

(19)日本国特許庁(JP)

(12)公表特許公報(A)

(11)公表番号
特表2024-534816
(P2024-534816A)

(43)公表日 令和6年9月26日(2024.9.26)

(51)国際特許分類
H 0 4 N 19/70 (2014.01)

F I
H 0 4 N 19/70

テーマコード (参考)
5 C 1 5 9

審査請求 未請求 予備審査請求 未請求 (全51頁)

(21)出願番号 (86)(22)出願日 (85)翻訳文提出日 (86)国際出願番号 (87)国際公開番号 (87)国際公開日 (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	特願2024-510524(P2024-510524) 令和4年8月26日(2022.8.26) 令和6年2月20日(2024.2.20) PCT/US2022/075507 WO2023/028580 令和5年3月2日(2023.3.2) 63/260,600 令和3年8月26日(2021.8.26) 米国(US) 63/251,385 令和3年10月1日(2021.10.1) 米国(US) 63/262,078	(71)出願人 516227559 オッポ広東移動通信有限公司 GUANGDONG OPPO MOBI LE TELECOMMUNICATI ONS CORP., LTD. 中華人民共和國広東省東莞市長安鎮烏沙 海浜路18号 No. 18 Haibin Road, Wusha, Chang'an, Don gguan, Guangdong 52 3860 China (74)代理人 100120031 弁理士 宮嶋 学 (74)代理人 100107582 弁理士 関根 毅
最終頁に続く		最終頁に続く

(54)【発明の名称】 多用途ビデオコーディングのための操作範囲拡張

(57)【要約】

いくつかの実施形態において、ビデオデコーダーは、多用途ビデオコーディング（VVC）を利用してエンコードされたビデオビットストリームからビデオをデコードする。当該ビデオデコーダーは、値が0から8の範囲内にあるシーケンスパラメータセット（SPS）シンタックス要素sps_bitdepth_minus8に基づいて、当該ビデオのサンプルのビット深度を確定する。当該ビデオデコーダーはさらに、値が0から8の範囲内にあるビデオパラメータセット（VPS）シンタックス要素vps_ols_dpb_bitdepth_minus8に基づいて、デコードピクチャバッファ（DPB）のサイズを確定する。当該ビデオデコーダーは、確定されたDPBのサイズを有する記憶スペースを割り当て、確定されたビット深度に基づいて、ビデオビットストリームを複数のピクチャにデコードし、デコードされた複数のピクチャをDPBに記憶する。当該ビデオデコーダーはさらに、デコードされた複数のピクチャを表示のために出力する。

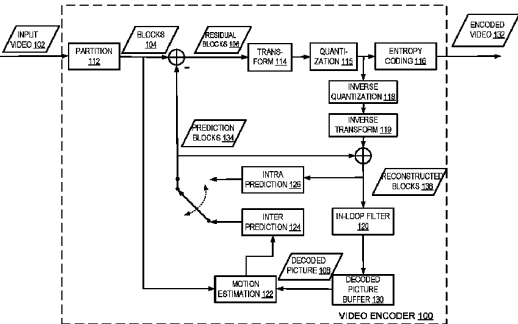


FIG. 1

【特許請求の範囲】

【請求項 1】

多用途ビデオコーディング（VVC）を利用してエンコードされたビデオビットストリームからビデオをデコードするための方法であって、前記方法は、

シーケンスパラメータセット（SPS）シンタックス要素 `sps__bitdepth__minus 8` に基づいて、前記ビデオのサンプルのビット深度を確定することであって、前記 SPS シンタックス要素 `sps__bitdepth__minus 8` の値が 0 から 8 の範囲内にある、確定することと、

ビデオパラメータセット（VPS）シンタックス要素 `vps__ols__dpb__bitdepth__minus 8` に基づいて、デコードピクチャバッファ（DPB）のサイズを確定することであって、前記 VPS シンタックス要素 `vps__ols__dpb__bitdepth__minus 8` の値が 0 から 8 の範囲内にある、確定することと、

確定された前記 DPB の前記サイズを有する記憶スペースを割り当てることと、

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることと、

デコードされた前記複数のピクチャの 1 つ又は複数を前記 DPB に記憶することと、

デコードされた前記複数のピクチャの前記 1 つ又は複数を表示のために出力することと、を含む、

ことを特徴とする VVC を利用してエンコードされたビデオビットストリームからビデオをデコードするための方法。

【請求項 2】

前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオビットストリームの第 1 の部分を第 1 のピクチャセットにデコードすることと、

前記第 1 のピクチャセットを前記 DPB に記憶することと、

前記 DPB における前記第 1 のピクチャセットを参照ピクチャとして利用することにより、前記ビデオビットストリームの第 2 の部分を第 2 のピクチャセットにデコードすることと、を含む、

ことを特徴とする請求項 1 に記載の方法。

【請求項 3】

前記方法は、

出力順序に従って、前記 DPB におけるデコードされた前記複数のピクチャの前記 1 つ又は複数を改めて並べることと、

前記出力順序に従って、デコードされた前記複数のピクチャの前記 1 つ又は複数を出力することと、をさらに含む、

ことを特徴とする請求項 1 に記載の方法。

【請求項 4】

前記方法は、

デコードされた前記複数のピクチャの前記 1 つ又は複数を前記 DPB に一定期間とどめておくことと、

前記一定期間経過後、デコードされた前記複数のピクチャの前記 1 つ又は複数を出力することと、をさらに含む、

ことを特徴とする請求項 1 に記載の方法。

【請求項 5】

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオのパーティションを表すバイナリ文字列にアクセスすることであって、前記パーティションが、1 つ又は複数のコーディングツリーユニット（CTU）行を形成する複数の CTU を含む、アクセスすることと、

前記パーティションにおける前記複数の CTU の各々に対して、

前記 C T U をデコードする前に、並列コーディングが有効にされたこと、及び前記 C T U が前記パーティションにおける前記 1 つ又は複数の C T U 行のうちの現在の C T U 行の 1 番目の C T U であることを確定することと、

前記並列コーディングが有効にされたこと、及び前記 C T U が前記パーティションにおける前記現在の C T U 行の前記 1 番目の C T U であることを確定することに応じて、

確定された前記ビット深度に基づいて、ライスパラメータを計算するための色成分の履歴カウンタを初期値に設定することと、

前記 C T U をデコードすることであって、

前記履歴カウンタに基づいて、前記 C T U における変換ユニット (T U) の前記ライスパラメータを計算することと、

計算された前記ライスパラメータに基づいて、前記 C T U における前記 T U に対応する前記バイナリ文字列を、前記 T U の係数値にデコードすることと、

前記係数値から前記 C T U における前記 T U のピクセル値を確定することと、を含む、前記 C T U をデコードすることと、

前記ビデオのデコードされたパーティションを出力することであって、デコードされた前記パーティションが、前記パーティションにおけるデコードされた前記複数の C T U を含む、出力することと、

を含む、

ことを特徴とする請求項 1 に記載の方法。

【請求項 6】

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオのパーティションを表すバイナリ文字列にアクセスすることであって、前記パーティションが、1 つ又は複数の C T U 行を形成する複数の C T U を含む、アクセスすることと、

前記パーティションにおける前記複数の C T U の各々に対して、

前記 C T U をデコードする前に、且つ、並列コーディングが有効にされたこと、及び前記 C T U が現在の C T U 行の 1 番目の C T U であることを確定することに応じて、前記現在の C T U 行が前記パーティションにおける 1 番目の C T U 行であるか否かを確定することと、

前記現在の C T U 行が前記パーティションにおける前記 1 番目の C T U 行であると確定することに応じて、確定された前記ビット深度に基づいて、ライスパラメータを計算するための色成分の履歴カウンタを初期値に設定することと、

前記現在の C T U 行が前記パーティションにおける前記 1 番目の C T U 行ではないと確定することに応じて、前記色成分の履歴カウンタを、履歴カウンタストレージ変数に記憶された値に設定することと、

前記 C T U をデコードすることであって、

前記履歴カウンタの前記値に基づいて、前記 C T U における T U の前記ライスパラメータを計算することと、

計算された前記ライスパラメータに基づいて、前記 C T U における前記 T U に対応する前記バイナリ文字列を、前記 T U の係数値にデコードすることと、

前記係数値から前記 C T U における前記 T U のピクセル値を確定することと、を含む、前記 C T U をデコードすることと、

前記 C T U をデコードした後、並列コーディングが有効にされたこと、及び前記 C T U が前記現在の C T U 行の前記 1 番目の C T U であることを確定することに応じて、前記履歴カウンタの現在の値を前記履歴カウンタストレージ変数に記憶することと、

を含む、

ことを特徴とする請求項 1 に記載の方法。

【請求項 7】

非一時的なコンピュータ可読媒体であって、前記非一時的なコンピュータ可読媒体にプ

10

20

30

40

50

ログラムコードが記憶されおり、

以下の操作を実行するように、前記プログラムコードは、1つ又は複数の処理デバイスによって実行可能であり、前記操作は、

ビデオのビデオビットストリームに含まれるビデオパラメータセット (VPS) シンタックス要素 `vps__ols__dpb__bitdepth__minus8` に基づいて、前記ビデオのデコードピクチャバッファ (DPB) のサイズを確定することであって、多用途ビデオコーディング (VVC) を利用して前記ビデオビットストリームがエンコードされ、前記 VPS シンタックス要素 `vps__ols__dpb__bitdepth__minus8` の値が 0 から 8 の範囲内にある、確定することと、

確定された前記 DPB の前記サイズを有する記憶スペースを割り当てることと、

前記ビデオビットストリームを複数のピクチャにデコードすることと、

デコードされた前記複数のピクチャの 1 つ又は複数を前記 DPB に記憶することと、

デコードされた前記複数のピクチャの前記 1 つ又は複数を表示のために出力することと

、

を含む、

ことを特徴とする非一時的なコンピュータ可読媒体。

【請求項 8】

前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオビットストリームの第 1 の部分を第 1 のピクチャセットにデコードすることと、

前記第 1 のピクチャセットを前記 DPB に記憶することと、

前記 DPB における前記第 1 のピクチャセットを参照ピクチャとして利用することにより、前記ビデオビットストリームの第 2 の部分を第 2 のピクチャセットにデコードすることと、を含む、

ことを特徴とする請求項 7 に記載の非一時的なコンピュータ可読媒体。

【請求項 9】

前記操作は、

出力順序に従って、前記 DPB におけるデコードされた前記複数のピクチャの前記 1 つ又は複数を改めて並べることと、

前記出力順序に従って、デコードされた前記複数のピクチャの前記 1 つ又は複数を出力することと、をさらに含む、

ことを特徴とする請求項 7 に記載の非一時的なコンピュータ可読媒体。

【請求項 10】

前記操作は、

デコードされた前記複数のピクチャの前記 1 つ又は複数を前記 DPB に一定期間とどめておくことと、

前記一定期間経過後、デコードされた前記複数のピクチャの前記 1 つ又は複数を出力することと、をさらに含む、

ことを特徴とする請求項 7 に記載の非一時的なコンピュータ可読媒体。

【請求項 11】

前記操作は、

シーケンスパラメータセット (SPS) シンタックス要素 `sps__bitdepth__minus8` に基づいて、前記ビデオのサンプルのビット深度を確定することをさらに含み、前記 SPS シンタックス要素 `sps__bitdepth__minus8` の値が 0 から 8 の範囲内にあり、前記ビデオビットストリームを前記複数のピクチャにデコードすることは、確定された前記ビット深度に基づいて実行される、

ことを特徴とする請求項 7 に記載の非一時的なコンピュータ可読媒体。

【請求項 12】

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることは、

10

20

30

40

50

前記ビデオのパーティションを表すバイナリ文字列にアクセスすることであって、前記パーティションが、1つ又は複数のコーディングツリーユニット（CTU）行を形成する複数のCTUを含む、アクセスすることと、

前記パーティションにおける前記複数のCTUの各々に対して、

前記CTUをデコードする前に、並列コーディングが有効にされたこと、及び前記CTUが前記パーティションにおける前記1つ又は複数のCTU行のうちの現在のCTU行の1番目のCTUであることを確定することと、

前記並列コーディングが有効にされたこと、及び前記CTUが前記パーティションにおける前記現在のCTU行の前記1番目のCTUであることを確定することに応じて、

確定された前記ビット深度に基づいて、ライスパラメータを計算するための色成分の履歴カウンタを初期値に設定することと、

前記CTUをデコードすることであって、

前記履歴カウンタに基づいて、前記CTUにおける変換ユニット（TU）の前記ライスパラメータを計算することと、

計算された前記ライスパラメータに基づいて、前記CTUにおける前記TUに対応する前記バイナリ文字列を、前記TUの係数値にデコードすることと、

前記係数値から前記CTUにおける前記TUのピクセル値を確定することと、を含む、前記CTUをデコードすることと、

前記ビデオのデコードされたパーティションを出力することであって、デコードされた前記パーティションが、前記パーティションにおけるデコードされた前記複数のCTUを含む、出力することと、

を含む、

ことを特徴とする請求項11に記載の非一時的なコンピュータ可読媒体。

【請求項13】

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオのパーティションを表すバイナリ文字列にアクセスすることであって、前記パーティションが、1つ又は複数のCTU行を形成する複数のCTUを含む、アクセスすることと、

前記パーティションにおける前記複数のCTUの各々に対して、

前記CTUをデコードする前に、且つ、並列コーディングが有効にされたこと、及び前記CTUが現在のCTU行の1番目のCTUであることを確定することに応じて、前記現在のCTU行が前記パーティションにおける1番目のCTU行であるか否かを確定することと、

前記現在のCTU行が前記パーティションにおける前記1番目のCTU行であると確定することに応じて、確定された前記ビット深度に基づいて、ライスパラメータを計算するための色成分の履歴カウンタを初期値に設定することと、

前記現在のCTU行が前記パーティションにおける前記1番目のCTU行ではないと確定することに応じて、前記色成分の履歴カウンタを、履歴カウンタストレージ変数に記憶された値に設定することと、

前記CTUをデコードすることであって、

前記履歴カウンタの前記値に基づいて、前記CTUにおけるTUの前記ライスパラメータを計算することと、

計算された前記ライスパラメータに基づいて、前記CTUにおける前記TUに対応する前記バイナリ文字列を、前記TUの係数値にデコードすることと、

前記係数値から前記CTUにおける前記TUのピクセル値を確定することと、を含む、前記CTUをデコードすることと、

前記CTUをデコードした後、並列コーディングが有効にされたこと、及び前記CTUが前記現在のCTU行の前記1番目のCTUであることを確定することに応じて、前記履歴カウンタの現在の値を前記履歴カウンタストレージ変数に記憶することと、

10

20

30

40

50

を含む、

ことを特徴とする請求項 1 1 に記載の非一時的なコンピュータ可読媒体。

【請求項 1 4】

システムであって、

処理デバイス及び非一時的なコンピュータ可読媒体を備え、

前記非一時的なコンピュータ可読媒体は前記処理デバイスに通信的に結合されており、
前記処理デバイスは、前記非一時的なコンピュータ可読媒体に記憶されたプログラムコードを実行するように構成されており、それによって、以下の操作を実行し、前記操作は、
ビデオのビデオビットストリームに含まれるビデオパラメータセット（VPS）シン
タックス要素 `vps__ols__dpb__bitdepth__minus8` に基づいて、前
記ビデオのデコードピクチャバッファ（DPB）のサイズを確定することであって、多用
途ビデオコーディング（VVC）を利用して前記ビデオビットストリームがエンコードさ
れ、前記 VPS シンタックス要素 `vps__ols__dpb__bitdepth__minus8` の値が 0 から 8 の範囲内にある、確定することと、

確定された前記 DPB の前記サイズを有する記憶スペースを割り当てることと、

前記ビデオビットストリームを複数のピクチャにデコードすることと、

デコードされた前記複数のピクチャの 1 つ又は複数を前記 DPB に記憶することと、

デコードされた前記複数のピクチャの前記 1 つ又は複数を表示のために出力すること
と、

を含む、

ことを特徴とするシステム。

【請求項 1 5】

前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオビットストリームの第 1 の部分を第 1 のピクチャセットにデコードすること
と、

前記第 1 のピクチャセットを前記 DPB に記憶することと、

前記 DPB における前記第 1 のピクチャセットを参照ピクチャとして利用することによ
り、前記ビデオビットストリームの第 2 の部分を第 2 のピクチャセットにデコードするこ
とと、を含む、

ことを特徴とする請求項 1 4 に記載のシステム。

【請求項 1 6】

前記操作は、

出力順序に従って、前記 DPB におけるデコードされた前記複数のピクチャの前記 1 つ
又は複数を改めて並べることと、

前記出力順序に従って、デコードされた前記複数のピクチャの前記 1 つ又は複数を出力
することと、をさらに含む、

ことを特徴とする請求項 1 4 に記載のシステム。

【請求項 1 7】

前記操作は、

デコードされた前記複数のピクチャの前記 1 つ又は複数を前記 DPB に一定期間とどめ
ておくことと、

前記一定期間経過後、デコードされた前記複数のピクチャの前記 1 つ又は複数を出力す
ることと、をさらに含む、

ことを特徴とする請求項 1 4 に記載のシステム。

【請求項 1 8】

前記操作は、

シーケンスパラメータセット（SPS）シンタックス要素 `sp_s__bitdepth__minus8` に基づいて、前記ビデオのサンプルのビット深度を確定することをさらに含
み、前記 SPS シンタックス要素 `sp_s__bitdepth__minus8` の値が 0 から
8 の範囲内にあり、前記ビデオビットストリームを前記複数のピクチャにデコードするこ

10

20

30

40

50

とは、確定された前記ビット深度に基づいて実行される、
ことを特徴とする請求項 14 に記載のシステム。

【請求項 19】

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオのパーティションを表すバイナリ文字列にアクセスすることであって、前記パーティションが、1つ又は複数のコーディングツリーユニット（CTU）行を形成する複数の CTU を含む、アクセスすることと、

前記パーティションにおける前記複数の CTU の各々に対して、

前記 CTU をデコードする前に、並列コーディングが有効にされたこと、及び前記 CTU が前記パーティションにおける前記 1 つ又は複数の CTU 行のうちの現在の CTU 行の 1 番目の CTU であることを確定することと、

前記並列コーディングが有効にされたこと、及び前記 CTU が前記パーティションにおける前記現在の CTU 行の前記 1 番目の CTU であることを確定することに応じて、

確定された前記ビット深度に基づいて、ライスパラメータを計算するための色成分の履歴カウンタを初期値に設定することと、

前記 CTU をデコードすることであって、

前記履歴カウンタに基づいて、前記 CTU における変換ユニット（TU）の前記ライスパラメータを計算することと、

計算された前記ライスパラメータに基づいて、前記 CTU における前記 TU に対応する前記バイナリ文字列を、前記 TU の係数値にデコードすることと、

前記係数値から前記 CTU における前記 TU のピクセル値を確定することと、を含む、前記 CTU をデコードすることと、

前記ビデオのデコードされたパーティションを出力することであって、デコードされた前記パーティションが、前記パーティションにおけるデコードされた前記複数の CTU を含む、出力することと、

を含む、

ことを特徴とする請求項 18 に記載のシステム。

【請求項 20】

確定された前記ビット深度に基づいて、前記ビデオビットストリームを複数のピクチャにデコードすることは、

前記ビデオのパーティションを表すバイナリ文字列にアクセスすることであって、前記パーティションが、1つ又は複数の CTU 行を形成する複数の CTU を含む、アクセスすることと、

前記パーティションにおける前記複数の CTU の各々に対して、

前記 CTU をデコードする前に、且つ、並列コーディングが有効にされたこと、及び前記 CTU が現在の CTU 行の 1 番目の CTU であることを確定することに応じて、前記現在の CTU 行が前記パーティションにおける 1 番目の CTU 行であるか否かを確定することと、

前記現在の CTU 行が前記パーティションにおける前記 1 番目の CTU 行であると確定することに応じて、確定された前記ビット深度に基づいて、ライスパラメータを計算するための色成分の履歴カウンタを初期値に設定することと、

前記現在の CTU 行が前記パーティションにおける前記 1 番目の CTU 行ではないと確定することに応じて、前記色成分の履歴カウンタを、履歴カウンタストレージ変数に記憶された値に設定することと、

前記 CTU をデコードすることであって、

前記履歴カウンタの前記値に基づいて、前記 CTU における TU の前記ライスパラメータを計算することと、

計算された前記ライスパラメータに基づいて、前記 CTU における前記 TU に対応する前記バイナリ文字列を、前記 TU の係数値にデコードすることと、

前記係数値から前記CTUにおける前記TUのピクセル値を確定することと、を含む、前記CTUをデコードすることと、

前記CTUをデコードした後、並列コーディングが有効にされたこと、及び前記CTUが前記現在のCTU行の前記1番目のCTUであることを確定することに応じて、前記履歴カウンタの現在の値を前記履歴カウンタストレージ変数に記憶することと、

を含む、

ことを特徴とする請求項18に記載のシステム。

【発明の詳細な説明】

【技術分野】

【0001】

関連出願の参照

本出願は、発明の名称が「ビデオコーディングにおけるウェーブフロント並列処理のための履歴ベースのライスパラメータ導出」である、2021年8月26日に出願された米国仮特許出願第63/260,600号の優先権と、発明の名称が「ビデオコーディングにおけるウェーブフロント並列処理のための履歴ベースのライスパラメータ導出」である、2021年10月04日に出願された米国仮特許出願第63/262,078号の優先権と、発明の名称が「VVC操作範囲拡張のためのビット深度範囲の表現」である、2021年10月01日に出願された米国仮特許出願第63/251,385号の優先権と、を主張し、それらのすべての内容が引用として本出願に組み込まれる。

【0002】

本開示は一般に、ビデオ処理のためのコンピュータ実装方法及びシステムに関する。具体的に、本開示は、多用途ビデオコーディングのための操作範囲拡張(operation range extension)に関する。

【背景技術】

【0003】

スマートフォン、タブレット端末、コンピュータなどのカメラ付きのデバイスの遍在により、ビデオやイメージのキャプチャがかつてないほど簡単になった。しかし、短いビデオでもそのデータ量は相当なものになることができる。ビデオコーディング技術(ビデオのエンコーディング及びデコーディングを含む)によりビデオデータがより小さなサイズに圧縮され、それによって、様々なビデオが記憶され且つ伝送されることができるようになった。ビデオコーディングは、デジタルテレビ放送、インターネットやモバイルネットワークでのビデオ伝送、リアルタイムアプリケーション(例えば、ビデオチャット、ビデオ会議など)、デジタル多用途ディスク(digital versatile disc、DVD)やブルーレイ・ディスクなど、幅広い用途に使用されている。ビデオを記憶するための記憶スペース及び/又はビデオを伝送するためのネットワーク帯域幅の消費を低減するために、ビデオコーディング方案の効率を向上させることが望まれている。

【発明の概要】

【0004】

いくつかの実施形態は、ビデオコーディングにおけるウェーブフロント並列処理(wavefront parallel processing)のための履歴ベースのライスパラメータ導出(history-based Rice parameter derivation)に関する。一例では、多用途ビデオコーディング(VVC)を利用してエンコードされたビデオビットストリームからビデオをデコードするための方法は、以下の内容を含む。

シーケンスパラメータセット(SPS)シンタックス要素sps__bitdepth__minus 8に基づいて、ビデオのサンプルのビット深度を確定する。SPSシンタックス要素sps__bitdepth__minus 8の値が0から8の範囲内にある。

ビデオパラメータセット(VPS)シンタックス要素vps__ols__dpb__bitdepth__minus 8に基づいて、デコードピクチャバッファ(DPB)のサイズを確定する。VPSシンタックス要素vps__ols__dpb__bitdepth__min

10

20

30

40

50

u s 8 の値が 0 から 8 の範囲内にある。

確定された D P B のサイズを有する記憶スペースを割り当てる。

確定されたビット深度に基づいて、ビデオビットストリームを複数のピクチャにデコードする。

デコードされた複数のピクチャの 1 つ又は複数を D P B に記憶する。

デコードされた複数のピクチャの 1 つ又は複数を表示のために出力する。

【0005】

別の例では、非一時的なコンピュータ可読媒体は、非一時的なコンピュータ可読媒体にプログラムコードが記憶されている。以下の操作を実行するように、プログラムコードは、1 つ又は複数の処理デバイスによって実行可能である。上記操作は、

ビデオのビデオビットストリームに含まれるビデオパラメータセット (V P S) シンタックス要素 `v p s _ o l s _ d p b _ b i t d e p t h _ m i n u s 8` に基づいて、ビデオのデコードピクチャバッファ (D P B) のサイズを確定することであって、多用途ビデオコーディング (V V C) を利用してビデオビットストリームがエンコードされ、V P S シンタックス要素 `v p s _ o l s _ d p b _ b i t d e p t h _ m i n u s 8` の値が 0 から 8 の範囲内にある、確定することと、

確定された D P B のサイズを有する記憶スペースを割り当てることと、

ビデオビットストリームを複数のピクチャにデコードすることと、

デコードされた複数のピクチャの 1 つ又は複数を D P B に記憶することと、

デコードされた複数のピクチャの 1 つ又は複数を表示のために出力することと、を含む

【0006】

別の例では、システムは、処理デバイス及び非一時的なコンピュータ可読媒体を備える。非一時的なコンピュータ可読媒体は処理デバイスに通信的に結合されている。処理デバイスは、非一時的なコンピュータ可読媒体に記憶されたプログラムコードを実行するように構成されており、それによって、以下の操作を実行する。上記操作は、

ビデオのビデオビットストリームに含まれるビデオパラメータセット (V P S) シンタックス要素 `v p s _ o l s _ d p b _ b i t d e p t h _ m i n u s 8` に基づいて、ビデオのデコードピクチャバッファ (D P B) のサイズを確定することであって、多用途ビデオコーディング (V V C) を利用してビデオビットストリームがエンコードされ、V P S シンタックス要素 `v p s _ o l s _ d p b _ b i t d e p t h _ m i n u s 8` の値が 0 から 8 の範囲内にある、確定することと、

確定された D P B のサイズを有する記憶スペースを割り当てることと、

ビデオビットストリームを複数のピクチャにデコードすることと、

デコードされた複数のピクチャの 1 つ又は複数を D P B に記憶することと、

デコードされた複数のピクチャの 1 つ又は複数を表示のために出力することと、を含む

【0007】

これらの例示的な実施形態は、本開示を限定又は定義するために記載されているのではなく、本開示の理解を助けるための例を提供するために記載されている。追加の実施形態は、明細書の詳細に記述され、さらなる説明が明細書に提供される。

【図面の簡単な説明】

【0008】

添付図面を参照しながら以下の詳細な説明を読むと、本開示の特徴、実施形態及び利点がよりよく理解される。

【図 1】図 1 は、本明細書で説明される実施形態を実施するように構成されたビデオエンコーダーの一例を示すブロック図である。

【図 2】図 2 は、本明細書で説明される実施形態を実施するように構成されたビデオデコーダーの一例を示すブロック図である。

【図 3】図 3 は、本開示のいくつかの実施形態に係る、ビデオにおけるピクチャのコーデ

10

20

30

40

50

ィングツリーユニットの分割の一例を示す図である。

【図 4】図 4 は、本開示のいくつかの実施形態に係る、コーディングツリーユニットのコーディングユニットの分割の一例を示す図である。

【図 5】図 5 は、コーディングブロック、及びコーディングブロックの要素を処理するための予め確定された順序の例を示す図である。

【図 6】図 6 は、変換ユニットの境界付近に位置する係数のローカルサム変数を算出するためのテンプレートパターンの一例を示す図である。

【図 7】図 7 は、ウェーブフロント並列処理が有効にされたタイル (tile) の一例を示す図である。

【図 8】図 8 は、本開示のいくつかの実施形態に係る、フレームと、当該フレームに含まれる、タイルと、履歴カウンタが計算されるコーディングツリーユニットとの一例を示す図である。

【図 9】図 9 は、本開示のいくつかの実施形態に係る、ビデオのパーティションをエンコードするプロセスの一例を示す図である。

【図 10】図 10 は、本開示のいくつかの実施形態に係る、ビデオのパーティションをデコードするプロセスの一例を示す図である。

【図 11】図 11 は、本開示のいくつかの実施形態に係る、ビデオのパーティションをエンコードするプロセスの別の例を示す図である。

【図 12】図 12 は、本開示のいくつかの実施形態に係る、ビデオのパーティションをデコードするプロセスの別の例を示す図である。

【図 13】図 13 は、本開示のいくつかの実施形態を実施するために用いられることができるコンピューティングシステムの一例を示す図である。

【発明を実施するための形態】

【0009】

様々な実施形態は、ビデオコーディングにおけるウェーブフロント並列処理のための履歴ベースのライスパラメータ導出を提供する。上述したように、生成され、記憶され、伝送されるビデオデータがより多くなっている。ビデオコーディング技術の効率を向上させることが有益であり、それによって、デコードされたビデオの視覚的品質を損なうことなく、より少ないデータを利用してビデオを表現する。コーディング効率を向上させる 1 つの方法は、エントロピーコーディングによって、できるだけ少ないビットを利用して、処理されたビデオサンプルをバイナリビットストリームに圧縮することである。一方、ビデオには一般的に大量のデータが含まれるため、コーディング（エンコーディング及びデコーディング）中の処理時間を短縮することが有益である。そのために、並列処理がビデオのエンコーディング及びデコーディングに用いられることができる。

【0010】

エントロピーコーディングでは、ビデオサンプルがバイナリビン (binary bin) にバイナリ (2 値化) され、コンテキスト適応型バイナリ算術コーディング (context-adaptive binary arithmetic coding、CABAC) などのコーディングアルゴリズムにより、ビンがさらにビットに圧縮されることができる。2 値化には、多用途ビデオコーディング (Versatile Video Coding、VVC) 規格で規定された、トランケートライス (truncated Rice、TR) と限定された k 次指数ゴロム (k -th order Exp-Golomb、EGk) 2 値化プロセスとの組み合わせに用いられるライスパラメータなどの、2 値化パラメータの計算が必要である。コーディング効率を向上させるために、履歴ベースのライスパラメータ導出が利用される。当該履歴ベースのライスパラメータ導出では、パーティション（例えば、ピクチャ、スライス (slice)、タイル) の現在のコーディングツリーユニット (coding tree unit、CTU) における変換ユニット (transform unit、TU) のライスパラメータは、当該パーティションにおける現在の CTU 及び前の CTU における前の TU 内の係数に従って計算された履歴カウンタ (StatCoeff と表記され) に基づいて導出される。その後、履歴カウ

10

20

30

40

50

ンタは、ライスパラメータを導出するために用いられる置換変数（`H i s t V a l u e`と表記され）を導出するために用いられる。履歴カウンタは、`T U`を処理する際に更新されることができる。いくつかの例では、履歴カウンタが更新されても、`T U`の置換変数は同じままである。

【0011】

履歴カウンタを計算するためのパーティションにおける現在の`C T U`と前の`C T U`との間の依存性は並列処理の利用と衝突し、並列処理の利用を制限し、又は、さらに妨げ、それは不安定又は非効率的なビデオコーディングにつながる。並列処理が有効にされてビデオ処理プロセスを速めることができるように、パーティションにおけるいくつかの`C T U`間の依存性を低減し又は除去することによって、又は、衝突が発生する前に衝突を検出して回避することによって、本明細書で説明される様々な実施形態は上記問題を解決する。いくつかの実施形態を紹介するために、以下の非限定的な例が提供される。

10

【0012】

一実施形態では、履歴カウンタを計算する際の、異なる`C T U`行の`C T U`間の依存性が除去されることにより、並列処理との依存性の衝突が解消される。例えば、パーティションの各`C T U`行に対して、履歴カウンタが再初期化されることができる。`C T U`行における1番目の`C T U`のライスパラメータが計算される前に、履歴カウンタが初期値に設定されることができる。後続の履歴カウンタは、同じ`C T U`行における前の`T U`における履歴カウンタ値に基づいて計算されることができる。このようにして、履歴ベースのライスパラメータ導出における`C T U`の依存性は、同じ`C T U`行内に限定され、異なる`C T U`行間の並列処理を妨げないと同時に、履歴ベースのライスパラメータ導出によって得られるコーディングゲインの恩恵を受ける。さらに、履歴ベースのライスパラメータ導出のプロセスが簡略化され、計算の複雑性が軽減される。

20

【0013】

別の実施形態では、履歴カウンタを計算する際の`C T U`間の依存性は、並列処理における`C T U`間の依存性にアライメントする。例えば、並列コーディングは、パーティションの`C T U`行の間で実施され得、2つの連続する`C T U`行の間に`N - C T U`遅延が存在し得る。即ち、本`C T U`行の処理は、前の`C T U`行における`N`個の`C T U`が処理された後に開始される。このシナリオでは、本`C T U`行の履歴カウンタは、前の`C T U`行における前の`N`個以下の`C T U`におけるサンプルに基づいて計算されることができる。それは、記憶同期プロセスによって実現されることができる。`C T U`行の1番目の`C T U`における最後の`T U`が処理された後、履歴カウンタがストレージ変数に記憶されることができる。その後、後続の`C T U`行の1番目の`C T U`における1番目の`T U`を処理する前に、履歴カウンタとストレージ変数における記憶された値とを同期させることができる。

30

【0014】

いくつかの例では、代替の履歴ベースのライスパラメータ導出が利用される。このような代替の履歴ベースのライスパラメータ導出では、`T U`を処理する際に履歴カウンタ`S t a t C o e f f`が更新されると、置換変数`H i s t V a l u e`が更新される。並列処理との依存性の衝突を回避するために、履歴カウンタを計算する際の`C T U`間の依存性も同様に、`N`個の`C T U`以下に制限されることができる。同様に、記憶同期プロセスが実現されることができる。`C T U`行の1番目の`C T U`における最後の`T U`が処理された後、履歴カウンタ及び置換変数はそれぞれストレージ変数に記憶されることができる。そして、後続の`C T U`行の1番目の`C T U`における1番目の`T U`を処理する前に、履歴カウンタと置換変数とを、対応するストレージ変数における記憶された値と同期させることができる。

40

【0015】

このようにして、履歴カウンタを計算する際の2つの連続する`C T U`行における`C T U`間の依存性は、並列コーディングを実行する際の`C T U`間の依存性以下に制限される（即ち、アライメントする）。その結果、履歴カウンタ計算は、並列処理を妨げないと同時に、履歴ベースのライスパラメータ導出によって得られるコーディングゲインの恩恵を受ける。

50

【0016】

又は、並列処理及び履歴ベースのライスパラメータ導出がビットストリームに共存することを防止する。例えば、ビデオコーダーは、並列処理が有効にされるか否かを確定することができる。並列処理が有効にされる場合、履歴ベースのライスパラメータ導出が無効にされ、そうではない場合、履歴ベースのライスパラメータ導出が有効にされる。同様に、ビデオコーダーは履歴ベースのライスパラメータ導出が有効にされると確定した場合、並列処理が無効にされ、その逆も同様である。

【0017】

上述のように確定されたライスパラメータを利用して、ビデオエンコーダーは、予測残差データ（例えば、残差の量子化された変換係数）をバイナリビンに2値化し、エントロピーコーディングアルゴリズムを利用して、ビデオビットストリームに含まれるように、さらにビンをビットに圧縮することができる。デコーダー側では、デコーダーは、ビットストリームをバイナリビンにデコードし、上述の任意の方法又は任意の方法の組み合わせを利用してライスパラメータを確定し、その後、バイナリビンから係数を確定することができる。係数がさらに逆量子化され（*de-quantized*）逆変換されて、表示のためにビデオブロックを再構成することができる。

【0018】

いくつかの実施形態では、ビデオのサンプルのビット深度（例えば、履歴カウンタ *StatCoeff* の初期値を確定するために用いられるビット深度）は、シーケンスパラメータセット（*Sequence Parameter Set*、*SPS*）シンタックス要素 *sps__bitdepth__minus8* に基づいて確定されることができる。*SPS* シンタックス要素 *sps__bitdepth__minus8* の値は、0 から 8 の範囲内にある。同様に、デコードされたピクチャを記憶するために用いられるデコードピクチャバッファ（*decoded picture buffer*、*DPB*）のサイズは、ビデオパラメータセット（*Video Parameter Set*、*VPS*）シンタックス要素 *vps__ols__dpb__bitdepth__minus8* に基づいて確定されることができる。*VPS* シンタックス要素 *vps__ols__dpb__bitdepth__minus8* の値は、0 から 8 の範囲内にある。確定された *DPB* のサイズに応じて、記憶スペースが *DPB* に割り当てられることができる。確定されたビット深度及び *DPB* は、ビデオビットストリームをピクチャにデコードするプロセスの全体に用いられる。

【0019】

本明細書で説明されるように、いくつかの実施形態は、履歴ベースのライスパラメータ導出と並列コーディングとを協調する（*coordinate*）ことによって、ビデオコーディングの効率及び計算効率を改善する。そうすることにより、履歴ベースのライスパラメータ導出と並列コーディングとの間の衝突を回避することができ、それによって、コーディングプロセスの安定性を向上させることができる。さらに、履歴ベースのライスパラメータ導出における *CTU* 間の依存性を並列コーディングにおける依存性以下に制限することにより、コーディングプロセスの計算効率を犠牲にすることなく、履歴ベースのライスパラメータ導出によってコーディングゲインが得られることができる。本技術は、将来のビデオコーディング規格における効果的なコーディングツールとなり得る。

【0020】

ここで図面を参照すると、図1は、本明細書で説明される実施形態を実施するように構成されたビデオエンコーダー100の一例を示すブロック図である。図1に示される例では、ビデオエンコーダー100は、パーティションモジュール112と、変換モジュール114と、量子化モジュール115と、逆量子化モジュール118と、逆変換モジュール119と、インループフィルタモジュール120と、イントラ予測モジュール126と、インター予測モジュール124と、動き推定モジュール122と、デコードピクチャバッファ130と、エントロピーコーディングモジュール116とを含む。

【0021】

ビデオエンコーダー100への入力は、ピクチャ（フレーム又はイメージとも呼ばれる

10

20

30

40

50

）のシーケンスを含む入力ビデオ102である。ブロックベースのビデオエンコーダーでは、各ピクチャに対して、ビデオエンコーダー100は、パーティションモジュール112を利用して、ピクチャをブロック104に分割し、各ブロックは複数のピクセルを含む。ブロックは、マクロブロック、コーディングツリーユニット、コーディングユニット、予測ユニット、及び／又は予測ブロックであってもよい。1つのピクチャは異なるサイズを有するブロックを含むことができ、ビデオの異なるピクチャのブロック分割も異なることができる。各ブロックは、イントラ予測、インター予測、又は、イントラとインターのハイブリッド予測（*intra and inter hybrid prediction*）など、異なる予測を利用してエンコードされることができる。

【0022】

通常、ビデオ信号の一番目のピクチャは、イントラ予測のみを利用してエンコードされるイントラ予測ピクチャである。イントラ予測モードでは、同じピクチャからのデータのみを利用して当該ピクチャのブロックが予測される。イントラ予測されたピクチャは、他のピクチャからの情報なしでデコードされることができる。イントラ予測を実行するために、図1に示されたビデオエンコーダー100は、イントラ予測モジュール126を利用することができる。イントラ予測モジュール126は、同じピクチャの隣接ブロックの再構成されたブロック136における再構成サンプルを利用して、イントラ予測ブロック（予測ブロック134）を生成するように構成されている。イントラ予測は、当該ブロックのために選択されたイントラ予測モードに基づいて実行される。次に、ビデオエンコーダー100は、ブロック104とイントラ予測ブロック134との間の差分を計算する。この差分は残差ブロック106と呼ばれる。

【0023】

ブロックから冗長性をさらに除去するために、変換モジュール114は、ブロックにおけるサンプルに変換を適用することで、残差ブロック106を変換ドメインに変換する。変換の例としては、離散コサイン変換（*discrete cosine transform*、DCT）又は離散サイン変換（*discrete sine transform*、DST）が含まれるが、それらに限定されない。変換された値は、変換ドメインにおける残差ブロックを表す変換係数と呼ばれることができる。いくつかの例では、残差ブロックは、変換モジュール114によって変換されることなく、直接に量子化されることができる。それは変換スキップモードと呼ばれる。

【0024】

ビデオエンコーダー100はさらに、量子化モジュール115を利用して変換係数を量子化し、それによって、量子化された係数を得ることができる。量子化は、サンプルを量子化ステップサイズで割り、それに続いて四捨五入する（*rounding*）ことを含む、一方、逆量子化は、量子化された値に量子化ステップサイズを乗算することを含む。このような量子化プロセスはスカラー量子化（*scalar quantization*）と呼ばれる。量子化は、（変換された又は変換されていない）ビデオサンプルのダイナミックレンジを減少させるために用いられ、それによって、より少ないビットを利用してビデオサンプルを表現する。

【0025】

ブロック内の係数／サンプルの量子化は独立して行われることができ、このような量子化方法は、H.264、高効率ビデオ符号化（*high efficiency video coding*、HEVC）などのいくつかの既存のビデオ圧縮規格に用いられる。N×Mのブロックに対して、係数量子化及びコーディングに用いられるために、特定のスキャン順序で、ブロックの2D（*two-dimensional*）係数を1D（*one-dimensional*）配列に変換することができる。ブロック内の係数の量子化は、スキャン順序情報を利用することができる。例えば、ブロックにおける所与の係数の量子化は、スキャン順序に沿った前の量子化された値の状態に依存することができる。コーディング効率をさらに向上させるために、複数の量子化器を利用することができる。どの量子化器が現在の係数の量子化に用いられるかは、エンコーディング／デコーディングス

10

20

30

40

50

キャン順序における現在の係数の前の情報に依存する。このような量子化方式は、依存量子化 (dependent quantization) と呼ばれる。

【0026】

量子化の度合いは、量子化ステップサイズを利用して調整されることができる。例えば、スカラー量子化に対して、より細かい量子化又はより粗い量子化を得るために、異なる量子化ステップサイズを適用することができる。比較的小さい量子化ステップサイズは比較的小さい量子化に対応し、比較的大きい量子化ステップサイズは比較的小さい量子化に対応する。量子化ステップサイズは、量子化パラメータ (quantization parameter、QP) によって指示されることができる。ビデオデコーダーが同じ量子化パラメータを用いてデコーディングできるように、ビデオのエンコードされたビットストリームに量子化パラメータが提供される。

10

【0027】

次に、量子化されたサンプルは、エントロピーコーディングモジュール116によってコードされて、ビデオ信号のサイズをさらに縮小する。エントロピーエンコーディングモジュール116は、エントロピーエンコーディングアルゴリズムを量子化されたサンプルに適用するように構成されている。いくつかの例では、量子化されたサンプルはバイナリビンに2値化され、バイナリビンはコーディングアルゴリズムによりさらにビットに圧縮される。2値化方法の例としては、ランケートライス (TR) 及び限定されたk次指数ゴロム (EGk) 2値化が含まれるが、それらに限定されない。コーディング効率を向上させるために、履歴ベースのライスパラメータ導出の方法が利用され、ここで、変換ユニット (TU) のために導出されたライスパラメータは、前のTUから得られ又は更新された変数に基づいている。エントロピーエンコーディングアルゴリズムの例としては、可変長コーディング (variable length coding、VLC) スキーム、コンテキスト適応型可変長コーディング (context adaptive VLC、CAVLC) スキーム、算術コーディングスキーム、2値化、コンテキスト適応型バイナリ算術コーディング (CABAC)、シンタックスベースのコンテキスト適応型バイナリ算術コーディング (syntax-based context-adaptive binary arithmetic coding、SBAC)、確率間隔区分エントロピー (probability interval partitioning entropy、PIPE) コーディング、又は他のエントロピーエンコーディング技術が含まれるが、それらに限定されない。エントロピーコードされたデータは、出力エンコードビデオ132のビットストリームに追加される。

20

30

【0028】

上述したように、隣接ブロックからの再構成されたブロック136は、ピクチャのブロックのイントラ予測に用いられる。ブロックの再構成されたブロック136を生成することは、当該ブロックの再構成された残差を計算することに係る。再構成された残差は、逆量子化及び逆変換をブロックの量子化された残差に適用することによって確定されることができる。逆量子化モジュール118は、逆量子化を量子化されたサンプルに適用して、逆量子化された係数を得るように構成されている。逆量子化モジュール118は、量子化モジュール115と同じ量子化ステップサイズを利用することにより、量子化モジュール115によって適用された量子化スキームの逆を適用する。逆変換モジュール119は、変換モジュール114によって適用された変換の逆変換 (例えば、逆DCT又は逆DST) を、逆量子化されたサンプルに適用するように構成されている。逆変換モジュール119の出力は、ピクセルドメインにおけるブロックの再構成された残差である。再構成された残差がブロックの予測ブロック134に加算されて、ピクセルドメインにおける再構成されたブロック136を得ることができる。変換がスキップされたブロックに対して、逆変換モジュール119はそれらのブロックに適用されない。逆量子化されたサンプルは、それらのブロックの再構成された残差である。

40

【0029】

1番目のイントラ予測ピクチャに続く後続のピクチャにおけるブロックは、インター予

50

測又はイントラ予測のいずれかを利用してコードされることができる。インター予測では、ピクチャにおけるブロックの予測は、1つ又は複数の前にエンコードされたビデオピクチャからの予測である。インター予測を実行するために、ビデオエンコーダー100はインター予測モジュール124を利用する。インター予測モジュール124は、動き推定モジュール122によって提供される動き推定に基づいて、ブロックの動き補償を実行するように構成されている。

【0030】

動き推定モジュール122は、現在のピクチャの現在のブロック104と動き推定に用いられるデコードされた参照ピクチャ108とを比較する。デコードされた参照ピクチャ108は、デコードピクチャバッファ130に記憶されている。動き推定モジュール122は、デコードされた参照ピクチャ108から、現在のブロックに最もマッチングする参照ブロックを選択する。動き推定モジュール122はさらに、参照ブロックの位置（例えば、x、y座標）と現在のブロックの位置との間のオフセットを特定する。当該オフセットは動きベクトル（motion vector、MV）と呼ばれ、インター予測モジュール124に提供される。いくつかの場合では、複数のデコードされた参照ピクチャ108において、複数の参照ブロックがブロックのために特定される。それによって、複数の動きベクトルが生成されて、インター予測モジュール124に提供される。

【0031】

インター予測モジュール124は、動きベクトル及び他のインター予測パラメータを利用して動き補償を実行して、現在のブロック（即ちインター予測ブロック134）の予測を生成する。例えば、動きベクトルに基づいて、インター予測モジュール124は、対応する参照ピクチャにおける動きベクトルによって指向される予測ブロックを位置付けることができる。複数の予測ブロックが存在する場合、現在のブロックの予測ブロック134を生成するために、これらの予測ブロックはいくつかのウェイト（weight）と組み合わせられる。

【0032】

インター予測されたブロックに対して、ビデオエンコーダー100は、ブロック104からインター予測ブロック134を減算して、残差ブロック106を生成することができる。残差ブロック106は、上述したイントラ予測ブロックの残差と同様の方法で、変換され、量子化され、且つエントロピーコードされることができる。同様に、インター予測されたブロックの再構成されたブロック136は、残差を逆量子化し、逆変換し、続いて対応する予測ブロック134と組み合わせることによって得られることができる。

【0033】

動き推定に用いられるデコードされたピクチャ108を得るために、再構成されたブロック136は、インループフィルタモジュール120によって処理される。インループフィルタモジュール120は、ピクセル遷移（pixel transition）を平滑化するように構成されており、それによって、ビデオ品質を向上させる。インループフィルタモジュール120は、デブロックフィルタ（de-blocking filter）、サンプル適応オフセット（sample adaptive offset、SAO）フィルタ、又は適応ループフィルタ（adaptive loop filter、ALF）などの1つ又は複数のインループフィルタを実現するように構成されることができる。

【0034】

図2は、本明細書で説明される実施形態を実施するように構成されたビデオデコーダー200の一例を示すブロック図である。ビデオデコーダー200は、ビットストリームにおけるエンコードされたビデオ202を処理し、デコードされたピクチャ208を生成する。図2に示された例では、ビデオデコーダー200は、エントロピーデコーディングモジュール216、逆量子化モジュール218、逆変換モジュール219、インループフィルタモジュール220、イントラ予測モジュール226、インター予測モジュール224、及びデコードピクチャバッファ230を含む。

【0035】

エントロピーデコーディングモジュール216は、エンコードされたビデオ202のエントロピーデコーディングを実行するように構成されている。エントロピーデコーディングモジュール216は、量子化された係数と、イントラ予測パラメータ及びインター予測パラメータを含むコーディングパラメータと、他の情報とをデコードする。いくつかの例では、エントロピーデコーディングモジュール216は、エンコードされたビデオ202のビットストリームをバイナリ表現にデコードし、そして、このバイナリ表現を係数の量子化レベルに変換する。次に、エントロピーデコードされた係数は、逆量子化モジュール218によって逆量子化され、続いて、逆変換モジュール219によってピクセルドメインに逆変換される。逆量子化モジュール218及び逆変換モジュール219はそれぞれ、先に図1について述べたように、逆量子化モジュール118及び逆変換モジュール119と同様に機能する。逆変換された残差ブロックが対応する予測ブロック234に加算されて、再構成されたブロック236を生成することができる。変換がスキップされたブロックに対して、逆変換モジュール219がこれらのブロックに適用されない。逆量子化モジュール118によって生成された、逆量子化されたサンプルは、再構成されたブロック236を生成するために用いられる。

10

【0036】

特定のブロックの予測ブロック234は、当該ブロックの予測モードに基づいて生成される。ブロックのコーディングパラメータは当該ブロックがイントラ予測されることを指示する場合、同じピクチャにおける参照ブロックの再構成されたブロック236がイントラ予測モジュール226にフィードされ、それによって、当該ブロックの予測ブロック234を生成することができる。ブロックのコーディングパラメータは当該ブロックがインター予測されることを指示する場合、予測ブロック234はインター予測モジュール224によって生成される。イントラ予測モジュール226及びインター予測モジュール224は、それぞれ図1のイントラ予測モジュール126及びインター予測モジュール124と同様に機能する。

20

【0037】

先に図1について述べたように、インター予測は1つ又は複数の参照ピクチャに係る。ビデオデコーダー200は、インループフィルタモジュール220を参照ピクチャの再構成されたブロックに適用することによって、参照ピクチャのデコードされたピクチャ208を生成する。デコードされたピクチャ208は、インター予測モジュール224によって利用されるために、且つ出力のために、デコードピクチャバッファ230に記憶される。

30

【0038】

ここで図3を参照すると、図3は、本開示のいくつかの実施形態に係る、ビデオにおけるピクチャのコーディングツリーユニットの分割の一例を示す図である。先に図1及び図2について述べたように、ビデオのピクチャをエンコードするために、ピクチャは、例えば、図3に示されたようなVVCにおけるCTU302のようなブロックに分割される。例えば、CTU302は、 128×128 ピクセルのブロックであることができる。CTUは、図3に示されたような順序などの順序に従って処理される。いくつかの例では、ピクチャにおける各CTU302は、図4に示されるような1つ又は複数のCU402に分割されることができ、CU402は、予測及び変換のための予測ユニット又は変換ユニット(TU)にさらに分割されることができる。コーディングスキームに応じて、CTU302のCU402への分割は異なり得る。例えば、VVCでは、CU402は長方形又は正方形であり得、予測ユニット又は変換ユニットにさらに分割されることがなくコードされ得る。各CU402は、そのルートCTU302と同様に大きい、又は 4×4 ブロックと同様に小さい、ルートCTU302の細分であることができる。図4に示されるように、VVCにおけるCTU302のCU402への分割は、4分木分割、2分木分割又は3分木分割であり得る。図4において、実線は4分木分割を示し、破線は2分木分割又は3分木分割を示す。

40

50

【0039】

先に図1及び図2について述べたように、量子化は、ビデオ信号におけるブロックの要素のダイナミックレンジを減少させるために用いられ、それによって、より少ないビットを利用してビデオ信号を表現する。いくつかの例では、量子化の前に、ブロックの特定の位置における要素は係数と呼ばれる。量子化後、係数の量子化された値は、量子化レベル又はレベルと呼ばれる。量子化は通常、量子化ステップサイズで割り、それに続いて四捨五入する (rounding) ことを含み、逆量子化は量子化ステップサイズを乗算することを含む。このような量子化プロセスはスカラー量子化とも呼ばれる。ブロック内の係数の量子化は独立して行われることができ、このような独立した量子化方法は、H. 264、HEVCなどのいくつかの既存のビデオ圧縮規格に用いられる。他の例では、依存量子化がVVCなどに用いられる。

10

【0040】

$N \times M$ のブロックに対して、係数量子化及びコーディングに用いられるために、特定のスキャン順序でブロックの2D係数を1D配列に変換することができる。同じスキャン順序はエンコーディング及びデコーディングの両方に用いられる。図5は、コーディングブロック (例えばTU)、及びコーディングブロックの係数を処理するための予め確定された順序の例を示す図である。この例では、コーディングブロック500は 8×8 のサイズを有し、処理は、右下隅の位置 L_0 から始まり、左上隅の位置 L_{63} で終了する。ブロック500が変換されたブロックである場合、図5に示された予め確定された順序は、最も高い周波数から最も低い周波数までである。いくつかの例では、量子化及び2値化などの

20

【0041】

残差コーディング

【0042】

残差コーディングは、ビデオコーディングにおいて量子化レベルをビットストリームに変換するために用いられる。量子化後、 $N \times M$ 変換ユニット (TU) コーディングブロックは $N \times M$ 個の量子化レベルを有する。これらの $N \times M$ 個のレベルはゼロ又は非ゼロ値である。レベルがバイナリではない場合、非ゼロレベルはさらにバイナリビンに2値化される。CABACにより、ビンをビットにさらに圧縮することができる。また、コンテキストモデリングベースのコーディング方法が2種類ある。特に、1つの方法では、隣接コード情報 (neighboring coded information) に基づいてコンテキストモデルを適応的に更新する。このような方法はコンテキストコード法 (context coded method) と呼ばれ、この方法でコードされたビンはコンテキストコードビン (context coded bin) と呼ばれる。これに対して、もう1つの方法では、1か0の確率が常に50%であると仮定し、そのため、適応なしの固定のコンテキストモデリングを常に利用する。このような方法はバイパス法 (bypass method) と呼ばれ、この方法でコードされたビンはバイパスビンと呼ばれる。

30

40

【0043】

VVCにおけるレギュラー残差コーディング (regular residual coding、RRC) ブロックでは、最後の非ゼロレベルの位置は、コーディングスキャン順序に沿った最後の非ゼロレベルの位置として定義される。最後の非ゼロレベルの2D座標 ($last_sig_coeff_x$ 及び $last_sig_coeff_y$) の表現は、 $last_sig_coeff_x_prefix$ 、 $last_sig_coeff_y_prefix$ 、 $last_sig_coeff_x_suffix$ 、 $last_sig_coeff_y_suffix$ という4つのプレフィックス及びサフィックスのシンタックス要素を含む。シンタックス要素 $last_sig_coeff_x_prefix$ 及び $last_sig_coeff_y_prefix$ は、まずコンテキストコー

50

ド法でコードされる。last_sig_coeff_x_suffix及びlast_sig_coeff_y_suffixが存在する場合、それらはバイパス法でコードされる。RRCブロックは複数の予め定義されたサブブロックからなることができる。シンタックス要素sb_coded_flagは、現在のサブブロックのレベルがすべて0に等しいか否かを指示するために用いられる。sb_coded_flagが1に等しい場合、現在のサブブロックに少なくとも1つの非ゼロ係数がある。sb_coded_flagが0に等しい場合、現在のサブブロックにおけるすべての係数は0となる。しかしながら、最後の非ゼロレベルを有する最後の非ゼロサブブロックのsb_coded_flagは、ビットストリームにコードされることなく、コーディングスキャン順序に従って、last_sig_coeff_x及びlast_sig_coeff_yから1として導出される。また、DC位置を含む左上のサブブロックのsb_coded_flagも、ビットストリームにコードされることなく1として導出される。ビットストリームにおけるシンタックス要素sb_coded_flagは、コンテキストコード法によりコードされる。RRCは、先に図5について述べたような逆コーディングスキャン順序で、最後の非ゼロサブブロックからサブブロックごとにコードする。

10

【0044】

最悪の場合のスループットを確保するために、予め定義された値remBinsPass1はコンテキストコードビンの最大数を制限するために用いられる。サブブロック内では、RRCは逆コーディングスキャン順序で各位置のレベルをコードする。remBinsPass1が4より大きい場合、現在のレベルをコードする際、sig_coeff_flagと呼ばれるフラグがまずビットストリームにコードされて、当該レベルがゼロであるか又は非ゼロであることを指示する。当該レベルが非ゼロである場合、abs_level_gt_x_flag[n][0]（nは、サブブロック内の現在位置のスキャン順序に沿ったインデックスである）は、絶対レベルが1であるか又は1より大きいかを指示する。絶対レベルが1より大きい場合、par_level_flagがコードされて、当該レベルがVVCにおいて奇数であるか又は偶数であることを指示し、次にabs_level_gt_x_flag[n][1]が存在する。par_level_flag及びabs_level_gt_x_flag[n][1]のフラグは共に、レベルが2、又は3以上であることを指示するために用いられる。上記各シンタックス要素をコンテキストコードビンとしてコードした後、remBinsPass1の値は1を引く。

20

30

【0045】

絶対レベルが3より大きいか、又はremBinsPass1の値が4以下である場合、コンテキストコード法によって前述のビンをコードした後、さらなる2つのシンタックス要素、abs_remainder及びdec_abs_levelは、残りのレベルのバイパスコードビンとしてコードされることができる。さらに、ブロック内の各レベルのサインも、量子化レベルを表すようにコードされて、バイパスコードビンとしてコードされる。

【0046】

別の残差コーディング方法では、abs_level_gt_xX_flag及び残りのレベルを利用して、残差ブロックのレベルコーディングのシンタックス要素に対する条件付きの解析を可能にし、レベルの絶対値の対応する2値化を表1に示す。ここで、abs_level_gt_xX_flagは、レベルの絶対値がXより大きいか否かを記述し、Xは、例えば、0、1、2、又はNなどの整数である。abs_level_gt_xY_flagが0であり、Yが0からN-1の間の整数である場合、abs_level_gt_x(Y+1)フラグが存在しない。abs_level_gt_xY_flagが1である場合、abs_level_gt_x(Y+1)フラグが存在する。さらに、abs_level_gt_xN_flagが0である場合、残りのレベルが存在しない。abs_level_gt_xN_flagが1である場合、残りのレベルが存在し、且つレベルから(N+1)を除去した後の値を表す。通常、abs_level_gt_xX_flagはコンテキストコード法で、残りのレベルはバイパス法でそれぞれコードされる。

40

50

【0047】

【表1】

abs_level_gtxX_flag及び剰余(remainder)に基づいた残差コーディング

abs(level)	0	1	2	3	4	5	6	7	8	9	10	11	12	...
abs_level_gtx0_flag	0	1	1	1	1	1	1	1	1	1	1	1	1	...
abs_level_gtx1_flag		0	1	1	1	1	1	1	1	1	1	1	1	...
abs_level_gtx2_flag			0	1	1	1	1	1	1	1	1	1	1	...
abs_level_gtx3_flag				0	1	1	1	1	1	1	1	1	1	...
剰余					0	1	2	3	4	5	6	7	8	...

10

【0048】

変換スキップ残差コーディングモード(transform skip residual coding mode、TSRC)でコードされたブロックに対して、TSRCは、コーディングスキャン順序に沿って、左上のサブブロックからサブブロックごとにコードする。同様に、シンタックス要素sb_coded_flagは、現在のサブブロックの残差がすべて0に等しいか否かを指示するために用いられる。特定の条件が発生した場合、最後のサブブロック以外の、すべてのサブブロックのすべてのシンタックス要素sb_coded_flagは、ビットストリームにコードされる。最後のサブブロックの前のすべてのサブブロックのすべてのsb_coded_flagが1に等しくない場合、最後のサブブロックに対して、sb_coded_flagは1として導出され、且つこのフラグはビットストリームにコードされない。最悪の場合のスループットを確保するために、予め定義された値RemCcb sはコンテキストコードビンの最大数を制限するために用いられる。現在のサブブロックが非ゼロレベルを有する場合、TSRCはコーディングスキャン順序で各位置のレベルをコードする。RemCcb sが4より大きい場合、以下のシンタックス要素がコンテキストコード法でコードされる。各レベルに対して、まずsig_coeff_flagがビットストリームにコードされて、レベルがゼロであるか又は非ゼロであるかを指示する。レベルが非ゼロである場合、coeff_sign_flagはコードされて、レベルが正であるか又は負であるかを指示する。次に、abs_level_gtx_flag[n][0] (nは、サブブロック内の現在位置のスキャン順序に沿ったインデックスである)がコードされて、現在位置の現在の絶対レベルが1より大きいか否かを指示する。abs_level_gtx_flag[n][0]が0ではない場合、par_level_flagがコードされる。上記各シンタックス要素をコンテキストコード法でコードした後、RemCcb sの値は1を引く。

20

30

【0049】

現在のサブブロック内のすべての位置の上記シンタックス要素をコードした後、RemCcb sが依然として4より大きい場合、最多として別の4つのabs_level_gtx_flag[n][j] (nは、サブブロック内の現在位置のスキャン順序に沿ったインデックスであり、jは1から4までであり)がコンテキストコード法でコードされる。各abs_level_gtx_flag[n][j]がコードされた後、RemCcb sの値は1を引く。RemCcb sが4以下である場合、サブブロック内の現在位置に対して、シンタックス要素abs_remainderは、必要に応じてバイパス法でコードされる。絶対レベルが完全にabs_remainderシンタックス要素でバイパス法によってコードされる位置に対して、coeff_sign_flagsもバイパス法によってコードされる。要約すると、コンテキストコードビンの総数を制限し、且つ最悪の場合のスループットを確保するために、RRCに予め定義されたカウンタremBi

40

50

`nsPass1`が存在し、又は`TSRC`に`RemCcb`sが存在する。

【0050】

ライスパラメータ導出

【0051】

VVCにおける現在のRRC設計では、バイパスピンとしてコードされた`abs__remainder`及び`dec__abs__level`という2つのシンタックス要素は、残りのレベルに用いられるように、ビットストリームに存在する可能性がある。`abs__remainder`と`dec__abs__level`の両方は、VVC規格で規定されたトランケートライス (TR) と限定されたk次指数ゴロム (EGk) 2値化プロセスとの組み合わせによって2値化される。当該プロセスには、所与のレベルを2値化するために、ライスパラメータが必要となる。最適なライスパラメータを得るために、以下に説明されるようなローカルサム法 (`local sum method`) が採用される。

10

【0052】

配列`AbsLevel[xC][yC]`は、色成分インデックス`cIdx`の現在の変換ブロックの変換係数レベルの絶対値の配列を表す。色成分インデックス`cIdx`を有する変換ブロックの配列`AbsLevel[x][y]`と左上のルーマ位置 (`x0, y0`) とが与えられると、ローカルサム変数`locSumAbs`は、以下の擬似コードプロセス (`pseudo-code process`) で規定されるように導出される。

```
locSumAbs = 0
if( xC < ( 1 << log2TbWidth ) - 1 ) {
    locSumAbs += AbsLevel[ xC + 1 ][ yC ]
    if( xC < ( 1 << log2TbWidth ) - 2 )
        locSumAbs += AbsLevel[ xC + 2 ][ yC ]
    if( yC < ( 1 << log2TbHeight ) - 1 )
        locSumAbs += AbsLevel[ xC + 1 ][ yC + 1 ]
}
if( yC < ( 1 << log2TbHeight ) - 1 ) {
    locSumAbs += AbsLevel[ xC ][ yC + 1 ]
    if( yC < ( 1 << log2TbHeight ) - 2 )
        locSumAbs += AbsLevel[ xC ][ yC + 2 ]
}
locSumAbs = Clip3( 0, 31, locSumAbs - baseLevel * 5 )
```

20

30

40

ここで、`log2TbWidth`は、変換ブロックの幅の2を底とする対数であり、`log2TbHeight`は、変換ブロックの高さの2を底とする対数である。`abs__remainder`に対して、変数`baseLevel`は4であり、`dec__abs__level`に対して、変数`baseLevel`は0である。ローカルサム変数`locSumAbs`が与えられると、ライスパラメータ`cRiceParam`は表2に規定されたように導出される。

【0053】

50

【表 2】

locSumAbs に基づいた cRiceParam の規格

locSumAbs	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cRiceParam	0	0	0	0	0	0	0	1	1	1	1	1	1	1	2	2
locSumAbs	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
cRiceParam	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3

10

【0054】

履歴ベースのライスパラメータ導出

【0055】

係数が TU の境界に位置する場合、又は、まず Rice 法でデコードされる場合、ライスパラメータ導出に対して採用されたテンプレート計算は、不正確な係数の推定を生成する可能性がある。これらの係数に対して、一部のテンプレート位置が TU の外側に位置し、且つ値 0 に解釈され又は初期化されることができ、テンプレート計算が 0 に偏る。図 6 は、TU の境界付近に位置する係数の locSumAbs を算出するためのテンプレートパターンの一例を示す図である。図 6 は、複数の CU に分割された CTU 602 を示し、各 CU は複数の TU を含む。TU 604 に対して、現在の係数の位置が実線のブロックで示され、テンプレートパターンにおける隣接サンプルの位置がパターン化されたブロック (patterned block) で示されている。パターン化されたブロックは、ローカルサム変数 locSumAbs を計算するための、現在の係数の予め確定された近傍を示す。

20

【0056】

図 6 では、現在係数 606 が TU 604 の境界に近い、テンプレートパターンにおける隣接サンプルの一部は、TU の境界外にあり、例えば、隣接サンプル 608B 及び 608E が挙げられる。上述したライスパラメータ導出では、ローカルサム変数 locSumAbs を計算する際に、これらの境界外の隣接サンプルが 0 に設定され、その結果として、ライスパラメータ導出が不正確となる。ビット深度が高い (例えば 10 ビットより大きい) サンプルに対して、TU の境界外にある隣接サンプルが多くあることができる。このような多くの隣接サンプルを 0 に設定すると、ライスパラメータ導出にさらなる誤差が生じる。

30

【0057】

計算されたテンプレートからのライス推定の精度を向上させるために、現在の TU の外側にあるテンプレート位置に対して、ローカルサム変数 locSumAbs が 0 に初期化されるのではなく、履歴導出値 (history derived value) で更新されることが提案された。この方法の実施は、以下の VVC 規格テキスト抜粋の 9.3.3.2 節に示されており、提案されたテキストには下線が引かれている。

【0058】

隣接係数／サンプル値の履歴を維持するために、色成分ごとの履歴カウンタ StatCoeff [cIdx] (cIdx = 0、1、2 であり、それぞれ 3 つの色成分 Y、U、V を表す) が利用される。当該 CTU がパーティション (例えば、ピクチャ、スライス、又はタイル) における 1 番目の CTU である場合、StatCoeff [cIdx] は以下のように初期化される。

40

【数 1】

【式 1】

$$\text{StatCoeff}[\text{idx}] = 2 * \text{Floor}(\text{Log2}(\text{BitDepth} - 10))$$

50

ここで、`BitDepth`はビデオのルーマ及びクロマ配列のサンプルのビット深度を指定し、`Floor(x)`は x 以下の最大の整数を表し、`Log2(x)`は x の2を底とする対数である。TUのデコーディング及び履歴カウンタの更新の前に、置換変数`HistValue`が次のように初期化される。

【数2】

【式2】

`HistValue[cIdx] = 1 << StatCoeff[cIdx]`

置換変数`HistValue`は、TUの境界外にある隣接サンプル（例えば、隣接サンプルはTUの外側に水平座標又は垂直座標を有する）の推定として利用される。ローカルサム変数`locSumAbs`は、以下の擬似コードプロセスで規定されたように再導出され、変更は下線で示されている。

```

locSumAbs = 0
if( xC < ( 1 << log2TbWidth ) - 1 ) {
    locSumAbs += AbsLevel[ xC + 1 ][ yC ]
    if( xC < ( 1 << log2TbWidth ) - 2 )
        locSumAbs += AbsLevel[ xC + 2 ][ yC ]
    else
        locSumAbs += HistValue
    if( yC < ( 1 << log2TbHeight ) - 1 )
        locSumAbs += AbsLevel[ xC + 1 ][ yC + 1 ]
    else
        locSumAbs += HistValue
}
else
    locSumAbs += 2 * HistValue
if( yC < ( 1 << log2TbHeight ) - 1 ) {
    locSumAbs += AbsLevel[ xC ][ yC + 1 ]
    if( yC < ( 1 << log2TbHeight ) - 2 )
        locSumAbs += AbsLevel[ xC ][ yC + 2 ]
    else
        locSumAbs += HistValue
}
else
    locSumAbs += HistValue

```

【0059】

履歴カウンタ`StatCoeff`は、指数移動平均（`exponential moving average`）のプロセスを通じて、1番目の非ゼロのゴロム・ライスコード

変換係数 (`abs_remainder[cIdx]` 又は `dec_abs_level[cIdx]`) から、TUごとに1回更新される。TUにおける1番目の非ゼロのゴロム・ライスコード変換係数が `abs_remainder` としてコードされる場合、色成分 `cIdx` の履歴カウンタ `StatCoeff` は、以下のように更新される。

【数3】

【式3】

$$\text{StatCoeff}[cIdx] = (\text{StatCoeff}[cIdx] + \text{Floor}(\text{Log2}(\text{abs_remainder}[cIdx])) + 2) \gg 1$$

TUにおける1番目の非ゼロのゴロム・ライスコード変換係数が `dec_abs_level` としてコードされる場合、色成分 `cIdx` の履歴カウンタ `StatCoeff` は、以下のように更新される。

【数4】

【式4】

$$\text{StatCoeff}[cIdx] = (\text{StatCoeff}[cIdx] + \text{Floor}(\text{Log2}(\text{dec_abs_level}[cIdx])) + 2) \gg 1$$

次のTUをデコードする前に、更新された `StatCoeff` は、式2に従って次のTUの置換変数 `HistValue` を計算するために用いられることができる。

【0060】

ウェーブフロント並列処理 (`Wavefront Parallel Processing`、WPP)

【0061】

WPPは、並列コーディングメカニズムを提供するように設計されている。VVCにおいてWPPが有効にされる場合、フレーム、タイル、又はスライスの各CTU行が分離パーティション (`separation partition`) を構成する。WPPは、SPS要素 `sps_entropy_coding_sync_enabled_flag` によって有効／無効にされる。図7は、WPPが有効にされたタイルの一例を示す図である。図7では、タイルの各CTU行は、その前のCTU行に対して1つのCTUの遅延で処理される。このようにして、各CTU行の末にパレットコーディング (`palette coding`) が有効にされた場合、連続するCTU行間の依存性は、CABACコンテキスト変数及びパレット予測子 (`palette predictor`) を除いて、パーティション境界で破壊されない。コーディング効率の潜在的な損失を軽減するために、適応されたCABACコンテキスト変数及びパレット予測子の内容は、前のCTU行の1番目のコードされたCTUから現在のCTU行の1番目のCTUに伝播される。WPPは、CTUの通常のラスタスキャン順序を変更しない。

【0062】

WPPが有効にされる場合、パーティション (例えば、タイル、スライス、又はフレームなど) におけるCTU行の数までの数を有するスレッドが並列に動作し、個々のCTU行を処理することができる。WPPをデコーダーに利用することにより、各デコーディングスレッドはパーティションの単一のCTU行を処理する。スレッド処理のスケジューリングは整えられる必要があり、それによって、各CTUに対して、前のCTU行における上部に隣接するCTUのデコーディングが完成されるようにする。WPPの追加の小さなオーバーヘッドが加えられ、それによって、最後のCTU行を除く各CTU行における1番目のCTUのコーディングが終了した後、すべてのCABACコンテキスト変数及びパレット予測子の内容を記憶することができる。

【0063】

高ビット深度及び高ビットレートのビデオコーディングで上述した履歴ベースのライスパラメータ導出が有効にされる場合、前のCTU行における最後の `StatCoeff` が、現在のCTU行における1番目のTUに伝えられる。その結果、WPPが同時に有効に

される場合、このプロセスはWPPを妨げ、WPPの並列性を壊す。本開示では、並列コーディング（例えば、WPP）が有効にされる場合の上記問題を解決するためのいくつかの解決策が提案される。

【0064】

一実施形態では、履歴カウンタStatCoeffを計算する際、異なるCTU行におけるCTU間の依存性が除去され、それによって、並列コーディングに対する履歴ベースのライスパラメータ導出の妨害が除去される。この実施形態では、前のCTU行から得られた履歴カウンタStatCoeff値を利用するのではなく、StatCoeff[cIdx]の初期値を利用して、パーティション（例えば、フレーム、タイル、又はスライス）の各CTU行における1番目のabs_remainder[cIdx]又はdec_10064abs_level[cIdx]をコードする。ここで、cIdxは色成分のインデックスである。

10

【0065】

一例として、StatCoeff[cIdx]の初期値は以下のように確定されることができる。

【数5】

【式5】

$$\text{StatCoeff}[\text{idx}] = 2 * \text{Floor}(\text{Log2}(\text{BitDepth} - 10))$$

20

ここで、BitDepthはルーマ又はクロマ配列のサンプルのビット深度を指定し、Floor(x)はx以下の最大の整数を表す。別の例として、StatCoeff[cIdx]の初期値は以下のように確定されることができる。

【数6】

【式6】

$$\text{StatCoeff}[\text{idx}] = \text{Clip}(\text{MIN_Stat}, \text{MAX_Stat}, (\text{int})((19 - \text{QP}) / 6)) - 1$$

ここで、MIN_Stat及びMAX_Statは2つの予め定義された整数であり、QPは各スライスの初期QPであり、Clip()は以下のように定義された演算である。

30

【数7】

【式7】

$$\text{Clip}(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad \text{otherwise} \end{cases}$$

40

【0066】

パーティション（例えば、フレーム、タイル、又はスライス）の各CTU行の1番目のTUをコードする前に、置換変数HistValueが以下のように計算される。

【数8】

【式8】

$$\text{HistValue}[\text{cIdx}] = 1 \ll \text{StatCoeff}[\text{cIdx}]$$

50

`HistValue`は、上述したローカルサム変数`locSumAbs`を計算するために用いられることができる。`HistValue`は、指数移動平均のプロセスを通じて、1番目の非ゼロのゴロム・ライスコード変換係数(`abs_remainder[cIdx]`)又は`dec_abs_level[cIdx]`)から、TUごとに1回更新される。TUにおける1番目の非ゼロのゴロム・ライスコード変換係数が`abs_remainder`としてコードされる場合、色成分`cIdx`の履歴カウンタ`StatCoeff`は、以下のように更新される。

【数9】

【式9】

$$\text{StatCoeff}[cIdx] = (\text{StatCoeff}[cIdx] + \text{Floor}(\text{Log2}(\text{abs_remainder}[cIdx])) + 2) \gg 1$$

10

TUにおける1番目の非ゼロのゴロム・ライスコード変換係数が`dec_abs_level`としてコードされる場合、色成分`cIdx`の履歴カウンタ`StatCoeff`は、以下のように更新される。

【数10】

【式10】

$$\text{StatCoeff}[cIdx] = (\text{StatCoeff}[cIdx] + \text{Floor}(\text{Log2}(\text{dec_abs_level}[cIdx])) \gg 1$$

20

【0067】

更新された`StatCoeff[cIdx]`は、現在のCTUの次のTU又は現在のCTU行における次のCTUの1番目のTUに用いられるように、式8に示されたように置換変数`HistValue`を計算するために用いられる。

【0068】

図8は、フレーム802及び当該フレームに含まれるCTUの例を示す図である。この例では、フレーム802は、タイル804A及びタイル804Bという2つのタイルを含む。タイル804Aは、4つのCTU行(CTU行1~CTU行4)を含む。1番目のCTU行はCTU0~CTU9を含み、2番目のCTU行はCTU10~CTU19を含み、以下はこのように類推する。同様に、タイル804Bも4つのCTU行(CTU行1'~CTU行4')を含む。1番目のCTU行は、10のCTU(CTU0'~CTU9')を含み、2番目のCTU行はCTU10'~CTU19'を含み、以下はこのように類推する。

30

【0069】

本実施形態によれば、タイル804Aの`StatCoeff[cIdx]`の初期値は、式5又は式6に従って確定されることができる。CTU行1~CTU行4のうちの各CTU行の1番目のTUをコードする前に、置換変数`HistValue[cIdx]`は、`StatCoeff[cIdx]`の初期値を利用して式8を利用して計算される。例えば、CTU0の1番目のTUをコードする前に、変数`HistValue`は式8を利用して計算される。`HistValue`のこの値は、1番目のTUにおける係数のローカルサム変数`locSumAbs`を確定するために用いられ、さらに、この`locSumAbs`は、1番目のTUの対応する係数のライスパラメータを確定するために用いられる。現在のCTU0の1番目のTUを処理する際、履歴カウンタ`StatCoeff`は、式9又は式10に従って更新されることができる。CTU0における2番目のTUを処理する前に、`StatCoeff`の現在値は、式8に従って2番目のTUの`HistValue`を確定するために用いられる。その後、同様の処理が2番目のTUに用いられ、それによって、`HistValue`を利用してライスパラメータを確定し、`StatCoeff`を更新する。CTU1における1番目のTUに対して、`HistValue`は、CTU0におけるTUからの最後の`StatCoeff`を利用して式8に従って計算される。このプロセスは

40

50

、現在のCTU行1における最後のCTU（CTU9）の処理が完了されるまで繰り返されることができる。

【0070】

タイル804Aの2番目のCTU行に対して、CTU10（2番目のCTU行の1番目のCTU）の1番目のTUをコードする前に、履歴カウンタStatCoeffが式5又は式6に従って初期化される。2番目のCTU行のCTUにおけるTUに対して、CTU行1について上述した処理と同様の処理が実行される。同様に、CTU20及びCTU30の各々の1番目のTUをコードする前に、変数StatCoeffが式5又は式6に従って再び初期化される。

【0071】

タイル804Bも同様の方法で処理されることができる。CTU行1'～CTU行4'（即ち、CTU0'、CTU10'、CTU20'、及びCTU30'）の各々の1番目のTUをコードする前に、StatCoeff[cIdx]の値が式5又は式6に従って初期化され、履歴カウンタHistValueが式8を利用して計算される。計算された履歴カウンタHistValueは、対応するCTU行の1番目のCTU及び残りのCTUにおけるTUのLocSumAbs及びライスパラメータを計算するために用いられる。さらに、履歴カウンタStatCoeffは、式9又は式10に従って、各TUにおいて最多で1回更新されることができ、StatCoeffの更新値は、同じCTU行における次のTUのHistValueを確定するために用いられる。

【0072】

図8は、タイル804A及びタイル804Bという2つのタイルを含むフレーム802として説明されているが、同じプロセスが、複数のタイルを含むスライス、複数のスライスを含むフレームなどの他のシナリオにも適用される。これらのシナリオのいずれにおいても、パーティション（例えば、フレーム、タイル、又はスライス）の各CTU行における1番目のTUのコーディングの前に、履歴カウンタStatCoeff[cIdx]の値が初期値にリセットされて、ライスパラメータ導出におけるCTU行の依存性を除去する。

【0073】

下線で示されたVVCの可能な規格変更は、以下のように規定されている。

10

20

30

40

50

コンテキスト変数の同期プロセスは、以下のように呼び出される。

- `availableFlagT` が 1 に等しい場合、以下が適用される。
- `TableStateIdx0Wpp` 及び `TableStateIdx1Wpp` を入力として、エラー！参照ソースが見つからない。節に規定されたコンテキスト変数の同期プロセスが呼び出される。
- `sps_palette_enabled_flag` が 1 に等しい場合、0 節に規定されたパレット予測子の同期プロセスが呼び出される。
- `sps_persistent_Rice_adaptation_enabled_flag` が 1 に等しい場合、9. 3. 2. x1 節に規定されたライスパラメータ導出の同期プロセスが呼び出される。

10

9. 3. 2. x1 ライスパラメータ導出の同期プロセス

このプロセスは、以下のように配列 `StatCoeff` の値を同期させる。

```
for( cIdx = 0; cIdx < 3; cIdx++ ) {
    StatCoeff[ cIdx ] = 2 * Floor( Log2( BitDepth - 10 ) )
}
```

20

【0074】

VVC の 9. 3. 2. 1 に関する別の可能な規格変更は、以下のように規定されている。

- 配列 `StatCoeff[ i ]` ($i = 0.. 2$) は、以下のように初期化される。

$$\text{StatCoeff}[i] = \text{sps_persistent_rice_adaptation_enabled_flag} ? 2 * \text{Floor}(\text{Log2}(\text{BitDepth} - 10)) : 0$$
- CTU がスライス又はタイルにおける一番目の CTU である場合、コンテキスト変数の初期化プロセスは 9. 3. 2. 2 従属節に規定されたように呼び出され、配列 `PredictorPaletteSize[ chType ]` ($chType = 0, 1$) は 0 に初期化される。
- そうではない場合、`sps_entropy_coding_sync_enabled_flag` が 1 に等しく、且つ `CtbAddrX` が `CtbToTileColBd[ CtbAddrX ]` に等しいとき、以下が適用される。

30

40

【0075】

ビデオサンプルのビット深度

【0076】

VVC バージョン 2 によってサポートされる入力ビデオのビット深度は 10 ビット以上であることができる。ビデオの比較的高いビット深度は、デコードされたビデオに比較的高いビジュアル品質及び比較的小さい圧縮歪みを提供することができる。入力ビデオの高

50

いビット深度をサポートするために、対応するSPS（シーケンスパラメータセット）シンタックス要素 `sps_bitdepth_minus8` 及びVPS（ビデオパラメータセット）シンタックス要素 `vps_ols_dpb_bitdepth_minus8[i]` のセマンティクスは、以下のように変更されることができる。

`sps_bitdepth_minus8` は、ルーマ及びクロマ配列のサンプルのビット深度 `BitDepth` と、ルーマ及びクロマの量子化パラメータ範囲オフセットの値 `QpBdOffset` とを以下のように指定する。

【数 1 1】

【式 x 1】

$$\text{BitDepth} = 8 + \text{sps_bitdepth_minus8}$$

10

【式 x 2】

$$\text{QpBdOffset} = 6 * \text{sps_bitdepth_minus8}$$

`sps_bitdepth_minus8` は、0 から 8 の範囲内にある（0 及び 8 を含む）。

`sps_video_parameter_set_id` が 0 より大きく、且つ SPS が、VPS によって指定された i 番目（ i が 0 から `NumMultiLayerOls-1` までの範囲内にあり、0 及び `NumMultiLayerOls-1` を含む）の多層OLSに含まれる層によって参照される場合、`sps_bitdepth_minus8` の値が `vps_ols_dpb_bitdepth_minus8[i]` の値以下であることは、ビットストリーム一致性（`bitstream conformance`）の要件である。

20

`vps_ols_dpb_bitdepth_minus8[i]` は、 i 番目の多層OLSのCVSにおけるCLVSによって言及されたすべてのSPSの `sps_bitdepth_minus8` の最大許容値を指定する。`vps_ols_dpb_bitdepth_minus8[i]` の値は、0 から 8 の範囲内にある（0 及び 8 を含む）。

注 2 – i 番目の多層OLSをデコードすることについて、デコーダーは、シンタックス要素 `vps_ols_dpb_pic_width[i]`、`vps_ols_dpb_pic_height[i]`、`vps_ols_dpb_chroma_format[i]`、及び `vps_ols_dpb_bitdepth_minus8[i]` の値に基づいて、DPBにメモリを安全に割り当てることができる。

30

【0077】

以上から分かるように、ルーマ及びクロマ配列のサンプルのビット深度 `BitDepth` は、SPSシンタックス要素 `sps_bitdepth_minus8` に基づいて、式 x 1 に従って導出されることができる。確定された `BitDepth` 値により、履歴カウンタ `StatCoeff`、置換変数 `HistValue`、及びライスパラメータは、上述のように導出されることができる。

40

【0078】

VPSシンタックス要素 `vps_ols_dpb_bitdepth_minus8[i]` は、デコードピクチャバッファ（DPB）のサイズを導出するために用いられることができる。コードビットストリームには複数のビデオレイヤが存在することができる。ビデオパラメータセットは、対応するシンタックス要素を指定するために用いられる。ビデオデコーディングでは、DPBは参照ピクチャを記憶するために用いられ、それによって、前にコードされたピクチャは、他のピクチャをコードする際に用いられる予測信号を生成するために用いられることができる。また、デコードされたピクチャを正しい順序で出力及び／又は表示できるように、DPBはデコードされたピクチャを改めて並べるために

50

用いられることもできる。さらに、DPBは仮想参照デコーダー(hypothetical reference decoder)のために指定された出力遅延に用いられることもできる。デコードされたピクチャはDPBに、仮想参照デコーダーのために指定された予め確定された時間と定められることができ、デコードされたピクチャは、予め確定された時間が経過した後に出力される。

【0079】

DPBにメモリを安全に割り当てるために、DPBのサイズは、シンタックス要素vps_ols_dpb_pic_width[i]、vps_ols_dpb_pic_height[i]、vps_ols_dpb_chroma_format[i]、vps_ols_dpb_bitdepth_minus8[i]によって以下のように決定される。

```
picture_size1 (in bits) = vps_ols_dpb_pic_width[i] * vps_ols_dpb_pic_height[i] * (vps_ols_dpb_bitdepth_minus8[i] + 8)
```

```
if (vps_ols_dpb_chroma_format[i] == 0) // Monochrome
```

```
    picture_size = picture_size1;
```

```
else if (vps_ols_dpb_chroma_format[i] == 1) // 4:2:0
```

```
    picture_size = 1.5 * picture_size1;
```

```
else if (vps_ols_dpb_chroma_format[i] == 2) // 4:2:2
```

```
    picture_size = 2 * picture_size1;
```

```
else if (vps_ols_dpb_chroma_format[i] == 3) // 4:4:4
```

```
    picture_size = 3 * picture_size1
```

DPBのサイズはpicture_sizeによって確定される。言い換えれば、DPBのサイズは、サンプルのクロマフォーマットに基づいて確定されることができる。ビデオフレームがモノクロフレーム(monochrome frame)である場合、バッファリングされようとするフレームのサイズは基本ピクチャサイズpicture_size1に確定される。カラービデオフレームのカラーサブサンプリング(color subsampling)が4:2:0である場合、フレームのサイズは基本ピクチャサイズpicture_size1の1.5倍に確定され、カラービデオフレームのカラーサブサンプリングが4:2:2である場合、フレームのサイズは基本ピクチャサイズpicture_size1の2倍に確定され、カラービデオフレームのカラーサブサンプリングが4:4:4である場合、フレームのサイズは基本ピクチャサイズpicture_size1の3倍に確定される。また、DPBのサイズは、DPBに記憶されようとするフレームの数にカラーサブサンプリングに応じたフレームのサイズを乗算することで得た値に確定されることができる。

【0080】

図9は、本開示のいくつかの実施形態に係る、ビデオのパーティションをエンコードするプロセス900の一例を示す図である。1つ又は複数のコンピューティングデバイス(例えば、ビデオエンコーダー100を実現するコンピューティングデバイス)は、適切なプログラムコード(例えば、エントローピーコーディングモジュール116を実現するプログラムコード)を実行することによって、図9に示された操作を実現する。説明の目的として、プロセス900は、図面に示されたいくつかの例を参照しながら説明される。しかしながら、他の実施例も可能である。

【0081】

ブロック902において、プロセス900は、ビデオ信号のパーティションにアクセスすることを含む。パーティションは、ビデオフレーム、スライス、タイル、又は、エンコーディングを実行する際にビデオエンコーダーによってユニットとして処理される任意のタイプのパーティションであり得る。パーティションは、図8に示されたような、CTU行に配列されたCTUセットを含む。図6の例に示されたように、各CTUは1つ又は複数のCTUを含み、且つ各CTUはエンコーディング用の複数のTUを含む。

【0082】

10

20

30

40

50

ブロック 906～914 を含むブロック 904 において、プロセス 900 は、パーティションにおける C T U セットの各 C T U を処理することを含み、それによって、パーティションをビットにエンコードする。ブロック 906 において、プロセス 900 は、並列コーディングメカニズムが有効にされたか否か、且つ、現在の C T U が C T U 行の 1 番目の C T U であるか否かを確定することを含む。いくつかの例では、並列コーディングは、並列コーディングが無効にされたことを指示する値 0 と、並列コーディングが有効にされたことを指示する値 1 とを有するフラグによって指示されることができる。並列コーディングメカニズムが有効にされたこと、及び現在の C T U が C T U 行の 1 番目の C T U であると確定された場合、プロセス 900 は、ブロック 908 において、履歴カウンタ *StateCoeff* を初期値に設定することを含む。上述したように、履歴ベースのライスパラメータ導出が有効にされた場合、履歴カウンタの初期値は、式 5 又は式 6 に従って設定されることができ、そうではない場合、履歴カウンタの初期値はゼロに設定される。

【0083】

並列コーディングメカニズムが有効にされていない、もしくは現在の C T U が C T U 行の 1 番目の C T U ではないと確定された場合、又は、ブロック 908 において履歴カウンタを設定した後、プロセス 900 は、ブロック 910 において、履歴カウンタに基づいて C T U における T U のライスパラメータを計算することを含む。図 6～図 8 について上記に詳述したように、ブロック 908 において履歴カウンタがリセットされる場合、C T U における T U のライスパラメータは、リセットされた履歴カウンタ又は後続に更新される履歴カウンタに基づいて計算される。ブロック 908 において履歴カウンタがリセットされない場合、C T U における T U のライスパラメータは、前の C T U において更新された履歴カウンタ又は現在の C T U において後続に更新される履歴カウンタに基づいて計算される。

【0084】

ブロック 912 において、プロセス 900 は、（例えば、V V C 規格に規定された、T R と限定された k 次指数ゴロム（E G k）との組み合わせを介して、）計算されたライスパラメータに基づいて、C T U における T U をバイナリ表現にエンコードすることを含む。ブロック 914 において、プロセス 900 は、ビデオのビットストリームに含まれるために、C T U のバイナリ表現をビットにエンコードすることを含む。エンコーディングは、例えば、上述した C A B A C を利用して実行されることができる。ブロック 916 において、プロセス 900 は、エンコードされたビデオビットストリームを出力することを含む。

【0085】

図 10 は、本開示のいくつかの実施形態に係る、ビデオのパーティションをデコードするプロセス 1000 の一例を示す図である。1 つ又は複数のコンピューティングデバイスは、適切なプログラムコードを実行することによって、図 10 に示された操作を実現する。例えば、ビデオデコーダー 200 を実現するコンピューティングデバイスは、エントロピーデコーディングモジュール 216、逆量子化モジュール 218、及び逆変換モジュール 219 のために用いられるプログラムコードを実行することによって、図 10 に示された操作を実現することができる。説明の目的として、プロセス 1000 は、図面に示されたいくつかの例を参照しながら説明される。しかしながら、他の実施例も可能である。

【0086】

ブロック 1002 において、プロセス 1000 は、ビデオ信号のパーティションを表すバイナリ文字列又はバイナリ表現にアクセスすることを含む。パーティションは、ビデオフレーム、スライス、タイル、又は、エンコーディングを実行する際にビデオエンコーダーによってユニットとして処理される任意のタイプのパーティションであり得る。パーティションは、図 8 に示されたような、C T U 行に配列された C T U セットを含む。図 6 の例に示されたように、各 C T U は 1 つ又は複数の C T U を含み、且つ各 C T U はエンコーディング用の複数の T U を含む。

【0087】

10

20

30

40

50

ブロック1006～1014を含むブロック1004において、プロセス1000は、パーティションのデコードされたサンプルを生成するために、パーティションにおけるCTUセットの各CTUのバイナリ文字列を処理することを含む。ブロック1006において、プロセス1000は、並列コーディングメカニズムが有効にされたか否か、且つ、現在のCTUがCTU行の1番目のCTUであるか否かを確定することを含む。並列コーディングは、並列コーディングが無効にされたことを指示する値0と、並列コーディングが有効にされたことを指示する値1とを有するフラグによって指示されることができる。並列コーディングメカニズムが有効にされたこと、及び現在のCTUがCTU行の1番目のCTUであると確定された場合、プロセス1000は、ブロック1008において、履歴カウンタStatCoeffを初期値に設定することを含む。上述したように、履歴ベースのライスパラメータ導出が有効にされた場合、履歴カウンタの初期値は、式5又は式6に従って設定されることができ、そうではない場合、履歴カウンタの初期値はゼロに設定される。

10

【0088】

並列コーディングメカニズムが有効にされていない、もしくは現在のCTUがCTU行の1番目のCTUではないと確定された場合、又は、ブロック1008において履歴カウンタを設定した後、プロセス1000は、ブロック1010において、履歴カウンタに基づいてCTUにおけるTUのライスパラメータを計算することを含む。図6～図8について上記に詳述したように、ブロック1008において履歴カウンタがリセットされる場合、CTUにおけるTUのライスパラメータは、リセットされた履歴カウンタ又は後続に更新される履歴カウンタに基づいて計算される。ブロック1008において履歴カウンタがリセットされない場合、CTUにおけるTUのライスパラメータは、前のCTUにおいて更新された履歴カウンタ又は現在のCTUにおいて後続に更新される履歴カウンタに基づいて計算される。

20

【0089】

ブロック1012において、プロセス1000は、（例えば、VVC規格に規定された、TRと限定されたk次指数ゴロム（EGk）との組み合わせを介して、）計算されたライスパラメータに基づいて、CTUにおけるTUのバイナリ文字列又はバイナリ表現を係数値にデコードすることを含む。ブロック1014において、プロセス1000は、例えば、図2について上述したような逆量子化及び逆変換を介して、CTUにおけるTUのピクセル値を再構成することを含む。ブロック1016において、プロセス1000は、ビデオのデコードされたパーティションを出力することを含む。

30

【0090】

別の実施形態では、履歴カウンタStatCoeffを計算する際のCTU間の依存性は、WPPのような並列コーディングメカニズムにおけるCTU間の依存性にアライメントする。例えば、パーティション（例えば、フレーム、タイル又はスライス）のCTU行の履歴カウンタStatCoeffは、前のCTU行の最初のN個以下のCTUにおける係数値に基づいて計算され、ここで、Nは、並列コーディングメカニズムにおいて許容される2つの連続するCTU行の間の最大遅延である。このようにして、履歴カウンタStatCoeffを計算する際の2つの連続するCTU行におけるCTU間の依存性は、並列処理を実行する際のCTU間の依存性以下に制限される（それによって、アライメントする）。

40

【0091】

この実施形態は、記憶同期プロセスを利用して実現されることができる。例えば、上述のWPPでは、2つの連続するCTU行の間の遅延は1つのCTUであり、従って、N=1である。記憶のプロセスでは、各CTU行（最後のCTU行を除く）における1番目のCTUの最後のTUがコードされた後、StatCoeff[cIdx]はストレージ変数StatCoeffWpp[cIdx]に保存されることができる。1番目のCTU行以外の各CTU行に対して、1番目のTUのコーディングの前に、ライスパラメータ導出の同期プロセスが適用される。同期プロセスにおいて、StatCoeff[cIdx]

50

は、前のCTU行からの保存されたStatCoeffWpp[cIdx]と同期される。

【0092】

上述したように、各CTU行における1番目のTUのコーディングの前に、変数HistValueは以下のように計算される。

【数12】

【式11】

$$\text{HistValue}[\text{cIdx}] = 1 \ll \text{StatCoeff}[\text{cIdx}]$$

10

現在のCTU行がパーティションの1番目のCTU行である場合、StatCoeff[cIdx]は式5又は式6に従って初期化されることができる。計算されたHistValueは、ローカルサム変数locSumAbsを確定するために用いられることができ、このローカルサム変数locSumAbsは、現在のCTUにおけるTUのライスパラメータを確定するために用いられる。StatCoeffは、式9及び式10について上述したような指数移動平均のプロセスを通じて、1番目の非ゼロのゴロム・ライスコード変換係数(abs_remainder[cIdx]又はdec_abs_level[cIdx])から、TUごとに1回更新されることができる。

【0093】

1番目のCTU行における一番目のCTUの最後のTUがコードされた後、StatCoeff[cIdx]は、以下のような記憶ステップにおいてStatCoeffWpp[cIdx]として保存されることができる。

20

【数13】

【式12】

$$\text{StatCoeffWpp}[\text{cIdx}] = \text{StatCoeff}[\text{cIdx}]$$

1番目のCTU行における残りのCTUのコーディングは、第1の実施形態について上述したものと同様の方法で行われることができる。

【0094】

30

2番目のCTU行及び後続の任意のCTU行における1番目のTUのコーディングの前に、StatCoeff[cIdx]は同期ステップによって得られることができる。

【数14】

【式13】

$$\text{StatCoeff}[\text{cIdx}] = \text{StatCoeffWpp}[\text{cIdx}]$$

得られたStatCoeff[cIdx]値を利用して、式11に従ってHistValueが計算される。CTU行の残りのプロセスは、1番目のCTU行と同じである。

【0095】

40

可能なVVC規格変更は以下のように規定されている（変更は下線で示されている）。

9. 3. 1 概要

sps_persistent_Rice_adaptation_enabled_flagが1に等しい場合、前のTUから蓄積された統計データを利用したライスパラメータ導出の記憶プロセスは、以下のように適用される。

ー7. 3. 11. 2節におけるCTUSインタックスの解析及び8. 1. 2節におけるCTUにおける最後のCUのデコーディングプロセスを終了する際、sps_entropy_coding_sync_enabled_flagが1に等しく、且つCtbAddrXがCtbToTileColBd[CtbAddrX]に等しく、9. 3. 2. x2節に規定されたライスパラメータ導出の記憶プロセスが呼び出される。

10

9. 3. 2. 1 概要

コンテキスト変数の同期プロセスは、以下のように呼び出される。

ーavailableFlagTが1に等しい場合、以下が適用される。

ーTableStateIdx0Wpp及びTableStateIdx1Wppを入力として、9. 3. 2. 4節に規定されたコンテキスト変数の同期プロセスが呼び出される。

ーsps_palette_enabled_flagが1に等しい場合、9. 3. 2. 7節に規定されたパレット予測子の同期プロセスが呼び出される。

20

ーsps_persistent_Rice_adaptation_enabled_flagが1に等しい場合、9. 3. 2. x1節に規定されたライスパラメータ導出の同期プロセスが呼び出される。

9. 3. 2. x2 ライスパラメータ導出の記憶プロセス

このプロセスは以下のように、配列StatCoeffの値を配列StatCoeffWppに記憶する。

```
for( cIdx = 0; cIdx < 3; cIdx++ ) {  
    StatCoeffWpp[ cIdx ] = StatCoeff[ cIdx ]  
}
```

30

9. 3. 2. x1 ライスパラメータ導出の同期プロセス

このプロセスは以下のように、配列StatCoeffの値と配列StatCoeffWppとを同期させる。

```
for( cIdx = 0; cIdx < 3; cIdx++ ) {  
    StatCoeff[ cIdx ] = StatCoeffWpp[ cIdx ]  
}
```

40

【0096】

代替の履歴ベースのライスパラメータ導出

【0097】

履歴ベースのライスパラメータ導出は、代替的な方法で実現されることができる。この代替的な実施態様では、CTUがパーティション（例えば、ピクチャ、スライス、又はタイル）における1番目のCTUである場合、HistValueは以下のように、StatCoeff[cIdx]の初期値で初期化される。

【数15】

50

【式 1 4】

$$\text{HistValue} = \text{sps_persistent_Rice_adaptation_enabled_flag} ? 1 \ll \text{StatCoeff}[\text{cIdx}] : 0$$

【0 0 9 8】

この初期の HistValue は、 HistValue が以下のルールに従って更新されるまで、1 番目の $\text{abs_remainder}[\text{cIdx}]$ 又は $\text{dec_abs_level}[\text{cIdx}]$ をコードするために用いられる。TU における 1 番目の非ゼロのゴロム・ライスコード変換係数が abs_remainder としてコードされるとき、色成分 cIdx の履歴カウンタは、以下のように更新される。

10

【数 1 6】

【式 1 5】

$$\text{StatCoeff}[\text{cIdx}] = (\text{StatCoeff}[\text{cIdx}] + \text{Floor}(\text{Log2}(\text{abs_remainder}[\text{cIdx}])) + 2) \gg 1$$

TU における 1 番目の非ゼロのゴロム・ライスコード変換係数が dec_abs_level としてコードされるとき、色成分 cIdx の履歴カウンタは、以下のように更新される。

【数 1 7】

【式 1 6】

$$\text{StatCoeff}[\text{cIdx}] = (\text{StatCoeff}[\text{cIdx}] + \text{Floor}(\text{Log2}(\text{dec_abs_level}[\text{cIdx}]))) \gg 1$$

20

【0 0 9 9】

履歴カウンタ $\text{StatCoeff}[\text{cIdx}]$ が更新されると、式 1 7 に示されるように HistValue が更新され、更新された HistValue は、新しい $\text{StatCoeff}[\text{cIdx}]$ 及び $\text{HistValue}[\text{cIdx}]$ が再び更新されるまで、残りの abs_remainder 及び dec_abs_level シンタックス要素のライスパラメータの導出のために用いられる。

【数 1 8】

【式 1 7】

$$\text{HistValue}[\text{cIdx}] = 1 \ll \text{StatCoeff}[\text{cIdx}]$$

30

【0 1 0 0】

現在の VVC 規格に基づいて、可能な規格変更は以下のように規定されている。

7.3.11.11 節（残差コーディングシンタックス）を以下のように変更する（追加部分には下線が引かれている）。

40

50

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descrip tor
...	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight - (log2SbW + log2SbH))) - 1	
~~HistValue = sps_persistent_rice_adaptation_enabled_flag ?~~ (1 << StatCoeff[cIdx]) : 0	
updateHist = sps_persistent_rice_adaptation_enabled_flag ? 1 : 0	
...	
for(n = firstPosMode0; n > firstPosModel; n -) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(abs_level_gtx_flag[n][1]) {	
abs_remainder[n]	ae(v)
if(updateHist && abs_remainder[n] > 0) {	
StatCoeff[cIdx] = (StatCoeff[cIdx] + Floor(Log2(abs_remainder[n])) + 2) >> 1	
updateHist = 0	
HistValue = 1 << StatCoeff[cIdx]	
}	
}	
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] +2 * abs_remainder[n]	
}	
for(n = firstPosModel; n >= 0; n -) {	

10

20

30

40

50

$xC = (xS \ll \log2SbW) +$ $DiagScanOrder[\log2SbW][\log2SbH][n][0]$		
$yC = (yS \ll \log2SbH) +$ $DiagScanOrder[\log2SbW][\log2SbH][n][1]$		
$if(sb_coded_flag[xS][yS]) \{$		
$dec_abs_level[n]$	$ae(v)$	10
$if(updateHist \ \&\& \ dec_abs_level[n] > 0) \{$		
$StatCoeff[cIdx] = (StatCoeff[cIdx] +$ $Floor(Log2(dec_abs_level[n]))) \gg 1$		
$updateHist = 0$		
$HistValue = 1 \ll StatCoeff[cIdx]$		
$\}$		20
$\}$		
$if(AbsLevel[xC][yC] > 0) \{$		
$if(lastSigScanPosSb == -1)$		
$lastSigScanPosSb = n$		
$firstSigScanPosSb = n$		
$\}$		
$if(sh_dep_quant_used_flag)$		30
$QState =$ $QStateTransTable[QState][AbsLevel[xC][yC] \& 1]$		
$\}$		
$.....$		

【0101】

40

並列コーディングと代替の履歴ベースのライスパラメータ導出との間の依存性の衝突を解決するために、各CTU行における1番目のCTUの最後のTUがコードされた後、各色成分の $StatCoeff[cIdx]$ と $HistValue[cIdx]$ の両方が保存される。保存された $StatCoeff[cIdx]$ 及び $HistValue[cIdx]$ の値は、後続のCTU行の1番目のCTUの1番目のTUが処理される前に、 $StatCoeff[cIdx]$ 及び $HistValue[cIdx]$ を初期化するために用いられることができる。

【0102】

この実施形態は、記憶同期プロセスを利用して実現されることもできる。例えば、記憶プロセスでは、各CTU行における1番目のCTUの最後のTUが処理された後、Sta

50

`t C o e f f [ c I d x]` 及び `H i s t V a l u e [ c I d x]` は、それぞれ記憶変数（例えば、式 18 に示される `S t a t C o e f f W p p [ c I d x]` 及び式 19 に示される `H i s t V a l u e W p p [ c I d x]`）に保存されることができる。

【数 19】

【式 18】

`StatCoeffWpp[cIdx] = StatCoeff[cIdx]`

【式 19】

`HistValueWpp[cIdx] = HistValue[cIdx]`

10

【0103】

1 番目の C T U 行以外の各 C T U 行に対して、1 番目の T U のコーディングの前に、ライスパラメータ導出の同期プロセスが適用される。例えば、式 20 に示されるように、`S t a t C o e f f [ c I d x]` は前の C T U 行からの保存された `S t a t C o e f f W p p [ c I d x]` と同期され、式 21 に示されるように、`H i s t V a l u e [ c I d x]` は前の C T U 行からの保存された `H i s t V a l u e W p p [ c I d x]` と同期される。

【数 20】

【式 20】

`StatCoeff[cIdx] = StatCoeffWpp[cIdx]`

20

【式 21】

`HistValue[cIdx] = HistValueWpp[cIdx]`

同期変数 `H i s t V a l u e` は、`H i s t V a l u e` が更新されるまで、1 番目の `a b s _ r e m a i n d e r [ c I d x]` 又は `d e c _ a b s _ l e v e l [ c I d x]` をコードするために用いられる。

【0104】

上述したように、式 15 又は式 16 に示されたように、`S t a t C o e f f [ c I d x]` は、1 番目の非ゼロのゴロム・ライスコード変換係数（`a b s _ r e m a i n d e r [ c I d x]` 又は `d e c _ a b s _ l e v e l [ c I d x]`）から、T U ごとに 1 回更新されることができる。履歴カウンタ `S t a t C o e f f [ c I d x]` が更新されると、`H i s t V a l u e` が式 17 に従って更新され、更新された `H i s t V a l u e` は、新しい `S t a t C o e f f [ c I d x]` 及び `H i s t V a l u e` が再び更新されるまで、残りの `a b s _ r e m a i n d e r` 及び `d e c _ a b s _ l e v e l` シンタックス要素のライスパラメータの導出のために用いられる。

30

【0105】

現行の V V C 規格に基づいて、下線で示された可能な規格変更は以下のように規定されている。

40

9. 3. 1 概要

sps_persistent_Rice_adaptation_enabled_flagが1に等しい場合、前のTUから蓄積された統計データを利用したライスパラメータ導出の記憶プロセスは、以下のように適用される。

－7. 3. 1 1. 2節におけるCTUシンタックスの解析及び8. 1. 2節におけるCTUにおける最後のCUのデコーディングプロセスを終了する際、sps_entropy_coding_sync_enabled_flagが1に等しく、且つCtbAddrXがCtbToTileColBd [CtbAddrX]に等しく、9. 3. 2. x 2節に規定されたライスパラメータ導出の記憶プロセスが呼び出される。

10

9. 3. 2. 1 概要

コンテキスト変数の同期プロセスは、以下のように呼び出される。

－availableFlagTが1に等しい場合、以下が適用される。

－TableStateIdx0Wpp及びTableStateIdx1Wppを入力として、9. 3. 2. 4節に規定されたコンテキスト変数の同期プロセスが呼び出される。

20

－sps_palette_enabled_flagが1に等しい場合、9. 3. 2. 7節に規定されたパレット予測子の同期プロセスが呼び出される。

－sps_persistent_Rice_adaptation_enabled_flagが1に等しい場合、9. 3. 2. x 1節に規定されたライスパラメータ導出の同期プロセスが呼び出される。

9. 3. 2. x 2 ライスパラメータ導出の記憶プロセス

このプロセスは以下のように、配列StatCoeff及びHistValueの値を配列StatCoeffWpp及びHistValueWPPに記憶する。

30

```
for( cIdx = 0; cIdx < 3; cIdx++ ) {
    StatCoeffWpp [ cIdx ] = StatCoeff[ cIdx ]
    HistValueWpp [ cIdx ] = HistValue[ cIdx ]
}
```

9. 3. 2. x 1 ライスパラメータ導出の同期プロセス

このプロセスは以下のように、配列StatCoeff及びHistValueの値と配列StatCoeffWpp及びHistValueWppとを同期させ

40

る。

```
for( cIdx = 0; cIdx < 3; cIdx++ ) {
    StatCoeff [ cIdx ] = StatCoeffWpp[ cIdx ]
    HistValue[ cIdx ] = HistValueWpp[ cIdx ]
}
```

50

【0106】

図11は、本開示のいくつかの実施形態に係る、ビデオのパーティションをエンコードするプロセス1100の例を示す図である。1つ又は複数のコンピューティングデバイス（例えば、ビデオエンコーダー100を実現するコンピューティングデバイス）は、適切なプログラムコード（例えば、エントローピーコーディングモジュール116を実現するプログラムコード）を実行することによって、図11に示された操作を実現する。説明の目的として、プロセス1100は、図面に示されたいくつかの例を参照しながら説明される。しかしながら、他の実施例も可能である。

【0107】

ブロック1102において、プロセス1100は、ビデオ信号のパーティションにアクセスすることを含む。パーティションは、ビデオフレーム、スライス、タイル、又は、エンコーディングを実行する際にビデオエンコーダーによってユニットとして処理される任意のタイプのパーティションであり得る。パーティションは、図8に示されたような、CTU行に配列されたCTUセットを含む。図6の例に示されたように、各CTUは1つ又は複数のCTUを含み、且つ各CTUはエンコーディング用の複数のTUを含む。

【0108】

ブロック1106～1118を含むブロック1104において、プロセス1100は、パーティションにおけるCTUセットの各CTUを処理することを含み、それによって、パーティションをビットにエンコードする。ブロック1106において、プロセス1100は、並列コーディングメカニズムが有効にされたか否か、且つ、現在のCTUがCTU行の1番目のCTUであるか否かを確定することを含む。いくつかの例では、並列コーディングは、並列コーディングが無効にされたことを指示する値0と、並列コーディングが有効にされたことを指示する値1とを有するフラグによって指示されることができる。並列コーディングメカニズムが有効にされたこと、及び現在のCTUがCTU行の1番目のCTUであると確定された場合、プロセス1100は、ブロック1107において、現在のCTU行がパーティションにおける1番目のCTU行であるか否かを確定することを含む。そうである場合、プロセス1100は、ブロック1108において、履歴カウンタStatCoeffを初期値に設定することを含む。上述したように、履歴カウンタの初期値は、式5又は式6に従って設定されることができる。現在のCTU行がパーティションにおける1番目のCTU行ではない場合、プロセス1100は、ブロック1109において、式13又は式20に示されたように、履歴カウンタStatCoeffを、履歴カウンタストレージ変数に記憶された値に設定することを含む。いくつかの例では、例えば、代替のライスパラメータ導出が利用される場合、置換変数HistValueの値は、式21に示されるように、記憶された値にリセットされることができる。

【0109】

並列コーディングメカニズムが有効にされていない、もしくは現在のCTUがCTU行の1番目のCTUではないと確定された場合、又は、ブロック1108もしくは1109において履歴カウンタの値を設定した後、プロセス1100は、ブロック1110において、履歴カウンタに基づいて（置換変数HistValueもリセットされた場合、さらに置換変数HistValueに基づいて）、CTUにおけるTUのライスパラメータを計算することを含む。上述したように（例えば、図8が参照として、又は代替のライスパラメータ導出において）、ブロック1108又はブロック1109において履歴カウンタの値がリセットされる場合、CTUにおけるTUのライスパラメータは、リセットされた履歴カウンタ又は後続に更新される履歴カウンタに基づいて計算される。ブロック1108又は1109において履歴カウンタがリセットされない場合、CTUにおけるTUのライスパラメータは、前のCTUにおいて更新された履歴カウンタ又は現在のCTUにおいて後続に更新される履歴カウンタに基づいて計算される。

【0110】

ブロック1112において、プロセス1100は、（例えば、VVC規格に規定された、TRと限定されたk次指数ゴロム（EGk）との組み合わせを介して、）計算されたラ

10

20

30

40

50

イスパラメータに基づいて、CTUにおけるTUをバイナリ表現にエンコードすることを含む。ブロック1114において、プロセス1100は、ビデオのビットストリームに含まれるために、CTUのバイナリ表現をビットにエンコードすることを含む。エンコーディングは、例えば、上述したCABACを利用して実行されることができる。

【0111】

ブロック1116において、プロセス1100は、並列コーディングが有効にされたか否か、且つ、CTUが現在のCTU行の1番目のCTUであるか否かを確定することを含む。そうである場合、プロセス1100は、ブロック1118において、式12又は式18に示されるように、履歴カウンタの値を履歴カウンタストレージ変数に記憶することを含む。いくつかの例では、例えば、代替のライスパラメータ導出が利用される場合、置換変数HistValueの値は、式19に示されるように、ストレージ変数に記憶されることができる。ブロック1120において、プロセス1100は、エンコードされたビデオビットストリームを出力することを含む。

【0112】

いくつかのシナリオでは、1番目のCTU行以外のCTU行におけるCTUはパーティションの境界に位置することができる。例えば、2番目のCTU行における1番目のCTUの上部に、パーティション内のCTUがない。これらのシナリオでは、このCTUの履歴カウンタは、記憶された値ではなく初期値に設定されることができる。この場合、図11においてブロック1107とブロック1109との間に新たなブロック1107'を追加して、CTUがパーティションの境界にあるか否か（例えば、CTUがパーティション内の上隣接CTUを有しない）を確定することができる。そうである場合、プロセス1100はブロック1108に進んで履歴カウンタを初期値に設定し、そうではない場合、プロセス1100はブロック1109に進んで履歴カウンタを記憶された値に設定する。図11の残りのブロックは同じままであることができる。

【0113】

図12は、本開示のいくつかの実施形態に係る、ビデオのパーティションをデコードするプロセス1200の例を示す図である。1つ又は複数のコンピューティングデバイスは、適切なプログラムコードを実行することによって、図12に示された操作を実現する。例えば、ビデオデコーダー200を実現するコンピューティングデバイスは、エントリピーデコーディングモジュール216、逆量子化モジュール218、及び逆変換モジュール219のために用いられるプログラムコードを実行することによって、図12に示された操作を実現することができる。説明の目的として、プロセス1200は、図面に示されたいくつかの例を参照しながら説明される。しかしながら、他の実施例も可能である。

【0114】

ブロック1202において、プロセス1000は、ビデオ信号のパーティションを表すバイナリ文字列又はバイナリ表現にアクセスすることを含む。パーティションは、ビデオフレーム、スライス、タイル、又は、エンコーディングを実行する際にビデオエンコーダーによってユニットとして処理される任意のタイプのパーティションであり得る。パーティションは、図8に示されたような、CTU行に配列されたCTUセットを含む。図6の例に示されたように、各CTUは1つ又は複数のCTUを含み、且つ各CTUはエンコーディング用の複数のTUを含む。

【0115】

ブロック1206～1218を含むブロック1204において、プロセス1200は、パーティションのデコードされたサンプルを生成するために、パーティションにおけるCTUセットの各CTUのバイナリ文字列を処理することを含む。ブロック1206において、プロセス1200は、並列コーディングメカニズムが有効にされたか否か、且つ、現在のCTUがCTU行の1番目のCTUであるか否かを確定することを含む。並列コーディングは、並列コーディングが無効にされたことを指示する値0と、並列コーディングが有効にされたことを指示する値1とを有するフラグによって指示されることができる。並列コーディングメカニズムが有効にされたこと、及び現在のCTUがCTU行の1番目の

10

20

30

40

50

C T Uであると確定された場合、プロセス1200は、ブロック1207において、現在のC T U行がパーティションにおける1番目のC T U行であるか否かを確定することを含む。そうである場合、プロセス1200は、ブロック1208において、履歴カウンタStatCoef fを初期値に設定することを含む。上述したように、履歴カウンタの初期値は、式5又は式6に従って設定されることができる。現在のC T U行がパーティションにおける1番目のC T U行ではない場合、プロセス1200は、ブロック1209において、式13又は式20に示されたように、履歴カウンタStatCoef fを、履歴カウンタストレージ変数に記憶された値に設定することを含む。いくつかの例では、例えば、代替のライスパラメータ導出が利用される場合、置換変数HistValueの値は、式21に示されるように、記憶された値にリセットされることができる。

10

【0116】

並列コーディングメカニズムが有効にされていない、もしくは現在のC T UがC T U行の1番目のC T Uではないと確定された場合、又は、ブロック1208もしくは1209において履歴カウンタを設定した後、プロセス1200は、ブロック1210において、履歴カウンタに基づいて（置換変数HistValueの値もリセットされた場合、さらに置換変数HistValueに基づいて）、C T UにおけるTUのライスパラメータを計算することを含む。上述したように（例えば、図8が参照として、又は代替のライスパラメータ導出において）、ブロック1208又はブロック1209において履歴カウンタの値がリセットされる場合、C T UにおけるTUのライスパラメータは、リセットされた履歴カウンタ又は後続に更新される履歴カウンタに基づいて計算される。ブロック1208又は1209において履歴カウンタがリセットされない場合、C T UにおけるTUのライスパラメータは、前のC T Uにおいて更新された履歴カウンタ又は現在のC T Uにおいて後続に更新される履歴カウンタに基づいて計算される。

20

【0117】

ブロック1212において、プロセス1200は、（例えば、VVC規格に規定された、TRと限定されたk次指数ゴロム（EGk）との組み合わせを介して、）計算されたライスパラメータに基づいて、C T UにおけるTUのバイナリ文字列又はバイナリ表現を係数値にデコードすることを含む。ブロック1214において、プロセス1200は、例えば、図2について上述したような逆量子化及び逆変換を介して、C T UにおけるTUのピクセル値を再構成することを含む。

30

【0118】

ブロック1216において、プロセス1200は、並列コーディングが有効にされたか否か、且つ、C T Uが現在のC T U行の1番目のC T Uであるか否かを確定することを含む。そうである場合、プロセス1200は、ブロック1218において、式12又は式18に示されるように、履歴カウンタの値を履歴カウンタストレージ変数に記憶することを含む。いくつかの例では、例えば、代替のライスパラメータ導出が利用される場合、置換変数HistValueの値は、式19に示されるように、ストレージ変数に記憶されることができる。ブロック1216において、プロセス1200は、ビデオのデコードされたパーティションを出力することを含む。

【0119】

40

別の実施形態では、WPP又は他の並列コーディングメカニズム及び履歴ベースのライスパラメータ導出は、ビットストリームに共存することが防止される。例えば、WPPが有効にされた場合、履歴ベースのライスパラメータ導出が有効にされないことができる。WPPが有効にされない場合、履歴ベースのライスパラメータ導出が有効にされることができる。同様に、履歴ベースのライスパラメータ導出が有効にされた場合、WPPが有効にされないことができる。一例として、シンタックスが以下のように変更されることができる。

7.3.2.22 シーケンスパラメータセット範囲拡張シンタックス（追加部分に下線が引かれている）

50

sps_range_extension() {	Descriptor
sps_extended_precision_flag	u(1)
if(sps_transform_skip_enabled_flag)	
sps_ts_residual_coding_rice_present_in_sh_flag	u(1)
sps_rrc_rice_extension_flag	u(1)
<u>if(sps_entropy_coding_sync_enabled_flag)</u>	
sps_persistent_rice_adaptation_enabled_flag	u(1)
sps_reverse_last_sig_coeff_enabled_flag	u(1)
}	

10

【 0 1 2 0 】

別の例として、対応するセマンティクスが以下のように変更される（変更は下線で示されている）。

20

30

40

50

`sps_entropy_coding_sync_enabled_flag`が1に等しいことは、SPSを参照する各ピクチャにおける各タイルにおける、CTB行の1番目のCTBを含むCTUをデコードする前に、コンテキスト変数の特定の同期プロセスが呼び出され、SPSを参照する各ピクチャにおける各タイルにおける、CTB行の1番目のCTBを含むCTUをデコードした後、コンテキスト変数の特定の記憶プロセスが呼び出されることを規定する。

10

`sps_entropy_coding_sync_enabled_flag`が0に等しいことは、SPSを参照する各ピクチャにおける各タイルにおける、CTB行の1番目のCTBを含むCTUをデコードする前に、コンテキスト変数の特定の同期プロセスが呼び出される必要がなく、SPSを参照する各ピクチャにおける各タイルにおける、CTB行の1番目のCTBを含むCTUをデコードした後、コンテキスト変数の特定の記憶プロセスが呼び出される必要がないことを規定する。`sps_persistent_Rice_adaptation_enabled_flag`が1に等しい場合、`sps_entropy_coding_sync_enabled_flag`の値は0となるべきである。

20

`sps_persistent_Rice_adaptation_enabled_flag`が1に等しいことは、`abs_remainder[ ]`及び`dec_abs_level[ ]`の2値化のためのライスパラメータ導出が、前のTUから蓄積された統計データを利用して各TUのスタートに初期化されることを規定する。

30

`sps_persistent_Rice_adaptation_enabled_flag`が0に等しいことは、ライスパラメータ導出に前のTU状態が利用されていないことを規定する。`sps_persistent_Rice_adaptation_enabled_flag`の値が存在しない場合、その値が0に等しいと推断される。`sps_entropy_coding_sync_enabled_flag`が1に等しい場合、`sps_persistent_Rice_adaptation_enabled_flag`の値は0となるべきである。

40

【0121】

上記説明では、TUは図面（例えば、図6）に説明され図示されたが、同じ技術は変換ブロック（`transform_block`、TB）に適用されることができる。言い換えれば、上記に示された実施形態（図面を含む）では、TUはTBを表すこともできる。

【0122】

ビデオコーディングの依存量子化を実現するためのコンピューティングシステムの例

【0123】

任意の適切なコンピューティングシステムは、本明細書で説明される操作を実行するために用いられることができる。例えば、図13は、図1のビデオエンコーダー100又は

50

図 2 のビデオデコーダー 200 を実現することができるコンピューティングデバイス 1300 の一例を示す。いくつかの実施形態では、コンピューティングデバイス 1300 は、メモリ 1314 に通信可能に結合され、且つ、コンピュータ実行可能なプログラムコードを実行し及び／又はメモリ 1314 に記憶された情報にアクセスするプロセッサ 1312 を含むことができる。プロセッサ 1312 は、マイクロプロセッサ、特定用途向け集積回路 (application specific integrated circuit、ASIC)、ステートマシン (state machine)、又は他の処理デバイスを含むことができる。プロセッサ 1312 は、1 つを含む任意の数の処理デバイスを含むことができる。このようなプロセッサは、命令を記憶するコンピュータ可読媒体を含み得るか、又は命令を記憶するコンピュータ可読媒体と通信し得る。当該命令は、プロセッサ 1312 によって実行されると、プロセッサに本明細書で説明される操作を実行させる。

10

【0124】

メモリ 1314 は、任意の適切な非一時的な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は、コンピュータ可読命令又は他のプログラムコードをプロセッサに提供することができる任意の電子、光学、磁気、又は他の記憶装置を含むことができる。コンピュータ可読媒体の非限定的な例としては、磁気ディスク、メモリチップ、読み取り専用メモリ (read only memory、ROM)、ランダムアクセスメモリ (random access memory、RAM)、ASIC、構成されたプロセッサ、光学記憶装置、磁気テープ又は他の磁気記憶装置、又はコンピュータプロセッサがそこから命令を読み取ることができる他の媒体が挙げられる。命令は、例えば、C、C++、C#、Visual Basic、Java、Python、Perl、JavaScript、及び ActionScript を含む任意の適切なコンピュータプログラミング言語で記述されたコードから、コンパイル及び／又はインタプリタによって生成されたプロセッサ特定の命令を含むことができる。

20

【0125】

コンピューティングデバイス 1300 は、バス 1316 を含むこともできる。バス 1316 は、コンピューティングデバイス 1300 の 1 つ又は複数のコンポーネントに通信可能に結合されることができる。コンピューティングデバイス 1300 は、入力デバイス又は出力デバイスなどの複数の外部デバイス又は内部デバイスを含むこともできる。例えば、示されるコンピューティングデバイス 1300 は、1 つ又は複数の入力デバイス 1320 から入力を受信し、又は 1 つ又は複数の出力デバイス 1322 に出力を提供することができる入力／出力 (input/output、I/O) インターフェース 1318 を有する。1 つ又は複数の入力デバイス 1320 及び 1 つ又は複数の出力デバイス 1322 は、I/O インターフェース 1318 に通信可能に結合され得る。通信結合は、任意の適切な方法 (例えば、プリント回路基板を介した接続、ケーブルを介した接続、無線伝送を介した通信など) で実現され得る。入力デバイス 1320 の非限定的な例には、タッチスクリーン (例えば、タッチ領域をイメージングするための 1 つ又は複数のカメラ、又は、タッチによって引き起こされる圧力変化を検出するための圧力センサ)、マウス、キーボード、又は、コンピューティングデバイスのユーザの物理的動作に応答して入力イベントを生成するために用いられることができる任意の他のデバイスが含まれる。出力デバイス 1322 の非限定的な例には、LCD スクリーン、外部モニタ、スピーカ、又は、コンピューティングデバイスによって生成された出力を表示するか、もしくは他の方法で提示するために用いられることができる任意の他のデバイスが含まれる。

30

40

【0126】

コンピューティングデバイス 1300 は、プログラムコードを実行することができ、当該プログラムコードは、図 1 ～図 12 を参照して叙述した上記の操作の 1 つ又は複数を実行するようにプロセッサ 1312 を構成する。当該プログラムコードは、ビデオエンコーダー 100 又はビデオデコーダー 200 を含み得る。プログラムコードは、メモリ 1314 又は任意の適切なコンピュータ可読媒体に留まることができ、プロセッサ 1312 又は

50

任意の他の適切なプロセッサによって実行されることができる。

【0127】

コンピューティングデバイス1300はまた、少なくとも1つのネットワークインターフェースデバイス1324を含み得る。ネットワークインターフェースデバイス1324は、1つ又は複数のデータネットワーク1328と有線又は無線データ接続を確立することに適した任意のデバイス又はデバイスグループを含み得る。ネットワークインターフェースデバイス1324の非限定的な例としては、イーサネットネットワークアダプタ、モデムなどが含まれる。コンピューティングデバイス1300は、ネットワークインターフェースデバイス1324を介して、電子信号又は光信号としてメッセージを送信することができる。

10

【0128】

一般的な考慮

【0129】

特許請求の主題を十分に理解するために、本明細書には多数の具体的なディテールが記載されている。しかしながら、当業者であれば、これらの具体的なディテールがなくても、特許請求の主題を実施し得ることを理解するであろう。他の例では、当業者に知られることができる方法、装置、又はシステムは、特許請求の主題を不明瞭にしないように詳細には記載されていない。

【0130】

特に説明されない限り、本明細書を通じて、「処理」、「コンピューティング」、「計算」、「確定」、及び「識別」などの用語を利用する議論は、コンピューティングデバイス（例えば、1つ又は複数のコンピュータ又は類似の1つ又は複数の電子コンピューティングデバイス）の動作又は処理を指すことが理解される。当該コンピューティングデバイスは、メモリ、レジスタ、又は他の情報記憶デバイス、伝送デバイス、又はコンピューティングプラットフォームのディスプレイデバイス内の物理的な電子又は磁気量として表されるデータを操作し又は変換する。

20

【0131】

本明細書で論じる1つ又は複数のシステムは、任意の特定のハードウェアアーキテクチャ又は構成に限定されない。コンピューティングデバイスは、1つ又は複数の入力を条件として結果を提供する任意の適切なコンポーネントの配置を含むことができる。適切なコンピューティングデバイスは、記憶されたソフトウェアにアクセスする多目的マイクロプロセッサベースのコンピュータシステムを含み、当該ソフトウェアは、コンピューティングシステムを、汎用コンピューティング装置から本主題の1つ又は複数の実施形態を実現する特定のコンピューティング装置となるようにプログラムし又は構成する。任意の適切なプログラミング、スクリプト、又は他のタイプの言語又は言語の組み合わせは、本明細書におけるコンピューティングデバイスをプログラムし又は構成するために用いられるソフトウェアに含まれる教示を実現するために用いられることができる。

30

【0132】

本明細書に開示される方法の実施形態は、そのようなコンピューティングデバイスの操作において実行され得る。以上の例におけるブロックの順序は変化することができ、例えば、ブロックは、再順序付けされ、組み合わせられ、及び／又はサブブロックに分割され得る。いくつかのブロック又はプロセスは、並行して実行されることができる。

40

【0133】

本明細書における「に適合した」又は「ように構成された」の使用は、付加的なタスク又はステップを実行するように適合又は構成されたデバイスを排除せず、開放的且つ包括的な文言として意図されている。さらに、「に基づく」の使用は、開放的且つ包括的であることを意図しており、1つ又は複数の挙げられた条件又は値「に基づく」処理、ステップ、計算、又は他の動作は実際に、挙げられた条件又は値以外の追加の条件又は値に基づくことができる。本明細書に含まれる見出し、リスト、及び番号付けは、説明を容易にするためだけのものであり、限定を意味するものではない。

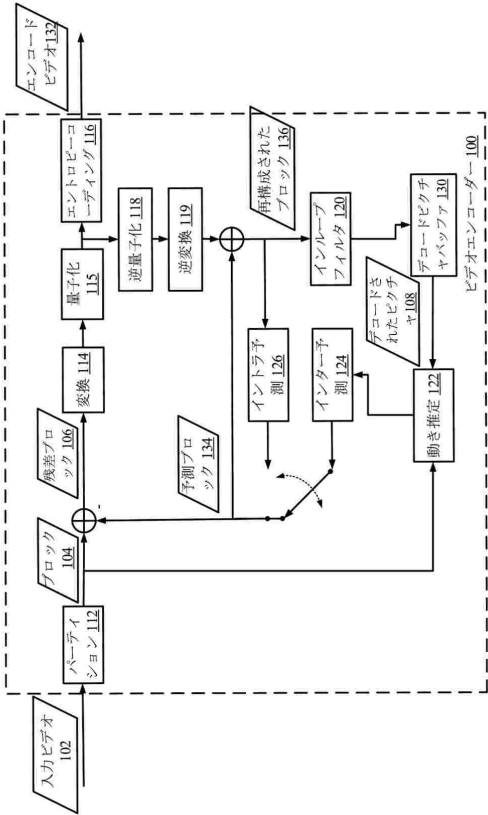
50

【 0 1 3 4 】

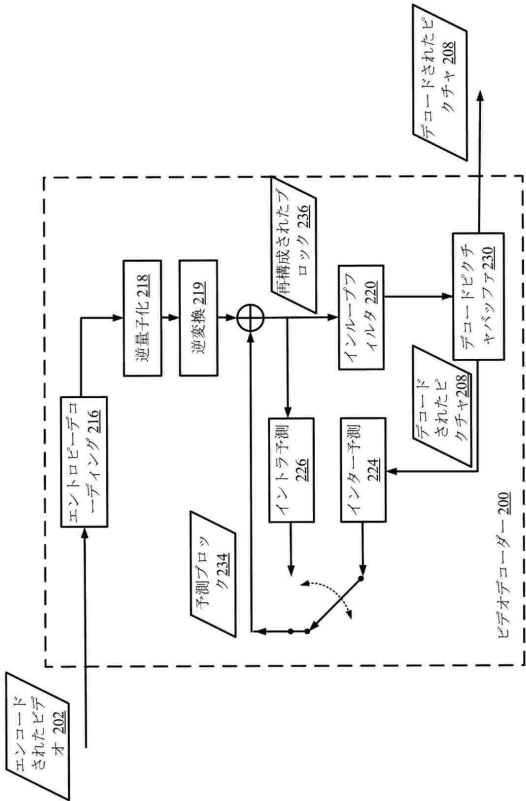
本主題は、その特定の実施形態を参照して詳細に説明されたが、当業者は、前述を理解した後、そのような実施形態に対する変更、変形、及び等価物を容易に提供することができることが理解される。従って、本開示は、限定ではなく例示の目的で提示されたものであり、当業者にとって明らかである変更、変形、及び／又は追加を含むことを排除しないことを理解されたい。

【 図 面 】

【 図 1 】



【 図 2 】



10

20

30

40

50

【 図 3 】

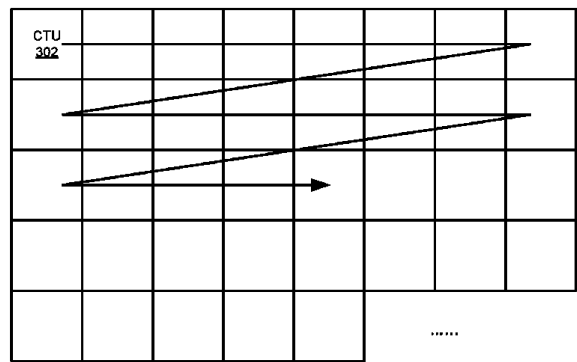


FIG. 3

【 図 4 】

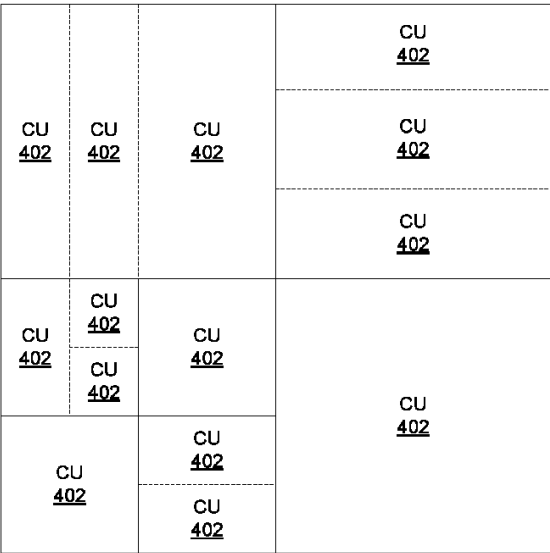


FIG. 4

【 図 5 】

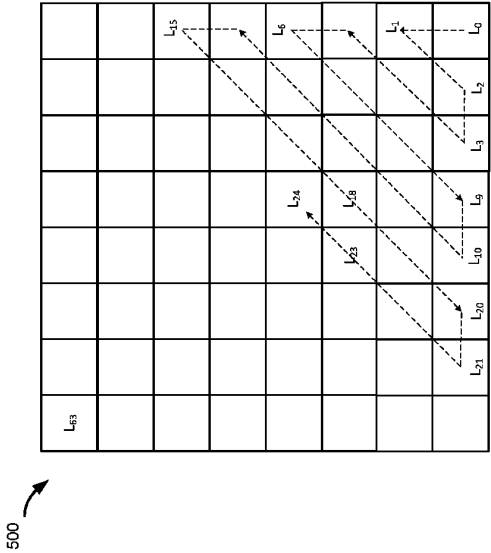
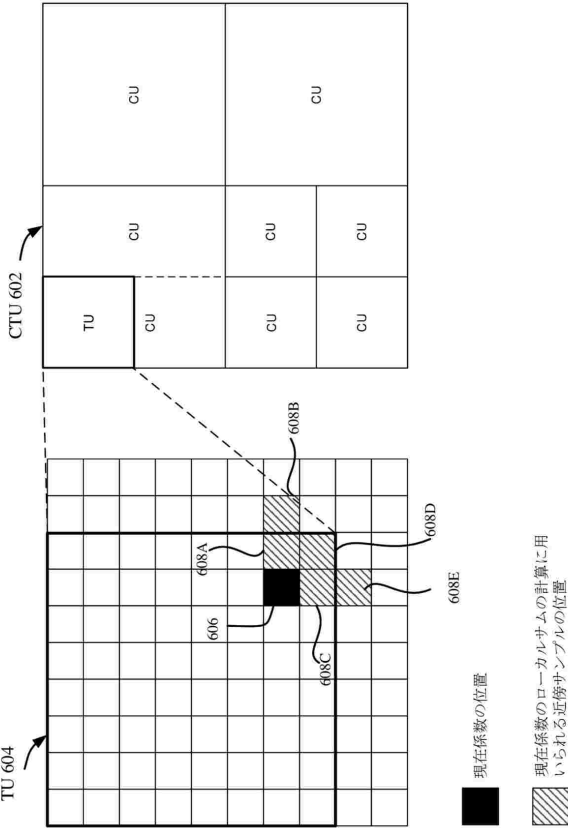


FIG. 5

【 図 6 】



現在係数の位置

現在係数のローカルサマの計算に用
いられる近傍サンプルの位置

10

20

30

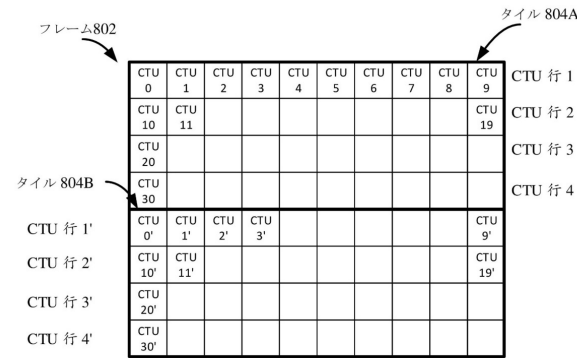
40

50

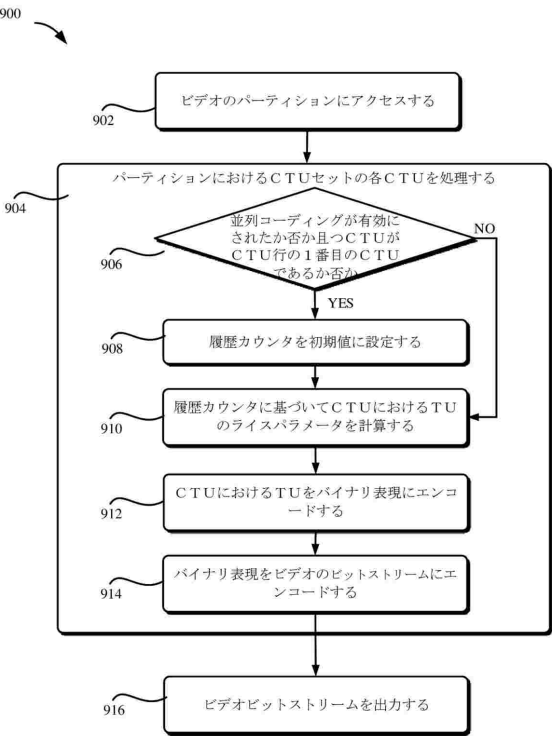
【図 7】



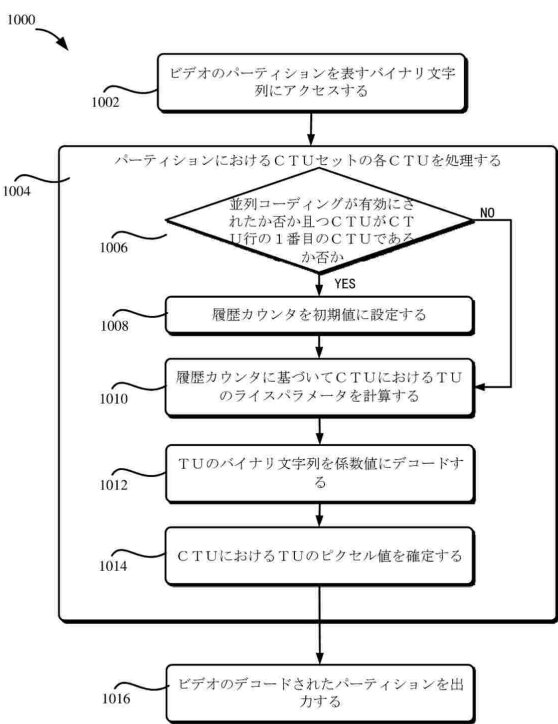
【図 8】



【図 9】



【図 10】



10

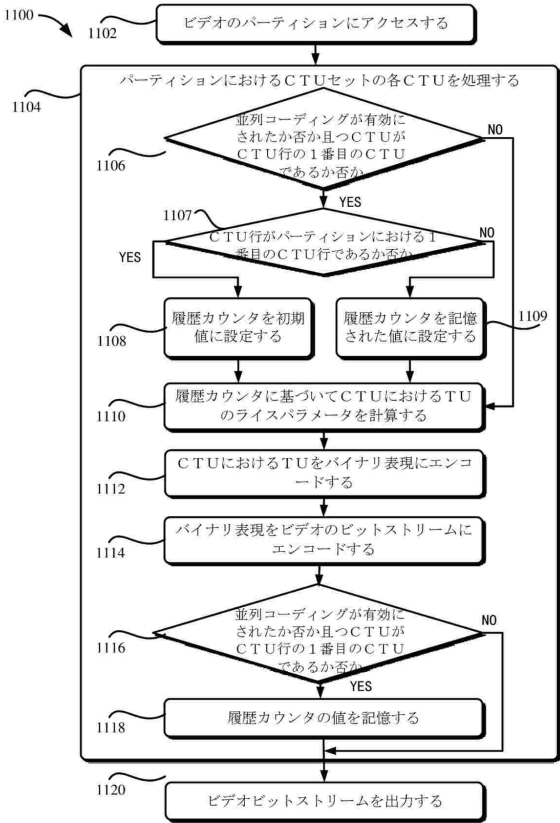
20

30

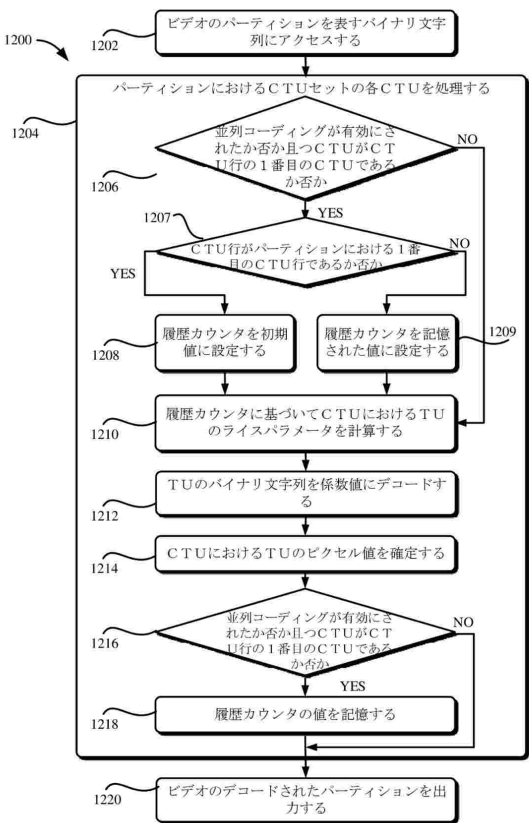
40

50

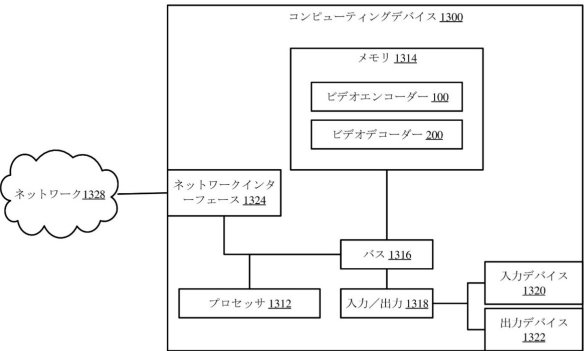
【図 1 1】



【図 1 2】



【図 1 3】



10

20

30

40

50

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 22/75507

A. CLASSIFICATION OF SUBJECT MATTER

IPC - INV. H04N 19/44, H04N 19/46 (2022.01)
 ADD. H04N 19/172, H04N 19/184 (2022.01)
 CPC - INV. H04N 19/44, H04N 19/46
 ADD. H04N 19/172, H04N 19/184

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

See Search History document

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

See Search History document

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

See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	WO 2021/136533 A1 (Huawei Technologies Co., Ltd.) 08 July 2021 (08.07.2021) entire document (especially pg. 21, ln 4-18; pg. 26, ln 34 to pg. 27, ln 5; pg. 58, ln 1-15; pg. 78, ln 27 to pg. 79, ln 11)	1-20
Y	WO 2021/061489 A1 (Futurewei Technologies, Inc.) 01 April 2021 (01.04.2021) entire document (especially para [0088], [0095], [00173]-[00178], [00188])	1-20

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

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"D" document cited by the applicant in the international application

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

08 November 2022

Date of mailing of the international search report

DEC 06 2022

Name and mailing address of the ISA/US

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents
 P.O. Box 1450, Alexandria, Virginia 22313-1450
 Facsimile No. 571-273-8300

Authorized officer

Karl Rodriguez

Telephone No. PCT Helpdesk: 571-272-4300

Form PCT/ISA/210 (second sheet) (July 2022)

フロントページの続き

(32)優先日 令和3年10月4日(2021.10.4)

(33)優先権主張国・地域又は機関
米国(US)

(81)指定国・地域 AP(BW,GH,GM,KE,LR,LS,MW,MZ,NA,RW,SD,SL,ST,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,RU,TJ,TM),EP(AL,AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HR,HU,IE,IS,IT,LT,LU,LV,MC,MK,MT,NL,NO,PL,PT,RO,RS,SE,SI,SK,SM,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,KM,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AO,AT,AU,AZ,BA,BB,BG,BH,BN,BR,BW,BY,BZ,CA,CH,CL,CN,CO,CR,CU,CV,CZ,DE,DJ,DK,DM,DO,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,GT,HN,HR,HU,ID,IL,IN,IQ,IR,IS,IT,JM,JO,JP,KE,KG,KH,KN,KP,KR,KW,KZ,LA,LC,LK,LR,LS,LU,LY,MA,MD,ME,MG,MK,MN,MW,MX,MY,MZ,NA,NG,NI,NO,NZ,OM,PA,PE,PG,PH,PL,PT,QA,RO,RS,RU,RW,SA,SC,SD,SE,SG,SK,SL,ST,SV,SY,TH,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN,WS,ZA,ZM,ZW

(特許庁注：以下のものは登録商標)

1. VISUAL BASIC

2. JAVA

3. PYTHON

4. JAVASCRIPT

(74)代理人 100152205

弁理士 吉田 昌司

(74)代理人 100137523

弁理士 出口 智也

(72)発明者 ユイ、ユエ

アメリカ合衆国カリフォルニア州、パロアルト、イースト、ベイショア、ロード、2479、スイート110、ケアオブ、イノピーク、テクノロジー、インコーポレイテッド

(72)発明者 ユイ、ハオピン

アメリカ合衆国カリフォルニア州、パロアルト、イースト、ベイショア、ロード、2479、スイート110、ケアオブ、イノピーク、テクノロジー、インコーポレイテッド

Fターム(参考) 5C159 KK14 LC09 MA04 MA05 MA21 MC11 ME01 UA02 UA05 UA16