イントラ予測モードのサブセットとして取り扱われる。したがって、隣接するブロックの一方がパレットモードでコード化されている場合、境界フィルタ強度は2に等しく設定される。より具体的には、境界フィルタ強度を計算するプロセスを詳述しているVVCドラフト8仕様の部分に変更されることで、シナリオ3が次のように記述するようになる：「サンプルp0又はq0が、イントラ予測モード又はパレットモードを用いてコード化されたコード化単位のコード化ブロックにある場合、 $bS[xDi][yDj]$ は、2に等しく設定される」。追加された言語に下線が引かれている。

10

【0104】

【0130】 いくつかの実施形態では、パレットモードが独立したコード化モードとして取り扱われる。2つの隣接するブロックのコード化モードが異なるとき、境界フィルタ強度は1に等しく設定される。より具体的には、境界フィルタ強度を計算するプロセスを詳述しているVVCドラフト8仕様の部分に変更されることで、シナリオ6が次のように記述するようになる：「サンプルp0を含んでいるコード化サブブロックの予測モードが、サンプルq0を含んでいるコード化サブブロックの予測モードとは異なる場合、 $bS[xDi][yDj]$ は、1に等しく設定される」。削除する言語に取り消し線が引かれている。

【0105】

【0131】 いくつかの実施形態では、パレットモードが独立したコード化モードとして取り扱われる。BDPCMモード設定と同様に、コード化ブロックのうちの1つがパレットモードであるとき、境界フィルタ強度は0に等しく設定される。より具体的には、境界フィルタ強度を計算するプロセスを詳述しているVVCドラフト8仕様の部分に変更されることで、新たなシナリオが、シナリオ3とシナリオ4の間に挿入され、次のように記述するようになる：「そうでない場合、ブロックエッジがコード化ブロックのエッジでもあり、サンプルp0又はq0が、 $pred_mode_plt_flag$ が1に等しいコード化ブロックにある場合、 $bS[xDi][yDj]$ は、0に等しく設定される」。この修正されたプロセスを使用すれば、シナリオ4～8がシナリオ5～9となるように再ナンバリングされて、9個のシナリオが存在することになる。留意すべきは、パレットモードでコード化されたブロックが、1に等しい $pred_mode_plt_flag$ と等価であることである。

20

30

【0106】

【0132】 いくつかの実施形態では、命令を含む非一時的コンピュータ可読記憶媒体もまた提供され、命令は、上記の方法を実行するための（開示されている符号器及び復号器などの）デバイスによって実行してもよい。一般的な形式の非一時的媒体は、例えば、フロッピー（登録商標）ディスク、フレキシブルディスク、ハードディスク、ソリッドステートドライブ、磁気テープ、若しくは任意の他の磁気データ記憶媒体、CD-ROM、任意の他の光学データ記憶媒体、孔のパターンを有する任意の物理媒体、RAM、PROM、及びEPROM、FLASH（登録商標）-EPROM若しくは任意の他のフラッシュメモリ、NVRAM、キャッシュ、レジスタ、任意の他のメモリチップ若しくはカートリッジ、及びそれらのもののネットワーク化されたバージョンを含む。デバイスは、1つ又は複数のプロセッサ（CPU）、入力／出力インタフェース、ネットワークインタフェース及び／又はメモリを含むことができる。

40

【0107】

【0133】 本明細書において、「第1の（first）」及び「第2の（second）」などの関係用語は、エンティティ又は動作を別のエンティティ又は動作から区別するためにのみ使用され、これらのエンティティ又は動作の間のいかなる実際の関係又は順序をも必要としたり示唆したりするものではないことに留意されたい。さらにその上に、「含む（comprising）」、「有する（having）」、「含有する（containing）」及び「含む（including）」並びに他の同様の形式の語は、意味の点で均等であることを意図し、これらの語のい

50

いずれか1つの後に続く単数又は複数のアイテムがこのような単数又は複数のアイテムの網羅的列挙であることを意図していないし、列挙する単数又は複数のアイテムのみに限定されることも意図していない点で非限定的であることを意図する。

【0108】

【0134】 本明細書で使用する場合、別段の定めがない限り、「又は」という用語は、実行不可能な場合を除いて、あり得るすべての組み合わせを包含する。例えば、あるデータベースがA又はBを含み得ると言う場合、別段の定めがない限り又は実行不可能でない限り、そのデータベースは、A、B、A及びBを含むことができる。第2の例として、あるデータベースがA、B、又はCを含み得ると言う場合、別段の定めがない限り又は実行不可能でない限り、そのデータベースは、A、B、C、A及びB、A及びC、B及びC、A及びB及びCを含むことができる。

10

【0109】

【0135】 上記の実施形態は、ハードウェア若しくはソフトウェア（プログラムコード）、又はハードウェアとソフトウェアとの組み合わせによって実装できることが理解されるであろう。ソフトウェアによって実装される場合、ソフトウェアは、上記のコンピュータ可読媒体に記憶することができる。ソフトウェアは、プロセッサによって実行されるとき、開示されている方法を実行することができる。本開示に記載されている計算ユニット及び他の機能ユニットは、ハードウェア、ソフトウェア、又はハードウェアとソフトウェアとの組み合わせによって実装することができる。当業者は、上記のモジュール／ユニットのうちの複数個を1つのモジュール／ユニットとして組み合わせることができ、また上記のモジュール／ユニットをそれぞれ複数のサブモジュール／サブユニットにさらに分割することもまた理解することであろう。

20

【0110】

【0136】 以下の条項を使用して実施形態をさらに説明することができる。

1. 1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することを含む映像データ処理方法であって、

1つ又は複数のパレット予測子のうちのパレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、及び、

前のパレット予測子からのパレットエントリを、前のパレット予測子のパレットエントリの再利用フラグの値に関わらず、パレット予測子の第2のエントリのセットであって、第1のエントリのセットの後にある第2のエントリのセットとして追加すること、

30

によって更新される、映像データ処理方法。

2. パレット予測子の第1のエントリのセット及び第2のエントリのセットの各パレットエントリが、再利用フラグを含む、条項1に記載の方法。

3. 処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することと、
をさらに含む、条項1に記載の方法。

4. パレット予測子の各パレットエントリが、対応する再利用フラグを有し、

パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、条項1～3のいずれか一項に記載の方法。

40

5. 1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することを含む映像データ処理方法であって、

1つ又は複数のパレット予測子のうちのパレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、

第1の範囲であって、前のパレット予測子の第1のパレットエントリから始まり、前のパレット予測子の、再利用フラグセットを有する第1のパレットエントリで終わる第1の範囲内にある、前のパレット予測子の1つ又は複数のパレットエントリを、パレット予測子の第2の1つ又は複数のエントリのセットとしてパレット予測子に追加すること、及

50

び、

第2の範囲であって、前のパレット予測子の、再利用フラグセットを有する最後のパレットエントリから始まり、前のパレット予測子の最後のパレットエントリで終わる第2の範囲内にある、前のパレット予測子の1つ又は複数のパレットエントリを、パレット予測子の第3の1つ又は複数のエントリのセットとしてパレット予測子に追加すること、

によって更新され、第2のエントリのセット及び第3のエントリのセットが第1のエントリのセットの後にある、映像データ処理方法。

6. パレット予測子の第1のエントリのセット、第2のエントリのセット、及び第3のエントリのセットの各パレットエントリが、再利用フラグを含む、条項5に記載の方法。

7. 処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することと、
をさらに含む、条項5に記載の方法。

8. 1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、

各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、条項5～7のいずれか一項に記載の方法。

9. 映像データ処理方法であって、

処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することと、

パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することと、

を含み、

1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、

各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、映像データ処理方法。

10. パレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、及び、

現パレットで再利用されない前のパレット予測子からのエントリを、パレット予測子の第2のエントリのセットであって、第1のエントリのセットの後にある第2のエントリのセットとして追加すること、

によって更新される、条項9に記載の方法。

11. 固定数が、スライスタイプ及びデュアルツリーモード設定に基づいて設定される、条項9に記載の方法。

12. 1つ又は複数のパレット予測子のうちの1つのパレット予測子のサイズが、非ウェーブフロントの場合にはスライスの初めに、事前定義された値に初期化される、条項9に記載の方法。

13. 1つ又は複数のパレット予測子のうちの1つのパレット予測子のサイズが、ウェーブフロントの場合にはコード化単位行の初めに、事前定義された値に初期化される、条項9に記載の方法。

14. 再利用フラグをシグナリングするときに、パレット予測子のサイズの値がパレット予測子の最大サイズに等しいというビットストリーム適合性を追加することをさらに含む、条項12又は13に記載の方法。

15. 1つ又は複数のパレット予測子を初期化するとき、各エントリの値及び各成分が、0又は $(1 < (シーケンスビット深度 - 1))$ に設定される、条項12～14のいずれか一項に記載の方法。

16. 再利用フラグの2値化値に範囲制約を加えることをさらに含む、条項9～15のいずれか一項に記載の方法。

17. 映像データ処理を実行するための装置であって、

10

20

30

40

50

命令を記憶するように構成されたメモリと、

メモリに結合されたプロセッサとを含み、プロセッサは、命令を実行して、装置に、

1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することを実行させるように構成され、

1つ又は複数のパレット予測子のうちのパレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、及び、

前のパレット予測子からのパレットエントリを、前のパレット予測子のパレットエントリの再利用フラグの値に関わらず、パレット予測子の第2のエントリのセットであって、第1のエントリのセットの後にある第2のエントリのセットとして追加すること、
によって更新される、装置。

10

18. パレット予測子の第1のエントリのセット及び第2のエントリのセットの各パレットエントリが、再利用フラグを含む、条項17に記載の装置。

19. プロセッサが、命令を実行して、装置に、

処理するために映像フレームを受信すること、及び、

映像フレームの1つ又は複数のコード化単位を生成すること

を実行させるようにさらに構成される、条項17に記載の装置。

20. パレット予測子の各パレットエントリが、対応する再利用フラグを有し、

パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、条項17～19のいずれか一項に記載の装置。

20

21. 映像データ処理を実行するための装置であって、

命令を記憶するように構成されたメモリと、

メモリに結合されたプロセッサとを含み、プロセッサは、命令を実行して、装置に、

1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することを実行させるように構成され、

1つ又は複数のパレット予測子のうちのパレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、

第1の範囲であって、前のパレット予測子の第1のパレットエントリから始まり、前のパレット予測子の、再利用フラグセットを有する第1のパレットエントリで終わる第1の範囲内にある、前のパレット予測子の1つ又は複数のパレットエントリを、パレット予測子の第2の1つ又は複数のエントリのセットとしてパレット予測子に追加すること、及び、

30

第2の範囲であって、前のパレット予測子の、再利用フラグセットを有する最後のパレットエントリから始まり、前のパレット予測子の最後のパレットエントリで終わる第2の範囲内にある、前のパレット予測子の1つ又は複数のパレットエントリを、パレット予測子の第3の1つ又は複数のエントリのセットとしてパレット予測子に追加すること、
によって更新され、第2のエントリのセット及び第3のエントリのセットが、第1のエントリのセットの後にある、装置。

22. パレット予測子の第1のエントリのセット及び第2のエントリのセットの各パレットエントリが、再利用フラグを含む、条項21に記載の装置。

40

23. プロセッサが、命令を実行して、装置に、

処理するために映像フレームを受信すること、及び、

映像フレームの1つ又は複数のコード化単位を生成すること

を実行させるようにさらに構成される、条項21に記載の装置。

24. パレット予測子の各パレットエントリが、対応する再利用フラグを有し、

パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、条項21～23のいずれか一項に記載の装置。

25. 映像データ処理を実行するための装置であって、

命令を記憶するように構成されたメモリと、

50

メモリに結合されたプロセッサとを含み、プロセッサは、命令を実行して、装置に、
処理するために映像フレームを受信すること、
映像フレームの1つ又は複数のコード化単位を生成すること、及び、
パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数の
のコード化単位を処理すること
を実行させるように構成され、
1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、
各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定
される、装置。

26. プロセッサが、命令を実行して、装置に、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセット
として追加すること、及び、

現パレットで再利用されない前のパレット予測子からのエントリを、パレット予測子
の第2のエントリのセットであって、第1のエントリのセットの後にある第2のエントリ
のセットとして追加すること、

によってパレット予測子を更新することを実行させるようにさらに構成される、条項2
5に記載の装置。

27. 固定数が、スライスタイプ及びデュアルツリーモード設定に基づいて設定される
、条項25に記載の装置。

28. 1つ又は複数のパレット予測子のうちの1つのパレット予測子のサイズが、非ウ
ェーブフロントの場合にはスライスの初めに、事前定義された値に初期化される、条項2
5に記載の装置。

29. 1つ又は複数のパレット予測子のうちの1つのパレット予測子のサイズが、ウェ
ーブフロントの場合にはコード化単位行の初めに、事前定義された値に初期化される、条
項25に記載の装置。

30. プロセッサが、命令を実行して、装置に、

再利用フラグをシグナリングするときに、パレット予測子のサイズの値がパレット予測
子の最大サイズに等しいというビットストリーム適合性を追加すること

を実行させるようにさらに構成される、条項28又は29に記載の装置。

31. 1つ又は複数のパレット予測子を初期化するとき、各エントリの値及び各成分が
、0又は $(1 << (\text{シーケンスビット深度} - 1))$ に設定される、条項28～30のいず
れか一項に記載の装置。

32. プロセッサが、命令を実行して、装置に、

再利用フラグの2値化値に範囲制約を加えること

を実行させるようにさらに構成される、条項25～31のいずれか一項に記載の装置。

33. 命令のセットを記憶する非一時的コンピュータ可読媒体であって、装置に映像デ
ータ処理を実行する方法を開始させるために、装置の1つ又は複数のプロセッサによって
実行可能であり、方法が、

1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理する
ことを含み、

1つ又は複数のパレット予測子のうちのパレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセット
として追加すること、及び、

前のパレット予測子からのパレットエントリを、前のパレット予測子のパレットエン
トリの再利用フラグの値に関わらず、パレット予測子の第2のエントリのセットであって
、第1のエントリのセットの後にある第2のエントリのセットとして追加すること

によって更新される、非一時的コンピュータ可読媒体。

34. パレット予測子の第1のエントリのセット及び第2のエントリのセットの各パレ
ットエントリが、再利用フラグを含む、条項33に記載の非一時的コンピュータ可読媒体。

35. 方法が、

10

20

30

40

50

処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することと、

をさらに含む、条項33に記載の非一時的コンピュータ可読媒体。

36. パレット予測子の各パレットエントリが、対応する再利用フラグを有し、

パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、条項33～35のいずれか一項に記載の非一時的コンピュータ可読媒体。

37. 命令のセットを記憶する非一時的コンピュータ可読媒体であって、装置に映像データ処理を実行する方法を開始させるために、装置の1つ又は複数のプロセッサによって実行可能であり、方法は、

1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することを含み、

1つ又は複数のパレット予測子のうちのパレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、

第1の範囲であって、前のパレット予測子の第1のパレットエントリから始まり、前のパレット予測子の、再利用フラグセットを有する第1のパレットエントリで終わる第1の範囲内にある、前のパレット予測子の1つ又は複数のパレットエントリを、パレット予測子の第2の1つ又は複数のエントリのセットとしてパレット予測子に追加すること、及び、

第2の範囲であって、前のパレット予測子の、再利用フラグセットを有する最後のパレットエントリから始まり、前のパレット予測子の最後のパレットエントリで終わる、第2の範囲内にある、前のパレット予測子の1つ又は複数のパレットエントリを、パレット予測子の第3の1つ又は複数のエントリのセットとしてパレット予測子に追加すること、

によって更新され、第2のエントリのセット及び第3のエントリのセットが、第1のエントリのセットの後にある、非一時的コンピュータ可読媒体。

38. パレット予測子の第1のエントリのセット及び第2のエントリのセットの各パレットエントリが、再利用フラグを含む、条項37に記載の非一時的コンピュータ可読媒体。

39. 方法が、

処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することと、

をさらに含む、条項37に記載の非一時的コンピュータ可読媒体。

40. パレット予測子の各パレットエントリが、対応する再利用フラグを有し、

パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、条項37～39のいずれか一項に記載の非一時的コンピュータ可読媒体。

41. 命令のセットを記憶する非一時的コンピュータ可読媒体であって、装置に映像データ処理を実行する方法を開始させるために、装置の1つ又は複数のプロセッサによって実行可能であり、方法は、

処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することと、

パレットエントリを有する1つ又は複数のパレット予測子を使用して、1つ又は複数のコード化単位を処理することと、

を含み、

1つ又は複数のパレット予測子の各パレットエントリが、対応する再利用フラグを有し、

各パレット予測子の再利用フラグの数が、対応するコード化単位に対して固定数に設定される、非一時的コンピュータ可読媒体。

42. パレット予測子が、

現パレットのすべてのパレットエントリをパレット予測子の第1のエントリのセットとして追加すること、及び、

現パレットで再利用されない前のパレット予測子からのエントリを、パレット予測子の

10

20

30

40

50

第2のエントリのセットであって、第1のエントリのセットの後にある第2のエントリのセットとして追加すること、

によって更新される、条項41に記載の非一時的コンピュータ可読媒体。

43. 固定数が、スライスタイプ及びデュアルツリーモード設定に基づいて設定される、条項41に記載の非一時的コンピュータ可読媒体。

44. 1つ又は複数のパレット予測子のうちの1つのパレット予測子のサイズが、非ウェブフロントの場合にはスライスの初めに、事前定義された値に初期化される、条項41に記載の非一時的コンピュータ可読媒体。

45. 1つ又は複数のパレット予測子のうちの1つのパレット予測子のサイズが、ウェブフロントの場合にはコード化単位行の初めに、事前定義された値に初期化される、条項41に記載の非一時的コンピュータ可読媒体。

46. 再利用フラグをシグナリングするときに、パレット予測子のサイズの値がパレット予測子の最大サイズに等しいというビットストリーム適合性を追加すること

をさらに含む、条項44又は45に記載の非一時的コンピュータ可読媒体。

47. 1つ又は複数のパレット予測子を初期化するとき、各エントリの値及び各成分が、0又は $(1 < (シーケンスビット深度 - 1))$ に設定される、条項44～46のいずれか一項に記載の非一時的コンピュータ可読媒体。

48. 再利用フラグの2値化値に範囲制約を加えること

をさらに含む、条項41～47のいずれか一項に記載の非一時的コンピュータ可読媒体。

49. パレットモードのフィルタをデブロッキングするための方法であって、

処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することであって、前記1つ又は複数のコード化単位のそれぞれのコード化単位が、1つ又は複数のコード化ブロックを有することと、

隣接する2つのコード化ブロックのうちの少なくとも第1のコード化ブロックが、パレットモードでコード化されていることに応答して、境界フィルタ強度を2に設定することと、

を含む方法。

50. パレットモードのフィルタをデブロッキングするための方法であって、

処理するために映像フレームを受信することと、

映像フレームの1つ又は複数のコード化単位を生成することであって、前記1つ又は複数のコード化単位のそれぞれのコード化単位が、1つ又は複数のコード化ブロックを有することと、

隣接する2つのコード化ブロックのうちの少なくとも第1のコード化ブロックがパレットモードでコード化され、隣接する2つのコード化ブロックのうちの第2のコード化ブロックがパレットモードとは異なるコード化モードを有することに応答して、境界フィルタ強度を1に設定することと、

を含む方法。

51. 映像データ処理を実行するための装置であって、

命令を記憶するように構成されたメモリと、

メモリに結合されたプロセッサとを含み、プロセッサは、命令を実行して、装置に、

処理するために映像フレームを受信すること、

映像フレームの1つ又は複数のコード化単位を生成することであって、前記1つ又は複数のコード化単位のそれぞれのコード化単位が、1つ又は複数のコード化ブロックを有すること、及び、

隣接する2つのコード化ブロックのうちの少なくとも第1のコード化ブロックがパレットモードでコード化され、隣接する2つのコード化ブロックのうちの第2のコード化ブロックがパレットモードとは異なるコード化モードを有することに応答して、境界フィルタ強度を1に設定すること

を実行させるように構成される、装置。

10

20

30

40

50

５２．命令のセットを記憶する非一時的コンピュータ可読媒体であって、装置に映像データ処理を実行する方法を開始させるために、装置の１つ又は複数のプロセッサによって実行可能であり、方法は、

処理するために映像フレームを受信することと、

映像フレームの１つ又は複数のコード化単位を生成することであって、前記１つ又は複数のコード化単位のそれぞれのコード化単位が、１つ又は複数のコード化ブロックを有することと、

隣接する２つのコード化ブロックのうちの少なくとも第１のコード化ブロックがパレットモードでコード化され、隣接する２つのコード化ブロックのうちの第２のコード化ブロックがパレットモードとは異なるコード化モードを有することに応答して、境界フィルタ強度を１に設定することと、

を含む、非一時的コンピュータ可読媒体。

【０１１１】

〔０１３７〕 上記の明細書では、実装形態ごとに変わり得る多数の具体的な詳細に関して実施形態を説明してきた。記載されている実施形態に対して、ある一定の適合及び修正を行うことができる。本明細書を検討し、本明細書に開示されている発明を実践することで、他の実施形態が当業者に明らかになる場合がある。本明細書及び例は、単なる例示として見なされ、本発明の真の範囲及び趣旨は、以下の特許請求の範囲によって示されることが意図されている。図に示されているステップの順序が単に説明目的であることもまた意図されており、任意の特定のステップの順序に限定されることは意図されていない。そのため、当業者は、これらのステップが同じ方法を実装しながら異なる順序で実行可能であることを理解することができる。

【０１１２】

〔０１３８〕 図面及び本明細書では、例示的な実施形態を開示してきた。しかしながら、これらの実施形態には多くの変形及び修正を加えることができる。それに相応して、特定の用語が用いられているが、それらの用語は、全般的及び説明的な意味合いにおいてのみ使用されており、限定目的で使用されていない。

10

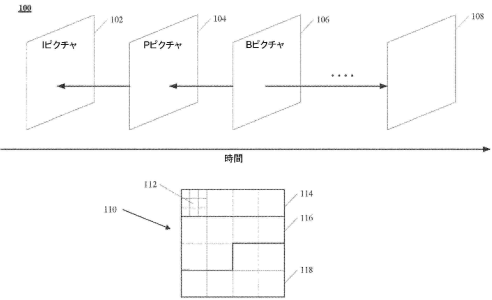
20

30

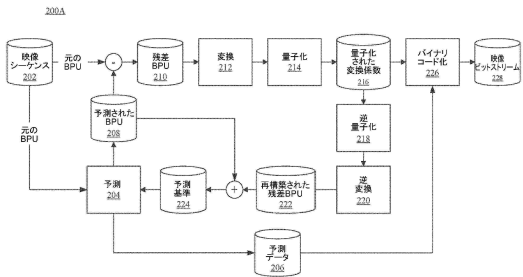
40

50

【図面】
【図 1】

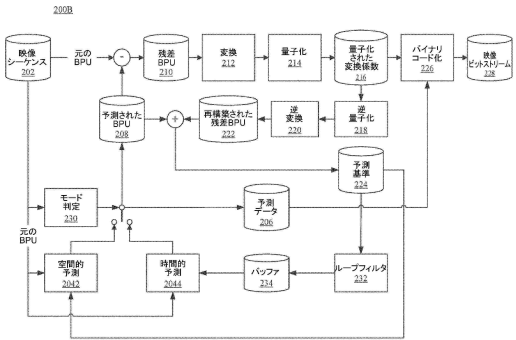


【図 2 A】

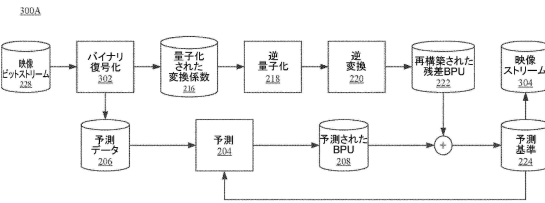


10

【図 2 B】



【図 3 A】



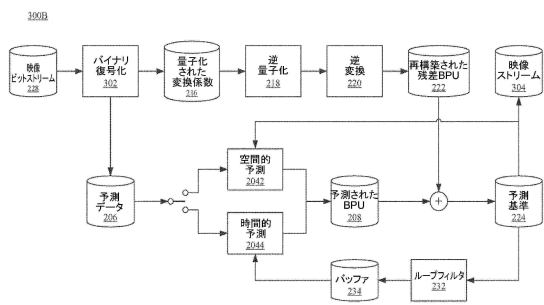
20

30

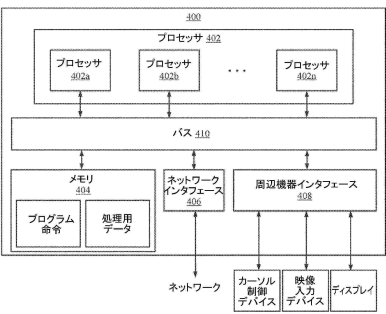
40

50

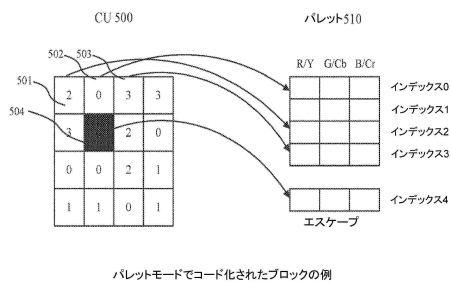
【図 3 B】



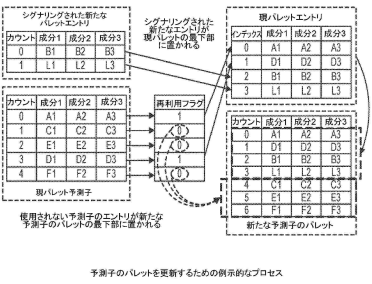
【図 4】



【図 5】



【図 6】



パレットモードでコード化されたブロックの例

予測子のパレットを更新するための例示的なプロセス

600

10

20

30

40

50

【図 7】

	記述子
	palette_coding(x0, y0, cbWidth, cbHeight, treeType) {
	startComp = (treeType == DUAL_TREE_CHROMA) ? 1 : 0
	numComps = (treeType == SINGLE_TREE) ? (ChromaArrayType == 0 ? 1 :
	3) : (treeType == DUAL_TREE_CHROMA) ? 2 : 1
	maxNumPaletteEntries = (treeType == SINGLE_TREE) ? 31 : 15
	palettePredictionFinished = 0
	NumPredictedPaletteEntries = 0
704	for(predictorEntryIdx = 0; predictorEntryIdx <
	PredictorPaletteSize[startComp] 1 && ! palettePredictionFinished &&
	NumPredictedPaletteEntries < maxNumPaletteEntries; predictorEntryIdx++) {
701	palette_predictor_run
	if(palette_predictor_run != 1) {
	if(palette_predictor_run > 1)
	predictorEntryIdx += palette_predictor_run - 1
	PalettePredictorEntryReuseFlags[predictorEntryIdx] = 1
	NumPredictedPaletteEntries++
	} else
	palettePredictionFinished = 1
	}
	if(NumPredictedPaletteEntries < maxNumPaletteEntries)
702	num_signalled_palette_entries
	for(cldx = startComp; cldx < (startComp + numComps); cldx++)
	for(i = 0; i < num_signalled_palette_entries; i++)
703	new_palette_entries[cldx][i]

【図 8】

800

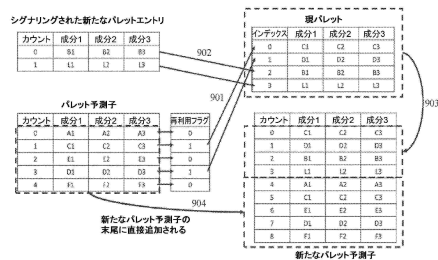
現パレットのすべてのパレットエントリをパレット
予測子の第1のエントリのセットとして追加する

802

前のパレット予測子からのパレットエントリを、
再利用フラグの値に関わらず、パレット予測子の
第2のエントリのセットとして追加し、第2のエントリの
セットが第1のエントリのセットの後にある

804

【図 9】



【図 10】

以下の条件のうちの1つが真であるとき:	
- cldxは0に等しい及びnumCompsは1に等しい;	
- cldxは2に等しい;	
値PredictorPaletteSize[startComp], 及びアレイPredictorPaletteEntriesが以下のように 導出又は修正される。	
for(i = 0; i < CurrentPaletteSize[startComp]; i++)	
for(cldx = startComp; cldx < (startComp + numComps); cldx++)	
newPredictorPaletteEntries[cldx][i] = CurrentPaletteEntries[cldx][i]	
newPredictorPaletteSize = CurrentPaletteSize[startComp]	
for(i = 0; i < PredictorPaletteSize[startComp] && newPredictorPaletteSize <	
maxNumPalettePredictorSize; i++)	
if(!PalettePredictorEntryReuseFlags[i]) {	-
for(cldx = startComp; cldx < (startComp + numComps); cldx++)	(457)
newPredictorPaletteEntries[cldx][newPredictorPaletteSize] =	
PredictorPaletteEntries[cldx][i]	
newPredictorPaletteSize++	
}	-
for(cldx = startComp; cldx < (startComp + numComps); cldx++)	
for(i = 0; i < newPredictorPaletteSize; i++)	
PredictorPaletteEntries[cldx][i] = newPredictorPaletteEntries[cldx][i]	
PredictorPaletteSize[startComp] = newPredictorPaletteSize	
qtbitt_dual_tree_intra_flagが0に等しいか、又はslice_typeが1に等しくないとき、以下のことがあてはまる:	
PredictorPaletteSize[1] = newPredictorPaletteSize	(458)
PredictorPaletteSize[startComp]の値が0からmaxNumPalettePredictorSizeまでの 間端値を含む範囲にあることが、ビットストリーム適合性の要件である。	

10

20

30

40

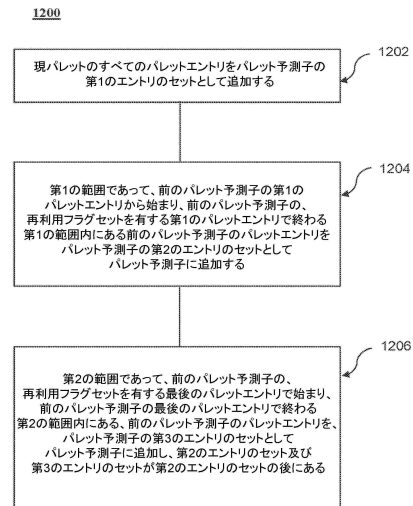
50

【図 1 1】

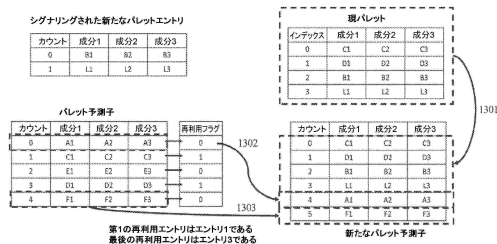
```

以下の条件のうちの1つが真であるとき、
-cldxは0に等しい及vnumCompsは1に等しい、
-cldxは2に等しい、
値PredictorPaletteSize[startComp]、及びアレIPredictorPaletteEntriesが以下のよう
に導出又は修正される。
for (i = 0; i < CurrentPaletteSize; i++)
    for (cldx = startComp; cldx < (startComp + numComps); cldx++)
        newPredictorPaletteEntries[cldx][i] = CurrentPaletteEntries[cldx][i]
newPredictorPaletteSize = CurrentPaletteSize; startComp =
for (i = 0; i < PredictorPaletteSize; i++)
    for (cldx = startComp; cldx < (startComp + numComps); cldx++)
        newPredictorPaletteEntries[cldx][i] = CurrentPaletteEntries[cldx][i]
        if (!PalettePredictorEntryReuseFinished) {
            for (cldx = startComp; cldx < (startComp + numComps); cldx++)
                newPredictorPaletteEntries[cldx][i] = PredictorPaletteEntries[cldx][i]
            newPredictorPaletteSize++
        }
for (cldx = startComp; cldx < (startComp + numComps); cldx++)
    for (i = 0; i < newPredictorPaletteSize; i++)
        PredictorPaletteEntries[cldx][i] = newPredictorPaletteEntries[cldx][i]
PredictorPaletteSize = newPredictorPaletteSize
qbit_dual_tree_intra_flagが0に等しいか、又はslice_typeが1に等しくないとき、
以下が当てはまる。
PredictorPaletteSize[1] = newPredictorPaletteSize
PredictorPaletteSize[startComp]の値が0から
maxNumPalettePredictorSizeまでの両端値を含む範囲にあることが、
ビットストリーム適合性の要件である。
    
```

【図 1 2】



【図 1 3】



【図 1 4】

```

palette_coding(x0, y0, cbWidth, cbHeight, treeType) {
    startComp = (treeType == DUAL_TREE_CHROMA) ? 1 : 0
    numComps = (treeType == SINGLE_TREE) ? (ChromaArrayType == 0 ? 1 : 3) : (treeType == DUAL_TREE_CHROMA) ? 2 : 1
    maxNumPaletteEntries = (treeType == SINGLE_TREE) ? 31 : 15
    palettePredictionFinished = 0
    NumPredictedPaletteEntries = 0
    firstReuseEntry = -1
    lastReuseEntry = -1
    for predictorEntryIdx = 0; predictorEntryIdx <
        PredictorPaletteSize[startComp] && !palettePredictionFinished &&
        NumPredictedPaletteEntries < maxNumPaletteEntries; predictorEntryIdx++
    {
        palette_predictor_run
        if (palette_predictor_run != 1) {
            if (palette_predictor_run > 1)
                predictorEntryIdx += palette_predictor_run - 1
            PalettePredictorEntryReuseFlag[predictorEntryIdx-1] =
                if (firstReuseEntry == -1)
                    firstReuseEntry = predictorEntryIdx
                if (lastReuseEntry < predictorEntryIdx)
                    lastReuseEntry = predictorEntryIdx
            NumPredictedPaletteEntries++
        } else
            palettePredictionFinished = 1
    }
    if (NumPredictedPaletteEntries == 0) {
        firstReuseEntry = 0
        lastReuseEntry = PredictorPaletteSize[startComp]
    }
    if (NumPredictedPaletteEntries < maxNumPaletteEntries)
        num_signalled_palette_entries
        for cldx = startComp; cldx < (startComp + numComps); cldx++
        for (i = 0; i < num_signalled_palette_entries; i++)
            new_palette_entries[cldx][i]
    }
    
```

10

20

30

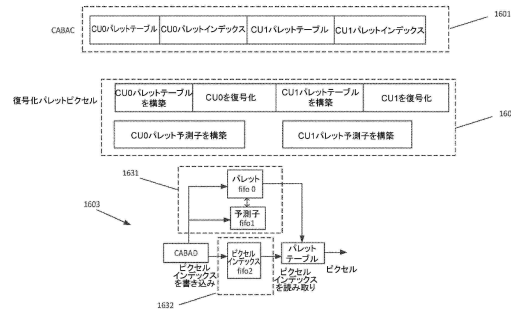
40

50

【図 15】

1502	以下の条件のうちの1つが真であるとき	
	-cldxは0に等しい及びnumCompsは1に等しい;	
	-cldxは2に等しい;	
	値PredictorPaletteSize[startComp]、及びアレイPredictorPaletteEntriesが以下のよう に導出又は修正される:	
	for (i = 0; i < CurrentPaletteSize[startComp]; i++)	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	
	newPredictorPaletteEntries[cldx][i] = CurrentPaletteEntries[cldx][i]	
	newPredictorPaletteSize = CurrentPaletteSize[startComp]	
	for (i = 0; i < firstReuseEntry && newPredictorPaletteSize < maxNumPalettePredictorSize; i++) {	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	
	newPredictorPaletteEntries[cldx][i] = newPredictorPaletteSize = PredictorPaletteEntries[cldx][i]	
	newPredictorPaletteSize++	
	}	
	for (i = lastReuseEntry + 1; i < PredictorPaletteSize[startComp] && newPredictorPaletteSize < maxNumPalettePredictorSize; i++) {	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	
	newPredictorPaletteEntries[cldx][i] = newPredictorPaletteSize = PredictorPaletteEntries[cldx][i]	
	newPredictorPaletteSize++	
	}	
	for (i = 0; i < PredictorPaletteSize[startComp] && newPredictorPaletteSize < maxNumPalettePredictorSize; i++)	
	if (!PalettePredictorEntryReuseFlags[i]) {	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	(457)
	newPredictorPaletteEntries[cldx][i] = newPredictorPaletteSize = PredictorPaletteEntries[cldx][i]	
	newPredictorPaletteSize++	
	}	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	
	for (i = 0; i < newPredictorPaletteSize; i++)	
	PredictorPaletteEntries[cldx][i] = newPredictorPaletteEntries[cldx][i]	
	PredictorPaletteSize[startComp] = newPredictorPaletteSize	
	qtbtt_dual_tree_intra_flagが0に等しいか、又はslice_typeが1に等しくないとき、 以下が当てはまる:	
	PredictorPaletteSize[i] = newPredictorPaletteSize	(458)
	PredictorPaletteSize[startComp]の値が0からmaxNumPalettePredictorSizeまでの 両端値を含む範囲にあることが、 ビットストリーム適合性の要件である。	

【図 16】



10

20

【図 17】

1702	以下の条件のうちの1つが真であるとき	
	-cldxは0に等しい及びnumCompsは1に等しい;	
	-cldxは2に等しい;	
	値PredictorPaletteSize[startComp]、及びアレイPredictorPaletteEntriesが以下のよう に導出又は修正される:	
	for (i = 0; i < CurrentPaletteSize[startComp]; i++)	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	
	newPredictorPaletteEntries[cldx][i] = CurrentPaletteEntries[cldx][i]	
	newPredictorPaletteSize = CurrentPaletteSize[startComp]	
	for (i = 0; i < PredictorPaletteSize[startComp] && newPredictorPaletteSize < maxNumPalettePredictorSize; i++)	
	if (!PalettePredictorEntryReuseFlags[i]) {	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	(457)
	newPredictorPaletteEntries[cldx][i] = newPredictorPaletteSize = PredictorPaletteEntries[cldx][i]	
	newPredictorPaletteSize++	
	}	
	for (cldx = startComp; cldx < (startComp + numComps); cldx++)	
	for (i = 0; i < newPredictorPaletteSize; i++)	
	PredictorPaletteEntries[cldx][i] = newPredictorPaletteEntries[cldx][i]	
	PredictorPaletteSize[startComp] = newPredictorPaletteSize	
	qtbtt_dual_tree_intra_flagが0に等しいか、又はslice_typeが1に等しくないとき、 以下が当てはまる:	
	PredictorPaletteSize[i] = newPredictorPaletteSize	(458)
	PredictorPaletteSize[startComp]の値が0からmaxNumPalettePredictorSizeに等しい ことが、ビットストリーム適合性の要件である。	
1701	PredictorPaletteSize[startComp]の値が0から maxNumPalettePredictorSizeまでの両端値を含む範囲にあることが、 ビットストリーム適合性の要件である。	

【図 18】

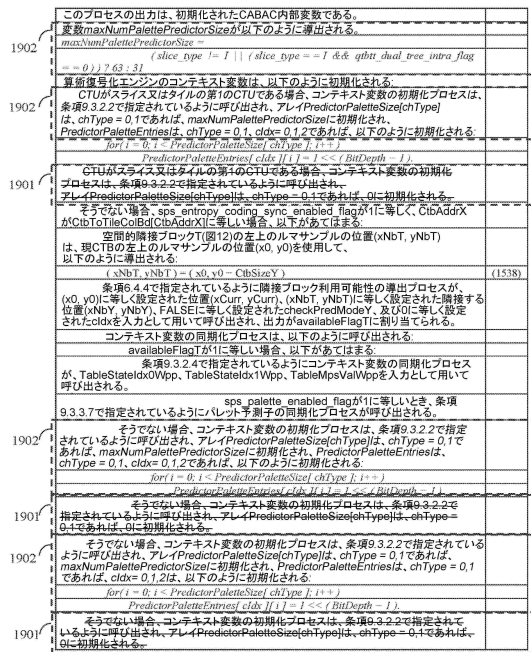
	このプロセスの出力は、初期化されたCABAC内部変数である。	
	算術復号化エンジンのコンテキスト変数は、以下のように初期化される:	
	CTUがスライス又はタイルの第1のCTUである場合、コンテキスト変数の 初期化プロセスは、条項9.3.2.2で指定されているように呼び出され、アレイ PredictorPaletteSize[chType]は、chType = 0, 1であれば、0に初期化される。	
	そうでない場合、sps_entropy_coding_sync_enabled_flagが1に等しく、 CtbAddrXがCtbToTileColBd[CtbAddrX]に等しい場合、以下が当てはまる:	
	空間的隣接ブロックT(図12)の左上のルマサンプルの位置 (xNbt, yNbt)は、現CTBの左上のルマサンプルの位置(x0, y0)を使用して、 以下のように導出される:	
	(xNbt, yNbt) = (x0, y0 - CtbSizeY)	(1538)
	条項6.4.4で指定されているように隣接ブロック利用可能性の導出 プロセスが、(x0, y0)に等しく設定された位置(xCurr, yCurr)、(xNbt, yNbt)に 等しく設定された隣接する位置(xNdy, yNdy)、FALSEに等しく設定された checkPredModeY、及び0に等しく設定されたcldxを入力として用いて 呼び出され、出力がavailableFlagTiに割り当てられる。	
	コンテキスト変数の同期化プロセスは、以下のように呼び出される:	
	availableFlagTiが1に等しい場合、以下が当てはまる:	
	条項9.3.2.4で指定されているようにコンテキスト変数の同期化 プロセスが、TableStateIdx0Wpp、TableStateIdx1Vpp、TableMpsValVppを 入力として用いて呼び出される。	
	sps_palette_enabled_flagが1に等しいとき、条項9.3.3.7で 指定されているようにパレット予測子の同期化プロセスが呼び出される。	
	そうでない場合、コンテキスト変数の初期化プロセスは、 条項9.3.2.2で指定されているように呼び出され、アレイ PredictorPaletteSize[chType]は、chType = 0, 1であれば、0に初期化される。	
	そうでない場合、コンテキスト変数の初期化プロセスは、条項9.3.2.2で 指定されているように呼び出され、アレイPredictorPaletteSize[chType]は、 chType = 0, 1であれば、0に初期化される。	

30

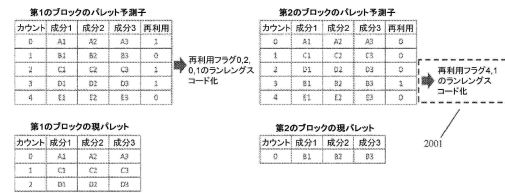
40

50

【図 19】



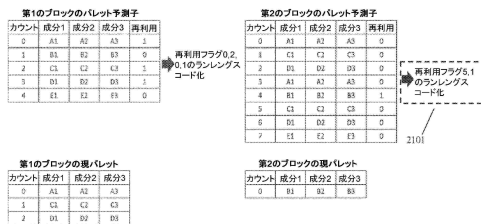
【図 20】



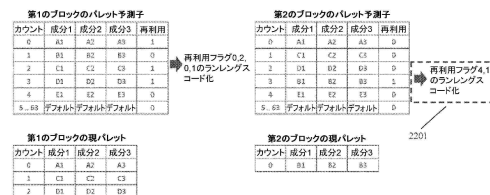
10

20

【図 21】



【図 22】



30

40

50

【図 2 3】

	palette_coding(s0, v0, cbWidth, cbHeight, treeType) {	記述子
	startComp = (treeType == DUAL_TREE_CHROMA) ? 1 : 0	
	numComps = (treeType == SINGLE_TREE) ? (ChromaArrayType == 0 ? 1 : 3) : (treeType == DUAL_TREE_CHROMA) ? 2 : 1	
	maxNumPaletteEntries = (treeType == SINGLE_TREE) ? 31 : 15	
2302	maxNumPalettePredictorSize = (slice_type != I) ? (slice_type == I && qbt_dual_tree_intra_flag == 0) ? 63 : 31	
	palettePredictionFinished = 0	
	NumPredictedPaletteEntries = 0	
2302	for(predictorEntryIdx = 0; predictorEntryIdx < maxNumPalettePredictorSize && !palettePredictionFinished && NumPredictedPaletteEntries < maxNumPaletteEntries; predictorEntryIdx++) {	
	for(predictorEntryIdx = 0; predictorEntryIdx < PredictorPaletteSize[startComp] && !palettePredictionFinished && NumPredictedPaletteEntries < maxNumPaletteEntries; predictorEntryIdx++) {	
2301	palette_predictor_run = 0	ae(v)
	if(palette_predictor_run != 1) {	
	if(palette_predictor_run > 1)	
	predictorEntryIdx += palette_predictor_run - 1	
	PalettePredictorEntryReuseFlags[predictorEntryIdx] = 1	
	NumPredictedPaletteEntries++	
	} else	
	palettePredictionFinished = 1	
	}	
	if(NumPredictedPaletteEntries < maxNumPaletteEntries)	
	num_signalled_palette_entries	ae(v)
	for(cidx = startComp; cidx < (startComp + numComps); cidx++)	
	for(i = 0; i < num_signalled_palette_entries; i++)	
	new_palette_entries[cidx][i]	ae(v)

【図 2 4】

palette_predictor_runを使用して、アレイPalettePredictorEntryReuseFlagsにおいて非ゼロエントリに先行するゼロの数を判定する。	
palette_predictor_runの値が0から(PredictorPaletteSize[startComp]-predictorEntryIdx)までの両端値を含む範囲にあることが、ビットストリーム適合性の要件である。なお、predictorEntryIdxは、アレイPalettePredictorEntryReuseFlagsにおける現在の位置に対応する。変数NumPredictedPaletteEntriesは、予測パレットから再利用されている現パレットのエントリの数を指定する。NumPredictedPaletteEntriesの値は、0からmaxNumPaletteEntriesまでの両端値を含む範囲にある。	

10

【図 2 5】

	palette_predictor_runを使用して、アレイPalettePredictorEntryReuseFlagsにおいて非ゼロエントリに先行するゼロの数を判定する。	
2502	(PredictorPaletteSize[startComp]-predictorEntryIdx)が0に等しい場合、palette_predictor_runの値は1になる。そうでない場合、(PredictorPaletteSize[startComp]-predictorEntryIdx)が0より大きい場合、palette_predictor_runの値が0から(PredictorPaletteSize[startComp]-predictorEntryIdx)までの両端値を含む範囲にあることが、ビットストリーム適合性の要件である。なお、predictorEntryIdxは、アレイPalettePredictorEntryReuseFlagsにおける現在の位置に対応する。変数NumPredictedPaletteEntriesは、予測パレットから再利用されている現パレットのエントリの数を指定する。NumPredictedPaletteEntriesの値は、0からmaxNumPaletteEntriesまでの両端値を含む範囲にある。	
2501	palette_predictor_runの値が0から(PredictorPaletteSize[startComp]-predictorEntryIdx)までの両端値を含む範囲にあることが、ビットストリーム適合性の要件である。なお、predictorEntryIdxは、アレイPalettePredictorEntryReuseFlagsにおける現在の位置に対応する。変数NumPredictedPaletteEntriesは、予測パレットから再利用されている現パレットのエントリの数を指定する。NumPredictedPaletteEntriesの値は、0からmaxNumPaletteEntriesまでの両端値を含む範囲にある。	

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)

(72)発明者 リャオ, ルーリン

中華人民共和国, ジャー جان 3 1 1 1 2 1, ハン チョウ, ユ ハン ディストリクト, ウェスト
ウェン イ ロード ナンバー 9 6 9 ビルディング 3 5 / エフ, アリババ グループ リーガル デ
パートメント

(72)発明者 サーヴァー, モハメッド, ゴラム

アメリカ合衆国, カリフォルニア州 9 4 4 0 2, サン マテオ, サウス エル カミーノ レアル
4 0 0, スイート 4 0 0

(72)発明者 イエ, ヤン

アメリカ合衆国, カリフォルニア州 9 4 4 0 2, サン マテオ, サウス エル カミーノ レアル
4 0 0, スイート 4 0 0

(72)発明者 ファン, シュアン

中華人民共和国, ジャー ジャン 3 1 1 1 2 1, ハン チョウ, ユ ハン ディストリクト, ウェスト
ウェン イ ロード ナンバー 9 6 9 ビルディング 3 5 / エフ, アリババ グループ リーガル デ
パートメント

審査官 岩井 健二

(56)参考文献 特表 2 0 1 7 - 5 2 2 8 0 4 (J P , A)

特表 2 0 1 7 - 5 1 9 4 6 0 (J P , A)

米国特許出願公開第 2 0 2 0 / 0 0 9 2 5 4 6 (U S , A 1)

Ru-Ling Liao, Yan Ye, and Mohammed Golam Sarwer, AHG11: Fixed number of reuse flags
for palette mode, Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC
1/SC 29/WG 11, JVET-R0229-v3, 18th Meeting: by teleconference, 2020年04月, pp.1-5

(58)調査した分野 (Int.Cl., D B 名)

H 0 4 N 1 9 / 0 0 - 1 9 / 9 8